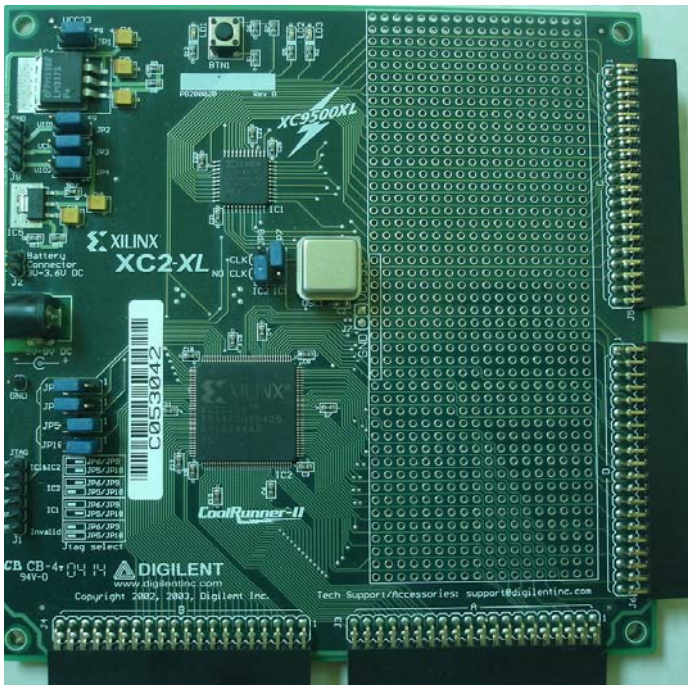
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM
KHOA ĐIỆN TỬ
BỘ MÔN VIỄN THÔNG

KỸ THUẬT PLD & ASIC

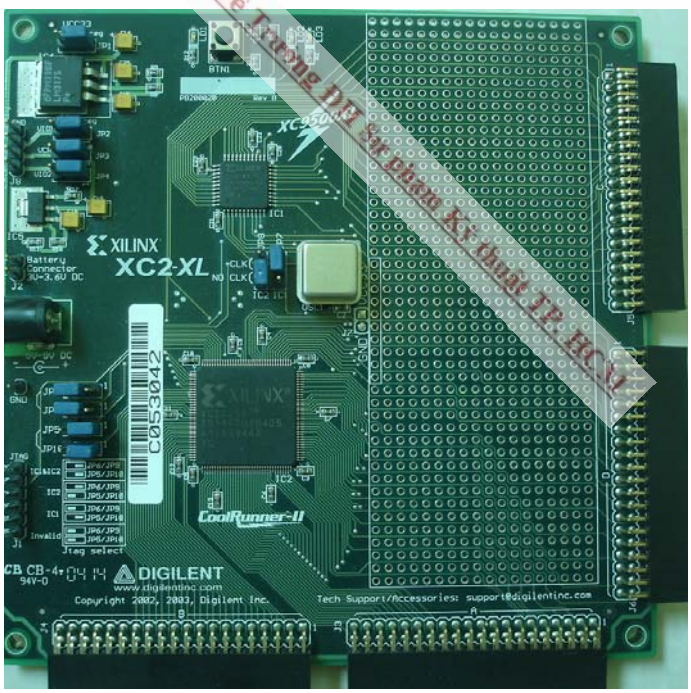


Biên soạn: Nguyễn Đình Phú

TP.HCM 2007

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM
KHOA ĐIỆN TỬ
BỘ MÔN VIỄN THÔNG

KỸ THUẬT PLD & ASIC



Biên soạn: Nguyễn Đình Phú

TP.HCM 2007

LỜI NÓI ĐẦU

Các hệ thống số lập trình ngày càng hiện diện trong nhiều thiết bị điện tử dân dụng cũng như trong các thiết bị điều khiển công nghiệp.

Ưu điểm của thiết bị số lập trình là làm cho mạch điện ngày càng nhỏ gọn do mật độ tích hợp cao, không mất nhiều thời gian cho việc kết nối và thử nghiệm so với IC rời, dễ thay đổi yêu cầu điều khiển của mạch, chiếm ít diện tích không gian, tốc độ hay tần số làm việc cao đáp ứng được các ứng dụng đòi hỏi về tốc độ hoặc xử lý khối lượng dữ liệu lớn.

Nội dung cuốn sách này được biên soạn gồm 4 chương nhằm phục vụ cho môn học 2 tín chỉ, trong đó chương 1 giới thiệu về các thiết bị số lập trình được, chương 2 trình bày ngôn ngữ VHDL dùng để lập trình cho hệ thống số, chương 3 trình bày cách lập trình cho các mạch điện tổ hợp, chương 4 trình bày cách lập trình cho các mạch điện tuần tự.

Nội dung trong cuốn sách nhằm trang bị các kiến thức cơ bản về kỹ thuật PLD và ASIC cho sinh viên ngành điện – điện tử.

Trong quá trình biên soạn có tham khảo nhiều tài liệu nên vẫn còn sai sót nên mong sự đóng góp xây dựng để bài giảng được hoàn thiện hơn xin hãy gởi về tác giả theo địa chỉ phu_nd@yahoo.com - xin chân thành cảm ơn.

MỤC LỤC

Chương 1

GIỚI THIỆU CÁC CẤU TRÚC LẬP TRÌNH ĐƯỢC

I. GIỚI THIỆU PLD:

Hai thành phần chính của thiết bị logic lập trình đơn giản SPLD (Simple Programmable Logic Device) là PAL và GAL. PAL tượng trưng cho logic mảng lập trình (**Programmable Array Logic**) và GAL tượng trưng cho logic mảng tổng quát (**Generic Array Logic**). Thường thì PAL chỉ lập trình 1 lần còn GAL thì cho phép lập trình lại, tuy nhiên có nhiều loại SPLD lập trình lại vẫn còn được gọi là PAL.

Thuật ngữ GAL là tên do hãng Lattice Semeconductor đặt và sau đó thì được cấp phép cho các nhà sản xuất khác.

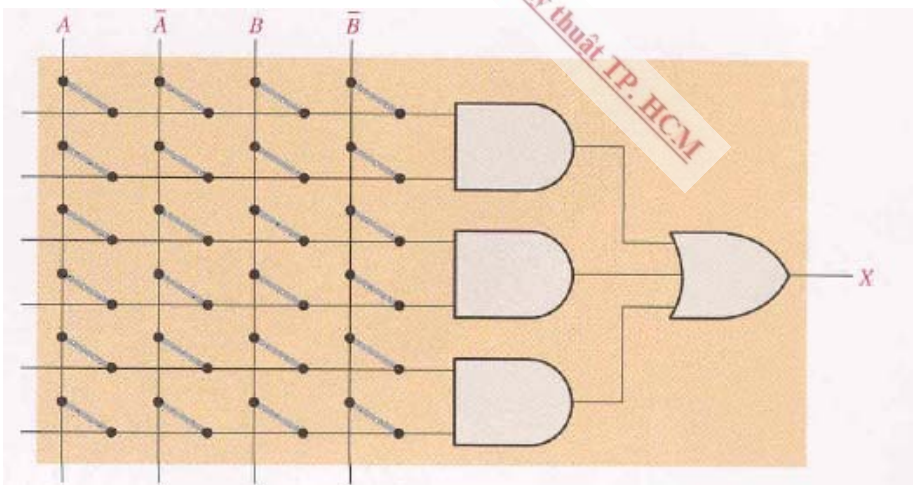
Cấu trúc cơ bản của PAL và GAL là mảng AND cho phép lập trình và mảng OR cố định tổ chức theo phương pháp tổng của các tích SOP (Sum-Of-Product). Với CPLD (Complex Programmable Logic Device) được tích hợp từ nhiều SPLD để có chức năng mạnh hơn cho các thiết kế phức tạp.

Trong phần này chúng ta sẽ khảo sát hoạt động của SPLD, phương pháp tổng của các tích được dùng trong PAL và GAL, giải thích được sơ đồ logic của PAL/GAL, mô tả macrocell cơ bản của PAL/GAL, khảo sát PAL16V8 và GAL22V10, mô tả CPLD cơ bản.

1. HOẠT ĐỘNG CỦA SPLD CƠ BẢN LÀ PAL

PAL chứa mảng cổng AND lập trình và được nối với mảng cổng OR cố định. Thường thì PAL dùng công nghệ xử lý cầu chì nên chỉ cho phép lập trình 1 lần OTP (One-time-Programmable).

Cấu trúc PAL cho phép thực hiện tất cả các hàm tổng của các tích với các biến đã được xác định. Cấu trúc của một PAL đơn giản được trình bày như hình 1-1 cho 2 biến ngõ vào và 1 biến ngõ ra:



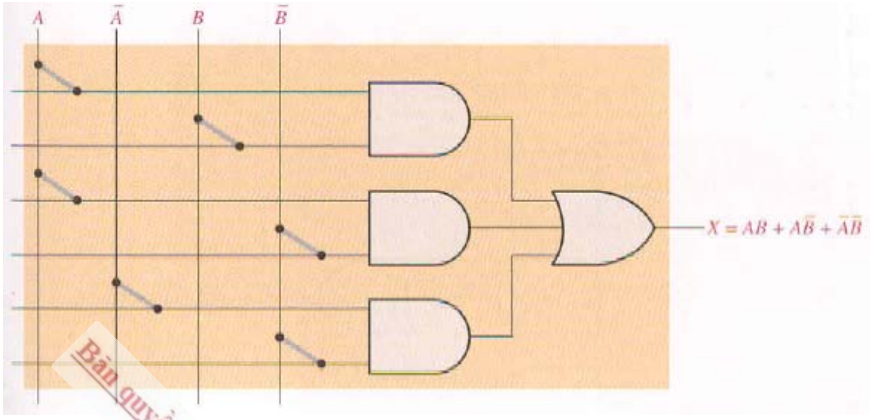
Hình 1-1. Cấu trúc của PAL.

Một mảng lập trình là một ma trận các dây dẫn gồm các hàng và các cột và chúng có thể lập trình để nối với nhau tại điểm giao nhau. Mỗi điểm nối lập trình có cấu tạo là cầu chì đối với loại PAL và được gọi là một tế bào ***cell***. Mỗi hàng có thể nối với một ngõ vào của cổng AND và mỗi cột là một biến ngõ vào hoặc biến phủ định. Bằng cách lập trình giữ nguyên cầu chì hay phá hỏng cầu chì thì có thể tạo ra bất kỳ hàm tổ hợp nào từ các biến ngõ vào để đưa

đến cổng AND tạo ra các thành phần tích mong muốn. Các cổng AND được kết nối với cổng OR để tạo nên các hàm ngõ ra tổng của các tích.

Ví dụ 1: Một PAL được lập trình như hình 1-2 để tạo ra thành phần AB, $A\overline{B}$ và $\overline{A}\overline{B}$. Trong hình 1-2 ta có thể nhìn thấy một số cấu chì bị phá hỏng và một số cầu chì còn nguyên để kết nối các biến ngõ vào với các ngõ vào của các cổng AND tạo ra hàm tích theo yêu cầu và sau cùng là hàm tổng của các tích:

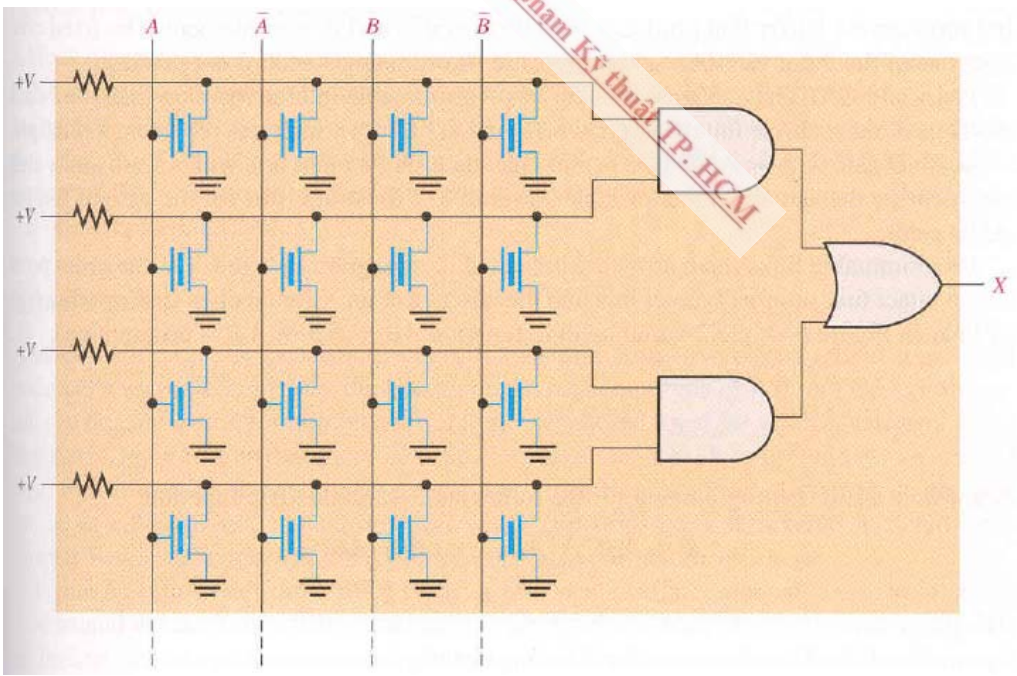
$$X = AB + A\overline{B} + \overline{A}\overline{B}$$



Hình 1-2. PAL sau khi lập trình để tạo hàm.

2. HOẠT ĐỘNG CỦA SPLD CƠ BẢN LÀ GAL

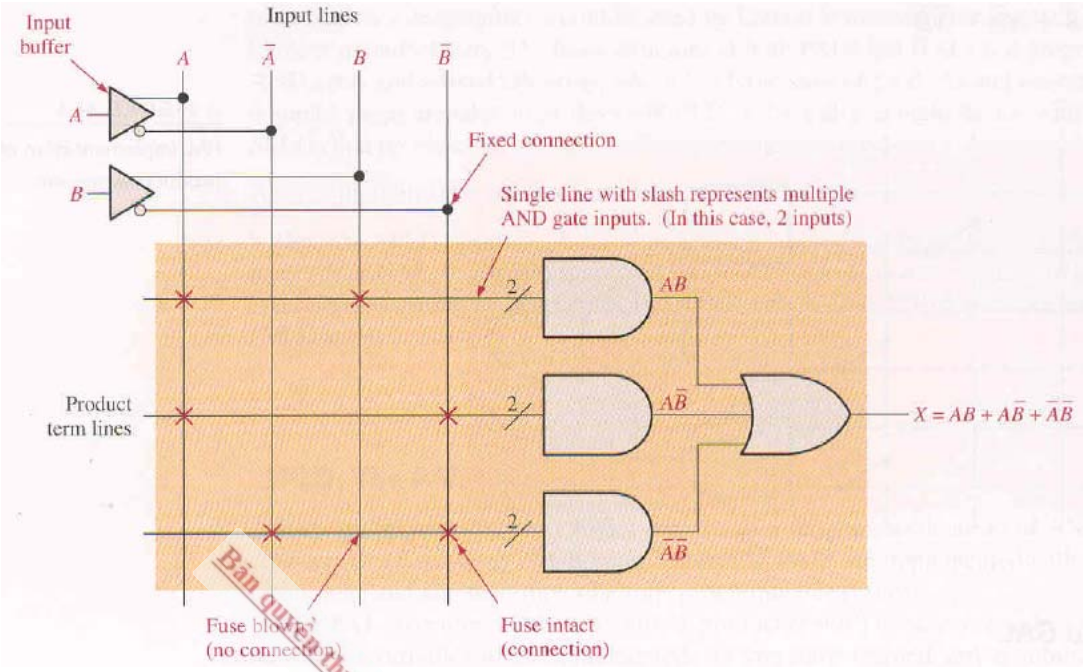
GAL về cơ bản chính là PAL có thể lập trình được, GAL có tổ chức AND/OR giống như PAL nhưng sự khác nhau cơ bản là GAL dùng công nghệ xử lý cho phép lập trình lại giống như EEPROM thay cho cầu chì được trình bày như hình 1-3.



Hình 1-3. Cấu trúc của GAL.

3. KÍ HIỆU ĐƠN GIẢN CHO SƠ ĐỒ CỦA PAL/GAL

Các thiết bị lập trình PAL và GAL có các cổng logic AND và OR và thêm một số phần tử khác cùng với các biến ngõ vào và các biến phủ định. Hầu hết các PAL và GAL đều có sơ đồ kí hiệu đơn giản như hình 1-4:



Hình 1-4. Kí hiệu đơn giản cho PAL/GAL.

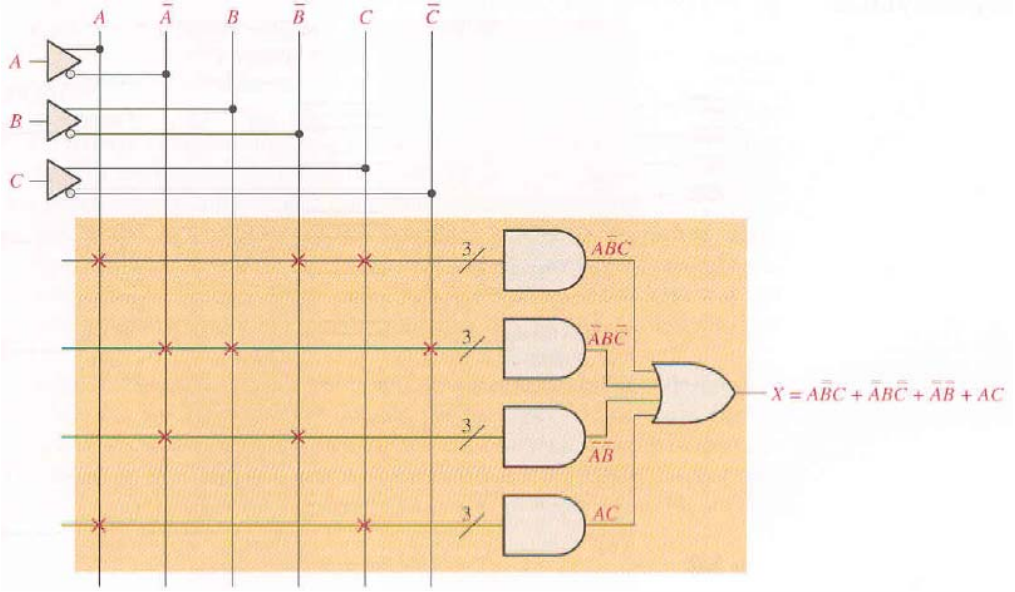
Các biến ngõ vào của PAL hoặc GAL thường có mạch đệm để ngăn chặn quá tải khi có quá nhiều cổng AND nối tới ngõ vào đó. Trong sơ đồ, khối đệm là khối tam giác vừa đệm tín hiệu ngõ vào và đảo tín hiệu để tạo ra biến phủ định của tín hiệu đó.

PAL và GAL đều có một lượng rất lớn các đường lập trình kết nối bên trong và mỗi cổng AND có nhiều ngõ vào. Thường thì trong sơ đồ mạch của PAL và GAL thay ***cổng AND nhiều ngõ vào*** bằng ***cổng AND chỉ có một đường ngõ vào*** cho gọn nhưng trên đó có ghi số lượng ngõ vào thực cho cổng AND đó. Trong hình 1-4 thì mỗi cổng AND đều có 2 ngõ vào.

Điểm nối lập trình nằm trong ma trận được xác định bằng dấu × nằm trên các đường giao nhau và cầu chì sẽ được giữ nguyên, còn các điểm không có đánh dấu × thì cầu chì sẽ bị phá hỏng. Hình 1-4 của ví dụ ở trên được lập trình để tạo ra hàm $X = AB + A\overline{B} + \overline{A}\overline{B}$.

Ví dụ 1-2: Hãy vẽ sơ đồ mạch cho một PAL đã lập trình để tạo ra hàm có 3 biến ngõ vào như sau: $X = A\overline{B}C + \overline{A}BC + \overline{A}\overline{B} + AC$

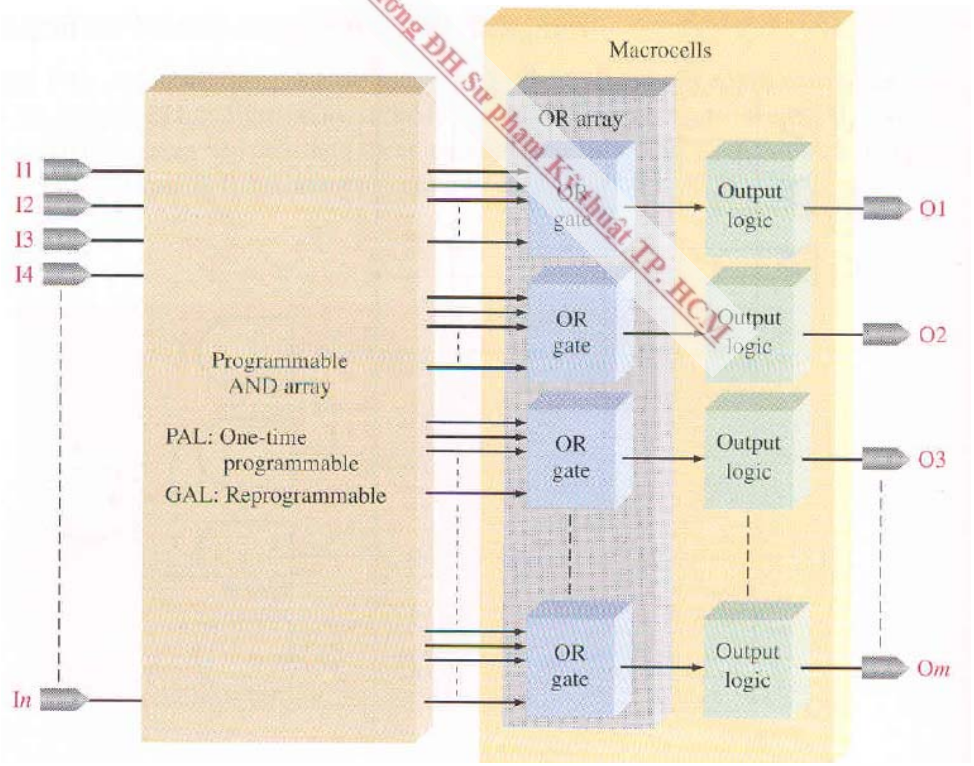
Giải: Sơ đồ mạch của PAL như hình 1-5:



Hình 1-5. Hình cho ví dụ 1-1.

4. SƠ ĐỒ KHỐI TỔNG QUÁT CỦA PAL/GAL

Sơ đồ khối của PAL hoặc GAL được trình bày ở hình 1-6. Nên nhớ rằng sự khác nhau cơ bản là GAL có mảng cho phép lập trình lại còn PAL thì chỉ lập trình một lần. Các ngõ ra của mảng cổng AND lập trình được đưa đến các cổng OR cố định đã được kết nối để tạo các hàm logic ngõ ra. Cổng OR kết hợp với hàm logic ngõ ra thường được gọi là ***macrocell***.



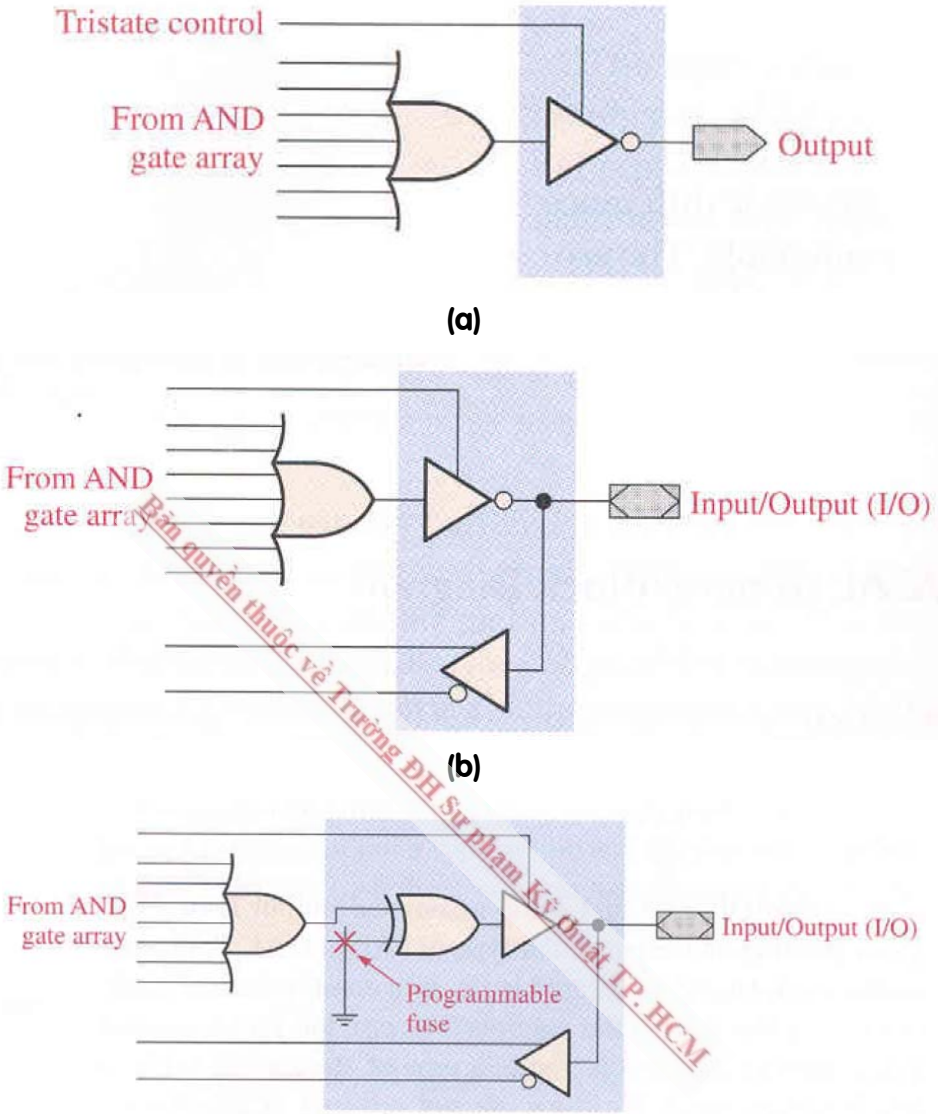
Hình 1-6. Sơ đồ khối của PAL/GAL.

5. MACROCELL

Một ***macrocell*** gồm một cổng OR và các hàm logic ngõ ra kết hợp. Mức độ phức tạp của ***macrocell*** tuỳ thuộc vào thiết bị cụ thể PAL hoặc GAL. Một ***macrocell*** có thể được định cấu hình cho một hàm tổ hợp, hàm thanh ghi hoặc cho cả hai.

Hàm thanh ghi có liên quan đến flip flop chính vì thế trong ***macrocell*** cũng có flip flop để tạo ra các hàm tuần tự.

Hình 1-7 trình bày 3 loại ***macrocell*** cơ bản với các hàm tổ hợp.



Hình 1-7. Sơ đồ mạch các Macrocell.

Hình 1-7a trình bày một ***macrocell*** đơn giản với một cổng OR và một cổng đảo ba trạng thái. Ngõ ra của cổng đảo ba trạng thái có thể hoạt động tạo ra mức HIGH, mức LOW và trạng thái tổng trở cao xem như hở mạch.

