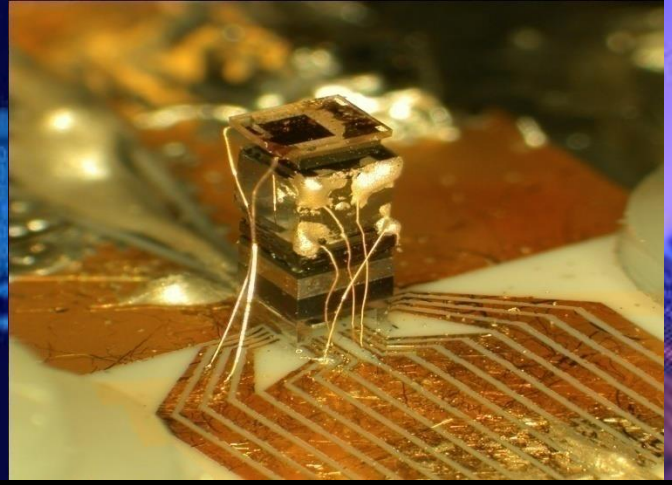
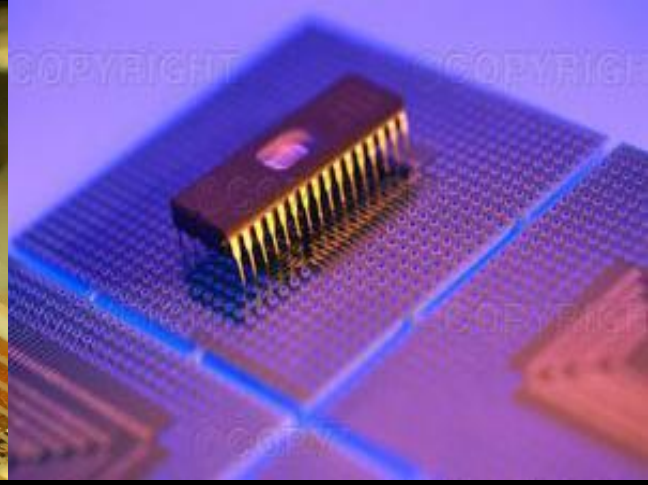



