

CHƯƠNG 2: CẤU TRÚC PHẦN CỨNG HỆ THỐNG NHÚNG

Bài 2: Bộ xử lý chức năng đơn chuyên dụng (Custom single-purpose processors)

cuu duong than cong. com

Tổng quan

- Giới thiệu
- Mạch tổ hợp
- Mạch tuần tự
- Thiết kế bộ xử lý chức năng đơn
- Thiết kế bộ xử lý chức năng đơn chuyên dụng thời gian thực

Giới thiệu

* Các loại vi điều khiển trên thị trường hiện nay:

- **Freescale 68HC11 (8-bit)**
- **Intel 8051**
- **STMicroelectronics STM8S (8-bit), ST10 (16-bit) và STM32 (32-bit)**
- **Atmel AVR (8-bit), AVR32 (32-bit), và AT91SAM (32-bit)**
- **Freescale ColdFire (32-bit) và S08 (8-bit)**
- **Hitachi H8 (8-bit), Hitachi SuperH (32-bit)**
- **MIPS (32-bit PIC32)**
- **PIC (8-bit PIC16, PIC18, 16-bit dsPIC33 / PIC24)**
- **PowerPC ISE**
- **PSoC (Programmable System-on-Chip)**
- **Texas Instruments Microcontrollers MSP430 (16-bit), C2000 (32-bit), và Stellaris (32-bit)**
- **Toshiba TLCS-870 (8-bit/16-bit)**
- **Zilog eZ8 (16-bit), eZ80 (8-bit)**
- **Philips Semiconductors LPC2000, LPC900, LPC700**

Giới thiệu

- * Ứng dụng các loại vi xử lý và vi điều khiển được sử dụng trên thị trường Việt Nam hiện nay.
 - Có thể nói việc sử dụng các loại vi điều khiển và vi xử lý trong các thiết bị điện tử tự động ở Việt Nam rất đa dạng, phong phú tùy vào yêu cầu kỹ thuật và giá thành sản phẩm.
 - Đối với các thiết bị như các máy ATM, máy giặt thường sử dụng vi điều khiển 8051, các bộ điều khiển trong robot công nghiệp, trong hệ thống ô tô thường sử dụng PIC, AVR, PSoC, còn trong điện thoại sử dụng các chip ARM...

cuu duong than cong. com

Giới thiệu

1. Vi điều khiển 8051.

- Intel 8051 - là vi điều khiển đơn tinh thể kiến trúc Harvard, lần đầu tiên được sản xuất bởi Intel năm 1980, để dùng trong các hệ thống nhúng. Trong những năm 1980 và đầu những năm 1990 đã rất nổi tiếng. Tuy nhiên hiện tại đã cũ và được thay thế bằng các thiết bị hiện đại hơn, với các lõi phối hợp 8051, được sản xuất bởi hơn 20 nhà sản xuất độc lập như Atmel, Maxim IC (công ty con của Dallas Semiconductor), NXP Semiconductors (Philips Semiconductor trước đây), Winbond, Silicon Laboratories, Texas Instruments và Cypress Semiconductor. Tên gọi chính thức của họ vi điều khiển Intel 8051 - MCS 51.
- Những vi điều khiển Intel 8051 được sản xuất với việc dùng công nghệ MOSFET, những những bản sau, chứa kí hiệu “C” trong tên, như 80C51, dùng công nghệ CMOS và yêu cầu công suất thấp, hơn những cái MOSFET trước (điều này cho phép trang bị cho các thiết bị với nguồn là pin).

cuu duong than cong. com

Giới thiệu

1. Vi điều khiển 8051.

* Các thông số kỹ thuật:

8 bit ALU, 8 bit thanh ghi.

8 bit dữ liệu bus

16 bit địa chỉ bus vì vậy không gian bộ nhớ tối đa cho ROM và RAM lên tới 64 kb

Bộ nhớ dữ liệu SRAM 128 bytes

Bộ nhớ chương trình ROM 4 kb.

32 chân vào/ra đa hướng.

Giao tiếp nối tiếp UART.

Hai bộ timer/counter 16 bit.

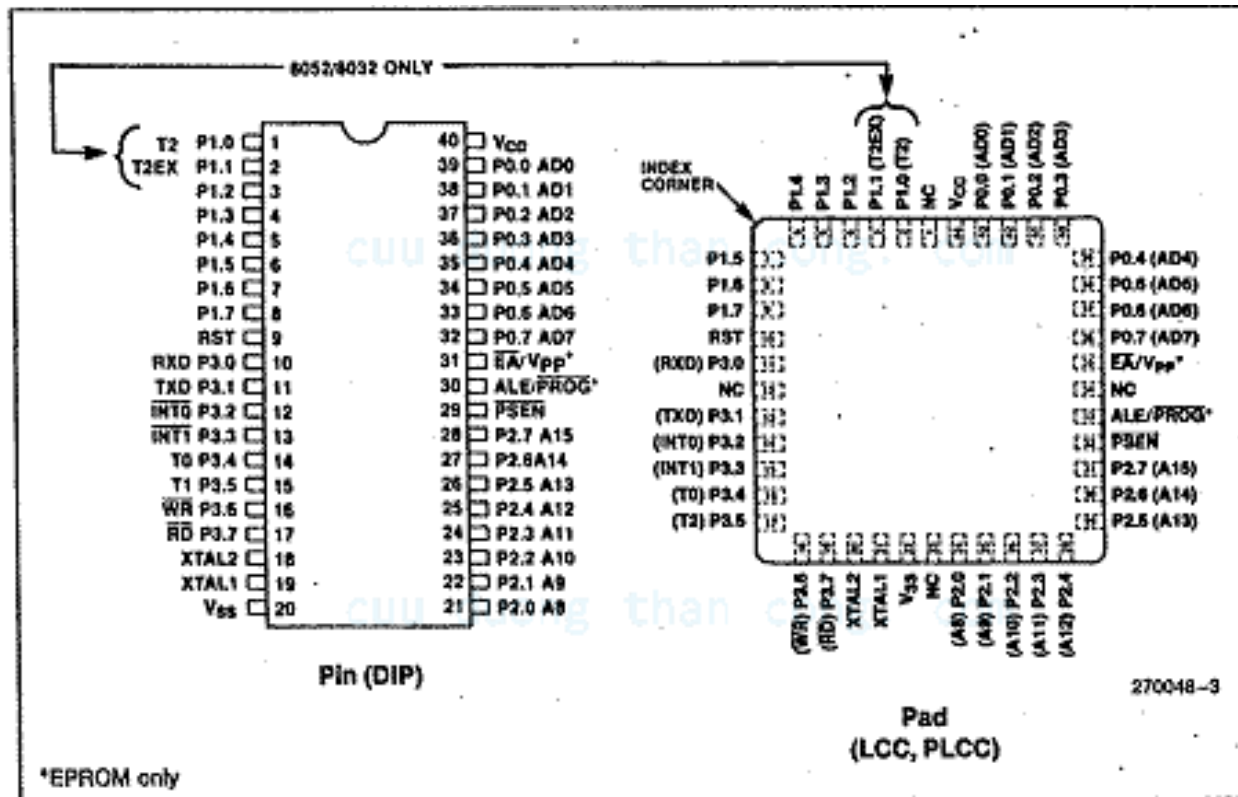
Hai ngắt ngoài.

cuu duong than cong. com

Giới thiệu

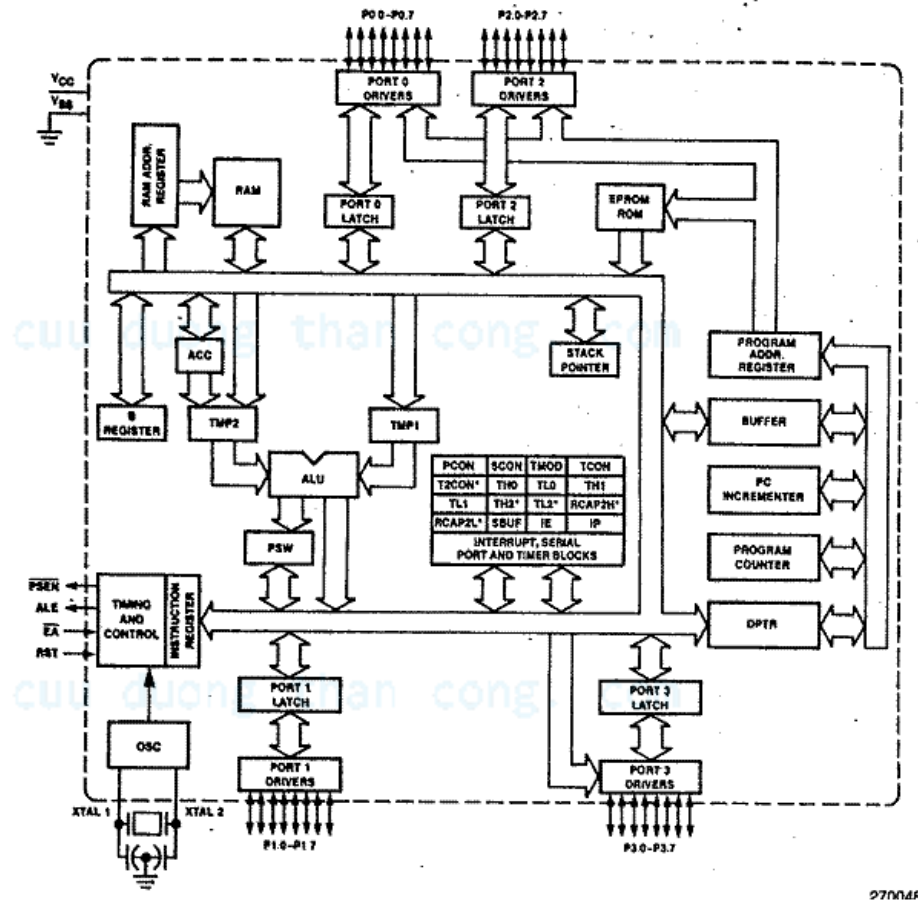
1. Vi điều khiển 8051.

Sơ đồ chân của 8051



Giới thiệu

1. Vi điều khiển 8051. Sơ đồ khối điều khiển:



Giới thiệu

1. Vi điều khiển 8051.

Lập trình cho 8051:

Các nhà sản xuất 8051 đều hỗ trợ ngôn ngữ lập trình Assembler tuy nhiên ngôn ngữ này thường ít được dùng cho những ứng dụng lớn do tính phù hợp của nó, vì vậy trong các ứng dụng thực tế hay sử dụng ngôn ngữ C. Ngoài ra còn một số ngôn ngữ khác được phát triển cho 8051 như Pascal, Basic, Forth.

cuu duong than cong. com

cuu duong than cong. com

Giới thiệu

2. Vi điều khiển AVR.

Là dòng vi điều khiển do hãng Atmel sản xuất có nhiều loại AVR như:

- 32-bit AVR UC3.
- 8/16-bit AVR XMEGA.
- 8-bit mega AVR.
- 8-bit tiny AVR.

•Vi điều khiển Atmega 16:

Là vi điều khiển 8 bit với tiêu thụ điện năng thấp dựa trên kiến trúc RISC (Reduced Instruction Set Computer). Vào ra Analog – digital và ngược lại. Với công nghệ này cho phép các lệnh thực thi chỉ trong một chu kì xung nhịp, vì thế tốc độ xử lý dữ liệu có thể đạt đến 1 triệu lệnh trên giây ở tần số 1Mhz. Vi điều khiển này cho phép người thiết kế có thể tối ưu hoá chế độ tiêu thụ năng lượng mà vẫn đảm bảo tốc độ xử lý.

Lỗi AVR có tập lệnh phong phú với số lượng với 32 thanh ghi làm việc chung với nhau. Tất cả 32 thanh ghi đều được nối trực tiếp với ALU (Arithmetic Logic Unit), cho phép 2 thanh ghi truy cập độc lập trong một chỉ lệnh đơn trong một chu kỳ xung nhịp. Kiến trúc đạt được có tốc độ xử lý nhanh gấp 10 lần vi điều khiển dạng CISC (Complex Instruction Set Computer) thông thường.