Hình 1-7b trình bày một ***macrocell*** có thể hoạt động như ngõ vào hoặc ngõ ra. Khi ngõ vào được dùng như ngõ ra thì cổng đảo phải ở trạng thái tổng trở cao để hở mạch và tín hiệu từ bên ngoài đưa đến bộ đệm và kết nối với mảng cổng AND bên trong.

Hình 1-7c trình bày một ***macrocell*** có thể lập trình để có ngõ ra tích cực mức HIGH hoặc mức tích cực mức LOW và cũng có thể sử dụng như ngõ vào. Một ngõ vào của cổng XOR (ex-or) có thể được lập trình ở mức HIGH hoặc mức LOW. Khi lập trình ngõ vào cổng XOR ở mức HIGH thì tín hiệu ngõ ra của cổng OR sẽ bị đảo vì : $0 \oplus 1 = 1$ và $1 \oplus 1 = 0$. Tương tự khi lập trình

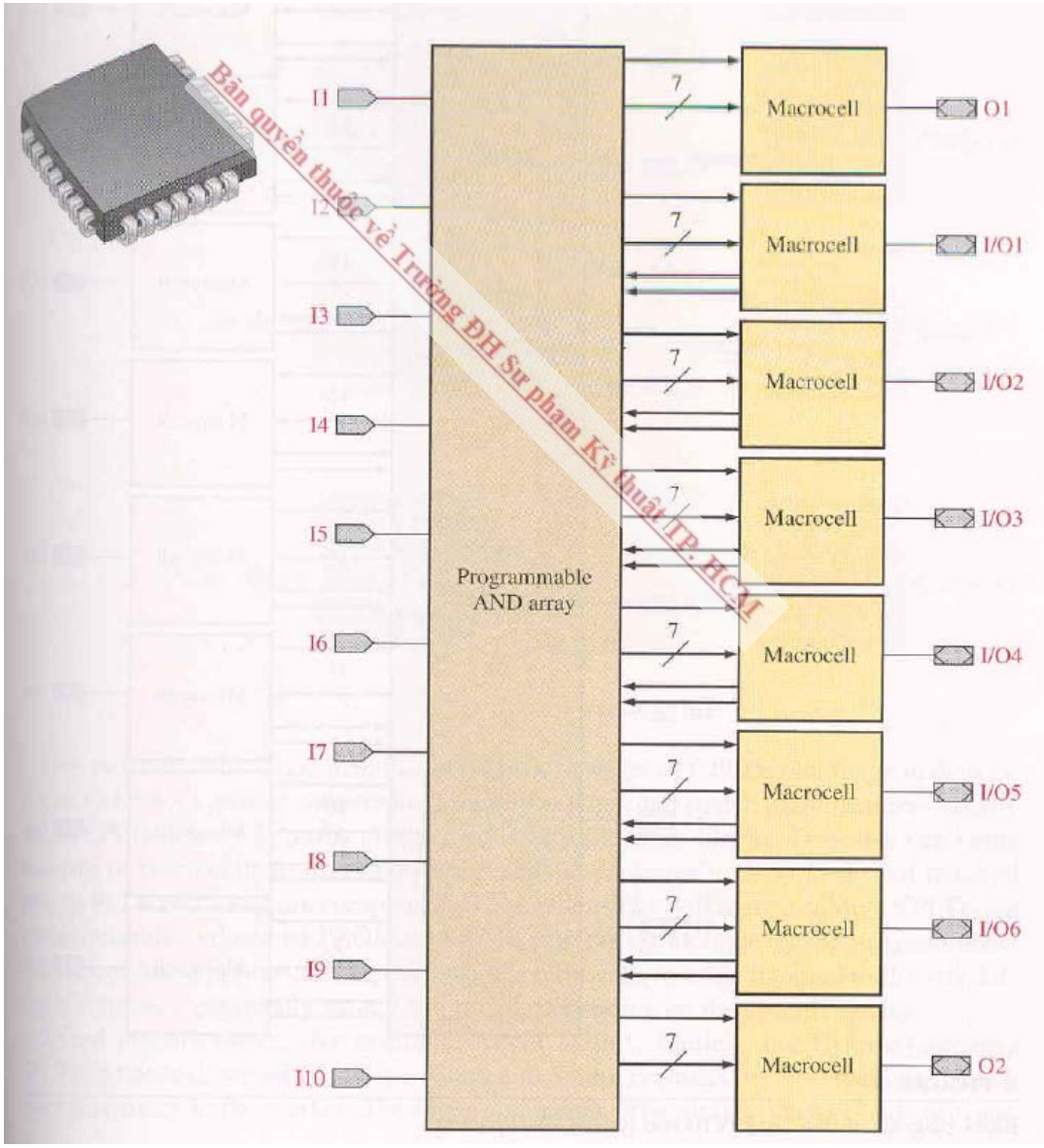
ngõ vào cổng XOR ở mức LOW thì tín hiệu ngõ ra cổng OR không bị đảo vì: $0 \oplus 0 = 0$ và $1 \oplus 0 = 1$.

6. CÁC SPLD THỰC TẾ

Thường thì hình dạng vỏ của SPLD có cấu hình chân nằm trong khoảng từ 20 đến 28 chân. Có 2 thành phần giúp chúng ta xác định PAL hoặc GAL một cách thích hợp cho các thiết kế logic đã cho là số lượng ngõ vào và ngõ ra cùng với số lượng cổng logic. Một vài thông số khác cần phải xem xét là tần số hoạt động cực đại, thời gian trễ và nguồn điện áp cung cấp.

Các nhà sản xuất Lattice, Actel, Atmel và Cypress là các công ty sản xuất SPLD.

Các loại PAL và GAL thường sử dụng là PAL16V8 và GAL22V10. Các mã số cho biết số lượng ngõ vào, số lượng ngõ ra và loại ngõ ra logic. Ví dụ: PAL16V8 sẽ cho biết thiết bị này có 16 ngõ vào, 8 ngõ ra và ngõ ra là biến (V: variable). Chữ H hoặc chữ L có nghĩa là ngõ ra tích cực mức HIGH hoặc mức LOW tương ứng. Sơ đồ khối của PAL16V8 và hình dạng vỏ được trình bày ở hình 1-8.

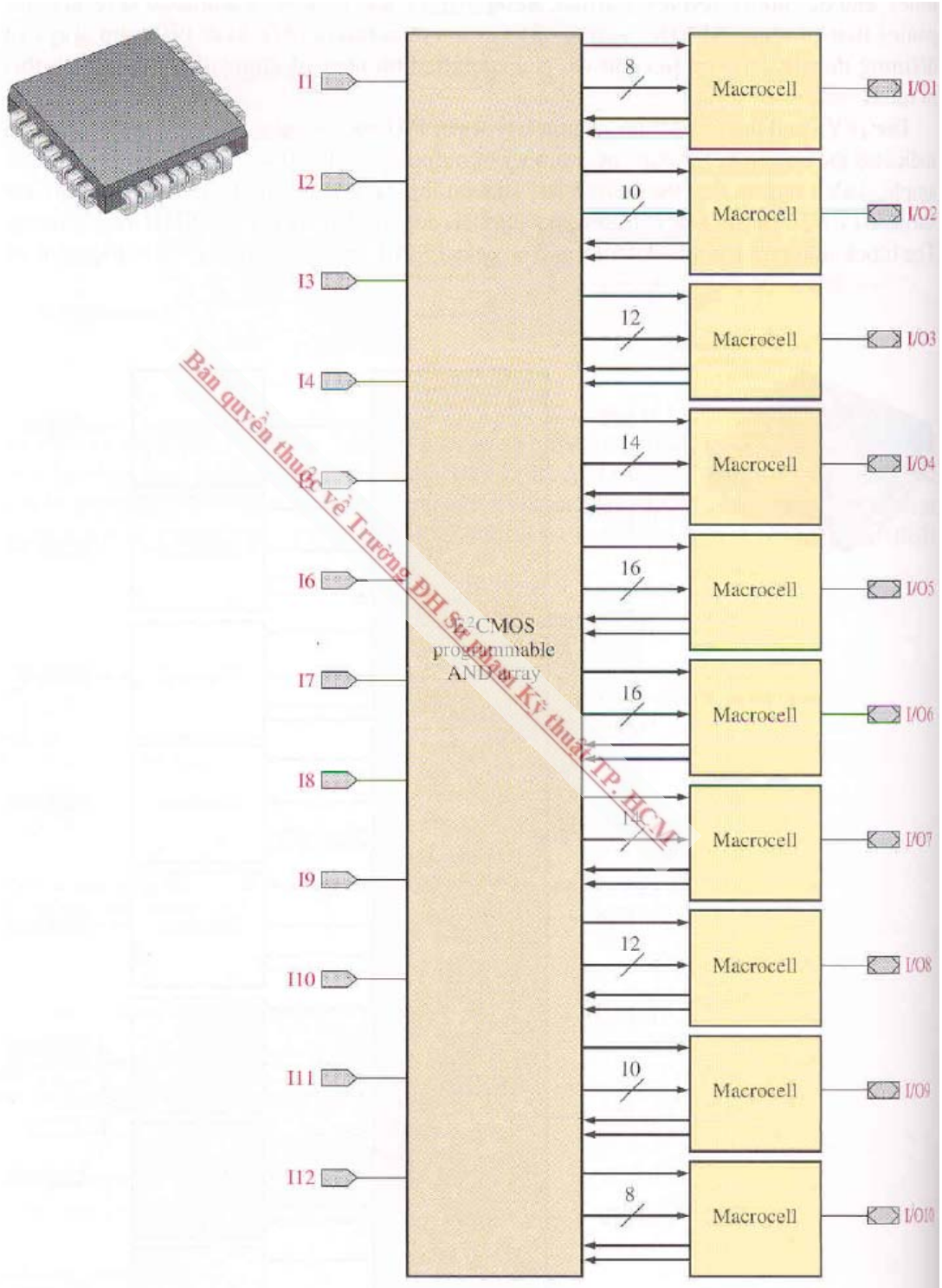


Hình 1-8. Sơ đồ khối và hình dạng vỏ của PAL16V8.

Mỗi ***macrocell*** có 8 ngõ vào lấy từ mảng cổng AND nên có thể có tới 8 thành phần tích cho mỗi ngõ ra. Có 10 ngõ vào kí hiệu là I, 2 ngõ ra kí hiệu là O và 6 chân có thể được dùng

như là ngõ vào hoặc ngõ ra và kí hiệu là I/O. Mỗi ngõ ra tích cực mức LOW. PAL16V8 có mật độ tích hợp khoảng 300 cổng.

Sơ đồ khối của GAL 22V10 và hình dạng vỏ như hình 1-9. GAL này có 12 ngõ vào và 10 chân có thể sử dụng như ngõ vào hoặc ngõ ra. Các ***macrocell*** có các ngõ vào kết nối với mảng cổng AND có thể thay đổi số lượng kết nối từ 8 đến 16. GAL 22V10 có mật độ tích hợp khoảng 500 cổng.



Hình 1-9. Sơ đồ khối và hình dạng vỏ của GAL22V10.

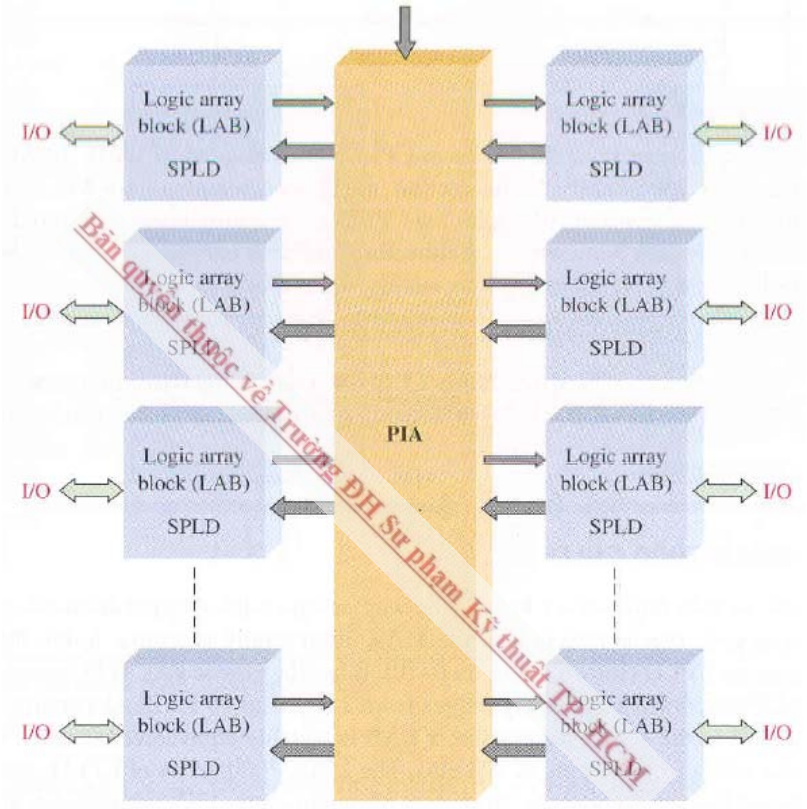
7. CÁC CPLD

Một CPLD chứa nhiều mảng SPLD với kết nối bên trong cho phép lập trình như hình 1-10.

Chúng ta xem mỗi mảng SPLD trong CPLD là một ***khối mảng logic* LAB** (Logic Array Block). Một tên khác đôi khi cũng được dùng là ***khối chức năng***, ***khối logic*** hoặc ***khối tổng quát***.

Các kết nối lập trình bên trong thường được gọi là PIA (**Programmable Interconnect Array**) nhưng một số nhà chế tạo như Xilinx dùng thuật ngữ AIM (**Advance Interconnect Matrix**) hoặc các thuật ngữ tương tự.

Các LAB và các kết nối bên trong được lập trình bằng phần mềm. Một CPLD có thể được lập trình cho các chức năng phức tạp dựa vào cấu trúc tổng của các tích cho mỗi LAB độc lập hay chính xác hơn là mỗi SPLD. Các ngõ vào có thể kết nối tới bất kỳ khối LAB nào và các ngõ ra cũng có thể kết nối tới bất kỳ LAB nào thông qua PIA.



Hình 1-10. Sơ đồ khối của CPLD tổng quát.

Hầu hết các nhà chế tạo ra một chuỗi CPLD được sắp xếp theo mật độ tích hợp, công nghệ xử lý, công suất tiêu thụ, nguồn cung cấp và tốc độ. Các nhà chế tạo thường cung cấp mật độ CPLD theo các thành phần ***macrocell*** hoặc **LAB**. Mật độ tích hợp có thể sắp xếp từ 10 macrocell đến 2000 macrocell trong một vỏ có thể lên đến vài trăm chân.

PLD càng phức tạp thì mật độ tích hợp càng cao. Một vài CPLD có thể lập trình lại và dùng công nghệ xử lý EEPROM hoặc SRAM cho các điểm kết nối lập trình. Công suất tiêu tán có thể nằm trong khoảng từ vài mili watt đến vài trăm mili watt. Nguồn cung cấp DC thường thì nằm trong khoảng từ 2,5V đến 5V tuỳ thuộc vào các chỉ định của thiết bị. Có nhiều nhà sản xuất CPLD như Altera, Xilinx, Lattice và Cypress.

Trong phần tiếp theo chúng ta sẽ khảo sát các CPLD của hai nhà sản xuất là Altera và Xilinx bởi vì hai công ty này đang chiếm lĩnh thị trường. Các nhà chế tạo khác thì cũng sản xuất thiết bị và phần mềm tương tự.

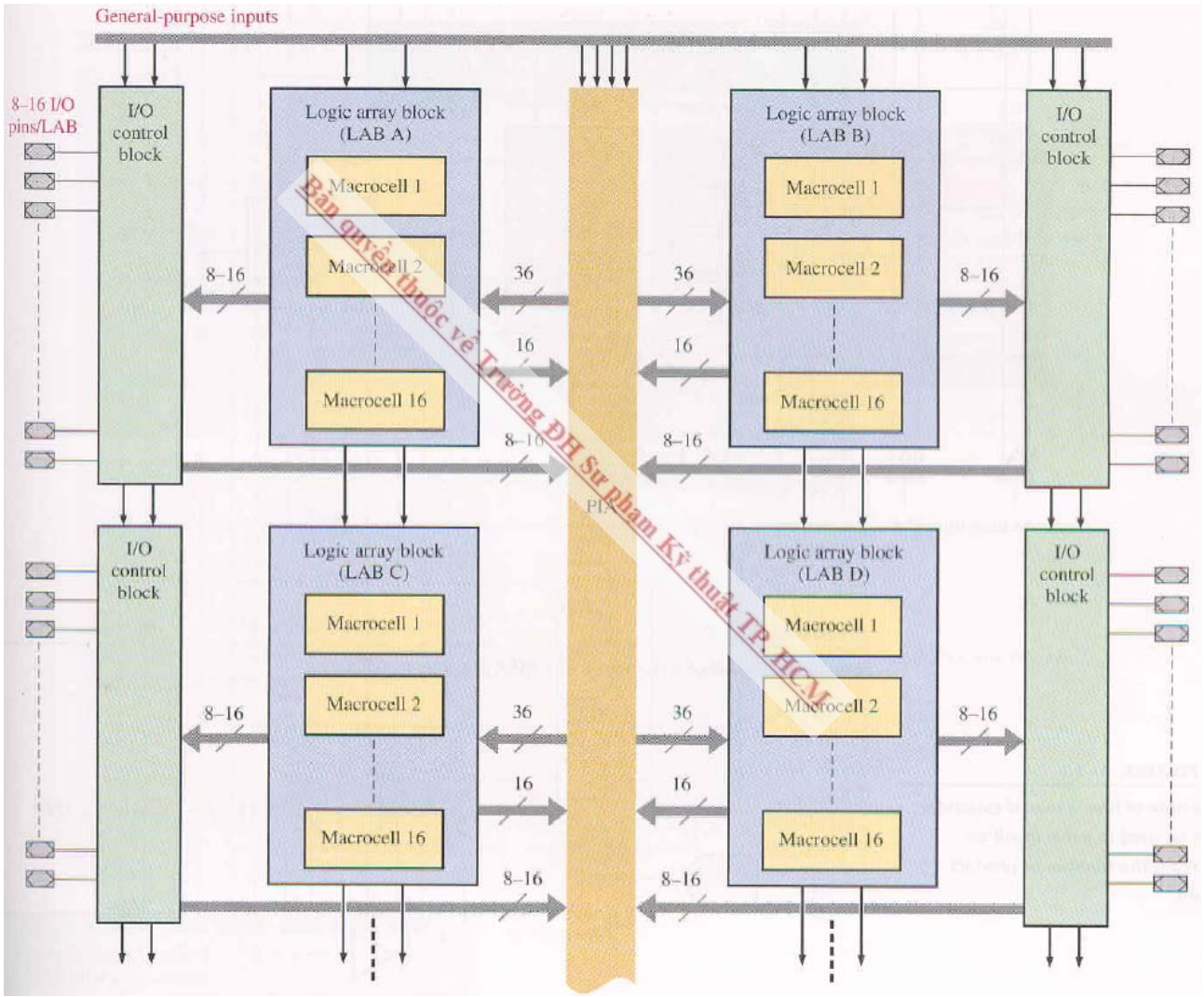
II. CPLD CỦA HÃNG ALTERA

Altera sản xuất ra nhiều họ CPLD như MAX II, MAX 3000 và MAX 7000. Trong phần này chỉ trình bày chủ yếu họ MAX 7000.

Sau khi hoàn tất phần này thì bạn có thể: mô tả được họ CPLD MAX, thảo luận về cấu trúc của CPLD MAX 7000 và CPLS MAX II, giải thích cách tạo các thành phần tích được tạo ra trong CPLD.

1. CPLD MAX 7000

Cấu trúc của CPLD là cách thức mà các thành phần bên trong được tổ chức và được sắp xếp. Cấu trúc của họ CPLD MAX 7000 thì giống như sơ đồ khối của CLPD tổng quát được trình bày ở hình 1-11.



Hình 1-11. Cấu trúc CPLD MAX 7000.

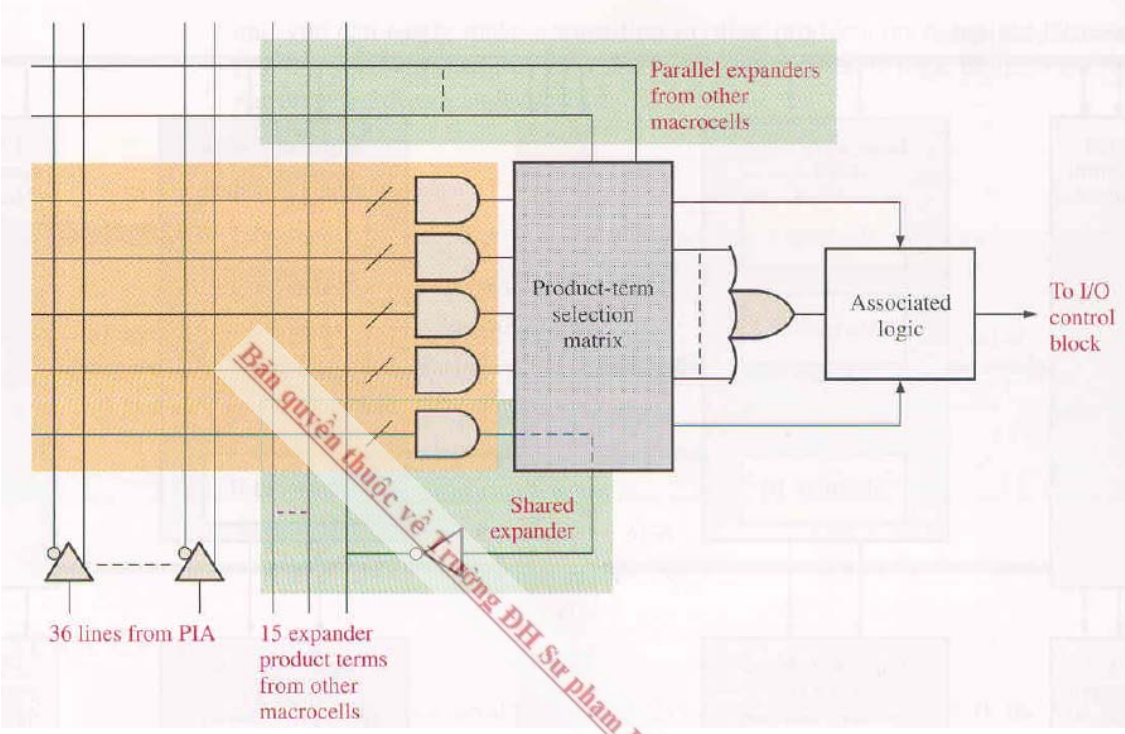
CPLD MAX 7000 có cấu trúc lớp PAL/GAL để tạo ra các hàm SOP. Mật độ nằm trong khoảng từ 2 LAB đến 16 LAB tuỳ thuộc vào CPLD cụ thể. Nên nhớ là một LAB tương đương với một SPLD dùng công nghệ xử lý EEPROM. Kiểu lập trình trong hệ thống ISP (In-System Programmable) dùng giao tiếp chuẩn JTAG.

Hình 1-11 trình bày sơ đồ khối tổng quát CPLD họ MAX 7000 của Altera. Bốn khối LAB được trình bày nhưng số lượng có thể lên đến 16 khối LAB. Mỗi khối LAB có 16 macrocell,

nhiều khối LAB được kết nối với nhau thông qua PIA, PIA là cấu trúc bus lập trình toàn cục (cho tất cả các LAB) bao gồm các ngõ vào có cùng chức năng, I/O và các macrocell.

2. MACROCELL

Sơ đồ khối macrocell đơn giản của họ MAX 7000 được trình bày trong hình 1-12. Macrocell chứa một mảng cổng AND lập trình gồm 5 cổng AND, một cổng OR, một ma trận lựa chọn thành phần tích để kết nối các ngõ ra của cổng AND với cổng OR, và logic kết hợp để có thể lập trình cho ngõ vào, ngõ ra logic tổng hợp hoặc ngõ ra thanh ghi dịch.



Hình 1-12. Sơ đồ khối macrocell đơn giản của MAX 7000.

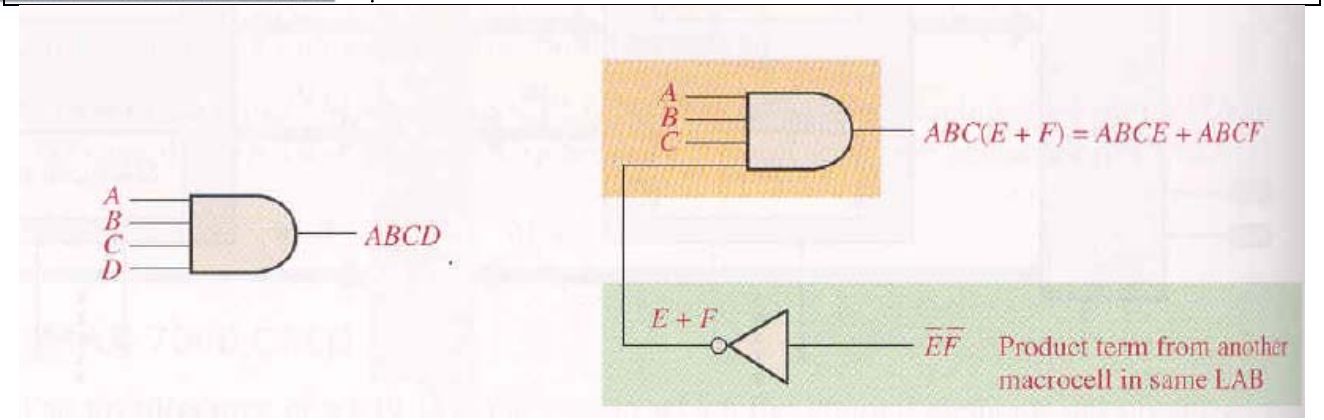
Mặc dù vẫn dùng cùng một khái niệm nhưng ***macrocell*** này khác với ***macrocell*** đã trình bày ở phần SPLD bởi vì nó có mảng cổng AND lập trình và ma trận lựa chọn thành phần tích. Trong hình 1-12 có 5 cổng AND tạo ra các thành phần tích từ PIA vào ma trận lựa chọn thành phần tích. Thành phần tích từ cổng AND nằm dưới cùng có thể được hồi tiếp trở lại ma trận lập trình xem như phần mở rộng chia sẻ để sử dụng bởi các ***macrocell*** khác.

Các ngõ vào mở rộng song song cho phép mượn các thành phần tích không dùng từ các ***macrocell*** khác để mở rộng biểu thức SOP. Ma trận lựa chọn thành phần tích là một ma trận của các kết nối lập trình được dùng để kết nối các ngõ ra đã lựa chọn từ mảng cổng AND và từ ngõ vào mở rộng đến cổng OR.

3. KHỐI MỞ RỘNG CHIA SẺ

Bù của thành phần tích được dùng để tăng số lượng thành phần tích trong biểu thức SOP thì có thể dùng được cho mỗi ***macrocell*** trong LAB. Hình 1-13 minh hoạ cách thức thành phần mở chia sẻ từ ***macrocell*** khác có thể được dùng để thiết lập thêm các thành phần tích.

Trong trường hợp này một trong 5 cổng AND trong 1 mảng macrocell bị giới hạn, chỉ có 4 ngõ vào và do đó có thể tạo ra 1 thành phần tích có 4 biến khác nhau được minh họa trong hình (a). Hình (b) trình bày phần mở rộng cho 2 thành phần tích.