HỆ NHÚNG

Ngô Lam Trung
Bộ môn Kỹ thuật Máy tính
Viện CNTT&TT- ĐHBK HN

Chương 2: Tổ chức cơ bản của hệ thống nhúng

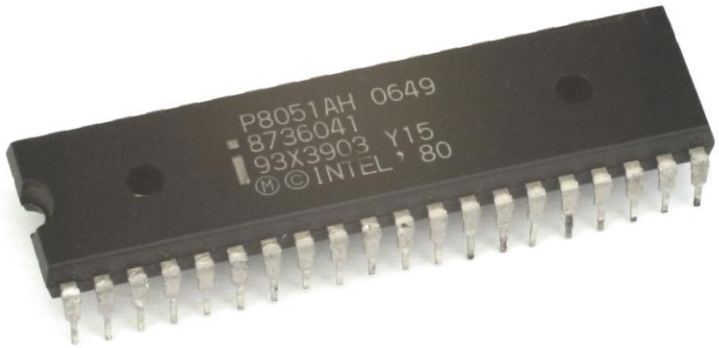
2.1 Phần cứng

2.2 Tập lệnh

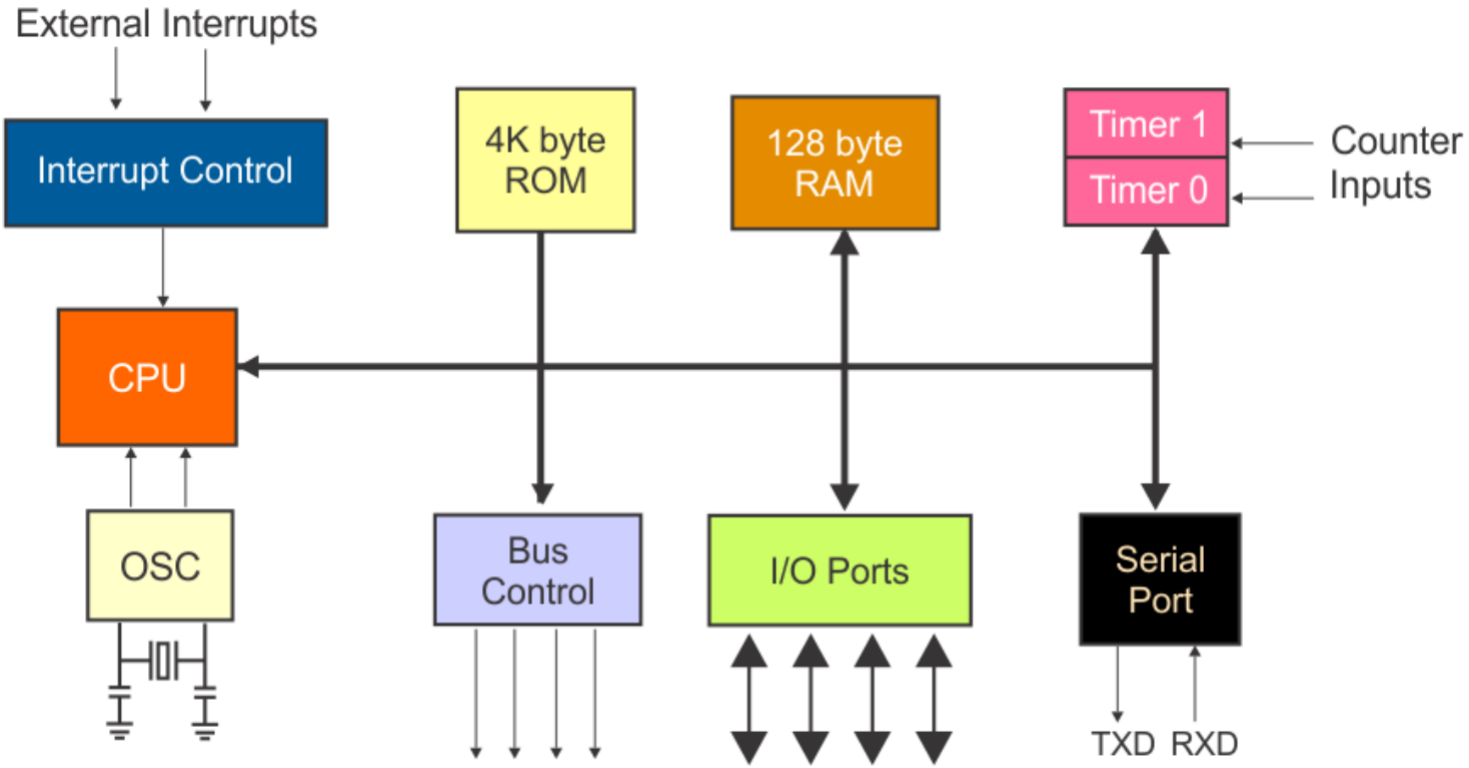
2.3 Lập trình hợp ngữ

2.2.1. Giới thiệu 8051

- Được Intel giới thiệu năm 1981 với tên MCS-51, rất phổ biến cho tới đầu 1990.
- Intel cho phép các hãng khác phát triển vi điều khiển dựa trên lõi của 8051, do đó rất nhiều phiên bản VĐK tương thích 8051 đã được chế tạo và còn phổ thông tới ngày nay.
- Một số vendor tiêu biểu: Atmel, SiliconLab, TI…