Giới thiệu

2. Vi điều khiển AVR.

Thông số kỹ thuật:

- Bộ nhớ:

Flash 16KB

EEPROM 512 Byte

SRAM 1KB.

- Ngoại vi:

Hai timer 8 bit

Một timer 16 bit

Bộ counter với tần số riêng

Bốn bộ điều chế độ rộng xung PWM.

Tám kênh ADC 10 bit.

USART.

Giao tiếp SPI, Giao diện I2C.

Watchdog timer.

Bộ so sánh tương tự trên chip.

Giới thiệu

2. Vi điều khiển AVR.

Thông số kỹ thuật:

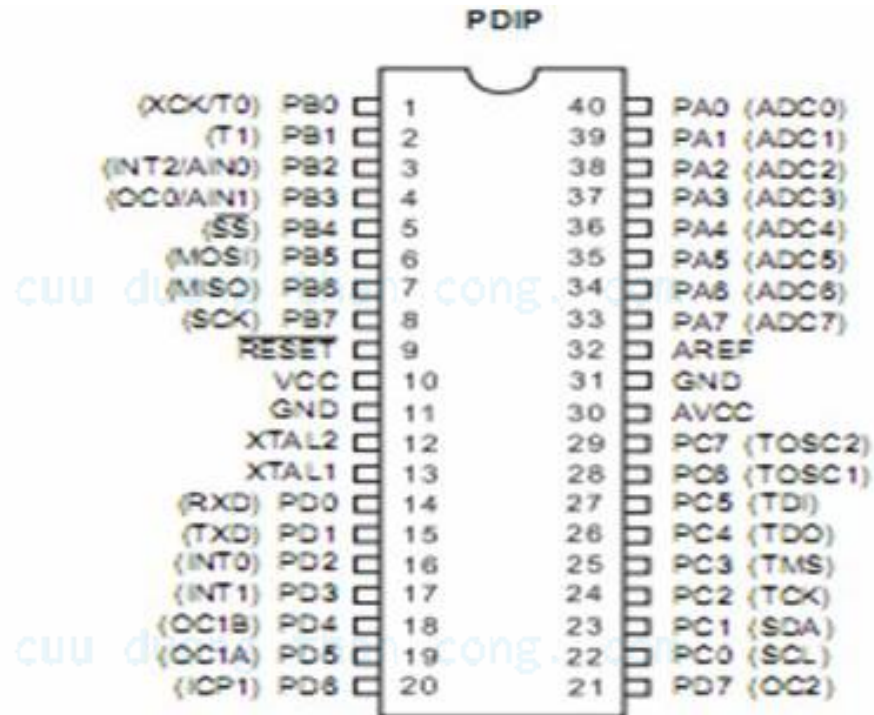
Tính năng:

- Tập lệnh gồm 131 lệnh, hầu hết thực hiện trong một chu kỳ máy.
- Xử lý 16 triệu lệnh ở tần số 16 MHZ.
- 32 chân vào/ra có thể lập trình được.
- Sáu chế độ sleep .
- 40 pin kiểu PDIP, 44 pin kiểu TQFP và kiểu QFL/MLF.
- 32 thanh ghi 8 bit đa dụng.
- Ngắt trong và ngắt ngoài.
- Điện áp hoạt động từ 2,7-5,5V cho Atmega 16A.

Giới thiệu

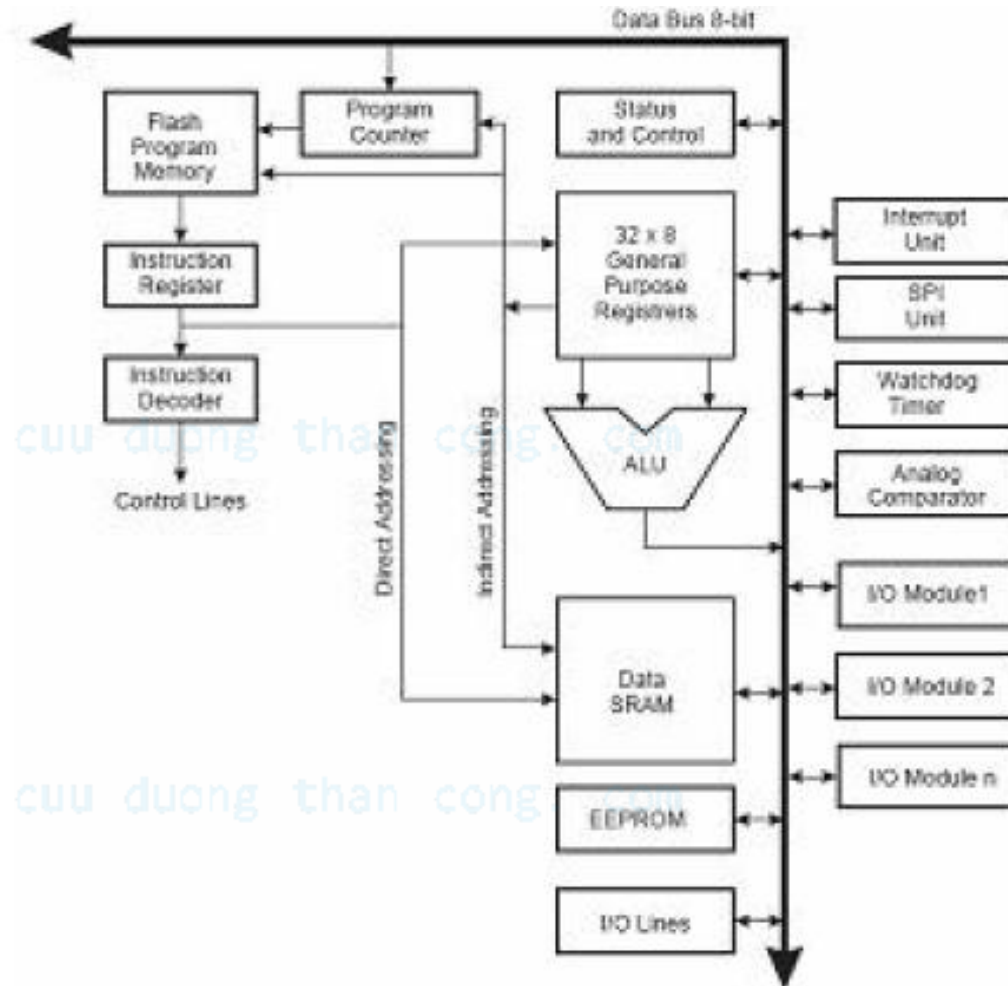
2. Vi điều khiển AVR.

Sơ đồ chân



Giới thiệu

2. Vi điều khiển AVR. Sơ đồ khối điều khiển



Giới thiệu

2. Vi điều khiển AVR.

Atmega 16 được hỗ trợ đầy đủ phần mềm và công cụ phát triển hệ thống bao gồm:

Trình dịch Assembly như AVR studio của Atmel, Trình dịch C như win AVR, CodeVisionAVR C, ICCAVR. C - CMPPILER của GNU... Trình dịch C đã được nhiều người dùng và đánh giá tương đối mạnh, dễ tiếp cận đối với những người bắt đầu tìm hiểu AVR, đó là trình dịch CodeVisionAVR C. Phần mềm này hỗ trợ nhiều ứng dụng và có nhiều hàm có sẵn nên việc lập trình tốt hơn.

cuu duong than cong. com

cuu duong than cong. com

Giới thiệu

3. Vi điều khiển PIC.

PIC là một họ vi điều khiển RISC được sản xuất bởi công ty Microchip Technology. Dòng PIC đầu tiên là PIC1650 được phát triển bởi Microelectronics Division thuộc General Instrument.

PIC bắt nguồn là chữ viết tắt của "Programmable Intelligent Computer" (Máy tính khả trình thông minh). Là vi điều khiển với kiến trúc RISC thực thi một lệnh với một chu kỳ máy (bằng bốn chu kỳ của bộ dao động). Ngày nay có nhiều dòng PIC được sản xuất với hàng loạt các mô đun ngoại vi tích hợp sẵn như ADC, PWM, USART, SPI...với bộ nhớ chương trình từ 512 word đến 32 Kword.

Các họ vi điều khiển PIC:

- Họ 8 bit: PIC 10/ PIC 12/ PIC 16/ PIC 18
- Họ 16 bit: PIC 24F/ PIC 24H/ dsPIC 30/ dsPIC 33
- Họ 32 bit: PIC 32.

• Comparison of PIC families

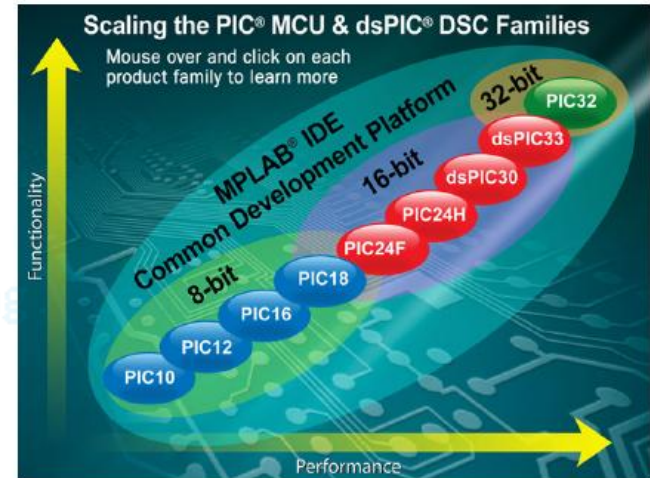
PIC family	Stack size (words)	Instruction word size	Number of instructions	Interrupt vectors
12CXXX/12FXXX	2	12- or 14-bit	33	None
16C5XX/16F5XX	2	12-bit	33	None
16CXXX/16FXXX	8	14-bit	35	1
17CXXX	16	16-bit	58, including hardware multiply	4
18CXXX/18FXXX	32	16-bit	75, including hardware multiply	2 (prioritised)

Giới thiệu

•3. Vi điều khiển PIC.

Thông số kỹ thuật

- Chân vào/ra I/O có thể lập trình được.
- Flash và ROM có thể tùy chọn từ 256 byte đến 512 Kbyte
- Bộ dao động bên trong.
- 8/16/32 bit Timers.
- Bộ nhớ EEPROM nội
- Chuẩn giao tiếp nối tiếp đồng bộ và không đồng bộ USART
- MSSP Peripheral cho giao tiếp I2C và SPI
- Các chế độ so sánh, bắt giữ và điều chế độ rộng xung PWM.
- Bộ so sánh điện áp.
- Bộ chuyển đổi ADC (tần số có thể lên tới 1 MHz).
- Hỗ trợ các giao thức USB, CAN, Ethernet.
- Mô đun điều khiển động cơ, mô đun đọc encoder.
- Hỗ trợ bộ nhớ ngoài.
- DSP những tính năng xử lý tín hiệu số (dsPIC)



Giới thiệu

•3. Vi điều khiển PIC.

Thông số kỹ thuật

Device number	No. of pins*	Clock speed	Memory (K = Kbytes, i.e. 1024 bytes)	Peripherals/special features
16F84A	18	DC to 20MHz	1K program memory, 68 bytes RAM, 64 bytes EEPROM	1 8-bit timer 1 5-bit parallel port 1 8-bit parallel port
16LF84A	As above	As above	As above	As above, with extended supply voltage range
16F84A-04	As above	DC to 4MHz	As above	As above
16F873A	28	DC to 20MHz	4K program memory, 192 bytes RAM, 128 bytes EEPROM	3 parallel ports, 3 counter/timers, 2 capture/compare/PWM modules, 2 serial communication modules, 5 10-bit ADC channels, 2 analog comparators
16F874A	40	DC to 20MHz	4K program memory, 192 bytes RAM, 128 bytes EEPROM	5 parallel ports, 3 counter/timers, 2 capture/compare/PWM modules, 2 serial communication modules, 8 10-bit ADC channels, 2 analog comparators

Giới thiệu

•3. Vi điều khiển PIC.