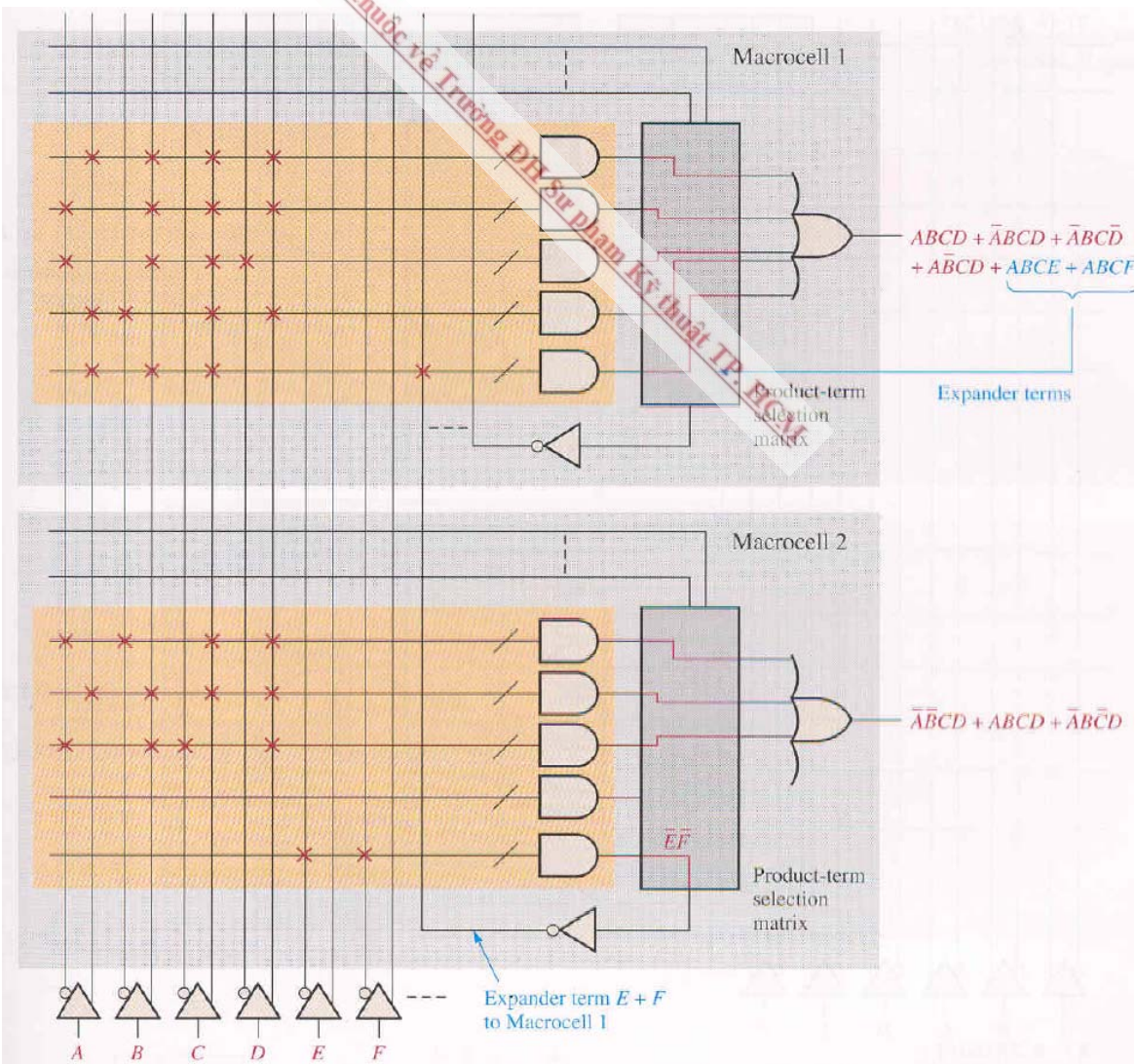


a. Cổng AND 4 ngõ vào tạo ra thành phần tích 4 biến.

b. Cổng AND được mở rộng để tạo ra 2 thành phần tích.

Hình 1-13. Ví dụ cách mở rộng.

Mỗi macrocell của MAX 7000 có thể tạo ra 5 thành phần tích từ mảng cổng AND. Nếu 1 macrocell cần nhiều hơn 5 thành phần tích cho hàm ngõ ra SOP thì nó phải dùng thêm thành phần mở rộng từ macrocell khác. Giả sử thiết kế cần biểu thức SOP chứa 6 thành phần tích. Hình 1-14 trình bày cách thành phần tích từ macrocell khác có thể được dùng để tăng biểu thức SOP ngõ ra.

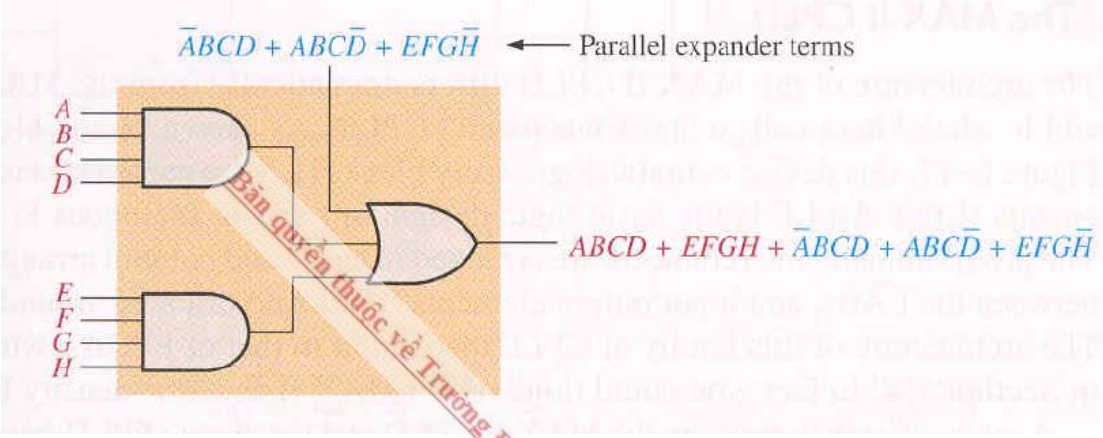


Hình 1-14. Minh họa cho việc chia sẻ.

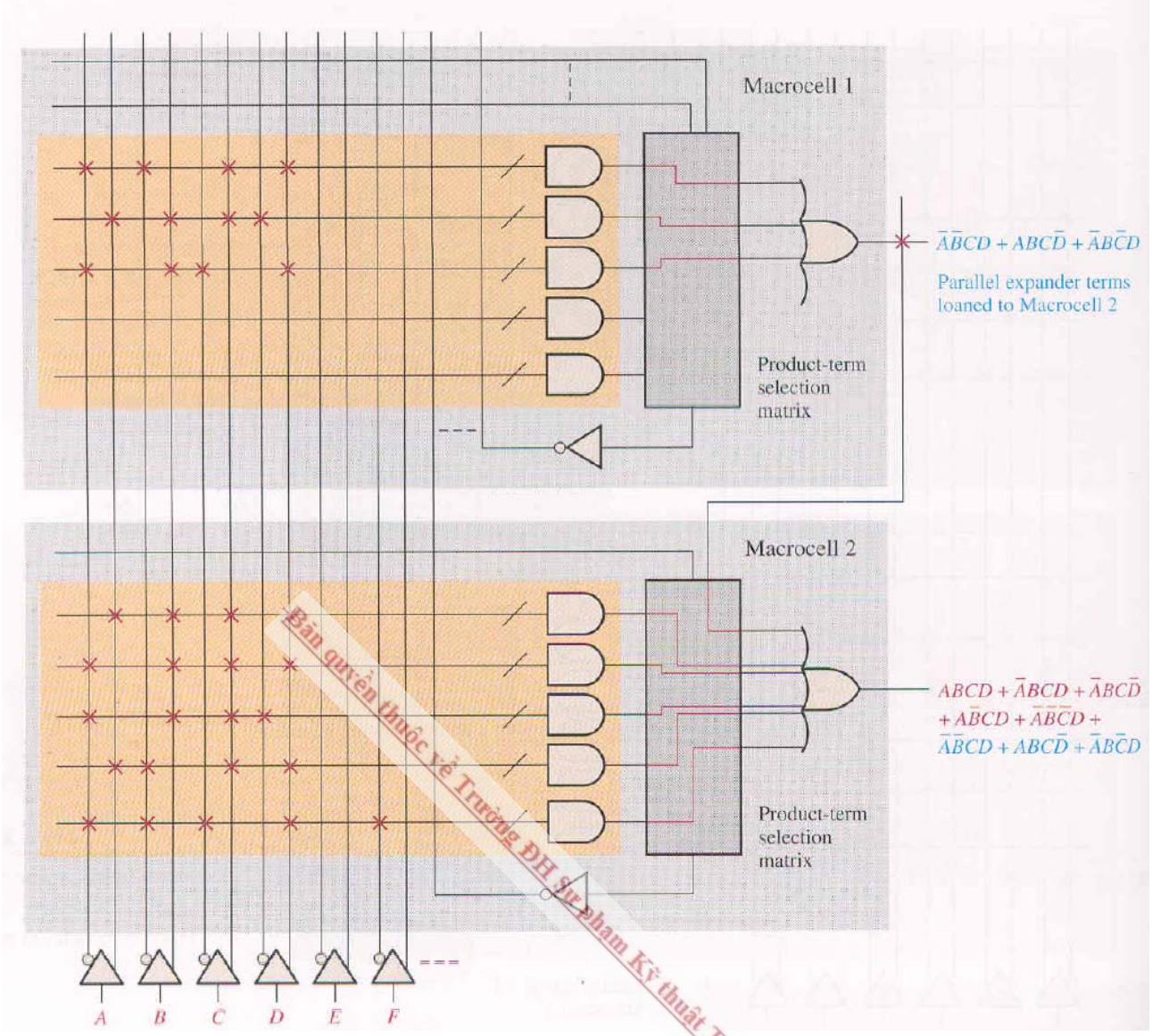
Macrocell thứ 2 tạo ra thành phần mở rộng chia sẻ $(E+F)$ được nối đến cổng AND thứ 5 trong ***macrocell*** thứ 1 để tạo ra biểu thức SOP với 6 thành phần tích. Các dấu kết nối × được tạo ra trong phần cứng từ chương trình thiết kế và phần mềm biên dịch rồi nạp vào chip.

4. KHỐI MỞ RỘNG SONG SONG

Một phương pháp khác để tăng số lượng các thành phần tích cho một ***macrocell*** bằng cách dùng bộ mở rộng song song – trong nó các thành phần tích mở rộng được OR với các thành phần được tạo ra ***macrocell*** thay vì dùng kết hợp trong ma trận AND như ở bộ mở rộng chia sẻ. Một ***macrocell*** đã cho có thể mượn các thành phần tích không dùng từ các macrocell lân cận (có thể lên đến 5 thành phần từ các macrocell khác đối với MAX 7000). Khái niệm này được minh họa như hình 1-15 trong đó mạch điện đơn giản được tạo ra từ 2 thành phần tích mượn thêm 3 thành phần tích mở rộng.



Hình 1-15. Minh họa cho bộ mở rộng song song.



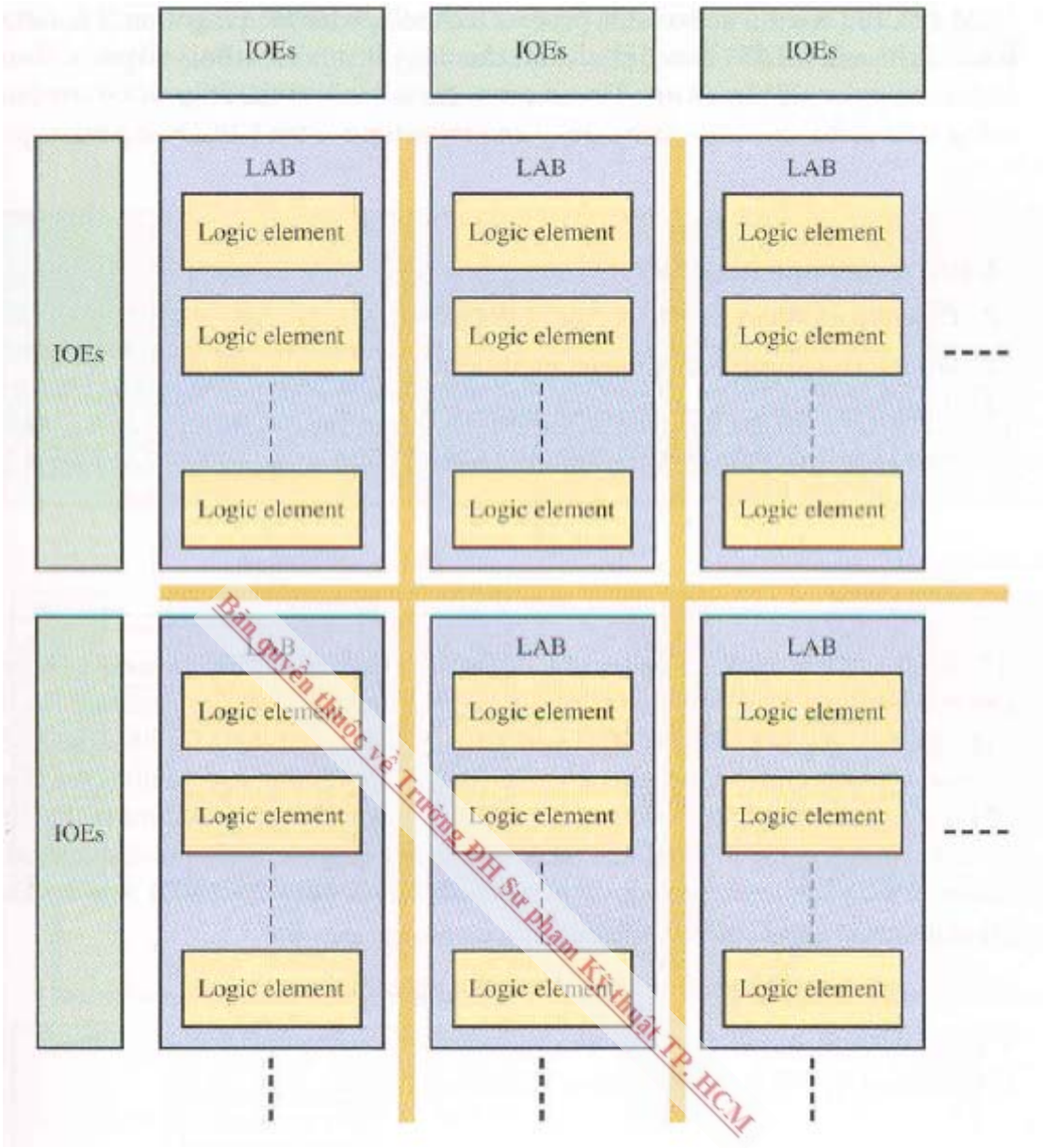
Hình 1-16. Minh họa cho bộ mở rộng song song từ macrocell khác.

Hình 1-16 trình bày cách một ***macrocell*** có thể mượn các thành phần mở rộng song song từ ***macrocell*** khác để tăng biểu thức ngõ ra SOP. ***Macrocell*** thứ 2 dùng 3 thành phần tích từ ***macrocell*** thứ 1 để tạo ra biểu thức SOP gồm 8 thành phần.

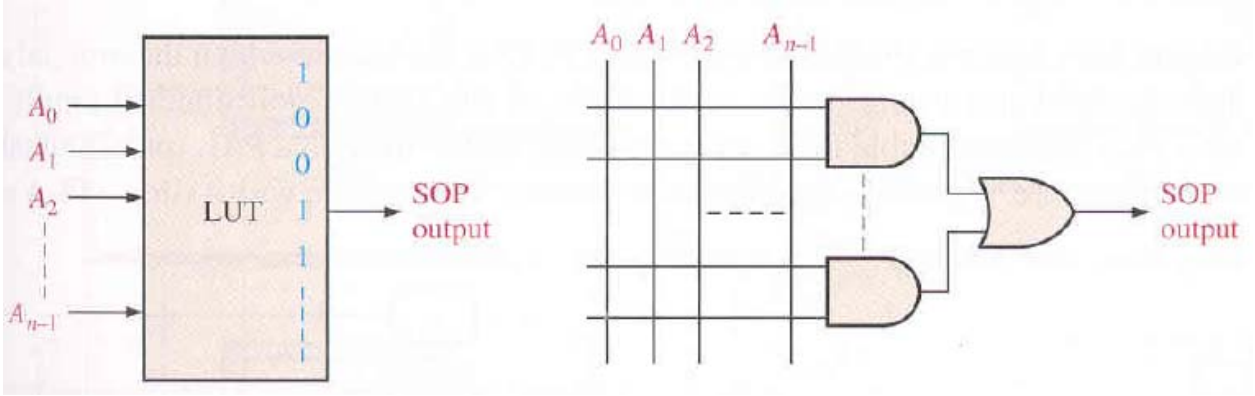
5. CPLD MAX II

Cấu trúc của CPLD MAX II khác với học MAX 7000 và được Altera gọi là CPLD "Post-macrocell". Như đã trình bày trong sơ đồ khối hình 1-17, thiết bị này chứa các khối LAB cùng với nhiều thành phần logic LE (***Logic Elements***). Một LE là một đơn vị thiết kế logic cơ bản và tương tự như ***macrocell***. Kết nối bên trong có thể lập trình được sắp xếp theo hàng và cột chạy giữa các LAB và các phần tử ngõ vào/ngõ ra (IOE: Input/Output Elements) được định hướng xung quanh. Cấu trúc của họ CPLD này giống như FPGA – có thể xem MAX II là FPGA có mật độ thấp.

Sự khác nhau giữa CPLD MAX II và các CPLD thiết kế từ SPLD là cách xây dựng một hàm logic. CPLD MAX II sử dụng các bảng tra LUT (Look-Up Tables) thay cho ma trận AND/OR. Một LUT về cơ bản là loại bộ nhớ có thể lập trình để tạo ra các hàm SOP. Hai phương pháp này được minh họa như hình 1-18.



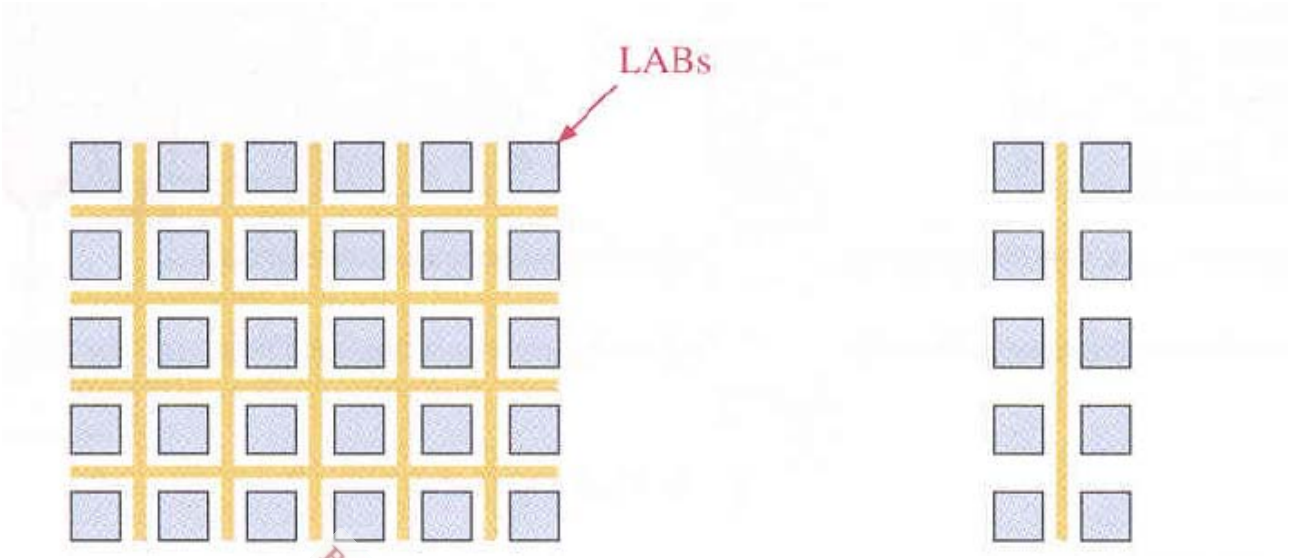
Hình 1-17. Sơ đồ khối của MAX II.



Loại dùng LUT. **Loại dùng logic mảng AND/OR**

Hình 1-18. Phân biệt 2 kiểu xây dựng hàm.

Như đã đề cập CPLD MAX II có cách sắp xếp hàng/cột của các kết nối bên trong thay cho cách kết nối bên trong theo loại kênh có trong hầu hết các CPLD. Có 2 phương pháp được minh họa trong hình 1-19.



Kết nối bên trong dùng hàng/cột **Kết nối theo kiểu kênh**

Hình 1-19. Phân biệt 2 kiểu kết nối.

Hầu hết các CPLD dùng công nghệ xử lý ***không bay hơi*** cho các điểm nối lập trình. Tuy nhiên MAX II dùng công nghệ xử lý như SRAM nên chúng ***có thể bay hơi*** – tất cả các logic đã lập trình sẽ mất hết khi mất điện. Bộ nhớ được gắn vào bên trong chip để lưu trữ dữ liệu chương trình dùng công nghệ bộ nhớ không bay hơi và sẽ định cấu hình lại cho CPLD khi có điện.

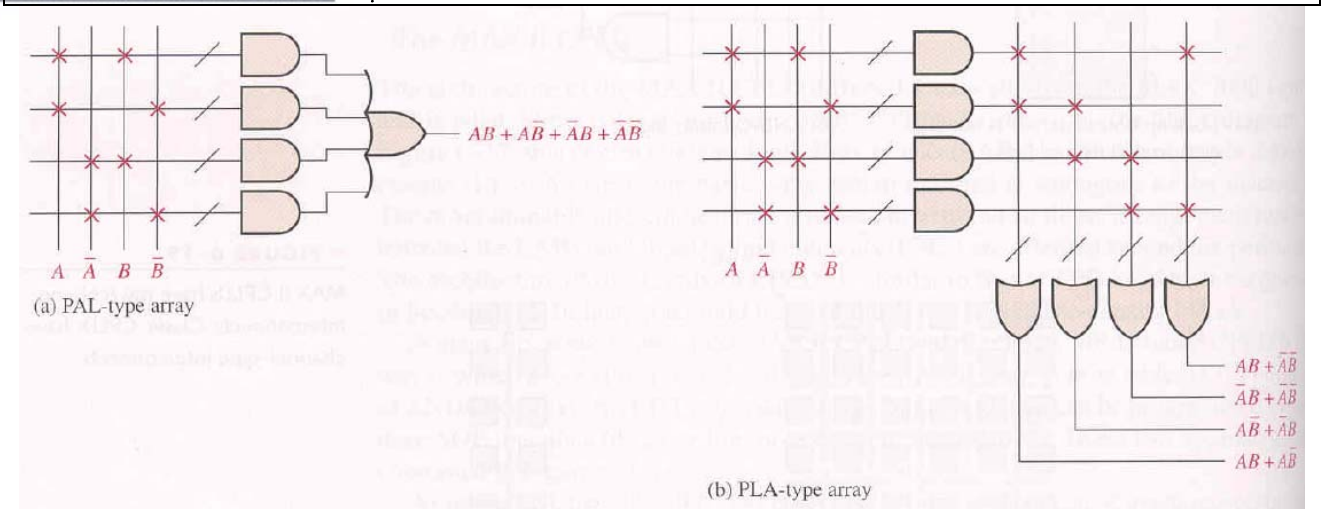
III. CPLD CỦA HÃNG XILINX:

Cũng giống như Altera, Xilinx sản xuất ra các họ CPLD được sắp xếp theo mật độ tích hợp, công nghệ xử lý, điện áp nguồn cung cấp và tốc độ. Xilinx chế tạo ra nhiều họ CPLD như Cool Runner II, Cool Runner XPLA3 và XC9500. Họ XC9500 thì có cấu trúc giống như họ CPLD MAX 7000 của Altera sử dụng cấu trúc loại PAL/GAL. Trong phần này chúng ta chỉ phân tích Cool Runner II.

Sau khi kết thúc phần này bạn có thể: mô tả PLA và so sánh với PAL, thảo luận về cấu trúc CPLD Cool Runner II và mô tả các khối chức năng.

1. PLA (PROGRAMMABLE LOGIC ARRAY)

Như đã trình bày, cấu trúc của CPLD là cách mà các thành phần bên trong được tổ chức và sắp xếp. Cấu trúc của họ Cool Runner II của Xilinx thì dựa vào cấu trúc mảng logic lập trình PLA (Programmable Logic Array) tốt hơn cấu trúc PAL (Programmable Array Logic). Hình 1-20 so sánh cấu trúc PAL với cấu trúc PLA đơn giản.

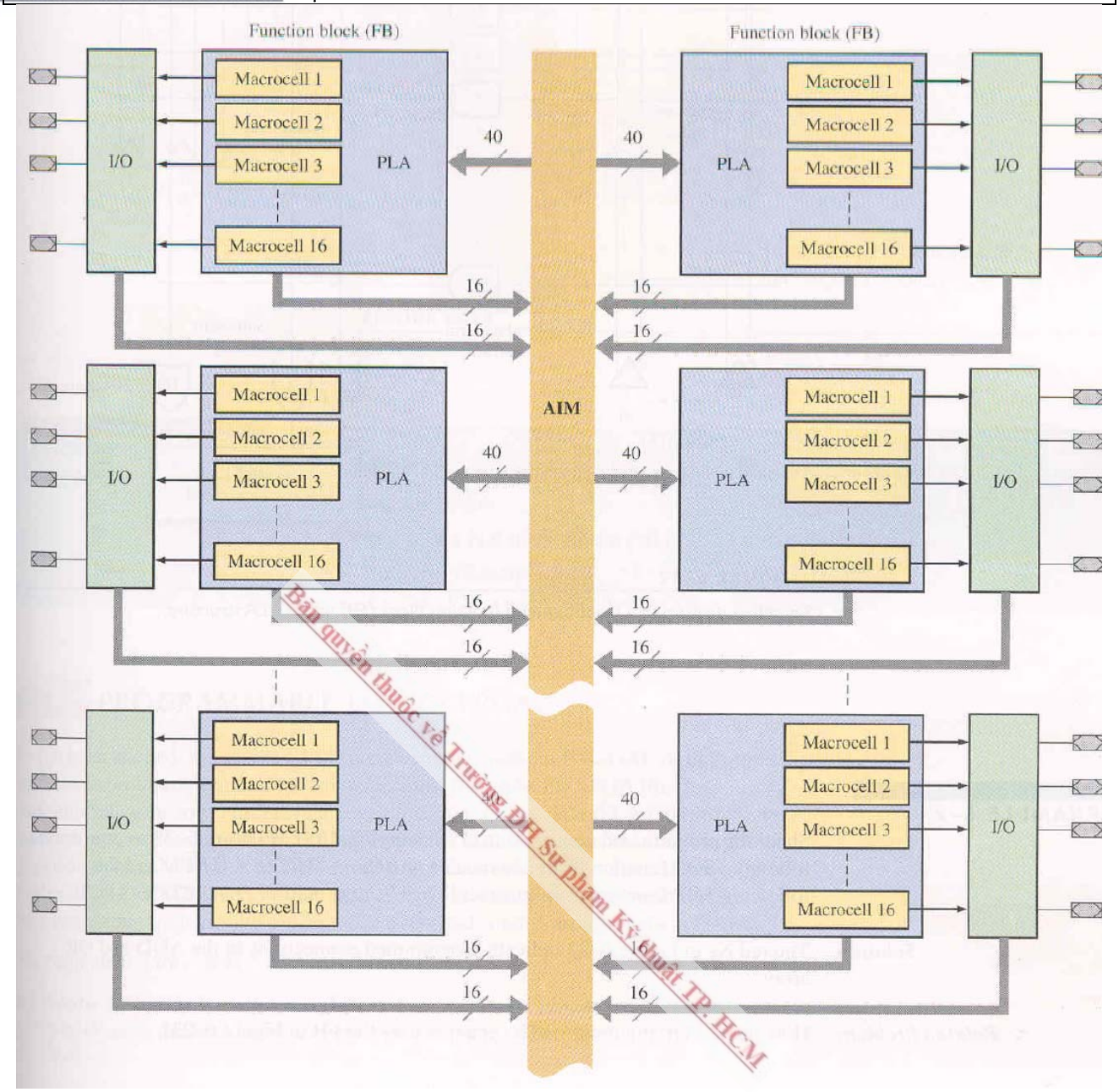


Hình 1-20. So sánh PAL với PLA.

