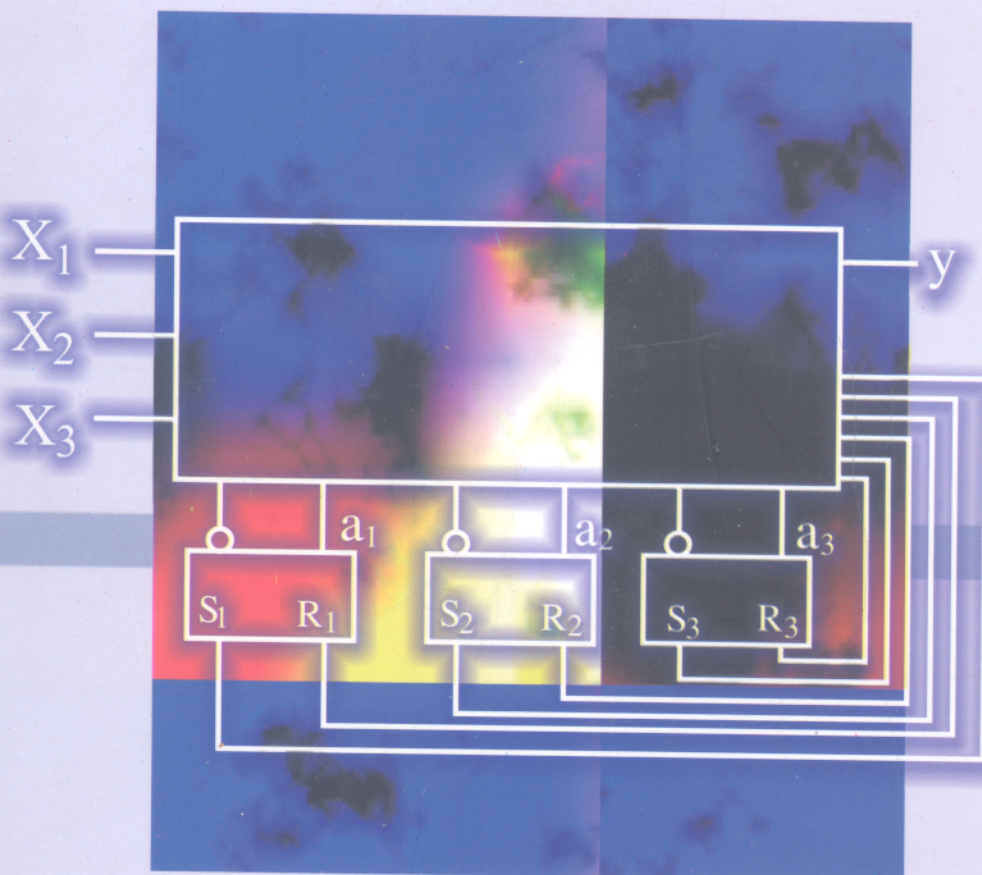


PGS. TS. NGUYỄN DUY BẢO



PHƯƠNG PHÁP PHÂN TÍCH & TỔNG HỢP THIẾT BỊ SỐ



NHÀ XUẤT BẢN KHOA HỌC VÀ KỸ THUẬT

PGS. TS. NGUYỄN DUY BẢO

PHƯƠNG PHÁP PHÂN TÍCH VÀ TỔNG HỢP THIẾT BỊ SỐ



NHÀ XUẤT BẢN KHOA HỌC VÀ KỸ THUẬT
HÀ NỘI - 2005

Chịu trách nhiệm xuất bản : PGS, TS. TÔ ĐĂNG HẢI
Biên tập : VŨ THỊ MINH LUẬN
Vẽ bìa : HƯƠNG LAN

NHÀ XUẤT BẢN KHOA HỌC VÀ KỸ THUẬT
70 TRẦN HUNG ĐẠO – HÀ NỘI

In 700 cuốn, khổ 19 x 27cm tại Xí nghiệp in NXB Lý luận chính trị
Giấy phép xuất bản số: 150 - 269 do Cục xuất bản cấp ngày 4/2/2005
In xong và nộp lưu chiểu tháng 7/2005.

MỤC LỤC

	Trang
MỞ ĐẦU	9
Phần I	
CƠ SỞ TOÁN CỦA THIẾT BỊ SỐ	
Chương 1. HỆ ĐẾM VÀ MỘT SỐ PHƯƠNG PHÁP BIỂU DIỄN THÔNG TIN	15
1.1. Các hệ đếm	15
1.2. Hệ đếm vị trí	15
1.2.1. Khái niệm	15
1.2.2. Đặc trưng của các hệ đếm vị trí	16
1.2.3. Hệ đếm cơ số 10	16
1.2.4. Hệ đếm cơ số 2	16
1.2.5. Hệ cơ số 8 và cơ số 16	17
1.3. Chuyển đổi giữa các hệ đếm	17
1.3.1. Quy tắc chuyển đổi chung	18
1.3.2. Trường hợp đặc biệt	20
1.4. Một số cách biểu diễn thông tin thường dùng trong các hệ xử lý tin	21
1.4.1. Các đơn vị thông tin	21
1.4.2. Biểu diễn số nguyên không dấu	21
1.4.3. Biểu diễn số nguyên có dấu	21
1.4.4. Cốt ngược và cốt bù của một số nhị phân	21
1.4.5. Cốt 2-10 hay còn gọi là mã BCD (Binary Coded Decimals)	23
1.4.6. Mã ASCII	25
1.4.7. Quan hệ giữa số BCD và số ASCII của một số	25
Bài tập chương 1	26
Chương 2. CÁC HÀM LÔGIC CƠ BẢN VÀ TÍNH CHẤT	28
2.1. Khái niệm về đại số logic	28
2.1.1. Các thuật toán logic cơ bản	28
2.1.2. Đại số Boole	30
2.1.3. Biểu diễn các phép toán logic cơ bản dùng giản đồ Venne	30
2.1.4. Biến và hàm logic	31
2.2. Các hàm logic cơ bản	32
2.2.1. Hàm logic không biến ($n = 0$)	32
2.2.2. Hàm logic một biến ($n = 1$)	32
2.2.3. Hàm logic hai biến ($n = 2$)	33
2.3. Các hàm logic quan trọng	35
2.3.1. Hàm y_7	35

2.3.2.	Hàm y_1	36
2.3.3.	Hàm y_8	36
2.3.4.	Hàm y_{14}	37
2.3.5.	Hàm y_6	37
2.4.	Các tính chất và định luật cơ bản.....	38
2.4.1.	Tính chất và định luật của các hàm HOAC, VA, KH.....	38
2.4.2.	Các tính chất và định luật quan trọng của các hàm VA-KH và HOAC-KH.....	39
2.4.3.	Các quan hệ và các định luật của hàm cộng môđun 2.....	40
2.5.	Phương pháp mô tả hàm logic.....	40
2.5.1.	Phương pháp bảng giá trị thực.....	41
2.5.2.	Phương pháp hình học.....	42
2.5.3.	Mô tả hàm logic dùng biểu thức toán học.....	44
2.6.	Hệ hàm đủ.....	50
2.7.	Phân tích và tổng hợp hàm logic.....	51
2.7.1.	Biểu diễn các hàm logic qua hệ hàm đủ.....	51
2.7.2.	Tổng hợp hàm logic.....	53
2.8.	Một số lớp hàm logic thường gặp.....	53
2.8.1.	Hàm logic tuyến tính.....	54
2.8.2.	Hàm logic bảo tồn giá trị 0.....	54
2.8.3.	Hàm logic bảo tồn giá trị 1.....	55
2.8.4.	Hàm logic đơn điệu.....	55
2.8.5.	Hàm logic đối ngẫu.....	56
2.9.	Mạch thực hiện các hàm logic cơ bản.....	58
2.9.1.	Mạch VA, HOAC dùng diốt - điện trở.....	59
2.9.2.	Mạch VA-KH và mạch HOAC-KH.....	60
	Bài tập chương 2.....	65
Chương 3.	PHƯƠNG PHÁP ĐƠN GIẢN CÁC HÀM LÓGIC.....	69
3.1.	Những khái niệm cơ bản.....	69
3.1.1.	Hàm đánh giá độ phức tạp.....	69
3.1.2.	Các dạng tuyến chuẩn khác nhau.....	70
3.2.	Phương pháp biến đổi toán học.....	71
3.3.	Đơn giản hàm bằng phương pháp hình học.....	74
3.3.1.	Phương pháp dùng bảng kế cận.....	74
3.3.2.	Phương pháp dùng bảng đối xứng.....	76
3.4.	Phương pháp McClusky.....	79
3.4.1.	Nội dung phương pháp McClusky.....	79
3.4.2.	Thí dụ.....	79
3.5.	Phương pháp ROT.....	81
3.5.1.	Khái niệm và cơ sở toán học của phương pháp.....	81
3.5.2.	Algôrit đơn giản hàm.....	88
3.6.	Đơn giản hệ hàm.....	89
3.6.1.	Phương pháp đơn giản riêng rẽ từng hàm.....	90
3.6.2.	Đơn giản kết hợp hệ hàm.....	91
3.6.3.	Phương pháp "Dấu" đơn giản hệ hàm.....	95
	Bài tập chương 3.....	102

Phần II

PHƯƠNG PHÁP TỔNG HỢP VÀ PHÂN TÍCH THIẾT BỊ SỐ TỔ HỢP

Chương 4. TỔNG HỢP CẤU TRÚC THIẾT BỊ SỐ TỔ HỢP	107
4.1. Tín hiệu và các mạch logic	107
4.1.1. Tín hiệu nhị phân	107
4.1.2. Khái niệm về mạch logic	108
4.1.3. Giá của mạch logic	111
4.2. Khái niệm về thiết bị số tổ hợp	111
4.2.1. Đặc điểm mạch logic tổ hợp	111
4.2.2. Các bước cơ bản tổng hợp thiết bị số tổ hợp	112
4.2.3. Vi mạch số	113
4.3. Tổng hợp cấu trúc mạch hai cấp	118
4.3.1. Trường hợp một hàm	119
4.3.2. Trường hợp một hệ hàm	126
4.4. Tổng hợp cấu trúc mạch nhiều cấp	129
4.4.1. Tổng hợp khi số đầu vào của các phần tử logic bị hạn chế	130
4.4.2. Tổng hợp khi khả năng tải của các phần tử logic bị hạn chế	132
4.4.3. Mạch VA-KH, HOAC-KH nhiều cấp	134
4.5. Tổng hợp theo cách phân giải hàm	138
4.5.1. Phân giải tách rời đơn giản (PTD)	138
4.5.2. Phân giải không tách rời đơn giản (PKTD)	143
4.5.3. Algorit thực hiện phân giải hàm	145
4.6. Thực hiện hàm logic dùng các mảng logic lập trình được và bằng các bộ nhớ	148
4.6.1. Thực hiện bằng PAL, PLA hay FPLA	149
4.6.2. Thực hiện trên RAM	153
4.6.3. Thực hiện hàm logic dùng chương trình	154
Bài tập chương 4	158
Chương 5. PHÂN TÍCH THIẾT BỊ SỐ TỔ HỢP	163
5.1. Phân tích tĩnh thiết bị số tổ hợp	163
5.1.1. Nhiệm vụ bài toán	163
5.1.2. Thủ tục chung phân tích tĩnh mạch tổ hợp	165
5.2. Phân tích động thiết bị số tổ hợp	166
5.2.1. Trường hợp khi thay đổi từ một tập giá trị biến này sang một tập giá trị biến khác	167
5.2.2. Khi tác động liên tiếp một chuỗi các tập giá trị biến vào	168
5.2.3. Các trường hợp khác	171
5.3. Bản chất của tranh chấp trạng thái trong Mạch tổ hợp, phương pháp phát hiện và khử bỏ	171
5.3.1. Bản chất của tranh chấp trạng thái trong mạch tổ hợp	171
5.3.2. Phân loại tranh chấp trạng thái	173
5.3.3. Tranh chấp trạng thái ở các mạch số	173
5.4. Kiểm tra và chẩn đoán hư hỏng trong thiết bị số tổ hợp	177
5.4.1. Các loại hư hỏng	177
5.4.2. Phương pháp phát hiện các hư hỏng logic đơn	179
5.4.3. Bài toán chẩn đoán kỹ thuật	181
Bài tập chương 5	186

Chương 6. TỔNG HỢP THIẾT BỊ SỐ TỔ HỢP THƯỜNG DÙNG	190
6.1. Khái quát chung	190
6.2. Tổng hợp các bộ biến đổi mã	191
6.2.1. Bộ biến đổi mã nhị phân sang mã thập phân (1 trong N) và ngược lại	191
6.2.2. Bộ trộn và bộ phân đường (Multiplexer và Demultiplexer)	196
6.2.3. Bộ so sánh	198
6.2.4. Các bộ cộng	200
Bài tập chương 6	202

Phần III

TỔNG HỢP VÀ PHÂN TÍCH MẠCH SỐ CÓ NHỚ

Chương 7. CÁC HÀM LÔGIC PHỤ THUỘC THỜI GIAN VÀ HÀM LÔGIC TRUY TOÁN	205
7.1. Hàm logic phụ thuộc thời gian	205
7.1.1. Những khái niệm cơ bản	205
7.1.2. Tính chất của hàm logic phụ thuộc thời gian	206
7.2. Các hàm logic truy toán	208
7.2.1. Hàm logic truy toán loại 1 (HLTR-1)	208
7.2.2. Hàm logic truy toán suy biến loại 1 (HLTRS-1)	209
7.2.3. Hàm logic truy toán loại 2 (HLTR-2)	210
Bài tập chương 7	211
Chương 8. KHÁI NIỆM VỀ THIẾT BỊ SỐ CÓ NHỚ VÀ ÔTÔMÁT HỮU HẠN	213
8.1. Khái niệm chung về thiết bị số có nhớ	213
8.2. Mô hình của ôtomát hữu hạn	215
8.2.1. Mô hình cấu trúc	215
8.2.2. Mô hình toán của ôtomát hữu hạn	215
8.3. Các phương pháp mô tả ôtomát	216
8.3.1. Khái quát chung	216
8.3.2. Phương pháp bảng chuyển đổi trạng thái và bảng ra	217
8.3.3. Phương pháp ma trận chuyển đổi trạng thái của ôtomát	222
8.3.4. Đồ hình của ôtomát	224
8.4. Quan hệ giữa ôtomát Milly và ôtomát Moore	225
8.4.1. Vấn đề tương đương của hai loại ôtomát	225
8.4.2. Phương pháp chuyển đổi từ ôtomát Moore sang ôtomát Milly	227
8.4.3. Chuyển từ dạng ôtomát Milly sang ôtomát Moore	229
8.5. Ôtomát đồng bộ và không đồng bộ	233
8.5.1. Trường hợp trạng thái vào có dạng thế	234
8.5.2. Trường hợp trạng thái vào có dạng xung	236
8.6. Mô hình kỹ thuật của ôtomát hữu hạn	240
Bài tập chương 8	243
Chương 9. TỔNG HỢP THIẾT BỊ SỐ CÓ NHỚ	246
9.1. Thủ tục tổng hợp cấu trúc thiết bị số có nhớ	246
9.2. Đơn giản trạng thái trong của ôtomát	248
9.2.1. Những khái niệm cơ bản	248

9.2.2.	Phương pháp nhận biết hai trạng thái tương đương hay thực hiện nhau trên bảng chuyển đổi trạng thái và bảng ra.....	250
9.2.3.	Phương pháp xác định các cặp trạng thái tương đương và nhóm các trạng thái tương đương tối đa	255
9.2.4.	Phương pháp đơn giản trạng thái trong của ô tô máy không đồng bộ xác định không đầy đủ	262
9.3.	Mã hoá trạng thái của ô tô máy	268
9.3.1.	Hiện tượng tranh chấp trạng thái trong của ô tô máy	270
9.3.2.	Các mã chống tranh chấp trạng thái nguy hiểm của ô tô máy	271
9.3.3.	Phương pháp mã hoá chống tranh chấp trạng thái	274
9.4.	Xác định phần tử nhớ	278
9.5.	Nhận các hàm kích thích và các hàm ra	281
9.5.1.	Xây dựng cấu trúc mạch sau khi đã mã hoá các trạng thái	281
9.5.2.	Phương pháp rút ra các hàm ra	282
	Bài tập chương 9	293
Chương 10.	CÁC PHẦN TỬ NHỚ CƠ BẢN	296
10.1.	Khái niệm về phần tử nhớ cơ bản	296
10.2.	Phần tử nhớ một đầu vào	297
10.2.1.	Mô hình hàm chức năng của phần tử nhớ một đầu vào	297
10.2.2.	Xét hàm a_1	299
10.2.3.	Xét hàm a_7	300
10.2.4.	Xét hàm a_3	300
10.2.5.	Xét hàm a_6	300
10.3.	Các phần tử nhớ hai đầu vào tín hiệu	301
10.3.1.	Phần tử nhớ cơ bản xây dựng trên cơ sở hàm RS	301
10.3.2.	Các phần tử nhớ loại CT và CD	303
10.3.3.	Phần tử nhớ xây dựng trên cơ sở hàm JK	304
10.4.	Phần tử nhớ có từ ba đầu vào trở lên	305
10.4.1.	Phần tử nhớ RST	305
10.4.2.	Phần tử nhớ SR - CJK	305
10.5.	Tổng hợp các phần tử nhớ cơ bản	306
10.5.1.	Tổng hợp trigơ RS	306
10.5.2.	Chuyển đổi từ loại trigơ này sang loại trigơ khác	307
	Bài tập chương 10	310
Chương 11.	TỔNG HỢP HÀM LÔGIC PHỤ THUỘC THỜI GIAN VÀ HÀM LÔGIC TRUY TOÁN	311
11.1.	Tổng hợp thiết bị số cho bởi hàm logic phụ thuộc thời gian	311
11.2.	Tổng hợp thiết bị số có nhớ cho bởi hàm logic truy toán	315
11.2.1.	Tổng hợp thiết bị số cho bởi hàm logic truy toán suy biến loại I	316
11.2.2.	Tổng hợp thiết bị số cho bởi hàm logic truy toán loại I	319
11.3.	Tổng hợp thiết bị số cho bởi hàm logic truy toán loại II	323
11.4.	Phân tích thiết bị số xây dựng trên cơ sở Hàm logic phụ thuộc thời gian và hàm logic truy toán	325
	Bài tập chương 11	330

Chương 12. PHÂN TÍCH THIẾT BỊ SỐ CÓ NHỚ	332
12.1. Phân tích thiết bị số có nhớ dùng Phần tử nhớ được nhịp	332
12.1.1. Đặc điểm	332
12.1.2. Các bước phân tích	333
12.2. Phân tích thiết bị số dùng phần tử nhớ làm việc với tín hiệu thế không được nhịp	340
12.2.1. Phân tích tĩnh.....	340
12.2.2. Phân tích động.....	342
12.3. Phân tích thiết bị số dùng phần tử nhớ làm việc với tín hiệu xung	345
12.3.1. Phân tích tĩnh.....	346
12.3.2. Phân tích động.....	348
12.4. Những hư hỏng trong thiết bị số	352
12.4.1. Chuỗi dẫn và chuỗi vào thiết lập.....	353
12.4.2. Vấn đề phát hiện các hư hỏng trong ô tô mát.....	355
12.4.3. Phương hướng tìm chuỗi kiểm tra hư hỏng	358
Bài tập chương 12	364
Chương 13. TỔNG HỢP THIẾT BỊ SỐ CÓ TÍNH TỐI CÁC ĐẶC ĐIỂM CỦA CÁC PHẦN TỬ NHỚ	368
13.1. Tổng hợp thiết bị số dùng Phần tử nhớ được nhịp.....	368
13.1.1. Thủ tục chung.....	368
13.1.2. Một số thí dụ	369
13.2. Tổng hợp thiết bị số dùng Phần tử nhớ không nhịp, đầu vào xung	374
13.2.1. Thủ tục chung.....	374
13.2.2. Một số thí dụ	375
13.3. Tổng hợp thiết bị số có nhớ khi Phần tử nhớ không được nhịp, đầu vào thế.....	381
13.3.1. Thủ tục chung.....	381
13.3.2. Một số thí dụ	382
Bài tập chương 13	385
Chương 14. TỔNG HỢP CÁC THIẾT BỊ SỐ THÔNG DỤNG	389
14.1. Tổng hợp mạch cộng	389
14.2. Tổng hợp mạch so sánh có nhớ	391
14.2.1. Mạch so sánh hai số nhị phân n bit, xác định $A = B$, $A < B$ hay $A > B$	391
14.2.2. Mạch so sánh hai số nhị phân, phân biệt $A = B$ hay $A \neq B$	391
14.3. Tổng hợp các mạch đếm	393
14.3.1. Phân loại mạch đếm.....	394
14.3.2. Thiết kế mạch đếm.....	395
14.4. Tổng hợp các mạch ghi	403
14.4.1. Khái niệm chung	403
14.4.2. Phân loại các mạch ghi	403
14.4.3. Tổng hợp các mạch ghi	404
Bài tập chương 14	407
TÀI LIỆU THAM KHẢO	408

Cùng với sự phát triển mạnh mẽ của lý thuyết ô tô măt hữu hạn và những thành tựu rực rỡ của công nghệ chế tạo các vi mạch, kỹ thuật số trong những năm gần đây nổi lên như một mũi nhọn trong lĩnh vực kỹ thuật điện tử, đặc biệt từ giữa những năm 1970, khi xuất hiện hàng loạt vi mạch cỡ lớn như các bộ nhớ tĩnh (SRAM), các bộ nhớ động (DRAM) có dung lượng hàng chục ngàn kilôbit, các bộ nối ghép vào - ra khả lập trình và các bộ vi xử lý làm việc theo chương trình đã thâm nhập vào hầu hết các lĩnh vực khoa học và kỹ thuật. Lĩnh vực mà kỹ thuật số có điều kiện thâm nhập sớm nhất và có hiệu quả cao nhất, đó là lĩnh vực máy tính số, tiếp đến là các lĩnh vực thông tin số, điều khiển số, đo lường và các thiết bị dân dụng khác. Ngày nay không một lĩnh vực khoa học, kỹ thuật nào mà không có sự tham gia của các thiết bị làm việc với tín hiệu số. Kỹ thuật số đã trở thành môn kỹ thuật cơ sở không thể thiếu của nhiều ngành khoa học, kỹ thuật.

I. LỊCH SỬ PHÁT TRIỂN CỦA KỸ THUẬT SỐ

Kỹ thuật số dựa trên cơ sở đại số lôgic là đại số Boole, có một lịch sử phát triển không lâu. Vào những năm 1950, các mạch số cơ bản được xây dựng chủ yếu trên các đèn điện tử, chưa được phát triển cả về chủng loại lẫn chức năng và rất hạn chế trong ứng dụng vì kích thước lớn, tiêu hao năng lượng nhiều và các mạch có chức năng rất hạn chế.

Vào những năm 1960, kỹ thuật số đã được phát triển thêm một bước quan trọng vì được dựa trên nền công nghệ bán dẫn. Các vi mạch bán dẫn số bắt đầu phát triển mạnh về chức năng và chủng loại ngày càng đa dạng. Song thời gian này chủ yếu vẫn chỉ có các vi mạch có độ tích hợp cỡ nhỏ (Small Scale Intergration - SSI) và cỡ trung bình (Middle Scale Intergration - MSI). Đặc biệt từ những năm 1970, nhờ tiến bộ vượt bậc trong công nghệ chế tạo các vi mạch đã cho ra đời những vi mạch có độ tích hợp cỡ lớn (Large Scale Intergration - LSI) và sau đó là các vi mạch có độ tích hợp cỡ cực lớn (Very LSI - VLSI) với hàng vạn linh kiện và transistor trong một vi mạch. Với độ tích hợp lớn như vậy, mỗi vi mạch, mặc dù có kích thước rất nhỏ, bây giờ không còn chỉ thực hiện một chức năng đơn giản mà đã có thể thực hiện một chức năng phức tạp và rất phức tạp, thậm chí chức năng của chúng có thể thay đổi theo từng nhiệm vụ - chúng ta có thể lập trình cho hoạt động chức năng của chúng. Ngày nay, người ta đã biết tới các vi mạch như các bộ nhớ động có dung lượng nhớ tới hàng chục ngàn bit; các vi mạch nối ghép có khả năng làm việc theo nhiều chương trình, khác nhau về tổ chức kết cấu và chức năng. Chúng là các thiết bị phức tạp và mềm dẻo, có thể đáp ứng được nhiều

nhiệm vụ khác nhau. Điển hình nhất cho các loại vi mạch cỡ cực lớn này là các bộ vi xử lý của các hãng INTEL, MOTOROLA, ZILOG, HITACHI... từ 8 bit tới 16, 32 và 64 bit dữ liệu và với số byte bộ nhớ có thể đánh địa chỉ trực tiếp lên tới 2^{21} byte.

II. ẢNH HƯỞNG CỦA TIẾN BỘ CÔNG NGHỆ CHẾ TẠO VI MẠCH TỚI LĨNH VỰC KỸ THUẬT SỐ

Nhờ sự tiến bộ trong công nghệ chế tạo vi mạch mà một vi mạch có kích thước rất nhỏ bé vẫn có thể thực hiện những chức năng phức tạp. Đó là cơ sở để có thể làm cho các thiết bị điện tử trở nên thông minh, có sức mạnh to lớn và cho phép ứng dụng vào hầu hết các trang thiết bị điện tử trong mọi lĩnh vực khoa học, kỹ thuật và đời sống. Vai trò và ảnh hưởng của sự tiến bộ trong công nghệ có thể là:

+ Làm cho khả năng ứng dụng kỹ thuật số vào hầu hết các lĩnh vực khoa học và kỹ thuật.

+ Làm tăng khả năng và hiệu quả của các trang thiết bị, trong nhiều trường hợp, có thể biến các trang thiết bị này thành các trang thiết bị thông minh.

+ Sự ra đời của các vi mạch có tính năng cao, thể tích nhỏ như một linh kiện thông thường, được sử dụng vào hầu hết các trang thiết bị đã đặt ra cho hoạt động nghiên cứu khoa học những nhiệm vụ mới, làm thay đổi cả những nguyên lý, nguyên tắc, các phương pháp phân tích và tổng hợp mạch vốn vẫn tồn tại từ trước và chỉ thích hợp với các mạch được xây dựng trên các phần tử rời rạc hay các vi mạch cỡ nhỏ. Giờ đây, với các vi mạch cỡ lớn và trung bình, bài toán phân tích và tổng hợp đã đổi khác: Sự phân tích mạch hầu như không còn phải tiến hành tới từng linh kiện điện trở, tụ điện, từng transistor nữa, mà chủ yếu chỉ cần dừng lại tới từng vi mạch. Đối với bài toán tổng hợp cũng vậy, nhiệm vụ tổng hợp bây giờ là tổng hợp mạch trên cơ sở các mạch cỡ lớn, đã có những chức năng định sẵn; thậm chí là tổng hợp mạch có khả năng làm việc theo chương trình trên cơ sở các vi mạch khả lập trình. Tất cả những điều này đã trực tiếp đặt ra cho các nhà nghiên cứu phải sáng tạo ra các phương pháp phân tích và tổng hợp mới; phải sáng tạo ra các ngôn ngữ lập trình mới cho phù hợp với yêu cầu của thực tiễn và thuận lợi trong thiết kế, khai thác và sử dụng.

Cũng chính vì đòi hỏi đó, trên cơ sở các phương pháp phân tích và tổng hợp dựa trên cơ sở toán logic và các vi mạch cỡ nhỏ và trung bình, vào cuối những năm 70 và những năm 80 của thế kỷ 20, người ta đã nghiên cứu và đề xuất ra hai phương pháp dùng trong thiết kế và phân tích các hệ thống lớn, làm việc theo trạng thái rời rạc. Đó là phương pháp đồ hình dùng mô tả hoạt động và thiết kế các hệ thống điều khiển phức tạp, gọi tắt là GRAFCET (Graphe fonctionnel de Commande d' etape - Transition - Đồ hình chức năng điều khiển sự chuyển đổi trạng thái) và phương pháp mạng Petri (Petri Nets). Đây là hai phương pháp hiện đại, có khả năng ứng dụng để mô tả, mô hình hoá và thiết kế các hệ thống lớn. Trong đó GRAFCET định hướng ứng dụng cho lĩnh vực thiết kế các hệ thống điều khiển tự động; mạng Petri là công cụ rất mạnh, tuy khá phức tạp, thường dùng cho việc mô hình hoá, đánh giá và thiết kế các hệ thống lớn, cả trong kỹ thuật, xã hội, kinh tế, quân sự.

III. NỘI DUNG CỦA TÀI LIỆU

Ở nước ta, các cơ quan, xí nghiệp, các trường học và viện nghiên cứu đã và đang được trang bị ngày càng nhiều máy móc, thiết bị hiện đại mà phần lớn hoạt động theo nguyên tắc xử lý số và xử lý số tín hiệu. Nhu cầu về khai thác, sử dụng, bảo quản, sửa chữa có hiệu quả các trang thiết bị hiện đại này ngày càng tỏ ra bức bách. Ngoài ra, sự phát triển tất yếu của nền khoa học và công nghệ của nước ta là từng bước làm chủ trang thiết bị hiện đại, tiến tới có thể thực hiện cải tiến và thiết kế mới các trang thiết bị phù hợp với điều kiện, nhu cầu và hoàn cảnh của nước ta. Điều này cũng đòi hỏi có sự chuẩn bị nhiều mặt khoa học và kỹ thuật, trong đó có kỹ thuật số.

Tài liệu "Phân tích và thiết kế thiết bị số" đã được xuất bản lần đầu vào năm 1987 tại Học viện Kỹ thuật quân sự. Căn cứ vào nhu cầu hiện tại đối với tài liệu và tính tới những thay đổi cần thiết sau hơn mười năm giảng dạy cũng như những tiến bộ mới trong các lĩnh vực kỹ thuật có liên quan, trong lần xuất bản này tài liệu sẽ được chỉnh lý và bổ sung những nội dung cần thiết.

Tuy vậy, trong phạm vi hạn chế của một tài liệu có tính vừa là giáo trình vừa là tài liệu tham khảo, nội dung của tài liệu về cơ bản vẫn xuất phát từ những yêu cầu cung cấp những phương pháp phân tích và tổng hợp các thiết bị số có tính thực tiễn và được lựa chọn những kiến thức cơ bản nhất, thiết thực trong thực tế nhưng mang tính phương pháp chung, có thể vận dụng trong mọi trường hợp của các bài toán phân tích và tổng hợp mạch. Đồng thời tài liệu cũng sẽ cố gắng không để mất đi tính khoa học cơ bản và cơ sở giúp cho việc mở rộng kiến thức. Mặt khác, để tạo điều kiện cho bạn đọc dễ dàng sử dụng các công cụ hiện đại như máy tính trong việc thực hiện tự động hoá các khâu phân tích và thiết kế các thiết bị số, nhiều nội dung trong tài liệu sẽ được trình bày dưới dạng algôrit hoặc dễ dàng đưa về dạng algôrit.

Phương pháp trình bày thiết bị số trong tài liệu này được nhất quán dưới cách nhìn thống nhất: Thiết bị số là một ô tô-mát hữu hạn. Trên cơ sở đó, toàn bộ chương trình phân tích và tổng hợp thiết bị số được trình bày theo lý thuyết ô tô-mát hữu hạn - một lý thuyết có độ chính xác cao và hết sức chặt chẽ. Cũng chính vì vậy, các phương pháp phân tích và tổng hợp thiết bị số trình bày trong tài liệu này mang tính khái quát cao, giúp cho bạn đọc có điều kiện vận dụng vào thực tiễn tương đối dễ dàng và trong phạm vi rộng.

Tài liệu chia thành 2 tập, 3 phần, 14 chương.

Phần I. Cơ sở toán của thiết bị số

Phần này cung cấp những kiến thức cơ bản về đại số logic, các biến và hàm logic, các hàm logic cơ bản và tính chất của chúng cũng như các phương pháp đơn giản về hàm và hệ hàm. Hiểu kỹ các hàm logic cơ bản là cơ sở quan trọng để tiếp thu các phương pháp phân tích và tổng hợp các thiết bị số.

Để giúp cho bạn đọc có điều kiện mở rộng thêm kiến thức trong lĩnh vực mạch số có nhớ, trong phần này còn đề cập tới khái niệm và tính chất của hàm logic phụ thuộc thời gian và hàm logic truy toán.

Trong các chương của phần I đều đưa ra nhiều thí dụ giúp người đọc dễ dàng nắm được những nội dung cơ bản của từng phần.

Phần II. Phương pháp tổng hợp và phân tích các thiết bị số tổ hợp

Phần này cung cấp những kiến thức cần thiết để tổng hợp và phân tích một thiết bị số tổ hợp. Với bài toán tổng hợp, phương pháp tổng hợp được trình bày có chú ý tới những hạn chế của thực tiễn. Đó là những hạn chế của các vi mạch như xây dựng mạch trên cơ sở các loại phần tử logic khác nhau, xây dựng các mạch logic với các vi mạch thực tế với số lượng các đầu vào bị hạn chế hay khả năng chịu tải có hạn. Những hạn chế này thường dẫn tới các mạch có nhiều cấp. Một thực tế khác của bài toán tổng hợp mạch là tổng hợp hệ hàm. Tối ưu hoá trong tổng hợp hệ hàm về độ đơn giản mạch và tăng tốc độ làm việc của mạch là mục tiêu của bài toán tổng hợp mạch nhiều cấp.

Trong phần này tài liệu cũng trình bày các phương pháp phân tích mạch tổ hợp dưới hai khía cạnh: Phân tích làm rõ chức năng của mạch (phân tích tĩnh) và phân tích làm rõ ảnh hưởng của việc giữ chậm tín hiệu của các phần tử logic tới hoạt động của mạch (phân tích động). Các bài toán phân tích, một mặt là công cụ để khai thác sử dụng và sửa chữa các thiết bị số. Mặt khác, bài toán phân tích lại là một khâu trong bài toán tổng hợp với tư cách là một phương pháp kiểm tra tính chân xác của chức năng mạch được thiết kế.

Phần III. Tổng hợp và phân tích thiết bị số có nhớ - Ôtômát hữu hạn

Trong phần này trình bày những kiến thức cơ bản về thiết bị số có nhớ. Chỉ rõ sự giống và khác nhau giữa thiết bị số có nhớ nói chung với ôtômát hữu hạn. Trên cơ sở đó, lý thuyết ôtômát hữu hạn đã được sử dụng để thực hiện bài toán tổng hợp và phân tích các thiết bị số có nhớ. Nhờ đó, các phương pháp phân tích và tổng hợp trình bày ở đây mang tính chặt chẽ và chuẩn xác, dễ dàng vận dụng một khi đã nắm vững.

Phương pháp tổng hợp được thực hiện trên cơ sở sử dụng các phần tử nhớ cơ bản có tính tới các đặc điểm của chúng. Cùng với việc chỉ ra các bước cơ bản cần thực hiện khi tổng hợp ôtômát hữu hạn, trong tài liệu cũng chỉ ra các bước quan trọng của bài toán tổng hợp, có ảnh hưởng nhiều đến những đặc trưng quan trọng của mạch như độ tin cậy về chức năng, độ đơn giản và tốc độ làm việc của thiết bị số.

Trong phần này còn đưa ra nhiều thí dụ về tổng hợp các mạch số thông dụng như những thí dụ về tổng hợp ôtômát hữu hạn. Đồng thời qua đó cũng làm rõ thêm tính vận năng của các phương pháp tổng hợp và phân tích đã trình bày.

Tác giả xin chân thành cảm ơn mọi đóng góp nhằm hoàn thiện hơn tài liệu này.

PHẦN I

CƠ SỞ TOÁN CỦA THIẾT BỊ SỐ

CHƯƠNG 1

HỆ ĐẾM VÀ MỘT SỐ PHƯƠNG PHÁP BIỂU DIỄN THÔNG TIN

1.1. CÁC HỆ ĐẾM

Sự phát triển của các hệ đếm gắn liền với sự phát triển của khoa học và kỹ thuật. Bên cạnh hệ đếm La Mã không phải là hệ đếm vị trí, các hệ đếm thông dụng còn lại đều là các hệ đếm vị trí. Các hệ đếm thông dụng là các hệ đếm cơ số 10, hệ đếm cơ số 2, cơ số 8 và cơ số 16.

Hệ đếm cơ số 10 là hệ đếm thông dụng và tiện lợi nhất trong đời sống con người. Mọi suy nghĩ của chúng ta về số lượng đều rõ ràng và dễ dàng cảm nhận khi chúng được thể hiện bằng con số dưới dạng cơ số 10. Nhưng hệ đếm cơ số 10 lại không thuận tiện khi thực hiện bằng kỹ thuật. Ngược lại, một số nhị phân (cơ số 2) lại rất dễ dàng thực hiện bằng kỹ thuật. Nhược điểm duy nhất của hệ nhị phân là để biểu diễn một con số ta cần nhiều chữ số hơn so với hệ cơ số 10. Để cách biểu diễn con số vừa gọn, dễ dàng chuyển đổi sang hệ nhị phân để thực hiện kỹ thuật, người ta sử dụng các hệ đếm có cơ số là 8 và 16. Tuy vậy, không ai thực hiện các phép tính trên cơ sở hệ đếm cơ số 8 và cơ số 16. Trong chương này chúng ta sẽ xét các hệ đếm vị trí thông dụng, quan hệ chuyển đổi giữa chúng và một vài phương pháp biểu diễn thông tin thường được sử dụng trong xử lý tin.

1.2. HỆ ĐẾM VỊ TRÍ

1.2.1. Khái niệm

Hệ đếm vị trí là hệ đếm mà mỗi chữ số của một số đã cho có giá trị cụ thể phụ thuộc vào vị trí của nó trong số đó. Giá trị cụ thể đó gọi là trọng số của nó.

Một số ở hệ đếm vị trí luôn có thể biểu diễn dưới dạng một đa thức của cơ số đếm với các trọng số là các hệ số tương ứng của các chữ số. Thí dụ số hệ 10 sau đây có thể viết như sau:

$$252,58 = 2 \cdot 10^2 + 5 \cdot 10^1 + 2 \cdot 10^0 + 5 \cdot 10^{-1} + 8 \cdot 10^{-2}$$

Mỗi chữ số, tùy thuộc vào vị trí của nó mà có giá trị khác nhau. Từ trái sang phải, số 2 đầu tiên có trọng số là 10^2 tức 100. Số 2 thứ hai có trọng số là 10^0 tức 1. Nghĩa là, do vị trí khác nhau trong số 252,58 mà số 2 thứ nhất có trọng số lớn gấp 100 lần trọng số của số 2 thứ hai... Như vậy, vị trí của chữ số quyết định trọng số của nó trong số đã cho.

1.2.2 Đặc trưng của các hệ đếm vị trí

Các hệ đếm vị trí có các đặc điểm sau đây:

- + Số lượng các chữ số khác nhau trong hệ đếm đúng bằng cơ số đếm của hệ đếm. Hệ đếm cơ số 2 chỉ có hai chữ số là chữ số 0 và 1; hệ đếm cơ số 10 có 10 chữ số là: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.
- + Chữ số lớn nhất kém cơ số 1 đơn vị.
- + Dưới dạng triển khai của một số, mỗi chữ số được nhân với trọng số của nó. Đó chính là cơ số có số mũ do trọng số của nó quyết định.
- + Nguyên tắc thực hiện các phép tính số học: Cộng, trừ, nhân, chia thực hiện theo nguyên tắc hoàn toàn như nhau, chỉ khác nhau về cơ số.

1.2.3. Hệ đếm cơ số 10

Là hệ đếm thông dụng nhất, có cơ số đếm là 10, sử dụng 10 chữ số khác nhau từ 0 đến 9. Dấu phẩy (,) dùng để phân biệt phần nguyên N_I và phần thập phân N_F của một số. Đối với một số được viết trong một hệ đếm vị trí, trong đó có hệ đếm cơ số 10, có thể viết dưới dạng tổng của phần nguyên và phần thập phân như sau:

$$N = N_I + N_F$$

Các phép tính số học thực hiện trong hệ đếm cơ số 10 chúng ta đã được biết từ bậc tiểu học.

1.2.4 Hệ đếm cơ số 2

Là hệ đếm vị trí đơn giản nhất, cơ số là 2 và sử dụng hai chữ số là 0 và 1. Số 1011,1101 ở hệ đếm cơ số 2 được biểu diễn dưới dạng đa thức:

$$1011,1101 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 1 \cdot 2^{-2} + 0 \cdot 2^{-3} + 1 \cdot 2^{-4}$$

Mười chữ số của hệ cơ số 10 viết trong hệ cơ số 2 được đưa ra trong bảng 1.1.

Các phép tính số học thực hiện trong hệ đếm cơ số 2 hoàn toàn tương tự như trong hệ đếm cơ số 10.

Thí dụ:

$$\begin{array}{r} + \text{ Phép cộng: } 1101 \\ + 1001 \\ \hline \end{array}$$

10110

$$\begin{array}{r} + \text{ Phép trừ: } 1101 \\ - 1001 \\ \hline \end{array}$$

0100

+ Phép nhân:

$$\begin{array}{r} 1101 \\ \times 101 \\ \hline \end{array}$$

1101

0000

1101

1000001

+ Phép chia:

$$\begin{array}{r|l} 1000001 & 1101 \\ 0001101 & 101 \\ \hline \end{array}$$

0000

Bảng 1.1. Mười chữ số của hệ 10 viết ở hệ 2

Cơ số 10	Cơ số 2
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001

1.2.5. Hệ cơ số 8 và cơ số 16

Là các hệ đếm vị trí có cơ số đếm là 8 (16) dùng 8 (16) chữ số là: 0,1,2,3,4,5,6,7 (và A, B, C, D, E, F).

Về nguyên tắc, các phép tính số học vẫn có thể thực hiện trong hệ cơ số 8 (16). Tuy vậy, do việc nhớ các bảng cửu chương ở hệ cơ số 8 và cơ số 16 rất khó, hệ đếm cơ số 8 và cơ số 16 thường chỉ dùng để ghi nhận thông tin mà không làm các phép tính trong các hệ đếm này. Việc ghi nhận thông tin dùng các hệ đếm cơ số 8 và 16 sẽ gọn hơn so với ghi nhận bằng các số nhị phân. Đồng thời việc chuyển đổi giữa các số viết ở hệ đếm cơ số 2 sang hệ đếm cơ số 8 hoặc 16 và ngược lại rất thuận tiện, dễ dàng và nhanh chóng.

1.3. CHUYỂN ĐỔI GIỮA CÁC HỆ ĐẾM

Việc chuyển đổi giữa các hệ đếm là một yêu cầu thường nhật. Lý thuyết thông tin đã chứng minh rằng, hệ đếm cơ số 16 là hệ đếm tối ưu về mặt tiết kiệm trong biểu diễn thông tin. Tuy vậy, việc thực hiện một hệ đếm cơ số 16 hay 10 trong thực tế kỹ thuật

rất không thuận tiện. Vì vậy, các thiết bị số được xây dựng để làm việc với các số cơ số 2 hay còn gọi là số nhị phân. Mặt khác, chúng ta dễ dàng cảm nhận độ lớn của sự vật hiện tượng khi các số đo của chúng được biểu diễn dưới dạng số hệ 10. Do đó việc chuyển đổi giữa các hệ đếm là không thể tránh khỏi.

Dưới đây chúng ta sẽ đưa ra phương pháp chuyển đổi chung giữa hai hệ đếm bất kỳ có cơ số đếm là α và β . Các cơ số α và β có thể là 2, 10, 8, 16 hay bất kỳ số nguyên dương nào khác.

1.3.1. Quy tắc chuyển đổi chung

a. Chuyển đổi phần N_I và phần N_F riêng

Số kết quả đã được chuyển đổi là tổng của hai số N_I và N_F đã được chuyển đổi.

b. Quy tắc đổi phần N_I

Muốn đổi phần N_I của một số viết ở hệ có cơ số α sang hệ có cơ số β , ta viết cơ số β ở hệ cơ số α , ký hiệu là $\beta(\alpha)$, rồi chia liên tiếp trong hệ α số N_I cho $\beta(\alpha)$ cho tới khi nào thương bằng 0 thì dừng. Số kết quả chuyển đổi sẽ là số gồm các chữ số (cần đổi sang hệ có cơ số β) là các số dư của các phép chia trên đây viết theo chiều ngược lại. Nghĩa là số dư sau viết trước và số dư trước viết sau.

Thí dụ 1.1

Đổi số $N_I(\alpha)$ viết ở hệ cơ số $\alpha = 2$ là $N_I(2) = 11001_2$ sang hệ cơ số $\beta = 10$. Thực hiện chuyển đổi theo quy tắc trên đây:

+ Viết cơ số $\beta = 10$ sang hệ cơ số 2 là: $\beta(\alpha) = 1010_2$

+ Chia liên tiếp ở hệ cơ số α :

$11001 : 1010$ được 1 dư 10; hạ nốt số 1 ta có 101 rồi chia tiếp cho 1010 ta được 0 dư 101. Kết quả phép chia thứ nhất cho thương là 10 và số dư là 101_2 tức 5_{10} . Chia tiếp $10 : 1010$ được 0 lần dư 10 tức 2_{10} .

Đến đây thì dừng lại.

Số kết quả chuyển đổi là :

$$N_{10} = 25_{10} \text{ hay } (11001)_2 = 25_{10}$$

Thí dụ 1.2

Chuyển số sau đây viết ở hệ cơ số 10 sang hệ cơ số 2: $N_I(10) = 11_{10}$

Thực hiện theo quy tắc trên đây:

+ Đổi cơ số 2 sang hệ cơ số 10, ta được: $\beta(\alpha) = 2$

+ Chia liên tiếp số 11_{10} cho 2_{10} :

$$\begin{array}{l} 11 : 2 \text{ được } 5, \text{ dư } 1. \text{ Dưới dạng nhị phân ta có số dư này bằng } 1_2 \\ 5 : 2 \text{ được } 2, \text{ dư } 1. \text{ Dưới dạng nhị phân ta có số dư này bằng } 1_2 \\ 2 : 2 \text{ được } 1, \text{ dư } 0. \text{ Dưới dạng nhị phân, số dư này bằng } 0_2 \\ 1 : 2 \text{ được } 0, \text{ dư } 1. \text{ Dưới dạng nhị phân số dư này bằng } 1_2 \end{array} \left| \begin{array}{l} \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \end{array} \right.$$

Vậy số kết quả này là 1011_2 hay $11_{10} = 1011_2$.

Thí dụ 1.3

Đổi số $N_1 (10) = 685_{10}$ sang hệ cơ số 16. Thực hiện theo quy tắc trên đây:

+ Đổi cơ số 16 sang hệ cơ số 10, ta được 16.

+ Chia liên tiếp:

$$\begin{array}{l} 685 : 16 \text{ được } 42, \text{ dư } 13_{10}, \text{ tức số D} \\ 42 : 16 \text{ được } 2, \text{ dư } 10_{10}, \text{ tức số A} \\ 2 : 16 \text{ được } 0, \text{ dư } 2_{10}, \text{ tức số 2} \end{array} \left| \begin{array}{l} \uparrow \\ \uparrow \\ \uparrow \end{array} \right.$$

Vậy số kết quả chuyển đổi là $2AD_{16}$ hay $685_{10} = 2AD_{16}$.

c. Nguyên tắc chuyển đổi phân thập phân N_F

Muốn đổi phân thập phân N_F của một số cho ở hệ cơ số α sang hệ cơ số β ta viết cơ số β ở hệ cơ số α , ký hiệu là $\beta(\alpha)$ rồi nhân liên tiếp trong hệ α số N_F với $\beta(\alpha)$ cho tới khi nào phần thập phân của tích bằng 0 hoặc độ chính xác cho phép (Trường hợp không bao giờ phần thập phân của phép nhân bằng 0) thì dừng. Số kết quả chuyển đổi sẽ là số gồm các phần nguyên của các phép nhân trên đây (đã đổi sang hệ cơ số β) viết theo chiều thuận, nghĩa là số nguyên trước viết trước, số nguyên sau viết sau.

Thí dụ 1.4

Đổi số $N_F = 0,8125_{10}$ sang hệ cơ số 2

Thực hiện quy tắc trên:

+ Đổi cơ số của hệ nhị phân sang hệ 10, ta được số 2

+ Nhân liên tiếp:

$$\begin{array}{l} 0,8125 \times 2 = 1,6250 \\ 0,6250 \times 2 = 1,2500 \\ 0,2500 \times 2 = 0,5000 \\ 0,5000 \times 2 = 1,0000 \end{array} \left| \begin{array}{l} \uparrow \\ \uparrow \\ \uparrow \\ \downarrow \end{array} \right.$$

Đến đây phần thập phân của tích bằng 0, ta dừng.

Số kết quả chuyển đổi là : $0,1101_2$ hay $0,8125_{10} = 0,1101_2$.

Thí dụ 1.5

Đổi số $N_1 = 0,12625_{10}$ sang hệ cơ số 8

+ Đổi cơ số 8 sang hệ cơ số 10 ta được 8

+ Nhân liên tiếp:

$$\begin{array}{rcl}
 0,12625 \times 8 & = & 1,01000 \\
 0,01000 \times 8 & = & 0,08000 \\
 0,08000 \times 8 & = & 0,64000 \\
 0,64000 \times 8 & = & 5,12000 \\
 0,12000 \times 8 & = & 0,96000 \\
 0,96000 \times 8 & = & 7,68000 \\
 0,68000 \times 8 & = & 5,44000 \\
 0,44000 \times 8 & = & 3,52020
 \end{array}$$

Vậy $(0,12625)_{10} = (0,10050753...)_{8}$.

1.3.2. Trường hợp đặc biệt

Nếu $\beta = \alpha^n$, trong đó n là số nguyên dương của hệ 10, thì muốn đổi một số N viết ở hệ α sang hệ β ta chỉ việc phân chia các chữ số của số N , tính từ dấu phẩy phân chia phần N_1 và N_2 về hai phía theo từng cụm n chữ số, nếu nhóm cuối cùng thiếu thì thêm số 0 vào cho đủ và thay mỗi cụm bằng một chữ số tương ứng của hệ β . Ngược lại, muốn đổi một số viết ở hệ β sang hệ α ta thay mỗi chữ số của số đã cho bằng một tổ hợp n chữ số tương ứng của hệ α , còn dấu phẩy giữ nguyên.

Thí dụ 1.6

Đổi số nhị phân sau đây sang hệ cơ số 8 và cơ số 16 và ngược lại:

$$N_2 = 101101,10101_2$$

Vì cơ số $8 = 2^3$ và cơ số $16 = 2^4$ nên xuất phát từ dấu phẩy của số N_2 ta phân các chữ số của nó thành từng cụm 3 và 4 chữ số tương ứng với việc đổi sang hệ cơ số 8 hay cơ số 16 và thay bằng các chữ số tương ứng của hệ 8 và hệ 16. Để đổi từ các số hệ cơ số 8 hoặc cơ số 16 sang số hệ cơ số 2 ta làm ngược lại: Thay mỗi số của hệ cơ số 8 hoặc cơ số 16 bằng một số nhị phân 3 hoặc 4 chữ số tương ứng như sau:

$$\begin{array}{ccccccc}
 \underline{1\ 0\ 1\ 1\ 0\ 1}, & \underline{1\ 0\ 1\ 0\ 1\ 0}_2 & & \underline{0\ 0\ 1\ 0\ 1\ 1\ 0\ 1}, & \underline{1\ 0\ 1\ 0\ 1\ 0\ 0\ 0}_2 \\
 \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow \\
 (5 & 5, & 5 & 2)_8 & & (2 & D, & A & 8)_{16}
 \end{array}$$

1.4. MỘT SỐ CÁCH BIỂU DIỄN THÔNG TIN THƯỜNG DÙNG TRONG CÁC HỆ XỬ LÝ TIN

1.4.1 Các đơn vị thông tin

Số nhị phân biểu diễn trong các hệ xử lý số liệu thường được tính theo các đơn vị: Bít (bit), Bai (byte), Từ (word).

Các đơn vị trên còn có các bội số của chúng là: Bít (b), kilôbít (Kb), Mêga bít (Mb); Bai (byte), Kilô bai (Kbyte), Mêga bai (Mbyte); Từ, Kilô từ và Mêga từ (Mword). Trong đó, một bai có tám bít, một từ có 16 bít và 1K(b/byte/word) bằng 2^{10} tức 1024 (b/byte/word) và 1 Mêga (b/byte/word) bằng 2^{10} tức 1024 (Kb/Kbyte/Kword).

Trong các byte hoặc word, bít đầu mút bên trái có trọng số cao nhất, viết tắt là MSB. Bít đầu mút bên phải là bít có trọng số nhỏ nhất, viết tắt là LSB.

1.4.2. Biểu diễn số nguyên không dấu

Với số không dấu, một byte tám bít có thể biểu diễn 256 số khác nhau, từ 00H tới FFH.

Một từ không dấu có thể biểu diễn 65536 số khác nhau, từ 0000H tới FFFFH, trong đó số nào có LSB = 0 là số chẵn.

1.4.3. Biểu diễn số nguyên có dấu

Một số có dấu có thể là số âm, số dương hay bằng 0. Trong biểu diễn thông tin dưới dạng số nhị phân, một số là số âm được thể hiện bằng MSB = 1; Nếu MSB = 0 thì số đó là số dương hoặc bằng 0.

1.4.4. Cốt ngược và cốt bù của một số nhị phân

a. Số ngược hay còn gọi là số bù 1: Số ngược của một số nhị phân là một số mà mỗi bít của nó là đảo của bít tương ứng của số đã cho.

Thí dụ $A = 10010100$ thì $A_{ng} = 01101011$

b. Số bù hay số bù 2: Số bù 2 của một số nhị phân là một số bằng số ngược của nó cộng thêm 1.

Thí dụ $B = 10011101$ thì $B_{ng} = 01100010$ và $B_{bù} = 01100010 + 1 = 01100011$.

Trong máy tính, một số âm được lưu giữ dưới dạng số bù 2 của giá trị tuyệt đối của nó.

Thí dụ số -97 được lưu giữ như sau:

- Số 97 viết ở hệ cơ số 16 là: 61H = 0110 0001

- Số bù của số 61H là: 1001 1110
 - Số bù 2 của số 61H là: 1001 1110 + 1 = 1001 1111, tức 9FH
- Cũng số này, nếu lưu giữ ở dạng số nhị phân 16 bit thì sẽ là số FF9FH.

c. Ứng dụng của cốt ngược và cốt bù

Trong xử lý thông tin, có thể dùng cốt ngược hay cốt bù để thay thế phép trừ bằng phép cộng. Mặt khác, các phép nhân, chia lại có thể thay thế bằng phép cộng và trừ. Do đó mọi phép tính đều có thể đưa về dạng phép cộng.

Hai trường hợp có thể xảy ra khi thay hiệu hai số nhị phân $A_1 - A_2$ bằng phép cộng.

+ Trường hợp $A_1 \geq A_2$

Trong trường hợp này, hiệu hai số nhận được bằng cách thực hiện $A_1 + A_{2\text{bù}}$ hoặc $A_1 + A_{2\text{ng}}$, nhưng nếu xuất hiện số nhớ ở bit cao nhất (MSB) thì:

- Bỏ qua số nhớ đó nếu dùng cốt bù.
- Cộng số nhớ đó vào kết quả nếu dùng cốt ngược.

Thí dụ $A_1 = 01011$, $A_2 = 00101$. Vì $A_1 > A_2$ nên hiệu $A_1 - A_2$ thực hiện qua cốt ngược và cốt bù như sau:

$$\begin{array}{r}
 A_1 = 01011 \\
 + \\
 A_{2\text{bù}} = 11011 \\
 \hline
 A_1 + A_{2\text{bù}} = 100110. \text{ Vậy, } A_1 - A_2 = 00110 \text{ hay } +6 \\
 \downarrow \\
 \text{Số nhớ bỏ qua}
 \end{array}$$

Hoặc:

$$\begin{array}{r}
 A_1 = 01011 \\
 + \\
 A_{2\text{ng}} = 11010 \\
 \hline
 A_1 + A_{2\text{ng}} = 100101. \text{ Lấy số nhớ cộng vào kết quả, ta có:} \\
 \downarrow \\
 \text{Số nhớ } + 1 \\
 \hline
 00110. \text{ Vậy } A_1 - A_2 = 00110 \text{ hay } +6.
 \end{array}$$

Kết quả thực hiện phép trừ trên đây bằng hai phương pháp dùng cốt bù và cốt ngược là như nhau.

+ Trường hợp $A_1 < A_2$

Trong trường hợp này, hiệu thu được thông qua phép cộng với số $A_{2\text{bù}}$ hoặc $A_{2\text{ng}}$ sẽ còn ở dạng cốt bù và cốt ngược một cách tương ứng. Do đó muốn có kết quả ở dạng nhị phân cần đổi kết quả đó sang cốt nhị phân.

Thí dụ:

* Thực hiện $A_1 - A_2$ với $A_1 = 00111$ và $A_2 = 01011$ dùng số bù:

$$\begin{array}{r} A_1 = 00111 \\ + \\ A_{2bù} = 10101 \\ \hline A_1 + A_{2bù} = 11100_{bù} \end{array}$$

Đổi lại, ta có số nhị phân tương ứng: 10100_2 , tức số -4 . Chú ý, để giữ nguyên dấu, ta không đổi dấu của kết quả.

* Thực hiện phép trừ trên đây dùng số ngược:

$$\begin{array}{r} A_1 = 00111 \\ + \\ A_{2ng} = 10100 \\ \hline A_1 + A_{2ng} = 11011_{ng} \end{array}$$

Đổi lại, ta có 10100 , tức số -4 .

1.4.5. Cốt 2-10 hay còn gọi là mã BCD (Binary Coded Decimals)

Trong thực tế thường phải dùng số BCD, trong đó 10 con số của hệ cơ số 10 được biểu diễn ở dạng số nhị phân 4 bit tương ứng. Bảng 1.2 cho một số loại mã thông dụng của 10 con số của hệ đếm cơ số 10.

Bảng 1.2. Một số loại mã (cốt)

Số hệ 10	Cốt BCD	Cốt 2421	Cốt dư 3	Cốt Grey	Cốt ASCII
0	0000	0000	0011	0000	00110000
1	0001	0001	0100	0001	00110001
2	0010	0010	0101	0011	00110010
3	0011	0011	0110	0010	00110011
4	0100	0100	0111	0110	00110100
5	0101	1011	1000	0111	00110101
6	0110	1100	1001	0101	00110110
7	0111	1101	1010	0100	00110111
8	1000	1110	1011	1100	00111000
9	1001	1111	1100	1000	00111001

Một số hệ cơ số 10 gồm nhiều con số, khi chuyển sang số BCD, mỗi con số được thay bằng một số nhị phân 4 bit tương ứng.

Thí dụ số 59_{10} viết dưới dạng số BCD là: $0101\ 1001_{\text{BCD}}$

Với 4 bit nhị phân ta có thể có 16 tổ hợp khác nhau, như vậy sẽ thừa 6 tổ hợp. Khi ta thực hiện các phép tính cộng hay trừ đối với số BCD sẽ xảy ra trường hợp tổng hay hiệu của chúng vượt quá giá trị 9, tức ra ngoài phạm vi sử dụng của số BCD. Trong trường hợp này, muốn kết quả vẫn ở dạng số BCD, ta cần cộng thêm hoặc trừ đi số hiệu chỉnh, gọi là số hiệu chỉnh BCD. Đó là số 6, tức 0110 .

Thí dụ cộng hai số BCD sau đây:

$$\begin{array}{r} 5_{10} = 0101_{\text{BCD}} \\ + \\ 8_{10} = 1000_{\text{BCD}} \end{array}$$

$13 = 1101$ là số 13 dưới dạng số nhị phân, không phải là số BCD.

Muốn kết quả ở dạng số BCD, cần cộng thêm số 0110 vào kết quả như sau:

$1101 + 0110 = 0001\ 0011$, tức số 13 dưới dạng số BCD.

Khi thực hiện phép trừ hai số BCD sau đây: $36_{10} - 19_{10}$

$$\begin{array}{r} 36_{10} = 0011\ 0110_{\text{BCD}} \\ - \\ 19_{10} = 0001\ 1001_{\text{BCD}} \end{array}$$

$17 = 0001\ 1101$, tức kết quả không phải là một số BCD.

Muốn có kết quả là một số BCD đúng, ta phải trừ kết quả đi số 6 tại tổ hợp 4 chữ số cho kết quả > 9 . Cụ thể:

$$\begin{array}{r} 0001\ 1101 \\ - \\ 0110 \\ \hline 0001\ 0111_{\text{BCD}} \end{array}$$

Khi thực hiện cộng hai số BCD sau đây:

$$\begin{array}{r} 49,08_{10} = 0100\ 1001, 0000\ 1000_{\text{BCD}} \\ + \\ 89,18_{10} = 1000\ 1001, 0001\ 1000_{\text{BCD}} \\ \hline 138,26_{10} = 1101\ 0010, 0010\ 0000 \end{array}$$

Kết quả cũng không phải là số BCD của số $138,26_{10}$. Trong trường hợp này, để có số BCD đúng, cần cộng thêm số hiệu chỉnh BCD vào các cột BCD có tổng > 9 hay có nhớ sang số BCD bên trái nó. Cụ thể:

$$\begin{array}{r} 138,26_{10} = 1101\ 0010, 0010\ 0000_{\text{BCD}} \\ + \quad 0110\ 0110 \quad 0110 \\ \hline 1\ 0011\ 1000, 0010\ 0110_{\text{BCD}} \end{array}$$

1.4.6. Mã ASCII

Mã ASCII là mã trao đổi thông tin chuẩn của Mỹ (American Standard Codes for Information Interchanges). Loại mã này sử dụng rộng rãi trong hầu hết các loại máy tính và các thiết bị xử lý tin hiện đại, trong đó gồm mã của tất cả các chữ viết thường và viết hoa, các ký hiệu thường sử dụng trong thông tin. Mã ASCII thông dụng cho trong bảng 1.3 dưới đây.

Bảng 1.3. Mã ASCII

	0H	1H	2H	3H	4H	5H	6H	7H
0H	<NUL>	<DLE>	<SP>	0	@	P		p
1H	<SOH>	<DC1>	!	1	A	Q	a	q
2H	<SIX>	<DC2>	"	2	B	R	b	r
3H	<ETX>	<DC3>	#	3	C	S	c	s
4H	<EOT>	<DC4>	\$	4	D	T	d	t
5H	<ENQ>	<NAK>	%	5	E	U	e	u
6H	<ACK>	<SYN>	&	6	F	V	f	v
7H	<BEL>	<ETB>	'	7	G	W	g	w
8H	<BS>	<CAN>	(	8	H	X	h	x
9H	<HT>		)	9	I	Y	i	y
AH	<LF>	<SUB>	*	:	J	Z	j	z
BH	<VT>	<ESC>	+	;	K	[	k	{
CH	<FF>	<FS>	,	<	L	\	l	
DH	<CR>	<GS>	-	=	M	]	m	}
EH	<SO>	<RSI>	.	>	N	^	n	
FH	<SI>	<US>	/	?	O	-	o	

Thí dụ Mã ASCII của số 5 là 35H hay 0011 0101

Mã ASCII của số 9 là 39H hay 0011 1001.

1.4.7. Quan hệ giữa số BCD và số ASCII của một số

Trong khi làm các phép tính với số BCD, các số hệ 10 có nhiều chữ số được hình thành dưới dạng BCD bằng cách ghép các số BCD lại với nhau. Thí dụ, số 36_{10} được viết là: 00110110_{BCD} . Cách biểu diễn như thế gọi là dạng BCD chuẩn hay dạng BCD gói (Packed BCD). Khi đưa các số vào từ máy tính, truyền hay lưu trữ chúng, ta lại dùng dạng ASCII của chúng. Như trong mã ASCII đã thể hiện rõ, mỗi chữ số của hệ 10 được

mã bằng tám bit mã nhị phân, trong đó bốn bit cao là mã BCD của số 3_{10} . Bốn bit tiếp thấp hơn là số BCD của chính số đó. Vì vậy, nếu ta cho bốn bit cao của mã ASCII của một số bằng 0 thì ta được số đó ở dạng cốt BCD. Trong trường hợp này ta có dạng BCD không gói của số đã cho.

Thí dụ: Mã ASCII của 4 là $0011\ 0100 = 34H$

Mã ASCII của 7 là $0011\ 0111 = 37H$

Mã BCD không gói của 4 là $0000\ 0100 = 04H$

Mã BCD không gói của 7 là $0000\ 0111 = 07H$

Trong máy tính, khi chuyển từ một số ASCII sang số BCD chuẩn như sau: Ta cho bốn bit cao của các số ASCII bằng 0 còn giữ lại bốn bit thấp của chúng, rồi ghép chúng lại theo thứ tự trọng số của chúng.

BÀI TẬP CHƯƠNG 1

- Đổi các số sau đây:
 - $10110011, 101_2$ sang hệ đếm cơ số 10
 - Các số 47_{10} , $0,37_{10}$ và $25,51_{10}$ sang hệ đếm cơ số 2
- Thực hiện các phép tính sau đây trong hệ đếm cơ số 2:
 - $10011101 + 01100110$; $10010110 + 00111101$
 - $1101 - 1001$; $11011001 - 10110111$
 - $1001,0111 + 101,0110$
 - $10001110.1010 - 01101010.0011$
- Đổi các số sau đây từ hệ cơ số 8 sang hệ đếm cơ số 2 và sau đó sang hệ đếm cơ số 10:
 - 57_8 ; 25_8 ; $12,068_8$; $0,103_8$
 - $12345,9857_8$; $7864,0352_8$
- Đổi các số sau đây sang hệ đếm cơ số 2 và sau đó sang hệ đếm cơ số 10:
 - $18A_{16}$; $C83_{16}$; $2B,56_{16}$; $5E3,D5_{16}$
 - $156B9C,2A7_{16}$
- Biểu diễn các số không dấu sau đây dưới dạng cốt ngược và cốt bù:
 - 11001001_2 ; 00111011_2
 - 01100101_2 ; 11101011_2
- Biểu diễn các số có dấu sau đây dưới dạng cốt ngược và cốt bù:
 - -25_{10}
 - $+432_{10}$
 - -7321_{10}

7. Thực hiện các phép tính sau đây bằng cách sử dụng cốt ngược và cốt bù:
- a) $946 - 503$
 - b) $257 - 782$
8. Thực hiện các phép tính dùng cốt BCD:
- a) $2449 + 2168$; $3695 + 7763$; $3582,036 + 3527,963$
 - b) $358 - 264$; $3378 - 6241$; $1459,461 - 3528,447$.

Chương 2

CÁC HÀM LÔGIC CƠ BẢN VÀ TÍNH CHẤT

2.1. KHÁI NIỆM VỀ ĐẠI SỐ LÔGIC

2.1.1. Các thuật toán logic cơ bản

Trong lĩnh vực logic học, mọi sự kiện, hiện tượng hay khẳng định đều được đánh giá chỉ bằng một trong hai giá trị: Thực và giả hay đúng và sai. Một sự kiện chỉ có thể là đúng hay sai, không thể có khái niệm trung gian nào khác. Nếu ta nói tam giác ABC là tam giác cân thì khẳng định đó nhận giá trị Thực, nếu tam giác ABC đúng là tam giác cân, còn nếu nó không là tam giác cân mà là một trong các loại tam giác khác thì câu khẳng định trên đây nhận giá trị Giả.

Các khẳng định trong logic học được chia thành các khẳng định đơn giản và các khẳng định phức tạp. Các khẳng định đơn giản là các khẳng định chỉ khẳng định một ý, không phụ thuộc vào bất kỳ ý hay điều kiện nào khác. Thí dụ, hôm nay trời mưa; diện tích S_1 lớn hơn diện tích S_2 ...

Một khẳng định đơn giản có thể được thể hiện bằng một ký hiệu nào đó, như x hay y ... Đối với các khẳng định đơn giản trong logic học tồn tại một thuật toán cơ bản và quan trọng sau đây:

Đối với mọi khẳng định đơn giản đều có thể tìm thấy một khẳng định đơn giản khác là phủ định của nó.

Nếu ta nói “Hôm nay trời mưa” và ký hiệu khẳng định này là x thì khẳng định: “Hôm nay trời không mưa”, ký hiệu là y , sẽ là phủ định của khẳng định x . Khẳng định “Hôm nay trời nắng” hay “Hôm nay trời râm mát” không chắc chắn là phủ định của khẳng định đã nêu trên đây.

Thuật toán phủ định được biểu diễn như sau:

$$y = \bar{x}$$

Đây là một trong ba thuật toán cơ bản, quan trọng nhất của logic học.

Khẳng định phức tạp là một khẳng định trong đó thể hiện sự phụ thuộc của một khẳng định này vào các khẳng định khác. Khẳng định phức tạp có hai dạng cơ bản là khẳng định phức tạp loại tuyển và loại hội.

Một khẳng định phức tạp loại tuyển là khẳng định trong đó sự kiện chính phụ thuộc vào các khẳng định thành phần theo nguyên tắc sau đây: Khẳng định chính sẽ là thực khi ít nhất một trong các khẳng định thành phần là thực.

Thí dụ 2.1: Tôi sẽ đi tham quan nếu có xe của cơ quan hoặc mua được xe máy.

Khẳng định đi tham quan là khẳng định chính, ký hiệu là y ; khẳng định có xe cơ quan và khẳng định mua được xe máy là các khẳng định thành phần, ký hiệu là x_1 và x_2 một cách tương ứng. Khi đó sự phụ thuộc theo phép tuyển được biểu diễn như sau:

$$y = x_1 \vee x_2$$

Bảng giá trị thực của phép tuyển cho trên bảng 2.1a.

Bảng 2.1. Bảng giá trị thực của phép tuyển và phép hội: S - sai; Đ - đúng

x_1	x_2	y	x_1	x_2	y
S	S	S	S	S	S
S	Đ	Đ	S	Đ	S
Đ	S	Đ	Đ	S	S
Đ	Đ	Đ	Đ	Đ	Đ

a)

b)

Phép tuyển còn được gọi là phép cộng logic và để cho tiện, người ta dùng ngay ký hiệu của phép cộng số học để thay cho ký hiệu phép tuyển $\vee$. Ta viết:

$$y = x_1 + x_2$$

Một khẳng định phức tạp loại hội là khẳng định trong đó sự kiện chính phụ thuộc vào các khẳng định thành phần theo nguyên tắc sau đây: Khẳng định chính sẽ là thực khi và chỉ khi tất cả các sự kiện thành phần đều là thực.

Thí dụ 2.2: Tôi sẽ đi du lịch nước ngoài nếu có hộ chiếu và có tiền.

Khẳng định đi du lịch nước ngoài là khẳng định chính, ký hiệu là y . Các khẳng định thành phần là có hộ chiếu và có tiền, ký hiệu bằng x_1 và x_2 một cách tương ứng. Khi đó y phụ thuộc vào x_1 và x_2 theo phép hội và được biểu diễn dưới dạng biểu thức toán như sau:

$$y = x_1 \wedge x_2$$

Bảng giá trị thực của phép hội cho trên bảng 2.1b.

Phép hội thường còn gọi là phép nhân logic, nên từ đây, để ký hiệu phép hội ta dùng dấu nhân số học “.” là một dấu chấm hay không viết dấu gì như sau:

$$y = x_1 \cdot x_2 \text{ hay } y = x_1 x_2$$

Trong logic học, ba phép toán: Phủ định, tuyển và hội là ba phép toán cơ bản và tạo thành một hệ hàm đủ. Chỉ bằng các hàm logic cơ bản này, ta có thể biểu diễn bất kỳ một khẳng định hay sự kiện phức tạp nào.

2.1.2. Đại số Boole

Đại số Boole là đại số logic trong đó hai giá trị Thực và Giả được mã hoá bằng hai con số “1” và “0”.

Khi áp dụng đại số Boole trong kỹ thuật thì hai giá trị “1” và “0” được dùng để thể hiện các đại lượng vật lý khác nhau. Thí dụ: Có điện áp là giá trị “1”. Không có điện áp là giá trị “0”. Tần số cao là giá trị “1”; tần số thấp là giá trị “0”, v.v...

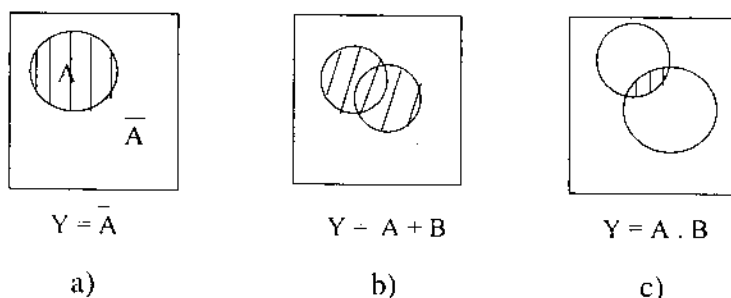
Những quy định cụ thể của các giá trị vật lý tương ứng với giá trị logic “1” và “0” sẽ được đề cập tới trong phần nói về tín hiệu và mạch logic.

Với các mã của các giá trị logic như trên, các phép toán cơ bản của đại số Boole được mô tả như sau:

Phép phủ định	Phép cộng logic	Phép nhân logic
$\bar{0} = 1$	$0 + 0 = 0$	$0 \cdot 0 = 0$
$\bar{1} = 0$	$0 + 1 = 1$	$0 \cdot 1 = 0$
$\bar{0} = 0$	$1 + 0 = 1$	$1 \cdot 0 = 0$
$\bar{1} = 1$	$1 + 1 = 1$	$1 \cdot 1 = 1$

2.1.3. Biểu diễn các phép toán logic cơ bản dùng giản đồ Venne

Như ta đã biết, không gian giá trị của đại số logic là $\{0, 1\}$. Nếu ta biểu diễn không gian giá trị của các đại lượng logic bằng diện tích hình vuông còn các khẳng định A, B, C là diện tích các hình tròn thì theo giản đồ Venne, các phép phủ định, phép cộng logic và phép nhân logic có thể biểu diễn bằng các phần gạch chéo như trên hình 2.1a, b, c tương ứng dưới đây.



Hình 2.1. Giản đồ Venne của các phép phủ định, cộng và nhân logic

2.1.4. Biến và hàm logic

Biến logic là một biến chỉ nhận một trong hai giá trị 0 và 1. Biến logic có thể ký hiệu bằng một ký tự hay một ký hiệu nào đó. Thí dụ: $x_i, i = \overline{1, n}$.

Hàm logic là một hàm phụ thuộc vào các biến logic và bản thân hàm logic cũng chỉ có thể nhận một trong hai giá trị là 0 hoặc 1. Hàm logic có thể ký hiệu là y . Khi đó hàm y phụ thuộc n biến logic có thể viết như sau:

$$y = f(x_i), i = \overline{1, n}$$

Đối với hàm logic, các biến logic có thể được sắp xếp theo thứ tự tăng dần hoặc giảm dần của các chỉ số. Thí dụ: $x_1 x_2 x_3 \dots x_n$ hay $x_n x_{n-1} x_{n-2} \dots x_1$. Một khi thứ tự các biến đã được xác định thì trong quá trình làm việc với hàm đã cho thứ tự đó không được thay đổi.

Đối với một thứ tự nhất định các biến, gán cho mỗi biến một giá trị cụ thể ta sẽ có một tập giá trị biến. Nếu có tất cả n biến thì số tập giá trị biến khác nhau của n biến là N mà:

$$N = 2^n$$

Với $n = 3$ ta có $N = 2^3 = 8$ tập giá trị biến khác nhau như bảng 2.2.

Bảng 2.2. Tám tập giá trị biến của ba biến x_1, x_2, x_3

x_1	x_2	x_3
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Một câu hỏi đặt ra là có bao nhiêu hàm logic n biến khác nhau? Vì với n biến ta có N tập giá trị biến và đối với mỗi tập giá trị biến hàm chỉ có thể nhận một trong hai giá trị là 0 và 1. Nếu ký hiệu số lượng các hàm logic khác nhau của n biến là M , ta có:

$$M = 2^N = 2^{2^n}$$

Quan hệ giữa M và n là quan hệ phụ thuộc theo hàm số mũ. Do đó M tăng rất nhanh khi số biến n tăng. Thí dụ:

$$\begin{aligned} n = 1 \text{ thì } N = 2^1 = 2 \text{ và } M = 2^2 = 4 \\ n = 2 \text{ thì } N = 2^2 = 4 \text{ và } M = 2^4 = 16 \\ n = 3 \text{ thì } N = 2^3 = 8 \text{ và } M = 2^8 = 256 \\ n = 4 \text{ thì } N = 2^4 = 16 \text{ và } M = 2^{16} = 65536 \end{aligned}$$

Khác với số học hay đại số thường gặp, trong đại số logic việc so sánh hai hàm số có các đặc điểm sau đây:

- Chỉ so sánh hai hàm có cùng số biến.

- Không có khái niệm hàm này lớn hơn hay nhỏ hơn hàm kia, chỉ có khái niệm hai hàm logic giống nhau hay khác nhau mà thôi.

Hai hàm logic f_1 và f_2 gọi là giống nhau hay như nhau khi chúng nhận những giá trị như nhau đối với mọi tập giá trị biến của chúng. Ngược lại, chúng là khác nhau khi chỉ cần chúng nhận các giá trị khác nhau đối với một tập giá trị biến. Trong trường hợp này ta có thể nói chúng giống nhau từng phần.

Một khái niệm quan trọng khác là khái niệm hàm xác định đầy đủ hay không đầy đủ. Một hàm logic gọi là xác định đầy đủ nếu nó nhận các giá trị xác định (0 hay 1) đối với mọi tập giá trị biến. Hàm logic gọi là xác định không đầy đủ nếu dù chỉ đối với một tập giá trị biến nó không có giá trị xác định. Trong trường hợp hàm xác định không đầy đủ ta nói hàm là bất định đối với tập hay những tập giá trị biến mà giá trị của nó là không xác định.

Một hàm logic $f(x_1, x_2, \dots, x_n)$ gọi là không phụ thuộc vào biến x_i nếu thỏa mãn điều kiện:

$$f(x_1, x_2, \dots, x_{i-1}, 0, x_{i+1}, \dots, x_n) = f(x_1, x_2, \dots, x_{i-1}, 1, x_{i+1}, \dots, x_n)$$

Các hàm logic luôn luôn nhận giá trị “0” hay “1” gọi là các hằng số logic.

Các hàm logic phức tạp, nhiều biến luôn luôn có thể biểu diễn qua các hàm logic một và hai biến và các hằng số logic. Nói cách khác, các hàm logic tuân theo định lý xếp chồng.

2.2. CÁC HÀM LÔGIC CƠ BẢN

Hàm logic cơ bản là các hàm có 1 hoặc 2 biến. Gọi các hàm này là các hàm cơ bản vì tất cả các hàm logic khác, dù phức tạp đến đâu cũng có thể biểu diễn qua các hàm này. Hiểu chức năng, tính chất của các hàm logic cơ bản sẽ giúp ta dễ dàng thiết lập và biến đổi các hàm phức tạp khác. Dưới đây chúng ta sẽ xét các hàm với $n = 0$; $n = 1$ và $n = 2$.

2.2.1. Hàm logic không biến ($n = 0$)

Với $n = 0$, ta có $N = 2^0 = 1$ và $M = 2^1 = 2$. Hai hàm logic không biến này là hai hằng số logic: $y = 0$ và $y = 1$.

2.2.2. Hàm logic một biến ($n = 1$)

Với $n = 1$, $N = 2$ và $M = 4$. Các hàm logic một biến này cho trong bảng 2.3.

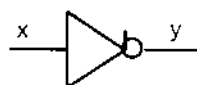
Bảng 2.3. Các hàm logic một biến

Hàm Biến x	y_0	y_1	y_2	y_3
0	0	0	1	1
1	0	1	0	1

Nhận xét: Hàm y_0 và hàm y_3 là các hằng số logic. Hàm y_1 luôn lặp lại giá trị của x , tức $y_1 = x$. Hàm y_2 luôn có giá trị là đảo của giá trị của biến x gọi là hàm đảo hay hàm phủ định, nó thực hiện thuật toán phủ định. Dưới dạng phương trình toán học ta có:

$$y_2 = \bar{x}$$

Hàm đảo là một trong các hàm logic cơ bản và rất quan trọng. Vì vậy trong thực tế hàm này được sản xuất dưới dạng các vi mạch tích hợp (IC – Integrated Circuits). Ký hiệu của phần tử mạch đảo như trên hình 2.2.



Hình 2.2. Ký hiệu mạch đảo

2.2.3. Hàm logic hai biến ($n = 2$)

Với $n = 2$, $N = 4$ ta có $M = 16$ hàm logic khác nhau – bảng 2.4.

Bảng 2.4. 16 hàm hai biến

Hàm $x_1 \ x_2$	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}	y_{11}	y_{12}	y_{13}	y_{14}	y_{15}
0 0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0 1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1 0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1 1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Các hàm hai biến này có tên, ký hiệu và biểu thức toán học tương ứng như ở bảng 2.5.

Dễ dàng nhận thấy rằng, tám hàm đầu là phủ định của tám hàm sau và ngược lại theo từng đôi một. Trong đó:

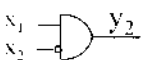
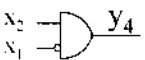
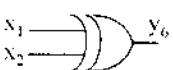
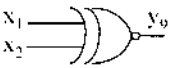
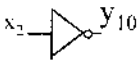
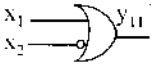
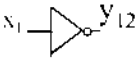
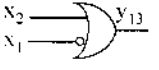
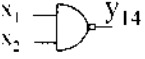
+ Hàm y_0 và y_{15} là các hằng số logic “0” và “1”.

+ Hàm y_1 là nhân logic và y_{14} là hàm phủ định của nhân logic, hay còn gọi là hàm VA-KH.

+ Hàm y_2 gọi là hàm Cấm, ở đây là cấm theo x_2 , và y_{13} là hàm kéo theo, x_1 kéo theo x_2 , có chức năng là phủ định của hàm y_2 .

+ Các hàm y_3 và y_{12} không phụ thuộc vào x_2 . Hàm y_3 bằng x_1 và hàm y_{12} là phủ định của x_1 .

Bảng 2.5. Tên, ký hiệu và biểu thức toán của các hàm logic cơ bản

Hàm	Tên gọi	Ký hiệu	Biểu thức
y_0	Hằng số "0"		$y_0 = 0$
y_1	Nhân logic		$y_1 = x_1 \cdot x_2$
y_2	Hàm cấm theo x_2		$y_2 = x_1 \cdot \bar{x}_2$
y_3	Hàm x_1		$y_3 = x_1$
y_4	Hàm cấm theo x_1		$y_4 = \bar{x}_1 \cdot x_2$
y_5	Hàm x_2		$y_5 = x_2$
y_6	Hàm cộng môđun 2		$y_6 = x_1 \oplus x_2$
y_7	Hàm cộng logic		$y_7 = x_1 + x_2$
y_8	Hàm Hoặc-KH		$y_8 = \overline{x_1 + x_2}$
y_9	Hàm đẳng trị		$y_9 = \overline{x_1 \oplus x_2}$
y_{10}	Hàm đảo của x_2		$y_{10} = \bar{x}_2$
y_{11}	x_2 kéo theo x_1		$y_{11} = x_1 + \bar{x}_2$
y_{12}	Hàm đảo của x_1		$y_{12} = \bar{x}_1$
y_{13}	x_1 kéo theo x_2		$y_{13} = \bar{x}_1 + x_2$
y_{14}	Hàm VA - KH		$y_{14} = \overline{x_1 \cdot x_2}$
y_{15}	Hằng logic "1"		$y_{15} = 1$

+ Hàm y_1 là hàm cấm theo x_1 . Hàm y_{11} là hàm kéo theo, x_2 kéo theo x_1 , và là đảo của hàm y_4 . Chúng tương tự như cặp hàm y_4 và y_{12} .

+ Hai hàm y_3 và y_{10} là hàm x_2 và đảo của x_2 .

+ Hàm y_6 là hàm cộng môđun hai hay còn gọi là hàm bất đẳng trị. Đảo của hàm bất đẳng trị là hàm đẳng trị y_9 .

+ Hàm y_7 là hàm cộng logic. Hàm đảo của hàm cộng logic là hàm y_8 . Hàm y_8 còn có tên gọi là hàm HOAC – KH.

Trong số các hàm trên đây, 5 hàm quan trọng nhất, có nhiều ý nghĩa và ứng dụng thực tiễn nhiều nhất là các hàm: y_1 , y_{14} , y_7 , y_8 và y_6 . Chúng ta sẽ xét các hàm này một cách chi tiết hơn.

2.3. CÁC HÀM LÔGIC QUAN TRỌNG

2.3.1. Hàm y_7

Hàm y_7 là hàm cộng logic. Bảng giá trị thực của y_7 cho trên bảng 2.6a. Chúng ta có thể định nghĩa: Hàm cộng logic là hàm chỉ nhận giá trị 0 khi tất cả các biến của nó đồng thời bằng 0 hay hàm cộng logic sẽ nhận giá trị 1 khi ít nhất một biến của nó có giá trị 1.

Biểu thức toán học của phép cộng logic là:

$$y_7 = x_1 + x_2$$

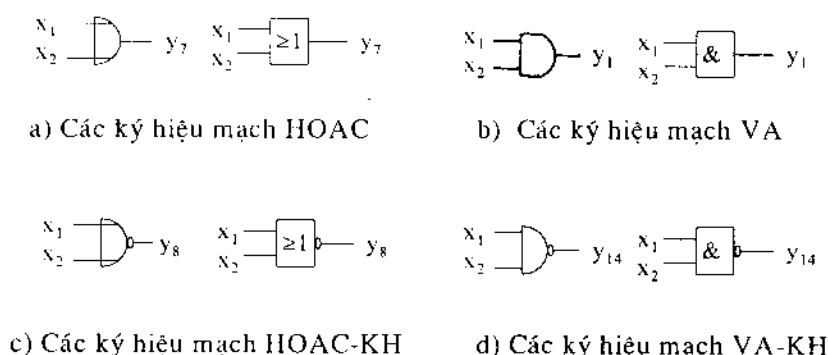
Mạch thực hiện hàm y_7 gọi là mạch cộng logic hay mạch HOAC (OR).

Mạch cộng logic là một mạch số logic cơ bản. Vì vậy loại mạch này được sản xuất hàng loạt dưới dạng các IC với số lượng các đầu vào khác nhau, thông thường từ 2 đến 5 đầu vào.

Mạch cộng logic có ký hiệu khác nhau tùy theo hãng sản xuất. Ngoài ký hiệu cho trong bảng 2.5, mạch HOAC còn có các ký hiệu như ở hình 2.3a.

Bảng 2.6. Bảng giá trị thực của các hàm cộng logic, nhân logic, VA – KH, HOAC – KH, và cộng môđun 2

x_1	x_2	y_7	x_1	x_2	y_7	x_1	x_2	y_8	x_1	x_2	y_{14}	x_1	x_2	y_6
0	0	0	0	0	0	0	0	1	0	0	1	0	0	0
0	1	1	0	1	0	0	1	0	0	1	1	0	1	1
1	0	1	1	0	0	1	0	0	1	0	1	1	0	1
1	1	1	1	1	1	1	1	0	1	1	0	1	1	0
a)			b)			c)			d)			e)		



Hình 2.3. Các ký hiệu mạch logic

2.3.2. Hàm y_1

Hàm y_1 là hàm nhân logic. Bảng giá trị thực của y_1 cho trên bảng 2.6b. Chúng ta có thể định nghĩa: Hàm nhân logic là hàm chỉ nhận giá trị 1 khi tất cả các biến của nó đồng thời bằng 1 hay hàm nhân logic sẽ nhận giá trị 0 khi ít nhất một biến của nó có giá trị 0.

Biểu thức toán học của phép nhân logic là:

$$y_1 = x_1 \cdot x_2$$

Mạch thực hiện hàm y_1 gọi là mạch nhân logic hay mạch VA (AND).

Mạch nhân logic là một mạch số logic cơ bản. Vì vậy loại mạch này được sản xuất hàng loạt dưới dạng các IC với số lượng các đầu vào khác nhau, thông thường từ 2 đến 5 đầu vào. Mạch có số lượng đầu vào lớn rất hiếm.

Mạch nhân logic có ký hiệu khác nhau tùy theo hãng sản xuất. Ngoài ký hiệu cho trên bảng 2.5 còn dùng các ký hiệu như ở hình 2.3b.

2.3.3. Hàm y_8

Hàm y_8 là hàm phủ định của cộng logic hay còn gọi là hàm HOAC – KH (NOR). Bảng giá trị thực của y_8 cho trên bảng 2.6c. Chúng ta có thể định nghĩa: Hàm HOAC-KH là hàm chỉ nhận giá trị 1 khi tất cả các biến của nó đồng thời bằng 0 hay hàm HOAC-KH sẽ nhận giá trị 0 khi ít nhất một biến của nó có giá trị 1. Hàm HOAC-KH có ký hiệu toán học là một mũi tên thẳng đứng “ $\downarrow$ ”.

Biểu thức toán học của phép phủ định cộng logic là:

$$y_8 = \overline{x_1 + x_2} \quad \text{hay} \quad y_8 = x_1 \downarrow x_2$$

Mạch thực hiện hàm y_8 gọi là mạch HOAC – KH (mạch NOR).

Mạch HOAC-KH là một mạch số logic cơ bản. Vì vậy loại mạch này được sản xuất hàng loạt dưới dạng các IC với số lượng các đầu vào khác nhau, thông thường từ 2 đến 5 đầu vào. Mạch có số lượng đầu vào lớn rất hiếm.

Mạch HOAC-KH có ký hiệu khác nhau tùy theo hãng sản xuất, ngoài ký hiệu cho trên bảng 2.5 còn dùng các ký hiệu như ở hình 2.3c.

2.3.4. Hàm y_{14}

Hàm y_{14} là hàm phủ định của nhân logic hay còn gọi là hàm VA – KH (NAND). Bảng giá trị thực của y_{14} cho trên bảng 2.6d. Chúng ta có thể định nghĩa: Hàm VA–KH là hàm chỉ nhận giá trị 0 khi tất cả các biến của nó đồng thời bằng 1 hay hàm VA – KH sẽ nhận giá trị 1 khi ít nhất một biến của nó có giá trị 0. Hàm VA–KH có ký hiệu toán học là một đường gạch thẳng đứng “|”.

Biểu thức toán học của phép phủ định của nhân logic là:

$$y_{14} = \overline{x_1 \cdot x_2} \quad \text{hay} \quad y_{14} = x_1 | x_2$$

Mạch thực hiện hàm y_{14} gọi là mạch VA – KH (mạch NAND).

Mạch VA–KH là một mạch số logic cơ bản. Vì vậy loại mạch này được sản xuất hàng loạt dưới dạng các IC với số lượng các đầu vào khác nhau, thông thường từ 2 đến 5 đầu vào. Mạch có số lượng đầu vào lớn rất hiếm.

Mạch VA–KH có ký hiệu khác nhau tùy theo hãng sản xuất. Ngoài ký hiệu cho trên bảng 2.5 còn dùng các ký hiệu như ở hình 2.3d.

2.3.5. Hàm y_6

Hàm y_6 gọi là hàm cộng môđun 2 hay hàm bất đẳng trị. Bảng giá trị thực của y_6 cho trên bảng 2.6e. Chúng ta có thể định nghĩa: Hàm cộng môđun 2 là hàm chỉ nhận giá trị 1 khi hai biến của nó có giá trị khác nhau.

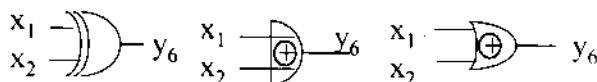
Biểu thức toán học của phép cộng môđun 2 là:

$$y_6 = \bar{x}_1 x_2 + x_1 \bar{x}_2$$

$$\text{hay } y_6 = x_1 \oplus x_2$$

Mạch thực hiện hàm y_6 gọi là mạch cộng môđun 2.

Mạch cộng môđun 2 là một mạch số logic cơ bản, đặc biệt hay dùng trong xử lý tin. Vì vậy loại mạch này được sản xuất hàng loạt dưới dạng các IC.



Ký hiệu riêng Các ký hiệu dùng ký hiệu của mạch cộng logic

Hình 2.4. Các ký hiệu mạch của hàm y_6

Mạch cộng môđun 2 có ký hiệu khác nhau. Ký hiệu cho trên bảng 2.5 là ký hiệu riêng cho mạch cộng môđun 2. Ngoài ra, để chỉ mạch cộng môđun 2 còn dùng các ký hiệu của mạch cộng logic có thêm dấu cộng môđun 2 như hình 2.4.

2.4. CÁC TÍNH CHẤT VÀ ĐỊNH LUẬT CƠ BẢN

Nắm được các tính chất và các định luật cơ bản của các hàm logic giúp cho việc tổng hợp và phân tích các hệ thống logic được thuận lợi, nhanh chóng và đạt được những tính chất tối ưu. Trong mục này chúng ta sẽ xem xét các tính chất hay quan hệ và các định luật cơ bản của các hàm logic cơ bản và quan trọng đã nêu trên. Các quan hệ quan trọng và các định luật này được thể hiện trong từng nhóm các hàm logic có mối quan hệ chặt chẽ. Đó là các nhóm hàm: VA, HOAC, KH; VA – KH, HOAC – KH; Cộng môđun 2.

2.4.1. Tính chất và định luật của các hàm HOAC, VA, KH

1) Các quan hệ logic quan trọng đối với các phép VA và HOAC được cho trong bảng 2.7.

Bảng 2.7. Các tính chất và các định luật quan trọng

STT	Các tính chất quan trọng	Đối với phép VA	Đối với phép HOAC
1	Phép tính với 0	$x.0 = 0$	$x + 0 = x$
2	Phép tính với 1	$x.1 = x$	$x + 1 = 1$
3	Phép tính với $\bar{x}$	$x.\bar{x} = 0$	$x + \bar{x} = 1$
4	Phép tính với x	$x.x = x$	$x + x = x$

STT	Các định luật	Đối với phép VA	Đối với phép HOAC
1	Định luật giao hoán	$x_1 . x_2 = x_2 . x_1$	$x_1 + x_2 = x_2 + x_1$
2	Định luật kết hợp	$x_1 (x_2 . x_3) = (x_1 . x_2) x_3$	$x_1 + (x_2 + x_3) = (x_1 + x_2) + x_3$
3	Định luật phân phối	$x_1 (x_2 + x_3) = x_1 . x_2 + x_1 . x_3$	$x_1 + x_2 . x_3 = (x_1 + x_2) (x_1 + x_3)$
4	Định luật dính	$x_1 x_2 + x_1 \bar{x}_2 = x_1$	$(x_1 + x_2) (x_1 + \bar{x}_2) = x_1$
5	Định luật nuốt	$x_1 + x_1 x_2 = x_1$	$x_1 (x_1 + x_2) = x_1$
6	Định luật giản ước	$x_1 + \bar{x}_1 x_2 = x_1 + x_2$	$x_1 (\bar{x}_1 + x_2) = x_1 \bar{x}_2$
7	Định luật De Morgan	$\overline{x_1 x_2} = \bar{x}_1 + \bar{x}_2$	$\overline{x_1 + x_2} = \bar{x}_1 . \bar{x}_2$
8	Định luật khử biến	$x_i f(x_1, x_2, \dots, x_i, \dots, x_n) = x_i f(x_1, x_2, \dots, 1, \dots, x_n)$	$x_i + f(x_1, x_2, \dots, x_i, \dots, x_n) = x_i + f(x_1, x_2, \dots, 0, \dots, x_n)$
9	Định luật phân rã	$f(x_1, x_2, \dots, x_i, \dots, x_n) = x_i f(x_1, x_2, \dots, 1, \dots, x_n) + \bar{x}_i f(x_1, x_2, \dots, 0, \dots, x_n)$	$f(x_1, x_2, \dots, x_i, \dots, x_n) = (x_i + f(x_1, x_2, \dots, 0, \dots, x_n)) . (\bar{x}_i + f(x_1, x_2, \dots, 1, \dots, x_n))$

Phương pháp chung để chứng minh các định luật trên là lập bảng chân lý của hai vế các đẳng thức trên và chứng tỏ chúng bằng nhau.

Qua bảng 2.7 ta thấy rằng, giữa phép nhân và phép cộng logic có tính đối lẫn: Cột các tính chất của phép nhân có thể nhận được từ cột các tính chất của phép cộng và ngược lại bằng cách thay các phép cộng bằng phép nhân và các phép nhân bằng phép cộng; các biến $\bar{x}_i$ bằng các biến x_i và ngược lại, các biến x_i bằng các biến $\bar{x}_i$.

Cần lưu ý rằng một số tính chất và định luật có thể mở rộng cho n biến như:

$$\begin{aligned} x \cdot x \dots x &= x \quad \text{và} \quad x + x + x + \dots + x = x \\ \overline{x_1 | x_2 | \dots | x_n} &= \bar{x}_1 \downarrow \bar{x}_2 \downarrow \dots \downarrow \bar{x}_n \\ \overline{x_1 \downarrow x_2 \downarrow \dots \downarrow x_n} &= \bar{x}_1 | \bar{x}_2 | \dots | \bar{x}_n \end{aligned}$$

2.4.2. Các tính chất và định luật quan trọng của các hàm VA-KH và HOAC-KH

Các quan hệ và các định luật cơ bản của các hàm VA-KH và HOAC - KH cho ở bảng 2.8.

Bảng 2.8. Các tính chất và định luật của các hàm VA-KH và HOAC-KH

STT	Tính chất & Định luật	Đối với phép VA-KH	Đối với phép HOAC-KH
1	Phép tính với 0	$x 0 = 1$	$x \downarrow 0 = \bar{x}$
2	Phép tính với 1	$x 1 = \bar{x}$	$x \downarrow 1 = 0$
3	Phép tính với $\bar{x}$	$x \bar{x} = 1$	$x \downarrow \bar{x} = 0$
4	Phép tính với x	$x x = \bar{x}$	$x \downarrow x = \bar{x}$
5	Quan hệ với phép nhân và phép cộng logic	$x_1 x_2 = \overline{x_1 x_2} = \bar{x}_1 + \bar{x}_2$	$x_1 \downarrow x_2 = \overline{x_1 + x_2} = \bar{x}_1 \cdot \bar{x}_2$
6	Định luật giao hoán	$x_1 x_2 = x_2 x_1$	$x_1 \downarrow x_2 = x_2 \downarrow x_1$
7	Không có định luật kết hợp	$x_1 x_2 x_3 \neq (x_1 x_2) x_3$	$x_1 \downarrow x_2 \downarrow x_3 \neq (x_1 \downarrow x_2) \downarrow x_3$
8	Định luật De Morgan	$\overline{x_1 x_2 \dots x_n} = \bar{x}_1 \downarrow \bar{x}_2 \downarrow \dots \downarrow \bar{x}_n$	$\overline{x_1 \downarrow x_2 \downarrow \dots \downarrow x_n} = \bar{x}_1 \bar{x}_2 \dots \bar{x}_n$
9	Định luật phân phối	$(x_1 x_2) (x_1 x_3) = \overline{x_1 (\bar{x}_2 \bar{x}_3)}$	$(x_1 \downarrow x_2) \downarrow (x_1 \downarrow x_3) = \overline{x_1 \downarrow (\bar{x}_2 \downarrow \bar{x}_3)}$
10	Định luật bỏ dấu ngoặc	$(x_1 \downarrow x_2) (x_1 \downarrow x_3) = \overline{x_1 \downarrow (\bar{x}_2 \bar{x}_3)}$ $= \bar{x}_1 (x_2 \downarrow x_3)$ $(x_1 \downarrow x_2) (x_3 \downarrow x_4) = \overline{x_1 \bar{x}_2 \bar{x}_3 \bar{x}_4}$	$(x_1 x_2) \downarrow (x_1 x_3) = \overline{x_1 (x_2 \downarrow x_3)} =$ $= \bar{x}_1 \downarrow (x_2 x_3)$ $(x_1 x_2) \downarrow (x_3 x_4) = \overline{x_1 \downarrow \bar{x}_2 \downarrow \bar{x}_3 \downarrow \bar{x}_4}$

Những quan hệ và các định luật trên đây thường được sử dụng để đơn giản hàm logic trong hệ hàm đủ VA – KH và HOAC – KH. Tuy nhiên biến đổi hàm trong hệ hàm đủ VA–KH hoặc HOAC – KH rất không thuận tiện. Trong thực tiễn cho phép thực hiện các biến đổi hàm logic trong hệ hàm đủ VA, HOAC, KH. Từ kết quả của các biến đổi đó ta chuyển sang dạng VA–KH, HOAC–KH hoặc một dạng hỗn hợp.

2.4.3. Các quan hệ và các định luật của hàm cộng môđun 2

Các quan hệ và các định luật này cho trên bảng 2.9.

Bảng 2.9. Các quan hệ và các định luật của hàm cộng môđun 2

STT	Quan hệ & Định luật	Nội dung
1	Phép tính với 0 Phép tính với 1 Phép tính với $\bar{x}$ Phép tính với x	$x \oplus 0 = x$ $x \oplus 1 = \bar{x}$ $x \oplus \bar{x} = 1$ $x \oplus x = 0$
2	Định luật giao hoán	$x_1 \oplus x_2 = x_2 \oplus x_1$
3	Định luật kết hợp	$x_1 \oplus (x_2 \oplus x_3) = (x_1 \oplus x_2) \oplus x_3$
4	Định luật phân phối	$x_1 (x_2 \oplus x_3) = x_1 x_2 \oplus x_1 x_3$
5	Quan hệ đặc biệt	$x_1 \oplus x_2 \oplus x_1 x_2 = x_1 + x_2$

2.5. PHƯƠNG PHÁP MÔ TẢ HÀM LÔGIC

Như chúng ta đã biết, một hàm logic n biến có $N = 2^n$ tập giá trị biến và ứng với mỗi tập giá trị biến hàm sẽ nhận một trong hai giá trị là 0 hoặc 1. Như vậy, một hàm logic sẽ được hoàn toàn xác định nếu đối với mỗi tập giá trị biến của nó ta biết rõ hàm đó nhận giá trị nào.

Mô tả một hàm logic là làm thế nào để ai cũng có thể nhận biết được hàm đã cho một cách chính xác và như nhau. Các phương pháp ghi nhận một hàm logic để phục vụ cho mục đích này gọi là các phương pháp mô tả hàm logic. Như vậy, để mô tả một hàm logic ta cần có cách mô tả các tập giá trị biến của nó và gán với mỗi tập giá trị biến giá trị tương ứng của hàm. Vì đối với hàm logic xác định đầy đủ, các giá trị của hàm chỉ có thể là 0 hay 1, nên đối với các hàm này việc xác định được đối với tập giá trị biến nào hàm nhận giá trị 1 tương đương với việc hàm đã được xác định. Đối với các hàm logic xác định không đầy đủ, giá trị của hàm có thể là một trong ba giá trị: 0, 1, – (bất định). Vì vậy, muốn một hàm không xác định đầy đủ được mô tả đúng thì phải chỉ rõ hai

trong ba nhóm tập giá trị biến: Hoặc nhóm mà hàm nhận giá trị 1 và nhóm mà hàm nhận giá trị bất định hoặc nhóm mà hàm nhận giá trị 0 và nhóm mà hàm nhận giá trị bất định.

Ba phương pháp thường dùng mô tả các hàm logic là:

- Phương pháp bảng giá trị thực;
- Phương pháp hình học;
- Phương pháp dùng biểu thức toán học.

2.5.1. Phương pháp bảng giá trị thực

Để mô tả hàm logic dùng phương pháp bảng giá trị thực, ta liệt kê tất cả $N = 2^n$ tập giá trị biến của hàm thành một bảng và bên cạnh mỗi tập giá trị biến ta ghi giá trị tương ứng của hàm.

Đây là phương pháp rất phổ biến, thuận tiện cho việc mô tả chức năng của các mạch logic tổ hợp như các bộ biến đổi mã, các bộ phân kênh, các bộ chọn và các bộ cộng không nhớ... Nó cũng là một phương pháp quan trọng đầu tiên, là tiền đề của các phương pháp khác.

Thí dụ 2.3. Một trong các hàm ba biến có thể được mô tả bằng bảng giá trị thực như sau:

x_1	x_2	x_3	y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Cùng hàm trên đây, người ta có thể viết dưới dạng gọn hơn như sau:

Cách thứ nhất:

$$y(x_1, x_2, x_3) = f(001, 010, 101, 111)$$

Cách thứ hai:

$y(x_1, x_2, x_3) = f(1, 2, 5, 7)$ với $n = 3$ và $b = 10$. Tức là 3 biến, các tập giá trị biến viết ở hệ cơ số 10.

Cách thứ ba:

Viết dưới dạng tương minh của các biến:

$$y(x_1, x_2, x_3) = f(\bar{x}_1 \bar{x}_2 x_3, \bar{x}_1 x_2 \bar{x}_3, x_1 \bar{x}_2 x_3, x_1 x_2 x_3)$$

Cần chú ý rằng, đối với một hàm logic, việc sắp xếp thứ tự các biến là tùy ý, nhưng một khi đã xác định thứ tự thì không được thay đổi.

Đối với các hàm logic xác định không đầy đủ, giá trị không xác định của hàm được biểu diễn bằng một dấu gạch ngang và hàm được cho bởi các tập giá trị biến mà hàm nhận giá trị 1 và giá trị bất định, hoặc giá trị 0 và giá trị bất định.

Thí dụ 2.4:

x_1	x_2	x_3	y
0	0	0	0
0	0	1	1
0	1	0	–
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	–
1	1	1	1

Dưới dạng ngắn gọn, hàm không xác định đầy đủ trên đây được mô tả như sau:

Cách thứ nhất:

$$y(x_1, x_2, x_3) = \begin{cases} f^1(001, 101, 111) \\ f^-(010, 110) \end{cases}$$

Cách thứ hai:

$$y(x_1, x_2, x_3) = \begin{cases} f^1(1, 5, 7) \text{ với } n=3 \text{ và } b=10 \\ f^-(2, 6) \end{cases}$$

Tức là 3 biến, các tập giá trị biến viết ở hệ cơ số 10.

Cách thứ ba:

$$y(x_1, x_2, x_3) = \begin{cases} f^1(\bar{x}_1 \bar{x}_2 x_3, x_1 \bar{x}_2 x_3, x_1 x_2 x_3) \\ f^-(\bar{x}_1 x_2 \bar{x}_3, x_1 x_2 \bar{x}_3) \end{cases}$$

2.5.2. Phương pháp hình học

Để mô tả một hàm logic có thể dùng giản đồ thời gian, không gian hai hoặc ba chiều. Không gian ba chiều (hay nói chung là nhiều chiều) thực tế ít được dùng. Trong tài liệu này chúng ta chỉ xét hai trường hợp đầu.

Phương pháp hình học có ưu điểm là trực quan, song phạm vi mô tả có phần bị hạn chế.

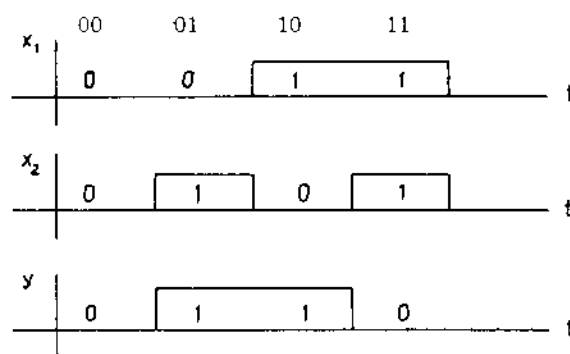
a. Mô tả dùng giản đồ thời gian

Giản đồ thời gian là một công cụ tiện lợi trong mô tả các đại lượng logic thay đổi theo thời gian như các tham số, các biến và các hàm trong các thiết bị điều khiển.

Để mô tả một hàm có n biến, ta cần $(n + 1)$ giản đồ thời gian để mô tả giá trị của các biến và của hàm. Trong đó cần tạo ra tất cả $N = 2^n$ tập giá trị biến với các giá trị tương ứng của hàm đó.

Thí dụ, giản đồ thời gian của hàm bất đẳng trị hai biến x_1, x_2 cho trên hình 2.5. Trong đó kết hợp hai giản đồ thời gian của x_1 và x_2 ta có 4 tập giá trị biến của hai biến là: 00, 01, 10 và 11.

Trong các khoảng thời gian tiếp theo, ứng với mọi sự lặp lại của một trong các tập giá trị biến, giá trị của hàm sẽ là giá trị với tập giá trị biến tương ứng đã chỉ ra trong hình 2.5.



Hình 2.5. Giản đồ thời gian hàm bất đẳng trị

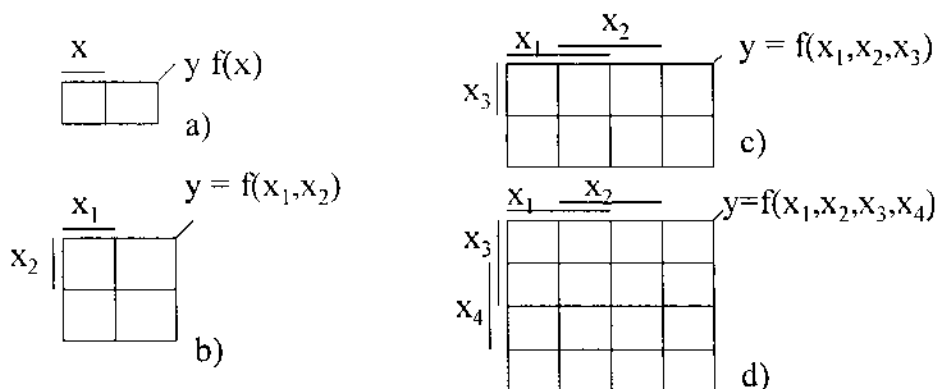
b. Mô tả hàm logic dùng không gian hai chiều

Phương pháp dùng không gian hai chiều còn gọi là phương pháp mô tả hàm logic dùng mặt phẳng hay dùng bảng kề cận.

Nguyên tắc xây dựng bảng kề cận như sau: Để mô tả hàm logic có n biến, cần kẻ một bảng có $N = 2^n$ ô trên mặt phẳng và bố trí giá trị các biến sao cho mỗi ô ứng với một tập giá trị biến. Một trong các cách bố trí các biến tuân theo nguyên tắc sau đây: Giả sử hàm cần mô tả có n biến thì bố trí sao cho:

- + Trong một nửa số ô x_1 có giá trị 1, trong nửa còn lại x_1 có giá trị 0.
- + Đối với x_2 cũng sẽ bố trí sao cho trong một nửa số ô x_2 có giá trị 1, còn trong nửa còn lại x_2 có giá trị 0. Nhưng sao cho các ô có giá trị 1 và giá trị 0 của hai biến x_1 và x_2 phải giao nhau 50%.
- + Đối với các biến còn lại cũng bố trí tương tự.

Thí dụ, các bảng để mô tả các hàm logic 1, 2, 3 và 4 biến cho trên hình 2.6.



Hình 2.6. a) Bảng cho hàm 1 biến; b) Bảng cho hàm 2 biến; c) Bảng cho hàm 3 biến; d) Bảng cho hàm 4 biến

Sau khi đã có bảng tương ứng, để mô tả hàm ta chỉ việc ghi giá trị của hàm ứng với từng tập giá trị biến vào các ô tương ứng của bảng.

Thí dụ, hàm ba biến cho trên hình 2.7a có thể mô tả như ở hình 2.7b.

x_1	x_2	x_3	y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

a)

	x_1 x_2		y
x_3	0	1	0
	1	0	1

b)

Hình 2.7. Bảng giá trị thực của hàm ba biến và biểu diễn lên bảng kế cận của nó

Đặc điểm của bảng kế cận xây dựng theo nguyên tắc này là:

- + Hai ô nằm kế nhau, kể cả khi cuốn tròn theo chiều đứng và chiều ngang, luôn luôn đại diện cho các tập giá trị biến chỉ khác nhau ở một bit. Các tập giá trị biến mà chỉ khác nhau ở một bit gọi là các tập giá trị biến kế cận. Bảng xây dựng có tính chất như vậy gọi là bảng kế cận.
- + Bảng kế cận rất thuận tiện cho việc mô tả các hàm logic với số biến $n \leq 4$. Khi đó chỉ có các ô nằm sát nhau mới là kế cận nhau. Các ô ở cách xa nhau không là kế cận. Nhưng khi số biến $n > 4$ thì tính chất này không còn nữa. Các ô kế cận khi đó không còn chỉ là các ô nằm sát nhau mà nhiều ô không nằm sát nhau cũng vẫn là các ô kế cận.
- + Tính kế cận được sử dụng để đơn giản hàm trong phương pháp Karnaugh.

Chúng ta sẽ trở lại vấn đề này khi nghiên cứu phương pháp đơn giản hàm bằng bảng kế cận hay còn gọi là bảng Karnaugh.

2.5.3. Mô tả hàm logic dùng biểu thức toán học

Phương pháp này rất quan trọng vì trên cơ sở mô tả hàm logic bằng biểu thức toán học, chúng ta có thể biến đổi, đơn giản biểu thức hay rút ra những quy luật biến đổi của hàm một cách dễ dàng.

Mô tả chức năng của một thiết bị dùng biểu thức toán học có thể thực hiện trực tiếp từ nhiệm vụ bài toán nhưng cũng thường xuất phát từ dạng mô tả dùng bảng giá trị thực. Trong trường hợp này ta có một số khái niệm sau:

a. Hàm đơn 1

Hàm đơn 1 là hàm logic n biến nhưng chỉ nhận giá trị 1 ở một tập giá trị biến duy nhất. Hàm đơn 1 ký hiệu là m_i , trong đó i là tập giá trị biến mà hàm nhận giá trị 1.

Thí dụ, với $n = 3$ biến, hàm đơn 1:

m_2 là hàm nhận giá trị 1 ở tập giá trị biến 010. Tức là $m_2 = f(010)$.

m_3 là hàm nhận giá trị 1 chỉ đối với tập giá trị biến 011, hay $m_3 = f(011)$;

m_6 là hàm nhận giá trị 1 chỉ đối với tập giá trị biến 110, hay $m_6 = f(110)$;

m_7 là hàm nhận giá trị 1 chỉ đối với tập giá trị biến 111, hay $m_7 = f(111)$

b. Hàm đơn 0

Hàm đơn 0 là hàm lôgic n biến nhưng chỉ nhận giá trị 0 ở một tập giá trị biến duy nhất. Hàm đơn 0 ký hiệu là M_i , trong đó i là tập giá trị biến mà hàm nhận giá trị 0.

Ví dụ: Với hàm $n = 3$ biến, hàm đơn 0 :

M_0 là hàm nhận giá trị 0 chỉ đối với tập giá trị biến 000, hay $M_0 = f^0(000)$

M_1 là hàm nhận giá trị 0 chỉ đối với tập giá trị biến 001, hay $M_1 = f^0(001)$

M_4 là hàm nhận giá trị 0 chỉ đối với tập giá trị biến 100, hay $M_4 = f^0(100)$

M_5 là hàm nhận giá trị 0 chỉ đối với tập giá trị biến 101, hay $M_5 = f^0(101)$.

c. Tính chất của các hàm m_i và M_i

Căn cứ vào định nghĩa của m_i và M_i ta thấy:

$$+ M_i + M_j = \begin{cases} M_i & \text{nếu } i = j \\ 1 & \text{nếu } i \neq j \end{cases}$$

$$+ m_i m_j = \begin{cases} m_i & \text{nếu } i = j \\ 0 & \text{nếu } i \neq j \end{cases}$$

+ Tổng lôgic của tất cả các m_i bằng 1; tích lôgic của tất cả các M_i bằng 0:

$$\sum_{i=1}^{2^n-1} m_i = 1 \qquad \prod_{i=1}^{2^n-1} M_i = 0$$

+ Mọi m_i (M_i) có M_i (m_i) là phủ định của nó, tức là:

$$m_i = \overline{M_i}, \qquad M_i = \overline{m_i}$$

d. Dạng giải tích của m_i và M_i

Trong hệ hàm VA, HOAC, KH biểu thức giải tích của m_i và M_i được rút ra từ bảng giá trị thực của các hàm này.

+ Dạng giải tích của m_i :

– Viết tích tất cả các biến của hàm m_i .

– Căn cứ vào tập giá trị biến i , biến nào có giá trị 0 thì viết thêm dấu phủ định.

Thí dụ 2.5:

x_1	x_2	x_3	m_1	m_6
0	0	0	0	0
0	0	1	1	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	0
1	1	0	0	1
1	1	1	0	0

$$0 \rightarrow m_1 = \bar{x}_1 \bar{x}_2 x_3 = x_1 \downarrow x_2 \downarrow \bar{x}_3 = \overline{\bar{x}_1 | \bar{x}_2 | x_3}$$

$$1 \rightarrow m_6 = x_1 x_2 \bar{x}_3 = \bar{x}_1 \downarrow \bar{x}_2 \downarrow x_3 = \overline{x_1 | x_2 | \bar{x}_3}$$

+ Dạng giải tích của hàm M_i :

- Viết tổng của tất cả các biến của M_i ;

- Căn cứ vào tập i , biến nào có giá trị 1 thì lấy phủ định.

Thí dụ 2.6:

x_1	x_2	x_3	M_2	M_5
0	0	0	1	1
0	0	1	1	1
0	1	0	0	1
0	1	1	1	1
1	0	0	1	1
1	0	1	1	0
1	1	0	1	1
1	1	1	1	1

$$0 \rightarrow M_2 = x_1 + \bar{x}_2 + x_3 = \bar{x}_1 | x_2 | \bar{x}_3 = \overline{x_1 \downarrow x_2 \downarrow \bar{x}_3}$$

$$1 \rightarrow M_5 = \bar{x}_1 + x_2 + \bar{x}_3 = x_1 | \bar{x}_2 | x_3 = \overline{\bar{x}_1 \downarrow x_2 \downarrow \bar{x}_3}$$

Trong hệ hàm VA-KH và HOAC-KH, các m_i và M_i được rút ra như sau:

+ Viết m_i trong hệ HOAC-KH và VA-KH:

Viết “ $\downarrow$ ” của tất cả các biến. Trong tập giá trị biến i , biến nào có giá trị 1 thì lấy phủ định. Biểu thức dưới dạng VA-KH có thể nhận được bằng cách áp dụng định luật De Morgan - Xem thí dụ 2.5.

+ Viết M_i trong hệ VA-KH và HOAC-KH :

Viết “ $|$ ” của tất cả các biến. Trong tập giá trị biến i , biến nào có giá trị 0 thì lấy phủ định. Biểu thức dưới dạng HOAC-KH có thể nhận được bằng cách áp dụng định luật De Morgan - Xem thí dụ 2.6.

e) Phân tích một hàm logic n biến thành các hàm m_i hoặc các hàm M_i

Bất kỳ một hàm logic nào đều có thể phân tích thành các hàm đơn 1 hoặc thành các hàm đơn 0 của nó.

Thí dụ 2.7:

Phân tích hàm logic xác định đầy đủ $y = f(2, 3, 6, 7)$ thành các hàm đơn 1 và các hàm đơn 0 của nó. Hàm này nhận giá trị 1 ở các tập giá trị biến: 2, 3, 6 và 7 và nhận giá trị 0 ở các tập giá trị biến: 0, 1, 4 và 5. Vậy chúng ta có thể phân tích hàm này hoặc theo các m_i hoặc theo các M_i của nó như bảng 2.10 sau đây.

Bảng 2.10. Phân tích hàm thành các hàm đơn 1 và đơn 0

x_1	x_2	x_3	y	m_2	m_3	m_6	m_7	M_0	M_1	M_4	M_5
0	0	0	0	0	0	0	0	0	1	1	1
0	0	1	0	0	0	0	0	1	0	1	1
0	1	0	1	1	0	0	0	1	1	1	1
0	1	1	1	0	1	0	0	1	1	1	1
1	0	0	0	0	0	0	0	1	1	0	1
1	0	1	0	0	0	0	0	1	1	1	0
1	1	0	1	0	0	1	0	1	1	1	1
1	1	1	1	0	0	0	1	1	1	1	1

Định lý 1: Bất kỳ hàm logic n biến nào đều có thể phân tích thành tổng logic của các hàm đơn 1 của nó.

$$y(x_1, x_2, \dots, x_n) = \sum_i m_i(x_1, x_2, \dots, x_n) \quad (2.1)$$

Chứng minh: Lấy bất kỳ tổ hợp biến nào trong $N = 2^n$ tổ hợp biến của hàm:

a. Nếu $y(x_1, x_2, \dots, x_n) = 1$ thì ở vế phải của (2.1) sẽ có một m_i tương ứng bằng 1 và do đó vế phải của (2.1) bằng 1.

b. Nếu $y(x_1, x_2, \dots, x_n) = 0$ thì ở vế phải của (2.1) không có m_i nào bằng 1 cả. Nói cách khác vế phải của (2.1) bằng 0.

Vậy hai vế của (2.1) luôn luôn bằng nhau. Định lý được chứng minh.

Dạng (2.1) gọi là dạng tuyến chuẩn chính tắc của hàm. Dạng tuyến chuẩn chính tắc của một hàm n biến có thể viết như sau:

$$y(x_1, x_2, \dots, x_n) = \sum_i x^{a_1} x^{a_2} \dots x^{a_n} \quad (2.2)$$

Trong đó, $x^{a_1} x^{a_2} \dots x^{a_n}$ là tích của tất cả các biến ứng với tập giá trị biến mà $y(x_1, x_2, \dots, x_n) = 1$.

Tổ hợp $\alpha_1 \alpha_2 \dots \alpha_n$ tương ứng với tổ hợp giá trị biến i . Trong đó:

$$x^{\alpha_i} = \begin{cases} x_i, & \text{nếu } \alpha_i = 1 \\ \bar{x}_i, & \text{nếu } \alpha_i = 0 \end{cases}$$

Thủ tục để viết dạng tuyến chuẩn chính tắc của một hàm:

- Xác định tất cả các tập giá trị biến mà hàm có giá trị 1;
- Viết tất cả các m_i của hàm;
- Nối các m_i bằng các dấu cộng logic.

Thí dụ 2.8

Viết dạng tuyến chuẩn chính tắc của hàm sau: $y(x_1, x_2, x_3) = f(1, 4, 6)$.

Bảng giá trị thực của hàm đã cho như sau:

x_1	x_2	x_3	y	Viết các m_i của hàm:
0	0	0	0	
0	0	1	1	$+ m_1 = \bar{x}_1 \bar{x}_2 x_3$
0	1	0	0	
0	1	1	0	$+ m_4 = x_1 \bar{x}_2 \bar{x}_3$
1	0	0	1	
1	0	1	0	$+ m_6 = x_1 x_2 \bar{x}_3$
1	1	0	1	Hàm y :
1	1	1	0	$y = \bar{x}_1 \bar{x}_2 x_3 + x_1 \bar{x}_2 \bar{x}_3 + x_1 x_2 \bar{x}_3$

Định lý 2

Mọi hàm logic đều có thể biểu diễn dưới dạng tích logic của tất cả các hàm đơn 0 của nó.

$$y(x_1, x_2, \dots, x_n) = \prod_i M_i(x_1, x_2, \dots, x_n) \quad (2.3)$$

trong đó i là tập hợp giá trị biến mà hàm nhận giá trị 0.

Chứng minh: Lấy bất kỳ tổ hợp biến nào trong $N = 2^n$ tổ hợp biến của hàm:

a) Nếu $y(x_1, x_2, \dots, x_n) = 0$ thì ở vế phải của (2.3) sẽ có một M_i tương ứng bằng 0 và do đó vế phải của (2.3) bằng 0.

b) Nếu $y(x_1, x_2, \dots, x_n) = 1$ thì ở vế phải của (2.3) không có M_i nào bằng 0 cả. Nói cách khác, vế phải của (2.3) bằng 1.

Vậy hai vế của (2.3) luôn luôn bằng nhau. Định lý được chứng minh.

Dạng (2.3) gọi là dạng hội chuẩn chính tắc của hàm. Dạng hội chuẩn chính tắc của một hàm n biến có thể viết như sau:

$$y(x_1, x_2, \dots, x_n) = \prod_i (x^{u_1} + x^{u_2} + \dots + x^{u_n}) \quad (2.4)$$

Trong đó, $x_1^{u_1} + x_2^{u_2} + \dots + x_n^{u_n}$ là tổng logic của tất cả các biến ứng với tập giá trị biến mà $y(x_1, x_2, \dots, x_n) = 0$.

Tổ hợp $\alpha_1 \alpha_2 \dots \alpha_n$ tương ứng với tổ hợp giá trị biến i . Trong đó:

$$x^{u_i} = \begin{cases} \bar{x}_i, & \text{nếu } \alpha_i = 1 \\ x_i, & \text{nếu } \alpha_i = 0 \end{cases}$$

Thủ tục để viết dạng hội chuẩn chính tắc của một hàm:

- Xác định tất cả các tập giá trị biến mà hàm có giá trị 0;
- Viết tất cả các M_i của hàm;
- Nối các M_i bằng các dấu nhân logic.

Thí dụ 2.9:

Viết dạng hội chuẩn chính tắc của hàm sau: $y(x_1, x_2, x_3) = f(1, 2, 4, 6, 7)$.

Bảng giá trị thực của hàm đã cho như sau:

x_1	x_2	x_3	y	Viết các M_i của hàm:
0	0	0	0	$+ M_0 = x_1 + x_2 + x_3$
0	0	1	1	$+ M_3 = x_1 + \bar{x}_2 + \bar{x}_3$
0	1	0	1	$+ M_5 = \bar{x}_1 + x_2 + \bar{x}_3$
0	1	1	0	Hàm y :
1	0	0	1	$y = (x_1 + x_2 + x_3)(x_1 + \bar{x}_2 + \bar{x}_3)(\bar{x}_1 + x_2 + \bar{x}_3)$
1	0	1	0	
1	1	0	1	
1	1	1	1	

Ở dạng chính tắc, trong mỗi m_i hay M_i đều có mặt tất cả các biến của hàm dưới dạng có phủ định hay không phủ định. Bằng các biến đổi trên cơ sở các định luật và các quan hệ logic ta có thể đưa về dạng đơn giản hơn, trong đó, mỗi thành phần của hàm (m_i hay M_i) không nhất thiết phải có đủ mặt tất cả các biến của hàm. Dạng đó gọi chung là dạng tuyển chuẩn (DTC) hay dạng hội chuẩn (DHC).

Bên cạnh các dạng tuyển chuẩn và dạng hội chuẩn, trong thực tế chúng ta còn gặp dạng tuyển chuẩn hay dạng hội chuẩn có nhiều cấp lồng vào nhau, gọi chung là dạng tuyển chuẩn hay hội chuẩn nhiều cấp (DTCNC / DHCNC).

Thí dụ 2.10: Viết dạng tuyển chuẩn và dạng hội chuẩn của hàm logic:

+ DTC: $y(x_1, x_2, x_3, x_4) = x_1 \bar{x}_2 + x_3 x_4 + \bar{x}_1 \bar{x}_3 x_4$

+ DHC: $y(x_1, x_2, x_3, x_4) = (\bar{x}_1 + x_2)(\bar{x}_3 + \bar{x}_4)(x_1 + x_3 + \bar{x}_4)$

$$+ \text{DTCNC} : y(x_1, x_2, x_3, x_4) = x_1 \bar{x}_2 + x_4(x_3 + \bar{x}_1 \bar{x}_3)$$

$$+ \text{DHCNC} : y(x_1, x_2, x_3, x_4) = (x_2 \bar{x}_3 + x_2 \bar{x}_4) ((\bar{x}_2 + x_1)x_3 + x_1 \bar{x}_4)$$

2.6. HỆ HÀM ĐỦ

Hệ các hàm logic cơ bản $\{f_1, f_2, \dots, f_m\}$ gọi là hệ hàm đủ, nếu mọi hàm logic φ , dù phức tạp đến đâu cũng có thể biểu diễn qua các hàm logic này.

Hệ hàm đủ hiển nhiên là hệ gồm tất cả các hàm logic 0, 1 và 2 biến. Nhưng một hệ hàm đủ như vậy không mang lại ý nghĩa thực tiễn quan trọng nào. Để đơn giản hệ thống và các quy trình sản xuất các vi mạch mà vẫn có đủ các loại phần tử logic cần thiết, chúng ta cần xác định các hệ hàm đủ gồm một số ít chấp nhận được các hàm logic cơ bản và đặc biệt cần xác định các hệ hàm đủ nhưng là hệ hàm tối thiểu.

Hệ hàm đủ và tối thiểu là hệ hàm mà nếu bỏ đi bất kỳ hàm nào thì cũng trở nên không đủ nữa.

Hệ hàm đủ quan trọng nhất gồm ba hàm : HOAC, VA, KH. Mọi hàm logic khác đều có thể biểu diễn qua ba hàm này. Thật vậy, theo định lý 1 và định lý 2, trong đó mọi hàm logic đều có thể biểu diễn qua các công thức (2.1) và (2.3) là các công thức chỉ gồm toàn các phép nhân logic, cộng logic và phủ định. Tuy vậy, hệ ba hàm này chưa phải là tối thiểu, vì phép VA có thể biểu diễn qua phép HOAC và KH, phép HOAC lại có thể biểu diễn qua phép VA và KH như sau:

$$x_1 x_2 = \overline{\bar{x}_1 + \bar{x}_2} ; \quad x_1 + x_2 = \overline{\bar{x}_1 \bar{x}_2}$$

Từ đây ta có hệ hàm đủ và tối thiểu là các hệ hàm chỉ gồm hai hàm VA và KH và hệ hàm gồm HOAC và KH.

Hai hệ hàm tối thiểu thông dụng khác chỉ gồm mỗi hệ một hàm. Đó là các hệ hàm $\{VA-KH\}$ và $\{HOAC-KH\}$.

Để chứng tỏ các hàm VA-KH và HOAC-KH là các hệ hàm đủ và tất nhiên là tối thiểu vì mỗi hệ chỉ có một hàm, ta có thể biểu diễn các hàm VA, HOAC và KH qua các hàm VA-KH và HOAC-KH như sau:

$$\begin{aligned} x_1 x_2 &= \overline{x_1 \mid x_2} = \overline{x_1} \downarrow \overline{x_2} \\ x_1 + x_2 &= \overline{x_1 \downarrow x_2} = \overline{x_1} \mid \overline{x_2} \\ \bar{x} &= x \downarrow x = x \mid x \end{aligned}$$

Ngoài ra có thể viết : $x_1 x_2 = (x_1 \mid x_2) \mid (x_1 \mid x_2)$

$$x_1 + x_2 = (x_1 \downarrow x_2) \downarrow (x_1 \downarrow x_2)$$

Một số hệ hàm đủ khác ít thông dụng như: $\{VA, HOAC, 1\}$, $\{HOAC, KH, \text{ĐẰNG TRI}\}$, $\{\text{Hàm kéo theo}, KH\}$.

2.7. PHÂN TÍCH VÀ TỔNG HỢP HÀM LÔGÍC

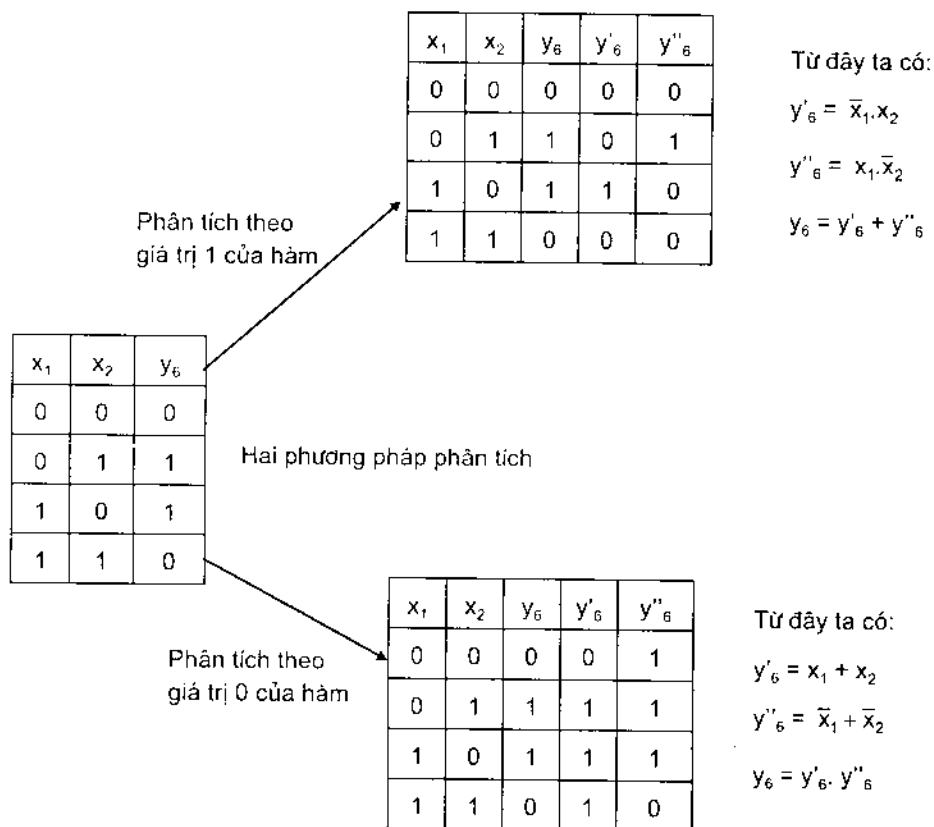
2.7.1. Biểu diễn các hàm lôgic qua hệ hàm đủ

Biểu diễn một hàm lôgic qua các hàm lôgic cơ bản là bài toán phân tích hàm lôgic. Bất kỳ hàm lôgic nào cũng có thể biểu diễn qua các hệ hàm đủ. Biểu diễn một hàm lôgic qua một hệ hàm đủ nào đó có thể thực hiện bằng cách phân tích bảng giá trị thực của hàm đã cho thành các hàm lôgic cơ bản hoặc dùng phương pháp giải tích.

Thí dụ 2.11: Giả thiết cho hàm lôgic y_6 có bảng giá trị thực như sau:

x_1	x_2	y_6
0	0	0
0	1	1
1	0	1
1	1	0

Để biểu diễn hàm này qua hệ hàm đủ {VA, HOAC, KH} ta phân tích hàm này theo sơ đồ ở hình 2.8.



Hình 2.8. Biểu diễn hàm số qua hệ hàm đủ

Như vậy hàm y_6 có thể phân tích thành hai dạng sau:

- 1) $x_1 \oplus x_2 = x_1 \bar{x}_2 + \bar{x}_1 x_2$
- 2) $x_1 \oplus x_2 = (x_1 + x_2) \cdot (\bar{x}_1 + \bar{x}_2)$

Thí dụ 2.12:

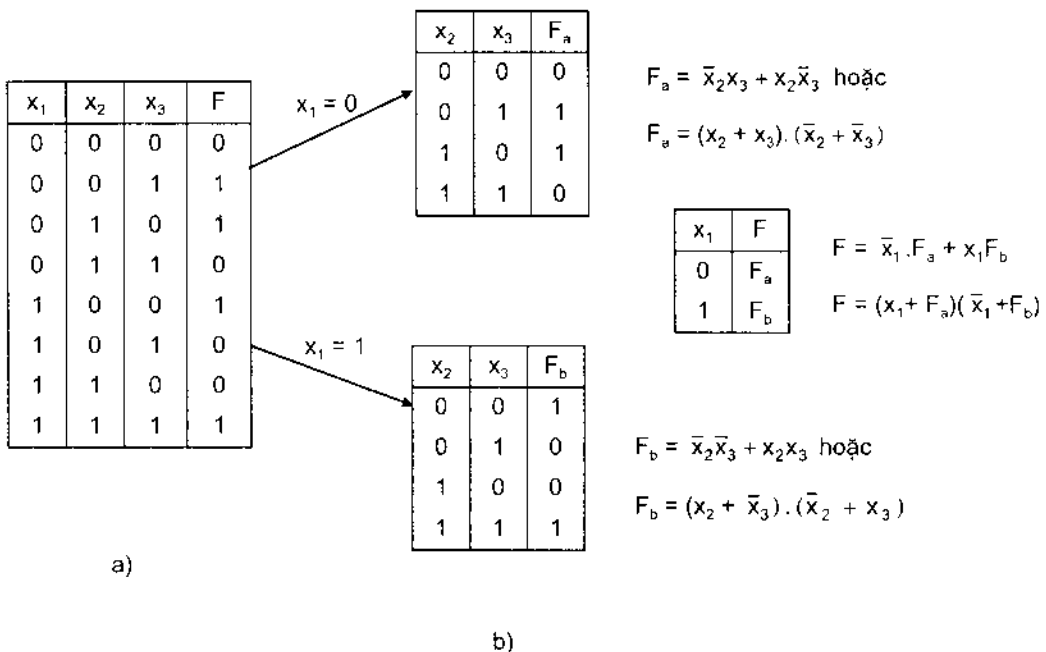
Hãy phân tích hàm ba biến cho trên hình 2.9a thành các hàm logic đơn giản hơn.

Từ bảng giá trị thực của hàm đã cho ta có thể phân tích như hình 2.9b. Kết quả phân tích cho phép biểu diễn hàm đã cho thành hai dạng sau:

- 1) $F = \bar{x}_1 \cdot F_a + x_1 \cdot F_b$
- 2) $F = (x_1 + F_a) \cdot (\bar{x}_1 + F_b)$

trong đó: $F_a = \bar{x}_2 x_3 + x_2 \bar{x}_3$ hoặc $F_a = (x_2 + x_3) \cdot (\bar{x}_2 + \bar{x}_3)$

$F_b = (x_2 + \bar{x}_3) \cdot (\bar{x}_2 + x_3)$ hoặc $F_b = \bar{x}_2 \bar{x}_3 + x_2 x_3$



Hình 2.9. Hàm và phân tích hàm

Qua hai thí dụ trên đây ta thấy rằng, để biểu diễn một hàm cho trước qua một hệ hàm ta có thể phân tích hàm đã cho theo nhiều cách khác nhau. Từ đó kết quả cụ thể sẽ có dạng khác nhau, nhưng chức năng giống nhau.

Phương pháp phân tích tiến hành trên hình 2.9 chính là phương pháp phân giải một hàm có nhiều biến qua những hàm ít biến hơn. Dạng giải tích tổng quát cho dạng phân giải này là:

- I. $f(x_1, x_2, \dots, x_n) = \bar{x}_1 \cdot f(x_1, \dots, 0, \dots, x_n) + x_1 \cdot f(x_1, \dots, 1, \dots, x_n)$
- II. $f(x_1, x_2, \dots, x_n) = (\bar{x}_1 + f(x_1, \dots, 1, \dots, x_n))(x_1 + f(x_1, \dots, 0, \dots, x_n))$

Như vậy, bất kỳ hàm logic nào cũng có thể biểu diễn qua hệ hàm đủ {VA, HOAC, KH}. Mặt khác, vì các hàm VA, HOAC, KH có thể biểu diễn qua hàm VA-KH hoặc HOAC-KH nên một hàm bất kỳ đều có thể biểu diễn qua hệ hàm đủ tối thiểu {VA-KH} và {HOAC-KH}.

Cũng từ những kết quả của hai thí dụ trên đây cho thấy có thể biểu diễn một hàm bất kỳ qua một số hàm logic cơ bản nhất định.

2.7.2. Tổng hợp hàm logic

Từ bài toán phân tích ta thấy một hàm logic được xác định bởi số lượng các biến và thứ tự sắp xếp các biến đó. Đối với một tập hợp thứ tự các biến thì quy luật xuất hiện các giá trị cụ thể của các biến này cũng ảnh hưởng tới quy luật thay đổi các giá trị của hàm. Nói cách khác, để tạo nên một hàm logic mới có thể thực hiện bằng hai cách:

- Đánh số lại các biến logic;
- Thay các biến logic bằng các hàm của chúng.

Hai thuật toán trên đây có thể tiến hành riêng rẽ hay kết hợp đồng thời; có thể thực hiện một lần hay nhiều lần. Phương pháp tổng hợp như vậy gọi là phương pháp xếp chồng.

Thí dụ, hàm logic f_0 dưới đây có thể nhận được bằng phương pháp xếp chồng các hàm logic cơ bản f_1 , f_6 và f_7 trong bảng 2.5.

$$f_0 = f_1(f_6(x_1, x_2), f_7(x_1, x_3)) = (x_1 \oplus x_2)(x_1 + x_3)$$

Nếu x_1, x_2, x_3 lại là kết quả của hàm f_a, f_b và f_s nào đó của các biến y_1, y_2, y_3 thì có thể viết:

$$f_0 = f_1(f_6(f_a(y_1, y_2), f_b(x_1, x_2), f_7(f_b(y_1, y_2), f_s(y_3)))$$

Giả sử $f_a(y_1, y_2) = y_1 \cdot y_2$; $f_b(x_1, x_2) = \bar{y}_1 \cdot \bar{y}_2$; $f_s(y_3) = \bar{y}_3$ thì:

$$f_0 = (y_1 y_2 \oplus \bar{y}_1 \cdot \bar{y}_2)(y_1 y_2 + \bar{y}_3).$$

Kết quả ta có một hàm mới mà biến của nó là y_1, y_2 và y_3 . Bảng giá trị thực của hàm này dễ dàng nhận được từ bảng giá trị thực của các hàm logic cơ bản thành phần.

2.8. MỘT SỐ LỚP HÀM LOGIC THƯỜNG GẶP

Trong các lớp hàm logic, năm lớp sau đây có những tính chất đặc biệt mà ta sẽ nghiên cứu kỹ hơn:

- + Hàm logic tuyến tính
- + Hàm logic bảo tồn giá trị 0
- + Hàm logic bảo tồn giá trị 1

+ Hàm logic đơn điệu

+ Hàm logic tự ngẫu.

Những tính chất đặc biệt đó là: Bất kỳ hàm logic nào nhận được nhờ phép xếp chồng và thay đổi biến từ các hàm cùng loại cũng sẽ là hàm logic cùng loại đó.

2.8.1. Hàm logic tuyến tính

Hàm logic gọi là tuyến tính nếu nó có thể được viết dưới dạng một đa thức bậc 1:

$$y(x_1, x_2, x_3, \dots, x_n) = a_0 + a_1x_1 + a_2x_2 + \dots + a_nx_n$$

trong đó $a_i, i = \overline{1, n}$, có thể có giá trị 0 hoặc 1.

Vì mỗi tập hợp giá trị của các hệ số $(a_0, a_1, \dots, a_n)$ là một số nhị phân có $(n+1)$ bit, do đó số lượng các hàm tuyến tính n biến là 2^{n+1} .

Định lý 1: Khi xếp chồng và khi thay đổi thứ tự các biến trong hàm tuyến tính sẽ chỉ được các hàm logic tuyến tính.

Chứng minh

a) Trường hợp xếp chồng

Giả thiết có các hàm logic tuyến tính sau đây:

$$G_1 = G_1(x_1, x_2, x_3, \dots, x_n) = a_0 + a_1x_1 + \dots + a_nx_n$$

$$y_1 = y_1(x_1, x_2, \dots, x_{m1}) = a_0 + a_1x_1 + \dots + a_{m1}x_{m1}$$

$$y_2 = y_2(x_1, x_2, \dots, x_{m2}) = a_0 + a_1x_1 + \dots + a_{m2}x_{m2}$$

$$\dots\dots\dots$$

$$y_n = y_n(x_1, x_2, \dots, x_{mn}) = a_0 + a_1x_1 + \dots + a_{mn}x_{mn}$$

Thực hiện nguyên lý xếp chồng, ta thay các x_i bằng các hàm $y_k, k = \overline{1, n}$ sẽ được hàm G_2 , trong đó các hệ số $a_i, i = \overline{1, n}$ được nhân với các hàm y_k . Nếu gộp các số hạng giống nhau lại, ta sẽ được hàm G_2 dưới dạng một đa thức tuyến tính.

b) Trường hợp thay đổi các biến

Từ biểu thức của G_1 ta thấy, dù thay đổi các biến $x_i, i = \overline{1, n}$ như thế nào ta vẫn nhận được hàm logic tuyến tính.

2.8.2. Hàm logic bảo tồn giá trị 0

Hàm logic được gọi là hàm logic bảo tồn giá trị 0, nếu khi mọi biến $x_i, i = \overline{1, n}$ đều nhận giá trị 0 thì hàm cũng nhận giá trị 0, tức là:

$$y(0, 0, \dots, 0) = 0$$

Định lý 2: Khi xếp chồng và khi thay đổi các biến trong các hàm logic bảo tồn giá trị 0 cũng chỉ nhận được hàm bảo tồn giá trị 0.

a) Trường hợp xếp chồng

Giả thiết có các hàm bảo tồn giá trị 0 sau đây:

$$G_1 = G_1(x_1, \dots, x_n)$$

$$\varphi_1 = \varphi_1(y_1, \dots, y_{m1})$$

$$\varphi_2 = \varphi_2(y_1, \dots, y_{m2})$$

$$\dots\dots\dots$$

$$\varphi_n = \varphi_n(y_1, \dots, y_{mn})$$

Nếu thay các biến $x_i, i = \overline{1, n}$ bằng các hàm $\varphi_k, k = \overline{1, n}$ ta sẽ được hàm G_2 , mà:

$$G_2(y_1, \dots, y_{mk}) = G_1(\varphi_1, \dots, \varphi_n)$$

trong đó y_{mk} là biến có mk lớn nhất trong các $m_i, i = \overline{1, n}$. Nếu ta cho $\forall y_i = 0, j = \overline{1, mk}$, ta sẽ có:

$$G_2(0, \dots, 0) = G_1(0, \dots, 0) = 0$$

Do đó G_2 vẫn là hàm bảo tồn giá trị 0.

Vì hàm logic chỉ có thể nhận một trong hai giá trị 0 hoặc 1 tại tập giá trị biến $(0, \dots, 0)$ nên số lượng hàm logic bảo tồn giá trị 0 có n biến là 2^{2^n-1} .

b) Trường hợp thay đổi biến

Tương tự như hàm logic tuyến tính, khi thay đổi các biến ta vẫn được hàm logic bảo tồn giá trị 0 là hiển nhiên.

2.8.3. Hàm logic bảo tồn giá trị 1

Hàm logic bảo tồn giá trị 1 là hàm mà khi tất cả các biến $x_i = 1; i = \overline{1, n}$ thì nó cũng nhận giá trị 1.

$$y(1, 1, \dots, 1) = 1$$

Định lý 3. Khi xếp chồng và khi thay đổi các biến của hàm bảo tồn giá trị 1 sẽ chỉ nhận được hàm bảo tồn giá trị 1.

Việc chứng minh định lý này được tiến hành tương tự như đối với hàm bảo tồn giá trị 0.

Số lượng hàm bảo tồn giá trị 1 cũng bằng 2^{2^n-1} .

2.8.4. Hàm logic đơn điệu

Hàm logic đơn điệu là hàm mà đối với hai tập giá trị biến X_1 và X_2 , trong đó $X_1 \geq X_2$ ta sẽ có $y(X_1) \geq y(X_2)$. Tập giá trị biến $X_1 \geq X_2$, nếu giá trị của mỗi biến trong X_1 đều lớn hơn hay bằng giá trị của biến tương ứng trong X_2 .

Thí dụ, nếu $X_1 = 1011$ và $X_2 = 1010$ thì $X_1 > X_2$.

Người ta đã chứng minh rằng số lượng các hàm logic đơn điệu $\varphi(n)$ nằm trong khoảng:

$$C_n^{(n/2)} \leq \varphi(n) \leq 2AC_n^{(n/2)}$$

trong đó: + $\varphi(n)$ là số lượng các hàm logic đơn điệu n biến;

+ A là một hằng số.

Định lý 4. Khi xếp chồng và khi thay đổi vị trí các biến của hàm logic đơn điệu sẽ chỉ nhận được các hàm logic đơn điệu.