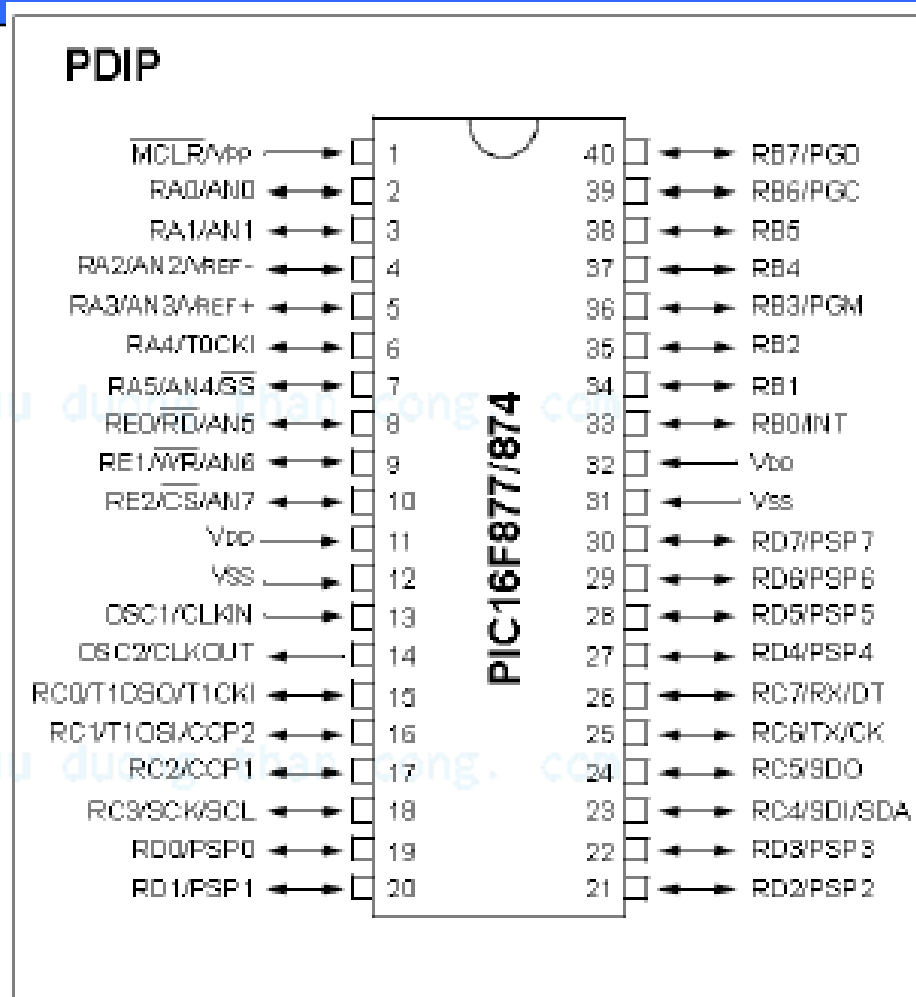
Thông số kỹ thuật

16F876A	28	DC to 20 MHz	8K program memory 368 bytes RAM, 256 bytes EEPROM	3 parallel ports, 3 counter/timers, 2 capture/compare/PWM modules, 2 serial communication modules, 5 10-bit ADC channels, 2 analog comparators
16F877A	40	DC to 20 MHz	8K program memory 368 bytes RAM, 256 bytes EEPROM	5 parallel ports, 3 counter/timers, 2 capture/compare/PWM modules, 2 serial communication modules, 8 10-bit ADC channels, 2 analog comparators

Giới thiệu

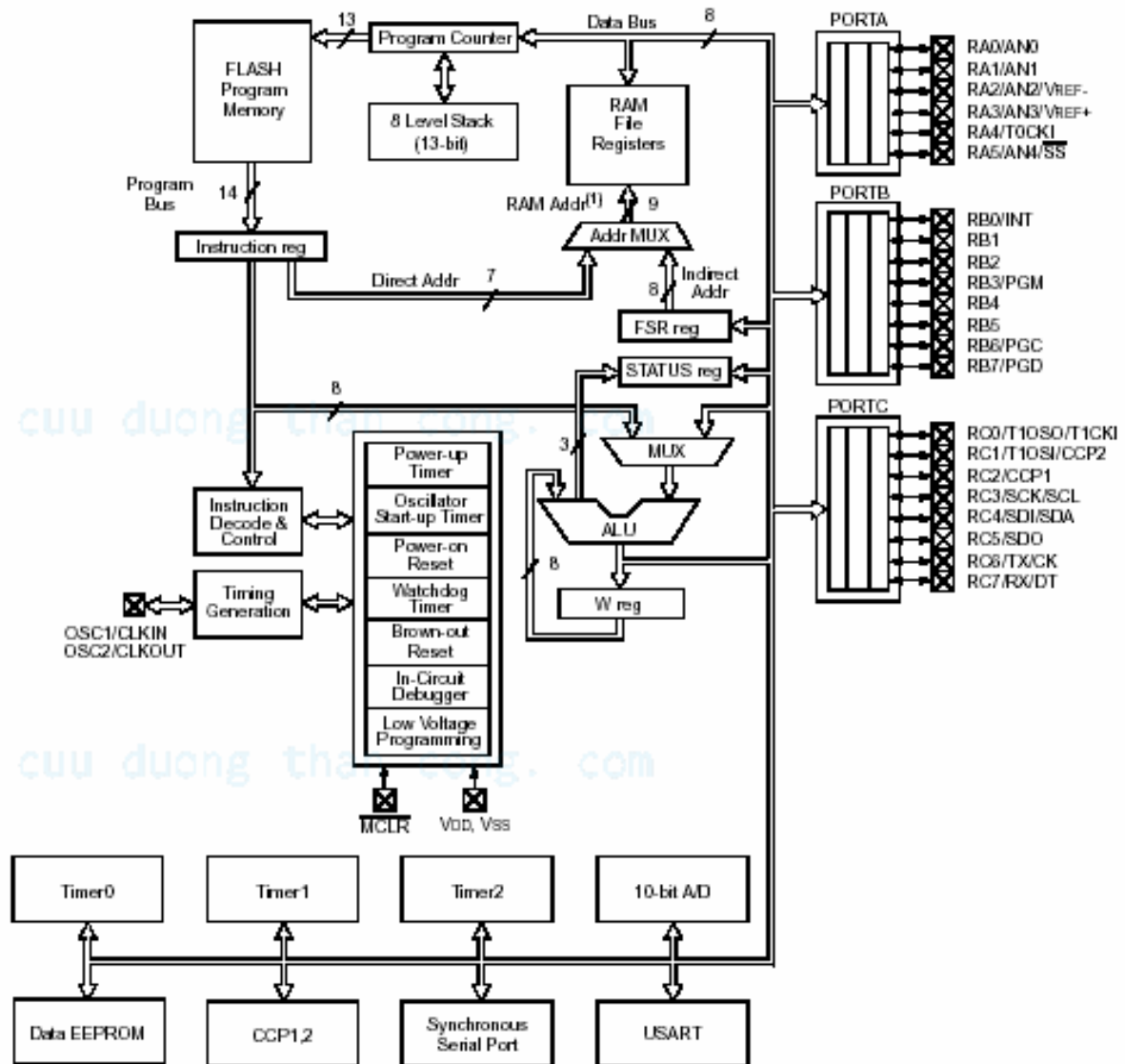
3. Vi điều khiển PIC.

Sơ đồ bố trí chân



Giới thiệu

3. Vi điều khiển PIC. Sơ đồ khối chức năng



Giới thiệu

3. Vi điều khiển PIC.

Lập trình cho PIC

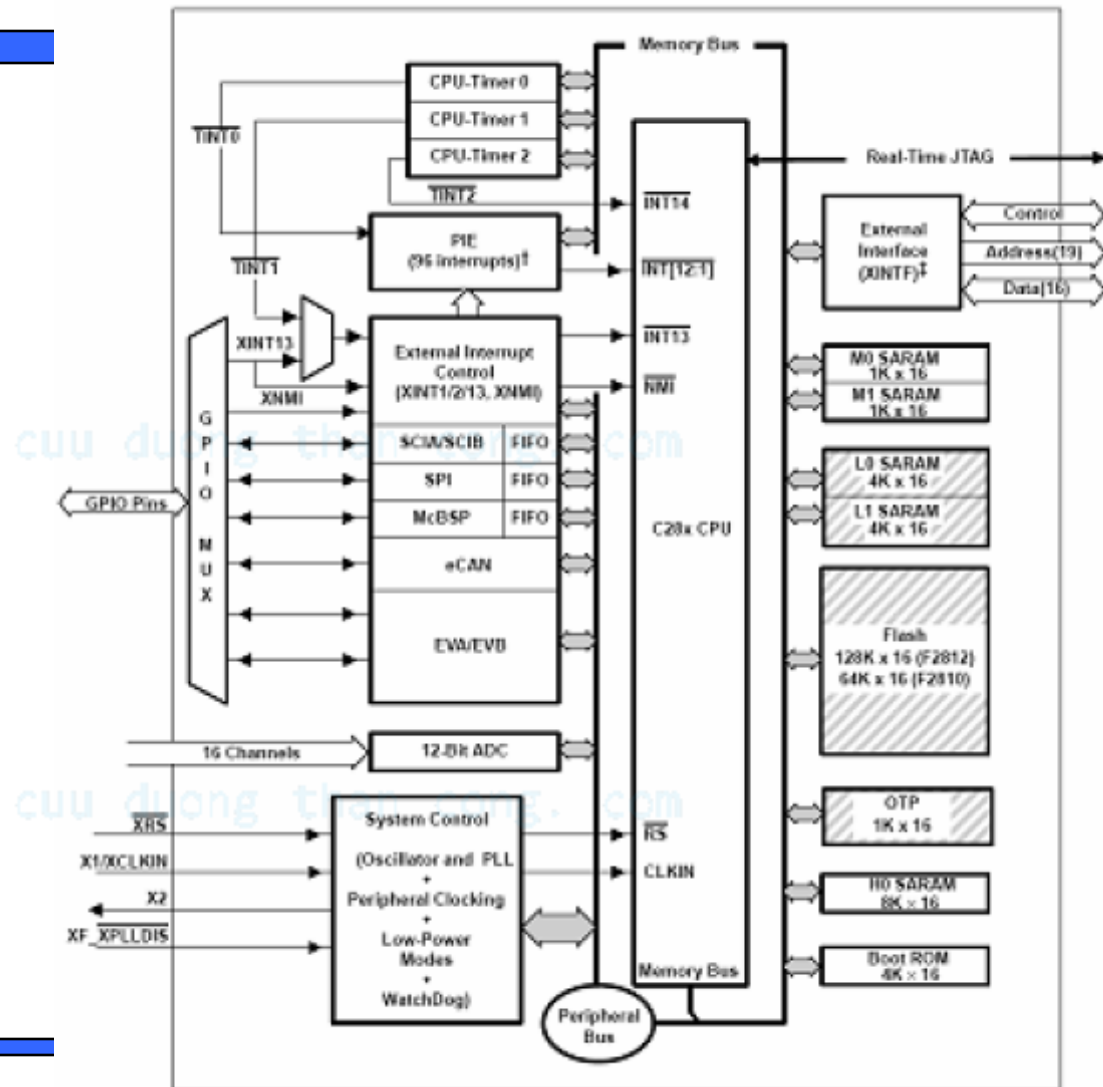
Hãng Microchip cung cấp môi trường lập trình MPLAB nó bao gồm phần mềm mô phỏng, trình dịch ASM, liên kết và gỡ rối. Ngoài ra hãng này cũng bán trình biên dịch C cho các dòng PIC18 và dsPIC tích hợp trong MPLAB.

Ngoài ra còn một số công ty khác cung cấp trình biên dịch C, PASCAL, BASIC cho PIC đó có thể là phần mềm thương mại hoặc phần mềm mã nguồn mở.

cuu duong than cong. com

Giới thiệu

4. Chip DSP



Giới thiệu

5. Vi điều khiển ARM.

Cấu trúc ARM (viết tắt từ tên gốc là Acorn RISC Machine) là một loại cấu trúc vi xử lý 32-bit kiểu RISC được sử dụng rộng rãi trong các thiết kế nhúng. Được phát triển lần đầu trong một dự án của công ty máy tính Acorn. Do có đặc điểm tiết kiệm năng lượng, các bộ CPU ARM chiếm ưu thế trong các sản phẩm điện tử di động, mà với các sản phẩm này việc tiêu tán công suất thấp là một mục tiêu thiết kế quan trọng hàng đầu.