Như đã trình bày, PAL có mảng cổng AND lập trình và theo sau là mảng cổng OR cố định để tạo ra các biểu thức SOP như hình 1-20a. PLA có mảng cổng AND lập trình và theo sau là mảng cổng OR lập trình như hình 1-20b.

2. COOLRUNNER II

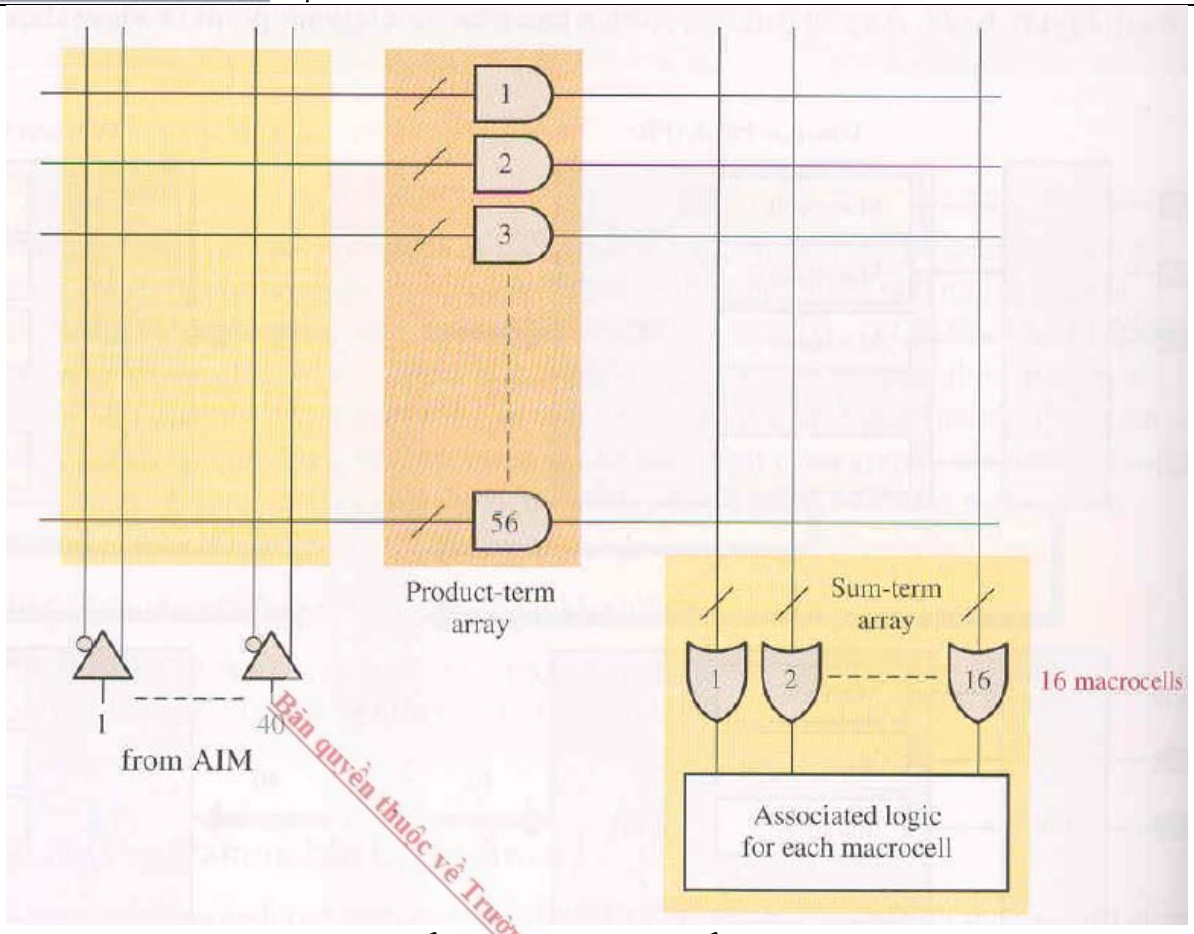
CPLD Cool Runner II dùng loại cấu trúc PLA. Cool Runner II có nhiều khối chức năng FB (Function Block) tương tự như LAB trong CPLD MAX 7000 của Altera. Mỗi khối chức năng FB chứa 16 macrocell. Các khối chức năng được kết nối bên trong bởi một ma trận kết nối bên trong cải tiến AIM tương tự như PIA trong MAX 7000. Sơ đồ cấu trúc cơ bản cho Cool Runner II được trình bày ở hình 1-21.



Hình 1-21. Sơ đồ cấu trúc của Cool runner II.

Sơ đồ khối CPLD của Xilinx và của Altera gần như là giống nhau tuy nhiên bên trong thì khác nhau.

CPLD họ Cool Runner II chứa từ 32 macrocell đến 512 macrocell. Do có 16 macrocell cho mỗi khối chức năng, số lượng khối chức năng nằm trong khoảng từ 2 đến 32. Sơ đồ khối của một khối chức năng FB được trình bày như hình 1-22.

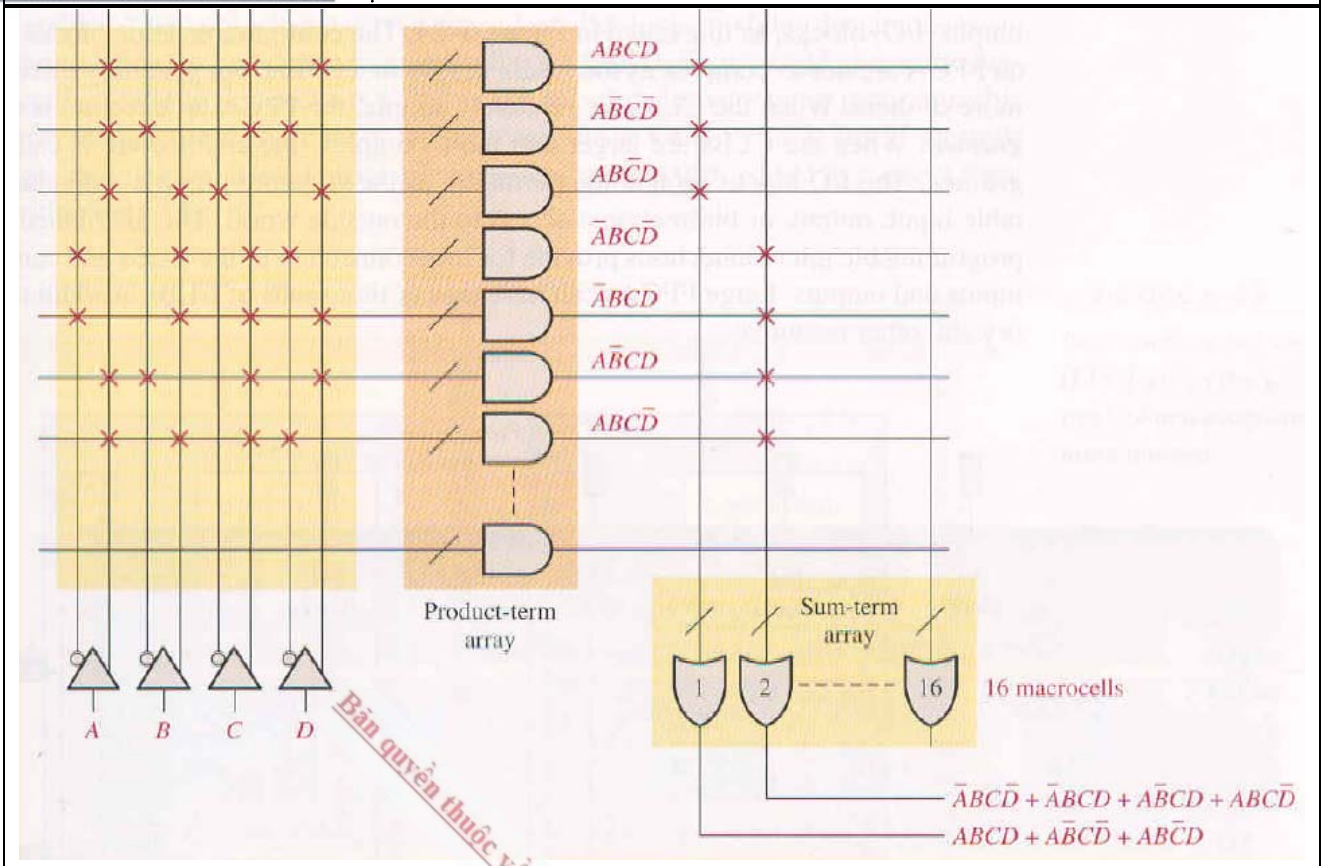


Hình 1-22. Cấu trúc của một khối chức năng FB.

Mảng cổng AND có 56 cổng AND và mảng cổng OR lập trình có 16 cổng OR. Với cấu trúc PLA thì bất kỳ thành phần tích nào cũng có thể nối tới cổng OR để tạo nên biểu thức SOP cho ngõ ra. Với khả năng cực đại mỗi khối chức năng có thể tạo ra 16 ngõ ra và mỗi ngõ ra có biểu thức SOP chứa 56 thành phần tích.

Ví dụ 1-2: Hãy lập trình kết nối bên trong khối FB của hình 1-22 để tạo ra hàm chức năng SOP từ macrocell thứ 1 là: $ABCD + A\overline{B}C\overline{D} + AB\overline{C}D$ và hàm cho macrocell thứ 2 là: $\overline{A}BC\overline{D} + \overline{A}BCD + A\overline{B}CD + ABC\overline{D}$

Giải: kết quả như hình 1-23:



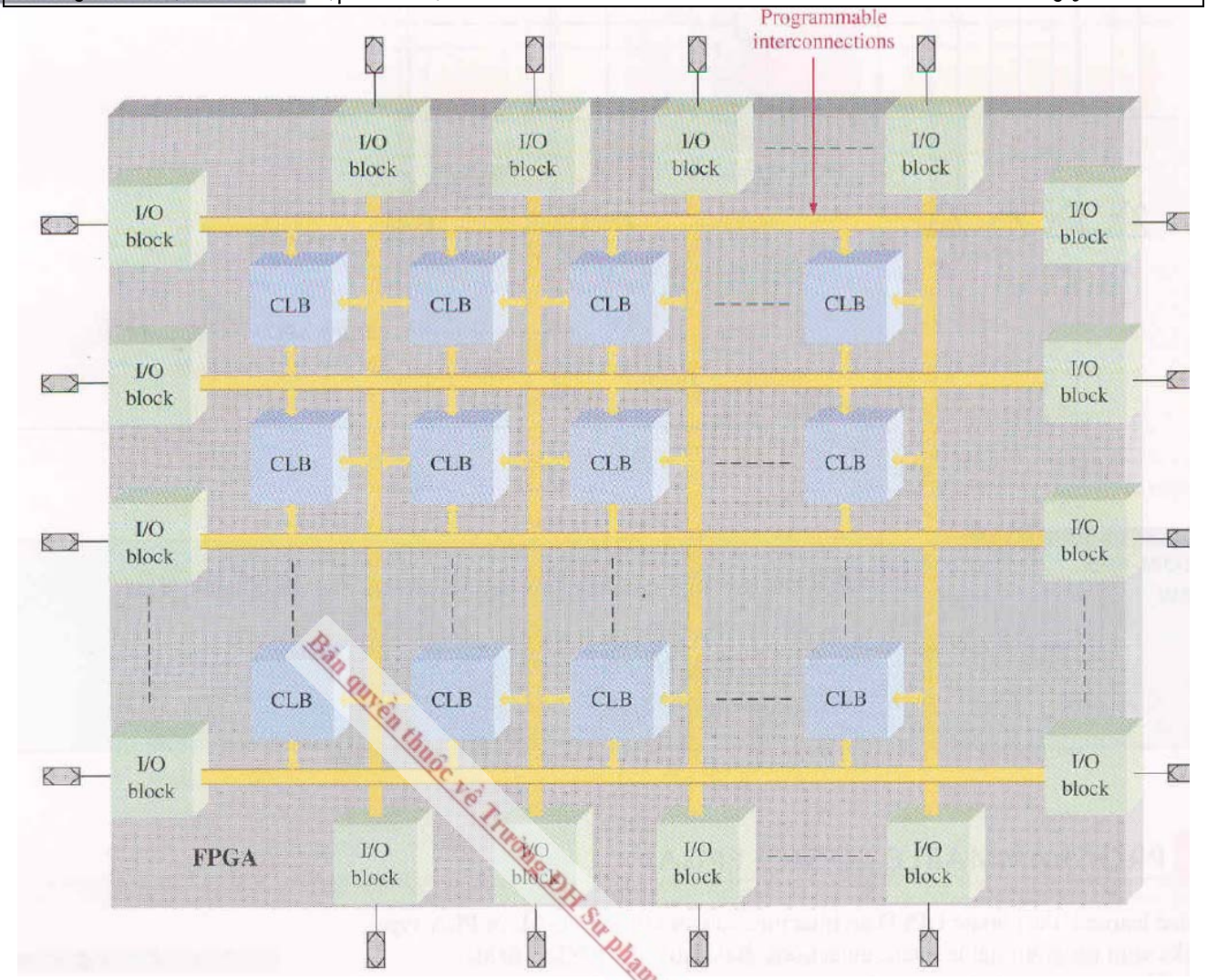
Hình 1-23. Minh họa cho ví dụ 1-2.

IV. LOGIC LẬP TRÌNH FPGA

Như đã trình bày ở trên, cấu trúc phân loại CPLD bao gồm các khối logic loại PAL/GAL hoặc PLA với các kết nối bên trong có thể lập trình. Về cơ bản FPGA (**Field Programmable Gate Array**) có cấu trúc khác – không dùng mảng loại PAL/PLA – có mật độ tích hợp cao hơn nhiều so với CPLD. Các phần tử dùng để tạo ra các hàm logic trong FPGA thường thì nhỏ hơn nhiều so với các thành phần trong CPLD. Tương tự trong FPGA thì các kết nối bên trong được tổ chức theo hàng và cột.

Sau khi kết thúc phần này bạn có thể: mô tả cấu trúc cơ bản của FPGA, so sánh FPGA với CPLD, thảo luận về LUT, thảo luận về FPGA dùng cấu trúc SRAM và định nghĩa lõi của FPGA.

Có 3 thành phần cơ bản trong FPGA là khối logic có thể định cấu hình logic CLB (**Configurable Logic Block**), các kết nối bên trong và các khối ngõ vào/ra được minh họa như hình 1-24.



Hình 1-24. Cấu trúc cơ bản của FPGA.

Các khối có thể định cấu hình logic CLB trong FPGA thì không phức tạp bằng các khối LAB hoặc FB trong CPLD nhưng thường thì có nhiều thành phần hơn. Khi CLB khá đơn giản thì cấu trúc FPGA được gọi là ***fine grained***. Các khối IO nằm xung quanh của cấu trúc tạo ra sự truy xuất ngõ vào, ngõ ra hoặc cả hai chiều có thể lựa chọn một cách độc lập đến thế giới bên ngoài.

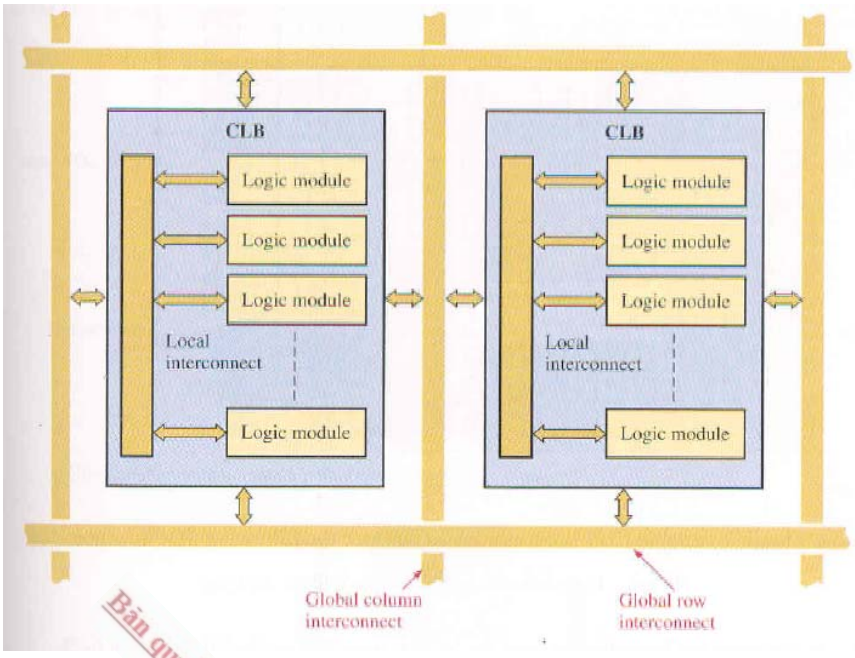
Ma trận phân loại của các kết nối bên trong có thể lập trình tạo ra các kết nối bên trong của CLB và kết nối đến các ngõ vào và các ngõ ra. Các FPGA lớn có thể có 10000 CLB và có thêm bộ nhớ và các nguồn tài nguyên khác.

Hầu hết các nhà chế tạo các thiết bị logic lập trình thường sắp xếp thành chuỗi FPGA phân loại theo mật độ, công suất tiêu tán, điện áp nguồn cung cấp, tốc độ và một vài mức độ khác nhau về cấu trúc. FPGA là thiết bị có thể lập trình lại và sử dụng công nghệ xử lý SRAM hoặc bán cầu chì để lập trình cho các điểm nối. Mật độ có thể nằm trong khoảng từ vài trăm module logic đến sấp xỉ khoảng 180000 module logic trong 1 vỏ với số lượng chân lên đến 1000. Nguồn cung cấp DC thường nằm trong khoảng 1,2V đến 2,5V tuỳ thuộc vào loại chip.

1. CÁC KHỐI LOGIC CÓ THỂ ĐỊNH CẤU HÌNH CLB

Thường thì khối logic của FPGA chứa một vài module logic khá nhỏ tương tự như macrocell trong CPLD. Hình 1-25 trình bày các khối CLB cơ bản nằm trong các kết nối bên trong có thể lập trình hàng/cột toàn cục – được dùng để kết nối các khối logic. Mỗi CLB được

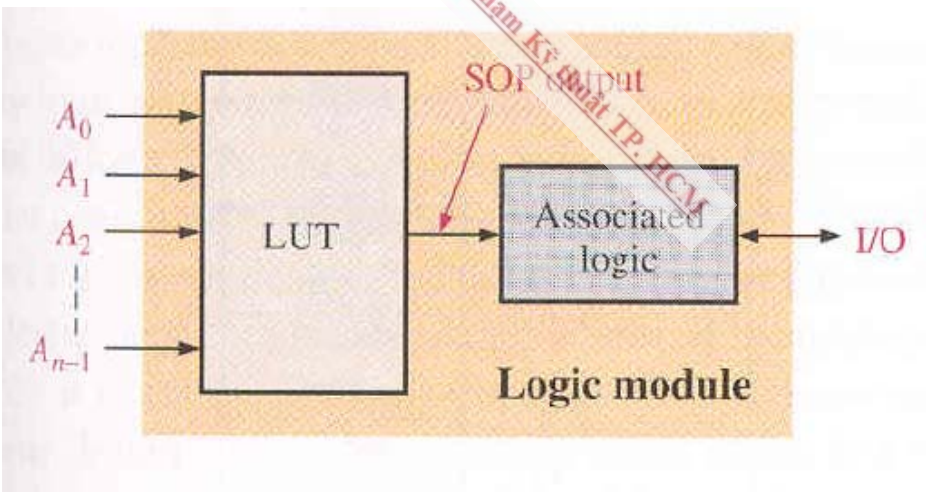
thiết lập từ nhiều module logic nhỏ hơn và các kết nối bên trong có thể lập trình cục bộ – được dùng để kết nối các module logic với CLB.



Hình 1-25. Các khối CLB của FPGA.

2. CÁC MODULE LOGIC

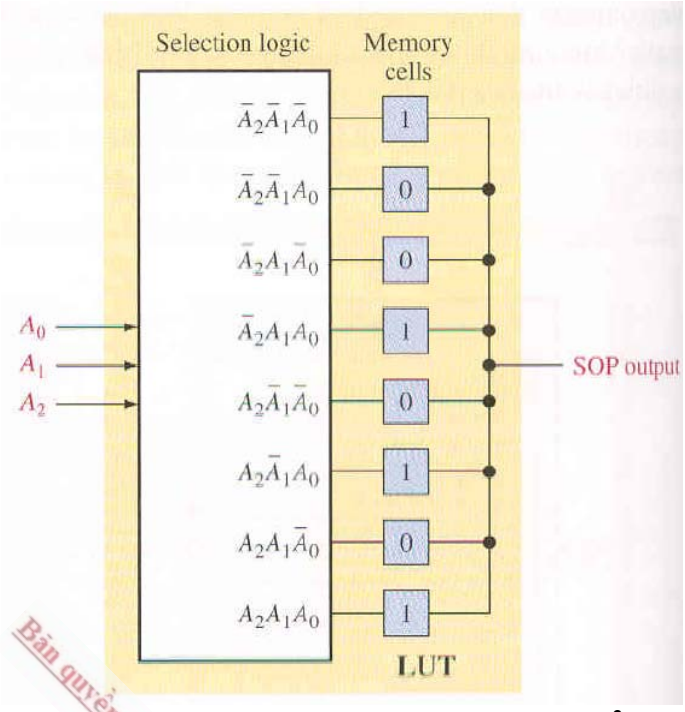
Một module logic trong một khối logic của FPGA có thể được định cấu hình cho hàm logic tổ hợp, hàm logic thanh ghi hoặc cho cả 2. Flip flop là thành phần logic kết hợp và được dùng cho các hàm logic thanh ghi. Sơ đồ khối của module logic tiêu biểu dùng cấu trúc LUT được trình bày như hình 1-26.



Hình 1-26. Sơ đồ khối cơ bản của 1 module logic trong FPGA.

Thường thì tổ chức của một LUT bao gồm một số các ô nhớ bằng với 2^n, trong đó n là số lượng các biến ngõ vào. Ví dụ: 3 ngõ vào có thể lựa chọn đến 8 ô nhớ, do đó LUT với biến ngõ vào có thể tạo ra biểu thức SOP lên đến 8 thành phần tích. Một mô hình mẫu của 1 và 0 có thể được lập trình vào trong các ô nhớ của LUT được minh họa như hình 1-27 để tạo ra hàm SOP theo chỉ định. Các ô nhớ chứa số 1 có nghĩa là thành phần tích được kết hợp trong biểu thức SOP của ngõ ra và ô nhớ chứa số 0 có nghĩa là thành phần tích kết hợp không xuất hiện trong biểu thức SOP của ngõ ra.

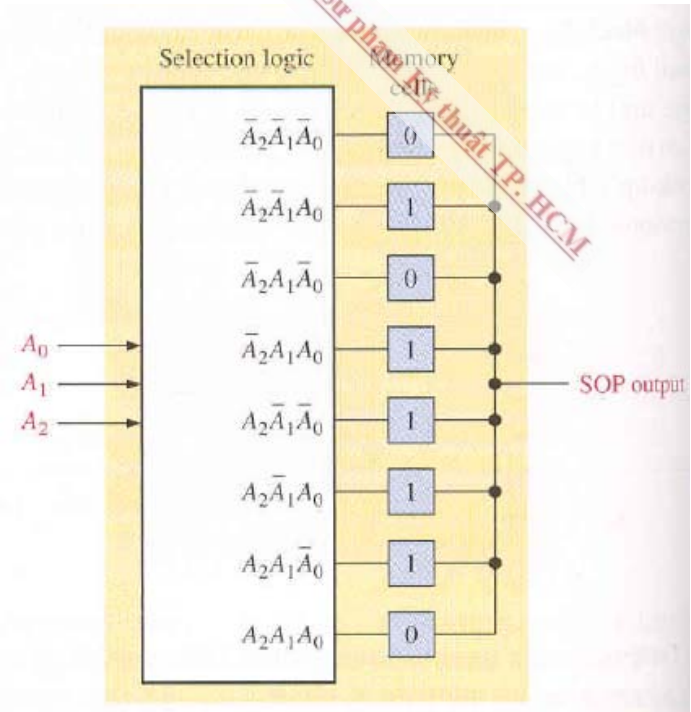
Kết quả biểu thức SOP ngõ ra là $\overline{A_2}\,\overline{A_1}\,\overline{A_0} + \overline{A_2}A_1A_0 + A_2\overline{A_1}A_0 + A_2A_1A_0$



Hình 1-27. Khái niệm cơ bản của LUT được lập trình để tạo SOP ngõ ra.

Ví dụ 1-3: Hãy thiết lập LUT có 3 biến cơ bản được lập trình để tạo ra biểu thức SOP theo sau: $A_2A_1\overline{A_0} + A_2\overline{A_1}\,\overline{A_0} + \overline{A_2}A_1A_0 + A_2\overline{A_1}A_0 + \overline{A_2}\,\overline{A_1}A_0$

Giải: kết quả như hình 1-28:



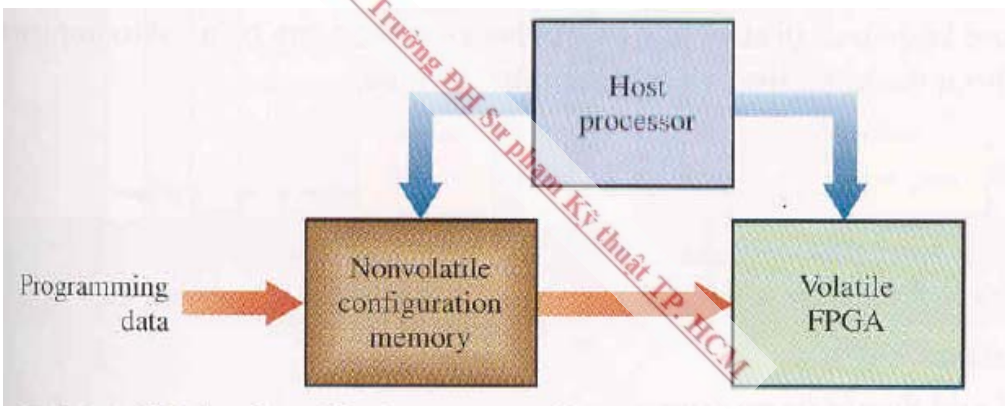
Hình 1-28. Minh họa cho ví dụ 1-3.