Chứng minh:

a) Trường hợp xếp chồng

Giả thiết có các hàm logic đơn điệu sau đây:

$$G_1 = G_1(x_1, \dots, x_n)$$

$$\varphi_1 = \varphi_1(y_1, \dots, y_{m1})$$

$$\varphi_2 = \varphi_2(y_1, \dots, y_{m2})$$

$$\dots\dots\dots$$

$$\varphi_n = \varphi_n(y_1, \dots, y_{mn})$$

Nếu thay các biến $x_i, i = \overline{1, n}$ bằng các hàm $\varphi_k, k = \overline{1, n}$ ta sẽ được hàm G_2 , mà:

$$G_2(y_1, \dots, y_{mk}) = G_1(\varphi_1, \dots, \varphi_n)$$

Vì đối với hai tập giá trị biến $\langle y_1^*, y_2^*, \dots, y_{mk}^* \rangle \geq \langle y_1, y_2, \dots, y_{mk} \rangle$, giá trị của các $\varphi_i, i = \overline{1, n}$ không giảm đi, tức là:

$$\varphi_i^*(\langle y_1^*, y_2^*, \dots, y_{mk}^* \rangle) \geq \varphi_i(\langle y_1, y_2, \dots, y_{mk} \rangle)$$

Do đó $G_1(\varphi_i^*) \geq G_1(\varphi_i)$ nên $G_2(\langle y_1^*, y_2^*, \dots, y_{mk}^* \rangle) \geq G_2(\langle y_1, y_2, \dots, y_{mk} \rangle)$

Định lý đã được chứng minh.

b) Trường hợp thay đổi vị trí các biến

Giả sử có hàm đơn điệu $G_1(x_1, x_2, \dots, x_n)$. Nếu ta thay đổi vị trí các biến, thí dụ như sau:

$$G_1(x_1, x_2, \dots, x_n) \Rightarrow G_1(x_5, x_n, \dots, x_6)$$

Khi đó, nếu tổ hợp biến $\langle x_1^*, x_2^*, \dots, x_n^* \rangle \geq \langle x_1, x_2, \dots, x_n \rangle$ thì tập giá trị biến $\langle x_5^*, x_n^*, \dots, x_6^* \rangle$ cũng $\geq \langle x_5, x_n, \dots, x_6 \rangle$. Điều này chứng tỏ G_2 cũng là hàm đơn điệu.

2.8.5. Hàm logic đối ngẫu

Hàm $G^*(x_1, x_2, \dots, x_n)$ gọi là đối ngẫu đối với hàm $G(x_1, x_2, \dots, x_n)$, nếu:

$$G^*(x_1, x_2, \dots, x_n) = \overline{G(\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n)}$$

Hàm $G(x_1, x_2, \dots, x_n)$ gọi là tự đối ngẫu, nếu:

$$G(x_1, x_2, \dots, x_n) = \overline{G}(\overline{x}_1, \overline{x}_2, \dots, \overline{x}_n)$$

Số lượng các hàm tự đối ngẫu bằng 2^{2^n-1} , trong đó n là số biến của hàm.

Định lý 5: Khi xếp chồng và khi thay đổi vị trí của các biến của hàm tự đối ngẫu sẽ chỉ được hàm tự đối ngẫu.

a) Trường hợp xếp chồng

Giả thiết có các hàm tự đối ngẫu sau:

$$G_1 = G_1(x_1, \dots, x_n)$$

$$\varphi_1 = \varphi_1(y_1, \dots, y_{m1})$$

$$\varphi_2 = \varphi_2(y_1, \dots, y_{m2})$$

$$\dots\dots\dots$$

$$\varphi_n = \varphi_n(y_1, \dots, y_{mn})$$

Nếu thay các biến $x_i, i = \overline{1, n}$ bằng các hàm $\varphi_k, k = \overline{1, n}$ ta sẽ được hàm G_2 , mà:

$$G_2(y_1, \dots, y_{mk}) = G_1(\varphi_1, \dots, \varphi_n)$$

Vì các $\varphi_i, i = \overline{1, n}$, là tự đối ngẫu, nên: $G_2(\overline{y}_1, \dots, \overline{y}_{mk}) = G_1(\overline{\varphi}_1, \dots, \overline{\varphi}_n)$.

Vì G_1 là tự đối ngẫu nên:

$$G_1(\overline{\varphi}_1, \dots, \overline{\varphi}_{mk}) = G_1(\varphi_1, \dots, \varphi_n)$$

Do đó: $G_2(y_1, \dots, y_{mk}) = \overline{G}_1(\overline{y}_1, \dots, \overline{y}_{mk})$

b) Trường hợp thay đổi vị trí các biến

Giả thiết có hàm tự đối ngẫu :

$$G_1(x_1, x_2, \dots, x_n) = \overline{G}_1(\overline{x}_1, \overline{x}_2, \dots, \overline{x}_n)$$

Vì việc thay đổi vị trí của các biến phải tiến hành đồng thời ở cả hai vế của đẳng thức, nên với bất kỳ sự thay đổi vị trí nào của các biến, hai tập giá trị biến tương ứng ở hai vế vẫn là ngược nhau. Vì vậy, hàm mới nhận được nhờ thay đổi vị trí của các biến vẫn là hàm tự đối ngẫu.

Ngoài các lớp hàm đã kể trên, còn một lớp hàm tuy không có tính chất giữ nguyên lớp hàm khi xếp chồng và thay đổi vị trí các biến, nhưng nó biểu hiện sự tương đương của một cặp hoặc một nhóm các biến. Đó là hàm đối xứng.

Hàm $G_1(x_1, x_2, \dots, x_n)$ được gọi là đối xứng toàn phần nếu nó không thay đổi khi ta đánh số lại một cách tùy ý các biến của nó. Tức là ta có:

$$G_1(x_1, x_2, \dots, x_n) = G_1(y_1, \dots, y_n)$$

trong đó $(y_1, \dots, y_n)$ là một tổ hợp biến đã đánh số lại một cách tùy ý của các biến $x_1, x_2, \dots, x_n$.

Nếu tính chất trên chỉ thoả mãn đối với một cặp biến hoặc một số biến thì ta nói hàm là hàm đối xứng đối với cặp biến hoặc nhóm biến đó.

Thí dụ hàm $y(x_1, x_2, x_3) = \bar{x}_1 x_2 x_3 + x_1 \bar{x}_2 x_3$ là hàm đối xứng đối với cặp biến x_1 và x_2 , vì nếu hoán vị hai biến x_1 và x_2 ta được $y(x_1, x_2, x_3) = \bar{x}_2 x_1 x_3 + x_2 \bar{x}_1 x_3 = \bar{x}_1 x_2 x_3 + x_1 \bar{x}_2 x_3$. Do đó :

$$y(x_1, x_2, x_3) = y(x_2, x_1, x_3)$$

2.9. MẠCH THỰC HIỆN CÁC HÀM LÔGIC CƠ BẢN

Đó là các mạch thực hiện các hàm logic KH, VA và HOAC; VA-KH và HOAC-KH. Ngày nay, các mạch logic cơ bản đã được sản xuất hàng loạt dưới dạng các mạch tích hợp IC (Integrated Circuits). Các vi mạch này gồm nhiều chủng loại, thuộc nhiều hãng sản xuất, đa dạng về số lượng đầu vào - số biến của các hàm logic; về số lượng các phần tử logic trong một vi mạch; về tốc độ, công suất tiêu thụ và khả năng chịu tải của chúng.

Thí dụ, mạch IC SN54/7416 J/W là vi mạch có 6 mạch đảo KH trong một vỏ IC; Vi mạch SN54/7432 là loại mạch HOAC có 4 mạch HOAC trong một vỏ IC, mỗi mạch có hai đầu vào; IC SN54/74S32 lại có 4 mạch HOAC, mỗi mạch có hai đầu vào, nhưng nhờ có hiệu ứng Shottky mà tốc độ làm việc của mạch nhanh hơn. Loại vi mạch SN54/74L32 cũng là loại vi mạch có 4 mạch HOAC hai đầu vào, nhưng sử dụng ít năng lượng hơn. Các vi mạch SN54/7421, SN54/74S21, SN54/74L21 hay SN54/74LS21 là các vi mạch có hai mạch VA 4 đầu vào nhưng khác nhau về tốc độ làm việc, về công suất tiêu thụ.

Tuy các vi mạch rất đa dạng do phải đáp ứng những yêu cầu kỹ thuật khác nhau và do các hãng khác nhau sản xuất bằng công nghệ khác nhau, nhưng về nguyên tắc vẫn thực hiện các chức năng của các hàm logic cơ bản. Do đó nếu nắm vững hoạt động và cấu tạo của các mạch logic điển hình, có thể hiểu và sử dụng chúng đúng, đồng thời vận dụng đối với các vi mạch khác.

Trong thiết kế và sử dụng các mạch logic, cần lưu ý rằng, cùng một mạch nhưng tùy thuộc vào việc vận dụng logic dương hay logic âm mà chức năng hàm sẽ khác nhau. Một cách cụ thể hơn, giữa hai miền logic dương và logic âm mạch có tính đối lẫn giữa các chức năng hàm như sau:

Lôgic dương		Lôgic âm
Mạch VA	$\longleftrightarrow$	Mạch HOAC
Mạch HOAC	$\longleftrightarrow$	Mạch VA
Mạch VA-KH	$\longleftrightarrow$	Mạch HOAC-KH
Mạch HOAC - KH	$\longleftrightarrow$	Mạch VA-KH
Mạch ĐAO	$\longleftrightarrow$	Mạch ĐAO

Chẳng hạn, một mạch nếu dùng logic dương, tức gán mức có điện áp cho giá trị 1, mức không có điện áp cho giá trị 0 là mạch VA thì nếu dùng logic âm, tức gán mức có điện áp cho giá trị 0, mức không có điện áp cho giá trị 1 thì lại là mạch HOAC và ngược lại.

Các mạch logic cơ bản rất đa dạng, nhiều chủng loại. Chức năng, các tham số điện và sơ đồ mạch điện cũng như công nghệ chế tạo và cách sử dụng đã được trình bày trong các sổ tay về vi mạch nên trong phần này chỉ xét một số mạch cơ bản, điển hình như mạch VA, HOAC, KH, VA-KH và HOAC-KH. Mạch KH thường là mạch khuếch đại đảo pha, làm việc ở chế độ tín hiệu lớn (chế độ bão hoà sâu).

2.9.1. Mạch VA, HOAC dùng điốt - điện trở

Các mạch VA và HOAC có thể dễ dàng thực hiện bằng điốt. Tuy nhiên, vì điốt không có khả năng khuếch đại tín hiệu và đặc biệt không có khả năng thực hiện logic KHÔNG (KH), nên không thể chỉ dùng điốt để thực hiện hệ hàm đủ VA, HOAC, KH.

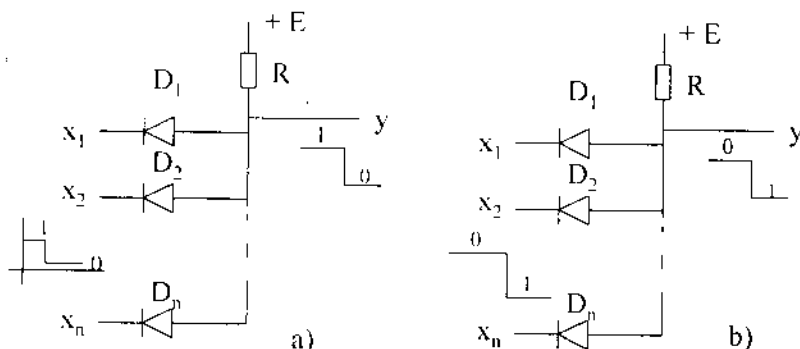
Mạch VA, HOAC dùng điốt cũng đa dạng về số lượng đầu vào và giữa mạch VA và mạch HOAC luôn có tính đối lẩn trong hai miền logic dương và âm.

Thí dụ 2.13

Trên hình 2.10 là mạch VA dùng điốt đối với logic dương, nhưng sẽ là mạch HOAC đối với logic âm.

Thật vậy, với logic dương, mức tín hiệu cao ở đầu vào hay đầu ra là mức 1, ngược lại, mức thấp là mức 0. Khi đó nếu tất cả các đầu vào $x_i, i = 1, n$ có mức logic 1 thì tất cả điốt đều bị ngắt (không cho dòng điện đi qua), do đó đầu ra y cũng có mức điện áp cao, tức mức logic 1.

Ta có $x_1 x_2 \dots x_n = 1.1 \dots 1 = y = 1$.



Hình 2.10. Mạch VA (logic +), mạch HOAC (logic -)

Như vậy mạch trên đây là mạch VA đối với logic dương.

Ngược lại, đối với logic âm - hình 2.10b, ta sẽ có mạch HOAC. Thật vậy, chỉ cần một trong các đầu vào là logic 1 (không có điện áp cao), khi đó điốt tương ứng sẽ thông. Dòng từ nguồn +E sẽ đi qua điốt này và điện áp trên đầu ra y xấp xỉ bằng 0, tức logic 1. Ta có mạch cộng logic (HOAC):

$$y = x_1 + \dots + x_n = 0 + \dots + 1 + \dots + 0 = 1$$

Chỉ khi nào tất cả đầu vào đều có mức logic 0 - tức có điện áp cao, khi đó tất cả điốt đều ngắt (không dẫn). Hầu hết điện áp nguồn sụt trên các điốt. Do đó đầu ra $y = 1$. Tức là:

$$y = x_1 + \dots + x_n = 0 + \dots + 0 + \dots + 0 = 0$$

Vậy mạch trên đối với logic âm là mạch HOAC.

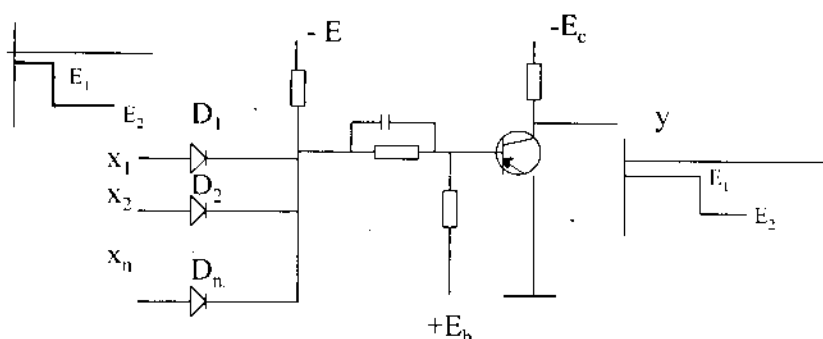
2.9.2. Mạch VA-KH và mạch HOAC-KH

a) Mạch dùng transistor - điốt

Mạch cho trên hình 2.11 là mạch VA-KH đối với logic dương và là mạch HOAC-KH đối với logic âm.

Thật vậy, đối với logic dương, nếu đồng thời tác động lên tất cả đầu vào $x_1, x_2, \dots, x_n$ mức E_2 (mức logic 1) thì tất cả điốt đều ngắt (không cho dòng qua). Khi đó điện áp trên cực gốc của transistor sẽ âm khiến transistor thông ở mức bão hoà. Đầu ra y có mức điện áp xấp xỉ bằng 0 vôn (mức E_1), tương ứng với mức logic 0. Tức là:

$$y = \overline{x_1 \cdot x_2 \dots x_n} = \overline{1 \cdot 1 \cdot 1 \dots 1} = 0$$



Hình 2.11. Mạch VA-KH (logic +), mạch HOAC-KH (logic -)

Chỉ cần ít nhất một trong các đầu vào x_i có mức E_1 , ứng với mức logic 0, ta sẽ có:

$$y = \overline{x_1 \cdot x_2 \dots x_i \dots x_n} = \overline{1 \cdot 1 \dots 0 \dots 1} = 1$$

Đó chính là chức năng mạch VA-KH.

Ngược lại, trong miền logic âm mạch trên sẽ là mạch HOAC-KH.

Thật vậy, nếu các đầu vào x_i đều có mức điện áp E_2 - tương ứng với mức logic 0, khi đó tất cả các diốt đều không thông, đầu ra y do transistor bão hoà, sẽ có mức điện áp E_1 , ứng với mức logic 1. Ta có:

$$y = x_1 + x_2 + \dots + x_i + \dots + x_n = 0 + 0 + \dots + 0 + \dots + 0 = 1$$

Ngược lại, chỉ cần một trong các đầu vào có mức điện áp E_1 , ứng với mức logic 1 thì diốt tương ứng sẽ thông, khiến cho transistor bị ngắt. Điện áp ra y khi đó có mức E_2 là mức logic 0. Ta có:

$$y = x_1 + x_2 + \dots + x_i + \dots + x_n = 1 + 1 + \dots + 0 + \dots + 1 = 0$$

Đó chính là chức năng mạch HOAC-KH.

Trong thực tế số lượng các đầu vào của một mạch và khả năng mắc các tải vào đầu ra không phải là vô hạn. Số lượng tối đa các đầu vào có thể có gọi là hệ số hợp mạch. Tương tự, hệ số phân mạch đầu ra là số lượng tối đa cho phép các mạch cùng loại mắc vào đầu ra của mạch. Các hệ số hợp mạch và phân (rẽ) mạch của các mạch logic dùng transistor thường từ 2 đến 5.

Loại mạch dùng transistor có mức nhiễu thuộc loại trung bình và có tính chống nhiễu cao.

b) Mạch dùng transistor mắc trực tiếp

Là loại mạch rất thông dụng trong vi mạch. Hệ số hợp mạch thường nhỏ hơn 8. Hệ số phân mạch từ 4 đến 6.

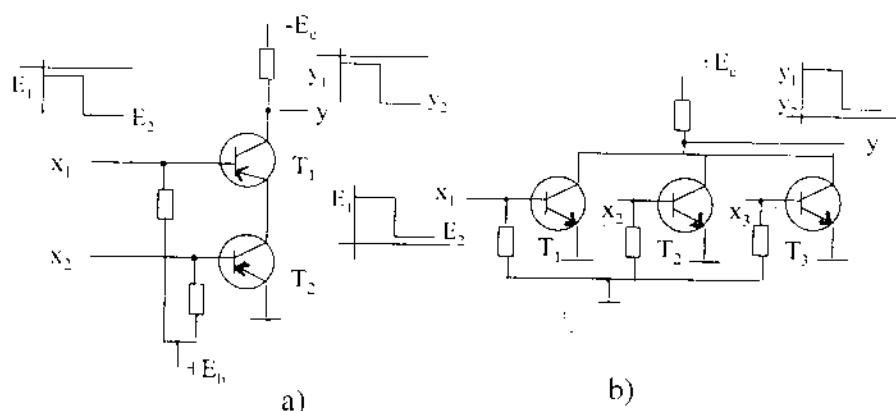
Các transistor trong các mạch này có thể mắc nối tiếp (hình 2.12a) hoặc mắc song song (hình 2.12b).

Đối với mạch ở hình 2.12a, nếu gán giá trị logic 0 cho mức E_1 và y_1 ; mức logic 1 cho mức E_2 và y_2 , tức dùng logic dương, sẽ có mạch VA-KH. Vì chỉ cần ít nhất một đầu vào có mức E_1 (mức logic 0) thì transistor tương ứng sẽ ngắt, không có dòng đi qua các transistor T_1 và T_2 . Khi đó điện áp ra trên y sẽ có mức y_2 (mức logic 1). Chỉ khi cả hai đầu vào x_1, x_2 đều có mức điện áp E_2 (mức logic 1) thì cả hai T_1 và T_2 đều thông bão hoà và đầu ra mới có mức điện áp y_1 (mức logic 0). Tức là:

$$y = x_1 \cdot x_2 = \overline{1 \cdot 0} = \overline{0 \cdot 1} = 1$$

$$y = x_1 \cdot x_2 = \overline{1 \cdot 1} = 0$$

Đó chính là chức năng của mạch VA-KH.



Hình 2.12. Mạch VA-KH hoặc mạch HOAC-KH mắc nối tiếp hoặc song song các transistor

Ngược lại, nếu gán giá trị logic 0 cho E_2 và y_2 ; giá trị logic 1 cho E_1 và y_1 , tức dùng logic âm thì khi đó, chỉ khi nào cả hai đầu vào x_1, x_2 đều ở mức điện áp E_2 (mức logic 0) thì y mới bằng y_1 , tức có mức logic 1. Trong tất cả các trường hợp khác đầu ra y sẽ có mức điện áp y_2 , tức mức logic 0. Đó chính là chức năng của mạch HOAC-KH.

Với hình 2.12b nếu gán mức logic 1 cho E_1 và y_1 , mức logic 0 cho E_2 và y_2 , ta sẽ có mạch VA-KH vì khi đó chỉ cần ít nhất một trong ba đầu vào có mức E_2 (mức logic 0) thì transistor tương ứng sẽ thông và bão hoà, mức điện áp ở đầu ra y sẽ là y_1 , tức mức logic 1. Tức là:

$$y = x_1 \cdot x_2 \cdot x_3 = \overline{0.1.1} = \overline{1.0.1} = 1$$

Chỉ khi nào tất cả ba đầu vào x_1, x_2, x_3 đều ở mức E_1 , tức mức logic 1 thì cả ba transistor đều ngắt, đầu ra y sẽ có điện áp mức y_2 , tức mức logic 0. Ta có:

$$y = x_1 \cdot x_2 \cdot x_3 = \overline{1.1.1} = 0$$

Như vậy với logic âm ta có mạch VA-KH.

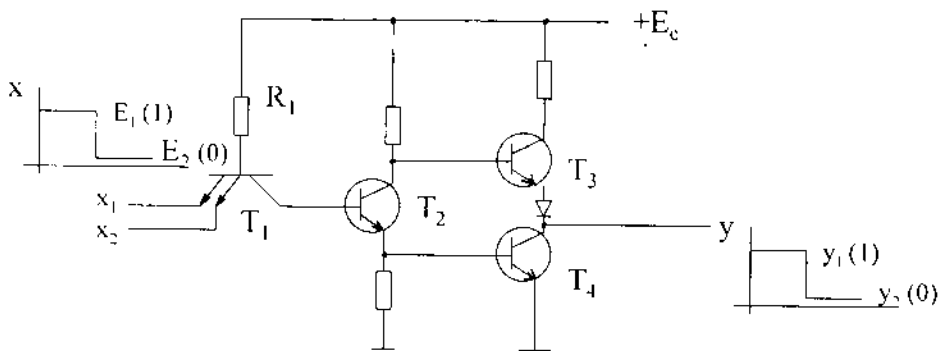
Bằng cách chứng minh tương tự với logic âm, mạch ở hình 2.12b là mạch HOAC-KH.

c) Mạch dùng transistor nhiều emitter

Hình 2.13 là mạch VA-KH hai đầu vào với logic dương. Hai đầu vào x_1, x_2 là hai emitter của transistor T_1 . Mức logic quy định như trên hình vẽ.

Nếu E_1 và E_2 có mức logic 1 thì dòng qua R_1 đến cực gốc của T_2 qua mặt ghép gốc - collector của T_1 . Khi đó có dòng phân cực thuận, dòng này làm T_2 thông. Điện áp trên cực góp T_2 giảm đi, làm thay đổi điện áp từ gốc T_3 đến gốc T_4 . Transistor T_3 ngắt còn T_4 bão hoà. Khi đó $y = y_2$, tức mức logic 0.

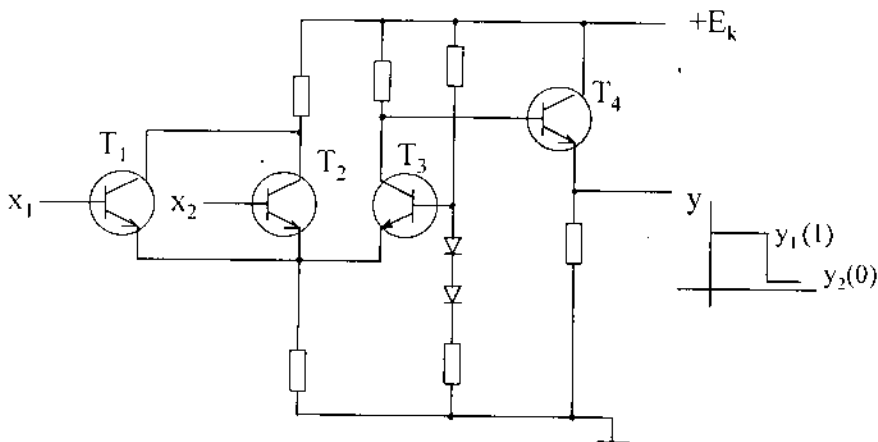
Nếu x_1 hoặc x_2 hoặc cả hai đều ở mức E_2 , tức logic 0 thì dòng qua R_1 sẽ qua emitter của T_1 xuống đất. Do đó T_2 ngắt, T_3 thông còn T_4 ngắt. Khi đó đầu ra $y = y_1$, tức mức logic 1. Hoạt động như vậy chính là chức năng của mạch VA-KH.



Hình 2.13. Mạch VA-KH dùng transistor nhiều emitter

d) Mạch logic ghép emitter (ECL)

Trong các mạch logic, nếu các transistor làm việc ở chế độ bão hoà sâu sẽ làm giảm tốc độ làm việc của mạch. Mạch logic loại ECL loại trừ hiện tượng bão hoà sâu của transistor. Hình 2.14 là mạch HOAC với logic dương thực hiện theo kiểu ECL. Mạch gồm một bộ khuếch đại thuật toán, mạch cung cấp thiên áp và bộ khuếch đại lặp lại ở đầu ra.



Hình 2.14. Mạch dùng transistor nhiều emitter

Mạch thiên áp làm việc sao cho T_3 thông nếu cả hai đầu vào x_1 và x_2 ở mức logic 0 (T_1, T_2 ngắt). Lúc đó T_4 ngắt và đầu ra y có mức logic 0.

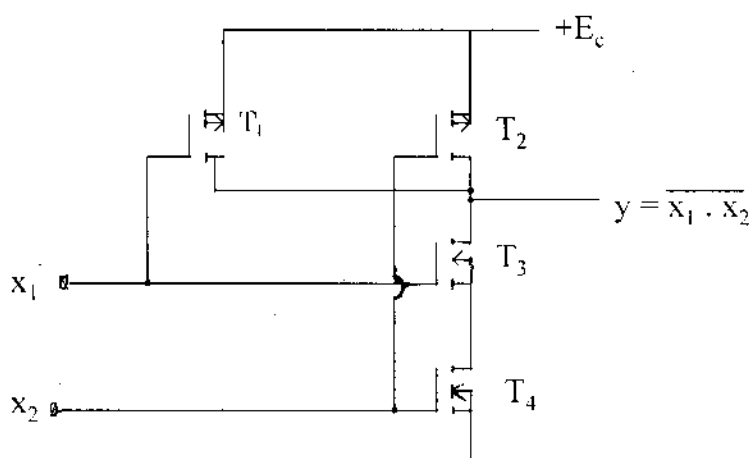
Nếu x_1 hoặc x_2 hoặc cả hai ở mức logic 1 thì T_1 hoặc T_2 hoặc cả hai thông, làm cho dòng đi qua T_3 giảm; điện áp còlectơ của T_3 tăng lên, T_4 thông và ở đầu ra y có giá trị logic 1. Đó chính là chức năng của mạch HOAC.

Nếu có bộ khuếch đại lặp lại mà cực gốc nối với côlectơ của T_2 thì đầu ra của bộ khuếch đại lặp lại này sẽ thực hiện chức năng HOAC-KH. Các loại mạch HOAC và HOAC-KH là các loại mạch thường dùng kiểu mắc ECL.

Trong các mạch ECL, họ mạch 1000/10.000 có hệ số hợp mạch đầu vào và hệ số rẽ mạch đầu ra là 5 và 50. Chúng thực hiện một lớp khá rộng các hàm lôgic và chúng có tốc độ làm việc cao hơn loại TTL bình thường và có khả năng làm việc kết hợp với loại TTL.

e) Mạch dùng transistor trường

Mạch lôgic dùng transistor trường tiêu thụ ít năng lượng, có độ chống nhiễu cao. Mạch VA-KH dùng transistor trường cho trên hình 2.15.



Hình 2.15. Mạch VA-KH dùng transistor trường

Khi $x_1 = x_2 = "1"$, T_1 và T_2 sẽ ngắt, T_3 và T_4 thông. Trên đầu ra của mạch y sẽ có lôgic 0, tức là $y = \overline{x_1 x_2} = \overline{1.1} = 0$. Chỉ cần một trong hai đầu vào hoặc cả hai đồng thời có mức lôgic 0 thì T_1 hoặc T_2 hoặc cả hai sẽ thông, trong khi T_3 , T_4 sẽ ngắt. Do đó y có lôgic 1. Tức là:

$$y = \overline{x_1 x_2} = \overline{0.1} = \overline{1.0} = \overline{0.0} = 1$$

Như vậy mạch thực hiện hàm VA-KH đối với lôgic dương.

Nếu đổi transistor loại n thành loại p và ngược lại thì mạch trên hình 2.15 sẽ trở thành mạch HOAC-KH.

Loại mạch điển hình dùng transistor trường là các loại mạch VA-KH và HOAC-KH. Chúng có hệ số tải lớn, công suất tiêu thụ rất nhỏ. Dưới dạng các vi mạch, các họ dùng transistor trường là: CD 4000, MM 54C/74C.

BÀI TẬP CHƯƠNG 2

- Hãy viết các tổ hợp biến có thể có dưới dạng số nhị phân của các hàm 3, 4 và 5 biến. Tính tổng số các hàm logic khác nhau có thể có của 3, 4 và 5 biến.
- Có bao nhiêu tổ hợp n biến ở hệ đếm cơ số 2 và có bao nhiêu hàm logic khác nhau có thể có của chúng?
- Chứng minh các đẳng thức sau đây:
 - $x_1 + x_1x_2 = x_1$; $x_2(x_2 + x_3) = x_2$; $x_1 + \bar{x}_1x_2 = x_1 + x_2$; $x_1(\bar{x}_1 + x_2) = x_1x_2$;
 - $(x_1 + \bar{x}_2)(x_2 + x_3) = x_2x_1 + \bar{x}_2x_3$; $x_2\bar{x}_3 + x_3\bar{x}_1 + \bar{x}_2x_1 = \bar{x}_2x_3 + \bar{x}_3x_1 + \bar{x}_1x_2$;
 $x_3x_1 + \bar{x}_2x_1 + x_2x_3 = x_2x_3 + \bar{x}_2x_1$; $x_2x_1 + x_1x_3 + x_2x_3 = (x_1 + x_3)(x_1 + x_2)(x_2 + x_3)$
 - $x_2\bar{x}_3 + \bar{x}_2\bar{x}_3 + x_2x_3 = x_2 + \bar{x}_3$; $x_1\bar{x}_2 + x_2x_3 + x_1\bar{x}_3 = x_1 + x_2x_3$;
 $(x_1 + \bar{x}_2\bar{x}_3)(x_1x_2 + \bar{x}_3)(x_2 + x_3) = x_1x_2$; $\bar{x}_1\bar{x}_2 + x_1\bar{x}_2 + \bar{x}_2x_3 + x_2x_3 = \bar{x}_2\bar{x}_3$
- Dùng tính chất của các hàm VA-KH, HOAC-KH và KH để chứng minh các đẳng thức sau đây:
 - $x_1 \mid (x_1 \mid (x_2 \mid x_2)) = x_1 \mid x_2$
 - $(x_1 \downarrow x_3) \downarrow (x_1 \downarrow (x_3 \downarrow x_3)) \downarrow ((x_1 \downarrow x_1) \downarrow x_2) \downarrow ((x_1 \downarrow x_1) \downarrow (x_2 \downarrow x_2)) = (x_1 \downarrow x_1) \downarrow x_1$
 - $x_1 \downarrow ((x_1 \downarrow x_1) \downarrow (x_2 \downarrow x_2)) = x_1 \downarrow x_1$
- Chứng minh định luật De Morgan cho n biến:
 - $\overline{(x_1x_2x_3 \dots x_n)} = \bar{x}_1 + \bar{x}_2 + \dots + \bar{x}_n$
 - $\overline{x_1 + x_2 + \dots + x_n} = \bar{x}_1 \cdot \bar{x}_2 \cdot \bar{x}_3 \dots \bar{x}_n$
- Chứng minh rằng:

$$f(x_1, x_2, \dots, x_n) = a_1x_1 \oplus a_2x_2 \oplus \dots \oplus a_nx_n = 1$$
 khi có một số lẻ các số hạng của đa thức bằng 1.
- Cho các hàm sau đây:
 - $y_1 = f^1(1001, 1101, 1110, 1010, 1000)$
 - $y_2 = f^1(111, 101, 010, 100)$
 - $y_3 = f^1(3, 5, 7, 8, 12); n = 4, b = 10$
 - $y_4 = \begin{cases} f^1(2, 4, 5, 12); n = 5, b = 10 \\ f^-(15, 22, 26, 32) \end{cases}$
 - $y_5 = f^1(0, 3, 5, 9, 17, 25, 36, 37, 42, 43, 44, 55, 56); n = 6, b = 10$
 Hãy biểu diễn chúng trên mặt phẳng và bằng bảng giá trị thực.
- Cho các hàm sau:
 - $y_1 = \begin{cases} f^1(001, 010, 101, 111) \\ f^-(000, 011) \end{cases}$

$$b) y_2 = \begin{cases} f^1(1, 3, 4, 8, 12) \\ f^0(2, 5); n = 4, b = 10 \end{cases}$$

Hãy biểu diễn chúng trên mặt phẳng và bằng bảng giá trị thực.

9. Viết dạng giải tích của $m_0, m_2, m_5, m_7, m_{12}, M_1, M_3, M_4, M_{16}$ trong các hệ hàm đủ sau:

a) $\{VA, HOAC, KH\}$

b) $\{VA - KH\}$

c) $\{HOAC - KH\}$

10. Tìm các dạng tuyến chuẩn, dạng hội chuẩn, VA-KH và HOAC-KH của các hàm sau đây:

a) $y_1 = x_1 (\overline{x_2 x_3} + \overline{x_2 x_3}) + (x_2 x_3 + \overline{x_2 x_3}) x_4$

b) $y_2 = x_1 (x_2 x_3 + x_4) + (x_2 + x_3) (x_3 + x_4)$

c) $y_3 = \overline{x_1} x_2 x_4 + x_1 (\overline{x_2 x_3})$

d) $y_4 = (\overline{x_3} + x_4) x_2 x_1 + (\overline{x_1 x_2}) x_4 + \overline{x_1} \overline{x_3}$

e) $y_5 = (\overline{x_2} \overline{x_4}) \overline{x_1} + x_5 (\overline{x_2 x_3 x_4})$

f) $y_6 = ((x_1 | x_2) | (x_1 | x_2) | x_3) | x_4$

g) $y_7 = (x_1 \downarrow (x_3 \downarrow x_4)) \downarrow (x_3 \downarrow (x_3 \downarrow x_4))$

h) $y_8 = (((x_1 + \overline{x_2}) x_3 \downarrow \overline{x_4}) + x_1 x_2 (\overline{x_3} + \overline{x_4}))$

i) $y_9 = \sum m_i; i = 0, 2, 3, 5, 9, 12; n = 4$

k) $y_{10} = \prod M_i; i = 4, 6, 8, 10, 14; n = 4$

11. Biểu diễn các hàm của bài tập 10 trên mặt phẳng.

12. Vẽ mạch thực hiện các hàm logic của bài tập 10.

13. Cho các hàm sau đây với $n = 4$ và cơ số đếm $b = 10$:

$$y_1 = f^1(0, 3, 5, 6, 10); y_2 = f^1(0, 5, 8, 12); y_3 = \{f^1(0, 4, 5, 6, 11); f^-(2, 7)\}$$

$$y_4 = \{f^1(0, 2, 4, 10); f^-(7)\}; y_5 = \{f^1(1, 2, 5, 7, 8); f^0(0, 9, 10, 11, 12, 13)\}$$

$$y_6 = \{f^1(0, 9, 10, 11, 12, 13); f^0(1, 2, 5, 7, 8)\}$$

Hãy viết các hàm:

a) $Z_1 = y_1 + y_2; Z_2 = y_3 + y_4; Z_3 = y_5 + y_6$

b) $Z_4 = y_1 \cdot y_2; Z_5 = y_3 \cdot y_4; Z_6 = y_5 \cdot y_6$

c) $(\overline{y_1} + y_2)(y_1 + \overline{y_2})$

14. Hãy kiểm tra xem các cặp hàm sau đây có bằng nhau hay không, biết $n = 4$ và $b = 10$?

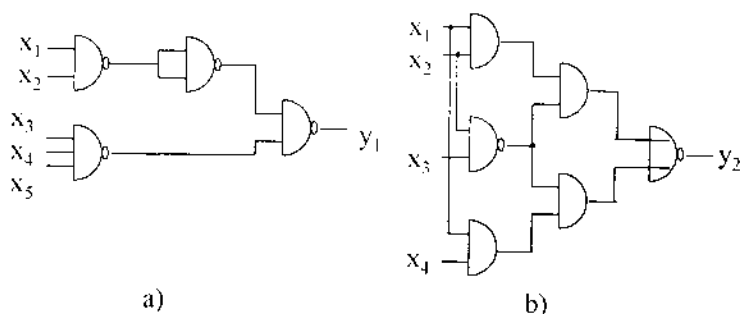
a) $y_1 = \{f^1(1, 4, 8, 12, 13); f^-(6, 9, 10, 11, 14, 15)\}$

$y_2 = \{f^1(0, 2, 3, 5, 7); f^0(1, 4, 8, 12, 13)\}$

b) $y_3 = \{f^1(1, 2, 5, 7, 8); f^1(0, 9, 10, 11, 12, 13)\}$

$y_4 = \{f^1(0, 9, 10, 11, 12, 13); f^0(1, 2, 5, 7, 8)\}$.

15. Cho các mạch sau đây, hãy viết dạng giải tích của các hàm y_1 và y_2 (hình 2.16).



Hình 2.16

16. Cho các cột như bảng dưới đây:

STT	Cột 8421	Cột dư 3	Cột 2421	Cột Gray
0	0000	0011	0000	0000
1	0001	0100	0001	0001
2	0010	0101	0010	0011
3	0011	0110	0011	0010
4	0100	0111	0100	0110
5	0101	1000	1011	0111
6	0110	1001	1100	0101
7	0111	1010	1101	0100
8	1000	1011	1110	1100
9	1001	1100	1111	1101

Hãy viết các hàm logic biến đổi từ cột này sang cột khác trong hệ hàm đủ VA-KH và HOAC - KH.

17. Hàm logic $f(x_1, x_2, x_3, x_4)$ được biểu diễn dưới dạng sau:

$$f(x_1, x_2, x_3, x_4) = \bar{x}_3 f_1(x_1, x_2, x_4) + x_3 f_2(x_1, x_2, x_4) = (\bar{x}_3 + f_3(x_1, x_2, x_4))(x_3 + f_4(x_1, x_2, x_4)).$$

Hãy tìm hàm lôgic f_1, f_2, f_3, f_4 nếu:

$$a) f(x_1, x_2, x_3, x_4) = (\bar{x}_1 + x_3)(\bar{x}_2 + x_4) + \bar{x}_3 \bar{x}_4$$

$$b) f(x_1, x_2, x_3, x_4) = (x_1 + \bar{x}_2 + x_3)(\bar{x}_1 + \bar{x}_2 + x_4)$$

$$c) f(x_1, x_2, x_3, x_4) = \bar{x}_1 x_2 + x_3 x_4 + x_1 x_2 \bar{x}_3$$

$$d) f(x_1, x_2, x_3, x_4) = x_1 \bar{x}_3 \bar{x}_4 + \bar{x}_1 x_3 + (x_1 + \bar{x}_2) x_3$$

18. Tìm các hàm lôgic tuyến tính, đơn điệu, đối ngẫu trong số các hàm lôgic cơ bản hai biến.

CHƯƠNG 3

PHƯƠNG PHÁP ĐƠN GIẢN CÁC HÀM LÔGIC

3.1. NHỮNG KHÁI NIỆM CƠ BẢN

3.1.1. Hàm đánh giá độ phức tạp

Thiết kế các mạch logic sao cho đơn giản, tin cậy vừa là chỉ tiêu kinh tế, vừa là chỉ tiêu kỹ thuật cần phải được quan tâm. Bớt đi được một số mạch logic không những giảm chi phí mà còn giảm bớt tiêu thụ nguồn, đặc biệt đối với các mạch hệ thống lớn việc giảm tiêu thụ nguồn là hết sức quan trọng. Trong thực tế tìm một dạng hàm đơn giản của các thiết bị phức tạp cho phép giảm đi rất nhiều linh kiện, có khi giảm đi vài lần. Do vậy đơn giản các hàm logic không những làm giảm giá thành sản phẩm mà còn làm tăng độ tin cậy của thiết bị. Nhưng cũng chú ý rằng, không phải bao giờ mạch đơn giản cũng tin cậy hơn.

Để đánh giá mức độ phức tạp của một mạch thực hiện một hàm đã cho, ta dùng hàm đánh giá F_d đối với một hàm ở dạng hàm tuyến chuẩn. Rõ ràng sự phức tạp của một hàm phụ thuộc vào số lượng các biến có phủ định, số lượng các số hạng của dạng đã cho và số lượng các biến trong mỗi số hạng. Do đó hàm đánh giá được xác định như sau:

$$F_d(y(x_1, \dots, x_n)) = \sigma(0) + \rho + \sum r_i$$

trong đó: $\sigma(0)$ là số lượng các biến có dấu phủ định;

ρ là số lượng các số hạng của dạng tuyến chuẩn của hàm đã cho;

r_i là số lượng biến trong số hạng thứ i của dạng tuyến chuẩn.

Thí dụ 3.1: Xác định hàm đánh giá của hàm sau đây:

$$y = x_1 \bar{x}_2 + x_1 x_2 x_3 + x_2 \bar{x}_3$$

$$F_d(y) = 2 + 3 + (2 + 3 + 2) = 12$$

Giá trị của hàm đánh giá càng cao thì độ phức tạp của mạch càng lớn. Trong hai hàm logic, hàm nào có hàm đánh giá thấp hơn thì hàm đó đơn giản hơn.

Từ hàm đánh giá chúng ta thấy rằng, các phương pháp đơn giản hàm phải làm sao giảm được số lượng các số hạng, giảm được số lượng các biến trong mỗi số hạng và

giảm số lượng các biến có phủ định. Tuy vậy, cho tới nay các phương pháp đơn giản hàm đều nhằm làm giảm số lượng các số hạng và số lượng các biến trong mỗi số hạng; mà chưa có phương pháp nào nhằm giảm số lượng các biến có phủ định. Lý do cũng đơn giản: Một là, nhiều khi giữa biến có phủ định và không có phủ định cũng không có sự phân biệt về độ phức tạp, vì có thể cùng lấy đồng thời. Mặt khác bản thân thuật toán để giảm số lượng các biến có phủ định cũng chưa được xác định.

3.1.2. Các dạng tuyển chuẩn khác nhau

Trong khi thực hiện đơn giản hàm, chúng ta thường gặp một số khái niệm sau:

1) Dạng tuyển chuẩn

Là dạng tổng các tích của các biến nói chung, nó có thể là dạng tuyển chuẩn chính tắc hay không chính tắc.

2) Dạng tuyển chuẩn ngắn nhất

Là dạng tuyển chuẩn mà mỗi số hạng của nó không bị bất kỳ số hạng nào khác phủ về mặt chức năng.

Nếu có hai số hạng $A = x_1x_2$ và $B = x_1x_2\bar{x}_3$, ta nói số hạng A phủ số hạng B hay B bị A phủ, vì nếu biến đổi A như sau: $A.(x_3 + \bar{x}_3) = x_1x_2(x_3 + \bar{x}_3) = x_1x_2x_3 + x_1x_2\bar{x}_3$. Rõ ràng B chỉ thực hiện một phần của A. Chức năng của A phải thực hiện lớn hơn của B, do đó A phủ B.

Mỗi số hạng của dạng tuyển chuẩn ngắn nhất gọi là một implicand đơn giản.

3) Dạng tuyển chuẩn tối thiểu

Dạng tuyển chuẩn ngắn nhất chưa chắc chắn đã là dạng đơn giản nhất, vì tuy mỗi số hạng không bị số hạng khác phủ, nhưng có thể bị một nhóm các số hạng khác phủ. Trong trường hợp này nhóm số hạng này có thể thay thế cho nó. Từ đó ta có khái niệm về dạng tuyển chuẩn tối thiểu.

Dạng tuyển chuẩn tối thiểu là dạng tuyển chuẩn ngắn nhất mà mỗi số hạng không bị các số hạng khác phủ.

Mỗi số hạng của dạng tuyển chuẩn tối thiểu là một implicand đơn giản thực thụ (thực sự).

Như vậy, mọi phương pháp đơn giản hàm đều phải dẫn đến dạng tuyển chuẩn tối thiểu.

Tuy dạng tuyển chuẩn tối thiểu là dạng hàm đơn giản nhất, nhưng đối với một hàm cho trước, có thể có một số dạng tuyển chuẩn tối thiểu, tuy khác nhau về thành phần kết cấu logic nhưng tương đương nhau về độ phức tạp và cùng thực hiện một chức năng logic. Ngoài ra cần chú ý rằng, các phương pháp đơn giản hàm có hiệu quả và được sử dụng rộng rãi hiện nay đều được thực hiện trong hệ hàm đủ VA, HOAC, KH.

Tới thời điểm hiện tại chưa xuất hiện phương pháp nào có hiệu quả mà thực hiện trong các hệ hàm đủ khác. Mặt khác, trên thực tế khi chuyển từ dạng tuyển chuẩn tối thiểu trong hệ hàm đủ VA, HOAC, KH sang dạng VA-KH hay HOAC-KH không làm phức tạp thêm mạch (tuy không được chứng minh một cách chặt chẽ). Do vậy người ta chấp nhận là khi đơn giản hàm thì thực hiện trong hệ hàm đủ VA, HOAC, KH, muốn thực hiện bằng các mạch VA-KH hay HOAC-KH ta dùng phép biến đổi toán học.

Trong tài liệu này sẽ đề cập một số phương pháp đơn giản hàm và hệ hàm thông dụng và có hiệu quả, phục vụ công tác học tập và nghiên cứu hàng ngày, đồng thời cũng là cơ sở cho việc nghiên cứu phát triển tiếp theo trong lĩnh vực này.

Ba phương pháp đơn giản hàm thông dụng là: (1) Phương pháp biến đổi toán học; (2) Phương pháp hình học, còn gọi là phương pháp Karnaugh hay phương pháp Bảng đối xứng; (3) Phương pháp McClusky.

3.2. PHƯƠNG PHÁP BIẾN ĐỔI TOÁN HỌC

Phương pháp biến đổi toán học sử dụng các quan hệ logic và các định luật giao hoán, kết hợp, phân phối, De Morgan và các định luật khác để biến đổi, nhằm rút gọn các biểu thức logic. Trong quá trình biến đổi, ta có thể cộng thêm vào một biểu thức một đại lượng có giá trị bằng 0 hay nhân nó với một đại lượng hay biểu thức tương đương giá trị 1. Tóm lại, trong đơn giản hàm cần chú ý các bước sau:

- Tách và bỏ đi các phần thừa;
- Dùng các quan hệ và các định luật của các hàm logic cơ bản để đơn giản hàm.

Thí dụ 3.2

1) Đơn giản hàm $y = x_1x_2x_3x_4 + \bar{x}_1x_2x_3x_4$

Sử dụng định luật phân phối, đặt thừa số chung ta có:

$$y = x_2x_3x_4(x_1 + \bar{x}_1) = x_2x_3x_4$$

2) Đơn giản hàm:

$$\begin{aligned} y &= x_1x_2x_3 + \bar{x}_1\bar{x}_2x_3 + \bar{x}_1\bar{x}_2\bar{x}_3 + x_1\bar{x}_2x_4 + \bar{x}_1x_2x_4 + x_1x_2\bar{x}_3x_4 \\ &= x_1x_2x_3(x_4 + \bar{x}_4) + \bar{x}_1\bar{x}_2x_3(x_4 + \bar{x}_4) + \bar{x}_1\bar{x}_2\bar{x}_3(x_4 + \bar{x}_4) + \bar{x}_1x_2x_4 + x_1\bar{x}_2x_4 + x_1x_2\bar{x}_3x_4 \\ &= x_4(x_1x_2x_3 + \bar{x}_1\bar{x}_2x_3 + \bar{x}_1\bar{x}_2\bar{x}_3 + x_1\bar{x}_2 + \bar{x}_1x_2 + x_1x_2\bar{x}_3) + \bar{x}_1x_2x_4 + x_1\bar{x}_2x_4 + x_1x_2\bar{x}_3x_4 \\ &= x_4(x_1x_2 + \bar{x}_1\bar{x}_2 + x_1\bar{x}_2 + \bar{x}_1x_2) + x_1x_2x_3 + \bar{x}_1\bar{x}_2x_3 + \bar{x}_1\bar{x}_2\bar{x}_3 \\ &= x_4 + x_1x_2x_3 + \bar{x}_1\bar{x}_2. \end{aligned}$$

Vậy cuối cùng ta có dạng tuyển chuẩn tối thiểu của hàm đã cho:

$$y = x_4 + x_1x_2x_3 + \bar{x}_1\bar{x}_2$$

Cần chú ý rằng phương pháp biến đổi toán học trong một số trường hợp không thể đơn giản hết các số hạng mà thực tế là thừa.

Thí dụ, hàm $y = x_1x_2 + \bar{x}_1x_3 + x_2x_3$ có số hạng thừa là x_2x_3 , nhưng không có cách biến đổi nào có thể khử bỏ được. Từ đây có thể rút ra quy tắc sau:

Trong một biểu thức có dạng: $x_i.A + \bar{x}_i.B$, trong đó A và B là các đa thức, ta có thể viết:

$$x_i.A + \bar{x}_i.B = x_i.A + \bar{x}_i.B + A.B$$

Nhờ đó trong một số trường hợp có thể tiếp tục đơn giản.

Thí dụ 3.3. Đơn giản hàm sau:

$$y = \bar{x}_1x_2 + x_1x_3 + x_2\bar{x}_3$$

ta biến đổi như sau:

$$y = \bar{x}_1x_2 + x_1x_2 + x_2\bar{x}_3 = \bar{x}_1x_2 + x_1x_3 + x_2\bar{x}_3 + x_2x_3$$

$$y = \bar{x}_1x_2 + x_1x_3 + x_2(\bar{x}_3 + x_3) = \bar{x}_1x_2 + x_1x_3 + x_2 = x_2(\bar{x}_1 + 1) + x_1x_3$$

$$\Rightarrow y = x_2 + x_1x_3$$

Ngoài ra, nhìn chung phương pháp biến đổi toán học chỉ đạt tới dạng tuyến chuẩn ngắn nhất. Đối với một hàm lôgic đã cho có thể có một số dạng tuyến chuẩn ngắn nhất.

Thí dụ 3.4

Tìm dạng tuyến chuẩn ngắn nhất của hàm sau đây:

$$y(x_1, x_2, x_3) = x_1x_2\bar{x}_3 + x_1\bar{x}_2x_3 + \bar{x}_1x_2x_3 + x_1\bar{x}_2\bar{x}_3 + \bar{x}_1x_2\bar{x}_3 + \bar{x}_1\bar{x}_2x_3$$

Nếu biến đổi ta có các dạng tuyến chuẩn ngắn nhất sau:

$$DTCNN1 = x_1\bar{x}_2 + \bar{x}_1x_3 + x_2\bar{x}_3$$

$$DTCNN2 = \bar{x}_1x_2 + x_1\bar{x}_3 + \bar{x}_2x_3$$

Để có dạng tuyến chuẩn tối thiểu cần áp dụng phương pháp Kvaina.

Phương pháp Kvaina là phương pháp lập bảng và thông qua xử lý thông tin để chọn lấy những implicăng đơn giản thực thụ của hàm.

Nội dung của phương pháp Kvaina như sau:

+ Lập một bảng có:

- Số cột bằng số lượng các số hạng của dạng tuyến chuẩn ban đầu. Mỗi cột ứng với một số hạng.

- Số hàng bằng số lượng các implicăng đơn giản của dạng tuyến chuẩn ngắn nhất của hàm. Mỗi hàng ứng với một implicăng đơn giản.

+ Điền vào bảng:

Việc điền vào các ô là giao của hàng và cột được thực hiện nhau sau: Nếu implicăng ở đầu hàng phủ số hạng ở đầu cột thì đánh dấu "x" vào ô tương ứng.

+ Thực hiện algôrit tìm các implicăng đơn giản thực thụ như sau:

a) Xem có cột nào chỉ có một dấu “x” không? Nếu có thì chọn implicăng đơn giản ở đầu hàng tương ứng làm implicăng đơn giản thực thụ. Sau đó gạch bỏ các cột có chứa dấu “x” của hàng đã chọn, rồi chuyển sang thực hiện bước b). Nếu không thì thực hiện bước c).

b) Xem các cột đã bị gạch bỏ hết chưa?

+ Nếu chưa thì thực hiện a).

+ Nếu đã hết, thực hiện bước d).

c) Xem hàng nào có nhiều dấu “x” nhất thì chọn implicăng ở đầu hàng đó làm implicăng đơn giản thực thụ; nếu có một số hàng có nhiều dấu “x” như nhau thì chọn implicăng ở đầu của một trong các hàng đó. Gạch bỏ tất cả các cột có chứa dấu “x” của hàng đã chọn. Quay lại b).

d) Kết thúc. Dạng tuyến chuẩn tối thiểu là tổng logic của những implicăng đơn giản thực thụ đã được chọn.

Thí dụ 3.5

Cho hàm logic dưới dạng tuyến và kết quả đơn giản bằng biến đổi toán học như sau:

$$y = (x_1, x_2, x_3, x_4) = \bar{x}_1 \bar{x}_2 x_3 x_4 + \bar{x}_1 x_2 \bar{x}_3 \bar{x}_4 + \bar{x}_1 x_2 \bar{x}_3 x_4 + \bar{x}_1 x_2 x_3 x_4 + x_1 \bar{x}_2 \bar{x}_3 x_4 + x_1 \bar{x}_2 x_3 x_4$$

$$DTCNN = \bar{x}_1 x_3 x_4 + \bar{x}_2 x_3 x_4 + \bar{x}_1 x_2 \bar{x}_3 + \bar{x}_1 x_2 x_4 + x_1 \bar{x}_2 x_4$$

Bảng 3.1 là bảng Kvaina đã điền các dấu “x” để tìm các implicăng đơn giản thực thụ cho hàm đã cho.

Bảng 3.1. Bảng Kvaina

	$\bar{x}_1 \bar{x}_2 x_3 x_4$	$\bar{x}_1 x_2 \bar{x}_3 \bar{x}_4$	$\bar{x}_1 x_2 \bar{x}_3 x_4$	$\bar{x}_1 x_2 x_3 x_4$	$x_1 \bar{x}_2 \bar{x}_3 x_4$	$x_1 \bar{x}_2 x_3 x_4$
$\bar{x}_1 x_3 x_4$	x			x		
$\bar{x}_2 x_3 x_4$	x					x
$\bar{x}_1 x_2 \bar{x}_3$		x	x			
$\bar{x}_1 x_2 x_4$			x	x		
$x_1 \bar{x}_2 x_4$					x	x

Áp dụng algôrit trên đây, ta được dạng tuyến chuẩn tối thiểu của hàm như sau:

$$y(x_1, x_2, x_3, x_4) = \bar{x}_1 x_3 x_4 + \bar{x}_1 x_2 \bar{x}_3 + x_1 \bar{x}_2 x_4$$

3.3. ĐƠN GIẢN HÀM BẰNG PHƯƠNG PHÁP HÌNH HỌC

3.3.1. Phương pháp dùng bảng kế cận

Phương pháp đơn giản hàm dựa trên biểu diễn hàm trên mặt phẳng hay trên bảng kế cận gọi là phương pháp Karnaugh. Phương pháp này cho phép tìm dạng tuyến chuẩn tối thiểu của hàm cho dưới dạng tuyến chuẩn chính tắc. Trong trường hợp hàm cho dưới dạng tuyến chuẩn không chính tắc thì để biểu diễn lên bảng kế cận ta lại phải quay lại dạng tuyến chuẩn chính tắc. Đó là một nhược điểm của phương pháp Karnaugh. Tuy vậy phương pháp Karnaugh là phương pháp rất trực quan, đặc biệt đối với các hàm có số biến $n \leq 4$. Với $n \geq 5$, phương pháp Karnaugh dựa trên tính chất kế cận của các ô nằm sát nhau, sẽ mất đi tính trực quan, nên khi đó thay vì dùng tính kế cận, năm 1978 trong [2] đã đề xuất phương pháp đơn giản hàm dựa trên tính đối xứng của bảng. Khi dựa trên tính đối xứng của bảng thì phương pháp và algôrit đơn giản hàm luôn đúng cho các hàm logic với mọi số biến.

Dưới đây chúng ta sẽ xét phương pháp đơn giản hàm dùng bảng kế cận.

Để đơn giản hàm dùng bảng kế cận, cần thực hiện các bước sau:

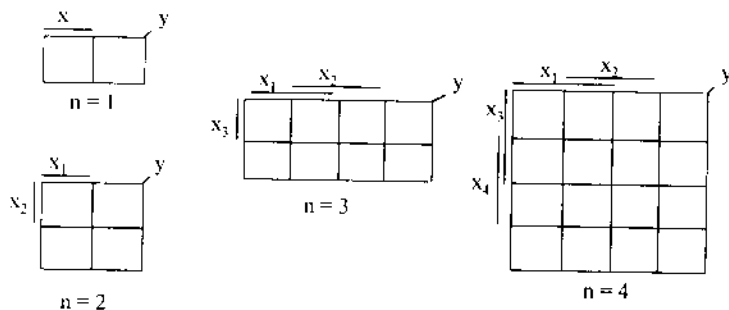
a) Biểu diễn hàm lên bảng kế cận

Để biểu diễn hàm có n biến, ta cần một bảng kế cận có $N = 2^n$ ô kế cận, mỗi ô đại diện cho một tập giá trị biến của hàm. Hình 3.1 là các bảng kế cận dùng biểu diễn các hàm 1, 2, 3 và 4 biến.

Như trong phần mô tả hàm logic đã trình bày, trong bảng kế cận, hai ô bất kỳ nằm cạnh nhau sẽ đại diện cho hai tập giá trị biến kế cận, tức giá trị của chúng chỉ khác nhau ở một bit; các bit còn lại hoàn toàn giống nhau.

Vì một hàm n biến có $N = 2^n$ tập giá trị biến và ứng với mỗi tập giá trị biến hàm sẽ có giá trị 0 hoặc 1 hoặc là giá trị bất định “-”. Giá trị của hàm được ghi trong ô có tập giá trị biến tương ứng. Trên hình 3.2 là bảng kế cận biểu diễn hàm logic 4 biến sau đây:

$$y(x_1, x_2, x_3, x_4) = \begin{cases} f'(0110, 1011, 0100, 1001, 0111, 1111) \\ f(1010, 0000, 1101) \end{cases}$$



Hình 3.1. Bảng kế cận cho hàm 1, 2, 3 và 4 biến

Do tính chất của các hàm không xác định, khi đơn giản hàm có quyền coi giá trị bất định là giá trị 1 hoặc 0, sao cho có lợi cho việc đơn giản hàm.

Thực chất của việc đơn giản hàm theo phương pháp Karnaugh là đối với các ô mà hàm có giá trị 1 và các ô mà hàm có giá trị bất định (nếu có), ta khoanh (tức thực hiện thuật toán cộng logic) các ô đó lại để tìm các implicăng đơn giản thực thụ theo nguyên tắc sau:

		x_1	x_2	y
x_3		—		1
x_4	1	1	1	
	1	—		
			1	—

Hình 3.2. Biểu diễn hàm lên bảng kế cận

1. Nếu ta khoanh 1 ô thì không khử được biến nào. Implicăng đơn giản thực thụ rút ra chính là tập giá trị biến của ô đó.

2. Khoanh hai ô kế cận sẽ rút ra được implicăng, trong đó khử được một biến. Biến bị khử là biến có giá trị thay đổi trong khoanh đã chọn. Implicăng là tích của các biến có giá trị không đổi, mà nếu có giá trị 0 thì viết là $\bar{x}$, nếu có giá trị là 1 thì viết là x không có dấu phủ định.

3. Một cách tổng quát, nếu ta khoanh $K = 2^k$ ô kế cận liên tiếp, thoả mãn điều kiện K ô này xếp thành hình chữ nhật, số ô của mỗi cạnh cũng thoả mãn điều kiện bằng 2^i và 2^j . Tức là:

$$K = 2^k = 2^i \cdot 2^j, \quad k, i, j = 0, 1, 2, \dots, n; \quad k = i + j$$

n là số nguyên dương.

Implicăng rút ra trong trường hợp này sẽ đơn giản được k biến. Các biến bị khử là các biến có giá trị thay đổi trong khoanh gộp. Các biến còn lại sẽ tham gia vào implicăng đơn giản, trong đó biến có giá trị 0 thì viết có dấu phủ định; biến có giá trị 1 thì viết không có dấu phủ định

b) Thực hiện algòrit đơn giản hàm sau

1. Xem các ô có giá trị 1 đã được khoanh hết chưa? Nếu đã hết thì thực hiện mục 4. Nếu chưa hết thì thực hiện mục 2.

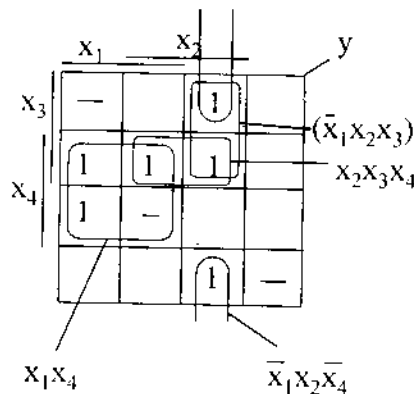
2. Khoanh một lượng K ô 1 và bất định, nếu có, *lớn nhất có thể được*, thoả mãn điều kiện $K = 2^k = 2^i \cdot 2^j$, $k, i, j = 0, 1, 2, \dots, n$ và rút ra implicăng đơn giản tương ứng.

3. Kiểm tra xem có khoanh gộp nào mà toàn bộ các ô của nó nằm gọn trong các khoanh gộp khác? Nếu có thì bỏ implicăng đơn giản tương ứng đã rút ra (Vì đó là implicăng đơn giản nhưng không phải là thực thụ) rồi quay lại 1. Nếu không có quay lại điểm 1.

Chú ý rằng, từ lần khoanh thứ hai trở đi phải khoanh cả các ô đã được khoanh trước đó nếu thoả mãn và có lợi cho việc đơn giản, nhưng ít nhất phải chứa một ô chưa được khoanh lần nào.

4. Kết thúc. Dạng tuyển chuẩn tối thiểu của hàm là tổng logic của tất cả implicăng đơn giản đã được rút ra.

Thí dụ 3.6: Tìm dạng tuyển chuẩn tối thiểu của hàm cho trên hình 3.2.



Hình 3.3. Hàm biểu diễn trên bảng kế cận

Lần lượt thực hiện algôrit trên đây, mỗi lần ta được một implicăng đơn giản theo thứ tự như sau (hình 3.3):

1. x_1x_4
2. $x_2x_3x_4$
3. $\bar{x}_1x_2x_3$
4. $\bar{x}_1x_2\bar{x}_4$

Thực hiện bước 3 của algôrit ta có thể bỏ một trong các implicăng đơn giản sau đây: $x_2x_3x_4$ và $\bar{x}_1x_2x_3$.

Nếu bỏ $\bar{x}_1x_2x_3$ ta được dạng tuyển chuẩn tối thiểu:

$$DTCTT = x_1x_4 + x_2x_3x_4 + \bar{x}_1x_2\bar{x}_4.$$

Nếu bỏ $x_2x_3x_4$ ta được dạng tuyển chuẩn tối thiểu sau:

$$DTCTT = x_1x_4 + \bar{x}_1x_2x_3 + \bar{x}_1x_2\bar{x}_4$$

3.3.2. Phương pháp dùng bảng đối xứng

Phương pháp dùng bảng đối xứng do chính tác giả tài liệu này đề xuất năm 1979, là sự phát triển của phương pháp dùng bảng kế cận bằng cách thay tính kế cận bằng tính đối xứng của bảng trong việc đơn giản hàm. Tính đối xứng hoàn toàn bao trùm lên tính kế cận, bất kỳ bảng kế cận nào cũng đều có tính đối xứng. Ngược lại, tính kế cận không hoàn toàn thay được tính đối xứng, vì khi số biến $n > 4$ thì không những các ô

nằm sát nhau là kế cận mà cả các ô nằm cách xa nhau cũng vẫn có thể là kế cận. Tính đối xứng luôn đồng nhất về tính chất với mọi bảng cho các giá trị n khác nhau.

Cách lập bảng đối xứng: Để đơn giản một hàm có n biến, cần:

1) Lập bảng có :

$2^{\frac{n}{2}}$ hàng và $2^{\frac{n}{2}}$ cột, khi n chẵn

$2^{\frac{n-1}{2}}$ hàng và $2^{\frac{n+1}{2}}$ cột, khi n lẻ.

2) Giá trị của các biến tại các hàng và các cột ghi sao cho các tập giá trị biến của các hàng là kế cận nhau và của các cột cũng là kế cận nhau, kể cả khi ta cuộn tròn bảng theo chiều dọc hay chiều ngang. Các trục đối xứng của các biến x_i , ký hiệu là các S_i , $i = 1, n$, là các trục chia các miền giá trị của chúng thành miền giá trị 0 và miền giá trị 1. Một bảng như vậy cho hàm $n = 5$ biến cho trên hình 3.4.

								$x_1 x_2$	
0	1	3	2	6	7	5	4	00	S_2
8	9	11	10	14	15	13	12	01	S_1
24	25	27	26	30	31	29	28	11	S_2
16	17	19	18	22	23	21	20	10	
x_3	0	0	0	0	1	1	1	1	
x_4	0	0	1	1	1	1	0	0	
x_5	0	1	1	0	0	1	1	0	
S_5				S_5				S_4	
S_3									

Hình 3.4. Bảng đối xứng dùng đơn giản hàm 5 biến

Các ô ứng với các tập giá trị biến viết dưới dạng số hệ 10 cho trong các ô của bảng. Trên thực tế giá trị này có thể đọc bằng cách đọc các giá trị của các biến ở hàng và cột của bảng và khi đó trong ô dùng để ghi giá trị tương ứng của hàm đối với tập giá trị biến tương ứng của ô đó. Bảng như trên có thể dùng để đơn giản bất kỳ hàm 5 biến nào.

3) Algorit đơn giản một hàm logic n biến dùng bảng đối xứng:

Hoàn toàn tương tự như đối với bảng kế cận. Sau khi đã biểu diễn hàm lên bảng đối xứng, áp dụng algorit đơn giản hàm bằng phương pháp Karnaugh, chỉ thay điều kiện *kế cận liên tiếp* bằng điều kiện *đối xứng liên tiếp qua các trục S_i* . Nói cách khác, ta khoanh một lượng các ô mà hàm nhận giá trị 1 (hoặc cả giá trị bất định) đối xứng liên tiếp qua các trục S_i và thỏa mãn điều kiện số lượng ô trong khoanh bằng K , mà:

Với phương pháp bảng đối xứng, việc đơn giản hàm nhiều biến thực hiện khá dễ dàng, nhanh chóng và trực quan.

3.4. PHƯƠNG PHÁP McCLUSKY

Đó là phương pháp do McClusky đề xuất năm 1956, tuy kém trực quan và nếu thực hiện bằng tay thì khá phức tạp, nhưng lại khắc phục được hiện tượng phải quay về dạng tuyến chuẩn chính tắc và cho phép dễ dàng thực hiện tự động hoá hơn.

3.4.1. Nội dung phương pháp McClusky

1) Viết các số hạng của dạng tuyến chuẩn ban đầu dưới dạng "Khối", nghĩa là: trong số hạng A nào đó, nếu các biến tham gia là biến không có phủ định thì chuyển thành giá trị 1, biến nào có phủ định thì chuyển thành giá trị 0; Biến nào không có mặt trong số hạng thì chuyển vị trí tương ứng của nó thành dấu bất định "-".

Thí dụ, nếu số hạng A của hàm ba biến có dạng: $A = x_1 \bar{x}_3$ thì khi chuyển thành dạng "Khối", có dạng: $A = 1 - 0$

2) Phân loại các "Khối" này thành lớp, tùy theo số lượng số 0 trong "Khối". Nghĩa là có lớp 0, không chứa số 1 nào; Lớp 1 chứa 1 số 1; Lớp 2 chứa 2 số 1, ... lớp n chứa cả n số 1. Rõ ràng có thể một số lớp không xuất hiện đối với một số hàm.

3) So sánh các lớp liên kế nhau để rút ra các "Khối" đơn giản hơn. Rõ ràng, chỉ có thể rút ra "Khối" đơn giản hơn khi ta so sánh hai "Khối" kế cận nhau, nói cách khác chúng chỉ khác nhau một bit.

Trong trường hợp rút ra được "Khối" đơn giản hơn, ta đánh dấu các "Khối" thành phần đã tham gia để tạo được "Khối" mới bằng một dấu "x" ở bên cạnh.

4) Xếp lại các lớp sau khi đã so sánh hết các lớp kế nhau và lại tiếp tục so sánh như trên. Chú ý rằng, khi đã xuất hiện các dấu gạch bất định "-" thì chỉ có thể so sánh hai "Khối" có cùng số lượng và vị trí của các dấu bất định.

Việc so sánh và xếp lại lớp cứ tiếp tục cho tới khi không còn so sánh được nữa thì thôi. Khi đó tổng của tất cả các "Khối" không bị đánh dấu chính là dạng tuyến chuẩn ngắn nhất của hàm.

5) Để có dạng tuyến chuẩn tối thiểu cần áp dụng phương pháp Kvaina.

3.4.2. Thí dụ

Thí dụ 3.8: Đơn giản hàm sau đây dùng phương pháp McClusky:

$$y(x_1, x_2, x_3, x_4) = f'(0, 1, 2, 3, 4, 6, 7, 8, 9, 11, 15)$$

Kết hợp điểm 1) và 2) của algôrit ta được các nhóm sau:

Nhóm 0 gồm: 0000^x

Nhóm 1: 0001^x, 0010^x, 0100^x, 1000^x

Nhóm 2: 0011^x, 0110^x, 1001^x

Nhóm 3: 0111^x, 1011^x

Nhóm 4: 1111^x

- Thực hiện bước 3) so sánh các nhóm 0 và 1, 1 và 2; 2 và 3; 3 và 4 để tìm các "Khối" đơn giản hơn và sau khi xếp lại các nhóm, ta được:

Nhóm 0: 000-^x; 00-0^x; 0-00^x; -000^x.

Nhóm 1: 00-1^x; -001^x, 001-^x; 0-10^x; 10-0^x; 100-^x.

Nhóm 2: 0-11^x; -011^x; 011-^x; 10-1^x.

Nhóm 3: -111^x; 1-11^x.

- Thực hiện thủ tục so sánh lần thứ ba và sau khi xếp lại nhóm, ta được:

Nhóm 0: 00--; -00-; 0--0

Nhóm 1: -0-1; 0-1-;

Nhóm 2: --11

Đến đây ta không thể tiếp tục so sánh nữa. Quá trình so sánh kết thúc. Các implicãng đơn giản (hay các "Khối" đơn giản) không bị đánh dấu chính là các implicãng đơn giản thực thụ. Nói cách khác, dạng tuyến chuẩn ngắn nhất của hàm này như sau:

$$y = F(00--, -00-, 0--0, -0-1, 0-1-, --11)$$

Áp dụng bước 5) ta có bảng Kvaina - Bảng 3.2.

Bảng 3.2. Bảng Kvaina tìm các implicãng đơn giản thực thụ

	0000	0001	0010	0011	0100	0110	0111	1000	1001	1011	1111
00--	x	x	x	x							
0--0	x		x		x	x					
-00-	x	x						x	x		
-0-1		x		x					x	x	
0-1-			x	x		x	x				
--11				x			x			x	x

Áp dụng algôrit chọn, lần lượt ta được các implicãng :

1) -00-; 2) 0--0; 3) --11.

Sau khi đã chọn ba implicãng đơn giản này đủ để phủ tất cả các cột của bảng. Vậy dạng tuyến chuẩn tối thiểu của hàm như sau:

$$y = \bar{x}_2 \bar{x}_3 + \bar{x}_1 \bar{x}_4 + x_3 x_4$$

Ta nhận thấy rằng, phương pháp đơn giản hàm của McClusky không những giảm bớt rất nhiều các bước tìm kiếm so sánh giữa các tổ hợp giá trị biến mà còn cho phép hàm cho trước không nhất thiết phải quay trở lại dạng tuyến chuẩn chính tắc. Phương pháp này dễ dàng thực hiện bằng máy tính điện tử.

3.5. PHƯƠNG PHÁP ROT

3.5.1. Khái niệm và cơ sở toán học của phương pháp

Phương pháp này cho phép toán hoá các bước tìm các implicăng và do đó có triển vọng tự động hoá quá trình đơn giản hàm dùng máy tính.

Để áp dụng phương pháp ROT các tổ hợp biến mà hàm nhận giá trị 1 được biểu diễn dưới dạng phức hợp khối. Nghĩa là, nếu trong tổ hợp biến thay biến x_i bằng giá trị "1", còn $\bar{x}_i$ bằng giá trị "0", không có mặt thay bằng dấu "x".

Thí dụ, với $n = 4$:

Nếu là $x_1x_2\bar{x}_3$ thì phức hợp khối là: 110x;

Nếu là $x_1x_3x_4$ thì phức hợp khối là: 1x11.

Khi đó cho một hàm là cho tập hợp các phức hợp khối mà hàm nhận giá trị 1.

Thí dụ cho một hàm như sau:

$$y = \begin{vmatrix} 110x \\ 0x11 \\ 10xx \end{vmatrix}$$

Tập hợp các phức hợp khối không chứa dấu "x" gọi là khối 0, ký hiệu là K^0 ; Tập hợp các implicăng đơn giản thuộc K^0 , ký hiệu là Z^0 .

Một cách tương tự, K^1 là tập hợp các phức hợp khối chứa một dấu "x" và Z^1 là tập hợp các implicăng đơn giản thuộc K^1 .

Một cách tổng quát, ta gọi K^i là tập hợp phức hợp khối và Z^i là tập hợp các implicăng đơn giản thuộc K^i .

Toàn bộ các phức hợp khối mà hàm nhận giá trị 1 bằng K:

$$K = K^0 + K^1 + \dots + K^n$$

và không gian các implicăng đơn giản của một hàm là Z:

$$Z = Z^0 + Z^1 + \dots + Z^n.$$

Khi đó bài toán đơn giản hàm đặt ra một cách tổng quát như sau:

Cho một hàm xác định đầy đủ, nhận giá trị 1 đối với tập hợp các tổ hợp giá trị biến L, cần tìm không gian các implicăng đơn giản Z của L, sau đó tìm một phủ tối thiểu E của L mà E thuộc Z. Tức là cần có:

$$L \subseteq E_{\min} \subseteq Z$$

Thí dụ 3.9:

Ta có $L = \{0001, 0010, 0000, 0011, 0100, 0110, 0111, 1000, 1001, 1011, 1111\}$

$$Z = \{00xx, 0xx0, x00x, x0x1, 0x1x, xx11\}$$

$$E = \{0xx0, x00x, xx11\}$$

Như vậy, để tìm hàm đơn giản nhất cần:

1) Tìm không gian các implicăng đơn giản của L :

$$Z = Z^0 + Z^1 + \dots + Z^n$$

2) Tìm không gian các đầu mút E (tức tìm không gian các implicăng đơn giản thực sự).

Trước khi thực hiện các bước trên đây, ta tìm hiểu một số thuật toán sau đây:

a) Phép nhân hai khối, ký hiệu là “*”

Nội dung phép nhân này là thực hiện nhân theo bit của hai phức hợp khối đó theo quy luật cho trên bảng 3.3

Nếu khối C^r và C^s có dạng:

$$C^r = (a_1 a_2 \dots a_n)$$

$$C^s = (b_1 b_2 \dots b_n)$$

thì kết quả phép nhân “*” sẽ là:

$$C^r * C^s = \begin{cases} \emptyset, \text{ nếu } a_i * b_i = y \text{ và có 2 giá} \\ \text{trị } y \text{ trở lên.} \\ m(a_1 * b_1) m(a_2 * b_2) \dots m(a_n * b_n) \end{cases}$$

với:

$$m(a_i * b_i) = \begin{cases} m(0) = 0 \\ m(1) = 1 \\ m(y) = m(x) = x \end{cases}$$

Bảng 3.3. Quy luật nhân hai phức hợp khối theo bit

*	0	1	x
0	1	y	0
1	y	1	1
x	0	1	x

Thí dụ 3.10: $000 * 101 = y0y = \emptyset$, vì có hai giá trị y ;

$000 * 001 = 00y = 00x$, vì chỉ có một giá trị y ;

$0xx * 1x0 = yx0 = xx0$

Tính chất của phép nhân hai phức hợp khối: Có tính giao hoán nhưng không có tính kết hợp và nếu $C^r \subseteq C^s$ thì $C^r * C^s = C^r$. Vì nếu giả sử $C^r = (x10)$ và $C^s = (xx0)$ thì $C^r * C^s = x10 * xx0 = x10 = C^r$.

Phép tính “*” cho phép ta xác định xem các khối trong tập hợp các phức hợp khối có khối nào là implicăng đơn giản không. Nói cách khác, xác định xem khối đó có thể kết hợp một khối nào đó trong phức hợp khối để khử đi một biến nào đó hay không. Nếu không kết hợp được thì nó là một implicăng đơn giản.

Thực vậy, nếu nhân phức hợp khối C^r với một tập hợp các phức hợp khối L , giả thiết C^r chứa một dấu “x” thì nếu $C^r * L$ sinh ra ít nhất một khối C^i nào đó có chứa hai dấu “x” thì có nghĩa rằng C^r và tập hợp L có thể kết hợp được với nhau để khử biến.

Thí dụ 3.11:

Trường hợp 1: Cho $C^r = (1x0)$; $L = \{1x0, x00, 01x, x10, 0x0\}$

thì $C^r * L = \{1x0, 100, 110, x10, xx0\}$

Như vậy, tích $1x0 * 0x0$ sinh ra khối 2, tức là $xx0$. Nói cách khác, hai khối đã cho có thể kết hợp với nhau để khử biến. Chúng là các tập giá trị biến kế cận nhau.

Trường hợp 2: Cho $C^r = (01x)$

$$C^r * L = \{x10, 0x0, 01x, 010, 010\}$$

Kết quả không sinh ra bất kỳ khối 2 nào. Do đó khối $C^r = 01x$ là implicăng đơn giản của L .

Để tìm các implicăng đơn giản, ta có bổ đề sau:

Bổ đề

Không gian các implicăng đơn giản khối i (không gian các implicăng đơn giản chứa i dấu “x”) là tập hợp các khối i , tức $C^i \in L_i$, sao cho $C^i * L_i$ không chứa một khối $(i + 1)$ nào. Tức là:

$$Z^i = \{C^i \mid C^i * L_i \not\supset \text{một khối } (i + 1) \text{ bất kỳ nào}\}$$

trong đó: Z^i - không gian các implicăng đơn giản bậc i (có tối đa i dấu x trong một khối);

C^i - phức hợp khối bậc i ;

L_i - không gian các implicăng (hoặc tập hợp các phức hợp khối nói chung) có bậc từ i trở lên.

Algôrit để tìm không gian các implicăng đơn giản của phức hợp khối L cho trước:

1. Xác định bậc thấp nhất của các phức hợp khối thuộc L . Giả sử bậc thấp nhất là i , $i = 0, 1, 2, \dots, n - 1$, trong đó n là số biến logic của hàm.

2. Xác định không gian các implicăng đơn giản khối i theo công thức:

$$Z^i = \{C^i \mid C^i * L_i \not\supset \text{một khối } (i + 1) \text{ bất kỳ nào}\}$$

3. Lập không gian các tập hợp khối mới L_{i+1} không chứa phức hợp khối bậc i :

$$\bar{L}_{i+1} = L_i \cup L_i * L_i$$

$$L_{i+1} = \bar{L}_{i+1} \setminus L_{i+1}^i \setminus \bar{L}_{i+1}^*$$

trong đó: L_{i+1}^i là không gian các phức hợp khối bậc i thuộc $\bar{L}_{i+1}$;

$\bar{L}_{i+1}^*$ là các phức hợp khối thuộc $\bar{L}_{i+1}$, nhưng bị các phức hợp khối khác phủ.

4. Xét xem $L_{i+1} = \emptyset$ hay không? Nếu $L_{i+1} = \emptyset$ thì chuyển sang bước 5); Nếu $L_{i+1} \neq \emptyset$ thì quay lại bước 2.

5. Không gian các implicăng đơn giản của tập hợp các phức hợp khối L là Z :

$$Z = \sum_i Z^i$$

b) Phép giao toạ độ, ký hiệu là " $\cap$ "

Phép giao toạ độ " $\cap$ " thực hiện giữa hai phức hợp khối theo bit. Luật giao toạ độ cho trên bảng 3.4.

Giao toạ độ của hai phức hợp khối:

$$C^r = (a_1 a_2 \dots a_n)$$

$$C^s = (b_1 b_2 \dots b_n)$$

được thể hiện như sau:

$$C^r \cap C^s = \begin{cases} \emptyset, \text{ nếu } a_i \cap b_i = \emptyset \text{ với một } i \\ \text{ bất kỳ;} \\ (a_1 \cap b_1 \dots a_n \cap b_n), \text{ trong} \\ \text{ mọi trường hợp khác.} \end{cases}$$

Phép giao toạ độ có tính giao hoán và tính kết hợp. Kết quả trống $\emptyset$ của phép giao toạ độ chứng tỏ C^r và C^s không phải là phủ của nhau.

Bảng 3.4. Bảng luật giao toạ độ của hai phức hợp khối

$\cap$	0	1	x
0	0	$\emptyset$	0
1	$\emptyset$	1	1
x	0	1	x

Trong các trường hợp khác chứng tỏ chúng là phủ của nhau: Hoặc C^r là phủ của C^s hoặc ngược lại. Trong trường hợp này kết quả sẽ là:

$$C^r, \text{ nếu } C^s \supseteq C^r;$$

$$C^s, \text{ nếu } C^s \subseteq C^r;$$

$$C^i, \text{ nếu } C^r \text{ và } C^s \text{ phủ từng phần của nhau, không khối nào phủ hoàn toàn khối nào.}$$

Nội dung và tính chất của phép giao toạ độ cho phép tìm không gian các implicăng đơn giản thực thụ E của tập hợp các phức hợp khối L từ không gian của các implicăng đơn giản Z của L , tức là:

$$L \subseteq Z \subseteq E$$

E chính là tập hợp các khối của dạng tuyến chuẩn tối thiểu của hàm đã cho trên L .

Định lý 1. Implicăng đơn giản $e \in Z$ là implicăng đơn giản thực thụ của L nếu nó thoả mãn hai điều kiện sau đây:

$$1. K(e \cap L) \neq \emptyset$$

$$2. K(e \cap L) \neq K(e \cap U'(e, Z) \cap L)$$

$$\text{Trong đó: } + U'(e, Z) = U(e, Z) \setminus e$$

$$+ U(e, Z) = \{C \mid C \in Z \text{ và } C \cap e \neq \emptyset\}$$

Điều kiện thứ nhất chứng tỏ e là implicăng đơn giản của L (là phủ ít nhất của một phức hợp khối thuộc L). Kết quả $K(e \cap L)$ là tập hợp các phức hợp khối thuộc L mà e phủ.

Điều kiện thứ hai thể hiện rằng có những phức hợp khối e phủ, nhưng các implicăng đơn giản khác không phủ. Do đó trong trường hợp này e phải là implicăng đơn giản thực thụ.

Thí dụ 3.12

Cho $L = \{0010, 0011, 0100, 0110\}$ và $Z = \{0xx0, x0x1, 0x1x\}$

Lấy $e = 0xx0$, khi đó:

$$K(e \cap L) = \{0010, 0100, 0110\}$$

Ta tìm $U(e, Z) = \{0xx0, 0x1x\}$

$$U'(e, Z) = U(e, Z) \setminus e = \{0xx0, 0x1x\} \setminus 0xx0 = 0x1x$$

$$K(e \cap U'(e, Z) \cap L) = \{0010, 0110\} \neq K(e \cap L)$$

Rõ ràng phức hợp khối 0100 thuộc L chỉ do e phủ. Khối e là khối $0xx0$, nó là một implicăng đơn giản thực thụ.

c) Phép trừ toạ độ, ký hiệu là “#”

Nếu tập hợp các phức hợp khối L_r , trừ đi tập hợp các phức hợp khối E_{r-1} sẽ cho ta các tập hợp khối thuộc L_{r-1} mà E_{r-1} không phủ. Phép trừ toạ độ này được ký hiệu như sau:

$$L_r = L_{r-1} \# E_{r-1}$$

Nếu $L_r = \emptyset$ có nghĩa là E_{r-1} phủ toàn bộ các tập hợp khối của L_{r-1} .

Thuật toán trừ toạ độ “#” thực hiện theo bảng 3.5.

Phép trừ toạ độ “#” là phép tính không có tính giao hoán và tính kết hợp, song có tính phân bố.

Nội dung của phép # có thể mô tả như sau:

Giả sử có hai khối C^r và C^s mà:

$$C^r = (a_1 a_2 \dots a_n) \text{ và } C^s = (b_1 b_2 \dots b_n)$$

thì:

$$C^r \# C^s = \begin{cases} C^r, & \text{nếu ít nhất có một } y \text{ nào đó;} \\ \emptyset, & \text{nếu } a_i \# b_i = z \text{ với mọi } i; \\ U_i (a_1 a_2 \dots a_n), & \text{nếu } a_i \# b_i = \alpha_i, \\ & \text{mà } \alpha_i = \{0, 1, z\}. \end{cases}$$

Bảng 3.5. Phép trừ toạ độ của hai phức hợp khối

#	0	1	x
0	z	y	z
1	y	z	z
x	1	0	z

Thí dụ 3.13

$$x10 \# xx1 = z0y = x10$$

$$x0x \# 1xx = 0zz = 00x$$

$$xxx \# 11x = 00z = 0xx + x0x, \text{ vì } \alpha_1 = \alpha_2 = 0$$

$$11x \# xxx = zzz = \emptyset.$$

Để nắm vững hơn phương pháp ROT, ta thực hiện thí dụ đơn giản hàm sau.

Thí dụ 3.14: Đơn giản hàm sau đây:

$$y = f(0011, 0100, 0101, 0111, 1001, 1011)$$

Ta có tập hợp phức hợp khối ban đầu L_0 như sau:

$$L_0 = \begin{array}{|c|} \hline 0011 \\ \hline 0100 \\ \hline 0101 \\ \hline 0111 \\ \hline 1001 \\ \hline 1011 \\ \hline \end{array}$$

Để tìm Z_0 ta tính:

$$\begin{aligned} L_0 * L_0 &= \begin{array}{|c|} \hline 0011 \\ \hline 0100 \\ \hline 0101 \\ \hline 0111 \\ \hline 1001 \\ \hline 1011 \\ \hline \end{array} * \begin{array}{|c|} \hline 0011 \\ \hline 0100 \\ \hline 0101 \\ \hline 0111 \\ \hline 1001 \\ \hline 1011 \\ \hline \end{array} = \\ &= (0011, \emptyset, \emptyset, 0x11, \emptyset, x011, 0100, 010x, 010x, 0101, 01x1, \\ &\quad 0x11, 01x1, 0111, 1001, 10x1, x011, 10x1, 1011). \end{aligned}$$

Không có khối nào trong 6 khối của L_0 mà không sinh ra khối 1. Do đó $Z^0 = \emptyset$.

Bây giờ xác định L_1 :

$$L_1 = \begin{array}{|c|} \hline 0x11 \\ \hline x011 \\ \hline 010x \\ \hline 01x1 \\ \hline 10x1 \\ \hline \end{array} \neq \emptyset$$

Do đó cần tìm Z^1 . Muốn vậy trước hết tính:

$$\begin{aligned} L_1 * L_1 &= (0x11, 0011, 01x1, 0111; 0011, x011, 0x11, 1011; 01x1, 010x, 0101; \\ &\quad 0111, 0x11, 0101, 01x1; x011, 1011, 10x1). \end{aligned}$$

Vì tất cả các khối của L_1 trong phép nhân $L_1 * L_1$ không sinh ra bất kỳ khối 2 nào.

Do đó $Z^1 = L_1$ và:

$$Z = Z^1 = \begin{array}{|c|} \hline 0x11 \\ \hline x011 \\ \hline 010x \\ \hline 01x1 \\ \hline 10x1 \\ \hline \end{array} \begin{array}{l} a \\ b \\ c \\ d \\ e \end{array}$$

Từ hai tập hợp các phức hợp khối L_0 và Z , ta tìm các implicăng đơn giản thực thụ E:

1. Kiểm tra các điều kiện của định lý 1 với khối a của Z:

$$K(a \cap L_0) = 0x11 \cap 0x11 \cap \begin{array}{|c|} \hline 0011 \\ 0100 \\ 0101 \\ 0111 \\ 1001 \\ 1011 \\ \hline \end{array} = (0111, 0011) \neq \emptyset$$

$$U'(a, z) = (b, d) = (x011, 01x1).$$

$$K(a \cap L_0 \cap U'(a, z)) = (0111, 0011) = K(a \cap L_0)$$

Do đó a không phải là implicăng đơn giản thực thụ.

2. Kiểm tra khối b của Z:

$$K(b \cap L_0) = x011 \cap L_0 = (0011, 1011) \neq \emptyset$$

$$U'(b, z) = (a, c) = (0x11, 10x1)$$

$$K(b \cap L_0 \cap U'(b, z)) = (0011, 1011) = K(b \cap L_0)$$

Do đó b không phải là một implicăng đơn giản thực thụ.

3. Kiểm tra khối c của Z:

$$K(c \cap L_0) = 010x \cap L_0 = (0100, 0101) \neq \emptyset$$

$$U'(c, z) = (d) = (01x1)$$

$$K(c \cap L_0 \cap U'(c, z)) = (0101, 0111) \neq K(c \cap L_0).$$

Do đó c là một implicăng đơn giản thực thụ.

4. Kiểm tra khối d của Z:

$$K(d \cap L_0) = (0101, 0111) \neq \emptyset$$

$$U'(d, z) = (a, c) = (0x11, 010x)$$

$$K(d \cap L_0 \cap U'(d, z)) = (0101, 0111) = K(d \cap L_0)$$

Do đó d không phải là một implicăng đơn giản thực thụ.

5. Kiểm tra khối e của Z:

$$K(e \cap L_0) = 10x1 \cap L_0 = (1001, 0111) \neq \emptyset$$

$$U'(e, z) = (x011)$$

$$K(e \cap L_0 \cap U'(e, z)) = (1011) \neq K(e \cap L_0)$$

Do đó e là một implicăng đơn giản thực thụ.

Cuối cùng ta có: $E_0 = c + e$.

Tìm L_1 :

$$L_1 = L_0 \# E_0 = \left\{ \begin{array}{l} 0011 \\ 0100 \\ 0101 \\ 0111 \\ 1001 \\ 1011 \end{array} \right\} \begin{array}{l} a \\ b \\ c \\ d \\ e \\ f \end{array} \# \left\{ \begin{array}{l} 010x \\ 10x1 \end{array} \right\} = (0011, 0111) \neq \emptyset$$

Do đó cần tìm Z^2 :

$$\hat{Z}^2 = Z^1 \setminus E_0 = (0x11, x011, 01x1)$$

Nhưng vì :

$$(x011 \cap L_1) = (0011) \subset (0x11 \cap L_1) = (0011, 0111) \supset (01x1 \cap L_1) = (0111)$$

nên: $Z^2 = a = 0x11$ và là khôi duy nhất phủ L_2 .

Từ đó ta có :

$$E_1 = a; E = E_0 + E_1 = a + c + e.$$

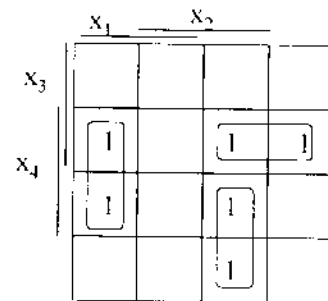
Như vậy, phủ tối thiểu của hàm đã cho dưới dạng L_0 là E , gồm ba phức hợp khối a, c và e .

Chúng ta có thể kiểm tra kết quả trên đây bằng cách áp dụng phương pháp Karnaugh để đơn giản hàm đã cho như trên hình 3.6.

Kết quả đơn giản ta được:

$$DTCTT = \bar{x}_1 x_3 x_4 + \bar{x}_1 x_2 \bar{x}_3 + x_1 \bar{x}_2 x_4$$

Kết quả này trùng khớp với tập E của hàm mà ta đã thu được bằng cách áp dụng các thuật toán đã chỉ ra trong phương pháp ROT.



Hình 3.6. Mô tả hàm trên bảng Karnaugh

3.5.2. Algôrit đơn giản hàm

Algôrit tổng quát tìm phủ tối thiểu C_{min} từ tập hợp không gian các implicăng đơn giản Z' của phức hợp khối L_r như sau:

1) Kiểm tra mọi phức hợp khối $z_i \in Z'$ với mọi i theo định lý 1 về tìm implicăng đơn giản thực thụ. Và lấy:

$$E_r = \bigcup_{i \in T} z_i$$

trong đó T là tập hợp các khối thoả mãn điều kiện của định lý 1 về một implicăng đơn giản thực thụ.

2) Dùng phép # để tính L_{r+1} như sau:

$$L_{r+1} = L_r \# E_r$$

Nếu $L_{r+1} = \emptyset$ thì có nghĩa là E_r phủ hết L_r . Khi đó ta có phủ tối thiểu của hàm đã cho $C_{\min}$:

$$C_{\min} = \bigcup_{i \in T} E_i$$

Sau đó chuyển sang bước 4). Nếu $L_{r+1} \neq \emptyset$ thì thực hiện bước 3).

3) Tính Z^{r+1} : Bằng cách loại bỏ các khối trong Z^r đã nằm trong E_r để có:

$$\hat{Z}^{r+1} = Z^r \setminus E_r$$

Sau đó rút gọn $\hat{Z}^{r+1}$ bằng cách loại đi những khối c thoả mãn điều kiện sau:

$$c < z \in \hat{Z}^{r+1}$$

trong đó c là một khối nào đó thuộc $\hat{Z}^{r+1}$.

Khái niệm $c < z$ được hiểu như sau:

Nếu bậc của c không thấp hơn bậc của z (số biến tham gia trong c không ít hơn số biến tham gia trong z) và:

$$c \cap L_r \subseteq z \cap L_r$$

thì $c < z$. Khi đó:

$$Z^{r+1} = \hat{Z}^{r+1} \setminus c$$

Sau đó quay lại bước 1).

4) Kết thúc.

Trên đây chúng ta đã xét một số phương pháp đơn giản hàm logic. Các phương pháp này được sắp xếp trong trình bày theo mức độ khó dần, nhưng cũng là hoàn thiện dần trên quan điểm sử dụng máy tính để tự động hoá khâu đơn giản hàm, đặc biệt đối với các hàm có số lượng biến lớn và rất lớn.

3.6. ĐƠN GIẢN HỆ HÀM

Đơn giản hệ các hàm logic có ý nghĩa thực tiễn lớn vì hầu hết các thiết bị điện tử đều là loại có nhiều hàm. Nếu mỗi đầu ra (một hàm) thực hiện độc lập với các hàm khác thì chắc chắn số linh kiện sẽ tăng lên. Trên thực tế, một số chức năng có thể được thực hiện chung bằng một hoặc một số phần tử logic cho cả hệ hoặc cho một số hàm trong hệ. Nhờ vậy mạch sẽ đơn giản đi rất nhiều. Do đó bài toán đơn giản hệ hàm là làm sao kết hợp những phần có thể thực hiện chung để tiết kiệm linh kiện, làm thiết bị gọn nhẹ, tăng độ tin cậy và dùng ít năng lượng hơn. Từ đó nhiều khâu kỹ thuật khác cũng được giảm nhẹ và thiết bị dễ dàng cấy vào các máy móc khác hơn.

Có nhiều phương pháp đơn giản hệ hàm. Song cho tới nay giải pháp cho việc tối ưu hoá việc đơn giản hệ hàm cũng còn nhiều khó khăn và còn cần tiếp tục nghiên cứu. Trong phần này sẽ trình bày một vài phương pháp đơn giản kết hợp hệ gồm m hàm với n biến.

3.6.1. Phương pháp đơn giản riêng rẽ từng hàm

Giả sử cần thực hiện hệ các hàm $(f_1, f_2, \dots, f_m)$, ta dùng các phương pháp đơn giản một hàm logic đã trình bày trước đây để đơn giản từng hàm. Trên cơ sở kết quả thu được, lựa chọn những thành phần chung cho tất cả các hàm trong hệ hoặc chung cho một số hàm. Các phần chung này được thực hiện bằng một phần tử hay một cụm các phần tử logic.

Thí dụ 3.15

Đơn giản hệ hai hàm sau:

$$y_1 = x_1x_2x_3 + x_1x_2\bar{x}_3 + x_1\bar{x}_2\bar{x}_3$$

$$y_2 = x_1\bar{x}_2x_3 + x_1x_2 + \bar{x}_1\bar{x}_2x_3$$

Nếu đơn giản bằng một trong các phương pháp đã trình bày, ta được:

$$y_1 = x_1x_2 + x_1\bar{x}_3; y_2 = x_1x_2 + \bar{x}_2x_3$$

Như vậy, trong tất cả các implicăng đơn giản thực thụ của hai hàm, có thành phần x_1x_2 là chung.

Có thể khẳng định như sau:

Các implicăng đơn giản của tích các hàm $y_1, y_2, \dots, y_m$, $m \geq 2$ sẽ là implicăng đơn giản chung cho m hàm đó.

Khẳng định này cho phép tìm các implicăng chung cho cả hệ hàm hay chung cho một số hàm trong hệ bằng cách tìm các implicăng đơn giản thực thụ của tích của chúng.

Thí dụ 3.16: Tìm các implicăng chung cho hai hàm sau đây:

$$y_1 = x_1x_2x_3 + x_1x_2\bar{x}_3 + x_1\bar{x}_2\bar{x}_3$$

$$y_2 = x_1x_2 + \bar{x}_2x_3$$

Tìm tích của hai hàm:

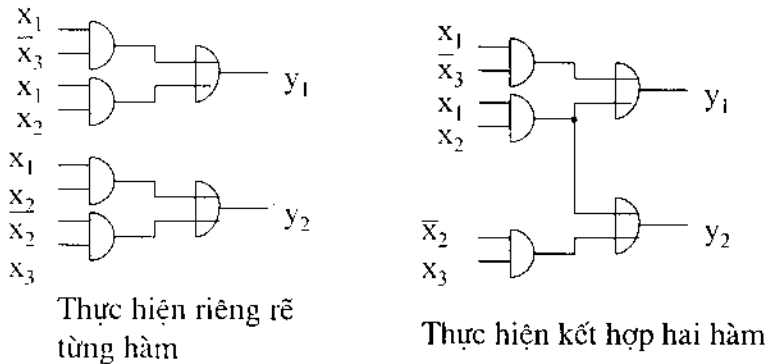
$$y_1 \cdot y_2 = (x_1x_2x_3 + x_1x_2\bar{x}_3 + x_1\bar{x}_2\bar{x}_3)(x_1x_2 + \bar{x}_2x_3) = x_1x_2x_3 + x_1x_2\bar{x}_3$$

Do đó các implicăng chung cho hai hàm tìm được từ các số hạng chung của chúng:

$$x_1x_2x_3 + x_1x_2\bar{x}_3 = x_1x_2$$

Rõ ràng, nếu ta đơn giản riêng từng hàm rồi tách các implicăng đơn giản chung của chúng ta cũng được x_1x_2 như thí dụ 3.15 đã thực hiện.

Khi thực hiện bằng các phần tử logic, phần tử thực hiện tích x_1x_2 là chung cho cả hai hàm (hình 3.7).



Hình 3.7. Các phương án thực hiện hàm

Cần lưu ý rằng, khi tìm implicăng đơn giản của tích các hàm, có thể có các implicăng đơn giản không phải là implicăng đơn giản của từng hàm đơn lẻ thì không có lợi gì khi dùng các implicăng đơn giản chung đó.

Thí dụ 3.17. Tìm implicăng đơn giản của hai hàm:

$$y_1 = x_1x_2 + x_1\bar{x}_2x_3$$

$$y_2 = x_1\bar{x}_3 + \bar{x}_2\bar{x}_3$$

Lấy tích của hai hàm này ta được:

$$y_1.y_2 = (x_1x_2 + x_1\bar{x}_2x_3)(x_1\bar{x}_3 + \bar{x}_2\bar{x}_3) = x_1x_2\bar{x}_3$$

Implicăng đơn giản này của tích hai hàm không phải là implicăng đơn giản của mỗi hàm, vì khi đơn giản từng hàm ta được:

$$y_1 = x_1x_2 + x_1x_3$$

$$y_2 = x_1\bar{x}_3 + \bar{x}_2x_3$$

Vì vậy khi thực hiện không được dùng $x_1x_2\bar{x}_3$.

Cần lưu ý rằng, với phương pháp này mạch thực hiện trong một số trường hợp còn chưa đạt được dạng tối thiểu.

3.6.2. Đơn giản kết hợp hệ hàm

Phương pháp đơn giản kết hợp hệ hàm hoàn toàn có thể áp dụng phương pháp đơn giản một hàm như phương pháp McClusky.

Để hiểu phương pháp này chúng ta xét thí dụ sau đây:

Thí dụ 3.18: Đơn giản hệ ba hàm sau đây:

$$y_1 = f_1(0, 2, 7, 10); \quad f_1(5, 12)$$

$$y_2 = f_2(2, 11, 15); \quad f_2(3, 7, 8, 10)$$

$$y_3 = f_3(2, 7, 8, 11, 15); \quad f_3(0, 9, 13)$$

Cách làm như sau:

- Liệt kê tất cả các tập giá trị biến của các hàm và bên phải mỗi tập giá trị biến ta ghi thêm ba cột, trong mỗi cột ghi giá trị lôgic ứng với ba hàm: Nếu tập giá trị biến đang xét thuộc hàm f_i , $i = \overline{1, 3}$ thì ghi số 1; nếu không thuộc hàm đó thì ghi số 0, như hình 3.8a.

$x_1 x_2 x_3 x_4$	$y_1 y_2 y_3$	$x_1 x_2 x_3 x_4$	$y_1 y_2 y_3$	$x_1 x_2 x_3 x_4$	$y_1 y_2 y_3$
0000	101	<u>0000</u> v	<u>101</u>	00-0	101
0010	111	0010	111	<u>-000</u>	001
0011	010	<u>1000</u>	<u>011</u>	001-	010
0101	100	0011v	010	-010	110
0111	111	0101v	100	100-	001
1000	011	1001v	001	10-0	010
1001	001	1010v	110	<u>1-00</u>	000
1010	110	<u>1100</u> v	<u>100</u>	0-11	010
1011	011	0111v	111	01-1	100
1100	100	1011v	011	-011	010
1101	001	<u>1101</u> v	<u>001</u>	10-1	001
1111	011	1111v	011	101-	010
				-101	000
				1-01	001
				<u>110-</u>	000
				-111	011
				1-11	011
				11-1	001
a)		b) Phân nhóm		c) Kết quả so sánh bước 1	

Hình 3.8. Quá trình đơn giản hệ 3 hàm

$x_1x_2x_3x_4$	$y_1y_2y_3$	$x_1x_2x_3x_4$	$y_1y_2y_3$	$x_1x_2x_3x_4$	$y_1y_2y_3$
-000	001	-0-0	000v	0010	111
-010	110	-01-	010	1000	011
-011	010	--11	010	1100	100
-111	011	--11	010	0111	111
0-11	010	1--1	001	-000	001
1-01	001	-0-0	000v	-010	110
1-11	011	10--	000v	-111	011
00-0	101	-1-1	000v	1-11	011
10-0	010	1--1	001	00-0	101
01-1	100	-01-	010	10-0	010
10-1	001	10--	000v	01-1	100
11-1	001			100-	001
001-	010			-01-	010
100-	001			--11	010
101-	010			1--1	001
d) Phân nhóm và kết quả so sánh bước 2				e) Tập hợp các implicăng đơn giản của hệ	

Hình 3.8. Quá trình đơn giản hệ 3 hàm (tiếp theo)

Thí dụ tập giá trị biến 1000 có các giá trị tương ứng ở ba cột $y_1y_2y_3$ là 011, vì đối với tập giá trị biến này hàm y_2 và y_3 có giá trị 1, còn y_1 có giá trị 0. Trong trường hợp hàm nhận giá trị bất định đối với một tập giá trị biến nào đó ta cũng ghi giá trị tương ứng là 1.

Dùng phương pháp đơn giản McClusky, ta phân nhóm các tập giá trị biến theo các lớp - hình 3.8b. Sau đó so sánh các nhóm kế nhau. Mỗi lần so sánh một cặp tập giá trị biến kế cận, ta sẽ đơn giản được một biến. Implicăng thu được này sẽ thuộc các hàm y_i , $i = 1, 3$ mà có hai tập giá trị biến vừa được so sánh. Nói cách khác, ba cột bên phải của implicăng đơn giản thu được sẽ là tích logic của ba cột bên phải của hai tập giá trị biến so sánh.

Thí dụ, tập -011 có ba cột bên phải là 010, còn -111 có ba cột bên phải là 011 (hình 2.8c). Khi so sánh hai tập giá trị biến này: -011 với -111, ta được implicăng đơn giản --11 và tích logic của hai tập 010 và 011 cho ta 010. Tức là implicăng --11 thuộc hàm y_2 .

Trong hai tập giá trị biến đem so sánh, ta sẽ đánh dấu “v” cạnh tập giá trị biến nào mà cùng thuộc một số hàm vừa là các hàm của implicăng đơn giản thu được.

Thí dụ, khi so sánh 1-01 / 001 với 1-11 / 011 ta được 1--1 / 001 thì sẽ đánh dấu tập giá trị biến 1-01 vì 1-01 và 1--1 cùng thuộc một hàm là y_3 . Hàm y_2 không nhận 1--1 làm implicăng đơn giản của mình nên tập giá trị biến 1-11 không được đánh dấu.

Kết quả so sánh bước hai cho trên hình 3.8d. Trên hình 3.8e là tập hợp các implicăng đơn giản của hệ.

Bước tiếp theo là lập bảng Kvaina để tìm các implicăng đơn giản thực thụ. Bảng được xây dựng với các tập giá trị biến của cả ba hàm xác định không đầy đủ, mỗi hàm thành một nhóm cột riêng - hình 3.9.

$x_1x_2x_3x_4$	y_1				y_2			y_3				
	0000	0010	0111	1010	0010	1011	1111	0010	0111	1000	1011	1111
0010		x			x			x				
1000										x		
1100												
0111			x						x			
-000										x		
-010		x		x	x							
-111							x		x			x
1-11						x	x				x	x
00-0	x	x						x				
10-0												
01-1			x									
100-										x		
-01-					x	x						
--11						x	x					
1--1											x	x

Hình 3.9. Bảng Kvaina đơn giản hệ 3 hàm đã cho

Vì những implicăng mà cả ba cột bên đều là 0 thì không thuộc bất kỳ hàm nào trong ba hàm nói trên, nên không đánh dấu x với các implicăng đó. Đó là trường hợp tập giá trị biến 1100 trong hình 3.9. Tương tự, đối với tập 1000 ta không đánh dấu x trong các cột của y_1 và y_2 vì đối với hai hàm này, 1000 là tập giá trị biến mà hàm bất định.

Dùng algôrit tìm các implicăng đơn giản thực thụ ta thấy hai implicăng đơn giản thực thụ nhất thiết phải chọn là -010 và 00-0. Phần còn lại cho trên hình 3.10.

$x_1x_2x_3x_4$	y_1	y_2		y_3			
	0111	1011	1111	0111	1000	1011	1111
0010							
1000					x		
0111	x			x			
-000					x		
-111			x	x			x
1-11		x	x			x	x
01-1	x						
100-					x		
-01-		x					
--11		x	x				x
1--1						x	x

Hình 3.10. Bảng Karnaugh sau khi đã rút hai implicăng đơn giản

Kết quả thực hiện algôrit tìm các implicăng đơn giản thực thụ cho ta phủ tối thiểu của hệ gồm:

$$\bar{x}_1\bar{x}_2\bar{x}_4, \bar{x}_2x_3\bar{x}_4, \bar{x}_1x_2x_3x_4, \bar{x}_2\bar{x}_3\bar{x}_4, x_1x_3x_4$$

trong đó: $y_1 = \{\bar{x}_1\bar{x}_2\bar{x}_4, \bar{x}_2\bar{x}_3\bar{x}_4, \bar{x}_1x_2x_3x_4\}$,

$$y_2 = \{x_1x_3x_4, \bar{x}_2x_3\bar{x}_4\},$$

$$y_3 = \{\bar{x}_1x_2x_3x_4, x_1x_3\bar{x}_4, \bar{x}_2\bar{x}_3\bar{x}_4, \bar{x}_1\bar{x}_2\bar{x}_4\}$$

3.6.3. Phương pháp “Dấu” đơn giản hệ hàm

Một trong những phương pháp đơn giản hệ hàm cho algôrit chính xác để lập trình là phương pháp “Dấu”. Nguyên tắc của phương pháp này là biến hệ m hàm có n biến thành một hàm có $(m+n)$ biến.

Algôrit thực hiện phương pháp “Dấu” đơn giản hệ hàm như sau:

+ Đơn giản từng hàm riêng rẽ. Bước này có thể dùng bất cứ phương pháp đơn giản hàm nào.

+ Kết quả mỗi hàm cho ta một phủ tối thiểu dưới dạng các phức hợp khối:

$$I_i, i = \overline{1, m} : L_1, L_2, \dots, L_m$$

+ Lấy giao của chúng:

$$L_1 \cap L_2 ; L_1 \cap L_3 ; \dots ; L_1 \cap L_m$$

$$L_1 \cap L_2 \cap L_3 ; \dots ; L_1 \cap L_2 \cap L_3 \cap \dots \cap L_m$$

+ Đối với mỗi khối thuộc L_i ; $i = \overline{1, m}$ và thuộc giao của chúng ta lập các dấu như sau:

- Các khối thuộc L_1 dấu có dạng: $x11...1$

- Các khối thuộc L_2 dấu có dạng: $1x1...1$

-

- Các khối thuộc $L_1 \cap L_2$ dấu có dạng: $xx1...1$

-

- Các khối thuộc $L_1 \cap L_2 \cap L_3 \cap ... \cap L_m$ dấu có dạng: $xx...x$.

Các dấu này được viết kể bên trái của các phức hợp khối.

+ Tất cả các khối cùng với dấu của nó được coi như tập hợp các khối của một hàm logic có $(m + n)$ biến. Ký hiệu toàn bộ tập hợp khối này là C .

+ Tìm không gian các implicăng đơn giản Z của C theo phương pháp ROT, sử dụng phép nhân tọa độ $"*"$. Kết quả ta được không gian các implicăng đơn giản Z của C .

+ Tìm phủ tối thiểu của (Z, L_i) , tức là tìm các implicăng đơn giản thực thụ, sử dụng phép giao tọa độ $"\cap"$ và phép trừ tọa độ $"\#"$. Kết quả ta sẽ được một phủ tối thiểu dưới dạng $C_{\min}$ mà phần dấu của mỗi implicăng đơn giản vẫn giữ nguyên ý nghĩa như ban đầu.

Để minh họa phương pháp "Dấu" ta sẽ xét một thí dụ sau đây.

Thí dụ 3.19

Dùng phương pháp "Dấu", đơn giản hệ hàm sau đây:

$$y_1 = \begin{vmatrix} 1100x01 \\ 1101xx1 \end{vmatrix} = L_1 \quad y_3 = \begin{vmatrix} 011100x \\ 0100xxx \\ 0101x0x \end{vmatrix} = L_3$$

$$y_2 = \begin{vmatrix} 011101x \\ 0101xxx \\ 1100x1x \end{vmatrix} = L_2 \quad y_4 = \begin{vmatrix} 1011xxx \\ 01111xx \\ 1101xxx \\ 1111xxx \end{vmatrix} = L_4$$

Thực hiện các bước theo algôrit trên ta có:

1. Lấy giao của L_1, L_2, L_3 và L_4 : Kết quả hầu hết nhận giá trị trống ($\emptyset$), riêng:

$$L_2 \cap L_1 = 0101x1x$$

$$L_3 \cap L_1 = 0101x0x$$

Do đó phủ C có dạng như hình 3.11a, trong đó đã bỏ đi khối $1x11010x1x$ và $11x10101x0x$ vì chúng không bị phủ bởi hai khối tương ứng là: $1x1x010x1x$ và $11xx0101x0x$.

2. Tìm không gian các implicăng đơn giản của C:

a) Vì C không chứa một khối 0 nào nên $Z^0 = \emptyset$.

Tương tự $Z^1 = Z^0 = \emptyset$.

C =	x1111100x01	C*C =	x111110xx01	1x1101x101x	111x01x111x
	x1111101xx1		11111100xx1	1111x100x1x	111x01x110x
	1x11011101x		1111x100x01	1111110xx10	111xx101xx0
	1x111100xxx		1111110xx11	11110111x0x	111x0101x1x
	11x1011100x		1111x101xx1	111101x100x	111x0101x0x
	11x10100xxx		111111x1xx1	1111x11100x	111x11x1xx0
	111x1011xxx		1111x101x11	11x101x100x	111xx101x10
	111x01111xx		1111x101x11	11x101x100x	111xx101x00
	111x0101xxx		1111x101x01	1111010xxxx	111x0101xxx
	111x1101xx0		1111x101x01	1111010xx1x	
	111x1111xxx		111101110xx	11x1010xx0x	b)
	1x1x0101x1x		11110111x1x	11x1010xx0x	
	11xx0101x0x		111101x101x	111x1x11xxx	
	a)		1111x11101x	111x01x11xx	
			1x1101x101x	111xx1111xx	

$\hat{C} =$	x1111100x01	11x101x100x	$C^* =$	x1111100x01
	x1111101xx1	11x101x100x		1x110111xx1
	1x11011101x	11x1010xx0x		11x1011100x
	1x111100x1x	11x1010xx0x		111x01111xx
	11x1011100x	111x1x111xx		111x0101xxx
	11x10100xxx	111x01x11xx		1x1101x101x
	111x1011xxx	111xx1111xx		11x101x100x
	111x01111xx	111x01x110x		11x1010xx0x
	111x0101xxx	111xx101xx0		111x01x110x
	111x1101xx0	111x0101x1x		111x0101x1x
	111x1111xxx	111x0101x0x		111x0101x0x
	1x1x0101x1x	111x11x1xx0		111xx101x10
	11xx0101x0x	111xx101x10		111xx101x00
	x111110xx01	111xx101x00		d)
	1x1101x101x	111x0101xxx		
	1x1101x101x	c)		

Hình 3.11. Đơn giản hệ hàm theo phương pháp “Đấu”

b) Để tìm Z^2 , ta phải tính C^*C như hình 3.11b. Các tập hợp khối $\hat{C}$ và C^* - hình 3.11c và d, trong đó:

$$\hat{C} = C \cup (C^*C)$$

Còn C^* là tập hợp các khối bị phủ bởi các khối khác.

Tập hợp các khối $C_2 = C_2 \setminus C^* \setminus \{Z^0, Z^1\}$ cho trên hình 3.12.

$C_2 =$	x111110xx01	111x1x111xx
	011111101xx1	111x01x11xx
	1x1101x101x	111xx1111xx
	1x111100x1x	111xx101x00
	11x10100xxx	11xx11x1x00
	11x101x100x	111x0101xxx
	11x1010xx0x	11xx0101x0x
	111x1011xxx	1x1x0101x1x
	111x1101xx0	
	111x1111xxx	

Hình 3.12. Tập hợp các khối C_2

Vì trong C_2 không có khối nào chứa hai dấu x nên $Z^2 = \emptyset$.

c) Để tìm Z^3 ta lại tìm $C_2 * C_2$ và trên cơ sở đó tìm C_3 như sau:

$$C_3 = C_3 \setminus C_3 \setminus \{Z^0, Z^1, Z^2\}$$

Kết quả ta có Z^3 như hình 3.13a.

$Z^3 =$	x111110xx01	$Z^4 =$	11x10100xxx
	x11111101xx1		111x0101xxx
	1x1101x101x		111x11x1xx0
	1x111100x1x		111xx101xx0
	11x110x100x		111xx1111xx
	11x1010xx0x		111x01x11xx
			111xx1x11x0
			11xx0101x0x
a)		b)	

Hình 3.13. Các khối Z^3 và Z^4

d) Tiến hành một cách tương tự ta có Z^4 - hình 3.13b.

e) Tiếp tục tìm Z^5 . Vì:

$$C_5 = C_4 \cup (C_4 * C_4) \setminus C_4^* \setminus \{Z^0, Z^1, Z^2, Z^3, Z^4\}$$

chỉ còn lại một khối duy nhất là 111x1x11xxx, nên:

$$Z^5 = 111x1x11xxx$$

Như vậy không gian các implicăng đơn giản của C là Z - hình 3.14.

$Z = (Z^0, Z^1, Z^2, Z^3, Z^4, Z^5) =$	x111110xx01	111xx101xx0
	x1111101xx1	111xx1111xx
	1x1101x101x	111x01x11xx
	1x111100x1x	111xx1x11x0
	11x101x100x	11xx0101x0x
	11x1010xx0x	1x1x0101x1x
	11x10100xxx	111x1x11xxx
	111x0101xxx	
	111x11x1xx0	

Hình 3.14. Các khối thuộc Z

3) Tìm phủ tối thiểu (Z_i, L_i) , $i = 1, 2, 3, 4$.

Ở đây ta có:

$$Z_1^m = \begin{vmatrix} 110xx01 \\ 1101xx1 \end{vmatrix} \begin{matrix} a \\ b \end{matrix} \quad L_1 = \begin{vmatrix} 1100x01 \\ 1101xx1 \end{vmatrix}$$

Ta tìm $U(a, Z_1^m)$. Đó là những khối của Z_1^m mà giao của chúng với khối a là không trống. Ta có:

$$110xx01 \cap 110xx01 = 110xx01 \neq \emptyset$$

$$1101xx1 \cap 110xx01 = 1101x01 \neq \emptyset$$

$$\text{Do đó: } U(a, Z_1^m) = U(a, Z_1^m) \setminus a = 1101xx1$$

Kiểm tra điều kiện:

$$K(a \cap L_1) = 110xx01 \cap (1101x01, 1101xx1) = 1100x01 \cap 1101x01$$

$$K(a \cap L_1(a, Z_1^m)) = K((1101x01, 1101x01) \cap 1101xx1) = 1101x01$$

$$\neq K(a, L_1).$$

Do đó suy ra $a = 110xx01$ là implicăng đơn giản thực thụ thuộc phủ của L_1 .

Kiểm tra tương tự đối với khối b của Z_1^m , dễ dàng thấy rằng:

$$K(b \cap L_1) = K(b \cap L_1 \cap U(b, Z_1^m))$$

Do đó b không phải là implicăng đơn giản thực thụ của L_1 .

Như vậy $E_1 = a = 110xx01$.

Vì không phải tất cả các khối thuộc L_1 được phủ bởi E_1 , nên cần tìm E_2 . Để nhằm mục đích đó ta tìm L_2 . Đó là tập hợp các khối thuộc L_1 mà không bị phủ bởi E_1 .

$$L_2 = L_1 \# E_1 = L_1 \# a = (1100x01, 1101xx1) \# 110xx01 = 1101xx1$$

$$Z_2^m = Z_1^m \setminus E_1 = 1101xx1$$

Vì Z_2^m chỉ có một khối duy nhất và khối này phủ L_2 , nên: $E_2 = 1101xx1$.

Như vậy, đối với Z_1^m, L_1 ta có:

$$E = E_1 \cup E_2 = 110xx01 \cup 1101xx1$$

Cũng theo cách trên ta xác định các implicăng đơn giản thực thụ cho y_2 , nhưng bây giờ ta có:

$$L_1 = \begin{vmatrix} 011101x \\ 0101x1x \\ 1100x1x \end{vmatrix} \quad Z_2^m = \begin{vmatrix} 01x101x \\ 1100x1x \\ 0101x1x \end{vmatrix} \begin{matrix} a \\ b \\ c \end{matrix}$$

$$U'(a, Z_2^m) = (011101x, 0101x1x) \setminus 01x101x = 0101x1x$$

Kiểm tra điều kiện:

$$K(a \cap L_1) = (011101x, 010101x)$$

$$K(a \cap L_1 \cap U'(a, Z_2^m)) = 010101x \neq K(a \cap L_1)$$

Do đó a là implicăng đơn giản thực thụ của y_2 .

Xác định theo cách tương tự, ta có b và c cũng là implicăng đơn giản thực thụ của y_2 . Tức là:

$$E_1 = \begin{vmatrix} 01x101x \\ 1100x1x \\ 0101x1x \end{vmatrix}. \text{ Vì } L_2 = L_1 \neq E_1 = \emptyset \text{ nên } E_2 = \emptyset$$

Đối với hàm y_3 , ta có:

$$L_1 = \begin{vmatrix} 011100x \\ 0101x1x \\ 0101x0x \end{vmatrix} \quad Z_2^m = \begin{vmatrix} 01x100x \\ 010xx0x \\ 0100xxx \\ 0101x0x \end{vmatrix} \begin{matrix} a \\ b \\ c \\ d \end{matrix}$$

Vì $a \cap b = 010100x$; $a \cap c = \emptyset$; $a \cap d = 010100x$;

$$U'(a, Z_2^m) = (010xx0x, 0101x0x)$$

$$K(a \cap L_1) = 010100x \cap 011100x = 01x100x$$

$$K(a \cap L_1 \cap U'(a, Z_2^m)) = 010100x \neq K(a \cap L_1)$$

Do đó a là implicăng đơn giản thực thụ của y_3 .

Cũng bằng cách tương tự có thể nhận được kết quả sau:

- b và c là implicăng đơn giản thực thụ.

- d không là implicăng đơn giản thực thụ.

Như vậy ta có:

$$E_1 = \begin{vmatrix} 01x100x \\ 010xx0x \\ 0100xxx \end{vmatrix}$$

Tính $L_2 = L_1 \# E_1 = \emptyset$. Do đó $E_2 = \emptyset$.

Đối với hàm y_4 , ta có:

$$L_1 = \begin{vmatrix} 1011xxx \\ 01111xx \\ 0101xxx \\ 1101xx0 \\ 1111xxx \end{vmatrix} \quad Z_4^m = \begin{vmatrix} 0101xxx & a \\ 11x1xx0 & b \\ x101xx0 & c \\ x1111xx & d \\ 01x11xx & đ \\ x1x11x0 & e \\ 0101x0x & g \\ 0101x1x & h \\ 1x11xxx & i \end{vmatrix}$$

Kiểm tra điều kiện là implicăng đơn giản thực thụ đối với các khối của Z_4^m , ta có:

$$U(a, Z_4^m) = \begin{vmatrix} x101xx0 \\ 01x11xx \\ x1x11x0 \\ 0101x0x \\ 0101x1x \end{vmatrix} \quad \begin{aligned} K(a \cap L_1) &= 0101xxx, \text{ còn} \\ K(a \cap L_1 \cap U(a, Z_4^m)) &= \\ &0101xx0 \cup 01011xx \cup 01011x0 \cup \\ &0101x0x \cup 0101x1x \end{aligned}$$

Do đó a là implicăng đơn giản thực thụ đối với L_1 , tức đối với y_4 . Một cách tương tự ta có các khối c , $đ$ và i cũng là các implicăng đơn giản thực thụ đối với y_4 . Vậy:

$$E_1 = \begin{vmatrix} 0101xxx \\ x101xx0 \\ 01x11xx \\ 1x11xxx \end{vmatrix}$$

Vì $L_2 = L_1 \# E_1 = \emptyset$, nên $E_2 = \emptyset$.

Kết quả cuối cùng ta có $C_{\min}$ của hệ hàm như sau:

$$C_{\min} = \begin{vmatrix} x111110xx0x \\ x1111101xx1 \\ 1x1101x101x \\ 1x111100x1x \\ 1x110101x1x \\ 11x101x100x \\ 11x1010xx0x \\ 11x10100xxx \\ 111x0101xxx \\ 111xx101xx0 \\ 111x01x11xx \\ 111x1x11xxx \end{vmatrix}$$

BÀI TẬP CHƯƠNG III

1. Tìm hàm đánh giá độ phức tạp F_d của các hàm sau đây:

$$a) y_1 = \bar{x}_1 x_2 + x_3 x_4 + x_1 x_2 \bar{x}_3$$

$$b) y_2 = \bar{x}_1 x_4 + x_2 x_3 \bar{x}_4 + \bar{x}_3 \bar{x}_4$$

$$c) y_3 = x_1 x_2 + (\bar{x}_1 + \bar{x}_2) x_3 + \overline{x_1 x_2 + \bar{x}_3}$$

2. Dùng các định luật và các tính chất cơ bản của đại số lôgic để đơn giản các hàm sau đây:

$$2.1 \quad y_1 = \bar{x}_1 x_2 \bar{x}_3 + x_1 x_2 x_4 + x_1 x_3$$

$$y_2 = \bar{x}_1 \bar{x}_2 + \bar{x}_1 \bar{x}_2 x_3 + x_1 x_3 x_4 + \bar{x}_1 x_3 x_4$$

$$y_3 = x_2 x_3 + x_2 x_4 + \bar{x}_1 x_2$$

$$y_4 = x_1 x_2 x_3 \bar{x}_4 + x_1 x_2 \bar{x}_3 + x_1 \bar{x}_2 x_3 \bar{x}_4 + \bar{x}_1 x_2 x_3$$

$$2.2 \quad y_1 = (x_1 + \bar{x}_2 + x_3)(x_1 + \bar{x}_2 + \bar{x}_3)(\bar{x}_1 + \bar{x}_2 + x_3)(x_1 + x_2 + x_3)$$

$$y_2 = (\bar{x}_1 + \bar{x}_2 + x_3 + \bar{x}_5 + \bar{x}_6)(\bar{x}_1 + \bar{x}_2 + \bar{x}_4 + \bar{x}_5 + x_6)(x_1 + x_3 + x_5 + x_6).$$

$$.(x_1 + x_2 + \bar{x}_3 + \bar{x}_4 + x_6)(\bar{x}_1 + \bar{x}_2 + x_3 + x_5 + x_6)(x_1 + x_2 + \bar{x}_3 + \bar{x}_5 + x_6).$$

$$.(x_2 + x_3 + x_4 + x_5 + x_6)$$

$$2.3 \quad y_1 = f(3, 4, 7, 8, 9, 10, 11, 12); n = 4, b = 10$$

$$y_2 = f(3, 4, 6, 8, 10, 13, 15); n = 4, b = 10$$

$$y_3 = f(0000, 0001, 0010, 0011, 1000, 1010, 0111, 1101); n = 4, b = 2$$

$$y_4 = f(0010, 0100, 0101, 0111, 1000, 1001, 1110, 1111); n = 4, b = 2$$

$$2.4 \quad y_1 = (x_1 x_2 + \bar{x}_1 \bar{x}_2) x_4 + (x_1 x_2 + x_1 \bar{x}_2) x_3$$

$$y_2 = x_2(x_1 x_3 + x_4) + \overline{(x_1 + x_3)}(x_2 + x_4)$$

$$y_3 = ((x_1 + x_3)(x_1 + x_2)x_4)(x_1 + x_2 + x_3 + x_4)$$

$$y_4 = (x_1 + x_2 x_3) \bar{x}_4 x_5 + \overline{(x_2 + x_1 x_3)}(x_1 + \bar{x}_3)$$

3. Dùng bảng Karnaugh đơn giản các hàm lôgic sau đây:

3.1 Đơn giản các hàm cho trong mục 2.1; 2.2; 2.3.

$$3.2 \quad y_1 = f(0, 1, 4, 5, 7, 9, 10, 11); n = 4, b = 10$$

$$y_2 = f(0, 2, 3, 4, 6); n = 3, b = 10$$

$$y_3 = f(3, 4, 5, 6); n = 3, b = 10$$

$$y_4 = f(1, 4, 5, 8, 9, 12, 13); n = 4, b = 10$$

$$3.3 \quad y_1 = f'(0, 2, 5, 7, 8, 11); f'(1, 3, 6, 9); n = 4, b = 10$$

$$y_2 = f'(1, 2, 3, 4, 8, 9, 10, 12); f'(0, 6, 7, 15); n = 4, b = 10$$

$$y_3 = f'(4, 5, 9); f'(1, 3, 6); n = 4, b = 10$$

$$y_4 = f^l(0, 4, 5, 7, 11); f^r(1, 3, 6); n = 4, b = 10$$

$$y_5 = f^l(0, 1, 7, 8, 11, 12); f^r(3, 4, 6); n = 4, b = 10$$

$$3.4 \quad y_1 = f^0(2, 3, 6, 7, 8, 11); f^-(12, 15); n = 4, b = 10$$

$$y_2 = f^0(5, 6, 7, 11, 15); f^-(2, 3, 4, 14); n = 4, b = 10$$

$$y_3 = f^0(0, 1, 2, 3, 7, 9, 12, 13, 15); f^-(5, 6, 8, 11); n = 4, b = 10.$$

4. Dùng phương pháp bảng đối xứng đơn giản các hàm sau đây:

$$4.1 \quad y_1 = f(0, 1, 3, 5, 7, 8, 10, 12, 17, 20, 25, 26, 27); n = 5, b = 10$$

$$y_2 = f(0, 3, 4, 12, 14, 15, 17, 20 \div 30); n = 5, b = 10$$

$$y_3 = f(1 \div 6, 11, 13, 14, 15, 17 \div 22, 28 \div 32); n = 5, b = 10.$$

$$4.2 \quad y_1 = f(0, 1, 2, 10 \div 15, 20 \div 26, 30 \div 40, 60 \div 62); n = 6, b = 10$$

$$y_2 = f(3 \div 10, 12 \div 15, 21 \div 25, 38 \div 41, 60 \div 62); n = 6, b = 10$$

$$y_3 = f(0 \div 12, 17 \div 23, 31 \div 38, 40 \div 55); n = 6, b = 10$$

$$4.3 \quad y_1 = f^l(0 \div 3, 7 \div 9, 16 \div 21, 25 \div 29); f^r(4 \div 6, 22 \div 24); n = 6, b = 10$$

$$y_2 = f^l(10 \div 20, 29 \div 32); f^r(0 \div 9, 21 \div 25); n = 5, b = 10$$

$$y_3 = f^l(0 \div 12, 17, 19, 24 \div 32, 41 \div 49, 52 \div 56);$$

$$f^r(14, 15, 36 \div 39); n = 6, b = 10$$

$$y_4 = f^l(20 \div 32, 41 \div 46, 52 \div 55, 60 \div 62); n = 6, b = 10$$

$$f^r(0 \div 9, 36 \div 40).$$

5. Dùng phương pháp McClusky đơn giản các hàm sau:

$$5.1 \quad y_1 = f(0, 2, 3, 5, 7), n = 3, b = 10$$

$$y_2 = f(0, 1, 4, 5, 6), n = 3, b = 10$$

$$y_3 = f(2, 3, 5, 7), n = 3, b = 10$$

$$5.2 \quad y_1 = f(0, 1, 2, 4, 8, 10, 11, 15), n = 4, b = 10$$

$$y_2 = f(2, 3, 5, 9, 10, 12), n = 4, b = 10$$

$$y_3 = f(8, 9, 11, 12, 15), n = 4, b = 10$$

$$y_4 = f(0000, 0001, 0011, 0100, 0101, 0111, 1011, 1100), n = 4, b = 2.$$

$$5.3 \quad y_1 = x_1 x_2 x_3 + \bar{x}_2 x_4 + \bar{x}_1 x_2 x_4 + \bar{x}_1 \bar{x}_2 x_3$$

$$y_2 = x_1 x_2 x_3 + \bar{x}_1 x_2 x_4 + x_1 \bar{x}_2 x_4 + \bar{x}_1 \bar{x}_2 x_3$$

$$y_3 = \bar{x}_1 \bar{x}_3 \bar{x}_4 + x_2 \bar{x}_3 x_4 + \bar{x}_1 x_2 x_4 + \bar{x}_2 x_3 \bar{x}_4 + \bar{x}_1 \bar{x}_2 \bar{x}_3 \bar{x}_4 \\ + x_1 x_2 x_3 x_4 + \bar{x}_1 x_2 x_3 x_4$$

$$5.4 \quad y_1 = (x_1 x_2 + \bar{x}_1 x_2) x_4 + (\bar{x}_1 \bar{x}_2 + x_1 \bar{x}_3) x_4$$

$$y_2 = ((\bar{x}_1 \bar{x}_4) \bar{x}_3 + (\bar{x}_2 x_4) x_3)$$

$$y_3 = ((x_1 + x_2)(x_1 + x_3) x_4)(x_2 + x_3 + x_4)$$

6. Thực hiện các phép toán sau đây:

$$\begin{array}{ll}
 6.1 & 0101 * 0x01 \qquad 1xx0 * x00x \\
 & 0110 * 0111 \qquad 10x1 * x0x1 \\
 & 1101 * 0101 \qquad 1xx1 * 0xxx \\
 & 1010 * 1x11 \qquad x1x1 * x0x0
 \end{array}$$

$$6.2 \quad L = (0x1x, 00x0, 1x0x, xx01, x01x)$$

$$C = (01x1, x0x1, x1x0, x100)$$

Tính $L * C$.

$$\begin{array}{ll}
 6.3 & 0010 \cap 10x0 \qquad 01x1 \cap 0x1x \\
 & x001 \cap 1x01 \qquad 10xx \cap 10x1 \\
 & xx00 \cap x000 \qquad x0xx \cap 10x0 \\
 & 1x0x \cap 0x1x \qquad x1x1 \cap 0x0x
 \end{array}$$

$$6.4 \quad L = (0x1x, x0xx, 1xx0, x1x1)$$

$$C = (01x1, x0x1, x1x0, x100)$$

Tính $L \cap C$.

$$\begin{array}{ll}
 6.5 & x10 \# xx1 \qquad 0x1x \# 10xx \\
 & x0x \# 1xx \qquad 1xx0 \# 1xxx \\
 & xxx \# 11x \qquad x0x1 \# 00xx \\
 & x00 \# 10x \qquad x1x1 \# xx0x
 \end{array}$$

$$6.6 \quad \text{Cho: } E = (00x, 10x, x00, 1xx, 00x)$$

$$C = (0xx, 1xx, xx0)$$

Tính $E \# C$.

7. Dùng phương pháp ROT đơn giản các hàm sau:

$$7.1 \quad y_1 = f(000, 1x0, 11x, 01x, 1x0)$$

$$y_2 = f(001, x10, x11, 1x0, xx1)$$

$$7.2 \quad y_1 = (000, 00x1, xx00, xx10, xx11, 0xxx, 0xx1)$$

$$y_2 = (xx00, x010, 0x10, 00x1, 00x0, x11x)$$

8. Dùng bảng Karnaugh đơn giản các hệ hàm sau:

$$8.1 \text{ a) } y_1 = x_1 \bar{x}_2 x_3 + \bar{x}_3 \bar{x}_4 + x_1 \bar{x}_2 x_4$$

$$y_2 = x_1(x_3 + \bar{x}_2)(x_3 + x_4)$$

$$\text{b) } y_1 = x_1 x_2 + x_1 \bar{x}_2 \bar{x}_3 \bar{x}_4$$

$$y_2 = \bar{x}_1 x_3 x_4 + x_1 x_2 \bar{x}_3 + x_1 x_2 x_4$$

$$y_3 = \bar{x}_1 \bar{x}_2 \bar{x}_4 + x_3 x_4 + x_1 x_2 x_4$$

$$\begin{aligned} \text{c) } y_1 &= (x_3 + x_4)(x_1 + x_2 + x_3 + x_4) \\ y_2 &= (x_2 + \bar{x}_4)(x_1 + x_3 + x_4)(x_1 + \bar{x}_2 + x_3 + x_4) \\ y_3 &= (\bar{x}_1 + \bar{x}_2 + \bar{x}_4)(\bar{x}_1 + x_3 + x_4)(x_3 + x_2 + x_4)(\bar{x}_1 + \bar{x}_2 + x_4) \end{aligned}$$

$$8.2 \quad y_1 = f^1(0, 3, 6, 10, 13, 14), f^-(4, 8, 9, 15); n = 4, b = 10$$

$$y_2 = f^1(1, 3, 4, 13, 15); f^-(0, 5, 7, 8, 9, 14); n = 4, b = 10$$

9. Dùng phương pháp “Đấu” đơn giản hệ hàm sau đây:

$$9.1 \text{ a) } y_1 = f(1, 2, 5, 6)$$

$$y_2 = f(0, 1, 5, 6, 7); n = 3, b = 10$$

$$\text{b) } y_1 = f^1(0, 2, 5, 6); f^-(1, 3)$$

$$y_2 = f^1(0, 4, 7); f^-(3, 6); n = 3, b = 10$$

$$\text{c) } y_1 = f(0, 1, 2, 5, 6, 9, 12)$$

$$y_2 = f(0, 1, 3, 5, 8, 9, 14, 15); n = 4, b = 10$$

9.2 Đơn giản kết hợp hệ hàm cho trong bài 8.1.

PHẦN II

PHƯƠNG PHÁP TỔNG HỢP VÀ PHÂN TÍCH THIẾT BỊ SỐ TỔ HỢP

Chương 4

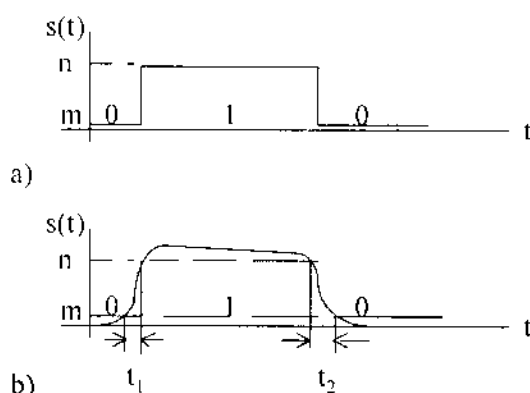
TỔNG HỢP CẤU TRÚC THIẾT BỊ SỐ TỔ HỢP

4.1. TÍN HIỆU VÀ CÁC MẠCH LÔGÍC

4.1.1. Tín hiệu nhị phân

Tín tức được biểu hiện qua các đại lượng vật lý đặc trưng như dòng điện, điện áp, tần số... Những đại lượng hay quá trình vật lý mang tín tức gọi là các tín hiệu. Thông thường các tín hiệu thay đổi liên tục theo thời gian, nhưng cũng có loại tín hiệu chỉ nhận giá trị ở những thời điểm nhất định. Tín hiệu chỉ nhận hai giá trị gọi là tín hiệu nhị phân hay tín hiệu lôgíc vì hai giá trị của nó có thể gán với hai đại lượng lôgíc là thực hoặc giả (hay 1 hoặc 0).

Tín hiệu nhị phân hay tín hiệu lôgíc lý tưởng là tín hiệu mà thời gian chuyển từ mức ứng với giá trị 1 sang mức ứng với giá trị 0 hay ngược lại bằng 0. Một dạng tín hiệu như vậy có sườn thẳng đứng như hình 4.1a.



Hình 4.1. Tín hiệu lôgíc lý tưởng và thực tế

Trong trường hợp như vậy, ta có thể gán một mức nào đó, thí dụ mức m với giá trị lôgíc 0 (hoặc với giá trị 1), còn n với giá trị lôgíc 1 (hoặc với giá trị 0). Nếu mức cao n được gán với giá trị lôgíc 1, còn mức thấp - với giá trị lôgíc 0, ta có lôgíc dương. Nếu gán ngược lại ta có lôgíc âm. Trên hình 4.1a là lôgíc dương.

Trên thực tế, các tín hiệu thường không có dạng lý tưởng, chúng thường biến đổi liên tục theo thời gian và các đột biến từ mức này sang mức khác có dạng hàm mũ, với biên độ không phải là hai mức rõ rệt - hình 4.1b. Trong trường hợp này, tín hiệu vẫn có thể coi là tín hiệu logic, nếu ta coi biên độ tín hiệu $s(t) \geq n$ là giá trị logic 1, còn $s(t) \leq m$ là giá trị logic 0. Khi đó ta vẫn có tín hiệu logic. Tuy vậy, tín hiệu logic này có thời gian chuyển đổi từ mức 0 sang mức 1 và ngược lại không bằng 0. Thời gian mà tín hiệu có giá trị $m < s(t) < n$ sẽ là thời gian không phải là mức 1 cũng không phải là mức 0. Nói cách khác, lúc này tín hiệu không xác định là 1 hay 0. Tại thời gian đó, một mạch logic có thể hiểu đó là 1, cũng có thể hiểu đó là 0. Ta nói mạch bất định đối với tín hiệu đó.

Rõ ràng, để mạch làm việc tin cậy, không lâm vào tình trạng bất định, thời gian này phải càng ngắn càng tốt. Cũng chính vì thế mà trong thực tế ta thường phải làm cho các sườn của tín hiệu logic thẳng đứng.

Các mức m và n trong các hệ mạch logic thông thường là:

+ Mức m (mức 0) có giá trị điện áp từ 0 đến 0,8 vôn. Mức 0,8 vôn là mức điện áp dư trên transistor khi nó ở chế độ bão hoà.

+ Mức n (mức 1) có giá trị từ 2,8 vôn đến 5 vôn, là mức điện áp trên transistor khi nó ở chế độ ngắt. Đó là mức nguồn cung cấp cho transistor. Trong các ứng dụng chuyên dụng khác, mức logic 1 có thể lên tới 18 vôn.

4.1.2. Khái niệm về mạch logic

Mạch logic là tập hợp các phần tử logic, được nối với nhau theo một quy luật nhất định, nhằm thực hiện một chức năng nhất định.

Tuỳ thuộc vào kết cấu và tính chất của các mạch mà người ta chia mạch logic thành hai loại chính: Mạch logic tổ hợp và mạch logic loại dây hay có nhớ.

Mạch logic tổ hợp là mạch mà tín hiệu ra tại một thời điểm chỉ phụ thuộc vào tín hiệu vào tại thời điểm đó. Đối với mạch số có nhớ, tín hiệu ra tại một thời điểm phụ thuộc không những vào tín hiệu vào tại thời điểm đó mà còn phụ thuộc vào tín hiệu vào tại các thời điểm trước đó.

Trong một mạch logic chỉ có thể nối đầu ra của một phần tử này với các đầu vào của các phần tử khác, không nối trực tiếp các đầu ra của chúng với nhau.

Nếu phần tử logic A có đầu ra nối với đầu vào của phần tử B, ta nói B là tải của A. Nói chung, đầu vào của B phải phối hợp với đầu ra của A về mức logic m và n và về trở kháng.

Mỗi phần tử logic không thể có quá nhiều tải. Số tải lớn nhất có thể mắc vào đầu ra của phần tử A gọi là hệ số phân tải của phần tử logic đó. Với các họ IC sản xuất công nghiệp hiện nay, hệ số phân tải thường từ 4 đến 6, tức có thể nối vào đầu ra của một phần tử logic tối đa từ 4 đến 6 phần tử logic khác. Trong trường hợp hệ số phân tải của

A nhỏ hơn số tải cần thiết, phải dùng phương pháp mắc song song hai hoặc nhiều phần tử A để đảm bảo khả năng chịu tải, vừa đảm bảo quan hệ thời gian của các đường truyền tín hiệu - hình 4.2.

Trong các mạch logic, các tín hiệu vào được biến đổi lần lượt khi đi qua các phần tử logic thành các tín hiệu trung gian, rồi trung gian thành tín hiệu ra. Đường đi của tín hiệu từ đầu vào tới đầu ra của mạch gọi là *đường truyền tín hiệu*. Mỗi đường truyền tín hiệu đều đi qua một số nhất định các phần tử logic. Đường truyền tín hiệu đi qua nhiều phần tử logic nhất gọi là *đường truyền của mạch*.

Số lượng các phần tử trên đường truyền của mạch là *cấp của mạch*.

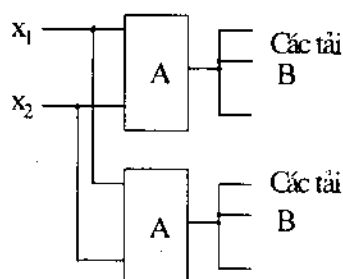
Về phương diện cấp của mạch, các mạch được chia ra thành mạch hai cấp và mạch nhiều cấp. Trong đó mạch nhiều cấp có từ 3 cấp trở lên.

Số cấp của một mạch càng lớn thì tốc độ làm việc của mạch càng thấp, vì mỗi lần đi qua một phần tử logic, tín hiệu bị giữ chậm một thời gian bằng τ . Trong đó τ là thời gian giữ chậm trung bình của một phần tử logic.

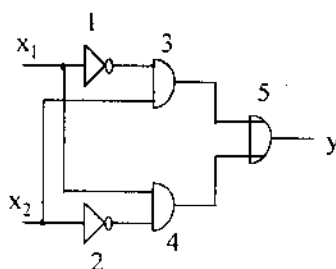
Nếu cấp của mạch là D thì thời gian tín hiệu bị giữ chậm nhiều nhất là $D\tau$ tính từ thời điểm tín hiệu tác động vào đầu vào. Nếu kể tới thời gian hữu hạn của các sườn xung tín hiệu vào thì thời gian cần thiết để xác lập một cách ổn định giá trị logic trên đầu ra của mạch là $D\tau + t_{1,2}$. Đó chính là thời gian quyết định tốc độ làm việc của mạch.

Trên hình 4.3 là mạch logic và các tín hiệu vào, ra, đường truyền và cấp của đường truyền. Cấp của mạch bằng 3.

Đầu vào	Đầu ra	Các phần tử đi qua	Cấp
x_1	y	1, 3, 5	3
		4, 5	2
x_2	y	2, 4, 5	3
		3, 5	2



Hình 4.2. Mắc song song để đảm bảo yêu cầu về tải



Hình 4.3. Mạch logic, các tham số vào, ra và cấp của mạch

Cấp của các đường truyền tín hiệu từ đầu vào tới đầu ra khác nhau là một trong những nguyên nhân gây nên *hiện tượng tranh chấp trạng thái* của các mạch logic. Do thời gian giữ chậm khác nhau, các tín hiệu khác nhau xuất hiện ở đầu ra vào những thời điểm khác nhau, do đó tín hiệu ra sẽ không còn như thiết kế nữa.

Trong mạch logic, các phần tử logic đứng cách đều đầu vào tạo thành một lớp hay cấp. Các tín hiệu đầu vào được coi là lớp 0. Các lớp phần tử của mạch ở hình 4.3 như sau:

Lớp 0: x_1, x_2

Lớp 1: 1, 2

Lớp 2: 3, 4

Lớp 3: 5

Trên đầu ra của các phần tử cùng lớp, tín hiệu xuất hiện cùng một lúc.

Trong các mạch logic chứa các đường phản hồi tạo thành chu trình kín không thể xác định rõ đường truyền của tín hiệu. Mạch logic chứa các đường phản hồi nói chung thuộc loại *mạch có nhớ* hay còn gọi là *mạch dây*.

Mô tả một mạch logic có thể mô tả tậpô logic hay mô tả chức năng.

+ Mô tả tậpô logic là mô tả thành phần và sự nối ghép giữa chúng. Để mô tả tậpô logic có thể dùng ma trận quan hệ. Hình 4.4 là ma trận quan hệ mô tả tậpô logic của mạch ở hình 4.3, trong đó các cột được đánh số theo số thứ tự của các phần tử logic và các đầu ra của mạch. Cột cuối cùng là số tải của các phần tử.

Các hàng được đánh số theo số thứ tự của các phần tử logic và các đầu vào x_i . Hàng cuối cùng ghi số lượng các đầu vào của các phần tử ở cột tương ứng. Tại giao của hàng i và cột j ta ghi giá trị 1 nếu đầu ra của phần tử ứng với hàng i được nối với đầu vào của phần tử ứng với cột j .

+ Việc mô tả chức năng của mạch được thực hiện thông qua việc mô tả toán học mối liên kết giữa các chức năng φ_i của phần tử logic i . Thí dụ chức năng của mạch ở hình 4.3 có thể viết như sau:

$$y = \varphi_5 (\varphi_3(\varphi_1(x_1), x_2), \varphi_4(\varphi_2(x_2), x_1))$$

Việc mô tả tậpô logic và mô tả chức năng như trên cho phép kiểm tra chức năng và chẩn đoán hư hỏng của mạch theo phương pháp lập trình trên máy tính.