Kiến trúc 8051

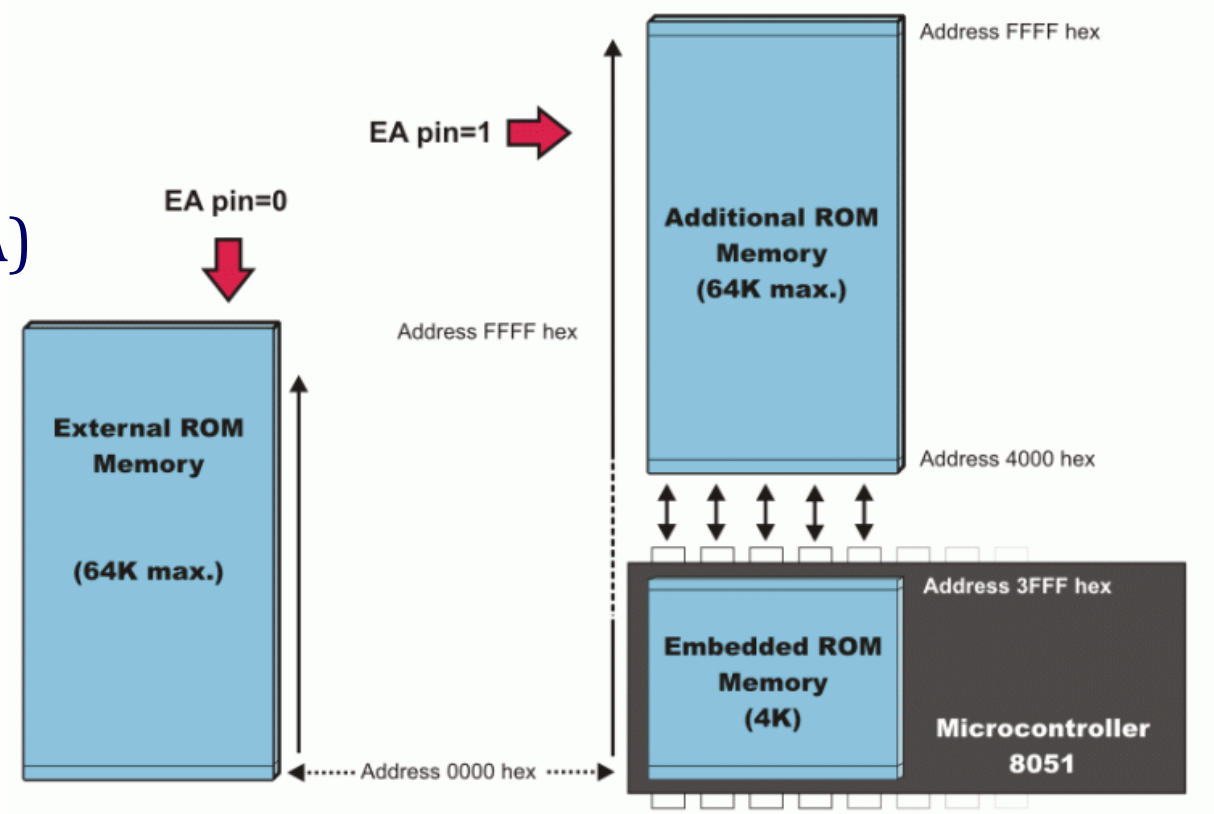


Vi xử lý 8 bit, CPU 24MHz max
128 Bytes RAM, 4KB ROM
Kiến trúc Havard

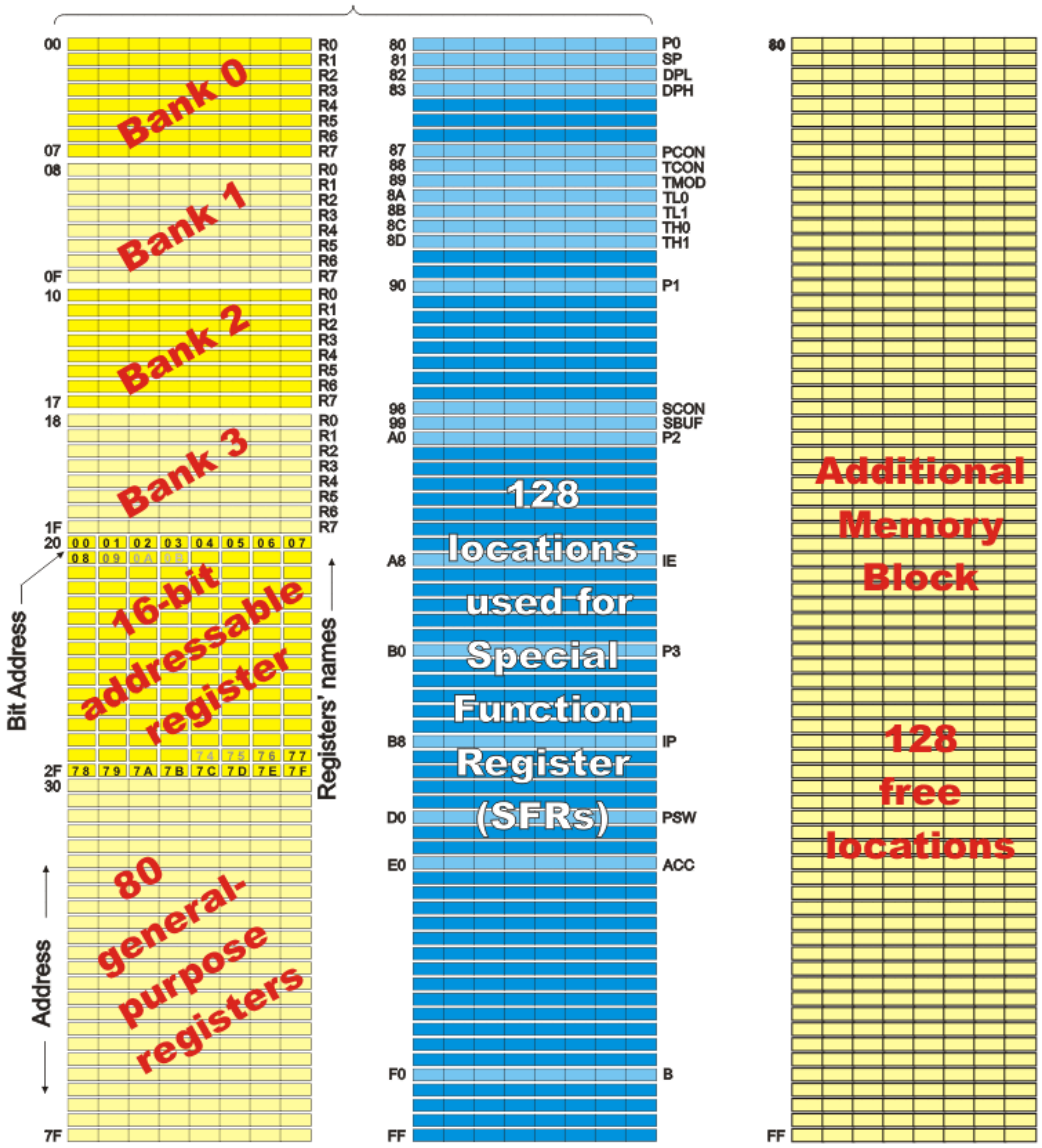
4 cổng 8 bit: P0, P1, P2, P3
1 UART
2 Interrupt (External)
2 Timer/Counter

Tổ chức bộ nhớ

- ROM: Bộ nhớ chứa chương trình điều khiển (firmware)
- RAM: Bộ nhớ chứa dữ liệu và các thanh ghi SFRs phục vụ hoạt động của CPU
- Không gian địa chỉ riêng biệt
- 16 bit địa chỉ
- Có thể ghép nối bộ nhớ ngoài (chân EA)



8051 data memory map



Tập thanh ghi

- R0-R7: các thanh ghi đa năng, được ánh xạ vào 1 trong 4 bank nhớ
- Các thanh ghi đặc biệt:
  - ACC: thanh ghi accumulator
  - B: thanh ghi đa năng, và dùng cho lệnh MUL và DIV
  - DPH/DPL
  - PC
  - SFRs cho các chức năng điều khiển ngoại vi
  - PSW: program status word

Program Status Word

(MSB)							(LSB)
CY	AC	F0	RS1	RS0	OV	-	P

Symbol	Position	Name and Significance
CY	PSW.7	Carry flag
AC	PSW.6	Auxiliary Carry flag. (For BCD operations.)
F0	PSW.5	Flag 0 (Available to the user for general purposes.)
RS1	PSW.4	Register bank Select control bits 1 & 0. Set/cleared by software to determine working register bank (see Note).
RS0	PSW.3	
OV	PSW.2	Overflow flag.
-	PSW.1	(reserved)
P	PSW.0	Parity flag. Set/cleared by hardware each instruction cycle to indicate and odd/even number of "one" bits in the accumulator, i.e., even parity.