3. FPGA DÙNG CÔNG NGHỆ SRAM

Các FPGA cũng có thể là không bay hơi nếu dùng công nghệ bán cầu chì hoặc có thể bay hơn nếu dùng công nghệ SRAM. Khái niệm bay hơi có nghĩa là tất cả các dữ liệu đã lập trình

vào trong các khối CLB sẽ bị mất hết khi mất điện. Do đó, các FPGA dùng công nghệ SRAM chứa cả bộ nhớ không bay hơi tích hợp bên trong chip để lưu trữ chương trình và dữ liệu và định cấu hình lại cho thiết bị mỗi khi có điện trở lại hoặc chúng dùng bộ nhớ bên ngoài với việc chuyển dữ liệu được điều khiển vi xử lý chủ. Khái niệm bộ nhớ tích hợp trong chip được minh họa như hình 1-29a và khái niệm định cấu hình lại dùng vi xử lý được trình bày như hình 1-29b.

a. FPGA bay hơi định lại cấu hình dùng bộ nhớ không bay hơi bên trong.



b. FPGA bay hơi định lại cấu hình dùng bộ nhớ không bay hơi và vi xử lý.

Hình 1-29. Khái niệm về FPGA bay hơi.

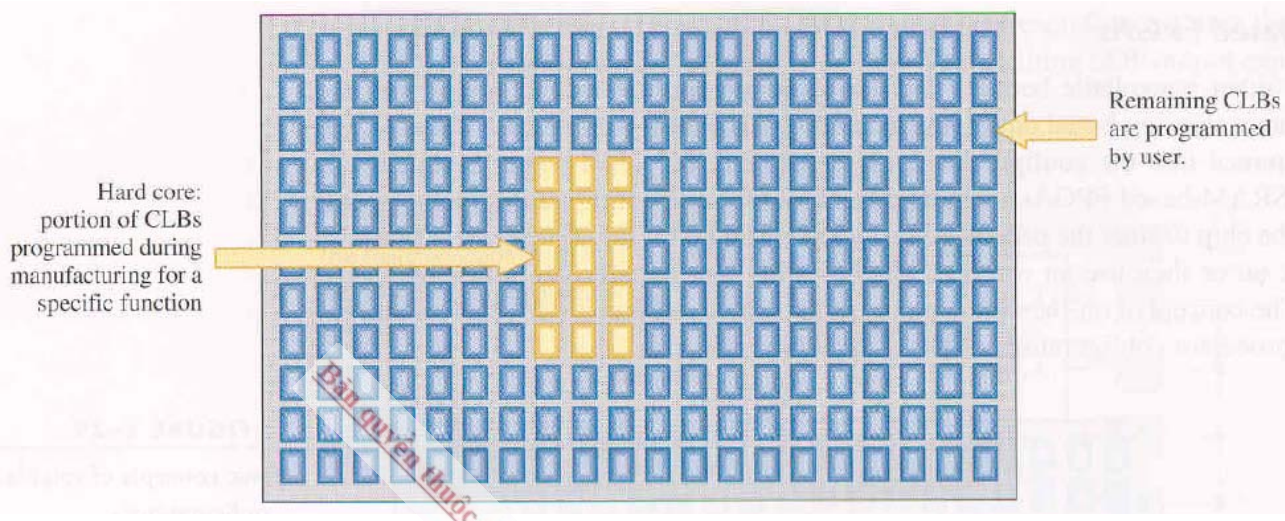
4. CÁC LÕI CỦA FPGA

Các FPGA về cơ bản giống như các phiến trắng mà người dùng có thể lập trình cho các thiết kế logic. Các FPGA tiện lợi khi mà nó chứa các mạch logic "lõi phần cứng" (***hard core***). Một mạch logic lõi phần cứng là một phần logic trong FPGA được đặt vào bên trong bởi nhà chế tạo để cung cấp các chức năng đặt biệt và không thể lập trình lại. Ví dụ nếu khách hàng cần một vi xử lý nhỏ như là một phần của thiết kế hệ thống thì nó có thể được lập trình vào trong FPGA cho khách hàng hoặc nó có thể được cung cấp như là một lõi phần cứng bởi nhà chế tạo. Nếu chức năng được tích hợp vào bên trong có vài cấu trúc có thể lập trình được thì nó được xem như là chức năng "lõi mềm" (***soft core***).

Ưu điểm của phương pháp dùng lõi phần cứng là cùng một thiết kế có thể thực hiện đầy đủ dùng khả năng của FPGA ít hơn nếu so với cách người dùng sử dụng cách lập trình, kết quả là không gian trên chip nhỏ hơn và thời gian thiết kế ngắn hơn.

Khuyết điểm của phương pháp dùng lõi phần cứng là các thông số kỹ thuật là cố định trong quá trình chế tạo và khách hàng phải có khả năng dùng được chức năng đó. Nó không thể thay đổi về sau.

Các lõi phần cứng thường có tác dụng cho các chức năng mà chúng được sử dụng phổ biến trong các hệ thống số như vi xử lý, giao tiếp ngõ vào/ngõ ra và xử lý tín hiệu số (***Digital Signal Processor***). Có nhiều chức năng lõi phần cứng có thể lập trình trong FPGA. Hình 1-30 minh hoạ cho khái niệm lõi phần cứng được bao quanh bởi CLB được lập trình bởi người sử dụng.



Hình 1-30. Khái niệm chức năng lõi phần cứng trong FPGA.

Việc thiết kế các lõi phần cứng thường được xây dựng bởi nhà chế tạo FPGA và chúng thuộc sở hữu của nhà chế tạo. Các thiết kế riêng bởi nhà chế tạo được đặt tên là ***Intellectual Property*** (IP) – sở hữu trí tuệ. Một công ty thường liệt các loại sở hữu trí tuệ mà chúng có hiệu lực trên các website. Nhiều sở hữu trí tuệ là sự kết hợp của lõi phần cứng và lõi phần mềm. Vi xử lý là một ví dụ minh họa – có vài tính năng mềm dẻo trong lựa chọn và điều chỉnh một vài thông số bởi người dùng.

Các FPGA chứa các vi xử lý tích hợp một trong hai hoặc cả hai lõi phần cứng và lõi phần mềm và nhiều chức năng khác thì được đặt tên là **Platform FPGA** bởi vì chúng có thể được dùng để điều khiển một hệ thống đầy đủ mà không cần thêm một thiết bị hỗ trợ nào.

V. FPGA CỦA ALTERA

Altera sản xuất ra nhiều họ FPGA bao gồm Stratix II, Stratix , Cyclone và ACEX. Trong phần này chúng ta chỉ khảo sát họ Stratix II để minh họa cho các khái niệm.

Sau khi kết thúc phần này chúng ta có thể:

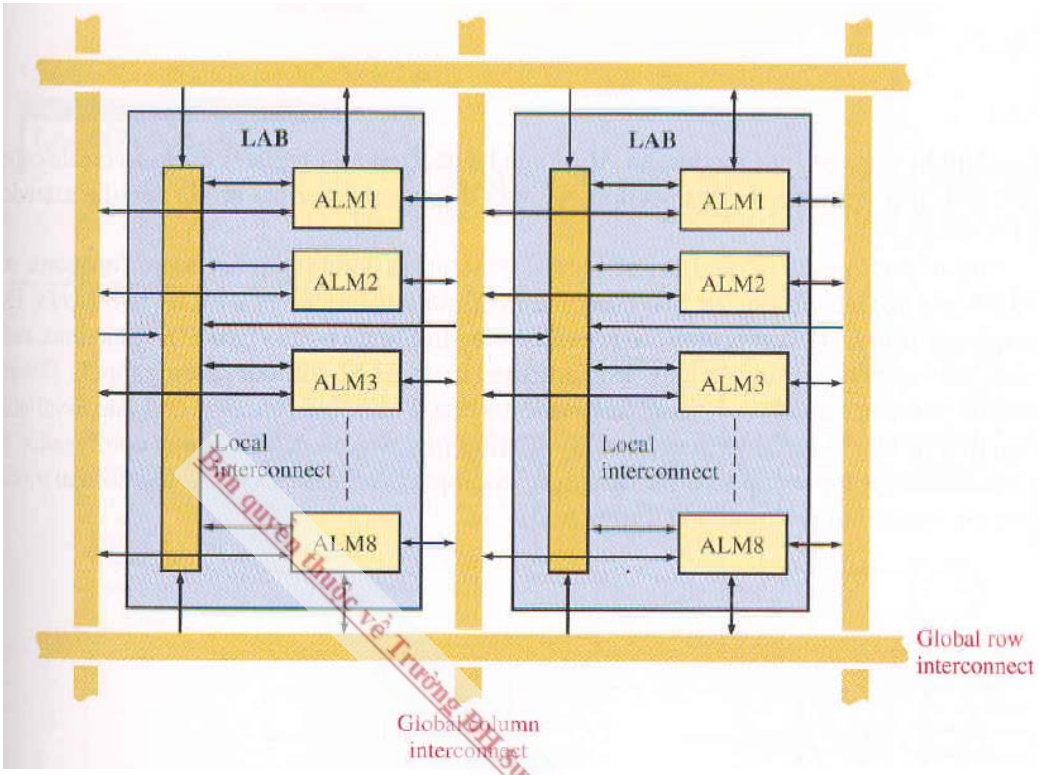
Thảo luận về cấu trúc cơ bản của FPGA họ Stratix II, giải thích cách thành phần được tạo ra trong FPGA, thảo luận về các chức năng được tích hợp.

1. KHỐI MẢNG LOGIC (LAB – LOGIC ARRAY BLOCK)

Sơ đồ khối của FPGA tổng quát đã được trình bày như hình 1-24; cấu trúc của Stratix II và các họ Altera khác thì giống nhau. Chúng đều có cấu trúc loại LUT cho các module logic – được gọi là module logic thích nghi ALM (**Adaptive Logic Module**) được trình bày trong thiết bị tổng quát LAB. Mật độ được phân loại từ 2000 LAB cho đến 22000 LAB tuỳ thuộc vào các họ cụ thể và mỗi LAB có 8 ALM. Kích thước vỏ thay đổi từ 314 chân đến 1173 chân. Thiết bị

yêu cầu sử dụng nguồn DC cung cấp từ 1,2V; 1,5V và 2,5V. Họ FPGA Stratix II sử dụng công nghệ của SRAM.

Hình 1-31 trình bày sơ đồ khối của cấu trúc LAB của Stratix II. Mỗi LAB chứa 8 ALM, các LAB được liên kết với nhau thông qua các kết nối hàng và cột bên trong. Các điểm kết nối cục bộ bên trong liên kết các ALM với mỗi LAB.



Hình 1-31. Sơ đồ khối của cấu trúc LAB của Stratix II và ALM

2. MODULE LOGIC THÍCH NGHI ALM

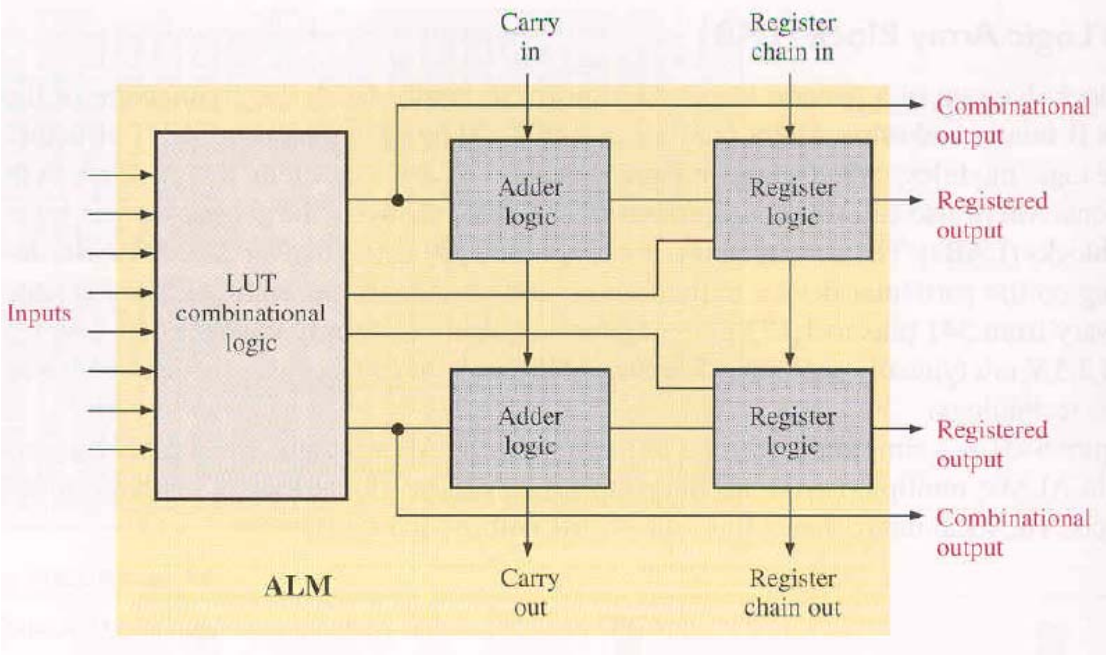
ALM là đơn vị thiết kế cơ bản trong FPGA Stratix II. Mỗi ALM chứa một phần tổ hợp logic dùng cấu trúc LUT và mạch logic kết hợp có thể được lập trình cho 2 ngõ ra logic tổ hợp hoặc hai ngõ ra thanh ghi dịch. Bên cạnh đó, ALM có mạch cộng logic, các flip flop và các mạch logic khác – cho phép thực hiện chức năng tính toán số học, chức năng đếm và thanh ghi dịch. Sơ đồ khối ALM của Stratix II được trình bày như hình 1-32.

Hoạt động của ALM:

Một ALM có thể được lập trình cho ra nhiều kiểu hoạt động như sau:

- Kiểu hoạt động bình thường.
- Kiểu hoạt động LUT mở rộng.
- Kiểu tính toán số học.
- Kiểu tính toán số học dùng chung.

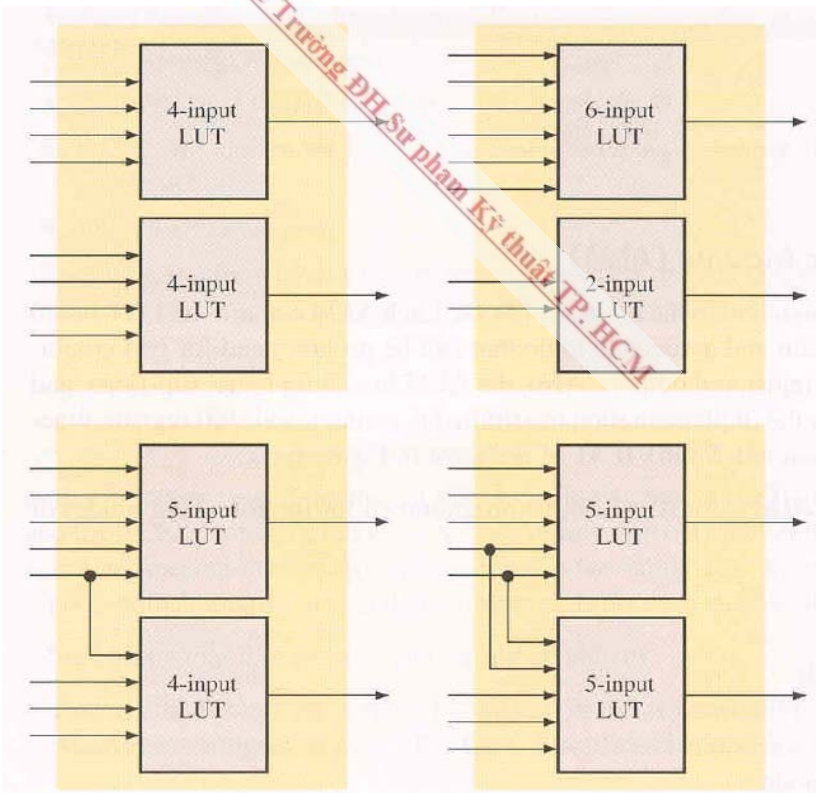
Ngoài 4 kiểu hoạt động thì ALM có thể được dùng như là 1 chuỗi thanh ghi để xây dựng bộ đếm và thanh ghi dịch. Trong phần này chúng ta sẽ khảo sát kiểu hoạt động bình thường và kiểu hoạt động LUT mở rộng.



Hình 1-32. Sơ đồ khối ALM của Stratix II.

a. Kiểu hoạt động bình thường

Được sử dụng đầu tiên để tạo các hàm logic tổ hợp. Một ALM có thể thực hiện một hoặc hai hàm ngõ ra tổ hợp với hai LUT của nó. Ví dụ về 4 cấu hình LUT được minh họa ở hình 1-33.

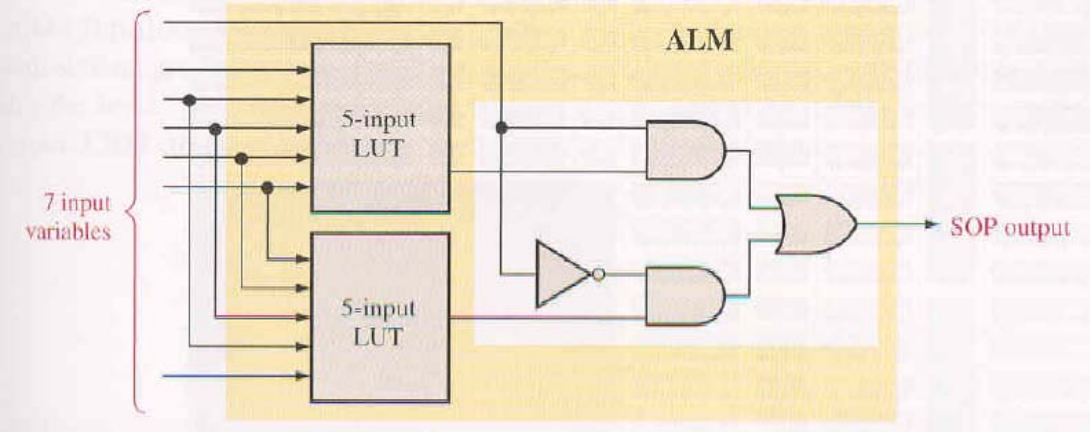


Hình 1-33. Các cấu hình có thể có của LUT trong ALM ở kiểu bình thường.

Hai hàm SOP – mỗi hàm có 4 biến hoặc ít hơn – có thể được thực hiện trong một ALM mà không cần dùng các ngõ vào chia sẽ. Ví dụ bạn có thể có "2 hàm 4 biến", "một hàm có 4 biến và một hàm 3 biến" hoặc "hai hàm 3 biến". Bằng cách chia sẽ các ngõ vào, bạn có thể có bất kỳ tổ hợp nào của 8 ngõ vào lên đến tối đa 6 ngõ vào cho mỗi LUT. Trong kiểu hoạt động bình thường thì bạn bị giới hạn là các hàm SOP chỉ có tối đa là 6 biến.

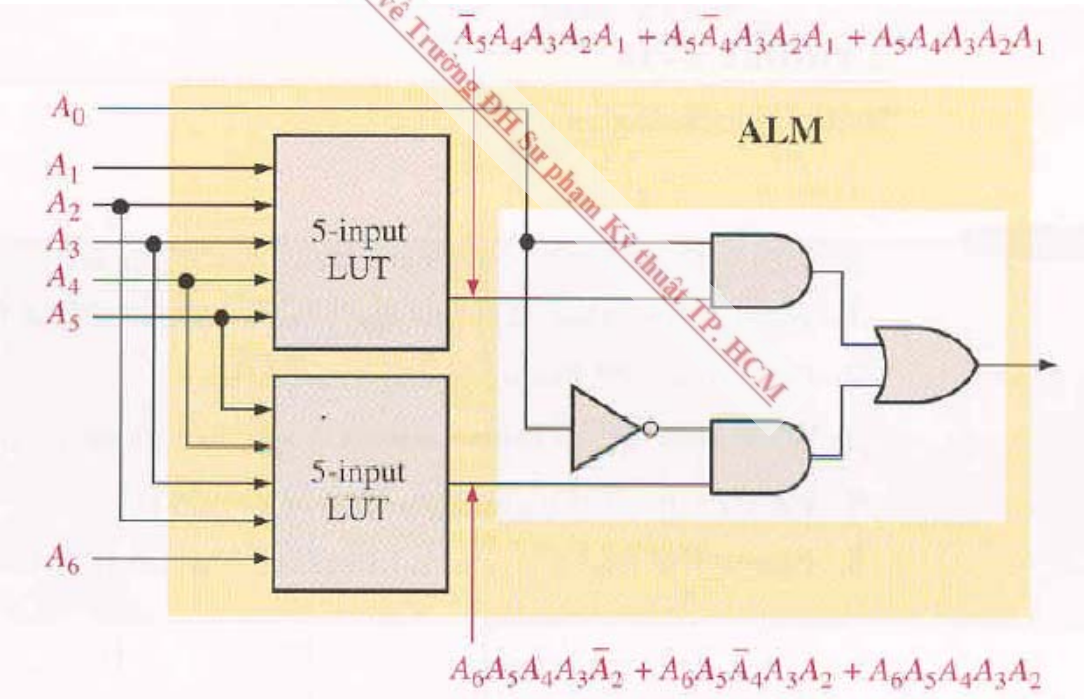
b. Kiểu hoạt động LUT mở rộng

Cho phép mở rộng hàm lên đến 7 biến được minh họa ở hình 1-34. Mạch điện AND – OR với ngõ vào đảo là một ví dụ đơn giản của mạch dồn kênh. Mạch dồn kênh là một phần của mạch logic dùng riêng trong ALM.



Hình 1-34. Mở rộng ALM để tạo ra hàm SOP 7 biến trong kiểu LUT mở rộng.

Ví dụ 1-4: Một ALM trong FPGA Stratix II được định cấu hình hoạt động ở kiểu LUT mở rộng được trình bày ở hình 1-35. Hãy xác định biểu thức ngõ ra SOP.



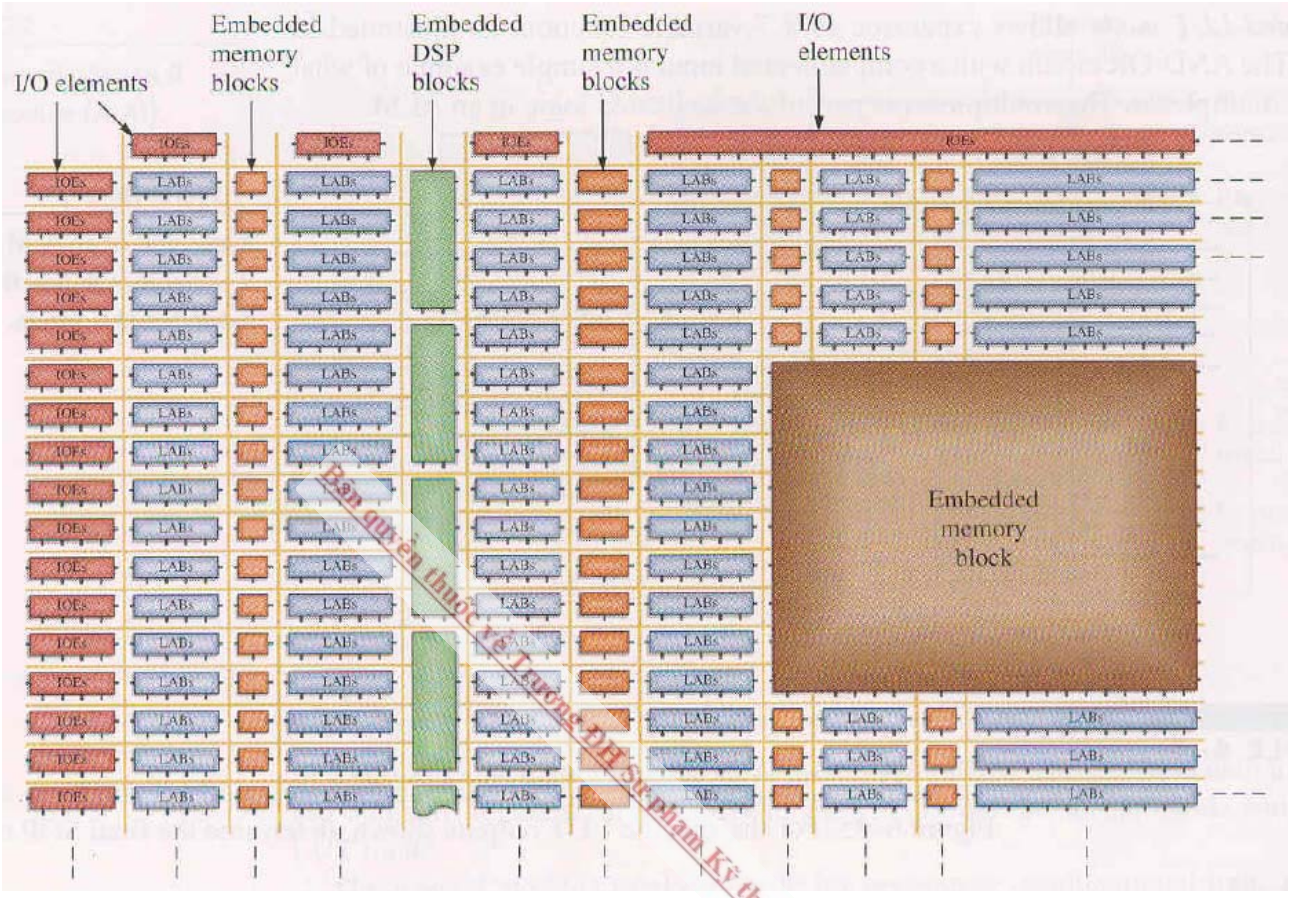
Hình 1-35. Minh họa cho ví dụ 1-4.

Giải: biểu thức ngõ ra ở trên thì AND với biến ngõ vào A_0 và biểu thức ngõ ra bên dưới thì AND với $\overline{A_0}$. Biểu thức sau cùng như sau:

$$\overline{A_5}A_4A_3A_2A_1A_0 + A_5\overline{A_4}A_3A_2A_1A_0 + A_5A_4A_3A_2A_1A_0 + A_6A_5A_4A_3\overline{A_2}\,\overline{A_0} + A_6A_5\overline{A_4}A_3A_2\overline{A_0} + A_6A_5A_4A_3A_2\overline{A_0}$$

3. CÁC CHỨC NĂNG TÍCH HỢP

Sơ đồ khối tổng quát của FPGA Stratix II được trình bày ở hình 1-36. FPGA chứa các thành phần bộ nhớ và chức năng xử lý tín hiệu số DSP. Chức năng của DSP như các mạch lọc số thường được sử dụng nhiều trong các hệ thống. Khi quan sát sơ đồ khối, các khối tích hợp bên trong được sắp xếp ở khắp nơi trong ma trận kết nối bên trong của FPGA và các phần tử ngõ vào/ngõ ra được đặt xung quanh chu vi FPGA.



Hình 1-36. Sơ đồ khối của FPGA Stratix II.