	1	2	3	4	5	y	Số tải
1			1				1
2				1			1
3					1		1
4					1		1
5						1	1
x_1	1			1			2
x_2		1	1				2
Số đầu vào	1	1	2	2	2	1	

Hình 4.4. Mô tả tậpô của mạch

4.1.3. Giá của mạch logic

Khi thiết kế, xây dựng một mạch logic, đặc biệt trong trường hợp các mạch lớn, phức tạp, không thể không tính tới giá phải trả cho mạch cần thiết kế. Giá của mạch logic được tính theo giá của các phần tử logic dùng trong mạch:

$$C = \sum_{i=1}^n k_i C_i$$

trong đó, k_i - số lượng phần tử loại i ;

C_i - giá của một phần tử loại i ;

n - tổng số các loại phần tử logic dùng trong mạch.

Trong công nghiệp chế tạo vi điện tử, số lượng các chân của IC có ý nghĩa quan trọng trong giá C_i của mỗi phần tử mạch. Số lượng chân IC vừa thể hiện sự phức tạp của cấu trúc bên trong vừa thể hiện sự phức tạp và tốn kém trong công nghệ chế tạo.

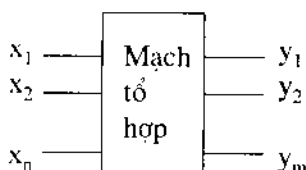
4.2. KHÁI NIỆM VỀ THIẾT BỊ SỐ TỔ HỢP

4.2.1. Đặc điểm mạch logic tổ hợp

Mạch mô tả cấu trúc logic của một thiết bị số bao gồm các ký hiệu quy ước của các phần tử logic của mạch và các đường nối liên hệ giữa chúng.

Về mặt chức năng, mạch logic tổ hợp là mạch mà tín hiệu ra tại thời điểm t chỉ phụ thuộc vào tín hiệu vào tại thời điểm t đó. Nói cách khác, đối với mạch tổ hợp ở hình 4.5 có thể viết:

$$y_j(t) = f(x_i(t)), \quad i = \overline{1, n}, j = \overline{1, m}$$



Hình 4.5. Mô hình mạch tổ hợp

Để xây dựng được cấu trúc logic của một thiết bị, trước hết cần mô tả chức năng hàm của nó một cách đầy đủ và đơn trị. Trên cơ sở đó cần rút ra được một hàm logic mô tả mỗi phụ thuộc của các đầu ra y_j vào các đầu vào x_i . Sau đó hàm được đơn giản và rút ra dưới dạng tuyến chuẩn tối thiểu hay dạng hội chuẩn tối thiểu; được biến đổi thành dạng toàn thuật toán VA-KH hai hay nhiều cấp hoặc dưới dạng HOAC-KH hai hay nhiều cấp hoặc một dạng hỗn hợp nào đó. Trong mọi trường hợp, hàm nhận được phải là hàm đã được đơn giản hoá và có thể đã phải tính tới độ tin cậy chức năng của mạch.

Hàm mô tả chức năng được đơn giản sẽ làm cho mạch trở nên đơn giản, đỡ tốn linh kiện, ít tốn năng lượng, làm việc tin cậy hơn, mang lại cả lợi ích về kinh tế lẫn kỹ thuật.

Như vậy, đơn giản hàm logic là một khâu quan trọng trong tổng hợp thiết bị số tổ hợp. Bên cạnh đó, đặc trưng kỹ thuật và công nghệ của các phần tử logic có ảnh hưởng không ít tới kết cấu và tính năng của mạch được thiết kế. Chẳng hạn, nếu các phần tử logic có số đầu vào nhỏ, không đủ điều kiện để thực hiện bằng mạch hai cấp thì bắt buộc phải thực hiện bằng mạch nhiều cấp. Khi đó tốc độ chậm hơn và độ tin cậy chức năng kém hơn, dễ có tranh chấp trạng thái hơn.

Ngoài ra, một thiết bị số thường được mô tả không phải bằng một hàm mà là một hệ các hàm. Trong trường hợp này việc đơn giản kết hợp một hệ hàm sẽ mang lại nhiều lợi ích. Trong phần sau của chương này chúng ta cũng sẽ đề cập tới vấn đề này một cách cụ thể hơn.

Trong một số trường hợp, khi số biến của thiết bị quá lớn, có thể áp dụng phương pháp phân hoạch hàm để có các dạng mà các phép tính được thực hiện với số biến phù hợp và do đó có thể thực hiện bằng các phần tử nhớ có số đầu vào hạn chế.

4.2.2. Các bước cơ bản tổng hợp thiết bị số tổ hợp

Các bước cơ bản trong tổng hợp một thiết bị số tổ hợp có thể là:

- 1) Phát biểu chức năng của thiết bị một cách đầy đủ và đơn trị.
- 2) Biểu diễn chức năng của thiết bị dưới dạng chuẩn như bảng giá trị thực hay dưới dạng một biểu thức đại số nào đó.
- 3) Đơn giản hàm hoặc hệ hàm logic của thiết bị để có được dạng tuyến chuẩn tối thiểu hay dạng hội chuẩn tối thiểu.
- 4) Căn cứ vào tính năng và đặc điểm của các phần tử logic sẽ được sử dụng để xây dựng thiết bị, biến đổi hàm về dạng thích hợp.
- 5) Thực hiện hàm hoặc hệ hàm theo cách nối ghép các phần tử logic theo từng cấp, có tính tới khả năng phân tải, số đầu vào của các phần tử logic cũng như sự phối hợp trở kháng giữa các phần tử mạch.

Tổng hợp cấu trúc thiết bị số là một bước quan trọng trong thiết kế một thiết bị. Để kiểm tra lại tính chuẩn xác trong tổng hợp, phát hiện những sai sót về kết cấu logic, phải tiến hành phân tích mạch ở chế độ tĩnh. Để làm việc này có thể kết hợp thử nghiệm với phân tích lý luận; cũng có thể tiến hành mô hình hoá thiết bị. Nếu thiết bị có độ phức tạp cao thì cần dùng máy tính với chương trình kiểm tra. Nhưng phân tích tĩnh chỉ cho phép phát hiện các sai sót về mặt kết cấu logic. Trên thực tế hoạt động của mạch còn phụ thuộc vào dạng tín hiệu vào và thời gian truyền các tín hiệu từ đầu vào tới đầu ra. Vì vậy tiếp theo việc phân tích tĩnh ta cần phân tích thiết bị ở chế độ động.

Trong đó có kể tới tham số thời gian của các tín hiệu và các phần tử logic. Chương 5 dành cho việc phân tích mạch tổ hợp.

4.2.3. Vi mạch số

1) Những khái niệm cơ bản

Vi mạch, ký hiệu là IC (Intergrated Circuits), là những phần tử mạch thực hiện một chức năng nhất định, được thực hiện bằng công nghệ tích hợp, có độ tin cậy cao và tính lắp lẫn rộng rãi.

Vi mạch có thể phân loại theo nhiều cách:

- + Phân loại theo bản chất tín hiệu vào và tín hiệu ra.
- + Phân loại theo mức độ tích hợp.
- + Phân loại theo công nghệ chế tạo.

Trường hợp thứ nhất: Phân loại theo tín hiệu

Tín hiệu được chia thành tín hiệu tương tự và tín hiệu số. Tín hiệu tương tự là loại có biên độ thay đổi một cách liên tục theo thời gian. Tín hiệu số là tín hiệu mà giá trị biên độ của nó chỉ có thể quy về một trong hai giá trị là 0 và 1, tương ứng với hai mức logic thấp L (Low) và cao H (High).

Như vậy, nếu phân loại theo tính chất của tín hiệu vào và ra thì ta có các loại vi mạch sau:

- + Tín hiệu vào và ra đều là loại tương tự thì IC là IC tương tự.
- + Tín hiệu vào là tương tự, tín hiệu ra là số thì IC là loại biến đổi tương tự- số (ADC).
- + Tín hiệu vào là số, còn tín hiệu ra là tương tự thì IC là bộ biến đổi số - tương tự (DAC).
- + Tín hiệu vào, ra đều là loại số thì IC là IC số (các mạch logic).

Vi mạch số thông dụng nhất là các cổng logic đơn giản như mạch VA, HOAC, VA-KH, HOAC-KH... và các vi mạch phức tạp về chức năng như các mạch nối ghép, các bộ biến đổi và các mạch nhớ ROM, RAM; các bộ vi xử lý...

Trường hợp thứ hai : Phân loại theo độ tích hợp

Độ tích hợp thể hiện mật độ các linh kiện mà cụ thể là mật độ các transistor trên một đơn vị diện tích của đơn tinh thể silic của IC. Theo độ tích hợp ta có các loại IC sau:

- + Vi mạch cỡ nhỏ (SSI - Small Scale Intergration): Loại có hàng chục transistor trong một vi mạch.
- + Vi mạch cỡ trung bình (MSI - Middle Scale Intergration): Là loại có hàng trăm transistor trong một vi mạch.

- + Vi mạch cỡ lớn (LSI - Large Scale Intergration): Là loại IC có hàng ngàn transistor trong một vi mạch.
- + Vi mạch cỡ cực lớn (VLSI- Very LSI): Là loại IC có hàng vạn transistor trong một vi mạch.

Trường hợp thứ ba: Phân loại theo công nghệ chế tạo

Có nhiều loại IC: IC khối rắn, IC màng mỏng, màng dây, IC lai... Phổ biến nhất là IC khối rắn với hai loại: IC số và IC tương tự.

IC số lại chia thành IC lưỡng cực và IC đơn cực.

Lưỡng cực có các loại:

- RDL (Resistor - Diode - Logic): Trong IC chỉ dùng điện trở và điốt.
- RTL (Resistor - Transistor - Logic): Trong vi mạch chỉ dùng điện trở và transistor.
- DTL (Diode - Transistor - Logic): Trong vi mạch chỉ dùng điốt và transistor.
- TTL (Transistor - Transistor - Logic): Trong vi mạch chỉ dùng transistor.
- ECL (Emitter Coupled Logic): Mạch logic có áp dụng phản hồi emitter có tốc độ làm việc nhanh.

Đơn cực có các loại:

Transistor dùng hiệu ứng trường (MOSFET – Metal - Oxide Semiconductor Field Effect Transistor) kênh dẫn loại p và kênh dẫn loại n. Loại dùng kênh dẫn loại p thường cần nhiều loại nguồn nuôi. Loại dùng kênh dẫn loại n thường dùng một nguồn nuôi duy nhất và loại CMOS (Complementary MOS) trong đó có dùng cả hai loại transistor loại p và loại n. Các loại vi mạch cỡ lớn như các bộ nhớ dung lượng cao, các bộ nối ghép hay các vi xử lý 8, 16 hay 32 bit... thường được chế tạo theo công nghệ này.

2) Các tham số cơ bản của vi mạch số

Trong các sổ tay IC, đối với mỗi IC thường cho các tham số cơ bản sau đây:

- + Điện áp nguồn: Thường cho điện áp danh định với sai số cho phép.
- + Thời gian trễ: Là thời gian cần thiết để mạch cho ra tín hiệu ổn định tại đầu ra tính từ khi có tác động đầu vào. Thông thường thời gian cần thiết để điện áp đầu ra thay đổi từ mức cao xuống mức thấp và từ mức thấp lên mức cao là khác nhau. Sổ tay IC thường cho giá trị lớn nhất.

+ Dòng vào cần thiết và dòng ra IC có thể cung cấp. Hai tham số này có liên quan trực tiếp tới khả năng chịu tải của các IC. Nếu dòng vào yêu cầu càng lớn thì làm giảm khả năng tải của phần tử cung cấp tín hiệu vào cho nó. Ngược lại, nếu dòng ra của một IC càng lớn thì nó có thể cung cấp đồng thời cho nhiều IC khác là tải của nó hơn, khả năng tải của nó cao hơn. Nếu dòng tải cung cấp nhỏ hơn dòng vào yêu cầu thì mạch làm việc không ổn định. Ví dụ, các tham số của IC VA hai đầu vào 7402 như sau:

	Thời gian giữ chậm điển hình	V_{cc}	Nguồn dòng
7402	10 ns	$V_{cc} = 5\text{ V} \pm 5\%$ $t = 0^\circ\text{C} \div 70^\circ\text{C}$	11 mA
74S02	3,5 ns		22 mA

Bảng tải đầu ra và vào (Input and output and Fan-out table)

	Mô tả	Họ 74	Họ 74S	Họ 74LS
A, B	Vào	1ul	1Sul	1LSul
y	Ra	10ul	10Sul	10LSul

Một đơn vị tải của họ 74 (ul) được hiểu là $40\text{ }\mu\text{A } I_{ih}$ và $-1,6\text{ mA } I_{il}$. Đối với 74S, đơn vị Sul là $50\text{ }\mu\text{A } I_{ih}$ và $-2,0\text{ mA } I_{il}$. Đối với họ 74LS, đơn vị LSul là $20\text{ }\mu\text{A } I_{ih}$ và $-0,4\text{ mA } I_{il}$.

3) Sơ đồ chân, sơ đồ logic và bảng chân lý

Trong các sổ tay IC, ngoài các tham số cơ bản còn cho sơ đồ chân, sơ đồ logic và bảng chân lý của IC.

+ **Sơ đồ chân:** Được dùng khi lắp ráp mạch. Các chân của IC được đánh số từ 1 trở đi. Cách đếm chân: Đặt úp IC, nhìn từ mặt có ký hiệu IC, bắt đầu từ ký hiệu đặc biệt (Một khuyên tròn nhỏ hay một rãnh nhỏ ở đầu IC) theo chiều ngược kim đồng hồ là chân 1, 2, ...

Các chân thường chia thành các nhóm chức năng khác nhau:

- **Chân nguồn:** Ký hiệu chung V_{cc} và V_{ss} . Là các chân cấp nguồn điện áp cho IC. Chân nối đất có ký hiệu là GND. Bố trí các chân này trong IC tùy thuộc từng loại IC cụ thể.

- Các chân đầu vào tín hiệu (Inputs) dùng đưa thông tin vào.

- Các chân thông tin đầu ra (Outputs) dùng lấy thông tin ra.

- Các chân điều khiển (Controls) dùng xác lập các chế độ như chế độ xác lập trạng thái ban đầu hay các chế độ làm việc khác nhau của IC.

- Các chân chọn chip (CS) và các chân đồng bộ (CLK) hay xóa (CLR) và thiết lập trạng thái ban đầu (PR, SET).

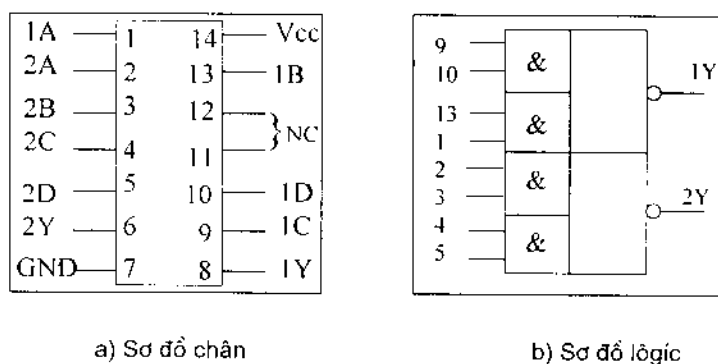
+ **Sơ đồ logic:** Biểu thị chức năng logic của IC. Sơ đồ logic được vẽ ngắn gọn phân kết cấu logic của mạch.

Ví dụ IC 7451 có sơ đồ chân và sơ đồ logic như trên hình 4.6.

Trong sơ đồ nguyên lý chỉ rõ cấu trúc logic của mạch gồm hai phần, mỗi phần gồm hai mạch VA hai đầu vào và một mạch HOAC-KH liên kết với nhau.

+ **Bảng chân lý (True table) của IC:** Bảng này cho biết mối quan hệ giữa các đại lượng đầu ra với các đại lượng đầu vào của IC thông qua tác động của các tập giá trị

biến đầu vào và các tín hiệu điều khiển. Hình 4.7 là thí dụ về sơ đồ chân và bảng chân lý của IC giải mã 74138.



Hình 4.6. Sơ đồ chân và sơ đồ logic của IC

Mạch có ba đầu vào số liệu và ba đầu vào điều khiển. Trong ba đầu vào điều khiển, hai đầu vào (E_1 , E_2) là tích cực ở mức thấp (L); một đầu vào (E_3) tích cực ở mức cao (H). IC 74138 chỉ làm việc khi nào toàn bộ các đầu vào điều khiển đều tích cực.

4) Các loại công nghệ vi mạch số thông dụng

Các loại công nghệ vi mạch số thông dụng là TTL, NMOS và CMOS.

a. Loại TTL

Là các vi mạch dùng transistor lưỡng cực. Đặc điểm cơ bản của loại vi mạch này là:

- + Transistor làm việc trong chế độ bão hoà và cắt sâu nên tốc độ làm việc chậm.
- + Độ tích hợp không lớn, thường các vi mạch SSI và MSI chế tạo theo công nghệ này.
- + Mức logic 0 thường từ 0 đến 0,8 vôn; mức logic 1 thường là 2,8 đến 5 vôn.

Các họ IC như 54 và 74 có hay không có hiệu ứng Shottky và sử dụng ít hay nhiều năng lượng nguồn đều dùng loại công nghệ TTL. Trong hai họ IC trên thì họ IC 54 có công nghệ chế tạo hoàn thiện hơn, dải nhiệt độ làm việc từ 0°C đến 125°C, độ tin cậy cao. Trong khi họ 74 có nhiệt độ làm việc chỉ từ 0°C đến 75°C, nhưng giá thành rẻ hơn, thường dùng trong các thiết bị dân dụng thông thường.

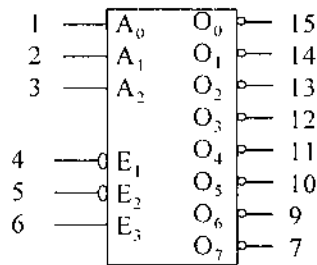
Một số loại IC dùng công nghệ TTL như:

SN54/7400; SN54/74S00; SN54/74LS00 là các IC mạch VA-KH hai đầu vào, không sử dụng và có sử dụng hiệu ứng Shottky (S) để làm tăng tốc độ làm việc hoặc vừa dùng cả hiệu ứng Shottky vừa đảm bảo tiêu hao ít năng lượng (LS).

SN54/7402; SN54/74S02; SN54/74LS02: Vi mạch có 4 cổng HOAC-KH hai đầu vào, không sử dụng và có sử dụng hiệu ứng Shottky (S) để làm tăng tốc độ làm việc hoặc vừa dùng cả hiệu ứng Shottky vừa đảm bảo tiêu hao ít năng lượng (LS).

SN54/7408; SN54/74S08; SN54/74LS08: Các vi mạch có 4 cổng VA hai đầu vào, không sử dụng và có sử dụng hiệu ứng Shottky (S) để làm tăng tốc độ làm việc hoặc vừa dùng cả hiệu ứng Shottky vừa đảm bảo tiêu ít năng lượng (LS).

SN54/7432; SN54/74S32; SN54/74LS32: Các vi mạch có 4 cổng HOAC hai đầu vào, không sử dụng và có sử dụng hiệu ứng Shottky (S) để làm tăng tốc độ làm việc hoặc vừa dùng cả hiệu ứng Shottky vừa đảm bảo tiêu hao ít năng lượng (LS).



a)

Đầu vào điều khiển	Đầu vào dữ liệu	Các đầu ra
E ₁ E ₂ E ₃	A ₂ A ₁ A ₀	O ₀ O ₁ O ₂ O ₃ O ₄ O ₅ O ₆ O ₇
1 x x	x x x	1 1 1 1 1 1 1 1
x 1 x	x x x	1 1 1 1 1 1 1 1
x x 0	x x 1	1 1 1 1 1 1 1 1
0 0 1	0 0 0	0 1 1 1 1 1 1 1
0 0 1	0 0 1	1 0 1 1 1 1 1 1
0 0 1	0 1 0	1 1 0 1 1 1 1 1
0 0 1	0 1 1	1 1 1 0 1 1 1 1
0 0 1	1 0 0	1 1 1 1 0 1 1 1
0 0 1	1 0 1	1 1 1 1 1 0 1 1
0 0 1	1 1 0	1 1 1 1 1 1 0 1
0 0 1	1 1 1	1 1 1 1 1 1 1 0

b)

Hình 4.7. Sơ đồ chân a) và bảng chân lý b) của IC 74138

SN54/7404; SN54/74S04; SN54/74LS04: Các IC có 6 mạch đảo, không sử dụng và có sử dụng hiệu ứng Shottky (S) để làm tăng tốc độ làm việc hoặc vừa dùng cả hiệu ứng Shottky vừa đảm bảo tiêu hao ít năng lượng (LS).

b. Loại công nghệ CMOS

Công nghệ CMOS (Complementary Metal oxide Semiconductor) có cấu trúc dùng kết hợp transistor loại P với transistor loại N, nên được gọi là bù (Complementary). Điện thế điểm chung A (cực G) của transistor giữa 0 vôn và $-U_{dd}$. Khi A là âm thì transistor P dẫn điện, N không dẫn điện và ngược lại, khi A là 0 vôn thì transistor loại N dẫn, loại P không dẫn điện. Vì vậy luôn luôn có một transistor không dẫn. Do đó hầu như không tiêu tốn năng lượng trong chế độ tĩnh.

Các IC số loại CMOS thường có các họ là SN40xxx; SN45xxx, có nguồn cấp từ 3 V đến 18 V, trong đó hai hay ba số trong phần "xxx" chỉ chủng loại IC. Ngoài ra, chủng loại IC CMOS như 74Cxxx và 74HCxxx có mức logic và nguồn tương thích với họ TTL.

Các IC công nghệ CMOS thường tiêu thụ rất ít năng lượng. Các họ 45 và 40 có thời gian trễ lớn, nhưng độ ổn định cao.

Thí dụ: IC 4501 là IC có bốn cổng HOAC-KH; 4511 là IC 4 cổng VA-KH; IC 4549 là loại có 6 mạch đảo; Loại 4550 là 6 mạch đệm; 4570 là IC bốn cổng cộng môđun 2...

c. Loại công nghệ NMOS

Công nghệ NMOS có cấu trúc gồm một nền P và hai vùng N, là các cực nguồn (S: Source) và cực máng (D: Drain) của transistor. Mặt trên cùng có lớp SiO_2 tạo ra cực cửa (G: Gate).

Nếu nối nền với đất và đưa điện áp dương vào G sẽ có một điện trường thẳng góc qua phần SiO_2 làm xuất hiện điện tích âm trong miền kênh. Nếu nối D với nguồn dương, S với nguồn âm sẽ có dòng điện chạy mà độ lớn phụ thuộc vào điện áp trên G.

Họ NMOS thường được sử dụng để chế tạo các IC có độ tích hợp cao như các bộ nối ghép và các bộ vi xử lý của các hãng INTEL, MOTOROLA, ZILOG...

IC dùng công nghệ NMOS và PMOS có mức logic và nguồn tương thích với họ TTL, có thể nối trực tiếp giữa chúng.

4.3. TỔNG HỢP CẤU TRÚC MẠCH HAI CẤP

Mạch logic hai cấp là loại mạch mà tín hiệu x_i và $\bar{x}_i$ đi từ đầu vào tới đầu ra tối đa chỉ qua hai phần tử logic. Mạch hai cấp có tốc độ làm việc nhanh và ít xảy ra tranh chấp trạng thái hơn so với mạch nhiều cấp.

Tuy vậy không phải lúc nào cũng có thể thực hiện được bằng mạch hai cấp, vì để đảm bảo lúc nào cũng có thể thực hiện được bằng mạch hai cấp thì các IC phải có số đầu vào không bị hạn chế. Đó là một yêu cầu không phải lúc nào cũng đáp ứng được trên thực tế, vì do yếu tố kinh tế và công nghệ, các IC chỉ được sản xuất với số lượng đầu vào hạn chế, thông thường từ 2 đến 5 đầu vào. Các IC có số đầu vào lớn hơn 5 tuy có nhưng không nhiều.

Chúng ta sẽ xét phương pháp tổng hợp cấu trúc mạch hai cấp đối với trường hợp một hàm và nhiều hàm.

4.3.1. Trường hợp một hàm

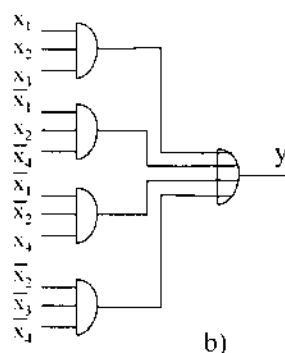
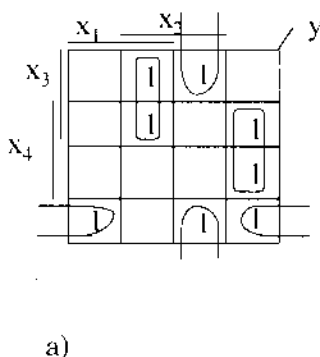
Trường hợp thiết bị cần thiết kế chỉ thực hiện chức năng của một hàm logic, phương pháp tổng hợp chung nhất sẽ là:

- + Phát biểu chức năng của mạch và ghi lại dưới một dạng chuẩn nào đó như bảng giá trị thực, giản đồ thời gian hay dưới dạng một biểu thức đại số.
- + Rút ra hàm logic và đơn giản để có dạng tuyến chuyển tối thiểu hoặc dạng hội chuẩn tối thiểu.
- + Căn cứ vào hệ phần tử logic sử dụng để xây dựng mạch, biến đổi kết quả trên đây về dạng cần thiết:
 - Nếu thực hiện bằng các mạch họ {VA, HOAC, KH} (AND, OR, NOT) thì không cần biến đổi gì thêm, thực hiện mạch trực tiếp từ dạng tuyến chuẩn tối thiểu hoặc dạng hội chuẩn tối thiểu.
 - Nếu thực hiện chỉ bằng các phần tử loại VA-KH (NAND) hoặc loại HOAC-KH (NOR) hay hỗn hợp của một số loại phần tử thì cần biến đổi sang dạng thích hợp theo cách sẽ chỉ ra ở mục tiếp theo.
 - Thực hiện kỹ thuật, đảm bảo sự phối hợp trở kháng, sự phối hợp về mức tín hiệu... nếu cần.

a) Thực hiện bằng các phần tử logic họ {VA, HOAC, KH} (AND, OR, NOT)

Thí dụ 4.1: Cần thực hiện hàm chức năng được cho dưới dạng sau:

$$y = f(0, 1, 3, 4, 6, 8, 14, 15), n = 4; b = 10.$$



Hình 4.8. Bảng Karnaugh dùng đơn giản hàm (a) và mạch thực hiện (b)

+ Đơn giản hàm dùng phương pháp Karnaugh như hình 4.8a.

Sau khi đưa hàm đã cho lên bảng Karnaugh, áp dụng algôrit đơn giản hàm, ta được dạng tuyến chuẩn tối thiểu như sau:

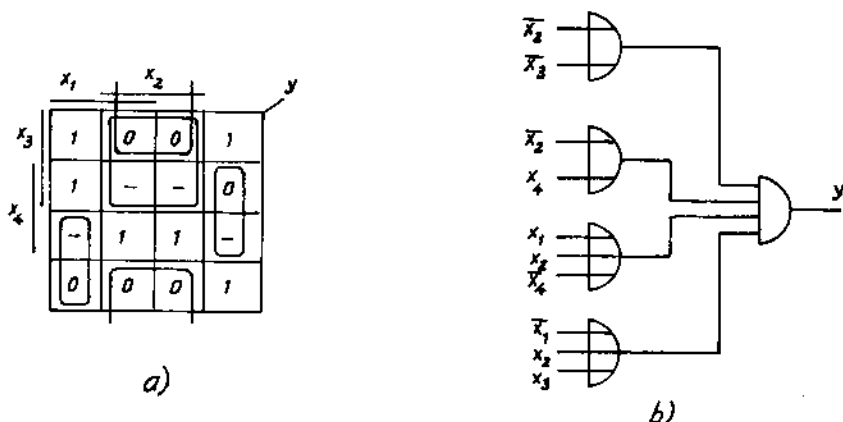
$$y = x_1x_2x_3 + \bar{x}_1x_2\bar{x}_4 + \bar{x}_1\bar{x}_2x_4 + \bar{x}_2\bar{x}_3\bar{x}_4$$

Mạch hai cấp thực hiện dạng tuyến chuẩn tối thiểu này dùng các phần tử mạch loại VA, HOAC, KH cho trên hình 4.8b. Để thực hiện hàm này chúng ta cần có 4 mạch VA ba đầu vào, một mạch HOAC bốn đầu vào. Để thực hiện một hàm cụ thể nào đó ở dạng hai cấp, số lượng các mạch và số đầu vào cần có của chúng có thể rất khác nhau. Chính vì vậy thực tế không phải lúc nào cũng có thể thực hiện bằng mạch hai cấp được.

Thí dụ 4.2: Thực hiện hàm cho sau đây bằng mạch hai cấp dùng các phần tử logic họ VA, HOAC, KH dưới dạng hội chuẩn tối thiểu.

$$y = \begin{cases} f^1(0, 2, 5, 10, 11, 13); & n = 4; b = 10 \\ f^-(1, 7, 9, 15) \end{cases}$$

Trước hết biểu diễn hàm này lên bảng Karnaugh – hình 4.9a.



Hình 4.9. Bảng Karnaugh dùng đơn giản hàm (a) và mạch thực hiện (b)

Áp dụng algôrit đơn giản hàm dùng bảng Karnaugh, ta có dạng hội chuẩn tối thiểu như sau:

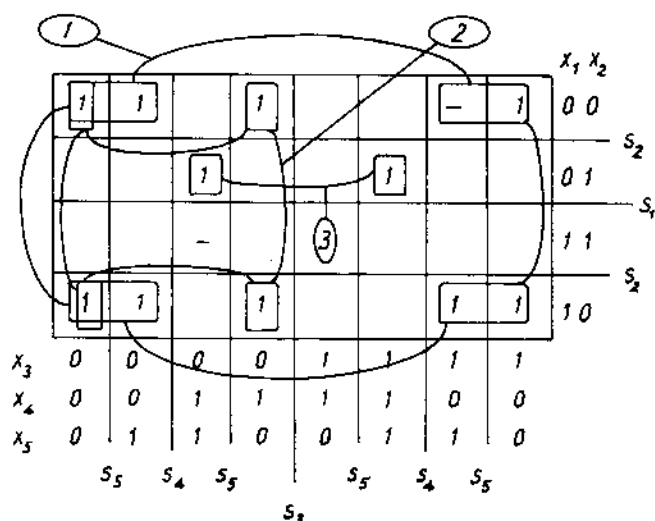
$$y = (\bar{x}_2 + \bar{x}_3)(\bar{x}_2 + x_4)(x_1 + x_2 + \bar{x}_4)(\bar{x}_1 + x_2 + x_3)$$

Thực hiện hàm này chúng ta cần bốn mạch HOAC, trong đó hai mạch có hai đầu vào, hai mạch còn lại - ba đầu vào và một mạch VA bốn đầu vào. Mạch thực hiện cho trên hình 4.9b.

Thí dụ 4.3: Đơn giản và thực hiện hàm logic 5 biến sau đây bằng mạch hai cấp dùng họ {VA, HOAC, KH}.

$$y = \begin{cases} f^1(00000, 00001, 00010, 00100, 01011, 01111, 10000, 10001, 10010, 10101, 10100) \\ f^-(00101, 11011) \end{cases}$$

Đưa lên bảng đối xứng và thực hiện đơn giản hàm, ta được ba implicăng đơn giản thực thụ (1), (2) và (3) - hình 4.10.



Hình 4.10. Đơn giản hàm dùng bảng đối xứng

Do đó dạng tuyến chuẩn tối thiểu của hàm:

$$y = \bar{x}_2 \bar{x}_4 + \bar{x}_2 \bar{x}_3 \bar{x}_5 + \bar{x}_1 x_2 x_4 x_5$$

Mạch thực hiện cho trên hình 4.11.

b) Thực hiện dùng họ VA-KH và HOAC-KH (NAND và NOR)

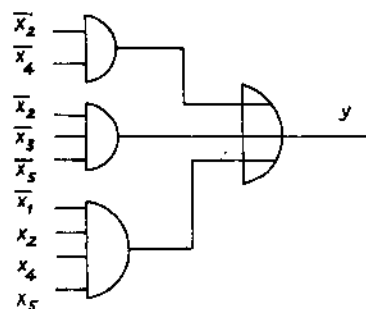
Trong trường hợp này ta cần nhận được dạng toàn thuật toán VA-KH hoặc HOAC-KH. Đã có một số công trình nghiên cứu nhằm thực hiện đơn giản hàm trong họ hàm đủ và tối thiểu {VA-KH} và {HOAC-KH} nhưng không có kết quả. Nói đúng hơn, các phương pháp đó đều phức tạp và kém hiệu quả. Chính vì vậy trên thực tế người ta chấp nhận, muốn có dạng hai cấp dùng mạch VA-KH hoặc HOAC-KH, người ta đơn giản hàm trong hệ hàm {VA, HOAC, KH}, sau đó biến đổi kết quả từ dạng tuyến chuẩn tối thiểu hoặc dạng hội chuẩn tối thiểu thành dạng toàn thuật toán VA-KH hoặc toàn thuật toán HOAC-KH.

+ Nhận dạng VA-KH hai cấp (NAND - NAND):

- Đơn giản hàm để nhận dạng tuyến chuẩn tối thiểu.

- Biến đổi dạng tuyến chuẩn tối thiểu thành dạng VA-KH hai cấp theo hai cách:

* Dùng quy luật $y = \bar{\bar{y}}$ và định luật De Morgan để biến đổi.



Hình 4.11. Mạch thực hiện hàm

* Thực hiện algôrit chuyển đổi sau đây:

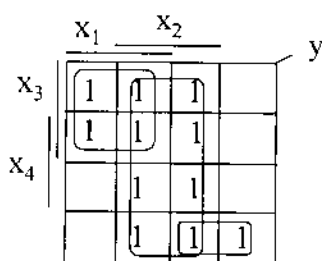
“Xuất phát từ dạng tuyến chuẩn tối thiểu, thay tất cả phép tính cộng và nhân logic bằng phép NAND “|”. Số hạng nào chỉ có một biến thì lấy phủ định”.

Thí dụ 4.4: Thực hiện mạch có chức năng biểu diễn bằng hàm logic sau đây:

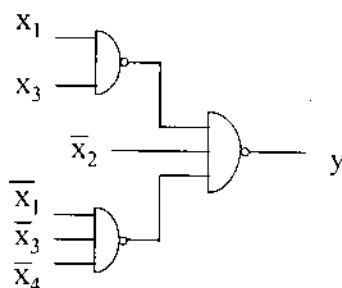
$$y = f(0, 4, 5, 6, 7, 10, 11, 12, 13, 14, 15); n = 4, b = 10$$

+ Đơn giản hàm bằng phương pháp Karnaugh (hình 4.12) ta có:

$$y = x_1x_3 + x_2 + \bar{x}_1\bar{x}_3\bar{x}_4$$



a)



b)

Hình 4.12. Bảng Karnaugh dùng đơn giản hàm (a) và mạch thực hiện (b)

* Biến đổi về dạng NAND dùng hai lần phủ định, ta có:

$$y = x_1x_3 + x_2 + \bar{x}_1\bar{x}_3\bar{x}_4 = \overline{\overline{x_1x_3} \cdot \overline{x_2} \cdot \overline{x_1x_3x_4}} = (x_1 | x_3) | \bar{x}_2 | (\bar{x}_1 | \bar{x}_3 | \bar{x}_4)$$

Mạch thực hiện cho trên hình 4.12b.

* Biến đổi bằng cách áp dụng trực tiếp algôrit trên ta cũng có ngay kết quả như trên, vì trong trường hợp này chỉ có một số hạng x_2 chỉ gồm một biến nên khi chuyển sang dạng NAND x_2 phải lấy phủ định. Đối với các toán hạng khác các biến không có gì thay đổi so với dạng tuyến chuẩn tối thiểu.

+ Nhận dạng HOAC - KH hai cấp (NOR-NOR)

- Đơn giản hàm và nhận dạng hội chuẩn tối thiểu và dùng định luật De Morgan để biến đổi.

- Nhận dạng NOR - NOR tối thiểu bằng cách biến đổi theo một trong hai cách sau đây:

* Lấy hai lần phủ định của dạng hội chuẩn tối thiểu và dùng định luật De Morgan để biến đổi.

* Thực hiện algôrit sau đây: Xuất phát từ dạng hội chuẩn tối thiểu, thay tất cả các phép nhân và cộng logic bằng phép NOR “ $\downarrow$ ”. Thừa số nào chỉ có một biến thì lấy phủ định.

Thí dụ 4.5: Thực hiện mạch có chức năng được mô tả bằng hàm logic sau đây:

$$y = f(1, 2, 3, 8, 9); n = 4, b = 10$$

Để đơn giản hàm này và nhận dạng hội chuẩn tối thiểu, ta căn cứ vào các giá trị 0 của hàm, từ đó nhận dạng hội chuẩn tối thiểu. Sau khi đưa hàm lên bảng Karnaugh và thực hiện đơn giản để nhận dạng hội chuẩn tối thiểu (hình 4.13a), ta được:

$$y = (\bar{x}_1 + \bar{x}_3)(\bar{x}_2)(x_1 + x_3 + x_4)$$

Sau đó áp dụng lấy hai lần phủ định ta được:

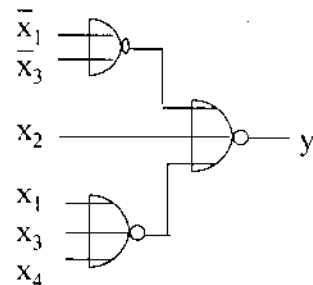
$$\begin{aligned} y &= \overline{(\bar{x}_1 + \bar{x}_3)(\bar{x}_2)(x_1 + x_3 + x_4)} = \overline{(\bar{x}_1 + \bar{x}_3)} + \overline{(\bar{x}_2)} + \overline{(x_1 + x_3 + x_4)} \\ &= (\bar{x}_1 \downarrow \bar{x}_3) \downarrow x_2 \downarrow (x_1 \downarrow x_3 \downarrow x_4) \end{aligned}$$

Mạch thực hiện cho trên hình 4.13b.

a)

	x_1	x_2	
x_3	0	0	0
x_4	0	0	0
		0	0
		0	0

a)



b)

Hình 4.13. Bảng Karnaugh dùng đơn giản hàm (a) và mạch thực hiện (b)

Nếu hàm cho dưới dạng tuyến chuẩn hoặc dạng bất kỳ, để có dạng hội chuẩn tối thiểu, ta làm như sau:

- Biến đổi hàm về dạng tuyến chuẩn.
- Đưa hàm lên bảng Karnaugh và tìm dạng tuyến chuẩn tối thiểu của hàm $\bar{y}$.
- Tìm dạng hội chuẩn bằng cách lấy phủ định của hàm $\bar{y}$.
- Áp dụng phương pháp biến đổi từ dạng hội chuẩn thành dạng NOR - NOR theo một trong hai cách đã nêu trên đây.

Thí dụ 4.6: Cho hàm:

$$y = x_4 (\bar{x}_1 + x_1 x_2 (\bar{x}_2 + x_3 + \bar{x}_4)) + \bar{x}_1 \bar{x}_2 \bar{x}_3$$

+ Bỏ dấu ngoặc ta được:

$$y = \bar{x}_1 x_4 + x_1 x_2 x_3 x_4 + \bar{x}_1 \bar{x}_2 \bar{x}_3$$

+ Đưa lên bảng Karnaugh và đơn giản (hình 4.14a).

+ Dạng tuyến chuẩn tối thiểu của hàm $\bar{y}$ là:

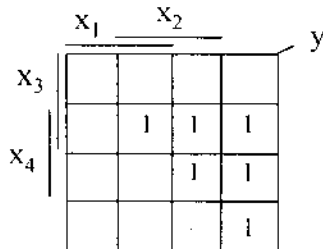
$$\bar{y} = x_1 \bar{x}_3 + x_1 \bar{x}_2 + x_3 \bar{x}_4 + x_2 \bar{x}_4$$

+ Dạng hội chuẩn tối thiểu của hàm y nhận được bằng cách lấy phủ định của hàm $\bar{y}$:

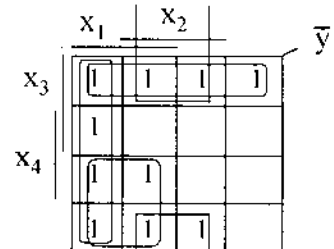
$$y = \overline{x_1 \bar{x}_3 + x_1 \bar{x}_2 + x_3 \bar{x}_4 + x_2 \bar{x}_4} = (\bar{x}_1 + x_3)(\bar{x}_1 + x_2)(\bar{x}_3 + x_4)(\bar{x}_2 + x_4)$$

+ Chuyển thành dạng NOR-NOR theo algôrit đã nêu trên đây:

$$y = (\bar{x}_1 \downarrow x_3) \downarrow (\bar{x}_1 \downarrow x_2) \downarrow (\bar{x}_3 \downarrow x_4) \downarrow (\bar{x}_2 \downarrow x_4)$$



a)



b)

Hình 4.14. Biểu diễn hàm (a) và hàm đảo của hàm đã cho (b)

Mạch NOR-NOR hai cấp thực hiện hàm trên đây cho trên hình 4.15.

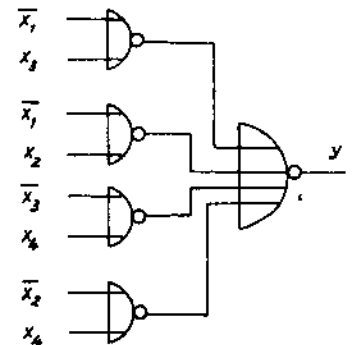
c. Xây dựng mạch dùng hỗn hợp các mạch logic loại:

+ HOAC - KH và mạch HOAC

Cách làm như sau:

- Đơn giản hàm để nhận dạng tuyến chuẩn tối thiểu của hàm đã cho.

- Các phép nhân logic "." thay bằng phép HOAC-KH " $\downarrow$ ". Trong đó tất cả các biến là thừa số của các phép VA phải lấy phủ định.



Hình 4.15. Thực hiện mạch bằng NOR

Thí dụ 4.7:

Giả sử kết quả của bước đơn giản hàm cho ta hai hàm sau đây:

$$y_1 = x_1 + \bar{x}_2 \bar{x}_4 \bar{x}_5$$

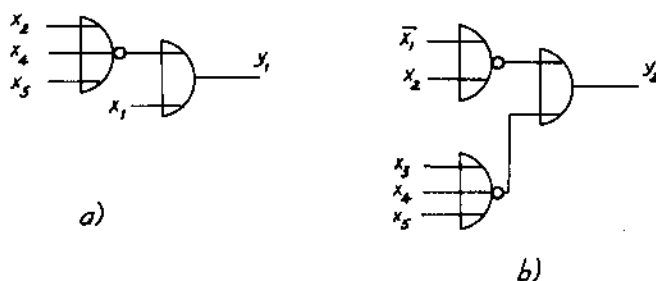
$$y_2 = x_1 \bar{x}_2 + \bar{x}_3 \bar{x}_4 \bar{x}_5$$

Áp dụng bước thứ hai ta được dạng HOAC-KH và HOAC như sau:

$$y_1 = x_1 + (x_3 \downarrow x_4 \downarrow x_5)$$

$$y_2 = (\bar{x}_1 \downarrow x_2) + (x_3 \downarrow x_4 \downarrow x_5)$$

Mạch hai cấp thực hiện hai hàm này cho trên hình 4.16a và 4.16b.



Hình 4.16. Thực hiện mạch dùng hỗn hợp các loại IC

+ VA-KH và mạch VA

Dạng này nhận được bằng cách:

- Đơn giản hàm để nhận dạng hội chuẩn tối thiểu.
- Thay tất cả các phép cộng bằng phép VA-KH, trong đó các số hạng của tổng phải lấy phủ định.

Thí dụ 4.8:

Giả sử sau bước đơn giản hàm ta được dạng hội chuẩn tối thiểu như sau:

$$y_1 = (x_1 + x_2)(x_3 + \bar{x}_4)$$

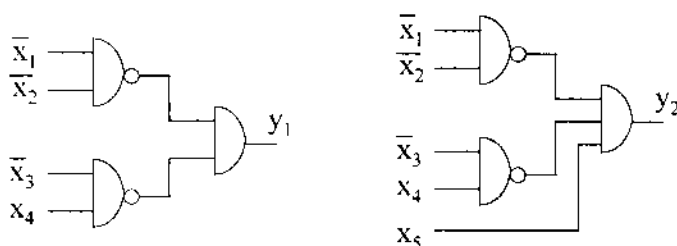
$$y_2 = (x_1 + x_2)(x_3 + \bar{x}_4)x_5$$

Dùng quy luật biến đổi trên đây, ta có dạng VA-KH-VA như sau:

$$y_1 = (\bar{x}_1 | \bar{x}_2)(\bar{x}_3 | x_4)$$

$$y_2 = (\bar{x}_1 | \bar{x}_2)(\bar{x}_3 | x_4)x_5$$

Mạch hai cấp thực hiện hai hàm này cho trên hình 4.17.



Hình 4.17. Thực hiện mạch dùng hỗn hợp các loại IC VA-KH và VA

Khái quát lại, bằng cách biến đổi và thực hiện ở các cấp biến đổi trung gian, ta có thể thực hiện một hàm logic đã cho dưới những dạng rất khác nhau, gồm các loại phần tử logic có chức năng khác nhau. Sự linh hoạt này tùy thuộc vào khả năng vận dụng một cách nhuần nhuyễn các quy luật biến đổi của người thiết kế.

4.3.2. Trường hợp một hệ hàm

Trong thực tế, nhiều mạch số tổ hợp thực hiện không phải một hàm mà nhiều hàm. Đó là các thiết bị có nhiều đầu ra, mỗi đầu ra là một hàm và có số biến vào như nhau. Vấn đề đặt ra khi thiết kế các mạch này cũng là tiết kiệm linh kiện và từ đó kéo theo các đặc tính kỹ thuật khác như độ tin cậy và tốc độ cao hơn...

Vấn đề đặt ra được giải quyết chủ yếu trong khâu đơn giản hàm, tức là áp dụng các phương pháp đơn giản hàm kết hợp để có dạng tuyến chuẩn tối thiểu và dạng hội chuẩn tối thiểu chung đối với cả hệ hàm và đối với từng nhóm con các hàm của hệ. Nói cách khác, trong đơn giản hàm kết hợp, ta coi các hàm trong hệ là có mối quan hệ chặt chẽ. Trong đó cần cố gắng biểu diễn hàm này qua hàm khác. Bằng cách đó sẽ làm cho mạch được thiết kế có độ đơn giản cao hơn. Các phương pháp này đã được xét tới trong chương đơn giản các hàm lôgic.

Thí dụ, có hai hàm y_1 và y_2 , trong đó chúng đều nhận giá trị 1 đối với các tập giá trị biến vào X_i , $i = \overline{1, k}$. Ngoài ra, y_1 còn nhận giá trị 1 trên các tập giá trị biến khác sau đây: X_{k+m} , $m = \overline{1, l}$. Trong đó $k + l \leq N = 2^n$; n là số biến của các hàm. Khi đó có thể viết:

$$y_1 = y_2 + X_{k+1} + \dots + X_{k+l}$$

Trong trường hợp biểu diễn như thế này, các phần tử lôgic dùng để thực hiện hàm y_2 đồng thời cũng được dùng để thực hiện hàm y_1 . Rõ ràng tiết kiệm linh kiện hơn trong trường hợp thực hiện từng hàm riêng rẽ.

Sau đây là cách thức thực hiện các ý muốn đã trình bày trên đây.

1) Tìm dạng tuyến chuẩn tối thiểu của mỗi hàm trong hệ

Một cách độc lập theo một trong các phương pháp đơn giản hàm lôgic đã trình bày trước đây. Khi lắp ráp mạch, những phần giống nhau của tất cả các hàm hoặc của một số hàm được thực hiện bằng các phần tử mạch chung cho chúng.

Cách làm này đơn giản và dễ thực hiện, có thể áp dụng đối với hệ hàm không lớn với số biến vào cũng không lớn. Nó có nhược điểm là dễ bỏ sót các phần có thể là chung cho một số hàm do cách đơn giản hoàn toàn độc lập.

Thí dụ 4.9:

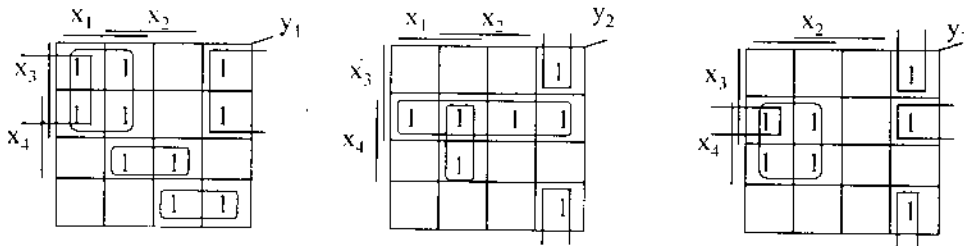
Thiết kế thiết bị số thực hiện hệ hàm sau đây:

$$y_1 = f(0, 2, 3, 4, 5, 10, 11, 13, 14, 15)$$

$$y_2 = f(0, 2, 3, 7, 11, 13, 15)$$

$$y_3 = f(0, 2, 3, 9, 11, 13, 15).$$

Thực hiện đơn giản hàm một cách độc lập, ta dùng ba bảng Karnaugh như hình 4.18.



Hình 4.18. Bảng Karnaugh đơn giản hệ ba hàm

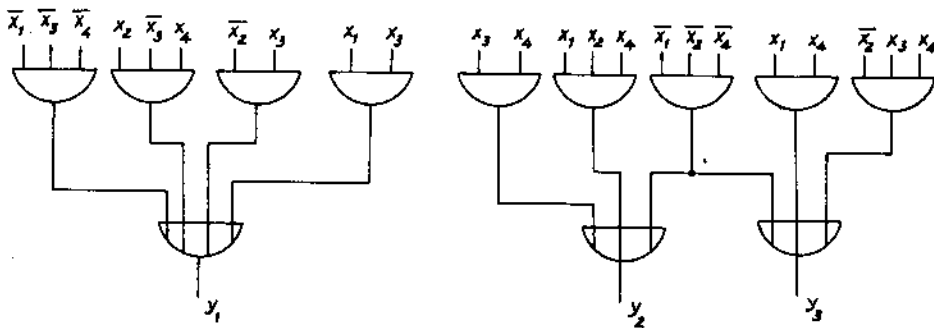
Sau khi đơn giản ta được các dạng tuyến chuẩn tối thiểu như sau:

$$y_1 = x_1x_3 + \bar{x}_2x_3 + x_2\bar{x}_3x_4 + \bar{x}_1\bar{x}_3\bar{x}_4$$

$$y_2 = x_3x_4 + x_1x_2x_4 + \bar{x}_1\bar{x}_2\bar{x}_4$$

$$y_3 = x_1x_4 + \bar{x}_2x_3x_4 + \bar{x}_1\bar{x}_2\bar{x}_4$$

Mạch cấu trúc hai cấp dùng các phân tử VA, HOAC và KH, trong đó phân tử $\bar{x}_1\bar{x}_3\bar{x}_4$ của hai hàm y_2 và y_3 được thực hiện bởi cùng một mạch VA, cho trên hình 4.19.



Hình 4.19. Thực hiện kết hợp ba hàm đã đơn giản

2) Dùng phương pháp phân hoạch

Nội dung của phương pháp phân hoạch là tách phần chung cho cả hệ gồm m hàm ($y_1, y_2, \dots, y_m$) có n biến mà chúng đều nhận giá trị 1, gọi là phần φ . Phần còn lại của các hàm, ký hiệu là y_j^* , $j = \overline{1, m}$, mà $y_j^* = y_j \setminus \varphi$. Khi đơn giản hệ hàm, phần φ được coi là một hàm riêng biệt có n biến và được đơn giản theo một trong các phương pháp đơn giản hàm đã trình bày trên đây. Các phần y_j^* được đơn giản như hàm n biến, nhưng đối với chúng phần φ được coi như phần mà hàm đó bất định. Kết quả đã đơn giản của mỗi hàm y_j là tổng logic của phần φ và phần y_j^* đã được đơn giản, tức là:

$$y_j = \varphi + y_j^*$$

Sơ đồ thực hiện cho trên hình 4.20.

Phương pháp này thích hợp nhất đối với hệ các hàm mà có nhiều tập giá trị biến các hàm đều nhận giá trị 1.

Để minh họa cho phương pháp này ta xét thí dụ sau đây.

Thí dụ 4.10:

Thiết kế mạch thực hiện chức năng cho bởi ba hàm logic sau đây:

$$y_1 = f(0, 2, 3, 4, 5, 10, 11, 13, 14, 15); n = 4, b = 10$$

$$y_2 = f(0, 2, 3, 9, 11, 13, 15)$$

$$y_3 = f(0, 2, 3, 7, 11, 13, 15)$$

Ba hàm đã cho có phần chung là φ , cho cả ba hàm như sau:

$$\varphi = f(0, 2, 3, 11, 13, 15)$$

Phần còn lại của các hàm:

$$y_1^* = f(4, 5, 10, 14); \bar{f}(0, 2, 3, 11, 13, 15) = \varphi$$

$$y_2^* = f(9); \bar{f}(0, 2, 3, 11, 13, 15) = \varphi$$

$$y_3^* = f(7); \bar{f}(0, 2, 3, 11, 13, 15) = \varphi$$

Thực hiện đơn giản ba hàm này và hàm φ , dùng một trong các phép đơn giản hàm đã trình bày trong chương III, ta được:

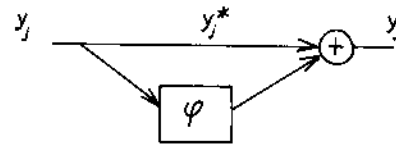
$$\varphi = x_1 x_2 x_4 + \bar{x}_2 x_3 x_4 + \bar{x}_1 \bar{x}_2 \bar{x}_4$$

$$y_1^* = x_1 x_3 + \bar{x}_1 x_2 \bar{x}_3$$

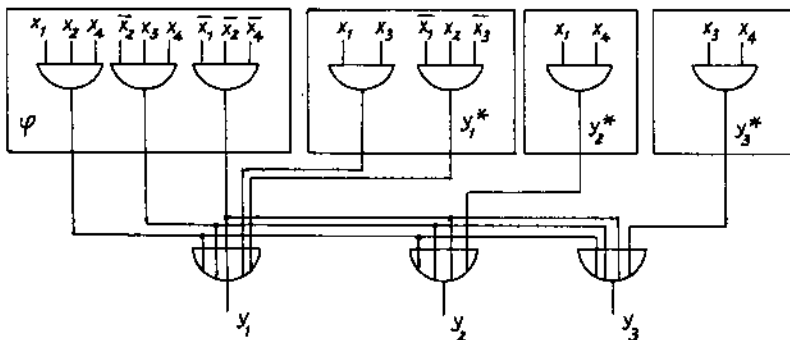
$$y_2^* = x_1 x_4$$

$$y_3^* = x_3 x_4$$

Mạch thực hiện cho hệ hàm này cho trên hình 4.21.



Hình 4.20. Mô hình tách hàm φ



Hình 4.21. Thực hiện hàm theo kiểu đơn giản phân hoạch

3) Dùng phương pháp đồng thời đơn giản cả hệ hàm

Phương pháp này cho phép giảm việc chọn theo kinh nghiệm, giảm những sai sót do sơ suất và có thể áp dụng các công cụ nghiên cứu hiện đại như các chương trình chuyên dụng của máy tính để tự động hoá quá trình đơn giản hệ hàm.

Thí dụ 4.11:

Thiết kế thiết bị số có chức năng cho bởi hệ ba hàm logic sau đây:

$$y_1 = f^1(0, 2, 7, 10); f^-(5, 12)$$

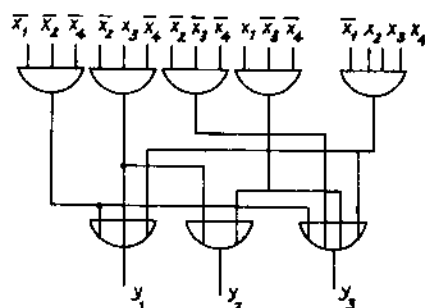
$$y_2 = f^1(2, 11, 15); f^-(3, 7, 8, 10)$$

$$y_3 = f^1(2, 7, 8, 11, 15); f^-(0, 9, 13), n = 4, b = 10$$

Dùng phương pháp đơn giản kết hợp hệ hàm, ta được phủ tối thiểu như sau:

x_1	x_2	x_3	x_4	y_1	y_2	y_3
0	0	-	1	1	0	-
-	0	1	0	1	1	0
0	1	1	1	1	0	1
-	0	0	0	0	0	1
1	-	0	0	0	1	1

Mạch thực hiện hệ ba hàm trên cho trên hình 4.22.



Hình 4.22. Thực hiện hệ hàm sau khi đã đơn giản kết hợp

4.4. TỔNG HỢP CẤU TRÚC MẠCH NHIỀU CẤP

Như đã nói trên đây, để luôn luôn có thể thực hiện mạch bằng mạch hai cấp, đòi hỏi số đầu vào của các loại IC phải không bị hạn chế. Đó là một đòi hỏi trên thực tế không phải lúc nào cũng đáp ứng được, nhất là khi số biến nhiều, dẫn đến dạng tuyến chuẩn tối thiểu (dạng hội chuẩn tối thiểu) có nhiều số hạng (thừa số) và mỗi số hạng (thừa số) lại có nhiều biến. Thực tế ngành công nghiệp vi điện tử chỉ sản xuất các phần tử logic có số đầu vào hạn chế, thông thường từ 1 đến 5 đầu vào. Các loại vi mạch có số đầu vào lớn thường sản xuất rất hạn chế do không có tính kinh tế trong sản xuất và trong sử dụng.

Khi số đầu vào của IC không đủ khả năng để thực hiện mạch dưới dạng hai cấp, ta phải chuyển sang thực hiện ở dạng nhiều cấp.

Ngoài lý do số đầu vào hạn chế, khi khắc phục khả năng chịu tải hạn chế của vi mạch cũng thường phải chuyển thành mạch nhiều cấp. Dưới đây chúng ta sẽ xét hai trường hợp này.

4.4.1. Tổng hợp khi số đầu vào của các phần tử logic bị hạn chế

a) Trường hợp một hàm

Khi số lượng đầu vào của các IC không đủ để thực hiện mạch theo mạch hai cấp, ta cần đặt thừa số chung để từng bước làm giảm số đầu vào cần thiết của các IC. Vấn đề đặt ra là nhóm các số hạng như thế nào cho có lợi nhất, với nghĩa là để có mạch có ít cấp nhất và cần ít phần tử logic nhất có thể được. Để đạt được mục đích trên đây, với số đầu vào của mỗi phần tử logic là “k”, ta làm như sau:

+ Chia số số hạng của dạng tuyến chuẩn (tối thiểu) thành k cụm, sao cho có khả năng đặt thừa số chung đối với từng cụm và sao cho số lượng số hạng trong mỗi cụm càng như nhau càng tốt.

+ Đặt thừa số chung của các số hạng trong mỗi cụm để giảm bớt yêu cầu về số đầu vào của phần tử logic. Trong trường hợp không đặt được thừa số chung thì phân nhóm tiếp để xử lý.

Hai bước trên đây có thể phải làm nhiều cấp cho tới khi nào số đầu vào của các phần tử logic đủ để thực hiện hàm đã cho.

Thí dụ 4.12:

Thực hiện hàm y_1 sau đây dùng các phần tử logic hai đầu vào:

$$y_1 = x_1x_2x_3 + \bar{x}_1x_2x_4 + \bar{x}_1\bar{x}_2x_3 + x_1\bar{x}_2x_4$$

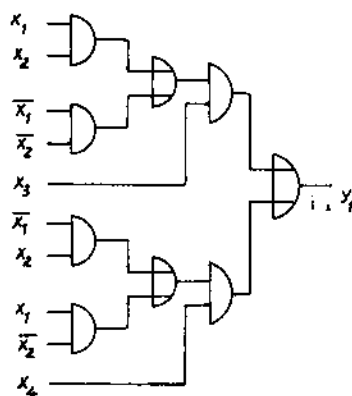
Nếu thực hiện bằng mạch hai cấp, ta cần bốn mạch VA ba đầu vào và một mạch HOAC bốn đầu vào. Để thực hiện bằng mạch chỉ có hai đầu vào, ta phân nhóm các số hạng của dạng tuyến chuẩn thành hai nhóm với mục đích đặt thừa số chung như sau:

$$y_1 = x_3(x_1x_2 + \bar{x}_1\bar{x}_2) + x_4(x_1\bar{x}_2 + \bar{x}_1x_2)$$

Thực hiện hàm này ta cần 6 mạch VA hai đầu vào, 3 mạch cộng hai đầu vào và số cấp của mạch không phải là 2 mà là 4. Trong đó không có phần tử logic nào cần số đầu vào lớn hơn hai. Mạch thực hiện cho trên hình 4.23.

Dạng giải tích của hàm sau khi đặt thừa số chung gọi là dạng dấu ngoặc. Dạng dấu ngoặc là dạng ứng với thiết bị số nhiều cấp.

Số lượng các dạng dấu ngoặc khác nhau của hàm có n biến rất lớn. Vì vậy việc tìm dạng dấu ngoặc đơn giản nhất của một hàm phức tạp hơn nhiều so với việc tìm dạng tuyến chuẩn tối thiểu. Tới nay chưa có phương pháp nào giải quyết nhiệm vụ này một cách có hiệu quả.



Hình 4.23. Thực hiện hàm y_1 dùng phần tử VA, HOAC, KH hai đầu vào

Thí dụ 4.13:

Cho hàm dưới dạng tuyến chuẩn sau đây:

$$y = \bar{x}_1 x_2 x_3 + x_3 x_4 + x_1 \bar{x}_2 x_4 + x_1 \bar{x}_2 x_3$$

Nếu ta nhóm các số hạng thành nhóm không đều nhau và đặt thừa số chung như sau:

$$y = (\bar{x}_1 x_2 + x_4 + x_1 \bar{x}_2) x_3 + (x_1 \bar{x}_2) x_4$$

Trường hợp này số phần tử VA là 5, HOAC là 3, tổng cộng là 8. Cấp của mạch là 5.

Nếu ta nhóm các số hạng sao cho số lượng thành phần ngang nhau và đặt thừa số chung, thí dụ như sau:

$$y = x_3 (\bar{x}_1 x_2 + x_1 \bar{x}_2) + x_4 (x_1 + x_1 \bar{x}_2)$$

thì số mạch VA hai đầu vào là 5, số mạch HOAC hai đầu vào là 3, tổng cộng là 8, nhưng số cấp là 4, nhỏ hơn trường hợp trên một đơn vị.

Thí dụ 4.14:

Thực hiện bằng mạch NAND hai đầu vào hàm sau đây:

$$y_1 = x_1 x_2 \bar{x}_4 + x_2 x_3 x_4 + \bar{x}_1 \bar{x}_2 x_4 + \bar{x}_1 \bar{x}_3 \bar{x}_4$$

Hai cách nhóm số hạng, đưa ra ngoài dấu ngoặc và sau đó chuyển đổi sang dạng NAND theo cách lấy hai lần phủ định sẽ cho hai kết quả như nhau về mặt đơn giản hàm:

$$y_1 = (\bar{x}_4 | ((x_1 | x_2) | (\bar{x}_1 | \bar{x}_3))) | (x_4 | ((x_2 | x_3) | (\bar{x}_1 | \bar{x}_2))) \quad \text{và}$$

$$y_1 = (x_2 | ((x_1 | \bar{x}_1) | (x_3 | x_4)) | ((\bar{x}_3 | \bar{x}_4) | (\bar{x}_2 | x_4)) \bar{x}_1)$$

Cả hai phương án đều có 4 cấp và số phần tử logic bằng 9.

Thí dụ 4.15:

Hàm trong thí dụ 4.14 cho dưới dạng hội chuẩn sau đây, hãy thực hiện hàm này dùng mạch logic hai đầu vào.

$$y = (x_1 + x_2)(x_1 + \bar{x}_3 + x_4)(\bar{x}_2 + x_3 + \bar{x}_4)$$

Ta có thể đưa về dạng dấu ngoặc để thực hiện bằng các phần tử logic hai đầu vào như sau:

$$y = ((\bar{x}_1 + x_2)((x_1 + \bar{x}_3) + x_4))((\bar{x}_2 + x_3) + \bar{x}_4)$$

Với cách biến đổi như trên, số cấp bằng 4, nhưng số lượng các phần tử chỉ bằng 7. Như vậy trong trường hợp này, dạng dấu ngoặc nhận từ dạng hội chuẩn lại ít tốn kém linh kiện hơn.

b) Trường hợp một hệ hàm

- Phân nhóm và đặt thừa số chung đối với hàm ϕ và các hàm y_j^* như đối với các hàm riêng lẻ. Trong đó nếu chọn hàm ϕ một cách thích hợp thì có thể giảm số đầu vào của các phần tử cần dùng.

Thí dụ 4.16:

Thực hiện hệ hàm sau đây, dùng các phần tử logic hai và ba đầu vào.

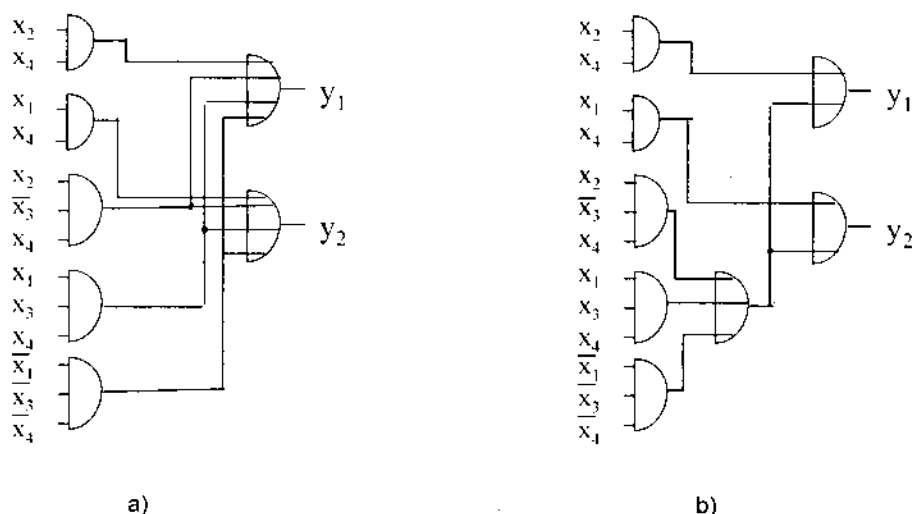
$$y_1 = x_2 \bar{x}_3 x_4 + x_1 x_3 x_4 + \bar{x}_1 \bar{x}_3 \bar{x}_4 + x_2 x_4$$

$$y_2 = x_2 \bar{x}_3 x_4 + x_1 x_3 x_4 + \bar{x}_1 \bar{x}_3 \bar{x}_4 + x_1 x_4$$

Nếu thực hiện chúng bằng mạch hai cấp phải có hai mạch HOAC 4 đầu vào (hình 4.24a). Nếu chọn phần chung:

$$\phi = x_2 \bar{x}_3 x_4 + x_1 x_3 x_4 + \bar{x}_1 \bar{x}_3 \bar{x}_4$$

thì có thể thoả mãn yêu cầu đặt ra (hình 4.24b).



Hình 4.24. Thực hiện hệ hàm dạng VA, HOAC, KH hai và ba đầu vào

4.4.2. Tổng hợp khi khả năng tải của các phần tử logic bị hạn chế

Khả năng chịu tải của các phần tử logic phụ thuộc vào công nghệ chế tạo chúng. Các họ IC chế tạo theo công nghệ TTL thường có khả năng chịu tải cao hơn so với các họ khác. Tuy vậy, khả năng chịu tải của IC nói chung là bị hạn chế. Trong nhiều trường hợp, khi tổng hợp mạch đôi khi đầu ra của một phần tử logic phải cấp tín hiệu cho nhiều mạch logic khác. Khi đó nếu số lượng mạch logic cần cung cấp tín hiệu vượt quá khả năng tải của IC thì mạch làm việc trong chế độ quá tải và không ổn định. Trong trường hợp này cần có biện pháp khắc phục để đảm bảo mạch làm việc ổn định. Biện pháp thường làm là thay đổi cấu trúc của mạch. Có hai cách biến đổi cấu trúc của mạch:

- Biến đổi cấu trúc logic.
- Biến đổi cấu trúc mạch mà không thay đổi cấu trúc logic.

a) Biến đổi cấu trúc logic

Đó là loại biến đổi dạng của hàm thực hiện chức năng của mạch nhằm làm cho các phần tử logic không bị quá tải. Để thay đổi được cấu trúc của mạch, ta dựa vào tính đa trị trong cách mô tả hàm logic. Một hàm logic có thể có nhiều dạng tối thiểu và càng có nhiều hơn dạng tuyến chuẩn đầy đủ. Chọn một dạng tuyến chuẩn tối thiểu hay đầy đủ phù hợp có thể tránh hiện tượng quá tải cho các phần tử logic.

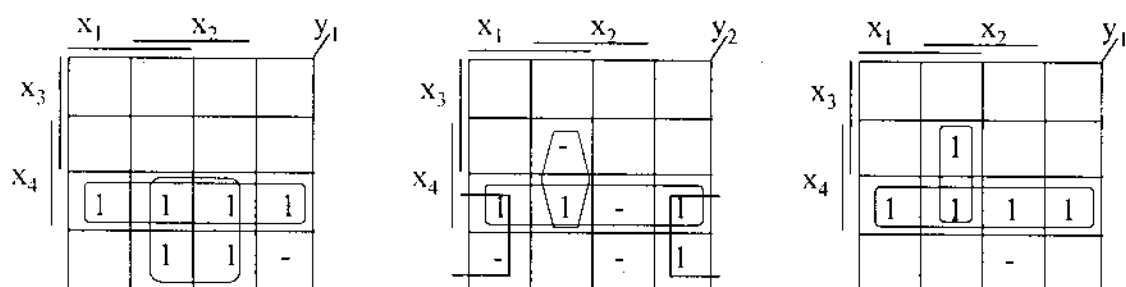
Thí dụ 4.17:

Thực hiện thiết bị số có chức năng được mô tả bằng hệ hàm sau đây, dùng các vi mạch có hệ số tải bằng 2:

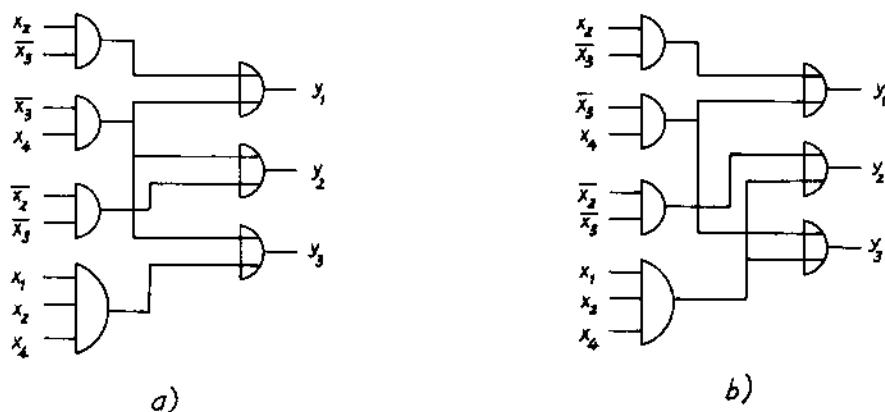
$$y_1 = f^1(1, 4, 5, 9, 12, 13); f^-(0)$$

$$y_2 = f^1(0, 1, 9, 13); f^-(4, 5, 8, 15)$$

$$y_3 = f^1(1, 5, 9, 13, 15); f^-(4); n = 4, b = 10$$



Hình 4.25. Đơn giản ba hàm logic đã cho



Hình 4.26. Thay đổi cấu trúc logic để đáp ứng yêu cầu chịu tải của IC

Đơn giản hệ hàm này bằng cách đưa lên bảng Karnaugh (hình 4.25), ta được dạng tuyến chuẩn tối thiểu của các hàm như sau:

1) Dạng NAND nhiều cấp nhận được từ dạng tuyến chuẩn nhiều cấp, tức dạng dấu ngoặc. Do đó để nhận dạng NAND nhiều cấp có thể làm theo hai cách sau:

+ Lấy hai lần phủ định dạng NAND (dạng dấu ngoặc) nhiều cấp.

+ Dùng algôrit:

- Xuất phát từ dạng NAND nhiều cấp.

- Thay tất cả các phép cộng “+” và nhân “.” logic bằng phép NAND “|”. Số hạng nào, dù ở cấp nào, chỉ có một biến thì lấy phủ định.

Thí dụ 4.18:

Biến đổi về dạng NAND nhiều cấp hàm sau đây bằng cách lấy hai lần phủ định:

$$y = (x_1 \bar{x}_2 + \bar{x}_1 x_2) x_3 + (x_1 \bar{x}_2 + \bar{x}_1 x_2) x_4$$

Lấy hai lần phủ định:

$$\begin{aligned} y &= \overline{\overline{(x_1 \bar{x}_2 + \bar{x}_1 x_2) x_3 + (x_1 \bar{x}_2 + \bar{x}_1 x_2) x_4}} \\ &= \overline{(((x_1 \bar{x}_2) + (\bar{x}_1 x_2)) x_3) \cdot (((\bar{x}_1 x_2) + (x_1 \bar{x}_2)) x_4)} \\ &= \overline{(((x_1 \bar{x}_2) + (\bar{x}_1 x_2)) | x_3) | (((\bar{x}_1 x_2) + (x_1 \bar{x}_2)) | x_4)} \end{aligned}$$

Vậy:

$$y = (((x_1 | \bar{x}_2) | (\bar{x}_1 | x_2)) | x_3) | (((\bar{x}_1 | x_2) | (x_1 | \bar{x}_2)) | x_4)$$

Như vậy cuối cùng chúng ta đã có dạng NAND nhiều cấp.

Phương pháp dùng hai lần phủ định hàm đã cho khá phức tạp, nhất là đối với các hàm có nhiều biến. Để thuận tiện cho những biến đổi tương tự, người ta khái quát việc biến đổi bằng một algôrit biến đổi như trên đã trình bày.

Thí dụ 4.19:

Áp dụng algôrit chuyển đổi từ dạng tuyến chuẩn nhiều cấp sang dạng NAND nhiều cấp, thực hiện hàm sau đây bằng các mạch NAND hai và ba đầu vào:

$$y = \bar{x}_1 x_3 + x_2 x_3 + x_1 x_4 x_6 + \bar{x}_5 x_6 + x_7$$

Nhóm các số hạng:

$$y = (\bar{x}_1 + x_2) x_3 + (x_1 x_4 + \bar{x}_5) x_6 + x_7$$

Thay tất cả các phép nhân và cộng logic bằng phép NAND, các số hạng $\bar{x}_1$, x_2 , $\bar{x}_5$ và x_7 có một biến nên phải lấy phủ định. Ta có:

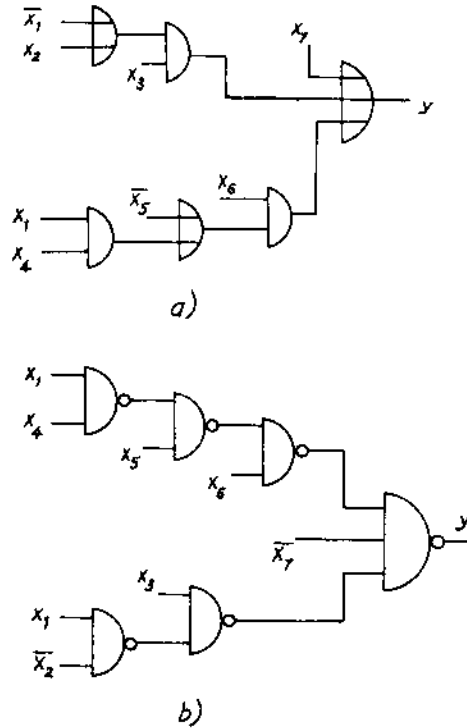
$$y = ((x_1 | \bar{x}_2) | x_3) | (((x_1 | x_4) | x_5) | x_6) | \bar{x}_7$$

Mạch thực hiện hàm đã cho dùng các mạch VA, HOAC, KH cho trên hình 4.28a và mạch NAND nhiều cấp của nó cho trên hình 4.28b.

2) Dạng NOR nhiều cấp nhận được từ dạng hội chuẩn nhiều cấp bằng hai cách:

+ Lấy hai lần phủ định hàm y ở dạng hội chuẩn nhiều cấp;

+ Thực hiện algôrit biến đổi từ dạng hội chuẩn nhiều cấp sang dạng NOR nhiều cấp: Thay tất cả các phép cộng và nhân logic bằng phép NOR, thừa số nào, dù ở cấp nào, chỉ có một biến thì phải lấy phủ định.



Hình 4.28. Mạch nhiều cấp dùng VA, HOAC, KH (a) và dùng mạch VA-KH (b)

Để minh họa cho các cách làm này, chúng ta xét các thí dụ sau đây:

Thí dụ 4.20:

Cho hàm dưới dạng hội chuẩn nhiều cấp. Hãy thực hiện bằng các mạch NOR hai đầu vào, dùng hai cách biến đổi trên đây.

$$y = (x_1(x_2 + \bar{x}_3) + x_4)(x_2x_3 + \bar{x}_5)$$

a) Lấy hai lần phủ định

$$\begin{aligned} y &= \overline{\overline{(x_1(x_2 + \bar{x}_3) + x_4)(x_2x_3 + \bar{x}_5)}} = \overline{\overline{(x_1(x_2 + \bar{x}_3) + x_4)} + \overline{\overline{(x_2x_3 + \bar{x}_5)}}} \\ &= \overline{(x_1(x_2 + \bar{x}_3) \downarrow x_4) \downarrow (x_2x_3 \downarrow \bar{x}_5)} = \overline{(x_1(x_2 + \bar{x}_3) \downarrow x_4) \downarrow (x_2x_3 \downarrow \bar{x}_5)} \\ &= \overline{(\bar{x}_1 + (\bar{x}_2 + \bar{x}_3) \downarrow x_4) \downarrow ((\bar{x}_2 + \bar{x}_3) \downarrow \bar{x}_5)} \\ &= ((\bar{x}_1 \downarrow (x_2 \downarrow \bar{x}_3)) \downarrow x_4) \downarrow ((\bar{x}_2 \downarrow \bar{x}_3) \downarrow x_5) \end{aligned}$$

Biểu thức cuối cùng chứa toàn phép NOR.

b) Dùng *algorit*: Thay tất cả các phép cộng và nhân logic trong biểu thức của y bằng phép NOR. Các thừa số x_1 đầu tiên và x_2 và x_3 là hai thừa số tham gia vào tích x_2x_3 , mỗi thừa số chỉ có một biến nên chúng phải lấy phủ định. Kết quả ta cũng được dạng toàn NOR:

$$y = ((\bar{x}_1 \downarrow (x_2 \downarrow \bar{x}_3)) \downarrow x_4) \downarrow ((\bar{x}_2 \downarrow \bar{x}_3) \downarrow x_5)$$

Mạch thực hiện cho trên hình 4.29a.

Thí dụ 4.21:

Tìm dạng HOAC - KH nhiều cấp của hàm sau đây:

$$y = x_1x_2 + \bar{x}_3\bar{x}_4$$

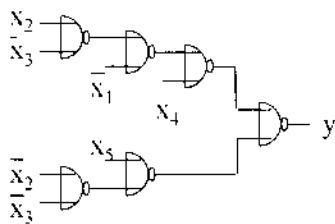
+ Biến đổi thành dạng hội chuẩn nhiều cấp:

$$y = (x_1x_2 + \bar{x}_3)(x_1x_2 + \bar{x}_4)$$

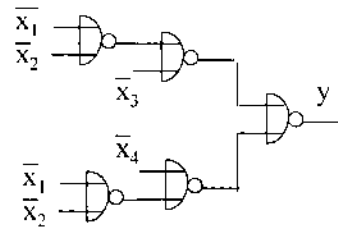
+ Áp dụng *algorit* chuyển đổi thành dạng HOAC-KH nhiều cấp:

$$y = ((\bar{x}_1 \downarrow \bar{x}_2) \downarrow \bar{x}_3) \downarrow ((\bar{x}_1 \downarrow \bar{x}_2) \downarrow \bar{x}_4)$$

Mạch thực hiện cho trên hình 4.29b.



a)



b)

Hình 4.29. Mạch dùng toàn mạch HOAC-KH hai đầu vào đối với các hàm y trong thí dụ 4.20 và 4.21

Bằng cách biến đổi khác đi một chút, ta có thể không cần sử dụng thể phủ định của biến như sau:

- Nhận dạng tuyến chuẩn nhiều cấp :

$$y = (x_1 + \bar{x}_3\bar{x}_4)(x_2 + \bar{x}_3\bar{x}_4)$$

- Chuyển thành dạng HOAC - KH nhiều cấp:

$$y = (x_1 \downarrow (x_3 \downarrow x_4)) \downarrow (x_2 \downarrow (x_3 \downarrow x_4))$$

Trong trường hợp này không cần dùng dạng phủ định của các biến mà vẫn thực hiện được hàm đã cho. Như vậy việc lựa chọn một dạng tuyến chuẩn nhiều cấp thích hợp là cần thiết. Đáng tiếc là cho tới nay chưa có một tài liệu nào nói về khả năng lựa chọn tối ưu này.

4.5. TỔNG HỢP THEO CÁCH PHÂN GIẢI HÀM

Chúng ta đã biết, một hàm logic bất kỳ đều có thể biểu diễn qua các hàm logic cơ bản. Nói cách khác, một hàm logic có thể coi là một hàm của các hàm logic đơn giản hơn. Tổng hợp theo phương pháp biểu diễn hàm đã cho qua các hàm logic đơn giản hơn gọi là tổng hợp theo phương pháp phân giải hàm.

Chúng ta đã biết, mọi hàm logic đều có thể viết dưới dạng:

$$f(x_1, x_2, \dots, x_i, \dots, x_n) = \bar{x}_i \cdot f_1(x_1, x_2, \dots, 0, \dots, x_n) + x_i \cdot f_2(x_1, x_2, \dots, 1, \dots, x_n)$$

$$\text{Hoặc } f(x_1, x_2, \dots, x_i, \dots, x_n) = (x_i + f_1(x_1, x_2, \dots, 0, \dots, x_n))(\bar{x}_i + f_2(x_1, x_2, \dots, 1, \dots, x_n))$$

Trong đó các hàm f_1 và f_2 là các hàm logic có $(n - 1)$ biến.

Mặt khác ta có thể viết:

$$f(x_1, x_2, \dots, x_i, \dots, x_n) = F(f_1, f_2, x_i) \quad (1)$$

Một cách tổng quát có thể viết:

$$f(x_1, x_2, \dots, x_n) = F_a(F_b(y_1, y_2, \dots, y_m), Z_1, Z_2, \dots, Z_p) \quad (2)$$

Trong đó $y_1, y_2, \dots, y_m$ là các biến x_i mà hàm F_b phụ thuộc; các $Z_1, Z_2, \dots, Z_p$ là các biến x_i còn lại của hàm F_a ; $m + p = n$. Các biến $y_1, y_2, \dots, y_m$ gọi là các biến liên kết. Các biến $Z_1, Z_2, \dots, Z_p$ gọi là các biến tự do.

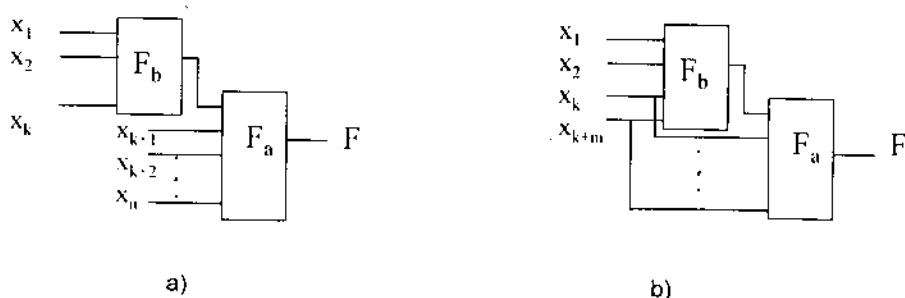
Chúng ta thường gặp hai trường hợp sau đây:

a) Không tồn tại x_k vừa là biến liên kết, vừa là biến tự do.

Trường hợp này gọi là phân giải tách rời đơn giản (PTD). Mô hình cấu trúc của nó biểu diễn trên hình 4.30a.

b) Tồn tại các biến x_k vừa là biến liên kết vừa là biến tự do.

Trường hợp này gọi là phân giải không tách rời đơn giản (PKTD). Mô hình cấu trúc cho trên hình 4.30b.



Hình 4.30. Trường hợp phân giải tách rời đơn giản (a) và không tách rời đơn giản (b)

4.5.1. Phân giải tách rời đơn giản (PTD)

Để biết được khi nào một hàm tồn tại phân giải tách rời đơn giản, ta hãy xét thí dụ sau đây:

Thí dụ 4.22:

Tìm dạng phân giải tách rời đơn giản của hàm sau:

$$y = x_1 \bar{x}_2 x_4 + \bar{x}_1 x_3 + x_2 x_3$$

Ta giả thiết biến liên kết là x_1 và x_2 và phân giải hàm này theo hai biến tự do x_3 và x_4 .

$$F(x_1, x_2, x_3, x_4) = F_a(F_b(x_1, x_2), x_3, x_4) \quad (3)$$

Ta sẽ biểu diễn hàm đã cho lên bảng Karnaugh (hình 4.31a).

		x_1		x_2		y
		$\bar{x}_1$	x_1	$\bar{x}_2$	x_2	
x_3	$\bar{x}_4$	0	1	1	1	x_3
	x_4	1	1	1	1	
	$\bar{x}_4$	1	0	0	0	
	x_4	0	0	0	0	

a)

		x_1		x_1		y
		$\bar{x}_1$	x_1	$\bar{x}_1$	x_1	
x_2	$\bar{x}_4$	0	0	1	0	x_2
	x_4	0	0	1	0	
	$\bar{x}_4$	1	1	1	1	
	x_4	0	1	1	1	

b)

Hình 4.31. Hàm y (a) và phân tích thành các hàm con theo hai biến tự do x_3 và x_4 (b)

Ta hãy phân tích bảng này thành bốn bảng con, ứng với bốn tổ hợp biến của hai biến tự do là x_3 và x_4 (hình 4.31b). Từ đây ta có:

$$\begin{aligned}
 y &= 0 \cdot \bar{x}_3 \bar{x}_4 + (x_1 \bar{x}_2) \bar{x}_3 x_4 + (x_2 + \bar{x}_1) x_3 \bar{x}_4 + 1 \cdot x_3 x_4 \\
 &= x_1 \bar{x}_2 \bar{x}_3 x_4 + \overline{x_1 \bar{x}_2 x_3 \bar{x}_4} + x_3 x_4 (x_1 \bar{x}_2 + \overline{x_1 \bar{x}_2}) \\
 &= (x_1 \bar{x}_2) x_4 + (\bar{x}_1 + x_2) \cdot x_3 \quad (A) \text{ hay} \\
 &= (\bar{x}_1 + x_2) \cdot x_4 + (\bar{x}_1 + x_2) \cdot x_3 \quad (B)
 \end{aligned} \quad (4)$$

Từ hai biểu thức (A) và (B), theo (2) ta có:

$$F_b = x_1 \bar{x}_2 \quad \text{hoặc} \quad F_b = \bar{x}_1 + x_2$$

Như vậy, (A) và (B) là hai khả năng phân giải tách rời đơn giản của hàm đã cho. Mạch thực hiện cho trên hình 4.32a và b tương ứng.

Từ thí dụ này ta rút ra cách tìm phân giải tách rời đơn giản như sau:

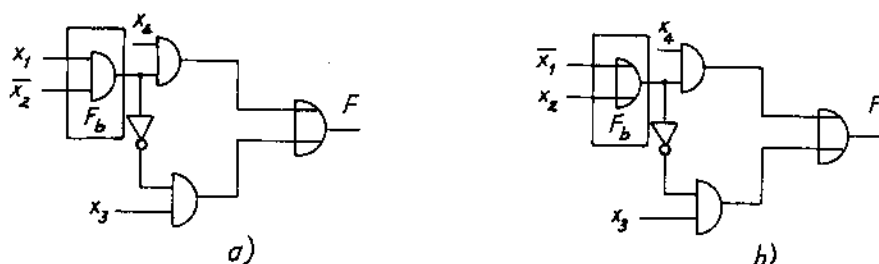
+ Giả thiết hàm có các biến tự do $x_i, i = \overline{1, k}$, các biến còn lại là các biến liên kết - các biến của F_b .

+ Sau khi đưa hàm lên bảng Karnaugh, ta phân tích bảng này thành 2^k bảng con tương ứng với 2^k tập giá trị biến của k biến tự do x_i .

+ Rút ra các hàm dưới dạng (4). Nếu đúng là hàm tồn tại phân giải tách rời đơn giản, trong đó các biến x_i là các biến tự do thì kết quả sẽ cho dạng phân giải tách rời đơn giản.

Nếu không thoả mãn, ta lại phải tìm các tổ hợp biến tự do khác. Điều kiện để tồn tại phân giải tách rời đơn giản dạng (1) là bất kỳ bảng con nào trong 2^k bảng con được tách ra phải có kết quả hàm ở một trong ba dạng dưới đây:

- Các ô của bảng đều bằng "0" (Bảng con thứ nhất trên hình 4.31).
- Các ô của bảng đều bằng "1" (Bảng con thứ tư trên hình 4.31).
- Ứng với dạng F_b và $\bar{F}_b$ (Bảng con thứ hai và thứ ba trên hình 4.31).



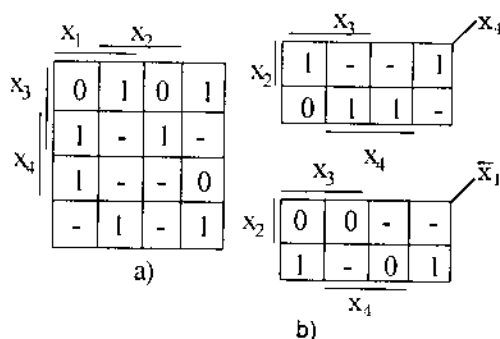
Hình 4.32. Hai phương án phân giải tách rời đơn giản

Trường hợp hàm cho trước là một hàm xác định không đầy đủ

Trong trường hợp này, các ô ứng với các tập giá trị biến mà hàm bất định có thể được điền bằng giá trị "0" hoặc "1" một cách tùy ý sao cho có lợi cho việc phân hoạch theo tiêu chuẩn có phân giải tách rời đơn giản. Vì vậy để biết hàm có tồn tại phân giải tách rời đơn giản hay không, cần kiểm tra mọi khả năng của hàm có tính tới các tổ hợp giá trị biến mà hàm bất định.

Thí dụ 4.23:

Cho hàm trên bảng Karnaugh ở hình 4.33a.



Hình 4.33. Hàm và hai bảng con kiểm tra khả năng tồn tại phân giải tách rời đơn giản

Hãy kiểm tra xem hàm có hay không có phân giải tách rời đơn giản?

Để trả lời cần kiểm tra đối với mọi biến của hàm.

- Kiểm tra đối với biến x_1 , tức tìm dạng:

$$F = F_a(F_b(x_2, x_3, x_4), x_1)$$

Ta có hai bảng con như ở hình 4.33b.

Nếu ta điền các giá trị bất định như hình 4.33c:

	x_3	x_4		x_1
x_2	1	0	1	1
	0	1	1	0

	x_3	x_4		$\bar{x}_1$
x_2	0	1	0	0
	1	0	0	1

Hình 4.33c. Phương án điền các giá trị bất định của hàm

ta sẽ được hai hàm là đảo của nhau. Như vậy hàm đã cho có phân giải tách rời đơn giản sau:

$$F = x_1(x_2x_3x_4 + \bar{x}_2x_4) + \bar{x}_1(x_2x_3x_4 + \bar{x}_2x_4)$$

- Kiểm tra đối với x_2 , tức kiểm tra dạng:

$$F = F_a(F_b(x_1, x_3, x_4), x_2)$$

Phân tách bảng theo x_2 ta có như ở hình 4.33d:

	x_3	x_4		x_2
x_1	1	-	-	1
	0	1	-	-

	x_3	x_4		$\bar{x}_2$
x_1	0	1	1	-
	1	-	0	1

Hình 4.33d. Phân tách hàm theo x_2

Đối với hai bảng này, dễ dàng điền các giá trị bất định để chúng trở thành đảo của nhau. Do đó hàm đã cho có phân giải tách rời đơn giản theo x_2 như sau:

$$y = x_2(\bar{x}_1x_4 + x_1\bar{x}_4) + \bar{x}_2(\bar{x}_1x_4 + x_1\bar{x}_4)$$

- Kiểm tra đối với x_3 , tức dạng:

$$F = F_a(F_b(x_1, x_2, x_4), x_3)$$

Ta phân tách bảng của hàm đã cho theo x_3 và điền các giá trị bất định theo cách như hình 4.33e ta được hai bảng như nhau.

Khi đó hàm sẽ có dạng phân giải tách rời đơn giản như sau:

$$\begin{aligned}
 F &= x_3(x_1x_2 + x_2x_4 + x_1x_4 + \bar{x}_1\bar{x}_2\bar{x}_4) + \bar{x}_3(x_1x_2 + x_2x_4 + x_1x_4 + \bar{x}_1\bar{x}_2\bar{x}_4) \\
 &= x_1x_2 + x_2x_4 + x_1x_4 + \bar{x}_1\bar{x}_3\bar{x}_4
 \end{aligned}$$

x_1		x_2		x_3
0	1	0	1	
1	-	1	-	x_4
0	1	0	1	

x_1		x_2		$\bar{x}_3$
0	1	0	1	
1	-	-	0	x_4
-	1	-	1	

x_1		x_2		x_3
0	1	0	1	
1	1	1	0	x_4
0	1	0	1	

x_1		x_2		$\bar{x}_3$
0	1	0	1	
1	1	1	0	x_4
0	1	0	1	

Hình 4.33e. Phân tích hàm theo x_3

- Kiểm tra phân giải tách rời đơn giản dạng:

$$F = F_a(F_b(x_1, x_2, x_3), x_4)$$

Ta có hai bảng con ở hình 4.33f sau:

x_1		x_2		x_3
0	1	0	1	
1	-	1	-	x_4
1	-	-	0	

x_1		x_2		$\bar{x}_3$
0	1	0	1	
0	1	0	1	x_4
-	1	-	1	

Hình 4.33f. Kiểm tra khả năng phân hoạch theo x_4

Rõ ràng không tồn tại phân giải tách rời đơn giản vì không một phương án sử dụng các giá trị bất định nào của hàm có thể làm cho hai bảng này trở thành đảo của nhau hoặc như nhau.

Từ đây ta suy ra cách tìm phân giải tách rời đơn giản đối với hàm xác định không đầy đủ như sau:

- Biểu diễn hàm lên bảng Karnaugh hay đối xứng.
- Chọn các biến tự do để kiểm tra khả năng có phân giải tách rời đơn giản. Đối với mỗi tập giá trị biến của mỗi biến tự do, trích bảng con.
- Chọn phương án điền các giá trị cụ thể vào các giá trị bất định của hàm trong các bảng con của từng biến tự do sao cho thoả mãn ít nhất một trong bốn điều kiện của phân giải tách rời đơn giản. Nếu tất cả các bảng con đều thoả mãn điều kiện của phân giải tách rời đơn giản thì chuyển sang bước d). Nếu không trở lại bước b) cho tới khi tìm

được dạng phân giải tách rời đơn giản hoặc khi hết phương án chọn các biến tự do thì kết thúc.

d) Viết hàm dưới dạng phân giải tách rời đơn giản và tiếp tục thực hiện đơn giản phần còn lại của hàm.

4.5.2. Phân giải không tách rời đơn giản (PKTD)

Tìm dạng phân giải không tách rời đơn giản của một hàm cho trước là một việc làm rất phức tạp. Để hiểu cách thực hiện ta thông qua thí dụ tìm phân giải không tách rời đơn giản của hàm cho trên hình 4.34.

	0	0	0	0	0	0	0	0	x_1x_2
	0	0	0	1	1	0	1	1	0 0
	1	0	0	1	1	0	1	1	0 1
	0	1	1	1	1	1	1	1	1 1
	1	1	1	1	1	1	1	1	1 0
x_3	0	0	0	0	1	1	1	1	
x_4	0	0	1	1	1	1	0	0	
x_5	0	1	1	0	0	1	1	0	
	s_5	s_4	s_5	s_3	s_5	s_4	s_5		

Hình 4.34. Hàm dùng phân giải tách rời không đơn giản

- Chọn các biến tự do: x_1, x_3 .
- Lập các bảng con ứng với các tổ hợp giá trị biến của x_1x_3 - hình 4.35.

	x_2	x_4	$\bar{x}_1\bar{x}_3$		x_2	x_4	$\bar{x}_1x_3$	
x_5	0	0	0	0	1	0	0	0
	1	1	0	0	1	1	0	0

	x_2	x_4	$x_1\bar{x}_3$		x_2	x_4	x_1x_3	
x_5	0	1	1	1	1	1	1	1
	1	1	1	1	1	1	1	1

Hình 4.35. Các bảng phân tích theo x_1x_3

Như vậy không tồn tại phân giải tách rời đơn giản theo x_1x_3 . Nếu ta tiếp tục phân tích các bảng con này theo x_5 , sẽ có như hình 4.36 sau đây.

x_5 :	$\begin{array}{c} \overline{x_2} \quad \overline{x_4} \\ \hline 0 \quad 0 \quad 0 \quad 0 \end{array}$ $\overline{x_1} \overline{x_3}$	x_5 :	$\begin{array}{c} \overline{x_2} \quad \overline{x_4} \\ \hline 1 \quad 0 \quad 0 \quad 0 \end{array}$ $\overline{x_1} x_3$
x_5 :	$\begin{array}{c} \overline{x_2} \quad \overline{x_4} \\ \hline 0 \quad 1 \quad 1 \quad 1 \end{array}$ $x_1 \overline{x_3}$	x_5 :	$\begin{array}{c} \overline{x_2} \quad \overline{x_4} \\ \hline 1 \quad 1 \quad 1 \quad 1 \end{array}$ $x_1 x_3$
$\overline{x_5}$:	$\begin{array}{c} \overline{x_2} \quad \overline{x_4} \\ \hline 1 \quad 1 \quad 0 \quad 0 \end{array}$ $\overline{x_1} \overline{x_3}$	$\overline{x_5}$:	$\begin{array}{c} \overline{x_2} \quad \overline{x_4} \\ \hline 1 \quad 1 \quad 0 \quad 0 \end{array}$ $\overline{x_1} x_3$
$\overline{x_5}$:	$\begin{array}{c} \overline{x_2} \quad \overline{x_4} \\ \hline 1 \quad 1 \quad 1 \quad 1 \end{array}$ $x_1 \overline{x_3}$	$\overline{x_5}$:	$\begin{array}{c} \overline{x_2} \quad \overline{x_4} \\ \hline 1 \quad 1 \quad 1 \quad 1 \end{array}$ $x_1 x_3$

Hình 4.36. Tiếp tục phân tích theo x_5

Với x_5 tất cả bảng con đều thỏa mãn điều kiện phân giải tách rời đơn giản. Vậy ta có:

$$F^* = x_5(x_1x_3 + \bar{x}_1x_3(x_2\bar{x}_4) + x_1\bar{x}_3(\overline{x_2\bar{x}_4})).$$

Vi $x_1x_3(x_2\bar{x}_4 + (\overline{x_2\bar{x}_4})) = x_1x_3.$

Thay vào biểu thức trên và đặt thừa số chung, ta có:

$$F^* = x_5((x_2\bar{x}_4)x_3 + (\overline{x_2\bar{x}_4})x_1)$$

Làm một cách tương tự với $\bar{x}_5$, cả bốn bảng con đều thỏa mãn điều kiện phân giải tách rời đơn giản, ta có:

$$F^{**} = \bar{x}_5(x_2(\bar{x}_1\bar{x}_3 + \bar{x}_1x_3) + x_1\bar{x}_3.1 + x_1x_3.1)$$

Hay $F^{**} = \bar{x}_5(\bar{x}_1x_2 + x_1) = \bar{x}_5(x_1 + x_2)$

Vậy hàm phân giải không tách rời đơn giản của hàm đã cho có dạng:

$$F = F^* + F^{**} = x_5(x_3(x_2\bar{x}_4) + x_1(\overline{x_2\bar{x}_4})) + \bar{x}_5(x_1 + x_2)$$

Qua việc nghiên cứu hai loại phân giải ta thấy rằng, tìm dạng phân giải của một hàm logic rất quan trọng trong tổng hợp thiết bị số phức tạp. Khi hàm có số biến lớn, phân giải hàm tạo điều kiện tốt cho việc đơn giản hàm và tổ chức kết cấu phần cứng đơn giản hơn, phù hợp với các vi mạch có sẵn.

Tuy quan trọng như vậy nhưng tới nay các phương pháp có hiệu quả trong việc tìm các dạng phân giải hàm vẫn chưa có. Một mặt không phải lúc nào việc tìm dạng phân giải hàm cũng có nghiệm. Mặt khác nhiều hàm lại có dạng phân giải ở nhiều cấp. Các cấp sau có thể là đối với các biến liên kết của cấp trước hoặc đối với các biến tự do. Nó có thể thuộc loại phân giải tách rời đơn giản hoặc phân giải không tách rời đơn giản.

4.5.3. Algôrit thực hiện phân giải hàm

Sau khi đã thực hiện tối thiểu hoá một hàm đã cho bằng một phương pháp nào đó, ta có một phủ tối thiểu C_{min} gồm các khối là các implicang đơn giản thực thụ của hàm. Khi hàm có nhiều biến, không phải bao giờ cũng có thể thực hiện dưới dạng mạch hai cấp, khi đó cần tiến hành phân giải hàm. Một trong các algôrit dùng để phân giải hàm trình bày sau đây cho phép đưa hàm cho dưới dạng phủ C_{min} về dạng có dấu ngoặc, thích hợp với việc thực hiện bằng các vi mạch số có số đầu vào bị hạn chế và sự phức tạp của mạch cũng giảm đi.

Trước khi trình bày algôrit, ta đưa ra thuật toán “Tích của các khối theo toạ độ” như bảng ở hình 4.37.

Trong đó giả sử tích m giữa hai khối:

$$C^r = (c_1, c_2, \dots, c_n)$$

$$C^s = (d_1, d_2, \dots, d_n)$$

thì:

$$C^r m C^s = (c_1 m d_1, \dots, c_n m d_n)$$

Ta gọi những toạ độ trong đó M chiếm chỗ là phần M của khối C_m , còn các toạ độ còn lại là phần λ của khối C_m .

Giả sử hàm cho dưới dạng phủ C_{min} thì algôrit phân giải hàm như sau:

a) Tính $C_{min} m C_{min}$, tức là phải thực hiện thuật toán $C^r m C^s$ đối với tất cả các khối của C_{min} với $r \neq s$. Trong kết quả ta chọn lấy khối mà có số lượng toạ độ phần λ lớn nhất và ký hiệu là C_{m1} .

Giả sử :

+ $\overline{M}_1$ là tập hợp các khối của C_{min} mà $c^r m c^s = c_{m1}$.

+ M_1 là tập hợp các khối của $\overline{M}_1$ mà ở những khối đó đã bỏ đi phần λ .

$$+ \overline{C}_1 = C_{min} \setminus \overline{M}_1$$

$$+ C_1 = \overline{C}_1 \cup C_{m1}$$

b) Viết C_{min} dưới dạng phân giải lần thứ nhất như sau:

$$C_{min1} = \left| \begin{array}{c} \overline{C}_1 \\ C_{m1} \\ M_1 \end{array} \right|$$

m	0	1	x	M
0	0	M	M	M
1	M	1	M	M
x	M	M	M	M
M	M	M	M	M

Hình 4.37. Thuật toán tích các khối theo toạ độ

Trong đó các toạ độ của các khối của M_1 được ghi trực tiếp dưới phần M của khối C_{m1} .

c) Tính $C_1 m C_1$ và lại chọn lấy khối có số toạ độ ở phần λ lớn nhất của kết quả phép tính. Ký hiệu khối này là C_{m2} .

d) Lập lại quá trình trên đây cho tới khi nào ở tất cả các khối chọn ra phần λ còn chưa trống. Nói cách khác toàn bộ là phần M .

Dạng của C_{min} cuối cùng là dạng phân giải cần thiết.

Thí dụ 4.24:

Tìm dạng phân giải của hàm cho dưới dạng C_{min} sau đây:

$$C_{min} = \begin{array}{c|ccccccc} & x_6 & x_5 & x_4 & x_3 & x_2 & x_1 & x_0 \\ \hline 1 & 1 & 0 & 0 & 0 & 0 & x & a \\ 1 & 1 & 0 & 0 & 1 & x & 0 & b \\ 1 & 1 & 0 & 0 & x & 0 & 1 & c \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & d \\ 1 & 0 & 1 & 1 & 1 & 1 & x & e \end{array}$$

Áp dụng algôrit tìm dạng phân giải hàm trên đây, ta có:

a) Tìm $C_{min} m C_{min}$

$$a \ m \ b = \underline{1100MMM}$$

$$a \ m \ c = \underline{1100MMM}$$

$$a \ m \ d = MM00M1M$$

$$a \ m \ e = 1MMMM1M$$

$$b \ m \ c = \underline{1100MMM}$$

$$c \ m \ d = MM00MM1$$

$$b \ m \ d = \underline{MM001MM}$$

$$c \ m \ e = 1MMMMMM$$

$$b \ m \ e = 1MMM1MM$$

$$d \ m \ e = M0MM11M$$

Vì trong kết quả của C_{min} khối 1100MMM có phần λ lớn nhất (bốn toạ độ). Do vậy ta lấy:

$$C_{m1} = 1100MMM$$

Ta có: $M_1 = (a, b, c)$, vì tích toạ độ của chúng cho ta khối 1100MMM.

$$\overline{M}_1 = (01x, 1x0, x01).$$

$$\overline{C} = C_{min} \setminus M_1 = (a, b, c, d, e) \setminus (a, b, c) = (d, e) = (0000111, 101111x)$$

$$C_1 = (d, e) \cup C_{m1} = (0000111, 101111x, 1100MMM).$$

b) Phân giải lần thứ nhất của C_{min} .

$$C_{min1} = \begin{array}{c|c} \begin{array}{c} 0000111 \\ 101111x \\ 1100MMM \\ 01x \\ 1x0 \\ x01 \end{array} & \begin{array}{l} \overline{C}_1 \\ \\ C_{m1} \\ M_1 \end{array} \end{array}$$

e) Thực hiện mục a) và b) đối với $C_1 = (0000111, 101111x, 1100MMM)$

+ Tính $C_1 m C_1$

$$d \ m \ e = 0000111 \ m \ 101111x = M0MM11M$$

$$d \ m \ C_{m1} = 0000111 \ m \ 1100MMM = MM00MMM$$

$$e \ m \ C_{m1} = 101111x \ m \ 1100MMM = 1MMMMMM$$

Trong kết quả, khối có phần λ lớn nhất là $M0MM11M$ nên:

$$C_{m2} = M0MM11M$$

$$\overline{M}_2 = (d, e) = (0000111, 101111x)$$

$$M_2 = (0-00--1, 1-11--x)$$

$$\overline{C}_2 = (C_{m1} \setminus \overline{M}_2 = (C_{m1}, M_1) = (1100MMM, 01x, 1x0, x01)$$

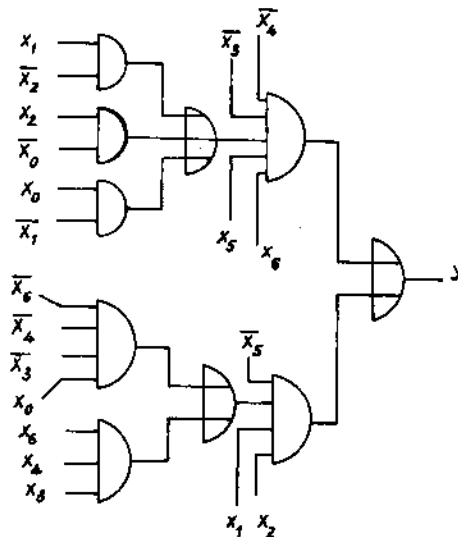
$$C_2 = \overline{C}_2 \cup C_{m2} = (1100MMM, 01x, 1x0, x01, M0MM11M)$$

Viết dưới dạng của C_{min2} :

$$C_{min2} = \begin{matrix} C_2 \\ C_{m2} \\ M_2 \end{matrix} = \begin{matrix} \left| \begin{array}{c} 1100MMM \\ 01x \\ 1x0 \\ x01 \\ M0MM11M \\ 0-00--1 \\ 1-11--x \end{array} \right| \\ C_{m1} \\ M_1 \\ C_{m2} \\ M_2 \end{matrix}$$

d) Thực hiện tiếp tục với C_2

Tìm $C_2 m C_2 = MMMMMMMM$ với mọi khối, nên kết thúc thuật toán phân giải ở C_{min2} .



Hình 4.38. Mạch thực hiện thí dụ 4.23

e) Hàm $y(x_6x_5x_4x_3x_2x_1x_0)$ cho dưới dạng C_{min} có thể viết dưới dạng giải tích như sau:

$$\begin{aligned} y(x_6x_5x_4x_3x_2x_1x_0) &= M_1.C_{m1} + M_2.C_{m2} = \\ &= (01x + 1x0 + x01).1100MMM + (0-00-1 + 1-11--x).M0MM11M = \\ &= (\bar{x}_2x_1 + x_2\bar{x}_0 + \bar{x}_1x_0) \cdot x_6x_5\bar{x}_4\bar{x}_3 + (\bar{x}_6\bar{x}_4\bar{x}_3x_0 + x_6x_4x_3) \cdot \bar{x}_5x_2x_1 \end{aligned}$$

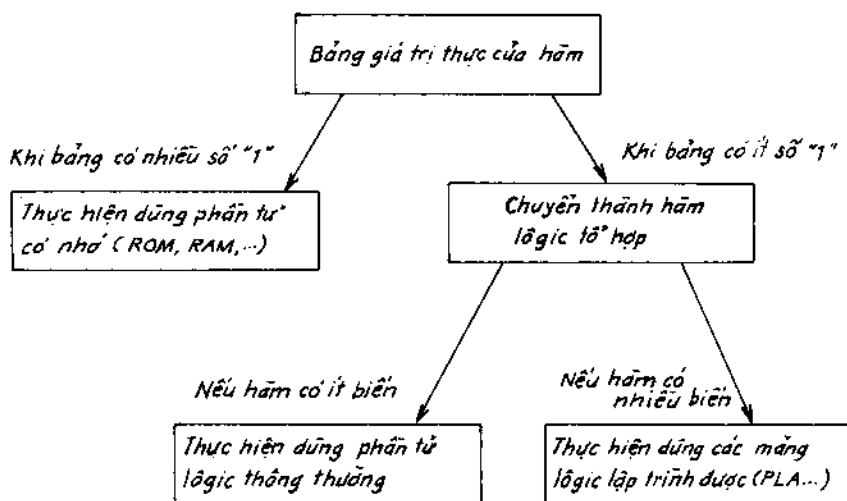
Mạch thực hiện hàm này cho trên hình 4.38.

4.6. THỰC HIỆN HÀM LÔGIC DÙNG CÁC MẢNG LÔGIC LẬP TRÌNH ĐƯỢC VÀ BẢNG CÁC BỘ NHỚ

Mạch số tổ hợp có thể được thực hiện theo nhiều cách khác nhau. Nó có thể và thông thường được thực hiện bằng các mạch logic tổ hợp đơn lẻ như dùng họ các mạch VA, HOAC, KH; mạch VA-KH hay mạch HOAC-KH hoặc hỗn hợp giữa chúng, như đã trình bày trong các mục trước của chương này. Nhưng ngày nay khi mà công nghệ điện tử đã rất phát triển, nền tảng để thực hiện các hàm logic tổ hợp không phải chỉ có các phần tử logic đơn lẻ. Chúng có thể được thực hiện trên cơ sở các mảng logic lập trình được, có độ tích hợp rất lớn, như PAL, PLA, FPLA; các bộ nhớ cố định như ROM, PROM, EPROM hay các bộ nhớ thuật toán RAM. Chúng cũng có thể được thực hiện bằng phần mềm trên cơ sở sử dụng kỹ thuật vi xử lý.

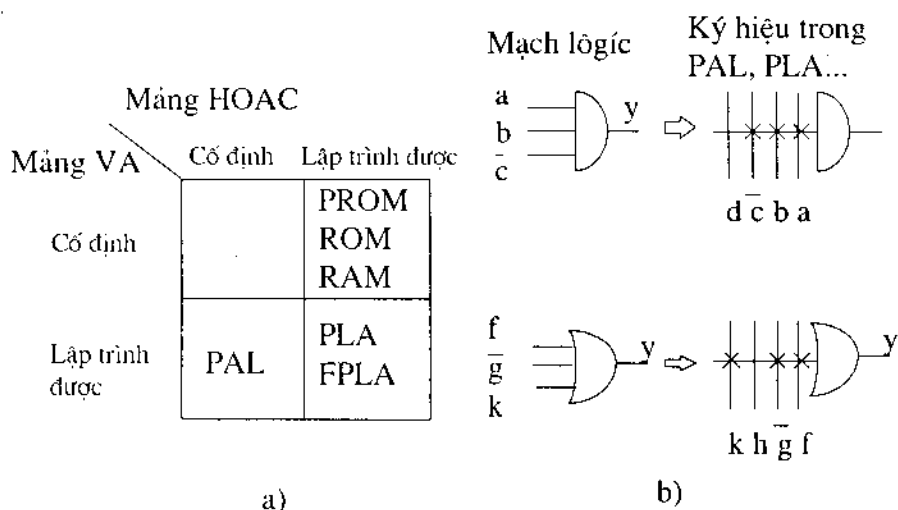
Trong những trường hợp hàm logic được thực hiện nhờ các phần tử có nhớ thì giá trị của hàm ra đối với các tập giá trị biến vào của nó được ghi lại thành nội dung tương ứng của các ô nhớ. Các tập giá trị biến tương ứng với các nội dung đó đóng vai trò địa chỉ của các ô nhớ này.

Sơ đồ khái quát các khuynh hướng và khả năng thực hiện một hàm logic, khi cho dưới dạng bảng giá trị thực, cho trên hình 4.39.



Hình 4.39. Khuynh hướng và khả năng thực hiện các hàm logic

Các mảng logic lập trình được loại PLA (Programmable Logic Array), FPLA (Functional PLA), PAL (Programmable Array Logic) và các bộ nhớ chỉ đọc ROM (Read Only Memory), PROM (Programmable ROM), các bộ nhớ truy nhập ngẫu nhiên RAM (Random Access Memory) cho phép ta thực hiện các hàm logic biểu diễn dưới dạng tuyến chuẩn một cách dễ dàng.



Hình 4.40. Phân loại các mảng logic khả lập trình và ký hiệu thực hiện các hàm logic trên chúng

Về mặt chức năng, các mạch này được phân loại theo khả năng lập trình được để có các mạch VA, HOAC trong mảng, nhằm thực hiện các hàm logic như hình 4.40a. Các mạch logic khi thực hiện trong các mảng logic lập trình được thường ký hiệu như hình 4.40b. Trong đó các dấu gạch chéo thể hiện là nối tắt tại nút đó. Không có dấu gạch chéo là không nối giữa hàng và cột.

Các mảng logic lập trình được dạng có nhớ và không có nhớ này có thể dùng làm môi trường thực hiện các hàm logic tổ hợp. Dưới đây ta sẽ xét một vài trường hợp thực hiện hàm logic dùng các mảng logic lập trình được này.

4.6.1. Thực hiện bằng PAL, PLA hay FPLA

Giữa PAL với PLA và FPLA chỉ khác nhau ở chỗ, đối với PAL, phần mảng logic VA và HOAC là hai phần tách biệt và được liên kết cứng với nhau. Trong phần mảng logic VA chỉ có thể lập trình để thực hiện các phép nhân logic. Ngược lại, trong phần mảng logic HOAC lại chỉ có thể lập trình để thực hiện các phép cộng logic. Trong khi đó các mảng lập trình được loại PLA hay FPLA ta có thể lập trình để có mạch thực hiện phép nhân logic, phép cộng logic hay nối tắt ở bất cứ nút nào trong mảng. Vì vậy ở đây chúng

ta chỉ trình bày việc thực hiện một hàm logic trên PAL, là loại hiện được dùng phổ biến nhất, đặc biệt trong lĩnh vực điện tử công nghiệp. Ngoài ra sẽ điểm qua phương pháp thực hiện hàm logic dùng bộ nhớ và phương hướng dùng phần mềm vi xử lý.

1) Thực hiện hàm logic dùng PAL

Trong trường hợp này hàm có thể viết dưới dạng tuyến chuẩn tối thiểu hay dạng tuyến chuẩn chính tắc cũng được. Vấn đề đơn giản hàm khi thực hiện bằng các mạch PAL không nhất thiết phải thực hiện. Chúng ta sẽ xem cách thực hiện một hệ hàm thông qua thí dụ sau đây.

Thí dụ 4.25:

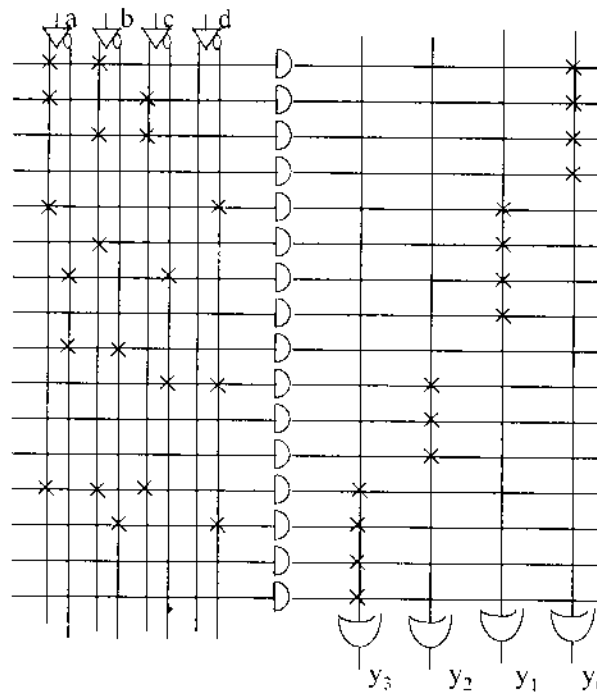
Cho hệ bốn hàm:

$$y_0 = ab + ac + bc$$

$$y_1 = a\bar{d} + b + \bar{a}\bar{c}$$

$$y_2 = \bar{a}\bar{b} + cd$$

$$y_3 = abc + \bar{b}\bar{d}$$

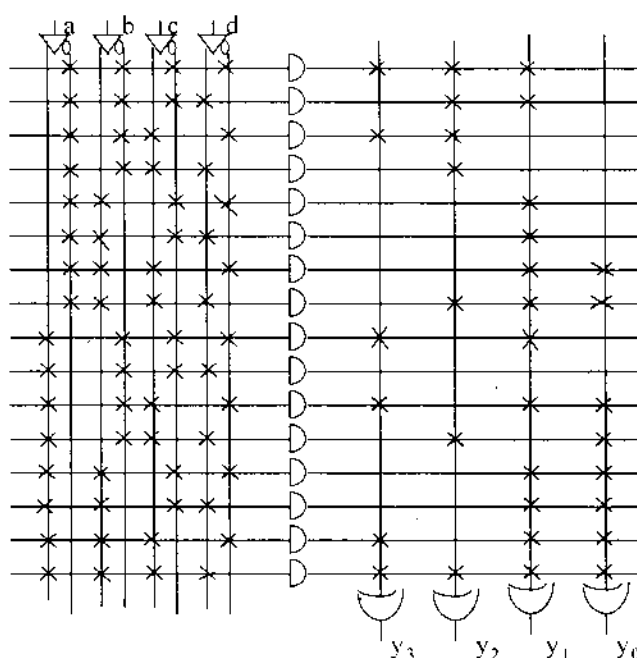


Hình 4.41. Lập trình trên PAL

Để thực hiện các hàm này dùng một PAL có bốn đầu vào a, b, c, d và bốn đầu ra cho bốn hàm y_0 , y_1 , y_2 và y_3 , cách lập trình chức năng của các nút của PAL cho trên hình 4.41.

Nếu bốn hàm đã cho được triển khai dưới dạng tuyến chuẩn chính tắc, chúng ta vẫn có thể thực hiện chúng bằng cách dùng một PAL bốn đầu vào và bốn đầu ra như trên nhưng việc lập trình sẽ khác đi (hình 4.42).

Trong trường hợp này phần mạch VA bên trái của PAL tạo thành 16 địa chỉ, tương ứng với 16 tập giá trị biến của hàm. Giá trị của mỗi hàm xác định bằng cách cộng logic tất cả giá trị 1 của nó đối với các tập giá trị biến tương ứng.



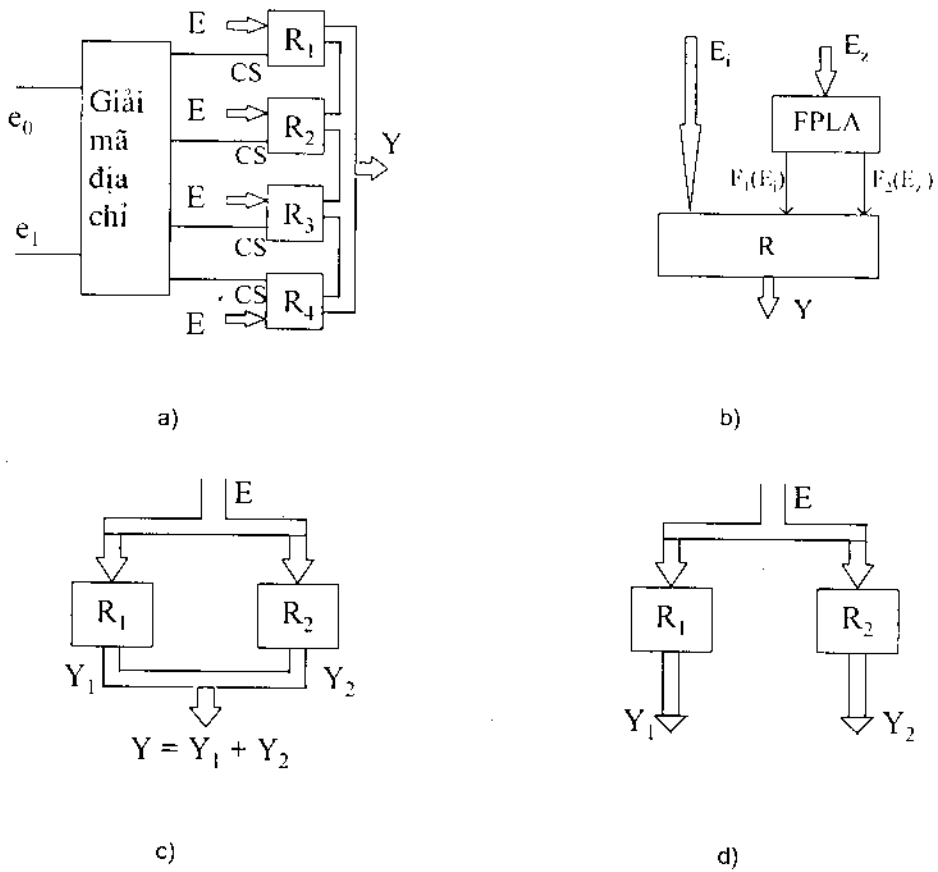
Hình 4.42. Thực hiện trên PAL 4 đầu vào

Trong một số trường hợp, khi số biến vào hay số lượng số hạng của dạng tuyến chuẩn của hàm lớn hay rất lớn, số đầu vào của một PAL hay FPLA, PLA hay ROM, RAM không đủ để thực hiện một hàm thì có thể nối ghép một số mạch này lại với nhau theo cách thực hiện các tín hiệu chọn chip. Phương thức nối ghép có nhiều cách. Hình 4.43a và 4.43b là các nối ghép để tăng thêm số lượng đầu vào. Hình 4.43c dùng cho trường hợp số lượng các số hạng của dạng tuyến chuẩn lớn và hình 4.43d dùng cho trường hợp cần tăng thêm số lượng đầu ra của hệ thống.

2) Thực hiện hàm logic bằng PLA, FPLA

PLA - Programmable Logic Array là loại mạch điện tử phức tạp, mỗi "ô" của nó là một phần tử đa năng, có thể điều khiển để có chức năng cụ thể như mạch VA, mạch HOAC hay ngăn mạch.

Trên một mạch điện tử đa năng lập trình được như vậy, có thể thực hiện các hàm logic. Thí dụ với 10 hàm 15 biến sau đây có thể thực hiện được bằng PLA bằng cách lập trình tương ứng.



Hình 4.43. Các phương thức mở rộng chức năng của PAL, PLA

Để thực hiện 10 hàm 15 biến:

$$Y_t = f_t(a, b, c, d, e, f, g, h, i, j, k, l, n, m, p); \quad t = \overline{1, 10}$$

Trước hết ta thực hiện một hàm, thí dụ hàm Y_4 . Biểu thức của hàm Y_4 này có thể như sau:

$$Y_4 = \left. \begin{aligned} & a \bar{b} c d \bar{e} f g h i j \bar{k} l n m p \\ & + a b c d \bar{e} f g h i j \bar{k} l n \bar{m} \bar{p} \\ & \vdots \\ & + \bar{a} \bar{b} c d e f g h i j k l n \bar{m} \bar{p} \end{aligned} \right\} \text{gồm 16 số hạng}$$

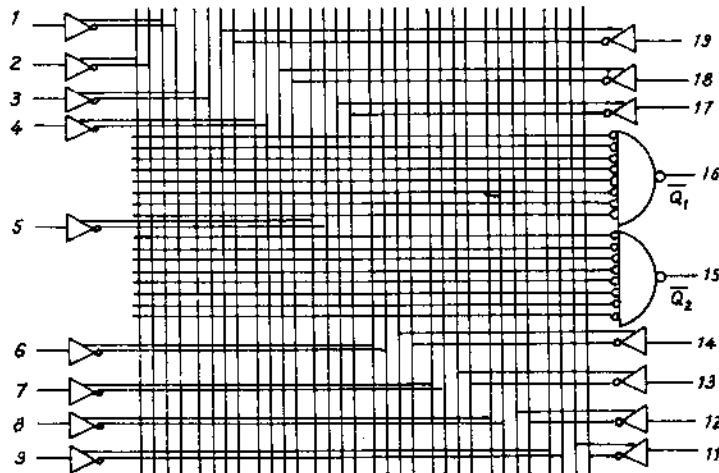
Để thực hiện, ta dùng một PLA có 16 đầu vào và hai đầu ra là $\bar{Q}_1$ và $\bar{Q}_2$ (hình 4.44). Hai đầu ra này là các hàm logic 16 đầu vào, dưới dạng tuyển chuẩn gồm mỗi hàm tám số hạng như sau:

$$\begin{aligned} \bar{Q}_1 &= P_{11} + P_{12} + P_{13} + \dots + P_{18} \\ \bar{Q}_2 &= P_{21} + P_{22} + P_{23} + \dots + P_{28} \end{aligned}$$

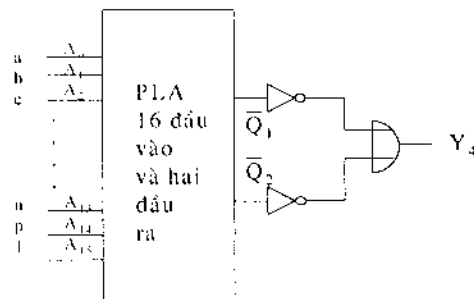
Vì hàm Y_1 có 16 số hạng, nó sẽ được thực hiện trên cả hai cổng ra $\overline{Q}_1$ và $\overline{Q}_2$. Bằng cách thêm hai mạch Đảo ở hai đầu ra rồi cộng lại như hình 4.45, kết quả ta có :

$$Y_4 = \overline{\overline{Q}_1} + \overline{\overline{Q}_2} = Q_1 + Q_2$$

như hình 4.45.



Hình 4.44. PLA có 16 đầu vào và 2 đầu ra $\overline{Q}_1$ và $\overline{Q}_2$



Hình 4.45. Tổng hợp kết quả ra để có Y_4

4.6.2. Thực hiện trên RAM

Giả sử ta có hệ 10 hàm logic 15 biến:

$$Y_{mt} = f_t(a, b, c, d, e, f, g, h, i, j, k, l, m, n, p); t = \overline{1, 10}$$

Để thực hiện hệ 10 hàm này, ta cần một bộ nhớ có dung lượng sao cho:

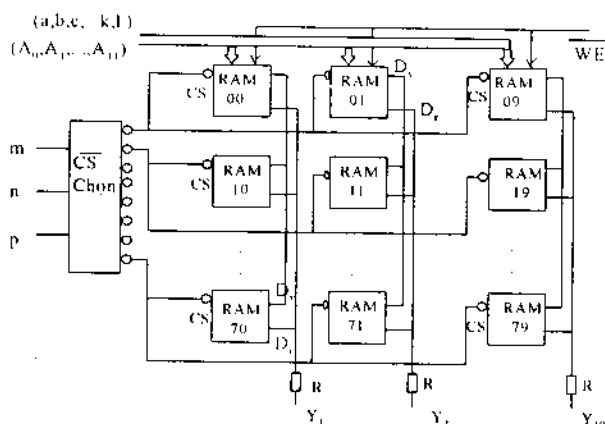
- Nhớ được các từ có 10 bit sau đây: $Y_1, Y_2, \dots, Y_{10}$.

- Có 15 bit địa chỉ ứng với 15 biến của các hàm trên là:

a, b, c, d, e, f, g, h, i, j, k, l, m, n, p

Để tổ chức một bộ nhớ như vậy, phải sử dụng, thí dụ 80 IC RAM loại 4096, tổ chức thành 10 cột để có các từ 10 bit và tám hàng. Như vậy số lượng từ 10 bit sẽ là: $4096 \times 8 = 32.768$ từ. Nói cách khác, số bit nhớ của bộ nhớ tổng thể khi đó là $32.768 \times 10 = 327.680$ bit. Tức có 32 Kilo từ loại 10 bit.

Tổ chức bộ nhớ như vậy cho trên hình 4.46.



Hình 4.46. Mảng các RAM dùng thực hiện hệ hàm

Như vậy mỗi cột tương ứng với một hàm. Các cột từ 0 đến 9 ứng với các hàm Y_1 đến Y_{10} .

Từ biểu thức đại số của hàm, căn cứ vào giá trị của hàm đối với mỗi tập giá trị biến của 15 biến (a, b, c, d, ..., m, n, p) ta ghi trước giá trị của hàm vào các ô tương ứng: Địa chỉ của ô là tập giá trị biến mà hàm có giá trị 1 hoặc 0.

Các tập giá trị biến xuất hiện khi quan sát sẽ tạo thành các địa chỉ thực và do đó tại đầu ra giá trị của hàm sẽ được xác định.

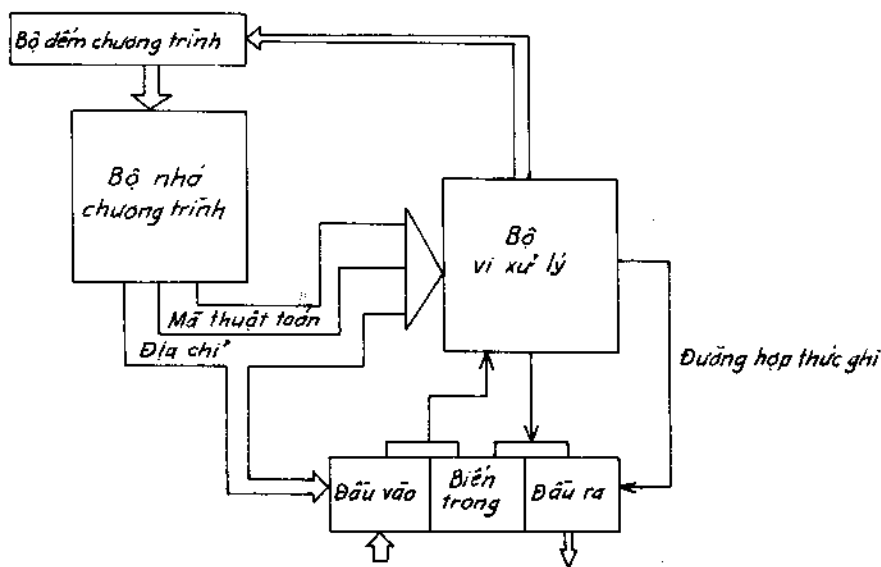
4.6.3. Thực hiện hàm logic dùng chương trình

Một hàm logic có thể được thực hiện bằng cách dùng phần mềm của một ô tô máy lập trình được.

1) Ô tô máy lập trình được

Các ô tô máy hay các bộ điều khiển lập trình được nói chung được thực hiện bằng các mạch từ các linh kiện rời rạc hoặc bằng kỹ thuật vi xử lý.

Trong trường hợp này thủ tục xử lý dựa trên việc dò xét theo chu trình bộ nhớ chương trình. Cấu trúc chung của một ô tô máy lập trình được cho trên hình 4.47.

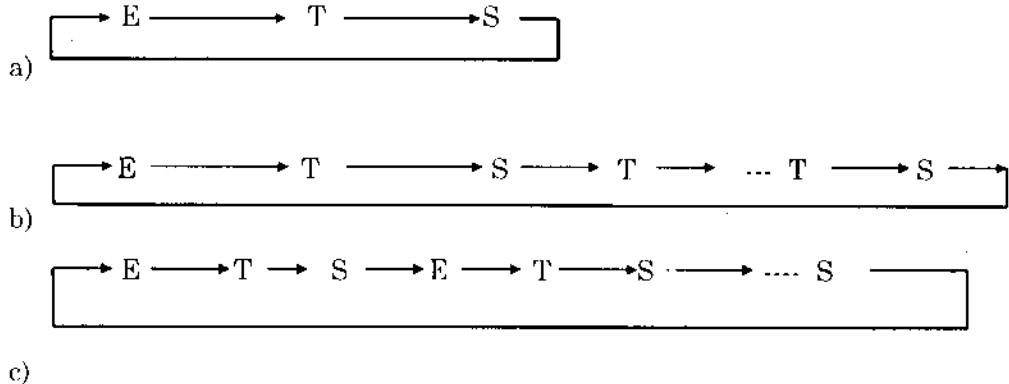


Hình 4.47. Cấu trúc chung của ô tô mát khả lập trình

Bộ đếm chương trình (CO) điều khiển bộ nhớ chương trình để từ đó cấp cho ta các lệnh. Mỗi lệnh có hai phần: Một phần là mã thuật toán của lệnh đó. Phần thứ hai là địa chỉ của các đầu vào, đầu ra hoặc của biến trong tùy theo lệnh.

Việc thực hiện một chương trình đơn thuần là bằng cách tăng dần nội dung của bộ đếm chương trình. Song khi gặp chương trình nhảy, phần địa chỉ của lệnh sẽ cung cấp giá trị cần nhảy để tìm đến lệnh cần thực hiện tiếp theo. Ở đây phép nhảy đều là nhảy về phía trước, tức là nhảy tới một địa chỉ có giá trị cao hơn.

Trong chu trình làm việc với bộ nhớ, thứ tự thực hiện các thuật toán nhận số liệu (E), gán kết quả cho đầu ra (S) và các phép xử lý trung gian khác (T) có thể có ba dạng như ở hình 4.48.



Hình 4.48. Các phương án làm việc với bộ nhớ

Các khả năng khác sẽ dẫn tới hoặc chu trình hoặc các chu trình con.

Trường hợp thứ nhất (a) là ứng với chế độ đồng bộ chặt chẽ giữa đầu vào và đầu ra. Sự thay đổi duy nhất trong trường hợp này có thể có là thay đổi thời gian của chu kỳ. Trong một số máy, lệnh nhảy sẽ là mất ít thời gian sử dụng bộ nhớ hơn. Trường hợp thứ nhất này cho phép tránh được đa phần điều ngẫu nhiên tại đầu ra.

Với trường hợp (b) ta duy trì việc dò xét các đầu vào ở đầu của chu trình nhưng lại dùng đầu ra theo sự xác định nó trong chương trình. Tức là có thể nhiều lần dùng đầu ra với một đầu vào. Cũng chính vì vậy, thứ tự sử dụng các đầu ra (thứ tự xuất hiện của nó) có ảnh hưởng lớn trên phương diện các ngẫu nhiên tại đầu ra (sự chồng chéo lên nhau của các đầu ra hay sự không liên tục của các tác động).

Trong trường hợp (c), người ta kiểm tra các đầu vào mỗi khi nó xuất hiện. Như vậy trong một số trường hợp, phương pháp này cho phép đối với một số xử lý khẩn cấp dùng nhiều lần cùng một lệnh hay cùng một chương trình trong một chu trình. Thủ tục này là nguy hiểm và bắt buộc phải giữ giá trị của một đầu vào tại một biến trong khi giá trị này có liên quan đến nhiều miền của một phương trình hay nhiều phương trình.

Tập các lệnh của một ô tô máy khả lập trình có thể phân chia thành ba loại:

- Các lệnh dùng để lập trình như bình thường (kiểu kinh điển).
- Các lệnh được định hướng chuyên dụng cho việc xử lý của các hệ tự động công nghiệp.
- Các lệnh dùng để đưa về 0 hay 1 các đầu ra hay biến trong, các lệnh nhảy, các lệnh thực hiện từng bước, các lệnh thực hiện phép đếm hay giữ chậm. Cuối cùng là sự mở rộng về số của các tính toán trên một từ.

2) Sơ lược về thực hiện một hàm Boole bằng chương trình

Để xét vấn đề này, giả sử có một số lệnh như sau:

CH Địa chỉ: Nạp giá trị của đầu vào có địa chỉ đã chỉ ra trong lệnh vào thanh chứa

CHC Địa chỉ: Nạp giá trị đảo của đầu vào có địa chỉ đã chỉ ra trong lệnh vào thanh chứa.

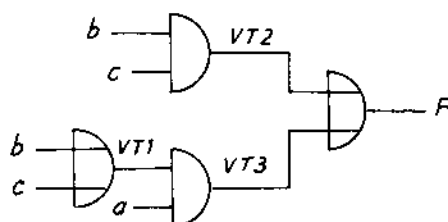
ET Địa chỉ: Phân tử VA (phép nhân logic). Đầu ra của nó sẽ được đưa đến địa chỉ đã chỉ ra trong lệnh.

OU Địa chỉ: Phân tử HOAC. Đầu ra của nó sẽ được đưa đến địa chỉ đã chỉ ra trong lệnh.

Áp dụng các lệnh này để giải phương trình:

$$F = a(b + c) + bc$$

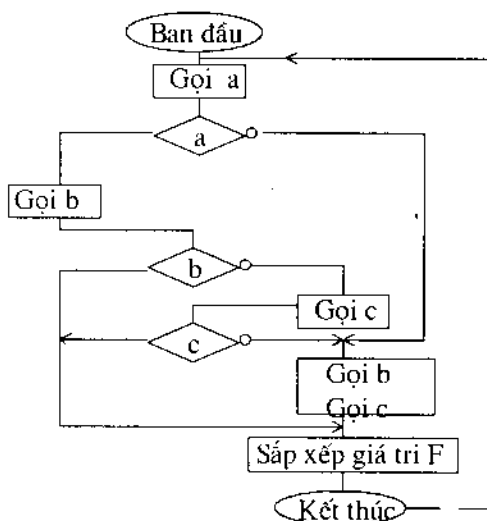
cho trên hình 4.49.



CH b
CH c
OU VT1
CH b
CH c
ET VT3
CH VT1
CH a
ET VT2
CH VT2
CH VT3
OU F

Hình 4.49. Sơ đồ giải phương trình

Để dễ lập trình, ta có thể biểu diễn hàm cho trước dưới dạng lưu đồ thuật toán. Thí dụ bài toán trên đây có thể biểu diễn dưới dạng lưu đồ thuật toán như ở hình 4.50.



Hình 4.50. Lưu đồ thuật toán

Nếu chúng ta có một bộ vi xử lý có các lệnh sau đây:

CH Địa chỉ: Nạp nội dung của biến có địa chỉ đã chỉ ra trong lệnh vào bộ ghi trong.

CHC Địa chỉ: Nạp giá trị đảo của nội dung của biến có địa chỉ đã chỉ ra trong lệnh vào bộ ghi trong.

RA Địa chỉ: Đưa kết quả vào địa chỉ đã chỉ ra trong lệnh.

SI "0" ALLER A + n: Nếu là 0 thì thực hiện lệnh tại A + n.

Trong đó n là giá trị mà nội dung của bộ đếm chương trình cần tăng lên.

SI “1” ALLER A + n: Tương tự như lệnh trên.

Khi đó hàm được biểu diễn dưới dạng lưu đồ ở hình 4.50 được thực hiện bằng chương trình sau đây với địa chỉ ban đầu từ 0.

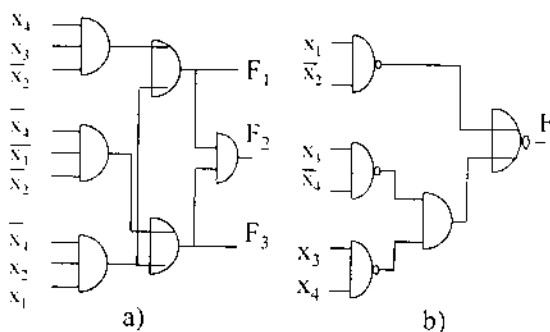
Địa chỉ	Lệnh
0	CH a
1	SI “0” ALLER A + 5
2	CH b
3	SI “1” ALLER A + 5
4	CH c
5	SI “1” ALLER A + 3
6	CH b
7	CH c
8	RA F

BÀI TẬP CHƯƠNG 4

- Thực hiện hàm cho trên hình 4.51. Xác định các đường truyền tín hiệu, đường truyền và cấp của của mạch.

	x_1	x_2	y
x_3	1	1	1
x_4	1	1	1
	1	1	1
		1	

Hình 4.51



Hình 4.52

- Đánh số các phần tử mạch của hình 4.52 và xác định lớp các phần tử mạch.
- Lập ma trận quan hệ cho các mạch ở hình 4.52a và b.
- Tìm dạng tối thiểu hai cấp loại dạng tuyển chuẩn tối thiểu và dạng hội chuẩn tối thiểu và thực hiện mạch của các hàm sau:

4.1 $y_1 = f(0, 4, 7, 8, 9, 10, 11, 12); n = 4, b = 10$

$y_2 = f(1001, 1110, 1000, 0111, 1010, 0101); n = 4, b = 10$

$y_3 = f(0, 2, 4, 6, 8, 10, 17, 18, 19, 20, 25, 26, 27, 28, 29, 30); n = 5, b = 10$

$$4.2 \quad y_1 = (x_1 x_2 + \bar{x}_1 \bar{x}_2) x_4 + (x_1 x_2 + \bar{x}_1 \bar{x}_2) x_3$$

$$y_2 = (x_1 x_2 + x_3) + x_4 + (x_1 + x_2)(x_3 + x_4)$$

$$y_3 = (x_1 + x_2)(x_1 + x_3 + x_4)(x_1 + x_3)$$

4.3. Đơn giản và thực hiện hệ hàm sau đây bằng mạch hai cấp dùng họ hàm (VA, HOAC, KH)

x_1	x_2	x_3	x_4	y_1	y_2	y_3	y_4	y_5	y_6	x_1	x_2	x_3	x_4	y_1	y_2	y_3	y_4	y_5	y_6
0	0	0	0	1	1	0	0	1	0	1	0	0	0	1	1	0	1	0	1
0	0	0	1	1	0	1	0	0	0	1	0	0	1	1	1	0	0	1	1
0	0	1	0	0	0	1	0	1	1	1	0	1	0	0	0	0	0	0	1
0	0	1	1	0	1	1	1	0	1	1	0	1	1	0	0	0	1	1	1
0	1	0	0	0	1	0	1	1	1	1	1	0	0	1	0	1	1	1	0
0	1	0	1	0	0	1	1	1	0	1	1	0	1	0	1	1	0	0	1
0	1	1	0	1	1	1	0	1	1	1	1	1	0	1	1	0	0	1	0
0	1	1	1	0	0	0	1	1	0	1	1	1	1	0	0	1	0	1	0

4.4. Tìm mạch hai cấp tối thiểu dùng các mạch họ (VA, HOAC, KH) của các hàm xác định không đầy đủ sau đây:

$$y_1 = f^1(0, 2, 5, 7, 8, 11); \quad f^-(1, 3, 4, 12, 13), \quad n = 4, b = 10$$

$$y_2 = f^1(1, 2, 4, 11, 12, 13); \quad f^-(3, 10, 14), \quad n = 4; b = 10$$

$$y_3 = f^1(0, 1, 7, 8, 11); \quad f^-(3, 4, 6), \quad n = 4; b = 10$$

$$y_4 = f^1(5, 6, 7, 11, 15); \quad f^-(2, 3, 4, 14), \quad n = 4; b = 10$$

4.5. Tìm mạch tối thiểu dạng VA-KH – VA-KH (NAND - NAND) hai cấp đối với các hàm cho trong nhiệm vụ 4.4.

4.6. Xây dựng mạch tối thiểu dạng VA-KH -- VA-KH (NAND - NAND) của các hàm sau đây có $n = 5, b = 10$:

$$y_1 = f^1(0, 4, 12, 14, 15, 17, 20, \text{đến } 28, 30); \quad f^-(13, 18, 19)$$

$$y_2 = f^1(0, 1, 3 \text{ đến } 5, 12 \text{ đến } 15, 17, 20 \text{ đến } 25); \quad f^-(5, 8, 9, 19)$$

$$y_3 = f(0, 1, 3 \text{ đến } 5, 15, 17, 18, 25, 28)$$

$$y_4 = f(0 \text{ đến } 7, 10, 12, 16 \text{ đến } 20, 26 \text{ đến } 29).$$

4.7. Tìm mạch tối thiểu dạng HOAC-KH – HOAC -KH (NOR - NOR) hai cấp của các hàm cho trong nhiệm vụ 4.5 và 4.6.

5. Đơn giản và thực hiện hệ hàm ở dạng hai cấp.