Ngày nay, hơn 75% CPU nhúng 32-bit là thuộc họ ARM, điều này khiến ARM trở thành cấu trúc 32-bit được sản xuất nhiều nhất trên thế giới. CPU ARM được tìm thấy khắp nơi trong các sản phẩm thương mại điện tử, từ thiết bị cầm tay (PDA, điện thoại di động, máy đa phương tiện, máy trò chơi cầm tay, và máy tính cầm tay) cho đến các thiết bị ngoại vi máy tính (ổ đĩa cứng, bộ định tuyến để bàn.). Một nhánh nổi tiếng của họ ARM là các vi xử lý Xscale của Intel.

cuu duong than cong. com

Giới thiệu

5. Vi điều khiển ARM.

Giới thiệu về vi điều khiển LPC2148:

Là dòng vi điều khiển ARM được sản xuất bởi hãng Philips.

- Vi điều khiển 16/32-bit ARM7TDMI-S
- 40k RAM tĩnh (8k +32k), 512k flash
- Tích hợp USB 2.0
- Hỗ trợ hai bộ ADC 10 bit
- Một bộ DAC 10 bit
- 2 bộ timer 32 bit, 6 ngõ điều chế độ rộng xung
- Đồng hồ thời gian thực hỗ trợ tần số 32kHz
- Khả năng thiết lập chế độ ưu tiên và định địa chỉ cho ngắt
- 45 chân GPIO vào ra đa dụng
- 9 chân ngắt ngoài (tích cực cạnh hoặc tích cực mức)
- CPU clock đạt tối đa 60MHz thông qua bộ PLL lập trình được
- Xung PLCK hoạt động độc lập.



LPC1754FBD80



AT91SAM7S64-AU



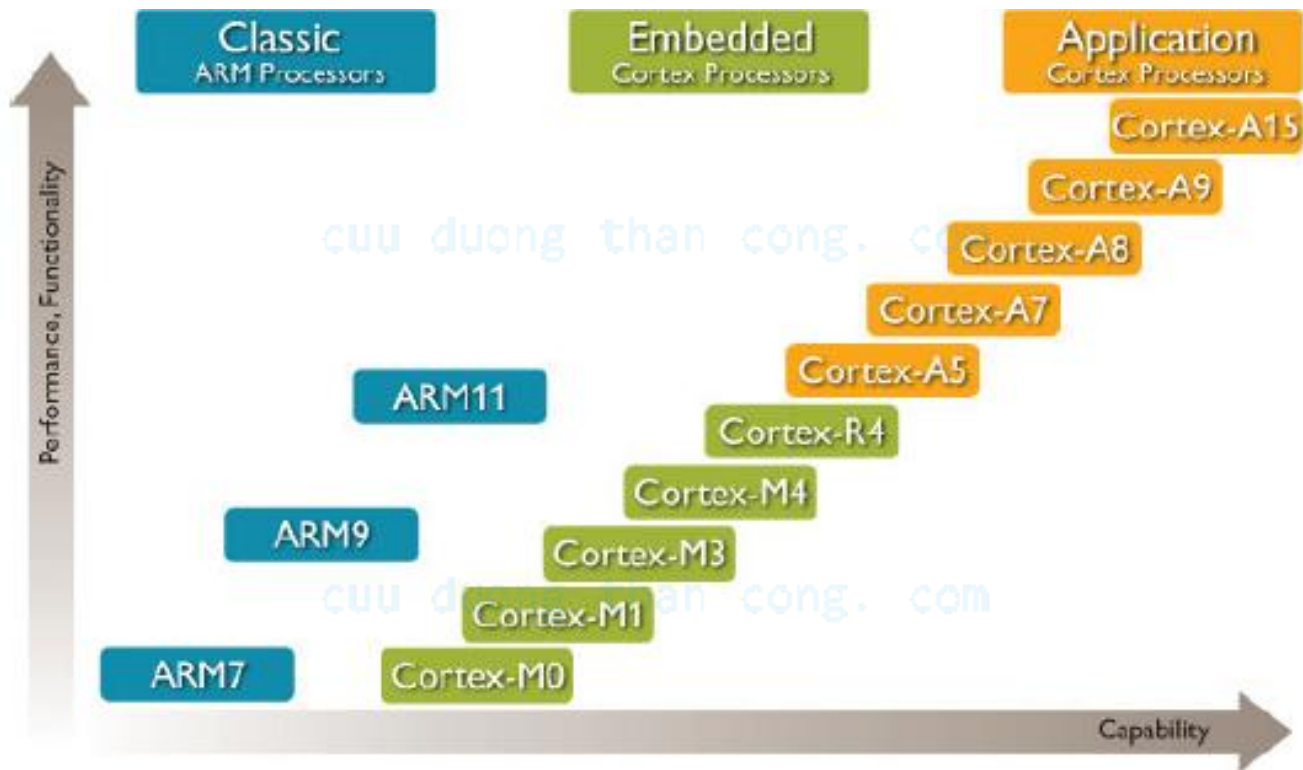
STM32F103RCT6



LM3S3749

Giới thiệu

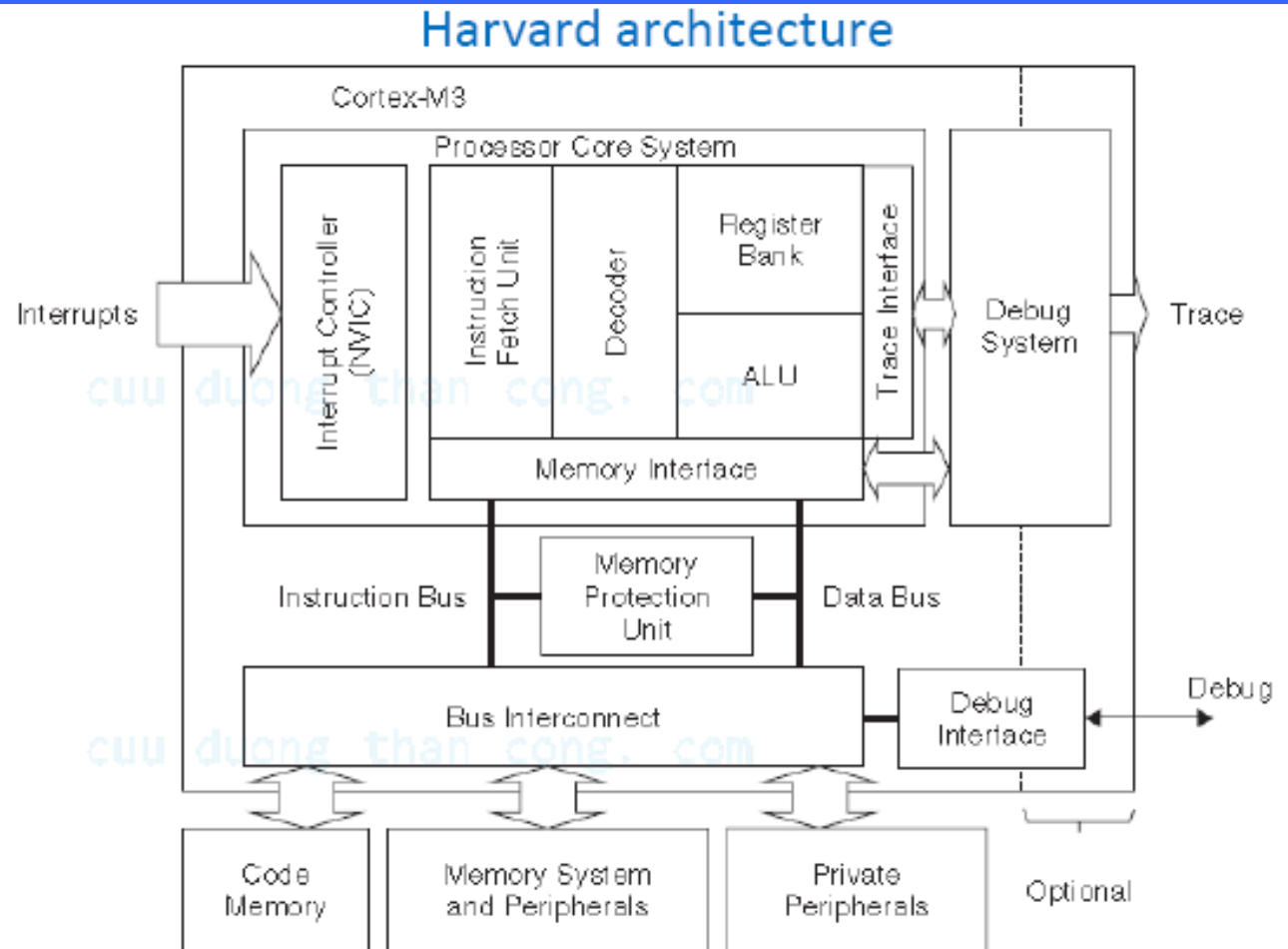
5. Vi điều khiển ARM.



Giới thiệu

5. Vi điều khiển

Sơ đồ kiến trúc



Giới thiệu

5. Vi điều khiển ARM.

Ngôn ngữ lập trình chính cho ARM hiện nay là ngôn ngữ C. Các trình biên dịch cho ARM thường được dùng:

- Keil ARM.
- IAR.
- HTPICC for ARM.
- ImageCraft ICCV7 for ARM

cuu duong than cong. com

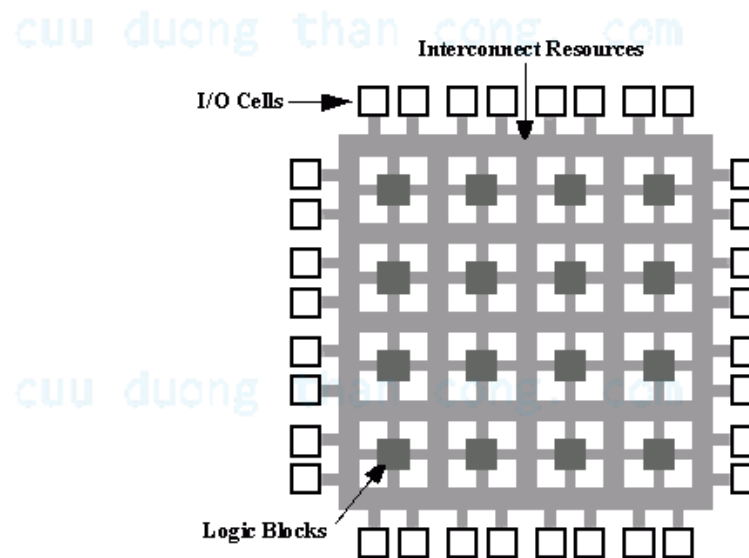
cuu duong than cong. com

Giới thiệu

6. FPGA và các ứng dụng nhúng

Hệ thống nhúng (embedded system) bắt đầu từ các họ Vi điều khiển 8051 rồi đến PIC, AVR và cao hơn nữa là họ ARM, AVR32 hay PSoC. Với FPGA tiếp cận hoàn toàn khác.

Vậy bao nhiêu công sức, kinh nghiệm về vi điều khiển và cả những công trình nghiên cứu bị bỏ rơi? thực ra FPGA chỉ phát huy sức mạnh của nó khi được ghép nối với vi điều khiển. Đó cũng chính là mục đích và tư tưởng thiết kế của co-design.



Giới thiệu

6. FPGA và các ứng dụng nhúng

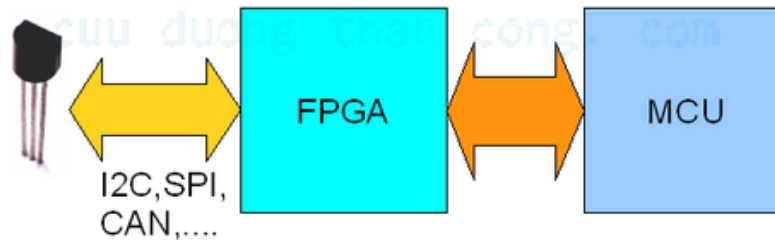
Co-design kết hợp năng lực về phần cứng của FPGA với ưu thế xử lý phần mềm của Vi điều khiển để tạo nên một hệ thống đầy sức mạnh.