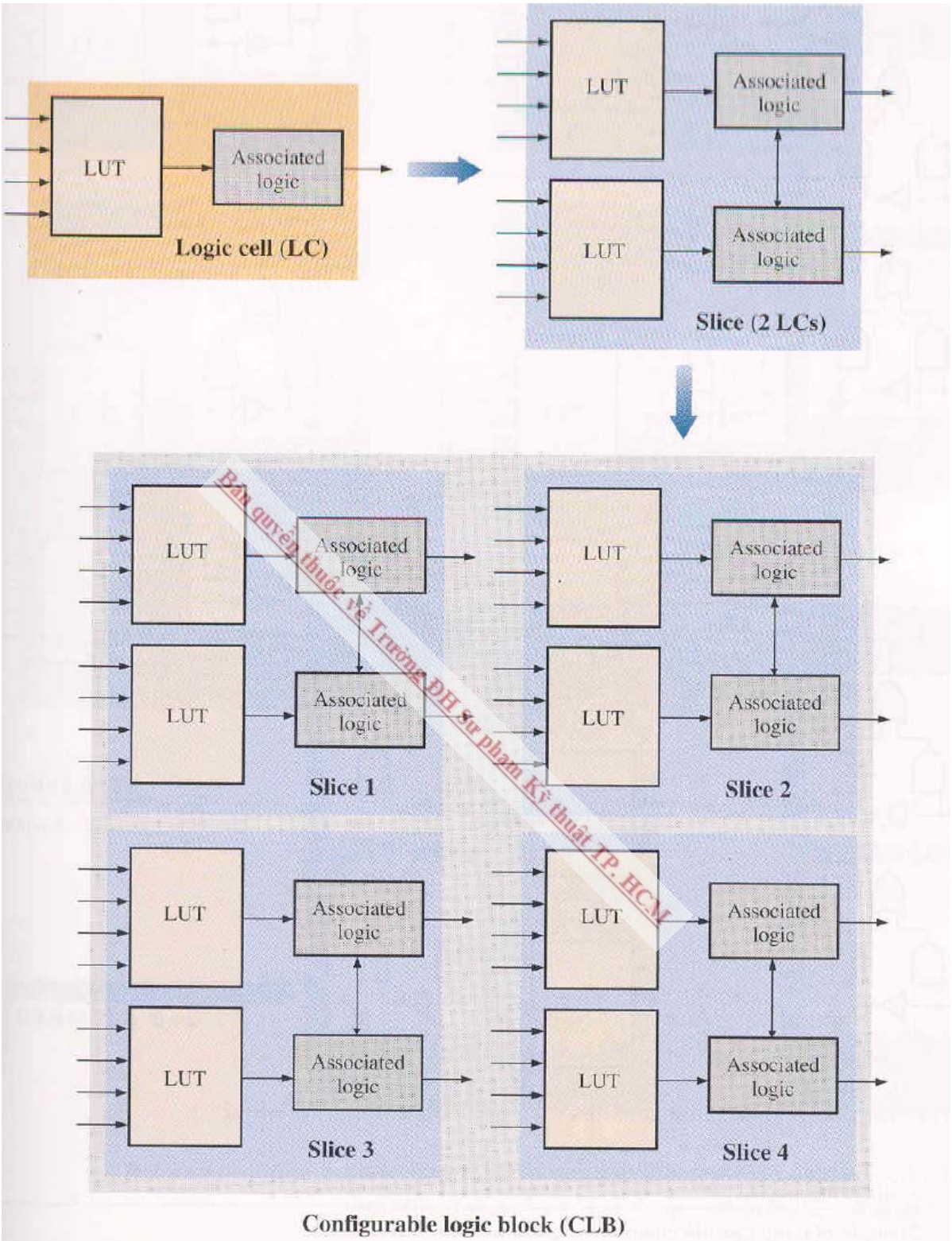
VI. FPGA CỦA XILINX

Xilinx có 2 họ FPGA chính là Spartan và Virtex và có nhiều loại khác nhau trong mỗi họ. Ví dụ Spartan 3 và Spartan IIE, Virtex-4, Virtex II và Virtex II Pro X. Xilinx định rõ Virtex-4, Virtex II và Virtex II Pro X là các FPGA loại platform (nền) bởi vì chúng tích hợp nhiều chức năng như bộ nhớ, vi xử lý, bộ thu phát và các phần cứng khác và các lõi phần mềm IP. Các họ FPGA thường thì khác về mật độ tích hợp và các thông số kỹ thuật. Hầu hết các thiết bị của Xilinx có cấu trúc FPGA truyền thống, tuy nhiên Virtex II Pro X có cái gọi là cấu trúc khối module chỉ định ứng dụng ASMBL (Application Specific Modular Block – được phát âm là assemble) có trên 1 tỉ transistor trong 1 chip đơn.

1. CÁC KHỐI LOGIC CÓ THỂ ĐỊNH CẤU HÌNH CLB (CONFIGURABLE LOGIC BLOCK)

Vùng logic định cấu hình của hầu hết các FPGA họ Xilinx được chia thành nhiều khối logic có thể định cấu hình CLB với mỗi CLB chứa nhiều đơn vị logic cơ bản được là các tế bào logic (logic cell - LC). Mỗi tế bào logic LC sử dụng mạch logic LUT truyền thống có 4 ngõ vào và thêm mạch logic cộng và một flip flop. Một LUT có 4 ngõ vào có thể tạo ra từ một thành phần tích cho đến hàm SOP chứa 16 thành phần tích. Hai tế bào logic LC giống nhau được gọi là ***slice (lát mỏng)***. Hình 1-37 minh họa các cấp logic định cấu hình từ tế bào logic cho đến

CLB. Mật độ tích hợp nằm trong khoảng từ 2000 đến 74000 tế bào logic LC trong một thiết bị Virtex đơn.

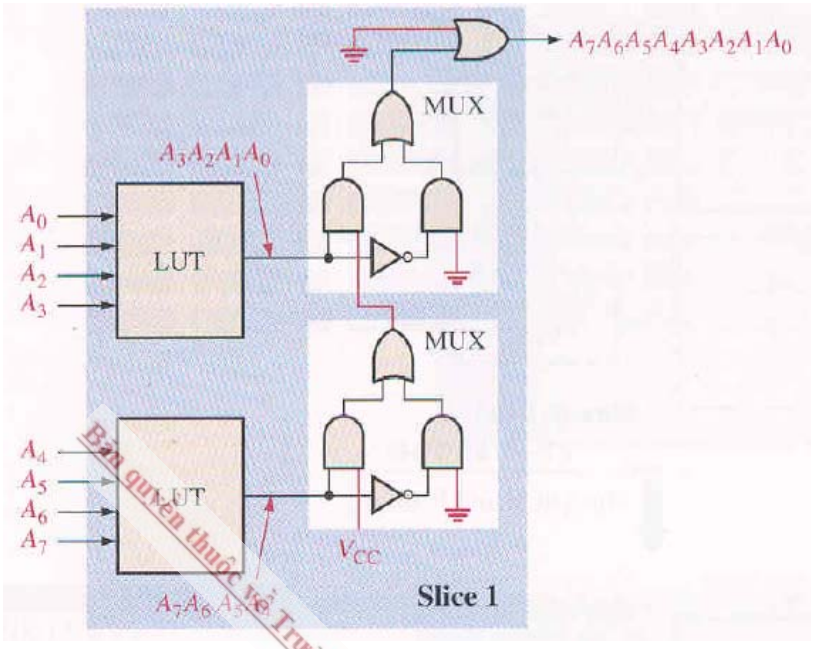


Hình 1-37. Minh họa các cấp logic định cấu hình từ tế bào logic cho đến CLB.

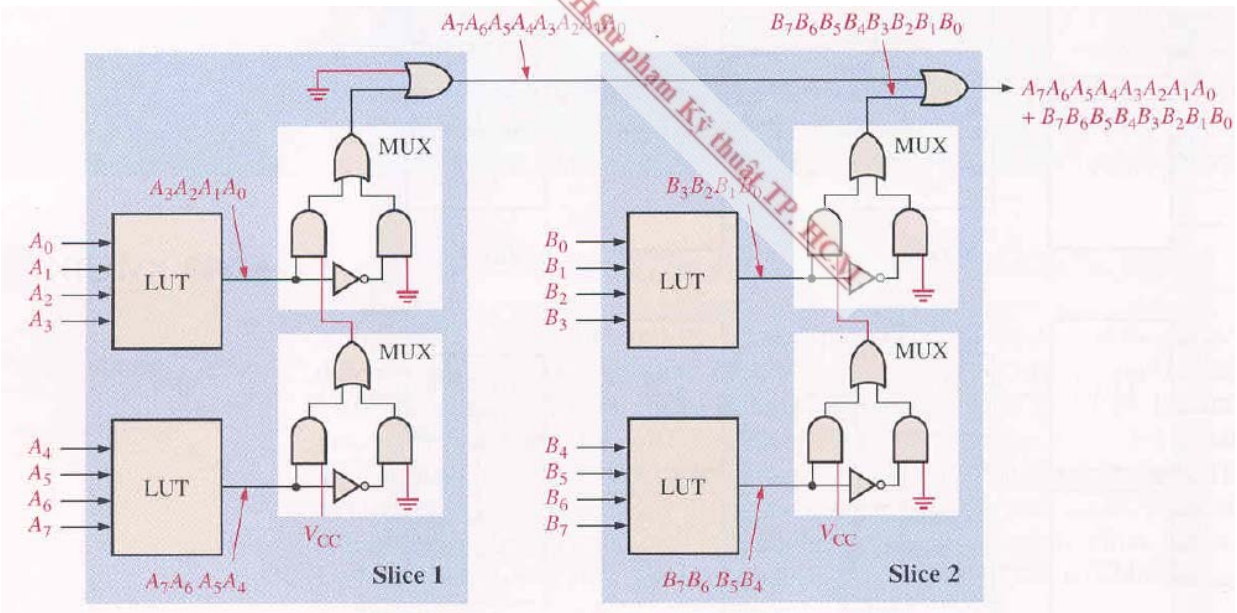
2. CHUỖI LIÊN TIẾP SOP

Slice đơn giản (hai tế bào logic LC) với logic chuỗi liên tiếp được trình bày ở hình 1-38. Có mạch đa hợp (MUX) dành riêng nằm trong mạch logic kết hợp của mỗi LC – được dùng trong chuỗi liên tiếp và một cổng OR dành riêng nằm trong slice.

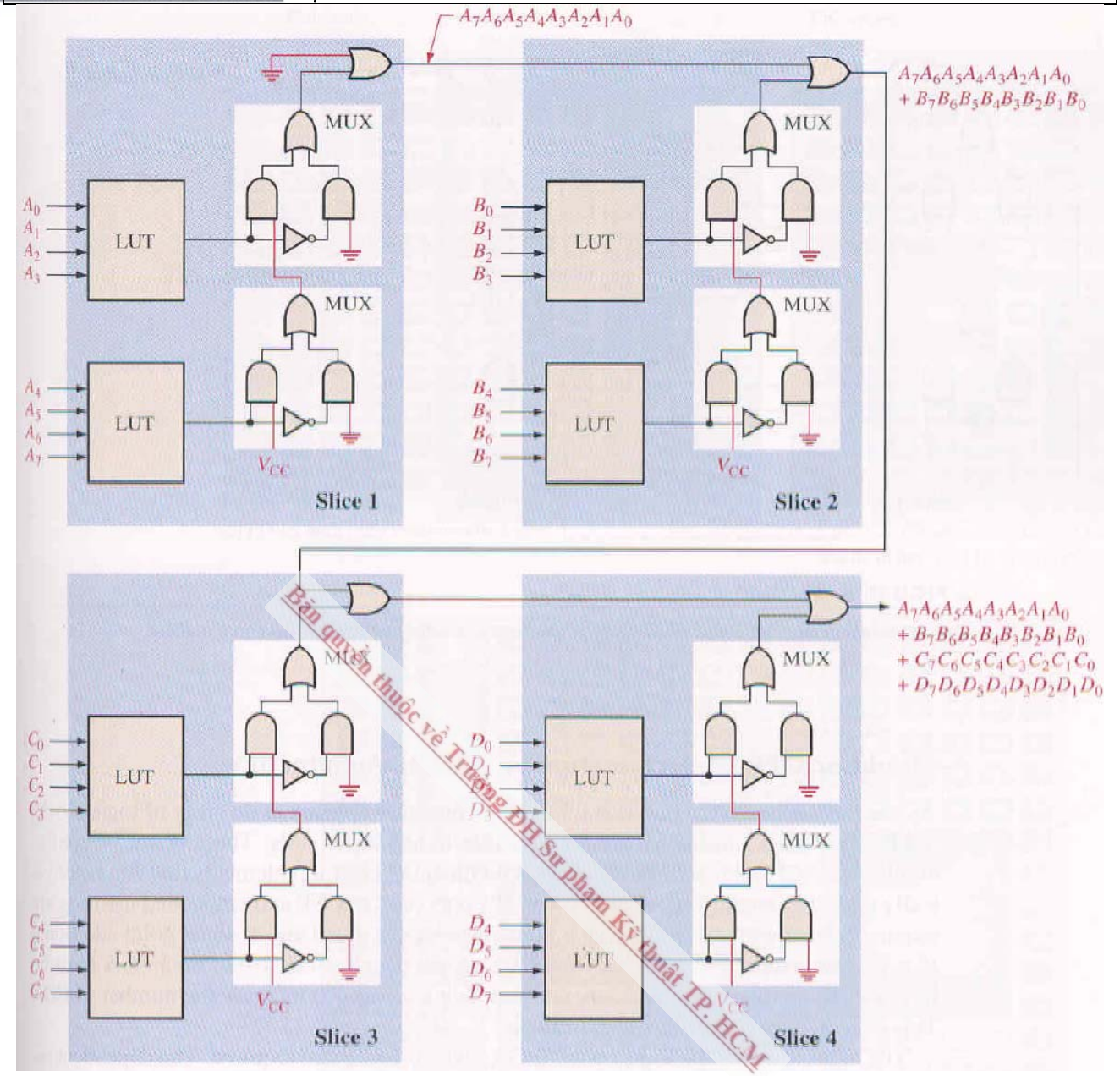
Hình 1-38a trình bày ví dụ cách mà một slice trong CLB có thể được định cấu hình như là một cổng AND để tạo ra thành phần tích 8 biến. Hai slice có thể được định cấu hình để tạo ra một hàm SOP với 2 thành phần tích 8 biến được trình bày ở hình 1-38b. Toàn bộ CLB của 4 slice có thể được định cấu hình thành một chuỗi liên tiếp để tạo ra một hàm SOP với 4 thành phần tích có 8 biến được trình bày như hình 1-38c. Biểu thức SOP dài hơn nữa có thể được thực hiện dùng thêm các CLB mở rộng.



(a)



(b)

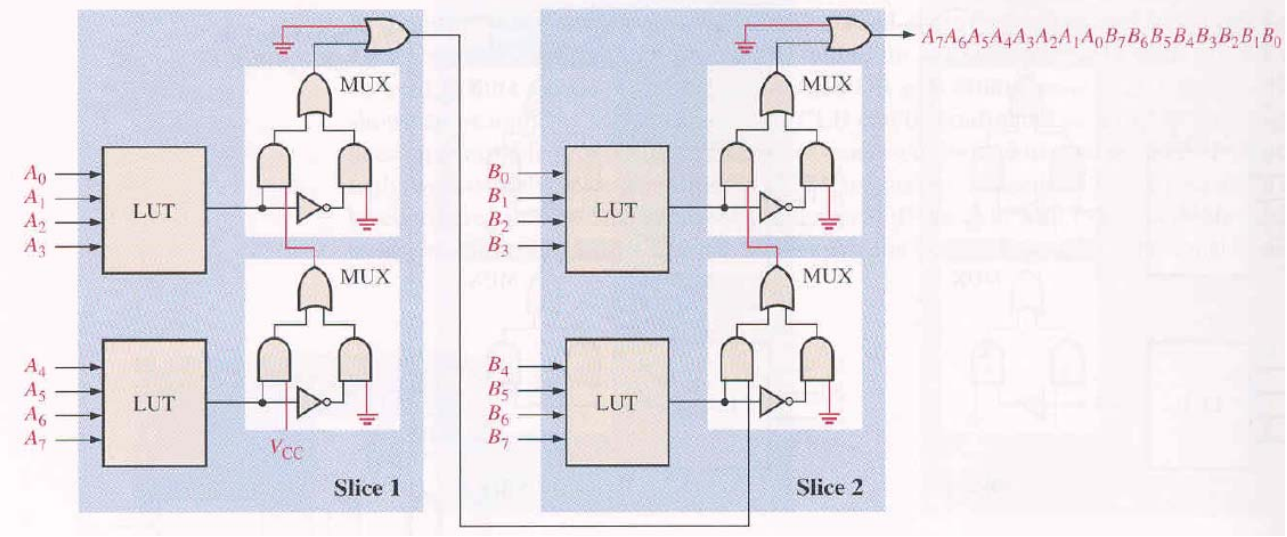


(c)

Hình 1-38. Ví dụ cách dùng chuỗi nối tiếp để mở rộng biểu thức SOP.

Ví dụ 1-5: Hãy trình bày cách cổng AND có 16 ngõ vào có thể tạo ra biểu thức trong CLB.

Giải: Hai slice được định cấu hình được trình bày ở hình 1-39 là kết quả của cổng AND có 16 ngõ vào.

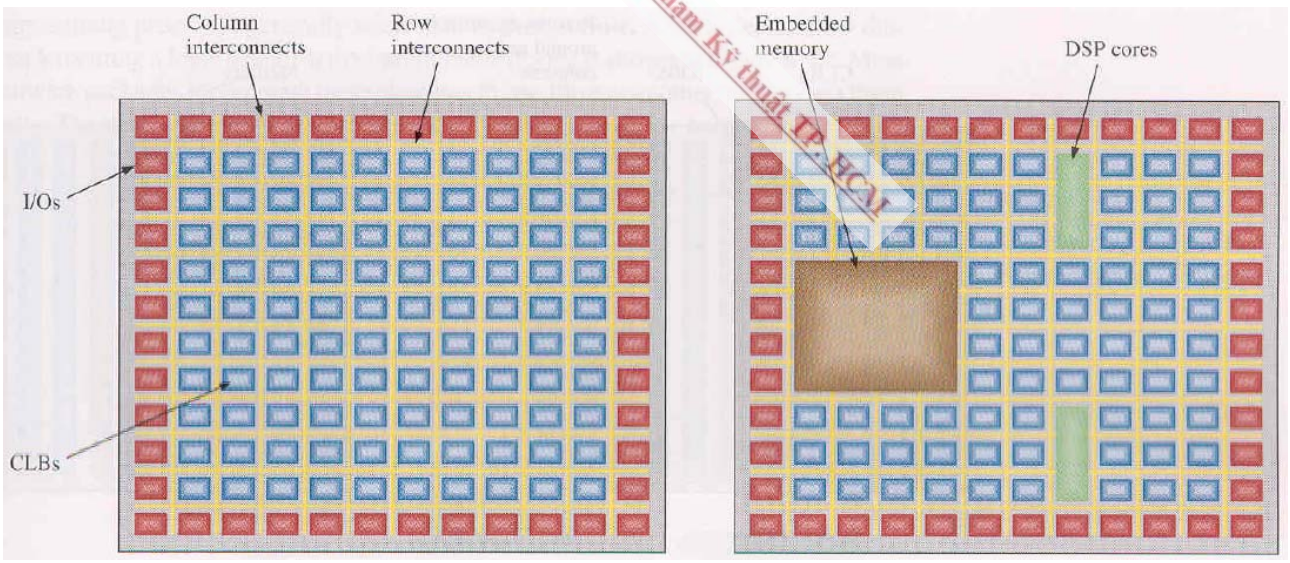


Hình 1-39. Minh họa cho ví dụ 1-5.

3. CẤU TRÚC FPGA TRUYỀN THỐNG VÀ CẤU TRÚC ASMBL

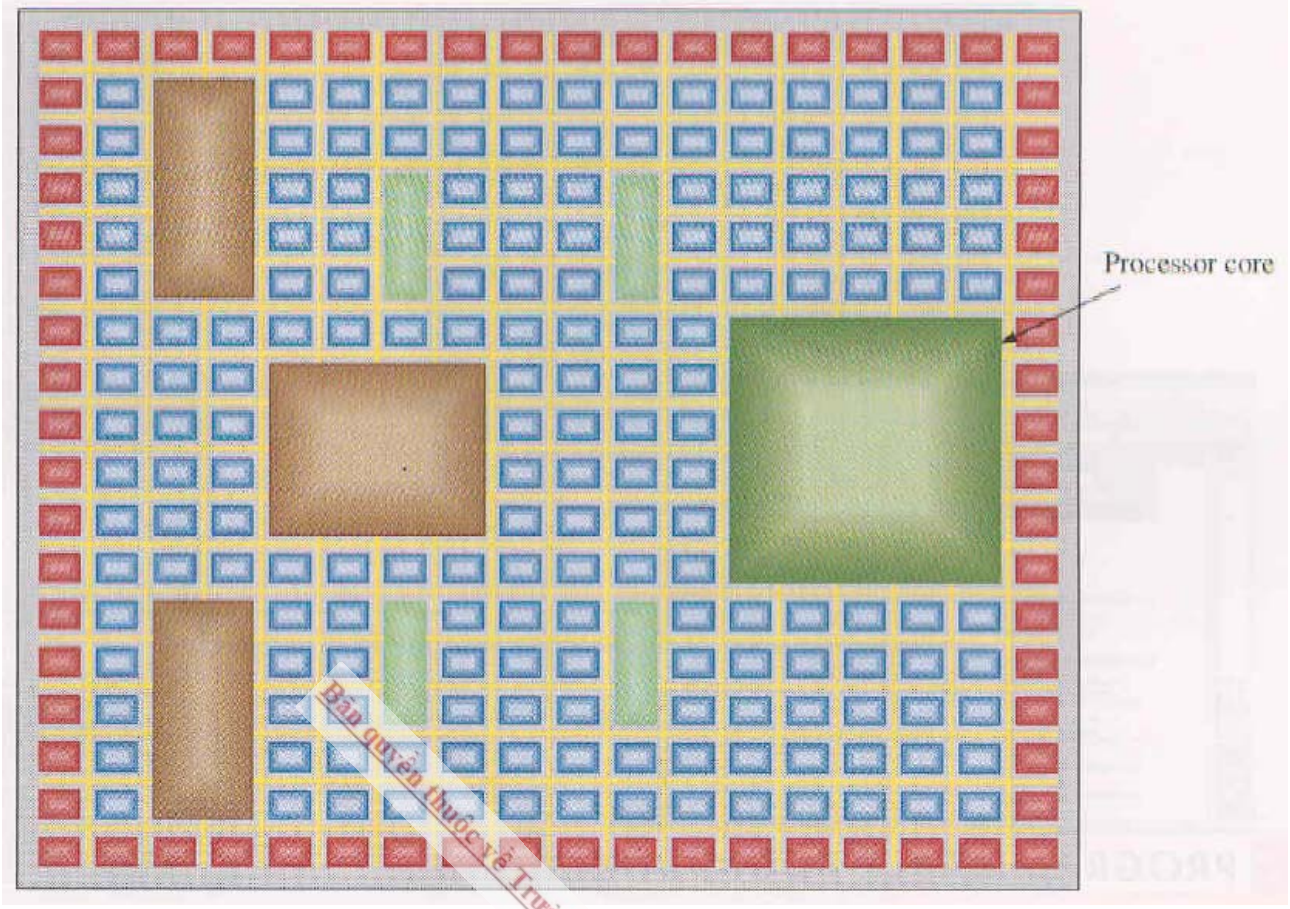
a. *Cấu trúc truyền thống*

Như đã biết, cấu trúc FPGA truyền thống xuất hiện như là một mảng của các khối logic (CLB hoặc LAB) được bao bọc xung quanh bởi các tế bào ngõ vào/ngõ ra có thể định cấu hình. Số LCB trong FPGA tuỳ thuộc vào số lượng các phần tử IO – có thể đặt xung quanh. Khi lõi IP như DSP và bộ nhớ tích hợp bên trong khi được yêu cầu thì một lượng logic định cấu hình phải mất đi và tại một vài vị trí được thay thế bằng IO nếu có yêu cầu. Khi nhiều lõi IP được thêm vào thì kích thước vật lý của FPGA phải tăng lên để đảm bảo số lượng logic cấu hình cần thiết và tăng thêm số lượng IO. Khái niệm này được minh họa bằng hình 1-40.



(a) FPGA với logic định hình đầy đủ.

(b) FPGA cùng kích thước với bộ nhớ và lõi IP (DSP) nên có ít CLB hơn.



(c) FPGA có nhiều bộ nhớ, thêm lõi DSP và lõi vi xử lý sẽ yêu cầu kích thước lớn hơn.

Hình 1-40. Tích hợp nhiều chức năng IP kết quả làm giảm CLB và/hoặc phải tăng kích thước chip.

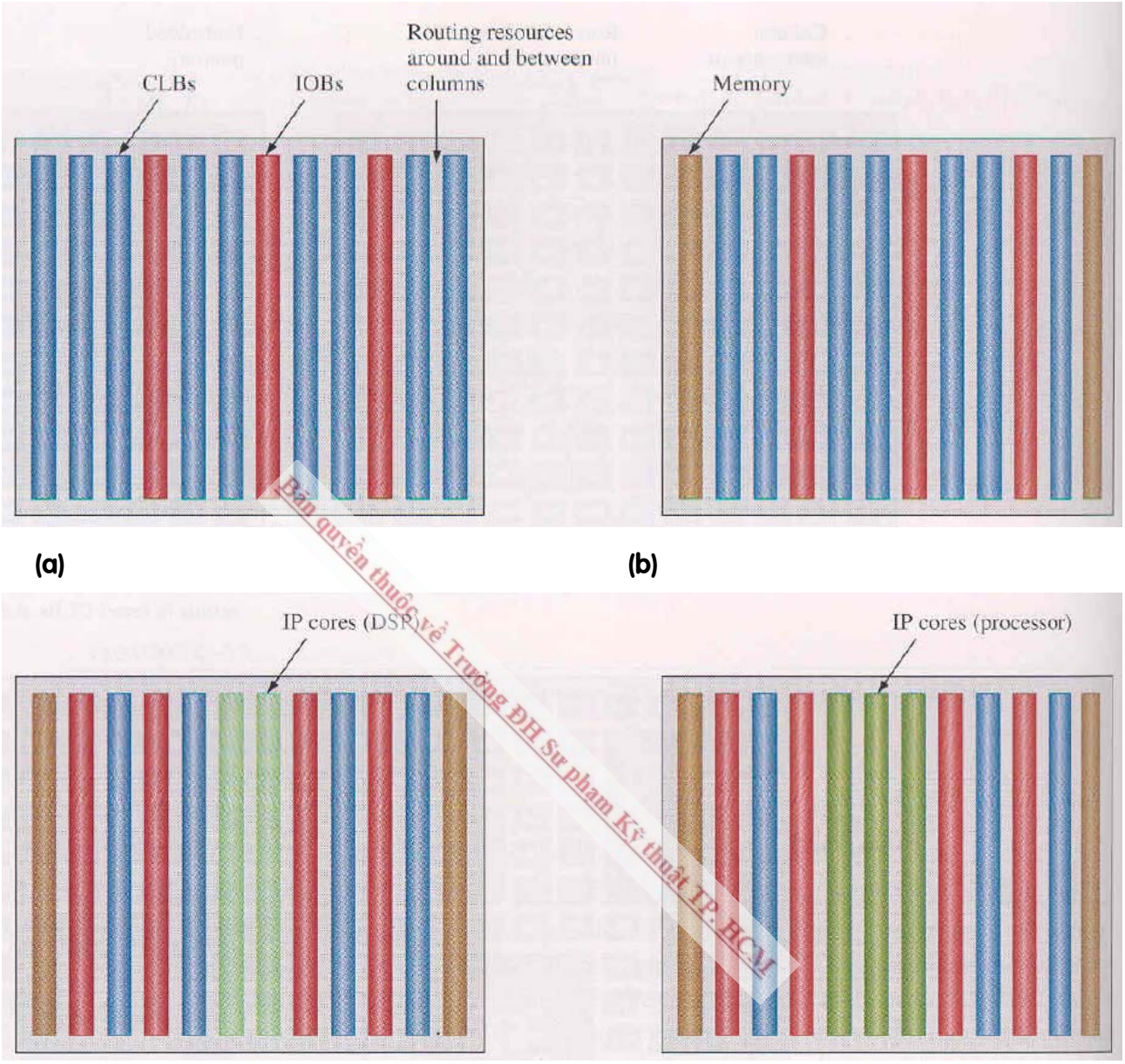
Logic cấu hình trong FPGA càng phức tạp thì càng dùng nhiều IO. Mối liên hệ ràng buộc giữa logic và IO sẽ dẫn đến tăng kích thước chip và tăng giá thành. Ngoài ra một vấn đề khác với FPGA platform là khi thêm các chức năng lõi IP tích hợp bên trong nếu có yêu cầu thì phải thiết kế lại thành phần chính hoặc thiết kế lại một phần trong cách bố trí chip (layout) có thể được yêu cầu sẽ làm tăng thêm giá thành.

a. Cấu trúc ASMBL

Xilinx đã xây dựng một phương pháp mềm dẻo cho FPGA platform ở chip Virtex II Pro X để khắc phục một vài hạn chế xuất hiện trong cấu trúc truyền thống. Cấu trúc ASMBL là cấu trúc sử dụng cột thay vì dùng cấu trúc hàng/cột. Các IO được đặt rải rác khắp nơi tốt hơn là đặt xung quanh, dẫn đến số lượng IO của nó tăng mà không cần làm tăng kích thước chip. Mỗi cột về cơ bản là một dải logic có thể được thay thế bằng dải logic khác mà không cần thiết kế lại cách bố trí chip. Các ví dụ về các loại của các dải logic là các khối logic định cấu hình CLB, khối IO, bộ nhớ và các lõi phần cứng và phần mềm như DSP và vi xử lý.