5.1. Tìm dạng tối thiểu hai cấp dùng các phần tử logic họ (VA, HOAC, KH) của hệ hàm cho trong nhiệm vụ 4.3.

5.2. Tìm dạng hai cấp tối thiểu dùng các phần tử họ (VA, HOAC, KH) của các hệ hàm sau đây:

a) $n = 3, b = 10: \quad y_1 = f(0, 3, 5, 6, 7)$

$$y_2 = f(2, 5, 6)$$

$$y_3 = f(0, 1, 4, 6)$$

b) $n = 4, b = 10$

$$y_1 = f(0, 1, 2, 6, 8, 9)$$

$$y_2 = f(0, 1, 4, 5, 12, 14, 15)$$

c) $n = 4, b = 10$

$$y_1 = f(1, 2, 5, 7, 8, 11, 13, 14)$$

$$y_2 = f(1, 2, 9, 10, 11, 15)$$

$$y_3 = f(0, 1, 4, 6, 12, 14, 15)$$

d) $n = 4, b = 10$

$$y_1 = f(1, 4, 5, 8, 9, 12, 13)$$

$$y_2 = f(0, 1, 3, 4, 5, 7, 11, 12, 15)$$

$$y_3 = f(1, 4, 5, 6, 12, 14, 15)$$

$$y_4 = f(0, 1, 2, 3, 7, 9, 12, 13, 15)$$

5.3. Với hệ hàm xác định không đầy đủ.

a) $n = 4, b = 10$

$$y_1 = f^1(0, 2, 5, 7, 8, 11); \quad f^-(1, 3, 4, 12).$$

$$y_2 = f^1(1, 2, 4, 11, 12, 13); \quad f^-(3, 10, 14)$$

b) $n = 3, b = 10$

$$y_1 = f^1(0, 2, 4, 6); \quad f^-(1),$$

$$y_2 = f^1(0, 4, 5); \quad f^-(7)$$

$$y_3 = f^1(1, 3, 5); \quad f^-(0, 2)$$

c) $n = 4, b = 10$

$$y_1 = f^1(1, 2, 5); \quad f^-(4, 6, 7)$$

$$y_2 = f^1(4, 6, 7, 14, 15); \quad f^-(0, 1, 2)$$

$$y_3 = f^1(0, 2, 10, 14, 15); \quad f^-(1, 3)$$

5.4. Tìm dạng mạch hai cấp tối thiểu loại VA-KH – VA-KH (NAND - NAND) của các hệ hàm cho trong nhiệm vụ 5.2 và 5.3.

5.5. Tìm dạng mạch hai cấp tối thiểu loại HOAC-KH – HOAC-KH (NOR - NOR) của các hệ hàm cho trong nhiệm vụ 5.2 và 5.3.

6. Tổng hợp mạch nhiều cấp.

6.1. Tổng hợp mạch nhiều cấp các hàm sau đây:

$$y_1 = x_1 x_2 x_3 + x_2 x_1 + x_1 \bar{x}_3 x_4 + \bar{x}_1 \bar{x}_2$$

$$y_2 = x_1 x_2 x_4 + x_1 \bar{x}_2 x_3 + x_1 \bar{x}_2 x_4 + \bar{x}_1 x_2 \bar{x}_3$$

$$y_3 = \bar{x}_1 \bar{x}_2 (\bar{x}_3 + \bar{x}_4) + (x_2 + \bar{x}_3) x_1 x_4$$

6.2. Tổng hợp mạch nhiều cấp đối với hệ hàm sau:

a) $y_1 = x_1 \bar{x}_2 x_3 + x_1 x_2 \bar{x}_3 + x_1 x_2 x_4$

$$y_2 = x_1 (x_2 + x_3) (x_4 + x_3)$$

b) $y_1 = \bar{x}_2 \bar{x}_3 \bar{x}_4 + x_1 x_2 \bar{x}_3 + x_1 x_2 x_4$

$$y_2 = x_1 x_2 + \bar{x}_1 x_2 \bar{x}_3 x_4$$

$$y_3 = \bar{x}_1 \bar{x}_2 \bar{x}_4 + x_3 \bar{x}_4 + x_1 x_2 x_4$$

c) $y_1 = (\bar{x}_3 + \bar{x}_4) (x_1 + x_2 + x_3 + x_4)$

$$y_2 = (x_2 + x_4) (x_1 + \bar{x}_3 + \bar{x}_4) (x_1 + \bar{x}_2 + x_3 + x_4)$$

$$y_3 = (\bar{x}_1 + \bar{x}_2 + \bar{x}_4) (\bar{x}_1 + x_3 + x_4) (x_2 + x_3 + x_4) (x_1 + x_3 + \bar{x}_4)$$

d) Với $n = 4, b = 10$

$$y_1 = f^+(0, 3, 6, 9, 11, 13); f^-(1, 2)$$

$$y_2 = f^+(1, 3, 4, 7, 11, 13, 15); f^-(0, 2)$$

6.3. Thực hiện các hàm của nhiệm vụ 6.1 sao cho đơn giản nhất và bằng các phần tử mạch VA và HOAC hai đầu vào.

6.4. Thực hiện mạch các hệ hàm cho trong nhiệm vụ 6.1 với các phần tử mạch không quá ba đầu vào.

6.5. Thực hiện mạch các hệ hàm cho trong nhiệm vụ 6.2 sao cho số tải của mỗi phần tử logic không quá hai.

a) Dùng phương pháp thay đổi mạch, không thay đổi cấu trúc logic.

b) Dùng phương pháp thay đổi cấu trúc logic của mạch.

7. Tìm dạng phân giải hàm:

7.1. Tìm dạng phân giải tách rời đơn giản của các hàm sau: $n = 4, b = 10$

$$y_1 = f(0, 3, 6, 10, 14, 15)$$

$$y_2 = f(2, 3, 4, 5, 9, 10, 11, 12, 15)$$

$$y_3 = f(1, 2, 3, 7, 14, 15); f^-(6, 10)$$

$$y_4 = f(0, 2, 4, 6, 10, 11); f^-(5, 7)$$

7.2. Tìm dạng phân giải không tách rời đơn giản các hàm sau: $n = 4, b = 10$

$$y_1 = f(4, 7, 8, 10, 12, 14)$$

$$y_2 = f(1, 2, 5, 7, 13, 15)$$

$$y_3 = f(1, 3, 6 \text{ đến } 10, 18 \text{ đến } 21, 29, 30); n = 5, b = 10$$

8. Dạng mạch VA-KH – VA-KH; HOAC-KH – HOAC-KH

8.1. Tìm dạng mạch đơn giản nhất thực hiện các hàm cho trong nhiệm vụ 6.2 bằng các phần tử logic VA-KH hoặc HOAC-KH có hai đầu vào.

8.2. Tìm các mạch đơn giản nhất thực hiện biến đổi các mã 8421 sang mã dư 3 và mã Grey và ngược lại, dùng các phần tử logic loại VA-KH và HOAC-KH có hai đầu vào.

9. Cho hàm dưới dạng phủ tối thiểu dưới đây, hãy áp dụng algôrit phân giải hàm để thực hiện chúng.

9.1 a) Hàm y_1 có phủ tối thiểu $C_{\min} = (110001x, 1100x01, 0011011, 10111x, x101xx0)$.

b) Hàm y_2 có phủ tối thiểu $C_{\min} = (x11x111xx, x110xx11x, x11x10x0x, x11x011xx, 1x10x0x0x, 1x1xx111x, 1x11xx11x, 11x0x0x0x, 11xx111xx)$.

9.2. Áp dụng algôrit phân giải hàm đối với các hàm sau đây sau khi đã đơn giản chúng bằng phương pháp bảng đối xứng:

a) $y_1 = f^1(1, 4, 6, 7, 11, 13, 15, 20, 23 \text{ đến } 25, 31); f^-(18, 28, 30); n = 5, b = 10$.

b) $y_2 = f^1(1, 2, 5, 7, 15 \text{ đến } 21, 26, 27); f(10, 11, 12); n = 5, b = 10$

c) $y_3 = f^1(0 \text{ đến } 5, 9 \text{ đến } 11, 18 \text{ đến } 22, 27, 29, 42 \text{ đến } 47, 50 \text{ đến } 55);$

$f^-(25, 26, 30 \text{ đến } 37, 61, 62); n = 6, b = 10$.

Chương 5

PHÂN TÍCH THIẾT BỊ SỐ TỔ HỢP

Thiết bị số chia thành hai loại: Thiết bị số tổ hợp và thiết bị số có nhớ, còn gọi là mạch dây hay mạch tuần tự.

Thiết bị số tổ hợp hay mạch tổ hợp là loại mạch mà tín hiệu ra tại một thời điểm cho trước t chỉ phụ thuộc vào tín hiệu vào tại thời điểm t đó mà thôi. Nếu các tín hiệu vào là các x_i , $i = 1, n$ và tín hiệu ra là y thì đối với mạch tổ hợp ta có phương trình:

$$y(t) = f(x_i(t))$$

Mạch có nhớ là loại mạch mà tín hiệu ra tại một thời điểm phụ thuộc không những vào tín hiệu vào tại thời điểm đó mà còn phụ thuộc vào tín hiệu vào tại các thời điểm trước đó. Chúng ta sẽ quay lại mạch có nhớ trong các chương sau.

Mục đích của bài toán phân tích mạch tổ hợp là tìm hiểu, xác định và đánh giá kết cấu logic và các đặc trưng của mạch trong mối quan hệ giữa tín hiệu vào và tín hiệu ra. Trong tổng hợp mạch, bài toán phân tích là một phần của bài toán tổng hợp, nhằm kiểm tra lại các đặc trưng và tính năng của mạch được thiết kế để trong trường hợp không đáp ứng yêu cầu thì sửa đổi lại thiết kế hoặc có các biện pháp khắc phục.

Trong khai thác sử dụng, phân tích mạch nhằm xác định chức năng của mạch hoặc xác định các đặc trưng thực tế của mạch khi có tính tới sự giữ chậm tín hiệu khi đi qua các phần tử logic, mà yếu tố này trong tổng hợp logic chưa thể tính tới được. Phân tích cho ta biết kết quả phản ứng của mạch lên những tập giá trị biến khác nhau, rút ra những phương trình toán học, mô tả chức năng logic của thiết bị. Trên cơ sở đó cho phép ta sửa chữa, thay thế, thay đổi kết cấu của mạch mà vẫn không làm thay đổi chức năng của thiết bị.

Bài toán phân tích tùy theo nội dung và mục đích có thể là bài toán phân tích tĩnh, phân tích động hay kiểm tra và chẩn đoán kỹ thuật. Trong chương này chúng ta sẽ đề cập tới những bài toán phân tích thuộc các nhiệm vụ nói trên.

5.1. PHÂN TÍCH TĨNH THIẾT BỊ SỐ TỔ HỢP

5.1.1. Nhiệm vụ bài toán

Nhiệm vụ của bài toán phân tích tĩnh mạch tổ hợp là xác định chức năng của mạch. Nói cách khác là phải xác định được mối quan hệ giữa đầu ra và đầu vào của

thiết bị và biểu diễn dưới một dạng nào đó, thí dụ dạng biểu thức đại số của quan hệ vào ra hay dạng bảng giá trị thực của hàm ra của thiết bị.

Để đạt được mục đích đó, trong phân tích tĩnh cho phép coi tất cả tín hiệu trong mạch là tín hiệu logic lý tưởng và các phần tử logic dùng trong mạch là không giữ chậm tín hiệu. Cũng chính vì thế, phân tích tĩnh chỉ cho phép xác định xem mạch có thực hiện đúng chức năng đã thiết kế hay không và cho biết chức năng thiết kế của mạch là gì, thông qua thí dụ bảng giá trị thực của hàm mà nó thực hiện. Phân tích tĩnh không có khả năng phát hiện những sai sót do yếu tố thời gian truyền tín hiệu gây ra.

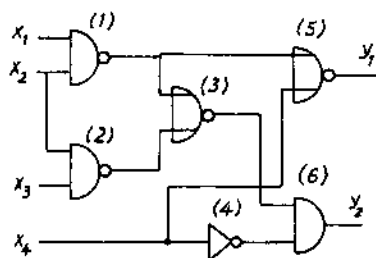
Để tiến hành phân tích tĩnh, ta cho tác động vào mạch từng tập giá trị biến rồi lần lượt xác định tín hiệu phản ứng ở mọi điểm trung gian - các đầu ra của tất cả phần tử logic, từ đầu vào tới đầu ra của mạch. Để làm việc này ta phân các phần tử mạch thành các lớp và tìm phản ứng ra lần lượt của các lớp.

Thí dụ 5.1: Tìm tín hiệu ra của mạch ở hình 5.1 dưới tác động của tập biến vào $(x_1 x_2 x_3 x_4) = 1010$.

Phân lớp các phần tử, ta có:

- Lớp 1 gồm các phần tử: (1), (2), (4)
- Lớp 2 gồm: (3), (5)
- Lớp 3 gồm: (6)

Tín hiệu tác động và tín hiệu ra ở các lớp phần tử như bảng 5.1.



Hình 5.1. Mạch dùng để phân tích tĩnh

Bảng 5.1. Bảng mô tả các lớp, tín hiệu vào, ra của các phần tử và của mạch

Lớp	Phần tử	Tín hiệu vào	Tín hiệu ra	y_1	y_2
1	(1)	10	1	1	0
	(2)	01	1		
	(4)	0	1		
2	(3)	11	0	1	0
	(5)	10	1		
3	(6)	01	0		

Cách tìm phản ứng của mạch đối với tập biến vào như vậy cho phép xác định cả các giá trị trung gian, đồng thời so sánh được độ dài đường truyền tín hiệu tới các đầu ra. Nếu tiến hành xác định phản ứng ra của mạch đối với tất cả 2^n tập giá trị của n biến thì kết quả thu được sẽ phản ánh chức năng của mạch ở dạng bảng giá trị thực. Từ đó dễ dàng rút ra hàm chức năng của mạch. Hình 5.2a là bảng giá trị thực thu được

do kết quả phân tích mạch ở hình 5.1. Hình 5.2b là mô tả các hàm ra y_1 và y_2 trên bảng Karnaugh.

x_1	x_2	x_3	x_4	y_1	y_2	x_1	x_2	x_3	x_4	y_1	y_2
0	0	0	0	1	0	1	0	0	0	1	0
0	0	0	1	1	0	1	0	0	1	1	0
0	0	1	0	1	0	1	0	1	0	1	0
0	0	1	1	1	0	1	0	1	1	1	0
0	1	0	0	1	0	1	1	0	0	1	0
0	1	0	1	1	0	1	1	0	1	0	0
0	1	1	0	1	0	1	1	1	0	1	0
0	1	1	1	1	0	1	1	1	1	0	1

a)

		x_1		x_2				x_1		x_2			
x_3	x_4	1	1	1	1	y_1	x_3	0	0	0	0	y_2	
		1	0	1	1			0	1	0	0		
		1	0	1	1			0	0	0	0		
		1	1	1	1			0	0	0	0		

b)

Hình 5.2. Hàm - kết quả phân tích mạch (a) và biểu diễn chúng lên bảng kế cận (b)

Dạng giải tích của hai hàm y_1 và y_2 :

$$\left. \begin{aligned} y_1 &= \bar{x}_1 + \bar{x}_2 + \bar{x}_4 \\ y_2 &= x_1 x_2 x_3 x_4 \end{aligned} \right\} \quad (5.1)$$

Hệ hàm (5.1) rút ra được trong trường hợp này không nhất thiết phản ánh đúng số lượng và chủng loại các phần tử logic trong mạch, nhưng bao giờ cũng phản ánh đúng chức năng mà mạch thực hiện.

5.1.2. Thủ tục chung phân tích tĩnh mạch tổ hợp

Trong trường hợp cần biết biểu thức toán học phản ánh cấu trúc logic gồm loại các phần tử, cách liên kết giữa chúng và chức năng của mạch, ta tiến hành theo các bước sau đây:

- Ký hiệu tất cả các đầu vào và đầu ra của tất cả các phần tử logic trong mạch.
- Xuất phát từ đầu ra tới đầu vào, mỗi lần gặp một ký hiệu là đầu ra của phần tử logic, ta thay ký hiệu đó bằng biểu thức mô tả chức năng của phần tử này.

Cuối cùng ta sẽ được một biểu thức đại số thể hiện mối quan hệ giữa tín hiệu đầu ra với các tín hiệu đầu vào, thể hiện chức năng của mạch.

Thí dụ 5.2

Tìm hàm y của mạch cho trên hình 5.3.

Bước 1: Ký hiệu tất cả các điểm vào, ra của các phần tử mạch: $x_1, x_2, x_3, x_4, a, b, c, d, e, f$ và y .

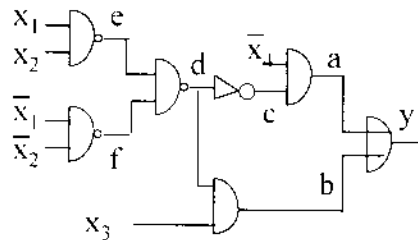
Bước 2: Ta có $y = a + b$

Thay a và b bằng hàm tương ứng của các phần tử logic, ta có:

$$y = \bar{x}_4.c + x_3.d$$

Thay c và d lần lượt, ta có:

$$y = \bar{x}_4.\bar{d} + x_3.(e \mid f) = \bar{x}_4(\overline{e \mid f}) + x_3.(e \mid f)$$



Hình 5.3. Mạch dùng thực hiện phân tích tĩnh

Thay e và f, ta có:

$$y = \bar{x}_4 \overline{((x_1 \mid x_2) \mid (\bar{x}_1 \mid \bar{x}_2))} + x_3((x_1 \mid x_2) \mid (\bar{x}_1 \mid \bar{x}_2)) \quad (5.2)$$

Biểu thức (5.2) là hàm chức năng của mạch ở hình 5.3, nó phản ánh đúng số lượng, loại và sự nối ghép giữa các phần tử mạch. Từ dạng (5.2), khi cần dễ dàng đưa về dạng mô tả chuẩn nào đó như bảng giá trị thực.

Như vậy phân tích cho phép tìm hàm chức năng của mạch và trên cơ sở đó, trong sửa chữa có thể biến đổi, thay thế các phần tử mạch sao cho phù hợp với điều kiện cụ thể mà vẫn không làm thay đổi chức năng của mạch.

5.2. PHÂN TÍCH ĐỘNG THIẾT BỊ SỐ TỔ HỢP

Phân tích động nhằm kiểm tra hoạt động của mạch về quan hệ thời gian, kiểm tra sự hoạt động trên thực tế của mạch để đối chiếu với mục đích đã đề ra trong thiết kế logic. Vì vậy phân tích động thường được tiến hành sau khi thiết kế. Một thiết bị sau khi thiết kế đem thử nghiệm nếu các tham số của tín hiệu ra không đúng như thiết kế thì bắt buộc phải tiến hành phân tích động.

Trong phân tích động có thể coi dạng của các loại tín hiệu là lý tưởng. Khi đó phân tích sẽ cho ta sự ảnh hưởng của việc giữ chậm tín hiệu khi đi qua các phần tử logic. Thực tế cho thấy thời gian giữ chậm của các phần tử logic thường khác nhau, ngay cả với các loại phần tử cùng một loại, cùng một loạt sản xuất. Tuy vậy để bài toán không quá phức tạp, chúng ta giả thiết thời gian giữ chậm trung bình của các phần tử logic là như nhau và ký hiệu là τ . Khi đó nhiệm vụ cơ bản của phân tích động là chỉ ra thời gian quá trình quá độ của mạch, những thay đổi gì so với thiết kế xảy ra đối với các tín hiệu ra trong các trường hợp sau đây:

1. Thay đổi từ tập giá trị biến này sang tập giá trị biến khác của các biến vào.

2. Khi có tác động liên tiếp một chuỗi các tập giá trị biến của các biến vào.

3. Khi thay đổi từ một tập giá trị biến này sang một tập giá trị biến khác bất kỳ trong $2^n - 1$ tập giá trị biến còn lại, hay thay đổi thứ tự và / hoặc thay đổi nội dung của chuỗi tác động.

Ta hãy xét lần lượt ba trường hợp trên đây.

5.2.1. Trường hợp khi thay đổi từ một tập giá trị biến này sang một tập giá trị biến khác

Để khảo sát ta giả thiết ở đầu vào của thiết bị có sự thay đổi từ tập X_i sang tập X_k của các biến vào.

Để thấy rõ ảnh hưởng của sự giữ chậm tín hiệu của các phần tử logic lên kết quả ra, trong trường hợp này ta tiến hành phân tích theo các bước sau:

1. Xác định cấp của mạch W . Cấp của mạch là số lượng lớn nhất các phần tử logic mà tín hiệu phải đi qua từ đầu vào tới đầu ra của mạch. Đánh số thứ tự tất cả các phần tử logic trong mạch.

2. Lập một bảng có $(W + 1)$ cột và $(n + m)$ hàng. Trong đó n là số biến vào của thiết bị và m là số phần tử logic trong mạch. Đánh số thứ tự các cột từ 0 đến W . n hàng đầu tương ứng với n biến; m hàng cuối tương ứng với m đầu ra của m mạch logic. Hàng cuối cùng chính là đầu ra của mạch.

3. Điền vào bảng theo cách sau:

- + Tập giá trị biến thứ hai, tức tập X_k , được điền vào tất cả các cột trong phần của các biến.
- + Trong cột 0 ta điền giá trị logic ở các đầu ra ở chế độ tĩnh của các phần tử logic khi tác động đầu vào là tập giá trị biến thứ nhất X_i .

4. Tiến hành điền vào các cột còn lại của bảng các giá trị đầu ra của các phần tử logic bắt đầu từ cột 1 đến cột W , xuất phát từ trạng thái ban đầu ở trên các đầu ra của các phần tử này, dưới tác động của tập giá trị biến thứ hai X_k . Khi điền vào cột thứ j ta lấy thông tin ở cột thứ $(j - 1)$.

Vì chiều ngang của bảng chính là trục thời gian, mà mỗi cột có thời gian là 1τ , nên diễn biến về giá trị ở đầu ra của phần tử cuối cùng của mạch sẽ chính là giản đồ thời gian thực tế, có tính tới sự giữ chậm tín hiệu khi đi qua các phần tử logic của mạch. So sánh kết quả sự phân tích động này với kết quả phân tích tĩnh đối với từng tập giá trị biến, chúng ta có thông tin về sự đúng đắn, chân thực của tín hiệu ra.

Thí dụ 5.3

Cho mạch ở hình 5.4. Hãy phân tích phản ứng của mạch khi thay đổi trạng thái vào từ tập giá trị biến $X_i = 000$ sang $X_k = 011$.

Ta nhận xét rằng, vì hai tập giá trị biến này có thể coi là hai tập thể nên đây là trường hợp thay đổi từ một tập thể này sang một tập thể khác.

Bước 1: Cấp của mạch trong trường hợp này bằng 3. Đánh số các phần tử mạch như trên hình 5.4, ta có $n = 3$, $m = 4$.

Bước 2: Vậy bảng có $W + 1 = 4$ cột. Số hàng bằng $3 + 4 = 7$.

Bước 3: Bố trí các cột và hàng như trên hình 5.5. Trong đó tập biến $X_k = 011$

được điền trong tất cả các cột, ứng với phần của các biến. Vì với tập $x_1x_2x_3 = 000$, tín hiệu ở các đầu ra của các phần tử logic như sau:

(1) : 1 (2) : 0
(3) : 0 (4) : 0

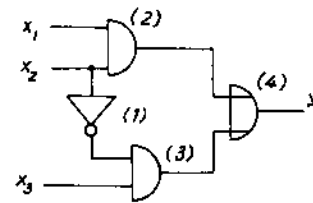
Do đó cột thứ nhất của bảng chứa các giá trị ban đầu là 1000, tương ứng với các phần tử (1), (2), (3), (4).

Bước 4: Vì khi tín hiệu truyền qua các lớp phần tử logic, giá trị của tập biến vào thứ hai 011 không đổi, nhưng giá trị logic ở đầu ra của các phần tử (1), (2), (3), (4) thì thay đổi tùy theo tín hiệu x_1, x_2, x_3 đã truyền tới phần tử logic nào. Kết quả ta có bảng ở hình 5.5. Kết quả xét thêm các cột tiếp theo $W + 1$, $W + 2$, ... nếu có sẽ hoàn toàn giống cột số 3, thể hiện trạng thái ổn định tại đầu ra của mạch.

Phân tích kết quả này ta thấy rằng, có một giai đoạn, tại cột 2 phản ứng của mạch là không giống như thiết kế tĩnh. Theo thiết kế tĩnh, với tập 011 thì đầu ra phải bằng 0, nhưng ở đây đầu ra lại bằng 1. Thời gian tồn tại của tín hiệu 1 không đúng này là 1τ .

Tín hiệu này gọi là tín hiệu tranh chấp trạng thái và thời gian cần thiết để tín hiệu ra ổn định là $t = W \cdot \tau$.

Để mạch trên đây làm việc đúng, cần có biện pháp khử bỏ tín hiệu tranh chấp trạng thái ở đầu ra của mạch.



Hình 5.4. Mạch dùng phân tích

	0	1	2	3
x_1	0	0	0	0
x_2	1	1	1	1
x_3	1	1	1	1
(1)	1	0	0	0
(2)	0	0	0	0
(3)	0	1	0	0
(4)	0	0	1	0

Hình 5.5. Bảng phân tích

5.2.2. Khi tác động liên tiếp một chuỗi các tập giá trị biến vào

Trong trường hợp này việc phân tích cũng tiến hành tương tự như trong trường hợp thứ nhất, nhưng có những đặc điểm sau đây:

1) Khung của bảng

Một khi đã đề cập tới chuỗi tập giá trị biến vào tác động liên tục thì tất yếu phải nói tới thời gian tồn tại của mỗi tập giá trị biến. Để tiện ta sẽ lấy đơn vị thời gian là thời gian giữ chậm trung bình của một phần tử logic và bằng τ . Như vậy mỗi tập giá trị biến có thể tồn tại từ 1τ đến nhiều τ . Để tập cuối ứng với trường hợp đầu ra tín hiệu đủ thời gian thiết lập giá trị ổn định, tập cuối cùng giả thiết tồn tại một thời gian có số τ lớn hơn hay bằng số cấp của mạch. Cách xây dựng bảng cũng khác:

+ Số cột của bảng bằng tổng số τ của chuỗi vào tác động, được đánh số thứ tự từ 0 trở đi.

+ Số hàng vẫn bằng $(n + m)$; n là số biến vào, m là số phần tử logic trong mạch.

2) Cách điền vào bảng

+ Trong phần các biến, tại các cột, theo thứ tự xuất hiện các tập giá trị biến, ta điền vào các cột các tập giá trị biến của chuỗi vào, tập biến nào tồn tại bao nhiêu τ thì điền vào bấy nhiêu cột.

+ Trong phần đầu ra các phần tử logic của bảng, cột đầu tiên ta ghi tất cả giá trị ra là bất định (x) vì chưa hết một τ thì chưa một tín hiệu nào đi qua được bất kỳ phần tử logic nào.

Để điền vào phần còn lại của bảng, vẫn áp dụng nguyên tắc điền vào cột j thì dùng thông tin ở cột $(j - 1)$.

Thí dụ 5.4

Cho mạch ở hình 5.4 trên đây, hãy xác định phản ứng ra của mạch khi cho chuỗi tác động vào là:

$$x_1x_2x_3 = (011(4\tau), 111(3\tau), 101(2\tau), 000(1\tau), 100(3\tau))$$

Thực hiện việc lập bảng và điền vào bảng theo nguyên tắc đã chỉ ra trên đây, ta có kết quả như ở hình 5.6a.

Tín hiệu trên đầu ra (4) chính là tín hiệu ra của mạch. Giảm đồ thời gian của tín hiệu ra trong chế độ động cho trên hình 5.6b. Để biết xem phản ứng ra ở chế độ động có giống như trong chế độ tĩnh - là chế độ làm việc theo thiết kế hay không, ta tiến hành phân tích tĩnh đối với mạch đã cho để tìm phản ứng của mạch đối với từng tập giá trị biến của chuỗi.

Ta có:

$$y = x_1x_2 + \bar{x}_2x_3$$

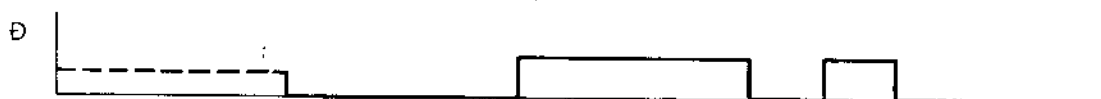
Bảng giá trị thực của hàm như sau:

x_1	x_2	x_3	y	x_1	x_2	x_3	y
0	0	0	0	1	0	0	0
0	0	1	1	1	0	1	1
0	1	0	0	1	1	0	1
0	1	1	0	1	1	1	1

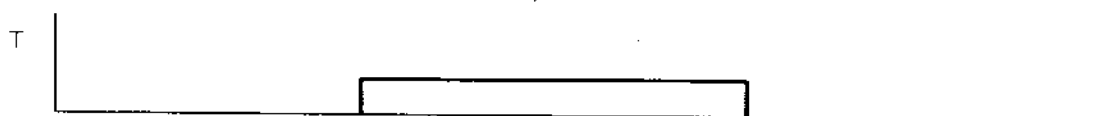
Từ đây dễ dàng vẽ được giản đồ thời gian của tín hiệu ra y trong chế độ tĩnh như hình 5.6c.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
x_1	0	0	0	0	1	1	1	1	1	0	1	1	1	x
x_2	1	1	1	1	1	1	1	0	0	0	0	0	0	x
x_3	1	1	1	1	1	1	1	1	1	0	0	0	0	x
(1)	x	0	0	0	0	0	0	0	1	1	1	1	1	1
(2)	x	0	0	0	0	1	1	1	0	0	0	0	0	0
(3)	x	x	1	0	0	0	0	0	0	1	0	0	0	0
(4)	x	x	x	0	0	0	1	1	1	0	1	0	0	0

a)



b)



c)

Hình 5.6. Bảng phân tích khi có một chuỗi tác động vào liên tiếp (a);
giản đồ thời gian của tín hiệu ra ở trạng thái động (b)
và ở trạng thái tĩnh (c)

So sánh hai kết quả ta thấy:

+ So với chế độ tĩnh, trong chế độ động ở ba cột đầu tiên tín hiệu không thể xác định rõ là 0 hay 1. Mạch ở chế độ bất định.

+ Tín hiệu "1" đối với các tập 111 và 101 bị co lại mất hai cột.

+ Tín hiệu ra tại cột 10 lẽ ra phải bằng 0, nhưng thực tế lại có một xung, với độ rộng là 1τ . Đó chính là do hậu quả của việc tranh chấp trạng thái của tín hiệu tại đầu ra của mạch. Lý do của sự tranh chấp này là sự giữ chậm tín hiệu khác nhau của các tín hiệu vào khi đi theo những đường khác nhau để tới đầu ra. Tín hiệu tranh chấp

trạng thái này cần phải khử bỏ vì trong nhiều trường hợp nó có thể gây nên những sự sai lệch về hoạt động chức năng của mạch.

5.2.3. Các trường hợp khác

Đó là các trường hợp khi thay đổi từ một tập giá trị biến này sang một tập bất kỳ trong $2^n - 1$ tập giá trị biến khác hay khi thay đổi về nội dung và thứ tự của chuỗi tác động vào.

Các trường hợp này về bản chất không khác hai trường hợp trên đây, có điều chúng ta phải lập cho mỗi trường hợp một bảng tương tự như của hai trường hợp trên. Khi đã biết phương pháp phân tích của hai trường hợp nói trên, dùng công cụ máy tính ta có thể lập trình cho các trường hợp khác bằng cách thay đổi tham số của chương trình.

5.3. BẢN CHẤT CỦA TRANH CHẤP TRẠNG THÁI TRONG MẠCH TỔ HỢP, PHƯƠNG PHÁP PHÁT HIỆN VÀ KHỬ BỎ

5.3.1. Bản chất của tranh chấp trạng thái trong mạch tổ hợp

Để hiểu bản chất của hiện tượng tranh chấp trạng thái và có biện pháp khắc phục, ta hãy xét nguyên tắc hoạt động của các mạch logic cơ bản VA và HOAC hai đầu vào.

Ta thấy rằng, đối với mạch VA hai đầu vào, khi hai đầu có giá trị tương ứng là 0 và 1 thì đầu ra bằng 0 và khi cả hai đầu cùng thay đổi ngược chiều nhau, nghĩa là đối với tập giá trị biến 1 và 0 thì đầu ra cũng nhận giá trị 0. Nhưng để đầu ra luôn luôn là 0 thì điều quan trọng là giá trị ở hai đầu vào *phải thay đổi đồng thời*. Đây là một điều kiện khó thực hiện hoặc không phải lúc nào cũng xảy ra đúng vì thực tế các tín hiệu đầu vào được hình thành một cách độc lập tương đối. Khi đã không đảm bảo chắc chắn thay đổi đồng thời thì giả sử đầu vào lúc đầu có giá trị 0 thay đổi trước, khi đó ở một khoảng khắc, giá trị của tập biến vào sẽ là 11. Khi đó đầu ra sẽ có giá trị tương ứng là 1 chứ không phải là 0 nữa. Hình ảnh của đầu ra sẽ là từ giá trị 0 lướt qua giá trị 1 rồi lại về 0, tức ta có : 0-1-0.

Lý luận một cách tương tự đối với mạch HOAC, ta sẽ có diễn biến của đầu ra là 1-0-1.

Một kết luận tương tự hoàn toàn đúng cho các mạch loại VA-KH, HOAC-KH. Nghiên cứu đối với một số mạch khác như mạch cộng môđun 2, mạch kéo theo, mạch đẳng trị và mạch cấm ta có thể rút ra kết luận đối với từng loại mạch (hình 5.7).

x_1	x_2	VA	HOAC	VA-KH	HOAC-KH	(+)	Đẳng trị	Kéo theo	Cấm
0	0	0	0	1	1	0	1	0	0
0	0-1	0	0-1	1	1-0	0-1	1-0	1	0
0	1-0	0	1-0	1	0-1	1-0	0-1	1	0
0	1	0	1	1	0	1	0	1	0
0-1	0	0	0-1	1	1-0	0-1	1-0	1-0	0-1
0-1	0-1	0-1	0-1	1-0	1-0	0-1-0	1-0-1	1-0-1	0-1-0
0-1	1-0	0-1-0	1-0-1	1-0-1	0-1-0	1-0-1	0-1-0	1-0	0-1
0-1	1	0-1	1	1-0	0	1-0	0-1	1	0
1-0	0	0	1-0	1	0-1	1-0	0-1	0-1	1-0
1-0	0-1	0-1-0	1-0-1	1-0-1	0-1-0	1-0-1	0-1-0	0-1	1-0
1-0	1-0	1-0	1-0	0-1	0-1	0-1-0	1-0-1	1-0-1	0-1-0
1-0	1	1-0	1	0-1	0	0-1	0-0	1	0
1	00	1	1	0	1	0	0	1	0
1	0-1	0-1	1	1-0	0	1-0	0-1	0-1	1-0
1	1-0	1-0	1	0-1	0	0-1	1-0	1-0	0-1
1	1	1	1	0	0	0	1	1	0

Hình 5.7. Bảng về hiện tượng tranh chấp của các mạch logic cơ bản

Từ kết quả ta rút ra các kết luận sau đây:

a) Đối với các mạch VA, HOAC, VA-KH và HOAC-KH sẽ có tranh chấp tĩnh, nếu hai biến vào thay đổi ngược chiều nhau.

b) Đối với các mạch Kéo theo và mạch Cấm sẽ có tranh chấp tĩnh nếu hai biến vào thay đổi giá trị cùng chiều.

c) Đối với các mạch cộng môđun 2 và đẳng trị: Khi các biến đầu vào thay đổi cùng chiều hoặc ngược chiều đều có tranh chấp.

Ta có thể khái quát về hiện tượng có tranh chấp trạng thái đối với mạch VA, HOAC, VA-KH và HOAC-KH nhiều đầu vào như sau.

Sẽ xuất hiện tranh chấp nếu:

- Ít nhất trên hai đầu vào các biến thay đổi giá trị của mình theo chiều ngược nhau.

- Những biến không thay đổi giá trị thì không có vai trò quyết định giữ cho biến ra không thay đổi.

5.3.2. Phân loại tranh chấp trạng thái

Việc phân loại tranh chấp được căn cứ vào phản ứng đúng ở đầu ra của mạch. Đó là các giá trị trong chế độ tĩnh, đối với hai tập biến đang xét, tại đầu ra của chúng.

Tranh chấp chia ra thành tranh chấp tĩnh và tranh chấp động.

Tranh chấp tĩnh là tranh chấp xảy ra khi phản ứng đúng của mạch đối với hai tập giá trị biến đang xét đều bằng 0 hay đều bằng 1.

Tranh chấp tĩnh lại chia ra thành tranh chấp tĩnh loại 0 và tranh chấp tĩnh loại 1.

Tranh chấp tĩnh loại 0 là tranh chấp xảy ra khi phản ứng đúng đối với hai tập giá trị biến đều bằng 0.

Tranh chấp tĩnh loại 1 là tranh chấp xảy ra khi phản ứng đúng của mạch đối với hai tập giá trị biến đang xét đều bằng 1.

Nếu tranh chấp xảy ra khi phản ứng của mạch đối với hai tập giá trị biến không giống nhau thì gọi là tranh chấp động.

Tranh chấp động lại chia ra tranh chấp động loại 0-1 và tranh chấp động loại 1-0.

Tranh chấp động loại 0-1 là loại tranh chấp xảy ra khi phản ứng đúng của mạch đối với tập giá trị biến thứ nhất bằng 0, còn đối với tập giá trị biến thứ hai bằng 1. Đối với tranh chấp động loại 1-0 thì ngược lại.

Thí dụ 5.5

Cho mạch ở hình 5.8. Hãy phân tích hoạt động của mạch trong chế độ động khi đầu vào thay đổi từ tập giá trị biến $X_i = 000$ sang $X_j = 111$.

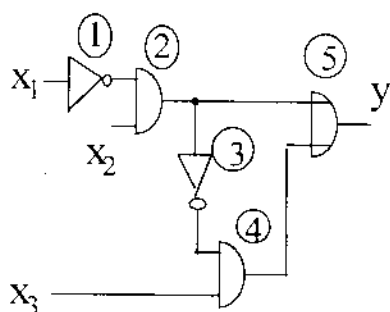
Sử dụng thủ tục phân tích động trên đây ta có bảng phân tích như ở hình 5.9.

Kết quả phân tích tĩnh để xác định chức năng của mạch cho thấy đối với tập giá trị biến 000, giá trị đúng của đầu ra là 0, còn đối với tập giá trị biến 111, giá trị đúng của nó là 1.

Trong chế độ động, khi thay đổi các tập giá trị biến vào từ trạng thái vào 000 sang 111 thì trong mạch xảy ra tranh chấp ở cột số 4. Tại đó lẽ ra giá trị đầu ra phải là 1 thì lại xuất hiện giá trị 0. Đó là tranh chấp động và là tranh chấp động loại 0-1.

5.3.3. Tranh chấp trạng thái ở các mạch số

Tranh chấp trạng thái ở các mạch, các thiết bị số là do có sự tranh chấp trạng thái của các phần tử logic tham gia trong mạch trong quá trình làm việc. Do đó việc phát hiện khả năng xuất hiện tranh chấp đối với các phần tử logic tổ hợp dẫn tới việc kiểm tra tuần tự từ đầu ra tới đầu vào của từng phần tử logic.



Hình 5.8. Mạch dùng cho thí dụ 5.5

	0	1	2	3	4	5
x_1	1	1	1	1	1	1
x_2	1	1	1	1	1	1
x_3	1	1	1	1	1	1
(1)	1	0	0	0	0	0
(2)	0	1	0	0	0	0
(3)	1	1	0	1	1	1
(4)	0	1	1	0	1	1
(5)	0	0	1	1	0	1

Hình 5.9. Bảng phân tích

Đối với mạch hai cấp thực hiện theo dạng tuyến chuẩn, các tranh chấp tĩnh chỉ có thể xảy ra giữa hai tập giá trị biến vào có tính chất sau đây:

- Chúng là các tập giá trị biến mà hàm nhận giá trị 1.
- Chúng là các tập giá trị biến kế cận.

- Trong khi thực hiện bằng mạch chúng không được thực hiện bởi một phần tử logic VA. Nói cách khác chúng không được phủ bởi cùng một implicăng đơn giản thực thụ. Trong trường hợp này, khi trạng thái vào thay đổi từ tập giá trị biến này sang tập giá trị biến khác, ở đầu ra của một phần tử logic VA sẽ có thay đổi từ 0 lên 1 và ở đầu ra của phần tử logic VA khác sẽ có thay đổi từ 1 xuống 0. Hai đầu ra này lại chính là hai đầu vào của phần tử logic HOAC. Do đó sẽ xảy ra tranh chấp mà là tranh chấp tĩnh loại 1, vì đối với cả hai tập giá trị biến này giá trị đúng của đầu ra của mạch (và cũng là của mạch HOAC) là 1.

Thí dụ 5.6

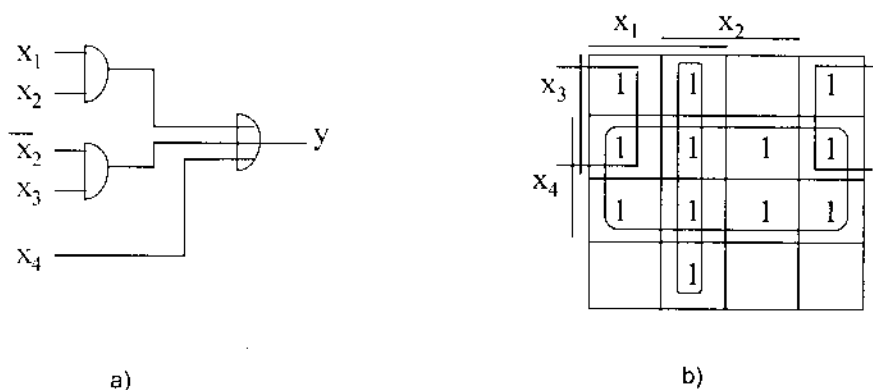
Cho mạch ở hình 5.10a. Tìm hai tập giá trị biến gây tranh chấp trạng thái của mạch.

Ta dễ dàng nhận được hàm chức năng của mạch như sau:

$$y = x_1x_2 + \bar{x}_2x_3 + x_1$$

Đưa hàm này lên bảng Karnaugh (hình 5.10b). Ta thấy rằng hai tập biến $x_1x_2x_3\bar{x}_4$ và $x_1\bar{x}_2x_3\bar{x}_4$ là hai tập biến mà nếu chúng tác động liên tiếp sẽ gây nên tranh chấp trạng thái ở đầu ra của mạch, vì chúng không được phủ bằng một implicăng đơn giản thực thụ chung, mà ở hai implicăng đơn giản khác nhau được thực hiện bằng hai phần tử logic khác nhau. Do đó khi chuyển từ một tập này sang một tập

khác thì sự thay đổi giá trị trên hai phần tử logic này là ngược chiều nhau. Kết quả là ở đầu ra của mạch HOAC sẽ có tranh chấp.



Hình 5.10. Mạch và hàm của thí dụ 5.6

Qua phân tích ta thấy, điều kiện để mạch hai cấp thực hiện theo dạng tuyến chuẩn không có tranh chấp tĩnh loại 1 là tất cả các giá trị 1 của hàm trong các ô là kế cận nhau thì bắt buộc phải thực hiện bằng một phần tử logic VA. Nói cách khác, để mạch hai cấp dạng tuyến chuẩn không có tranh chấp tĩnh loại 1, một dấu, nó cần phải được thực hiện không phải bằng dạng tuyến chuẩn tối thiểu mà bằng dạng tuyến chuẩn tối thiểu đầy đủ. Dạng tuyến chuẩn tối thiểu đầy đủ là dạng tuyến chuẩn mà theo bảng Karnaugh, tất cả các giá trị 1 của hàm trong các ô là kế cận nhau phải được thực hiện bằng một implicăng đơn giản.

Mạch ở hình 5.10a sẽ không có tranh chấp trạng thái tĩnh loại 1 một dấu, nếu được thực hiện bằng dạng tuyến chuẩn đầy đủ sau đây (mạch hình 5.11):

$$y = x_1x_2 + \bar{x}_2x_3 + x_1x_3 + x_4$$

Trong đó hai tập giá trị biến mà hàm nhận giá trị 1 là $x_1\bar{x}_2x_3\bar{x}_4$ và $x_1x_2x_3\bar{x}_4$ là kế cận và được thực hiện bằng một implicăng đơn giản là x_1x_3 . Có thể chứng minh rằng mạch hai cấp thực hiện theo dạng tuyến chuẩn đã không có tranh chấp tĩnh loại 1, một dấu, thì cũng không có tranh chấp tĩnh loại 0 hoặc tranh chấp động một dấu. Song không có nghĩa là sẽ không có tranh chấp loại 2, 3 ... khác.

Một cách tương tự, có thể kết luận cho mạch hai cấp thực hiện theo dạng hội chuẩn.

Mạch hai cấp dạng hội chuẩn chỉ có thể có tranh chấp loại 0, một dấu, giữa hai tập giá trị biến nếu chúng có tính chất sau:

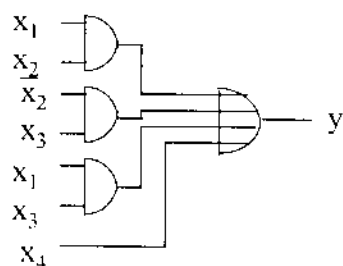
- + Chúng là các tập giá trị biến mà hàm nhận giá trị 0.
- + Chúng là các tập giá trị biến kế cận.
- + Chúng không được thực hiện bởi cùng một hàm M_i .

Để mạch hai cấp dạng hội chuẩn không có tranh chấp tĩnh loại 0, một dấu, điều kiện cần và đủ là hai tập giá trị biến mà hàm nhận giá trị 0 mà kế cận với nhau thì phải được thực hiện bằng một mạch HOAC, tức chúng phải được phủ chung bởi một thành phần M_i . Nói cách khác, mạch phải được thực hiện bằng dạng hội chuẩn tối thiểu đầy đủ.

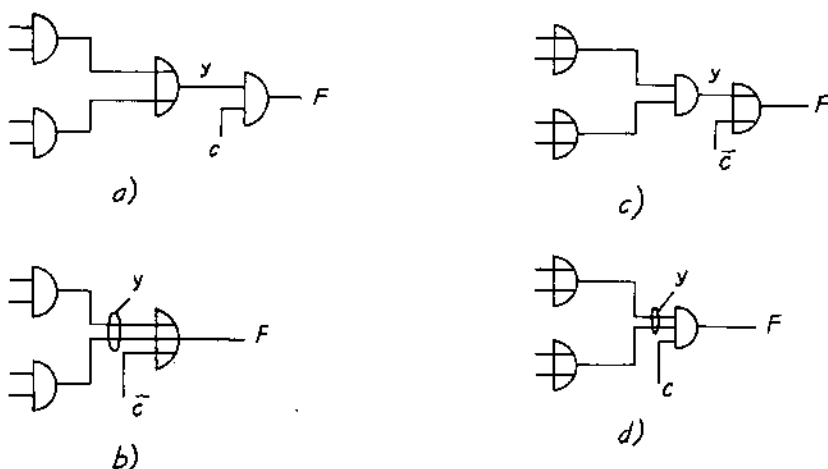
Nói tóm lại, dạng tuyến chuẩn tối thiểu đầy đủ hay dạng hội chuẩn tối thiểu đầy đủ là những dạng hạn chế rất nhiều hiện tượng tranh chấp trạng thái của mạch. Tuy vậy, cái giá phải trả là tốn kém nhiều linh kiện hơn.

Mặt khác cần thấy rằng, để tránh ảnh hưởng của hiện tượng tranh chấp trạng thái có thể dùng biện pháp đồng bộ ngoài. Dùng đồng bộ là chỉ lấy tín hiệu ra vào một thời điểm - thời điểm có xung đồng bộ, là thời điểm mà tín hiệu ra của mạch đã được ổn định.

Trên hình 5.12 là một số phương án sử dụng tín hiệu đồng bộ đối với hai dạng mạch: Đầu ra là mạch VA và mạch HOAC.



Hình 5.11. Mạch thực hiện bằng dạng tuyến chuẩn tối thiểu đầy đủ



Hình 5.12. Các phương án mắc tín hiệu đồng bộ

Tín hiệu ra y được lấy ra vào thời điểm tín hiệu đồng bộ $c = 1$. Khi $c = 0$ thì không lấy tín hiệu ra. Tín hiệu đồng bộ c là tín hiệu xung rất hẹp.

Với phương án ở hình 5.12a và d) ta có:

$$F = y \cdot c, \quad \text{do đó với } c = 1 \text{ thì } F = y; \text{ với } c = 0 \text{ thì } F = 0.$$

Như vậy bình thường $F = 0$ bất kể y bằng 0 hay 1. Khi $c = 1$ thì $F = y$. Tức là y chỉ được lấy ra trong thời gian rất ngắn và được lưu giữ lại nên không bị ảnh hưởng của hiện tượng tranh chấp trạng thái.

Với phương án ở hình 5.12b và c, ta có:

$$F = y + \bar{c}, \text{ do đó, với } c = 1 \text{ thì } F = y; c = 0 \text{ thì } F = 1.$$

Như vậy bình thường $F = 1$; khi $c = 1$ thì $F = y$.

Phương án a) và c) phải thêm mạch logic, còn phương án b) và d) phải cần thêm chân của mạch logic cuối cùng.

Ngoài phương án dùng đồng bộ ngoài, có thể dùng tụ có điện dung thích hợp mắc song song với đầu ra để khử các xung kim là các xung sinh ra do hiện tượng tranh chấp.

Vì hiện tượng tranh chấp sinh ra do việc truyền tín hiệu từ đầu vào tới đầu ra của mạch theo những con đường dài ngắn khác nhau, khi tổng hợp mạch đôi khi phải chấp nhận những phương án chưa tối ưu về mặt đơn giản để đảm bảo không có hiện tượng tranh chấp trạng thái.

5.4. KIỂM TRA VÀ CHẨN ĐOÁN HƯ HỎNG TRONG THIẾT BỊ SỐ TỔ HỢP

Kiểm tra và chẩn đoán các hư hỏng xảy ra trong thiết bị số tổ hợp là một bài toán phục vụ trực tiếp cho công tác khai thác, sửa chữa. Đó là một bài toán phức tạp và quan trọng. Ngày nay lý thuyết về dự báo và chẩn đoán kỹ thuật đã phát triển rất mạnh và trở thành một bộ môn khoa học riêng. Ở đây chỉ đề cập tới một số vấn đề cơ bản và cần thiết nhất.

5.4.1. Các loại hư hỏng

Việc phân tích thiết bị số, ngoài việc xác định và kiểm tra chức năng ra, nhiệm vụ phát hiện có hư hỏng hay không và nếu có thì hư hỏng ở đâu, hư hỏng kiểu gì, chấp xuống đất hay chấp lên nguồn, đứt mạch hay một dạng nào khác là rất thường gặp và quan trọng.

Những hư hỏng trong thiết bị số tổ hợp có những đặc trưng khác nhau và do nhiều nguyên nhân, có thể chia thành hai loại chính là: Hư hỏng cố định và hư hỏng không cố định; hư hỏng logic hay hư hỏng không logic.

Những hư hỏng làm thay đổi chức năng logic của mạch gọi là các *hư hỏng logic*. Ngoài ra có thể có những hư hỏng chỉ dẫn tới làm méo tín hiệu, là các hư hỏng không logic...

Trên thực tế hay gặp và quan trọng hơn cả vẫn là loại hư hỏng logic. Vì vậy ta sẽ chỉ đi sâu tìm hiểu phương pháp kiểm tra và phát hiện các hư hỏng logic.

Để rút ra thủ tục kiểm tra hư hỏng logic, ta đưa ra cách làm và một số khái niệm sau đây:

Đánh số thứ tự 1, 2, 3, ... tất cả đầu vào đầu ra của các phần tử logic, từ đầu vào tới đầu ra của của mạch. Nếu hư hỏng làm cố định giá trị logic tại điểm i ở mức 1 (viết tắt là (1/ i)) hay ở mức 0 (viết tắt là (0/ i)), ta có các hư hỏng logic loại hằng 1 và hằng 0 tương ứng.

Nếu trong mạch, tại một điểm i nào đó, có hư hỏng loại t ($t = 0$ hay 1), ta nói mạch sẽ thực hiện hàm $f(t/i)$. Trong đó f là hàm chức năng khi chưa có hư hỏng logic.

Thí dụ 5.7

Cho mạch thực hiện hàm $f = \overline{x_1 x_2} + \overline{x_3}$ như ở hình 5.13. Tìm hàm hỏng $f(t/i)$ của tất cả các điểm cần thiết trên mạch.

Các điểm được đánh dấu từ 1 đến 6.

- Nếu $t = 1$, lần lượt ở mọi điểm trong mạch, ta có:

$$f(1/1) = \overline{x_2} + \overline{x_3}; \quad f(1/2) = \overline{x_1} + \overline{x_3};$$

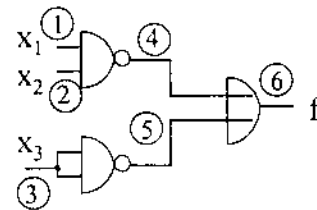
$$f(1/3) = \overline{x_1 x_2}; \quad f(1/4) = 1;$$

$$f(1/5) = 1; \quad f(1/6) = 1$$

- Nếu $t = 0$, lần lượt ở mọi điểm ta có các hàm sau:

$$f(0/1) = 1; \quad f(0/2) = 1; \quad f(0/3) = 1;$$

$$f(0/4) = \overline{x_3}; \quad f(0/5) = \overline{x_1 x_2}; \quad f(0/6) = 0$$



Hình 5.13. Mạch dùng xác định hàm hỏng

Trong thí dụ 5.7 ta chỉ xét các hư hỏng đơn, nghĩa là trong mạch luôn chỉ có một chỗ hỏng. Nếu trong mạch có từ hai chỗ hỏng trở lên cùng xảy ra thì ta có *hư hỏng kép*.

Số lượng hư hỏng đơn loại $f(t/i)$ là $2K$. Trong đó K là số lượng các điểm của mạch cần được đánh số. Số lượng các hư hỏng kép rất lớn và bằng $(3^K - 2K - 1)$. Hàm hư hỏng kép được ký hiệu như sau:

$f(t_1/i_1, t_2/i_2)$ là loại hai điểm hư hỏng đồng thời.

$f(t_1/i_1, t_2/i_2, t_3/i_3)$ là loại ba điểm hư hỏng đồng thời.

.....

Một vài loại hư hỏng kép của mạch ở hình 5.13 như sau:

$$f(1/1, 0/3) = 1; \quad f(1/1, 0/5) = \overline{x_2}$$

$$f(0/2, 1/3) = 1; \quad f(1/1, 0/2, 0/5) = 1$$

$$f(1/3, 0/4, 0/6) = 0; \quad \dots\dots\dots$$

Hư hỏng đơn là loại hư hỏng logic thường gặp. Phương pháp tìm kiếm các hư hỏng đơn tương đối đơn giản. Vì vậy trong phần này ta chỉ xem xét thủ tục phát hiện và khoanh vùng các hư hỏng logic đơn.

5.4.2. Phương pháp phát hiện các hư hỏng logic đơn

Để rút ra nhiệm vụ và thủ tục phát hiện các hư hỏng logic đơn trong mạch logic tổ hợp, ta hãy bắt đầu từ thí dụ 5.7. Trong hình 5.14 dưới đây là các hàm hư hỏng logic đơn của mạch ở hình 5.13.

$x_1x_2x_3$	$f1/1$	$f1/2$	$f1/3$	$f1/4$	$f1/5$	$f1/6$	$f0/1$	$f0/2$	$f0/3$	$f0/4$	$f0/5$	$f0/6$	f_d
0 0 0	1	1	0	1	1	1	1	1	1	1	0	0	1
0 0 1	0	0	0	1	1	1	0	0	1	0	0	0	0
0 1 0	1	1	0	1	1	1	1	1	1	1	0	0	1
0 1 1	1	0	0	1	1	1	0	0	1	0	0	0	0
1 0 0	1	1	0	1	1	1	1	1	1	1	0	0	1
1 0 1	0	1	0	1	1	1	0	0	1	0	0	0	0
1 1 0	1	1	1	1	1	1	1	1	1	1	1	0	1
1 1 1	1	1	1	1	1	1	0	0	1	0	1	0	1

Hình 5.14. Các hàm hư hỏng đơn của mạch ở hình 5.13

Trong đó cột đầu tiên ghi các tập giá trị biến có thể có của $x_1x_2x_3$. Cột cuối cùng ghi phản ứng đúng của mạch đối với mỗi tập giá trị biến. Trong các cột còn lại ghi giá trị của hàm hỏng đối với các loại hư hỏng logic đơn trong mạch.

Từ bảng này ta thấy rằng, đối với mạch ở hình 5.13, phản ứng đối với tất cả các tập giá trị biến của mạch khi có hỏng đều khác với phản ứng đúng của mạch. Do đó thủ tục đơn giản nhất và cũng là kém hiệu quả nhất để phát hiện xem trong mạch có hư hỏng hay không là :

- Cho tác dụng lần lượt tất cả các tập giá trị biến và ghi lại phản ứng ra tương ứng của mạch.

- So sánh kết quả với kết quả đúng đã biết trước.

Bằng cách đó có thể biết được có hay không có hỏng và hỏng liên quan tới tập giá trị biến nào.

Thủ tục để phát hiện xem trong mạch có hay không có hư hỏng gọi là *thủ tục kiểm tra*. Các tập giá trị biến vào được sử dụng trong thủ tục kiểm tra gọi là *các tập giá trị biến kiểm tra*.

Thủ tục kiểm tra chỉ cho phép phát hiện có hư hỏng logic hay không. Để biết được hư hỏng các loại, cần lập bảng kiểm tra. Muốn có bảng kiểm tra cần lập bảng phản ứng

của mạch giống như ở hình 5.14. Khi đó, bảng kiểm tra có thể nhận từ hình 5.14 bằng cách cộng môđun 2 giữa cột f_d với từng cột còn lại. Các giá trị 1 ở giao của cột i và hàng j sẽ cho biết rằng có hư hỏng loại $f(1/i)$. Bảng kiểm tra rút ra từ hình 5.14 cho trên hình 5.15.

$x_1x_2x_3$	$f1/1$	$f1/2$	$f1/3$	$f1/4$	$f1/5$	$f1/6$	$f0/1$	$f0/2$	$f0/3$	$f0/4$	$f0/5$	$f0/6$	f_d
000	0	0	1	0	0	0	0	0	0	0	1	1	0
001	0	0	0	1	1	1	0	0	1	0	0	0	0
010	0	0	1	0	0	0	0	0	0	0	1	1	0
011	1	0	0	1	1	1	0	0	1	0	0	0	0
100	0	0	1	0	0	0	0	0	0	0	1	1	0
101	0	1	0	1	1	1	0	0	1	0	0	0	0
110	0	0	0	0	0	0	0	0	0	0	0	1	0
111	0	0	0	0	0	0	1	1	0	1	0	1	0

Hình 5.15. Bảng kiểm tra

Vấn đề đặt ra là làm thế nào để có thể sử dụng một số lượng tối thiểu các tập giá trị biến kiểm tra mà vẫn có thể phát hiện được mọi hư hỏng logic có thể có trong mạch?

Để giải quyết vấn đề này ta có nhận xét sau đây: Bảng ở hình 5.15 cho kết quả cộng môđun 2 của hàm đúng với chính nó luôn luôn bằng 0. Điều đó có nghĩa là, nếu mạch làm việc đúng thì kết quả của bảng kiểm tra ở các cột đều phải bằng 0. Những cột chỉ chứa một số 1 thì có nghĩa rằng tập giá trị biến ở đầu hàng tương ứng là tập giá trị biến kiểm tra duy nhất có thể phát hiện hư hỏng $f(t/i)$, trong đó i là số thứ tự cột tương ứng. Nếu cột i nào đó có chứa từ N số 1, $0 \leq N \leq 2^n$, n là số biến vào của mạch, thì sẽ có N tập giá trị biến có khả năng phát hiện hư hỏng $f(t/i)$ tương ứng.

Như vậy sẽ có những tập giá trị biến cho phép phát hiện đồng thời một số loại hư hỏng khác nhau. Đó là các tập giá trị biến ở đầu hàng mà trong hàng đó có nhiều số 1. Mỗi số 1 của hàng này cho phép tập giá trị biến đó phát hiện hư hỏng tại cột tương ứng.

Như vậy chúng ta có thể dùng phương pháp Kvaiana tìm phủ để tìm tập hợp tối thiểu các tập giá trị biến cho phép phát hiện tất cả hư hỏng có thể có của mạch.

Tập hợp các tập giá trị biến kiểm tra cho phép phát hiện tất cả hư hỏng đơn của mạch gọi là *hệ các tập giá trị biến kiểm tra đầy đủ*. Nếu hệ này chỉ chứa một số tối thiểu các tập giá trị biến thì gọi là *hệ các tập giá trị biến kiểm tra tối thiểu*.

Thí dụ đối với bảng kiểm tra hình ở 5.15, hệ các tập giá trị biến kiểm tra tối thiểu gồm các tập sau: 0, 3, 5 và 7.

Các bước để tìm hệ tập các giá trị biến kiểm tra tối thiểu:

- Xác định bảng giá trị của hàm ra của mạch ứng với các trường hợp mạch làm việc đúng và các trường hợp hư hỏng loại $f(t/i)$.
- Lập bảng kiểm tra của mạch.
- Dùng phương pháp Kvaiana tìm phủ tối thiểu để xác định hệ các tập giá trị biến kiểm tra tối thiểu.

Áp dụng thủ tục này đối với mạch ở hình 5.13 ta được tập tối thiểu các tập giá trị biến kiểm tra như sau: 000, 011, 101, 111.

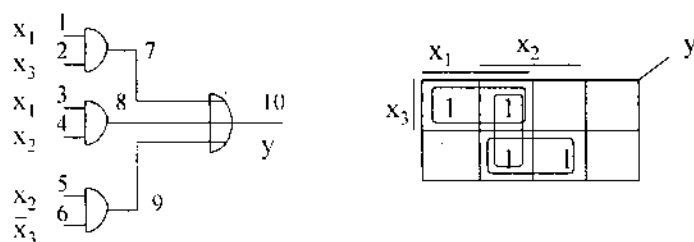
Tiếc rằng thủ tục kiểm tra và phát hiện hư hỏng trên đây chỉ có thể thực hiện tốt đối với các mạch được xây dựng trên cơ sở các hàm đã được tối thiểu hoá. Đối với các mạch được xây dựng theo phương pháp dùng các phần tử logic “thừa” để chống tranh chấp trạng thái, hay khi phải biến đổi biểu thức đại số logic để đảm bảo có thể thực hiện được bằng các phần tử logic có số đầu vào hạn chế thì phương pháp trên đây không xét hết được các hư hỏng.

Thí dụ đối với mạch xây dựng có dùng thêm phần tử logic để chống hiện tượng tranh chấp trạng thái (hình 5.16) thì không thể phát hiện hư hỏng loại $f(0/8)$.

Muốn phát hiện hư hỏng $f(0/8)$ phải tiến hành đo đạc ngay tại mạch VA này.

Ngoài ra cần chú ý rằng thủ tục kiểm tra chủ yếu cho phép xác định có các hư hỏng đơn hay không. Nó không cho phép phân biệt các hư hỏng, do đó không khoanh vùng được các hư hỏng một cách chi tiết và chưa đáp ứng được yêu cầu trong sửa chữa một cách cụ thể.

Nhiệm vụ khoanh vùng hư hỏng, tức xem hư hỏng xảy ra ở điểm nào trong mạch, là nhiệm vụ của *bài toán Chẩn đoán kỹ thuật*. Nó là bài toán tiếp theo của bài toán kiểm tra hư hỏng.



Hình 5.16. Mạch không thể phát hiện hư hỏng loại $f(0/8)$

5.4.3. Bài toán chẩn đoán kỹ thuật

Khi xây dựng bảng kiểm tra, ta thấy rằng đối với một tập giá trị biến nào đó có thể có một số hàm hỏng. Như vậy có thể dùng một chuỗi tập giá trị biến nào đó để

phân biệt từng hư hỏng. Thủ tục nhận chuỗi tập giá trị biến đó và cách tiến hành để phân biệt từng hư hỏng trong mạch gọi là *thủ tục chẩn đoán kỹ thuật*. Thủ tục chẩn đoán khái quát nhất sau khi đã có hệ các tập giá trị biến chẩn đoán gồm hai bước sau:

- Cho tác động vào mạch tất cả các tập giá trị biến của chuỗi và ghi lại kết quả.
- Phân tích kết quả để phân biệt từng hư hỏng, nghĩa là khoanh vùng được hư hỏng.

Các tập giá trị biến vào dùng trong trường hợp này gọi là các *tập giá trị biến chẩn đoán*.

Để phân tích chẩn đoán xem hư hỏng xảy ra ở đâu có thể thực hiện phân tích bằng kiểm tra. Qua đó rút ra bảng chẩn đoán và dấu hiệu phân biệt các hư hỏng.

Đối với thí dụ trên đây, bảng kiểm tra ở hình 5.15 có thể phân tích như sau:

- Đối với mỗi tập giá trị biến vào thì kết quả ra chia làm hai loại: Các cột chứa giá trị 0 và các cột chứa giá trị 1. Điều này có nghĩa là đối với mỗi tập giá trị biến kiểm tra (chẩn đoán) sẽ chia các hư hỏng của mạch thành hai loại, loại có phản ứng ra là 0 và loại có phản ứng ra là 1.

- Nếu ta tác động vào đầu vào một tập kiểm tra tiếp theo thì sẽ lại có sự phân chia thứ hai; kết quả của hai lần tác động sẽ cho ta bốn nhóm giá trị, ứng với bốn nhóm hư hỏng khác nhau loại: 0-0 ; 0-1 ; 1-0 ; 1-1.

- Nếu ta tác động liên tiếp không phải chỉ hai mà là ba, bốn và hơn nữa các tập giá trị biến kiểm tra thì tới một lúc nào đó, trong mỗi nhóm chỉ còn lại một hàm hỏng duy nhất, nghĩa là khi đó đã phân biệt được từng hàm hỏng. Nói cách khác, ta đã có thủ tục để khoanh vùng hư hỏng đơn.

Thí dụ, đối với bảng ở hình 5.15, cặp giá trị biến 011 và 111 chia các cột (các hàm hỏng) thành bốn nhóm sau đây:

- Ứng với các giá trị ghi được 0-0, có các cột: $f(0/1)$; $f(0/2)$; $f(0/4)$; $f(0/6)$
- Ứng với các giá trị ghi được 0-1, có các cột: $f(1/2)$; $f(1/3)$ $f(1/5)$ và f_0 .
- Ứng với các giá trị ghi được 1-0, không có cột nào.
- Ứng với các giá trị ghi được 1-1, có các cột: $f(1/1)$; $f(1/4)$; $f(1/5)$; $f(1/6)$ và $f(0/3)$.

Nếu dùng ba tập liên tiếp 3, 7, 0 ta sẽ có tám nhóm, số hàm hỏng trong mỗi nhóm sẽ ít đi:

- Ứng với các giá trị ghi được 0-0-0, có cột $f(0/6)$.
- Ứng với các giá trị ghi được 0-0-1, có các cột: $f(0/1)$; $f(0/2)$ $f(0/4)$.
- Ứng với các giá trị ghi được 0-1-0, có các cột: $f(1/3)$; $f(0/5)$.
- Ứng với các giá trị ghi được 0-1-1, có các cột: $f(1/2)$ và f_0 .
- Ứng với các giá trị ghi được 1-0-0, không có cột nào.

- Ứng với các giá trị ghi được 1-0-1, không có cột nào.
- Ứng với các giá trị ghi được 1-1-0, không có cột nào.
- Ứng với các giá trị ghi được 1-1-1, có các cột: $f(1/1)$; $f(1/4)$; $f(1/5)$; $f(1/6)$ và $f(0/3)$.

Như vậy, với tập hợp các tập giá trị biến 3, 7, 0 có thể phân biệt được hư hỏng đơn $f(0/6)$ với các hư hỏng khác, vì chỉ có $f(0/6)$ mới cho phản ứng ra 0-0-0.

Tất nhiên trong một mạch có thể có những hư hỏng không phân biệt được với nhau, đó là những hư hỏng mà các phản ứng ra của chúng hoàn toàn như nhau đối với mọi tập giá trị biến. Đối với chúng, các cột tương ứng trên bảng kiểm tra hoàn toàn giống nhau. Thí dụ trong các bảng ở hình 5.14 và 5.15 các hàm hỏng sau đây là không thể phân biệt được:

$f(1/4)$, $f(1/5)$, $f(1/6)$ và $f(0/3)$; $f(0/1)$, $f(0/2)$ và $f(0/4)$; $f(1/3)$ và $f(0/5)$.

- Nếu ta gộp các cột không phân biệt được lại với nhau, sẽ có bảng kiểm tra phân biệt được (hình 5.17). Qua bảng này thấy rằng, hư hỏng của nhóm $f(1/4)$ sẽ được phát hiện và phân biệt với các nhóm còn lại nhờ tập giá trị các biến vào 001, vì chỉ có nhóm hỏng $f(1/4)$ khi đó mới có phản ứng ra là 1.

Tập biến	$f(1/1)$	$f(1/2)$	$f(1/3)$	$f(1/4)$	$f(0/1)$	$f(0/6)$	f_d
0	1	1	0	1	1	0	1
1	0	0	0	1	0	0	0
2	1	1	0	1	1	0	1
3	1	0	0	1	0	0	0
4	1	1	0	1	1	0	1
5	0	1	0	1	0	0	0
6	1	1	1	1	1	0	1
7	1	1	1	1	1	0	1

Hình 5.17. Bảng kiểm tra phân biệt được

Tập hợp các tập giá trị biến vào cho phép phát hiện và khoanh vùng hư hỏng loại $f(t/i)$ gọi là tập hợp các tập giá trị biến chẩn đoán đối với hư hỏng $f(t/i)$.

Việc tìm tập hợp các tập giá trị biến chẩn đoán đối với hư hỏng loại $f(t/i)$ cũng thực hiện như cách tìm tập hợp các tập giá trị biến kiểm tra. Tức là từ bảng chẩn đoán đối với các hư hỏng phân biệt được, cộng môđun 2 cột $f(t/i)$ với các cột còn lại và bỏ cột $f(t/i)$ đi. Tập hợp các tập giá trị biến chẩn đoán tối thiểu của hư hỏng $f(t/i)$ sẽ bao gồm các tập giá trị biến là phủ của bảng vừa tìm được. Với bảng ở hình 5.17 ta có bảng chẩn đoán hư hỏng $f(1/1)$ như hình 5.18.

Tập biến	$f(1/2)$	$f(1/3)$	$f(1/4)$	$f(0/1)$	$f(0/6)$	f_d
0	0	1	0	0	1	0
1	0	0	1	0	0	0
2	0	1	0	0	1	0
3	1	1	0	1	1	1
4	0	1	0	0	1	0
5	1	0	1	0	0	0
6	0	0	0	0	1	0
7	0	0	0	1	1	0

Hình 5.18. Bảng chẩn đoán hư hỏng $f(1/1)$

Từ đây ta thấy, tập hợp các tập giá trị biến chẩn đoán đối với hư hỏng loại $f(1/1)$ là 1, 3 hoặc 3, 5. Vì các tập này cho kết quả phủ kín bảng.

Tập hợp tất cả các tập giá trị biến chẩn đoán cho phép phát hiện và khoanh vùng mọi hư hỏng và các nhóm hư hỏng đơn không phân biệt gọi là *tập hợp các tập giá trị biến chẩn đoán đầy đủ*.

Nếu tập hợp các tập giá trị biến chẩn đoán đầy đủ chỉ bao gồm một số tối thiểu các tập giá trị biến thì gọi là *tập hợp các tập giá trị biến chẩn đoán tối thiểu*.

Để tìm tập hợp các tập giá trị biến đầy đủ và tối thiểu cần:

1. Xây dựng tất cả các bảng chẩn đoán cho mọi hư hỏng $f(t/i)$.
2. Tìm phủ đồng thời của tất cả các bảng.

Để giảm bớt những việc phải làm trong việc tìm kiếm này, có thể dựa vào hai điểm sau đây:

a) Có những tập giá trị biến vào hoàn toàn tương đương về mặt phân biệt hư hỏng. Đó là các tập giá trị biến mà trong bảng chẩn đoán có các hàng tương ứng hoàn toàn giống nhau. Do đó trong khi lập bảng chẩn đoán, các hàng tương đương này được thay thế bằng một hàng đại diện chung cho chúng.

Trên bảng chẩn đoán ở hình 5.18, các tập giá trị biến 0, 2 và 4 là các tập tương đương.

b) Khi lập bảng chẩn đoán đầy đủ, thay cho việc xây dựng tất cả các bảng chẩn đoán hư hỏng $f(t/i)$ ta chỉ cần làm như sau:

Bước 1: Từ bảng đối với các hư hỏng đơn phân biệt được, đã thực hiện bước a) trên đây (hình 5.19), ta xây dựng bảng chẩn đoán cho cột hư hỏng cực trái $f(1/1)$.

Tập biến	$f(1/1)$	$f(1/2)$	$f(1/3)$	$f(1/4)$	$f(0/1)$	$f(0/6)$	f_d
0	1	1	0	1	1	0	1
1	0	0	0	1	0	0	0
3	1	0	0	1	0	0	0
5	0	1	0	1	0	0	0
6	1	1	1	1	1	0	1
7	1	1	1	1	1	0	1

Hình 5.19. Bảng hư hỏng đơn phân biệt được

Bước 2: Kiểm tra xem trong bảng chẩn đoán vừa xây dựng số cột có lớn hơn 1 hay không? Nếu số cột lớn hơn 1 thì làm lại bước 1. Nếu không, bảng chẩn đoán đầy đủ đã được xây dựng xong.

Áp dụng thủ tục trên đây, từ bảng ở hình 5.19 ta xây dựng được bảng chẩn đoán đầy đủ như ở hình 5.20.

Tập biến	$f(1/1)$						$f(1/2)$					$f(1/3)$					$f(1/4)$			$f(0/1)$		$f(0/6)$	
	1/2	1/3	1/4	0/1	0/6	f_d	1/3	1/4	0/1	0/6	f_d	1/4	0/1	0/6	f_d	0/1	0/6	f_d	0/1	0/6	f_d	0/6	f_d
0	0	1	0	0	1	0	1	0	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1
1	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0
3	1	1	0	1	1	1	0	1	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0
5	1	0	1	0	0	0	1	0	1	1	1	1	0	0	0	1	1	1	0	0	0	0	0
6	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	1	0	1	0	1	0	1
7	0	0	0	1	1	0	0	0	1	1	0	0	1	1	0	1	1	0	0	1	0	1	1

Hình 5.20. Bảng chẩn đoán đầy đủ

Trong bảng này, theo algôrit tìm phủ tối thiểu như đã trình bày trong phần đơn giản hàm, dễ dàng thu được phủ tối thiểu là các tập giá trị biến: 0, 3, 5 và 7.

Sau khi đã tìm được phủ tối thiểu của bảng chẩn đoán đầy đủ, ta hãy tách riêng từng bảng phần ứng với các hư hỏng phân biệt được (hình 5.17) phần ứng với các tập giá trị biến chẩn đoán 0, 3, 5, 7. Ta được bảng ở hình 5.21.

Tập biến	$f(1/1)$	$f(1/2)$	$f(1/3)$	$f(1/4)$	$f(0/1)$	$f(0/6)$	f_d
0	1	1	0	1	1	0	1
3	1	0	0	1	0	0	0
5	0	1	0	1	0	0	0
7	1	1	1	1	0	0	1

Hình 5.21. Phần bảng ứng với các hư hỏng phân biệt được

Bảng ở hình 5.21 có thể coi là từ điển các mã hư hỏng phân biệt được của mạch đã cho. Vì khi tác động vào mạch chuỗi các tập giá trị biến 0, 3, 5, 7 thì phản ứng của mạch ghi được ở đầu ra của mạch tại các cột, ứng với các hư hỏng khác nhau là khác nhau. Ví dụ đối với hư hỏng loại $f(1/2)$ ta có chuỗi 1011, với hư hỏng loại $0/6$ ta có 0000...

Kết quả ghi trong bảng ở hình 5.21 vừa cho phép phát hiện từng loại hư hỏng phân biệt được, vừa khoanh vùng được các hư hỏng, tức chỉ rõ hư hỏng là loại gì và tại chỗ nào.

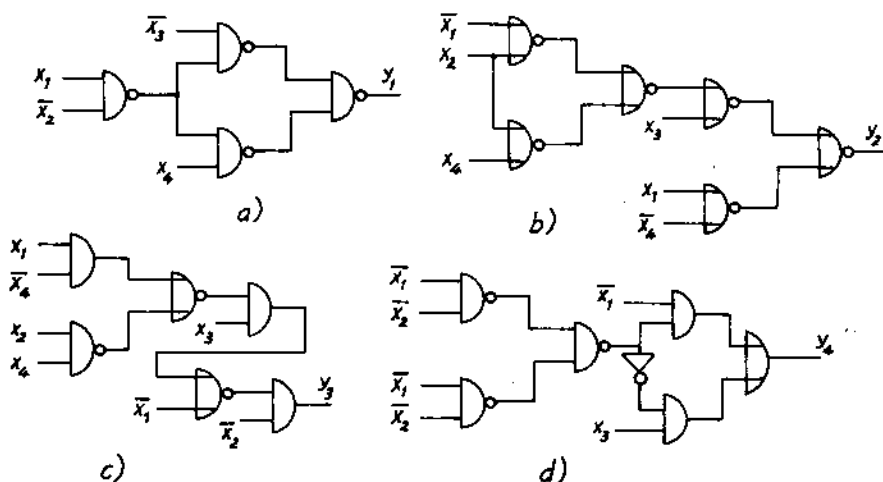
Tóm lại, các bước tìm tập hợp các tập giá trị biến chẩn đoán đầy đủ tối thiểu cho mạch như sau:

1. Xây dựng bảng giá trị phản ứng ra với tất cả hàm hư hỏng và hàm không hỏng.
2. Lập bảng phản ứng ra đối với các hư hỏng phân biệt được và hàm không hỏng bằng cách gộp các cột giống nhau thành một cột.
3. Từ bảng phản ứng ra của các hư hỏng phân biệt được, gộp các hàng giống nhau và thay chúng bằng một hàng đại diện.
4. Xây dựng bảng chẩn đoán đầy đủ theo algôrit đã nêu ở trên.
5. Tìm phủ tối thiểu của bảng chẩn đoán đầy đủ.
6. Xây dựng từ điển các mã hư hỏng của mạch.

Với algôrit này, sau khi đã xây dựng được từ điển chẩn đoán cho một mạch nào đó, việc tìm kiếm các hư hỏng đơn giản dẫn tới việc cho tác động vào mạch các tập giá trị biến chẩn đoán tối thiểu đã tìm được và ghi lại kết quả phản ứng ra. Sau đó so sánh kết quả này với từ điển các hư hỏng để biết là có hư hỏng hay không. Nếu có thì là hư hỏng loại gì và ở chỗ nào một cách nhanh chóng và chính xác.

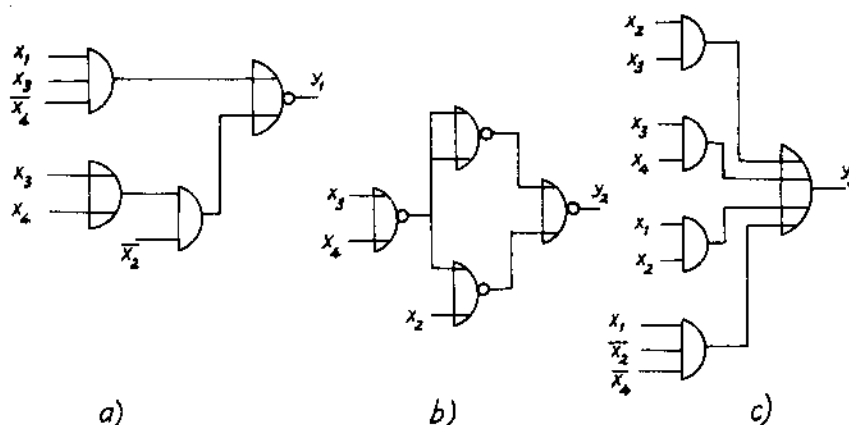
BÀI TẬP CHƯƠNG 5

1. Phân tích tĩnh



Hình 5.22

- 1.1. Hãy phân tích tĩnh để xác định chức năng của mạch cho trên hình 5.22 dưới dạng:
- Nhận biểu thức đại số phản ánh chức năng và cấu trúc của mạch.
 - Bảng giá trị thực của các hàm y_1, y_2, y_3, y_4 .
 - Nhận dạng tuyển chuẩn tối thiểu của các hàm trên.
- 1.2. Hãy kiểm tra xem ba mạch ở hình 5.23a, b, c có thực hiện cùng một chức năng hay không?



Hình 5.23

2. Phân tích động

2.1. Hãy phân tích động mạch ở hình 5.22a, b, c và d khi đầu vào thay đổi từ tập giá trị biến này sang tập giá trị biến khác như sau. Trong đó đối với hình 5.22d chỉ áp dụng đối với các tập giá trị biến không lớn hơn 7 trong hệ cơ số 10:

- | | | |
|-----------------|-----------------|-----------------|
| a) Từ 0 sang 5 | b) Từ 4 sang 7 | c) Từ 8 sang 9 |
| d) Từ 0 sang 11 | e) Từ 5 sang 6 | f) Từ 0 sang 1 |
| g) Từ 10 sang 9 | h) Từ 13 sang 8 | i) Từ 15 sang 9 |

2.2. Hãy phân tích động các mạch ở hình 5.22a, b, c và d khi có chuỗi tác động vào liên tục như sau:

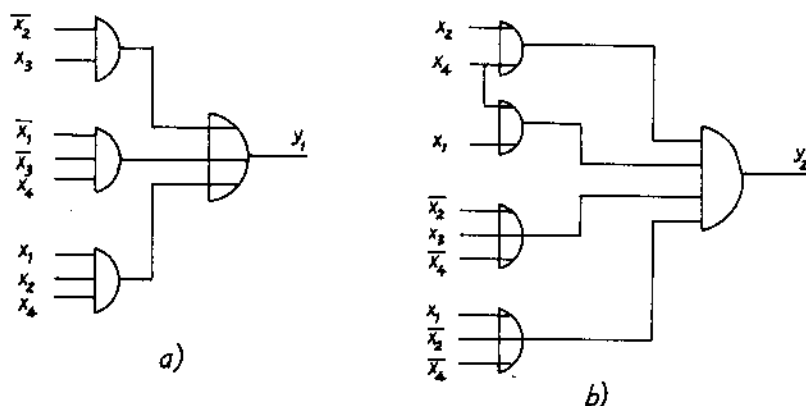
- $0(2\tau), 4(1\tau), 3(3\tau)$
- $0(2\tau), 1(2\tau), 5(3\tau), 12(4\tau)$ - đối với mạch ở hình 5.22a, b, c.
- $0(1\tau), 3(2\tau), 6(1\tau), 11(4\tau)$ - đối với các mạch ở hình 5.22a, b, c.

3. Phát hiện hiện tượng tranh chấp trạng thái

3.1. Phân tích các mạch cho trên hình 5.23a, b, c. Hãy chỉ rõ, các mạch này có điều kiện để làm xuất hiện:

- a) Tranh chấp trạng thái loại 1, một dấu không?
- b) Tranh chấp loại 0, một dấu không?
- c) Có tranh chấp động không?

3.2. Phát hiện các cặp tập giá trị biến có tranh chấp loại 0 và loại 1 một dấu đối với các mạch ở hình 5.24a và b.



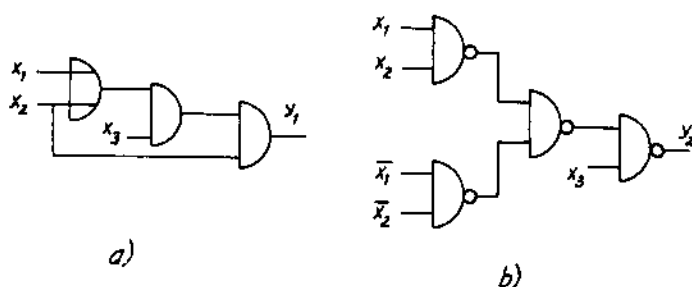
Hình 5.24

3.3. Tổng hợp các mạch thực hiện các hàm bốn biến dưới đây không có tranh chấp 1, một dấu.

- a) $y_1 = f_1(2, 3, 4, 9, 10, 12, 14, 15)$
- b) $y_2 = f_2(0, 1, 4, 6, 7, 9, 12, 14)$
- c) $y_3 = f_3(0, 3, 5, 7, 8, 10, 12, 15); f_3^-(11, 13, 14)$

4. Phát hiện các hư hỏng trong mạch

- 4.1. Xây dựng bảng kiểm tra hư hỏng đơn cho các mạch ở hình 5.25a, b.
- 4.2. Tìm chuỗi kiểm tra tối thiểu cho các mạch ở hình 5.25a, b.

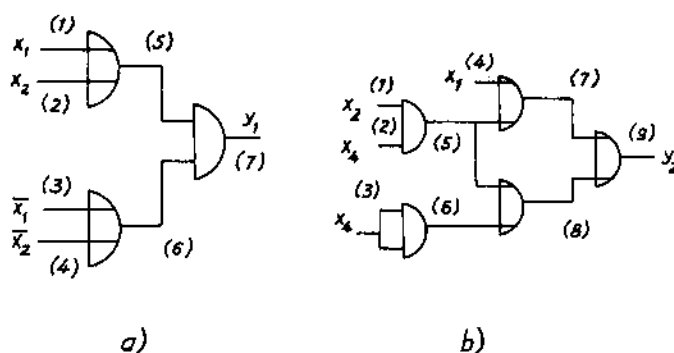


Hình 5.25

4.3. Xây dựng bảng chân đoán cho các hư hỏng sau đây đối với mạch ở hình 5.26a:

a) $f(1/6)$ và $f(0/5)$; b) $f(1/3)$ và $f(0/4)$

4.4. Tìm bảng chân đoán đầy đủ cho các mạch ở hình 5.26a và b.



Hình 5.26

Chương 6

TỔNG HỢP THIẾT BỊ SỐ TỔ HỢP THƯỜNG DÙNG

6.1. KHÁI QUÁT CHUNG

Thiết bị số tổ hợp là loại thiết bị số không có nhớ, tín hiệu ở đầu ra tại thời điểm t chỉ phụ thuộc vào tín hiệu vào tại thời điểm đó, không phụ thuộc vào tín hiệu vào tại các thời điểm trước đó. Nếu ký hiệu các tín hiệu ra là $y_j(t)$, $j = \overline{1, m}$, các tín hiệu vào của thiết bị là x_i , $i = \overline{1, n}$ thì quan hệ giữa đầu ra và đầu vào được biểu diễn bằng hệ các phương trình sau:

$$y_j(t) = f_j(x_i(t))$$

Cần chú ý rằng, nói mạch tổ hợp không có nhớ là ý nói nó không nhớ, không bị ảnh hưởng bởi các tập giá trị biến trước đó đã tác động vào thiết bị. Đó là điều khác với thiết bị số có nhớ. Song không phải vì thế mà thiết bị số tổ hợp không nhớ một cái gì đó trong nó. Điều này có liên quan tới cách thực hiện mạch số tổ hợp.

Mạch số tổ hợp có thể được thực hiện theo nhiều cách khác nhau. Như trong chương 4 đã nói rõ, nó có thể và thông thường được thực hiện bằng các mạch logic tổ hợp đơn lẻ như dùng họ các mạch VA, HOAC, KH; mạch VA-KH hay mạch HOAC-KH hoặc hỗn hợp giữa chúng. Nhưng ngày nay, khi mà công nghệ điện tử đã rất phát triển, nên tảng để thực hiện các hàm logic tổ hợp không phải chỉ có các phần tử logic đơn lẻ. Chúng có thể được thực hiện trên cơ sở các mảng logic lập trình được như PAL, PLA, FPLA; các bộ nhớ dạng nhớ cố định như ROM, PROM, EPROM hay các bộ nhớ thuật toán SRAM và DRAM. Chúng cũng có thể được thực hiện bằng phần mềm trên cơ sở sử dụng kỹ thuật vi xử lý. Trong những trường hợp hàm logic được thực hiện nhờ các phần tử có nhớ thì giá trị của hàm ra đối với các tập giá trị biến của nó được ghi lại thành nội dung tương ứng của các ô nhớ. Các tập giá trị biến tương ứng với các nội dung đó đóng vai trò địa chỉ của các ô nhớ này.

Như vậy, một hàm logic có thể được thực hiện bằng các phần tử logic (không có nhớ) hay bằng các phần tử có nhớ. Trong thực tế, nếu trong bảng giá trị thực, hàm không có nhiều giá trị 1 hay thiết bị không có nhiều biến vào thì nên thực hiện thiết bị

bằng các phần tử logic không có nhớ, vì trong trường hợp này không cần nhiều đường nối ghép. Để khắc phục về số lượng các đường nối, người ta thường tổ chức thành một mảng logic có khả năng chương trình hoá như PLA, PAL hay FPLA. Nếu số lượng các giá trị 1 của hàm nhiều hay thiết bị có nhiều biến vào thì nên thực hiện thiết bị trên cơ sở các phần tử nhớ hoặc các mảng logic lập trình được như PAL, PLA hay FPLA.

Dưới đây chúng ta sẽ xem cách xây dựng một số thiết bị số tổ hợp dùng phần tử logic thông thường không có nhớ hay được dùng trong thực tế.

6.2. TỔNG HỢP CÁC BỘ BIẾN ĐỔI MÃ

Bộ biến đổi mã là thiết bị chuyển đổi từ một mã này sang một mã khác. Trong thực tế kỹ thuật và đời sống, chúng ta luôn luôn phải dùng tới các bộ biến đổi mã. Thí dụ, để nhận biết độ lớn của một con số chúng ta luôn quen với mã thập phân. Ngược lại, máy tính lại chỉ làm việc với mã nhị phân. Do đó trong quá trình làm việc với máy tính, ta luôn phải thực hiện việc chuyển đổi từ mã thập phân sang mã nhị phân và ngược lại.

Các bộ biến đổi mã là thí dụ điển hình về các thiết bị số tổ hợp nhiều đầu ra. Chức năng của bất kỳ bộ biến đổi mã nào cũng có thể mô tả bằng một hệ các hàm logic. Sau đây chúng ta sẽ xem cách xây dựng một số thiết bị chuyển đổi mã thông dụng nhất.

6.2.1. Bộ biến đổi mã nhị phân sang mã thập phân (1 trong N) và ngược lại

1) Biến đổi mã nhị phân sang mã 1 trong N

Đó là thiết bị biến đổi có n đầu vào và $N = 2^n$ đầu ra. Mỗi đầu ra ứng với một tập giá trị biến. Khi một tập giá trị biến xuất hiện tại đầu vào của thiết bị thì tại đầu ra tương ứng sẽ có tín hiệu tích cực.

Để xây dựng các thiết bị biến đổi mã kiểu này, chúng ta xây dựng bảng giá trị thực của các hàm đầu ra, phụ thuộc vào các biến vào. Sau đó rút ra các hàm, đơn giản chúng và biến đổi chúng về dạng cần thiết nếu cần.

Để minh họa chúng ta xét thí dụ sau.

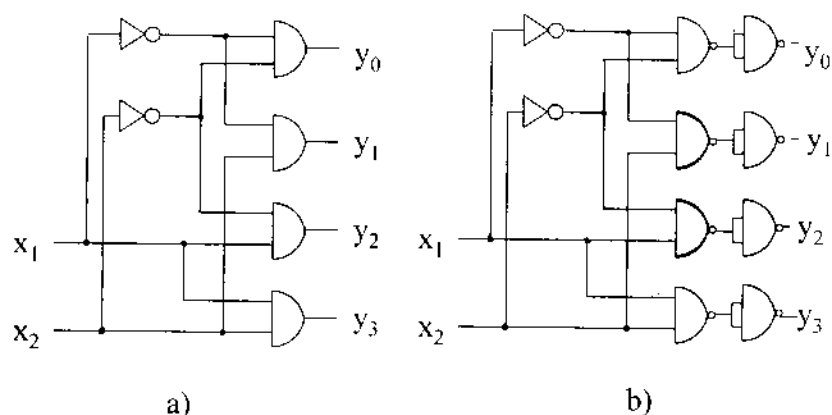
Thí dụ 6.1: Xây dựng bộ biến đổi mã nhị phân hai bit sang mã thập phân tương ứng.

Như vậy ta có hai biến vào, ký hiệu là x_1 và x_2 . Ta có 4 tập giá trị biến vì $N = 2^n = 4$. Do đó có bốn hàm ra, mỗi hàm tương ứng với một tập giá trị biến. Bảng giá trị thực của hệ bốn hàm hai biến và các hàm rút ra từ bảng này như sau:

x_1	x_2	y_0	y_1	y_2	y_3	
0	0	1	0	0	0	$y_0 = \bar{x}_1 \bar{x}_2$
0	1	0	1	0	0	$y_1 = \bar{x}_1 x_2$
1	0	0	0	1	0	$y_2 = x_1 \bar{x}_2$
1	1	0	0	0	1	$y_3 = x_1 x_2$

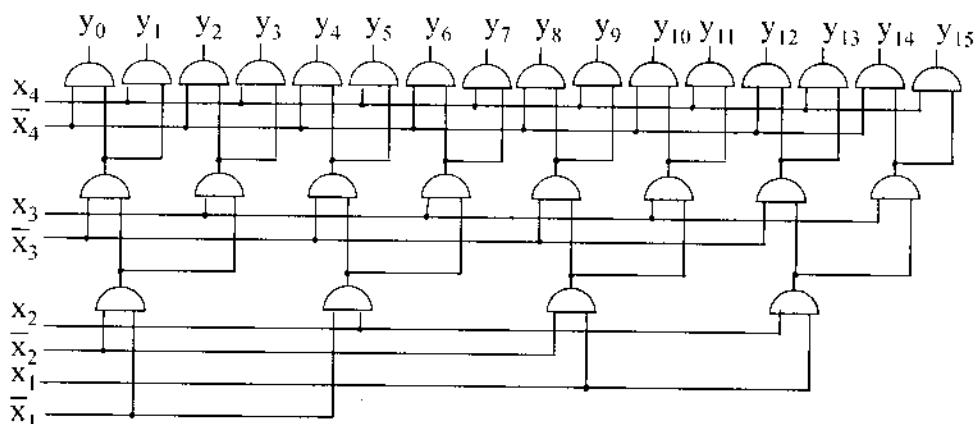
Mạch thực hiện thiết bị chuyển đổi mã này dùng các phần tử logic loại VA, HOAC, KH cho trên hình 6.1a, dùng phần tử logic hệ VA-KH cho trên hình 6.1b.

Cách xây dựng bộ biến đổi mã kiểu ở hình 6.1 gọi là kiểu một cấp hay kiểu ma trận. Khi số biến vào của thiết bị số lớn, kiểu ma trận cần nhiều phần tử logic, không tiết kiệm. Vì vậy kiểu ma trận chỉ dùng trong trường hợp khi số biến $n \leq 3$. Với $n > 3$, bộ chuyển đổi mã thường được xây dựng theo kiểu hình tháp hoặc theo kiểu nhiều cấp.



Hình 6.1. Mạch chuyển đổi mã nhị phân sang mã 1 trong N

+ Kiểu hình tháp được xây dựng dựa trên công thức dạng dấu ngoặc. Khi thực hiện người ta thực hiện từ dấu ngoặc bên trong cùng tới dấu ngoặc ngoài cùng.



Hình 6.2. Mạch giải mã theo hình tháp

Thí dụ với $n = 4$, bộ biến đổi mã từ mã nhị phân sang mã 1 trong $N = 2^4 = 16$, thực hiện theo kiểu nhiều cấp như sau:

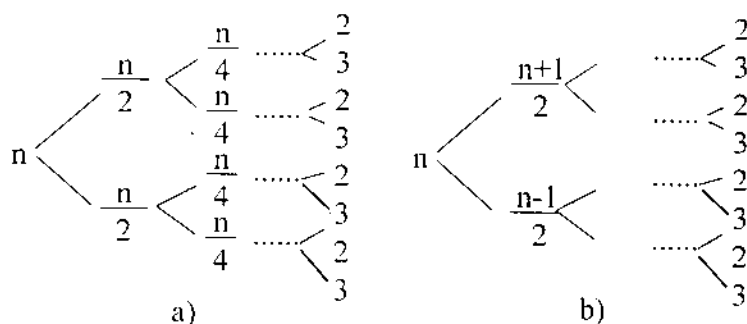
$$Y_j = ((\tilde{x}_1 \tilde{x}_2) \tilde{x}_3) \tilde{x}_4$$

trong đó $\tilde{x}_i$ có thể là x_i hoặc $\bar{x}_i$, tùy thuộc từng hàm cụ thể.

Trước tiên ta thực hiện bộ biến đổi đối với hai biến x_1 và x_2 . Ta được bộ biến đổi có 4 đầu ra. Từ 4 đầu ra này kết hợp với biến x_3 ta được 8 đầu ra. Từ 8 đầu ra này ta thực hiện các phép nhân với x_4 và $\bar{x}_4$ thì được 16 đầu ra của bộ biến đổi mã (hình 6.2).

Số mạch VA hai đầu vào cần thiết để thực hiện bộ biến đổi mã này là $M = 4(2^{n-1} - 1) = 28$. Nếu dùng kiểu một cấp phải mất 16 mạch VA bốn đầu vào hay $16 \times 3 = 48$ mạch VA hai đầu vào.

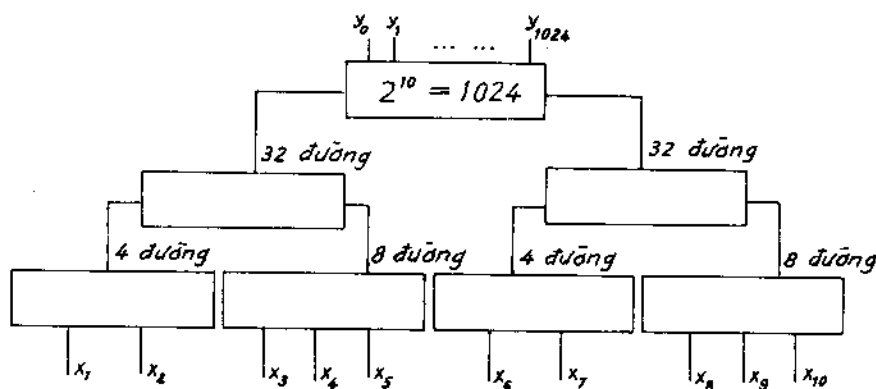
+ Kiểu nhiều cấp được xây dựng dựa trên nguyên tắc phân nhóm số biến n theo sơ đồ ở hình 6.3



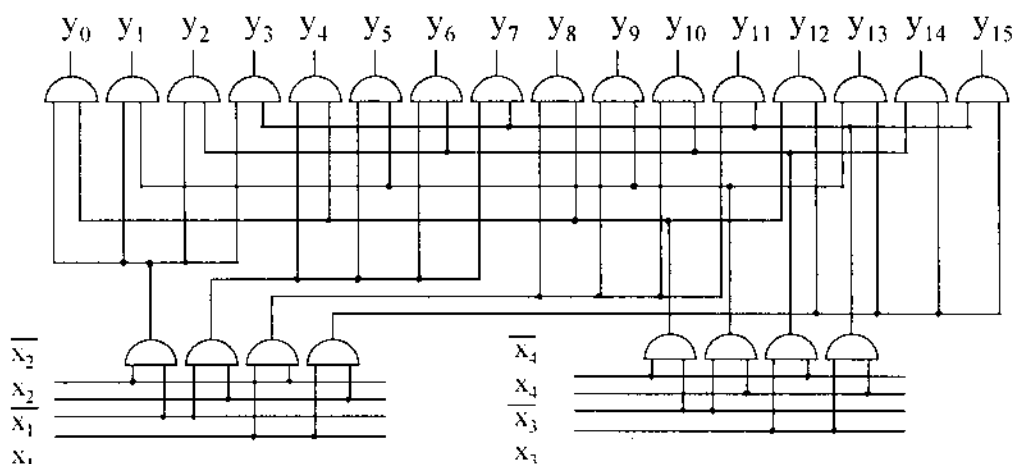
Hình 6.3. Phương thức phân cấp các biến

Việc phân nhóm tiến hành cho tới khi nào mỗi nhóm chỉ còn hai hay ba biến thì dừng lại.

Khi thực hiện, tiến hành từ mạch cấp hai hay ba biến trước theo kiểu một cấp. Kết quả sẽ được kết hợp với nhau để tạo thành các cấp cao hơn cho tới hết mới dừng. Sơ đồ tổ chức khái quát cho trên hình 6.4 với $n = 10$.



Hình 6.4. Sơ đồ tổ chức khái quát mạch giải mã nhiều cấp



Hình 6.5. Mạch giải mã nhiều cấp cho 4 biến

Mạch thực hiện cho trên hình 6.5.

Nếu thực hiện theo kiểu nhiều cấp số phần tử logic VA cần thiết khi số biến n là số chẵn và lẻ được tính như sau:

Với n là chẵn: Số lượng phần tử VA là M :

$$M = \frac{n}{2} \cdot 2^{n/2} + 2^n$$

Với n là lẻ: Số lượng phần tử VA là:

$$M = 2^{\frac{n+1}{2}} + 2^{\frac{n-1}{2}} + 2^n$$

Với $n = 4$, ta có cách phân nhóm như sau:

$$y = ((x_1 \ x_2)(x_3 \ x_4))$$

2) Bộ chuyển đổi mã 1 trong N sang mã nhị phân

Đó là bộ biến đổi ngược của bộ biến đổi từ mã nhị phân sang mã 1 trong N trên đây. Với bảng giá trị thực trên đây, nếu ta coi các tín hiệu đầu vào là đầu ra và đầu ra lại là đầu vào thì ta có bảng giá trị thực của bộ biến đổi mã 1 trong N sang mã nhị phân. Từ đó dễ dàng rút ra các hàm ra phụ thuộc vào các biến vào để thực hiện mạch mong muốn. Tuy vậy trong thực tế bộ biến đổi từ mã 1 trong N sang mã nhị phân thường được thực hiện theo nguyên tắc giải mã ưu tiên, nghĩa là tín hiệu trên đầu ra sẽ là một số nhị phân tương ứng với tín hiệu đầu vào trọng số cao nhất có giá trị 1, trong khi giá trị của các đầu vào có trọng số thấp hơn bằng 0 hay 1 đều không ảnh hưởng tới kết quả. Bảng chuyển đổi của những loại biến đổi mã loại này với $n = 7$ cho trên hình 6.6.

x_7	x_6	x_5	x_4	x_3	x_2	x_1	y_2	y_1	y_0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	1
0	0	0	0	0	1	x	0	1	0
0	0	0	0	1	x	x	0	1	1
0	0	0	1	x	x	x	1	0	0
0	0	1	x	x	x	x	1	0	1
0	1	x	x	x	x	x	1	1	0
1	x	x	x	x	x	x	1	1	1

Hình 6.6. Bảng giá trị thực của bộ giải mã 1 trong N sang nhị phân theo nguyên tắc ưu tiên

Một số mạch chuyển đổi mã loại này thường dùng là: SN 74147 (TTL) là loại biến đổi mã từ 1 trong 10 sang mã nhị phân; SN 74148 hay MC 10165 là các bộ biến đổi từ mã 1 trong 8 thành mã nhị phân.

3) Bộ chuyển đổi từ mã nhị phân sang mã Grey (mã kế cận)

Để thiết kế bộ chuyển đổi mã này, ta có thể xây dựng bảng chuyển đổi mã hay là bảng giá trị thực (hình 6.7) rồi từ đó rút ra các hàm logic đầu ra tương ứng, phụ thuộc vào các biến đầu vào như đã làm trong thí dụ trên đây hay đối với bất kỳ bộ biến đổi mã nào khác. Riêng trong trường hợp này, quan hệ giữa đầu ra và đầu vào của bộ biến đổi mã từ mã nhị phân sang mã Grey hay ngược lại là quan hệ tương đối đặc biệt mà ta có thể nhận xét và rút ra các hàm ra thông qua các thuật toán cộng môđun 2 như sau:

Mã nhị phân	Mã Grey	Mã nhị phân	Mã Grey
0000	0000	1000	1100
0001	0001	1001	1101
0010	0011	1010	1111
0011	0010	1011	1110
0100	0110	1100	1010
0101	0111	1101	1011
0110	0101	1110	1001
0111	0100	1111	1000

Hình 6.7. Một số loại mã thông dụng

$$y_0 = x_0 \oplus x_1$$

$$y_1 = x_1 \oplus x_2$$

$$y_2 = x_2 \oplus x_3$$

$$y_3 = x_3 \oplus x_4$$

.....

$$y_i = x_i \oplus x_{i+1}$$

Nhưng đối với bit cao nhất thì:

$$y_N = x_N \oplus 0$$

Ngược lại, khi cần biến đổi từ mã Grey sang mã nhị phân thì ta coi các $x_i, i = \overline{1, 4}$ của bảng ở hình 6.7 là các đầu ra, các biến $y_j, j = \overline{1, 4}$ là các đầu vào. Khi đó thì:

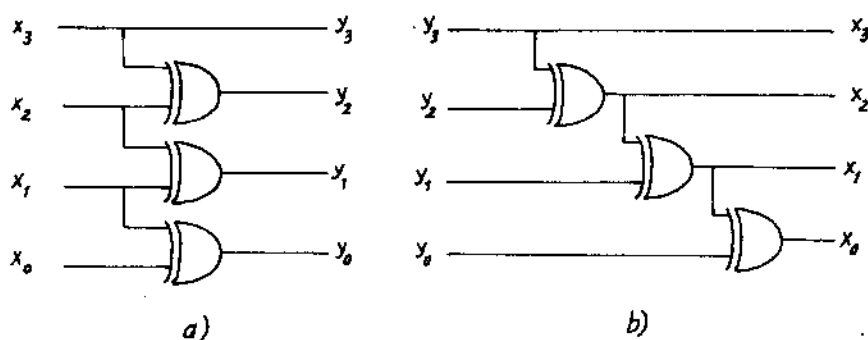
$$x_N = y_N$$

$$x_{N-1} = x_N \oplus y_{N-1}$$

.....

$$x_0 = x_1 \oplus y_0$$

Các mạch tương ứng cho trên hình 6.8a và hình 6.8b.



Hình 6.8. Mạch chuyển đổi mã nhị phân sang mã Grey

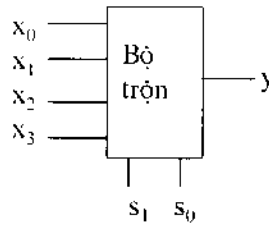
6.2.2. Bộ trộn và bộ phân đường (Multiplexer và Demultiplexer)

Trong thu thập và phân phối thông tin ta thường phải lấy thông tin từ nhiều nguồn để sau khi xử lý xong lại phân phối từ trung tâm thông tin tới các đơn vị cơ sở. Để điều khiển phối hợp hoạt động giữa trung tâm thông tin và các đơn vị cơ sở, trong những trường hợp trên đây cần có các thiết bị nối ghép giữa thiết bị trung tâm và các thiết bị đầu cuối. Đó chính là các bộ trộn và bộ phân đường.

1) Bộ trộn

Còn gọi là bộ thu thập thông tin, là một thiết bị có một đầu ra y nhưng có nhiều đầu vào $x_i, i = \overline{1, n}$. Tín hiệu từ đầu vào nào được đưa ra đầu ra là tùy thuộc vào từng

tập giá trị của các biến điều khiển $s_j, j = \overline{1, m}$. Hình 6.9 là bộ trộn 4 đầu vào x_3, x_2, x_1, x_0 và một đầu ra y . Đầu ra y được nối với đầu vào nào trong bốn đầu vào trên đây là tùy thuộc vào giá trị của hai biến điều khiển s_1 và s_0 . Số biến điều khiển tối thiểu cần thiết để điều khiển quan hệ của đầu ra y và n đầu vào là $\lceil \log_2 n \rceil$. Dấu $\lceil \dots \rceil$ chỉ số nguyên nhỏ nhất lớn hơn hay bằng $\log_2 n$.



Hình 6.9. Bộ trộn 4 đầu vào

Bảng giá trị thực mô tả hàm y phụ thuộc vào các biến vào x_i có dạng dưới đây.

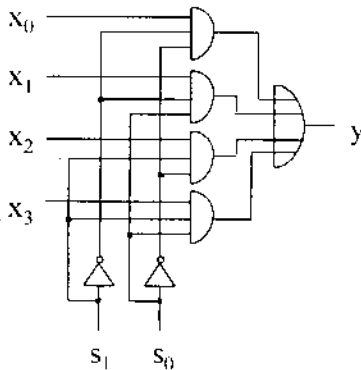
Từ bảng giá trị thực ta dễ dàng rút ra hàm sau đây:

$$y = \bar{s}_1 \bar{s}_0 x_0 + \bar{s}_1 s_0 x_1 + s_1 \bar{s}_0 x_2 + s_1 s_0 x_3$$

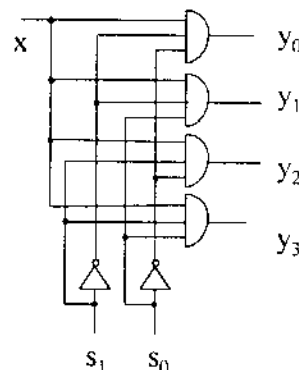
Mạch thực hiện cho trên hình 6.10a.

$s_1 s_0$	y
0 0	x_0
0 1	x_1
1 0	x_2
1 1	x_3

Theo nguyên tắc chung trên đây, các hãng sản xuất vi mạch đã cho ra các loại mạch trộn với số đầu vào khác nhau. Thí dụ SN 74150 (TTL) có 16 đầu vào, một đầu ra; SN 74151, MC 10164, MC 14512 là các vi mạch trộn có 8 đầu vào. Các mạch này được sử dụng rộng rãi trong các thiết bị số như máy tính, các thiết bị thu thập và xử lý tin...



a)



b)

Hình 6.10. Mạch trộn (a) và mạch phân đường (b)

2) Bộ phân đường

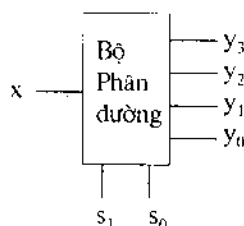
Bộ phân đường là thiết bị có một đầu vào x nhưng lại có nhiều đầu ra y_i , $i = \overline{1, n}$. Tín hiệu từ đầu vào được đưa ra đầu ra nào là tùy thuộc vào các tín hiệu điều khiển s_j , $j = \overline{1, m}$.

Hình 6.11 là bộ phân đường với một đầu vào x , bốn đầu ra y_3, y_2, y_1, y_0 và hai đầu điều khiển s_1, s_0 .

Số lượng đầu điều khiển cần thiết để mạch có n đầu ra là $\lceil \log_2 n \rceil$, trong đó dấu $\lceil \dots \rceil$ là số nguyên nhỏ nhất lớn hơn hay bằng $\log_2 n$.

Bảng giá trị thực của mạch phân đường ở hình 6.11 có thể như sau:

s_1	s_0	y_3	y_2	y_1	y_0
0	0	0	0	0	x
0	1	0	0	x	0
1	0	0	x	0	0
1	1	x	0	0	0



Hình 6.11. Mạch phân đường

Trên cơ sở bảng giá trị thực này ta dễ dàng rút ra hệ hàm đầu ra như sau:

$$y_0 = \bar{s}_1 \bar{s}_0 x$$

$$y_1 = \bar{s}_1 s_0 x$$

$$y_2 = s_1 \bar{s}_0 x$$

$$y_3 = s_1 s_0 x$$

Mạch thực hiện cho trên hình 6.10b.

Các loại vi mạch phân đường thường dùng như SN 74151, MC 74514 là loại 16 đầu ra; loại SN 7442 có 10 và SN 74138 và MC 101162 có 8 đầu ra.

6.2.3. Bộ so sánh

Các bộ so sánh là các mạch thực hiện so sánh hai từ nhị phân A và B để phát hiện xem $A = B$, $A < B$ hay $A > B$.

Hai số nhị phân A và B gọi là bằng nhau nếu chúng bằng nhau ở từng bit tương ứng. Trong thực tế thường tồn tại hai loại so sánh:

+ So sánh phát hiện $A = B$ hay $A \neq B$;

+ So sánh phát hiện $A = B$, $A < B$ hay $A > B$.

1) Bộ so sánh $A = B$ hay $A \neq B$

Trong trường hợp này thiết bị cần so sánh từng bit tương ứng của A và B. Nếu tất cả các bit tương ứng của chúng đều bằng nhau thì $A = B$. Ngược lại, chỉ cần phát hiện ít nhất một cặp bit không bằng nhau thì $A \neq B$.

Để phát hiện hai bit có bằng nhau hay không, ta dùng thuật toán cộng môđun 2. Cộng môđun 2 của hai bit bằng 0 nếu chúng bằng nhau. Nếu chúng khác nhau thì kết quả phép cộng là 1. Để biết cả từ A và B có giống nhau hay không, ta dùng phép HOAC-KH (NOR) kết quả của so sánh các bit của chúng. Nếu hai từ A và B bằng nhau thì kết quả này phải bằng 1. Ngược lại, kết quả này bằng 0. Tức là đối với hai từ :

$$A = (a_n a_{n-1} \dots a_1) \text{ và } B = (b_n b_{n-1} \dots b_1) \text{ thì:}$$

$$y = (a_n \oplus b_n) \downarrow (a_{n-1} \oplus b_{n-1}) \downarrow \dots \downarrow (a_1 \oplus b_1)$$

Như vậy, $y = 1$ nếu $A = B$ và $y = 0$ nếu $A \neq B$. Sơ đồ thực hiện mạch này cho trên hình 6.12a.

Hãng National sản xuất vi mạch so sánh kiểu này như DM 8131 với 2 x 6 đầu vào, hoặc AM 25LS21 với 2 x 8 đầu vào của hãng AMD.

2) Bộ so sánh $A = B, A < B, A > B$

Trong nhiều trường hợp, việc so sánh không phải chỉ để biết xem hai từ có giống nhau không mà còn phải phân biệt từ nào lớn hơn trong trường hợp chúng không bằng nhau (Bộ Magnitude Comparator).

Trong trường hợp này ta dùng ba biến $y_{a=b}$, $y_{a < b}$ và $y_{a > b}$ để chỉ ba trường hợp: $A = B$, $A < B$ và $A > B$ một cách tương ứng. Khi đó bảng giá trị thực so sánh hai số nhị phân, mỗi số một bit, được xây dựng như sau:

a	b	$y_{a=b}$	$y_{a < b}$	$y_{a > b}$
0	0	1	0	0
0	1	0	1	0
1	0	0	0	1
1	1	1	0	0

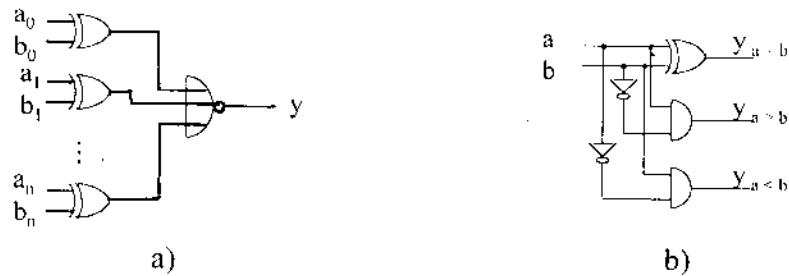
Từ đây ta rút ra:

$$y_{a=b} = \overline{a \oplus b}$$

$$y_{a < b} = \overline{a} \cdot b$$

$$y_{a > b} = a \cdot \overline{b}$$

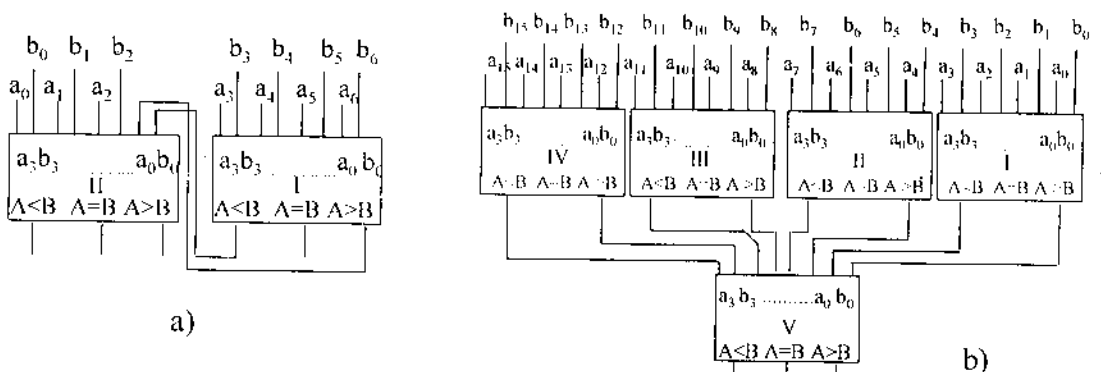
Mạch thực hiện cho trên hình 6.12b.



Hình 6.12. Hai loại mạch so sánh

Khi A và B là những từ nhiều bit, ta thực hiện so sánh theo cách sau: Đầu tiên so sánh hai bit có trọng số cao nhất của hai từ. Nếu chúng khác nhau thì kết quả đó chính là kết quả của so sánh hai từ. Nếu chúng bằng nhau thì tiếp tục so sánh hai bit tiếp theo. Cứ như vậy cho đến bit thấp nhất thì kết thúc. Mạch so sánh của mỗi bit có thể dùng mạch ở hình 6.12b.

Trên nguyên tắc đó, các hãng đã sản xuất các loại IC so sánh bốn bit như SN 485 (TTL), MC 10166 (ECL), MC 14585 (KMOS). Những IC này hoàn toàn có thể nối ghép để mở rộng phạm vi so sánh cho các số nhiều bit hơn. Cách nối ghép có thể là theo kiểu nối tiếp (hình 6.13a) hoặc theo kiểu song song (hình 6.13b). Khi số bit của A và B lớn thì nên mắc theo kiểu song song. Trong kiểu ghép nối tiếp khi các bit cao của A và B đều bằng nhau thì kết quả so sánh sẽ do các bit thấp ở bộ so sánh II quyết định và kết quả so sánh A và B được đọc ở bộ so sánh II.



Hình 6.13. Ghép nối các IC để so sánh hai từ nhiều bit

a) Ghép nối tiếp; b) Ghép song song

6.2.4. Các bộ cộng

Các bộ cộng là mạch thực hiện phép cộng hai số nhị phân A và B được biểu diễn dưới dạng một loại mã nào đó như mã nhị phân, mã BCD, mã Grey... Với mỗi loại mã ta có mạch cộng khác nhau. Dưới đây ta xây dựng bộ cộng hai số nhị phân.

1) Trường hợp đơn giản nhất: Cộng hai số, mỗi số có một bit

Bảng giá trị thực của phép cộng trong trường hợp cộng hai bit nhị phân a_0 và b_0 như sau:

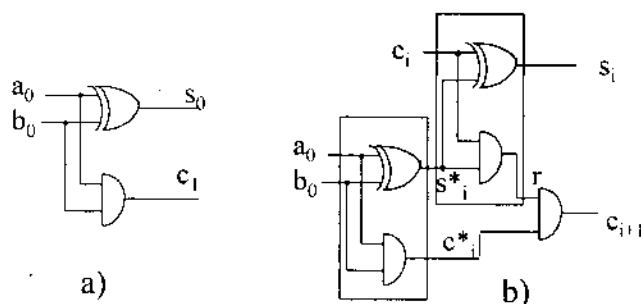
a_0	b_0	s_0	c_1	Trong đó s_0 là giá trị tổng, còn c_1 là số nhớ của tổng đó sang bit tiếp theo. Từ đó ta có:
0	0	0	0	
0	1	1	0	$s_0 = \bar{a}_0 b_0 + a_0 \bar{b}_0 = a_0 \oplus b_0$
1	0	1	0	$c_1 = a_0 b_0$
1	1	0	1	

Mạch thực hiện bộ cộng hai bit a_0 và b_0 cho trên hình 6.14a.

Mạch 6.14a thường gọi là mạch bán tổng, vì nó chưa tính đến số nhớ của bit phía trước nó. Khi trong phép cộng có kể tới số nhớ của bit trước nó ta sẽ có bộ tổng đầy đủ.

Bảng giá trị thực của một bộ tổng đầy đủ, thực hiện cộng hai số a_i và b_i cùng số nhớ của phép cộng bit trước nó c_i cho dưới đây. Trong đó s_i và c_{i+1} là giá trị tổng và số nhớ của phép cộng. Chúng ta có thể thực hiện bộ tổng bằng cách rút ra các hàm s_i và c_{i+1} và thực hiện chúng. Song cũng có thể biểu diễn qua các đại lượng trung gian để có thể thực hiện bộ cộng đầy đủ bằng hai mạch bán tổng - hình 6.14b. Cách biểu diễn qua các đại lượng trung gian như sau:

Vào			Giá trị trung gian			Ra	
a_i	b_i	c_i	s_i^*	c_i^*	r	s_i	c_{i+1}
0	0	0	0	0	0	0	0
0	1	0	1	0	0	1	0
1	0	0	1	0	0	1	0
1	1	0	0	1	0	0	1
0	0	1	0	0	0	1	0
0	1	1	1	0	1	0	1
1	0	1	1	0	1	0	1
1	1	1	0	1	0	1	1



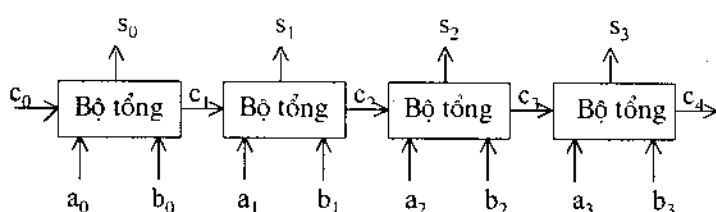
Hình 6.14. Mạch bán tổng (a) và mạch tổng đầy đủ (b)

Trong đó a_i và b_i được cộng với nhau và cho kết quả tổng s^*_i và số nhớ c^*_i . Sau đó s^*_i lại được cộng với số nhớ c_i để cho ra tổng s_i và số nhớ r . Cuối cùng hai số nhớ được cộng với nhau để được số nhớ c_{i+1} .

2) Để thực hiện cộng nhiều bit

Ta cần ghép nối các bộ tổng đầy đủ lại. Hình 6.15 là cách nối ghép các môđun cộng đầy đủ để thực hiện cộng hai số A và B bốn bit.

Trên nguyên tắc ở hình 6.15, các hãng sản xuất IC đã cho ra đời các bộ tổng khác nhau, thí dụ IC 74LS83 là loại cộng 4 bit.



Hình 6.15. Nối ghép các bộ tổng để cộng hai từ nhiều bit

Trên cơ sở các bộ tổng đã trình bày trên đây, dễ dàng nối ghép để thực hiện các phép cộng hai số cho dưới dạng BCD, thực hiện lấy bù của một số cho trước ở dạng nhị phân, thực hiện các phép trừ, phép nhân, phép chia hay phép dịch... Các IC có khả năng thực hiện các chức năng như trên như 82S82, 82S93 (loại TTL) của hãng Signetics hoặc 7480 PC, 7482 PC, 7483 PC... Đó là các bộ tổng 4 bit cho dưới dạng số BCD.

BÀI TẬP CHƯƠNG 6

- Tổng hợp các bộ biến đổi từ mã nhị phân sang mã 1 trong N với ba biến nhị phân x_1, x_2, x_3 :
 - Theo kiểu thông thường.
 - Theo kiểu hình tháp (nhiều cấp).
- Tổng hợp các bộ biến đổi giữa các mã cho trong bảng sau đây:

8421	Dư 3	2421	Grey	1 trong N
0000	0011	0000	0000	1 000 000 000
0001	0100	0001	0001	0 100 000 000
0010	0101	0010	0011	0 010 000 000
0011	0110	0011	0010	0 001 000 000
0100	0111	0100	0110	0 000 100 000

0101	1000	1011	0111	0 000 010 000
0110	1001	1100	0101	0 000 001 000
0111	1010	1101	0100	0 000 000 100
1000	1011	1110	1100	0 000 000 010
1001	1100	1111	1001	0 000 000 001

3. Tổng hợp bộ trộn (multiplexer) cho 4, 6, 8, 10 và 16 đầu vào, một đầu ra:
 - a) Dùng hệ hàm (VA, HOAC, KH).
 - b) Dùng mạch (VA-KH).
 - c) Dùng mạch (HOAC-KH).
4. Tổng hợp các bộ phân đường một đầu vào và có 4, 6, 8, 10 và 16 đầu ra:
 - a) Dùng hệ hàm (VA, HOAC, KH).
 - b) Dùng mạch (VA-KH).
 - c) Dùng mạch (HOAC-KH).
5. Tổng hợp bộ cộng 1 bit có nhớ:
 - a) Dùng mạch (VA-KH).
 - b) Dùng mạch (HOAC-KH).
6. Tổ chức bộ cộng hai từ 8 bit, dùng kết quả bài 5.

PHẦN III

TỔNG HỢP VÀ PHÂN TÍCH MẠCH SỐ CÓ NHỚ

Chương 7

CÁC HÀM LÔGIC PHỤ THUỘC THỜI GIAN VÀ HÀM LÔGIC TRUY TOÁN

Các hàm logic phụ thuộc thời gian và hàm logic truy toán cũng là một trong các công cụ mô tả chức năng của các thiết bị số có nhớ song song với các mô tả chức năng thiết bị số dùng phương pháp bảng giá trị thực, phương pháp bảng chuyển đổi trạng thái, bảng ra và phương pháp đồ hình trạng thái... Trong chương này chúng ta sẽ đề cập một cách chi tiết hơn các hàm phụ thuộc thời gian, các hàm logic truy toán và quan hệ của chúng với dạng thông thường của các hàm logic.

7.1. HÀM LÔGIC PHỤ THUỘC THỜI GIAN

7.1.1. Những khái niệm cơ bản

Ta hãy xét một hàm phụ thuộc vào các biến logic $x_1, x_2, \dots, x_n$ và biến thời gian t . Trong đó như chúng ta đã biết, các biến $x_i, i = \overline{1, n}$ có thể nhận giá trị 0 hoặc 1; biến thời gian t có thể nhận s giá trị, từ 0 đến $(s - 1)$.

Như vậy, ứng với mỗi giá trị của mỗi biến x_i và t , ký hiệu là (x^*, t^*) , ta có một tập giá trị biến $b^* \in B^* = \{x^*, t^*\}$, trong đó B^* là tập hợp tất cả các tập giá trị biến có thể có của tập biến x_i và t . Khi đó ứng với mỗi tập giá trị biến $b^* \in B^*$ ta có giá trị tương ứng của hàm logic y . Giá trị này cũng chỉ có thể là 0 hay 1. Vì hàm y trong trường hợp này vừa phụ thuộc vào các biến logic, vừa phụ thuộc vào biến thời gian, nên gọi là hàm logic phụ thuộc thời gian.

$$y = f(x_1, x_2, \dots, x_n, t)$$

Số lượng tập hợp giá trị biến của hàm logic phụ thuộc thời gian n biến là $N = s.2^n$. Số lượng hàm logic phụ thuộc thời gian n biến là $M = 2^N$.

Thí dụ, với $n = 2, s = 3$, số tập giá trị biến $N = 3.2^2 = 3.4 = 12$ và số hàm khác nhau sẽ là $M = 2^{12} = 4096$ hàm.

Rõ ràng, nếu hàm logic không phụ thuộc vào biến thời gian t một cách chính thức thì ta có hàm logic thông thường.

Bảng 7.1 là bảng giá trị thực của một trong 4096 hàm nói trên.

Bảng 7.1. Bảng giá trị thực của hàm logic phụ thuộc thời gian

x_1	x_2	t	$y = f(x_1, x_2, t)$	x_1	x_2	t	$y = f(x_1, x_2, t)$
0	0	0	1	1	0	1	0
0	1	0	0	1	1	1	1
1	0	0	0	0	0	2	0
1	1	0	1	0	1	2	0
0	0	1	1	1	0	2	1
0	1	1	0	1	1	2	0

7.1.2. Tính chất của hàm logic phụ thuộc thời gian

1) Tính chất tương đương

Bất kỳ hàm logic phụ thuộc thời gian nào cũng tương đương với một hàm logic trong chuỗi các hàm logic thông thường vào một thời điểm nhất định.

Thật vậy, nếu có hàm $y = f(x_1, x_2, \dots, x_n, t)$ với $t = \overline{0, (s-1)}$. Nếu $t = k \in \overline{0, (s-1)}$, thì:

$$y_k = f_k(x_1, x_2, \dots, x_n), k = \overline{0, (s-1)} \quad (7.1)$$

Với hàm cho ở bảng 7.1, ta có :

$$y_0 = \bar{x}_1 \bar{x}_2 + x_1 x_2, \quad \text{với } t = 0$$

$$y_1 = \bar{x}_1 \bar{x}_2 + x_1 x_2, \quad \text{với } t = 1$$

$$y_2 = x_1 \bar{x}_2, \quad \text{với } t = 2$$

Như vậy, ta có thể liên kết tất cả các hàm y_k thành một hàm bằng cách đưa thêm hàm đặc biệt τ_α như sau:

$$\tau_\alpha = \begin{cases} 0 & \text{nếu } t \neq \alpha \\ 1, & \text{nếu } t = \alpha \end{cases}$$

Khi đó có thể viết dạng khái quát của hàm y như sau:

$$y = y_0 \cdot \tau_0 + y_1 \cdot \tau_1 + y_2 \cdot \tau_2 + \dots + y_{(s-1)} \cdot \tau_{(s-1)} \quad (7.2)$$

2) Dạng tuyến chuẩn chính tắc của hàm logic phụ thuộc thời gian

Mọi hàm logic phụ thuộc thời gian đều có thể biểu diễn dưới dạng tuyến chuẩn chính tắc (7.2) và có thể dùng phương pháp đơn giản hàm logic thông thường để đơn giản mọi hàm $y_i, i = \overline{0, (s-1)}$.

Thí dụ 7.1: Hàm logic phụ thuộc thời gian cho trong bảng 7.2 dưới đây.

Bảng 7.2. Hàm logic phụ thuộc thời gian của thí dụ 7.1

x_1	x_2	t	$y = f(x_1, x_2, t)$	x_1	x_2	t	$y = f(x_1, x_2, t)$
0	0	0	0	0	0	1	0
0	1	0	1	0	1	1	1
1	0	0	1	1	0	1	0
1	1	0	1	1	1	1	0

có dạng: $y = (\bar{x}_1 x_2 + x_1 \bar{x}_2 + x_1 x_2) \cdot \tau_0 + \bar{x}_1 x_2 \cdot \tau_1$ (7.2)

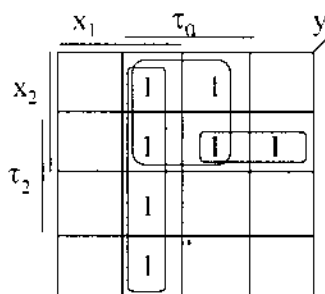
Đối với hàm logic phụ thuộc thời gian này, ta có thể thực hiện đơn giản cho từng hàm $y_0 = (\bar{x}_1 x_2 + x_1 \bar{x}_2 + x_1 x_2)$ và $y_1 = \bar{x}_1 x_2$. Tuy nhiên hoàn toàn có thể coi các τ_0 và τ_1 như các biến logic độc lập. Khi đó ta coi y như một hàm không phải hai biến mà là bốn biến. Từ đó áp dụng phương pháp đơn giản hàm bốn biến. Trường hợp hàm y trên đây có thể đơn giản bằng cách dùng bảng Karnaugh cho bốn biến như trên hình 7.1.

Đơn giản hàm ta có:

$$y = x_2 \tau_0 + x_1 \tau_0 + \bar{x}_1 x_2 \tau_1$$

$$y = (x_2 + x_1) \tau_0 + \bar{x}_1 x_2 \tau_1$$

Trong trường hợp số biến quá lớn thì việc giảm bớt số biến trong khi đơn giản hàm là cần thiết và do đó việc đơn giản hàm đối với từng thành phần là cần thiết.



Hình 7.1. Bảng Karnaugh đơn giản hàm (7.3)

Trong một số trường hợp khác lại có thể sử dụng các trường hợp bất định của hàm khi mọi $\tau_i, i = 0, (s-1)$ đều bằng 0 để đơn giản hàm. Tuy vậy, trong trường hợp của ta việc sử dụng các trường hợp bất định không mang lại kết quả tốt hơn.

Tương tự như các trường hợp hàm logic thông thường, ta có thể biểu diễn hàm trên đây dưới dạng hội chuẩn tối thiểu như sau:

$$y = (x_1 + x_2)(\bar{x}_1 + x_2)(\bar{x}_1 + \bar{x}_2) \tau_1 + (x_2 + x_1) \tau_0$$

3) Tính chất có chu kỳ của hàm logic phụ thuộc thời gian

a) Nếu hàm logic phụ thuộc thời gian thỏa mãn điều kiện: $y_{t+q} = y_t$ ta nói nó có chu kỳ, với chu kỳ là q .

b) Hàm logic có chu kỳ có thể cho dưới dạng bảng có q hàng khi giá trị của t không bị hạn chế trên. Đó là tính chất rất quan trọng đối với việc tổng hợp và phân tích các thiết bị số mà hoạt động của nó được mô tả bởi hàm logic phụ thuộc thời gian.

Thí dụ 7.2. Cho hàm logic phụ thuộc thời gian $y(x_1, x_2, t); 0 \leq t \leq \infty$ như sau:

x_1	x_2	t	y	x_1	x_2	t	y
0	0	0	0	0	0	2	0
0	1	0	0	0	1	2	0
1	0	0	1	1	0	2	1
1	1	0	1	1	1	2	1
0	0	1	1	0	0	3	1
0	1	1	0	0	1	3	0
1	0	1	1	1	0	3	1
1	1	1	0	1	1	3	0

Từ đây ta có:

$$y_{2k}(x_1, x_2) = x_1 \bar{x}_2 + x_1 x_2 = x_1; k = 0, 1, 2, 3, \dots$$

$$y_{2k+1}(x_1, x_2) = \bar{x}_1 \bar{x}_2 + x_1 \bar{x}_2 = \bar{x}_2; k = 0, 1, 2, \dots$$

c) Bất kỳ hàm logic phụ thuộc thời gian nào cũng có thể làm cho trở thành có chu kỳ với $q > s - 1$.

Thí dụ cho hàm sau:

$$y = \bar{x}_1 x_2 t_0 + t_1 + x_2 t_3 + (x_1 + \bar{x}_2) t_4$$

Ta có thể làm cho hàm này trở thành có chu kỳ $q = 5$ bằng cách xác định cách kéo dài hàm như sau:

$$y_{5k}(x_1, x_2) = \bar{x}_1 x_2; k = 0, 1, 2, \dots$$

$$y_{5k+1}(x_1, x_2) = 1; k = 0, 1, 2, \dots$$

$$y_{5k+2}(x_1, x_2) = 0; k = 0, 1, 2, \dots$$

$$y_{5k+3}(x_1, x_2) = x_2; k = 0, 1, 2, \dots$$

$$y_{5k+4}(x_1, x_2) = x_1 + \bar{x}_2; k = 0, 1, 2, \dots$$

7.2. CÁC HÀM LÔGIC TRUY TOÁN

7.2.1. Hàm logic truy toán loại 1 (HLTR-1)

Giả sử có hàm logic phụ thuộc thời gian φ_t , giá trị của hàm này tại thời điểm t là y_t , trong đó $y_t = 0$ hay 1 . Giả sử các biến $x_i, i = \overline{1, n}$ cũng phụ thuộc vào thời gian t và là loại biến logic, chỉ nhận giá trị 0 hay 1 . Khi đó:

+ Hàm ánh xạ đơn trị của tập hợp các vectơ:

$$Q = \{x_{1t}, x_{2t}, \dots, x_{nt}, y_{t-1}, y_{t-2}, \dots, y_{t-k}\} \rightarrow (0, 1) \quad (7.4)$$

lên tập hợp $y = (0, 1)$ gọi là hàm logic truy toán loại một (HLTR-1). Tức là:

$$y_t = \varphi(x_{1t}, x_{2t}, \dots, x_{nt}, y_{t-1}, y_{t-2}, \dots, y_{t-k}) \quad (7.5)$$

Trong đó k có thể có giá trị từ 1 tới t . Nếu $k = t$ thì y_0 là biến của hàm φ và trùng với giá trị ban đầu của hàm y_t .

Giả thiết giá trị ban đầu là cho trước, ta có:

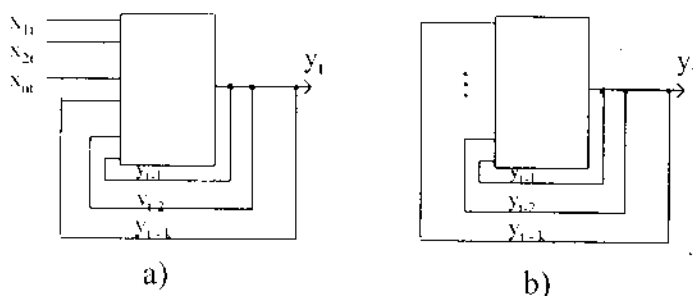
$$\left. \begin{aligned} y_0 &= q; \quad q = \{0, 1\}, \quad t \leq 0 \\ y_t &= \varphi(x_{1t}, x_{2t}, \dots, x_{nt}, y_{t-1}, y_{t-2}, \dots, y_{t-k}), \quad t \geq 0 \end{aligned} \right\} \quad (7.6)$$

7.2.2. Hàm logic truy toán suy biến loại 1 (HLTRS-1)

+ Nếu HLTR-1 lại không phụ thuộc vào các biến x_{it} , $i = \overline{1, n}$ thì gọi là hàm logic truy toán suy biến loại một (HLTRS-1). Biểu thức của hàm logic truy toán suy biến loại một:

$$\left. \begin{aligned} y_0 &= q; \quad q = \{0, 1\}, \quad t \leq 0 \\ y_t &= \varphi(y_{t-1}, y_{t-2}, \dots, y_{t-k}), \quad t \geq 0 \end{aligned} \right\} \quad (7.7)$$

Hình 7.2a là mô hình mạch thực hiện HLTR-1 và hình 7.2b là mô hình mạch thực hiện HLTRS-1.



Hình 7.2. Mô hình mạch thực hiện HLTR-1 (a) và HLTRS-1 (b)

Thí dụ về hàm logic truy toán loại một.

Thí dụ 7.3 Cho HLTR-1 sau:

$$y_0 = 0$$

$$y_t = y_{t-1} + 1$$

Tìm giá trị của hàm.

Hàm logic truy toán này thực chất có thể viết dưới dạng sau đây:

$$y_0 = 0; \quad y_t = 1 \text{ với mọi } t = 1, 2, \dots$$

Thí dụ 7.4: Tìm một số giá trị của HLTR-1 sau:

$$y_0 = 0$$

$$y_t = x_{1t} \cdot \bar{x}_{2t} + y_{t-1}$$

Bằng cách cho t các giá trị 1, 2, 3 ta có:

$$y_1 = x_{11} \cdot \bar{x}_{21} + y_0 = x_{11} \cdot \bar{x}_{21}$$

$$y_2 = x_{12} \cdot \bar{x}_{22} + y_1 = x_{12} \cdot \bar{x}_{22} + x_{11} \cdot \bar{x}_{21}$$

$$y_3 = x_{13} \cdot \bar{x}_{23} + y_2 = x_{13} \cdot \bar{x}_{23} + x_{12} \cdot \bar{x}_{22} + x_{11} \cdot \bar{x}_{21}$$

Một cách tổng quát có thể viết:

$$y_t = \sum_{k=1}^T x_{1k} \cdot x_{2k}$$

T là giá trị lớn nhất của biến t .

7.2.3. Hàm logic truy toán loại 2 (HLTR-2)

+ Nếu hàm logic truy toán thực hiện ánh xạ đơn trị tập hợp:

$$Q = \{x_{1t}, x_{2t}, \dots, x_{nt}, x_{1(t-1)}, x_{2(t-1)}, \dots, x_{n(t-1)}, \dots, x_{1(t-r)}, \dots, x_{n(t-r)}\}$$

lên tập hợp $y = \{0, 1\}$ thì gọi là hàm logic truy toán loại hai (HLTR-2). Nói cách khác, HLTR-2 chỉ phụ thuộc vào các biến vào x_{it} tại thời điểm t và các thời điểm trước đó.

Thí dụ 7.5: Xác định một vài giá trị của HLTR-2 sau đây:

$$y_0 = 0$$

$$y_t = x_{1(t-1)} + x_{2t}$$

Giả sử giá trị của chuỗi tín hiệu vào x_{1t} và x_{2t} vào các thời điểm $t = 1, 2, 3, \dots, 8$ như sau:

$$x_{1t} = 0, 1, 0, 1, 1, 1, 0, 0$$

$$x_{2t} = 0, 0, 1, 1, 1, 0, 1, 0$$

Khi đó ta có:

$$y_0 = 0 \qquad y_5 = 1 + 1 = 1$$

$$y_1 = 0 + 0 = 0 \qquad y_6 = 1 + 0 = 1$$

$$y_2 = 0 + 0 = 0 \qquad y_7 = 1 + 1 = 1$$

$$y_3 = 1 + 1 = 1 \qquad y_8 = 0 + 0 = 0$$

$$y_4 = 0 + 1 = 1$$

+ Trường hợp tổng quát nhất, khi hàm logic thực hiện ánh xạ một cách đơn trị tập hợp các vectơ:

$$Q = \{x_{1t}, x_{2t}, \dots, x_{nt}, x_{1(t-r)}, \dots, x_{n(t-r)}, y_{t-1}, \dots, y_{t-k}\}$$

lên tập hợp $y = \{0, 1\}$ thì gọi chung là hàm logic truy toán.

Thí dụ 7.6: Cho hàm logic truy toán sau:

$$y_{10} = 0$$

$$y_{20} = 1$$

$$y_{30} = 1$$

$$y_{1t} = x_{1t} + y_{2(t-1)}$$

$$y_{2t} = \bar{x}_{1t} + y_{2(t-1)} + y_{2(t-2)}$$

$$y_{3t} = y_{1(t-1)} + y_{2(t-1)}$$

Hãy xác định một số giá trị của y_{1t} , y_{2t} , y_{3t} tại các thời điểm $t = 1, 2, 3$.

Giả sử chuỗi các giá trị của biến vào x_{1t} tại các thời điểm đã cho là:

$$x_{1t} = \{0, 1, 0, 0, \dots\}. \text{ Khi đó:}$$

Với $t = 1$, ta có:

$$y_{11} = x_{11} + y_{20} = 0 + 1 = 1$$

$$y_{21} = \bar{x}_{11} + y_{20} + y_{2(-1)} = 1 + 1 + 1 = 1$$

$$y_{31} = y_{10} + y_{20} = 0 + 1 = 1$$

Với $t = 2$:

$$y_{12} = x_{12} + y_{21} = 1 + 1 = 1$$

$$y_{22} = \bar{x}_{12} + y_{21} + y_{20} = 0 + 1 + 1 = 1$$

$$y_{32} = y_{11} + y_{21} = 1 + 1 = 1$$

Với $t = 3$:

$$y_{13} = x_{13} + y_{22} = 0 + 1 = 1$$

$$y_{23} = \bar{x}_{13} + y_{22} + y_{21} = 1 + 1 + 1 = 1$$

$$y_{33} = y_{12} + y_{22} = 1 + 1 = 1$$

BÀI TẬP CHƯƠNG 7

1. Đơn giản các hàm logic phụ thuộc thời gian sau đây:

$$y_1 = (\bar{x}_1 \bar{x}_2 + \bar{x}_1 x_2 + x_1 \bar{x}_2 + x_1 x_2) \tau_1 + (\bar{x}_1 \bar{x}_2 + \bar{x}_1 x_2) \tau_2$$

$$y_2 = (\bar{x}_1 \bar{x}_2 \bar{x}_3 + \bar{x}_1 \bar{x}_2 x_3 + \bar{x}_1 x_2 \bar{x}_3 + \bar{x}_1 x_2 x_3 + x_1 \bar{x}_2 \bar{x}_3 + x_1 \bar{x}_2 x_3 + x_1 x_2 \bar{x}_3 + x_1 x_2 x_3) \tau_0 + (\bar{x}_1 \bar{x}_2 \bar{x}_3 + \bar{x}_1 \bar{x}_2 x_3 + \bar{x}_1 x_2 \bar{x}_3 + \bar{x}_1 x_2 x_3) \tau_1$$

2. Hãy lấy thí dụ về hàm logic phụ thuộc thời gian có chu kỳ.

3. Hãy xác định để hàm sau đây có chu kỳ:

$$a) y_1 = x_1 \bar{x}_2 \tau_0 + (x_1 + x_2) \tau_1 + \bar{x}_1 \tau_2 + \bar{x}_1 x_2 \tau_4$$

$$b) y_2 = (x_1 + x_2) \tau_0 + x_1 x_2 \tau_3 + x_2 \tau_5$$

4. Tìm một số giá trị đầu tiên của các hàm logic truy toán suy biến loại 1 sau đây:

$$a) \begin{cases} y_0 = 0 \\ y_t = y_{t-1} + y_{t-3} + 1 \end{cases}$$

$$b) \begin{cases} y_0 = 0 \\ y_t = x_{1t} + x_{2t} + y_{t-1} \end{cases}$$

$$c) \quad \begin{cases} y_0 = 1 \\ y_t = x_{1t}y_{t-1} + (x_{1t} + x_{2t})y_{t-2} \end{cases}$$

5. Tìm một số giá trị của hàm sau đây:

a) $y_t = x_{1t} + \bar{x}_{1t}x_{2(t-1)}$ đối với chuỗi tín hiệu vào:

$$x_{1t} = 0, 1, 1, 0, 1, 0, 0, 1, 0$$

$$x_{2t} = 0, 0, 1, 0, 0, 0, 1, 1, 0$$

b) Hàm:

$$y_0 = 1$$

$$y_t = x_{1t} + x_{1(t-1)}y_{t-1} \quad \text{đối với chuỗi tín hiệu vào sau:}$$

$$x_{1t} = 0, 0, 1, 0, 1, 1, 0, 1, 1$$

c) Hệ hàm:

$$y_{10} = 1; \quad y_{1t} = x_{1t}y_{2(t-1)}$$

$$y_{20} = 0; \quad y_{2t} = \bar{x}_{1t} + x_{1t}y_{1(t-1)}$$

$$y_{30} = 0; \quad y_{3t} = y_{2(t-1)}y_{3(t-1)} + y_{1(t-1)}$$

Với chuỗi vào $\{x_{1t}\} = (0, 0, 1, 0, 0, 1, 1, 1)$.

Chương 8

KHÁI NIỆM VỀ THIẾT BỊ SỐ CÓ NHỚ VÀ ÔTÔMÁT HỮU HẠN

8.1. KHÁI NIỆM CHUNG VỀ THIẾT BỊ SỐ CÓ NHỚ

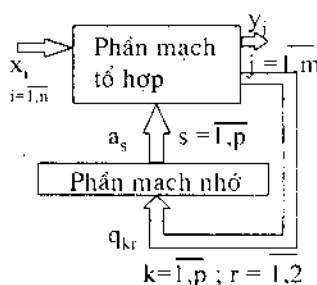
Thiết bị số có nhớ nói chung là một thiết bị mà tín hiệu ra $y(t)$ tại một thời điểm t nào đó phụ thuộc không những vào tín hiệu đầu vào $x_i(t)$ tại thời điểm đó mà còn phụ thuộc vào tín hiệu vào tại các thời điểm trước đó $x_i(t-1)$, $x_i(t-2)$, ..., $x_i(t-k)$. Tức là:

$$y(t) = f(x_i(t), x_i(t-1), \dots, x_i(t-k)) \quad (8.1)$$

Nói một cách khái quát, bất kỳ một thiết bị số nào đều có thể đưa về dạng thiết bị số có nhớ, gồm hai phần:

Phần mạch có nhớ và phần mạch không có nhớ (Phần mạch tổ hợp) - hình 8.1.

Phần mạch nhớ gồm p phần tử nhớ với r đầu vào kích thích dùng để thay đổi trạng thái nhớ của phần tử nhớ. Mỗi phần tử nhớ chỉ có thể nhớ được một trong hai trạng thái là trạng thái 0 và trạng thái 1. Tập hợp các trạng thái có thể có của các phần tử nhớ tạo thành các trạng thái trong của thiết bị số.



Hình 8.1. Mô hình thiết bị số có nhớ

Phần tổ hợp có thể có n đầu vào tín hiệu x_i , $i = \overline{1, n}$ và p đầu vào nội bộ từ các phần tử nhớ, gọi là các biến trong và có m đầu ra y_j , $j = \overline{1, m}$, gọi là các biến ra.

Đối với các mạch tổ hợp (không có nhớ) thì phần nhớ không tồn tại. Đối với các thiết bị số có nhớ thì có thể tồn tại cả phần mạch tổ hợp lẫn phần nhớ hoặc chỉ gồm phần nhớ, nhưng không bao giờ thiếu phần nhớ.

Chúng ta nhận thấy rằng, mạch số có nhớ có một số đặc điểm sau đây:

- Các mạch số có nhớ thường là các mạch có đường phản hồi. Đường phản hồi có thể từ đầu ra trở về đầu vào hoặc từ một điểm nào đó trở lại một điểm phía trước nó.

- Tín hiệu ra tại một thời điểm phụ thuộc không những vào các tín hiệu vào tại thời điểm đang xét mà còn phụ thuộc vào các tín hiệu vào tại các thời điểm trước đó.

Nói cách khác, một mạch có nhớ luôn lưu lại trong nó những trạng thái nào đó, gọi là các trạng thái trong của mạch.

- Sự phụ thuộc của các biến ra, hay còn gọi là các trạng thái ra vào các biến vào hay các trạng thái vào rất phức tạp:

+ Có những trạng thái vào làm thay đổi trạng thái trong và trạng thái ra. Nhưng cũng có những trạng thái vào không làm thay đổi trạng thái trong và trạng thái ra hoặc chỉ thay đổi một trong hai loại trạng thái đó.