(0.0)-Bank 0(00H-07H)
(0.1)-Bank 1(08H-0FH)
(1.0)-Bank 2(10H-17H)
(1.1)-Bank 3(18H-1FH)

Special Function Registers (SFRs)

0F8H									0FFH
0F0H	B 00000000								0F7H
0E8H									0EFH
0E0H	ACC 00000000								0E7H
0D8H									0DFH
0D0H	PSW 00000000								0D7H
0C8H									0CFH
0C0H									0C7H
0B8H	IP XX000000								0BFH
0B0H	P3 11111111								0B7H
0A8H	IE 0X000000								0AFH
0A0H	P2 11111111		AUXR1 XXXXXXX0				WDTRST XXXXXXXX		0A7H
98H	SCON 00000000	SBUF XXXXXXXX							9FH
90H	P1 11111111								97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000	AUXR XXX00XX0		8FH
80H	P0 11111111	SP 00000111	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000		PCON 0XXX0000	87H

Tập lệnh 8051

- Các nhóm thao tác lệnh:
  - Lệnh chuyển dữ liệu: MOV, PUSH, POP
  - Lệnh số học: ADD, SUB, INC, DEC, MUL, DIV
  - Lệnh logic: ANL, ORL, XRL, CLR
  - Lệnh rẽ nhánh: LCALL, RET, LJMP, JZ, JNZ, NOP
- Các chế độ địa chỉ (Addressing)
  - Địa chỉ tức thì
  - Địa chỉ thanh ghi
  - Địa chỉ trực tiếp
  - Địa chỉ gián tiếp

Các chế độ địa chỉ

- Trực tiếp: toán hạng là địa chỉ ngăn nhớ

ADD A, 7FH

- Gián tiếp: toán hạng là ngăn nhớ, địa chỉ để trong thanh ghi

ADD A, @R0

- Thanh ghi: toán hạng là thanh ghi

ADD A, R7

- Hằng số:

ADD A, #127

Biểu diễn dữ liệu

- Các hằng (literal)
  - Số nhị phân: 1011b, 1011B, ...
  - Số thập phân: 35, 35d, 35D, ...
  - Số Hexa: 4Ah, 0ABCDh, 0FFFFH, ...
  - Kí tự: "A", 'HELLO', "Bach Khoa", ...
- Tất cả các kiểu dữ liệu trên sau đó đều được trình dịch Assembler dịch ra mã nhị phân.
- Mỗi kí tự được dịch thành mã ASCII tương ứng
  - Chương trình không phân biệt 'A' với 41h hay 65

Kiểu dữ liệu

- Chỉ có một kiểu dữ liệu là kiểu dữ liệu 8 bit
- Để định nghĩa một dữ liệu kiểu byte, cần sử dụng chỉ dẫn **DB**

```
        ORG    500H
DATA1:  DB     28                ;Số thập phân (=1CH)
DATA2:  DB     00110101B         ;Số nhị phân (=35H)
DATA3:  DB     39H               ;Số dạng Hexa
        ORG    510H
DATA4:  DB     "2591"            ;Các số ASCII
        ORG    518H
DATA5:  DB "My name is Binh"     ;Ký tự ASCII
```

Các chỉ dẫn

- ORG: Báo địa chỉ bắt đầu
- EQU: dùng để định nghĩa hằng số
  - VD: COUNT EQU 25
- END: báo kết thúc file mã nguồn

Tập lệnh 8051

- Cú pháp lệnh hợp ngữ gồm có 4 thành phần

  [nhãn:] [mã thao tác] [các toán hạng] [;chú giải]

  Ví dụ:

  bat_dau: MOV A,#25 ;Khởi gán A=25

Lệnh chuyển dữ liệu	
Lệnh	**Giải thích**
MOV đích, nguồn	Đích = nguồn (Bộ nhớ trong)
MOVX đích, nguồn	Đích = nguồn (Thao tác bộ nhớ ngoài)
PUSH	Đẩy dữ liệu vào đỉnh ngăn xếp
POP	Lấy dữ liệu từ đỉnh ngăn xếp
XCH	Tráo đổi dữ liệu
XCHD	Tráo đổi dữ liệu (4 bit thấp)

Tập lệnh 8051

Mnemonic	Operation	Addressing Modes				Execution Time @ 12MHz (µs)
		Dir	Ind	Reg	Imm	
MOV A, <src>	A = <src>	X	X	X	X	1
MOV <dest>, A	<dest> = A	X	X	X		1
MOV <dest>, <src>	<dest> = <src>	X	X	X	X	2
MOV DPTR, # data 16	DPTR = 16-bit immediate constant				X	2
PUSH <src>	INC SP: MOV "@SP", <scr>	X				2
POP <dest>	MOV <dest>, "@SP": DEC SP	X				2
XCH A, <byte>	ACC and <byte> Exchange Data	X	X	X		1
XCHD A, @Ri	ACC and @ Ri exchange low nibbles		X			1