Số lượng khác nhau của mỗi loại dải logic có thể được trộn lại để tương thích với các yêu cầu ứng dụng riêng biệt. Ví dụ, trong cấu hình đơn giản nhất thì có thể pha trộn các dải CLB và các dải khối IO được minh họa như hình 1-41a. Nhiều hoặc ít hơn của cả 2 cũng có thể được sử dụng tuỳ thuộc vào các yêu cầu.

Nếu cần nhiều bộ nhớ thì một hoặc nhiều dải CLB có thể được thay thế như hình 1-41b. Nếu vùng riêng biệt trong ứng dụng là xử lý tín hiệu số thì có thể thêm vào các lõi IP DSP trộn với bộ nhớ như hình 1-41c. Hình 1-41d trình bày các lõi vi xử lý được thêm vào.



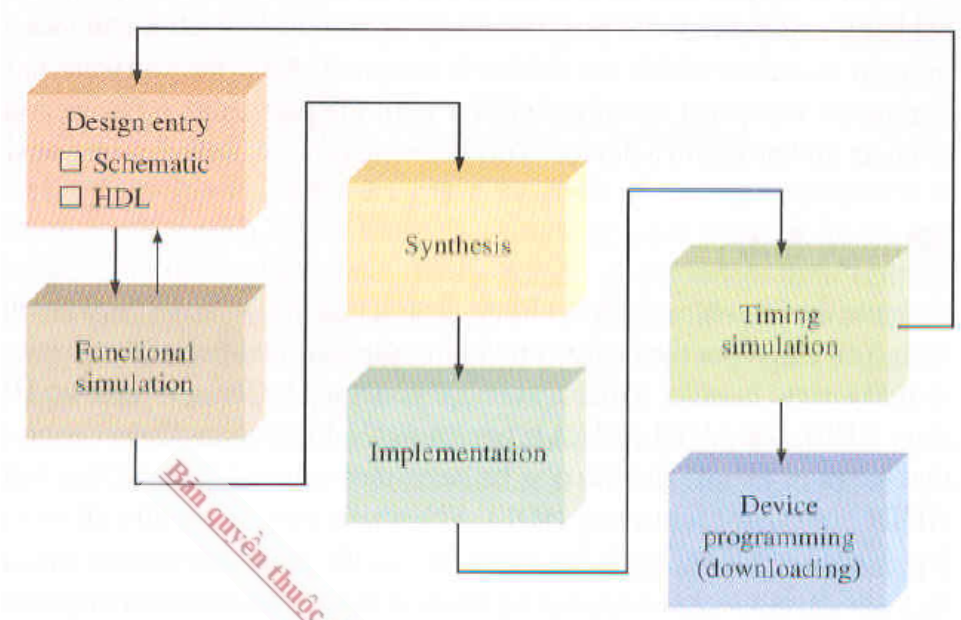
Hình 1-41. Minh họa cấu trúc ASMBL của FPGA platform.

VII. PHẦN MỀM LẬP TRÌNH

Để sử dụng thiết bị logic lập trình thì phải có phần cứng và phần mềm kết hợp với nhau. Tất cả các nhà chế tạo SPLD, CPLD và FPGA cung cấp phần mềm hỗ trợ cho mỗi thiết bị phần cứng. Các gói phần mềm nằm trong danh sách phần mềm được dùng để thiết kế dưới sự giúp đỡ của máy tính – CAD. Trong phần này phần mềm lập trình được giới thiệu một cách tổng quát.

Sau khi kết thúc phần này bạn có thể: giải thích quy trình lập trình cho các thành phần của thiết kế, mô tả giai đoạn thiết kế, mô tả giai đoạn mô phỏng chức năng, mô tả giai đoạn tổng hợp, mô tả giai đoạn thi hành, mô tả mô phỏng theo thời gian, mô tả cách tải chương trình.

Quy trình lập trình thiết kế được xem như là dòng thiết kế (***design flow***). Giản đồ dòng thiết kế cơ bản dùng để thực hiện thiết kế logic cho thiết bị lập trình được trình bày như hình 1-42. Hầu hết các gói phần mềm riêng lẻ sẽ kết hợp các công đoạn của quy trình lại với nhau và quá trình xử lý hoàn toàn tự động. Thiết bị để được lập trình thường được xem là thiết bị đích (***target device***)



Hình 1-42. Sơ đồ dòng thiết kế tổng quát để lập trình cho SPLD, CPLD hoặc FPGA.

Phải có 4 thiết bị để có thể lập trình cho thiết bị là: máy tính, phần mềm lập trình, thiết bị logic lập trình (SPLD, CPLD hoặc FPGA) và thiết bị kết nối máy tính với thiết bị lập trình (cáp hoặc mạch nạp). Tất cả các thành phần này được minh họa như hình 1-43.

(a) Computer (b) Software (CD or Website download)

(c) Device (d) Programming hardware (programming fixture or development board with cable for connection to computer port)

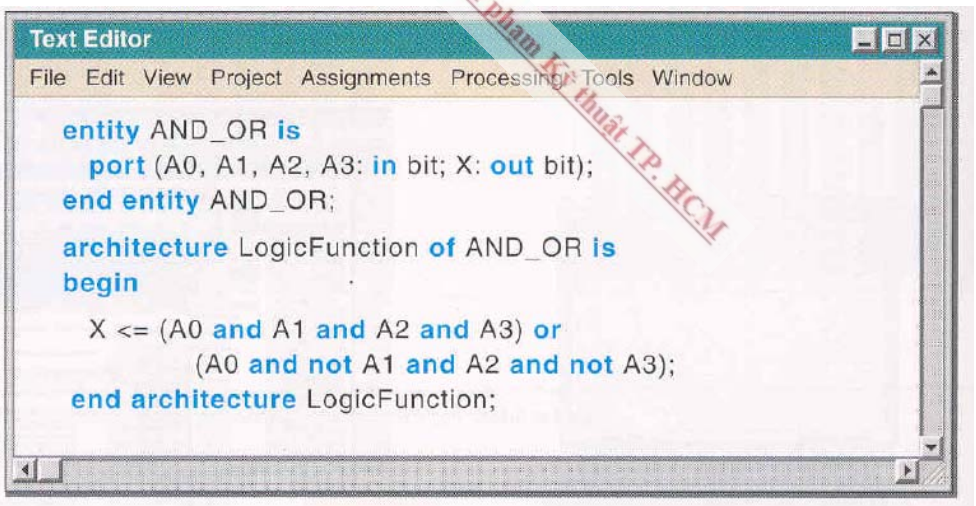
Hình 1-43. Các thiết bị cơ bản để lập trình cho SPLD, CPLD hoặc FPGA.

1. CÁCH THIẾT KẾ

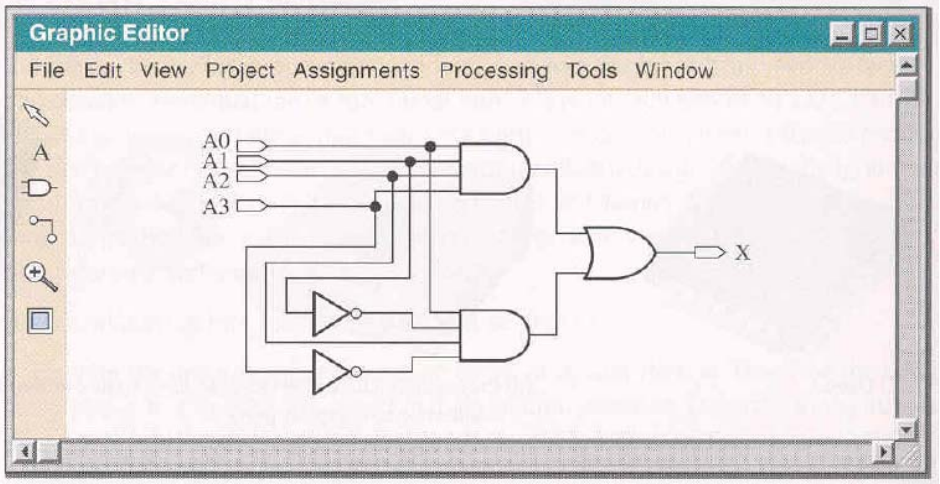
Giả sử rằng chúng ta có một thiết kế mạch điện logic muốn điều khiển bằng thiết bị lập trình thì chúng ta có thể thiết kế trên máy tính bằng một trong hai cách cơ bản: thiết kế dùng sơ đồ nguyên lý (***schematic entry***) và cách dùng ngôn ngữ (***text entry***).

Để dùng cách thiết kế bằng ngôn ngữ thì phải làm quen với ngôn ngữ HDL như VHDL, Verilog, ABEL hoặc AHDL. Hầu hết các nhà chế tạo thiết bị lập trình cung cấp các gói phần mềm hỗ trợ ngôn ngữ VHDL và Verilog bởi vì chúng là ngôn ngữ HDL chuẩn. Nhiều nhà chế tạo còn cung cấp thêm ngôn ngữ ABEL, AHDL.

Kiểu thiết kế dùng sơ đồ mạch cho phép chúng ta đặt các kí hiệu của các cổng logic và các chức năng logic khác từ thư viện lên màn hình và kết nối chúng theo yêu cầu của thiết kế. Với kiểu thiết kế này thì cần biết các ngôn ngữ HDL. Hình 1-44 minh họa cho cả 2 kiểu thiết kế cho một mạch điện logic AND-OR đơn giản.



(a) Text entry using VHDL to describe an AND-OR logic circuit



(b) Schematic entry of the same AND-OR logic circuit entered in (a)

Hình 1-44. Minh họa cho 2 kiểu lập trình.

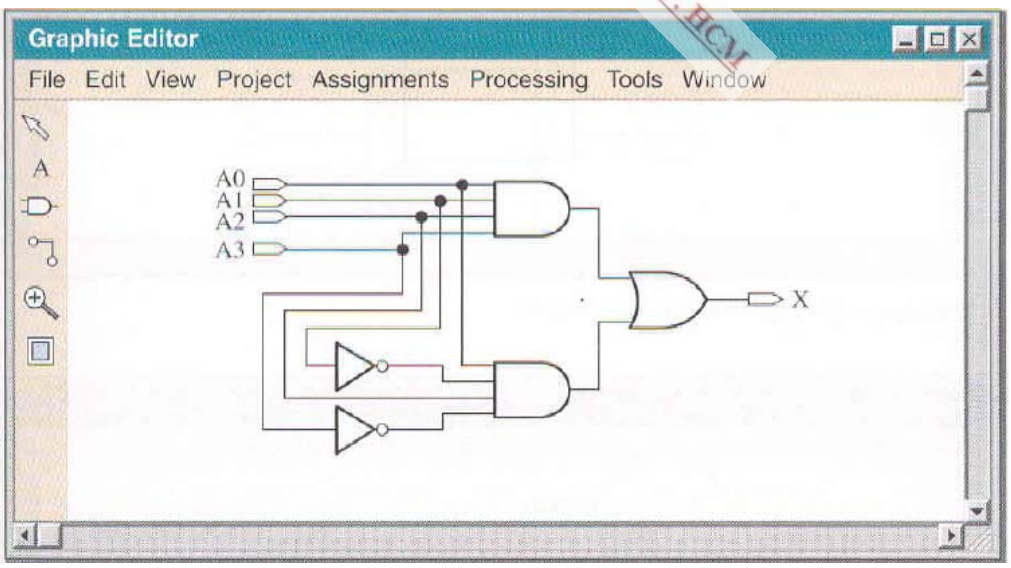
Xây dựng sơ đồ logic:

Khi xây dựng mạch điện logic đầy đủ trên màn hình thì nó được gọi là sơ đồ phẳng "flat". Các mạch điện logic phức tạp hơn thì khó mà tương thích với màn hình. Chúng ta có thể thiết kế mạch điện logic thành nhiều đoạn (segment), lưu trữ mỗi đoạn như là một kí hiệu khối và sau đó kết nối các kí hiệu khối lại với nhau để tạo thành một mạch điện hoàn chỉnh – được gọi là thiết kế có thứ tự.

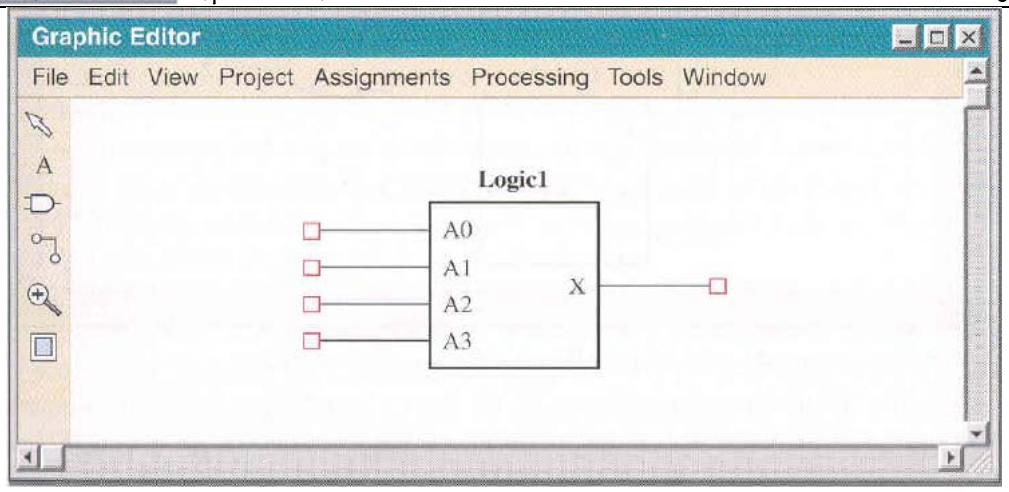
Ví dụ thiết kế mạch điện có biểu thức SOP như sau:

$$Z = \left(A_3A_2A_1A_0 + \overline{A_3}A_2\overline{A_1}A_0\right) + \left(A_3\overline{A_2}A_1A_0 + A_3\overline{A_2}A_1\overline{A_0} + \overline{A_3}A_2A_1\underline{A_0}\right)$$

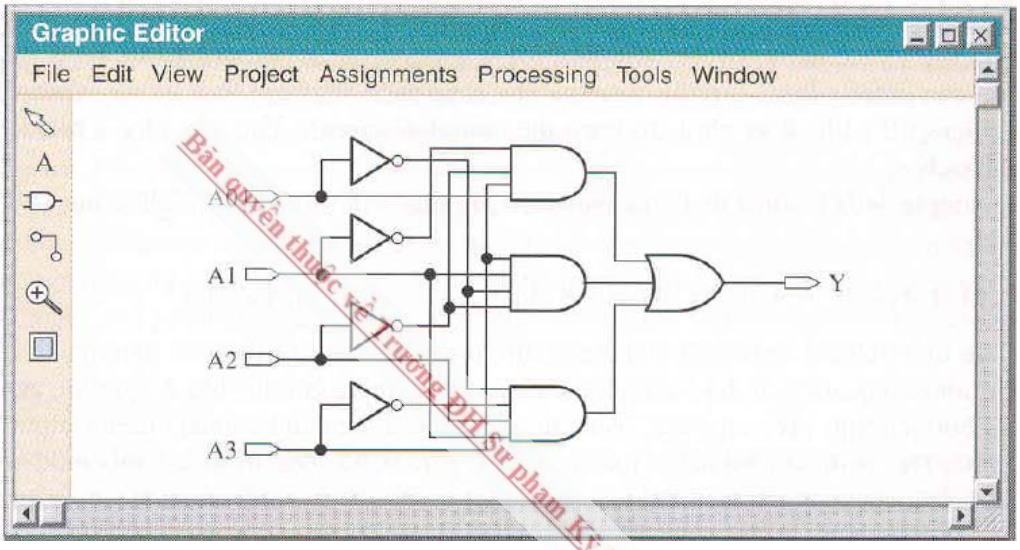
Chúng ta dùng phương pháp thiết kế có thứ tự và xây dựng mạch logic cho 2 thành phần tổng trong phương trình, làm đơn giản mỗi mạch điện logic bằng một kí hiệu duy nhất, sau khi thiết kế xong cả 2 mạch điện thì đặt chúng lên màn hình và kết nối các ngõ ra với cổng OR để tạo thành mạch hoàn chỉnh – tất cả được minh họa bằng hình 1-45.



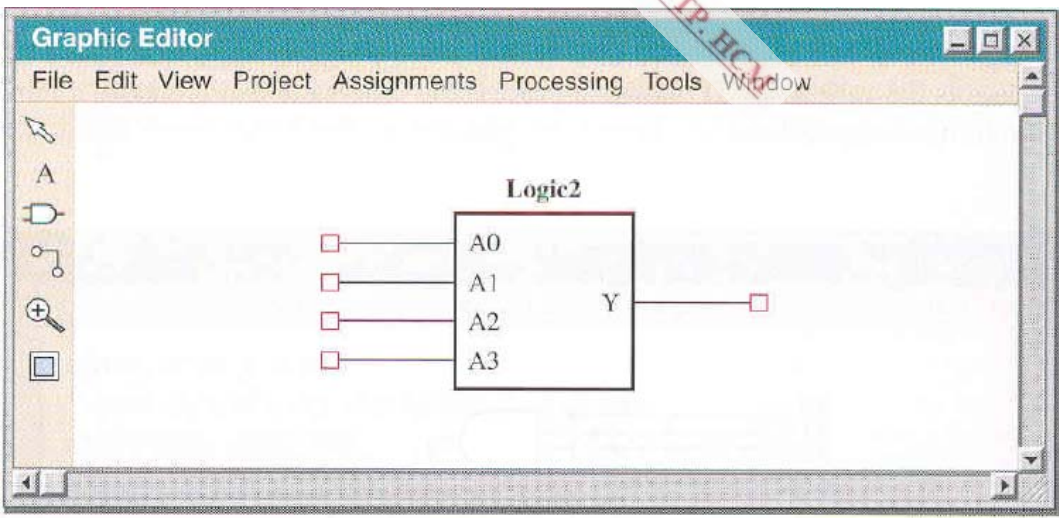
a. Thiết kế thành phần thứ 1 gồm tổng của 2 tích.



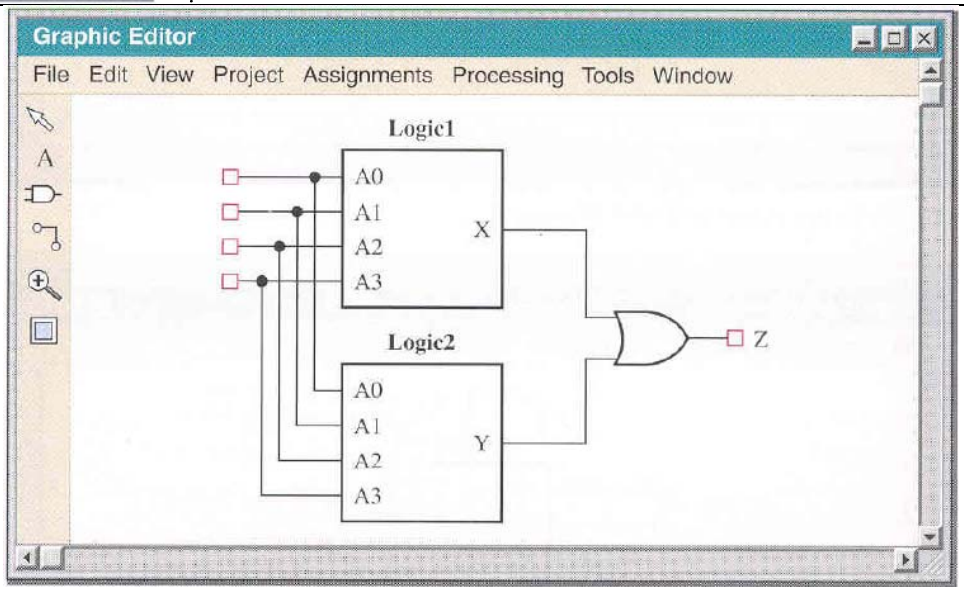
b. Làm đơn giản khối mạch điện bằng kí hiệu logic 1.



c. Thiết kế thành phần thứ 2 gồm tổng của 3 tích.



d. Làm đơn giản khối mạch điện bằng kí hiệu logic 2.

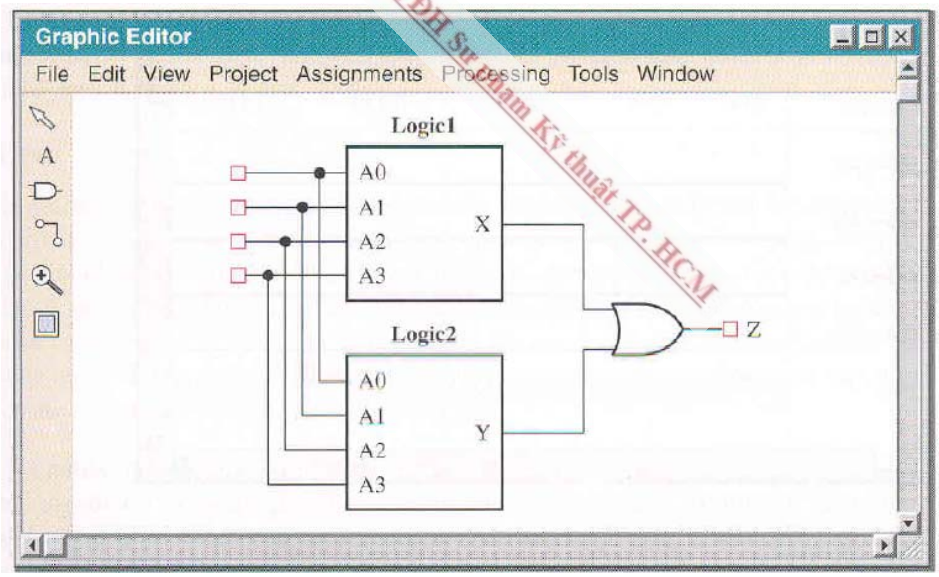


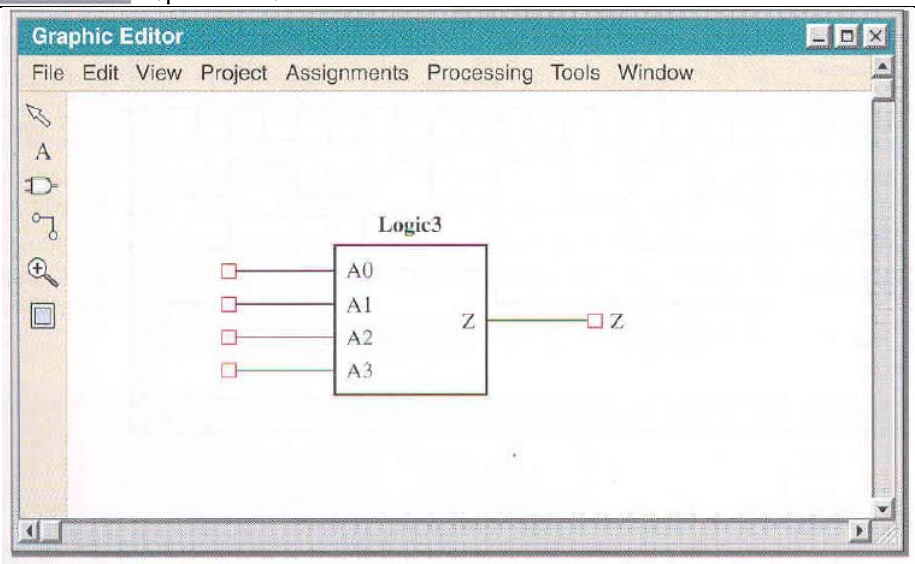
e. Kết nối 2 khối logic 1 và logic 2 bằng cổng OR.

Hình 1-45. Minh họa cho kiểu lập trình từng đoạn.

Toàn bộ mạch điện trên có thể đặt lên màn hình nhưng phương pháp thiết kế theo trình tự rất tiện lợi khi mạch điện logic lớn và phải chia ra thành nhiều phần.

Ở hình 1-46e, mạch điện logic có thể làm đơn giản bằng 1 kí hiệu khác và được sử dụng để thiết kế mạch điện lớn hơn hoặc có thể lưu và dùng lại cho các thiết kế khác được minh họa như hình 1-46.



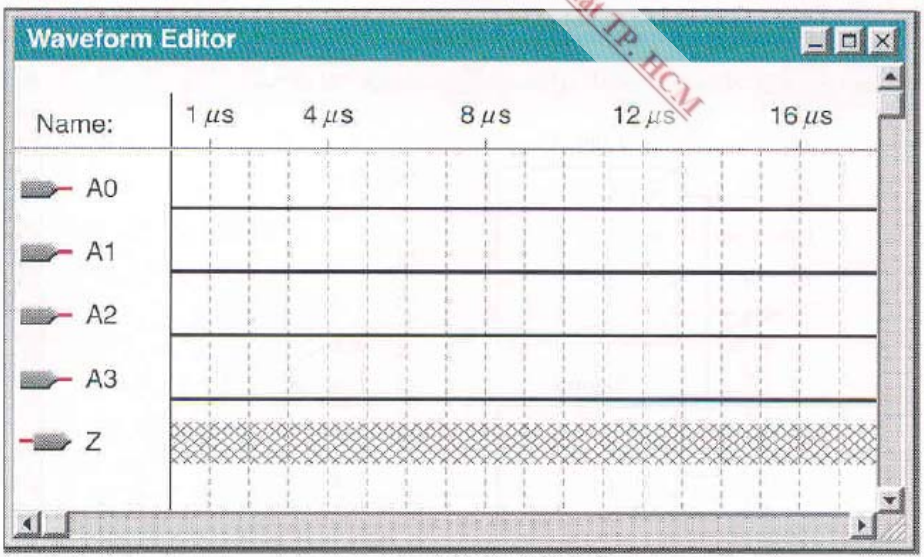


Hình 1-46. Lưu thành khối logic 3.