+ Có những trạng thái vào lại đưa trạng thái trong và trạng thái ra của mạch hoặc một trong hai trạng thái đó về tình trạng bất định (như trạng thái treo).

+ Nội dung một chuỗi tác động vào đầu vào, thứ tự sắp xếp của các thành phần của chuỗi, độ dài của chuỗi cũng như thời gian tồn tại của từng trạng thái trong chuỗi tác động vào đều ảnh hưởng trực tiếp tới kết quả làm việc của mạch có nhớ.

Từ một số nhận xét sơ bộ trên đây, chúng ta thấy việc nghiên cứu như phân tích và tổng hợp mạch có nhớ hay mạch dây khó khăn hơn nhiều so với nghiên cứu mạch số tổ hợp. Cho tới nay nghiên cứu mạch số nói chung và mạch số có nhớ nói riêng dựa trên lý thuyết ôtomát hữu hạn là phương pháp chặt chẽ, chính xác và khoa học, cho phép tổng hợp và phân tích mọi mạch số, trong đó có cả các mạch số có nhớ.

- Có những mạch làm việc với bất kỳ giới hạn trên nào của thời gian tồn tại của các tập giá trị biến vào (các trạng thái vào). Những mạch này gọi là các mạch loại thế. Chúng làm việc với các tín hiệu dạng thế.

- Có những mạch chỉ làm việc với các trạng thái vào có thời gian tồn tại hạn chế ở một mức nào đó. Các mạch này gọi là các mạch loại xung. Chúng chỉ làm việc đúng với các tín hiệu xung.

Chúng ta cũng cần làm rõ thêm khái niệm về mạch số có nhớ nói chung và ôtomát hữu hạn.

- Mạch số có nhớ nói chung, mà mô hình toán học của nó là biểu thức (8.1), là mạch số mà tín hiệu đầu ra, tại một thời điểm t , phụ thuộc không những vào tín hiệu vào tại thời điểm đó mà còn phụ thuộc vào các tín hiệu vào tại các thời điểm trước đó nữa.

- Ôtomát hữu hạn là một thiết bị số có nhớ, nhưng tín hiệu ra của nó tại một thời điểm t chỉ phụ thuộc vào tín hiệu vào tại thời điểm đó và một thời điểm trước đó. Mô hình toán học của ôtomát hữu hạn có thể viết như sau:

$$y(t) = f(x_i(t), x_i(t-1)) \quad (8.2)$$

Người ta cũng chứng minh rằng, mọi thiết bị số có nhớ đều có thể đưa về dạng ôtomát hữu hạn (8.2). Vì vậy chúng ta sẽ nghiên cứu các phương pháp phân tích và tổng hợp dựa trên lý thuyết của ôtomát hữu hạn.

8.2. MÔ HÌNH CỦA ÔTÔMÁT HỮU HẠN

8.2.1. Mô hình cấu trúc

Mọi thiết bị số có nhớ hay ôtomát hữu hạn đều có thể đưa về dạng cấu trúc (hình 8.1) gồm hai phần: Phần mạch tổ hợp và phần mạch nhớ. Trong đó:

- Tập hợp các tập giá trị của các biến vào $x_i, i = \overline{1, n}$, ký hiệu là $\{X_i\}$, mà:

$$\{X_i\} = (X_0, X_1, \dots, X_{N-1}); N = 2^n; |\{X_i\}| = N$$

gọi là tập hợp các trạng thái vào của thiết bị.

- Tập hợp các tập giá trị của các biến ra $y_j, j = \overline{1, m}$, ký hiệu là $\{Y_j\}$, mà:

$$\{Y_j\} = (Y_0, Y_1, \dots, Y_{M-1}); M = 2^m; |\{Y_j\}| = M$$

gọi là tập hợp các trạng thái ra.

- Tập hợp các tập giá trị các biến trong $a_s, s = \overline{1, p}$, ký hiệu là $\{A_s\}$, mà:

$$\{A_s\} = (A_0, A_1, \dots, A_{P-1}); P = 2^p; |\{A_s\}| = P$$

gọi là tập hợp các trạng thái trong.

8.2.2. Mô hình toán của ôtomát hữu hạn

Từ mô hình cấu trúc ta nhận thấy rằng, hoạt động của một thiết bị số có nhớ - ôtomát hữu hạn là sự thay đổi trạng thái trong A_s và trạng thái ra Y_j dưới tác động của các trạng thái vào X_i . Sự thay đổi của trạng thái trong khi ôtomát đang ở trạng thái A_c sang một trạng thái A_m nào đó trong số tập hợp các trạng thái trong có thể có của ôtomát, dưới tác động của trạng thái vào X_i , được quyết định bởi một hàm gọi là hàm chuyển đổi trạng thái, ký hiệu là φ . Sự phụ thuộc này được viết như sau:

$$A_m = \varphi(A_c, X_i) \quad (8.3)$$

Đồng thời với việc trạng thái trong chuyển từ trạng thái trong A_c sang trạng thái trong A_m dưới tác động của trạng thái vào X_i , ở đầu ra của ôtomát sẽ xuất hiện trạng thái ra mới, ký hiệu là Y_m . Nhưng quy luật xuất hiện trạng thái ra mới này do một hàm khác gọi là hàm ra, ký hiệu là ψ , quyết định. Ta có thể viết như sau:

$$Y_m = \psi(A_c, X_i) \quad (8.4)$$

Như vậy, nếu ta biết các trạng thái vào $X_i, i = \overline{1, N}$; các trạng thái trong $A_s, s = \overline{1, P}$; các trạng thái ra $Y_j, j = \overline{1, M}$; hàm chuyển đổi trạng thái φ và hàm ra ψ thì ta hoàn toàn xác định được cấu tạo và chức năng hoạt động của ôtomát đó.

Vậy có thể nói: Một ôtomát được xác định bằng bộ năm:

$$O = \{X_i, A_s, Y_j, \varphi, \psi\} \quad (8.5)$$

Nói cách khác, một ôtomát được xác định bằng hai công thức toán học sau đây:

$$\left. \begin{aligned} A_m &= \varphi(A_e, X_i) \\ Y_m &= \psi(A_e, X_i) \end{aligned} \right\} \quad (8.6)$$

Hai phương trình của (8.6) gọi là mô hình toán học của ôtomát hữu hạn. Ôtomát hữu hạn được mô tả bằng công thức toán (8.6) gọi là mô hình ôtomát Milly. Bằng cách biến đổi không phức tạp, mọi mô hình toán của ôtomát hữu hạn đều có thể đưa về dạng:

$$\left. \begin{aligned} A_m &= \varphi(A_e, X_i) \\ Y_m &= \psi(A_e) \end{aligned} \right\} \quad (8.7)$$

Nghĩa là trạng thái ra mới chỉ phụ thuộc vào các trạng thái trong mà không phụ thuộc vào trạng thái vào. Ôtomát hữu hạn được mô tả dùng hệ phương trình (8.7) gọi là ôtomát Moore.

Ôtomát Milly và ôtomát Moore chỉ là hai cách mô tả của ôtomát. Giữa ôtomát Milly và ôtomát More đều có thể chuyển đổi từ loại này sang loại kia bằng một biến đổi không lấy gì làm phức tạp. Trong phần sau chúng ta sẽ xét vấn đề này một cách chi tiết hơn.

8.3. CÁC PHƯƠNG PHÁP MÔ TẢ ÔTÔMÁT

8.3.1. Khái quát chung

Tới đây chúng ta đã đề cập tới cơ sở toán học của các loại mạch số. Trong đó một mạch số có thể được mô tả bằng một hàm logic thông thường, một hàm logic phụ thuộc thời gian, một hàm logic truy toán hay dưới dạng biểu thức toán học của ôtomát hữu hạn.

Vấn đề đặt ra là các mô hình toán học trên đây được mô tả dưới dạng triển khai hay dưới dạng tường minh như thế nào để cấu trúc và chức năng của một mạch số được xác định một cách đơn trị, mà ta đều đọc được và hiểu một cách chính xác và thống nhất. Những phương pháp ghi nhận cấu trúc và chức năng hàm của một thiết bị số có nhớ gọi là các phương pháp mô tả ôtomát hữu hạn.

Như chúng ta đã biết, một ôtomát hữu hạn được đặc trưng bởi bộ năm:

$$\{X_i, A_e, Y_o, \varphi, \psi\}$$

và dưới dạng toán học nó được mô tả bằng hệ các phương trình (8.6) và (8.7).

Như vậy, một ôtomát hữu hạn sẽ được hoàn toàn xác định nếu chúng ta biết tập hợp các trạng thái vào $\{X_i\}$; tập hợp các trạng thái trong $\{A_e\}$; tập hợp các trạng thái ra $\{Y_o\}$ và hai hàm φ và ψ . Khi đã biết cấu hình của ôtomát thì việc mô tả thực chất tập trung vào việc mô tả hai hàm φ và ψ .

Có nhiều phương pháp để mô tả hai hàm trên như phương pháp dùng bảng chuyển đổi trạng thái và bảng ra; phương pháp dùng ma trận trạng thái; phương pháp dùng đồ hình trạng thái; lưu đồ trạng thái, phương pháp Grafcet hay phương pháp mạng Pêtri và nhiều phương pháp khác. Trong tài liệu này chúng ta sẽ xét ba phương

pháp quan trọng nhất và cũng là cơ sở để tiếp thu tốt các phương pháp khác. Đó là phương pháp bảng chuyển đổi trạng thái và bảng ra; phương pháp ma trận chuyển đổi trạng thái và phương pháp đồ hình trạng thái.

8.3.2. Phương pháp bảng chuyển đổi trạng thái và bảng ra

1) Bảng chuyển đổi trạng thái

Bảng chuyển đổi trạng thái của ô tô máy dùng để mô tả một cách chi tiết hàm chuyển đổi trạng thái φ của nó. Để mô tả hàm chuyển đổi trạng thái của một ô tô máy nào đó, ta phải biết mọi khả năng chuyển trạng thái của nó xuất phát từ bất kỳ trạng thái ban đầu nào dưới tác động của bất kỳ trạng thái vào nào. Để mô tả điều này, có thể dùng một bảng gọi là bảng chuyển đổi trạng thái.

Bảng chuyển đổi trạng thái có:

- + Số cột bằng số lượng trạng thái vào N của ô tô máy.
- + Số hàng bằng số lượng trạng thái trong P của ô tô máy.

Giao của cột i và hàng s ta ghi trạng thái trong mới mà ô tô máy chuyển tới xuất phát từ trạng thái ban đầu A_s nào đó, ghi tại đầu hàng s , dưới tác động của trạng thái vào X_i ghi tại đầu cột i . Tức là ta thực hiện bước chuyển:

$$A_s \xrightarrow{X_i} A_m$$

thuộc tập hợp các bước chuyển có thể có do hàm φ quyết định.

Như vậy, nếu ta điền vào tất cả các ô của bảng thì tức là ta đã lường hết mọi khả năng của hàm chuyển đổi trạng thái φ .

Trong trường hợp khi trạng thái A_m là bất định, thay vì ghi trạng thái A_m ta ghi dấu bất định “-”. Vì hàm chuyển đổi trạng thái của ô tô máy Milly và của ô tô máy Moore đều giống nhau, phụ thuộc vào trạng thái xuất phát A_s và trạng thái tác động vào X_i , nên bảng chuyển đổi trạng thái của chúng hoàn toàn giống nhau (hình 8.2a).

A \ X	X			
	X_0	X_1	X_2	X_{N-1}
A_0		↓		
A_1	→	A_m		
⋮				
A_s			⋯	⋯
⋮				
A_{P-1}				

a)

A \ X	X			
	X_0	X_1	X_2	X_{N-1}
A_0		↓		
A_1	→	Y_m		
⋮				
A_s			⋯	⋯
⋮				
A_{P-1}				

b)

Hình 8.2. Bảng chuyển đổi trạng thái và bảng ra của ô tô máy Milly

Trên hình 8.2a chỉ ra hai trường hợp: Khi trạng thái mới của ôtomát được hình thành từ trạng thái trong ban đầu A_i dưới tác động của trạng thái vào X_i và trường hợp khi trạng thái mới là một trạng thái bất định khi xuất phát từ trạng thái trong ban đầu A_s nào đó, dưới tác động của trạng thái vào X_i , tức hai bước chuyển trạng thái sau đây:

$$A_i \xrightarrow{X_i} A_m$$

và

$$A_s \xrightarrow{X_i} "-" \text{ (bất định)}$$

2) Bảng ra

Bảng ra là bảng dùng để mô tả hàm ra Ψ của ôtomát, tức là bảng mô tả sự hình thành trạng thái ra mới khi hàm thực hiện bước chuyển trạng thái từ trạng thái ban đầu A_i nào đó sang trạng thái trong mới A_m dưới tác động của trạng thái vào X_i . Vì vậy khuôn dạng của bảng ra của ôtomát Milly hoàn toàn giống bảng chuyển đổi trạng thái. Điều khác duy nhất là trong ô là giao của cột i , $i = 0, (N-1)$ và hàng s , $s = 0, (P-1)$ ta không ghi trạng thái trong mới A_m mà ghi trạng thái ra mới Y_m , tức là thực hiện bước chuyển:

$$A_s \xrightarrow{X_i/Y_m} A_m$$

Trường hợp trạng thái ra mới là bất định, ta cũng ghi vào ô tương ứng như trường hợp bảng chuyển đổi trạng thái, bảng ra của ôtomát Milly cho trên hình 8.2b.

A \ X	X			
	X_0	X_1	X_i	X_{N-1}
A_0		↓		
A_1	→	A_m/Y_m		
⋮				
A_s			↓	
⋮				
A_{P-1}				

a)

Y \ A \ X	X			
	X_0	X_1	X_i	X_{N-1}
A_0		↓		
A_1	→	A_m		
⋮				
$Y_m \ A_s$			↓	
⋮				
A_{P-1}				

b)

Hình 8.3. Bảng chuyển đổi trạng thái và bảng ra của ôtomát bất định

Vì hàm ra của ôtomát Moore không phụ thuộc vào trạng thái vào nên bảng ra của nó chỉ còn lại một cột Y mà mỗi hàng của nó tương ứng với một trạng thái trong của ôtomát. Ngoài ra, nếu ta kết hợp hai bảng chuyển đổi trạng thái và bảng ra thành một

bảng bằng cách chồng khít chúng lên nhau (đối với ô tô máy Milly) hoặc để chúng bên cạnh nhau (đối với ô tô máy Moore) thì ta có bảng chuyển đổi trạng thái và bảng ra kết hợp. Hình 8.3a là bảng chuyển đổi trạng thái và bảng ra kết hợp đối với ô tô máy Milly. Hình 8.3b là bảng chuyển đổi trạng thái và bảng ra kết hợp đối với ô tô máy Moore.

Trên hình 8.4a là một ô tô máy Moore và hình 8.4b là một ô tô máy Milly (xác định không đầy đủ).

Y	X			
	A	X ₀	X ₁	X ₂
Y ₁	A ₀	A ₀	A ₁	A ₂
Y ₁	A ₁	A ₁	A ₂	A ₃
Y ₂	A ₂	A ₂	A ₁	A ₀
Y ₂	A ₃	A ₁	A ₂	A ₂

a)

A	X			
	X ₀	X ₁	X ₂	X ₃
A ₀	A ₁ /Y ₀	A ₀ /Y ₁	A ₁ /Y ₂	A ₀ /Y ₀
A ₁	A ₂ /Y ₂	A ₂ /-	A ₀ /Y ₁	-/-
A ₂	A ₂ /Y ₁	A ₁ /Y ₀	-/Y ₀	A ₀ /Y ₀

b)

Hình 8.4. Bảng chuyển đổi trạng thái và bảng ra của ô tô máy Moore và ô tô máy Milly

3) Trạng thái bền của ô tô máy

Khi ô tô máy đang ở một trạng thái A_i nào đó và có một tác động vào X_k . Nếu ô tô máy không chuyển sang bất kỳ một trạng thái mới nào khác mà vẫn ở lại trạng thái ban đầu A_i thì ta nói A_i là bền đối với X_k .

Trạng thái bền có một ý nghĩa rất quan trọng đối với ô tô máy không đồng bộ, tức là các ô tô máy mà các phần tử nhớ của chúng không làm việc với tín hiệu nhịp hay ô tô máy không làm việc với các tín hiệu vào là các tín hiệu xung. Khi đó tại các trạng thái không phải là các trạng thái bền, ô tô máy không thể cho ra một tín hiệu ra nào vì chúng chỉ lướt qua các trạng thái này một cách hết sức nhanh chóng và không kịp gây tác động đối với các thiết bị ngoại vi bên ngoài, thường là các thiết bị có quán tính lớn.

Trạng thái bền thể hiện trong bước chuyển loại:

$$A_k \xrightarrow{X_i} A_k$$

Trong đó trạng thái mới ghi trong ô là giao của cột X_i và hàng A_k cũng chính là trạng thái A_k . Để dễ nhận biết, các trạng thái bền thường được khoanh tròn bằng một vòng tròn nhỏ như trên hình 8.5a. Trong đó cả bốn trạng thái đều bền đối với trạng thái vào X_0 . Hình 8.5b là thí dụ về một ô tô máy có bốn trạng thái vào: X_0, X_1, X_2, X_3 ; bốn trạng thái trong: A_0, A_1, A_2 và A_3 ; hai trạng thái ra Y_0 và Y_1 . Trong đó các trạng thái trong lại bền đối với các trạng thái vào khác nhau và các trạng thái trong A_1, A_2 và A_3

sẽ trở thành trạng thái không xác định dưới tác động của các tập biến vào X_2 , X_1 và X_0 . Trong ba trường hợp này các trạng thái ra đi kèm cũng trở thành bất định. Ngoài ra, khi ôtomát đang ở trạng thái trong A_3 mà có tác động trạng thái vào X_0 thì trạng thái ra sẽ trở nên bất định.

A \ X	X			
	X_0	X_1	X_2	X_3
A_0	$\textcircled{A_0}$	A_1	A_2	A_3
A_1	$\textcircled{A_1}$	A_2	A_3	A_0
A_2	$\textcircled{A_2}$	A_3	A_0	A_1
A_3	$\textcircled{A_3}$	A_0	A_1	A_2

a)

A \ X	X			
	X_0	X_1	X_2	X_3
A_0	$\textcircled{A_0}, Y_0$	A_2, Y_0	A_3, Y_1	$\textcircled{A_0}, Y_{11}$
A_1	A_0, Y_0	A_2, Y_0	—, —	$\textcircled{A_1}, Y_{11}$
A_2	A_1, Y_0	—, —	$\textcircled{A_2}, Y_1$	A_0, Y_0
A_3	$\textcircled{A_3}, —$	A_0, Y_1	A_0, Y_0	—, —

b)

Hình 8.5. a) Đánh dấu trạng thái bền bằng khoanh tròn;
b) Thí dụ về ôtomát Milly có các trạng thái bền

4) Bảng trạng thái và bảng ra đã mã hoá

Các ôtomát cho dưới dạng hình 8.4 hay 8.5 là cho dưới dạng hình thức. Ta nói đó là các ôtomát hình thức. Để có thể chuyển từ một bảng chuyển đổi trạng thái và bảng ra thành một kết cấu mạch cụ thể, phải thực hiện mã hoá các trạng thái vào, ra và các trạng thái trong của ôtomát.

Mã hoá một trạng thái là gán cho nó một số nhị phân cụ thể nào đó. Nếu mỗi trạng thái vào, trạng thái ra và trạng thái trong đều được gán một số nhị phân nào đó thì ta có bảng chuyển đổi trạng thái và bảng ra đã mã hoá.

Mã hoá là một khâu rất quan trọng trong quá trình thiết kế một thiết bị số loại có nhớ, nó liên quan tới độ đơn giản, độ tin cậy chức năng và tốc độ làm việc của thiết bị. Tức là nó liên quan tới những đặc trưng quan trọng nhất của thiết bị. Chúng ta sẽ nói kỹ hơn vấn đề này trong phần mã hoá trạng thái trong của ôtomát.

Thí dụ, có thể gán các trạng thái của ôtomát ở hình 8.4a và hình 8.5b với các mã như ở hình 8.6.

Thay các giá trị cụ thể này vào các bảng ở hình 8.4a và 8.5b ta được bảng chuyển đổi trạng thái và bảng ra của các ôtomát Moore (hình 8.7a) và ôtomát Milly (hình 8.7b) đã mã hoá.

A	Các biến trong	
	a_1	a_0
A_0	0	0
A_1	0	1
A_2	1	0
A_3	1	1

a)

X	Các biến vào	
	x_1	x_0
X_0	0	0
X_1	0	1
X_2	1	0
X_3	1	1

b)

Y	Các biến ra	
	y_1	y_0
Y_0	0	0
Y_1	0	1
Y_2	1	0

c)

Hình 8.6. Mã hoá các trạng thái trong (a); các trạng thái vào (b) và các trạng thái ra (c)

y_1y_0	x_1x_0 a_1a_0	00	01	10
01	00	00	01	10
01	01	01	10	11
10	10	10	01	00
10	11	01	10	10

a)

x_1x_0 a_1a_0	00	01	10	11
00	00, 00	10, 00	11, 01	00, 00
01	00, 00	10, 00	—, —	01, 00
10	01, 00	—, —	10, 01	00, 00
11	11, —	00, 01	00, 00	—, —

b)

Hình 8.7. Ôtômát Moore và Milly đã được mã hoá

Khi mã hoá ôtômát, có thể sử dụng hết hoặc không sử dụng hết các tập giá trị biến của các biến mã. Thí dụ, để mã hoá ôtômát Moore trên đây, khi mã hoá ba trạng thái vào, ta đã không dùng tập mã 11 của hai biến mã x_1x_0 . Tương tự như vậy, đối với hai tập mã của 00 và 11 của y_1y_0 cũng không được sử dụng. Ngược lại, khi mã hoá ôtômát Milly, tất cả các tập giá trị của các biến mã đều đã được sử dụng.

Trong trường hợp có các tập mã thừa thì việc chọn tập mã nào để mã hoá và bỏ tập mã nào là tùy ý. Tất nhiên, khi đi sâu vào việc mã hoá và tính chất của các thiết bị được mã hoá thì việc lựa chọn tập mã nào để mã hoá là rất có ý nghĩa. Trong phần này chúng ta chưa có điều kiện để đề cập cụ thể tới vấn đề này.

Khi có các tập mã thừa không được dùng để mã hoá các trạng thái vào, các trạng thái trong và các trạng thái ra của ôtômát thì các ôtômát này được gọi là các ôtômát không xác định toàn phần theo các tập biến vào, các tập biến trong hay các tập biến ra, tùy theo loại tập biến nào thừa không được sử dụng.

Như vậy, tính không xác định toàn phần của ôtomát có liên quan tới việc mã hoá các trạng thái của nó. Mã hoá là một khâu quan trọng trong thiết kế ôtomát, ta sẽ xét kỹ hơn trong phần tổng hợp ôtomát.

8.3.3. Phương pháp ma trận chuyển đổi trạng thái của ôtomát

Ma trận mô tả ôtomát là một ma trận vuông. Các hàng và các cột của ma trận này tương ứng với các trạng thái trong của ôtomát. Nghĩa là số hàng và số cột của ma trận bằng số trạng thái trong của ôtomát. Trong ô là giao của cột A_i và hàng A_j , ta ghi trạng thái vào (tập giá trị biến vào) tương ứng mà dưới tác dụng của nó ôtomát chuyển từ trạng thái A_j sang trạng thái A_i , đồng thời có ghi cả trạng thái ra tương ứng (hình 8.8a).

	A_1		A_j		A_p
A_1					
A_2					
$\vdots$					
A_i					
$\vdots$					
A_p					

a)

	A_0	A_1	A_2	—
A_0	X_1/Y_1 X_3/Y_0	X_0/Y_0 X_2/Y_1	—	—
A_1	X_2/Y_1	—	X_0/Y_2 $X_1/-$	$X_3/-$
A_2	X_3/Y_0	X_1/Y_0	X_0/Y_1	X_2/Y_0

b)

Hình 8.8. Dạng của ma trận chuyển đổi và dạng ma trận của ôtomát Milly trên hình 8.4b

Ôtomát Milly cho trên hình 8.4b có thể viết dưới dạng ma trận chuyển đổi trạng thái như ở hình 8.8b. Ma trận ở hình 8.8b có thể viết dưới dạng rút gọn như hình 8.9. Trong đó ma trận có bốn cột vì có thêm trạng thái bất định “—”. Số hàng vẫn chỉ là 3 vì các bước chuyển không bao giờ xuất phát từ trạng thái bất định. Nếu từ trạng thái A_i ứng với một hàng nào đó mà không tồn tại bước chuyển tới một trạng thái tương ứng với cột thì trong ô tương ứng ta ghi giá trị bất định “—”.

Dạng ma trận chuyển đổi trạng thái ở hình 8.8 và 8.9 có thể phân tích thành các ma trận con của các chuyển đổi, tương ứng mỗi trạng thái vào X_0, X_1, X_2 và X_3 .

$$MR = \begin{bmatrix} X_1/Y_1 & VX_3/Y_0 & X_0/Y_0 & VX_2/Y_1 \\ X_2/Y_1 & & X_0/Y_2 & VX_1/- \\ X_3/Y_0 & & X_1/Y_0 & X_0/Y_1 & X_2/Y_0 \end{bmatrix}$$

Hình 8.9. Dạng ma trận của ôtomat dùng để phân tích thành các ma trận con

Hình 8.10 là các ma trận con, tương ứng với bốn trạng thái vào X_0, X_1, X_2 và X_3 của ôtomát Milly ở hình 8.8.

$$\begin{aligned} \text{MR } X_0 &= \begin{bmatrix} 0 & 1 & - & - \\ 0 & - & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} & \text{MR } X_1 &= \begin{bmatrix} 1 & 0 & - & - \\ 0 & - & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \\ \text{MR } X_2 &= \begin{bmatrix} 0 & 1 & - & - \\ 1 & - & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} & \text{MR } X_3 &= \begin{bmatrix} 1 & 0 & - & - \\ 0 & - & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \end{aligned}$$

Hình 8.10. Các ma trận con tương ứng

Trong đó thành phần “1” của của ma trận cho biết dưới tác động của trạng thái vào tương ứng sẽ xảy ra sự chuyển trạng thái từ A_i , ứng với đầu hàng sang trạng thái A_j , ứng với đầu cột. Giá trị “0” của ma trận chỉ rằng không có bước chuyển xảy ra. Ký hiệu bất định “-” là những chuyển biến không thể có đối với ô tô máy đã cho.

Dạng ma trận của ô tô máy, ngoài ma trận chuyển trạng thái và ma trận ra, còn có ma trận các biến vào.

B.chuyển	Các biến vào			
$A_1 \rightarrow A_1$	X_1		X_k	X_N
$A_1 \rightarrow A_2$				
$A_1 \rightarrow A_p$				
$A_1 \rightarrow A_1$				
$A_p \rightarrow A_1$				
$A_p \rightarrow A_p$				

a)

Chuyển trạng thái	Các biến vào	
	X_1	X_0
$00 \rightarrow 00$	*	1
$00 \rightarrow 01$	*	0
$00 \rightarrow 10$	-	-
$00 \rightarrow --$	-	-
$01 \rightarrow 00$	1	0
$01 \rightarrow 01$	-	-
$01 \rightarrow 10$	0	*
$01 \rightarrow --$	1	1
$10 \rightarrow 00$	1	1
$10 \rightarrow 01$	0	1
$10 \rightarrow 10$	0	0
$10 \rightarrow --$	1	0

b)

Hình 8.11. Dạng ma trận chuyển đổi đã mã hoá

Dấu G: Ghi giá trị của X_k , khi đó bước chuyển A_i sang A_j

Ma trận các biến vào có số hàng bằng số lượng các bước chuyển trạng thái có thể có của ô tô máy và có số cột bằng số lượng các biến vào của ô tô máy đó. Trong ô là giao của hàng, ứng với bước chuyển A_i sang A_j và cột, ứng với trạng thái vào X_k ta ghi giá trị cụ thể như sau:

- Nếu giá trị của X_k đối với bước chuyển đó là xác định thì ghi rõ là giá trị 0 hay 1.

- Nếu giá trị của x_k là tùy ý là 0 hay 1 thì ghi kí hiệu “*”.
- Nếu là ứng với tập biến vào bất định thì ghi giá trị bất định “-”.

Dạng của ma trận các biến vào cho trên hình 8.11a.

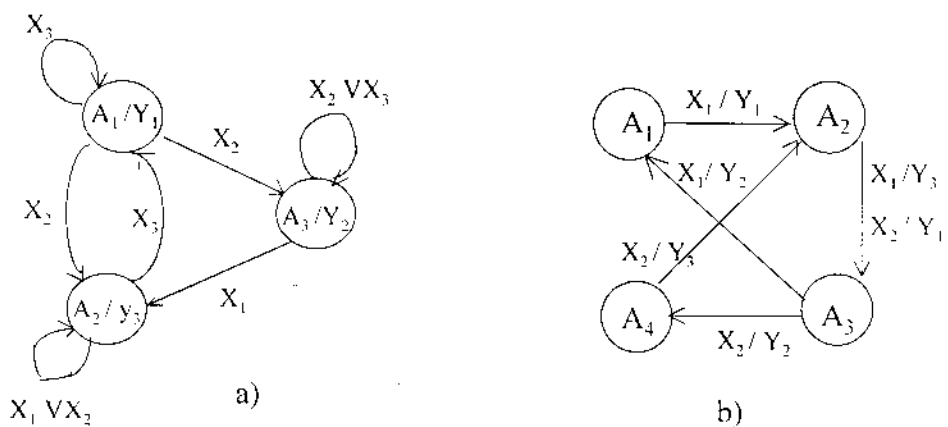
Ma trận các biến vào của ôtomát ở hình 8.4b đã mã hoá theo bộ mã ở hình 8.6 cho trên hình 8.11b.

8.3.4. Đồ hình của ôtomát

Đồ hình của ôtomát được xây dựng trên cơ sở đồ hình định hướng. Dạng đồ hình của ôtomát là dạng trực quan, dễ nhận biết các bước chuyển và những đặc điểm làm việc của ôtomát nên thường được sử dụng trong bài toán phân tích.

Giữa bảng chuyển đổi trạng thái và bảng ra với đồ hình trạng thái có một quan hệ chuyển đổi đơn trị. Trong đó:

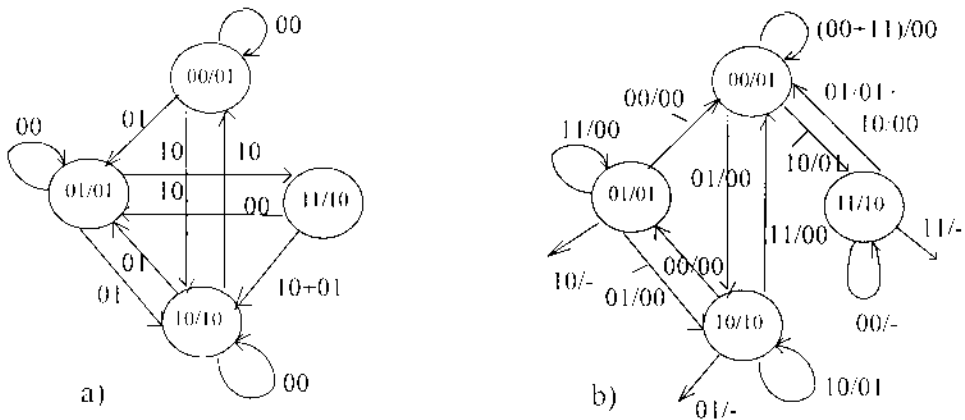
- Mỗi trạng thái trong A_i của ôtomát được mô tả bằng một đỉnh của đồ hình. Vậy nếu muốn mô tả một ôtomát có P trạng thái trong ta cần dùng một đồ hình có P đỉnh.
- Mỗi bước chuyển từ trạng thái A_i sang trạng thái A_j của ôtomát dưới tác động của trạng thái vào X_r và cho ra trạng thái ra Y_m (đối với ôtomát Milly) ta dùng một cung có hướng đi từ trạng thái A_i sang trạng thái A_j . Trên cung ta ghi X_r/Y_m .
- Trường hợp đối với ôtomát Moore, vì trạng thái ra chỉ phụ thuộc vào trạng thái trong, mỗi trạng thái trong có một trạng thái ra tương ứng mà không phụ thuộc vào trạng thái vào, do đó trong đồ hình của ôtomát Moore ta ghi trạng thái ra ngay tại đỉnh tương ứng với trạng thái trong mà trạng thái ra đó phụ thuộc.
- Nếu nhiều trạng thái vào cùng gây nên một bước chuyển trạng thái trong như nhau, trên đồ hình ta ghi chung một cung với cộng logic của các trạng thái vào và trạng thái ra đi kèm.



Hình 8.12. Dạng đồ hình của ôtomát Moore (a) và Milly (b)

Hình 8.12a là đồ hình của một ô tô măt Moore có 3 trạng thái trong, ba trạng thái ra và ba trạng thái vào. Hình 8.12b là đồ hình của ô tô măt Milly có bốn trạng thái trong, hai trạng thái vào và ba trạng thái ra.

Ô tô măt Moore (hình 8.7a) và ô tô măt Milly (hình 8.7b) chuyển sang dạng đồ hình trạng thái cho trên hình 8.13a và b.



Hình 8.13. Dạng đồ hình của ô tô măt ở hình 8.7a, b

Ba phương pháp mô tả ô tô măt đã xét trên đây là hoàn toàn tương đương. Ta có thể dễ dàng chuyển từ dạng này sang dạng khác. Việc sử dụng dạng mô tả nào là tùy thuộc vào hoàn cảnh và điều kiện cụ thể, sao cho thuận tiện. Dạng đồ hình đơn giản và trực quan, dễ dàng cho việc nhận xét đánh giá về sự chuyển đổi trạng thái của ô tô măt, nhưng ít thuận tiện hơn trong việc biến đổi, đơn giản trạng thái trong của ô tô măt. Nó phù hợp với trường hợp ô tô măt có ít trạng thái. Dạng bảng chuyển đổi trạng thái và bảng ra và dạng ma trận ít trực quan hơn trong nhận xét đánh giá hoạt động của ô tô măt nhưng lại thuận tiện hơn trong biến đổi, đơn giản trạng thái trong của nó.

Do điều kiện hạn chế của tài liệu này, phương pháp mô tả hoạt động của ô tô măt bằng sơ đồ logic các algorit; bằng lưu đồ hoạt động cũng như bằng Grafcet và bằng mạng Pêtri chưa được đề cập tới. Phương pháp Grafcet và lý thuyết mạng Pêtri được viết trong các tài liệu chuyên khảo khác của tác giả.

8.4. QUAN HỆ GIỮA Ô TÔ MẶT MILLY VÀ Ô TÔ MẶT MOORE

8.4.1. Vấn đề tương đương của hai loại ô tô măt

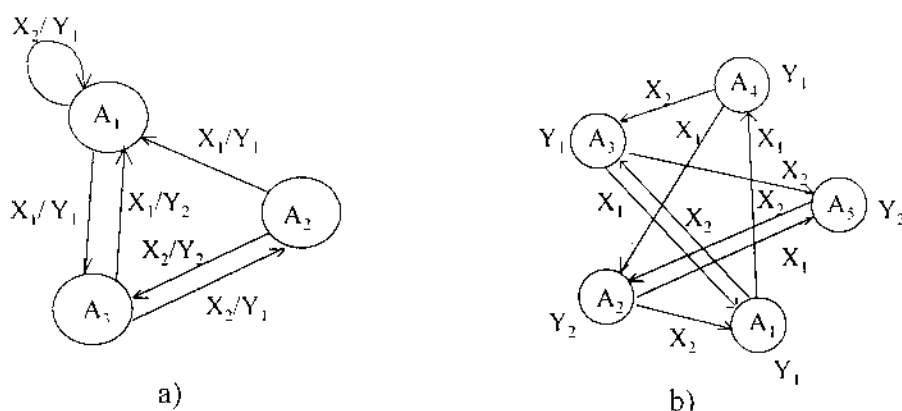
Như ta đã biết, một ô tô măt bất kỳ đều có thể được mô tả dưới dạng ô tô măt Milly hoặc dưới dạng ô tô măt Moore. Nói cách khác, mô hình ô tô măt Milly và mô hình ô tô măt Moore chỉ là dạng mô tả toán học khác nhau của một ô tô măt. Như vậy, với mỗi ô tô măt Milly có một ô tô măt Moore tương ứng và ngược lại, với mỗi ô tô măt Moore có một ô tô măt Milly tương ứng.

Việc chuyển đổi từ mô hình ôtomát Milly sang mô hình ôtomát Moore dựa trên khái niệm hai ôtomát tương đương.

Hai ôtomát S_1 và S_2 là tương đương nếu chúng cùng xuất phát từ những trạng thái ban đầu như nhau, dưới tác động của chuỗi các trạng thái vào như nhau thì chuỗi các trạng thái ra của chúng phải như nhau.

Qua khái niệm trên đây ta thấy, hai ôtomát tương đương S_1 và S_2 cũng phải có các trạng thái vào và ra như nhau. Nhưng số lượng trạng thái trong của chúng có thể khác nhau.

Thí dụ, cho hai ôtomát Milly S_1 và Moore S_2 như ở hình 8.14a và b sau đây. Ta sẽ xét kết quả hoạt động của hai ôtomát này khi cả hai đều xuất phát từ trạng thái A_1 , dưới tác động của chuỗi các trạng thái vào là: $\{X_i\} = X_1, X_1, X_2, X_1, X_2, X_2$.



Hình 8.14. Ôtomát Moore (b) và ôtomát Milly (a).

Căn cứ vào đồ hình trạng thái của ôtomát Milly (hình 8.14a), ta có chuỗi các trạng thái trong $\{A\}$ và chuỗi các trạng thái ra $\{Y\}$ tương ứng khi có tác động của chuỗi các trạng thái vào đã cho $\{X\}$ như sau:

$$\{X\} = X_1 \ X_1 \ X_2 \ X_1 \ X_2 \ X_2$$

$$\{A\} = A_1 \ A_3 \ A_1 \ A_1 \ A_3 \ A_2 \ A_3$$

$$\{Y\} = Y_1 \ Y_2 \ Y_1 \ Y_1 \ Y_1 \ Y_2$$

Qua đây ta thấy phản ứng đối với chuỗi gồm sáu trạng thái vào, ôtomát S_1 cho ta chuỗi gồm 7 trạng thái trong và chuỗi gồm 6 trạng thái ra tương ứng.

Đối với ôtomát Moore (hình 8.14b), từ đồ hình trạng thái ta có các chuỗi trạng thái vào, trong và ra như sau:

$$\{X\} = X_1 \ X_1 \ X_2 \ X_1 \ X_2 \ X_2$$

$$\{A\} = A_1 \ A_1 \ A_2 \ A_1 \ A_4 \ A_3 \ A_5$$

$$\{Y\} = Y_1 \ Y_1 \ Y_2 \ Y_1 \ Y_1 \ Y_1 \ Y_2$$

Từ thí dụ trên đây ta thấy rằng hai ô tô máy S_1 và S_2 đều có các trạng thái vào và ra như nhau là X_1, X_2 và Y_1, Y_2 mặc dù số lượng trạng thái trong của chúng là khác nhau: S_1 có 3 trạng thái trong, S_2 có 5 trạng thái trong. Khi hoạt động, khi cùng xuất phát từ trạng thái A_1 dưới tác động vào cùng một chuỗi trạng thái vào $\{X\} = X_1 X_1 X_2 X_1 X_2 X_2$ như nhau, với độ chính xác tới một nhịp công tác, chuỗi ra của chúng hoàn toàn như nhau. Chuỗi trạng thái ra $\{Y\}$ của S_2 trong khung hoàn toàn giống chuỗi ra của S_1 . Vậy hai ô tô máy S_1 và S_2 là tương đương nhau. Nói cách khác, chúng thực hiện cùng một chức năng.

Tuy vậy cũng cần chú ý rằng, trên thực tế không phải lúc nào hai loại ô tô máy cú tương đương nhau là thay thế cho nhau được. Đó là vì đối với ô tô máy Moore, tín hiệu ra chỉ có quan hệ với trạng thái trong của nó mà không trực tiếp phụ thuộc vào các trạng thái vào. Ngược lại, tín hiệu ra ở ô tô máy Milly lại chỉ xuất hiện khi có tác động của trạng thái vào. Nó chỉ tồn tại liên tục với cùng một giá trị khi tất cả các trạng thái của chuỗi trạng thái vào đều bền đối với trạng thái trong mà nó chuyển đến. Ngoài ra, như chúng ta sẽ chỉ rõ dưới đây, số lượng trạng thái trong của ô tô máy Moore thường lớn hơn hay bằng số lượng trạng thái trong của ô tô máy Milly tương đương. Mặt khác, trên thực tế cũng rất hay phải chuyển đổi từ loại ô tô máy này sang loại ô tô máy khác

8.4.2. Phương pháp chuyển đổi từ ô tô máy Moore sang ô tô máy Milly

Xuất phát từ đồ hình và nguyên tắc hoạt động của hai loại ô tô máy, ta có thể chuyển từ đồ hình của ô tô máy Moore :

$$S_{Mu} = \{X_u, A_u, Y_u, \varphi_u, \psi_u \text{ và } A_{u0}\}$$

trong đó: X_u - tập hợp các trạng thái vào;

A_u - tập hợp các trạng thái trong;

Y_u - tập hợp các trạng thái ra;

φ_u - hàm chuyển đổi trạng thái của ô tô máy;

ψ_u - hàm ra của ô tô máy;

A_{u0} - trạng thái ban đầu.

sang đồ hình của ô tô máy Milly tương đương :

$$S_{Mi} = \{X_i, A_i, Y_i, \varphi_i, \psi_i \text{ và } A_{i0}\}$$

trong đó: $X_i = X_u$ - tập hợp các trạng thái vào;

$A_i = A_u$ - tập hợp các trạng thái trong;

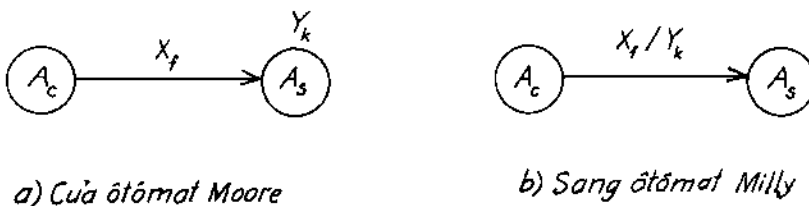
$Y_i = Y_u$ - tập hợp các trạng thái ra;

$\varphi_i = \varphi_u$ - hàm chuyển đổi trạng thái của ô tô máy;

$A_{i0} = A_{u0}$ - trạng thái ban đầu.

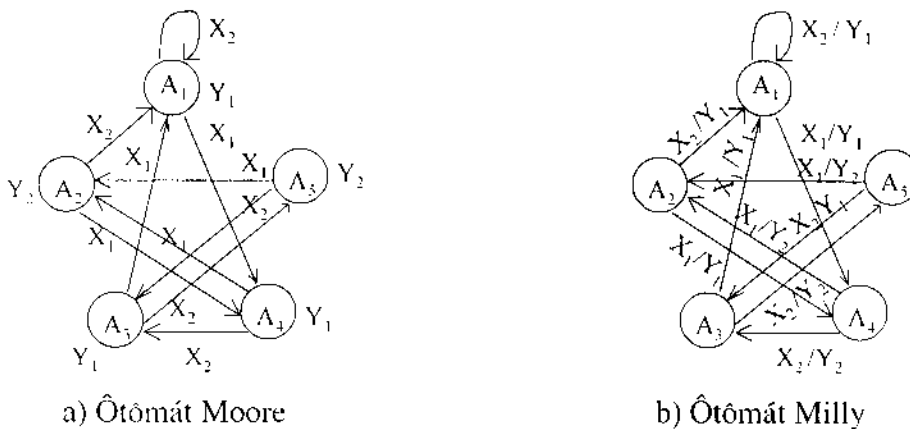
Để chuyển từ đồ hình của ôtomát Moore sang đồ hình của ôtomát Milly, ta xác định hàm ra của ôtomát Milly. Việc xác định hàm ra của ôtomát Milly thực hiện như sau:

Nếu trong ôtomát Moore có bước chuyển đổi $\varphi_u(A_i, x_f) \longrightarrow A_s$ và $\psi_u(A_s) = Y_k$, trong ôtomát Milly tương đương ta ghi $\psi_u(A_i, X_f) = Y_k$. Trên đồ hình trạng thái, việc chuyển đổi này cho trên hình 8.15. Nói cách khác, nếu tại đỉnh A_s của ôtomát Moore có gán giá trị Y_k thì khi chuyển sang ôtomát Milly trạng thái ra Y_k được gán cho bất kỳ cung nào của đồ hình đi tới đỉnh A_s . Bằng cách đó ôtomát Moore (hình 8.16a) có ôtomát Milly tương đương (hình 8.16b).



Hình 8.15. Bước chuyển dạng Moore sang bước chuyển dạng Milly

Khi các ôtomát cho dưới dạng bảng, bảng chuyển đổi trạng thái của ôtomát Milly hoàn toàn giống bảng chuyển đổi trạng thái của ôtomát Moore; bảng ra của ôtomát Milly dễ dàng nhận được từ bảng chuyển đổi trạng thái của ôtomát Moore bằng cách thay trạng thái trong A_s trong ô là giao điểm của hàng A_i với cột X_f của bước chuyển trạng thái $\varphi_u(A_i, X_f) \longrightarrow A_s$ bằng giá trị Y_k tương ứng với trạng thái trong A_s của ôtomát Moore.



Hình 8.16. Chuyển đổi từ ôtomát Moore sang ôtomát Milly

Thí dụ, bằng cách chuyển đổi như đã chỉ dẫn, từ bảng chuyển đổi trạng thái và bảng ra kết hợp của ô tô măt Moore cho trên hình 8.17a, ta nhận được bảng chuyển đổi trạng thái và bảng ra kết hợp của ô tô măt Milly tương đương như ở hình 8.17b.

Y_u	X_u	X_1	X_2
A_u	A_u	A_u	A_u
Y_1	A_1	A_4	A_1
Y_2	A_2	A_4	A_1
Y_3	A_3	A_1	A_5
Y_4	A_4	A_2	A_3
Y_5	A_5	A_2	A_3

a) Bảng chuyển đổi trạng thái và bảng ra của ô tô măt Moore

X_i	X_1	X_2
A_i	A_i	A_i
A_1	A_4/Y_1	A_1/Y_1
A_2	A_4/Y_1	A_1/Y_1
A_3	A_1/Y_1	A_5/Y_2
A_4	A_2/Y_2	A_3/Y_1
A_5	A_2/Y_2	A_3/Y_1

b) Bảng chuyển đổi trạng thái và bảng ra của ô tô măt Milly tương đương

Hình 8.17. Chuyển từ ô tô măt Moore sang ô tô măt Milly dưới dạng bảng

Từ chính phương pháp chuyển tương đương ta thấy rằng, S_i hoàn toàn tương đương với S_u . Thật vậy, nếu có tập biến vào $X_i \in \{X\}$ tác động lên S_u đang ở trạng thái A_u thì sẽ chuyển sang trạng thái A_s : $\varphi_u(A_u, X_i) \longrightarrow A_s$ và cho ra tín hiệu ra Y_k : $\psi(A_u) = Y_k$. Đối với ô tô măt Milly S_i , khi đó ta cũng có:

$$\varphi_u(A_u, X_i) \longrightarrow A_s$$

và cũng cho ra $\psi(A_u, X_i) = Y_k$.

Vì A_u và A_s là các trạng thái trong bất kỳ của ô tô măt A_u và X_i là một trạng thái vào bất kỳ thuộc tập hợp các trạng thái vào $\{X_u\} = \{X_i\}$, nên hoạt động của hai ô tô măt S_u và S_i là như nhau.

Qua phương pháp chuyển đổi tương đương ta thấy rằng, bất kỳ một ô tô măt loại Moore nào đều có thể chuyển thành một ô tô măt loại Milly có cùng số lượng trạng thái trong.

8.4.3. Chuyển từ dạng ô tô măt Milly sang ô tô măt Moore

Trước khi xét phương pháp chuyển từ dạng ô tô măt Milly sang ô tô măt Moore, ta giả thiết rằng ô tô măt Milly không có các trạng thái đặc biệt. Trạng thái đặc biệt là trạng thái mà ở dạng đồ hình trạng thái không có cung đi vào, nhưng có ít nhất một cung đi ra từ trạng thái đó.

Bây giờ giả thiết rằng cần chuyển ô tô măt Milly S_i sau đây:

$$S_i = \{X_i, A_i, Y_i, \varphi_i, \psi_i, \Lambda_{i0}\}$$

sang ôttômát Moore tương đương, được xác định bởi các đại lượng tương ứng sau:

$$S_u = \{X_u, A_u, Y_u, \varphi_u, \psi_u \text{ và } \Lambda_{u0}\}$$

trong đó: $X_u = X_i$ - tập hợp các trạng thái vào;

A_u - tập hợp các trạng thái trong;

$Y_u = Y_i$ - tập hợp các trạng thái ra.

Trong trường hợp này chúng ta cần xác định tập hợp các trạng thái trong A_u , các hàm chuyển đổi trạng thái φ_u và hàm ra ψ_u của ôttômát Moore S_u tương đương.

a) Xác định tập hợp các trạng thái trong A_u

Đối với mỗi trạng thái A_i thuộc tập hợp các trạng thái trong của ôttômát Milly sẽ có một tập hợp các trạng thái A_s thuộc tập hợp các trạng thái trong của ôttômát Moore tương ứng. Trong đó mỗi trạng thái của tập hợp này ứng với một giá trị Y_k của các cung đi vào đỉnh A_s . Tức là:

$$\Lambda_s = \{(A_s, Y_k) \mid \varphi_i(A_i, X_i) \longrightarrow A_s \text{ và } \psi(A_i, X_i) = Y_k\}$$

Số lượng phần tử của tập hợp A_s đúng bằng số các tập tín hiệu ra Y_k khác nhau trên các cung đi vào đỉnh A_s của ôttômát S_i .

Như vậy số lượng trạng thái của ôttômát Moore A_u sẽ phải là:

$$A_u = \bigcup_{s=1}^n \Lambda_s$$

trong đó n là số lượng trạng thái trong của S_i .

b) Xác định hàm chuyển đổi trạng thái và hàm ra

Hàm chuyển đổi trạng thái φ_u và hàm ra ψ_u được xác định như sau:

Mỗi trạng thái của ôttômát Moore S_u ứng với một cặp nhất định (A_s, Y_k) của ôttômát Milly S_i ta gán với một trạng thái ra Y_k và nếu trong S_i có bước chuyển $\varphi_i(A_i, X_i) \longrightarrow A_s$, trong đó sinh ra trạng thái ra $\psi(A_i, X_i) = Y_k$ thì trong đồ hình trạng thái của S_u tương đương, từ tất cả các đỉnh trong tập hợp đỉnh $\{A_s\}$ do A_i của S_i sinh ra phải có cung tới trạng thái của S_u , ứng với cặp (A_s, Y_k) dưới tác động của X_i .

Trạng thái ban đầu A_{u0} có thể là bất kỳ trạng thái nào trong tập hợp các trạng thái ban đầu $\{A_{u0}\}$ của S_u , do A_{i0} của S_i sinh ra.

Thí dụ, ôttômát Milly (hình 8.18a) được biến đổi thành ôttômát Moore (hình 8.18b).

Đối với ôttômát Milly này ta có:

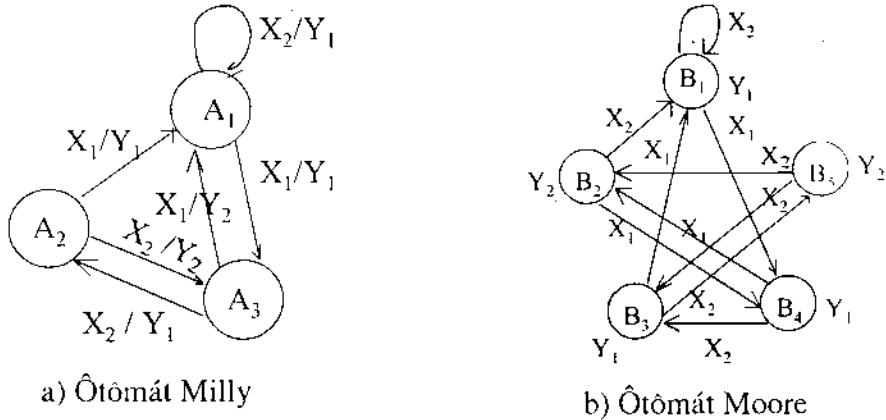
$$X_i = \{X_1, X_2\}$$

$$\Lambda_i = \{A_1, A_2, A_3\}$$

$$Y_i = \{Y_1, Y_2\}$$

$$A_{10} = A_1$$

Các hàm chuyển trạng thái φ_i và hàm ra ψ_i được thể hiện bằng các cung có hướng của đồ hình 8.18a và các giá trị X_i và Y_i tương ứng ghi trên các cung.



Hình 8.18. Chuyển đổi từ ôtômát Milly sang ôtômát Moore

Trong ôtômát Moore tương đương, bắt buộc phải có:

$$X_u = X_i = \{X_1, X_2\} \text{ và}$$

$$Y_u = Y_i = \{Y_1, Y_2\}$$

Việc xác định tập hợp các trạng thái trong A_u của ôtômát Moore tương đương S_u được tiến hành như sau:

Với ba định A_1, A_2 , và A_3 của ôtômát Milly ta có:

$$A_1 = \{(A_1, Y_1), (A_1, Y_2)\} = (B_1, B_2)$$

$$A_2 = \{(A_2, Y_1)\} = (B_3)$$

$$A_3 = \{(A_3, Y_1), (A_3, Y_2)\} = (B_4, B_5)$$

Vậy tập hợp các trạng thái trong của ôtômát Moore tương đương có năm trạng thái, tức năm phần tử của tập hợp là $\{B_1, B_2, B_3, B_4$ và $B_5\}$:

$$A_u = A_1 \cup A_2 \cup A_3 = \{B_1, B_2, B_3, B_4 \text{ và } B_5\}$$

Hàm ra của ôtômát Moore được xác định trên cơ sở các trạng thái trong B_m ; $m = \overline{1, 5}$ của nó với các giá trị của các tập biến ra kèm theo như sau:

$$\psi_u(B_1) = Y_1; \psi_u(B_2) = Y_2; \psi_u(B_3) = Y_1; \psi_u(B_4) = Y_1; \psi_u(B_5) = Y_2$$

Hàm chuyển đổi trạng thái φ_u của ôtômát Moore được xác định như sau:

Vì trong ôtômát S_i có bước chuyển trạng thái $\varphi_i(A_1, X_1) \rightarrow A_3$ và cho ra Y_1 nên trong ôtômát Moore tương đương S_u , từ các trạng thái thuộc $A_1 = \{B_1, B_2\}$, do A_1 sinh ra, phải có bước chuyển tới B_3 , là trạng thái do A_3 sinh ra, dưới tác động của trạng thái vào X_1 . Tức là phải có:

$$\varphi_u(B_1, X_1) \longrightarrow B_2 \text{ và } \varphi_u(B_2, X_1) \longrightarrow B_3$$

Cùng lý luận tương tự, ta thấy từ tập hợp các trạng thái $A_1 = \{B_1, B_2\}$, phải có các bước chuyển vào trạng thái $B_1 = (A_1, Y_1)$ dưới tác động của X_2 ...

Cùng lý luận tương tự đối với các trạng thái và bước chuyển khác của S_1 ta có đồ hình trạng thái của ôtomát Moore tương đương như trên hình 8.18b.

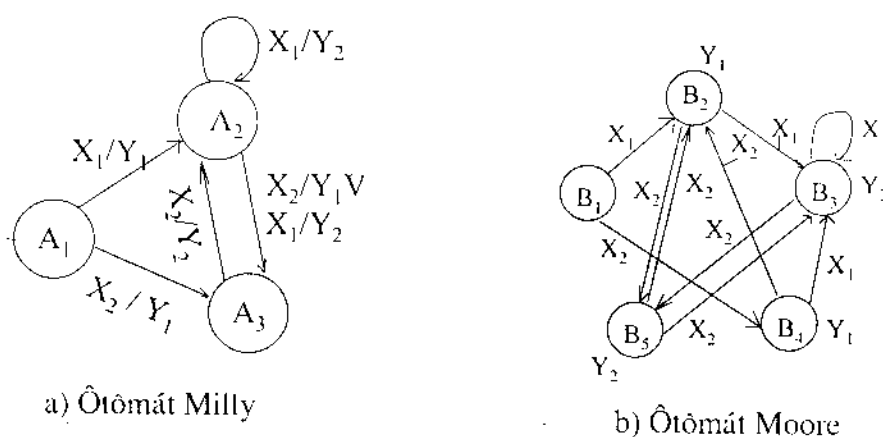
Trạng thái ban đầu trong ôtomát Moore S_u có thể là bất kỳ trạng thái nào trong tập hợp các trạng thái $A_1 = \{B_1, B_2\}$.

Trên đây ta giả thiết là ôtomát Milly S_1 không có trạng thái đặc biệt, chỉ có các cung đi ra mà không có cung đi vào trạng thái đó. Đó thường là trạng thái xuất phát ban đầu. Nếu trong đồ hình trạng thái của ôtomát Milly có trạng thái đặc biệt thì khi chuyển sang đồ hình trạng thái của ôtomát Moore S_u tương đương, trạng thái đặc biệt A_0 của S_1 sẽ trở thành trạng thái B_0 của S_u mà:

$$B_0 = (A_0, -)$$

Dấu bất định “-” là biểu hiện rằng tín hiệu ra không được xác định. Trạng thái B_0 cũng là trạng thái ban đầu của ôtomát Moore S_u . Để thực hiện việc chuyển đổi, cách làm cũng tương tự trường hợp trên đây. Tính tương đương của hai ôtomát S_1 và S_u được xác định từ bản thân cách làm.

Trên hình 8.19a là ôtomát Milly ba trạng thái trong với trạng thái A_1 là trạng thái đặc biệt. Trên hình 8.19b là ôtomát Moore tương đương của nó, với năm trạng thái trong $B_j, j = \overline{1, 5}$.



Hình 8.19. Ôtomát S_u tương đương khi S có trạng thái đặc biệt

Trong trường hợp này ta có:

$$A_1 = \{A_1, -\} = \{B_1\}$$

$$A_2 = \{(A_2, Y_1), (A_2, Y_2)\} = \{B_2, B_3\}$$

$$A_3 = \{(A_3, Y_1), (A_3, Y_2)\} = \{B_4, B_5\}$$

Khi xác định các bước chuyển của S_0 cần chú ý rằng, từ B_1 chỉ có hai bước chuyển sang B_2 dưới tác dụng của trạng thái vào X_1 và sang B_4 dưới tác động của trạng thái vào X_2 . Mặc dù B_4 và B_5 cũng do A_2 và A_3 sinh ra, nhưng vì giá trị trạng thái ra trong S_1 là $Y_2 \neq Y_1$, nên không tồn tại bước chuyển từ B_1 tới B_3 và B_5 .

Từ B_4 và B_5 đều có hai bước chuyển sang B_2 và B_3 với tác động của X_1 và X_2 và các hàm ra Y_1 và Y_2 tương ứng như trên đồ hình của S_0 , vì trên đồ hình của S_1 có các bước chuyển tương ứng: Từ A_4 , dưới tác động của X_2 , ô tô máy chuyển sang A_2 và cho ra Y_1 . Nhưng nếu dưới tác động của X_1 cũng sẽ sang A_2 , nhưng cho ra Y_2 .

Ta nhận xét rằng, từ mô hình ô tô máy Milly chuyển sang mô hình ô tô máy Moore, số lượng trạng thái trong có thể tăng lên. Ở ví dụ trên đây, số trạng thái trong của S_1 là ba, còn số trạng thái trong của S_0 tương đương là năm. Mặt khác, từ thí dụ biến đổi từ ô tô máy Moore sang ô tô máy Milly tương đương, số lượng trạng thái trong của hai ô tô máy là như nhau. Như vậy, nếu ô tô máy Moore (hình 8.19b) lại được biến đổi ngược trở lại thành ô tô máy Milly tương đương thì ô tô máy Milly mới này sẽ có năm trạng thái. Nhưng dù có năm trạng thái, ô tô máy Milly năm trạng thái này vẫn thực hiện cùng một chức năng như ô tô máy Milly ba trạng thái đã cho ban đầu. Đương nhiên chúng ta thấy sẽ xuất hiện bài toán tìm dạng ô tô máy tương đương có số lượng trạng thái tối thiểu. Nhiệm vụ này được xem xét trong chương tiếp theo: Tổng hợp ô tô máy hữu hạn.

8.5. Ô TÔ MÁY ĐỒNG BỘ VÀ KHÔNG ĐỒNG BỘ

Về mặt cách thức thực hiện chức năng, các ô tô máy hữu hạn chia thành hai loại:

- Ô tô máy làm việc theo chế độ không đồng bộ;
- Ô tô máy làm việc theo chế độ đồng bộ.

Trong chế độ đồng bộ lại chia ra thành đồng bộ ngoài và đồng bộ trong.

Ô tô máy làm việc theo chế độ đồng bộ ngoài là loại mà hoạt động của nó luôn được nhịp bởi chuỗi xung nhịp có độ rộng xung rất hẹp và có độ ổn định tần số lặp lại của xung rất cao. Trong trường hợp này, dưới tác động của các xung đồng bộ, các phần tử nhớ được nhịp đồng thời bởi cùng một nguồn xung đồng bộ và do đó khi ô tô máy thay đổi trạng thái của mình, không xảy ra tranh chấp trạng thái. Điều này cho phép ta khi dùng đồng bộ ngoài không nhất thiết phải mã hoá trạng thái trong bằng các loại mã chống tranh chấp trạng thái như mã hoá kế cận chẳng hạn. Hơn thế nữa, có thể áp dụng các biện pháp mã hoá làm cho mạch đơn giản nhất. Tất nhiên vì ô tô máy làm việc theo nhịp, nên tốc độ làm việc của ô tô máy đồng bộ ngoài bị hạn chế bởi tần số xung nhịp và chậm hơn tốc độ làm việc của ô tô máy không đồng bộ. Ngoài ra, còn cần phải có nguồn tín hiệu đồng bộ có độ chính xác cao nên thiết bị trở nên phức tạp thêm.

Các ô tô máy làm việc ở chế độ không đồng bộ thực hiện chức năng của mình vào bất cứ lúc nào có thông tin hữu ích đầu vào, không cần chờ xung nhịp. Do đó tốc độ làm

việc của ôtomát không đồng bộ cao hơn so với tốc độ làm việc của ôtomát đồng bộ. Song do các phần tử nhớ không được nhịp nên sự thay đổi trạng thái của chúng khó có thể xảy ra đồng thời. Đó chính là nguyên nhân gây ra hiện tượng tranh chấp trạng thái nguy hiểm, dẫn tới mạch làm việc không đúng chức năng như đã thiết kế.

Để đi sâu tìm hiểu kỹ hơn hoạt động của ôtomát đồng bộ và không đồng bộ, chúng ta cần làm rõ khái niệm “trạng thái bền” và “trạng thái không bền” của ôtomát.

Trạng thái A_s của ôtomát gọi là bền đối với trạng thái vào X_i , nếu thoả mãn điều kiện:

$$\varphi(A_s, X_i) = A_s$$

Nghĩa là xuất phát từ trạng thái trong A_s , dưới tác động của trạng thái vào X_i , ôtomát vẫn ở trạng thái A_s , không thay đổi trạng thái của mình.

Thí dụ, ôtomát cho trên hình 8.20 có các trạng thái bền sau đây:

- Trạng thái A_0 là trạng thái bền đối với trạng thái vào X_1 .
- Trạng thái A_1 là trạng thái bền đối với trạng thái vào X_2 và X_3 .
- Trạng thái A_2 là trạng thái bền đối với trạng thái vào X_2 .

Để hiểu rõ hoạt động của ôtomát này khi tín hiệu vào có dạng thế và khi tín hiệu vào có dạng xung, ta xét hoạt động của nó đối với hai trường hợp sau đây.

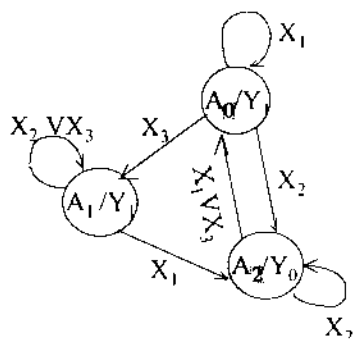
8.5.1. Trường hợp trạng thái vào có dạng thế

Giả sử trạng thái ban đầu của ôtomát là A_0 . Khi xuất hiện trạng thái vào X_2 , theo đồ hình trạng thái ở hình 8.20, từ trạng thái A_0 , dưới tác động của X_2 , ôtomát chuyển sang trạng thái A_2 . Ta có:

$$A_0 \xrightarrow{X_2} A_2$$

Vì trạng thái A_2 của ôtomát là bền đối với trạng thái vào X_2 , nên chừng nào còn tồn tại X_2 thì ôtomát vẫn ở A_2 .

Nếu bây giờ xuất hiện trạng thái vào X_3 thì theo đồ hình, ôtomát sẽ chuyển từ trạng thái A_2 sang A_0 . Nhưng vì A_0 không phải là trạng thái bền đối với trạng thái vào X_3 , nên khi đã sang A_0 rồi mà vẫn tồn tại trạng thái vào X_3 thì theo đồ hình ôtomát lại tiếp tục chuyển sang trạng thái A_1 và sẽ ở lại A_1 chừng nào còn tồn tại X_3 , vì A_1 là trạng thái bền đối với trạng thái vào X_3 . Ta có các bước chuyển trạng thái sau đây:



Hình 8.20. Ôtomát Milly

$$\begin{array}{ccccccc} A_0 & \xrightarrow{X_2} & A_2 & \xrightarrow{X_1} & A_0 & \xrightarrow{X_1} & A_1 \\ Y_1 & & Y_0 & & (Y_1) & & Y_1 \end{array}$$

Ta nói, dưới tác động của trạng thái vào X_3 , ô tô máy chỉ lướt qua trạng thái A_0 . Trên thực tế trạng thái Y_1 tại A_0 không gây được tác động vì thời gian tồn tại của nó quá nhỏ.

Từ thí dụ trên cho thấy rằng, trong trường hợp này thời gian tồn tại của trạng thái vào có ảnh hưởng quan trọng tới hoạt động chức năng của ô tô máy và thực tế thời gian tồn tại này không bị hạn chế. Nhưng vấn đề là ở chỗ, với thời gian tồn tại khác nhau của các trạng thái vào thì ô tô máy lại hoạt động khác nhau và gây nên những tác động khác nhau.

Một câu hỏi đặt ra là thời gian tồn tại của trạng thái vào phải như thế nào để ô tô máy làm việc đúng chức năng đã định mà không làm mất thông tin?

Giả sử thời gian tối thiểu để ô tô máy đủ thực hiện một bước chuyển đơn là t_0 . Ta hãy nghiên cứu hoạt động của ô tô máy khi có các chuỗi vào tác động, trong đó trạng thái X_3 có thời gian tồn tại khác nhau, còn X_2 tồn tại lâu dài như sau:

$$\begin{aligned} &+ X_3(t_0), X_2; \\ &+ X_3(2t_0), X_2; \\ &+ X_3(3t_0), X_2; \end{aligned}$$

Giả sử đối với cả ba chuỗi tác động vào, ô tô máy đều xuất phát từ trạng thái A_2 . Kết quả xảy ra với mỗi chuỗi như sau:

+ Với chuỗi $X_3(t_0), X_2$ ta có các bước chuyển sau đây:

$$\begin{array}{ccccc} A_2 & \xrightarrow{X_3(t_0)} & A_0 & \xrightarrow{X_2} & A_2 \\ Y_0 & & (Y_1) & & Y_0 \end{array}$$

Ô tô máy lướt qua trạng thái A_0 và Y_1 không gây được tác động gì.

+ Với chuỗi vào $X_3(2t_0), X_2$, ta có các bước chuyển sau:

$$\begin{array}{ccccccc} A_2 & \xrightarrow{X_3(t_0)} & A_0 & \xrightarrow{X_3(t_0)} & A_1 & \xrightarrow{X_2} & A_1 \\ Y_0 & & (Y_1) & & Y_1 & & Y_1 \end{array}$$

Ô tô máy đủ thời gian để đạt tới trạng thái bền đối với nó và tiếp theo nó là trạng thái X_2 cũng bền đối với trạng thái A_1 , cho nên kết quả của X_3 và của X_2 tiếp theo đều được ghi nhận.

+ Với chuỗi vào $X_3(3t_0), X_2$ ta có các bước chuyển sau đây:

$$\begin{array}{ccccccccc} A_2 & \xrightarrow{X_3(t_0)} & A_0 & \xrightarrow{X_3(t_0)} & A_1 & \xrightarrow{X_3(t_0)} & A_1 & \xrightarrow{X_2} & A_1 \\ Y_0 & & (Y_1) & & Y_1 & & Y_1 & & Y_1 \end{array}$$

Trong trường hợp này, thời gian tồn tại của trạng thái vào X_3 đủ để ô tô máy đạt tới và ổn định ở trạng thái bền đối với X_3 . Ô tô máy vẫn lướt qua A_0 nên Y_1 tại A_0 không

gây nên tác động gì. Tuy vậy các Y_i tại trạng thái A_i là kết quả tác động của X_3 và X_2 tiếp theo.

Kết luận:

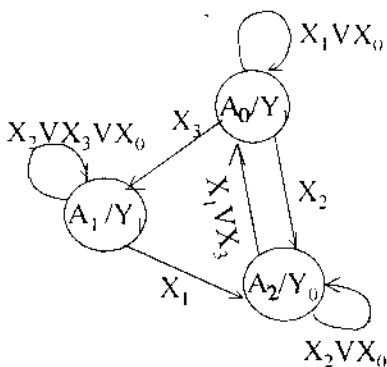
Yêu cầu cơ bản khi ô tômát làm việc với tín hiệu thế là thời gian tồn tại của các trạng thái vào phải đủ lớn để ít nhất ô tômát đạt tới trạng thái trong bền đối với trạng thái vào đó. Chỉ có như vậy tác dụng của trạng thái vào mới được ghi nhận và không bị mất thông tin. Chế độ làm việc này gọi là chế độ không đồng bộ.

8.5.2. Trường hợp trạng thái vào có dạng xung

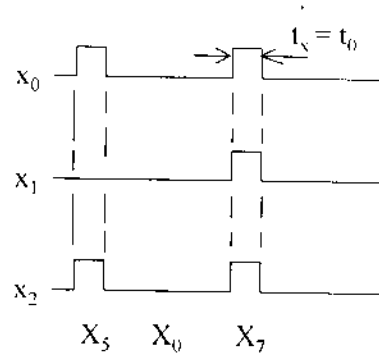
Trong trường hợp này ta giả thiết rằng thời gian tồn tại của trạng thái vào t_0 chỉ đủ để ô tômát thực hiện xong một chuyển biến đơn. Như vậy đối với một trạng thái vào tích cực, ô tômát chỉ có thể thực hiện một bước chuyển biến đơn mà thôi. Nếu nhiều trạng thái vào tích cực tác động liên tiếp thì tất cả các trạng thái mà ô tômát đi qua chỉ được lướt qua rất nhanh và có thể không bao giờ gây nên các tác động ra. Thông tin vào trong trường hợp này sẽ mất tác dụng.

Như vậy, đối với ô tômát làm việc với trạng thái vào là xung, bắt buộc phải có một trạng thái dạng thế mà tất cả trạng thái của ô tômát đều bền đối với nó. Ta ký hiệu trạng thái này là trạng thái X_0 . Thông thường trạng thái X_0 là trạng thái mà tất cả các biến vào đều bằng 0 hoặc đều bằng 1. Khi đó quy tắc tồn tại xen kẽ của trạng thái thụ động X_0 và các trạng thái tích cực khác là bắt buộc.

Tính tối sự cần thiết phải có trạng thái thụ động, đối với ô tômát ở hình 8.20 phải có thêm trạng thái X_0 , loại thế và mọi trạng thái đều phải bền đối với nó. Từ đó ta có ô tômát như ở hình 8.21.



Hình 8.21. Ô tômát làm việc với trạng thái vào tích cực dạng xung



Hình 8.22. Chuỗi trạng thái vào X_5 , X_0 , X_7

Trên hình 8.22 mô tả một chuỗi trạng thái vào hợp thức đối với ô tô máy làm việc với trạng thái vào dạng xung. Hai trạng thái tích cực là trạng thái 5 và trạng thái 7 tương ứng với giá trị các biến vào x_0, x_1, x_2 là 101 và 111 và trạng thái X_0 tương ứng với giá trị các biến vào là 000, xen kẽ giữa các trạng thái 5 và 7.

Đối với ô tô máy ở hình 8.21, ứng với chuỗi trạng thái vào:

$$X_1 X_0 X_1 X_0 X_3 X_0 X_1$$

và trạng thái trong xuất phát là trạng thái A_2 , ta sẽ có các bước chuyển sau đây:

$$\begin{array}{ccccccc} X_1 & X_0 & X_1 & X_0 & X_3 & X_1 & \\ A_2 \rightarrow A_0 \rightarrow A_0 \rightarrow A_0 \rightarrow A_0 \rightarrow A_1 \rightarrow A_2 \\ Y_0 & Y_1 & Y_1 & Y_1 & Y_1 & Y_1 & Y_0 \end{array}$$

Trong trường hợp này mọi trạng thái vào tích cực đều gây nên các tác động tương ứng. Các trạng thái thụ động X_0 xen giữa các trạng thái tích cực chỉ duy trì thêm giá trị đã đạt được của các trạng thái vào tích cực. Trong trường hợp là ô tô máy loại Milly, tác động ra hoàn toàn phụ thuộc vào từng trạng thái vào.

Kết luận chung

Qua hai trường hợp trên đây ta thấy, khi trạng thái vào là loại thế, ô tô máy có thể thực hiện các bước chuyển đơn hoặc thực hiện nhiều bước chuyển đối với cùng một trạng thái vào. Thời gian khi ô tô máy thực hiện nhiều bước chuyển đối với cùng một trạng thái vào là khác nhau đối với các trạng thái vào có thời gian tồn tại khác nhau. Ô tô máy làm việc theo cách này gọi là ô tô máy không đồng bộ.

Nếu trạng thái vào là loại xung, ô tô máy chỉ thực hiện các bước chuyển đơn đối với mỗi trạng thái vào. Trong trường hợp này thời gian đối với mỗi bước chuyển là như nhau. Những ô tô máy này gọi là các ô tô máy đồng bộ, nói một cách chính xác hơn đó là các ô tô máy đồng bộ trong. Nó khác với các ô tô máy đồng bộ ngoài chỉ ở chỗ, đối với ô tô máy đồng bộ ngoài, việc "xung hoá" các trạng thái vào là do xung nhịp tạo ra. Vì thế mọi ô tô máy làm việc với chế độ đồng bộ ngoài bắt buộc phải có chân đồng bộ "CLOCK". Trong chế độ đồng bộ ngoài, mạch sẽ làm việc theo chế độ đồng bộ, nhưng các trạng thái vào không nhất thiết phải ở dạng xung. Tốc độ làm việc của hệ thống và sự phối hợp về tốc độ giữa tốc độ xuất hiện các trạng thái vào và xung nhịp là tùy thuộc vào từng hệ thống. Ngoài ra, khi trạng thái vào là loại xung thì dễ xảy ra mất thông tin, nếu tín hiệu vào không đồng bộ với xung nhịp.

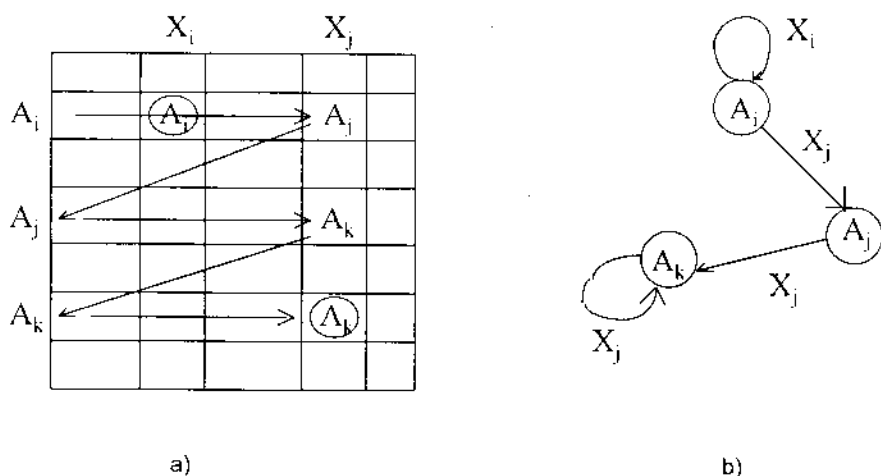
Ta sẽ xét kỹ hơn hoạt động của hai loại ô tô máy này.

a) Đối với ô tô máy không đồng bộ

Về mặt các biến vào, yêu cầu cơ bản đối với chúng là không một biến nào được thay đổi giá trị của mình chừng nào ô tô máy chưa thiết lập xong trạng thái bên của

mình đối với trạng thái vào hiện hành. Như vậy, đối với mỗi trạng thái vào, ô tômát không đồng bộ phải có ít nhất một trạng thái bền đối với nó. Quá trình chuyển từ một trạng thái bền này sang một trạng thái bền khác có thể phải lướt qua một số trạng thái không bền. Điều này sẽ gây nên các phản ứng tác động phụ ở đầu ra của ô tômát mà nhiệm vụ của người thiết kế là phải tìm biện pháp khử bỏ các phản ứng phụ này.

Thật vậy, giả sử trạng thái A_i bền vững đối với trạng thái vào X_i , còn A_k bền đối với trạng thái vào X_i thì trong bảng chuyển đổi trạng thái, tại ô là giao của cột X_i và hàng A_i ta ghi trạng thái mới là A_i , còn tại ô là giao của cột X_i với hàng A_k ta ghi trạng thái mới A_k như trên hình 8.23a. Các trạng thái bền ta đánh dấu bằng một khoanh tròn.



Hình 8.23. Hoạt động khi A_i và A_j không bền đối với X_i , nhưng A_k bền đối với X_i .

Hoạt động của ô tômát khi đó được thể hiện trên hình 8.23b. Sự chuyển trạng thái của ô tômát xảy ra theo thứ tự sau:

$$A_i \xrightarrow{X_i} A_i \xrightarrow{X_j} A_k \xrightarrow{X_i} A_k$$

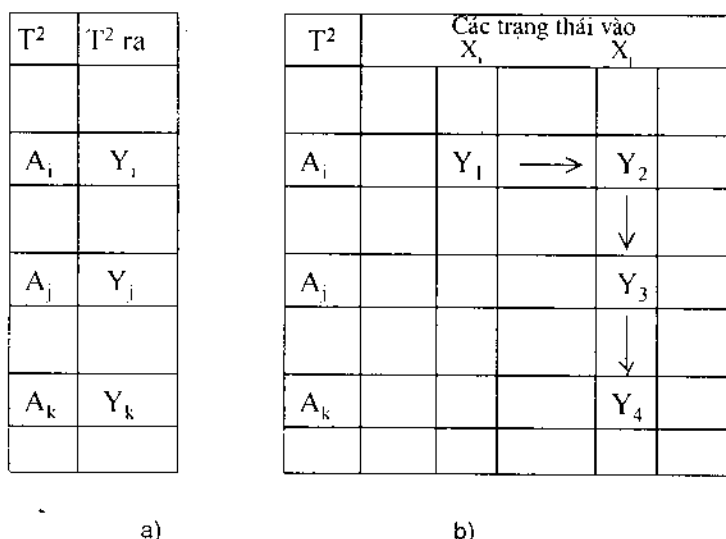
Trong trường hợp này ô tômát đang ở trạng thái bền A_i , dưới tác động của trạng thái vào A_j nó chuyển sang trạng thái bền A_k bằng cách lướt qua trạng thái không bền A_j . Số lượng các trạng thái không bền cần lướt qua trên thực tế có thể nhiều hơn và số lượng khác nhau đối với các chuyển trạng thái khác nhau. Mỗi lần lướt qua một trạng thái không bền, ô tômát lại sinh ra một phản ứng phụ tại đầu ra. Giả sử ô tômát đang xét là ô tômát loại Moore hoặc là ô tômát loại Milly, khi đó chuỗi ra gồm các phản ứng chính và các phản ứng phụ cho trên hình 8.24a hoặc hình 8.24b tương ứng. Chuỗi các phản ứng ra sẽ là:

$$Y_i \rightarrow Y_j \rightarrow Y_k; \text{ Đối với ô tômát Moore;}$$

$$Y_1 \rightarrow Y_2 \rightarrow Y_3 \rightarrow Y_4; \text{ Đối với ô tômát Milly;}$$

Trong trường hợp là ô tô máy Moore, đúng ra chỉ cần các trạng thái: Y_i và Y_k . Trạng thái ra Y_j là phản ứng phụ, cần khử bỏ.

Trong trường hợp là ô tô máy Milly, các trạng thái ra chính là Y_1 và Y_4 . Các trạng thái Y_2 và Y_3 là các phản ứng ra phụ, cần phải có biện pháp khử bỏ hoặc làm cho nó không gây ảnh hưởng trên thực tế.



Hình 8.24. Chuỗi ra của ô tô máy Moore và ô tô máy Milly

b) Đối với ô tô máy đồng bộ

Khi làm việc, mỗi lần có trạng thái vào tác động, ô tô máy đồng bộ chỉ thực hiện một bước chuyển trạng thái đơn. Có hai phương thức để thực hiện chế độ công tác này:

+ Dựa trên đặc trưng của trạng thái vào: Trong trường hợp này, trong số các biến vào x_i của ô tô máy, ít nhất phải có một biến tồn tại dưới dạng xung, tức là thời gian tồn tại của các xung chỉ đủ để thực hiện một chuyển biến đơn. Như vậy mọi trạng thái tích cực đều phải là xung. Trong trường hợp này, để tác động ra có thể được thể hiện, nhất thiết phải có một trạng thái vào mà mọi trạng thái trong đều bên đối với nó. Trạng thái mang tính chất một trạng thái dạng thế, ký hiệu là X_0 . Đó là trạng thái vào dạng thế duy nhất và nó được xen giữa bất kỳ hai trạng thái vào xung tích cực nào. Chế độ đồng bộ này gọi là chế độ đồng bộ trong.

+ Trong trường hợp tất cả các biến vào đều có dạng thế, tức là không tồn tại các trạng thái vào dạng xung, có thể đưa ô tô máy về chế độ làm việc xung bằng cách đưa vào xung đồng bộ. Về phương diện tín hiệu, xung đồng bộ hoàn toàn có thể coi là một tín hiệu hữu ích như các tín hiệu vào khác. Chế độ này gọi là chế độ đồng bộ ngoài. Đặc điểm của ô tô máy làm việc ở chế độ đồng bộ là:

- Không có khái niệm về trạng thái bền và không bền.
- Mọi chuyển biến trạng thái chỉ là chuyển biến đơn.
- Không tồn tại các trạng thái ra phụ.

Trên hình 8.25a và b là các thí dụ về bảng chuyển đổi trạng thái và bảng ra của ôtomát Moore và ôtomát Milly làm việc ở chế độ đồng bộ.

Với ôtomát Milly, trạng thái ra phụ thuộc “một cách xung” vào các trạng thái vào và các trạng thái trong. Đối với ôtomát Moore, trạng thái ra phụ thuộc “một cách thê” vào các trạng thái trong của ôtomát. Hai loại ôtomát này chỉ khác nhau về dạng tồn tại của trạng thái ra.

Ra	T^2	Các trạng thái vào			
		X_0		X_1	
Y_0	A_i	A_i		A_j	
Y_k	A_j	A_j		A_k	

T^2	Các trạng thái vào			
	X_0		X_1	
Λ_i	A_i, Y_0		A_j, Y_j	
Λ_j	A_j, Y_0		Λ_k, Y_k	

a) Ôtomát Moore

b) Ôtomát Milly

Hình 8.25. Bảng ra của ôtomát Moore và Milly ở chế độ đồng bộ

8.6. MÔ HÌNH KỸ THUẬT CỦA ÔTÔMÁT HỮU HẠN

Trong mục 8.1 chúng ta đã đề cập tới mô hình toán học của ôtomát hữu hạn. Để tiếp cận tới kỹ thuật, đối với mỗi ôtomát cụ thể không những phải mô tả cấu trúc phần cứng, tức là số lượng biến vào n , số lượng biến ra m , và số lượng biến trong p , mà còn phải mô tả một cách chi tiết hai hàm chuyển đổi trạng thái φ và hàm ra ψ . Việc xác định các đầu vào, đầu ra và các biến trong không phải là vấn đề phức tạp. Trái lại, khâu quan trọng nhất, khó khăn và phức tạp nhất lại là việc mô tả hàm chuyển đổi trạng thái và hàm ra. Từ các mô tả hai hàm này chúng ta dễ dàng hình thành mô hình cấu trúc của ôtomát.

Người ta chứng minh được rằng, bất kỳ ô tômát hữu hạn nào có cấu trúc tổng quát theo mô hình toán học như ở hình 8.26a, đều có thể biểu diễn dưới dạng mô hình kỹ thuật gồm hai phần (hình 8.26b):

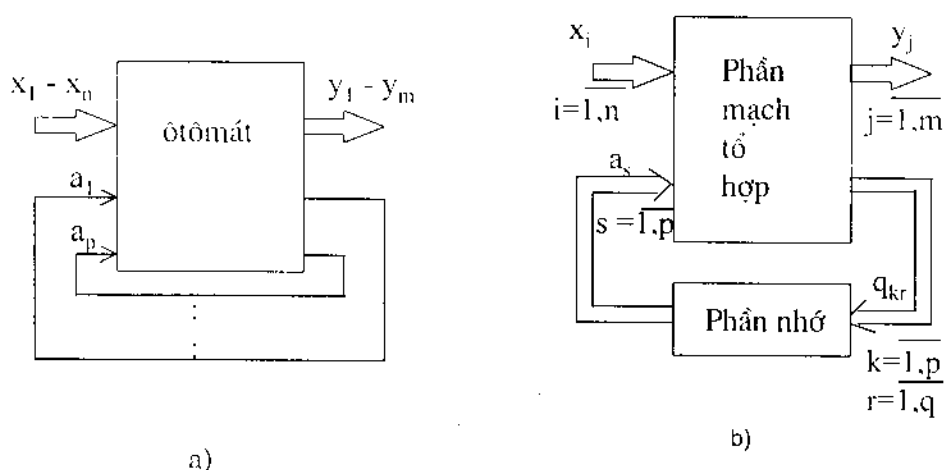
- + Phần mạch tổ hợp, gồm toàn bộ phần mạch không có nhớ của ô tômát.
- + Phần nhớ, gồm toàn bộ các phần tử nhớ của ô tômát. Số lượng các phần tử nhớ luôn luôn bằng số lượng các biến trong.

Phần tổ hợp có $(n + p)$ đầu vào, gồm các đầu vào thông tin của ô tômát và các đầu vào là các biến trong của nó và có số đầu ra bằng tổng các đầu ra của ô tômát và các đầu ra là các tín hiệu kích thích cho các phần tử nhớ của phần nhớ.

Phần nhớ gồm p phần tử nhớ cơ bản, mỗi phần tử nhớ cơ bản thực hiện một biến trong a_s . Các phần tử nhớ cơ bản là một mạch có phản hồi (một ô tômát hữu hạn loại Moore) có một đầu ra a_s có thể nhớ được một bit, có thể có một hay hai đầu vào tín hiệu và các đầu vào điều khiển khác. Ta ký hiệu số lượng đầu vào của mỗi phần tử nhớ cơ bản là q . Đầu ra a_s của phần tử nhớ cơ bản có khả năng thay đổi giá trị của mình theo các phương án sau:

0 sang 0; 0 sang 1; 1 sang 0; 1 sang 1

tùy theo giá trị ban đầu của a bằng 0 hay 1 và trạng thái tác động vào của các biến và q_{kr} của nó. Chúng ta sẽ dành riêng một chương để nghiên cứu kỹ về các phần tử nhớ cơ bản này.



Hình 8.26. a) Mô hình tổng quát của ô tômát;
b) Mô hình kỹ thuật.

Từ mô hình cấu trúc của ô tômát, chúng ta dễ dàng thấy mối liên hệ giữa các hàm chuyển đổi trạng thái và hàm ra của ô tômát với hàm thay đổi trạng thái của các phần tử nhớ cơ bản như sau:

$$\left. \begin{aligned} a_{sm} &= \varphi(x_i, a_s) \\ y_j &= \psi(x_i, a_s) \end{aligned} \right\} \quad (8.1)$$

Mặt khác, đầu ra a_s của các phần tử nhớ cơ bản lại là hàm của các đầu vào kích thích của chúng là các q_{kr} theo hàm ξ nào đó. Ta có:

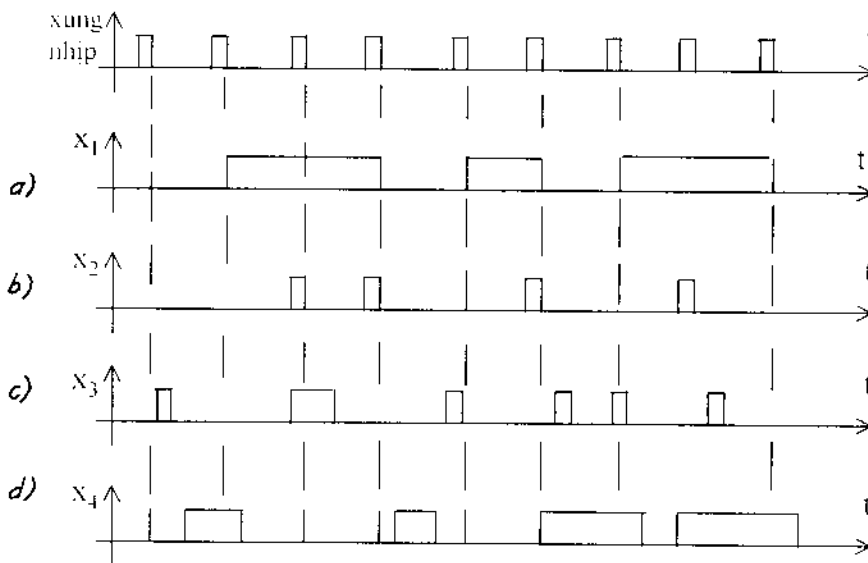
$$a_s = \xi(q_{kr}) \quad (8.2)$$

Nhưng q_{kr} lại là đầu ra của phần mạch tổ hợp và do đó là hàm của x_i và a_s . Tức là:

$$q_{kr} = f_{kr}(x_i, a_s) \quad (8.3)$$

Các công thức toán học (8.1), (8.2) và (8.3) là phản ánh mô hình kỹ thuật của ôtomát.

Khi xây dựng phần tổ hợp phải xây dựng một hệ hàm gồm các hàm ra y_i và các hàm kích thích q_{kr} , mỗi hàm có $(n + p)$ biến.



Hình 8.27. Tín hiệu vào đồng bộ (a, b) và không đồng bộ (c, d)

Như phần sau sẽ trình bày rõ, phần tử nhớ cơ bản có nhiều loại: Làm việc với tín hiệu vào dạng thế hay dạng xung. Các phần tử nhớ có đầu vào xung nhịp hay không có đầu vào xung nhịp. Bản thân tín hiệu đầu vào của ôtomát có đồng bộ với xung nhịp của nó hay không... Trong hình 8.27 cho ta thí dụ về tín hiệu vào x_1 và x_2 được đồng bộ đối với tín hiệu nhịp, còn tín hiệu vào x_3 và x_4 không xuất hiện đồng bộ với xung nhịp.

Trong phần tổng hợp và phân tích thiết bị số có nhớ tiếp theo, chúng ta sẽ đi sâu nghiên cứu các loại thiết bị số điển hình sau đây:

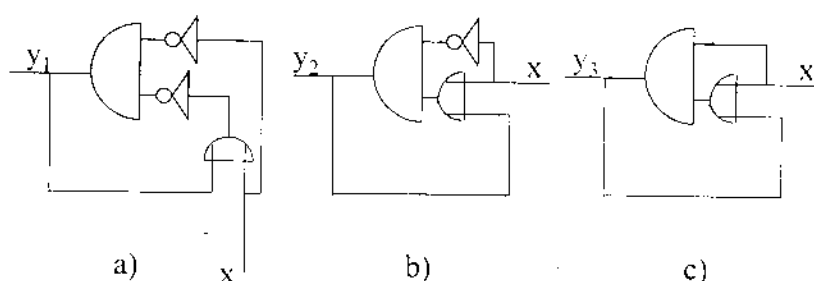
+ Làm việc với tín hiệu đồng bộ: tín hiệu vào có dạng thế hoặc xung.

+ Làm việc không có tín hiệu đồng bộ; tín hiệu vào có dạng thế hoặc xung.

Các phương pháp phân tích và tổng hợp các loại thiết bị số này là các phương pháp tổng quát, nhưng rất cụ thể, có thể áp dụng cho tất cả các loại thiết bị số thường gặp.

BÀI TẬP CHƯƠNG 8

- Phân tích phản ứng ra của các mạch ở hình 8.28a, b, c. Cho kết luận về hoạt động của các mạch logic có phản hồi.



Hình 8.28

- Cho bảng chuyển đổi trạng thái và bảng ra của ô tômát Milly và Moore như ở hình 8.29a và b. Hãy:
 - Xây dựng đồ hình chuyển đổi trạng thái của các ô tômát này.
 - Lập ma trận chuyển đổi trạng thái và ma trận ra của chúng.

A	X_0	X_1
A_0	A_0/Y_2	A_1/Y_2
A_1	A_1/Y_1	A_3/Y_2
A_2	A_3/Y_2	A_2/Y_1
A_3	A_3/Y_2	A_0/Y_1

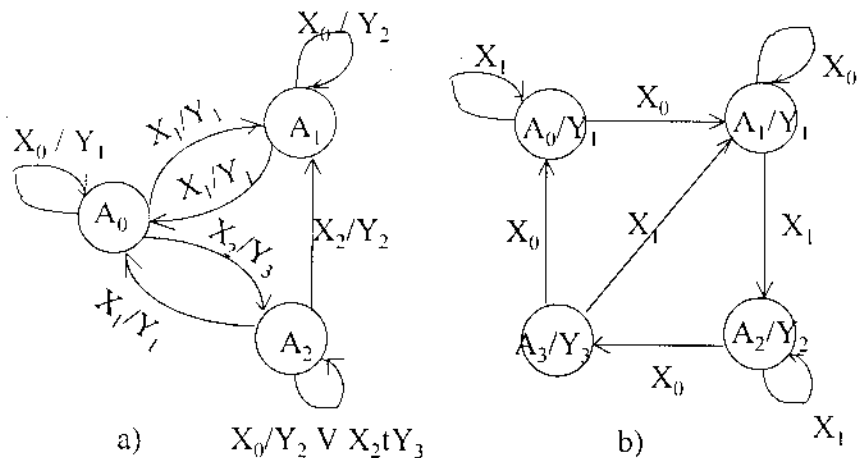
a)

Y	A	X_0	X_1	X_2	X_3
Y_2	A_0	A_0	A_1	A_2	A_1
Y_1	A_1	A_1	A_2	A_1	A_0
Y_2	A_2	A_2	A_0	A_1	A_0
Y_3	A_3	A_3	A_0	A_1	A_3

b)

Hình 8.29. Ô tômát Milly và Moore

3. Cho đồ hình trạng thái của ôtomát Milly và Moore trên hình 8.31a và b. Hãy:
 - a) Lập bảng chuyển đổi trạng thái và bảng ra của chúng;
 - b) Lập ma trận chuyển đổi trạng thái và ma trận ra của chúng.
4. Hãy chuyển ôtomát Milly (hình 8.30a) sang ôtomát dạng Moore và ôtomát Moore (hình 8.30b) sang ôtomát dạng Milly.
5. Mã hoá và sau đó chuyển các ôtomát ở hình 8.30a và b sang dạng:
 - a) Bảng chuyển đổi trạng thái và bảng ra đã mã hoá.
 - b) Đồ hình trạng thái đã mã hoá.



Hình 8.30. Đồ hình của ôtomát Milly và Moore

6. Hãy lập bảng chuyển đổi trạng thái và bảng ra, đồ hình trạng thái của bộ đếm có cơ số đếm thay đổi phụ thuộc vào tín hiệu điều khiển S: Nếu $S = 0$ thì hệ số đếm là 2 (Ôtomát sẽ chuyển trạng thái theo chu trình 0, 1, 2, 0...). Nếu $S = 1$ thì hệ số đếm là 3 (Chu trình trạng thái lúc đó là 0, 1, 2, 3, 0...).
7. Lập bảng chuyển đổi trạng thái của ôtomát có 5 trạng thái trong và ba trạng thái vào X_0, X_1, X_2 . Năm trạng thái là A_0, A_1, A_2, A_3 và A_4 , với điều kiện:
 - Khi X_0 tác động thì không làm thay đổi trạng thái của bất kỳ trạng thái nào trong năm trạng thái trong của ôtomát đã cho.
 - Khi X_1 tác động thì ở bất kỳ trạng thái nào ôtomát cũng chuyển về A_0 .
 - Khi X_2 tác động thì thực hiện đếm với cơ số là 5.
8. Lập đồ hình trạng thái cho nhiệm vụ 7.
9. Mã hoá các biến vào, các biến trong và lập bảng chuyển đổi trạng thái và đồ hình đã mã hoá cho nhiệm vụ 7.
10. Cho hai ôtomát dưới đây (hình 8.31a và b) Hãy cho biết ôtomát nào là ôtomát đồng bộ và ôtomát nào là ôtomát không đồng bộ? Tại sao?

Y	A	X ₀	X ₁
Y ₁	A ₀	A ₀	A ₁
Y ₂	A ₁	A ₁	A ₂
Y ₁	A ₂	A ₂	A ₃
Y ₁	A ₃	A ₃	A ₀

a)

A	X ₀	X ₁	X ₂	X ₃
A ₀	-	A ₁ / Y ₁	A ₀ / -	A ₀ / Y ₀
A ₁	A ₁ / Y ₁	A ₂ / Y ₂	-	A ₀ / Y ₀
A ₂	A ₂ / Y ₁	A ₂ / Y ₀	A ₁ / Y ₁	A ₀ / Y ₀

b)

Hình 8.31

Chương 9

TỔNG HỢP THIẾT BỊ SỐ CÓ NHỚ

9.1. THỦ TỤC TỔNG HỢP CẤU TRÚC THIẾT BỊ SỐ CÓ NHỚ

Như chúng ta đã biết, thiết bị số có nhớ nói chung và các thiết bị số có nhớ - các ô tô măt hữu hạn nói riêng đã thâm nhập vào hầu hết các lĩnh vực điện tử trong thông tin liên lạc, trong kỹ thuật tính toán, điều khiển, đo lường và trong hầu hết các thiết bị chuyên dùng khác.

Bài toán tổng quát cho việc thiết kế một ô tô măt hữu hạn thường được đặt ra như sau: Chức năng của thiết bị được phát biểu bằng lời, yêu cầu thiết kế một thiết bị hoạt động theo chức năng đã phát biểu và đáp ứng một số yêu cầu kỹ thuật nhất định.

Thông thường việc phát biểu chức năng của thiết bị bằng lời khó có thể chặt chẽ và khó tránh khỏi có những sai sót, mâu thuẫn. Những phát biểu sơ bộ ban đầu như vậy thường chỉ đưa ra những yêu cầu cơ bản cần thực hiện. Để chính xác hoá việc mô tả chức năng, người thiết kế phải phát biểu chức năng của thiết bị và ghi nó dưới dạng chuẩn như bảng chuyển đổi trạng thái và bảng ra hay đồ hình trạng thái ... Mặt khác, ngày nay do sự phát triển mạnh mẽ của khoa học, kỹ thuật và công nghệ chế tạo các vi mạch mà cùng một chức năng, thiết bị có thể được xây dựng theo nhiều phương án khác nhau, không những về kết cấu cụ thể mà còn khác nhau về các tính năng quan trọng như độ tin cậy, tốc độ làm việc...

Trước đây, khi công nghệ vi mạch còn hạn chế, người thiết kế bắt buộc phải thiết kế thiết bị trên cơ sở các linh kiện điện tử riêng lẻ. Đó là công việc hết sức phức tạp, nặng nề. Việc thiết kế không chỉ dừng ở chức năng logic mà còn phải giải quyết cả vấn đề chế độ công tác của các linh kiện và phối hợp trở kháng, phối hợp tín hiệu...

Sự ra đời của các vi mạch điện tử, chuẩn hoá các chức năng cơ bản về mọi chỉ tiêu kỹ thuật đã làm cho việc thiết kế được dễ dàng hơn. Tuy vậy, ở giai đoạn đầu vào những năm 1960, người ta vẫn phải thiết kế các thiết bị, kể cả các thiết bị phức tạp, trên cơ sở các vi mạch tích hợp cỡ nhỏ, với số lượng đầu vào thông thường từ 1 đến 8. Do đó việc thiết kế tuy đã chuẩn hoá nhưng vẫn phức tạp, đặc biệt đối với các hệ thống lớn. Sự ra đời của các vi mạch có độ tích hợp trung bình như các bộ đếm, các bộ ghi, chốt, so sánh và các mảng logic khả lập trình không những có chức năng lớn hơn mà còn có thể

thực hiện được các chức năng khác nhau trong một phạm vi nhất định nhờ có các đầu điều khiển. Việc thiết kế các thiết bị số trên cơ sở các vi mạch cỡ nhỏ kết hợp với các vi mạch cỡ trung bình đã thuận lợi hơn nhiều. Về phương pháp, thủ tục thiết kế, công cụ phục vụ cho thiết kế cũng được nghiên cứu phát triển nhiều. Song chức năng của các thiết bị được thiết kế ra chưa có tính mềm dẻo, khó thay đổi được.

Vào những năm 70 và 80 của thế kỷ 20, công nghệ chế tạo vi mạch đã có những tiến bộ vượt bậc. Các vi mạch cỡ lớn và cực lớn ra đời với chức năng của mỗi vi mạch rất phức tạp, đa năng và có thể lập trình để thực hiện từng loại chức năng được. Mặt khác, các hệ thống trang thiết bị ngày càng phức tạp, tính năng có thể đòi hỏi phải thay đổi trong quá trình làm việc. Các vi mạch cỡ lớn và rất lớn, các môđun chức năng định sẵn là cơ sở thuận tiện cho việc thiết kế các hệ thống lớn làm việc theo chương trình. Các phương pháp thiết kế và tổng hợp mới cũng được nghiên cứu và phát triển. Đó là các phương pháp phân tích và tổng hợp các hệ thống số phức tạp, làm việc theo chương trình trên cơ sở các phần tử vi mạch cũng làm việc theo chương trình. Các hệ thống thường bao gồm trung tâm xử lý thông tin, có các bộ xử lý, lưu trữ thông tin và các bộ nối ghép hệ xử lý với các thiết bị ngoại vi khác nhau. Giai đoạn này làm xuất hiện nhiều vấn đề lý thuyết lớn về phân tích và tổng hợp hệ thống để phù hợp với sự phát triển và đòi hỏi của thực tiễn kỹ thuật. Sự xuất hiện của các môi trường tính toán đồng nhất, lý thuyết về mạng thông tin mới... còn đặt ra những yêu cầu mới về ngôn ngữ thiết kế, điều khiển và các phương pháp tổng hợp và phân tích cụ thể.

Tuy lý thuyết tổng hợp thiết bị số phát triển rất nhanh, nhưng những bước cơ bản vẫn là:

1. Trên cơ sở nhiệm vụ thiết kế được phát biểu bằng lời, cần mô tả chức năng của nó bằng một algôrit nào đó, để đảm bảo tính lôgic, loại trừ mọi khả năng không đúng, không đơn trị hay không dẫn tới kết quả...

2. Mô tả algôrit vừa nhận được bằng một phương pháp mô tả chuẩn nào đó, thí dụ, bằng chuyển đổi trạng thái và bảng ra hay đồ hình trạng thái... một lần nữa nhằm tránh xảy ra mâu thuẫn, bỏ sót hay bất định... đồng thời tạo điều kiện sử dụng các công cụ toán học để xử lý về sau.

3. Đối với các thiết bị phức tạp, nếu cần thiết, cần thực hiện phân hoạch, phân chia thiết bị thành từng khối chức năng, thuận tiện cho việc tính toán, thiết kế và thi công lắp ráp trong các công đoạn sau.

Các phần trên đây thuộc phần thiết kế hệ thống. Sau khi đã mô tả thiết bị bằng một ngôn ngữ chuẩn nào đó, ta thực hiện thiết kế lôgic. Phần thiết kế lôgic gồm các bước tiếp theo sau đây:

4. Đơn giản các trạng thái trong của ô tômát. Mặc dù ở phần thiết kế hệ thống đã cố gắng loại trừ các trạng thái thừa. Song trên thực tế thường vẫn nhận được các ô tômát có số trạng thái trong chưa phải là tối thiểu.

5. Mã hoá các trạng thái: vào, ra và trong của ôtomát theo yêu cầu về độ tin cậy chức năng, về độ đơn giản của thiết bị hay về tốc độ làm việc của nó.

6. Xác định mô hình cấu trúc cụ thể của ôtomát, trong đó cần xác định loại phần tử nhớ cơ bản cần dùng.

7. Rút ra các hàm ra y , các hàm kích thích q_{kr} của ôtomát.

8. Tổng hợp phần tổ hợp và phần nhớ. Kết quả của phần tổng hợp logic là chức năng của thiết bị được mô tả tối chức năng của các vi mạch.

Năm bước từ (4) đến (8) hoàn toàn mang tính tổng quát. Đối với mỗi loại ôtomát như: Ôtomát đồng bộ hay không đồng bộ, làm việc theo chương trình hay không, dùng phần tử nhớ loại xung hay thế sẽ có những điểm riêng trong thiết kế cấu trúc logic.

Tiếp theo phần thiết kế cấu trúc là phần thiết kế kỹ thuật và thiết kế công nghệ, gồm các bước sau:

9. Tiến hành xác định các loại vi mạch tổ hợp sẽ sử dụng để thực hiện phần tổ hợp, xuất phát từ yêu cầu về tốc độ, yêu cầu về công suất tín hiệu và khả năng hiện có về các chủng loại vi mạch... Thực hiện phần nối ghép phối hợp về trở kháng và về mức tín hiệu nếu cần. Tiến hành thiết kế công nghệ như thiết kế mạch in, lắp ráp.

10. Lập các hồ sơ kỹ thuật về thiết kế, chế tạo và hướng dẫn sử dụng.

Tất cả các khâu trong quá trình thiết kế trên đây đều có ảnh hưởng tới chất lượng của thiết bị và là các công đoạn khá phức tạp. Ngày nay trong mọi khâu người ta đều có thể sử dụng các công cụ chuyên dụng hay máy tính đa năng để thực hiện tự động hoá thiết kế. Việc dùng máy tính để tính toán thiết kế cho phép xử lý một khối lượng thông tin khá lớn, thuận tiện cho việc lựa chọn phương án và đảm bảo rút ngắn thời gian thiết kế, mau chóng có kết quả, đảm bảo chất lượng và khả năng tối ưu đối với các chỉ tiêu kỹ thuật của máy.

Trong các phần tiếp theo, do hạn chế của tài liệu, chúng ta chỉ xét về mặt nguyên tắc những khâu cơ bản của quá trình thiết kế cấu trúc logic của ôtomát, dựa trên kết quả của phần thiết kế hệ thống. Tức là ta coi chức năng của ôtomát đã được mô tả bằng một ngôn ngữ chuẩn thông dụng nào đó.

9.2. ĐƠN GIẢN TRẠNG THÁI TRONG CỦA ÔTOMÁT

9.2.1. Những khái niệm cơ bản

Trong phần trước chúng ta đã nghiên cứu một số ngôn ngữ mô tả ôtomát chuẩn thường được dùng trong thực tế. Bước tiếp theo sau mô tả chức năng của ôtomát là đơn giản trạng thái trong của nó.

Trước hết chúng ta chỉ tiến hành đơn giản trạng thái trong của ôtomát vì trạng thái vào và trạng thái ra là những yêu cầu tất yếu phải có của nó.

Đơn giản trạng thái trong của ô tômát là một khâu hết sức quan trọng trong tổng hợp cấu trúc logic của ô tômát hữu hạn. Nó liên quan tới độ đơn giản của thiết bị và trực tiếp làm giảm độ phức tạp của thiết bị, giảm công suất tiêu thụ năng lượng và gián tiếp ảnh hưởng tới độ tin cậy của thiết bị.

Về nguyên tắc, việc đơn giản trạng thái trong của ô tômát phải đảm bảo không làm thay đổi chức năng của thiết bị khi ta giảm bớt số lượng trạng thái trong của nó. Nói cách khác, đơn giản trạng thái trong phải dựa trên khái niệm tương đương của các ô tômát.

Định nghĩa 1

Hai ô tômát S_1 và S_2 gọi là tương đương với nhau, nếu với điều kiện ban đầu như nhau, với tác động của chuỗi các trạng thái vào như nhau, phản ứng đầu ra của chúng phải như nhau, còn số trạng thái trong của hai ô tômát không nhất thiết phải như nhau.

Như vậy, đơn giản trạng thái trong của ô tômát thực chất là tìm một ô tômát tương đương với ô tômát đã cho, nhưng có số trạng thái trong ít nhất có thể được.

Nói cách khác, việc đơn giản trạng thái trong của ô tômát là tìm những lớp trạng thái trong sao cho có thể thay chúng bằng một trạng thái đại diện.

Từ những ý tưởng trên đây chúng ta đi đến khái niệm về hai trạng thái tương đương của ô tômát.

Định nghĩa 2

Hai trạng thái tương đương:

Hai trạng thái A_i và A_k của ô tômát gọi là tương đương, nếu ô tômát xuất phát từ chúng, dưới tác động của chuỗi các trạng thái vào như nhau, chuỗi ra phải như nhau.

Những khái niệm này là hoàn toàn chính xác đối với các ô tômát xác định hoàn toàn. Đối với các ô tômát xác định không hoàn toàn thì, vì có những trạng thái không xác định đối với ô tômát nên ta có thể lợi dụng chúng để làm cho việc đơn giản được triệt để hơn. Tuy vậy, trong trường hợp này khái niệm tương đương cần thay thế bằng một khái niệm rộng hơn, linh động hơn, nhưng vẫn đảm bảo yêu cầu về việc thực hiện các chức năng đã đề ra. Đó là khái niệm về trạng thái này thực hiện trạng thái kia và ô tômát này thực hiện ô tômát kia.

Để đi đến khái niệm về ô tômát này thực hiện ô tômát kia, ta đưa ra một số định nghĩa sau đây.

Định nghĩa 3

Chuỗi các trạng thái vào cho phép đối với một trạng thái của ô tômát:

Ta nói chuỗi trạng thái vào $\{X_i\}$ được gọi là cho phép đối với trạng thái A nào đó của ô tô máy S , nếu xuất phát từ A , dưới tác động của chuỗi vào này thì chuỗi trạng thái trong mà ô tô máy đi qua phải không chứa bất kỳ trạng thái bất định nào.

Định nghĩa 4

Trạng thái này thực hiện trạng thái kia:

Trạng thái A_j của ô tô máy S_2 được gọi là thực hiện trạng thái A_i của ô tô máy S_1 nếu bất kỳ chuỗi vào $\{X_i\}$ nào là cho phép đối với A_i của S_1 thì cũng cho phép đối với A_j của S_2 và chuỗi các trạng thái ra tương ứng nhận từ S_1 và S_2 xuất phát từ các trạng thái S_1 và S_2 phải là như nhau hoặc có thể làm cho như nhau bằng cách điền vào các đại lượng bất định những giá trị thích hợp.

Chúng ta dễ dàng thấy quan hệ tương đương là quan hệ hai chiều và bắc cầu, nhưng quan hệ trạng thái này thực hiện trạng thái kia là quan hệ tự nó thực hiện nó và là quan hệ bắc cầu, tức nếu A_j thực hiện A_i và A_i thực hiện A_k thì A_j cũng thực hiện A_k . Nhưng quan hệ này không phải là quan hệ đối xứng, nghĩa là A_j thực hiện A_i thì không thể suy ra một cách đương nhiên là A_i cũng thực hiện A_j , vì trạng thái A_i trong trường hợp này có thể có nhiều chuỗi vào cho phép hơn so với A_j .

Định nghĩa 5

Ô tô máy S_2 thực hiện ô tô máy S_1 :

Ô tô máy S_2 gọi là thực hiện ô tô máy S_1 nếu đối với mỗi trạng thái A_i của S_1 ít nhất phải có một trạng thái A_j của S_2 thực hiện nó. Kết quả hành vi của S_2 phải trùng với kết quả hành vi của S_1 trong mọi lĩnh vực mà S_1 được xác định.

Những khái niệm trên đây thường gắn trực tiếp với các ô tô máy không đồng bộ. Tuy vậy, mọi khái niệm mà chúng ta đã đưa ra trên đây đều có thể áp dụng cho loại ô tô máy đồng bộ, nếu chúng ta coi đầu vào tín hiệu đồng bộ như mọi đầu vào tín hiệu thông tin khác.

9.2.2. Phương pháp nhận biết hai trạng thái tương đương hay thực hiện nhau trên bảng chuyển đổi trạng thái và bảng ra

Nhận biết được các trạng thái tương đương hay trạng thái nào thực hiện trạng thái nào trên bảng chuyển đổi trạng thái và bảng ra của ô tô máy là một vấn đề quan trọng trong đơn giản trạng thái trong của ô tô máy.

Chúng ta biết rằng, chức năng của một ô tô máy có thể được mô tả bằng bảng chuyển đổi trạng thái, đồ hình trạng thái hay ma trận trạng thái... Mọi cách mô tả đều có cách để đơn giản các trạng thái trong của chúng, phù hợp với ngôn ngữ mô tả. Chúng ta sẽ xem xét trường hợp thông dụng nhất, đó là trường hợp ô tô máy được mô tả bằng bảng chuyển đổi trạng thái và bảng ra.

Để tổng quát, ta sẽ xét việc nhận biết các trạng thái tương đương hay thực hiện nhau cũng như hai ô tô máy tính tương đương hay thực hiện nhau đối với loại ô tô máy tính không xác định hoàn toàn. Vì ô tô máy tính xác định hoàn toàn chỉ là một trường hợp cụ thể của ô tô máy tính không xác định hoàn toàn.

Từ khái niệm về ô tô máy tính S_2 thực hiện ô tô máy tính S_1 , giả sử ô tô máy tính S_1 là ô tô máy tính cho ban đầu, xuất phát từ bảng chuyển đổi trạng thái và bảng ra của nó, ta có thể tìm tập hợp các trạng thái tương đương với nhau để rồi thay chúng bằng một trạng thái đại diện. thực hiện chúng thì ta sẽ có một ô tô máy tính S_2 có số trạng thái trong ít hơn, nhưng vẫn đảm bảo thực hiện đầy đủ chức năng của ô tô máy tính S_1 đã cho. Từ đây ta lại có khái niệm "Nhóm (hay lớp) các trạng thái tương đương".

Định nghĩa 6: Nhóm các trạng thái tương đương của ô tô máy tính.

Tập hợp các trạng thái của ô tô máy tính, trong đó chúng tương đương với nhau từng đôi một, gọi là nhóm các trạng thái tương đương.

Tất nhiên, để có ô tô máy tính tương đương S_2 có số trạng thái tối thiểu, chúng ta phải tìm nhóm trạng thái tương đương có chứa một số trạng thái tối đa có thể được. Ta có định nghĩa sau.

Định nghĩa 7: Nhóm các trạng thái tương đương gọi là tối đa, nếu ta thêm bất kỳ trạng thái nào vào thì không còn là nhóm các trạng thái tương đương nữa.

Như đã nói trên đây, trong nhóm các trạng thái tương đương hay nhóm các trạng thái tương đương tối đa, các trạng thái phải tương đương với nhau từng đôi một. Khái niệm tổng quát về hai trạng thái tương đương đã cho tại định nghĩa 2. Để có thể nhận biết hai trạng thái nào là tương đương (thực hiện nhau) trên bảng chuyển đổi trạng thái và bảng ra, ta có thể chỉ ra điều kiện hai trạng thái tương đương như sau.

Điều kiện này gồm hai phần: Đối với trạng thái ra và đối với trạng thái trong.

Hai trạng thái A_i và A_k của ô tô máy tính đã cho là tương đương nếu:

a) Đối với trạng thái ra

Trạng thái ra trong mỗi cột của bảng ra thuộc hai hàng A_i và A_k phải thuộc một trong hai điều kiện sau đây:

- + Như nhau (Giống hệt nhau).
- + Có thể làm cho như nhau (Trường hợp có một dấu bất định trong một hàng).

b) Đối với trạng thái trong

Các trạng thái trong trong mỗi cột thuộc hai hàng A_i và A_k phải thuộc một trong ba điều kiện sau đây:

- + Như nhau (Giống hệt nhau)
- + Có thể làm cho như nhau (Trường hợp có một dấu bất định trong một hàng).
- + Chúng là hai trạng thái tương đương.

Nếu đảm bảo cả hai điều kiện trên đây thì định nghĩa 2 về hai trạng thái tương đương sẽ được thoả mãn. Có nghĩa là xuất phát từ trạng thái trong A_i hoặc A_k , dưới tác động của cùng một chuỗi trạng thái vào, chuỗi trạng thái ra sẽ như nhau.

Trạng thái thay thế hai trạng thái tương đương này là một trạng thái đại diện mà:

+ **Trạng thái ra** của nó trong cột X_i là hội của trạng thái ra của hàng A_i và hàng A_k tại cột đó. Trong đó giá trị bất định tại một hàng nào đó có thể nhận giá trị 0 hoặc 1.

Trong trường hợp ta thay một nhóm tối đa các trạng thái tương đương bằng một trạng thái đại diện thì trạng thái ra của trạng thái đại diện tại cột X_i nào đó cũng được tính một cách tương tự, tức là bằng hội của các trạng thái ra của tất cả các hàng thuộc cột X_i mà trạng thái trong ở đầu hàng thuộc nhóm các trạng thái tương đương này.

+ **Trạng thái trong** tại ô là giao của nó với cột X_i là một trạng thái được xác định như sau:

- Nếu các trạng thái trong tại cột X_i của các hàng có trạng thái trong ở đầu hàng nằm trong nhóm các trạng thái tương đương nào thì trạng thái mới thay thế sẽ là trạng thái thay thế của nhóm các trạng thái tương đương tối đa của nhóm đó.

- Nếu các trạng thái trong tại cột X_i của các hàng này lại thuộc các nhóm trạng thái tương đương tối đa khác nhau thì trạng thái mới thay thế tại ô đang xét có thể là một trong các trạng thái thay thế cho các nhóm tương đương tối đa này.

Thí dụ 9.1: Hãy tìm một số cặp các trạng thái tương đương của ô tô máy ở hình 9.1.

Ô tô máy này có 9 trạng thái trong, đánh số từ 1 đến 9; có 4 trạng thái vào là 00, 01, 10, 11 và có hai trạng thái ra là 0 và 1.

X \ A	$X_1 X_2$				Y
	00	01	10	11	
1	(1)	9	-	2	0
2	3	-	-	(2)	0
3	(3)	4	-	8	0
4	5	(4)	-	-	0
5	(5)	6	-	7	0
6	1	(6)	-	-	1
7	3	-	-	(7)	0
8	3	-	-	(8)	0
9	1	(9)	-	-	0

Hình 9.1. Ô tô máy Moore

Giả sử muốn xét xem hai trạng thái 1 và 3 có là các trạng thái tương đương hay không, ta xét hai tiêu chuẩn về trạng thái ra và trạng thái trong như sau:

- Đối với trạng thái ra:

Trạng thái ra tại hàng 1 và hàng 3 đều bằng 0. Tức các trạng thái này là hoàn toàn như nhau. Điều kiện a) trên đây thoả mãn.

- Đối với trạng thái trong:

Tại cột $X_1 = 00$, các trạng thái trong tương ứng tại hai hàng 1 và 3 là (1) và (3). Hai trạng thái này chính là hai trạng thái ở đầu hàng và đang xét tính tương đương của chúng nên không cần xét gì thêm, chúng được coi là tự thoả mãn.

Tại cột $X_1 = 01$ và $X_1 = 11$, trạng thái trong tương ứng tại hai hàng 1 và 3 là các cặp (9, 4) và (2, 8). Vậy để hai trạng thái 1 và 3 tương đương thì nhất định (4, 9) và (2, 8) phải là các cặp trạng thái tương đương.

Hãy xét cặp (4, 9) trước.

Căn cứ vào bảng chuyển đổi trạng thái và bảng ra của ô tô máy, hai trạng thái 4 và 9 ứng với hai hàng có các trạng thái ra như nhau - đều bằng 0 và các trạng thái trong đều thoả mãn điều kiện tương đương, trừ cặp 1 và 5 phải xét. Khi xét tính tương đương của cặp (1, 5) ta lại phải xét tính tương đương của các cặp trạng thái trong (6, 9) và (2, 7). Từ bảng ra của ô tô máy ta thấy ngay rằng, đối với cặp (6, 9) hai trạng thái ra tương ứng là 1 và 0. Vậy hai trạng thái (6, 9) không tương đương một cách hiển nhiên. Suy ra hai trạng thái 1 và 3 không tương đương. Chúng ta không cần thiết phải xét các cặp trạng thái khác nữa.

Thí dụ 9.2: Cho ô tô máy Milly với bảng chuyển trạng thái φ và ψ cho trên hình 9.2. Hãy tìm lớp các trạng thái tương đương tối đa và thay thế chúng bằng trạng thái thực hiện tương ứng.

$A \backslash X$	X_0	X_1	X_2
1	1	2	—
2	3	2	5
3	3	4	3
4	4	4	—
5	—	6	5
6	1	3	6

$A \backslash X$	X_0	X_1	X_2
1	1	0	—
2	1	0	0
3	1	0	1
4	1	0	—
5	—	0	1
6	1	0	1

a) Bảng chuyển đổi trạng thái

b) Bảng ra

Hình 9.2. Ô tô máy Milly

Trước hết ta xét hai trạng thái 1 và 3 xem có tương đương hay không?

a) *Điều kiện trạng thái ra*: Căn cứ vào bảng ra của hàm ψ , ta thấy ứng với hai trạng thái trong 1 và 3, các trạng thái ra của chúng là 1, 0, - và 1, 0, 1. Vậy nếu các điều kiện tiếp theo mà đủ thì chúng có thể thay thế bằng trạng thái ra 1, 0, 1. Điều kiện a) được thoả mãn.

b) *Điều kiện trạng thái trong*: Tại hai hàng 1 và 3 của bảng chuyển đổi trạng thái, để hai trạng thái trong 1 và 3 là tương đương thì hai trạng thái 2 và 4 cũng phải là hai trạng thái tương đương.

Đối với cặp trạng thái (2, 4), trong bảng ra ta có các trạng thái ra tương ứng là 1, 0, 0 và 1, 0, - nên điều kiện a) được thoả mãn. Nhưng trong bảng chuyển đổi trạng thái, để cặp trạng thái (2, 4) là tương đương, cặp trạng thái (3, 4) trong cột X_n cũng phải là tương đương. Vì với cặp trạng thái (3, 4), trong bảng ra ta có các trạng thái ra tương ứng là: 1, 0, 1 và 1, 0, - nên điều kiện a) được thoả mãn. Trong bảng chuyển đổi trạng thái, các cặp trạng thái trong tại tất cả các cột đều thoả mãn điều kiện tương đương đối với trạng thái trong. Nói cách khác, điều kiện b) được thoả mãn. Do đó ta có thể kết luận chung là cặp trạng thái (1, 3) là tương đương.

Bằng cách xem xét một cách tương tự, ta có các cặp trạng thái sau đây là các cặp tương đương: (3, 4); (1, 4); (4, 6); (3, 6). Riêng trạng thái 5 không tương đương với bất kỳ trạng thái nào, một mình nó tạo thành một "nhóm tương đương tối đa".

Từ các cặp trạng thái tương đương, ta có thể lập thành các nhóm tương đương tối đa như sau:

- (1, 3), (3, 4), (1, 4) → Có nhóm tương đương tối đa (1, 3, 4)
- (3, 4), (4, 6), (3, 6) → Có nhóm tương đương tối đa (3, 4, 6)
- (2, 4) → Có nhóm tương đương tối đa (2, 4)
- (5) → Có nhóm tương đương tối đa (5).

Nếu bây giờ ta gán cho mỗi nhóm một trạng thái thay thế, thực hiện chúng như sau:

- Nhóm (1, 3), (3, 4), (1, 4) thay bằng trạng thái 1.
- Nhóm (3, 4), (4, 6), (3, 6) thay bằng trạng thái 2.
- Nhóm (2, 4) thay bằng trạng thái 3.
- Nhóm (5) thay bằng trạng thái 4.

Trên cơ sở đó ta có bảng chuyển đổi trạng thái và bảng ra mới như ở hình 9.3.

Trong bảng chuyển đổi trạng thái, ta viết 1/3 để chỉ trạng thái thay thế 1 hoặc 3.

Như vậy, từ một ô tô-mát có sáu trạng thái trong, bằng cách thay thế các nhóm trạng thái tương đương tối đa bằng các trạng thái đại diện thực hiện chúng, ta có được

một ô tô máy tương đương với ô tô máy đã cho, nhưng số trạng thái trong chỉ là 4. Số trạng thái được đã giảm đi là 2.

$\hat{A}$	X_0	X_1	X_2
1	1	2	1/3
2	1/3	2	4
3	1	3	1/3
4	-	3	4

a) Bảng chuyển đổi trạng thái

A	X_0	X_1	X_2
1	1	0	1
2	1	0	0
3	1	0	0
4	-	0	0

b) Bảng ra

Hình 9.3. Ô tô máy đã được rút gọn

Từ mô hình kỹ thuật của ô tô máy ta thấy rằng, khi số lượng trạng thái trong ít đi, có thể dẫn đến số lượng biến trong cũng ít đi và do đó phần nhớ và phần tổ hợp của ô tô máy đều được đơn giản hơn. Kết quả cuối cùng là toàn bộ hệ thống được đơn giản và tất nhiên đem lại hiệu quả về chất lượng và kinh tế đối với thiết bị.

Qua thí dụ trên đây ta thấy rằng, tính tương đương của một nhóm các trạng thái thường phải xác định thông qua tính tương đương của những cặp trạng thái khác nhau. Khi số trạng thái trong lớn thì đó là một việc làm phức tạp. Vì vậy, tìm một phương pháp để dễ dàng xác định được nhóm các trạng thái tương đương tối đa của một ô tô máy là một việc làm rất có ý nghĩa. Sau đây ta nghiên cứu một trong các phương pháp đó.

9.2.3. Phương pháp xác định các cặp trạng thái tương đương và nhóm các trạng thái tương đương tối đa

Giả sử ô tô máy được cho dưới dạng bảng chuyển đổi trạng thái và bảng ra như ở hình 9.4. Phương pháp để xác định các trạng thái tương đương và các nhóm các trạng thái tương đương tối đa gọi là *phương pháp bảng tam giác*.

Trước hết, để tìm một cách nhanh chóng tất cả các cặp trạng thái tương đương của ô tô máy, ta lập một bảng hình tam giác, cho phép xét tất cả các cặp trạng thái có thể có. Trong bảng, theo hai trục ta bố trí các trạng thái trong như ở hình 9.5.

A	X ₁	X ₂	X ₃	X ₄
1	(1),0	7	2	6
2	1	7	(2),0	3
3	1	4	5	(3),1
4	1	(4),0	2	3
5	1	4	(5),0	3
6	1	4	5	(6),0
7	1	(7),0	5	8
8	1	4	5	(8),0

Hình 9.4. Ôtômát dùng để áp dụng phương pháp bảng tam giác

2	3,6xx						
3	4,7 xx 2,5:3,6	4,7 2,5					
4	4,7 xx 3,6	4,7	2,5				
5	4,7. xx 2,5:3,6	4,7	V	2,5			
6	4,7 2,5	4,7 xx 2,5:3,6	xx	2,5 xx 3,6	3,6 xx		
7	2,5 xx 6,8	2,5 3,8	4,7 3,8	2,5 3,8	4,7 3,8	4,7xx 6,8	
8	4,7 xx 2,5:6,8	4,7 x x 2,5:3,8	V	2,5 3,8	3,8	x x	4,7

1	2	3	4	5	6	7
(1,6)	(2,3)	(3,4)	(4,5)	(5,7)	-	(7,8)
	(2,4)	(3,5)	(4,7)	<u>(5,8)</u>		
	(2,5)	(3,7)	<u>(4,8)</u>		<u>(5,7,8)</u>	
	(2,7)	<u>(3,8)</u>		<u>(4,5,7,8)</u>		
	<u>(2,8)</u>		<u>(3,4,5,7,8)</u>			
			<u>(2,3,4,5,7,8)</u>			

Hình 9.5. Bảng tam giác tìm cặp các trạng thái tương đương và các nhóm tương đương tối đa

Cách điền vào bảng:

- Trong ô là giao của hàng A_i và cột A_j, ta điền dấu “x”, nếu cặp trạng thái (A_i, A_j) là không tương đương một cách hiển nhiên, tức là khi xét tính tương đương của hai

trạng thái A_i và A_j thì ít nhất một trong các điều kiện tương đương của hai trạng thái không thoả mãn. Thí dụ, đối với bảng chuyển đổi trạng thái ở hình 9.4, khi xét tính tương đương của hai trạng thái 3 và 6, hai trạng thái ra ứng với X_1 là 0 và 1. Như vậy, điều kiện về trạng thái ra không thoả mãn. Do đó trong bảng tam giác, trong ô là giao của hàng 6 và cột 3 ta ghi dấu "x". Cũng tương tự như vậy, trong ô là giao của hàng 8 và cột 6 ta cũng ghi dấu "x" vì khi xét tính tương đương của hai trạng thái 6 và 8, hai trạng thái ra đối với cột X_1 của hai hàng 6 và 8 trong bảng chuyển đổi trạng thái ở hình 9.4 là không giống nhau: 0 và 1, nên hai trạng thái 6 và 8 là không tương đương một cách hiển nhiên.

- Trong ô là giao của hàng A_i và cột A_j ta ghi dấu "V", nếu hai trạng thái A_i và A_j là tương đương hiển nhiên. Nghĩa là khi xét tính tương đương của hai trạng thái A_i và A_j nào đó của bảng chuyển đổi trạng thái và bảng ra kết hợp thì tất cả các điều kiện tương đương về trạng thái ra và trạng thái trong đều được thoả mãn ngay mà không phải xét thêm điều kiện gì nữa.

Trong ô tômát ở hình 9.4, các cặp trạng thái trong (3, 5) và (3, 8) đều là các cặp trạng thái tương đương một cách hiển nhiên, nên trong các ô là giao của cột 3 và các hàng 5 và 8 ta ghi dấu "V". Thật vậy, khi xét sự tương đương của hai cặp trạng thái trong (3, 5) và (3, 8), ta có các hàng chứa trạng thái trong và trạng thái ra như sau:

3:	1	4	5	3, 1	
5:	1	4	5, 0	3	và
3:	1	4	5	3, 1	
8:	1	4	5	8, 1	

Rõ ràng các điều kiện đối với các trạng thái ra và các trạng thái trong về tính tương đương của hai cặp trạng thái này được thoả mãn ngay mà không cần xét thêm điều kiện gì nữa.

- Trong ô là giao của trạng thái A_i và A_j mà khi xét tính tương đương của chúng ta cần phải xét tính tương đương của các cặp trạng thái khác thì ghi các cặp trạng thái cần phải xét này. Thí dụ, khi xét tính tương đương của cặp trạng thái (1, 2), ta cần phải xét tính tương đương của hai trạng thái 3 và 6. Nói cách khác, cặp (1, 2) chỉ là cặp tương đương khi cặp (3, 6) là tương đương. Trong ô là giao của cột 1 và hàng 2 ta ghi cặp trạng thái cần xét (3, 6)...

Xử lý bảng:

Sau khi đã ghi đầy đủ vào bảng tam giác, cần cử bảng chuyển đổi trạng thái và bảng ra đã cho, ta xử lý bảng để biết các cặp trạng thái không tương đương và các cặp trạng thái tương đương. Cách xử lý như sau:

- Dùng cặp trạng thái không tương đương hiển nhiên đã được đánh dấu bằng dấu "x" trong bảng để đánh dấu tất cả các cặp trạng thái khác cũng là các trạng thái không tương đương bằng cách rà soát trong tất cả các ô của bảng, nếu ô nào có chứa cặp trạng thái cần xét là cặp không tương đương hiển nhiên này thì đánh dấu "x" vào ô đó. Bởi vì khi xét điều kiện về trạng thái trong của hai trạng thái A_i và A_j này ta sẽ gặp phải cặp trạng thái cần xét đó là cặp không tương đương, và do đó cặp trạng thái đang xét A_i và A_j rõ ràng sẽ là không tương đương.

Để biết cặp không tương đương nào đã được dùng để đánh dấu các cặp trạng thái không tương đương khác, ta ghi thêm một dấu "x" vào ô đó. Như vậy, cặp trạng thái không tương đương nào đã được dùng để đánh dấu các cặp khác thì sẽ có hai dấu "x" trong ô tương ứng. Chú ý rằng sau lần đánh dấu thứ nhất, có thể sẽ lại xuất hiện các trạng thái không tương đương mới. Các cặp này lại được dùng để đánh dấu các cặp trạng thái còn lại. Cứ như vậy cho tới khi nào trong bảng, các ô ứng với cặp trạng thái không tương đương đều chứa hai dấu "x" thì kết thúc. Toàn bộ các ô không có dấu "x" đều ứng với các cặp trạng thái tương đương của ô tô máy. Như vậy, sau khi đánh dấu ta có tất cả các cặp trạng thái tương đương.

- Bước tiếp theo là *xác định các nhóm tương đương tối đa*:

Để xác định các nhóm tương đương tối đa của ô tô máy, ta làm như sau:

+ Theo từng cột, ta ghi lại dưới các cột tất cả các cặp trạng thái tương đương, ứng với các ô không có dấu "x" (hình 9.5).

+ Tiếp theo, lần lượt từ phải sang trái đối với các cột, ta ghép các cặp tương đương để hình thành các nhóm tương đương lớn hơn cho tới khi nào không thể ghép được nữa thì thôi. Trong khi ghép cần nhớ rằng, các trạng thái chỉ tạo thành nhóm các trạng thái tương đương khi chúng tương đương với nhau từng đôi một. Đối với ô tô máy ở hình 9.4 và bảng đã xử lý ở hình 9.5, ta lần lượt có các nhóm tương đương được mở rộng dần để cuối cùng có được hai nhóm tương đương tối đa như sau:

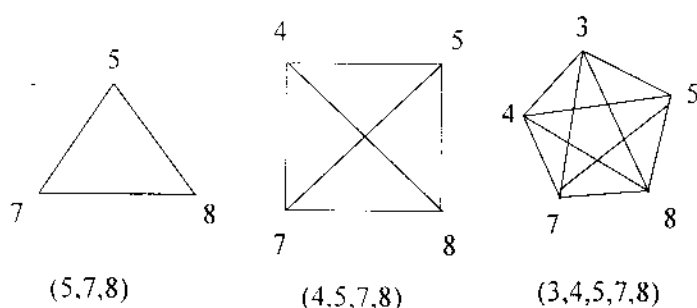
- Đầu tiên ta ghép ba cặp trạng thái tương đương (7, 8), (5, 7) và (5, 8) để có nhóm tương đương gồm ba trạng thái là (5, 7, 8).
- Tiếp theo ta lại ghép nhóm trạng thái này với nhóm các cặp trạng thái tương đương tại cột 4 là (4, 5), (4, 7), (4, 8) để có nhóm tương đương lớn hơn gồm 4 trạng thái trong tương đương với nhau từng đôi một. Đó là nhóm (4, 5, 7, 8).
- Ghép tiếp nhóm này với các cặp trạng thái trong cột 3 là : (3, 4), (3, 5), (3, 7), (3, 8) để có nhóm trạng thái tương đương gồm 5 trạng thái tương đương với nhau từng đôi một. Đó là nhóm (3, 4, 5, 7, 8).
- Tiếp theo ta ghép nhóm này với các cặp trạng thái tương đương của cột 2, gồm các cặp: (2, 3), (2, 4), (2, 5), (2, 7), (2, 8) để có nhóm tương đương gồm 6

trạng thái tương đương với nhau từng đôi một. Đó là nhóm (2, 3, 4, 5, 7, 8).

- Bây giờ, nếu ta muốn ghép nhóm vừa nhận được với cặp trạng thái tương đương tại cột 1 là (1, 6) thì cũng không thể tạo thành nhóm tương đương tối đa lớn hơn được. Do đó bản thân cặp trạng thái tương đương (1, 6) tạo thành một nhóm tương đương tối đa chỉ gồm hai trạng thái 1 và 6 mà thôi.

Cuối cùng ta có hai nhóm tương đương tối đa là (1, 6) và (2, 3, 4, 5, 7, 8).

Ta nhận xét rằng, một nhóm các trạng thái tương đương, dưới dạng đồ hình sẽ tạo thành một đồ hình đầy đủ. Thí dụ, ba nhóm tương đương (5, 7, 8), (4, 5, 7, 8) và (3, 4, 5, 7, 8) biểu diễn dưới dạng đồ hình có dạng như trên hình 9.6.



Hình 9.6. Đồ hình của các nhóm tương đương

Ta thấy tất cả trạng thái của ô tô máy đã cho đều tham gia vào một trong các nhóm tương đương tối đa. Ta nói các nhóm tương đương tối đa này là một tập hợp đầy đủ. Tính đầy đủ của các nhóm tương đương tối đa đảm bảo không có trạng thái nào của ô tô máy mà không được tính tới.

- Hình thành bảng chuyển đổi trạng thái và bảng ra mới, ít trạng thái trong hơn

Để hình thành bảng chuyển đổi trạng thái và bảng ra mới ít trạng thái trong hơn, ta gán cho mỗi nhóm trạng thái tương đương tối đa một trạng thái thực hiện nhóm đó theo nguyên tắc đã trình bày trên đây. Trong trường hợp cụ thể của ô tô máy ở hình 9.4, ta thay nhóm (1, 6) bằng trạng thái "a" và nhóm (2, 3, 4, 5, 7, 8) bằng trạng thái thực hiện chúng là trạng thái "b":

(1, 6) $\longrightarrow$ a

(2, 3, 4, 5, 7, 8) $\longrightarrow$ b

Vậy ô tô máy mới chỉ gồm có hai trạng thái trong là a và b.

Để điền vào các ô của bảng chuyển đổi trạng thái và bảng ra mới, ta làm như sau:

Ta nói, xuất phát từ trạng thái a , tức từ một trong các trạng thái 1 và 6, dưới tác động của một X_k nào đó, nếu trong bảng ở hình 9.4 của ô tô máy đã cho, trạng thái trong mới thuộc nhóm nào trong hai nhóm tương đương tối đa (1, 6) và (2, 3, 4, 5, 7, 8) thì ta ghi trạng thái mới tương ứng là a hoặc b . Nếu thuộc (1, 6) thì ghi là a ; nếu thuộc (2, 3, 4, 5, 7, 8) thì ghi là b . Trạng thái ra tương ứng của bảng ra mới là hội của các trạng thái ra thuộc cột X_k của các hàng thuộc nhóm (1, 6) hoặc (2, 3, 4, 5, 7, 8) tùy thuộc trạng thái trong mới thuộc nhóm trạng thái nào.

Với cách làm như vậy, cuối cùng ta được bảng chuyển đổi trạng thái và bảng ra kết hợp của ô tô máy đã cho nhưng đã được đơn giản số trạng thái trong xuống tối thiểu như ở hình 9.6.

Ta gọi ô tô máy cho bởi bảng chuyển đổi trạng thái và bảng ra ở hình 9.6 là ô tô máy S_2 và ô tô máy cho bởi bảng chuyển đổi trạng thái và bảng ra ở hình 9.4 là ô tô máy S_1 . Hai ô tô máy S_1 và S_2 là tương đương nhau. Chúng có cùng số trạng thái vào, cùng số trạng thái ra nhưng khác nhau về số trạng thái trong: S_1 có 8, còn S_2 có 2. Với tám trạng thái trong, để thực hiện ô tô máy S_1 cần ít nhất ba phần tử nhớ. Trong khi đó với hai trạng thái để thực hiện ô tô máy S_2 ta chỉ cần một phần tử nhớ cơ bản.

$\begin{matrix} & X \\ A & \end{matrix}$	X_1	X_2	X_3	X_4
$\begin{matrix} a \\ (1,6) \end{matrix}$	$(a).0$	b	b	$(a).0$
$\begin{matrix} b \\ (2,3,4,5,7,8) \end{matrix}$	a	$(b).0$	$(b).0$	$(b).1$

Hình 9.7. Bảng chuyển đổi trạng thái và bảng ra sau khi đã đơn giản trạng thái trong của ô tô máy ở hình 9.4

Trong bảng chuyển đổi trạng thái của ô tô máy S_2 , thực hiện ô tô máy S_1 có thể có nhiều trạng thái bên trong một hàng.

Khi áp dụng phương pháp đơn giản trạng thái trong đối với ô tô máy xác định không hoàn toàn có thể có hiện tượng cùng một trạng thái của ô tô máy đã cho sẽ tham gia vào một vài nhóm trạng thái tương đương tối đa. Nói cách khác, các nhóm tương đương tối đa có thể có hiện tượng giao nhau. Điều đó có thể giải thích, trong khi tìm kiếm trạng thái tương đương, trạng thái bất định có thể được gán cho các trạng thái khác nhau. Vì vậy đối với các ô tô máy không xác định hoàn toàn, số lượng các nhóm trạng thái tương đương tối đa tìm được bằng cách áp dụng phương pháp bảng tam giác, có thể không phải là số lượng tối thiểu các nhóm. Nói cách khác, đối với ô tô máy xác định không hoàn toàn, sau khi xác định được các nhóm tương đương tối đa, nếu có hiện tượng giao nhau của các nhóm, cần tiến hành bước tiếp theo để xác định một phủ tối thiểu đối với các nhóm đó, đảm bảo tính đầy đủ và tính đóng kín đối với các trạng thái trong của ô tô máy. Sau đó, mỗi nhóm trong phủ tối thiểu được gán với một trạng thái thay thế của ô tô máy S_2 thực hiện ô tô máy S_1 .

Ta có nhận xét rằng, trong các nhóm cực đại của phủ tối thiểu tìm được, có thể loại trừ những trạng thái không làm thay đổi tính đầy đủ và tính đóng kín của tập hợp các nhóm thuộc phủ đó. Trong trường hợp này ta có thể sử dụng cả các nhóm không

phải là tối đa và chúng không giao nhau. Tuy vậy, trong nhiều trường hợp nếu chỉ dùng các nhóm không giao nhau sẽ không cho ta ô tô máy thay thế có số lượng trạng thái trong tối thiểu.

Để nghiên cứu vấn đề vừa đặt ra, chúng ta xét thí dụ sau đây.

Thí dụ 9.3: Hãy đơn giản trạng thái trong của ô tô máy cho trên hình 9.8.

Để tìm các nhóm trạng thái tương đương tối đa, ta lập bảng tam giác như trên hình 9.9. Các cặp trạng thái tương đương và không tương đương được chỉ ra trên hình 9.9.

Kết quả xử lý bằng tam giác ta được các nhóm tương đương tối đa như sau:

$$a = (1, 2, 5, 6, 7)$$

$$b = (1, 3, 4, 5, 7)$$

Đây là hai nhóm giao nhau vì các trạng thái 1, 5, 7 tham gia trong cả hai nhóm.

	X_1	X_2	X_3
1	(1),-	4	6
2	(2),0	5	-
3	(3),1	-	7
4	3	(4),0	6
5	1	(5),0	-
6	2	-	(6),1
7	1	-	(7),1

Hình 9.8. Ô tô máy Milly

Gán mỗi nhóm với một trạng thái của ô tô máy S_2 thực hiện ô tô máy S_1 đã cho, ta được bảng chuyển đổi trạng thái và bảng ra của S_2 gồm hai trạng thái trong a và b như ở hình 9.10a.

2	4,5					
3	6,7	x				
4	1,3	$\frac{2,3}{4,5}$	6,7			
5	4,5	1,2	1,5	1,3		
6	1,2	v	$\frac{2,3}{6,7}$	$\frac{2,3}{x}$	1,2	
7	6,7	1,2	1,3	$\frac{1,3}{6,7}$	v	1,2
	1	2	3	4	5	6
	(1,2)	(2,5)	(3,4)	(4,5)	(5,6)	(6,7)
	(1,3)	(2,6)	(3,5)	(4,7)	(5,7)	
	(1,4)	(2,7)	(3,7)		(5,6,7)	
	(1,5)			(4,5,7)		
	(1,6)					
	(1,7)					

Hình 9.9. Bảng tam giác xét các cặp trạng thái tương đương của ô tô máy ở hình 9.8

Nếu làm cho nhóm a và nhóm b không giao nhau thì không tránh khỏi làm tăng số lượng trạng thái trong của ô tômát S_2 thực hiện S_1 . Thật vậy, nếu giả sử không dùng trạng thái 1 trong nhóm a thì lúc đó không thể đưa trạng thái trong 6 và trạng thái 7 vào cùng một nhóm tương đương, vì trạng thái 6 và 7 chỉ tương đương với điều kiện hai trạng thái 1 và 2 tương đương (hình 9.9).

Đến lượt mình, nếu đưa trạng thái 7 ra khỏi nhóm b sẽ dẫn tới trạng thái 1 và 3 không thể cùng đưa vào cùng một nhóm vì chúng chỉ là các trạng thái tương đương khi hai trạng thái 6 và 7 là tương đương, tức phải trong cùng một nhóm.

Như vậy, dù a hay b không chứa trạng thái 1 thì số cặp tương đương chỉ còn lại hai: (2, 6) và (5, 7). Vì vậy trong trường hợp này ô tômát S_2 sẽ có năm trạng thái trong như hình 9.10b đã chỉ rõ.

	X_1	X_2	X_3
a	(a), 0	(b), 0	(a), 1
b	(b), 1	(a), 1	(a), 1

a)

	X_1	X_2	X_3
1	(1), -	4	2
2	(2), 0	5	(2), 1
3	(3), 1	-	5
4	3	(4), 0	2
5	1	(5), 0	(5), 1

b)

Hình 9.10. a) Ô tômát S_2 dùng các nhóm tương đương giao nhau
b) Ô tômát S_2 dùng các nhóm tương đương không giao nhau

9.2.4. Phương pháp đơn giản trạng thái trong của ô tômát không đồng bộ xác định không đầy đủ

Khi cần đơn giản trạng thái trong của ô tômát không xác định đầy đủ, để có được tất cả các nhóm tương đương tối đa và không tối đa đảm bảo tính đóng kín và tính đầy đủ và có phủ tối thiểu (ít nhóm trạng thái tương đương nhất), thường phải tiến hành chọn nhiều lần và có nhiều phương án. Trong đó để tìm phủ tối thiểu của các nhóm tương đương có tính đầy đủ và đóng kín có thể dùng phương pháp sau:

a) Căn cứ vào ô tômát đã cho, lập bảng tam giác để xác định các nhóm tương đương tối đa.

b) Lập bảng phủ: Mỗi hàng thứ i ứng với một nhóm tối đa các trạng thái tương đương. Mỗi cột thứ j ứng với một trạng thái trong của ô tômát đã cho. Giao của cột j và hàng i , trong đó j là các trạng thái có trong nhóm tối đa cực đại i , được đánh dấu bằng dấu "x".

Sau đó sẽ xét, nếu cột nào chỉ có một dấu "x" thì dấu "x" đó được đánh dấu bằng cách đưa vào ngoặc (x). Những nhóm cực đại nào thuộc hàng dù chỉ có một dấu "x" đã được đánh dấu (tức (x)) sẽ là những thành phần cơ bản của phủ trạng thái. Ta làm tiếp tục đối với các cột còn lại tương tự như khi thực hiện phương pháp Kvaína trong đơn giản các hàm lôgic. Kết quả các nhóm được chọn sẽ tạo thành một phủ tối thiểu các nhóm tối đa các trạng thái tương đương, có tính đầy đủ và tính đóng kín.

c) Gán cho mỗi nhóm cực đại trong tập hợp các nhóm tìm được một trạng thái của S_2 thực hiện ô tô măt S_1 . Nói cách khác, thay mỗi nhóm này bằng một trạng thái tương đương. Sau đó lập bảng chuyển đổi trạng thái như thường lệ. Ô tô măt nhận được sẽ tương đương với ô tô măt đã cho và có số lượng trạng thái tối thiểu.