Ví dụ : Muốn thiết kế một ứng dụng đo nhiệt độ phòng với cảm biến nhiệt có giao tiếp I²C.

Nếu chỉ dùng MCU thông thường không có giao tiếp I²C thì sẽ gặp rất nhiều khó khăn (Phải lập trình ngắt, bắt sườn, mức của xung,...).

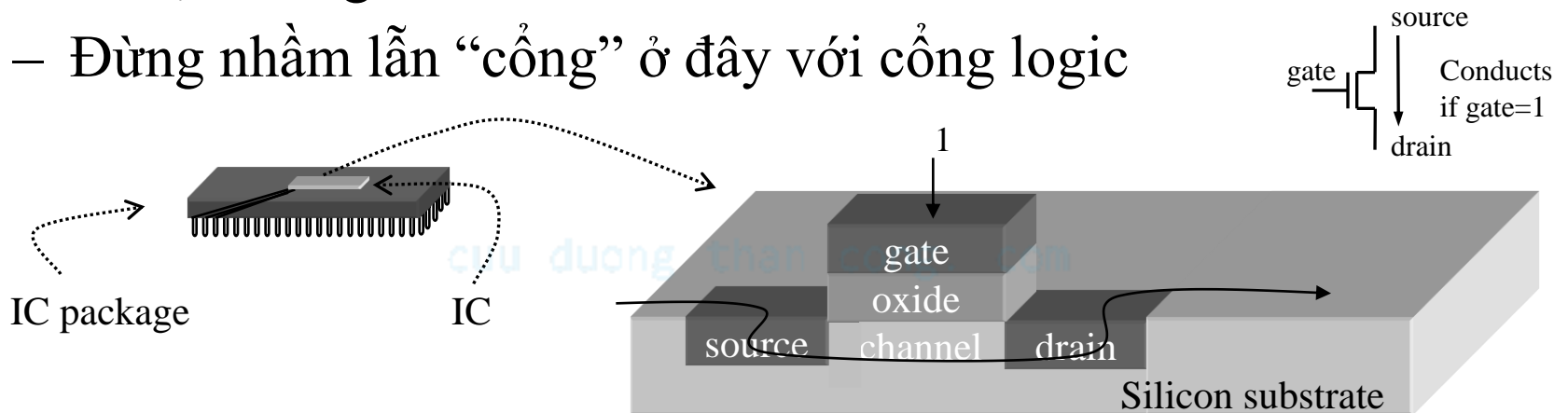
Còn nếu chỉ sử dụng FPGA trong ứng dụng này cũng không ổn vì lúc đó bạn sẽ gặp khó khăn nhất định trong các tính toán số học.

Ví dụ cảm biến đo nhiệt độ bằng đơn vị độ F, trong khi ta muốn hiển thị độ C, mà muốn thực hiện các phép toán cộng trừ nhân chia để chuyển đổi độ F với độ C bằng FPGA là không hề đơn giản. Trong trường hợp này ta thiết kế theo phương thức co-design. FPGA phụ trách giao tiếp với cảm biến I²C và trả về các số liệu thô để MCU thực hiện các tính toán số học.



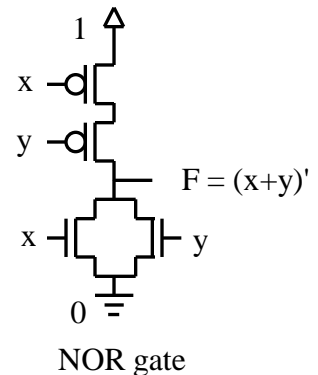
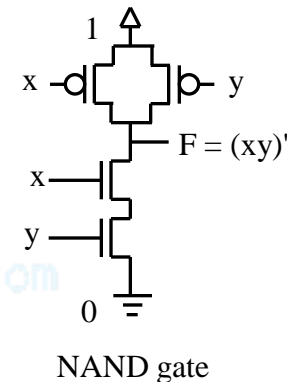
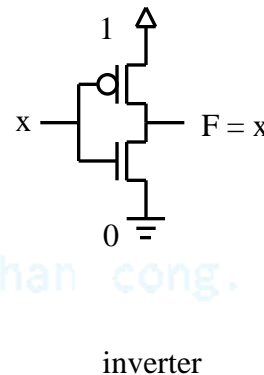
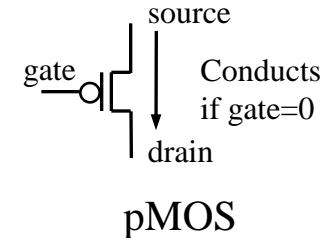
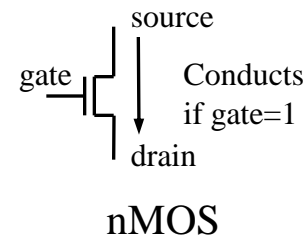
Transistor CMOS trên silicon

- Transistor
 - Là phần điện tử cơ bản trên mạch số
 - Hoạt động như một chuyển mạch on/off
 - Điện áp ở cực “cổng” điều khiển dòng giữa cực nguồn và cực máng
 - Đừng nhầm lẫn “cổng” ở đây với cổng logic

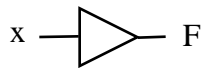


Thực hiện transistor CMOS

- CMOS - Complementary Metal Oxide Semiconductor
- Chúng ta quan tâm đến mức logic
 - Điện hình 0 là 0V, 1 là 5V
- Hai kiểu CMOS cơ bản
 - nMOS dẫn nếu “công” = 1
 - pMOS dẫn nếu “công” = 0
 - Vì thế gọi là “complementary”
- Các cổng cơ bản
 - Đảo, NAND, NOR

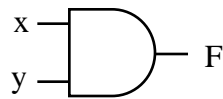


Các cổng logic cơ bản



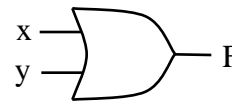
x	F
0	0
1	1

$F = x$
Driver



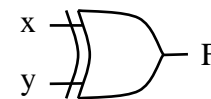
x	y	F
0	0	0
0	1	0
1	0	0
1	1	1

$F = x \cdot y$
AND



x	y	F
0	0	0
0	1	1
1	0	1
1	1	1

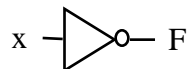
$F = x + y$
OR



x	y	F
0	0	0
0	1	1
1	0	1
1	1	0

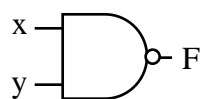
$F = x \oplus y$
XOR

cuu duong than cong. com



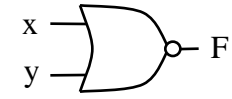
x	F
0	1
1	0

$F = x'$
Inverter



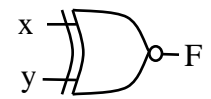
x	y	F
0	0	1
0	1	1
1	0	1
1	1	0

$F = (x \cdot y)'$
NAND



x	y	F
0	0	1
0	1	0
1	0	0
1	1	0

$F = (x + y)'$
NOR



x	y	F
0	0	1
0	1	0
1	0	0
1	1	1

$F = x \odot y$
XNOR

cuu duong than cong. com

Thiết kế logic tổ hợp

A) Mô tả vấn đề

y là 1 nếu a là 1, hoặc b và c là 1. z là 1 nếu b hoặc c là 1, nhưng không phải cả hai, hoặc nếu tất cả là 1.

B) Bảng chân lý

Inputs			Outputs	
a	b	c	y	z
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	1	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

C) Phương trình đầu ra

$$y = a'bc + ab'c' + ab'c + abc' + abc$$

$$z = a'b'c + a'bc' + ab'c + abc' + abc$$

D) Phương trình đầu ra tối thiểu

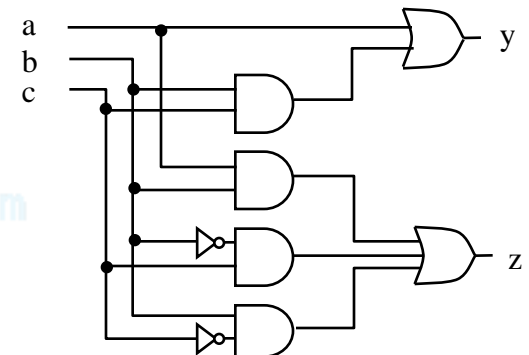
y	a	bc			
		00	01	11	10
0	0	0	0	1	0
1	1	1	1	1	1

$$y = a + bc$$

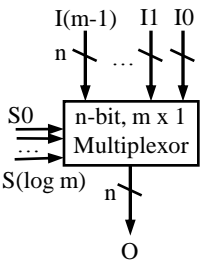
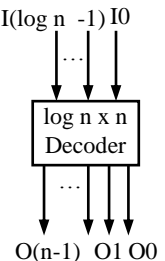
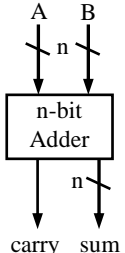
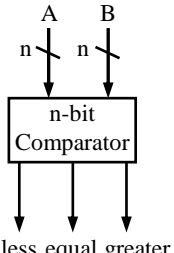
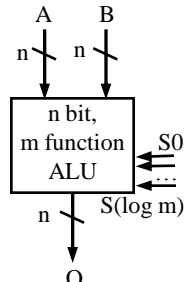
z	a	bc			
		00	01	11	10
0	0	0	1	0	1
1	1	0	1	1	1

$$z = ab + b'c + bc'$$

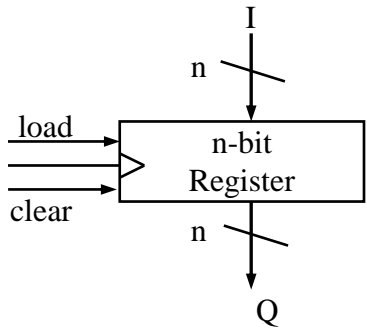
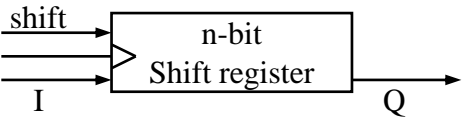
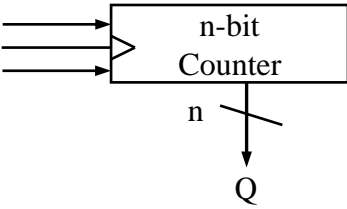
E) Mạch logic



Các phần tử mạch tổ hợp

				
<p> $O =$ I_0 nếu $S=0..00$ I_1 nếu $S=0..01$ \dots $I_{(m-1)}$ nếu $S=1..11$ </p>	<p> $O_0 = 1$ nếu $I=0..00$ $O_1 = 1$ nếu $I=0..01$ \dots $O_{(n-1)} = 1$ nếu $I=1..11$ </p>	<p> $sum = A + B$ (n bit đầu) $carry = \text{bit thứ } (n+1) \text{ của } A+B$ </p>	<p> $Less = 1$ nếu $A < B$ $equal = 1$ nếu $A = B$ $greater = 1$ nếu $A > B$ </p>	<p> $O = A \text{ op } B$ op được xác định bởi S. </p>
	<p>Với đầu vào enable e \rightarrow tất cả các bit đầu ra là 0 nếu e=0</p>	<p>Với đầu vào Carry-in $C_i \rightarrow$ $sum = A + B + C_i$</p>		<p>Có thể có các đầu ra trạng thái.</p>