2. MÔ PHỎNG CHỨC NĂNG

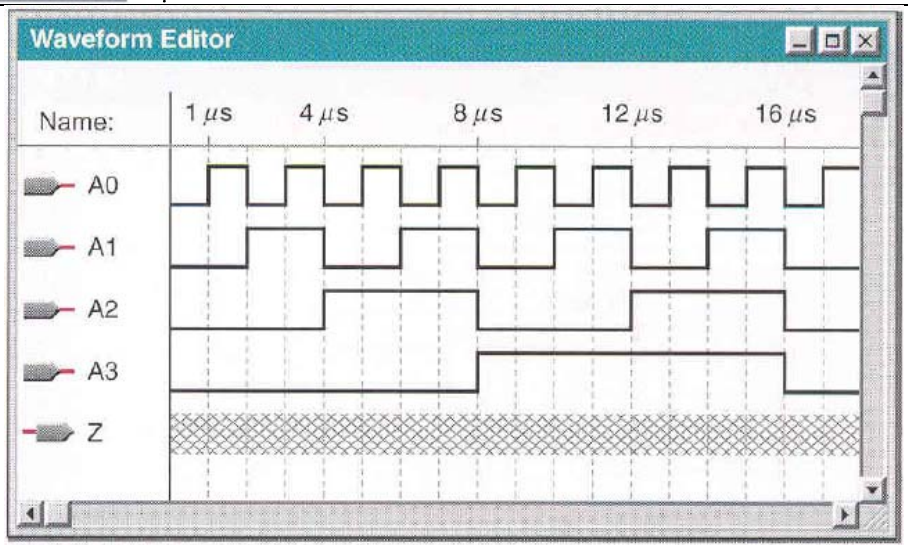
Mục đích của chức năng mô phỏng trong ***dòng thiết kế*** là để đảm bảo ***chắc chắn thiết kế hoạt động đúng theo yêu cầu*** trước khi tổng hợp thành thiết kế phần cứng. Về cơ bản sau khi mạch điện logic được biên dịch thì sau đó có thể mô phỏng bằng cách cung cấp các dạng sóng đầu vào và kiểm tra dạng sóng ngõ ra cho các tổ hợp ngõ vào có thể có dùng trình soạn thảo dạng sóng.

Trình soạn thảo dạng sóng cho phép lựa chọn các nút (các ngõ vào và các ngõ ra) muốn kiểm tra. Tên các ngõ vào và ngõ ra đã chọn xuất hiện trên màn hình soạn thảo dạng sóng bằng kí hiệu hoặc tên khác để xác định cho mỗi một ngõ vào hoặc một ngõ ra – được trình bày ở hình 1-47. Khi bắt đầu thì tất cả các ngõ vào mặc nhiên ở mức 0 và các đường chéo song song tượng trưng cho tín hiệu chưa xác định. Có thể lựa chọn các khoảng thời gian để hiển thị.



Hình 1-47. Màn hình soạn thảo dạng sóng tổng quát .

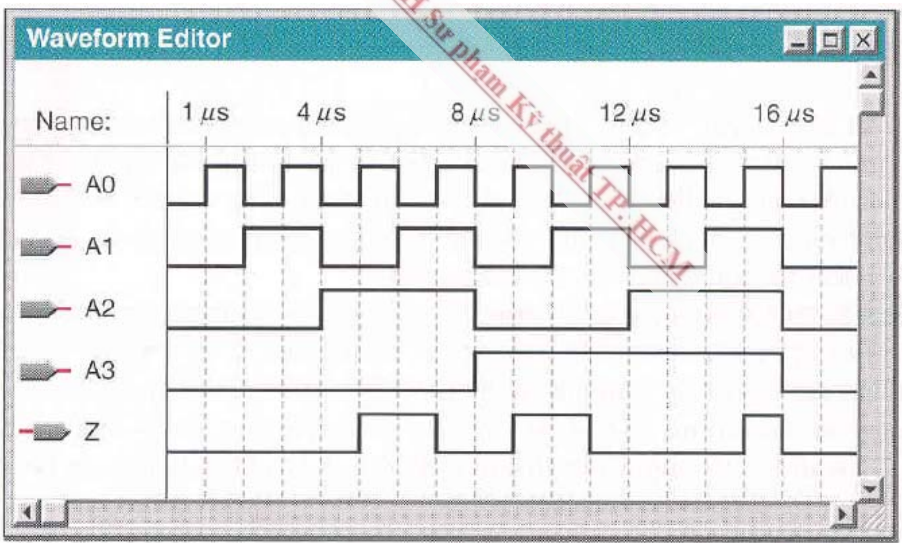
Bước tiếp theo chúng ta xây dựng dạng sóng cho mỗi ngõ vào bằng cách nhập vào 1 hoặc 0 cho mỗi khoảng thời gian. Hình 1-48 trình bày các dạng sóng ngõ vào.



Hình 1-48. Thiết lập các dạng sóng ngõ vào.

Sau khi thiết lập các dạng sóng ngõ vào thì mở cửa sổ điều khiển mô phỏng để thiết lập thời gian bắt đầu và thời gian kết thúc cho việc mô phỏng và chỉ định các khoảng thời gian hiển thị. Khi bắt đầu mô phỏng thì dạng sóng của tín hiệu Z sẽ được hiển thị trên màn hình dạng sóng như hình 1-49.

Kết quả dạng sóng ngõ ra Z của ví dụ này sẽ cho chúng ta biết thiết kế hoạt động đúng hay không đúng. Trong trường hợp này dạng sóng ngõ ra là đúng với dạng sóng ngõ vào đã chọn. Khi dạng sóng ngõ ra không đúng thì phải quay lại kiểm tra thiết kế ban đầu cho đến khi mạch hoạt động đúng.



Hình 1-49. Dạng sóng ngõ vào và ra khi chạy mô phỏng.

3. TỔNG HỢP

Mỗi khi mạch logic được xây dựng và được mô phỏng chức năng để kiểm tra đúng sai của mạch logic thiết kế thì phần mềm biên dịch đã tự động thực hiện một vài công đoạn để chuẩn bị cho việc nạp thiết kế vào cho thiết bị lập trình.

Trong ***công đoạn tổng hợp*** của ***dòng thiết kế*** thì thiết kế được tối ưu theo các thành phần để làm giảm số lượng cổng, thay thế các phần tử logic bằng các phần tử logic khác mà chúng có thể thực hiện cùng một chức năng nhưng hiệu quả hơn và loại trừ các thành phần logic thừa.

Ngõ ra cuối cùng từ công đoạn tổng hợp là liệt kê kết nối (netlist) – chúng diễn tả trạng thái tối ưu của mạch điện logic.

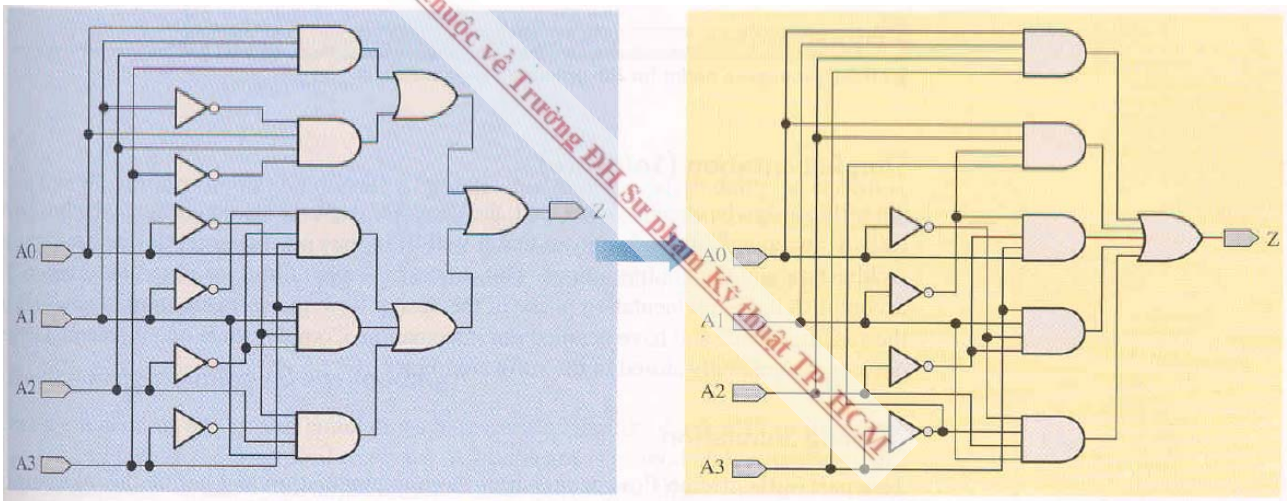
4. LIỆT KÊ LƯỚI (NETLIST)

Liệt kê lưới về cơ bản là một danh sách liệt lưới mà chúng mô tả các thành phần và cách kết nối với nhau. Tổng quát, liệt kê lưới chứa các tham chiếu mô tả các thành phần và các phần tử được sử dụng.

Mỗi lần một thành phần như cổng logic được sử dụng trong liệt kê lưới thì nó được gọi là ***instance***. Mỗi ***instance*** có xác định liệt kê các kết nối. Các điểm kết nối được gọi là các ***cảng*** (port) hoặc các ***chân (pin)***.

Thường thì mỗi instance sẽ có một tên duy nhất, ví dụ như nếu có 2 instance của các cổng AND thì một là "and1" và cổng còn lại là "and2". Ngoài tên ra còn có tên khác, các lưới là các đường dây – nối với nhau trong mạch điện. Bảng liệt kê các lưới thường mô tả tất cả các instance và các thuộc tính của chúng, sau đó mô tả từng lưới và đặt biệt là các port nối với mỗi instance.

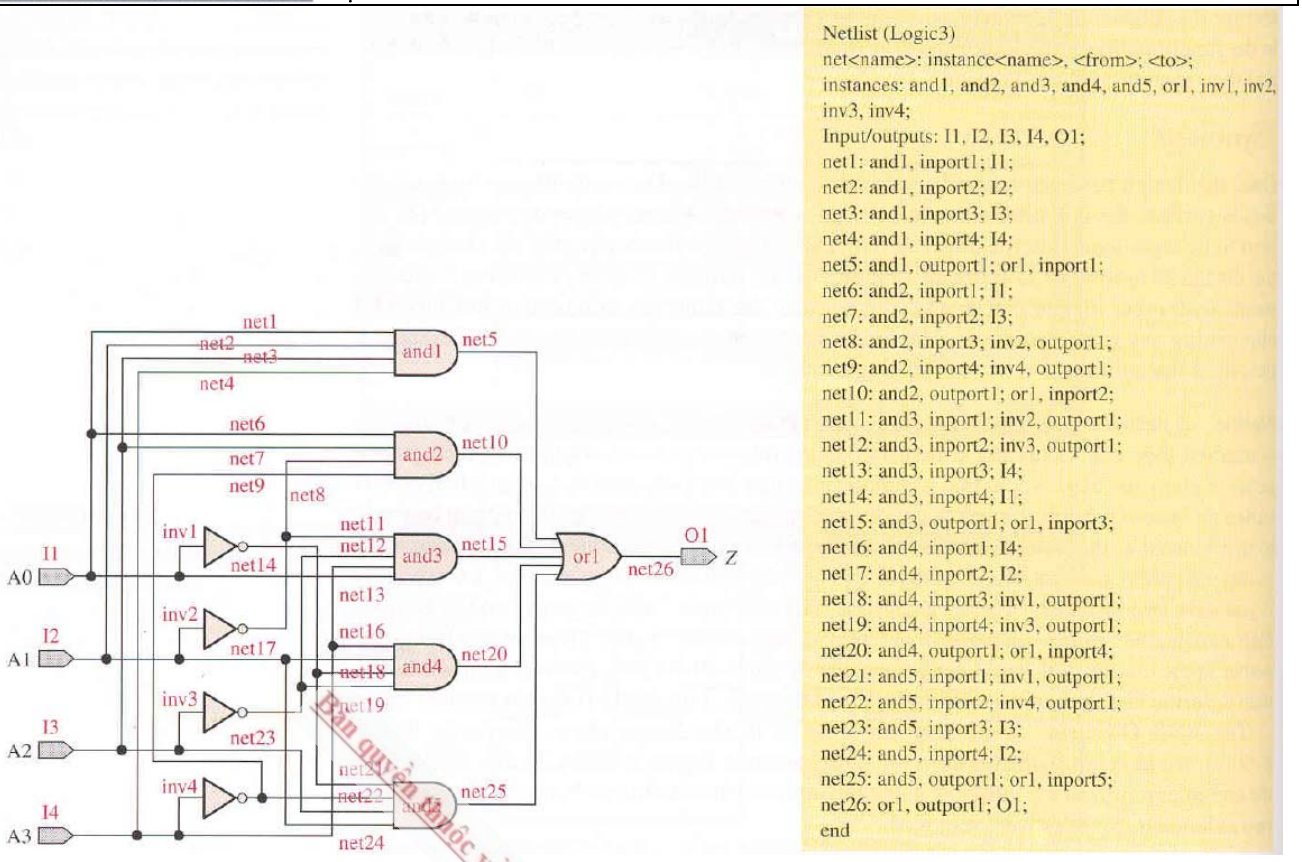
Mạch điện logic AND-OR đã thiết kế ở trên được trình bày ở hình 1-50a có thể được tối ưu thành mạch điện hình 1-50b. Trong phần minh họa này, trình biên dịch thay thế các cổng OR và bằng một cổng OR có 5 ngõ vào, bỏ hai cổng đảo thừa trong mạch.



(a). Mạch điện thiết kế **(b). Mạch tối ưu sau khi tổng hợp**

Hình 1-50. Minh họa cho chức năng tổng hợp.

Phần mềm tổng hợp tạo ra danh sách liệt kê lưới. Để minh họa cho khái niệm tạo ra danh sách lưới thì hình 1-51a sẽ trình bày cách gán tên cho lưới, gán tên cho instance và gán tên cho IO. Danh sách liệt kê lưới được trình bày ở hình 1-51b không cần thiết phải giống bất kỳ danh sách liệt kê nào về cú pháp và khuôn khổ. Danh sách liệt kê nhằm xác định các loại thông tin cần để mô tả mạch điện. Một khuôn khổ được dùng cho bảng liệt kê các lưới là EDIF (Electronic Design Interchange Format).



(a) (b)

Hình 1-51. Sơ đồ mạch và danh sách liệt kê.

5. PHẦN MỀM THI HÀNH

Sau khi thiết kế đã được tổng hợp thì trình biên dịch thi hành thiết kế – về cơ bản công việc này chính là sắp xếp thiết kế để nó có thể tương thích với thiết bị lập trình đã chọn bằng cách dựa vào cấu trúc và cấu hình chân.

Quá trình xử lý này gọi là làm cho tương thích (***fitting***). Để kết thúc công đoạn thi hành của dòng thiết kế thì phần mềm phải biết thiết bị rõ ràng và có đầy đủ các thông tin chi tiết về chân. Dữ liệu đầy đủ cho tất cả các thiết bị thường được lưu trong thư viện của bộ nhớ và người thiết kế chỉ cần chọn đúng thiết bị lập trình.

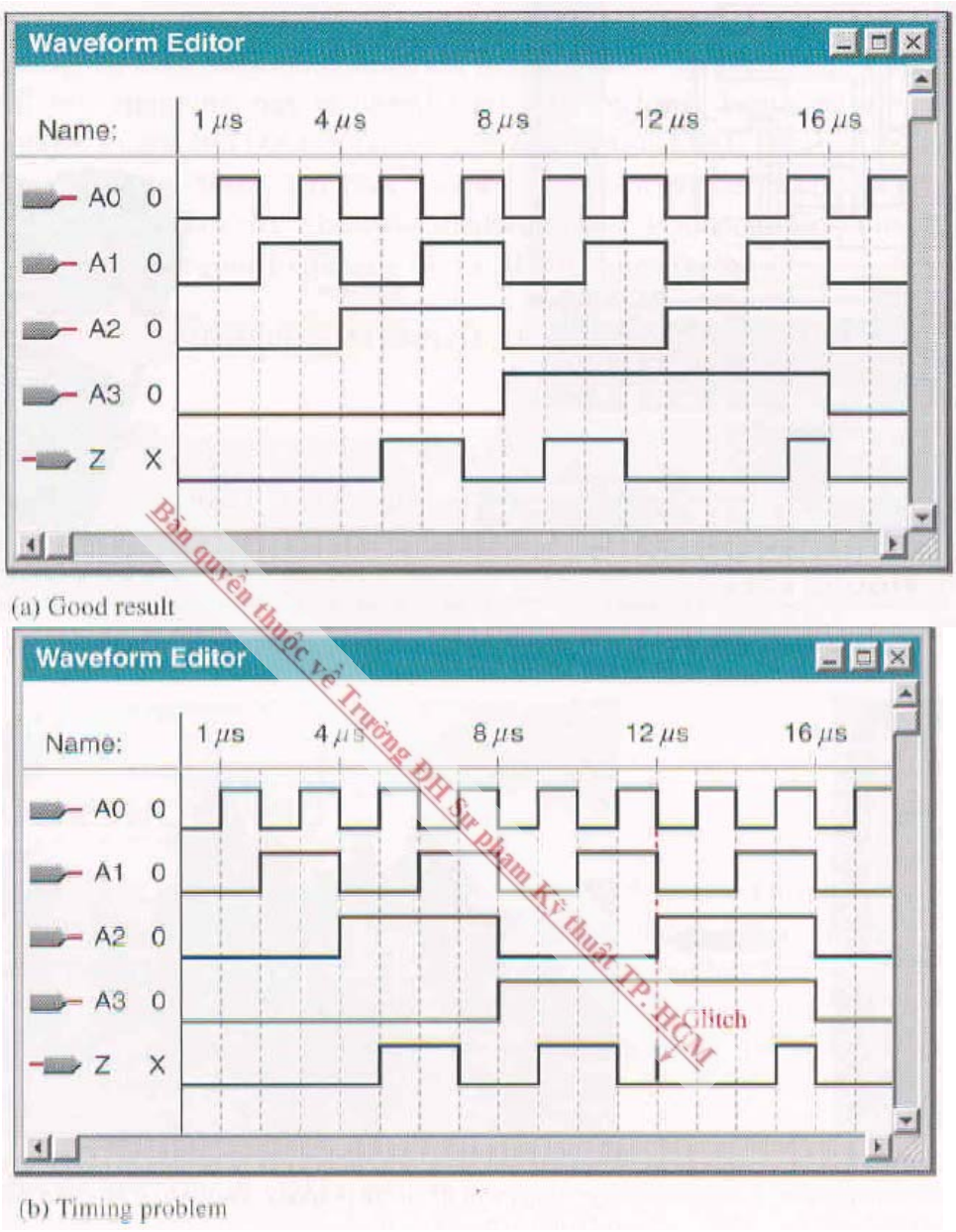
6. MÔ PHỎNG THỜI GIAN

Phần này nằm trong dòng thiết kế được thực hiện sau khi phần mềm thi hành biên dịch và trước khi nạp chương trình vào thiết bị. Mô phỏng theo thời gian để kiểm tra mạch điện hoạt động tại tần số thiết kế và không có thời gian trễ hoặc các vấn đề về thời gian khác làm ảnh hưởng đến hoạt động của mạch.

Phần mềm thiết kế dùng các thông tin của thiết bị lập trình như thời gian trì hoãn của các cổng để thực hiện mô phỏng theo thời gian của thiết kế.

Khi mô phỏng chức năng đã được thực hiện thì mạch điện sẽ hoạt động đúng theo quan điểm logic. Khi mô phỏng chức năng thì các thông số chỉ định về thiết bị đích là không cần thiết nhưng khi mô phỏng về thời gian thì phải lựa chọn thiết bị đích. Phần mềm soạn thảo dạng sóng có thể được dùng để xem kết quả mô phỏng cũng như mô phỏng chức năng được minh họa như hình 1-52.

Nếu không có vấn đề gì với kết quả mô phỏng như được trình bày ở hình 1-52a thì thiết kế có thể nạp vào thiết bị lập trình. Tuy nhiên, giả sử rằng các khoảng mô phỏng thời gian phát hiện không đều hay không giống nhau phụ thuộc vào thời gian trễ như được trình bày ở hình 1-52b.

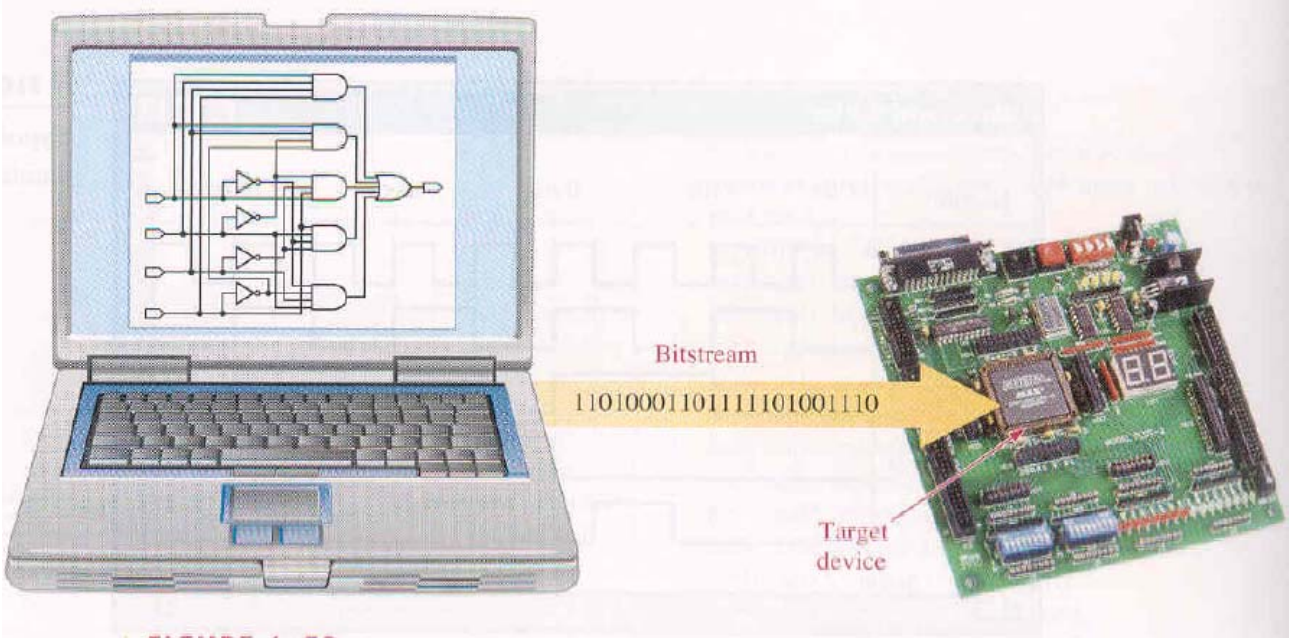


Hình 1-52. Minh họa cho mô phỏng thời gian.

Việc thực thi không giống nhau chỉ xảy ra trong một khoảng thời gian rất ngắn trong dạng sóng. Trong trường hợp này cần phải phân tích thiết kế một cách cẩn thận để tìm ra nguyên nhân và sau đó hiệu chỉnh lại thiết kế và lặp lại các bước thiết kế.

7. LẬP TRÌNH CHO THIẾT BỊ – HAY NẠP CHƯƠNG TRÌNH CHO THIẾT BỊ

Sau khi kiểm tra mô phỏng chức năng và mô phỏng theo thời gian và thiết kế đã hoạt động đúng thì có thể tiến hành download. Chuỗi bit nhị phân được tạo ra tượng trưng cho thiết kế và được gởi đến ***thiết bị đích*** để tự động định cấu hình cho thiết bị. Sau khi thực hiện xong thì thiết kế có thể được kiểm tra bằng mạch điện thực tế. Hình 1-53 trình bày khái niệm cho quá trình download.



Hình 1-53. Download thiết kế vào thiết bị lập trình.

VIII. CÂU HỎI ÔN TẬP VÀ BÀI TẬP

Câu 1-1. PAL tượng trưng cho cái gì ?

Câu 1-2. GAL tượng trưng cho cái gì ?

Câu 1-3. Sự khác nhau giữa PAL và GAL là gì ?

Câu 1-4. Một macrocell cơ bản chứa các thành phần nào?

Câu 1-5. CPLD là gì?

Câu 1-6. LAB tượng trưng cho cái gì ?

Câu 1-7. Mô tả LAB trong CPLD MAX 7000 ?

Câu 1-8. Mục đích cửa bộ mở rộng chia sẽ là gì ?

Câu 1-9. Mục đích cửa bộ mở rộng song song là gì ?

Câu 1-10. CPLD MAX II khác với MAX 7000 ở điểm nào ?

Câu 1-11. Sự khác nhau cơ bản của CPLD hãng Altera và hãng Xilinx là gì?

Câu 1-12. Hãy mô tả PLA ?

Câu 1-13. PLA khác với PAL là gì ?

Câu 1-14. FB tượng trưng cho cái gì ?

Câu 1-15. FPGA khác với CPLD như thế nào ?

Câu 1-16. CLB tượng trưng cho cái gì ?

Câu 1-17. Mô tả LUT và cho biết chức năng của nó ?

Câu 1-18. Sự khác nhau giữa kết nối bên trong toàn cục và cục bộ trong FPGA là gì ?

Câu 1-19. Lõi FPGA là gì ?

Câu 1-20. Định nghĩa thuật ngữ IP có liên đến nhà sản xuất FPGA ?

Câu 1-21. Đơn vị thiết kế logic cơ bản trong FPGA Stratix II là gì?

Câu 1-22. Có bao nhiêu ALM trong LAB?

Câu 1-23. Cái gì tạo ra các hàm logic tổ hợp trong ALM?

Câu 1-24. Có bao nhiêu hàm SOP có thể được tạo ra từ một ALM ?

Câu 1-25. Hãy cho biết tên của 2 loại chức năng tích hợp trong Stratix II ?

Câu 1-26. CLB trong FPGA của Xilinx chứa cái gì ?

Câu 1-27. LC chứa cái gì ?

Câu 1-28. Hãy mô tả slice trong FPGA của Xilinx ?

Câu 1-29. Chuỗi nối tiếp SOP là gì ?

Câu 1-30. ASMBL tượng trưng cho cái gì ?

Câu 1-31. Hãy liệt kê các công đoạn của dòng thiết kế cho một thiết bị lập trình ?

Câu 1-32. Hãy liệt kê các phần tử cơ bản để lập trình cho CPLD và FPGA?

Câu 1-33. Hãy cho biết chức năng của bảng liệt kê cá lưới ?

Câu 1-34. Có bao nhiêu hàm SOP có thể được tạo ra từ một ALM ?

Câu 1-35. Hãy cho biết tên của 2 loại chức năng tích hợp trong Stratix II ?

end