Nhưng phương pháp này chỉ áp dụng được đối với một lớp các ô tô măt xác định không hoàn toàn, thoả mãn các điều kiện sau đây:

- Các trạng thái của ô tô măt không chứa các trạng thái bất định của trạng thái ra.
- Trong mỗi hàng không có quá một trạng thái bền. Nếu các trạng thái bền trong cùng một cột thì các trạng thái bất định của chúng phải ở những vị trí (cột) như nhau.

Thí dụ 9.4: Hãy đơn giản trạng thái trong của ô tô măt cho trên hình 9.11.

	X_1	X_2	X_3	X_4
1	(1),000	2	7	-
2	3	(2),100	-	10
3	(3),000	4	9	-
4	5	(4),010	-	11
5	(5),000	6	8	-
6	1	(6),001	-	-
7	5	-	(7),001	10
8	3	-	(8),010	12
9	1	-	(9),100	11
10	-	2	7	(10),000
11	-	4	9	(11),000
12	-	6	8	(12),000

Hình 9.11. Ô tô măt xác định không đầy đủ, mỗi hàng chỉ có một trạng thái bền và không có trạng thái ra bất định

Ta nhận thấy rằng, bảng chuyển đổi trạng thái của ô tô măt này thoả mãn các điều kiện đã nêu trên đây. Do đó có thể áp dụng phương pháp nói trên để tìm phủ tối thiểu các nhóm trạng thái tương đương tối đa.

Trong trường hợp này nếu lập bảng tam giác để xác định các nhóm trạng thái tương đương cực đại, ta sẽ được các nhóm cực đại sau đây: (1, 10), (2, 10), (3, 11), (4, 11), (5, 12), (6, 12), (7, 10), (8, 12), (9, 11).

Bảng phủ tối thiểu từ các nhóm cực đại tìm được cho trên hình 9.12.

Ta nhận thấy rằng, tất cả chín hàng, ứng với chín nhóm các trạng thái tương đương đều chứa dấu "x", nên cả chín nhóm đó đều nằm trong phủ tối thiểu. Ngoài ra không có trạng thái nào khác của ô tô máy đã cho nằm ngoài chín nhóm tối đa đó, nên phủ tối thiểu ở đây gồm chín nhóm tối đa. Bảng chuyển đổi trạng thái và bảng ra kết hợp của ô tô máy S_2 , thực hiện ô tô máy đã cho, cho trên hình 9.13a.

	1	2	3	4	5	6	7	8	9	10	11	12
(1,10)	x									x		
(2,10)		x								x		
(3,11)			x								x	
(4,11)				x							x	
(5,12)					x							x
(6,12)						x						x
(7,10)							x			x		
(8,12)								x				x
(9,11)									x		x	

Hình 9.12. Bảng tìm phủ tối thiểu

	X_1	X_2	X_3	X_4
1	(1).000	2	7	(10).000
2	3	(2).100	7	(10).000
3	(3).000	4	9	(11).000
4	5	(4).010	9	(11).000
5	(5).000	6	8	(12).000
6	1	(6).001	8	(12).000
7	5	2	(7).001	(10).000
8	3	6	(8).010	(12).000
9	1	4	(9).100	(11).000

	X_1	X_2	X_3	X_4
1	(1).000	2	7	(1).000
2=(2)	3	(2).100	7	1
3	(3).000	4	9	-
4=(4)	5	(4).010	-	(3).000
5	(5).000	6	8	-
6=(6)	1	(6).001	-	(5).000
7=(7)	5	-	(7).001	1
8=(8)	3	6	(8).010	(12).000
9=(9)	1	-	(9).100	3

a) Dùng các nhóm giao nhau

b) Dùng các nhóm không giao nhau

Hình 9.13. Ô tô máy S_2 thực hiện ô tô máy S_1

Nếu ta sử dụng các nhóm không giao nhau $1 = (1, 10)$; $2 = (2)$; $3 = (3, 11)$; $4 = (4)$; $5 = (5, 12)$; $6 = (6)$; $7 = (7)$; $8 = (8)$; $9 = (9)$ sẽ có bảng chuyển đổi trạng thái và bảng ra kết hợp cho trên hình 9.13b, trong đó sẽ có một số trạng thái không xác định. Nhờ những trạng thái bất định này, cấu trúc của ô tô máy có thể đơn giản hơn. Vì vậy xu hướng chung là nên sử dụng các nhóm tương đương tối đa không giao nhau, nếu điều đó không làm tăng số trạng thái trong của ô tô máy thực hiện ô tô máy đã cho.

Cần chú ý rằng phương pháp trên đây khi áp dụng cho trường hợp tổng quát có thể dẫn tới hiện tượng lẫn mất tính đóng kín của các nhóm tìm được. Hiện tại đã có các phương pháp tìm phủ tối thiểu, bao gồm các nhóm tương đương tối đa, giao nhau hoặc không giao nhau, với số lượng tối thiểu, đảm bảo tính đóng kín và tính đầy đủ. Trong khuôn khổ của tài liệu này ta chỉ trình bày một trong các phương pháp đó. Phương pháp này dựa trên khái niệm gọi là tập hợp tương đương gắn bó của các nhóm trạng thái tương đương.

Một tập hợp có thứ tự của các nhóm các trạng thái tương đương được gọi là tập hợp tương đương gắn bó, nếu các trạng thái trong của nhóm thứ N_i của tập hợp này chỉ tương đương khi các trạng thái trong thuộc nhóm N_{i+1} cũng là tương đương.

Tập hợp tương đương gắn bó của các nhóm trạng thái tương đương được gọi là chuẩn, nếu nó không chứa các nhóm lặp lại hoặc các nhóm nằm trong các nhóm khác của tập hợp này.

Người ta đã chứng minh được rằng, để ô tô máy S_2 thực hiện được ô tô máy S_1 đã cho thì điều kiện cần và đủ là gán cho mỗi trạng thái trong của S_2 một nhóm các trạng thái tương đương của ô tô máy S_1 đã cho.

+ Mỗi trạng thái của ô tô máy S_1 phải tham gia ít nhất trong một nhóm các trạng thái tương đương (tối đa hoặc không tối đa) được gán cho trạng thái của S_2 .

+ Nếu ta đã gán nhóm các trạng thái tương đương N cho một trạng thái của S_2 thì những trạng thái khác của nó phải được gán với tất cả các nhóm của tập hợp tương đương gắn bó mà xuất phát từ N hoặc với những nhóm khác bao trùm những nhóm này. Chú ý rằng, mỗi nhóm đều được gán với một trạng thái của ô tô máy S_2 .

Như vậy, thủ tục tổng quát để đơn giản trạng thái trong của ô tô máy không đồng bộ, không xác định hoàn toàn là:

a) Tìm các nhóm trạng thái tương đương tối đa của ô tô máy S_1 đã cho bằng cách dùng bảng tam giác.

b) Đối với mỗi nhóm trạng thái tương đương tối đa tìm tập hợp tương đương gắn bó của các nhóm trạng thái tương đương. Nếu trong những tập hợp này gặp những nhóm các trạng thái tương đương không tối đa thì đối với chúng cũng lập các tập hợp tương đương gắn bó tương ứng.

c) Từ các nhóm của tất cả các tập hợp tương đương gắn bó này ta lập thành một tập hợp chung đầy đủ và có tính đóng kín. Từ tập hợp chung này cần tìm một tập hợp con gồm các nhóm trạng thái tương đương thoả mãn các điều kiện đầy đủ và đóng kín.

d) Để tìm tập hợp con nói trên ta lập bảng phủ mà mỗi cột ứng với một trạng thái trong của ô tô máy S_1 cho trước. Mỗi hàng ứng với một nhóm các trạng thái tương đương tối đa đã tìm được qua bảng tam giác và các nhóm tương đương không tối đa đã xây dựng các tập hợp tương đương gắn bó.

e) Xác định các tập hợp tương đương gần bó nòng cốt. Đó là những tập hợp do những nhóm tương đương cực đại sinh ra mà các hàng của các nhóm này trong bảng phủ chứa dấu “x” duy nhất của một hoặc một số cột nào đó. Ký hiệu tích logic của các nhóm tương đương thuộc tập hợp tương đương gần bó nòng cốt thứ j là $\{\alpha_j\}$, mà $j = 1, 2, 3, \dots$

f) Trừ những cột chỉ chứa một dấu “x”, đối với các cột còn lại, ta lập hàm F . Hàm F được xác định như sau:

$$F = \prod_{i=1}^N \left(\sum_{k=1}^M (\beta_k) \right)$$

trong đó, N là số cột có từ hai dấu “x” trở lên;

M là số hàng chứa dấu “x” trong cột thứ i ;

β_k là tích logic của các nhóm tương đương trong tập hợp các nhóm tương đương gần bó ứng với tập hợp đầy đủ hoặc không đầy đủ thuộc hàng thứ k sinh ra. Hàng thứ k là hàng mà tại cột thứ i có dấu “x”. Biến i lấy lần lượt theo số cột của bảng, trừ những cột có chứa độc nhất một dấu “x”.

g) Biến đổi hàm F thành dạng tổng của các tích tối thiểu mà mỗi thừa số của tích, tức mỗi biến, là những nhóm tương đương cực đại hoặc không cực đại được sử dụng để lập lên các tập hợp tương đương gần bó. Ký hiệu dạng tối thiểu của hàm F là F_T .

h) Mở rộng hàm F_T thành hàm F_M bằng cách nhân logic tất cả các $\{\alpha_j\}$, mà $j = 1, 2, 3, \dots$ với F_T , tức:

$$F_M = F_T \cdot (\prod_j (\alpha_j))$$

và đưa F_M về dạng tổng của các tích bằng cách thực hiện phép nhân logic trên đây.

i) Mỗi số hạng (mỗi tích) của F_M là một khả năng lựa chọn các nhóm trạng thái tương đương tối đa và không tối đa thoả mãn các điều kiện đầy đủ và đóng kín. Vì vậy những nhóm của ô tô máy S_2 thay thế ô tô máy S_1 sẽ là các nhóm của F_M mà trong tích có ít thừa số nhất.

Thí dụ 9.5: Áp dụng thủ tục đơn giản trạng thái trong của ô tô máy không đồng bộ, xác định không đầy đủ để đơn giản trạng thái trong của ô tô máy cho trên hình 9.14.

Ta lập bảng tam giác như ở hình 9.15 để tìm các nhóm tương đương tối đa. Ta được hai nhóm sau:

$$a = (1, 3, 4, 5)$$

$$b = (2, 3, 5, 6)$$

Đây là hai nhóm giao nhau. Vậy tiếp tục áp dụng thủ tục đơn giản đã trình bày trên đây để tìm kiếm khả năng đơn giản tiếp tục như thế nào? Để làm việc này ta làm như sau:

A	X_1	X_2	X_3
1	(1), 00	2	4
2	3	(2), 10	6
3	(3), 00	-	5
4	1	-	(4), -0
5	1	2	(5), 1-
6	1	-	(6), 11

Hình 9.14. Ôtômat không đồng bộ, không xác định đầy đủ

2	1,3 4,6x				
3	4,5	5,6			
4	V	1,3 4,6 x	1,3 4,5		
5	4,5	5,6	1,3	V	
6	4,6	1,3	1,3 5,6	x	V
	1	2	3	4	5

Hình 9.15. Bảng tam giác tìm các cặp trạng thái tương đương

- Tìm các tập hợp tương đương gần bó của các nhóm các trạng thái tương đương tối đa:

+ Từ nhóm $a = (1, 3, 4, 5)$ ta có:

$(1,3) \rightarrow (4,5) \rightarrow (1,3,4,5)$
 $(1,4) \rightarrow$
 $(1,5) \rightarrow (4,5)$
 $(3,4) \rightarrow (1,3) \rightarrow (4,5) \rightarrow (1,3,4,5)$
 $(3,5) \rightarrow (1,3)$
 $(4,5) \rightarrow$

Trong đó các cặp trạng thái tương đương cần phải xét $(1, 3)$, $(1, 4)$, $(3, 5)$ và $(4, 5)$ đều nằm trong nhóm a . Vậy ta có $a \rightarrow a$.

Từ nhóm $b = (2, 3, 5, 6)$, ta có:

$(2,3) \rightarrow (5,6)$
 $(2,5) \rightarrow (5,6)$
 $(2,6) \rightarrow (1,3) \rightarrow (4,5) \rightarrow (1,3,4,5)$, tức $b \rightarrow a$
 $(3,5) \rightarrow (1,3) \rightarrow (4,5) \rightarrow$ tức
 $(3,5) \rightarrow (1,3,4,5)$ hay $b \rightarrow a$
 $(3,6) \rightarrow (1,3) \rightarrow (4,5)$, tức
 $(3,6) \rightarrow (1,3,4,5)$ hay $b \rightarrow a$
 $(5,6) \rightarrow$ (Không cần xét gì thêm).

Vì các cặp trạng thái (5, 6) và (4, 5) là các cặp tương đương hiển nhiên, không cần xét gì thêm. Do đó ta có tập hợp tương đương gần bó của các nhóm trạng thái tương đương tối đa là: $b \rightarrow a$.

Vì trong quá trình tìm các tập hợp tương đương gần bó không xuất hiện nhóm không tối đa nào nên bảng phủ chỉ gồm có hai hàng như hình 9.16 đã chỉ ra.

Trong bảng phủ này, các cột 1, 2, 4 và 6 chỉ chứa một dấu "x". Do đó các tập hợp tương đương gần bó nồng cốt tương ứng là:

$$\alpha_1 = \{a\}; \quad \alpha_2 = \{b\};$$

$$\alpha_3 = \{a\}; \quad \alpha_4 = \{b\}$$

Hàm F tính theo các cột 3 và 5:

$$F = (a + b)(a + b) = a + ba = a = F_T$$

$$\begin{aligned} \text{Vậy hàm } F_M &= F.\alpha_1.\alpha_2.\alpha_3.\alpha_4 = \\ &= a.a.b.a.b = ab \end{aligned}$$

	1	2	3	4	5	6
a	(x)		x	(x)	x	
b		(x)	x		x	(x)

Hình 9.16. Bảng phủ của các nhóm tương đương tối đa

Trong trường hợp đang xét, dạng tổng của các tích của F_M chỉ có một số hạng duy nhất là $a.b$, như vậy không cần phải chọn gì thêm. Rõ ràng a và b là hai trạng thái của ô tô máy S_2 , thực hiện ô tô máy S_1 , đã cho trên hình 9.14. Các nhóm tối đa a và b là hai nhóm giao nhau, nhưng không thể bỏ đi bất kỳ trạng thái nào trong a hoặc trong b mà không làm tăng số lượng trạng thái của ô tô máy S_2 .

9.3. MÃ HOÁ TRẠNG THÁI CỦA Ô TÔ MÁY

Tổng hợp cấu trúc của thiết bị số có nhớ gồm nhiều bước, trong đó mã hoá trạng thái mà trước hết là trạng thái trong của ô tô máy có tính quyết định tới nhiều đặc trưng kỹ thuật quan trọng của nó.

Mã hoá thực chất là bước đầu tiên tiệm cận tới việc thực hiện kỹ thuật, xuất phát từ dạng chung nhất, khái quát nhất là dạng hình thức của ô tô máy.

Nguyên tắc chung khi mã hoá trạng thái của ô tô máy là mã hoá trạng thái vào, trạng thái ra và trạng thái trong riêng biệt. Vì việc mã hoá trạng thái trong của ô tô máy liên quan tới các đặc trưng kỹ thuật của ô tô máy như độ tin cậy chức năng, độ đơn giản mạch và tốc độ làm việc của ô tô máy nên chúng ta sẽ tập trung hơn vào việc mã hoá các trạng thái trong.

Mã hoá thực chất là gán cho mỗi trạng thái của ô tô máy một tổ hợp mã nhị phân, có độ dài D nào đó. Vì vậy, nếu số lượng trạng thái cần mã hoá là T thì số bit mã tối thiểu cần thiết để mã hoá chúng là D , tính theo công thức:

$$D = \lceil \log_2 T \rceil$$

trong đó dấu $\lceil \dots \rceil$ chỉ rằng phải lấy số nguyên nhỏ nhất lớn hơn hay bằng $\log_2 T$.

Như vậy:

+ Để mã hoá N trạng thái vào của ô tô-mát, ta cần số bit mã tối thiểu là n , mà:

$$n = \lceil \log_2 N \rceil$$

Số n đồng thời cũng là số biến vào hay số đầu vào của ô tô-mát cần thiết kế.

+ Để mã hoá M trạng thái ra của ô tô-mát, cần số bit mã là m , mà:

$$m = \lceil \log_2 M \rceil$$

Số m đồng thời cũng là số biến ra hay số đầu ra của ô tô-mát.

+ Để mã hoá P trạng thái trong của ô tô-mát, số bit mã tối thiểu cần dùng là p , mà:

$$p = \lceil \log_2 P \rceil$$

Đối với các mã dùng mã hoá các trạng thái trong của ô tô-mát có liên quan tới loại phần tử nhớ sẽ sử dụng, các phần tử nhớ thường chia ra làm các phần tử nhớ có xung đồng bộ (có đầu vào xung nhịp) và không có xung đồng bộ (không có đầu vào xung nhịp). Trong trường hợp dùng các phần tử nhớ có đầu vào xung nhịp thì việc mã hoá đơn giản vì không phải dùng biện pháp mã hoá chống tranh chấp trạng thái. Ngược lại, nếu dùng các phần tử nhớ không có đầu vào xung nhịp (không có xung đồng bộ) thì việc mã hoá trạng thái trong nhất thiết phải đảm bảo chống tranh chấp trạng thái. Trong trường hợp này số bit mã có thể phải tăng lên khá nhiều.

Ngoài ra cần lưu ý rằng, số bit mã dùng mã hoá các trạng thái trong của ô tô-mát có liên quan trực tiếp tới độ phức tạp của ô tô-mát. Vì mỗi bit mã phải được giữ (được thực hiện) bằng một phần tử nhớ, nên khi số bit mã tăng thì số phần tử nhớ cũng tăng và do đó phần nhớ cũng phức tạp hơn và kéo theo làm phần tổ hợp của ô tô-mát thêm phức tạp. Ngoài ra, khi ô tô-mát chuyển từ trạng thái này sang trạng thái khác thì tùy theo số lượng phần tử nhớ cần thay đổi trạng thái nhiều hay ít mà thời gian cần thiết để ô tô-mát chuyển đổi xong trạng thái là lâu hay chóng. Từ đó tốc độ làm việc của ô tô-mát cũng chậm hay nhanh.

Mặt khác, khi có từ hai phần tử nhớ trở lên cần thay đổi trạng thái trong một bước chuyển trạng thái của ô tô-mát thì khả năng các phần tử nhớ này thay đổi đồng thời là rất khó. Cũng chính vì thế mà dễ có khả năng xảy ra hiện tượng tranh chấp trạng thái trong của ô tô-mát và là nguyên nhân gây nên hiện tượng làm không đúng chức năng đã thiết kế. Hiện tượng này thường gọi là hiện tượng tranh chấp trạng thái nguy hiểm. Vì một khi chức năng đã không được đảm bảo thì những tính năng khác không còn ý nghĩa nữa. Do đó mã hoá chống tranh chấp trạng thái của ô tô-mát là một việc có ý nghĩa quan trọng hàng đầu. Để hiểu rõ hiện tượng tranh chấp trạng thái của ô tô-mát, chúng ta hãy xét trường hợp mã hoá trạng thái trong ảnh hưởng tới độ tin cậy chức năng như thế nào.

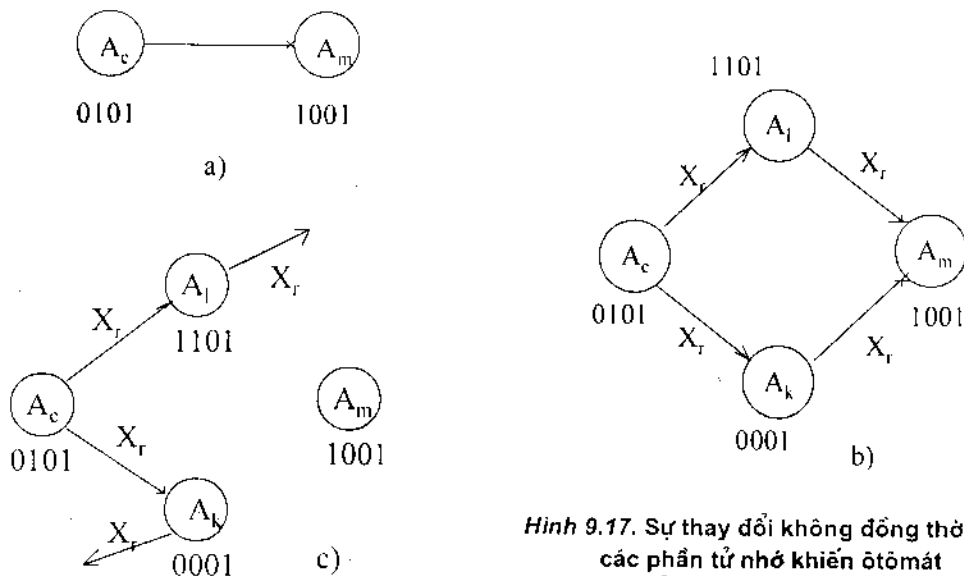
9.3.1. Hiện tượng tranh chấp trạng thái trong của ô tô máy

Để hiểu bản chất của hiện tượng tranh chấp trạng thái trong của ô tô máy, chúng ta hãy xét một bước chuyển trạng thái của ô tô máy giữa hai trạng thái A_c và A_m dưới tác động của trạng thái vào X_r với hai mã dùng để mã hoá hai trạng thái A_c và A_m như sau: A_c có mã bốn bit là 0101, còn A_m cũng có mã bốn bit là 1001. Lưu ý rằng, trong trường hợp này khi ô tô máy thay đổi từ trạng thái A_c sang trạng thái A_m thì có hai phần tử nhớ, ứng với hai bit thứ nhất và thứ hai phải thay đổi một cách đồng thời:

- Phần tử nhớ thứ nhất thay đổi từ 0 sang 1;
- Phần tử nhớ thứ hai thay đổi từ 1 sang 0.

Các phần tử nhớ thứ ba và thứ tư không thay đổi trạng thái (hình 9.17a).

Trên thực tế thời điểm và thời gian cần thiết để thay đổi trạng thái của các phần tử nhớ là khác nhau, ngay cả khi chúng được sản xuất cùng một loạt. Để thay đổi trạng thái, các phần tử nhớ cần nhận được các tín hiệu kích thích mà các tín hiệu này lại lấy từ những nguồn khác nhau. Do đó thời điểm xuất hiện của chúng cũng khác nhau. Từ đó ta thấy việc chuyển trạng thái của các phần tử nhớ khó có thể xảy ra đồng thời.



Hình 9.17. Sự thay đổi không đồng thời của các phần tử nhớ khiến ô tô máy chuyển trạng thái không đúng

Giả thiết phần tử nhớ thứ nhất chuyển trạng thái trước. Khi đó từ trạng thái A_c ô tô máy sẽ chuyển sang trạng thái A_l , là trạng thái được mã bằng mã 1101. Nếu phần tử nhớ thứ hai chuyển trạng thái trước, ô tô máy sẽ chuyển sang trạng thái A_k là trạng thái được mã hoá bằng mã 0001. Nếu trong hai trường hợp trên, trong đồ hình của ô tô máy, từ trạng thái A_l và A_k , dưới tác động của trạng thái vào X_r , ô tô máy vẫn có bước chuyển về trạng thái A_m thì cuối cùng ô tô máy vẫn về đúng trạng thái A_m (hình 9.17b). Trong

trường hợp này chức năng của ôtomát không bị thay đổi nên gọi là hiện tượng tranh chấp trạng thái không nguy hiểm. Ta nói ôtomát chỉ lướt qua các trạng thái A_i hoặc A_k trước khi về trạng thái A_m . Khi lướt qua các trạng thái trên, do thời gian rất ngắn nên ôtomát không để lại hậu quả gì. Trạng thái A_i và A_k là các trạng thái không bền đối với trạng thái vào X_i .

Nhưng nếu xuất phát từ A_i và A_k ôtomát không có các bước chuyển, dưới tác động của X_i , tới trạng thái A_m thì rõ ràng do hiện tượng chuyển trạng thái không đồng thời của các phần tử nhớ mà ôtomát không thực hiện được chức năng của mình (hình 9.17c). Hiện tượng này gọi là hiện tượng tranh chấp trạng thái nguy hiểm và cần phải loại bỏ. Việc loại bỏ này có thể thực hiện bằng phương pháp xử lý phần cứng hay mã hoá trạng thái trong của ôtomát một cách thích hợp.

Phương pháp chống tranh chấp trạng thái nguy hiểm có nhiều:

- Dùng các phần tử nhớ được nhịp hay xử lý bằng kết cấu phần cứng.
- Có thể dùng mã chống được tranh chấp như mã đơn vị, mã kế cận...
- Có thể dùng một thủ tục tạo mã (algôrit mã hoá) thích hợp để có những tổ hợp mã và phương thức mã hoá không xảy ra tranh chấp trạng thái nguy hiểm.

Trong phần tiếp theo chúng ta sẽ nghiên cứu một số mã và một thủ tục tạo mã và mã hoá cho phép chống được hiện tượng tranh chấp trạng thái nguy hiểm nói trên.

9.3.2. Các mã chống tranh chấp trạng thái nguy hiểm của ôtomát

Như trên đã trình bày, mã hoá trạng thái trong của ôtomát có ảnh hưởng trực tiếp đến ba đặc trưng quan trọng nhất của một thiết bị. Đó là độ tin cậy chức năng, độ đơn giản và tốc độ làm việc của thiết bị. Về phần mình, mỗi đặc trưng lại đòi hỏi những điều kiện khác nhau, trong đó có những yêu cầu có tính mâu thuẫn. Do đó tới nay chưa có một phương pháp nào có thể đảm bảo tối ưu đồng thời cả ba đặc trưng trên đây. Tuy vậy người ta đã xây dựng những thủ tục mã hoá để đảm bảo tối ưu một trong ba đặc trưng và các thủ tục đảm bảo hai trong ba đặc trưng đó. Tất nhiên bao giờ đặc trưng về độ tin cậy chức năng của thiết bị cũng là đặc trưng được đặt lên hàng đầu vì một khi chức năng đã không được đảm bảo chắc chắn thì mọi đặc trưng khác đều trở nên vô nghĩa.

Nếu chỉ để đảm bảo độ tin cậy chức năng của thiết bị, có thể dùng các mã sau đây để mã hoá trạng thái trong của ôtomát:

Mã đơn vị: Mã đơn vị là mã đơn giản nhất, nhưng độ dài mã là lớn nhất. Đó là ưu và nhược điểm chính của loại mã này.

Mã đơn vị về bản chất là mã nhị phân mà mỗi bit của mã được dùng để mã hoá một trạng thái trong của ôtomát. Như vậy nếu ôtomát có P trạng thái trong thì độ dài của mã đơn vị dùng để mã hoá là P bit. Bằng việc mã hoá như vậy, bất kỳ bước chuyển trạng thái nào của ôtomát từ một trạng thái nào đó sang trạng thái mới A_m chỉ phụ

thuộc vào việc thiết lập trạng thái 1 của bit tương ứng với trạng thái A_m đó. Do đó không bao giờ có hiện tượng tranh chấp trạng thái.

Độ dài của mã lớn làm thiết bị thêm phức tạp cả đối với phần mạch tổ hợp lẫn phần mạch nhớ.

II - mã: Là loại mã cho phép chống được tranh chấp nguy hiểm và có đặc điểm sau đây:

- Hai trạng thái trong bất kỳ của ô tômát phải được mã hoá bằng các tổ hợp mã khác nhau.

- Đối với mỗi cặp chuyển biến trạng thái bất kỳ:

$$A_g \xrightarrow{x_u} A_j \text{ và } A_l \xrightarrow{x_l} A_k$$

trong đó $A_g \neq A_k$; $A_g \neq A_l$; $A_l \neq A_k$ và $A_j \neq A_l$, trong các biến tồn tại, dù chỉ là một biến, sao cho nó nhận giá trị 1/0 ở trạng thái A_g và A_j thì lại nhận giá trị 0/1 ở các trạng thái A_l và A_k .

II - mã là loại mã có mức độ trung bình về mặt độ đơn giản và tốc độ làm việc của mạch.

Mã kế cận: Hai tổ hợp mã gọi là kế cận nhau nếu chúng chỉ khác nhau ở một bit. Thí dụ: 0110 và 0111, chúng khác nhau ở bit cuối cùng. Chúng là hai tổ hợp mã kế cận.

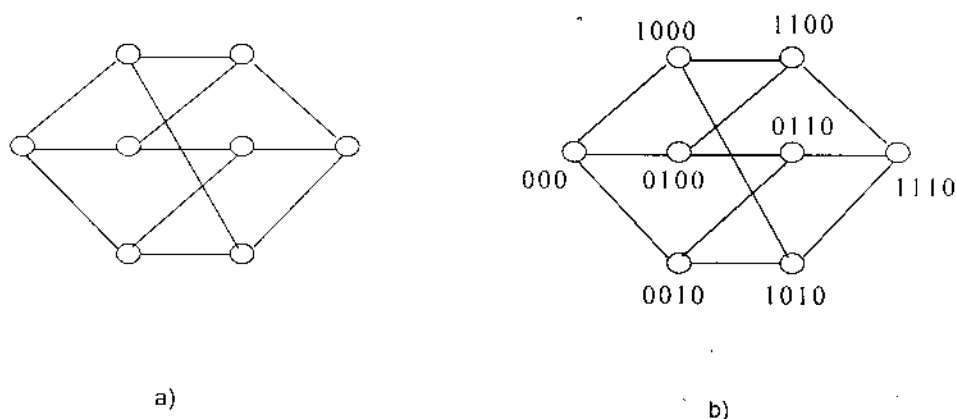
Nếu ta mã hoá bất kỳ hai trạng thái nào mà giữa chúng có bước chuyển đổi trạng thái, bằng hai tổ hợp mã kế cận thì khi ô tômát chuyển từ trạng thái này sang trạng thái khác luôn chỉ có một bit thay đổi, nghĩa là chỉ có một phần tử nhớ cần thay đổi trạng thái của mình. Do đó không bao giờ có hiện tượng tranh chấp trạng thái. Từ đây chúng ta suy ra nguyên tắc mã hoá kế cận đối với một ô tômát như sau:

+ Chọn số bit mã của bộ mã kế cận bằng p . Trong đó $p = \max(\lceil \log_2 P \rceil, K)$, K là số lượng bước chuyển lớn nhất của một đỉnh tới các đỉnh khác của đồ hình trạng thái của ô tômát.

+ Thực hiện mã hoá trạng thái trong của ô tômát như sau:

- Bất kỳ hai trạng thái nào có tồn tại bước chuyển trạng thái đều phải được mã hoá bằng hai tổ hợp mã kế cận.
- Cách mã: Gán tổ hợp mã đầu tiên gồm toàn bit 0, tức 00...0 cho trạng thái có nhiều mối liên hệ bước chuyển nhất.
- Lần lượt gán các tổ hợp mã kế cận với tổ hợp mã đầu tiên cho các trạng thái có bước chuyển liên quan tới trạng thái đã mã hoá trước đó. Nếu số bit mã không đủ, có thể mở rộng thêm.
- Cứ thế phát triển việc gán các tổ hợp mã cho tất cả các đỉnh còn lại theo nguyên tắc hai trạng thái bất kỳ, nếu giữa chúng có bước chuyển, phải mã hoá bằng hai tổ hợp mã kế cận.

Thí dụ về việc mã hoá kế cận của ô tômát có đồ hình trạng thái ở hình 9.18a cho trên hình 9.18b.

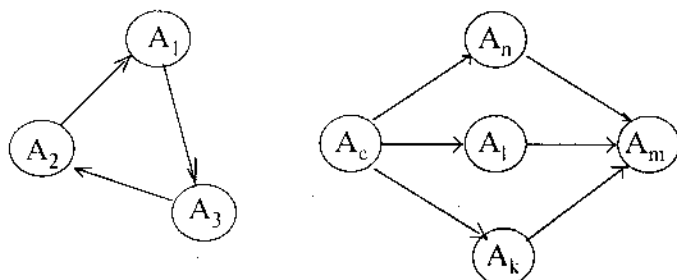


Hình 9.18. Ô tômát có 8 trạng thái (a) và mã hoá các trạng thái trong bảng mã kế cận (b)

Tất nhiên không phải lúc nào cũng có thể thực hiện mã hoá bằng mã kế cận. Cụ thể, nếu đồ hình của ô tômát có chứa những chu trình kín với một số lẻ các trạng thái hoặc nếu giữa hai trạng thái kế cận bậc hai A_c và A_m , tức hai trạng thái mà trong đồ hình kế cận chúng nằm cách nhau một trạng thái khác, có từ ba con đường trở lên qua các trạng thái A_n , A_l và A_k chẳng hạn, thì sẽ không thể mã hoá bằng mã kế cận được. Hình 9.19a và b là hai trường hợp tương ứng không thể mã hoá bằng mã kế cận.

Trong trường hợp này, để có thể mã hoá được bằng mã kế cận, người ta thường thêm các trạng thái phụ sao cho phá được cấu trúc của hình 9.19 nhưng không gây ảnh hưởng tới trạng thái ra khi ô tômát hoạt động.

Ngoài các bộ mã trên đây, còn nhiều phương pháp mã hoá cho phép chống được hiện tượng tranh chấp trạng thái của ô tômát.



Hình 9.19. Những đồ hình không thể mã bằng mã kế cận

Hiện tượng tranh chấp trạng thái sẽ được khắc phục nếu các phần tử nhớ được dùng là các phần tử nhớ được nhịp. Tín hiệu nhịp là một loại tín hiệu xung mà xung tích cực là loại xung có độ rộng tồn tại rất hẹp, chỉ đủ để thực hiện một bước chuyển

đơn trong quá trình hoạt động của ô tô máy. Trong trường hợp này, các phần tử nhớ cần thay đổi trạng thái sẽ thay đổi đồng thời trong thời gian có xung nhịp tác động và do đó tránh được hiện tượng tranh chấp trạng thái.

Qua đây chúng ta cũng thấy rằng, các phương pháp mã hoá chống tranh chấp chỉ liên quan tới các ô tô máy không đồng bộ. Đối với các ô tô máy đồng bộ, việc mã hoá trạng thái trong có thể thực hiện trong mối quan tâm tới các chỉ tiêu khác như độ đơn giản và tốc độ làm việc của mạch.

Do hạn chế của tài liệu này, dưới đây chúng ta chỉ xét một phương pháp mã hoá đảm bảo độ tin cậy chức năng của ô tô máy. Bạn đọc có thể tìm hiểu các thuật toán đảm bảo hai trong ba chỉ tiêu nói trên trong các tài liệu đã chỉ ra trong phần tài liệu tham khảo.

9.3.3. Phương pháp mã hoá chống tranh chấp trạng thái

a. Khái niệm: Giả sử có hai cặp tổ hợp mã viết dưới dạng nhị phân (α, β) và (γ, σ) có độ dài L . Ta gọi hai tổ hợp mã này là không liên quan với nhau, nếu với một i nào đó ($1 \leq i \leq L$) thì bit thứ i của tổ hợp mã nhận giá trị 1/0 trong tổ hợp mã (α, β) thì trong tổ hợp mã (γ, σ) nó có giá trị 0/1. Trong trường hợp ngược lại, các tổ hợp mã này gọi là có liên quan.

Người ta đã chứng minh rằng, trong ô tô máy hữu hạn hiện tượng tranh chấp trạng thái sẽ được khử bỏ nếu các trạng thái trong của nó được mã hoá bằng các tổ hợp mã có độ dài hữu hạn, nhưng đối với bất kỳ hai bước chuyển trạng thái $(A_m \rightarrow A_c)$ và $(A_k \rightarrow A_l)$ nào, với $A_c \neq A_l$, dưới tác động của cùng một trạng thái vào X , thì những cặp tổ hợp mã tương ứng phải là không liên quan.

Trên cơ sở điều đã chứng minh trên đây, người ta đã xây dựng mã chống tranh chấp trạng thái bằng cách lần lượt xét tất cả các chuyển biến trạng thái dưới tác động dù chỉ là của một trạng thái vào chung, đã gây nên các chuyển biến đó, rồi gán cho các bit của các tổ hợp mã những giá trị sao cho các cặp tổ hợp mã đó là không liên quan.

Thuật toán mã hoá thực hiện theo cách từng bước làm cho các tổ hợp mã trở thành không liên quan. Trong quá trình thực hiện, nếu bit nào của tổ hợp mã chưa được xác định ta ghi dấu bất định “-”. Giả sử ta cần làm cho trở thành không liên quan hai cặp tổ hợp mã dùng để mã hoá bốn trạng thái $(A_m \rightarrow A_c)$, $(A_k \rightarrow A_l)$ của ô tô máy S là (α, β) và (γ, σ) .

Thuật toán này thực hiện như sau:

1. Đặt $i = 0$;
2. Nếu $i = 0$, thực hiện bước 8, nếu $i \neq 0$, thực hiện bước 3;
3. Nếu với r nào đó, ($1 \leq r \leq i$) các giá trị của bit thứ r của bốn tổ hợp mã $(\alpha, \beta, \gamma, \sigma)$ tạo thành tập giá trị 0001 hoặc 1100 thì chuyển sang bước 9, nếu không thì thực hiện bước 4.

4. Nếu với giá trị r , ($1 \leq r \leq i$), các giá trị của bit thứ r của bốn tổ hợp mã (α , β , γ , σ) tạo thành một trong các tổ hợp mã sau đây:

- 0 1 1	- - 1 1	- - - 1
0 - 1 1	0 - - 1	0 - - -
0 0 - 1	0 0 - -	- 0 - -
0 0 1 -	- 0 - 1	- - 1 -
- 0 1 -	0 - 1 -	- - - -

thì thực hiện bước 5, nếu không thì thực hiện bước 6.

5. Điền tiếp vào những giá trị của bit r của bốn tổ hợp mã (α , β , γ , σ) sao cho giá trị của chúng ở các bit đó tạo thành giá trị 0011. Chuyển sang bước 9.

6. Nếu r nào đó, ($1 \leq r \leq i$), các giá trị của bit thứ r của các tổ hợp mã (α , β , γ , σ) tạo thành một trong các tổ hợp mã sau đây:

- 1 0 0	- - 0 0	- - - 0
1 - 0 0	1 - - 0	1 - - -
1 1 - 0	1 1 - -	- 1 - -
1 1 0 -	- 1 - 0	- - 0 -
- 1 0 -	1 - 0 -	

thì chuyển sang bước 7. Nếu không, thực hiện bước 8.

7. Điền tiếp vào những giá trị không xác định của bit thứ r của (α , β , γ , σ) sao cho những giá trị của bit này tạo thành 1100. Chuyển sang bước 9.

8. Tăng thêm một bit bất định nữa cho các tổ hợp mã rồi chuyển sang bước 4.

9. Cặp mã cho các bước chuyển ($A_m \rightarrow A_n$), ($A_k \rightarrow A_l$) là không liên quan.

Độ dài của bộ mã nhận được bằng algôrit này thường không phải là tối thiểu. Do đó sau khi nhận được có thể lại cần thực hiện đơn giản độ dài của tổ hợp mã.

Cách làm như sau:

Bỏ một trong các bit của tổ hợp mã. Kết quả là một số cặp tổ hợp mã có thể trở thành liên quan. Tiếp tục áp dụng algôrit trên đây đối với các cặp chuyển trạng thái này. Sau đó lại bỏ tiếp một bit nữa và lại dùng algôrit trên đây cho tới khi nào độ dài của mã không giảm được nữa thì dừng.

Nếu kết quả của algôrit còn có những bit không xác định “-” thì có thể điền vào các bit này giá trị “0” hay “1” tùy ý.

Thí dụ 9.6: Mã hoá chống tranh chấp trạng thái cho ô tô máy trên hình 9.20 dưới đây.

Căn cứ vào bảng chuyển đổi trạng thái, ta xếp các bước chuyển trạng thái có chung tập giá trị biến vào thành các cột như sau:

X_1	X_2	X_3
$(a_1 \rightarrow a_2)$	$(a_1 \rightarrow a_1)$	$(a_2 \rightarrow a_5)$
$(a_2 \rightarrow a_2)$	$(a_2 \rightarrow a_3)$	$(a_3 \rightarrow a_7)$
$(a_3 \rightarrow a_1)$	$(a_3 \rightarrow a_3)$	$(a_5 \rightarrow a_5)$
$(a_4 \rightarrow a_1)$	$(a_4 \rightarrow a_1)$	$(a_7 \rightarrow a_7)$
$(a_5 \rightarrow a_6)$	$(a_5 \rightarrow a_3)$	
$(a_6 \rightarrow a_6)$		

Vì hai cặp $(a_1 \rightarrow a_2)$ và $(a_2 \rightarrow a_2)$ có cùng trạng thái chuyển đến nên không cần phải xét. Bắt đầu áp dụng algôrit từ cặp $(a_1 \rightarrow a_2)$ với $(a_3 \rightarrow a_4)$ vì đây là cặp đầu tiên ($i = 0$) nên với biến $r = 1$ ta có bảng mã đầu tiên như trên hình 9.21a.

Cặp tiếp theo cần xét là cặp $(a_1 \rightarrow a_2)$ và $(a_4 \rightarrow a_4)$. Đối với cặp này tự động đã có 0011, tức chúng đã là không liên quan.

Λ	X_1	X_2	X_3
1	2	1	-
2	2	3	5
3	4	3	7
4	4	1	-
5	6	3	5
6	6	-	-
7	-	-	7

Hình 9.20. Ôtômat dùng mã hoá

Cặp $(a_1 \rightarrow a_2)$ với $(a_5 \rightarrow a_6)$: Để có dạng không liên quan chỉ việc điền giá trị 1 ứng với a_5 và a_6 , tức là tạo thành 0011 - hình 9.21b.

Cặp $(a_1 \rightarrow a_2)$ và $(a_6 \rightarrow a_6)$ cũng tự động trở thành không liên quan (có dạng 0011). Tương tự các cặp $(a_2 \rightarrow a_3)$ với tất cả bốn cặp còn lại cũng là không liên quan một cách tự động, vì bit r_1 tạo thành 0011.

A	r ₁	A	r ₁	A	r ₁	r ₂	A	r ₁	r ₂	r ₃
a ₁	0	a ₁	0	a ₁	0	-	a ₁	0	1	1
a ₂	0	a ₂	0	a ₂	0	-	a ₂	0	0	0
a ₃	1	a ₃	1	a ₃	1	0	a ₃	1	0	0
a ₄	1	a ₄	1	a ₄	1	0	a ₄	1	0	1
a ₅	-	a ₅	1	a ₅	1	1	a ₅	1	1	0
a ₆	-	a ₆	1	a ₆	1	1	a ₆	1	1	-
a ₇	-	a ₇	-	a ₇	-	-	a ₇	-	-	-

a) b) c) d)

Hình 9.21. Bảng các tổ hợp mã thực hiện theo algôrit

Các cặp tiếp theo cần xét là $(a_3 \rightarrow a_4)$ và $(a_5 \rightarrow a_6)$. Vì trong hình 9.21b có dạng 1111 nên hai cặp này là liên quan. Vì vậy cần quay lại thực hiện điểm 4. Tăng thêm một bit và giả thiết như ở hình 9.21c, với hai bit r_1 và r_2 toàn bộ các cặp chuyển biến trạng thái dưới tác động của trạng thái vào X_1 đều trở thành không liên quan.

Tiến hành làm tương tự đối với các cặp chuyển trạng thái dưới tác động của các trạng thái vào X_2 và X_3 ta được như ở hình 9.21d và 9.22a.

A	r ₁	r ₂	r ₃	r ₄	A	r ₂	r ₃	r ₄	A	r ₂	r ₃	r ₄	r ₅
a ₁	0	1	1	-	a ₁	1	1	-	a ₁	1	1	0	0
a ₂	0	0	0	0	a ₂	0	0	0	a ₂	0	0	0	0
a ₃	1	0	0	1	a ₃	0	0	1	a ₃	0	0	1	-
a ₄	1	0	1	-	a ₄	0	1	-	a ₄	0	1	1	-
a ₅	1	1	0	0	a ₅	1	0	0	a ₅	1	0	0	1
a ₆	1	1	-	-	a ₆	1	-	-	a ₆	1	-	-	1
a ₇	-	0	-	1	a ₇	0	-	1	a ₇	0	-	1	-

a) b) c)

Hình 9.22. Tiếp tục hoàn thiện việc mã hoá

Đơn giản bộ mã

Để đơn giản, đầu tiên ta bỏ bớt bit r_1 của bộ mã (hình 9.22b). Sau đó lặp lại quá trình thực hiện algôrit để làm sao cho các tổ hợp mã thuộc các cặp chuyển trạng thái trở thành không liên quan. Ta thấy rằng cặp $(a_1 \rightarrow a_2)$ và $(a_5 \rightarrow a_6)$ lúc này có các mã tương ứng là 11 - ; 000; 100 và 1 - -. Như vậy không ở một bit i nào, $i = 2, 3, 4$, thoả mãn

điều kiện không liên quan. Các cặp chuyển đổi trạng thái còn lại đều thoả mãn điều kiện không liên quan hoặc có thể điền vào các vị trí bất định những giá trị thích hợp làm cho chúng trở thành không liên quan. Để làm cho cặp $(a_1 \rightarrow a_2)$ và $(a_5 \rightarrow a_6)$ trở thành không liên quan, áp dụng algôrit trên đây, ta thêm bit r_3 vào bộ mã và gán cho a_1 và a_2 giá trị logic "0", còn cho a_5 và a_6 giá trị logic "1" (hình 9.22c). Đồng thời ta điền giá trị của bit r_1 cho $a_1 = 0$ và cho $a_1 = 1$ để cho cặp $(a_1 \rightarrow a_2)$ và $(a_5 \rightarrow a_6)$ trở thành không liên quan. Như vậy số bit của tổ hợp mã lại vẫn là 4. Bây giờ bỏ r_2 , ta được hình 9.23a.

A	r_3	r_1	r_5
a_1	1	0	0
a_2	0	0	0
a_3	0	1	-
a_4	1	1	-
a_5	0	0	1
a_6	-	0	1
a_7	-	1	-

a)

A	r_3	r_4	r_5
a_1	1	0	0
a_2	0	0	0
a_3	0	1	0
a_4	1	1	0
a_5	0	0	1
a_6	1	0	1
a_7	0	1	1

b)

Hình 9.23. Mã sau khi bỏ bit r_2

Sau khi điền các bit bất định bằng các giá trị cụ thể thích hợp như hình 9.23b thì mã tất cả các cặp chuyển đổi trạng thái đều trở nên không liên quan.

Nếu tiếp tục đơn giản nữa thì không thể được, vì để mã hoá 7 trạng thái trong của ô tô mét ít nhất cũng phải sử dụng bộ mã có số bit bằng 3.

9.4. XÁC ĐỊNH PHẦN TỬ NHỚ

Khi mã hoá trạng thái trong của ô tô mét, ta được số bit mã cần thiết. Mỗi bit mã này là một biến trong của ô tô mét. Mỗi biến trong được thực hiện bằng một phần tử nhớ cơ bản, có thể nhớ một bit thông tin có giá trị "0" hoặc "1".

Trên thực tế, do những mục đích rất đa dạng về kỹ thuật, người ta thiết kế và chế tạo nhiều loại phần tử nhớ khác nhau về nguyên lý hoạt động. Đó là các loại trigơ (Flip - Flop). Chúng ta sẽ dành riêng chương 10 để nghiên cứu về bản chất cấu tạo và hoạt động của các trigơ cũng như phương pháp chuyển đổi giữa chúng với tư cách là một ô tô mét hữu hạn. Trong mục này chúng ta chỉ đề cập tới một số vấn đề về lựa chọn các trigơ này trong khi thiết kế ô tô mét hữu hạn.

Các trigơ thông thường có 4 loại: trigơ T, trigơ D, trigơ JK và trigơ RS. Đó là các loại trigơ không đồng bộ, nghĩa là các trigơ này làm việc không có xung nhịp, làm việc như một ô tômát không đồng bộ.

Trong thiết kế ô tômát, nếu sử dụng trigơ không đồng bộ thì nhất thiết phải áp dụng mã hoá chống tranh chấp trạng thái. Ngoài ra cần nhớ rằng, các trigơ loại D và loại RS có thể làm việc với các trạng thái tín hiệu vào X_i là xung hay thế cũng được. Ngược lại, các trigơ loại T và loại JK chỉ có thể làm việc đúng với các trạng thái tín hiệu vào X_i là xung. Những đặc điểm này sẽ được làm rõ trong chương 10 khi nghiên cứu chi tiết về các phần tử nhớ cơ bản.

Để khắc phục hiện tượng tranh chấp trạng thái của ô tômát, người ta thiết kế các loại trigơ làm việc có tín hiệu nhịp. Đó là các trigơ đồng bộ. Các trigơ loại CT, CD, CJK và CRS là các loại trigơ đồng bộ, làm việc theo nhịp của các xung đồng bộ Clock (CLK). Vì bản thân tín hiệu nhịp là loại tín hiệu xung, có thời gian tồn tại rất nhỏ hoặc là sườn của các tín hiệu thế. Vì vậy khi các trạng thái vào là thế thì xung nhịp sẽ làm cho ô tômát hoạt động như làm việc với các trạng thái vào là xung. Vì vậy sẽ không có hiện tượng tranh chấp trạng thái. Do đó khi thiết kế, nếu chọn phần tử nhớ để thực hiện các biến trong là các trigơ đồng bộ thì không nhất thiết phải mã hoá trạng thái trong của ô tômát bằng mã chống tranh chấp trạng thái. Điều này làm cho việc thiết kế trở nên đơn giản và mạch cũng có phần đơn giản hơn. Tuy vậy, giá phải trả lại là việc phải dùng nguồn xung đồng bộ và về nguyên tắc, tốc độ làm việc của ô tômát sẽ bị hạn chế bởi tần số của xung đồng bộ.

Ngoài ra chúng ta cần biết rằng, các trigơ có tính chất chung là cùng là ô tômát loại Moore, có một đầu ra "a", có thể ghi nhận một trong hai giá trị logic là "0" và "1" và có thể có 4 bước chuyển ở đầu ra "a": 0 - 0; 0 - 1; 1 - 0 và 1 - 1. Nhưng về cấu tạo và phương thức hoạt động khác nhau. Trigơ T (CT) và trigơ D (CD) chỉ có một đầu vào tín hiệu. Trong khi đó hai loại trigơ JK (CJK) và RS (CRS) có hai đầu vào tín hiệu. Do đó sự chuyển biến trạng thái "a" của chúng ở đầu ra phụ thuộc vào trạng thái vào của các tín hiệu này cũng khác nhau. Để phục vụ cho việc thiết kế các ô tômát hữu hạn, trên hình 9.24a, b, c và d cho ký hiệu khái quát và bảng kích thích của các loại trigơ nói trên. Bảng kích thích thể hiện, để đầu ra "a" chuyển trạng thái theo bốn bước chuyển: 0 - 0; 0 - 1; 1 - 0 và 1 - 1 thì đầu vào D, T, JK và RS của các loại trigơ đó phải có trạng thái như thế nào.

Từ các bảng kích thích này ta có nhận xét sau:

- Trigơ D là loại trigơ mà trạng thái ra "a" mới hoàn toàn giống trạng thái vào D, chỉ có điều nó bị giữ chậm khi truyền từ đầu vào tới đầu ra của trigơ. Do đó đầu vào của trigơ này có ký hiệu là D, viết tắt của từ tiếng Anh "Delay - giữ chậm".

- Trơ RS có đặc điểm là hai đầu R và S không bao giờ được đồng thời bằng 1. Nếu trong quá trình làm việc của trơ RS mà để hai đầu R và S đồng thời bằng 1 thì sẽ đưa trơ về trạng thái bất định. Đó là điều cần tránh.

- Trơ T là loại, nếu trạng thái vào T là tín hiệu thế và tích cực, tức T luôn bằng 1 thì trạng thái đầu ra luôn thay đổi từ trạng thái 0 sang trạng thái 1 và ngược lại. Đó chính là dao động ở đầu ra. Vì vậy trơ này chỉ làm việc với trạng thái vào là trạng thái xung.

- Trơ JK là loại mà nếu đầu J và đầu K đồng thời bằng 1 thì nó sẽ tạo nên dao động ở đầu ra tương tự như trơ T. Vì vậy trơ JK cũng chỉ làm việc với trạng thái vào tích cực là xung.



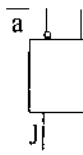
$a_c \rightarrow a_m$	D
0 - 0	0
0 - 1	1
1 - 0	0
1 - 1	1

a) Trơ D



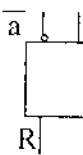
$a_c \rightarrow a_m$	T
0 - 0	0
0 - 1	1
1 - 0	1
1 - 1	0

b) Trơ T



$a_c \rightarrow a_m$	JK
0 - 0	0-
0 - 1	1-
1 - 0	-1
1 - 1	-0

c) Trơ JK



$a_c \rightarrow a_m$	RS
0 - 0	0-
0 - 1	10
1 - 0	01
1 - 1	-0

d) Trơ RS

Hình 9.24. Các loại trơ và bảng kích thích của chúng

Ngoài những điều đã trình bày trên đây, cần lưu ý rằng bảng kích thích của các trơ CD, CT, CJK và CRS hoàn toàn giống bảng kích thích của các trơ không đồng bộ cùng loại, nhưng với điều kiện là tín hiệu đồng bộ C (hay CLK) bằng 1. Hơn nữa, khi dùng các loại trơ đồng bộ mà tín hiệu vào X_i là loại xung thì nhất thiết tín hiệu vào xung X_i này cần được đồng bộ với tín hiệu đồng bộ của các trơ để tránh mất thông tin.

Trên đây là một số đặc trưng của các loại phần tử nhớ cơ bản thường dùng để thực hiện như là một biến trong của ô tô-mát. Mỗi biến trong được thực hiện bằng một trơ. Về nguyên tắc, để thực hiện một biến trong có thể sử dụng bất kỳ loại trơ nào. Tuy vậy, tùy thuộc vào loại trơ được dùng, độ phức tạp của mạch và một số yêu cầu đảm bảo sẽ khác nhau, mặc dù chức năng của ô tô-mát vẫn được đảm bảo.

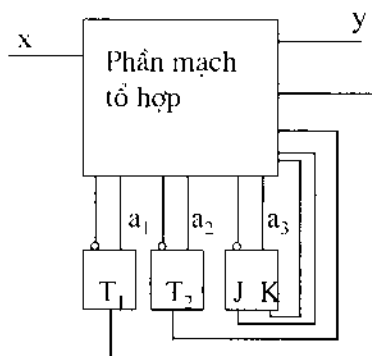
Những điều vừa trình bày trên đây là cơ sở để lựa chọn và xác định các phần tử nhớ dùng thực hiện các biến trong của ô tô-mát.

9.5. NHẬN CÁC HÀM KÍCH THÍCH VÀ CÁC HÀM RA

9.5.1. Xây dựng cấu trúc mạch sau khi đã mã hoá các trạng thái

Sau khi đã mã hoá các trạng thái vào $\{X_i\}$, $i = \overline{1, N}$; các trạng thái ra $\{Y_j\}$, $j = \overline{1, M}$ và các trạng thái trong $\{A_s\}$, $s = \overline{1, P}$ và sau khi đã lựa chọn các phần tử nhớ dùng thực hiện phần nhớ của ô tô-mát, chúng ta hoàn toàn đã xác định xong phần tử nhớ. Điều này có nghĩa là số lượng các biến vào, số lượng các biến ra, số lượng các biến trong và số lượng các hàm kích thích cho các phần tử nhớ đã được xác định. Từ đây chúng ta có thể chi tiết hoá thêm một bước mạch cấu trúc của ô tô-mát.

Trên hình 9.25 là thí dụ về mạch cấu trúc sau khi đã thực hiện mã hoá và chọn các phần tử nhớ. Trong đó số biến vào được giả thiết là $n = 2$, tức có hai đầu vào x_1 và x_2 ; số biến ra $m = 1$, tức có một đầu ra y và số biến trong $p = 3$, tức có ba biến trong a_1 , a_2 , a_3 . Để thực hiện ba biến trong này giả sử ta chọn ba trigơ: Để thực hiện a_1 ta dùng trigơ T_1 , để thực hiện a_2 ta dùng trigơ T_2 và để thực hiện a_3 ta dùng trigơ JK.



Hình 9.25. Cấu trúc ô tô-mát sau khi đã mã hoá và chọn phần tử nhớ

Như vậy, phần duy nhất chưa được xác định là phần mạch tổ hợp. Để thực hiện phần tổ hợp đòi hỏi phải xác định được các hàm ra:

$$y_j = f_j(x_i, a_s), j = \overline{1, m} \quad (9.1)$$

và các hàm kích thích cho các phần tử nhớ q_k :

$$q_{kl} = f_k(x_i, a_s); s = \overline{1, p} \quad (9.2)$$

$$l = \overline{1, r}; k = \overline{1, p}$$

trong đó r là số đầu vào kích thích của trigơ thứ k .

Trong trường hợp hình 9.25, hàm ra là hàm y ; các hàm kích thích là các hàm T_1 , T_2 và hai hàm J và K . Trong đó:

$$y = f_1(x, a_1, a_2, a_3)$$

$$T_1 = f_2(x, a_1, a_2, a_3)$$

$$T_2 = f_3(x, a_1, a_2, a_3)$$

$$J = f_4(x, a_1, a_2, a_3)$$

$$K = f_5(x, a_1, a_2, a_3)$$

Một cách khái quát, số biến của các hàm này luôn được coi là bằng tổng số biến vào và số biến trong $(n + p)$ và như vậy, khi xuất phát từ một trạng thái trong ban đầu nào đó $(a^*_1 a^*_2 \dots a^*_p)$, $a^*_s = (0/1)$, dưới tác động của một trạng thái vào $(x^*_1 x^*_2 \dots x^*_n)$ nào đó, ô tômát sẽ cho ra một trạng thái ra $(y^*_1 y^*_2 \dots y^*_j)$ tương ứng, đồng thời tạo ra các tín hiệu kích thích cho các phần tử nhớ $(q^*_{11} q^*_{21} \dots q^*_{pn})$. Các tín hiệu kích thích này tác động vào các đầu vào của các phần tử nhớ và do đó làm các trạng thái ra của các phần tử nhớ này thay đổi, tạo ra một trạng thái mới của ô tômát.

Từ các mối quan hệ trên đây chúng ta thấy rằng, để xác định được các hàm ra y_j và các hàm kích thích q_{ki} của các phần tử nhớ, cần sử dụng bảng chuyển đổi và bảng ra đã mã hoá và bảng kích thích của các phần tử nhớ (các trigơ) đã được xác định để thực hiện phần có nhớ của ô tômát.

9.5.2. Phương pháp rút ra các hàm ra

Phương pháp rút ra các hàm ra và các hàm kích thích của ô tômát gồm các bước sau đây:

1. Mã hoá các trạng thái vào $\{X_i\}$, các trạng thái ra $\{Y_j\}$ và các trạng thái trong $\{A_s\}$ theo yêu cầu của bước mã hoá.

2. Căn cứ vào bảng ra đã mã hoá của ô tômát, biểu diễn các hàm $y_j, j = \overline{1, m}$ theo mối quan hệ phụ thuộc của chúng vào các biến vào x_i và các biến trong a_s bằng cách dùng bảng giá trị thực hay bảng Karnaugh/hoặc bảng đối xứng.

Đối với các tập mã không được sử dụng để mã hoá các trạng thái các hàm $y_j, j = \overline{1, m}$ tương ứng được coi là có giá trị hàm bất định.

3. Rút ra các hàm và đơn giản chúng. Trường hợp biểu diễn hàm dưới dạng bảng Karnaugh, việc rút ra các hàm này đồng thời cũng là đơn giản chúng. Trong trường hợp cần thiết, có thể tiến hành đơn giản hệ hàm để đảm bảo sự đơn giản chung của hệ thống được thiết kế.

Để nhận các hàm kích thích của các phần tử nhớ của ô tômát, cần sử dụng bảng chuyển đổi trạng thái đã mã hoá và các bảng kích thích của các loại phần tử nhớ đã được dùng.

Phương pháp nhận gồm các bước sau đây:

1. Từ bảng chuyển đổi trạng thái đã mã hoá của ô tômát và các bảng kích thích của các phần tử nhớ, ta suy ra giá trị tương ứng cần thiết của các hàm q_{ki} đối với từng bước chuyển và biểu diễn chúng dưới dạng các bảng giá trị thực hoặc bảng kế cận hay bảng đối xứng của các biến vào $x_i, i = \overline{1, n}$ và các biến trong $a_s, s = \overline{1, p}$.

Đối với các tổ hợp mã của các biến mã $a_1, a_2, \dots, a_n$ không dùng để mã hoá các trạng thái trong thì giá trị tương ứng của các hàm q_{ki} là bất định. Nếu biểu diễn dưới dạng bảng giá trị thực thì mỗi hàng của bảng tương ứng với một bước chuyển trạng thái của ô tô máy. Nếu biểu diễn dưới dạng bảng kế cận hoặc bảng đối xứng thì số ô của bảng bằng $N \times P$. Trong đó N là số lượng trạng thái vào: $N = 2^n$ và P là số lượng trạng thái trong: $P = 2^p$.

2. Căn cứ vào kết quả bước thứ nhất để rút ra các hàm kích thích cho các phần tử nhớ của ô tô máy dưới dạng đã được đơn giản.

3. Căn cứ vào bảng ra đã mã hoá của ô tô máy, lập bảng giá trị thực cho các hàm ra $y_j, j = 1, m$. Ta cũng có thể biểu diễn các hàm ra này dưới dạng bảng kế cận hay bảng đối xứng. Cần chú ý rằng, dù biểu diễn dưới dạng nào thì các hàm y_j này vẫn là hàm của các biến vào x_i và các biến trong a_s .

4. Rút ra các hàm ra y_j dưới dạng đã đơn giản hoá.

Sau khi rút ra được các hàm ra và các hàm kích thích, ta hoàn toàn xác định cấu trúc logic của ô tô máy và bước tiếp theo chỉ còn là bước thực hiện kỹ thuật cụ thể.

Dưới đây là thí dụ minh hoạ cho phương pháp rút ra các hàm kích thích và các hàm ra.

Thí dụ 9.7: Cho ô tô máy ở hình 9.26. Hãy mã hoá, xác định các phần tử nhớ và nhận các hàm ra và các hàm kích thích.

Λ	X_0	X_1	X_2
Λ_1	Λ_2/Y_3	Λ_3/Y_4	Λ_2/Y_2
Λ_2	$- / Y_4$	Λ_1/Y_3	$\Lambda_3 / .$
Λ_3	Λ_1/Y_2	$- / Y_1$	Λ_3/Y_3

Hình 9.26. Ô tô máy dùng rút ra các hàm kích thích và hàm ra

a. Mã hoá trạng thái

Vì có ba trạng thái vào X_0, X_1 và X_2 nên số bit mã cần thiết là $n = 2$. Ký hiệu hai bit mã này là x_0 và x_1 . Trong đó việc gán các tổ hợp mã cho các trạng thái vào cho trên hình 9.27a. Ô tô máy có ba trạng thái trong Λ_1, Λ_2 và Λ_3 nên ít nhất phải dùng hai bit mã để mã hoá các trạng thái này. Ký hiệu các bit mã này là a_1 và a_2 . Thí dụ về việc gán các tổ hợp mã cho các trạng thái trong cho trên hình 9.27b.

Để mã hoá bốn trạng thái ra Y_1, Y_2, Y_3 và Y_4 , số bit mã cần thiết là 2. Ký hiệu các bit mã này là y_1 và y_2 và việc gán các tổ hợp mã cho các trạng thái ra cho trên hình 9.27c.

Cuối cùng ta có bảng chuyển đổi trạng thái và bảng ra cho trên hình 9.28a và b tương ứng.

X	x_0 x_1	A	a_1 a_2	Y	y_1 y_2
X_0	0 0	A_1	0 0	Y_1	0 0
X_1	0 1	A_2	0 1	Y_2	0 1
X_2	1 0	A_3	1 1	Y_3	1 0
-	1 1	-	1 0	Y_4	1 1

a) b) c)

Hình 9.27. Mã hoá các trạng thái của ô tô mât

$A \backslash x_0 x_1$	00	01	10	11
00	01	11	01	--
01	--	00	11	--
11	00	--	11	--
10	--	--	--	--

a) Bảng chuyển đổi trạng thái

$A \backslash x_0 x_1$	00	01	10	11
00	10	11	01	--
01	11	10	--	--
11	01	00	10	--
10	--	--	--	--

b) Bảng ra

Hình 9.28. Bảng chuyển đổi trạng thái và bảng ra đã mã hoá

b. Lập bảng giá trị thực

Giả sử ta dùng hai trigơ RS. Như vậy bảng giá trị thực cho hàm ra y_1, y_2 và bảng giá trị thực cho các hàm kích thích S_1, R_1, S_2 và R_2 cho trên hình 9.29a, b. Trong đó để có giá trị điền vào bốn cột R_1, S_1, R_2 và S_2 ta phải căn cứ vào các bước chuyển $a_{1c} \rightarrow a_{1m}$ và $a_{2c} \rightarrow a_{2m}$ ở cột thứ hai và ba của bảng hình 9.29b, đồng thời đối chiếu với bảng hoạt động của trigơ RS. Hình 9.30 là biểu diễn các hàm này lên bảng kế cận.

c. Rút ra các hàm $y_1, y_2, S_1, R_1, S_2, R_2$

Từ các bảng kế cận như ở hình 9.30, chúng ta dễ dàng rút ra các hàm $y_1, y_2, S_1, R_1, S_2, R_2$ như sau:

$$y_1 = a_2 x_0 + \bar{a}_1 \bar{x}_0$$

$$y_2 = \bar{a}_1 x_0 + \bar{a}_2 x_1 + a_2 \bar{x}_0 \bar{x}_1$$

$$S_1 = a_2 x_0 + \bar{a}_2 x_1$$

$$R_1 = a_1 \bar{x}_0$$

$$S_2 = \bar{a}_2$$

$$R_2 = a_1 \bar{x}_0 + a_2 x_1$$

Mạch cấu trúc thực hiện các hàm vừa rút ra trên đây cho trên hình 9.31.

x_0	x_1	a_1	a_2	y_1	y_2
0	0	0	0	1	0
0	0	0	1	1	1
0	0	1	1	0	1
0	1	0	0	1	1
0	1	0	1	1	0
0	1	1	1	0	0
1	0	0	0	0	1
1	0	0	1	-	-
1	0	1	1	1	0

a)

x_0	x_1	a_{1c}	a_{2m}	a_{1m}	a_{2m}	R_1	S_1	R_2	S_2
0	0	0	0	0	1	-	0	0	1
0	0	0	1	-	-	-	0	0	-
0	0	1	1	0	0	1	0	1	0
0	1	0	0	1	1	0	1	0	1
0	1	0	1	0	0	-	0	1	0
0	1	1	1	-	-	1	0	1	0
1	0	0	0	0	1	-	0	0	1
1	0	0	1	1	1	0	1	0	-
1	0	1	1	1	1	0	-	0	-

b)

Hình 9.29. Bảng giá trị thực của các hàm

	x_0	x_1	y_1
a_1	-	-	-
a_2	1	-	0
-	-	1	1
0	-	1	1

	x_0	x_1	y_2
a_1	-	-	-
a_2	0	-	1
-	-	0	1
1	-	1	0

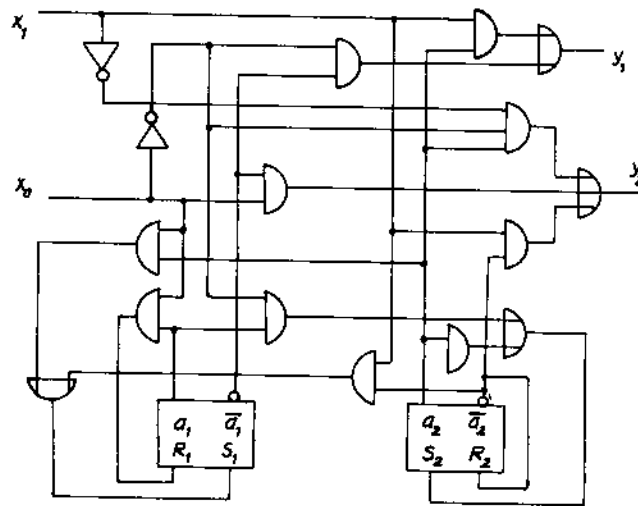
	x_0	x_1	R_1
a_1	-	-	-
a_2	0	-	1
-	-	-	-
0	-	0	-

	x_0	x_1	S_1
a_1	-	-	-
a_2	-	-	0
1	-	0	0
0	-	1	0

	x_0	x_1	R_2
a_1	-	-	-
a_2	0	-	1
-	-	1	0
0	-	0	0

	x_0	x_1	S_2
a_1	-	-	-
a_2	-	-	0
-	-	0	-
1	-	1	-

Hình 9.30. Biểu diễn các hàm ra và các hàm kích thích lên bảng kế cận



Hình 9.31. Mạch cấu trúc của ô-tô-mát của thí dụ 9.7

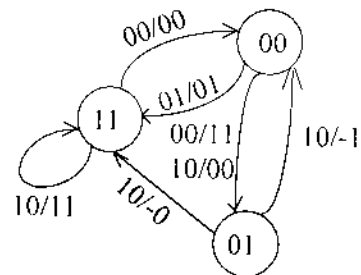
Thí dụ 9.8: Hãy thiết kế ô-tô-mát đã mã hoá cho dưới dạng đồ hình trạng thái (hình 9.32).

Từ ô-tô-mát đã mã hoá ở hình 9.32, ta thấy:

- Ô-tô-mát có hai biến vào, ký hiệu là x_1 và x_2 .
- Ô-tô-mát có hai biến ra, ký hiệu là y_1 và y_2 .
- Ô-tô-mát có hai biến trạng, ký hiệu là a_1 và a_2 .

Giả sử để thực hiện hai biến trạng của ô-tô-mát, ta dùng hai loại trigơ:

- Trigơ T để thực hiện a_1 ; trigơ JK để thực hiện a_2 .



Hình 9.32. Ô-tô-mát cho để thiết kế mạch

Bảng kích thích của hai loại trigơ này đối với trường hợp có hoặc không có xung nhíp như sau:

$a_c \rightarrow a_m$	T
0 $\rightarrow$ 0	0
0 $\rightarrow$ 1	1
1 $\rightarrow$ 0	1
1 $\rightarrow$ 1	0

a) Không nhíp

$a_c \rightarrow a_m$	T
0 $\rightarrow$ 0	0
0 $\rightarrow$ 1	1
1 $\rightarrow$ 0	1
1 $\rightarrow$ 1	0

b) Có nhíp: C = 1

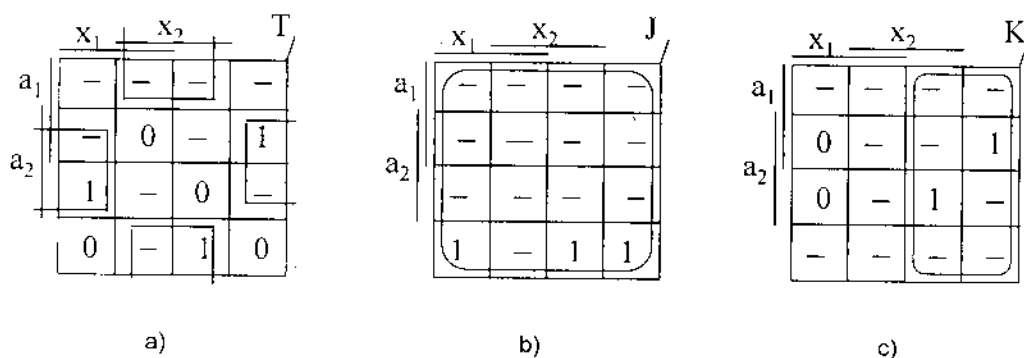
$a_c \rightarrow a_m$	JK
0 $\rightarrow$ 0	0 -
0 $\rightarrow$ 1	1 -
1 $\rightarrow$ 0	- 1
1 $\rightarrow$ 1	- 0

a) Không nhíp

$a_c \rightarrow a_m$	JK
0 $\rightarrow$ 0	0 -
0 $\rightarrow$ 1	1 -
1 $\rightarrow$ 0	- 1
1 $\rightarrow$ 1	- 0

b) Có nhíp: C=1

Căn cứ vào đồ hình chuyển đổi trạng thái và vào các bảng kích thích của các trigơ T và JK, ta xác định các hàm T, J và K. Sau khi đưa các hàm này lên bảng kế cận, ta có hình 9.33.



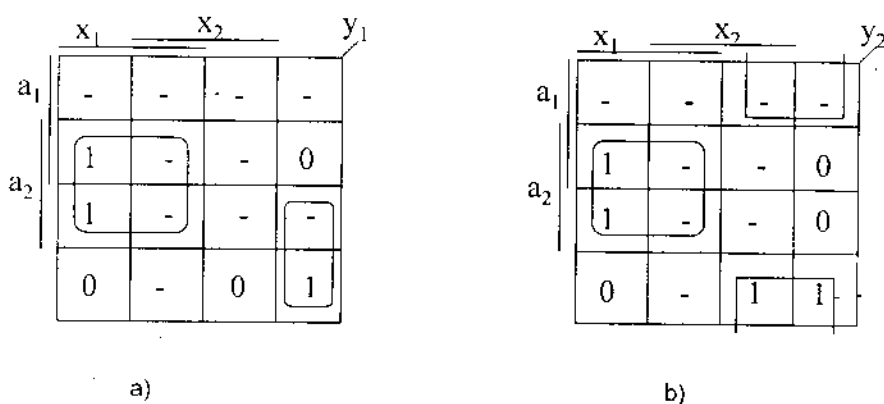
Hình 9.33. Bảng Karnaugh tìm các hàm kích thích

Từ đó ta rút ra:

$$T = x_2 \bar{a}_2 + \bar{x}_2 a_2$$

$$J = 1; K = \bar{x}_1$$

Để rút ra các hàm ra, ta có bảng ở hình 9.34.



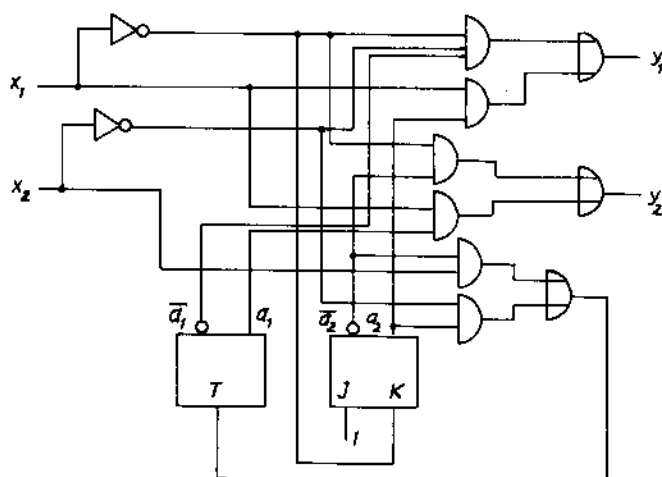
Hình 9.34. Bảng Karnaugh tìm các hàm ra

Từ đó ta có:

$$y_1 = x_1 a_2 + \bar{x}_1 \bar{x}_2 \bar{a}_1$$

$$y_2 = \bar{x}_1 \bar{a}_2 + x_1 a_2$$

Mạch thực hiện dùng một trigơ T và một trigơ JK cho trên hình 9.35.



Hình 9.35. Mạch thực hiện thiết bị số của thí dụ 9.8

Cần lưu ý rằng, trong các thí dụ trên đây chúng ta chỉ chú ý tới phương pháp rút ra các hàm ra và các hàm kích thích sau khi đã mã hoá mà không quan tâm tới các phương pháp mã hoá nhằm đạt được những chỉ tiêu chất lượng của việc thiết kế.

Thí dụ 9.9: Cần thiết kế một thiết bị số thực hiện chức năng như sau:

Thiết bị có hai đầu vào x_1, x_2 . Đó là các đầu vào tín hiệu xung. Trong đó cứ mỗi xung của đầu vào x_1 lại cộng thêm vào các số xung đã có trong thiết bị thêm 1 đơn vị. Ngược lại, xung x_2 lại làm giảm số xung của thiết bị đi 1 đơn vị. Các phép tăng và giảm của thiết bị đều theo môđun 4. Trạng thái trong của mạch luôn biểu thị số xung có trong nó.

Để thiết kế một ô tô-mát như vậy, ta có nhận xét như sau:

Vì cộng và trừ đều theo môđun 4 nên cần có 4 trạng thái trong để biểu thị 4 trạng thái. Ký hiệu các trạng thái này là A_0, A_1, A_2 và A_3 , trong đó:

A_0 là không có xung.

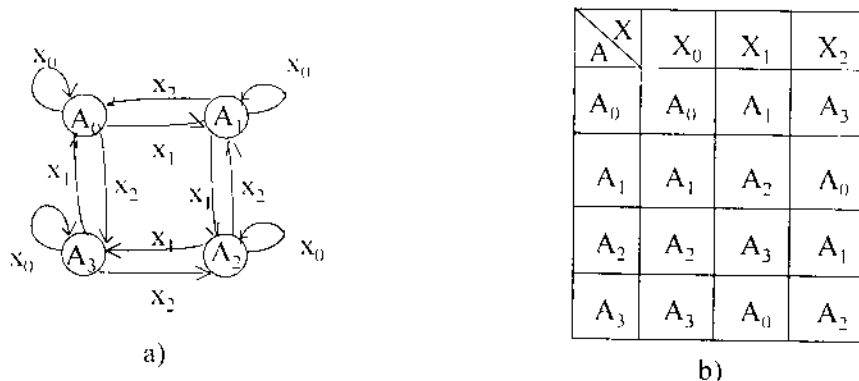
A_1 là có 1 xung.

A_2 là có 2 xung.

A_3 là có 3 xung.

Xung thứ 4 sẽ đưa trạng thái của thiết bị về A_0 .

Vì theo đầu bài, trạng thái trong của ô tô-mát đồng thời phản ánh số xung, nên ta dùng ngay trạng thái trong của nó là trạng thái ra, tức là ta có ô tô-mát loại Moore. Đồ hình trạng thái và bảng chuyển đổi trạng thái cho trên hình 9.36a và b.



Hình 9.36. Đồ hình trạng thái và bảng chuyển đổi trạng thái

Trong số các trạng thái vào của ôlômát, trạng thái X_0 là trạng thái thế, trạng thái thụ động, không làm thay đổi bất kỳ trạng thái trong nào. Trong khi đó hai trạng thái X_1 và X_2 là các trạng thái vào xung, tương ứng với các tập vào thực hiện phép cộng (X_1) và trừ (X_2).

Mã hoá trạng thái:

+ Với ba trạng thái vào ta cần ba tổ hợp mã. Vì vậy ít nhất phải có hai bit mã, tức hai biến vào, ký hiệu x_1 và x_2 . Tất nhiên với hai bit mã có thể tạo thành 4 tổ hợp mã. Vậy sẽ có một tổ hợp mã không dùng tới. Bộ tổ hợp mã nào là tùy người thiết kế.

+ Với bốn trạng thái trong ta cần hai bit mã để tạo thành bốn tổ hợp mã, gán cho bốn trạng thái trong.

+ Vì trạng thái trong đồng thời cũng là trạng thái ra nên không cần mã hoá khác đi nữa.

Sau khi gán các tổ hợp mã cho các trạng thái vào, trạng thái trong của ôlômát, đồ hình trạng thái và bảng chuyển đổi trạng thái đã mã hoá cho trên hình 9.37. Trong trường hợp này:

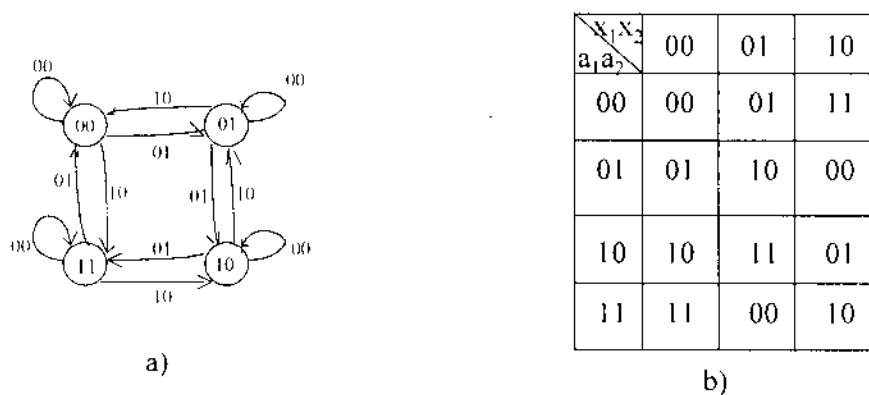
- Trạng thái vào $X_0 = 00$ ($x_1 = 0, x_2 = 0$) là trạng thái thụ động.

- Trạng thái $X_1 = 01$ ($x_1 = 0, x_2 = 1$) là trạng thái vào thực hiện phép cộng, hay còn gọi là thực hiện phép đếm xuôi.

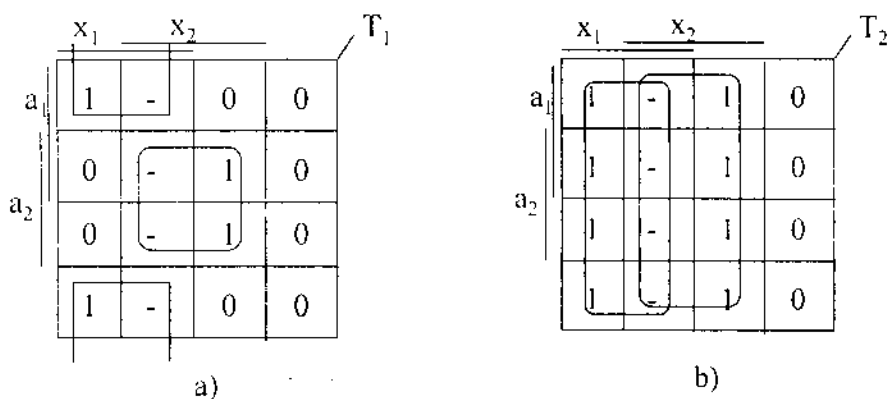
- Trạng thái $X_2 = 10$ ($x_1 = 1, x_2 = 0$) là trạng thái thực hiện phép trừ hay phép đếm ngược.

- Trạng thái $X_3 = 11$ ($x_1 = 1, x_2 = 1$) là trạng thái không dùng. Để có thể tận dụng điều kiện này trong đơn giản hàm, phải đảm bảo sao cho trạng thái này không được xuất hiện trên đầu vào của mạch.

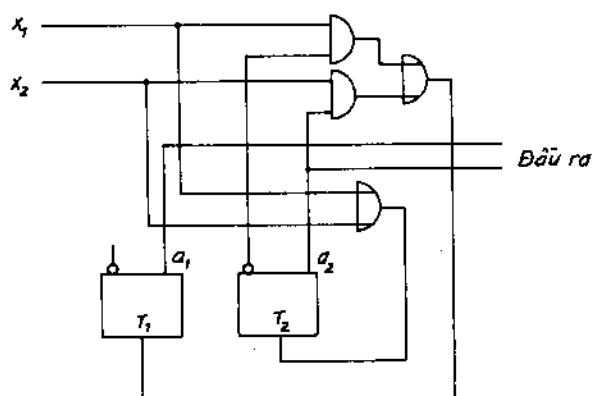
- Các trạng thái trong A_0, A_1, A_2 và A_3 được gán cho các tổ hợp mã tương ứng là: 00, 01, 10 và 11. Cần lưu ý rằng việc mã hoá kế cận chống tranh chấp trạng thái hoàn toàn có thể thực hiện được. Tuy vậy, trong trường hợp này các trạng thái vào tích cực X_1 và X_2 đều là các trạng thái xung, do đó không nhất thiết phải thực hiện mã hoá kế cận.



Hình 9.37. Đồ hình trạng thái và bảng chuyển đổi trạng thái đã mã hoá



Hình 9.38. Bảng Karnaugh xác định các hàm kích thích T_1 và T_2



Hình 9.39. Mạch thực hiện thiết bị số của thí dụ 9.9

+ Với hai bit mã mã hoá trạng thái trong, để thực hiện ô tô-mát, ta cần có hai tri-gơ. Giả sử ta dùng hai tri-gơ loại T (hoặc CT). Sử dụng bảng chuyển đổi trạng thái đã mã hoá hình 9.37 và bảng kích thích của tri-gơ T đã cho trong chương *Các phần tử nhớ cơ bản* có thể xác định được các hàm T_1 và T_2 như trên bảng Karnaugh (hình 9.38).

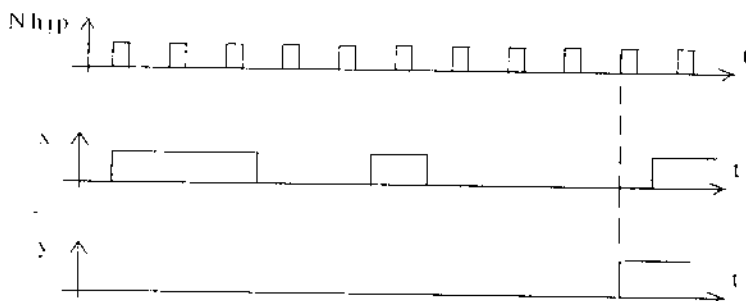
Từ đó xác định T_1 và T_2 :

$$T_1 = x_1 \bar{a}_2 + x_2 a_2 \quad \text{và} \quad T_2 = x_1 + x_2$$

Mạch thực hiện nhiệm vụ đã cho trình bày trên hình 9.39.

Thí dụ 9.10: Cần thiết kế một thiết bị phân biệt được khi nào khoảng trống của biến x (tức khoảng $x = 0$) lớn hơn bốn nhịp của xung nhịp. Nếu khoảng trống này bằng hoặc nhỏ hơn 4 nhịp thì tín hiệu ra $y = 0$, ngược lại thì $y = 1$.

Để thiết kế mạch này, ta lập giản đồ thời gian các tham số như ở hình 9.40.



Hình 9.40. Giản đồ thời gian các tham số của thiết bị

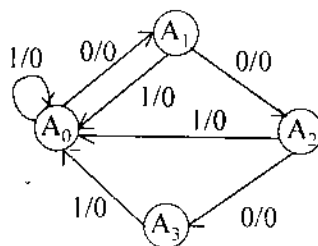
Từ nhiệm vụ thiết kế ta thấy rằng, khoảng trống cần phát hiện lớn hơn 4 nhịp. Vì vậy cần có một ô tô-mát có 4 trạng thái để ghi lại trạng thái của ô tô-mát tương ứng với 4 nhịp. Ký hiệu các trạng thái đó là: A_0, A_1, A_2, A_3 .

Theo điều kiện của đầu bài thì khi $x = 1$, mọi trạng thái sẽ chuyển về trạng thái ban đầu A_0 với $y = 0$. Còn tại trạng thái A_3 , tại nhịp tiếp theo nếu $x = 0$ thì ô tô-mát không thay đổi trạng thái, nhưng cho ra $y = 1$. Nếu $x = 1$ thì ô tô-mát chuyển về trạng thái A_0 .

Với các điều kiện như trên, bảng chuyển đổi trạng thái và đồ hình trạng thái của ô tô-mát cho trên hình 9.41.

$A \setminus x$	0	1
A_0	$A_1 / 0$	$A_0 / 0$
A_1	$A_2 / 0$	$A_0 / 0$
A_2	$A_3 / 0$	$A_0 / 0$
A_3	$A_3 / 1$	$A_0 / 0$

a)



b)

Hình 9.41. Bảng chuyển đổi và đồ hình trạng thái của ô tô-mát đã cho

Trên cơ sở đó ta rút ra các hàm sau:

$$y = a_1 a_2 \bar{x}$$

$$D_1 = a_1 \bar{x} + a_2 \bar{x} = \bar{x}(a_1 + a_2)$$

$$D_2 = a_1 \bar{x} + \bar{a}_2 \bar{x} = \bar{x}(a_1 + \bar{a}_2)$$

Mạch thực hiện cho trên hình 9.43.

BÀI TẬP CHƯƠNG 9

- Hãy tìm các cặp trạng thái tương đương của ô tô mât cho trên hình 9.44a, b dưới đây:

A \ X	X ₀	X ₁	X ₂	X ₃
1	(1),0	7	2	6
2	1	4	5	(2),1
3	1	(3),0	2	3
4	1	(4),0	2	3
5	1	4	6	(5),0
6	1	7	(6),0	8
7	1	(7),0	5	8
8	1	4	5	(8),1

a)

A \ X	X ₀	X ₁	X ₂
1	(1),b ₁	2	6
2	1	-	(2),b ₂
3	-	5	(3),b ₃
4	(4),b ₁	-	4
5	4	-	(5),b ₁
6	-	(6),0	1

b)

Hình 9.44

- Hãy tìm các lớp tương đương tối đa của ô tô mât ở hình 9.44a, b.
- Đơn giản các trạng thái trong của các ô tô mât Milly và Moore cho trên hình 9.45a, b:

A \ X	X ₀	X ₁	X ₂
1	(1),00	2	4
2	3	(2),10	5
3	(3),00	-	4
4	1	2	(4),10
5	1	-	(5),11

a)

Y	A \ X	X ₀	X ₁	X ₂	X ₃
0	1	(1)	7	-	2
0	2	3	-	-	(2)
0	3	(3)	4	-	7
1	4	5	(4)	-	-
0	5	(5)	6	-	-
1	6	1	(6)	-	7
0	7	3	-	-	(7)

b)

Hình 9.45

4. Áp dụng phương pháp tìm phủ tối thiểu của McClusky để đơn giản trạng thái trong của ô tô máy cho trên hình 9.46a và b.

A \ X	X ₀	X ₁	X ₂
1	6	-	(1),11
2	3	(2),10	1
3	6	2	(3),1-
4	6	-	(4),-0
5	(5),00	-	3
6	(6),00	2	4

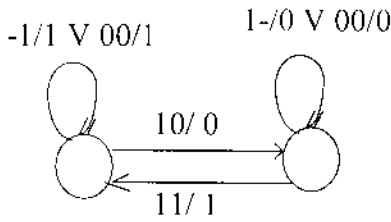
a)

A \ X	X ₀	X ₁	X ₂	X ₃
1	(1),000	4	11	-
2	1	(2),100	-	10
3	(3),000	2	5	-
4	7	(4),010	-	9
5	7	-	(5),001	10
6	3	(6),001	-	12
7	(7),000	6	8	-
8	1	-	(8),010	12
9	-	4	11	(9),000
10	-	2	5	(10),000
11	3	-	(11),100	9
12	-	6	8	(12),000

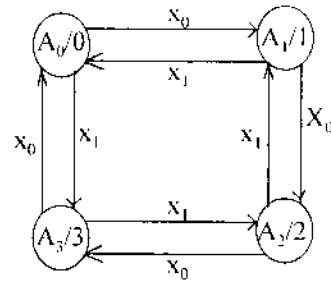
b)

Hình 9.46

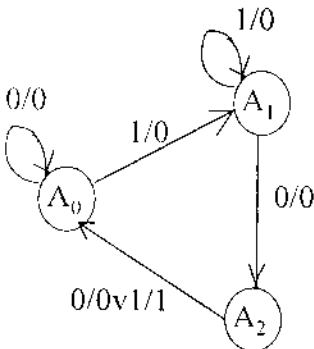
5. Hãy mã hoá kế cận các ô tô máy cho trên hình 9.47.



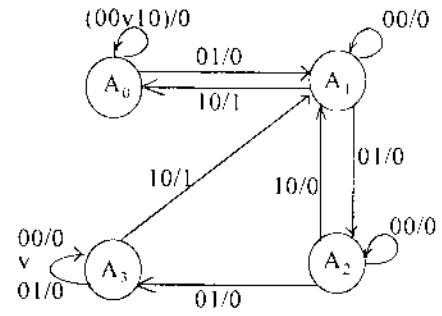
a)



b)



c)



d)

Hình 9.47

7. Thực hiện các ô tô măt ở hình 9.47, sau khi đã mã hoá chống tranh chấp trạng thái, dùng các loại trigơ:
- a) T
 - b) D
 - c) RS
 - d) JK
8. Mã hoá trạng thái trong của ô tô măt ở hình 9.47c và d theo cách gán sau:
- a) Mã 00 gán cho A_0
 - b) Mã 01 gán cho A_1
 - c) Mã 10 gán cho A_2
 - d) Mã 11 gán cho A_3 .

Hãy nhận các hàm kích thích và hàm ra khi:

- a) Biến a_1 thực hiện bằng trigơ D, a_2 thực hiện bằng trigơ RS
- b) Biến a_1 thực hiện bằng trigơ T, a_2 thực hiện bằng trigơ JK
- c) Biến a_1 thực hiện bằng trigơ JK, a_2 thực hiện bằng trigơ RS.

Nhận xét về độ phức tạp của mạch.

9. Mã hoá ô tô măt hình ở 9.47a, b một cách tùy ý (không nhất thiết dùng mã chống tranh chấp) và rút ra các hàm kích thích và hàm ra của chúng, nếu:
- a) Phần tử nhớ dùng trong mạch là trigơ T
 - b) Phần tử nhớ dùng trong mạch là trigơ D
 - c) Phần tử nhớ dùng trong mạch là trigơ JK
 - d) Phần tử nhớ dùng trong mạch là trigơ RS.

Chương 10

CÁC PHẦN TỬ NHỚ CƠ BẢN

10.1. KHÁI NIỆM VỀ PHẦN TỬ NHỚ CƠ BẢN

Phần tử nhớ cơ bản là một mạch điện tử có một đầu ra logic, ký hiệu là a , một hoặc hai đầu vào tín hiệu, ký hiệu chung là x_i , $i = \overline{1, 2}$. Dưới tác động của các tín hiệu đầu vào, tín hiệu ra a có thể có khả năng xảy ra một trong bốn loại chuyển đổi sau:

$$\begin{array}{lcl} a: & 0 & \longrightarrow 0 \\ & 0 & \longrightarrow 1 \\ & 1 & \longrightarrow 0 \\ & 1 & \longrightarrow 1 \end{array}$$

Khi không còn các tín hiệu vào thì a vẫn lưu giữ lại tín hiệu đã có dưới dạng bit có giá trị 0 hoặc 1.

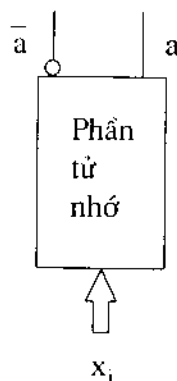
Như vậy, phần tử nhớ cũng là một ô tô-mát hữu hạn và là ô tô-mát hữu hạn loại Moore, đầu ra a vừa là biến trong, vừa là biến ra.

Ký hiệu khái quát của phần tử nhớ cơ bản cho ở hình 10.1.

Như vậy, phần tử nhớ cơ bản luôn luôn có hai trạng thái trong ($a = 0$ hoặc $a = 1$); có hai hoặc 4 trạng thái vào, tương ứng với loại có một đầu vào hay hai đầu vào. Hàm chuyển đổi trạng thái φ của nó dưới dạng tổng quát như sau:

$$A_m = \varphi(A_c, X_i)$$

trong đó A_m là trạng thái trong mới được hình thành trên cơ sở trạng thái trong cũ A_c dưới tác động của trạng thái vào X_i .



Hình 10.1. Mô hình phần tử nhớ cơ bản

Hàm φ khi đó sẽ là hàm hai hoặc ba biến, tùy theo số biến đầu vào x_i là một hoặc hai biến.

Như chúng ta đã biết, các phần tử nhớ cơ bản đã được sản xuất dưới dạng mạch tích hợp (IC). Bốn loại phần tử nhớ cơ bản là bốn loại trigơ (Flip - Flop) một đầu vào: D (C'D), T (C'T) và hai đầu vào: JK (C'JK), RS (CRS). Tuy vậy việc nghiên cứu nguyên tắc hình thành cấu trúc và nguyên lý hoạt động của chúng sẽ giúp chúng ta hiểu một cách đầy đủ tính năng, đặc trưng và cách sử dụng chúng một cách chính xác. Đồng thời qua nghiên cứu cấu trúc có thể dễ dàng tiến hành chuyển đổi từ một dạng trigơ này sang một dạng khác khi cần thiết, một việc mà trong thực tế cũng hay gặp phải.

Dưới đây chúng ta sẽ xét nguyên tắc hình thành cấu trúc, nguyên tắc hoạt động của các loại phần tử nhớ cơ bản này cũng như nguyên tắc chuyển đổi giữa chúng.

10.2. PHẦN TỬ NHỚ MỘT ĐẦU VÀO

10.2.1. Mô hình hàm chức năng của phần tử nhớ một đầu vào

Đối với các phần tử nhớ có một đầu vào x , hàm chuyển đổi trạng thái của nó là một hàm hai biến. Ta đã có:

$$A_m = \varphi(A_c, X_i)$$

Vì chỉ có một biến trong và một biến vào, nên có thể viết trực tiếp như sau:

$$a_m = \varphi(a_c, x)$$

Chúng ta cũng biết rằng, với hàm hai biến ta có số tập giá trị biến có thể có là $N = 2^2 = 4$ và số hàm lôgic khác nhau có thể có là $M = 2^N = 2^4 = 16$.

Các hàm này cho trên bảng ở hình 10.2.

X	A	$A_m = (A_c, X_i)$																
x	a	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	

Hình 10.2. Bảng giá trị thực của các hàm hai biến

Vậy trong số 16 hàm hai biến này, những hàm nào có thể dùng để xây dựng thành các phần tử nhớ cơ bản một đầu vào?

Trước khi chọn các hàm để thực hiện chức năng của trigơ, ta có nhận xét rằng bảng ở hình 10.2 có thể chia thành hai phần: Phần thứ nhất từ hàm a_0 đến hàm a_7 . Phần thứ hai từ hàm a_8 đến hàm a_{15} . Các hàm trong mỗi phần đều có hàm tương ứng trong phần kia là phủ định của nhau. Thí dụ hàm a_8 là phủ định của a_7 , hàm a_{15} là phủ định của hàm a_0 ... Vì vậy thực chất chỉ cần quan tâm đến tám hàm.

Trên hình 10.3 cho bảng chuyển đổi trạng thái, đồ hình trạng thái; bảng để xác định trạng thái mới và bảng kích thích của các mạch được xây dựng trên cơ sở tám hàm này. Trong số tám hàm, trên thực tế chỉ có các hàm sau đây là có ý nghĩa đối với việc xây dựng các phần tử nhớ cơ bản. Đó là các hàm a_1 , a_3 , a_4 , a_6 và a_7 . Các hàm còn lại không đủ đảm bảo cho mạch có những đặc trưng của một trigơ. Sau đây ta sẽ xét từng hàm.

Hàm	Bảng chuyển đổi trạng thái	Đồ hình trạng thái	Bảng tính a_m	Bảng kích thích																											
a_0	<table> <tr> <td>$\begin{matrix} & x \\ a & \end{matrix}$</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> </table>	$\begin{matrix} & x \\ a & \end{matrix}$	0	1	0	0	0	1	0	0		<table> <tr> <td>$\begin{matrix} & x \\ a & \end{matrix}$</td> <td>$a_m$</td> </tr> <tr> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>0</td> </tr> <tr> <td>$a_m = 0$</td> <td></td> </tr> </table>	$\begin{matrix} & x \\ a & \end{matrix}$	a_m	0	0	0	0	$a_m = 0$		<table> <tr> <td>$a_c \rightarrow a_m$</td> <td>K.t</td> </tr> <tr> <td>0 0</td> <td>0V1</td> </tr> <tr> <td>0 1</td> <td>-</td> </tr> <tr> <td>1 0</td> <td>0V1</td> </tr> <tr> <td>1 1</td> <td>-</td> </tr> </table>	$a_c \rightarrow a_m$	K.t	0 0	0V1	0 1	-	1 0	0V1	1 1	-
$\begin{matrix} & x \\ a & \end{matrix}$	0	1																													
0	0	0																													
1	0	0																													
$\begin{matrix} & x \\ a & \end{matrix}$	a_m																														
0	0																														
0	0																														
$a_m = 0$																															
$a_c \rightarrow a_m$	K.t																														
0 0	0V1																														
0 1	-																														
1 0	0V1																														
1 1	-																														
a_1	<table> <tr> <td>$\begin{matrix} & x \\ a & \end{matrix}$</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> </table>	$\begin{matrix} & x \\ a & \end{matrix}$	0	1	0	0	0	1	0	1		<table> <tr> <td>$\begin{matrix} & x \\ a & \end{matrix}$</td> <td>$a_m$</td> </tr> <tr> <td>1</td> <td>0</td> </tr> <tr> <td>0</td> <td>0</td> </tr> <tr> <td>$a_m = a.x$</td> <td></td> </tr> </table>	$\begin{matrix} & x \\ a & \end{matrix}$	a_m	1	0	0	0	$a_m = a.x$		<table> <tr> <td>$a_c \rightarrow a_m$</td> <td>K.t</td> </tr> <tr> <td>0 0</td> <td>0V1</td> </tr> <tr> <td>0 1</td> <td>-</td> </tr> <tr> <td>1 0</td> <td>0</td> </tr> <tr> <td>1 1</td> <td>-</td> </tr> </table>	$a_c \rightarrow a_m$	K.t	0 0	0V1	0 1	-	1 0	0	1 1	-
$\begin{matrix} & x \\ a & \end{matrix}$	0	1																													
0	0	0																													
1	0	1																													
$\begin{matrix} & x \\ a & \end{matrix}$	a_m																														
1	0																														
0	0																														
$a_m = a.x$																															
$a_c \rightarrow a_m$	K.t																														
0 0	0V1																														
0 1	-																														
1 0	0																														
1 1	-																														
a_2	<table> <tr> <td>$\begin{matrix} & x \\ a & \end{matrix}$</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> </table>	$\begin{matrix} & x \\ a & \end{matrix}$	0	1	0	0	1	1	0	0		<table> <tr> <td>$\begin{matrix} & x \\ a & \end{matrix}$</td> <td>$a_m$</td> </tr> <tr> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> </tr> <tr> <td>$a_m = \bar{a}.x$</td> <td></td> </tr> </table>	$\begin{matrix} & x \\ a & \end{matrix}$	a_m	0	0	1	0	$a_m = \bar{a}.x$		<table> <tr> <td>$a_c \rightarrow a_m$</td> <td>K.t</td> </tr> <tr> <td>0 0</td> <td>0</td> </tr> <tr> <td>0 1</td> <td>1</td> </tr> <tr> <td>1 0</td> <td>0V1</td> </tr> <tr> <td>1 1</td> <td>-</td> </tr> </table>	$a_c \rightarrow a_m$	K.t	0 0	0	0 1	1	1 0	0V1	1 1	-
$\begin{matrix} & x \\ a & \end{matrix}$	0	1																													
0	0	1																													
1	0	0																													
$\begin{matrix} & x \\ a & \end{matrix}$	a_m																														
0	0																														
1	0																														
$a_m = \bar{a}.x$																															
$a_c \rightarrow a_m$	K.t																														
0 0	0																														
0 1	1																														
1 0	0V1																														
1 1	-																														
a_3	<table> <tr> <td>$\begin{matrix} & x \\ a & \end{matrix}$</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> </table>	$\begin{matrix} & x \\ a & \end{matrix}$	0	1	0	0	1	1	0	1		<table> <tr> <td>$\begin{matrix} & x \\ a & \end{matrix}$</td> <td>$a_m$</td> </tr> <tr> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> </tr> <tr> <td>$a_m = x$</td> <td></td> </tr> </table>	$\begin{matrix} & x \\ a & \end{matrix}$	a_m	1	0	1	0	$a_m = x$		<table> <tr> <td>$a_c \rightarrow a_m$</td> <td>K.t</td> </tr> <tr> <td>0 0</td> <td>0</td> </tr> <tr> <td>0 1</td> <td>1</td> </tr> <tr> <td>1 0</td> <td>0</td> </tr> <tr> <td>1 1</td> <td>1</td> </tr> </table>	$a_c \rightarrow a_m$	K.t	0 0	0	0 1	1	1 0	0	1 1	1
$\begin{matrix} & x \\ a & \end{matrix}$	0	1																													
0	0	1																													
1	0	1																													
$\begin{matrix} & x \\ a & \end{matrix}$	a_m																														
1	0																														
1	0																														
$a_m = x$																															
$a_c \rightarrow a_m$	K.t																														
0 0	0																														
0 1	1																														
1 0	0																														
1 1	1																														

Hàm	Bảng chuyển đổi trạng thái	Đồ hình trạng thái	Bảng tính a_m	Bảng kích thích																												
a_4	<table> <tr> <td>$\begin{matrix} x \\ a \end{matrix}$</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </table>	$\begin{matrix} x \\ a \end{matrix}$	0	1	0	0	0	1	1	0		<table> <tr> <td>$\begin{matrix} x \\ a \end{matrix}$</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> </table> $a_m = a \cdot \bar{x}$	$\begin{matrix} x \\ a \end{matrix}$	0	1	0	0	0	1	0	0	<table> <tr> <th>$a_c \rightarrow a_m$</th> <th>K.t</th> </tr> <tr> <td>0 0</td> <td>0V1</td> </tr> <tr> <td>0 1</td> <td>-</td> </tr> <tr> <td>1 0</td> <td>1</td> </tr> <tr> <td>1 1</td> <td>0</td> </tr> </table>	$a_c \rightarrow a_m$	K.t	0 0	0V1	0 1	-	1 0	1	1 1	0
$\begin{matrix} x \\ a \end{matrix}$	0	1																														
0	0	0																														
1	1	0																														
$\begin{matrix} x \\ a \end{matrix}$	0	1																														
0	0	0																														
1	0	0																														
$a_c \rightarrow a_m$	K.t																															
0 0	0V1																															
0 1	-																															
1 0	1																															
1 1	0																															
a_5	<table> <tr> <td>$\begin{matrix} x \\ a \end{matrix}$</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> </table>	$\begin{matrix} x \\ a \end{matrix}$	0	1	0	0	0	1	1	1		<table> <tr> <td>$\begin{matrix} x \\ a \end{matrix}$</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> </table> $a_m = a$	$\begin{matrix} x \\ a \end{matrix}$	0	1	0	1	1	1	0	0	<table> <tr> <th>$a_c \rightarrow a_m$</th> <th>K.t</th> </tr> <tr> <td>0 0</td> <td>0V1</td> </tr> <tr> <td>0 1</td> <td>-</td> </tr> <tr> <td>1 0</td> <td>-</td> </tr> <tr> <td>1 1</td> <td>0V1</td> </tr> </table>	$a_c \rightarrow a_m$	K.t	0 0	0V1	0 1	-	1 0	-	1 1	0V1
$\begin{matrix} x \\ a \end{matrix}$	0	1																														
0	0	0																														
1	1	1																														
$\begin{matrix} x \\ a \end{matrix}$	0	1																														
0	1	1																														
1	0	0																														
$a_c \rightarrow a_m$	K.t																															
0 0	0V1																															
0 1	-																															
1 0	-																															
1 1	0V1																															
a_6	<table> <tr> <td>$\begin{matrix} x \\ a \end{matrix}$</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </table>	$\begin{matrix} x \\ a \end{matrix}$	0	1	0	0	1	1	1	0		<table> <tr> <td>$\begin{matrix} x \\ a \end{matrix}$</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </table> $a_m = a \cdot \bar{x} + \bar{a} \cdot x$	$\begin{matrix} x \\ a \end{matrix}$	0	1	0	0	1	1	1	0	<table> <tr> <th>$a_c \rightarrow a_m$</th> <th>K.t</th> </tr> <tr> <td>0 0</td> <td>0</td> </tr> <tr> <td>0 1</td> <td>1</td> </tr> <tr> <td>1 0</td> <td>1</td> </tr> <tr> <td>1 1</td> <td>0</td> </tr> </table>	$a_c \rightarrow a_m$	K.t	0 0	0	0 1	1	1 0	1	1 1	0
$\begin{matrix} x \\ a \end{matrix}$	0	1																														
0	0	1																														
1	1	0																														
$\begin{matrix} x \\ a \end{matrix}$	0	1																														
0	0	1																														
1	1	0																														
$a_c \rightarrow a_m$	K.t																															
0 0	0																															
0 1	1																															
1 0	1																															
1 1	0																															
a_7	<table> <tr> <td>$\begin{matrix} x \\ a \end{matrix}$</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> </table>	$\begin{matrix} x \\ a \end{matrix}$	0	1	0	0	1	1	1	1		<table> <tr> <td>$\begin{matrix} x \\ a \end{matrix}$</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </table> $a_m = a + x$	$\begin{matrix} x \\ a \end{matrix}$	0	1	0	1	1	1	1	0	<table> <tr> <th>$a_c \rightarrow a_m$</th> <th>K.t</th> </tr> <tr> <td>0 0</td> <td>0</td> </tr> <tr> <td>0 1</td> <td>1</td> </tr> <tr> <td>1 0</td> <td>-</td> </tr> <tr> <td>1 1</td> <td>0V1</td> </tr> </table>	$a_c \rightarrow a_m$	K.t	0 0	0	0 1	1	1 0	-	1 1	0V1
$\begin{matrix} x \\ a \end{matrix}$	0	1																														
0	0	1																														
1	1	1																														
$\begin{matrix} x \\ a \end{matrix}$	0	1																														
0	1	1																														
1	1	0																														
$a_c \rightarrow a_m$	K.t																															
0 0	0																															
0 1	1																															
1 0	-																															
1 1	0V1																															

Hình 10.3. Các hàm hai biến được xem xét khả năng thực hiện chức năng của phần tử nhớ cơ bản

10.2.2. Xét hàm a_1

Khi xây dựng phần tử nhớ dùng hàm a_1 sẽ có đặc điểm sau:

Khi đầu vào x bằng 0 thì luôn luôn đưa trạng thái của phần tử nhớ về 0, bất kỳ trước đó phần tử nhớ ở trạng thái 0 hay 1. Còn giá trị 1 của x sẽ không làm thay đổi trạng thái trong của phần tử nhớ. Đầu vào này thường gọi là đầu vào xoá (RESET) nên thay vì ký hiệu là x , người ta ký hiệu bằng chữ R.

Như vậy phần tử nhớ mà chỉ có một đầu vào R (hay $\bar{R}$) có giá trị 1 (hay $\bar{R}$ có giá trị 0) thì không tồn tại một tín hiệu vào nào để có thể đưa trạng thái trong của phần tử nhớ từ 0 lên 1. Muốn đưa từ trạng thái 0 lên trạng thái 1, cần phải sử dụng một đầu vào khác, đầu vào S mà ta sẽ xét sau đây.

10.2.3. Xét hàm a_7

Đặc điểm của mạch khi xây dựng dựa trên hàm a_7 như sau:

Khi đó nếu giá trị của đầu vào x là 1 thì không phụ thuộc vào trạng thái trong của phần tử nhớ, sẽ đưa trạng thái trong này về trạng thái 1. Nếu $x = 0$ thì không làm thay đổi trạng thái của phần tử nhớ. Vì đầu vào x trong trường hợp này luôn đưa trạng thái trong của phần tử nhớ về trạng thái 1, nên nó được ký hiệu bằng chữ S, viết tắt của chữ SET là lập giá trị 1.

Như vậy trái với đầu vào R, đầu vào S không có giá trị nào để có thể đưa trạng thái của phần tử nhớ từ trạng thái 1 về trạng thái 0. Để phần tử nhớ có thể chuyển từ trạng thái 1 về trạng thái 0 và ngược lại, có thể dùng kết hợp hai đầu vào R và S. Chúng ta sẽ xét loại phần tử nhớ có hai đầu vào là R và S sau.

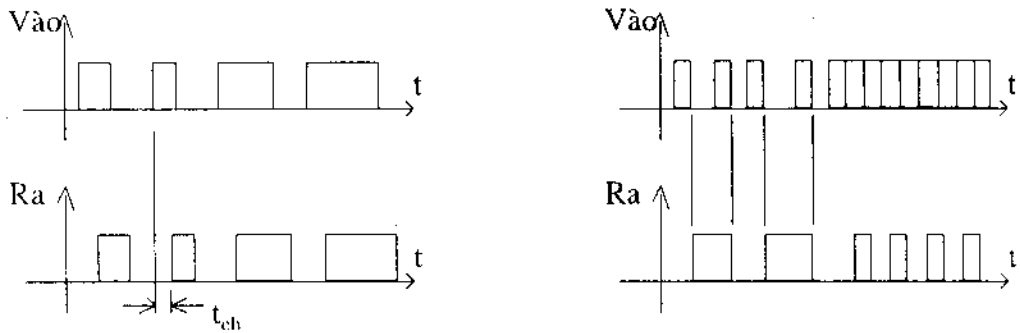
10.2.4. Xét hàm a_3

Phần tử nhớ xây dựng trên cơ sở hàm logic a_3 có đặc điểm là giá trị tín hiệu đầu ra a mới được thiết lập luôn bằng giá trị đầu vào x tác động vào nó, nhưng bị giữ chậm một nhịp. Vì vậy đầu vào x trong trường hợp này được ký hiệu là D, viết tắt của từ giữ chậm trong tiếng Anh là Delay. Từ bảng chuyển đổi trạng thái và đồ hình trạng thái của phần tử nhớ xây dựng trên cơ sở a_3 , ta thấy rằng trạng thái 0 luôn bền với giá trị vào trạng thái là 0. Ngược lại, trạng thái trong bằng 1 luôn bền đối với đầu vào bằng 1. Phần tử nhớ này có thể làm việc với trạng thái vào là xung hay thế cũng được.

Vì tồn tại cả bốn bước chuyển: 0 sang 0; 0 sang 1; 1 sang 0 và 1 sang 1, nên phần tử nhớ xây dựng trên cơ sở hàm a_3 thỏa mãn điều kiện của một phần tử nhớ cơ bản. Nó là một trigơ - Trigơ D.

10.2.5. Xét hàm a_6

Khi phần tử nhớ được xây dựng trên cơ sở hàm a_6 sẽ có các đặc điểm sau đây: Khi đầu vào $x = 1$ thì trạng thái trong của phần tử nhớ luôn thay đổi, dù trạng thái ban đầu ở 0 hay 1. Khi trạng thái của $x = 0$ thì không làm thay đổi trạng thái trong của nó. Nói cách khác, $x = 1$ là trạng thái tích cực, còn $x = 0$ là trạng thái thụ động của đầu vào. Vì vậy đầu vào x trong trường hợp này được ký hiệu là T, là viết tắt của từ tiếng anh Trigger - là nhảy liên tiếp. Phần tử nhớ trong trường hợp này cũng có đủ bốn bước chuyển giống như trong trường hợp xây dựng bằng hàm a_3 . Do đó trigơ này gọi là trigơ T. Cũng từ đây chúng ta thấy rằng trigơ T chỉ có thể làm việc đúng với trạng thái vào tích cực của T là xung. Nếu trạng thái 1 của T là thế thì sẽ xảy ra dao động.



a) Trơ D: t_{ch} là thời gian bị giữ chậm

b) Trơ T: Xung vào rộng sẽ gây dao động

Hình 10.4. Tín hiệu ra phụ thuộc vào tín hiệu vào đối với trơ D và T

Trên hình 10.4 vẽ các tín hiệu vào và ra tương ứng của hai loại trơ D và T ứng với các trường hợp tín hiệu vào khác nhau.

Tính chất có đầy đủ các bước chuyển trạng thái là rất quan trọng, vì nếu không mạch tổng hợp trên cơ sở các phần tử nhớ này sẽ không thể thực hiện được một số chức năng nào đó.

Trong bảng ở hình 10.3, các hàm a_1 , a_4 và a_7 tạo nên các mạch tuy không có đủ 4 bước chuyển trạng thái, song trên thực tế chúng thường được dùng kết hợp. Các hàm khác không có ý nghĩa thực tiễn nên không xét trong tài liệu này.

10.3. CÁC PHẦN TỬ NHỚ HAI ĐẦU VÀO TÍN HIỆU

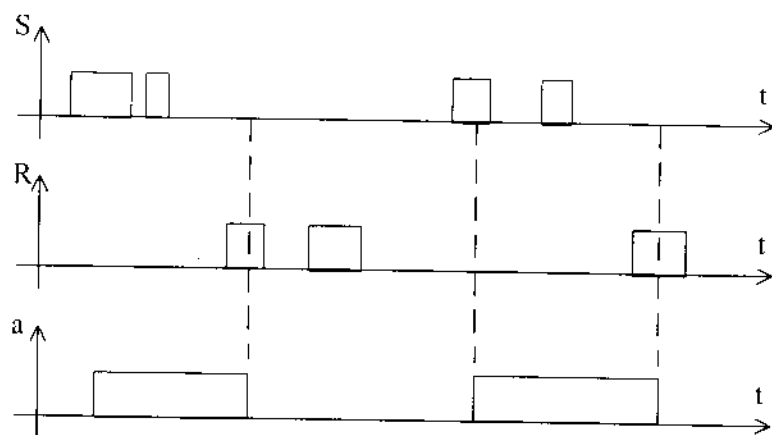
Vì số lượng các phần tử nhớ hai đầu vào rất lớn, trên bảng ở hình 10.5 chỉ xét một vài loại thông dụng. Trong đó các hàm có hai đầu vào là R và S; J và K là các phần tử nhớ có hai đầu vào chính thức là các đầu vào tín hiệu. Trong khi các mạch thực hiện bằng các hàm có một đầu vào tín hiệu là D và T, còn một đầu vào khác là đầu vào nhịp (C). Đầu vào nhịp tuy không phải là đầu vào tín hiệu chính thức, nhưng về mặt thông tin nó hoàn toàn có thể coi như một đầu vào. Vì vậy ở đây bên cạnh các hàm có các đầu vào R, S và J, K, ta sẽ xét thêm cả hai hàm với các đầu vào là C và D, C và T để làm rõ thêm tính khái quát của nội dung lý thuyết.

10.3.1. Phần tử nhớ cơ bản xây dựng trên cơ sở hàm RS

Phần tử nhớ này có hai đầu vào R và S và gọi là trơ RS. Sự kết hợp giữa hai đầu vào R và S làm cho phần tử nhớ này có đủ cả 4 bước chuyển: 0 - 0, 0 - 1, 1 - 0, 1 - 1. Song như bảng chuyển đổi trạng thái đã chỉ rõ, tập giá trị biến 11 của hai biến vào SR sẽ đưa phần tử nhớ tới trạng thái bất định. Cho nên điều kiện làm việc đúng của phần tử nhớ RS là: $S \cdot R = 0$. Đây là loại trơ không đồng bộ, nó có thể làm việc với các tín hiệu vào R, S là xung hay thế cũng được. Giản đồ thời gian của các tín hiệu của trơ RS cho trên hình 10.6.

Tên hàm	Bảng chuyển đổi trạng thái	Đồ hình trạng thái	Bảng tính trạng thái a_m	Bảng kích thích																																								
Phần tử RS	<table><tr><th>SR \ A</th><th>00</th><th>01</th><th>10</th><th>11</th></tr><tr><th>0</th><td>0</td><td>0</td><td>1</td><td>-</td></tr><tr><th>1</th><td>1</td><td>0</td><td>1</td><td>-</td></tr></table>	SR \ A	00	01	10	11	0	0	0	1	-	1	1	0	1	-	Điều kiện: $R.S = 0$	<table><tr><th>S</th><th>a</th><th>a_m</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table> $a_m = S + a.R$	S	a	a_m	0	0	0	0	1	0	1	0	1	1	1	0	<table><tr><th>$a_c \rightarrow a_m$</th><th>SR</th></tr><tr><td>0 0</td><td>0-</td></tr><tr><td>0 1</td><td>10</td></tr><tr><td>1 0</td><td>01</td></tr><tr><td>1 1</td><td>-0</td></tr></table>	$a_c \rightarrow a_m$	SR	0 0	0-	0 1	10	1 0	01	1 1	-0
SR \ A	00	01	10	11																																								
0	0	0	1	-																																								
1	1	0	1	-																																								
S	a	a_m																																										
0	0	0																																										
0	1	0																																										
1	0	1																																										
1	1	0																																										
$a_c \rightarrow a_m$	SR																																											
0 0	0-																																											
0 1	10																																											
1 0	01																																											
1 1	-0																																											
Phần tử CD	<table><tr><th>CD \ A</th><th>00</th><th>01</th><th>10</th><th>11</th></tr><tr><th>0</th><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><th>1</th><td>1</td><td>1</td><td>0</td><td>1</td></tr></table>	CD \ A	00	01	10	11	0	0	0	0	1	1	1	1	0	1		<table><tr><th>C</th><th>a</th><th>a_m</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table> $a_m = a.C + CD$	C	a	a_m	0	0	0	0	1	0	1	0	1	1	1	0	<table><tr><th>$a_c \rightarrow a_m$</th><th>CD</th></tr><tr><td>0 0</td><td>0-/0</td></tr><tr><td>0 1</td><td>11</td></tr><tr><td>1 0</td><td>10</td></tr><tr><td>1 1</td><td>-0/-1</td></tr></table>	$a_c \rightarrow a_m$	CD	0 0	0-/0	0 1	11	1 0	10	1 1	-0/-1
CD \ A	00	01	10	11																																								
0	0	0	0	1																																								
1	1	1	0	1																																								
C	a	a_m																																										
0	0	0																																										
0	1	0																																										
1	0	1																																										
1	1	0																																										
$a_c \rightarrow a_m$	CD																																											
0 0	0-/0																																											
0 1	11																																											
1 0	10																																											
1 1	-0/-1																																											
Phần tử CT	<table><tr><th>CT \ A</th><th>00</th><th>01</th><th>10</th><th>11</th></tr><tr><th>0</th><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><th>1</th><td>1</td><td>1</td><td>1</td><td>0</td></tr></table>	CT \ A	00	01	10	11	0	0	0	0	1	1	1	1	1	0		<table><tr><th>C</th><th>a</th><th>a_m</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table> $a_m = C.a + \bar{C}.a + a.\bar{C}$	C	a	a_m	0	0	0	0	1	0	1	0	1	1	1	0	<table><tr><th>$a_c \rightarrow a_m$</th><th>CT</th></tr><tr><td>0 0</td><td>0-/0</td></tr><tr><td>0 1</td><td>11</td></tr><tr><td>1 0</td><td>11</td></tr><tr><td>1 1</td><td>-0/-0</td></tr></table>	$a_c \rightarrow a_m$	CT	0 0	0-/0	0 1	11	1 0	11	1 1	-0/-0
CT \ A	00	01	10	11																																								
0	0	0	0	1																																								
1	1	1	1	0																																								
C	a	a_m																																										
0	0	0																																										
0	1	0																																										
1	0	1																																										
1	1	0																																										
$a_c \rightarrow a_m$	CT																																											
0 0	0-/0																																											
0 1	11																																											
1 0	11																																											
1 1	-0/-0																																											
Phần tử JK	<table><tr><th>JK \ A</th><th>00</th><th>01</th><th>10</th><th>11</th></tr><tr><th>0</th><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><th>1</th><td>1</td><td>0</td><td>1</td><td>0</td></tr></table>	JK \ A	00	01	10	11	0	0	0	1	1	1	1	0	1	0		<table><tr><th>J</th><th>a</th><th>a_m</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table> $a_m = J.a + \bar{J}.\bar{a} + a.\bar{K}$	J	a	a_m	0	0	0	0	1	0	1	0	1	1	1	0	<table><tr><th>$a_c \rightarrow a_m$</th><th>JK</th></tr><tr><td>0 0</td><td>0-</td></tr><tr><td>0 1</td><td>1-</td></tr><tr><td>1 0</td><td>-1</td></tr><tr><td>1 1</td><td>-0</td></tr></table>	$a_c \rightarrow a_m$	JK	0 0	0-	0 1	1-	1 0	-1	1 1	-0
JK \ A	00	01	10	11																																								
0	0	0	1	1																																								
1	1	0	1	0																																								
J	a	a_m																																										
0	0	0																																										
0	1	0																																										
1	0	1																																										
1	1	0																																										
$a_c \rightarrow a_m$	JK																																											
0 0	0-																																											
0 1	1-																																											
1 0	-1																																											
1 1	-0																																											

Hình 10.5. Bảng mô tả chức năng của phần tử nhớ hai đầu vào tín hiệu



Hình 10.6. Giản đồ thời gian của các tín hiệu của trigơ RS

Quan hệ giữa tín hiệu đầu vào S và R như sau: S quyết định việc chuyển trạng thái của phần tử nhớ từ 0 lên 1, ngược lại, R lại quyết định việc chuyển trạng thái từ 1 về 0. Ngoài ra ta thấy rằng, một chuỗi các tín hiệu S (hay R) mà không có các tín hiệu R (hay S) xen kẽ thì chuỗi tín hiệu này của S và R chỉ có tác dụng như một tín hiệu mà thôi.

10.3.2. Các phần tử nhớ loại CT và CD

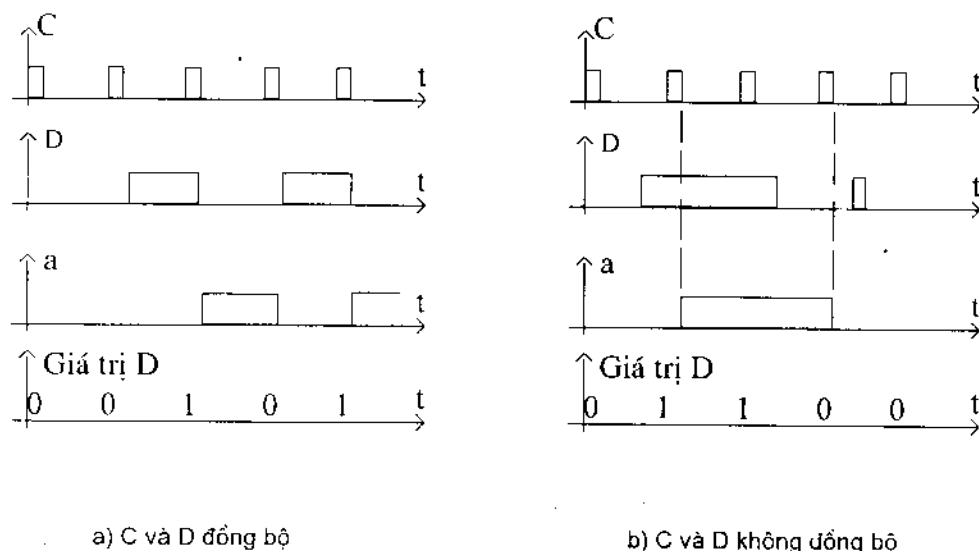
Trong trường hợp này, các phần tử nhớ ngoài đầu vào thông tin D hoặc T còn có đầu vào đồng bộ C mà ở đây cũng coi như một đầu vào thông tin. Từ bảng chuyển đổi trạng thái và bảng ra ta thấy đầu vào C có đặc điểm sau:

- Khi $C = 0$, không phụ thuộc vào T và D bằng 0 hay 1, trạng thái của phần tử nhớ không thay đổi và giữ nguyên trạng thái vốn có.

- Khi $C = 1$, phần tử CT và CD sẽ thay đổi trạng thái hay không là tùy thuộc vào trạng thái ở đầu vào T và D một cách tương ứng.

Do tín hiệu C là tín hiệu nhịp, nên nó thường là loại xung hẹp. Khi đầu vào C có xung thì tín hiệu tại các đầu D và T mới có ý nghĩa và có thể là loại thế hay xung cũng được. Tuy vậy, nếu là xung thì để không mất thông tin, tín hiệu này phải được đồng bộ với tín hiệu nhịp C.

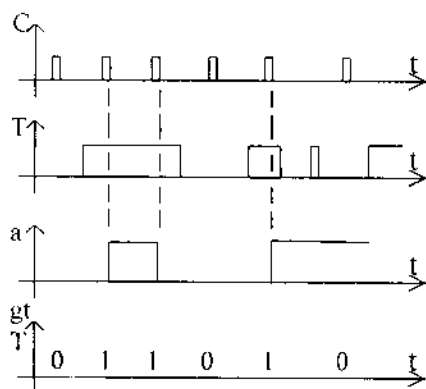
Ngoài ra cần nhận xét rằng, nếu D đồng bộ với tín hiệu C thì tín hiệu ra của phần tử CD hoàn toàn giống tín hiệu của đầu vào D (0 hay 1) nhưng chậm so với tín hiệu vào một nhịp của xung nhịp (hình 10.7a).



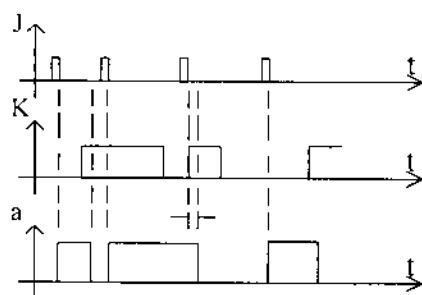
Hình 10.7. Giản đồ thời gian của các tín hiệu của trigơ D khi các tín hiệu C và D đồng bộ và không đồng bộ

Nếu D xuất hiện không đồng bộ với C thì mặc dù sự thay đổi trạng thái vẫn diễn ra tại sườn sau của xung nhịp, nhưng vì sự xuất hiện của D lúc này là tùy ý nên thời gian trễ tính từ khi xuất hiện tín hiệu tại D tới khi tín hiệu đó xuất hiện tại đầu ra là khác nhau. Đồng thời các tín hiệu D ở khoảng giữa hai tín hiệu đồng bộ C sẽ bị bỏ qua. Các tình huống trên đây được mô tả trên hình 10.7b. Nói cách khác, trong hai trường hợp này, tín hiệu D sẽ chỉ được ghi nhận là 1 hay 0 khi tồn tại $C = 1$. Vậy có thể nói, tất cả các phần tử CD đều làm việc giống nhau và không phụ thuộc vào dạng tín hiệu vào D, được đồng bộ với C hay không, là dạng xung hay thế.

Giản đồ thời gian của trigơ CT cho trên hình 10.8a, trong đó tín hiệu C chỉ có ý nghĩa khi $C = 1$. Trong trường hợp này, vì C vốn đã là tín hiệu loại xung nên T có thể là thế. Đồng thời cần lưu ý rằng, nếu T là tín hiệu xung thì để phần tử CT làm việc đúng, T phải đồng bộ với C. Những tín hiệu T không đồng bộ với C sẽ bị bỏ qua.



a) T là thế, không đồng bộ với C



b) Giản đồ thời gian của trigơ JK

Hình 10.8. Giản đồ thời gian của các loại trigơ CT và JK

10.3.3. Phần tử nhớ xây dựng trên cơ sở hàm JK

Phần tử nhớ loại trigơ JK làm việc như tổ hợp của phần tử RS và T. Nó khác với phần tử RS ở chỗ, tập giá trị $JK = 11$ vẫn là tập cho phép và với tập giá trị biến 11 này, phần tử JK làm việc như phần tử T, tức luôn thay đổi trạng thái từ 0 sang 1 và ngược lại từ 1 sang 0.

Vì với tập giá trị biến $JK = 11$, phần tử JK không bền vững, nên trong hoạt động thực tế để nó làm việc đúng, một trong hai tín hiệu J và K phải là xung. Trên hình 10.8b ta đã giả thiết J là loại xung với thời gian nhỏ nhất đủ để trigơ JK thực hiện xong một bước chuyển trạng thái.

Nếu không dùng tập giá trị biến $JK = 11$ thì phần tử nhớ JK làm việc như trigơ RS và trên cả hai đầu vào J và K đều có thể là tín hiệu xung hay thế cũng được.

10.4. PHẦN TỬ NHỚ CÓ TỪ BA ĐẦU VÀO TRỞ LÊN

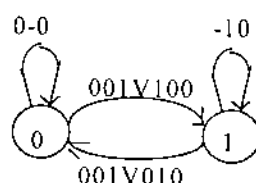
Trên thực tế các phần tử nhớ có nhiều đầu vào cũng rất đa dạng, chúng là kết hợp của của một số loại và có thể có xung nhịp hoặc không có xung nhịp. Chúng ta sẽ xét hai loại sau: Phần tử nhớ RST và SR - CJK.

10.4.1. Phần tử nhớ RST

Đó là loại phần tử nhớ không có xung nhịp, có ba đầu vào. Bảng chuyển đổi trạng thái và đồ hình trạng thái của loại phần tử nhớ này cho trên hình 10.9.

Phần tử nhớ RST là phần tử kết hợp trực tiếp của hai phần tử nhớ RS và T, trong đó đầu vào T là đầu vào xung, đầu vào S và R là các đầu vào thế. Tại một thời điểm, phần tử nhớ chỉ có thể thực hiện chức năng của một trong hai phần tử RS và T. Vì vậy các tổ hợp vào cấm đối với RS vẫn giữ nguyên giá trị. Nói cách khác, điều kiện làm việc của phần tử nhớ này là $R.S = 0$. Do đó trong bảng chuyển đổi trạng thái, giá trị của trạng thái mới trong các ô ứng với các tổ hợp RS là 11 đều là giá trị bất định. Ngoài ra, khi cần chuyển trạng thái thì T làm việc trong điều kiện $R = S = 0$. Vì vậy trong bảng chuyển đổi trạng thái, giá trị của a_m trong tất cả các ô ứng với trường hợp có từ hai biến trong các biến R, S, T bằng 1 thì đều là bất định.

RST a	000	001	010	011	100	101	110	111
0	0	1	0	–	1	–	–	–
1	1	0	0	–	1	–	–	–



a) Bảng chuyển đổi trạng thái

b) Đồ hình trạng thái

Hình 10.9. Bảng chuyển đổi trạng thái và đồ hình trạng thái của trigơ RST

Đối với loại phần tử nhớ RST đầu vào R và S có thể là xung hay thế, còn đầu vào T phải là đầu vào xung.

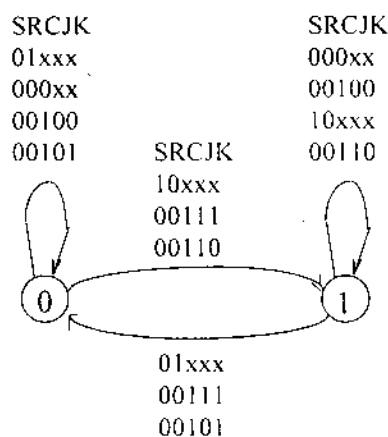
10.4.2. Phần tử nhớ SR - CJK

Đây là phần tử nhớ kết hợp giữa hai phần tử nhớ là SR và CJK, trong đó phần tử SR có ưu tiên cao hơn. Điều đó có nghĩa là tổ hợp $SR = 11$ vẫn là tổ hợp cấm và đối với tổ hợp này, phần tử nhớ SR - CJK có trạng thái bất định.

Bảng chuyển đổi trạng thái và đồ hình trạng thái của loại trigơ này cho trên hình 10.10.

S	R	C	J	K	a_c	a_m
0	0	0	x	x	a_c	a_c
0	0	1	0	0	a_c	a_c
0	0	1	0	1	a_c	0
0	0	1	1	0	a_c	1
0	0	1	1	1	a_c	$\overline{a_c}$
0	1	x	x	x	a_c	0
1	0	x	x	x	a_c	1
1	1	x	x	x	a_c	cắm

a) Bảng chuyển đổi trạng thái



b) Đồ hình trạng thái

Hình 10.10. Bảng chuyển đổi và đồ hình trạng thái của trigơ SR-CJK

Với tổ hợp SR = 01, trạng thái của phần tử nhớ SR - CJK về trạng thái 0. Với SR = 10, trạng thái trong mới của SR - CJK là 1. Chỉ khi nào SR = 00 thì phần tử nhớ SR-CJK mới làm việc như một phần tử nhớ CJK. Trong đó nếu C = 0 thì SR-CJK không thay đổi trạng thái với bất kỳ tác động nào của các đầu vào J và K. Khi C = 1, phần tử SR - CJK làm việc như một phần tử nhớ JK.

10.5. TỔNG HỢP CÁC PHẦN TỬ NHỚ CƠ BẢN

Các phần tử nhớ cơ bản cũng như bất kỳ thiết bị điện tử số nào đều có thể tổng hợp bằng các mạch logic cơ bản. Chúng có đặc điểm chung là có thể thực hiện bằng các loại mạch khác nhau, nhưng chức năng thì như nhau (tính không đơn trị). Dưới đây chúng ta sẽ xét phương pháp tổng hợp một phần tử nhớ cơ bản như một thí dụ điển hình và sau đó xét phương pháp chuyển đổi từ một phần tử nhớ cơ bản này sang phần tử nhớ cơ bản khác, một việc vẫn hay gặp trong hoạt động thực tiễn.

10.5.1. Tổng hợp trigơ RS

Muốn tổng hợp trigơ RS, ta xuất phát từ bảng chuyển đổi trạng thái của nó và chuyển thành bảng Karnaugh đối với trạng thái trong mới a_m của nó như trên hình 10.11a. Từ đó ta rút ra các hàm sau:

$$a_m = S + a_c \overline{R} = \overline{S} | (a_c | \overline{R})$$

hay:

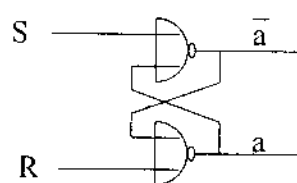
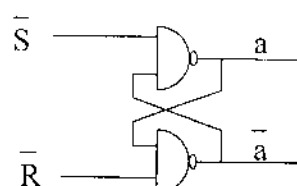
$$a_m = \overline{R} \cdot (S + a_c) = S \downarrow (a_c \downarrow R)$$

Mạch thực hiện đối với hai trường hợp này cho trên hình 10.11b.

	S		a_c		a_m
R	-	-	0	0	
	1	1	1	0	

	S		a_c		a_m
R	-	-	0	0	
	1	1	1	0	

a) Bảng Karnaugh dùng lấy dạng tuyến chuẩn tối thiểu và dạng hội chuẩn tối thiểu của a_m



b) Mạch của trigơ RS dùng NAND và dùng NOR

Hình 10.11. Sơ đồ mạch của trigơ RS

Một cách tương tự, ta có thể tổng hợp các mạch thực hiện chức năng của các loại trigơ khác.

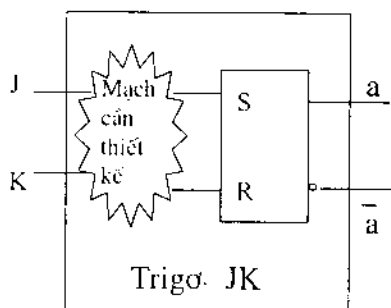
10.5.2. Chuyển đổi từ loại trigơ này sang loại trigơ khác

Dưới đây chúng ta sẽ trình bày một nguyên tắc chung dùng chuyển đổi từ một trigơ bất kỳ nào đó sang một trigơ khác. Để dễ hiểu, chúng ta lấy thí dụ chuyển từ trigơ RS sang trigơ JK chẳng hạn. Thí dụ cụ thể này chỉ nhằm minh họa cho phương pháp chung, có thể áp dụng cho bất kỳ chuyển đổi nào.

Giả sử có trigơ RS, muốn cải tiến trigơ này để có trigơ JK, chúng ta thấy cần phải thiết kế một bộ phận phụ thực hiện các hàm:

$$R = f(J, K, a) \text{ và } S = f(J, K, a)$$

như hình 10.12a đã chỉ ra.



a) Sơ đồ thể hiện nguyên lý cần thực hiện để chuyển đổi

a_c	a_m	J	K	S	R
0	0	0	-	0	-
0	1	1	-	1	0
1	0	-	1	0	1
1	1	-	0	-	0

b) Bảng giá trị thực của R và S

Hình 10.12. Nguyên tắc thiết kế chuyển đổi

Để tìm được các hàm này, có thể sử dụng các hàm kích thích của các trigơ, trong trường hợp này là của các trigơ JK và RS. Chúng ta có bảng giá trị thực của các hàm R và S như bảng ở hình 10.12b. Từ đó ta có thể rút ra bảng Karnaugh dùng xác định các hàm S và R như bảng ở hình 10.13a.

Mạch thực hiện trigơ JK trên cơ sở trigơ RS như trên hình 10.13b.

Một cách tương tự, ta có thể biến đổi từ bất kỳ một loại trigơ này sang một loại trigơ khác bằng cách xác định hàm kích thích của trigơ đã có sẵn như một hàm của các biến vào muốn có và của biến ra của chính nó.

Ngoài ra có thể nhận xét rằng, các vi mạch trigơ ngoài các chân là các đầu vào tín hiệu và các đầu ra, có thể còn có các đầu vào điều khiển sau:

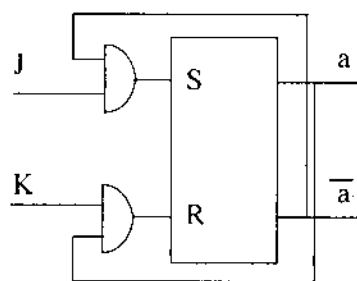
+ Đầu vào đặt trước PRESET (PR): Đầu vào này dùng để đặt trước một giá trị là 0 hay 1 cho đầu ra của trigơ, tùy theo $PR = 0$ hay 1, trước khi nó bước vào chu trình làm việc chính.

	a_c	J			S
K	0	0	1	0	
	-	-	1	0	

	a_c	J			R
K	1	1	0	-	
	0	0	0	-	

$$S = \bar{a}_c \cdot J \quad \text{và} \quad R = a_c \cdot K$$

a) Bảng Karnaugh của S và R



b) Trilog JK

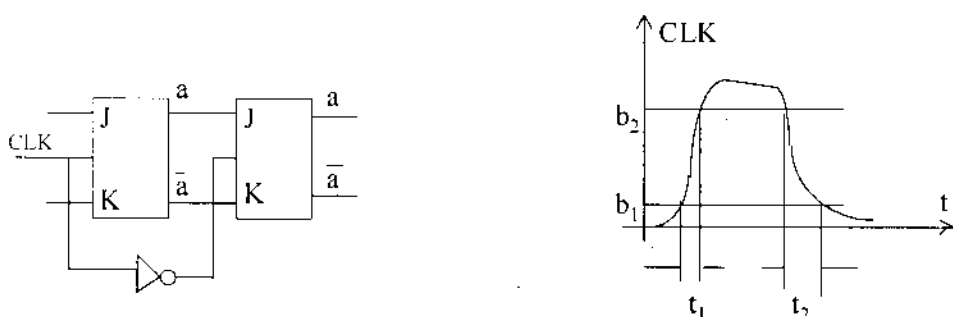
Hình 10.13. Mạch thực hiện trigơ JK trên cơ sở trigơ RS

+ Đầu vào Xóa về trạng thái ban đầu - CLEAR (CLR). Nếu đầu vào này tích cực trong khi đầu vào PR không tích cực thì trạng thái ra a sẽ nhận giá trị 0. Đầu CLR có mức ưu tiên nhỏ hơn đầu PR.

+ Đầu vào nhịp CLOCK (CLK): Là đầu vào dùng các xung nhịp có thời gian tồn tại rất ngắn để đồng bộ hoạt động của trigơ. Đầu vào CLK có thể làm việc với mức hoặc xung hay với sườn xung: sườn lên hoặc sườn xuống.

Cuối cùng chúng ta cần nói tới một loại phần tử nhớ cơ bản - Một loại trigơ hay được dùng, đó là trigơ Chủ - Tớ (Master - Slave = MS). Trilog M⁻ S là loại trigơ mà thời điểm ghi thông tin vào và đưa thông tin ra đầu ra của nó là hai thời điểm tách rời

nhau. Để có thể đạt được điều đó, người ta xây dựng trigơ M - S dựa trên hai trigơ RS hay hai trigơ JK như ở hình 10.14.



a) Mạch trigơ M-S dùng trigơ JK

b) t_1 - thời gian trong đó thông tin được ghi vào;
 t_2 - thời gian trong đó thông tin được đưa ra đầu ra

Hình 10.14. Cấu trúc trigơ Chủ -Tớ (M - S)

Chúng ta cũng lưu ý rằng, trong thực tế các trigơ JK và RS có thể có nhiều đầu vào J và nhiều đầu vào K cũng như nhiều đầu vào S và R. Hình 10.15 là ký hiệu của các trigơ hai đầu vào J, hai đầu vào K và trigơ hai đầu vào S và hai đầu vào R.



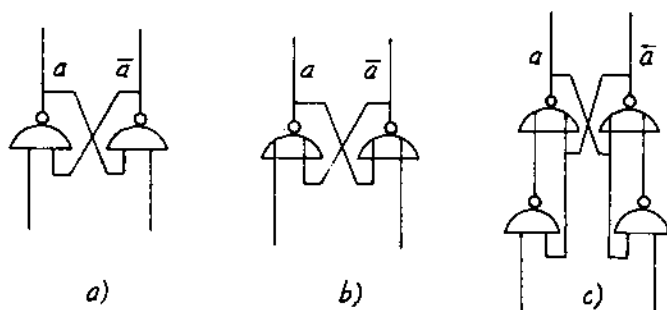
Hình 10.15. Dạng nhiều đầu vào J, K và R, S

Bảng dưới đây cho một số loại trigơ thông dụng:

Ký hiệu vi mạch	Đặc điểm
7470 PC	Trigơ CJK, 2 đầu vào J và 2 đầu vào K
7472 PC	Trigơ CJK loại M - S, 3 đầu vào J và 3 đầu vào K
7473 PC	2 trigơ JK có đầu CLR và CLK riêng cho từng trigơ
7474 PC	2 trigơ CD
7475 PC	4 trigơ CD
7476 PC	2 trigơ JK có các đầu CLR và CLK cho từng trigơ
74174 PC	6 trigơ CD, có đầu CLR
74175 PC	4 trigơ CD, có đầu CLR

BÀI TẬP CHƯƠNG 10

- Hãy vẽ biểu đồ thời gian của trigơ D và CD khi chuỗi tác động đầu vào D là: 0, 1, 0, 1, 1 với trạng thái ban đầu của $a = 0$, khi:
 - Tín hiệu vào là loại xung đồng bộ với xung nhịp.
 - Tín hiệu vào là loại xung không đồng bộ với xung nhịp.
 - Tín hiệu vào là loại thế đồng bộ với xung nhịp.
 - Tín hiệu vào là loại thế không đồng bộ với xung nhịp.
- Hãy vẽ biểu đồ thời gian của trigơ CJK khi chuỗi tác động đầu vào là JK = 10, 01, 11, 01, 00, 11, nếu:
 - Tín hiệu vào là loại xung đồng bộ với xung nhịp.
 - Tín hiệu vào là loại xung không đồng bộ với xung nhịp.
 - Tín hiệu vào là loại thế đồng bộ với xung nhịp.
 - Tín hiệu vào là loại thế không đồng bộ với xung nhịp.
- Hãy xác định xem các phần tử nhớ cho trên hình 10.16a, b, c là loại gì? Hãy điền tên vào các đầu vào tương ứng.



Hình 10.16

- Hãy thiết kế phần bổ sung để biến trigơ JK thành các trigơ CT hoặc CD.
- Hãy nhận trigơ RS từ trigơ JK.

Chương 11

TỔNG HỢP HÀM LÔGÍC PHỤ THUỘC THỜI GIAN VÀ HÀM LÔGÍC TRUY TOÁN

Như trong chương 4 đã trình bày, hàm logic phụ thuộc thời gian và hàm logic truy toán là các hàm logic trong đó tham số thời gian được coi như một biến tương minh đặc biệt. Giá trị của biến thời gian có thể là các số nguyên, dương, âm hay bằng 0. Nhờ có biến thời gian, qua biểu thức của hàm logic ta dễ dàng nhận thấy sự phụ thuộc vào thời gian của chuỗi tín hiệu ra. Ngược lại, trong tổng hợp thiết bị số thường phải giải quyết bài toán xây dựng một thiết bị mà tín hiệu ra của nó là chuỗi tín hiệu phụ thuộc vào thời gian và giữa chúng có mối quan hệ ràng buộc nhất định. Thí dụ, chuỗi các tín hiệu dùng để điều khiển hoạt động của một ô tô máy thuật toán đòi hỏi các tín hiệu này phải tồn tại theo thời gian theo một quy luật nhất định và quan hệ thời gian giữa chúng cũng đóng vai trò quyết định tới hoạt động của ô tô máy.