Các phần tử mạch tuần tự

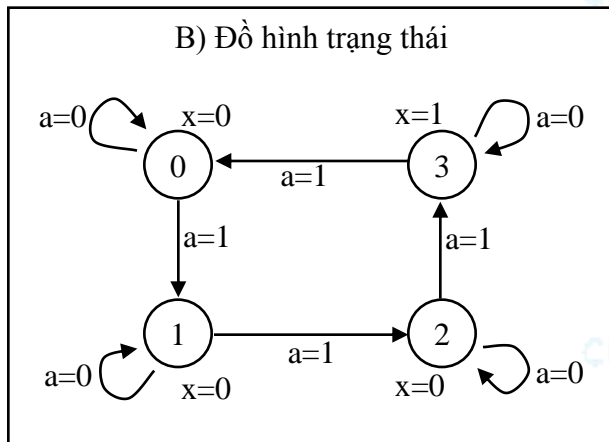
		
<p>Q = 0 nếu clear=1, I nếu load=1 and clock=1, Q(previous) nếu khác.</p>	<p>Q = lsb - Nội dung được dịch - I được lưu trong msb</p>	<p>Q = 0 nếu clear=1, Q(prev)+1 nếu count=1 và clock=1.</p>

Thiết kế mạch tuần tự

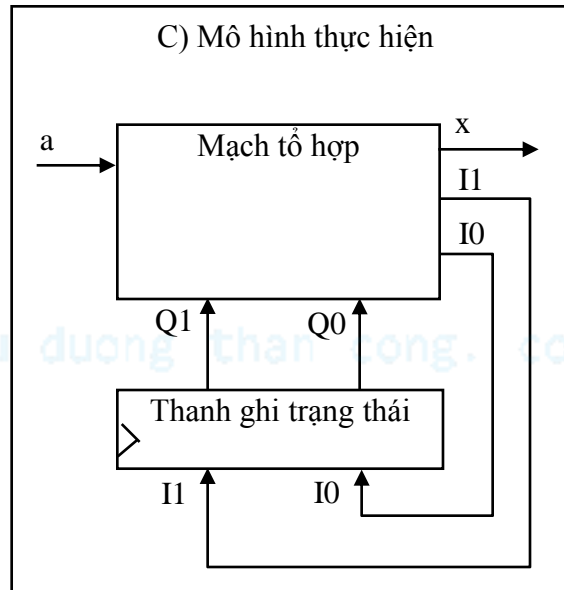
A) Mô tả vấn đề

Giả sử ta muốn xây dựng 1 bộ chia xung clock. Đầu ra có tín hiệu “1” sau mỗi 4 xung clock.

B) Đồ hình trạng thái



C) Mô hình thực hiện



D) Bảng trạng thái (Moore-type)

Inputs			Outputs		
Q1	Q0	a	I1	I0	x
0	0	0	0	0	0
0	0	1	0	1	
0	1	0	0	1	0
0	1	1	1	0	
1	0	0	1	0	0
1	0	1	1	1	
1	1	0	1	1	1
1	1	1	0	0	

- Cho mô hình thực hiện này
 - Vấn đề thiết kế mạch tuần tự trở thành thiết kế mạch tổ hợp

Thiết kế mạch tuần tự (cont.)

E) Phương trình đầu ra tối thiểu

I1

	Q1Q0	00	01	11	10
a					
0		0	0	1	1
1		0	1	0	1

$$I1 = Q1'Q0a + Q1a' + Q1Q0'$$

I0

	Q1Q0	00	01	11	10
a					
0		0	1	1	0
1		1	0	0	1

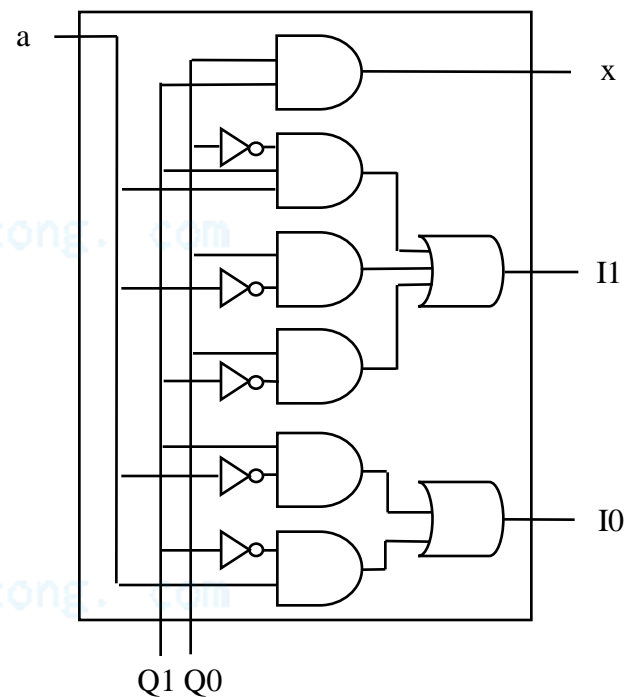
$$I0 = Q0a' + Q0'a$$

x

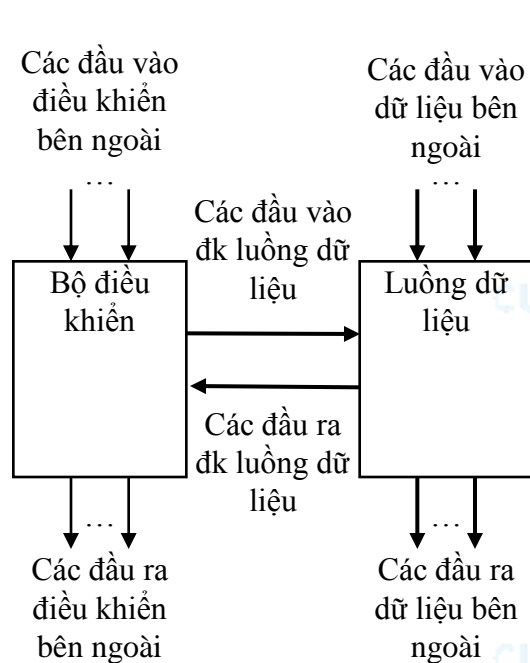
	Q1Q0	00	01	11	10
a					
0		0	0	1	0
1		0	0	1	0

$$x = Q1Q0$$

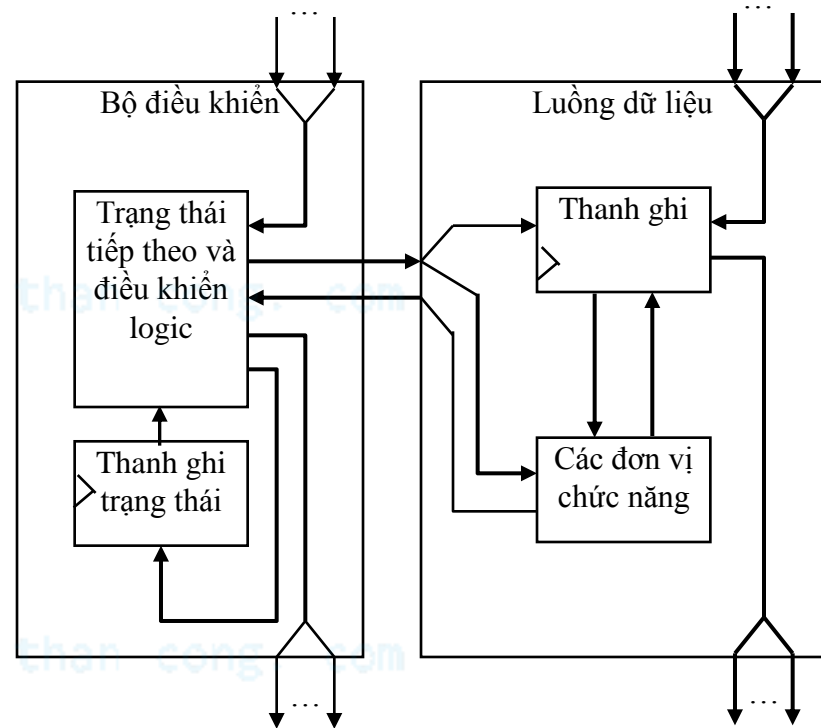
F) Mạch tổ hợp



Mô hình cơ bản của bộ xử lý chức năng đơn chuyên biệt



Bộ điều khiển và luồng dữ liệu



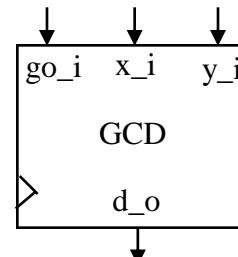
Nhìn bên trong bộ điều khiển và luồng dữ liệu

Ví dụ bộ xử lý chức năng đơn chuyên biệt

Tìm ước số chung lớn nhất

- Đầu tiên tạo ra thật toán
- Biến đổi thuật toán thành giản đồ trạng thái
 - Thường gọi là FSMD (finite-state machine with datapath)
 - Có thể sử dụng biểu tượng để thực hiện việc chuyển đổi

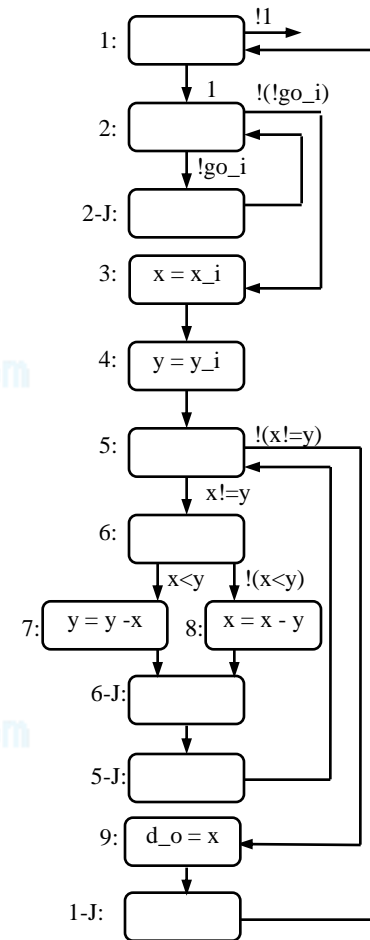
(a) Sơ đồ khối



(b) Chức năng yêu cầu

```
0: int x, y;
1: while (1) {
2:   while (!go_i);
3:   x = x_i;
4:   y = y_i;
5:   while (x != y) {
6:     if (x < y)
7:       y = y - x;
8:     else
9:       x = x - y;
10:  }
11:  d_o = x;
12: }
```

(c) Giản đồ trạng thái

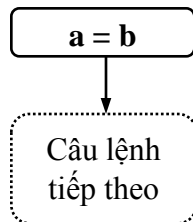


Biểu tượng giản đồ trạng thái

Câu lệnh gán

a = b

Câu lệnh tiếp
theo

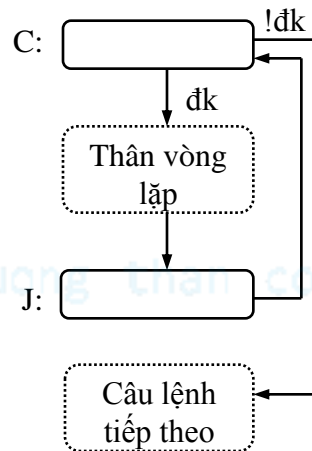


Câu lệnh lặp

while (đk)

```
{  
    thân vòng lặp  
}
```

Câu lệnh tiếp theo



Câu lệnh rẽ nhánh

if (c1)

c1 stmts

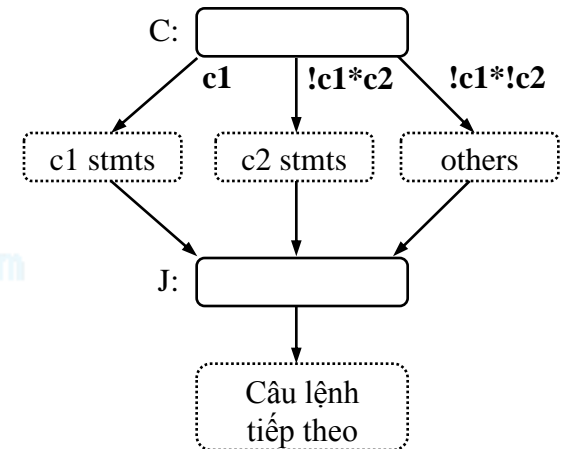
else if c2

c2 stmts

else

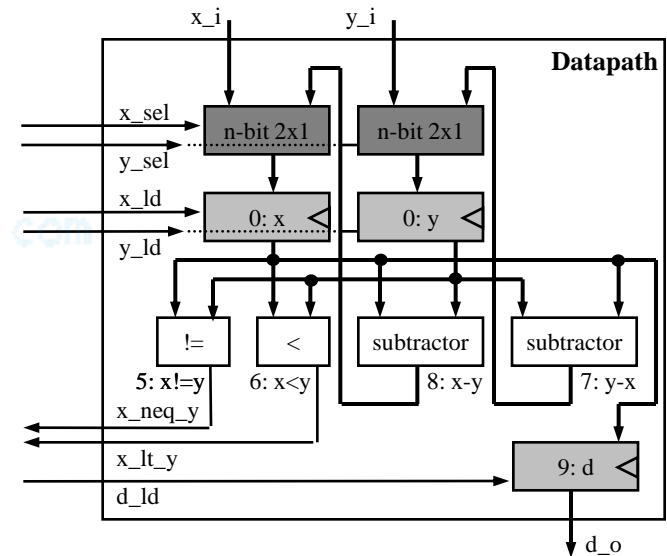
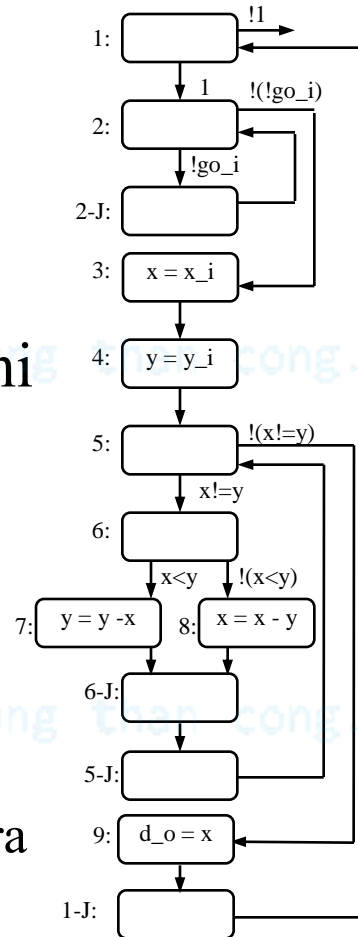
other stmts

Câu lệnh tiếp theo

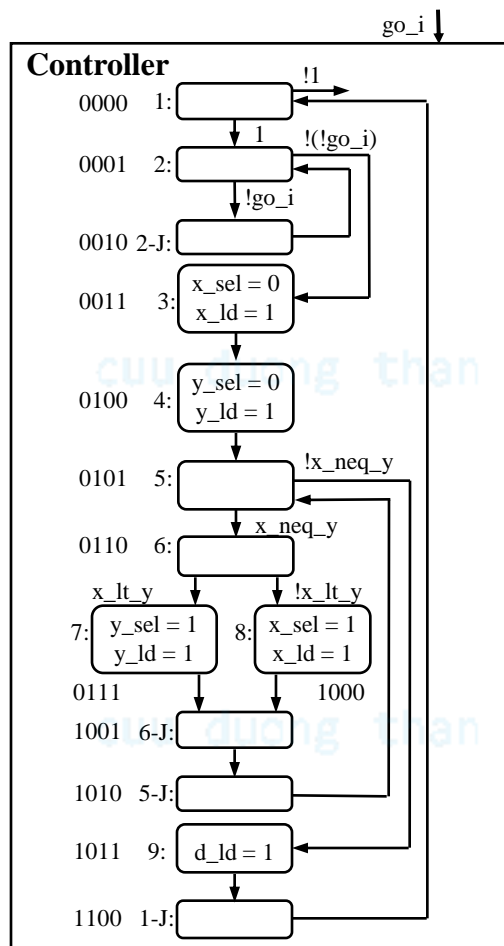
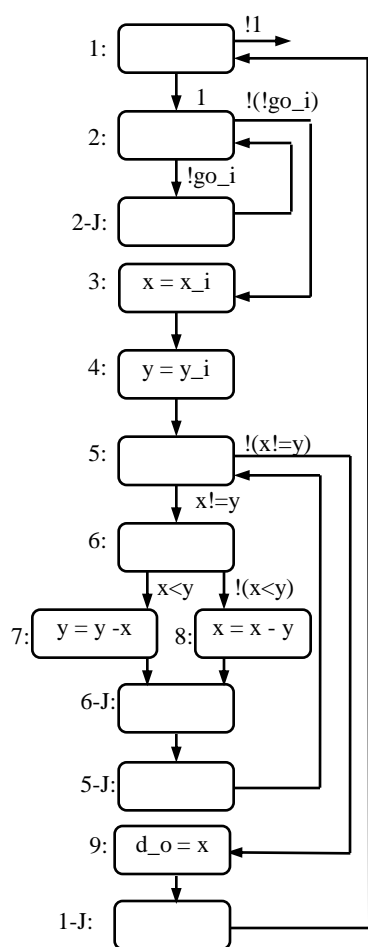


Tạo ra tuyến dữ liệu

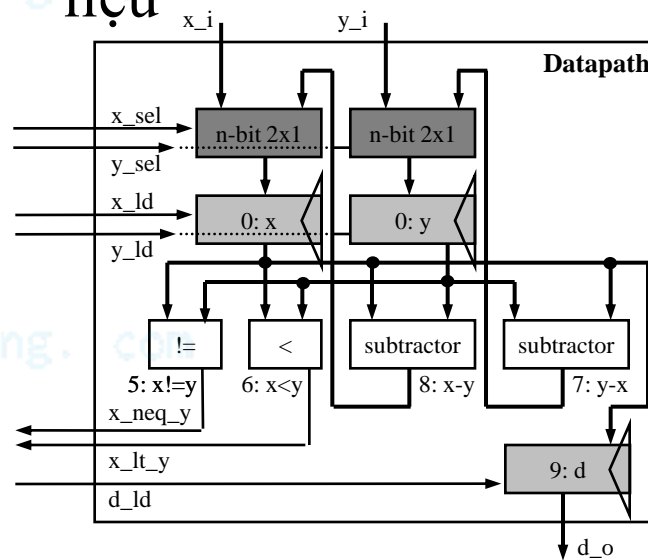
- Tạo ra một thanh ghi cho biến được khai báo
- Tạo ra một hàm cho mỗi thuật toán tính toán
- Kết nối các cổng, thanh ghi và hàm
 - Dựa trên việc đọc và ghi
- Tạo ra bộ nhận dạng đồng nhất
 - Đối với mỗi luồng dữ liệu, điều khiển đầu vào và đầu ra



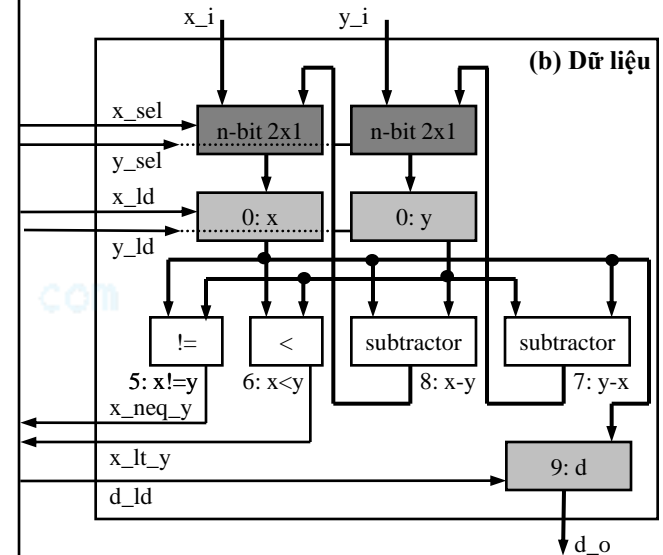
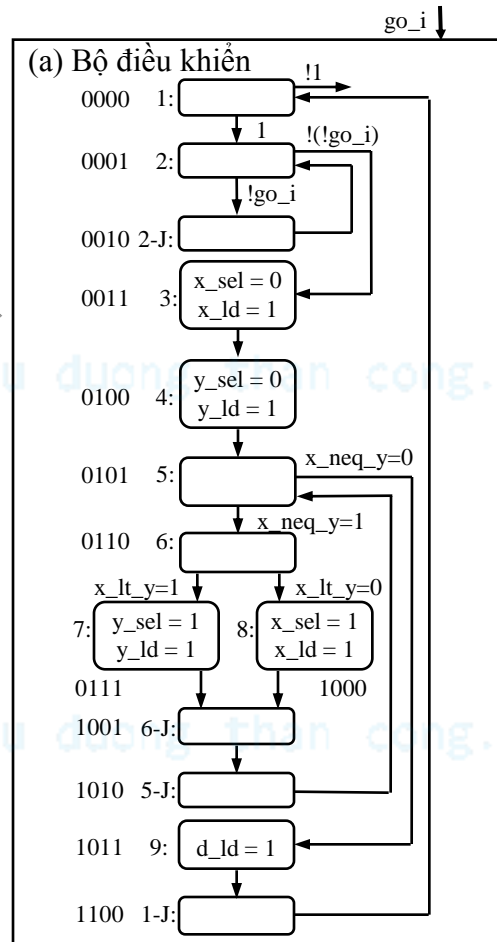
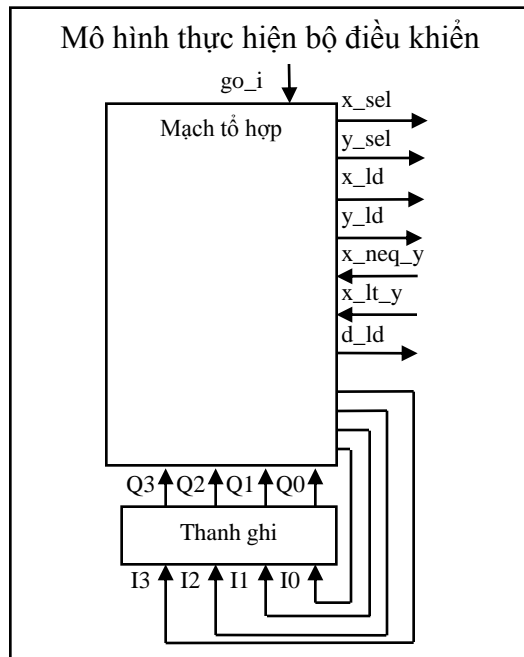
Tạo ra biểu đồ trạng thái (FSM) cho bộ điều khiển



- Có cùng cấu trúc như FSMD
- Thay thế các phép tính phức tạp bằng luồng dữ liệu



Tách thành bộ điều khiển và luồng dữ liệu

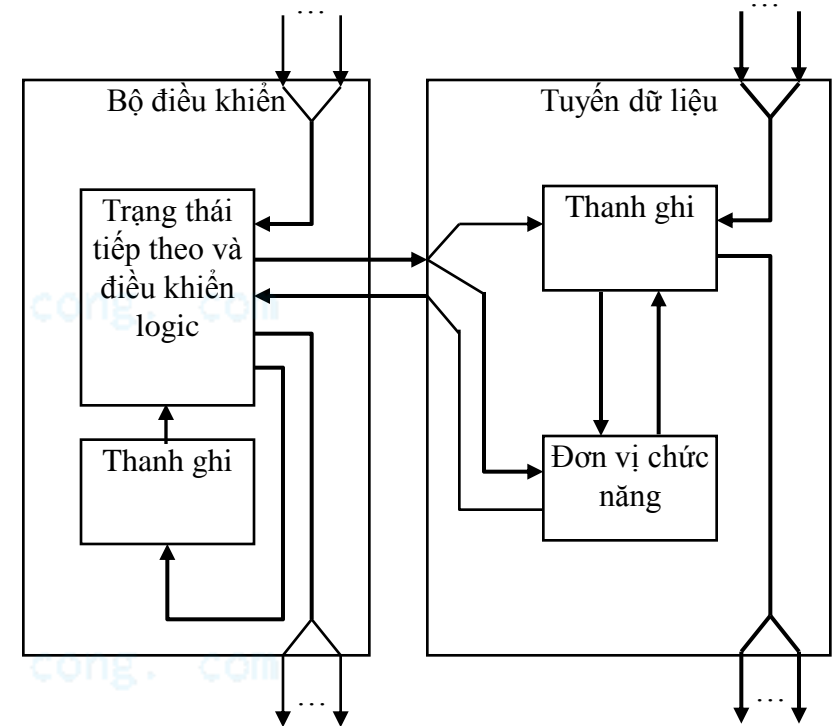


Bảng trạng thái điều khiển cho ví dụ Tìm ước số chung lớn nhất (GCD)