Thông thường chuỗi tín hiệu như vậy dễ dàng mô tả bằng hàm logic phụ thuộc thời gian và hàm logic truy toán. Trong đó đơn vị của biến thời gian được tính theo tham số thời gian của của tín hiệu chuẩn như tín hiệu nhịp chẳng hạn. Vì vậy tổng hợp thiết bị số cho bởi các hàm logic phụ thuộc thời gian và hàm logic truy toán là có ý nghĩa thực tế.

11.1. TỔNG HỢP THIẾT BỊ SỐ CHO BỞI HÀM LÔGÍC PHỤ THUỘC THỜI GIAN

Giả sử có thiết bị số như ở hình 11.1 mà hoạt động của nó như sau: Khi cho tác động một tập giá trị biến vào nào đó vào đầu vào của thiết bị và giữ nguyên giá trị của các biến x_i trong một thời gian nhất định thì ở đầu ra sẽ xuất hiện chuỗi các tín hiệu phụ thuộc thời gian với độ dài R nào đó.

Hoạt động của thiết bị này có thể mô tả bằng hàm logic phụ thuộc thời gian như sau:

$$Y = \varphi_0 t_0 + \varphi_1 t_1 + \varphi_2 t_2 + \dots + \varphi_{k-1} t_{k-1} \quad (11.1)$$

trong đó φ_i quyết định thành phần tín hiệu thứ i trong chuỗi tín hiệu ra. φ_i , $i = \overline{1, (k-1)}$ là các hàm logic n biến thông thường. Các tham số thời gian t_i là các mốc thời gian xuất hiện các φ_i tương ứng.

Bài toán tổng hợp đặt ra là cho dạng (11.1) của thiết bị, hãy thiết kế mạch thực hiện chức năng hàm đã cho.

Để thực hiện nhiệm vụ này người ta có thể biến đổi dạng (11.1) thành dạng hệ K hàm, mô tả thiết bị có n đầu vào và K đầu ra. Nhưng cũng có thể tổng hợp mạch thực hiện trực tiếp hàm logic phụ thuộc thời gian (11.1) nói trên. Do đó việc tổng hợp phân giải thành hai phần:

- Tổng hợp các mạch thực hiện các hàm logic thông thường φ_i .
- Tổng hợp thiết bị phân phối φ_i theo các mốc thời gian như hình 11.2.

Ở đây giả thiết rằng dạng (11.1) đã là dạng tối thiểu. Nếu (11.1) chưa phải là dạng tối thiểu, ta có thể thực hiện đơn giản hàm dùng phương pháp hệ số bất định. Nếu biến thời gian chỉ nhận hai giá trị 0 và 1 thì có thể dùng bất kỳ phương pháp đơn giản hàm thông thường nào như phương pháp dùng bảng Karnaugh, bảng đối xứng, hay các phương pháp khác.

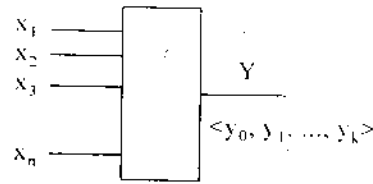
Khó khăn thường xảy ra khi t không phải chỉ nhận hai giá trị 0 và 1 mà là vô số giá trị. Trong trường hợp đó thường dùng phương pháp đơn giản không đầy đủ để đơn giản hàm (11.1). Đơn giản không đầy đủ là chỉ đơn giản hệ K hàm φ_i có n biến. Phương pháp đơn giản hệ các hàm logic n biến đã xét trong chương 3.

Nếu sau khi đơn giản hàm mà xuất hiện một φ_i không phụ thuộc vào thời gian t thì đầu ra của nó trong mạch ở hình 11.2 được nối trực tiếp với Y mà không cần qua bộ phận phân phối thời gian. Ngoài ra, nếu xuất hiện một thành phần nào đó chỉ có t_i mà không có thành phần φ_i thì có nghĩa là từ thời điểm t_i trên đầu ra Y xuất hiện hằng số logic 1 (vì $\varphi_i = 1$).

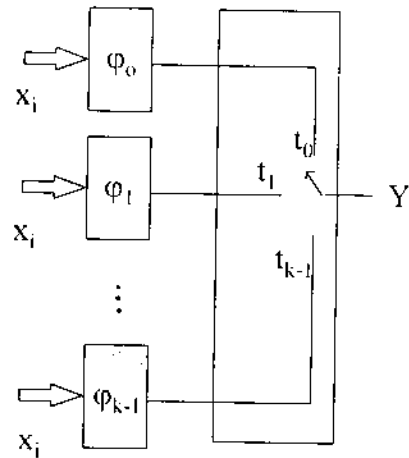
Thí dụ 11.1

Cho hàm logic phụ thuộc thời gian sau:

$$Y = (\bar{x}_1 x_2 x_3 + x_1 \bar{x}_2 (\bar{x}_2 + x_3)) t_0 + (\bar{x}_1 \bar{x}_2 (x_3 + \bar{x}_3) + x_1 x_2 x_3) t_1$$



Hình 11.1. Mô hình thiết bị hoạt động phụ thuộc thời gian



Hình 11.2. Mô hình với bộ phân phối thời gian φ_i

Nếu coi t_0 và t_1 là hai biến logic, nhận các giá trị 0 hoặc 1, ta có thể dùng bảng đối xứng để đơn giản hàm Y năm biến này như trên hình 11.3.

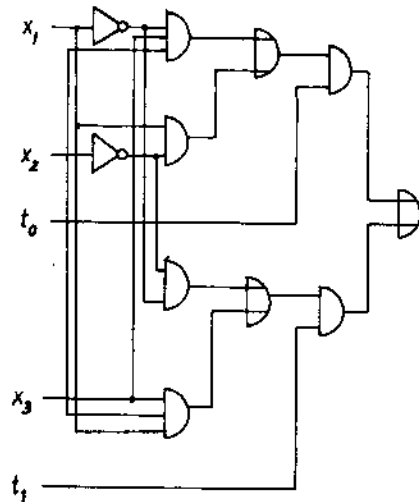
Từ đó có thể rút ra phương trình của hàm Y như sau:

$$\begin{aligned} Y &= \bar{x}_1 \bar{x}_2 t_1 + x_1 x_2 x_3 t_1 + x_1 \bar{x}_2 t_0 + \bar{x}_1 x_2 x_3 t_0 = \\ &= (x_1 \bar{x}_2 + \bar{x}_1 x_2 x_3) t_0 + (\bar{x}_1 \bar{x}_2 + x_1 x_2 x_3) t_1 \end{aligned}$$

Mạch chức năng thực hiện hàm Y cho trên hình 11.4, trong đó các tín hiệu t_0 và t_1 nhận giá trị 1 là biểu hiện sự xuất hiện thời điểm t_0 và t_1 . Trong thực tế có thể dễ dàng thực hiện các tín hiệu t_0 , t_1 nhờ một bộ đếm và một bộ giải mã.

					$x_1 x_2 x_3$
		1	1		000
		1	1		001 S_3
		1	1		011 S_2
		1	1		010 S_3
					110 S_1
					111 S_3
		1	1		101 S_2
		1	1		100 S_3
t_0	1	1	0	0	
t_1	0	1	1	0	
	S_{t1}	S_{t0}	S_{t1}	S_{t1}	

Hình 11.3. Đơn giản hàm dùng bảng đối xứng



Hình 11.4. Mạch thực hiện hàm thời gian của thí dụ 11.1

Thật vậy, Zakrepski A. D. đã đưa ra các biến tham số nhị phân: $\xi_1, \xi_2, \xi_3, \dots, \xi_m$, mà:

$$m = \log_2 r \quad (11.2)$$

trong đó r là số lượng các hàm logic φ_i , $i = \overline{1, r}$. Nếu số hàm φ_i không đủ để thỏa mãn biểu thức lôgarit (11.2) trên đây, ta có thể thêm các hàm tương đương với giá trị hằng "0". Khi đó nếu cho hàm logic phụ thuộc thời gian:

$$Y = \varphi_0(x_1, x_2, \dots, x_n) \cdot t_0 + \varphi_1(x_1, x_2, \dots, x_n) \cdot t_1 + \dots + \varphi_{r-1}(x_1, x_2, \dots, x_n) \cdot t_{r-1}$$

Ta có thể xây dựng một hàm hoàn toàn tương đương với Y , gọi là hàm $F(x_1, x_2, \dots, x_n, \xi_0, \xi_1, \dots, \xi_{m-1})$ như sau:

$$F(x_1, x_2, \dots, x_n, \xi_0, \xi_1, \dots, \xi_{m-1}) = \sum_{i=0}^{r-1} \left(\varphi_i \prod_{k=0}^{m-1} \xi_k^{\alpha_k(i)} \right) \quad (11.3)$$

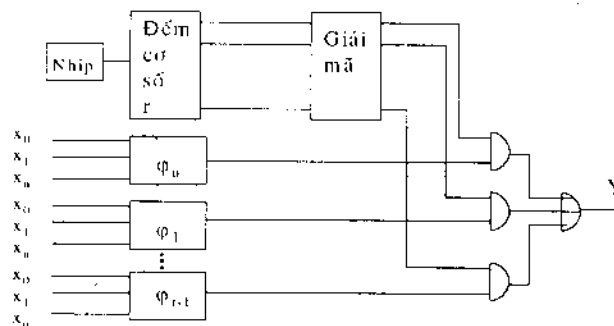
trong đó $\alpha_k(i)$ là giá trị của bit thứ k trong mã nhị phân của số tự nhiên i .

$$\xi^{\alpha_k(i)} = \begin{cases} \xi_k, & \text{nếu } \alpha_k(i) = 1 \\ \overline{\xi_k}, & \text{nếu } \alpha_k(i) = 0 \end{cases}$$

Như vậy, nếu ta làm tương thích các thời gian $t_0, t_1, \dots, t_{r-1}$ với các số nhị phân:

$$t_j, j = \overline{0, r-1} \leftrightarrow \prod_{k=0}^{m-1} \xi^{\alpha_k(i)} = \xi_0^{\alpha_0(i)} \xi_1^{\alpha_1(i)}, \dots, \xi_{m-1}^{\alpha_{m-1}(i)} \quad (11.4)$$

bằng cách xây dựng một bộ đếm nhị phân, với nguồn xung nhịp thích hợp; đồng thời kết hợp với một bộ giải mã, làm tương thích mỗi tổ hợp của bộ mã nhị phân (11.4) với một logic 1 thì có thể kết luận rằng hàm $Y(x_1, x_2, \dots, x_n)$ hoàn toàn tương thích với hàm $F(x_1, x_2, \dots, x_n, \xi_0, \xi_1, \dots, \xi_{m-1})$.



Hình 11.5. Mạch thực hiện $Y(x_1, x_2, \dots, x_n)$

Mạch thực hiện hàm $Y(x_1, x_2, \dots, x_n)$ theo phương pháp trên đây cho trên hình 11.5.

Ngày nay thiết bị phân phối các hàm φ_i theo thời gian thường dùng các bộ trộn như trong phần thiết bị tổ hợp đã trình bày.

Thí dụ 11.2

Tổng hợp thiết bị số cho bởi hàm logic phụ thuộc thời gian sau đây:

$$Y(x_1, x_2) = x_1 x_2 t_0 + x_2 t_2 + t_1 + (x_1 + x_2) t_3$$

Ta có $r = 4, m = 2$, do đó:

$$\prod_{k=0}^{m-1} \xi^{\alpha_k(i)} = \xi_0^{\alpha_0(i)} \cdot \xi_1^{\alpha_1(i)}$$

$$i = 0 \rightarrow \xi_0 \cdot \xi_1, \quad \text{vì } \alpha_0 = 0 \quad \text{và} \quad \alpha_1 = 0$$

$$i = 1 \rightarrow \xi_0 \cdot \xi_1, \quad \text{vì } \alpha_0 = 0 \quad \text{và} \quad \alpha_1 = 1$$

$$i = 2 \rightarrow \xi_0 \cdot \xi_1, \quad \text{vì } \alpha_0 = 1 \quad \text{và} \quad \alpha_1 = 0$$

$$i = 3 \rightarrow \xi_0 \cdot \xi_1, \quad \text{vì } \alpha_0 = 1 \quad \text{và} \quad \alpha_1 = 1$$

Để đảm bảo bốn tổ hợp giá trị của ξ_0, ξ_1 ta cần một bộ đếm nhị phân cơ số 4 và một bộ giải mã từ mã nhị phân sang mã thập phân.

Sơ đồ của mạch thực hiện bài toán trên cho trên hình 11.6.

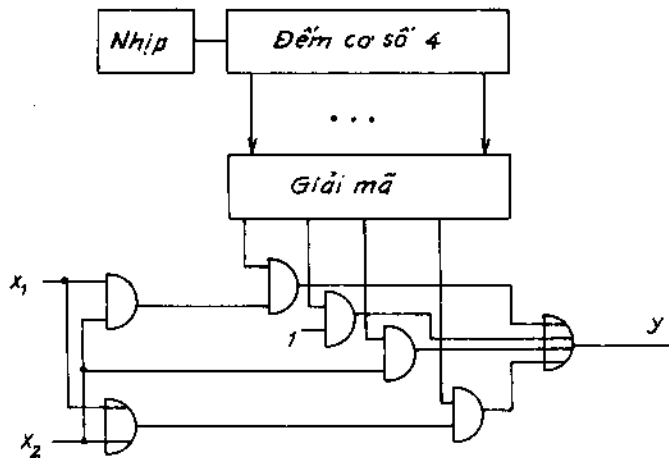
Với sơ đồ cấu trúc như vậy, ta có thể xác định giá trị của hàm tại các điểm của giá trị biến i như sau:

$i = 0$, hàm $Y(x_1, x_2) = x_1 x_2$.

$i = 1$, hàm $Y(x_1, x_2) = 1$, là hằng logic 1.

$i = 2$, hàm $Y(x_1, x_2) = x_2$.

$i = 3$, hàm $Y(x_1, x_2) = x_1 + x_2$.



Hình 11.6. Mạch thực hiện hàm của thí dụ 11.2

11.2. TỔNG HỢP THIẾT BỊ SỐ CÓ NHỚ CHO BỞI HÀM LÔGIC TRUY TOÁN

Trong chương 7 ta đã trình bày khái niệm về hàm logic truy toán, trong đó hàm logic truy toán loại I là hàm logic truy toán mà giá trị của nó tại thời điểm t phụ thuộc không những vào giá trị của các biến vào x_i , $i = \overline{1, n}$ tại thời điểm t đó, mà còn phụ thuộc vào giá trị các biến ra tại các thời điểm trước đó. Đồng thời hàm cũng phụ thuộc vào giá trị ban đầu tại đầu ra y_0 khi hệ thống bắt đầu làm việc. Biểu thức tổng quát của hàm logic truy toán loại I như sau:

$$y_0 = a$$

$$y_t = \varphi(x_{1t}, x_{2t}, \dots, x_{nt}, y_{t-1}, y_{t-2}, \dots, y_{t-k}) \quad (11.5)$$

Trong nhiều hệ thống tự động, các hệ thống điều khiển từ xa cũng như những thiết bị trong kỹ thuật tính toán sử dụng các mã nhị phân với chu kỳ lặp lại nhất định. Những thiết bị như vậy có thể mô tả bằng hàm logic truy toán, trong đó giá trị của hàm tại thời điểm t chỉ phụ thuộc vào trạng thái ban đầu y_0 và các trạng thái của hàm ở các thời điểm trước đó mà không phụ thuộc vào các biến vào x_i . Những hàm logic truy toán

loại này gọi là các hàm logic truy toán suy biến loại I. Biểu thức tổng quát của hàm logic suy biến loại I như sau:

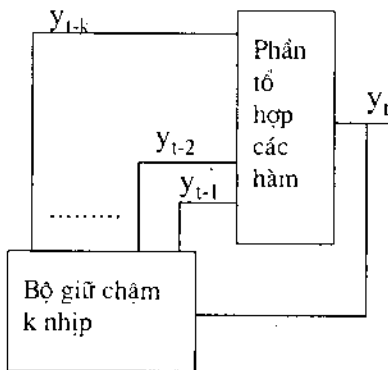
$$\begin{aligned} y_0 &= a, t \leq 0 \\ y_t &= \varphi(y_{t-1}, y_{t-2}, \dots, y_{t-k}), t > 0 \end{aligned} \quad (11.6)$$

Dưới đây sẽ xét phương pháp và khả năng thực hiện các thiết bị số cho bởi hàm logic truy toán suy biến và hàm logic truy toán loại I.

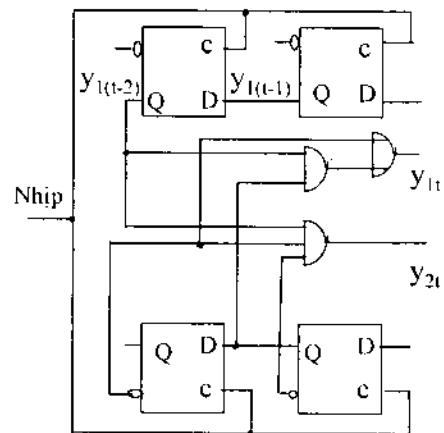
11.2.1. Tổng hợp thiết bị số cho bởi hàm logic truy toán suy biến loại I

Khi chức năng của thiết bị số cho dưới dạng hàm logic truy toán suy biến loại I (11.6), ta thấy rằng tín hiệu ra của thiết bị tại thời điểm t sẽ phụ thuộc vào k nhịp trước đó. Nói cách khác, tín hiệu ra ở nhịp này sẽ là tín hiệu vào của k nhịp tiếp theo. Muốn duy trì được các tín hiệu ra này trong k nhịp ta có thể dùng phần tử nhớ hay phần tử giữ chậm. Thí dụ, mỗi trigơ D có thể giữ chậm tín hiệu được một nhịp. Nếu nối tiếp k trigơ D, ta sẽ có bộ giữ chậm với k đầu ra, tương ứng với k đầu ra bị giữ chậm từ y_{t-1} đến y_{t-k} . Như vậy mạch thực hiện hàm logic truy toán suy biến loại I sẽ gồm các phần tử giữ chậm tín hiệu ra và phần mạch tổ hợp theo sơ đồ ở hình 11.7.

Nếu là một hệ hàm logic truy toán loại I thì đối với mỗi hàm cần phải có một phần tử giữ chậm riêng. Phần tổ hợp sẽ nhận tất cả các hàm $y_{i(t-j)}$, $i = \overline{1, s}$ và $j = \overline{1, k}$, trong đó s là số lượng hàm trong hệ, đồng thời cũng là số lượng các biến vào.



Hình 11.7. Mạch thực hiện hàm logic truy toán suy biến loại I



Hình 11.8. Mạch thực hiện hệ hai hàm logic truy toán suy biến loại I

Thí dụ 11.3

Tổng hợp thiết bị số cho bởi hệ hàm logic truy toán suy biến loại I sau đây:

$$\begin{aligned}y_{1t} &= y_{1(t-2)} \cdot y_{2(t-1)} + \bar{y}_{2(t-2)} \\ y_{2t} &= y_{1(t-2)} \cdot \bar{y}_{2(t-2)} \cdot \bar{y}_{2(t-1)}\end{aligned}\quad (11.7)$$

Hệ này gồm hai hàm y_{1t} và y_{2t} , phụ thuộc vào các giá trị của hàm ở hai nhịp trước đó. Như vậy, đối với mỗi hàm y_{1t} và y_{2t} cần một bộ giữ chậm hai nhịp, mỗi bộ giữ chậm cần hai trigơ D. Phần tổ hợp sẽ thực hiện hàm (11.7) với các hàm $y_{1(t-1)}$, $y_{1(t-2)}$, $y_{2(t-1)}$, $y_{2(t-2)}$ là biến vào.

Mạch thực hiện hệ hai hàm logic truy toán suy biến loại I (11.7) cho trên hình 11.8.

Cần chú ý rằng, trong trường hợp này chúng ta chưa chú ý tới trạng thái ban đầu của hệ thống. Nếu trạng thái ban đầu được xác định trước thì các trigơ D dùng làm phần tử giữ chậm cần có các đầu đặt trước (PRESET): Căn cứ vào trạng thái ban đầu y_{10} và y_{20} và căn cứ vào phần logic tổ hợp để xác định giá trị cần đặt trước cho các trigơ. Điều này dễ dàng thực hiện vì trong thực tế hầu hết các trigơ đều đã có các đầu đặt trước này.

Trạng thái xuất phát ban đầu của hệ thường quyết định hoạt động cụ thể của thiết bị. Trong phần phân tích chúng ta sẽ làm rõ thêm điều này.

Trên đây ta đã xét trường hợp tổng quát của thiết bị số có nhớ, được mô tả bằng hàm logic truy toán suy biến loại I. Trên thực tế rất hay gặp trường hợp đặc biệt với $k = 1$, tức là khi hàm logic truy toán suy biến ở thời điểm $(t + 1)$ chỉ phụ thuộc vào giá trị của hàm ở nhịp t trước đó. Ôtômát loại này gọi là ôtômát ôônôm. Phương trình tổng quát của ôtômát hữu hạn loại ôônôm được mô tả bởi phương trình (11.8). Trường hợp ôtômát có s đầu ra thì ta có hệ phương trình (11.9).

$$\left. \begin{aligned}y_0 &= a \\ y_{t+1} &= \varphi(y_t)\end{aligned} \right\} \quad (11.8)$$

$$\left. \begin{aligned}y_{10} &= a_{10} \\ y_{20} &= a_{20} \\ &\dots\dots\dots \\ y_{s0} &= a_{s0} \\ y_{1(t+1)} &= \varphi_1(y_{1t}, y_{2t}, \dots, y_{st}) \\ y_{2(t+1)} &= \varphi_2(y_{1t}, y_{2t}, \dots, y_{st}) \\ &\dots\dots\dots \\ y_{s(t+1)} &= \varphi_s(y_{1t}, y_{2t}, \dots, y_{st})\end{aligned} \right\} \quad (11.9)$$

Quá trình tổng hợp ôtômát hữu hạn ôônôm cho bởi hệ phương trình (11.9) cũng được tiến hành theo các bước tổng hợp một ôtômát hữu hạn được mô tả bởi phương trình logic truy toán suy biến loại I trên đây. Trong đó vì $y_{j(t+1)}$ chỉ phụ thuộc vào y_{jt} , $j = \overline{1, s}$, nên chỉ cần dùng phần tử giữ chậm một nhịp đối với mỗi y_{jt} . Việc tính tới trạng

thái ban đầu ($a_0, a_1, \dots, a_n$) cũng được thực hiện bằng cách đặt trước giá trị ban đầu đối với các trigô.

Chức năng của ô tô măt hữu hạn ô tô nôm cũng có thể cho dưới dạng bảng giá trị thực. Việc tổng hợp chúng, dù cho dưới dạng nào cũng có thể tiến hành theo các bước sau đây:

a) Từ dạng bảng giá trị thực nhận hàm dạng (11.9).

b) Đơn giản hệ hàm.

c) Tổng hợp phần giữ chậm.

d) Tổng hợp phần mạch lôgic tổ hợp.

Dưới đây là một vài thí dụ.

Thí dụ 11.4

Tổng hợp ô tô măt ô tô nôm cho dưới dạng bảng giá trị thực sau:

y_{1t} y_{2t} y_{3t}	$y_{1(t+1)}$ $y_{2(t+1)}$ $y_{3(t+1)}$	y_{1t} y_{2t} y_{3t}	$y_{1(t+1)}$ $y_{2(t+1)}$ $y_{3(t+1)}$
0 0 0	0 1 0	1 0 0	1 1 1
0 0 1	1 0 1	1 0 1	1 1 0
0 1 0	1 1 1	1 1 0	0 0 0
0 1 1	0 1 1	1 1 1	0 1 0

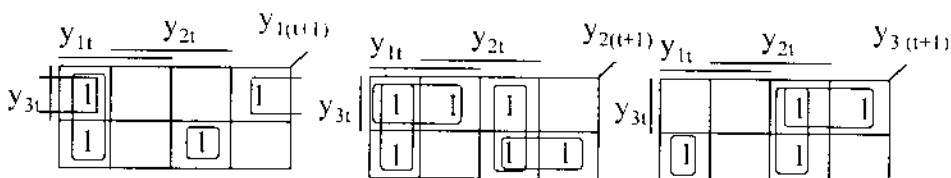
Từ bảng trên chúng ta rút ra các hàm sau đây:

$$y_{1(t+1)} = \bar{y}_{1t} \cdot \bar{y}_{2t} \cdot y_{3t} + \bar{y}_{1t} \cdot y_{2t} \cdot \bar{y}_{3t} + y_{1t} \cdot \bar{y}_{2t} \cdot \bar{y}_{3t} + y_{1t} \cdot \bar{y}_{2t} \cdot y_{3t}$$

$$y_{2(t+1)} = \bar{y}_{1t} \cdot \bar{y}_{2t} \cdot \bar{y}_{3t} + \bar{y}_{1t} \cdot y_{2t} \cdot \bar{y}_{3t} + \bar{y}_{1t} \cdot y_{2t} \cdot y_{3t} + y_{1t} \cdot \bar{y}_{2t} \cdot \bar{y}_{3t} + \\ + y_{1t} \cdot \bar{y}_{2t} \cdot y_{3t} + y_{1t} \cdot y_{2t} \cdot y_{3t}$$

$$y_{3(t+1)} = \bar{y}_{1t} \cdot \bar{y}_{2t} \cdot y_{3t} + \bar{y}_{1t} \cdot y_{2t} \cdot \bar{y}_{3t} + \bar{y}_{1t} \cdot y_{2t} \cdot y_{3t} + y_{1t} \cdot \bar{y}_{2t} \cdot \bar{y}_{3t}$$

Đơn giản các hàm trên đây dùng phương pháp Karnaugh, ta được:

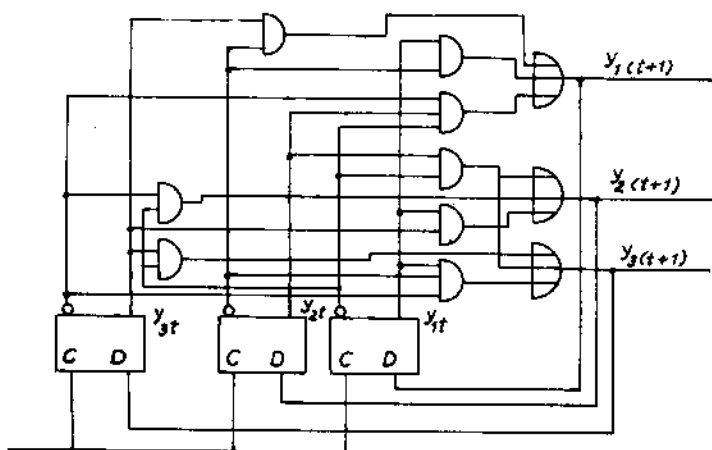


$$y_{1(t+1)} = \bar{y}_{1t} \cdot \bar{y}_{2t} + \bar{y}_{1t} \cdot y_{2t} \cdot \bar{y}_{3t} + \bar{y}_{2t} \cdot y_{3t}$$

$$y_{2(t+1)} = y_{1t} \cdot \bar{y}_{2t} + \bar{y}_{1t} \cdot y_{2t} + y_{1t} \cdot y_{3t} + \bar{y}_{1t} \cdot \bar{y}_{3t}$$

$$y_{3(t+1)} = \bar{y}_{1t} \cdot y_{3t} + \bar{y}_{1t} \cdot y_{2t} + y_{1t} \cdot \bar{y}_{2t} \cdot \bar{y}_{3t}$$

Phần giữ chậm ở đây đối với mỗi đầu ra chỉ cần một khâu giữ chậm một nhịp. Sơ đồ mạch thực hiện hệ phương trình (11.10) cho trên hình 11.9.



Hình 11.9. Mạch thực hiện hệ hàm 11.10

11.2.2. Tổng hợp thiết bị số cho bởi hàm logic truy toán loại I

Thiết bị số có chức năng cho bởi hàm logic truy toán loại I (công thức 11.5) trong nhiều trường hợp có thể viết dưới dạng (11.11) sau đây:

$$\left. \begin{aligned} Y_{1(t+1)} &= \Phi_1(X_{1(t+1)}, \dots, X_{n(t+1)}, Y_{1t}, Y_{2t}, \dots, Y_{1(t-k)}, \dots, Y_{s(t-k)}) \\ Y_{2(t+1)} &= \Phi_1(X_{1(t+1)}, \dots, X_{n(t+1)}, Y_{1t}, Y_{2t}, \dots, Y_{1(t-k)}, \dots, Y_{s(t-k)}) \\ &\vdots \\ Y_{s(t+1)} &= \Phi_s(X_{1(t+1)}, \dots, X_{n(t+1)}, Y_{1t}, Y_{2t}, \dots, Y_{1(t-k)}, \dots, Y_{s(t-k)}) \\ Z_{1(t+1)} &= \Psi_1(X_{1(t+1)}, \dots, X_{n(t+1)}, Y_{1t}, Y_{2t}, \dots, Y_{1(t-k)}, \dots, Y_{s(t-k)}) \\ Z_{2(t+1)} &= \Psi_2(X_{1(t+1)}, \dots, X_{n(t+1)}, Y_{1t}, Y_{2t}, \dots, Y_{1(t-k)}, \dots, Y_{s(t-k)}) \\ &\vdots \\ Z_{r(t+1)} &= \Psi_r(X_{1(t+1)}, \dots, X_{n(t+1)}, Y_{1t}, Y_{2t}, \dots, Y_{1(t-k)}, \dots, Y_{s(t-k)}) \end{aligned} \right\} \quad (11.11)$$

trong đó $y_i, i = \overline{1, s}$ là các biến của ô tô măt. Các $Z_r, r = \overline{1, r}$ là các biến ra. Các biến trong và các biến ra đều là hàm của các biến vào tại thời điểm đang xét và của các biến trong tại các thời điểm trước đó.

Trường hợp đơn giản thường gặp là khi $k = 0$. Khi đó hệ phương trình (11.11) trở thành hệ phương trình (11.12):

$$\begin{aligned} y_{1(t+1)} &= \phi_1(x_{1(t+1)}, \dots, x_{n(t+1)}, y_{1t}, y_{2t}, \dots, y_{st}) \\ y_{2(t+1)} &= \phi_2(x_{1(t+1)}, \dots, x_{n(t+1)}, y_{1t}, y_{2t}, \dots, y_{st}) \\ &\vdots \\ y_{s(t+1)} &= \phi_s(x_{1(t+1)}, \dots, x_{n(t+1)}, y_{1t}, y_{2t}, \dots, y_{st}) \end{aligned}$$

$$Z_{1(t+1)} = \psi_1(x_{1(t+1)}, \dots, x_{n(t+1)}, y_{1t}, y_{2t}, \dots, y_{st}) \quad (11.12)$$

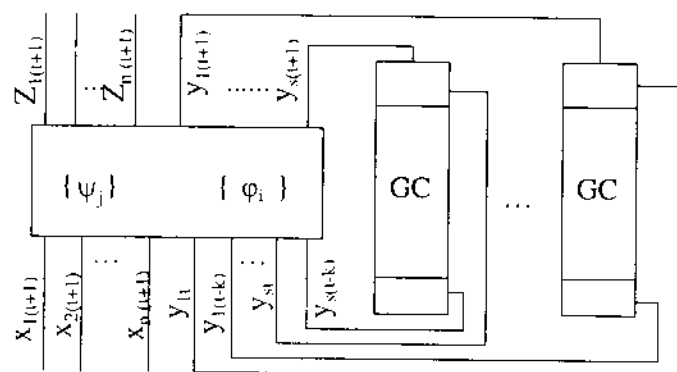
$$Z_{2(t+1)} = \psi_2(x_{1(t+1)}, \dots, x_{n(t+1)}, y_{1t}, y_{2t}, \dots, y_{st})$$

.....

$$Z_{r(t+1)} = \psi_r(x_{1(t+1)}, \dots, x_{n(t+1)}, y_{1t}, y_{2t}, \dots, y_{st})$$

Để thực hiện các hệ phương trình (11.11) và (11.12), ta cần thiết kế hai bộ phận: Bộ phận giữ chậm tín hiệu gồm s kênh, mỗi kênh có $(k+1)$ tín hiệu đầu ra ứng với các tín hiệu được giữ chậm $(k+1)$ nhịp, $k = 1, 2, 3, \dots$. Như vậy đối với hệ phương trình (11.12) mỗi kênh có thể giữ chậm một nhịp. Để giữ chậm tín hiệu 1 nhịp có thể dùng các loại mạch giữ chậm thụ động hay tích cực, các loại trigơ như trigơ D. Để giữ chậm $(k+1)$ nhịp cần $(k+1)$ trigơ D mắc nối tiếp và có đầu nhịp chung. Bộ phận thứ hai là hệ tổ hợp gồm $(s+n)$ đầu vào và $(s+r)$ đầu ra. Tổng hợp mạch tổ hợp đã được xét kỹ trong phần I của tài liệu này.

Như vậy, sơ đồ tổng quát để thực hiện thiết bị số có nhớ cho bởi hàm lôgic truy toán loại I (công thức 11.11) cho trên hình 11.10. Trong đó GC là phần mạch giữ chậm $(k+1)$ nhịp. Mạch thực hiện hệ hàm φ_i và ψ_j là một hệ tổ hợp.



Hình 11.10. Sơ đồ tổng quát thực hiện thiết bị số cho bởi hàm lôgic truy toán loại I

Có thể chứng minh rằng, hệ (11.12) hoàn toàn có thể thực hiện được nhờ các mạch lôgic cơ bản và các trigơ.

Thật vậy, đối với mỗi hàm lôgic n biến bất kỳ $F(x_1, x_2, \dots, x_n)$ ta có đẳng thức sau:

$$F(x_1, x_2, \dots, x_n) = \bar{x}_1 F(x_1 = 0) + x_1 F(x_1 = 1) \quad (11.13)$$

Trong khi đó phương trình mô tả hoạt động của trigơ hai đầu vào ở hình 11.11 có dạng:

$$y_{(t+1)} = \bar{x}_{2(t+1)} y_t + x_{1(t+1)} \bar{y}_t \quad (11.14)$$

trong đó y_t là trạng thái trong của trigơ tại thời điểm t ; $y_{(t+1)}$ là trạng thái của trigơ ở nhịp tiếp theo; $x_{1(t+1)}, x_{2(t+1)}$ là các tín hiệu kích thích của trigơ.

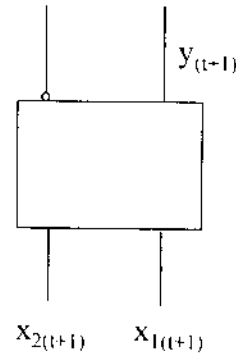
Phương trình (11.14) hoàn toàn phù hợp với (11.13), nếu ta coi:

$$x_{1(i+1)} = F(x_i = 0)$$

$$x_{2(i+1)} = \bar{F}(x_i = 1)$$

Đối với các phương trình φ_i trong (11.12) dễ dàng đưa về dạng (11.13), (11.14), nếu coi:

$$\left. \begin{aligned} x_{1(i+1)}^i &= \varphi_i(y_{it} = 0) \\ x_{2(i+1)}^i &= \bar{\varphi}_i(y_{it} = 1) \end{aligned} \right\} \quad (11.15)$$



Hình 11.11. Trigơ hai đầu vào

Khi đó mỗi phương trình thứ i , $i = \overline{1, s}$ của (11.12) có thể viết:

$$y_{i(i+1)} = \bar{y}_{it} \cdot \varphi_i(y_{it} = 0) + y_{it} \cdot \varphi_i(y_{it} = 1)$$

$$y_{i(i+1)} = \bar{y}_{it} \cdot x_{1(i+1)}^i + y_{it} \cdot \bar{x}_{2(i+1)}^i \quad (11.16)$$

Như vậy, hàm $y_{i(i+1)}$ với mọi i có thể thực hiện nhờ trigơ hai đầu vào. Nếu sử dụng trigơ một đầu vào, thí dụ trigơ T, ta cần đặt:

$$x_{(i+1)}^i = \bar{y}_{it} \cdot \varphi_i(y_{it} = 0) + y_{it} \cdot \bar{\varphi}_i(y_{it} = 1) \quad (11.17)$$

Với $i = \overline{1, s}$ thì:

$$y_{i(i+1)} = \bar{x}_{(i+1)}^i \cdot y_{it} + x_{(i+1)}^i \cdot \bar{y}_{it} \quad (11.18)$$

Đẳng thức (11.18) là phương trình của trigơ T. Thật vậy, nếu thay (11.17) vào (11.18) và biến đổi sẽ được dạng:

$$\begin{aligned} y_{i(i+1)} &= [\bar{y}_{it} \cdot \varphi_i(y_{it} = 0) + y_{it} \cdot \bar{\varphi}_i(y_{it} = 1)] \cdot y_{it} + [\bar{y}_{it} \cdot \varphi_i(y_{it} = 0) + y_{it} \cdot \bar{\varphi}_i(y_{it} = 1)] \cdot \bar{y}_{it} \\ &= y_{it} \cdot (y_{it} + \bar{\varphi}_i(y_{it} = 0)) \cdot (\bar{y}_{it} + \varphi_i(y_{it} = 1)) + \bar{y}_{it} \cdot \varphi_i(y_{it} = 0) \cdot \bar{y}_{it} + y_{it} \cdot \bar{y}_{it} \cdot \bar{\varphi}_i(y_{it} = 1) \\ &= y_{it} \cdot \varphi_i(y_{it} = 1) + y_{it} \cdot \bar{\varphi}_i(y_{it} = 0) \cdot \varphi_i(y_{it} = 1) + \bar{y}_{it} \cdot \varphi_i(y_{it} = 0) \\ &= y_{it} \cdot \varphi_i(y_{it} = 1) + \bar{y}_{it} \cdot \varphi_i(y_{it} = 0). \end{aligned}$$

Đó chính là dạng của (11.16). Như vậy ta đã chứng minh rằng, các phương trình $y_{i(i+1)}$ với mọi i có thể thực hiện được nhờ các trigơ một hoặc hai đầu vào.

Nhóm phương trình $Z_{(i+1)}$ thực hiện như các mạch tổ hợp bình thường, không phải dùng tới các phần tử nhớ.

Thí dụ 11.5

Thực hiện ô tô-mát cho bởi hệ hàm logic truy toán loại I sau đây:

$$\left. \begin{aligned} y_{1(i+1)} &= x_{1(i+1)} y_{1t} + x_{2(i+1)} \\ y_{2(i+1)} &= y_{1t} + y_{2t} \\ Z_{1(i+1)} &= x_{1(i+1)} \cdot x_{2(i+1)} + \bar{y}_{2(t-1)} \end{aligned} \right\} \quad (11.19)$$

a) Trường hợp dùng trigơ hai đầu vào loại JK:

Đối với hàm $y_{1(t+1)}$ ta đặt :

$$x_{1(t+1)}^1 = \varphi_1(y_{1t} = 0) = x_{2(t+1)}$$

$$x_{2(t+1)}^1 = \varphi_1(y_{1t} = 1) = x_{1(t+1)} + x_{2(t+1)}$$

Đối với hàm $y_{2(t+1)}$ ta đặt:

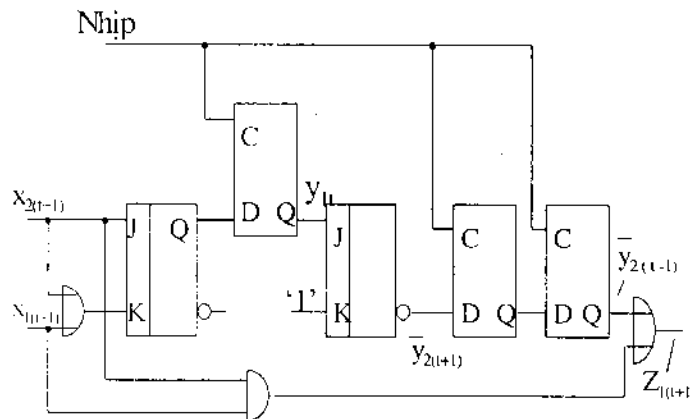
$$x_{1(t+1)}^2 = \varphi_2(y_{2t} = 0) = y_{1t}$$

$$x_{2(t+1)}^2 = \varphi_2(y_{2t} = 1) = y_{1t} + 1 = 1.$$

Áp dụng công thức (11.16), hệ phương trình (11.19) có thể viết dưới dạng thông qua hoạt động của trigơ JK như sau:

$$\left. \begin{aligned} y_{1(t+1)} &= x_{2(t+1)} \bar{y}_{1t} + (x_{2(t+1)} + x_{1(t+1)}) \cdot y_{1t} \\ y_{2(t+1)} &= y_{1t} \cdot \bar{y}_{2t} + y_{2t} \\ Z_{1(t+1)} &= x_{1(t+1)} \cdot x_{2(t+1)} + \bar{y}_{2(t+1)} \end{aligned} \right\} \quad (11.20)$$

Hệ hàm (11.20) có thể thực hiện bằng mạch ở hình 11.12.



Hình 11.12. Mạch thực hiện hệ hàm 11.20

b) Trường hợp dùng trigơ T (một đầu vào).

Sử dụng công thức (12.17) đối với $y_{1(t+1)}$, ta có:

$$\begin{aligned} x_{1(t+1)}^1 &= x_{2(t+1)} \cdot \bar{y}_{1t} + \overline{(x_{2(t+1)} + x_{1(t+1)}) \cdot y_{1t}} \\ &= x_{2(t+1)} \cdot \bar{y}_{1t} + \bar{x}_{2(t+1)} \cdot \bar{x}_{1(t+1)} \cdot y_{1t} \end{aligned}$$

Đối với $y_{2(t+1)}$ ta có:

$$x_{1(t+1)}^2 = y_{1t} \cdot \bar{y}_{2t} + y_{2t} \cdot 0 = y_{1t} \cdot \bar{y}_{2t} \quad (11.21)$$

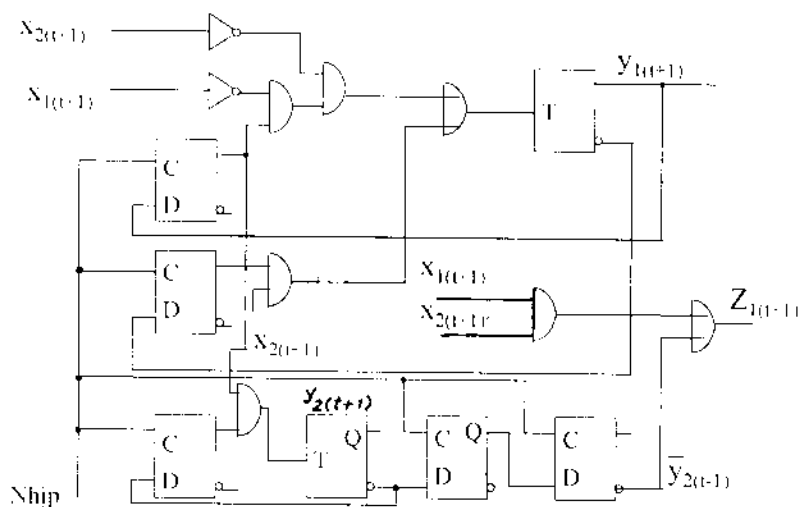
Do đó:

$$y_{1(t+1)} = \overline{x_{2(t+1)} \cdot \bar{y}_{1t} + \bar{x}_{2(t+1)} \cdot \bar{x}_{1(t+1)} \cdot y_{1t}} \cdot y_{1t} + [x_{2(t+1)} \cdot y_{1t} + \bar{x}_{2(t+1)} \cdot \bar{x}_{1(t+1)} \cdot y_{1t}] \bar{y}_{1t}$$

$$y_{2(t+1)} = y_{1t} \cdot \bar{y}_{2t} \cdot y_{2t} + y_{1t} \cdot \bar{y}_{2t} \cdot \bar{y}_{2t}$$

$$Z_{1(t+1)} = x_{1(t+1)} \cdot x_{2(t+1)} + \bar{y}_{2(t+1)}$$

Mạch thực hiện hệ phương trình (11.21) dùng trigơ T cho trên hình 11.13.



Hình 11.13. Mạch thực hiện hệ hàm 11.21

11.3. TỔNG HỢP THIẾT BỊ SỐ CHO BỞI HÀM LÔGIC TRUY TOÁN LOẠI II

Thiết bị số có chức năng được mô tả bằng hàm logic truy toán loại II có đặc trưng là tín hiệu ra của nó phụ thuộc không những vào giá trị trạng thái vào tại thời điểm đang xét, mà còn phụ thuộc vào các trạng thái vào tại những thời điểm trước đó. Như trong chương 4 đã chỉ rõ, hàm logic truy toán loại II có dạng:

$$y_t = \varphi(x_{1t}, x_{2t}, \dots, x_{1(t-1)}, \dots, x_{n(t-1)}, \dots, x_{1(t-r)}, \dots, x_{n(t-r)}) \quad (11.22)$$

Căn cứ vào (11.22) ta thấy rằng, để tổng hợp mạch, các tín hiệu đầu vào cần được giữ chậm từ 1 đến r nhịp. Như vậy một cách tổng quát, mạch thực hiện một hàm logic truy toán loại II sẽ gồm một bộ phận giữ chậm các tín hiệu vào từ 1 đến r nhịp và một hệ tổ hợp gồm $(n.r + n)$ biến vào và một biến ra y .

Như ta đã biết, mọi sự giữ chậm tín hiệu đều có thể thực hiện bằng các mạch tích cực hay thụ động. Trong nhiều trường hợp, thường dùng các phần tử nhớ như trigơ D để giữ chậm tín hiệu.

Để tổng hợp phần hệ tổ hợp, tất cả các tín hiệu x_{it} đến $x_{i(t-r)}$ được coi ngang bằng nhau và có thể thực hiện theo các bước tổng hợp hệ hàm như chương 5 đã trình bày.

Thí dụ 11.6

Thiết kế mạch thực hiện hàm logic truy toán loại II sau đây:

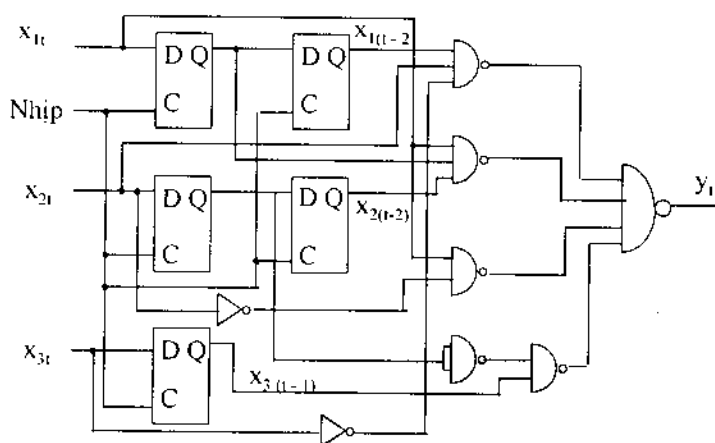
$$Y_t = x_{1t} \cdot x_{1(t-1)} \cdot x_{2(t-2)} + \bar{x}_{2t} \cdot x_{1t} + x_{1(t-2)} \cdot \bar{x}_{3t} \cdot x_{2t} + \bar{x}_{2(t-1)} \cdot x_{3(t-1)}$$

Để tạo các tín hiệu $x_{1(t-1)}$, $x_{1(t-2)}$, $x_{2(t-1)}$, $x_{2(t-2)}$ và $x_{3(t-1)}$ ta cần dùng các trigơ D. Mạch thực hiện tiếp theo hàm logic truy toán loại II đã cho thực chất là thực hiện mạch tổ hợp tám đầu vào, một đầu ra. Vì hàm logic tám biến đã cho là một hàm đã được đơn giản, do đó việc thực hiện bằng các mạch logic loại VA, HOAC và KH rất dễ dàng.

Để thực hiện mạch trên cơ sở các phần tử NAND và trigơ D, ta biến đổi theo algôrit đã trình bày trong chương 2:

$$y_t = (x_{1t} | x_{1(t-1)} | x_{2(t-2)}) ((\bar{x}_{2t} | x_{1t}) | (x_{1(t-2)} | \bar{x}_{3t} . x_{2t})) ((\bar{x}_{2(t-1)} | x_{3(t-1)})$$

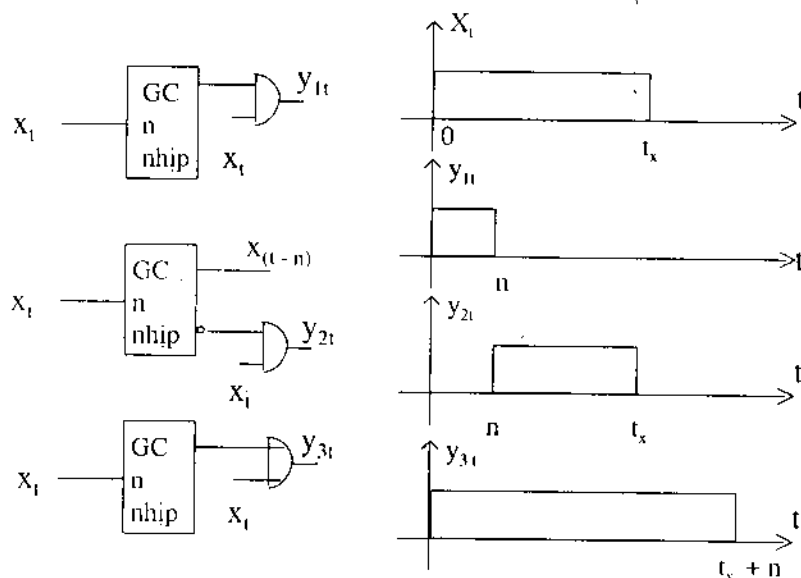
Mạch thực hiện cho trên hình 11.14.



Hình 11.14. Mạch thực hiện hàm của thí dụ 11.6

Nhờ có hàm logic truy toán loại II, ta có thể thực hiện nhiều loại mạch trong thực tế như các mạch làm ngắn hoặc mở rộng tín hiệu. Thí dụ, hàm thực hiện làm ngắn tín hiệu đi n nhịp kể từ sườn trước của tín hiệu có thể viết như sau:

$$y_{1t} = x_t . x_{t-n} \quad (11.23)$$



Hình 11.15. Mạch thực hiện các hàm (11.23) đến (11.25)

Hàm làm ngắn tín hiệu đi ($t_s - n$) đơn vị thời gian (n chu kỳ nhịp) có dạng:

$$y_{2t} = \overline{x_{(t-n)}} \cdot x_t \quad (11.24)$$

Hàm làm rộng tín hiệu ra thêm n chu kỳ xung nhịp có dạng:

$$y_{3t} = x_{(t-n)} + x_t \quad (11.25)$$

Mạch thực hiện các hàm (11.23), (11.24) và (11.25) cho trên hình 11.15.

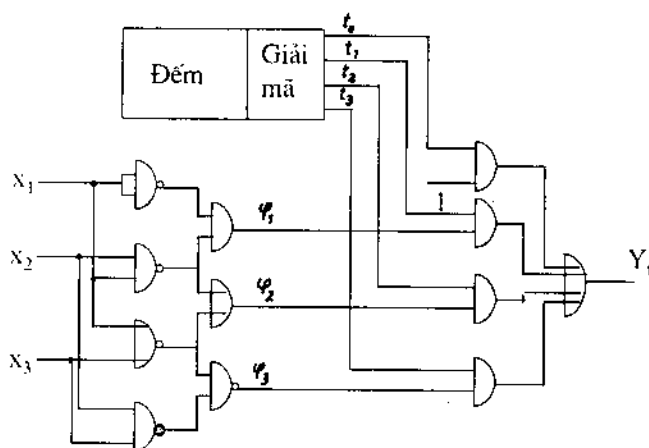
11.4. PHÂN TÍCH THIẾT BỊ SỐ XÂY DỰNG TRÊN CƠ SỞ HÀM LOGIC PHỤ THUỘC THỜI GIAN VÀ HÀM LOGIC TRUY TOÁN

Phân tích thiết bị số xây dựng trên cơ sở hàm logic phụ thuộc thời gian và hàm logic truy toán là nhằm rút ra được chức năng hoạt động của các mạch đó dưới dạng hàm logic phụ thuộc thời gian hay hàm logic truy toán. Các hàm này là biểu hiện của sự phụ thuộc của các biến ra vào các biến vào trong những khoảng thời gian cụ thể. Trên kết quả thu được ta có thể kiểm tra kết quả thiết kế hay chỉ ra những đặc điểm hoạt động của thiết bị.

Khi phân tích, nếu các thiết bị có sơ đồ chức năng dạng cây, không có phản hồi và có nhiều phần tử giữ chậm nằm rải rác trên đường truyền tín hiệu từ đầu vào tới đầu ra của mạch thì, trong nhiều trường hợp, bằng cách lần theo đường truyền của các tín hiệu từ đầu ra tới các đầu vào và quy toàn bộ thời gian giữ chậm của mỗi tín hiệu về một phần tử giữ chậm ở ngay đầu vào sẽ dễ dàng trong việc phân tích mạch.

Để phân tích mạch, trước hết cần xác định các đầu vào, đầu ra và các đầu điều khiển của mạch. Trên cơ sở đó áp dụng quy tắc chung về phân tích mạch để tìm các hàm chức năng của mạch, bằng cách thay thế dần các biến trung gian lần lượt từ đầu ra tới đầu vào.

Thí dụ 11.7: Phân tích mạch cho trên hình 11.16.



Hình 11.16. Mạch dùng để phân tích

Các tín hiệu vào x_1, x_2, x_3 . Tín hiệu ra $Y_i = f(x_1, x_2, x_3)$. Song sự phụ thuộc của y vào các biến vào như thế nào còn phụ thuộc vào giá trị thời gian $t_i, i = \overline{0, 3}$. Sự xuất hiện của các giá trị thời gian sẽ quyết định giá trị cụ thể của hàm y .

Theo sơ đồ mạch ở hình 11.16, ta có:

$$y = \varphi_0.t_0 + \varphi_1.t_1 + \varphi_2.t_2 + \varphi_3.t_3$$

trong đó:

$$\varphi_0 = 1$$

$$\varphi_1 = \overline{x_1}(\overline{x_1} \overline{x_2}) = \overline{x_1} + \overline{x_1} \overline{x_2} = \overline{x_1}$$

$$\varphi_2 = \overline{x_1 x_2} + \overline{x_1 + x_3} = \overline{x_1} + \overline{x_2} + \overline{x_1} \overline{x_3} = \overline{x_1} + \overline{x_2}$$

$$\varphi_3 = \overline{(x_1 + x_2) \cdot (x_2 x_3)} = x_1 + x_3$$

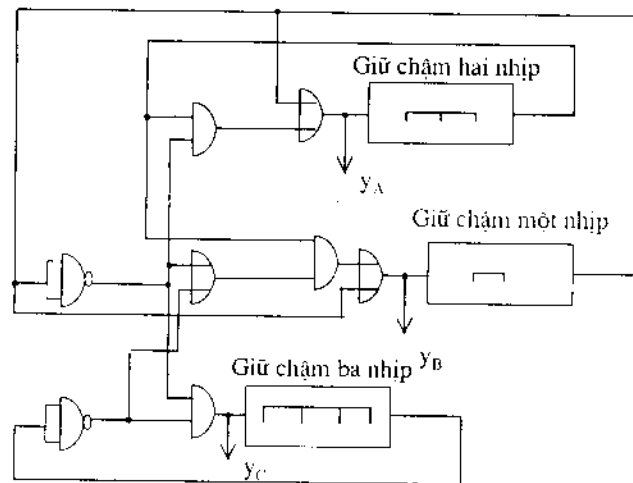
hay:

$$y = t_0 + \overline{x_1}.t_1 + (\overline{x_1} + \overline{x_2}).t_2 + (x_1 + x_3).t_3$$

Trong thời gian $t_0 = 1$, các $t_1 = t_2 = t_3 = 0$, đầu ra của thiết bị có giá trị logic 1, không phụ thuộc vào các giá trị của các biến vào x_1, x_2, x_3 . Khi $t_1 = 1$, các t_0, t_2 và t_3 đều bằng 0. Khi đó y là hàm đảo của biến x_1 .

Một cách tương tự, ta xét cho các trường hợp khi t_2, t_3 có giá trị bằng 1 và quá trình cứ như thế lặp lại.

Thí dụ 11.8: Phân tích hoạt động của ôtomát ôtonôm cho trên hình 11.17.



Hình 11.17. Ôtomát ôtonôm dùng phân tích

Trong trường hợp này, thiết bị có ba đầu ra: y_A, y_B, y_C , ta có:

$$y_{A(t+3)} = y_{A(t+1)} \cdot \overline{y_{B(t+2)}} + y_{B(t+2)}$$

$$y_{B(t+3)} = y_{A(t+1)} \cdot (\overline{y_{B(t+2)}} + \overline{y_{C1}}) + y_{B(t+2)}$$

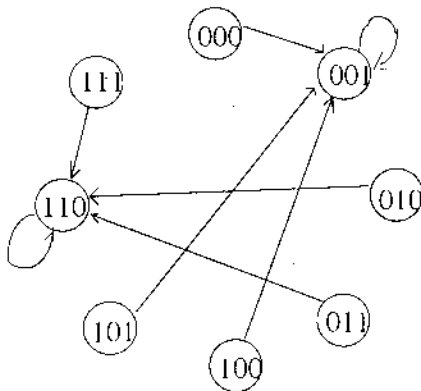
$$y_{C(t+3)} = \overline{y_{C1}} \cdot \overline{y_{B(t+2)}}$$

Để tiếp tục phân tích hoạt động của mạch, ta giả thiết rằng ở thời điểm trước khi xuất hiện t , toàn bộ $y_A = y_B = y_C = 0$, ứng với trạng thái ban đầu của mạch. Từ đó ta lập bảng giá trị thực cho y_A, y_B, y_C tại các thời điểm $(t + 1), (t + 2), (t + 3)$, căn cứ vào giá trị của chúng tại thời điểm t và trước thời điểm t .

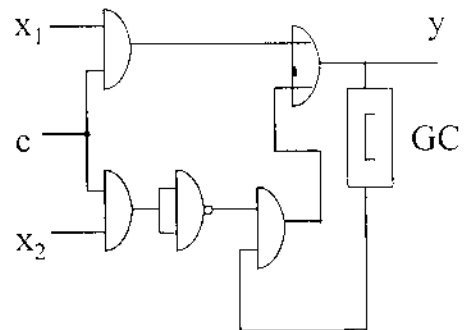
Từ hệ phương trình trên đây ta thu được bảng giá trị thực của các hàm như sau:

t			t + 1			t + 2			t + 3		
y_A	y_B	y_C	y_A	y_B	y_C	y_A	y_B	y_C	y_A	y_B	y_C
0	0	0	0	0	1	0	0	1	0	0	1
0	0	1	0	0	1	0	0	1	0	0	1
0	1	0	1	1	0	1	1	0	1	1	0
0	1	1	1	1	0	1	1	0	1	1	0
1	0	0	0	0	1	0	0	1	0	0	1
1	0	1	0	0	1	0	0	1	0	0	1
1	1	0	1	1	0	1	1	0	1	1	0
1	1	1	1	1	0	1	1	0	1	1	0

Trên cơ sở bảng này, có thể vẽ đồ hình trạng thái của mạch như ở hình 11.18.



Hình 11.18. Đồ hình trạng thái



Hình 11.19. Mạch cho thí dụ 11.9

Trong đó hình thành rõ nét hai lớp trạng thái ban đầu dẫn tới những hoạt động khác nhau của hệ. Cụ thể, nếu trạng thái ban đầu khi bật nguồn của y_A, y_B và y_C là một trong các trạng thái sau đây: 010, 011, 110 và 111 thì ngay sau chu kỳ tiếp theo, ô tô-mat đã rơi vào trạng thái bên 110 và không thay đổi trạng thái nữa. Ở đầu ra $y_A = y_B = 1$ và $y_C = 0$.

Nếu trạng thái ban đầu khi bật nguồn là 000, 001, 100 hoặc 101 thì ô tô máy sẽ rơi vào trạng thái bên là trạng thái 001.

Như vậy hoạt động của mạch đã cho phụ thuộc vào cấu tạo mạch và trạng thái ban đầu của nó.

Thí dụ 11.9: Phân tích cho trên hình 11.19, với điều kiện $x_1, x_2 = 0$.

Từ mạch đã cho, dễ dàng nhận được:

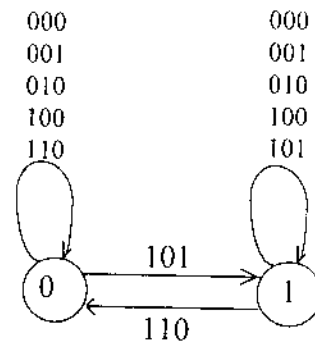
$$y = C_1 \cdot x_{1t} + \overline{C_1} \cdot x_{2t} \cdot y_{(t-1)}$$

Để hiểu rõ hoạt động của mạch, ta lập bảng giá trị thực và sau đó lập đồ hình trạng thái như ở hình 11.20.

So sánh với đồ hình và hoạt động của các phần tử nhớ có hai trạng thái, ta thấy mạch hình 11.20 làm việc như một trigơ CRS. Trong đó đã chú ý tới tổ hợp cấm: Nếu thoả mãn điều kiện $x_1, x_2 = 0$ thì có nghĩa là x_1 và x_2 không được phép đồng thời bằng 1. Đó cũng là điều kiện của trigơ CRS.

Đối với mạch này x_1 đóng vai trò như đầu vào S, còn x_2 như đầu vào R; c là tín hiệu đồng bộ; y_{t-1} là trạng thái trong ở thời điểm trước đó; y_t là trạng thái trong được xác lập (trạng thái mới).

c_t	x_{2t}	x_{1t}	y_{t-1}	y_t
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	-
0	1	1	1	-
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	-
1	1	1	1	-



Hình 11.20. Đồ hình trạng thái của mạch ở hình 11.19

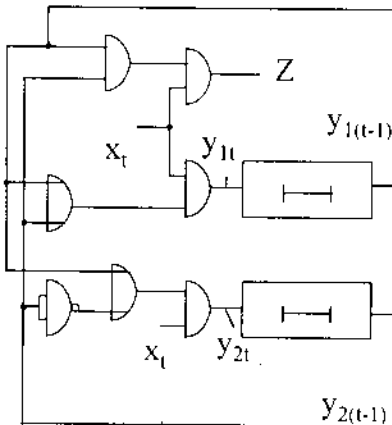
Thí dụ 11.10: Xác định chức năng hoạt động của mạch cho trên hình 11.21.

Trên hình 11.21, nếu trước phần tử giữ chậm ta ghi y_{1t} và y_{2t} thì sau phần tử giữ chậm sẽ là $y_{1(t-1)}$ và $y_{2(t-1)}$ tương ứng.

Từ mạch đã cho, theo phương pháp thay thế lần lượt các biến trung gian ta có:

$$\left. \begin{aligned} y_{1t} &= x_t(y_{1(t-1)} + y_{2(t-1)}) \\ y_{2t} &= x_t(y_{1(t-1)} + \bar{y}_{2(t-1)}) \\ Z &= x_t \cdot y_{1(t-1)} \cdot y_{2(t-1)} \end{aligned} \right\} \quad (11.23)$$

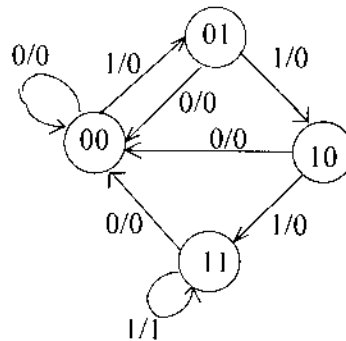
Trên cơ sở đó ta lập bảng giá trị thực của các hàm và được bảng giá trị thực hình 11.22.



Hình 11.21. Mạch cho thí dụ 11.10

x_t	$y_{1(t-1)}$	$y_{2(t-1)}$	y_{1t}	y_{2t}	Z
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	1	0
1	0	1	1	0	0
1	1	0	1	1	0
1	1	1	1	1	1

Hình 11.22. Bảng giá trị thực



Hình 11.23. Đồ hình trạng thái

Căn cứ vào giá trị đã có trong bảng, ta xây dựng đồ hình trạng thái của mạch như ở hình 11.23, trong đó trạng thái của mạch được xác định bởi hai biến y_1 và y_2 . Biến ra là Z . Biến x_t là biến kích thích. Ta nhận thấy rằng, với mọi trạng thái, nếu $x_t = 0$ thì dù xuất phát từ trạng thái nào, mạch cũng trở về trạng thái $y_1 y_2 = 00$ và tín hiệu ra $Z = 0$. Nếu xuất phát từ trạng thái trong $y_1 y_2 = 00$, dưới tác động của $x_t = 1$, mà thời gian tồn tại của x_t nhỏ hơn ba nhịp thì mạch lại trở lại trạng thái ban đầu $y_1 y_2 = 00$ và $Z = 0$.

Nhưng nếu thời gian tồn tại của x_i lớn hơn hay bằng ba nhịp thì mạch sẽ tới trạng thái $y_1 y_2 = 11$ và ở đó, dưới tác động của $x_i = 1$, tín hiệu đầu ra $Z = 1$. Khi x_i trở về trạng thái 0 thì mạch lại trở về trạng thái $y_1 y_2 = 00$.

Qua phân tích ta thấy rằng, mạch ở hình 11.21 là một mạch phân biệt độ rộng xung của tín hiệu x_i . Nếu x_i có độ rộng lớn hơn 3 nhịp thì đầu ra cho $Z = 1$, ngược lại thì $Z = 0$.

BÀI TẬP CHƯƠNG 11

1. Tổng hợp các hàm logic phụ thuộc thời gian sau đây, trong đó các biến thời gian chỉ nhận một trong hai giá trị 0 hoặc 1.

$$a) y_1 = ((x_1 + x_2 x_3) \cdot x_4 + \bar{x}_1 x_2 \bar{x}_4) t_0 + ((\bar{x}_1 x_3 + x_2 \bar{x}_3 x_4)(\bar{x}_1 x_4 + x_2)) t_1$$

$$b) y_2 = \bar{x}_1 \bar{x}_2 \bar{x}_4 t_0 + (x_1 + x_2) x_3 t_1 + \bar{x}_2 t_2 + x_2 x_3 t_3$$

$$c) y_3 = ((x_1 \bar{x}_3 + \bar{x}_1 x_2 x_3) t_0 + (\bar{x}_1 x_3 + x_2 x_4)) t_2 + \bar{x}_3 x_4 t_3 + (x_2 + x_3) t_4$$

$$d) y_4 = \varphi_1(1, 2, 3, 4, 5, 6, 7, 11) t_1 + \varphi_2(1, 2, 3, 8) t_2 + \varphi_3(1, 2, 5, 6, 10, 11, 12) t_3 + \varphi_4(5, 6, 9) t_4 + \varphi_5(1, 2, 5, 6, 10, 12) t_5.$$

2. Tổng hợp các hàm trong bài 1, trong đó phần phối hợp thời gian dùng các vi mạch trộn đường (multiplexer) thích hợp.
3. Tổng hợp thiết bị cho bởi hàm logic truy toán suy biến loại I sau đây:

$$a) y_{1t} = y_{1(t-1)} y_{2(t-1)} + \bar{y}_{1(t-2)}$$

$$y_{2t} = y_{1(t-1)} y_{2(t-2)} + y_{2(t-1)}$$

$$b) y_{1t} = y_{1(t-1)} + y_{2(t-1)} \bar{y}_{3(t-2)}$$

$$y_{2t} = y_{2(t-1)} + \bar{y}_{1(t-2)} y_{3(t-1)}$$

$$y_{3t} = \bar{y}_{1(t-1)} y_{2(t-1)} + y_{3(t-1)}$$

Trong đó dùng trigơ D, thí dụ 7474, làm phần tử giữ chậm.

4. Tổng hợp thiết bị số cho bởi bảng hoạt động sau:

$y_{1t} \ y_{2t} \ y_{3t}$	$y_{1(t+1)} \ y_{2(t+1)} \ y_{3(t+1)}$	$y_{1t} \ y_{2t} \ y_{3t}$	$y_{1(t+1)} \ y_{2(t+1)} \ y_{3(t+1)}$
0 0 0	1 0 0	1 0 0	1 1 0
0 0 1	0 1 1	1 0 1	1 1 0
0 1 0	1 0 1	1 1 0	0 0 1
0 1 1	1 1 0	1 1 1	0 1 1

5. Tổng hợp thiết bị số cho bởi hàm logic sau đây, dùng trigơ T hoặc JK:

$$\varphi_{1(t+1)} = x_{1(t+1)} x_{2(t+1)} \varphi_{1t} + x_{2(t+1)}$$

$$\varphi_{2(t+1)} = \varphi_{1t} + \varphi_{2t} \bar{x}_{1(t+1)}$$

$$\varphi_{3(t+1)} = \bar{x}_{1(t+1)} x_{2(t+1)} + \varphi_{2(t-1)}$$

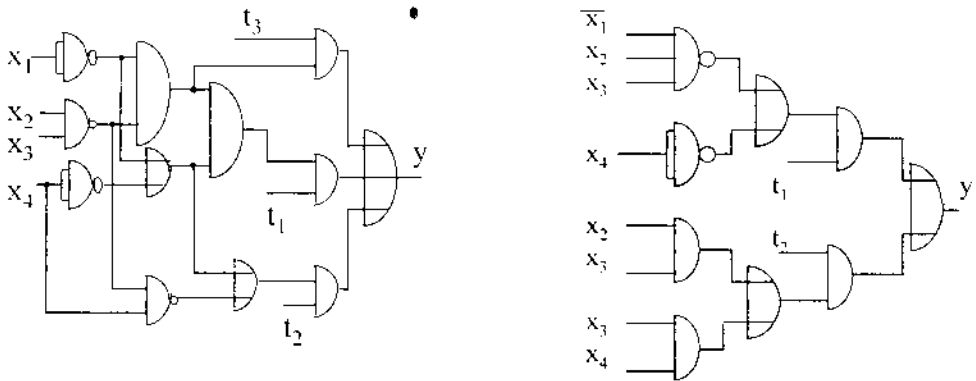
6. Tổng hợp thiết bị số cho bởi các hàm logic sau:

$$a) f_1 = x_{1t} x_{2(t-1)} + \bar{x}_{1t} (\bar{x}_{2(t-2)} + x_{3t})$$

$$b) f_1 = (x_{1t} \{ x_{2(t-1)} \} | (x_{1t} | x_{2(t-1)}) | x_{3(t-2)}) | \bar{x}_{3t}$$

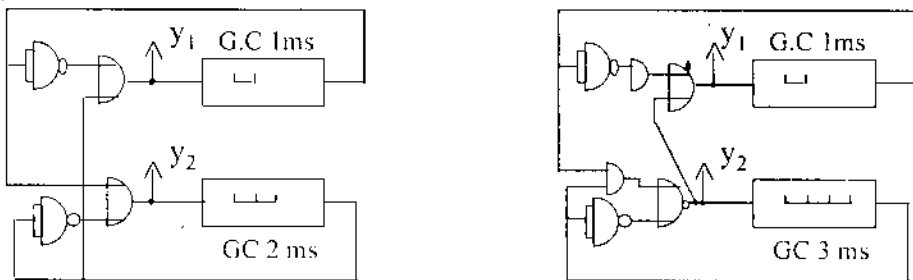
$$c) f_1 = x_{2t} x_{1(t-1)} + \bar{x}_{2(t-1)} x_{3t} + x_{3(t-2)} x_{1t}$$

7. Xác định hàm chức năng của mạch cho trên hình 11.24.



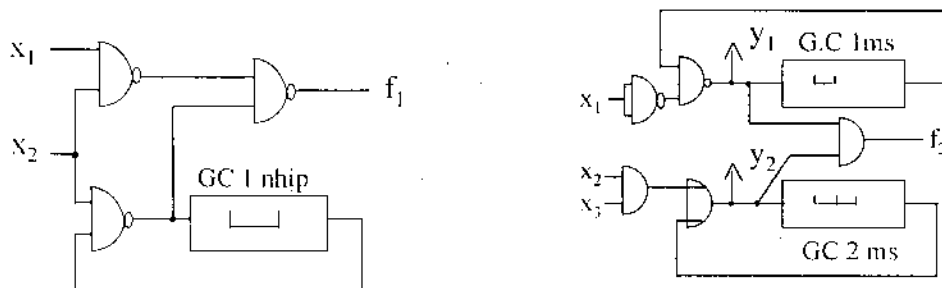
Hình 11.24

8. Xác định chức năng hoạt động của thiết bị số cho trên hình 11.25 dưới dạng biểu thức đại số. Vẽ đồ hình trạng thái của thiết bị.



Hình 11.25

9. Phân tích hoạt động của thiết bị số cho trên hình 11.26.



Hình 11.26

Chương 12

PHÂN TÍCH THIẾT BỊ SỐ CÓ NHỚ

Mục đích: Nhằm xác định chức năng và đặc điểm hoạt động của thiết bị số có nhớ cho trước. Chức năng của thiết bị trong trường hợp này phải được mô tả dưới dạng một ngôn ngữ chuẩn như bảng giá trị thực và bảng ra hoặc dưới dạng đồ hình trạng thái. Đồng thời thông qua các dạng ngôn ngữ mô tả chuẩn đó, nhận xét các đặc điểm hoạt động của thiết bị đã cho, những yêu cầu đối với các tín hiệu vào và thực hiện phân tích để phát hiện các nguyên nhân làm sai lệch hoạt động chức năng chung của hệ.

Mặt khác, kết quả của bài toán phân tích cũng cho phép ta có điều kiện thiết kế lại thiết bị đã cho bằng các phần tử logic và phần tử nhớ khác mà vẫn đảm bảo chức năng như thiết bị đã cho.

Cũng tương tự như bài toán thiết kế, phân tích mạch số có nhớ chia làm ba trường hợp:

- + Phân tích đối với thiết bị số có nhớ dùng phần tử nhớ được nhíp.
- + Phân tích đối với thiết bị số có nhớ làm việc với tín hiệu thế dùng phần tử nhớ không được nhíp.
- + Phân tích thiết bị số có nhớ làm việc với tín hiệu vào là xung.

12.1. PHÂN TÍCH THIẾT BỊ SỐ CÓ NHỚ DÙNG PHẦN TỬ NHỚ ĐƯỢC NHÍP

12.1.1. Đặc điểm

Hoạt động của thiết bị được thực hiện theo tín hiệu nhíp CLOCK (đồng bộ). Trong trường hợp này cần có một số hạn chế sau đây:

- Trong thời gian có xung nhíp, các tín hiệu vào không được thay đổi giá trị. Sự thay đổi của tín hiệu vào trong trường hợp này sẽ làm sai lệch hoạt động của thiết bị.
- Thời gian tồn tại của xung nhíp phải đủ lớn để đầu ra kịp thiết lập trạng thái mới ổn định.
- Tín hiệu vào có thể coi là lý tưởng.
- Thời gian giữ chậm tín hiệu của các phần tử logic được coi bằng 0.

Từ đây ta thấy, trong trường hợp này ô tômát chỉ thực hiện các chuyển biến đơn và các phần tử nhớ thay đổi trạng thái đồng thời khi có xung nhịp. Do đó vấn đề tranh chấp trạng thái là không có.

Bài toán phân tích trong trường hợp này chỉ là phân tích tĩnh để tìm ra chức năng của hệ.

12.1.2. Các bước phân tích

a) Đưa mạch về dạng chuẩn của ô tômát, gồm hai phần: Phần mạch tổ hợp và phân các phần tử nhớ. Ký hiệu và xác định số biến vào (n); số biến ra (m), số biến trong, tức số phần tử nhớ (p). Từ đó có thể xác định số trạng thái vào $N = 2^n$, số trạng thái ra $M = 2^m$ và số trạng thái trong $P = 2^p$.

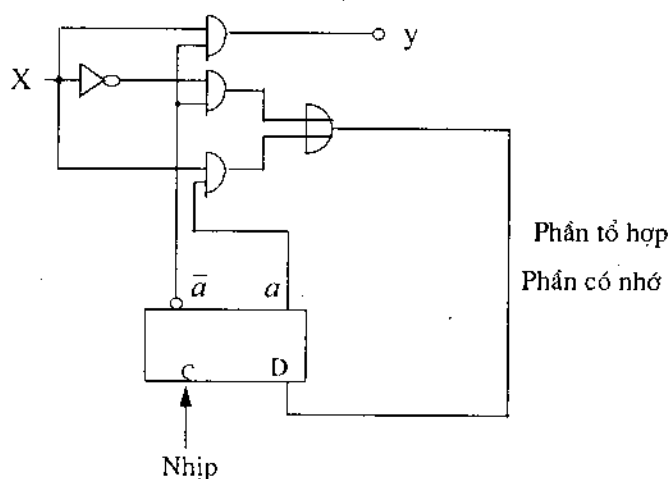
b) Bằng bài toán phân tích tĩnh mạch tổ hợp, rút ra các hàm kích thích và hàm ra của mạch và biểu diễn chúng lên bảng kế cận hoặc bảng đối xứng.

c) Căn cứ bảng kích thích của các phần tử nhớ của mạch và kết quả của bước b) để xây dựng bảng chuyển đổi trạng thái và bảng ra của mạch đã cho. Dựng đồ hình trạng thái và biểu đồ thời gian nếu cần. Đến đây về nguyên tắc chúng ta đã xác định được chức năng làm việc của mạch.

d) Xây dựng bảng chuyển đổi trạng thái và bảng ra hình thức, nếu cần thiết kẻ lại bằng các phần tử logic và nhớ khác.

f) Phân tích để biết hoạt động của thiết bị, đánh giá hoạt động của ô tômát.

Thí dụ 12.1: Cho thiết bị số có nhớ trên hình 12.1.



Hình 12.1. Thiết bị số có nhớ dùng phân tích

1. Để phân tích mạch, trước hết ta đưa mạch về dạng gồm hai phần: Phần mạch tổ hợp và phần mạch có nhớ. Ta thấy có một biến vào, ký hiệu là x ; một biến trong ký hiệu là a ; một biến ra ký hiệu là y ; một phần tử nhớ là trigơ CD, là loại phần tử nhớ làm việc có tín hiệu đồng bộ c .

2. Để rút ra các hàm ra và hàm kích thích, ta dùng phương pháp phân tích tĩnh mạch tổ hợp. Các hàm được rút ra như sau:

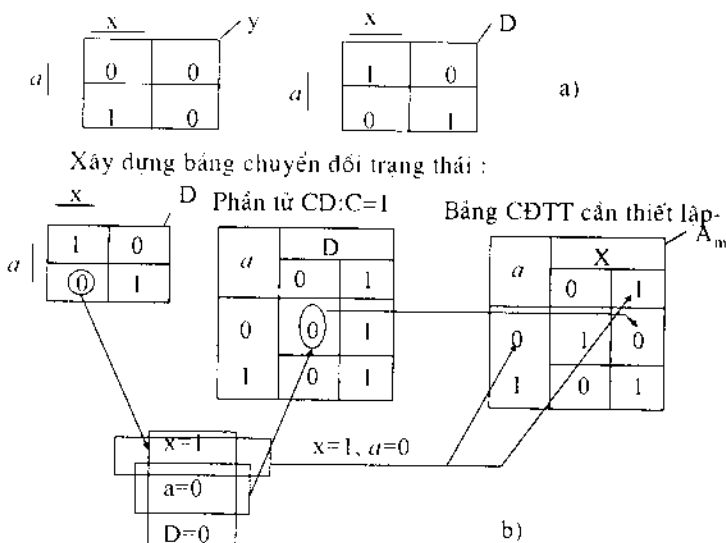
$$y = \bar{a}.x$$

$$D = \bar{a}.\bar{x} + a.x$$

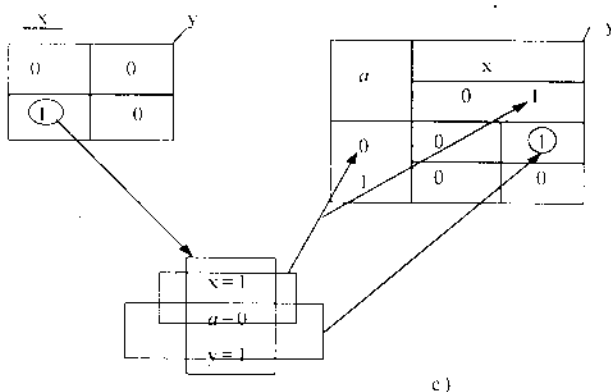
Hai hàm y và D được biểu diễn trên bảng kế cận như ở hình 12.2a.

3. Để thực hiện bước 3, tức xây dựng bảng chuyển đổi trạng thái và bảng ra của mạch đã cho, ta cần lưu ý rằng, vì có một biến vào, do đó có hai trạng thái vào, bảng chuyển đổi trạng thái và bảng ra có hai cột. Vì số biến trong bảng 1 và do đó có hai trạng thái ra. Bảng chuyển đổi trạng thái và bảng ra có hai hàng, mỗi hàng ứng với một trạng thái trong.

Phần còn lại là điền vào tất cả các ô của bảng các giá trị của trạng thái trong và trạng thái ra tương ứng. Để điền vào bảng ra, chúng ta chỉ việc chuyển các giá trị của hàm y trong hình 12.2a sang bảng ra theo các ô có cùng giá trị tọa độ. Để điền vào các ô của bảng chuyển đổi trạng thái, ta phải kết hợp giữa bảng hoạt động hay bảng kích thích của trigơ CD với bảng kế cận của hàm kích thích D . Các quá trình điền vào các ô tương ứng có thể mô tả như trên hình 12.2b, c.



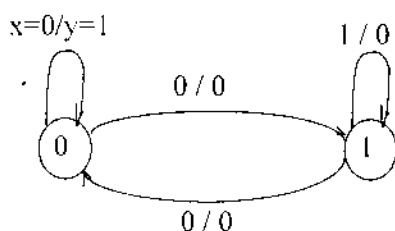
Hình 12.2a, b. Cách xây dựng bảng chuyển đổi trạng thái



Hình 12.2c. Cách xây dựng bảng ra

Thí dụ, khi $x = 1$, $a = 0$ thì $D = 0$. Ở đây $a = 0$ được coi là trạng thái xuất phát a_c của mạch. Đối chiếu hai điều kiện: $a_c = 0$ và $D = 0$ ở bảng hoạt động hay bảng kích thích của trigger CD khi $C = 1$, ta thấy trạng thái mới của nó bằng 0. Đó cũng chính là trạng thái mới a_m cần điền vào ô tương ứng $x = 1$ và $a_c = 0$ của bảng chuyển đổi trạng thái. Làm tương tự với mọi ô của bảng, ta có bảng chuyển đổi trạng thái như ở hình 12.2b.

Đồ hình trạng thái của mạch đã cho có thể nhận được bằng cách chuyển thẳng sang từ bảng chuyển đổi trạng thái và bảng ra. Ta có đồ hình có hai trạng thái với các bước chuyển như trong bảng chuyển đổi trạng thái (hình 12.3).



Hình 12.3. Đồ hình trạng thái xây dựng từ bảng chuyển đổi trạng thái và bảng ra ở hình 12.2

Bảng chuyển đổi trạng thái và bảng ra cũng như đồ hình trạng thái thu được trên đây còn đang ở dạng cụ thể. Có nghĩa là nội dung và cấu trúc cụ thể của chúng gắn chặt với các phần tử nhớ và phương pháp mã hoá các trạng thái vào, trong và các trạng thái ra của quá trình phân tích. Trong trường hợp cần thiết kế lại với dụng ý sẽ sử dụng các loại phần tử khác và có thể mã hoá lại hoàn toàn khác, ta cần đưa chúng về dạng hình thức. Để làm được việc này, ta gán cho mỗi tổ hợp mã một ký tự hay ký hiệu nhất định và thay chúng vào bảng và đồ hình, sẽ được bảng chuyển đổi trạng thái và bảng ra ở dạng hình thức.

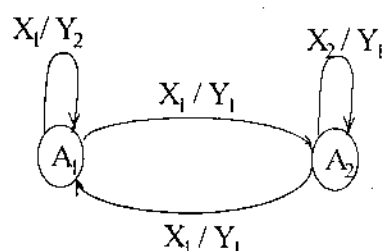
Thí dụ, một trong các phương án gán đối với ô tô máy đã cho trên bảng ở hình 12.3 như bảng ở hình 12.4.

Trạng thái trong	Ký hiệu	Trạng thái vào	Ký hiệu	Trạng thái ra	Ký hiệu
0	A_1	0	X_1	0	Y_1
1	A_2	1	X_2	1	Y_2

Hình 12.4. Một trong các phương án mã hoá các trạng thái

Bảng chuyển đổi trạng thái và bảng ra, đồ hình trạng thái đã được hình thức hoá cho trên hình 12.5a, b.

A \ X	X	
	X_1	X_2
A_1	A_2 / Y_1	A_1 / Y_2
A_2	A_1 / Y_1	A_2 / Y_1



a) Bảng chuyển đổi trạng thái và bảng ra

b) Đồ hình trạng thái

Hình 12.5. Bảng chuyển đổi trạng thái và bảng ra đã được mã hoá

Thông qua đồ hình trạng thái, ta dễ dàng thấy rằng mạch đã cho làm việc với tín hiệu vào là thế hay là xung cũng được vì nó làm việc với xung nhịp. Tuy nhiên, nếu trạng thái vào của các tín hiệu vào là xung thì nó phải được đồng bộ với xung nhịp, nếu không có thế sẽ bị mất thông tin. Nhận xét này được làm rõ hơn nếu ta xây dựng giản đồ thời gian của các tín hiệu a , D và y .

* Khi vẽ giản đồ điện áp của a , D , y , thí dụ cho chuỗi vào:

$x = \{0-1-1-0-1-0-0-1\}$. Chuỗi x có thể có bốn loại:

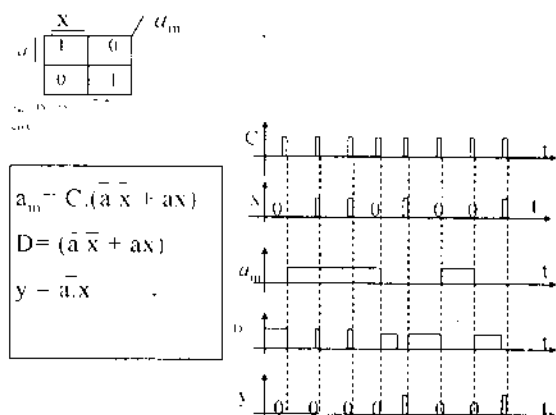
x đồng bộ với C và là chuỗi xung;

x đồng bộ với C và là chuỗi thế;

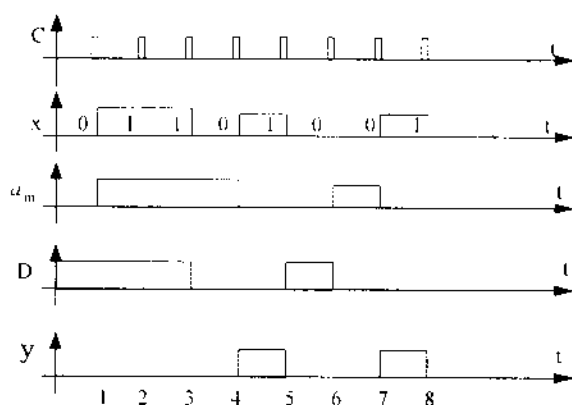
x không đồng bộ với C và là chuỗi xung;

x không đồng bộ với C và là chuỗi thế.

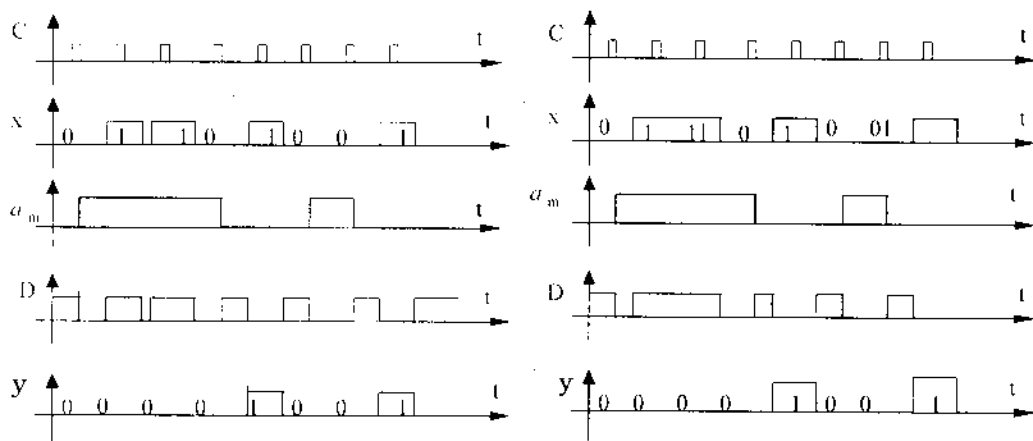
Dưới đây chúng ta sẽ vẽ các giản đồ thời gian cho từng trường hợp trên đây (hình 12.6).



a) x: Xung, đồng bộ với C; D và y cũng đồng bộ với C



b) X: thế, đồng bộ với C; D, y cũng đồng bộ với C



c) X: xung, không đồng bộ với C $\Rightarrow$ y và D sẽ không đồng bộ

d) X: thế, không đồng bộ với C $\Rightarrow$ y và D sẽ không đồng bộ với C

Hình 12.6. Các tình huống về quan hệ đồng bộ giữa trạng thái vào X và xung đồng bộ

Chúng ta nhận xét rằng, đối với phần tử CD, giá trị của các tín hiệu chỉ có ý nghĩa trong thời gian tồn tại của tín hiệu đồng bộ C. Thí dụ về cách vẽ biểu đồ thời gian như sau: Khi $C = 1$, với $x = 0$, $a = 0$ thì $D = 1$ nên tại thời điểm 1 trên trục tọa độ ứng với sườn âm của tín hiệu đồng bộ, phần tử CD chuyển về trạng thái 1, tức $a_m = 1$ và nằm ở trạng thái đó cho tới cuối của xung nhịp tiếp theo - thời điểm 2. Trong thời gian tồn tại xung nhịp thứ hai, ta có $x = 1$, $a = 1$, nên $D = 1$, tức phần tử CD không thay đổi trạng thái... Trạng thái của trigơ CD cũng là trạng thái của mạch chỉ trở về 0 tại cuối thời gian tồn tại của xung nhịp thứ tư vì trong thời gian tồn tại xung nhịp thứ 4, có $x = 0$, $a = 1$. Theo đồ hình trạng thái của trigơ CD, nó sẽ chuyển về trạng thái $a_m = 0$, ... Biến ra $y = 1$ khi và chỉ khi $x = 1$, $a = 0$.

Tóm lại, qua quá trình phân tích ta thấy rằng, mạch đã cho có thể làm việc với các loại tín hiệu vào x khác nhau: xung hoặc thế và độ rộng tín hiệu có khác nhau, nhưng chức năng không hề bị thay đổi.

Như vậy có thể rút ra một kết luận quan trọng là, tất cả thiết bị số có nhớ dùng phần tử được nhịp đều có thể phân tích và tổng hợp theo cùng một phương pháp.

Thí dụ 12.2: Xác định bảng chuyển đổi trạng thái và bảng ra của thiết bị số cho trên hình 12.7.

Mạch này có hai trigơ: CT và CJK. Do đó mạch có 4 trạng thái trong, $P = 4$. Có hai biến vào x_1, x_2 nên có 4 trạng thái vào: 00, 01, 10, 11. Có hai biến ra: y_1, y_2 nên có 4 trạng thái ra: 00, 01, 10, 11.

Các hàm kích thích của các trigơ CT và CJK là hàm T, hàm J và hàm K. Bằng phương pháp phân tích tĩnh mạch tổ hợp, ta dễ dàng rút ra các hàm này như sau:

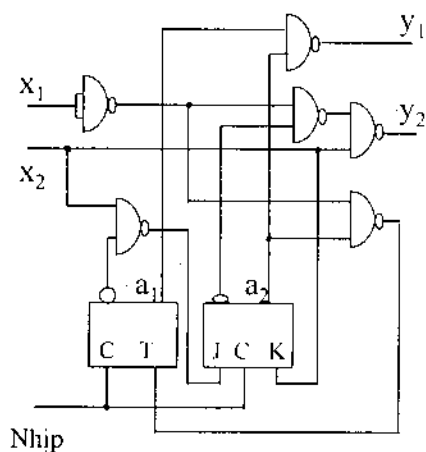
$$T = \overline{x_1} \cdot a_2 = x_1 + \overline{a_2}$$

$$J = \overline{x_1} \cdot \overline{a_1} = \overline{x_1} + a_1$$

$$K = x_2$$

$$y_1 = \overline{a_1} \cdot a_2 = \overline{a_1} + \overline{a_2}$$

$$y_2 = \overline{x_1} \cdot \overline{a_2} \cdot x_2 = \overline{x_2} + \overline{x_1} \overline{a_2}$$



Hình 12.7. Mạch cho thí dụ 12.2

Các hàm trên đây được biểu diễn trên bảng kế cận như ở hình 12.8.

Căn cứ vào các bảng kế cận biểu diễn các hàm T, J, K, y_1 và y_2 và bảng kích thích của hai loại trigơ CT và CJK, có thể xác định các trạng thái trong mới, ứng với các trạng thái trong cũ (a_i) và các trạng thái vào khác nhau cũng như các trạng thái ra để

điền vào bảng chuyển đổi trạng thái và bảng ra của mạch. Thí dụ, để điền vào ô của bảng chuyển đổi trạng thái và bảng ra có tọa độ: Cột $x_1x_2 = 00$, hàng $a_1a_2 = 00$, ta sẽ có $T = 1, J = 1, K = 0, y_1 = y_2 = 0$ và do đó trạng thái trong mới sẽ là $a_{1m}a_{2m} = 11$, trạng thái ra mới: $y_{1m}y_{2m} = 01$.

	x_1	x_2	T
a_1	1	1	1
a_2	1	1	0
a_1	1	0	0
a_2	1	0	0
a_1	0	1	1
a_2	0	1	1
a_1	0	0	0
a_2	0	0	0

	x_1	x_2	J
a_1	1	1	1
a_2	1	1	1
a_1	1	0	0
a_2	1	0	0
a_1	0	1	1
a_2	0	1	1
a_1	0	0	0
a_2	0	0	0

	x_1	x_2	K
a_1	0	1	0
a_2	0	1	0
a_1	0	1	0
a_2	0	1	0
a_1	0	0	0
a_2	0	0	0
a_1	0	0	0
a_2	0	0	0

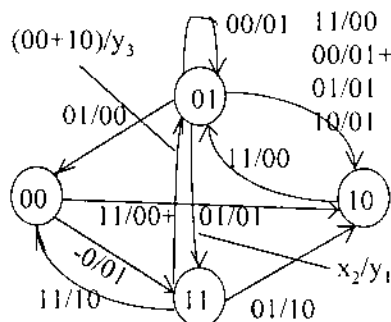
	x_1	x_2	y_1
a_1	1	0	1
a_2	1	0	0
a_1	1	0	0
a_2	1	0	0
a_1	0	1	1
a_2	0	1	1
a_1	0	0	0
a_2	0	0	0

	x_1	x_2	y_2
a_1	1	0	1
a_2	1	0	0
a_1	1	0	0
a_2	1	0	0
a_1	0	1	1
a_2	0	1	1
a_1	0	0	0
a_2	0	0	0

Hình 12.8. Biểu diễn các hàm lên bảng kế cận để đơn giản

Cuối cùng thực hiện với mọi ô của bảng, ta có bảng chuyển đổi trạng thái và bảng ra của mạch đã cho như trên hình 12.9a và đồ hình trạng thái tương ứng (hình 12.9b).

Λ	x_1x_2			
a_1a_2	00	01	10	11
00	11/01	10/01	11/01	10/00
01	01/01	00/00	11/01	10/00
10	01/01	01/01	01/01	01/00
11	01/11	10/10	01/11	00/10



a) Bảng chuyển đổi trạng thái và bảng ra

b) Đồ hình trạng thái

Hình 12.9. Bảng chuyển đổi trạng thái và bảng ra của mạch ở hình 12.7

Đây là loại ô tô-mát xác định đầy đủ với mọi tập giá trị biến vào và biến trong. Để đưa bảng chuyển đổi trạng thái và bảng ra của thiết bị số về dạng hình thức, ta có thể gán cho mỗi tập giá trị biến nhị phân trong bảng ở hình 12.9 trên đây một ký tự như sau:

Trạng thái trong

00	A_0
01	A_1
10	A_2
11	A_3

Trạng thái vào

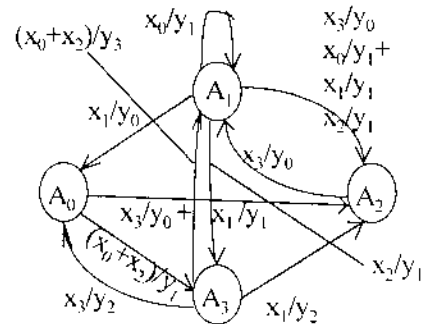
00	X_0
01	X_1
10	X_2
11	X_3

Trạng thái ra

00	Y_0
01	Y_1
10	Y_2
11	Y_3

và sẽ được dạng hình thức của bảng chuyển đổi trạng thái và bảng ra, đồ hình trạng thái như trên hình 12.10a, b.

Λ $a_1 a_2$	$x_1 x_2$			
	X_0	X_1	X_2	X_3
Λ_0	Λ_3/Y_1	Λ_2/Y_1	Λ_3/Y_1	Λ_2/Y_0
Λ_1	Λ_1/Y_1	Λ_0/Y_0	Λ_3/Y_1	Λ_2/Y_0
Λ_2	Λ_1/Y_1	Λ_1/Y_1	Λ_1/Y_1	Λ_1/Y_0
Λ_3	Λ_1/Y_3	Λ_2/Y_2	Λ_1/Y_3	Λ_0/Y_2



a) Bảng chuyển đổi trạng thái và bảng ra

b) Đồ hình trạng thái

Hình 12.10. Bảng chuyển đổi trạng thái và bảng ra dưới dạng hình thức

12.2. PHÂN TÍCH THIẾT BỊ SỐ DÙNG PHẦN TỬ NHỚ LÀM VIỆC VỚI TÍN HIỆU THỂ KHÔNG ĐƯỢC NHỊP

Phân tích thiết bị số dùng các phần tử nhớ không được nhịp phức tạp hơn nhiều. Sự phức tạp và khó khăn ở đây liên quan tới việc phải xác định xem hoạt động của thiết bị đã cho có hiện tượng tranh chấp trạng thái nguy hiểm hay không. Để phát hiện được hiện tượng này cần phải phân tích hiện tượng tranh chấp trạng thái trong phần mạch tổ hợp của thiết bị, trong đó phải tính tới hiện tượng giữ chậm tín hiệu của các phần tử logic. Nếu trong phần mạch tổ hợp hiện tượng tranh chấp chỉ gây nên sai số tại thời điểm nào đấy thì trong mạch có nhớ hiện tượng đó có thể gây nên sai lệch chức năng của thiết bị, vì sai lệch trong một thời điểm nào đó của phần mạch tổ hợp qua mạch phản hồi có thể gây nên sự chuyển biến trạng thái nhầm của các phần tử nhớ và cuối cùng có thể dẫn tới sự sai lệch về chức năng của thiết bị. Hơn thế nữa, sai lệch ấy lại có thể xảy ra một cách ngẫu nhiên nên khó có thể lường trước được.

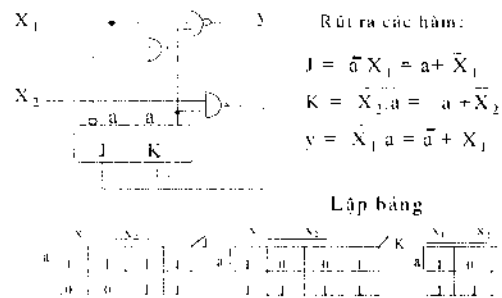
Khi phần tử nhớ không có đầu nhịp (đồng bộ) CLK, việc phân tích sẽ phức tạp hơn và bắt buộc phải tiến hành cả phân tích tĩnh và động.

12.2.1. Phân tích tĩnh

Thứ tự thực hiện nhiệm vụ phân tích tĩnh như trong mục 12.1 đã chỉ ra, trong phần nhận xét cần bổ sung thêm việc nhận xét những hạn chế đối với các tín hiệu vào. Trong bảng chuyển đổi trạng thái cần xác định và đánh dấu các trạng thái bền đối với các trạng thái vào tương ứng, thí dụ bằng một khoanh tròn.

Thí dụ 12.3. Phân tích tĩnh đối với mạch cho trên hình 12.11.

Tiến hành phân tích chúng ta sẽ được kết quả như ở hình 12.11.



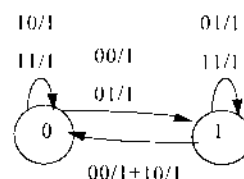
Hình 12.11. Mạch cho thí dụ 12.3

Để có thể xây dựng bảng chuyển đổi trạng thái và bảng ra, chúng ta dựa vào kết quả biểu diễn các hàm và bảng kích thích của trigger CJK khi C = 1 để xác định các trạng thái trong mỗi đối với các bước chuyển của ô tô măt như sau:

$$\begin{array}{l}
 \left. \begin{array}{l} x_1 = 0 \\ x_2 = 0 \\ a = 0 \end{array} \right\} \Rightarrow \begin{array}{l} J = 1 \\ K = 1 \\ y = 1 \end{array} \quad JK = 11 \quad \boxed{1/1} \quad \left. \begin{array}{l} x_1 = 0 \\ x_2 = 0 \\ a = 1 \end{array} \right\} \Rightarrow \begin{array}{l} J = 1 \\ K = 1 \\ y = 1 \end{array} \quad JK = 11 \quad \boxed{0/1} \\
 \left. \begin{array}{l} x_1 = 0 \\ x_2 = 1 \\ a = 0 \end{array} \right\} \Rightarrow \begin{array}{l} J = 1 \\ K = 1 \\ y = 1 \end{array} \quad JK = 11 \quad \boxed{1/1} \quad \left. \begin{array}{l} x_1 = 0 \\ x_2 = 1 \\ a = 1 \end{array} \right\} \Rightarrow \begin{array}{l} J = 1 \\ K = 1 \\ y = 1 \end{array} \quad JK = 11 \quad \boxed{1/1} \\
 \left. \begin{array}{l} x_1 = 1 \\ x_2 = 0 \\ a = 0 \end{array} \right\} \Rightarrow \begin{array}{l} J = 0 \\ K = 1 \\ y = 1 \end{array} \quad JK = 01 \quad \boxed{0/1} \quad \left. \begin{array}{l} x_1 = 1 \\ x_2 = 0 \\ a = 1 \end{array} \right\} \Rightarrow \begin{array}{l} J = 1 \\ K = 1 \\ y = 1 \end{array} \quad JK = 11 \quad \boxed{0/1} \\
 \left. \begin{array}{l} x_1 = 1 \\ x_2 = 1 \\ a = 0 \end{array} \right\} \Rightarrow \begin{array}{l} J = 0 \\ K = 1 \\ y = 1 \end{array} \quad JK = 01 \quad \boxed{0/1} \quad \left. \begin{array}{l} x_1 = 1 \\ x_2 = 1 \\ a = 1 \end{array} \right\} \Rightarrow \begin{array}{l} J = 1 \\ K = 0 \\ y = 0 \end{array} \quad JK = 10 \quad \boxed{1/0}
 \end{array}$$

A	JK			
	00	01	11	10
0	0	0	1	1
1	1	0	0	1

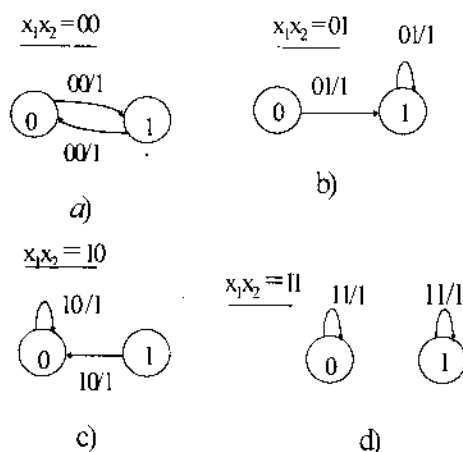
A	X ₁ X ₂			
	00	01	11	10
0	1/1	1/1	0/1	0/1
1	0/1	1/1	1/0	0/1



Hình 12.12. Bảng chuyển đổi trạng thái và bảng ra; đồ hình trạng thái của thí dụ 12.3

Cuối cùng ta có thể điền vào bảng chuyển đổi trạng thái và bảng ra cũng như đồ hình trạng thái của thiết bị như trên hình 12.12.

Để thấy rõ ảnh hưởng của các trạng thái vào tới hoạt động của mạch, ta phân tích đồ hình trạng thái theo các tổ hợp giá trị biến của x_1x_2 : (00, 01, 10, 11) $\Rightarrow$ có bốn đồ hình trạng thái thành phần như ở hình 12.13.



Hình 12.13. Bốn đồ hình trạng thái thành phần

Kết luận:

Tập $x_1x_2 = 00$ không có trạng thái nào bên đối với nó. Do đó với tập vào 00 sẽ có dao động giữa hai trạng thái trong 0 và 1. Trạng thái ra luôn luôn bằng 1. Để làm việc đúng, tập vào 00 phải là tập xung, tức ít nhất một trong x_1x_2 là xung.

Tập vào 11 là tập thụ động, không gây chuyển biến trạng thái đối với bất kỳ trạng thái vào nào.

Hai tập còn lại cũng là các tập tích cực: Tập 01 làm việc như tập đặt 1 cho trạng thái trong; tập 10 làm việc như tập xoá đối với trạng thái trong.

Tóm lại, để mạch làm việc đúng, tập 00 nhất thiết phải là tập xung.

Trong thực tế có thể dùng các đầu vào PR hay RESET để thiết lập trạng thái ban đầu.

12.2.2. Phân tích động

Đối với phân tích động có thể coi:

- Các tín hiệu vào là loại tín hiệu logic lý tưởng.
- Độ giữ chậm của các phần tử nhớ đối với các tín hiệu vào là như nhau và bằng τ .

Nhiệm vụ của bài toán phân tích động là xác định ảnh hưởng của hiện tượng giữ chậm tín hiệu kích thích của các phần tử nhớ đến độ tin cậy chức năng của mạch. Tức

cần xác định xem có hay không có sự tranh chấp trạng thái nguy hiểm đối với một mạch đã cho.

Để có thể phân tích động, bao giờ cũng phải tiến hành phân tích tĩnh và trên cơ sở đó tiếp tục phân tích ảnh hưởng của hiện tượng giữ chậm tín hiệu. Phân tích động gồm hai bước chính sau đây:

- Từ đồ hình trạng thái thành phần đối với các tập giá trị biến vào khác nhau, xác định xem có tồn tại chuyển biến động nguy hiểm không?
- Phân tích phần mạch tổ hợp xem có tồn tại hiện tượng tranh chấp trạng thái nguy hiểm của mạch tổ hợp hay không.

Để minh họa cho bài toán phân tích động này, chúng ta hãy xét thí dụ sau đây.

Thí dụ 12.4. Cho thiết bị số và kết quả phân tích tĩnh của nó như trên hình 12.14a, b, c, d, e, f, trong đó:

- Mạch và các hàm rút ra từ mạch tổ hợp của thiết bị đã cho.
- Hàm chuyển đổi trạng thái.
- Bảng ra.
- Đồ hình trạng thái.
- Đồ hình con khi $x = 0$.
- Đồ hình con khi $x = 1$.

Căn cứ vào kết quả phân tích tĩnh, ta thấy có hai trường hợp :

a) Khi mạch đang ở trạng thái 10, nếu $x = 0$, thì thiết kế mạch sẽ chuyển sang trạng thái 01 và cho ra $y = 0$ và trạng thái 01 là bền đối với tín hiệu vào $x = 0$. Điều này chỉ xảy ra khi hai phần tử nhớ a_1 và a_2 thay đổi trạng thái đồng thời. Nếu a_1 thay đổi trước thì ô tômát sẽ chuyển sang trạng thái 00. Ngược lại, nếu a_2 thay đổi trước thì ô tômát sẽ chuyển sang trạng thái 11. Song cả hai trạng thái 00 và 11 đều bền vững đối với tập vào $x = 0$, nên ô tômát không thể nào về trạng thái mong muốn của nó là 01 được.

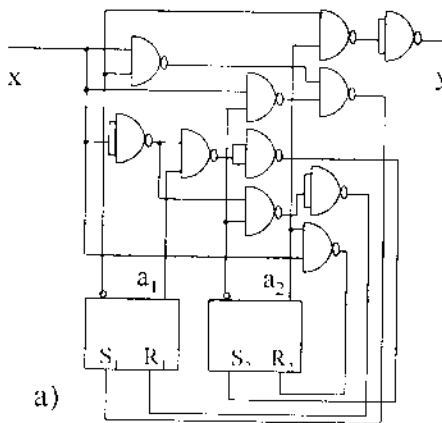
Vậy trong trường hợp này trong mạch xảy ra tranh chấp trạng thái nguy hiểm.

b) Nếu mạch đang ở trạng thái 01 mà tác động vào $x = 1$ thì đầu ra sẽ là $y = 1$. Theo thiết kế tĩnh, cả hai trigơ đều phải thay đổi trạng thái một cách đồng thời thì trạng thái mới là 10 và trạng thái này bền đối với trạng thái vào $x = 1$.

Nếu một trong hai phần tử nhớ chuyển trạng thái trước thì dù a_1 hay a_2 chuyển trước thì trạng thái mới của ô tômát vẫn lướt qua trạng thái 11 và cuối cùng vẫn tới trạng thái 10. Trong trường hợp này có tranh chấp trạng thái nhưng không nguy hiểm.

Từ sự phân tích trên đây ta rút ra nhận xét:

Do hiện tượng chuyển trạng thái không đồng thời của các phần tử nhớ, mạch có thể thực hiện các chuyển biến không có trong đồ hình trạng thái của nó.



Bảng phân tích tĩnh
mạch tổ hợp, rút ra các
hàm:

$$S_1 = (x | \bar{a}_1) | (x | \bar{a}_2)$$

$$R_1 = \bar{x} | \bar{a}_2$$

$$S_2 = (\bar{x} | a_1)$$

$$R_2 = (x | \bar{a}_2)$$

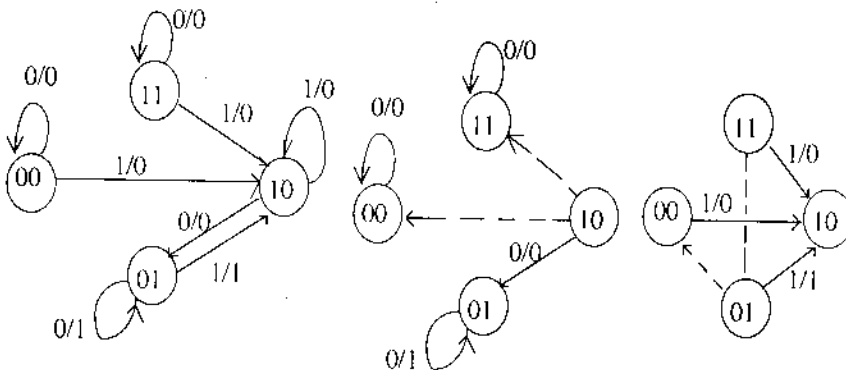
$$y = \bar{a}_1 | a_2$$

A	X	
$A_1 a_2$	0	1
0 0	00	10
0 1	01	10
1 0	01	10
1 1	11	10

b) Bảng chuyển đổi trạng thái

A	X	
$a_1 a_2$	0	1
0 0	0	0
0 1	1	1
1 0	0	0
1 1	0	0

c) Bảng ra



d) Đồ hình trạng thái

e) Khi $x = 0$

f) Khi $x = 1$

Hình 12.14. Mạch và kết quả phân tích của thí dụ 12.4

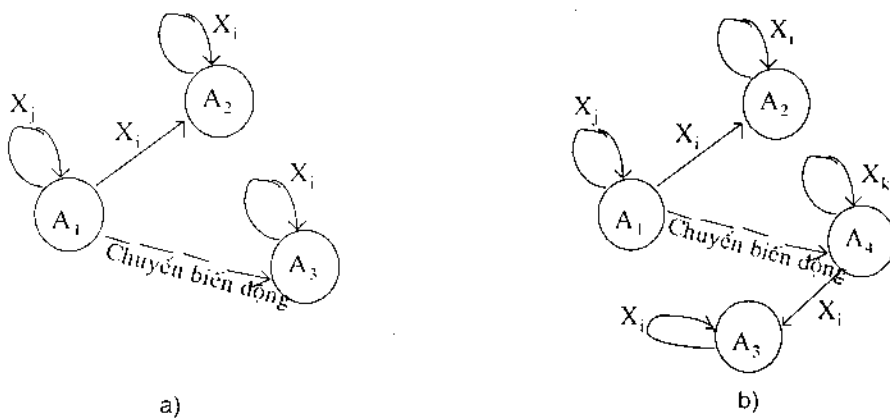
Các chuyển biến có trong đồ hình trạng thái của mạch gọi là các chuyển biến tĩnh. Các chuyển biến xảy ra trong quá trình làm việc của mạch gọi là các chuyển biến động. Như vậy các chuyển biến động có thể trùng với hoặc không trùng với các chuyển biến tĩnh.

Cần chú ý rằng, nếu các tổ hợp mã của hai trạng thái có chuyển biến tĩnh chỉ khác nhau ở một بیت thì chuyển biến động giữa chúng là duy nhất và trùng với chuyển

biến tĩnh, đó chính là trường hợp mã hoá kế cận. Mã kế cận làm cho các chuyển biến động luôn trùng với các chuyển biến tĩnh, do đó chống được tranh chấp trạng thái nguy hiểm.

Đối với mạch đã cho, do trong trường hợp a) có tranh chấp trạng thái nguy hiểm nên có nguy cơ làm việc không đúng. Do đó bắt buộc phải xem xét lại việc thiết kế và việc mã hoá trạng thái trong của mạch.

Điều kiện xuất hiện các tranh chấp trạng thái nguy hiểm dễ dàng nhận thấy trên đồ hình trạng thái của ô tô máy, khi đồ hình có chứa các yếu tố chỉ ra trên hình 12.15a, b. Trong đó điều kiện hình 12.15a là cơ bản. Trong đồ hình 12.15b có các chuyển biến tĩnh giữa A_1 và A_2 ; A_1 và A_3 và việc mã hoá đã dễ xảy ra chuyển biến động giữa A_1 và A_3 ; A_1 và A_4 .



Hình 12.15. Các trường hợp có tranh chấp trạng thái nguy hiểm

Kết quả của việc phân tích các chuyển biến trạng thái của các phần tử nhớ cho ta biết phần nhớ của ô tô máy đã cho được thiết kế và mã hoá có đúng hay không. Nếu phân tích cho thấy không có tranh chấp trạng thái nguy hiểm thì mạch được thiết kế và mã hoá trạng thái trong là đúng; nếu ngược lại phải xem lại việc mã hoá.

Mặt khác cần thấy rằng, khi phần nhớ thiết kế đúng thì mạch vẫn còn có thể sai do vẫn có thể có tranh chấp trạng thái tại đầu ra của mạch tổ hợp. Sự tranh chấp này có thể gây nên các tín hiệu giả hay làm méo các tín hiệu, làm cho tín hiệu kích thích của các phần tử nhớ thay đổi không như dự tính và đó chính là nguyên nhân của tranh chấp trạng thái nguy hiểm của mạch có nhớ. Vậy để mạch làm việc tốt cần khắc phục hiện tượng tranh chấp của cả phần tổ hợp lẫn phần nhớ của mạch.

12.3. PHÂN TÍCH THIẾT BỊ SỐ DÙNG PHẦN TỬ NHỚ LÀM VIỆC VỚI TÍN HIỆU XUNG

Ta biết rằng các phần tử nhớ làm việc với tín hiệu vào là xung chỉ có thể hoạt động đúng khi trạng thái vào tích cực có dạng xung, tức thời gian tồn tại của nó chỉ đủ để phần tử nhớ chuyển xong trạng thái và ổn định trạng thái mới. Nếu thời gian tồn tại của trạng thái vào này kéo dài, tức có dạng thế thì trạng thái mới cuối cùng của nó

phụ thuộc vào thời gian tồn tại của trạng thái vào này và tất nhiên phụ thuộc vào từng mạch cụ thể.

Từ đây ta thấy rằng, yêu cầu cơ bản để phần tử nhớ với tín hiệu vào xung làm việc đúng là ở trạng thái ban đầu, bình thường trạng thái của tín hiệu vào phải là trạng thái thụ động, không gây nên chuyển biến trạng thái và thường là tập giá trị biến 00...0 hay 11...1. Các tập giá trị biến khác có thể là các tập tích cực và là các trạng thái vào xung. Vậy các loại ô tômát có các phần tử nhớ làm việc với tín hiệu vào dạng xung làm việc tương tự như ô tômát đồng bộ, mặc dù không có xung đồng bộ ngoài.

Quá trình phân tích thiết bị số được xây dựng trên cơ sở các phần tử nhớ làm việc với tín hiệu xung gồm hai giai đoạn:

a) Thực hiện phân tích tĩnh như trong trường hợp làm việc với phần tử nhớ được nhíp, song điều quan trọng là sau đó phải chỉ ra trạng thái vào nào là thụ động, những trạng thái vào nào là tích cực và là xung.

b) Việc phân tích động trong trường hợp này có ý nghĩa cực kỳ quan trọng.

Để minh họa cho phương pháp phân tích này, ta xét thí dụ sau.

Thí dụ 12.5: Hãy phân tích thiết bị số có nhớ dùng phần tử nhớ loại xung - trigơ T cho trên hình 12.16.

12.3.1. Phân tích tĩnh

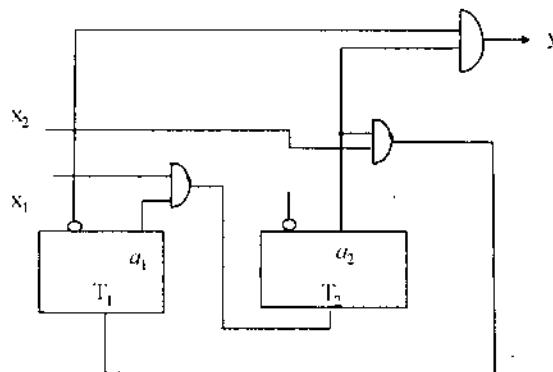
Bằng phân tích tĩnh mạch tổ hợp ta được các hàm sau:

$$y = \bar{a}_1 a_2$$

$$T_1 = x_2 a_2$$

$$T_2 = x_1 a_1$$

Trên hình 12.17 là các hàm y , T_1 , T_2 đã được đưa lên bảng kế cận và bảng chuyển đổi trạng thái của trigơ T.



Hình 12.16. Mạch dùng cho thí dụ 12.5

The diagram illustrates the transformation of a 2D array A into a 1D array T . The array A is a 4x4 grid with columns x_1 and x_2 . The array T is a 4x2 grid with columns x_1 and x_2 . The transformation is shown by mapping the elements of A to the elements of T . The mapping is: $A[0][0] \rightarrow T[0][0]$, $A[0][1] \rightarrow T[0][1]$, $A[1][0] \rightarrow T[1][0]$, $A[1][1] \rightarrow T[1][1]$, $A[2][0] \rightarrow T[2][0]$, $A[2][1] \rightarrow T[2][1]$, $A[3][0] \rightarrow T[3][0]$, and $A[3][1] \rightarrow T[3][1]$.

Array A (4x4):

	x_1	x_2	
a_1	0	0	0
a_2	0	0	0
	1	1	1
	0	0	0

Array T (4x2):

	x_1	x_2
a_1	0	0
a_2	0	1
	0	1
	0	0

Array T_2 (4x2):

	x_1	x_2
a_1	1	1
a_2	1	1
	0	0
	0	0

Array T (4x2):

	T	
A	0	1
0	0	1
1	1	0

 Hình 12.17. Các hàm y , T_1 , T_2 và bảng chuyển đổi trạng thái của mạch ở hình 12.16

Sử dụng các kết quả này để điền vào bảng chuyển đổi trạng thái và bảng ra, ta được hình 12.18a và đồ hình tương ứng (hình 12.18b).

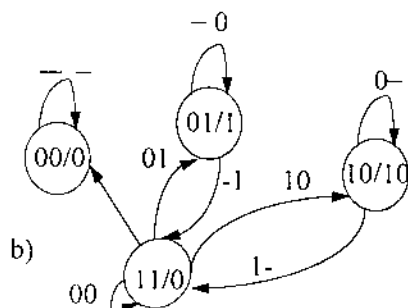
Từ đó có thể có dạng hình thức như ở hình 12.19.

Bây giờ ta hãy xét xem, từ yêu cầu đối với tín hiệu kích thích của phần tử nhớ làm việc với tín hiệu xung - trigơ T, sẽ dẫn tới những yêu cầu và hạn chế gì đối với các tập giá trị biến vào (các trạng thái vào).

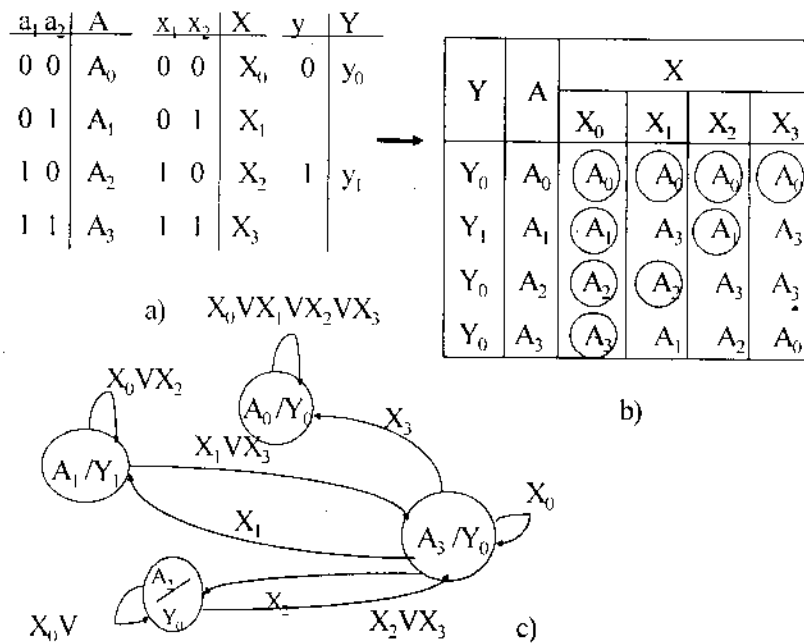
$a_1 a_2$	$x_1 x_2$			
	00	01	10	11
00	00	00	00	00
01	01	11	01	11
10	10	10	11	11
11	11	01	10	00

$a_1 a_2$	y
00	0
01	1
10	0
11	0

a)



Hình 12.18. Bảng chuyển đổi trạng thái và bảng ra (a); đồ hình trạng thái (b)



Hình 12.19. Bảng chuyển đổi trạng thái và bảng ra, đồ hình trạng thái dạng hình thức

- Để T_1 là tín hiệu xung thì x_2 phải là xung vì $T_1 = a_2 x_2$, mà a_2 là trạng thái của trigger T_2 nên a_2 nói chung có dạng thế.
- Tương tự $T_2 = a_1 x_1 \Rightarrow x_1$ phải xung.
- Tập $x_1 = 0, x_2 = 0$ rõ ràng là tập thụ động, mọi trạng thái đều bền với nó.
- Qua đồ hình hình thức ta thấy, nếu X_1 hoặc X_2 là thế sẽ gây nên dao động, vậy chúng phải là xung. Tập X_0 là thụ động, còn $X_2 = x_1 x_2$ và $X_1 = x_1 \bar{x}_2$ đều phải là xung.
- Tương tự X_3 nếu là thế sẽ có trường hợp đưa hệ về trạng thái liệt là trạng thái A_0 . Ở đó không một trạng thái vào nào có thể làm cho ô tô măt hoạt động được nữa, do đó X_3 cũng phải là trạng thái xung.

Mặc dù để một trạng thái vào, phụ thuộc vào hai biến x_1, x_2 là xung chỉ cần ít nhất một trong hai biến đó là xung. Song vì ba trạng thái X_1, X_2, X_3 trong bốn trạng thái của hai biến x_1 và x_2 đều phải là xung, nên cả x_1 và x_2 đều phải có dạng xung.

12.3.2. Phân tích động

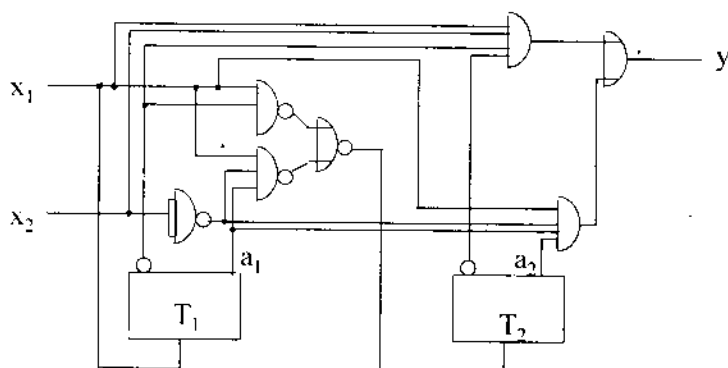
Từ đồ hình trạng thái (hình 12.18b) ta thấy rằng, khi ô tô măt chuyển từ trạng thái 11 sang trạng thái 00 thì yêu cầu cả hai biến trong đều phải thay đổi đồng thời. Nếu một trong hai biến thay đổi trước thì sẽ dẫn tới tranh chấp trạng thái nguy hiểm: ô tô măt không bao giờ đến được trạng thái 00 nữa.

Tuy vậy, cần lưu ý rằng không phải cứ phần tử nhớ làm việc với tín hiệu xung thì các tín hiệu vào x_i đều phải là xung hoặc cứ có x_i là xung thì dùng các phần tử nhớ loại xung là được. Thí dụ sau đây sẽ chứng minh điều đó.

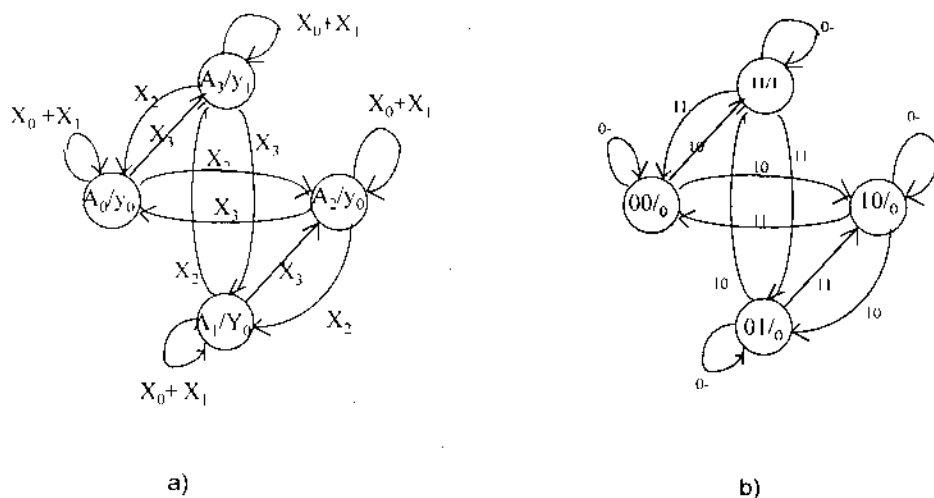
Thí dụ 12.6: Phân tích ô-tô-mát cho trên hình 12.20.

Kết quả của phân tích tĩnh ta sẽ được đồ hình trạng thái như ở hình 12.21.

Từ đồ hình ta thấy rằng, để ô-tô-mát làm việc như một ô-tô-mát đồng bộ, tức dùng hai trigơ T, hai tập giá trị biến vào X_2 và X_3 phải là xung. Song có nhất thiết cả hai biến vào x_1 và x_2 đều phải là xung hay không? Câu trả lời là không nhất thiết phải như vậy. Thật vậy, nếu ta mã hoá như đồ hình ở hình 12.21b thì rõ ràng chỉ cần biến x_1 là xung. Biến x_2 là xung hay thế cũng được.



Hình 12.20



Hình 12.21

Phân tích động

Tranh chấp trạng thái do mạch tổ hợp gây nên.

Ta đã biết để không xảy ra tranh chấp động trong phần mạch tổ hợp, tại một thời điểm, trong số các biến dạng xung yêu cầu chỉ một biến có giá trị 1. Vậy nếu có n biến thì ta chỉ có thể dùng $(n + 1)$ tập giá trị biến mà thôi, trong đó một tập giá trị biến phải là tập thụ động. Đối với mạch ở hình 12.20, nếu thoả mãn điều kiện vừa nêu ra, phân tích tĩnh sẽ cho kết quả như ở hình 12.22.

	x_1	x_2	y
a_1	0	-	0
	0	-	0
a_2	1	-	1
	0	-	0

	x_1	x_2	T_1
a_1	0	-	0
	0	-	0
a_2	0	-	1
	0	-	0

	x_1	x_2	T_2
a_1	0	-	1
	0	-	1
a_2	0	-	0
	0	-	0

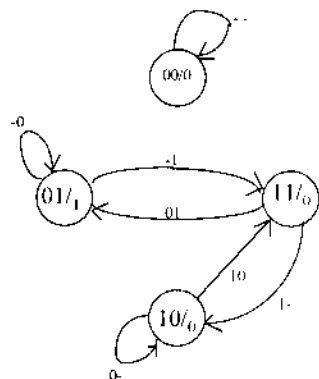
Hình 12.22. Biểu diễn các hàm y , T_1 , T_2 lên bảng kế cận

Bảng chuyển đổi trạng thái và bảng ra, đồ hình trạng thái cho trên hình 12.23.

Vì tổ hợp $x_1x_2 = 00$ là tổ hợp vào thụ động nên đồ hình 12.23 có thể rút gọn thành hình 12.24.

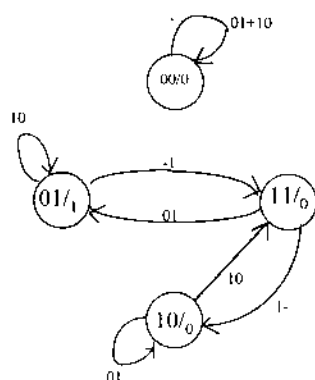
Một nguyên nhân khác làm cho mạch làm việc sai là hiện tượng làm hẹp xung ở đầu ra của một số phần tử logic do đường truyền của tín hiệu tới đầu vào của nó là khác nhau. Khi tín hiệu ra bị hẹp lại tới mức mà các phần tử nhớ không thể ghi nhận được thì hoạt động của toàn mạch sẽ bị sai.

Y	A	x_1x_2			
		00	01	10	11
0	00	00	00	00	-
1	01	01	11	01	-
0	10	10	10	11	-
0	11	11	01	10	-



Hình 12.23. Bảng chuyển đổi trạng thái và bảng ra

Y	A	X_1X_2	
		01	10
0	00	00	00
1	01	11	01
0	10	10	11
0	11	01	10



Hình 12.24. Dạng rút gọn của bảng chuyển đổi trạng thái và bảng ra, đồ hình trạng thái ở hình 12.23

Thực tế yêu cầu đủ để không có tranh chấp trạng thái trong phần mạch tổ hợp là:

- + Các tập giá trị biến vào tác động liên tiếp phải là các tập giá trị biến kế cận.
- + Phần mạch tổ hợp phải được xây dựng bằng dạng tuyến chuẩn tối thiểu đầy đủ.

Điều kiện thứ nhất có nghĩa là:

- + Sau mỗi tập xung vào phải kèm theo tập giá trị biến thụ động.

+ Các biến vào chỉ được thay đổi giá trị của mình trong giai đoạn tác động của tập biến vào thụ động.

+ Các phần tử nhớ không được đồng thời thay đổi trạng thái. Đây là điều có liên quan tới sự mã hoá các trạng thái trong. Mạch ở hình 12.20 không thoả mãn điều kiện này vì theo đồ hình trạng thái của nó, ta thấy có các bước chuyển trạng thái trong đó cả hai biến trong đều thay đổi. Do đó có thể xảy ra tranh chấp trạng thái.

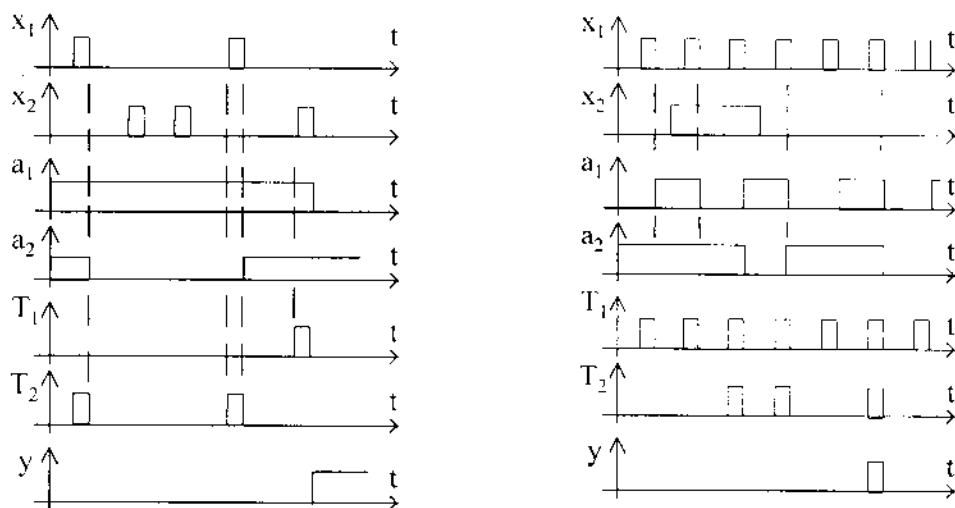
Đối với các xung được tạo nên do đường truyền của các tín hiệu khác nhau, nếu các phần tử nhớ không ghi nhận được thì cũng vẫn không gây nên tranh chấp nguy hiểm và mạch vẫn làm việc đúng.

Phân tích về sự tồn tại các chuyển biến động nguy hiểm của các mạch dây dùng các phần tử nhớ loại xung dễ dàng hơn đối với trường hợp khi dùng loại thế, vì đối với chúng chỉ có các chuyển biến đơn nên mọi chuyển biến gây thay đổi trạng thái của từ hai trigơ trở lên sẽ dẫn tới nguy cơ tranh chấp trạng thái nguy hiểm.

Như vậy, ô tômat sẽ không có chuyển biến động nguy hiểm khi hai trạng thái bất kỳ có bước chuyển được mã hoá bằng hai bộ mã kế cận. Điều này cho thấy việc phân tích ô tômat dùng phần tử nhớ đầu vào xung cũng tương tự như khi phân tích ô tômat dùng phần tử nhớ được nhịp. Điều khác là ở chỗ cần xác định xem đầu vào nào phải là xung và phân tích hoạt động của mạch về mặt tranh chấp nguy hiểm.

Trên hình 12.25 là giản đồ thời gian của các tín hiệu thuộc hình 12.16 và hình 12.20 với giả thiết là các trigger chuyển trạng thái theo sườn sau của xung kích thích.

Từ những phân tích trên đây ta có thể nói rằng, các thiết bị số dùng các phần tử nhớ loại xung đòi hỏi các trạng thái tác động vào X_i phải là xung, mà để đảm bảo điều này ít nhất một biến vào phải là xung. Mô hình ô tômát tương ứng là loại ô tômát đồng bộ. Việc phân tích các ô tômát loại này cũng tương tự như phân tích ô tômát dùng phần tử nhớ loại nhịp.



Hình 12.25. Giản đồ thời gian của hình 12.16 và 12.20

Sự khác nhau giữa hai loại ô tômát này là ở chỗ, đối với loại xung do tác động lên một số phần tử nhớ cùng một lúc nên ở một vài chuyển biến có thể xuất hiện tranh chấp trạng thái. Do đó khi tổng hợp sẽ phải đáp ứng một số đòi hỏi hạn chế.

12.4. NHỮNG HƯ HỎNG TRONG THIẾT BỊ SỐ

Tương tự như trong các mạch tổ hợp, đối với mạch dãy ta cũng xem xét vấn đề phát hiện các hư hỏng và phương pháp khử bỏ chúng. Đặc điểm của các hư hỏng trong các mạch dãy là ngoài những hư hỏng thể hiện ngay trên tín hiệu đầu ra, còn có các hư hỏng chỉ thể hiện ở đầu ra khi phần tử nhớ của nó hoạt động không đúng. Ngoài ra cần thấy rằng, những hư hỏng như thế thường lại không xảy ra theo quy luật nhất định mà tùy thuộc vào trạng thái trong ban đầu và các trạng thái vào tiếp theo xảy ra.

Do những đặc điểm trên đây, sự phát hiện và khoanh vùng hư hỏng trong thiết bị số có nhớ khó hơn nhiều so với trường hợp của mạch tổ hợp. Trước khi xem xét cách phát hiện các hư hỏng trong mạch dãy cần hiểu rõ khái niệm chuỗi dẫn và chuỗi vào thiết lập.

12.4.1. Chuỗi dẫn và chuỗi vào thiết lập

a) Chuỗi dẫn

Là chuỗi vào ngắn nhất để đưa ô tô măt từ một trạng thái này sang một trạng thái khác.

Thí dụ có ô tô măt cho trên hình 12.26. Giả sử ban đầu ô tô măt ở trạng thái A_2 , hãy xác định chuỗi trạng thái vào để ô tô măt chuyển sang trạng thái A_1 .

Để trả lời cho vấn đề này, từ đồ hình trạng thái của ô tô măt đã cho ta có sơ đồ tìm chuỗi dẫn của ô tô măt từ A_2 về A_1 như hình 12.27.

Từ hình 12.27 ta có 4 chuỗi vào ngắn nhất đưa ô tô măt từ A_2 về A_1 là:

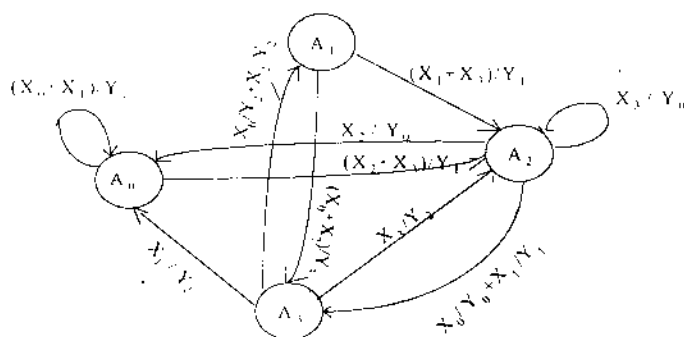
$$X_0 \dots X_0$$

$$X_1 \dots X_0$$

$$X_0 \dots X_2$$

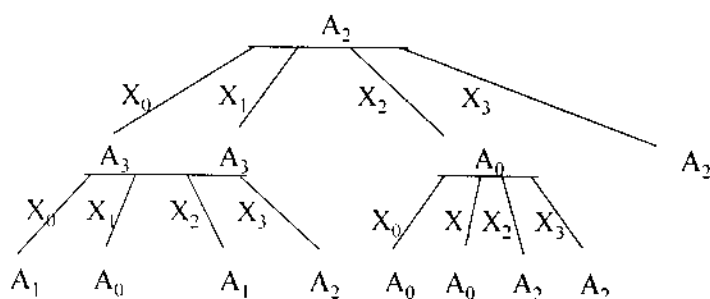
$$X_1 \dots X_2$$

Đó chính là các chuỗi dẫn.



Hình 12.26. Ô tô măt dùng nghiên cứu về chuỗi dẫn và chuỗi vào

Trong thực tế có thể có trường hợp không có chuỗi dẫn. Trong trường hợp này, sau một số bước nào đó không thể có trạng thái vào nào có thể đưa ô tô măt về trạng thái mong muốn.



Hình 12.27. Xác định 4 chuỗi vào ngắn nhất từ A_2 về A_1

b) Chuỗi thiết lập

Một trường hợp khác hay gặp là trong khi phân tích hoạt động hiện tại của ô tô máy lại không rõ nó đang ở trạng thái trong nào. Trong trường hợp này, để ô tô máy về một trạng thái định trước ta cần tác động vào ô tô máy chuỗi các trạng thái vào thích hợp, gọi là chuỗi thiết lập.

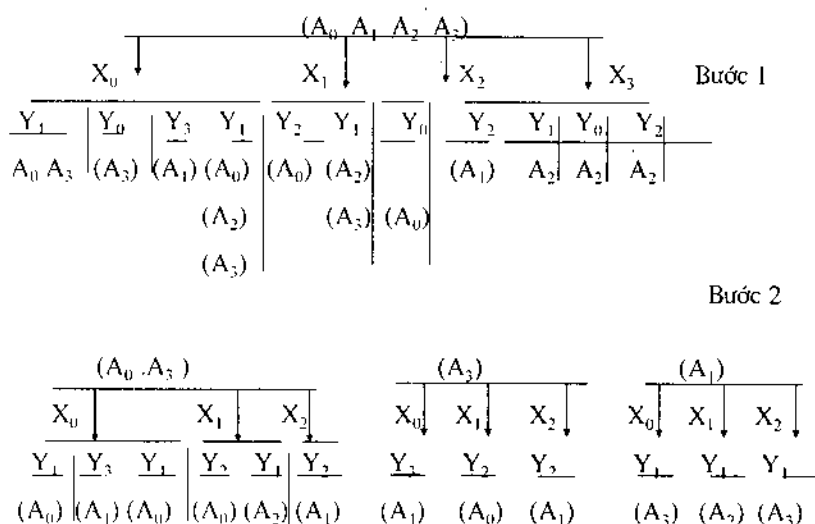
Chuỗi thiết lập là chuỗi các trạng thái vào của ô tô máy mà khi tác động, ô tô máy đang ở trạng thái nào đó sẽ về một trạng thái đã định trước.

Phương pháp xác định chuỗi thiết lập, thí dụ đối với ô tô máy ở hình 12.26 như sau: Giả sử không rõ trạng thái đang tồn tại của ô tô máy là trạng thái nào, ta hãy tác động vào nó trạng thái vào X_0 . Theo đồ hình của ô tô máy, kết quả trạng thái ra có thể là một trong ba giá trị: Y_0, Y_1, Y_3 . Nếu là Y_1 thì rõ ràng ô tô máy trước đó đang nằm ở trạng thái A_0 hoặc A_1 . Nếu là Y_0 thì là A_3 . Nếu là Y_3 thì ô tô máy trước đó đang ở A_1 .

Ta cũng có thể tác động vào ô tô máy các trạng thái vào X_1, X_2 hoặc X_3 và căn cứ vào các trạng thái ra thu được để xác định trước đó ô tô máy đang nằm ở trạng thái nào. Kết quả phân tích trong trường hợp này cho trên hình 12.28.

Từ hình 12.28 ta thấy rằng, tập vào X_3 bao giờ cũng đưa ô tô máy về trạng thái A_2 dù nó xuất phát từ bất kỳ trạng thái nào trong 4 trạng thái A_0, A_1, A_2 và A_3 . Đối với các chuỗi vào khác, muốn đi đến kết quả cần phải tiếp tục phân tích.

Thí dụ, tập vào X_0 phân biệt được trạng thái mới thiết lập là A_1 và A_3 căn cứ vào phản ứng ra là Y_3 hay Y_0 . Nhưng nếu phản ứng ra là Y_1 thì lại cho phép kết luận là mạch trước đó đã ở trạng thái A_0 hoặc A_3 ...



Hình 12.28. Tập vào X_3 đưa ô tô máy về trạng thái A_2 , xuất phát từ bất kỳ trạng thái nào trong các trạng thái A_0, A_1, A_2 và A_3

Với ô tô máy ở hình 12.26, chuỗi thiết lập ngắn nhất là X_3 . Sau khi phân tích bước 1, nếu tiếp tục phân tích bước 2 ta có thể xác định một cách chắc chắn ô tô máy đang ở trạng thái nào. Thí dụ, sau khi tác động X_0 ta tác động tiếp X_2 , nếu phản ứng ra sau hai bước là :

- + Y_1 và Y_1 thì trạng thái được thiết lập là trạng thái A_2 .
- + Y_1 và Y_2 thì trạng thái được thiết lập là trạng thái A_1 .
- + Y_0 và Y_2 thì trạng thái được thiết lập là trạng thái A_1 .
- + Y_3 và Y_1 thì trạng thái được thiết lập là A_3 .

Bằng cách phân tích như trên, ta hoàn toàn có thể xác định được các chuỗi thiết lập, đưa ô tô máy về trạng thái mong muốn. Từ đó thủ tục để thiết lập ô tô máy đã cho vào một trạng thái mong muốn gồm hai bước sau đây:

a) Chọn chuỗi thiết lập và cho tác động vào ô tô máy. Kiểm tra trạng thái được thiết lập bằng cách quan sát các trạng thái ra Y .

b) Nếu ô tô máy không được thiết lập vào trạng thái mong muốn thì dùng chuỗi dẫn để đưa ô tô máy về trạng thái mong muốn.

Vấn đề quan trọng là làm sao tìm được chuỗi thiết lập và chuỗi dẫn ngắn nhất. Ở thí dụ trên đây, chuỗi thiết lập ngắn nhất là X_3 . Chuỗi dẫn ngắn nhất là X_2 .

Trong quá trình tìm chuỗi thiết lập và dẫn, người ta cũng phải quan tâm giải quyết câu hỏi đặt ra là, đối với một ô tô máy cho trước:

- + Phải chăng luôn luôn tồn tại chuỗi thiết lập và chuỗi dẫn?
- + Nếu tồn tại thì phải có điều kiện gì hay không?
- + Ảnh hưởng của các hư hỏng lên chuỗi thiết lập và chuỗi dẫn ra sao?
- + Liệu có thể tìm được chuỗi thiết lập và chuỗi dẫn không phụ thuộc vào các hư hỏng trong mạch của ô tô máy hay không?

Giải đáp những vấn đề đặt ra còn nhiều phức tạp và còn phải tiếp tục nghiên cứu.

12.4.2. Vấn đề phát hiện các hư hỏng trong ô tô máy

Như chúng ta đã biết, việc xác định xem ô tô máy có hỏng hay không là một bài toán phức tạp và khó. Không những thế, khi một mạch đã bị hỏng lại còn cần phải biết hỏng ở chỗ nào trong mạch. Có như vậy mới có thể tiến hành sửa chữa, thay thế được.

Trong phần này sẽ xem xét cách phát hiện một số hư hỏng trong ô tô máy với giả thiết rằng đối với ô tô máy đã cho ta đã xác định được chuỗi thiết lập và chuỗi dẫn hoặc trong trường hợp thuận lợi là có các đầu vào đặc biệt để thiết lập trạng thái ban đầu của các phần tử nhỏ.

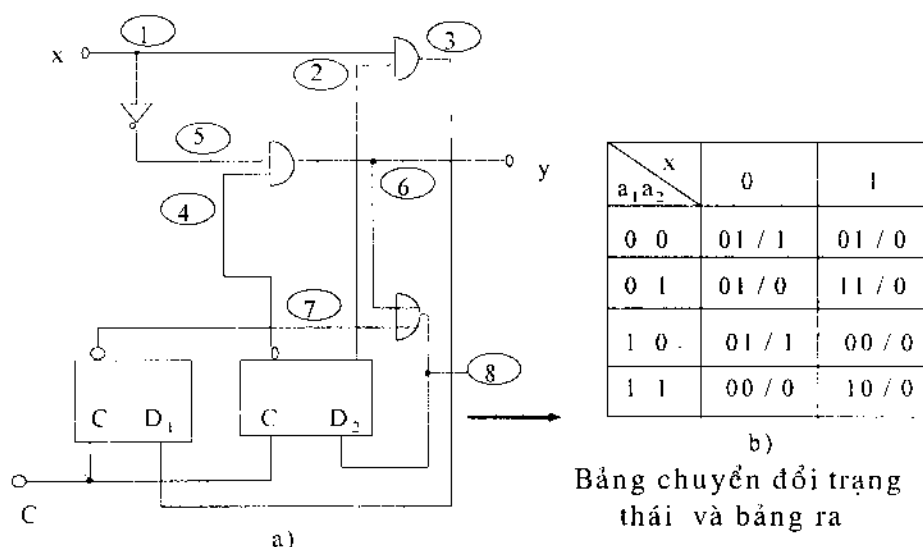
Để thấy rõ cách xác định và khoanh vùng hư hỏng trong ô tô máy như thế nào, ta xét thí dụ hình ở 12.29a trong thí dụ 12.6.