Inputs							Outputs								
Q3	Q2	Q1	Q0	x_neq_y	x_lt_y	go_i	I3	I2	I1	I0	x_sel	y_sel	x_ld	y_ld	d_ld
0	0	0	0	*	*	*	0	0	0	1	X	X	0	0	0
0	0	0	1	*	*	0	0	0	1	0	X	X	0	0	0
0	0	0	1	*	*	1	0	0	1	1	X	X	0	0	0
0	0	1	0	*	*	*	0	0	0	1	X	X	0	0	0
0	0	1	1	*	*	*	0	1	0	0	0	X	1	0	0
0	1	0	0	*	*	*	0	1	0	1	X	0	0	1	0
0	1	0	1	0	*	*	1	0	1	1	X	X	0	0	0
0	1	0	1	1	*	*	0	1	1	0	X	X	0	0	0
0	1	1	0	*	0	*	1	0	0	0	X	X	0	0	0
0	1	1	0	*	1	*	0	1	1	1	X	X	0	0	0
0	1	1	1	*	*	*	1	0	0	1	X	1	0	1	0
1	0	0	0	*	*	*	1	0	0	1	1	X	1	0	0
1	0	0	1	*	*	*	1	0	1	0	X	X	0	0	0
1	0	1	0	*	*	*	0	1	0	1	X	X	0	0	0
1	0	1	1	*	*	*	1	1	0	0	X	X	0	0	1
1	1	0	0	*	*	*	0	0	0	0	X	X	0	0	0
1	1	0	1	*	*	*	0	0	0	0	X	X	0	0	0
1	1	1	0	*	*	*	0	0	0	0	X	X	0	0	0
1	1	1	1	*	*	*	0	0	0	0	X	X	0	0	0

Hoàn thiện thiết kế bộ xử lý chức năng đơn chuyên biệt GCD

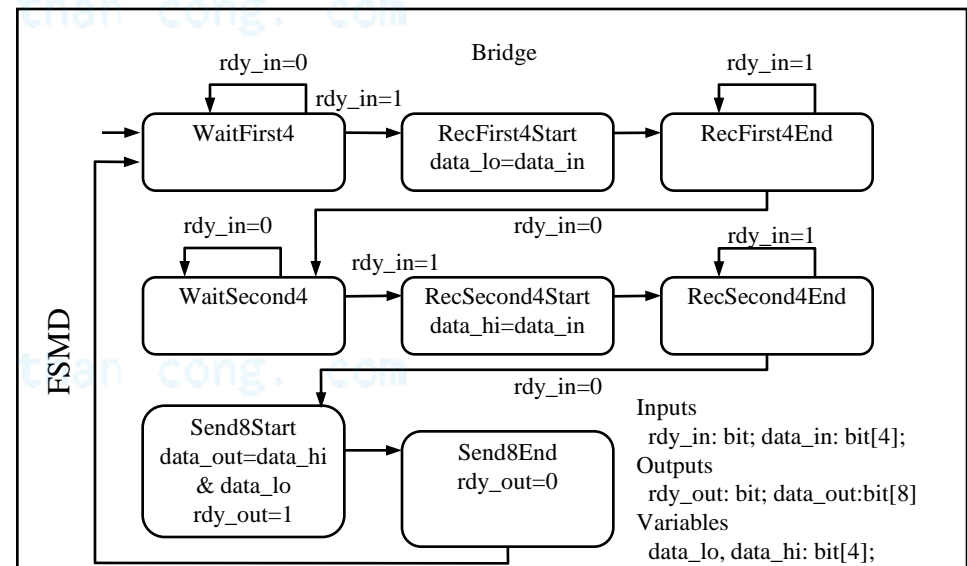
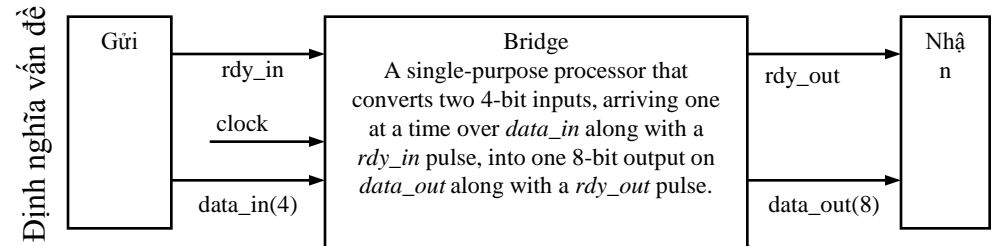
- Chúng ta đã tạo ra tuyến dữ liệu
- Chúng ta đã có bảng trạng thái và điều khiển logic
 - Bộ phận bên trái là thiết kế logic
- Không phải là một thiết kế tối ưu, nhưng chúng ta đã thực hiện các bước thiết kế cơ bản



Bên trong bộ điều khiển và tuyến dữ liệu

Thiết kế bộ xử lý chức năng đơn chuyên biệt mức chuyển đổi thanh ghi (RT)

- Chúng ta thường bắt đầu với một giản đồ trạng thái
 - Khác với thuật toán
 - Việc lặp lại chu trình thường chỉ chú ý đến chức năng
- Ví dụ
 - Một bộ chuyển đổi bus biến đổi một bus 4-bit sang bus 8-bit
 - Bắt đầu với FSMĐ
 - Thường được biết như mức chuyển đổi thanh ghi (RT)
 - Bài tập: hoàn thành thiết kế



Tối ưu bộ xử lý chức năng đơn

- Tối ưu là nhiệm vụ tạo ra các giá trị của thông số thiết kế phù hợp nhất có thể
- Các cơ hội tối ưu
 - Chương trình nguồn
 - FSMD
 - Tuyến dữ liệu
 - FSM

Tối ưu chương trình nguồn

- Phân tích thuộc tính của chương trình và tìm ra các khu vực có thể cải tiến được
 - Số phép tính
 - Kích thước biến
 - Độ phức tạp về thời gian và không gian
 - Các toán hạng sử dụng
 - Ví dụ phép nhân và phép chia rất phức tạp về độ tính toán

Tối ưu chương trình nguồn (cont')

original program

```
0: int x, y;  
1: while (1) {  
2:   while (!go_i);  
3:   x = x_i;  
4:   y = y_i;  
5:   while (x != y) {  
6:     if (x < y)  
7:       y = y - x;  
8:     else  
9:       x = x - y;  
10:  }  
11:  d_o = x;  
12: }
```

replace the subtraction
operation(s) with modulo
operation in order to speed
up program

optimized program

```
0: int x, y, r;  
1: while (1) {  
2:   while (!go_i);  
3:   // x must be the larger number  
4:   if (x_i >= y_i) {  
5:     x=x_i;  
6:     y=y_i;  
7:   }  
8:   else {  
9:     x=y_i;  
10:    y=x_i;  
11:  }  
12:  while (y != 0) {  
13:    r = x % y;  
14:    x = y;  
15:    y = r;  
16:  }  
17:  d_o = x;  
18: }
```

GCD(42, 8) - 9 iterations to complete the loop

x and y values evaluated as follows : (42, 8), (43, 8),
(26,8), (18,8), (10, 8), (2,8), (2,6), (2,4), (2,2).

GCD(42,8) - 3 iterations to complete the loop

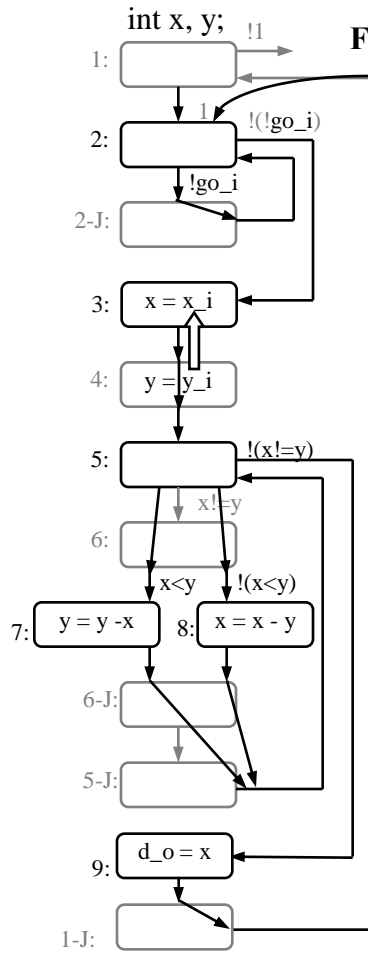
x and y values evaluated as follows: (42, 8), (8,2),
(2,0)

Tối ưu FSMD

- Các trường hợp có thể cải tiến
 - Ghép trạng thái
 - Trạng thái không có biến đổi (chuyển trạng thái) có thể bỏ đi
 - Trạng thái với các hoạt động độc lập có thể ghép
 - Tách trạng thái
 - Các trạng thái yêu cầu các toán hạng phức tạp ($a*b*c*d$) có thể tách thành các trạng thái nhỏ hơn để giảm kích thước phần cứng
 - Lập lịch

Tối ưu FSMD (cont.)

FSMD ban đầu



Loại bỏ trạng thái 1 – quá trình chuyển có giá trị không đổi

Ghép trạng thái 2 và trạng thái 2J – không có hoạt động lặp trong nó

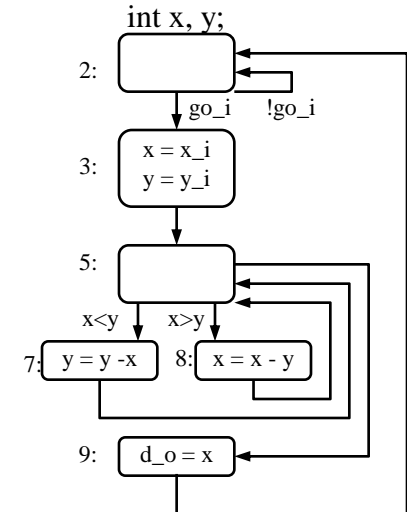
Ghép trạng thái 3 và 4 – hoạt động gán độc lập với các hoạt động khác

Ghép trạng thái 5 và 6 – việc chuyển sang trạng thái 6 có thể thực hiện ở trạng thái 5

Loại bỏ trạng thái 5J và 6J – việc chuyển từ mỗi trạng thái có thể thực hiện từ trạng thái 7 hoặc 8, tương ứng

Loại bỏ trạng thái 1-J – chuyển từ trạng thái 1-J có thể thực hiện trực tiếp từ trạng thái 9

FSMD tối ưu



Tối ưu tuyến dữ liệu

- Chia sẻ các bộ phận chức năng
 - Ghép từng chức năng, như thực hiện trong phần trước, là không cần thiết
 - Nếu có cùng hoạt động xảy ra trong các trạng thái khác nhau, chúng có thể chia sẻ một đơn vị chức năng đơn
- Các bộ phận đa chức năng
 - ALUs hỗ trợ một loạt các hoạt động, nó có thể chia sẻ các hoạt động trong các trạng thái khác nhau

Tối ưu FSM

- Mã hóa trạng thái
 - Là nhiệm vụ gán một mẫu bit duy nhất tới mỗi trạng thái trong một FSM
 - Kích thước của thanh ghi trạng thái và mạch tổ hợp biến đổi
 - Có thể được xem xét như một vấn đề sắp xếp
- Tối ưu trạng thái
 - Là nhiệm vụ ghép các trạng thái tương đồng thành một trạng thái duy nhất
 - Trạng thái tương đồng nếu kết hợp tất cả các đầu vào, hai trạng thái tạo ra cùng đầu ra và cùng chuyển tới trạng thái kế tiếp

Tóm tắt

- Bộ xử lý chức năng đơn chuyên biệt
 - Kỹ thuật thiết kế trực tiếp
 - Có thể được xây dựng để thực hiện các thuật toán
 - Thường bắt đầu với FSMD
 - Công cụ CAD có thể trợ giúp đắc lực

cuu duong than cong. com