Thí dụ 12.6

Cho ô tômát ở hình 12.29a. Giả thiết rằng có hư hỏng loại 1/(1), tức là tại điểm (1) luôn luôn có giá trị 1.

Bằng bài toán phân tích, dễ dàng thiết lập bảng chuyển đổi trạng thái và bảng ra kết hợp của ô tômát khi không có hư hỏng như ở hình 12.29b.

Đồng thời với giả thiết có hư hỏng loại 1/(1) ta cũng thiết lập được bảng chuyển đổi trạng thái và bảng ra như ở hình 12.30.



Hình 12.29. Mạch và bảng chuyển đổi trạng thái và bảng ra khi không có hư hỏng

So sánh kết quả trong hình 12.26b và 12.27 ta có nhận xét sau:

· Phản ứng ra khác nhau trong 2 trường hợp hỏng và không hỏng ở:

$$x = 0, a_1 a_2 = 00$$

$$x = 0, a_1 a_2 = 10$$

Do đó, nếu trạng thái trong của ô tômát là 00 hoặc 10 và trạng thái vào là $x = 0$, phản ứng ra sẽ là 1 khi không có hư hỏng, là 0 khi có hư hỏng loại 1/(1).

Bây giờ giả thiết có hư hỏng loại 1/(8). Trong trường hợp này ta có bảng chuyển đổi trạng thái và bảng ra kết hợp như trên hình 12.31a. So sánh hai bảng khi có và không có hư hỏng ta thấy:

x	0	1
$a_1 a_2$		
0 0	01/(0)	01/0
0 1	01/0	11/0
1 0	00/(0)	00/0
1 1	00/0	10/0

Hình 12.30. Bảng chuyển đổi trạng thái và bảng ra kết hợp khi có hư hỏng loại 1/(1)

- Bảng ra không thay đổi.

- Bảng chuyển đổi trạng thái thay đổi ở ba trường hợp, được đánh dấu bằng khoanh tròn.

Kết quả so sánh cho thấy hư hỏng 1/(8) không thể hiện ở trạng thái ra mà chỉ làm thay đổi các bước chuyển trạng thái.

Thí dụ, nếu trạng thái ban đầu là 11, trạng thái đầu vào $x = 0$ thì:

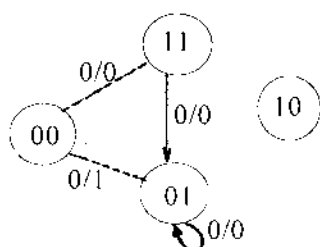
		x	
		0	1
a ₁	a ₂		
	0	01 / 1	01 / 0
	1	01 / 0	11 / 0
	1	01 / 1	01 / 0
1	0	01 / 1	01 / 0
	1	01 / 0	11 / 0

Hình 12.31. Bảng chuyển đổi trạng thái và bảng ra khi có hư hỏng 1/(8)

- Trạng thái trong mới của ô tô máy không có hư hỏng sẽ là 00. Trạng thái ra tương ứng là $y = 0$.

- Trạng thái trong mới của ô tô máy có hư hỏng loại 1/(8) sẽ là 01. Trạng thái ra tương ứng là 0.

Song vì đối với ô tô máy đã cho, khi $x = 0$ từ hai trạng thái 00 và 01 sẽ xảy ra những chuyển biến tiếp theo, có trạng thái ra khác nhau: Bằng 0 khi không có hư hỏng và bằng 1 khi có hư hỏng - đồ hình 12.32, nên cho phép ta xác định được chuỗi các trạng thái vào, cho phép phân biệt mạch còn làm việc tốt với mạch có hư hỏng 1/(8).



Hình 12.32. Trạng thái ra bằng 0 khi không hư hỏng, bằng 1 khi có hư hỏng 1/(8)

Thí dụ, chuỗi vào $x = \{0, 0\}$, trạng thái ban đầu 11, chuỗi ra $y = \{0, 1\}$ thì mạch không có hư hỏng. Ngược lại, nếu chuỗi ra là $y = \{0, 0\}$ thì mạch có hư hỏng loại 1/(8).

Như vậy trường hợp này hư hỏng thể hiện ở nhịp thứ hai.

Nhiều trường hợp khác phải sử dụng các chuỗi vào khác nhau có độ dài khác nhau mới có thể phát hiện được có hư hỏng hay không.

Thí dụ, theo bảng chuyển đổi trạng thái và bảng ra của ô tô máy ở hình 12.29, có hỏng loại 1/(2), cho trên hình 12.33. Ta đã biết, nếu mạch không có hư hỏng, từ trạng thái trong 11, ta tác động hai trạng thái vào $x = \{0, 0\}$, ô tô máy sẽ chuyển lần lượt sang các trạng thái 00 và 01. Nhưng nếu mạch có hư hỏng loại 1/(2) thì dưới tác động của $x = \{0, 0\}$, ô tô máy sẽ lần lượt chuyển sang các trạng thái 10 và 01. Tuy đường đi của ô tô máy qua các trạng thái trong của hai trường hợp là khác nhau, song kết quả ra ở hai nhịp vẫn là 0, 1, do đó ta không thể phân biệt được. Chẳng những thế, trong trường

hợp này nếu ta tiếp tục tác động bằng các trạng thái vào $x = 0$ thêm một hoặc một số nhịp nữa thì vì trạng thái trong 01 bền đối với trạng thái vào $x = 0$, nên kết quả vẫn luôn bằng 0. Do đó không thể phân biệt được mạch có hư hỏng hay không có hư hỏng.

Nếu ta dùng chuỗi vào là $x = \{0, 1\}$, thì chỉ bằng hai nhịp, không thể phân biệt được mạch có hư hỏng 1/(2) với mạch không hỏng. Nói một cách tổng quát hơn, đối với ô tô-mát ở hình 12.29, dù dùng chuỗi vào nào cũng không thể phân biệt được mạch có hư hỏng loại 1/(2) với mạch không hư hỏng.

Từ phân tích ta có thể đi đến kết luận sau:

Nếu mạch chứa W phần tử thì trạng thái không đúng sẽ biểu lộ ra sau khi tác động không quá W nhịp. Ở nhịp thứ $W + 1$ sự không đúng sẽ được thể hiện ở kết quả ra, hoặc không được thể hiện ra. Trong trường hợp không được thể hiện ra là trường hợp không thể phát hiện được.

A	X	
	0	1
$a_1 a_2$		
0 0	01 / 1	01 / 0
0 1	11 / 0	11 / 0
1 0	01 / 1	00 / 0
1 1	10 / 0	00 / 0

Hình 12.33. Bảng có hỏng 1/(2) của mạch ở hình 12.29

12.4.3. Phương hướng tìm chuỗi kiểm tra hư hỏng

Dưới đây chúng ta sẽ tìm chuỗi vào cần thiết để kiểm tra hư hỏng của ô tô-mát với các điều kiện sau:

- Mạch chỉ có hư hỏng đơn.
- Tồn tại chuỗi thiết lập và chuỗi dẫn không phụ thuộc vào các hư hỏng trong mạch.
- Mạch thiết kế ở dạng tối thiểu, không thừa về linh kiện.

Thí dụ 12.7: Tìm chuỗi kiểm tra cho ô tô-mát ở hình 12.34.

Nếu phân tích tĩnh ta thấy:

- Có 9 điểm cần kiểm tra và có $2 \times 9 = 18$ hư hỏng đơn.

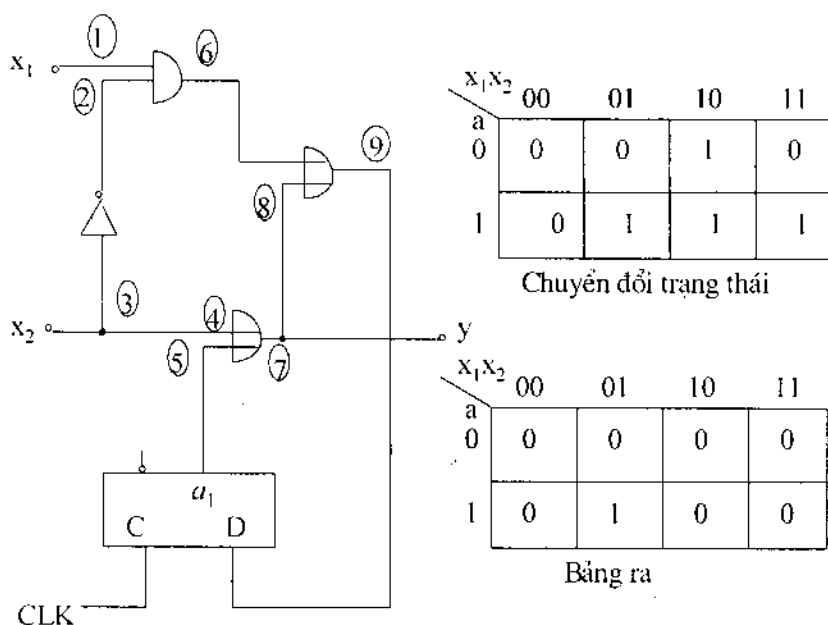
- Để tìm được chuỗi kiểm tra ta làm tương tự như tìm chuỗi kiểm tra trong phần tổ hợp, nghĩa là phải lập bảng chuyển đổi trạng thái và bảng ra khi:

- + Không có hư hỏng
- + Có 1 trong 18 hư hỏng.

Để tiện cho việc so sánh, bảng chuyển đổi trạng thái và bảng ra được viết dưới dạng các cột ứng với các bảng khi có một trong 18 loại hư hỏng trên đây. Bảng có dạng như đã chỉ ra trên hình 12.35.

Phân tích bảng nhận được ta thấy rằng:

+ Một số hư hỏng không thể phân biệt chúng với nhau bằng tín hiệu ra, thí dụ đối với các hư hỏng 0/(4), 0/(5) và 0/(7). Nhưng giữa chúng với các hư hỏng khác lại có thể phân biệt bằng các phản ứng đầu ra.



Hình 12.34. Ôtômát và bảng chuyển đổi trạng thái và bảng ra cho thí dụ 12.7

Thí dụ, với các hư hỏng loại 1/(4), 1/(5) và 1/(7).

+ Đối với các hư hỏng có thể phân biệt được, ta có thể lập một bảng riêng như bảng ở hình 12.36.

$x_1 x_2$	a	tốt	1/1	0/1	1/2	0/2	1/3	0/3	1/4	0/4	1/5	0/5	1/6	0/6	1/7	0/7	1/8	0/8	1/9	0/9
0 0 0	0/0	0/0	1/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	1/0	0/0	1/1	0/0	1/0	0/0	1/0	0/0
0 0 1	0/0	0/0	1/0	0/0	0/0	0/0	0/0	0/0	1/1	0/0	0/0	0/0	1/0	0/0	1/1	0/0	1/0	0/0	1/0	0/0
0 1 0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	1/1	0/0	1/0	0/0	1/1	0/0	1/0	0/0	1/0	0/0
0 1 1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	0/0	1/1	0/0	1/1	1/1	1/1	0/0	1/0	0/1	1/1	0/1	
1 0 0	1/0	1/0	0/0	1/0	0/0	0/0	1/0	1/0	1/0	1/0	1/0	1/0	0/0	1/1	1/0	1/0	1/0	1/0	0/0	0/0
1 0 1	1/0	1/0	0/0	1/0	0/0	0/0	1/0	1/1	1/0	1/0	1/0	1/0	0/0	1/1	1/0	1/0	1/0	1/0	0/0	0/0
1 1 0	0/0	0/0	0/0	1/0	0/0	0/0	1/0	0/0	0/0	1/1	0/0	1/0	0/0	1/1	0/0	1/0	0/0	1/0	0/0	0/0
1 1 1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	0/0	1/1	0/0	1/1	1/1	1/1	0/0	1/1	0/1	1/1	0/1	

Hình 12.35. Bảng viết thành cột khi có các hư hỏng

$x_1 x_2 a$	0/(4)	1/(4)	1/(5)	1/(7)	
0 0 0	0	0	0	1	
0 0 1	0	1	0	1	[0/4, 0/5, 0/7] $\rightarrow$ 0/4
0 1 0	0	0	1	1	[1/4]
0 1 1	0	1	1	1	[1/5]
1 0 0	0	0	0	1	[1/7]
1 0 1	0	1	0	1	Kiểm tra diễn theo phản ứng ra.
1 1 0	0	0	1	1	
1 1 1	0	1	1	1	

Hình 12.36. Bảng cho các hư hỏng phân biệt được

Chúng ta gọi bảng kiểm tra này là bảng kiểm tra thứ nhất. Bảng kiểm tra thứ nhất chứa đựng tất cả những hư hỏng phân biệt được. Nhờ bảng kiểm tra thứ nhất, mọi hư hỏng có trong bảng đều được thể hiện trực tiếp ở đầu ra chỉ qua một lần, tức chỉ một nhịp kiểm tra. Trong đó ứng với một tập giá trị của các biến vào và biến trong $x_1 x_2 a$ thì thể hiện ở phản ứng ra đối với trường hợp mạch còn tốt và khi mạch hỏng là hoàn toàn khác nhau đối với mỗi nhóm hư hỏng.

Thí dụ, với $x_1 x_2 a = 001$: nếu $Ra = 0 \rightarrow$ mạch tốt.

nếu $Ra = 1 \rightarrow$ hỏng loại 1/(4) hoặc 1/(7).

Một trong các phương án kiểm tra trong trường hợp này có thể cho trên bảng ở hình 12.37.

$x_1 \ x_2 \ a$	y tốt	y hỏng	Loại hư hỏng
0 0 1	0	1	1/4 loại 1/7
0 1 0	0	1	1/5 loại 1/7
0 1 1	0	0	0/4 hoặc Tương đương

Hình 12.37. Một trong các phương án kiểm tra loại hư hỏng

Như vậy, nếu dùng tới nhịp thứ hai sẽ phân biệt ra từng hư hỏng.

Tương tự, đối với các hư hỏng không thể thể hiện được trên các phản ứng ra của ô tô mét, sau khi đã gộp những hư hỏng không thể phân biệt được như 0/(1), 0/(2), 1/(3) và 0/(6); 0/(3) và 1/(2); 1/(6), 1/(8) và 1/(9); 0/(7) và 0/(8); 0/(9) và thay chúng bằng các hư hỏng đại diện như sau:

$$\{1/1\} \Rightarrow 1/1$$

$$\{0/1, 0/2, 1/3, 0/6\} \Rightarrow 0/1$$

$$\{1/6, 1/8, 1/9\} \Rightarrow 1/6$$

$$\{1/2, 0/3\}$$

$$\Rightarrow 1/2$$

$$\{0/9\}$$

$$\Rightarrow 0/9$$

$$\{0/7, 0/8\}$$

$$\Rightarrow 0/7 \Rightarrow \text{bảng}$$

Sau đó ta lập bảng kiểm tra như trên hình 12.38. Trong đó các trạng thái mới được khoanh bằng hình vuông là các trạng thái trong khác với trong trường hợp không có hư hỏng. Ta gọi bảng kiểm tra này là bảng kiểm tra thứ hai.

Như vậy:

- Bằng cách thực hiện kiểm tra hai nhịp ta có thể phân biệt được mạch tốt hay mạch hỏng. Thí dụ dùng hai tập giá trị biến vào: 001 và 010, ta có:

Nếu trạng thái ra là: $0 \rightarrow 0$ mạch tốt

$0 \rightarrow 1$: hỏng loại 1/1, 1/6 và tương đương.

Do đó một trong các phương án kiểm tra hai nhịp để phân biệt các nhóm hư hỏng trong mạch cho trên hình 12.39.

Nhận xét:

Kiểm tra một nhịp có thể được phủ bởi kiểm tra nhiều nhịp từng phần hay toàn bộ. Thật vậy, đối với ô tô máy ở hình 12.34, hàng thứ nhất của phương án kiểm tra một nhịp (hình 12.38) được phủ bởi hàng thứ tư của bảng kiểm tra hai nhịp - hình 12.39. Khi kết hợp chúng lại ta có chuẩn để kiểm tra như sau:

$x_1 x_2 a_1$	tốt	1/1	0/1	1/2	1/6	0/7	0/9
0 0 0	0	1	0	0	1	0	0
0 0 1	0	1	0	0	1	0	0
0 1 0	0	0	0	0	1	0	0
0 1 1	1	1	1	1	1	0	0
1 0 0	1	1	0	1	1	1	0
1 0 1	1	1	0	1	1	1	0
1 1 0	0	0	0	1	1	0	0
1 1 1	1	1	1	1	1	0	0

□ $\equiv$ Khác với khi tốt

Bảng kiểm tra thứ 2

Hình 12.38. Phương án kiểm tra một nhịp đối với ô tô máy ở hình 12.34

Trạng thái ban đầu là $a = 1$, trạng thái vào thứ nhất là $x_1 x_2 = 00$, trạng thái vào thứ hai là $x_1 x_2 = 01$ sẽ cho chuỗi ra:

0 - 0 : Mạch không có hư hỏng;

0 - 1 : Mạch có hư hỏng loại 1/(1), 1/(6) và tương đương;

1 - 1 : Mạch có hư hỏng loại 1/(4) hoặc 1/(7).

a_1	x_1x_2 (I)	x_1x_2 (II)	$y_1 - y_2$ (Tốt)	$y_1 - y_2$ (Hỏng)	Hư hỏng loại
0	11	01	0 - 0	0 - 1	1/(2) 1/(6)
1	11	01	1 - 1	1 - 0	0/(7) 0/(9)
1	10	01	0 - 1	0 - 0	0/(1) 0/(9)
1	00	01	0 - 0	0 - 1	1/(1) 1/(6)

Hình 12.39. Bảng kiểm tra hai nhip

Hàng thứ hai của bảng kiểm tra một nhip (hình 12.38) được chứa đựng trong hàng thứ nhất của bảng kiểm tra hai nhip (hình 12.39). Liên kết chúng lại ta có:

Trạng thái ban đầu là $a = 0$; trạng thái vào thứ nhất $x_1x_2 = 11$; trạng thái vào thứ hai $x_1x_2 = 01$, phản ứng ra là:

0 - 0: Mạch không có hư hỏng;

0 - 1: Mạch có hư hỏng loại 1/(2), 1/(6) hoặc tương đương;

1 - 1: Mạch có hư hỏng loại 1/(5) hoặc 1/(7).

Một cách tương tự, với hàng thứ 3 của hình 12.38 kết hợp với hàng thứ 2 của hình 12.39 ta có:

Trạng thái ban đầu là $a = 1$; trạng thái vào thứ nhất $x_1x_2 = 11$; trạng thái vào thứ hai $x_1x_2 = 01$, phản ứng ra là:

1 - 1: Mạch không có hư hỏng;

1 - 0: Mạch có hư hỏng loại 0/(7), 0/(9);

0 - 1: Mạch có hư hỏng loại 0/(4), 0/(5) hoặc 0/(7).

Do đó một trong những phương án kiểm tra hư hỏng của ô tô máy ở hình 12.34 cho trên hình 12.40.

Trạng thái ban đầu	x_1x_2 (I)	x_1x_2 (II)	Phản ứng ra mạch tốt	Những hư hỏng được phát hiện
0	11	01	0 - 0	1/(2), 0/(3), 1/(5), 1/(6), 1/(7), 1/(8), 1/(9)
1	11	01	1 - 0	0/(4), 0/(5), 0/(7), 0/(8), 0/(9)
1	10	01	0 - 1	0/(1), 0/(2), 1/(1), 0/(6), 0/(9)
1	00	01	0 - 0	1/(1), 1/(4), 1/(6), 1/(7), 1/(8), 1/(9)

Hình 12.40. Một trong những phương án kiểm tra đối với hình 12.34

Từ những phân tích trên đây có thể đưa ra thủ tục tìm bảng kiểm tra như sau:

Bước 1: Lập bảng chuyển đổi trạng thái và bảng ra của ô tô mát cho hai trường hợp khi không hỏng và khi có hỏng đơn.

Bước 2: Từ những hư hỏng được thể hiện trực tiếp trên bảng ra, lập bảng kiểm tra thứ nhất. Trong đó những nhóm các hư hỏng tương đương theo phản ứng ra được gộp lại và chọn một hư hỏng làm tiêu biểu.

Bước 3: Lập bảng kiểm tra thứ hai đối với các hư hỏng còn lại. Trong đó các hư hỏng tương đương theo bước chuyển trạng thái thì gộp lại và chọn một trường hợp đại diện. Tách riêng những hư hỏng có thể thể hiện rõ sau nhịp thử thứ hai.

Bước 4: Nếu còn có các hư hỏng chưa được thể hiện rõ qua phản ứng ra ở nhịp thử thứ hai thì đối với chúng lại tiếp tục tìm chuỗi tác động vào sao cho cuối cùng mạch không hỏng và mạch có hư hỏng sẽ cho những phản ứng ra khác nhau.

Bước 5: Bắt đầu từ chuỗi vào dài nhất đã tìm được, lần lượt đối với mọi chuỗi kiểm tra có trong bảng làm rõ xem chúng cho phát hiện những hư hỏng nào trên đầu ra.

Tất nhiên, để tìm cách kiểm tra ngắn nhất đòi hỏi phải qua nhiều chọn lọc và có khi không thể đạt được. Thông thường, để khẳng định xem ô tô mát có hỏng hay không, ta phải cho tác động vào đầu vào tất cả các chuỗi kiểm tra và so sánh với những số liệu chuẩn. Nếu trùng khớp với các số liệu của mạch tốt thì mạch không có hư hỏng. Nếu ngược lại là có hư hỏng.

Tất nhiên, trước khi cho tác động vào mạch các chuỗi kiểm tra ta phải đưa mạch về trạng thái ban đầu nhất định bằng cách dùng chuỗi thiết lập và chuỗi dẫn. Thủ tục thực hiện việc này như sau:

Bước 1: Chọn chuỗi kiểm tra nào cho phép phát hiện nhiều hư hỏng nhất. Đối với thí dụ trên đây, đó là chuỗi ứng với hàng thứ nhất của hình 12.40. Nó cho phép phát hiện 7 loại hư hỏng. Chuỗi đó gồm: Trạng thái ban đầu $a = 0$, hai chuỗi vào $x_1, x_2 = (11 \text{ và } 01)$.

Bước 2: Nếu không rõ mạch đang ở trạng thái nào thì cho tác động vào mạch chuỗi thiết lập. Thí dụ đối với ô tô mát trong thí dụ trên đây, chuỗi thiết lập ngắn nhất là 00, luôn luôn đưa ô tô mát về trạng thái $a = 0$.

Bước 3: Nếu mạch không được thiết lập vào trạng thái mong muốn thì dùng chuỗi dẫn. Trường hợp của ta không cần.

Bước 4: Cho tác động vào mạch các chuỗi kiểm tra và quan sát ở đầu ra. Nếu kết quả tỏ rõ là mạch không hỏng thì thực hiện bước 5. Nếu kết quả cho thấy mạch đã bị hỏng thì kết thúc và kết luận là mạch bị hỏng.

Bước 5: Nếu còn các chuỗi kiểm tra chưa dùng thì chuyển sang bước 6. Nếu đã hết thì kết thúc và kết luận là mạch không bị hỏng. Đối với thí dụ trên đây, ta còn cần phải thực hiện thêm bước 6.

Bước 6: Chọn chuỗi kiểm tra chưa dùng mà có trạng thái là trạng thái ban đầu mà mạch được thiết lập và chuỗi đó cho phép phát hiện nhiều hư hỏng nhất và quay lại bước 4. Nếu không có chuỗi nào như vậy thì quay lại bước 1. Trong trường hợp thí dụ của ta, cần quay lại bước 1.

Bước 1: Chuỗi kiểm tra lúc đó là: Trạng thái ban đầu là $a = 1$, hai trạng thái vào cần tác động là: 00 và 01, nó cho phép phát hiện 6 hư hỏng.

Bước 2: Không cần chuỗi thiết lập.

Bước 3: Chuỗi dẫn là 0 rồi 1, đưa trạng thái từ 0 về 1.

Bước 4: Khi phản ứng ra là 0 - 0 thì thực hiện bước 5. Nếu không phải thì kết thúc và kết luận là mạch hỏng.

Bước 5: Cần tiếp tục bước 6.

Bước 6: Tiếp tục như đã làm...

Kết quả cuối cùng ta có chuỗi kiểm tra và kết quả phản ứng ra đối với mạch ở hình 12.34 như sau:

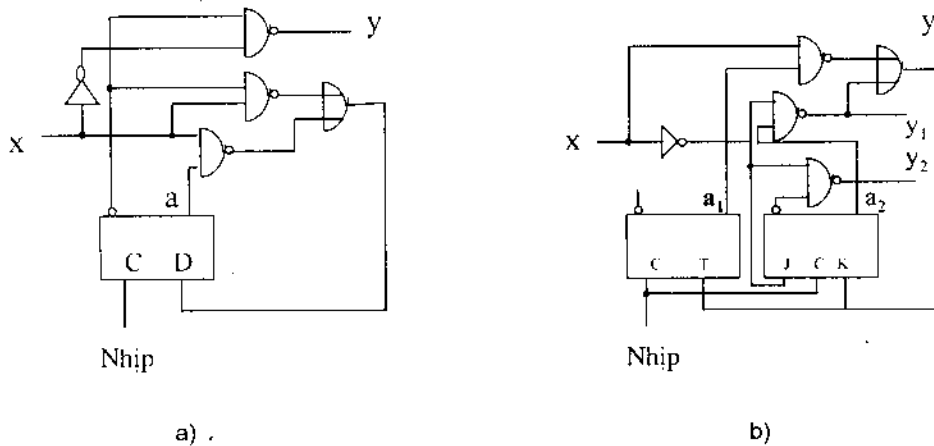
+ Chuỗi kiểm tra: 00 - 11 - 01 - 10 - 00 - 01 - 10 - 11 - 01 - 10 - 10 - 01

+ Chuỗi ra: - 0 0 - 0 0 - 1 1 - 0 1

với mạch đúng. Các phản ứng ra khi thiết lập mạch về trạng thái cần thiết để bắt đầu kiểm tra là không có ý nghĩa. Trong thí dụ trên đây, với mạch không hỏng, các phản ứng ra đó đều bằng 0.

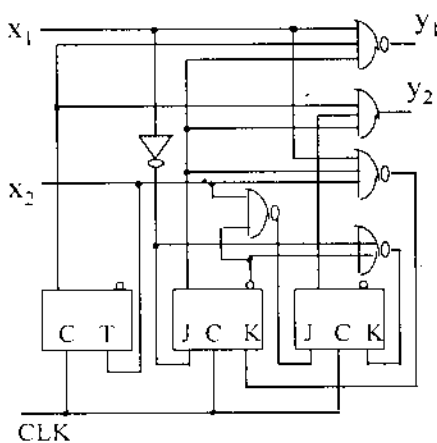
BÀI TẬP CHƯƠNG 12

- Hãy phân tích các mạch cho trên hình 12.41 bằng cách dùng:
 - Bảng hoạt động - bảng chuyển đổi trạng thái của các phần tử nhớ.
 - Đồ hình trạng thái của các phần tử nhớ để nhận được:
 - Bảng chuyển đổi trạng thái và bảng ra của mạch.
 - Đồ hình trạng thái của mạch.
- Vẽ biểu đồ thời gian của tín hiệu ra y của hình 12.41a, khi:
 - Các tín hiệu vào là thế, đồng bộ.
 - Các tín hiệu vào là xung, đồng bộ.
 với chuỗi vào x là: 1011010 và trạng thái ban đầu $a = 1$.
- Vẽ biểu đồ thời gian của y_1 và y_2 của sơ đồ ở hình 12.41b, khi:
 - Các tín hiệu là loại thế đồng bộ.
 - Các tín hiệu là loại xung đồng bộ
 với chuỗi vào $x = 1011001$, trạng thái ban đầu $a_1 a_2 = 01$.

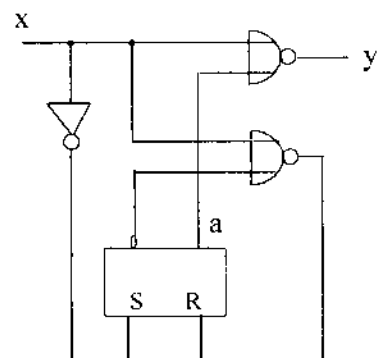


Hình 12.41

4. Cho hình 12.42, tìm:
 - a) Đồ hình chuyển đổi trạng thái?
 - b) Chuỗi tín hiệu ra khi chuỗi vào tác động là $x_1x_2 = 00, 10, 11, 01, 11$ với trạng thái ban đầu $a_1a_2a_3 = 000$.
5. Hãy phân tích ô-tô-mát cho trên hình 12.43 để:
 - a) Nhận bảng chuyển đổi trạng thái và bảng ra?
 - b) Đồ hình trạng thái?
 - c) Xét xem có tranh chấp nguy hiểm hay không?

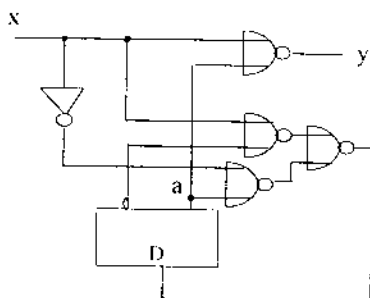


Hình 12.42

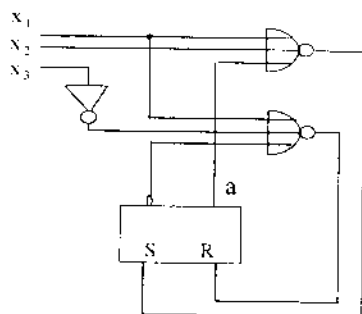


Hình 12.43

6. Xác định tính chất của tín hiệu vào x và tiến hành phân tích động để xem có tranh chấp nguy hiểm mạch ở hình 12.44 hay không?

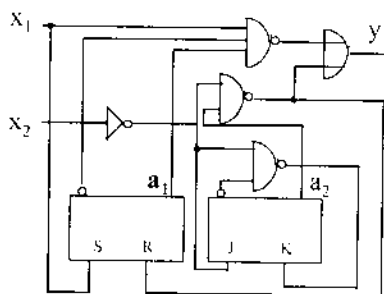


Hình 12.44

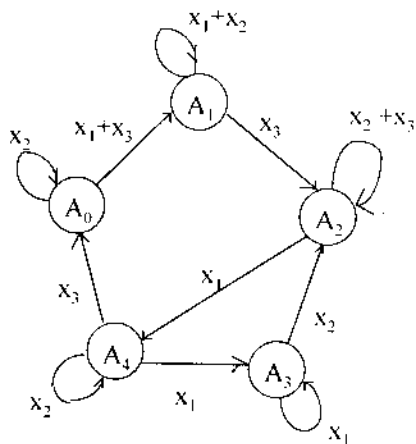


Hình 12.45

7. Phân tích mạch ở hình 12.45 để nhận được:
 - a) Bảng chuyển đổi trạng thái và bảng ra?
 - b) Đồ hình trạng thái?
 - c) Những điều kiện hạn chế đối với các trạng thái vào và các chuỗi vào?
8. Phân tích mạch ở hình 12.46 xem có chuyển biến động và tranh chấp nguy hiểm hay không?
9. Hãy xét xem có tồn tại điều kiện chuyển biến động nguy hiểm đối với mạch ở hình 12.43 hay không?
10. Hãy mã hoá trạng thái trong của ô tômát ở hình 12.43 theo bảng ở hình 12.44 và phân tích xem có tồn tại chuyển biến động nguy hiểm hay không?



Hình 12.46



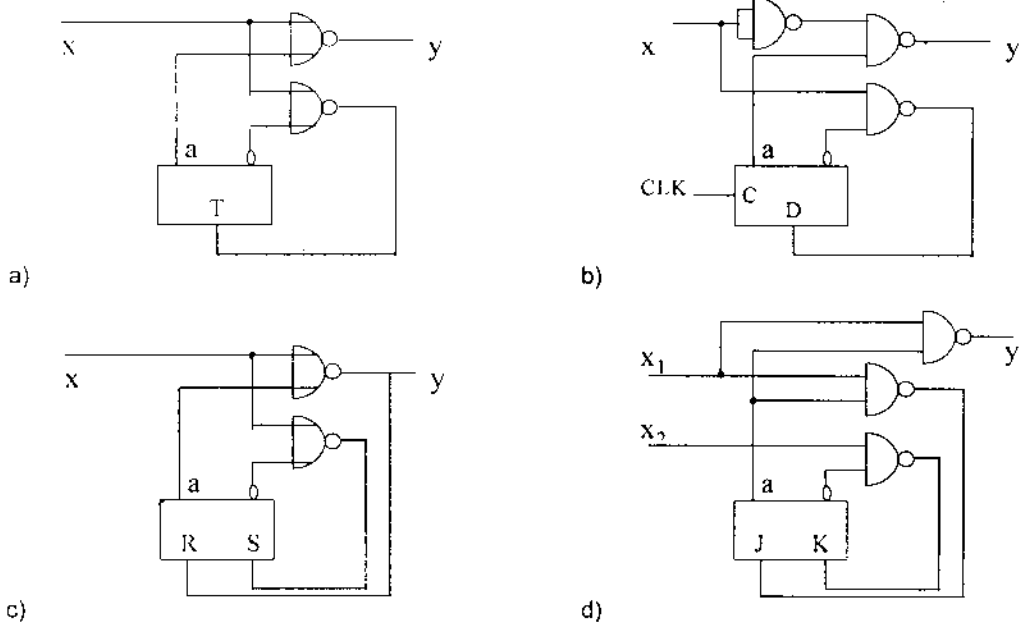
Hình 12.47

A	$a_1a_2a_3$	A	$a_1a_2a_3$
A_0	0 0 0	A_0	0 1 1
A_1	0 0 1	A_1	0 0 1
A_2	0 1 0	A_2	0 1 0
A_3	0 1 1	A_3	1 0 0
A_4	1 0 0	A_4	0 0 0

a) b)

Hình 12.48

11. Tìm chuỗi dẫn từ A_4 đến A_1 đối với thiết bị số cho trên hình 12.47, trong đó:
- Chuỗi dẫn?
 - Chuỗi thiết lập ngắn nhất?
12. Tìm:
- Chuỗi dẫn từ trạng thái 00 đến trạng thái 11?
 - Chuỗi thiết lập? đối với ô tômát cho trên hình 12.48.
13. Tìm bảng kiểm tra đối với mạch ở hình 12.49a, b, c và d.



Hình 12.49

14. Tìm bảng chẩn đoán hư hỏng:
- a) Đối với hình 12.49a b) Đối với hình 12.49d
- c) Đối với hình 12.44 d) Đối với hình 12.45.

Chương 13

TỔNG HỢP THIẾT BỊ SỐ CÓ TÍNH TỚI CÁC ĐẶC ĐIỂM CỦA CÁC PHẦN TỬ NHỚ

Để thực hiện phần nhớ của thiết bị số có nhớ ta dùng các phần tử giữ chậm hay các phần tử nhớ cơ bản. Để thoả mãn các yêu cầu khác nhau của thực tế, người ta sản xuất một số loại phần tử nhớ khác nhau về phương thức làm việc và những yêu cầu khác nhau về tín hiệu đầu vào. Những đặc trưng khác nhau là: Phần tử nhớ làm việc với tín hiệu xung hoặc thế; có đầu vào nhịp hay không có đầu vào nhịp; tín hiệu vào là loại xung hay thế... Khi sử dụng các phần tử nhớ khác nhau thì các bước cần thực hiện trong thiết kế cũng khác nhau. Dưới đây chúng ta sẽ xét một số trường hợp.

13.1. TỔNG HỢP THIẾT BỊ SỐ DÙNG PHẦN TỬ NHỚ ĐƯỢC NHỊP

13.1.1. Thủ tục chung

Các thiết bị số có đầu vào nhịp có thể làm việc với các tín hiệu vào là loại thế hay loại xung, nhưng những tín hiệu này không được thay đổi trong thời gian có xung nhịp.

Do có xung nhịp, khi ô tô mát thay đổi trạng thái, tất cả các phần tử nhớ cần thay đổi trạng thái của mình sẽ được thay đổi đồng thời theo xung nhịp. Do đó không có hiện tượng tranh chấp trạng thái. Nhờ vậy khi mã hoá trạng thái trong không nhất thiết phải mã hoá chống tranh chấp trạng thái, công việc thiết kế sẽ đơn giản hơn nhiều và có thể tập trung vào việc tối ưu các chỉ tiêu khác như đơn giản mạch hay tăng tốc độ làm việc của mạch.

Thủ tục thiết kế các thiết bị số có nhớ dùng các phần tử nhớ được nhịp gồm các bước sau:

1) Phát biểu chức năng của thiết bị, ghi dưới dạng bảng chuyển đổi trạng thái và bảng ra. Đơn giản trạng thái trong của mạch.

2) Mã hoá các trạng thái vào, trạng thái trong và các trạng thái ra. Trong đó không nhất thiết phải mã hoá trạng thái trong bằng các loại mã chống tranh chấp trạng thái. Nhận bảng chuyển đổi trạng thái và bảng ra đã mã hoá.

3) Chọn các loại phần tử nhớ có đầu vào nhậ sẽ sử dụng. Căn cứ vào kết quả bước 2) và các bảng kích thích của các phần tử nhớ đã chọn, rút ra các hàm ra và các hàm kích thích của các phần tử nhớ. Đơn giản các hàm đã nhận được. Trong trường hợp cần thiết, biến đổi các hàm này về dạng mong muốn.

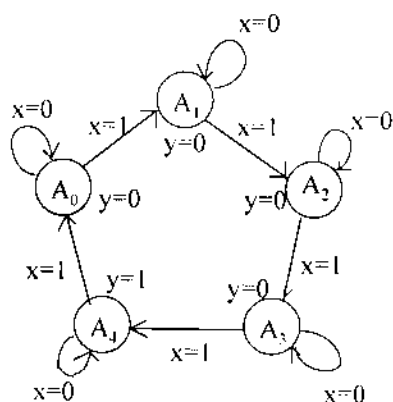
4) Vẽ mạch cấu trúc của mạch.

13.1.2. Một số thí dụ

Để minh hoạ các bước tổng hợp, dưới đây sẽ xét một vài thí dụ tổng hợp các thiết bị số cho dưới dạng bảng chuyển đổi trạng thái và bảng ra.

Thí dụ 13.1: Tổng hợp thiết bị số cho dưới dạng bảng chuyển đổi trạng thái và bảng ra (hình 13.1).

A \ X	0	1	Y
A_0	A_0	A_1	0
A_1	A_1	A_2	0
A_2	A_2	A_3	0
A_3	A_3	A_4	0
A_4	A_4	A_0	1



Hình 13.1. Ôtômát cho thí dụ 13.1

Hình 13.2. Đồ hình trạng thái

Để mã hoá 5 trạng thái trong của ôtômát, ta cần ít nhất ba bit mã, ký hiệu là a_1, a_2, a_3 , chúng đồng thời là ba biến trong của ôtômát. Chọn các phần tử nhớ để thực hiện ba biến trong là ba trigơ CT: CT_1, CT_2, CT_3 .

Vì các trigơ là loại được nhậ, do đó không nhất thiết phải mã hoá chống tranh chấp trạng thái. Một trong các phương án mã hoá có thể như sau:

a_1	a_2	a_3	A
0	0	0	A_0
0	0	1	A_1
0	1	0	A_2
0	1	1	A_3
1	0	0	A_4

Ba tổ hợp mã còn lại của ba bit mã không dùng. Đồ hình trạng thái đã được mã hoá cho trên hình 13.2.

Từ đồ hình trạng thái hoặc bảng chuyển đổi trạng thái ta có thể thiết lập các hàm kích thích của các trigơ CT_1 , CT_2 và CT_3 và của hàm y khi một trong các hàm này bằng 1 như sau:

Các trường hợp $a_1a_2a_3 = 101$ đến 111 không dùng nên với $x = 0$ hay 1 thì các tập hợp giá trị biến tương ứng cũng luôn được coi là bất định (6 trường hợp). Đối với các tổ hợp giá trị biến còn lại của 4 biến $xa_1a_2a_3$, các hàm trên đây đều nhận giá trị 0, nếu $x = 0$ và nhận giá trị cụ thể 0 hay 1 tùy theo từng trường hợp khi $x = 1$ như ở hình 13.3.

$xa_1a_2a_3$	$xa_1a_2a_3^*$	T_1	T_2	T_3	y
1 0 0 0	1 0 0 1	0	0	1	0
1 0 0 1	1 0 1 0	0	1	1	0
1 0 1 0	1 0 1 1	0	0	1	0
1 0 1 1	1 1 0 0	1	1	1	0
1 1 0 0	1 0 0 0	1	0	0	1

Hình 13.3. Giá trị xác định của các hàm khi $x = 1$.

Trong đó, $a_1a_2a_3$ là trạng thái cũ (trạng thái xuất phát) mà dưới tác động của biến x , ô tô máy chuyển sang trạng thái mới $a_1a_2a_3^*$

Đưa các hàm này lên bảng kế cận như trên hình 13.4.

$\begin{matrix} a_2 \\ a_3 \end{matrix}$	$\begin{matrix} x & a_1 & & y \end{matrix}$			
	0	-	-	0
	0	-	-	0
	0	-	-	0
	0	1	1	0

$\begin{matrix} a_2 \\ a_3 \end{matrix}$	$\begin{matrix} x & a_1 & & T_1 \end{matrix}$			
	0	-	-	0
	1	-	-	0
	0	-	-	0
	0	1	0	0

$\begin{matrix} a_2 \\ a_3 \end{matrix}$	$\begin{matrix} x & a_1 & & T_2 \end{matrix}$			
	0	-	-	0
	1	-	-	0
	1	-	-	0
	0	0	0	0

$\begin{matrix} a_2 \\ a_3 \end{matrix}$	$\begin{matrix} x & a_1 & & T_3 \end{matrix}$			
	1	-	-	0
	1	-	-	0
	1	-	-	0
	1	0	0	0

Hình 13.4. Biểu diễn các hàm lên bảng kế cận

Rút ra các hàm :

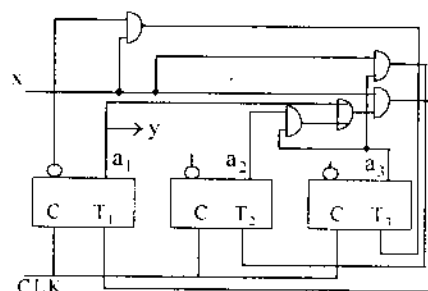
$$y = a_1$$

$$T_1 = x(a_1 + a_2a_3)$$

$$T_2 = xa_3$$

$$T_3 = x\bar{a}_1$$

Mạch thực hiện các hàm trên đây vẽ trên hình 13.5.



Hình 13.5. Mạch thực hiện ô tô mét ở hình 13.1

Thí dụ 13.2. Tổng hợp thiết bị số có nhớ dưới dạng bảng chuyển đổi trạng thái và bảng ra cho trên hình 13.6a, sử dụng các trigơ CJK.

X \ A	X ₀	X ₁
A ₀	A ₁ /Y ₁	A ₀ /Y ₀
A ₁	A ₂ /Y ₂	A ₀ /Y ₀
A ₂	A ₃ /Y ₁	A ₀ Y ₀
A ₃	A ₀ /Y ₂	A ₀ /Y ₀

a)

A	x ₁	x ₂
A ₀	0	0
A ₁	0	1
A ₂	1	0
A ₃	1	1

X	x
X ₀	0
X ₁	1

Y	y ₁	y ₂
Y ₀	0	0
Y ₁	0	1
Y ₂	1	0
-	1	1

b)

Hình 13.6. Ô tô mét cho thí dụ 13.2

Bảng chuyển đổi trạng thái và bảng ra trong trường hợp này đã được tối thiểu hoá.

Mã hoá các trạng thái:

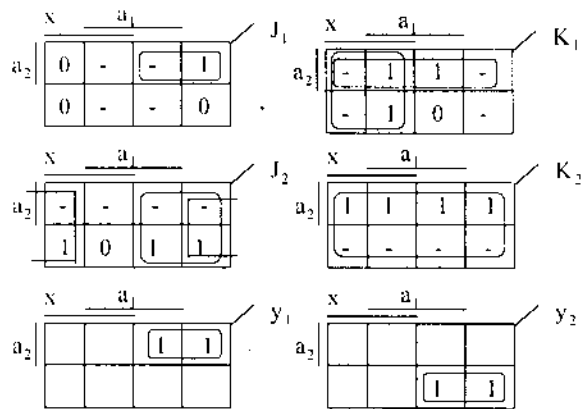
- Vì có hai trạng thái vào X₀ và X₁ nên chỉ cần một biến vào (một بیت mã) x.
- Vì có bốn trạng thái trong A₀, A₁, A₂ và A₃ nên cần hai بیت mã (hai biến trong a₁ và a₂). Vì bắt buộc phải dùng trigơ CJK nên để thực hiện hai biến trong ta dùng hai trigơ CJK trong mạch của ô tô mét.
- Vì có ba trạng thái ra Y₀, Y₁ và Y₂ nên cũng cần hai بیت mã (hai biến ra) y₁ và y₂.

Bảng gán các tổ hợp mã cho các trạng thái trong, vào và ra ở hình 13.6b.

Bảng chuyển đổi trạng thái và bảng ra sau khi đã mã hoá cho trên hình 13.7. Từ đó căn cứ vào bảng kích thích của trigơ CJK ta rút ra các hàm ra và các hàm kích thích của trigơ như trên hình 13.8.

$x \backslash a_1 a_2$	0	1
00	01/01	01/00
01	10/10	00/00
10	11/01	00/00
11	00/10	00/00

Hình 13.7. Bảng chuyển đổi trạng thái và bảng ra kết hợp

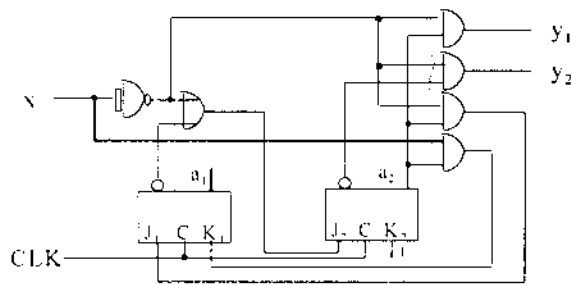


$$J_1 = a_2 \bar{x} \quad K_1 = a_2 + x \quad y_1 = a_2 \bar{x}$$

$$J_2 = \bar{x} + \bar{a}_1 \quad K_2 = 1 \quad y_2 = \bar{a}_2 \bar{x}$$

Hình 13.8. Đơn giản các hàm

Mạch thực hiện cho trên hình 13.9.



Hình 13.9. Mạch thực hiện ô tô-mát ở hình 13.6

Thí dụ 13.3 Tổng hợp ô tô-mát xác định không đầy đủ, có bảng chuyển đổi trạng thái và bảng ra cho trên hình 13.10a.

$X \backslash A$	X_0	X_1	X_2
1	1/0	2/0	1/0
2	1/0	3/0	4/1
3	1/0	4/-	-/-
4	1/0	5/1	-/0
5	1/0	1/0	2/-

a)

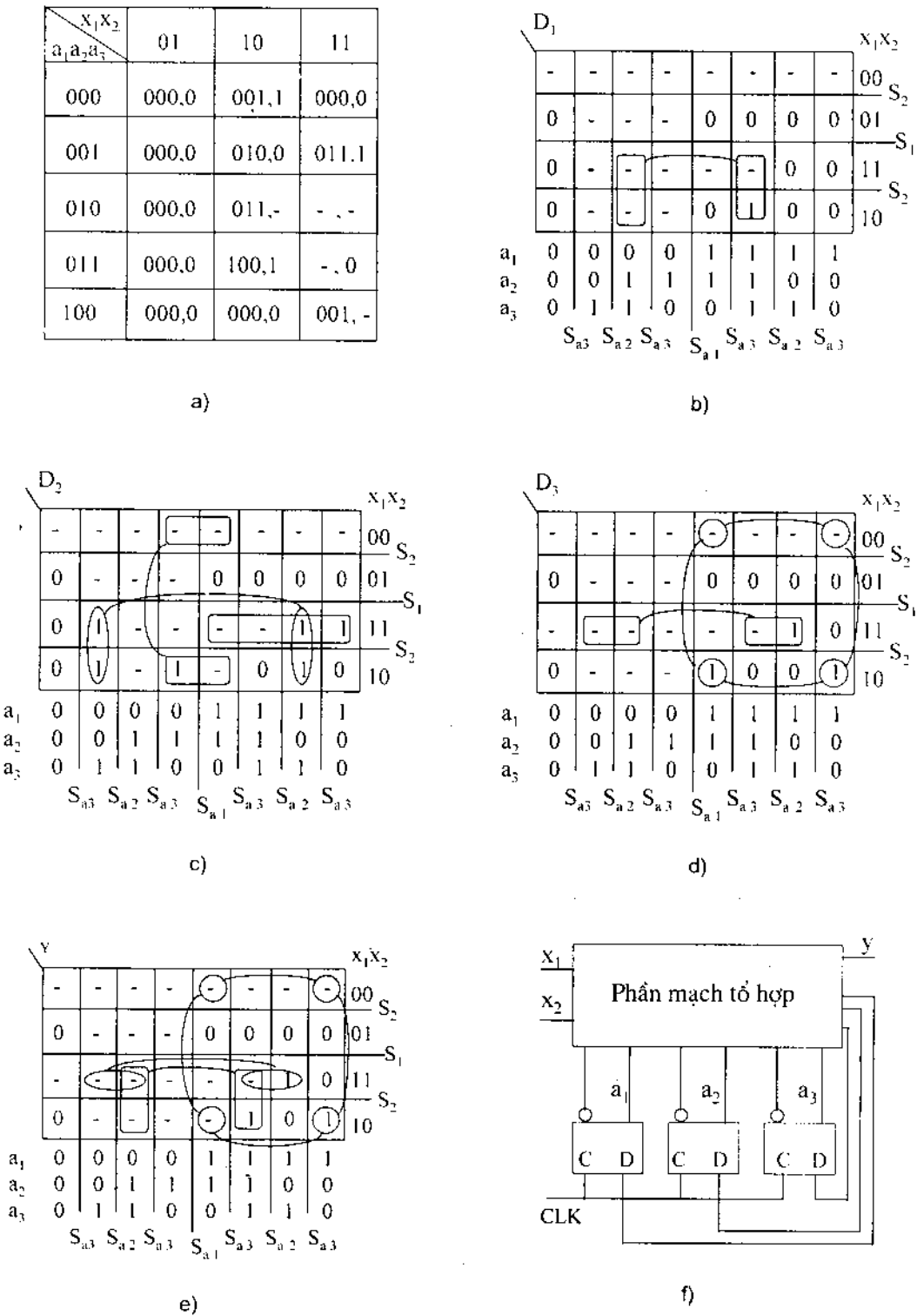
TT	$a_1 a_2 a_3$
1	0 0 0
2	0 0 1
3	0 1 0
4	0 1 1
5	1 0 0

TTv	$x_1 x_2$
x_0	01
x_1	10
x_2	11

TTra	Y
0	0
1	1

b)

Hình 13.10. Ô tô-mát cho thí dụ 13.3



Hình 13.11. Đơn giản, rút ra các hàm kích thích, hàm ra và mạch thực hiện

Để mã hoá các trạng thái vào, ra và trong của ô tô-mát, ta dùng bảng gán mã như ở hình 13.10b.

Sau khi thay các tổ hợp mã vào bảng chuyển đổi trạng thái và bảng ra, ta được bảng chuyển đổi trạng thái và bảng ra đã mã hoá như ở hình 13.11a. Kết hợp với bảng kích thích của các trigger CD, dễ dàng rút ra các hàm D_1 , D_2 , D_3 và y từ các bảng biểu diễn các hàm này trên hình 13.11b, c, d, e như sau:

$$D_1 = x_1 a_2 a_3$$

$$D_2 = x_1 x_2 a_1 + \bar{x}_2 a_2 \bar{a}_3 + x_1 \bar{a}_2 a_3$$

$$D_3 = \bar{x}_2 a_1 \bar{a}_3 + x_1 x_2 a_3$$

$$y = \bar{x}_2 a_1 \bar{a}_3 + x_1 x_2 a_3 + x_1 a_2 a_3$$

Mạch thực hiện cho trên hình 13.11f.

Qua thí dụ trên đây ta thấy rằng, các hàm kích thích và hàm ra là các hàm xác định không đầy đủ. Đó chính là do sự xác định không đầy đủ của các trạng thái trong, các trạng thái vào và sự không đầy đủ của các bước chuyển trạng thái.

Cần lưu ý rằng, mạch được xây dựng từ các hàm kích thích và hàm ra xác định không đầy đủ phải đảm bảo sao cho không xuất hiện các trạng thái vào mà dẫn tới trạng thái không xác định của mạch.

13.2. TỔNG HỢP THIẾT BỊ SỐ DÙNG PHẦN TỬ NHỚ KHÔNG NHỊP, ĐẦU VÀO XUNG

13.2.1. Thủ tục chung

Khi các phần tử nhớ của mạch là các phần tử có đầu vào nhịp thì ô tô-mát làm việc như một ô tô-mát đồng bộ. Khi đó hoạt động của ô tô-mát có các đặc điểm sau đây:

- Chuyển biến trạng thái trong là các chuyển biến đơn, nghĩa là mỗi nhịp chỉ chuyển biến từ một trạng thái này sang một trạng thái khác.

- Tồn tại trạng thái vào thụ động X_0 .

Từ những đặc điểm trên đây, khi thiết kế các thiết bị số có nhớ không được nhịp, đầu vào xung có thể không phải mã hoá chống tranh chấp trạng thái. Do đó có thể tập trung vào việc mã hoá làm cho mạch đơn giản hoặc có tốc độ nhanh. Tuy vậy, trong trường hợp này phải chọn các phần tử nhớ sao cho thời gian chuyển trạng thái của chúng như nhau hoặc gần như nhau. Do đó trong trường hợp cần thận trọng vẫn nên mã hoá chống tranh chấp trạng thái.

Như chúng ta đã biết, khi các phần tử nhớ không có đầu vào nhịp thì để tránh nguy hiểm nên đảm bảo các điều kiện sau:

- Tại một thời điểm chỉ một tín hiệu đầu vào thay đổi giá trị tạo nên trạng thái xung của trạng thái vào.

- Mỗi lần thay đổi trạng thái trong chỉ thay đổi một biến trong - trạng thái ra của một trigơ.

- Phân mạch tổ hợp phải được thực hiện bằng dạng chống được tranh chấp, tức dạng tuyến chuẩn tối thiểu đầy đủ.

Dưới đây là thủ tục thiết kế cho hai trường hợp.

a) Khi không dùng mã hoá chống tranh chấp trạng thái

+ Thực hiện mã hoá các trạng thái vào, ra và trong của ô tô mát bằng các tổ hợp mã nhị phân thông thường.

+ Chọn các phần tử nhớ và thực hiện rút ra các hàm ra và các hàm kích thích của các trigơ.

+ Biến đổi các hàm nhận được trên đây về dạng cần thiết và thực hiện bằng các mạch lôgic.

+ Thực hiện phân tích mạch. Nếu thấy có hiện tượng tranh chấp trạng thái thì cần mã hoá lại đối với trạng thái trong của mạch.

b) Trường hợp mã hoá chống tranh chấp trạng thái

+ Mã hoá các trạng thái vào, ra bằng mã thông thường. Các trạng thái trong mã bằng các mã chống tranh chấp trạng thái.

+ Rút ra các hàm ra và các hàm kích thích của các phần tử nhớ.

+ Biến đổi về dạng cần thiết nếu cần và xây dựng mạch.

Trong trường hợp này sẽ không có tranh chấp trạng thái nguy hiểm.

13.2.2. Một số thí dụ

Thí dụ 13.4

Tổng hợp ô tô mát cho dưới dạng bảng chuyển đổi trạng thái và bảng ra (hình 13.12a), không mã hoá chống tranh chấp trạng thái.

Vì ô tô mát có 4 trạng thái trong nên cần hai biến trong, ký hiệu a_1 và a_2 . Để thực hiện hai biến này ta dùng hai phần tử nhớ cơ bản. Cụ thể chọn hai trigơ D: D_1 và D_2 .

Với hai trạng thái vào X_1 và X_2 là các trạng thái tích cực, nên cần ít nhất một trạng thái không tích cực nữa. Do đó có ba trạng thái vào, ký hiệu X_0 , X_1 , và X_2 . Ta dùng hai biến vào x_1 , x_2 để mã hoá ba trạng thái vào này.

Với hai trạng thái ra: 0 và 1, ta dùng một biến ra y để mã hoá hai trạng thái này.

Bảng mã hoá các trạng thái của ô tô mát cho trên hình 13.12b. Bảng chuyển đổi trạng thái và bảng ra kết hợp cho trên hình 13.12c.

Y	X	X ₁	X ₂
0	0	1	0
0	1	2	0
0	2	3	1
1	3	0	0

a)

Λ	a ₁ a ₂	X	x ₁ x ₂		Y	y
0	00	X ₀	00	Thu động	0	0
1	01	X ₁	10	Tích cực	1	1
2	10	X ₂	01	Tích cực		
3	11	-	11	Bỏ		

y	x ₁ x ₂ a ₁ a ₂	00	01	11	10
0	00	00	00	-	01
0	01	01	00	-	10
0	10	10	01	-	11
1	11	11	00	-	00

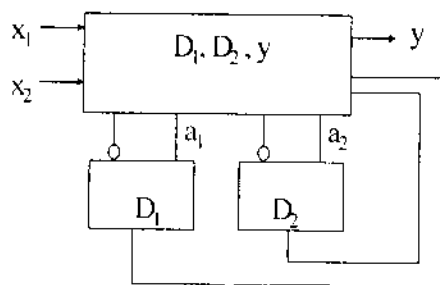
c)

Hình 13.12. Ôtômát cho thí dụ 13.4

Từ bảng chuyển đổi trạng thái và bảng ra đã mã hoá kết hợp với bảng kích thích của trigơ D đã chọn, ta có thể biểu diễn các hàm ra và các hàm kích thích của các trigơ D trên bảng kế cận, rút ra các hàm và thực hiện chúng như trên hình 13.13.

$$\begin{array}{c}
 \begin{array}{c} x_1 \quad x_2 \\ \begin{array}{|c|c|c|c|} \hline 1 & - & 0 & 1 \\ \hline 0 & - & 0 & 1 \\ \hline 1 & - & 0 & 0 \\ \hline 0 & - & 0 & 0 \\ \hline \end{array} \\ \begin{array}{l} a_1 \\ a_2 \end{array} \end{array} \quad D_1 \quad \begin{array}{c} x_1 \quad x_2 \\ \begin{array}{|c|c|c|c|} \hline 1 & - & 1 & 0 \\ \hline 0 & - & 0 & 1 \\ \hline 0 & - & 0 & 1 \\ \hline 1 & - & 0 & 0 \\ \hline \end{array} \\ \begin{array}{l} a_1 \\ a_2 \end{array} \end{array} \quad D_2 \quad \begin{array}{c} x_1 \quad x_2 \\ \begin{array}{|c|c|c|c|} \hline 0 & - & 0 & 0 \\ \hline 1 & - & 0 & 1 \\ \hline 0 & - & 0 & 0 \\ \hline 0 & - & 0 & 0 \\ \hline \end{array} \\ \begin{array}{l} a_1 \\ a_2 \end{array} \end{array} \quad y
 \end{array}$$

$$D_1 = a_1 \bar{x}_1 \bar{x}_2 + a_1 \bar{a}_2 x_1 + \bar{a}_1 a_2 x_1 \quad D_2 = a_2 \bar{x}_1 \bar{x}_2 + \bar{a}_2 x_1 + a_1 \bar{a}_2 x_2 \quad y = a_1 a_2$$



Hình 13.13. Mạch thực hiện ôtômát ở hình 13.12

Việc phân tích mạch đã được thiết kế có thể xem trong phần phân tích mạch dây. Trong trường hợp này, vì phần mạch tổ hợp đều thực hiện ở dạng tuyến chuẩn tối thiểu đầy đủ nên tránh được tranh chấp trạng thái một vị trí.

Mặt khác, do cách mã hoá đã chọn, trong chuyển đổi trạng thái xuất hiện các bước chuyển trong đó có hai biến trong thay đổi giá trị. Đó là các bước chuyển giữa các trạng thái trong sau đây:

$$10-01; 01-10; 11-00.$$

Do đó muốn biết mạch có tranh chấp nguy hiểm hay không cần áp dụng các phương pháp phân tích để làm rõ.

Bây giờ chúng ta sẽ thiết kế theo cách sẽ mã hoá chống tranh chấp trạng thái.

Như ta đã rõ, việc mã hoá kế cận không phải lúc nào cũng có thể thực hiện được. Nhưng đối với ô tô măt ở hình 13.12a hoàn toàn có thể mã hoá kế cận như trên hình 13.14 và hình 13.15. Trên cơ sở đó ta biểu diễn các hàm D_1 , D_2 và y trên bảng kế cận trên hình 13.16. Các hàm này được rút ra như sau:

$$D_1 = a_2x_1 + a_1\bar{x}_1\bar{x}_2$$

$$D_2 = \bar{a}_1x_1 + a_1a_2x_2 + a_2\bar{x}_1\bar{x}_2$$

$$y = a_1\bar{a}_2$$

A	a_1a_2
0	00
1	01
2	11
3	10

Hình 13.14. Mã hóa kế cận

	x_1x_2		
y	a_1a_2	10	01
0	00	01	00
0	01	11	00
0	11	10	01
1	10	00	00

Hình 13.15

	x_1	x_2	
y	a_1	a_2	
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0

	x_1	x_2	
D_1	a_1	a_2	
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0

Hình 13.16

Mạch thực hiện cho trên hình 13.17.

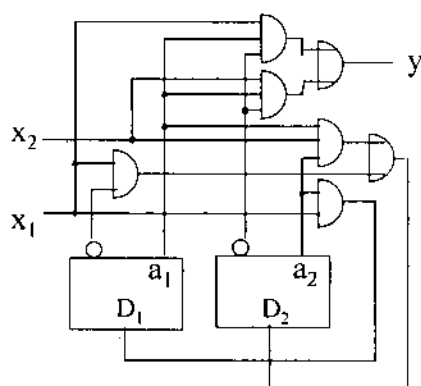
Trong mạch này D_2 không được thực hiện bằng dạng tối thiểu đầy đủ nên có khả năng có tranh chấp loại 1 một vị trí. Nếu thay biểu thức của D_2 bằng dạng tuyến chuẩn tối thiểu đầy đủ thì có thể tránh được hiện tượng tranh chấp trên.

Vì trạng thái trong của ô tô măt được mã hoá bằng các mã kế cận nên sẽ không có tranh chấp trạng thái nguy hiểm.

Một lần nữa cần lưu ý rằng, nếu đồ hình của ô tô măt có dạng không cho phép mã hoá kế cận thì nên:

- + Hoặc thực hiện bằng phần tử nhớ có đầu vào nhíp.
- + Hoặc tìm algôrit mã hoá sao cho chống được tranh chấp trạng thái.

Phần mạch tổ hợp thực hiện dưới dạng tuyến chuẩn tối thiểu đầy đủ để chống tranh chấp trạng thái của mạch tổ hợp.

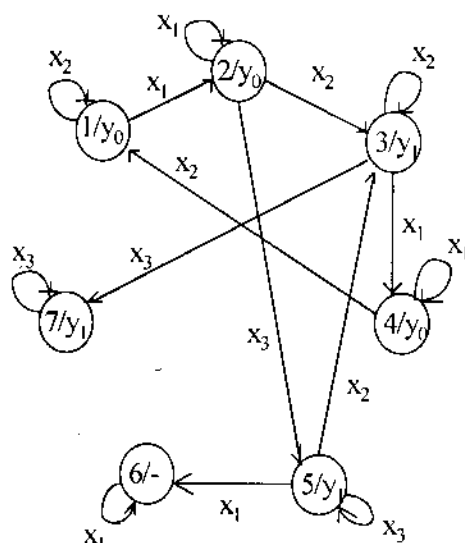


Hình 13.17. Mạch thực hiện khi mã hoá kế cận

Thí dụ 13.5: Áp dụng algôrit mã hoá chống tranh chấp trạng thái để tổng hợp ôôtômát cho trên hình 13.18a.

A	X			Y
	x_1	x_2	x_3	
1	2	1	-	y_0
2	2	3	5	y_0
3	4	3	7	y_1
4	4	1	-	y_0
5	6	3	3	y_1
6	6	-	-	-
7	-	-	7	y_1

a)



b)

Hình 13.18. Ôôtômát cho thí dụ 13.5

Vì đồ hình trạng thái của ôôtômát ở hình 13.18b không thoả mãn điều kiện mã hoá kế cận, ta áp dụng algôrit mã hoá chống tranh chấp trạng thái đã trình bày trong phần trước.

Để mã hoá 7 trạng thái của ô tô máy, sau khi đã dùng phương pháp đơn giản số bit của bộ mã, kết quả áp dụng algôrit đã trình bày trên đây, ta gán các bộ mã như trình bày trên hình 13.19a. Việc mã hoá các trạng thái vào và trạng thái ra tiến hành bình thường và cho trên hình 13.19b, c.

Như vậy cũng chỉ cần ba trigô để thực hiện ba biến trong a_1, a_2 và a_3 .

Kết quả ta có bảng chuyển đổi trạng thái và bảng ra đã mã hoá cho trên hình 13.20a. Từ đó có thể rút ra các hàm $R_1, S_1; R_2, S_2; R_3, S_3$ như đã biểu diễn trên hình 13.20b, c, d, e, f, g và h.

A	$a_1 a_2 a_3$
1	100
2	000
3	010
4	110
5	001
6	101
7	011

a)

X	$x_1 x_2 x_3$
x_0	000
x_1	100
x_2	010
x_3	001

b)

y	Y
0	0
1	1

c)

Hình 13.19. Phương án gán mã

A	X			Y
$a_1 a_2 a_3$	100	010	001	y
100	000	100	-	0
000	000	010	001	0
010	110	010	011	1
110	110	100	-	0
001	101	010	010	1
101	101	-	-	-
011	-	-	011	1

a) Bảng chuyển đổi trạng thái và bảng ra đã mã hóa

								$x_1 x_2 x_3$
-	-	-	-	-	-	-	-	000
0	0	0	0	0	0	0	-	001
-	-	-	-	-	-	-	-	011
0	-	-	0	-	-	-	-	010
-	-	-	-	-	-	-	-	110
-	-	-	-	-	-	-	-	111
-	-	-	-	-	-	-	-	101
0	0	0	0	0	-	0	-	100
1	1	1	1	0	0	0	0	a_1
0	0	1	1	1	1	0	0	a_2
0	1	1	0	0	1	1	0	a_3

b) Hàm R_1

Hình 13.20a, b

-	-	-	-	-	-	-	-	$x_1x_2x_3$	000
-	-	-	-	0	0	0	0		001
-	-	-	-	-	-	-	-		011
-	-	-	-	0	-	0	0		010
-	-	-	-	-	-	-	-		110
-	-	-	-	-	-	-	-		111
-	-	-	-	-	-	-	-		101
0	-	-	-	1	-	1	0		100
1	1	1	1	0	0	0	0	a_1	
0	0	1	1	1	1	0	0	a_2	
0	1	1	0	0	1	1	0	a_3	

c) Hàm S_1

-	-	-	-	-	-	-	-	$x_1x_2x_3$	000
-	-	-	-	0	0	-	-		001
-	-	-	-	-	-	-	-		011
-	-	-	1	0	-	0	-		010
-	-	-	-	-	-	-	-		110
-	-	-	-	-	-	-	-		111
-	-	-	-	-	-	-	-		101
-	-	-	0	0	-	-	-		100
1	1	1	1	0	0	0	0	a_1	
0	0	1	1	1	1	0	0	a_2	
0	1	1	0	0	1	1	0	a_3	

d) Hàm R_2

-	-	-	-	-	-	-	-	$x_1x_2x_3$	000
-	-	-	-	-	-	0	0		001
-	-	-	-	-	-	-	-		011
0	-	-	0	-	-	1	1		010
-	-	-	-	-	-	-	-		110
-	-	-	-	-	-	-	-		111
-	-	-	-	-	-	-	-		101
0	0	-	-	-	-	0	0		100
1	1	1	1	0	0	0	0	a_1	
0	0	1	0	0	1	1	0	a_2	
0	1	1	0	0	1	1	0	a_3	

e) Hàm S_2

-	-	-	-	-	-	-	-	$x_1x_2x_3$	000
-	-	-	-	0	0	0	0		001
-	-	-	-	-	-	-	-		011
-	-	-	-	-	-	1	-		010
-	-	-	-	-	-	-	-		110
-	-	-	-	-	-	-	-		111
-	-	-	-	-	-	-	-		101
-	0	-	-	-	-	0	-		100
1	1	1	1	0	0	0	0	a_1	
0	0	1	0	0	1	1	0	a_2	
0	1	1	0	0	1	1	0	a_3	

f) Hàm R_3

-	-	-	-	-	-	-	-	$x_1x_2x_3$	000
-	-	-	-	1	-	-	-		001
-	-	-	-	-	-	-	-		011
0	-	-	0	0	-	0	0		010
-	-	-	-	-	-	-	-		110
-	-	-	-	-	-	-	-		111
-	-	-	-	-	-	-	-		101
0	-	-	0	0	-	-	0		100
1	1	1	1	0	0	0	0	a_1	
0	0	1	0	0	1	1	0	a_2	
0	1	1	0	0	1	1	0	a_3	

g) Hàm S_3

-	-	-	-	-	-	-	-	$x_1x_2x_3$	000
0	-	-	0	1	1	1	0		001
-	-	-	-	-	-	-	-		011
-	-	-	-	1	1	1	0		010
-	-	-	-	-	-	-	-		110
-	-	-	-	-	-	-	-		111
-	-	-	-	-	-	-	-		101
0	-	-	0	1	1	1	0		100
1	1	1	1	0	0	0	0	a_1	
0	0	1	0	0	1	1	0	a_2	
0	1	1	0	0	1	1	0	a_3	

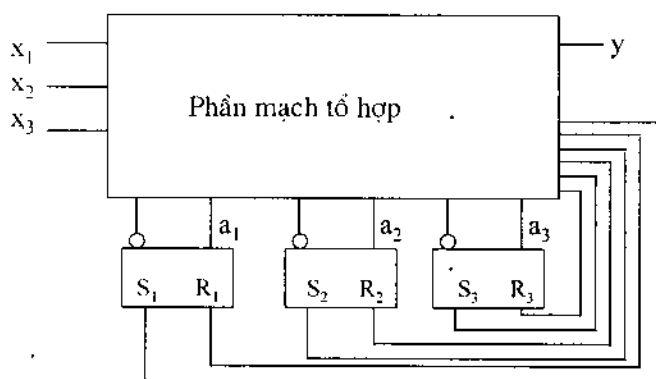
h) Hàm y

Hình 13.20. Đơn giản và rút ra các hàm kích thích

Từ đó rút ra các hàm sau:

$$\begin{aligned} R_1 &= \bar{a}_2 \bar{a}_3 x_1; & R_2 &= a_1 x_2; & R_3 &= x_2; & y &= \bar{a}_1 a_2 + a_3 \\ S_1 &= a_3 x_1 + a_2 x_1; & S_2 &= \bar{a}_1 x_2; & S_3 &= x_3. \end{aligned}$$

Mạch thực hiện cho trên hình 13.21, trong đó phần mạch tổ hợp thực hiện các hàm $R_1, S_1, R_2, S_2, R_3, S_3$ và y đã nhận được trên đây.



Hình 13.21. Mạch thực hiện ô tô mát ở hình 13.18

13.3. TỔNG HỢP THIẾT BỊ SỐ CÓ NHỚ KHI PHẦN TỬ NHỚ KHÔNG ĐƯỢC NHẬP, ĐẦU VÀO THỂ

13.3.1. Thủ tục chung

Khi các trạng thái vào của ô tô mát là loại thể, ô tô mát có các đặc điểm sau đây:

- Là loại ô tô mát không đồng bộ.
- Các trạng thái trong chia thành các trạng thái bền và không bền.

Để ô tô mát loại này làm việc đúng, yêu cầu đối với các trạng thái vào là:

- + Tại một thời điểm chỉ được thay đổi giá trị của một biến vào.
- + Trạng thái vào chỉ thay đổi sau khi ô tô mát đã thiết lập trạng thái bền.

-- Để ô tô mát làm việc đúng, cần đảm bảo không có tranh chấp trạng thái trong phần mạch tổ hợp và không có tranh chấp động đối với trạng thái trong của ô tô mát. Vì vậy cần:

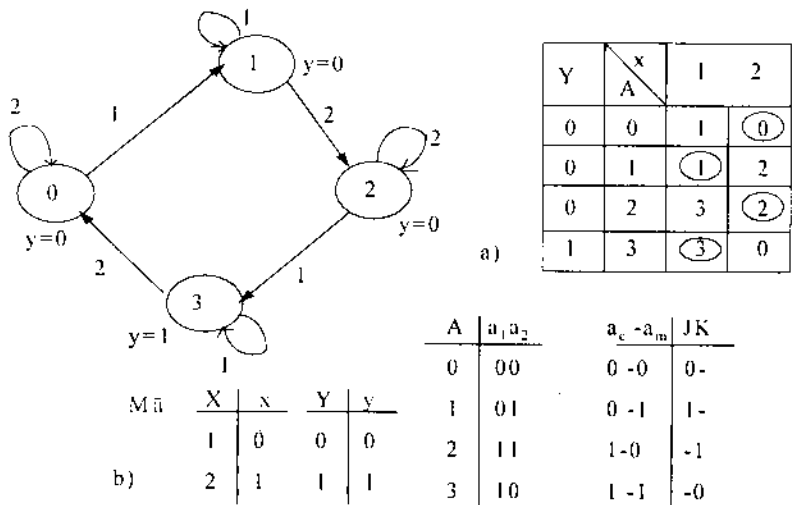
- + Phần mạch tổ hợp thực hiện bằng dạng tuyển chuẩn tối thiểu đầy đủ.
- + Trạng thái trong phải được mã bằng các mã hoá chống tranh chấp trạng thái như mã kế cận chẳng hạn.

Từ các phân tích trên đây, thủ tục tổng hợp mạch trong trường hợp này là:

- + Mã hoá các trạng thái vào, ra và các trạng thái trong. Các trạng thái trong phải được mã bằng các mã chống tranh chấp trạng thái. Lập bảng chuyển đổi trạng thái và bảng ra đã mã hoá.
- + Chọn các phần tử nhớ cụ thể.
- + Rút ra các hàm ra và các hàm kích thích của các trigơ dưới dạng tuyến chuẩn tối thiểu đầy đủ.
- + Thực hiện kỹ thuật các hàm và toàn mạch của ô tômát.

13.3.2. Một số thí dụ

Thí dụ 13.6. Tổng hợp ô tômát cho dưới dạng bảng chuyển đổi trạng thái và bảng ra ở hình 13.22.



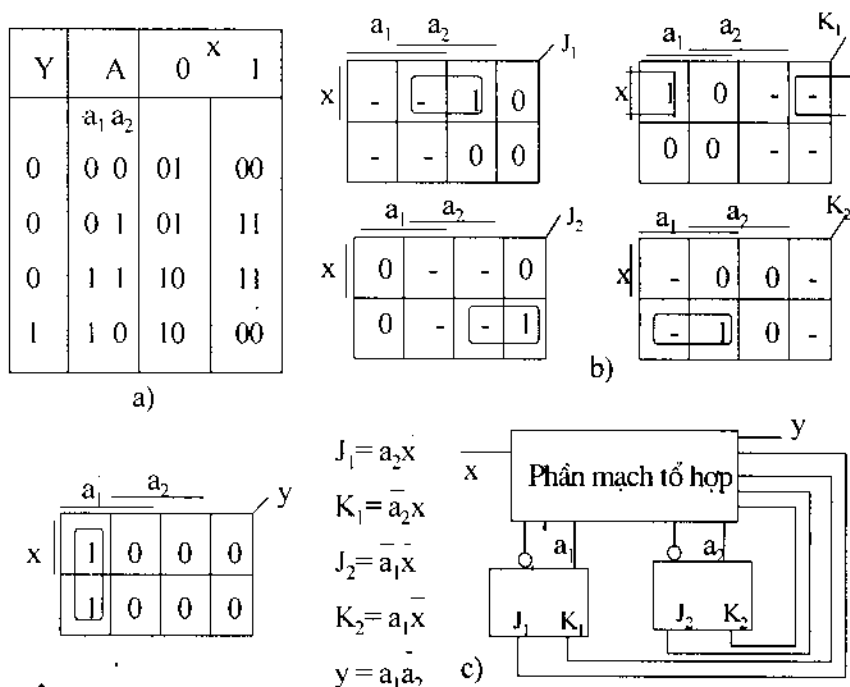
Hình 13.22. Ô tômát cho thí dụ 13.6

Để mã hoá hai trạng thái vào chỉ cần một biến vào, ký hiệu là x . Để mã hoá hai trạng thái ra cũng chỉ cần một biến ra, ký hiệu là y . Bốn trạng thái trong, với hai biến trong a_1 và a_2 , căn cứ vào đồ hình trạng thái của ô tômát, có thể mã hoá kế cận - hình 13.22b. Cuối cùng ta có bảng chuyển đổi trạng thái và bảng ra đã mã hoá như ở hình 13.23a.

Từ bảng chuyển đổi trạng thái và bảng ra, kết hợp với bảng kích thích của trigơ JK, dễ dàng rút ra các hàm J_1, K_1, J_2, K_2 và y - hình 13.23b.

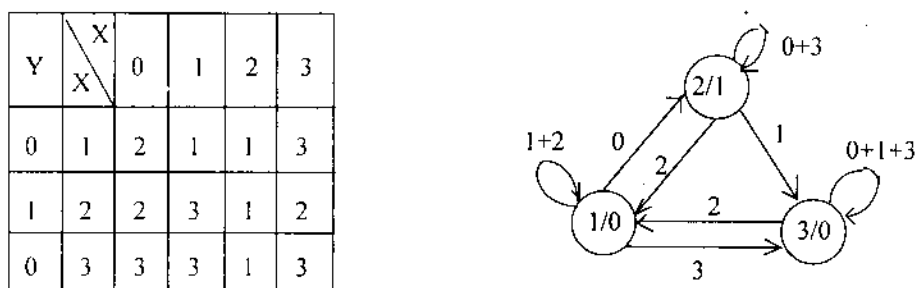
Mạch thực hiện ô tômát đã cho trên hình 13.23c, trong đó phần mạch tổ hợp thực hiện các hàm J_1, K_1, J_2, K_2, y .

Vì mạch phần tổ hợp được thực hiện bằng dạng tuyến chuẩn tối thiểu đầy đủ, còn các trạng thái trong được mã hoá bằng các mã kế cận nên ô tômát này làm việc sẽ không có hiện tượng tranh chấp trạng thái, mặc dù các trigơ JK không có đầu vào nhệp.



Hình 13.23. Bảng mã hoá, các hàm kích thích và mạch thực hiện ôtomát ở hình 13.22

Thí dụ 13.7: Thiết kế ôtomát cho trên hình 13.24.



Hình 13.24. Ôtomát cho thí dụ 13.7

Ôtomát này có ba trạng thái. Đồ hình của ôtomát thể hiện không thể mã hoá kế cận ngay được. Để có thể mã hoá kế cận, cần thêm trạng thái phụ - trạng thái 4 như ở hình 13.25a. Trên cơ sở đó có thể mã hoá kế cận như đã chỉ ra trên hình 13.25b.

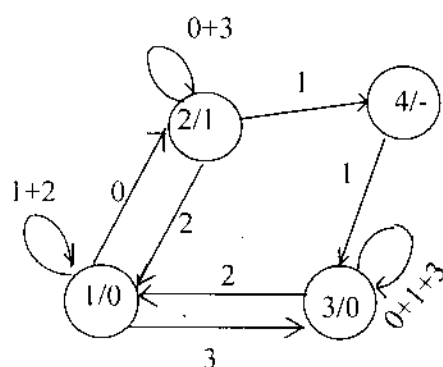
Trên cơ sở bảng chuyển đổi trạng thái và bảng ra đã mã hoá (hình 13.25), biểu diễn các hàm y và T_1, T_2 trên bảng kế cận như trên hình 13.26.

Từ đó rút ra các hàm sau:

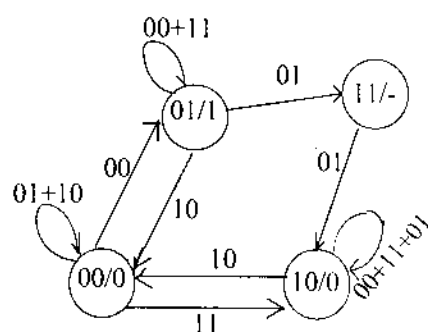
$$T_1 = x_1 \bar{x}_2 a_1 + \bar{x}_1 x_2 \bar{a}_1 a_2 + x_1 x_2 \bar{a}_1 \bar{a}_2$$

$$T_2 = x_1 \bar{x}_2 a_2 + \bar{x}_1 \bar{x}_2 \bar{a}_1 \bar{a}_2 + a_1 a_2$$

$$y = a_2$$



a)



b)

Hình 13.25. Thêm trạng thái phụ a) và thực hiện mã hoá kế cận b)

	x_1	x_2	T_1	
a_1	1	0	0	0
a_2	-	-	0	-
	0	0	1	0
	0	1	0	0

	x_1	x_2	T_2	
a_1	0	0	0	0
a_2	-	-	1	-
	1	0	0	0
	0	0	0	1

	x_1	x_2	y	
a_1	0	0	0	0
a_2	-	-	-	-
	1	1	1	1
	0	0	0	0

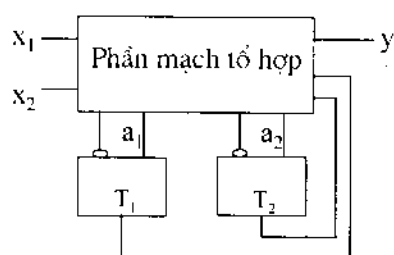
Hình 13.26. Biểu diễn hàm lên bảng kế cận

Mạch thực hiện ô tô-mát đã cho như trên hình 13.27, trong đó phần mạch tổ hợp thực hiện các hàm logic y , T_1 và T_2 .

Nhận xét:

- Có thể có nhiều phương án bổ sung các trạng thái phụ để phá vỡ thế không thể thực hiện bằng mã kế cận. Tất nhiên mỗi phương án có thể dẫn đến độ phức tạp khác nhau của mạch. Trong mọi trường hợp, việc gán các trạng thái ra cho các trạng thái phụ cần sao cho ô tô-mát mới và cũ phải là các ô tô-mát tương đương.

- Trong trường hợp không dùng phương pháp thêm trạng thái phụ, có thể dùng phương pháp mã hoá các trạng thái trong bảng algôrit mã hoá chống tranh chấp trạng thái.



Hình 13.27. Mô hình thực hiện ô tô-mát ở hình 13.25

BÀI TẬP CHƯƠNG 13

1. Tổng hợp ô tô máy Milly có bảng chuyển đổi trạng thái và bảng ra cho trên hình 13.28.

a) Dùng phần tử nhớ CT.

b) Dùng phần tử nhớ CD.

2. Tổng hợp thiết bị số có nhớ có bảng chuyển đổi trạng thái và bảng ra kết hợp như ở hình 13.29, dùng trigger:

a) CT;

b) CD;

c) CJK;

d) CRS

A \ X	0	1
	0	1
0	1/0	0/1
1	0/0	1/0

Hình 13.28

A \ X	0	1
	00	01
00	01/01	00/00
01	10/10	00/00
10	11/01	00/00
11	00/10	00/00

Hình 13.29

3. Tổng hợp ô tô máy ở hình 13.30, nếu trạng thái vào được mã hoá như ở hình 13.31. Kiểm tra xem chu trình làm việc của mạch có vấn đề gì đặc biệt hay không?

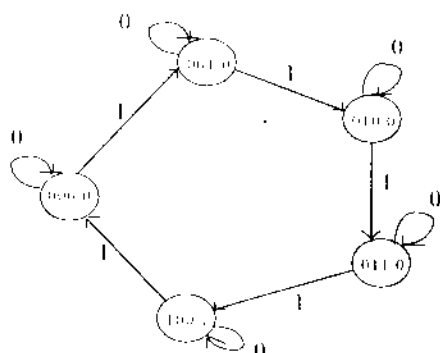
A \ X	1	2	3
	1	2	3
1	1/0	2/1	3/0
2	1/0	3/0	4/1
3	1/0	4/-	-/-
4	1/0	5/1	-/0
5	1/0	1/0	2/-

Hình 13.30

X	x_1x_2
1	00
2	01
3	10

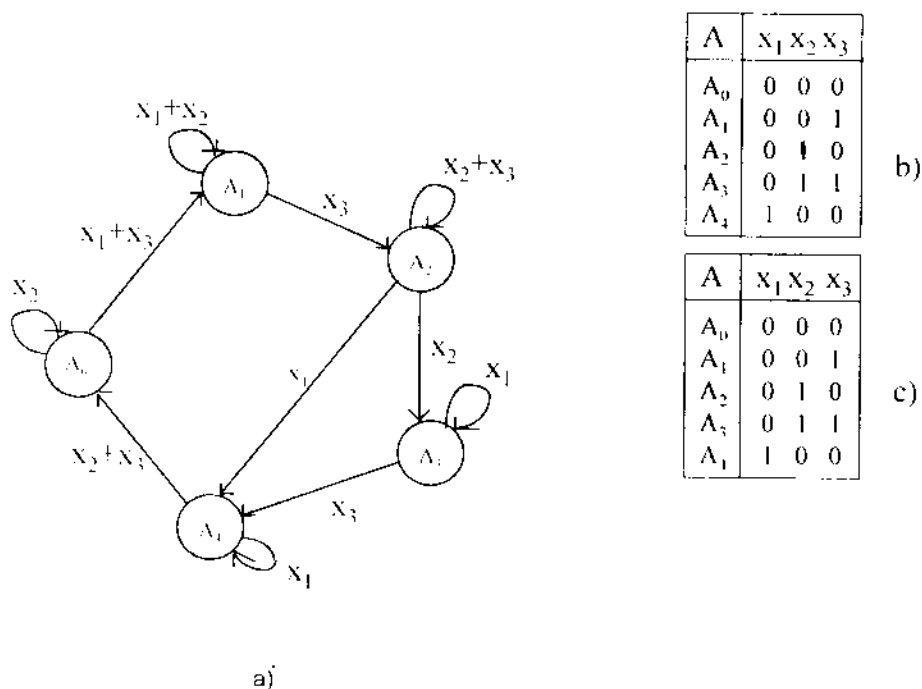
Hình 13.31

4. Tổng hợp mạch dây có đồ hình trạng thái cho trên hình 13.32, dùng phần tử nhớ loại CJK. Hãy chỉ ra những đặc điểm hoạt động của mạch?



Hình 13.32

5. Thực hiện ôtomát cho trên hình 13.33a khi mã hoá các trạng thái trong theo các phương án cho trên hình 13.33b, c.



Hình 13.33

- a) Dùng trigơ CT.
 b) Dùng trigơ CD.
 c) Dùng trigơ CRS.
 d) Dùng trigơ CJK.

Hãy so sánh cấu trúc của các mạch đã thiết kế về mặt độ phức tạp của mạch.

6. Tổng hợp thiết bị số hoạt động theo bảng chuyển đổi trạng thái và bảng ra ở hình 13.34a, b, nếu các trạng thái vào X_1 , X_2 và X_3 là xung:

- a) Dừng trigơ D
b) Dừng trigơ T
c) Dừng trigơ RS
d) Dừng trigơ JK

$\begin{matrix} X \\ A \end{matrix}$	X_1	X_2	X_3
1	1/0	2/0	3/1
2	2/0	3/0	4/0
3	3/1	4/1	1/0
4	4/0	1/0	2/0

a)

$\begin{matrix} X \\ A \end{matrix}$	X_1	X_2	X_3
1	1/0	2/0	4/0
2	2/0	3/1	1/0
3	3/1	4/0	2/0
4	4/0	1/0	1/1

b)

Hình 13.34

7. Tổng hợp bộ đếm môđun 10, đầu vào xung x. Trạng thái trong đồng thời cũng là trạng thái ra của bộ đếm, được biểu diễn dưới dạng mã:

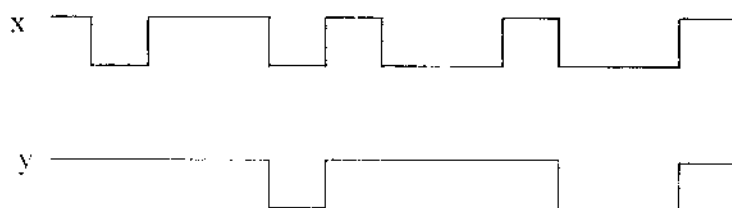
- a) 8421
b) 2421
c) 1 trong N.

8. Hãy tổng hợp thiết bị số có hai đầu vào x_1 , x_2 và một đầu ra y. Hoạt động của thiết bị như sau: Đầu ra y thay đổi từ 0 sang 1 khi xuất hiện xung cuối cùng của chuỗi:

- a) $x_1 \cdot x_2 = x_1$
b) $x_1 = x_2 = x_1 \cdot x_2$
c) $x_1 = x_1 = x_2$

Đầu ra y sẽ đổi giá trị của mình từ 1 sang 0 khi xuất hiện đầu tiên trên đầu x_2 sau khi đầu ra của thiết bị đã ở trạng thái 1.

9. Tổng hợp thiết bị số một đầu vào x và một đầu ra y, hoạt động của thiết bị đảm bảo sao cho đầu ra y theo sát được mọi biến đổi lần thứ hai của cặp biến đổi đầu vào 1 - 0 - 1 như ở hình 13.35.



Hình 13.35

10. Tổng hợp thiết bị số mà hoạt động cho trên hình 13.36a, b, c dùng các phần tử nhỏ không được nhép sau đây:

- a) 2 trigơ D; 2 trigơ RS; 2 trigơ JK; 1 trigơ T và 1 trigơ RS đối với hình 13.36a.
 b) 2 trigơ T; 2 trigơ RS; 2 trigơ JK; 1 trigơ RS và 1 trigơ JK với hình 13.36b.
 c) 3 trigơ; 3 trigơ RS; 3 trigơ JK; 1 trigơ T, 1 trigơ RS và 1 trigơ JK với hình 13.36c.

X_1X_2				
A	00	01	10	11
1	(1),0	(1),1	3,-	2,-
2	1,-	(2),0	4,-	(2),0
3	1,-	1,-	(3),1	(3),1
4	1,-	2,-	(4),0	3,-

a)

X_1X_2				
A	00	01	10	11
1	(1),0	2,-	1,01	4,-
2	1,-	(2),10	2,00	3,-
3	4,-	(3),10	(3),10	(3),10
4	4,01	3,-	3,-	(4),10

b)

X_1X_2				
A	00	01	10	11
1	(1),1	3,-	2,-	(1),1
2	(2),0	4,-	(2),0	1,-
3	1,-	(3),0	5,-	(3),0
4	2,-	(4),1	6,-	(4),1
5	(5),1	6,-	(5),1	3,-
6	1,-	(6),0	(6),1	1,-

c)

Hình 13.36

11. Tổng hợp ôtomát cho dưới dạng bảng chuyển đổi trạng thái và bảng ra ở hình 13.37a, b dùng trigơ JK hoặc RS. Phần mạch tổ hợp thực hiện bằng các loại mạch sau:

- a) VA, KH, HOAC
 b) VA-KH
 c) HOAC-KH.

X_1X_2			
A	00	01	10
1	(1),0	(1),1	2
2	1	(2),1	(2),0
3	(3),1	2	(3),0

a)

Y	X_1X_2				
	A	00	01	10	11
1	1	1	2	-	3
0	2	1	2	4	-
1	3	1	-	4	3
0	4	-	2	4	-

b)

Hình 13.37

Chương 14

TỔNG HỢP CÁC THIẾT BỊ SỐ THÔNG DỤNG

Các thiết bị số thông dụng loại có nhớ như các mạch cộng, các thiết bị so sánh, các mạch đếm, các thanh ghi hay thanh ghi dịch, các mạch tạo từ hay các thiết bị thực hiện các chức năng đa dạng khác như các bộ nhớ, các bộ vi xử lý đã được sản xuất hàng loạt dưới dạng các vi mạch số từ cỡ trung bình đến cỡ lớn. Tuy vậy, nghiên cứu tổng hợp các loại thiết bị này vẫn có nhiều ý nghĩa vì:

- Trước hết nó giúp ta hiểu một cách cặn kẽ nguyên lý hoạt động và thiết kế, cấu tạo của chúng, giúp ta khai thác, sử dụng chúng được tốt hơn.

- Trên cơ sở nắm vững nguyên lý thiết kế các vi mạch cỡ nhỏ và trung bình, ta dễ dàng vận dụng để thiết kế các thiết bị số khác phức tạp hơn và đa dạng hơn.

Vì vậy dưới đây ta sẽ xem xét việc tổng hợp một số thiết bị số có nhớ thuộc loại vi mạch cỡ trung bình.

14.1. TỔNG HỢP MẠCH CỘNG

Trong phần mạch tổ hợp chúng ta đã trình bày phương pháp thiết kế mạch cộng không có nhớ. Trong phần này ta sẽ tổng hợp mạch cộng có nhớ hoạt động theo nguyên tắc các bit đầu vào của hai số hạng tác động liên tiếp tuần tự từ bit thấp tới bit cao. Một mạch hoạt động như vậy là một ô tômát hữu hạn, cho phép ta dùng lý thuyết của ô tômát hữu hạn để thực hiện việc tổng hợp.

Giả sử cần cộng hai số nhị phân có n bit A và B :

$$A = (a_n \ a_{n-1} \ \dots \ a_1)$$

$$B = (b_n \ b_{n-1} \ \dots \ b_1)$$

Với nguyên tắc cộng lần lượt các bit từ thấp tới bit cao, mạch cần có hai đầu vào và mỗi lần thực hiện phép cộng đối với một bit nào đó, cần chú ý tới số nhớ từ phép cộng của bit trước nó.

Thuật toán cộng bit thứ i , $i = 1, 2, 3, \dots, n$ dưới dạng tổng quát có dạng như sau:

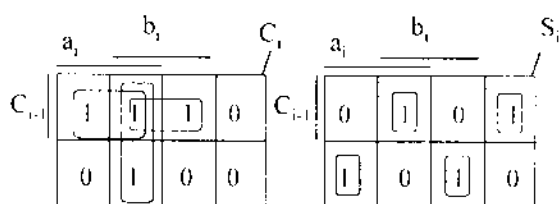
$$\left. \begin{aligned} C_i &= F_1(a_i, b_i, C_{i-1}) \\ S_i &= F_2(a_i, b_i, C_{i-1}) \end{aligned} \right\} \quad (14.1)$$

trong đó a_i và b_i là hai bit thứ i của hai số A và B ; B_i là giá trị của phép cộng; C_i là số nhớ của phép cộng; C_{i-1} là số nhớ từ phép cộng của hai bit trước a_{i-1} và b_{i-1} .

Hệ hai phương trình (14.1) là các phương trình mô tả chức năng của một ô tômát hữu hạn, trong đó i đóng vai trò như một biến thời gian. C_i là biến của trạng thái trong; S_i là biến ra. Như vậy, theo quan điểm của ô tômát, ô tômát này có hai trạng thái trong, với các giá trị có thể có là: $C_i = 0$ và $C_i = 1$ và có hai trạng thái ra: $S_i = 0$ và $S_i = 1$. Bảng chuyển đổi trạng thái và bảng ra của thiết bị cộng này có thể mô tả như trên hình 14.1a.

$a_i b_i$ C_{i-1}	00	01	10	11
0	0,0	0,1	0,1	1,0
1	0,1	1,0	1,0	1,1

a)



b)

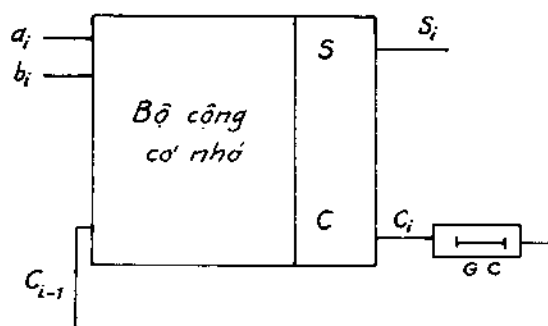
Hình 14.1. a) Bảng chuyển đổi trạng thái và bảng ra của thiết bị cộng;
b) Biểu diễn C_i và S_i lên bảng kế cận

Trên cơ sở đó có thể rút ra các hàm C_i và S_i sau khi đã biểu diễn chúng lên bảng kế cận ở hình 14.1b như sau:

$$C_i = a_i b_i + a_i C_{i-1} + b_i C_{i-1}$$

$$S_i = a_i b_i C_{i-1} + \bar{a}_i \bar{b}_i C_{i-1} + a_i \bar{b}_i \bar{C}_{i-1} + \bar{a}_i b_i \bar{C}_{i-1}$$

Mạch thực hiện cho trên hình 14.2, trong đó để tạo tín hiệu C_{i-1} ta dùng bộ giữ chậm một nhịp GC.



Hình 14.2. Mô hình thực hiện mạch cộng

Khi làm việc, trên đầu ra S_i lần lượt xuất hiện giá trị của tổng S_i của các phép cộng các bit thứ i và có tính tới số nhớ của phép cộng trước nó.

Các loại vi mạch thực hiện phép cộng có nhớ loại này gồm:

- Loại cộng hai từ 4 bit: SN54/74183; SN54/74283; SN54/7483AE; 9383PC...
- Loại cộng hai từ 1 bit: SN54/74183; ...

14.2. TỔNG HỢP MẠCH SO SÁNH CÓ NHỚ

14.2.1. Mạch so sánh hai số nhị phân n bit, xác định $A = B$, $A < B$ hay $A > B$

Giả sử có hai số nhị phân n bit A và B sau đây:

$$A = (a_n, a_{n-1}, a_{n-2}, a_1)$$

$$B = (b_n, b_{n-1}, b_{n-2}, b_1)$$

Để phân biệt ba trường hợp: $A > B$, $A < B$, $A = B$ ta cần hai biến y_1 và y_2 . Trong đó nếu:

$$\left. \begin{matrix} y_1 = 0 \\ y_2 = 0 \end{matrix} \right\} \longrightarrow A = B \quad \left. \begin{matrix} y_1 = 1 \\ y_2 = 0 \end{matrix} \right\} \longrightarrow A < B \quad \left. \begin{matrix} y_1 = 0 \\ y_2 = 1 \end{matrix} \right\} \longrightarrow A > B$$

Algôrit để so sánh là: So sánh từ các bit tương ứng của hai từ A và B từ các bit có trọng số cao tới các bit có trọng số thấp. Nếu $a_i = b_i$ thì so sánh bit tiếp theo cho tới a_1 , b_1 . Nếu tất cả các bit của hai từ đều bằng nhau thì kết luận $A = B$. Nếu trong quá trình so sánh phát hiện một cặp bit $a_i > b_i$ thì không cần so sánh tiếp và kết luận $A > B$; nếu $a_i < b_i$ thì $A < B$.

Từ các lập luận trên đây, ta có bảng chuyển đổi trạng thái và bảng ra như trên hình 14.3a. Đồ hình trạng thái cho trên hình 14.3b. Căn cứ vào bảng chuyển đổi trạng thái và bảng ra ta có thể rút ra các hàm ra y_1 và y_2 như ở hình 14.3c và mạch thực hiện các hàm đã rút ra cho trên hình 14.3d.

14.2.2. Mạch so sánh hai số nhị phân, phân biệt $A = B$ hay $A \neq B$

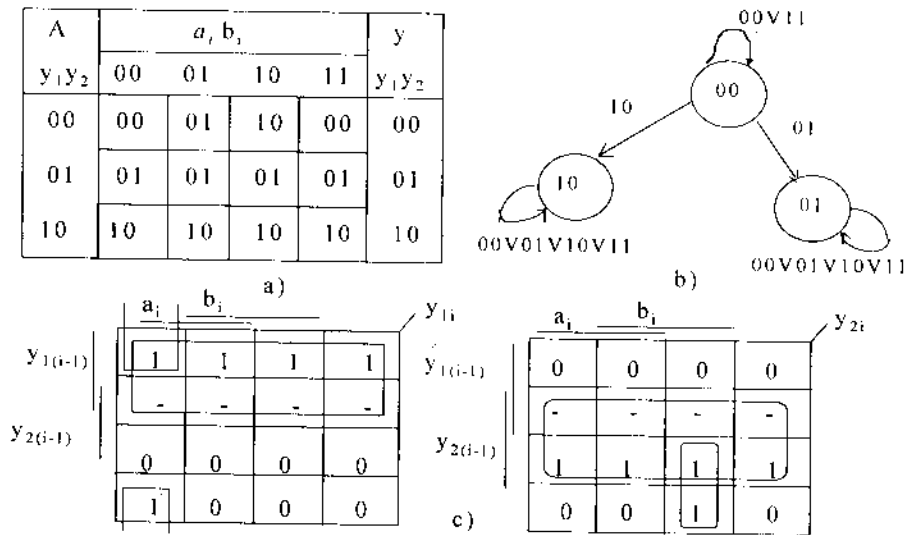
Trong trường hợp này chỉ cần một biến để phân biệt hai trường hợp trên đây. Biến này có thể đồng thời là biến trong của ô tômát. Do đó mạch thực hiện trong trường hợp này chỉ cần một phần tử nhớ. Thí dụ ta dùng phần tử JK với đầu ra là Q . Khi đó:

Nếu $Q = 0$ thì $A = B$.

Nếu $Q = 1$ thì $A \neq B$.

Thuật toán so sánh là so sánh từ hai bit cao của A và B đến các bit thấp của chúng. Nếu trong quá trình so sánh, một cặp bit nào đó khác nhau thì trạng thái của

$Q = 1$, chứng tỏ $A \neq B$ và dừng việc so sánh. Nếu tất cả các cặp bit so sánh đều bằng nhau thì $Q = 0$, chứng tỏ $A = B$.



$$y_{1i} = y_{1(i-1)} + a_i \bar{b}_i \bar{y}_{2(i-1)}$$

$$y_{2i} = y_{2(i-1)} + a_i b_i \bar{y}_{1(i-1)}$$

Hình 14.3. a) Bảng chuyển đổi trạng thái; b) Đồ hình trạng thái;
c) Biểu diễn hàm trên bảng kế cận; d) Mạch so sánh phân biệt $A = B$ và $A \neq B$

Bảng chuyển đổi trạng thái và bảng ra mô tả hoạt động của mạch so sánh loại này cho trên hình 14.4a. Từ đó rút ra các hàm kích thích cho trigger JK - hình 14.4b và mạch thực hiện chúng cho trên hình 14.4c.

Đây là loại ô tômát Moore, biến trong Q đồng thời là biến ra. Một mạch so sánh có nhớ như đã xem xét sẽ giữ lại kết quả so sánh ngay cả khi các bit của hai số A và B đầu vào đã hết. Đó là ưu điểm nổi bật của bộ so sánh loại có nhớ so với loại không có nhớ.

$a_i b_i$ Q	00	01	10	11
0	0	1	1	0
1	1	1	1	1

a)

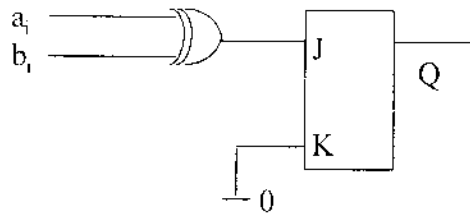
a_i	b_i	J
-	-	-
1	0	1

$$J = a_i \bar{b}_i + \bar{a}_i b_i = a_i \oplus b_i$$

a_i	b_i	K
0	0	0
-	-	-

$$K = 0$$

b)



c)

Hình 14.4. a) Bảng chuyển đổi trạng thái bảng ra; b) Xác định các hàm kích thích; c) Mạch thực hiện mạch so sánh xác định $A = B$ hay $A \neq B$

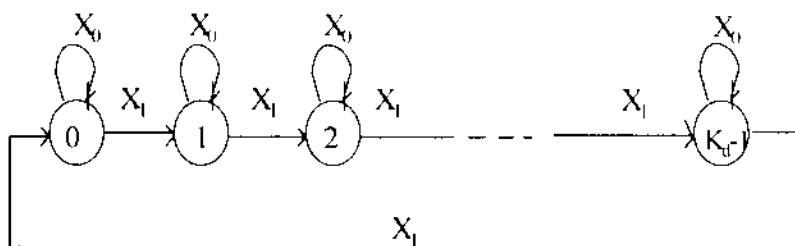
14.3. TỔNG HỢP CÁC MẠCH ĐẾM

Các mạch đếm là một trong các mạch có nhớ được sản xuất và sử dụng rộng rãi. Các vi mạch đếm có nhiều loại với các tính năng và đặc điểm sử dụng khác nhau. Mạch đếm thường được cấu tạo từ một chuỗi các mạch nhớ cơ bản - các trigơ được nối với nhau theo một quy luật nhất định để thực hiện việc đếm số xung.

Đặc trưng quan trọng nhất của bất kỳ mạch đếm nào cũng là cơ số đếm. Cơ số đếm thể hiện số xung tối đa mà mạch đếm đó có thể đếm. Khi số xung bằng cơ số đếm, mạch đếm lại trở về trạng thái ban đầu. Trạng thái ban đầu của mạch đếm thông thường là trạng thái không (không có xung), nhưng cũng có thể là trạng thái có số xung lớn nhất hoặc một trạng thái nào đó trong số các trạng thái mà nó có thể thể hiện.

Với quan điểm nhìn nhận mạch đếm như một ô tômát hữu hạn thì mạch đếm là một ô tômát có số trạng thái trong bảng cơ số đếm của nó và thường được xây dựng theo mô hình ô tômát Moore, trạng thái trong của ô tômát đồng thời cũng là trạng thái ra.

Một mạch đếm có một đầu vào (một biến vào) đếm. Các biến trong đồng thời là các biến ra, trong quá trình thực hiện đếm sẽ thay đổi giá trị của mình và tạo thành các trạng thái trong - các trạng thái của mạch đếm. Nếu ký hiệu X_0 là trạng thái vào không có xung đếm và X_1 là trạng thái vào có xung đếm thì một mạch đếm có cơ số đếm là K_d , tức có K_d trạng thái trong, đồ hình trạng thái khái quát của nó được trình bày như ở hình 14.5.



Hình 14.5. Mạch đếm có cơ số đếm bằng K .

14.3.1. Phân loại mạch đếm

Một số cách phân loại thường dùng như dưới đây.

a. Phân loại theo cơ số đếm

+ Mạch đếm nhị phân: Là các mạch đếm có cơ số đếm $K_d = 2^n$, $n = 1, 2, \dots$. Thí dụ các IC loại 7493, 74197 là các mạch đếm nhị phân có $K_d = 16$.

+ Mạch đếm thập phân: Là các mạch đếm có cơ số đếm $K_d = 10$. Thí dụ các IC loại 7490; 74176; 74196.

+ Các mạch đếm có cơ số đếm khác cơ số đếm của mạch đếm nhị phân và mạch đếm thập phân.

b. Phân loại theo cách làm việc

Chia ra thành mạch đếm đồng bộ và không đồng bộ hay còn gọi là các mạch đếm song song và các mạch đếm nối tiếp.

+ **Mạch đếm đồng bộ** (Mạch đếm song song) là mạch đếm mà khi chuyển từ trạng thái này sang trạng thái khác các biến trong cần thay đổi giá trị thì thay đổi một cách đồng thời. Để đạt được điều này, xung đếm thường được kích thích đồng thời vào các đầu vào CLK của các phần tử nhớ của mạch đếm. Thí dụ các mạch đếm loại IC 7493; 7490.

+ **Mạch đếm không đồng bộ** (Mạch đếm nối tiếp) là loại mạch đếm mà nói chung khi chuyển từ trạng thái này sang trạng thái khác, các biến trong cần thay đổi trạng

thái thì thay đổi một cách tuần tự (không đồng thời). Thí dụ loại IC 74196; 74197. Phương thức kích thích đếm của loại mạch đếm này thường là đưa xung đếm vào đầu vào thông tin hay đầu vào CLK của phần tử đếm (tức biến trong) có trọng số nhỏ nhất. Các phần tử nhớ được mắc theo kiểu nối tiếp và khi đó sự thay đổi trạng thái ra của một phần tử này có thể dẫn tới sự thay đổi trạng thái ra của phần tử tiếp theo nó.

c. Phân loại theo hướng đếm

Chia thành mạch đếm thuận và mạch đếm ngược.

+ *Mạch đếm thuận* là mạch đếm mà mỗi xung đếm lại làm tăng trạng thái của mạch đếm thêm một đơn vị. Trạng thái ban đầu của mạch đếm thuận luôn bằng 0.

+ *Mạch đếm ngược* là mạch đếm mà mỗi xung đếm lại làm giảm trạng thái trong của mạch đếm đi một đơn vị. Để thực hiện đếm ngược, ban đầu người ta nạp vào mạch đếm giá trị tối đa mà mạch đếm đó ghi nhận được.

Nói chung khái niệm đếm thuận hay đếm ngược chỉ là tương đối và có thể thiết kế hệ điều khiển để một mạch đếm vừa có thể đếm thuận, vừa có thể đếm ngược. Thí dụ về mạch đếm thuận nghịch như vậy là các IC đếm loại 74190 và 74193.

Đối với các mạch đếm cần chú ý thêm rằng, trong nhiều trường hợp người ta chỉ quan tâm tới cơ số đếm của mạch đếm như khi dùng mạch đếm với tư cách là một mạch chia thì việc mã hoá các trạng thái trong của mạch đếm là tùy ý. Sự thay đổi trạng thái trong của mạch đếm khi đó không nhất thiết theo tuần tự mà chỉ cần sau một số xung vào kích thích bằng cơ số đếm thì mạch đếm lại trở về trạng thái ban đầu.

14.3.2. Thiết kế mạch đếm

Mạch đếm là một ô tô-mát nên có thể dùng lý thuyết thiết kế ô tô-mát để thiết kế mạch đếm. Trong đó có thể dùng bất kỳ loại phần tử nhớ nào. Tuy vậy đối với các mạch đếm, để thuận lợi thường hay dùng các phần tử nhớ loại CT và CJK.

1) Các bước thiết kế

+ Xác định cơ số đếm K_d của mạch đếm; phương thức làm việc: song song hay nối tiếp; đếm ngược hay đếm xuôi...

+ Mô tả hoạt động của mạch đếm dưới dạng bảng chuyển đổi trạng thái và bảng ra hoặc đồ hình trạng thái. Trong đó số trạng thái trong của ô tô-mát đúng bằng cơ số đếm K_d của bộ đếm. Từ đó số trigơ cần thiết để thiết kế mạch đếm, đồng thời cũng là số biến trong, được xác định như sau:

$$p = \lceil \log_2 K_d \rceil$$

Dấu $\lceil \dots \rceil$ dùng để chỉ số nguyên nhỏ nhất lớn hơn hay bằng $\log_2 K_d$.

+ Tuỳ theo loại đếm ngược hay đếm xuôi hay chỉ cần quan tâm đến cơ số đếm mà thực hiện việc mã hoá các trạng thái trong của mạch đếm cho phù hợp.

+ Căn cứ vào bảng chuyển đổi trạng thái và bảng ra hay đồ hình trạng thái và bảng kích thích của các trigơ để rút ra các hàm kích thích cho các phần tử nhớ.

+ Thực hiện mạch.

2) Thí dụ về thiết kế các bộ đếm thông dụng

a) Bộ đếm nhị phân làm việc theo kiểu đồng bộ và đếm thuận; cơ số đếm $K_d = 16$

+ Xác định số trigơ cần thiết: $p = \lceil \log_2 K_d \rceil = 4$

+ Xây dựng bảng chuyển đổi trạng thái:

Thí dụ dùng 4 trigơ CJK, xếp theo thứ tự trọng số: DCBA, ta có bảng chuyển đổi trạng thái của bộ đếm theo số xung đếm như sau:

Số xung	DCBA	Số xung	DCBA	Số xung	DCBA	$a_c - a_m$	JK
0	0000	5	0101	11	1011	0 - 1	1 -
1	0001	6	0110	12	1100	0 - 0	0 -
2	0010	7	0111	13	1101	1 - 0	- 1
3	0011	8	1000	14	1110	1 - 1	- 0
4	0100	9	1001	15	1111		
		10	1010	16	0000		

Từ đây có thể đưa lên bảng kế cận và rút ra các hàm $J_A K_A$; $J_B K_B$; $J_C K_C$; $J_D K_D$ như ở hình 14.6.

		D C J _A K _A						D C J _B K _B			
B	A	1 -	- 1	- 1	1 -	B	A	0 -	- 0	- 0	- 0
1 -		- 1	- 1	- 1	1 -	- 1		- 1	- 1	- 1	- 1
1 -		- 1	- 1	- 1	1 -	1 -		1 -	1 -	1 -	1 -
1 -		- 1	- 1	- 1	1 -	0 -		0 -	0 -	0 -	0 -
		D C J _C K _C						D C J _D K _D			
0 -	A	- 0	- 0	- 0	0 -	B	A	0 -	0 -	0 -	0 -
1 -		- 1	- 1	- 1	1 -	0 -		- 1	1 -	0 -	
0 -		- 0	- 0	- 0	0 -	- 0		- 0	0 -	0 -	
0 -		- 0	- 0	- 0	0 -	- 0		- 0	0 -	0 -	

$J_A = K_A = 1$, $J_B = K_B = A$, $J_C = K_C = BA$, $J_D = K_D = ABC$.

Hình 14.6. Xác định các hàm kích thích của các phần tử nhớ

$a_3 a_2 a_1 a_0$	a_3	a_2	a_1	a_0	$J_0 K_0$	$J_1 K_1$
0 0 0 0 0	--	--	1-	1-	--	--
1 0 0 0 1	--	--	-1	-1	--	--
2 0 0 1 0	-1	--	-1	-1	0-	0-
3 0 0 1 1	1-	--	1-	1-	0-	0-
4 0 1 0 0	--	--	-0	-0	--	--
5 0 1 0 1	--	--	-1	-1	--	--
6 0 1 1 0	0-	--	-0	-0	--	--
7 0 1 1 1	0-	--	-0	-0	--	--
8 1 0 0 1	--	--	1-	0-	--	--
9 1 0 0 1	-1	--	0-	0-	--	--
10 0 0 0 0	0-	--	0-	0-	--	--

a)

$J_0 = K_0 = 1$

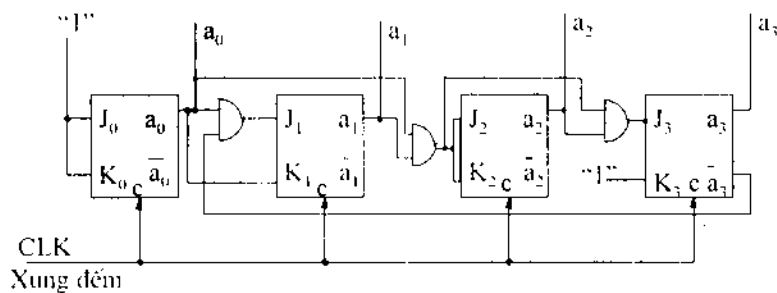
$J_2 = K_2 = a_1 a_0$

b)

$J_1 = a_0 \bar{a}_3, K_1 = \bar{a}_0$

$J_3 = a_2 a_1 a_0, K_3 = 1$

Hình 14.7. a) Bảng chuyển đổi trạng thái;
b) Xác định các hàm kích thích của các trigơ



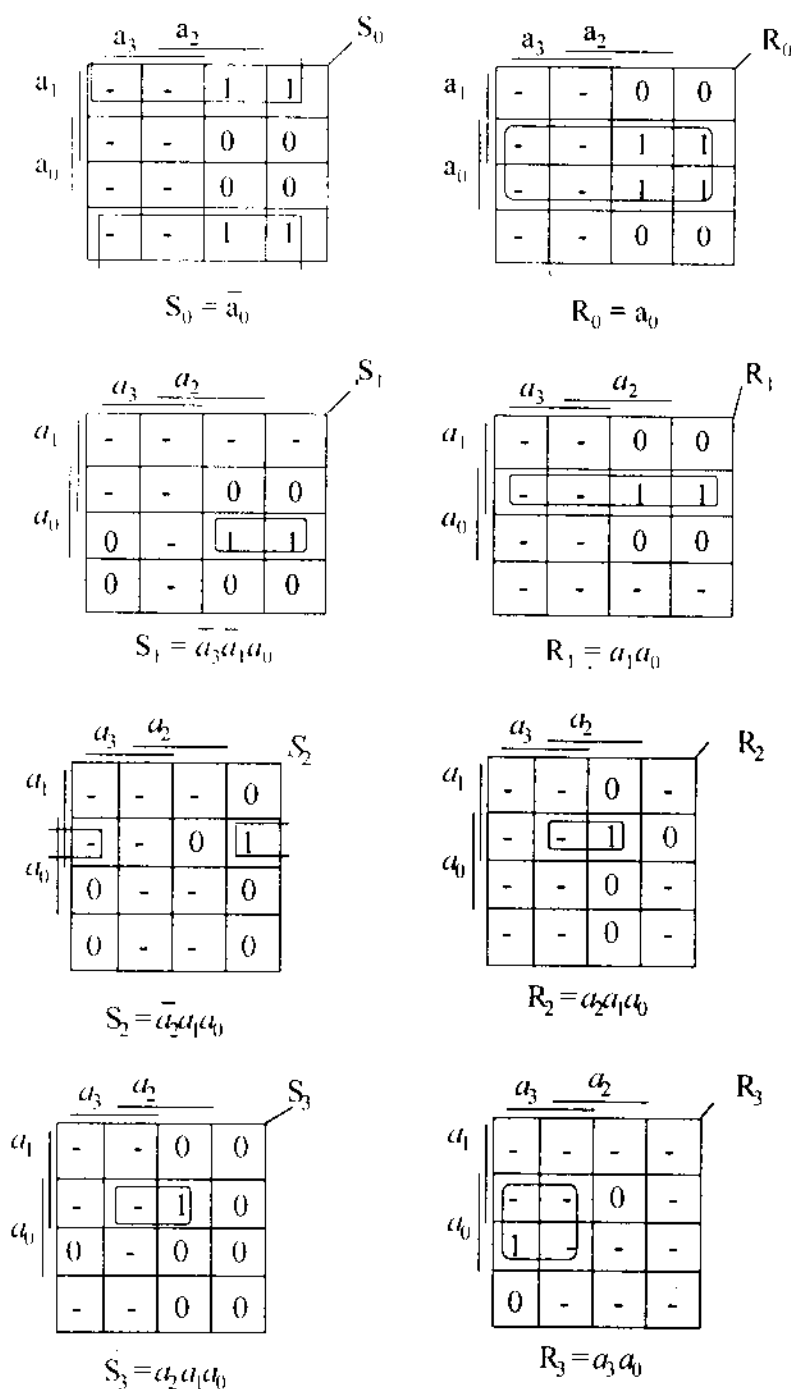
Hình 14.8. Mạch thực hiện bộ nhớ có $K_d = 10$, dùng trigơ JK

Cùng với mạch đếm trên đây, nhưng thiết kế trên cơ sở các trigơ CRS thì các hàm được rút ra như trên hình 14.9 và mạch thực hiện cho trên hình 14.10.

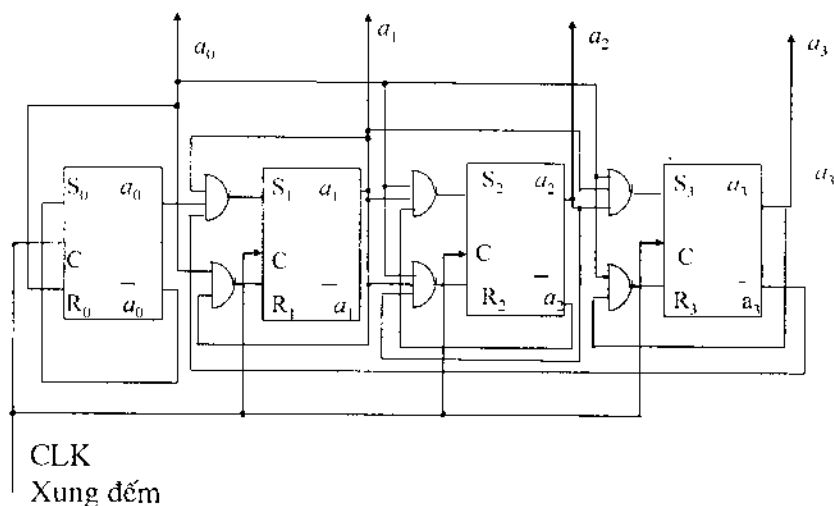
Mạch đếm ở hình 14.9, trong trường hợp dùng loại trigơ nhiều đầu vào J, nhiều đầu vào K, có thể biểu diễn như trên hình 14.11.

Cần chú ý rằng, trong thực tế người ta thường sản xuất các bộ đếm dưới dạng vi mạch có cơ số đếm nhất định. Để tiện cho việc mở rộng cơ số đếm trong việc nối ghép nhiều vi mạch đếm với nhau, đầu ra tín hiệu nhớ của mạch đếm cũng được thiết kế. Đó là tín hiệu tràn hay tín hiệu chuyển từ trạng thái cao nhất của mạch đếm sang

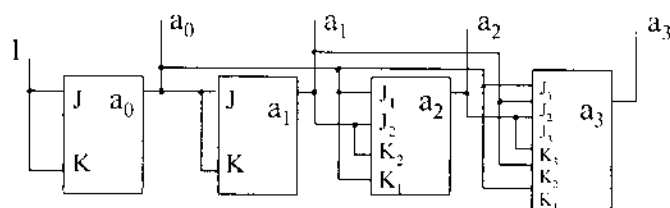
trạng thái thấp nhất của nó cũng vậy. Để có được tín hiệu tràn này, đối với mạch đếm có cơ số đếm bằng 16, xây dựng trên cơ sở các trigger CJK, trạng thái cao nhất là DCBA = 1111 và trạng thái thấp nhất là 0000. Mạch tạo tín hiệu tràn cho trên hình 14.12a và mạch nối ba mạch đếm như trên để có cơ số đếm bằng $16 \times 16 \times 16$ cho trên hình 14.12b.



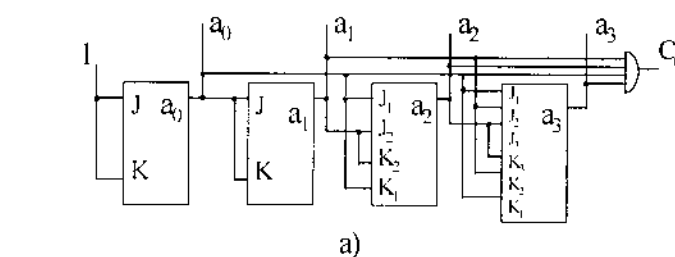
Hình 14.9. Xác định các hàm kích thích của các trigger CRS



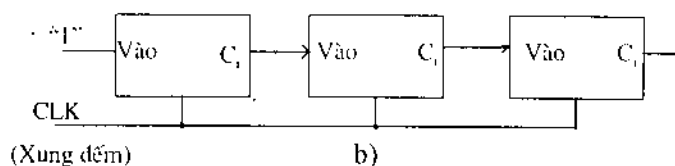
Hình 14.10. Mạch thực hiện bộ đếm cơ số 10 dùng trigơ CRS



Hình 14.11. Bộ đếm cơ số 10 thực hiện bằng các trigơ CJK nhiều đầu vào



a)

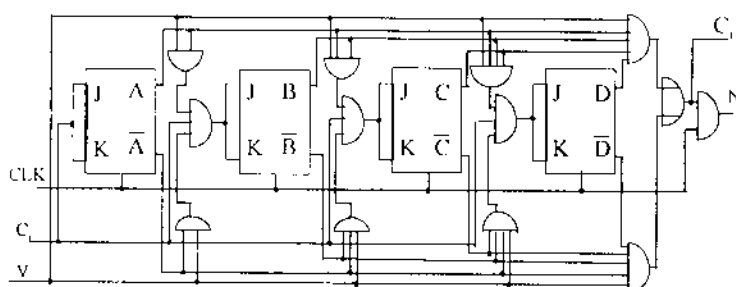


b)

Hình 14.12. a) Mạch đếm có tín hiệu tràn ; b) Nối ba bộ đếm để tạo mạch có cơ số đếm bằng tích của các cơ số đếm của ba bộ đếm thành phần

Đến đây chúng ta mới xây dựng các mạch đếm xuôi. Một cách hoàn toàn tương tự ta có thể thiết kế các mạch đếm ngược, tức mỗi lần có xung đếm nội dung của bộ đếm lại bớt đi một đơn vị. Song cũng dễ dàng nhận được mạch đếm ngược từ bộ đếm xuôi bằng cách nối các đầu vào JK của trigger phía sau vào đầu ra $\overline{A}$, $\overline{B}$, $\overline{C}$ của tầng trước.

Ngoài ra có thể thiết kế bộ đếm vừa có thể thực hiện đếm xuôi, vừa có thể thực hiện đếm ngược (mạch đếm xuôi ngược). Điều này có thể thực hiện được bằng cách dùng thêm các mạch điều khiển để khi thì các đầu ra A, B, C của tầng trước được nối vào các đầu vào J, K của các tầng sau, khi thì các đầu ra $\bar{A}$, $\bar{B}$, $\bar{C}$ nối vào các đầu vào này như ở hình 14.13.



Hình 14.13. Bộ đếm xuôi ngược

Trong mạch này tín hiệu V dùng làm tín hiệu điều khiển mạch đếm xuôi hay đếm ngược. Nếu $V = 1$, mạch thực hiện đếm xuôi; nếu $V = 0$, mạch thực hiện đếm ngược.

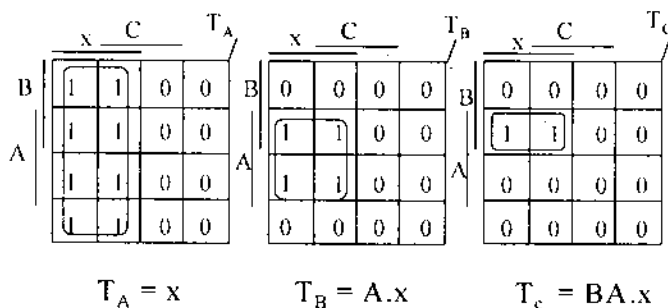
c) Mạch đếm nhị phân không đồng bộ, đếm xuôi với cơ số đếm $K_1 = 8$

Thiết kế mạch đếm làm việc theo kiểu nối tiếp (không đồng bộ) có thể theo nhiều cách.

Trước hết có thể thiết kế theo phương pháp thiết kế ô tô-mát thông thường, bằng cách căn cứ vào bảng chuyển đổi trạng thái, bảng ra đã mã hoá và bảng kích thích của các trigơ được sử dụng để tìm ra các hàm kích thích các trigơ và hàm ra của mạch (nếu có). Khi đó, với bài toán trên đây, số trigơ cần thiết bằng 4, ta có bảng chuyển đổi trạng thái như hình 14.14. Giả sử dùng trigơ loại CT. Từ đó có thể xác định và rút ra các hàm kích thích cho các trigơ T như trên hình 14.15.

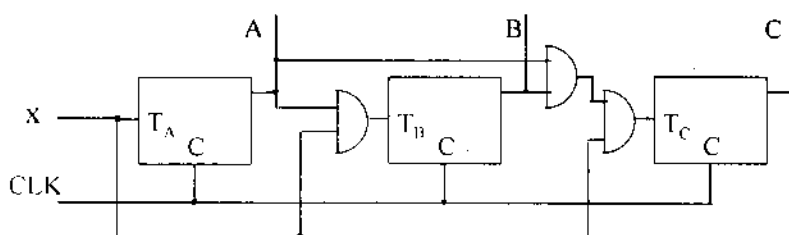
$\begin{matrix} x \\ a_1 a_2 a_0 \end{matrix}$	0	1
000	000	001
001	001	010
010	010	011
011	011	100
100	100	101
101	101	110
110	110	111
111	111	000

Hình 14.14. Bảng chuyển đổi trạng thái của bộ đếm $K_d = 8$



Hình 14.15. Xác định các hàm kích thích của các trigơ CT

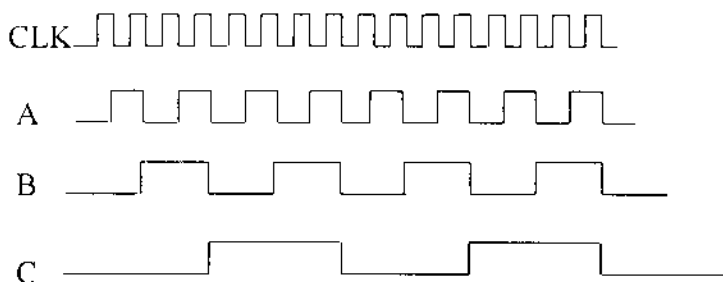
Mạch thực hiện cho trên hình 14.16.



Hình 14.16. Mạch đếm không đồng bộ có $K_d = 8$

Trong trường hợp này, để mạch đếm làm việc đúng, xung đếm x phải đồng bộ với xung CLK.

Khi thiết kế mạch đếm không đồng bộ có thể xuất phát từ nhận xét rằng, trong bảng chuyển đổi trạng thái của mạch đếm, trạng thái của biến A thay đổi nhanh gấp hai lần sự thay đổi của biến B và sự thay đổi của biến B lại nhanh gấp hai lần sự thay đổi của C. Nói cách khác, tín hiệu ra của trigơ A có tần số gấp hai lần tín hiệu ra của trigơ B. Đến lượt mình, tần số ra của trigơ B gấp hai lần tần số ra của trigơ C. Dưới dạng đồ thị thời gian điều này được vẽ trên hình 14.17.

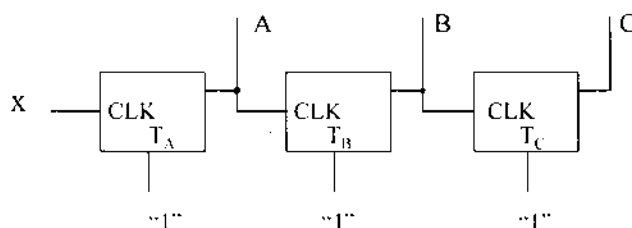


Hình 14.17. Đồ thị thời gian của các biến trong đồng thời là các biến ra của bộ đếm $K_d = 8$

Như vậy, có thể xây dựng mạch đếm loại này theo nguyên tắc chia tần, lấy đầu ra của A kích thích vào đầu vào CLK của trigơ B và đầu ra của B kích thích vào đầu vào CLK của trigơ C, trong khi các đầu vào tín hiệu của cả ba trigơ đều nối với logic "1"; đầu vào đếm xung được đưa vào đầu CLK của trigơ A như trên hình 14.18.

Dưới đây là một số loại IC đếm thường dùng:

- + Các mạch đếm nhị phân: SN 74160; SN 74162; MC 14160; MC 14162
- + Các mạch đếm thập phân: SN 74190; SN 7490; SN 74194; MC 10137; MC 14510
- + Đếm không đồng bộ 4 bit: SN 74161; SN 74163; MC 14161; MC 14163
- + Đếm xuôi ngược: SN 74191; MC 10136; MC 14516
- + Đếm xuôi ngược, đầu thuận nghịch riêng: SN 74193.



Hình 14.18. Một cách xây dựng bộ đếm không đồng bộ

14.4. TỔNG HỢP CÁC MẠCH GHI

14.4.1. Khái niệm chung

Các mạch ghi dùng lưu giữ thông tin dưới dạng số nhị phân và thực hiện các thuật toán dịch trái hoặc dịch phải đối với thông tin đã được ghi.

Các mạch ghi thông thường được xây dựng trên cơ sở các trigơ loại D hay các trigơ loại khác nhưng được mắc theo dạng loại trigơ D, vì đối với mạch ghi, một trong các yêu cầu quan trọng nhất là thông tin ra phải chính là thông tin đưa vào.

Để ghi nhận n bit thông tin, ta cần n trigơ. Các trigơ này được nối ghép với nhau sao cho việc đưa thông tin vào và lấy thông tin ra được thuận lợi và theo những dự kiến trước. Những dự kiến trước này đối với các mạch ghi cũng chính là các đặc trưng phân biệt các mạch ghi với nhau. Những đặc trưng quan trọng nhất là:

- Cách đưa thông tin vào và lấy thông tin ra;
- Khả năng dịch thông tin sang phải, sang trái hay cả hai trường hợp.

Từ đó có phân loại các mạch ghi.

14.4.2. Phân loại các mạch ghi

a) Theo cách đưa thông tin vào và lấy thông tin ra, có các loại sau:

- Vào nối tiếp, ra nối tiếp: Thông tin ghi vào mạch ghi được thực hiện lần lượt từng bit một. Lấy thông tin ra cũng lấy lần lượt từng bit một.
- Vào nối tiếp ra song song: Thông tin đưa vào từng bit một, sau khi đưa hết thông tin vào chúng được lấy ra đồng thời.
- Vào song song ra nối tiếp: Tất cả các bit thông tin được đưa vào mạch ghi một cách đồng thời, nhưng lấy thông tin ra theo cách lấy lần lượt từng bit một.
- Vào song song ra song song: Các bit thông tin cần ghi được đưa vào đồng thời và khi lấy ra cũng được lấy ra một cách đồng thời.

Chú ý rằng, một mạch ghi có thể đồng thời kết hợp các dạng đưa thông tin vào, lấy thông tin ra một cách linh hoạt.

b) Theo khả năng dịch, chia ra:

- Mạch ghi dịch phải
- Mạch ghi dịch trái
- Mạch ghi dịch trái, phải.

c) Theo chức năng, chia ra:

- Các mạch ghi đơn thuần: Chỉ cho phép ghi vào và lấy thông tin ra như một nơi chứa thông tin.
- Các mạch ghi dịch: Không những cho ghi thông tin mà còn có khả năng dịch thông tin sang trái hoặc sang phải theo từng bit một.
- Các thanh ghi chốt: Thường là loại vào, ra song song, nhưng đầu ra có ba trạng thái, thường dùng làm các phần tử nối ghép, thí dụ, giữa bộ vi xử lý và các thiết bị ngoại vi trên đường truyền thông tin.

14.4.3. Tổng hợp các mạch ghi

Để minh họa phương pháp thiết kế các mạch ghi, ta xét một số thí dụ sau.

a) Mạch ghi vào nối tiếp

Thí dụ thiết kế mạch ghi vào nối tiếp, bốn bit, có khả năng dịch trái, dịch phải, dùng trigơ CD.

Nếu ta ký hiệu tín hiệu vào x là Z_0 ; các tín hiệu ra của các trigơ từ 1 đến 4 là Z_1, Z_2, Z_3 và Z_4 ; tín hiệu tạo dịch trái là y_t ; tín hiệu tạo dịch phải đồng thời là tín hiệu ghi vào nối tiếp là y_g , được đồng bộ với tín hiệu nhịp bên ngoài là C . Như vậy, nguyên tắc hoạt động của mạch ghi vào nối tiếp, bốn bit, có khả năng dịch trái, dịch phải có thể mô tả bằng phương trình cho mỗi trigơ như sau:

$$Z_n = Z_{n-1} \cdot y_g + Z_{n+1} \cdot y_t, \text{ với } n = \overline{1, 4}$$

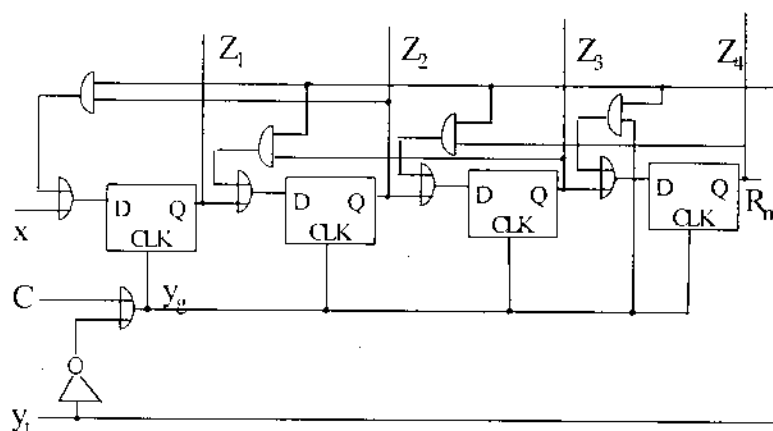
$$y_g = C \cdot \overline{y_t} + y_t$$

trong đó Z_n là tín hiệu vào của trigơ thứ n .

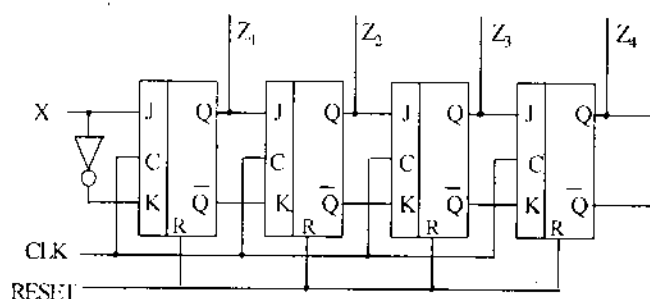
Như vậy, với nhịp C ($y_t = 0$) thiết bị sẽ ghi và dịch phải, với $y_t = 1$ số liệu đã ghi trong mạch ghi được dịch trái. Sơ đồ mạch ghi này cho trên hình 14.19.

Các mạch ghi như trên có thể lấy thông tin ra dưới dạng song song tại các đầu ra Z_1 đến Z_4 hoặc dưới dạng nối tiếp, mỗi lần một bit, theo xung nhịp, tại đầu ra nối tiếp R_n .

Nếu trigơ được sử dụng là loại trigơ M-S (Master – Slave) thì thông tin được đưa vào theo sườn xung nhịp C. Bộ ghi nối tiếp, dịch phải bốn bit dùng trigơ M-S CJK, trong trường hợp này cho trên hình 14.20.



Hình 14.19. Bộ ghi nối tiếp, dịch phải, dùng trigơ M-S



Hình 14.20. Bộ ghi 4 bit, dịch phải dùng trigơ M-S

Đối với mạch này, mỗi khi có xung nhịp, thông tin lại được dịch sang phải một bit. Đó cũng chính là động tác đưa thông tin vào. Tín hiệu RESET dùng để xoá tất cả các bit của bộ ghi về giá trị 0.

Nếu ta đưa chuỗi thông tin $D_1 D_2 D_3 D_4 D_5 D_6 D_7$ vào thì sau bảy xung nhịp, giá trị tại các đầu ra song song Z_1 đến Z_4 sẽ như trên hình 14.21.

Loại mạch sản xuất dưới dạng các IC:

- SN 74164
- MC 14021.

Nhịp	Z_1	Z_2	Z_3	Z_4
1	D_1	-	-	-
2	D_2	D_1	-	-
3	D_3	D_2	D_1	-
4	D_4	D_3	D_2	D_1
5	D_5	D_4	D_3	D_2
6	D_6	D_5	D_4	D_3
7	D_7	D_6	D_5	D_4

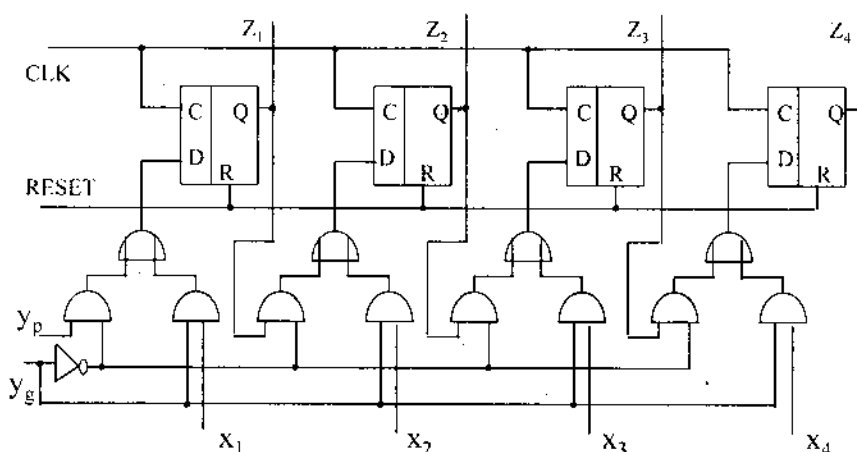
Hình 14.21. Trạng thái đầu ra các trigơ sau 7 xung nhịp

b) Mạch ghi bốn bit, vào song song, ra song song hoặc nối tiếp, dùng trigơ CD

Giả sử bốn bit số liệu đầu vào là x_1, x_2, x_3 và x_4 . Tín hiệu điều khiển ghi là y_g . Tín hiệu điều khiển dịch phải là y_p . Hàm mô tả đầu vào của trigơ CD đồng thời cũng là đầu ra của nó trong mối quan hệ với các đầu ra của các trigơ khác trong mạch và thực hiện chức năng ghi song song và dịch phải như sau:

$$Z_n = (X_n \cdot y_g + Z_{n-1} \cdot y_p \cdot \bar{y}_g) \cdot C$$

Mạch thực hiện cho trên hình 14.22.



Hình 14.22. Bộ ghi 4 bit, vào song song, ra song song/nối tiếp dùng trigơ CD

Trong đó nếu $y_g = 0$ và $y_p = 1$ thì khi có xung nhịp C, các số liệu đã được đưa vào trong mạch ghi sẽ dịch phải một bit với $Z_1 = y_p = (0/1)$. Nếu $y_g = 1$ thì thực hiện vào song song bốn bit từ x_1 đến x_4 .

Loại vi mạch :

Ghi dịch 4 bit: SN 74179; MC 14035

Ghi dịch 8 bit: SN 74199, vào, ra song song.

c) Mạch ghi vào song song, dịch phải, dịch trái

Để thực hiện chức năng vào song song và dịch phải, ta dùng tín hiệu y_g . Để dịch trái số liệu đã ghi trong mạch ghi ta dùng tín hiệu y_t .

Phương trình hàm nối ghép 4 trigơ của mạch ghi 4 bit như sau:

$$Z_n = x_n \cdot \bar{y}_g + Z_{n-1} \cdot \bar{y}_g \cdot y_p + Z_{n+1} \cdot y_t \cdot y_p$$

Mạch thực hiện cho trên hình 14.23.

Trong trường hợp này y_g và y_t cùng bằng 1 thì số liệu đã ghi được dịch trái. Để tránh lẫn lộn trong trường hợp này cần có $x_i = 0, i = \overline{1, 4}$.

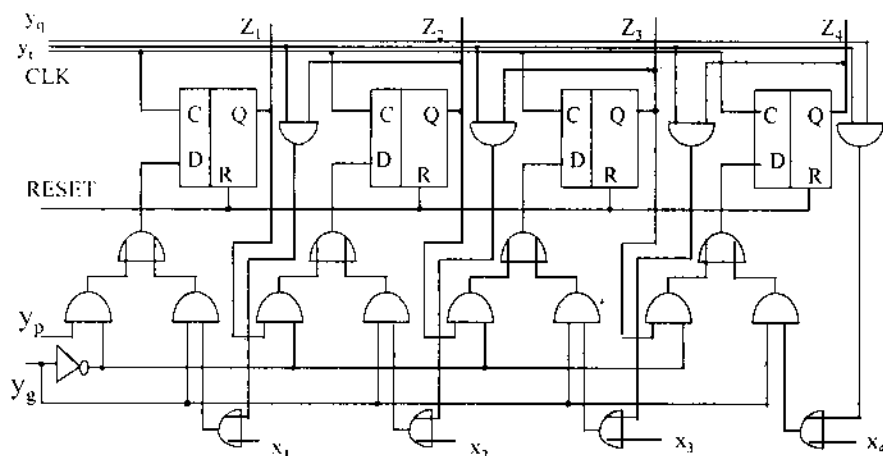
Nếu $y_t = 0$ và $y_g = 1$ thì thông tin x_1 đến x_4 được đưa song song vào các trigơ tương ứng của mạch ghi khi có xung nhịp $C = 1$.

Nếu $y_t = 0$, $y_g = 0$ thông tin đã được ghi được dịch phải, khi $C = 1$.

Loại vi mạch thông dụng:

+ Mạch ghi dịch phải, dịch trái 4 bít: SN 74194; MC 10141; MC 14194.

+ Mạch ghi dịch phải, dịch trái 8 bít: SN74198.



Hình 14.23. Bộ ghi vào song song, dịch phải, dịch trái

BÀI TẬP CHƯƠNG 14

- Thiết kế mạch cộng nối tiếp hai số nhị phân bốn bít A và B.
 - Dùng trigơ CD 7474 và họ IC VA, HOAC, KH.
 - Dùng trigơ CD 7475 và họ IC VA-KH, thí dụ 7400.
- Hãy thiết kế mạch so sánh hai số nhị phân A và B một bít phân biệt ba trường hợp $A \neq B$; $A = B$ dùng trigơ RS.
- Tổng hợp mạch đếm nhị phân cơ số 16, dùng:
 - Các trigơ CT.
 - Các trigơ CD.
 - Các trigơ CRS.
 - Các trigơ CJK có ba đầu vào J và ba đầu vào K.
- Tổng hợp mạch đếm thập phân, cơ số đếm bằng 10, dùng các trigơ CRS.
- Tổng hợp mạch đếm thập phân loại song song dùng trigơ CJK.
- Tổng hợp mạch ghi vào nối tiếp, dùng:
 - Các trigơ CT.
 - Trigơ CRS.
- Tổng hợp mạch ghi vào song song, dịch phải dùng các trigơ CJK.
- Tổng hợp mạch ghi vào song song, dịch phải, dịch trái, dùng trigơ CJK.

TÀI LIỆU THAM KHẢO

1. Nguyễn Duy Bảo. *Phân tích và tổng hợp thiết bị số*. Học viện Kỹ thuật quân sự, Hà Nội, 1987.
2. Nguyễn Duy Bảo. *Phương pháp GRAFCET thiết kế các hệ thống tự động điều khiển*. Học viện Kỹ thuật quân sự, Hà Nội, 2002.
3. Nguyễn Duy Bảo. *Проектиране на управляващите цифрови устройства с използване на еднородна среда*. Диссертация, София, 1979.
4. Баранов С. И. *Синтез микропрограмных автоматов*. М. Энергия, 1974.
5. Даковски Л. *Логически основи на ЦЕИМ*. Техника, София, 1978.
6. Глушков В. М. *Синтез конечных автоматов*. М. Физматгиз, 1962.
7. Закревский Н. В. *Алгоритмы синтеза дискретных автоматов*. М. Физматгиз, 1971.
8. Лазарев В. Г., Е. И. Пийль. *Синтез управляющих конечных автоматов*. М. Энергия, 1978.
9. Лазарев В. Г., Е. И. Пийль. *Синтез управляющих конечных автоматов*. М. Наука, 1964.
10. Майоров С. А. *Принципы организации цифровых машин*. Л. Мациностроение, 1974.
11. Милер Р. Е. *Теория переключательных схем. T1, T2*. М. Наука, 1984.
12. Поспелов Д. А. *Логические методы анализа и синтеза схем*. М., Энергия, 1974.
13. Almani A. E. *Electronic logic systems*. Prentice Hall, 1994.
14. Bredeson J. *Synthesis of multiple inputs change hazard-free combinational switching circuits without feedback*. International Journal of Electronics, 1975.
15. Courvoisier M. *An Asynchronous logic array for the realisation of logic systems with concurrency*. Electronics letters, 16 Fevrier, 1978, Vol. 14. N°4.
16. E. C. Tan and M. H. Ho. *Enhancing Matrix Method to detect logic Hazards*. Electronics, 1997, Vo. 82. N°4.
17. M. Botton. *Digital systems Design with programmable logic*. Addition - Wesley, 1990.
18. A. Miczo. *Digital logic testing and simulation*. Harper and row, 1986.
19. Michel G., C. Laugeau, B. Espiau. *Les automates programmables industriels*. Dunod, 1979.
20. David R. et al. *Du Crafcet aux reseaux de Petri*. Hermes, Paris, 1989.

205160

p phân tích và tổng hợp thi



69.000 VND

Giá : 69.000 đ