Thanh ghi đích có thể là A, B, R0-R7

```
MOV A,#55H ; reg. A ← 55h
MOV R6,#12 ; R6←12=0CH
MOV R0,A     ; A=55H, R0=55H
MOV R5,#0F9H ;load F9H into R5
MOV A,17H    ;load the value
MOV A,#’4’   ;A = ?
```

Lệnh số học	
Lệnh	**Giải thích**
ADD đích, nguồn	Đích = đích + nguồn
ADDC đích, nguồn	Đích = đích + nguồn + carry
SUBB đích, nguồn	Đích = đích – nguồn - carry
INC nguồn	Đích = đích + 1
DEC nguồn	Đích = đích - 1
MUL AB	A*B
DIV AB	A/B

Tập lệnh 8051

Mnemonic	Operation	Addressing Modes				Mode @12 MHz (µs)
		Dir	Ind	Reg	Imm	
ADD A, <byte>	A = A + <byte>	X	X	X	X	
ADDC A, <byte>	A = A + <byte> + C	X	X	X	X	1
SUBB A, <byte>	A = A – <byte> – C	X	X	X	X	1
INC A	A = A + 1	Accumulator only				1
INC <byte>	<byte> = <byte> + 1	X	X	X		1
INC DPTR	DPTR = DPTR + 1	Data Pointer only				2
DEC A	A = A – 1	Accumulator only				1
DEC <byte>	<byte> = <byte> – 1	X	X	X		1
MUL AB	B:A = B × A	ACC and B only				4
DIV AB	A = Int [A/B] B = Mod [A/B]	ACC and B only				4
DA A	Decimal Adjust	Accumulator only				1

Lệnh số học

- Toán hạng đích bắt buộc là thanh ghi A

```
MOV A,#25H    ;load 25H into A
MOV R2,#34H   ;load 34H into R2
ADD A,R2      ;add R2 to A=A+R2
```

➔ R2, A = ?

```
ADD R0,A      ;không hợp lệ
```

Lệnh logic	
Lệnh	**Giải thích**
ANL	Lệnh “AND”
ORL	Lệnh “OR”
XRL	Lệnh “XOR”
CLR	Xóa bit
RL, RLC	Lệnh quay trái
RR, RRC	Lệnh quay phải

Tập lệnh 8051

Mnemonic	Operation	Addressing Modes				Execution Time @ 12MHz (µs)
		Dir	Ind	Reg	Imm	
ANL A, <byte>	A = A AND <byte>	X	X	X	X	1
ANL <byte>, A	<byte> = <byte> AND A	X				1
ANL <byte>, # data	<byte> = <byte> AND # data	X				2
ORL A, <byte>	A = A OR <byte>	X	X	X	X	1
ORL <byte>, A	<byte> = <byte> OR A	X				1
ORL <byte>, # data	<byte> = <byte> OR # data	X				2
XRL A, <byte>	A = A XOR <byte>	X	X	X	X	1
XRL <byte>, A	<byte> = <byte> XOR A	X				1
XRL <byte>, # data	<byte> = <byte> XOR # data	X				2

Tập lệnh 8051

CLR A	A = 00H	Accumulator only	1
CLP A	A = NOT A	Accumulator only	1
RL A	Rotate ACC Left 1 bit	Accumulator only	1
RLC A	Rotate Left through Carry	Accumulator only	1
RR A	Rotate ACC Right 1 bit	Accumulator only	1
RRC A	Rotate Right through Carry	Accumulator only	1
SWAP A	Swap Nibbles in A	Accumulator only	1

Ví dụ

- Lệnh ANL

  ANL đích, nguồn ; đích=đích AND nguồn

  ❖Mục đích: che, xóa bit

  ❖VD: - Xóa 4 bit thấp của thanh ghi A

  ANL A, #0F0h

- Lệnh ORL

  ORL đích, nguồn ; đích=đích OR nguồn

  ❖Mục đích: thiết lập bit

  ❖VD: - Thiết lập 4 bit cao của thanh ghi A

  ORL A, #0F0h

Các lệnh logic, lệnh quay

- Lệnh XRL

XRL đích, nguồn ; đích=đích XOR nguồn

❖Mục đích: - Xóa thanh ghi (XOR với chính nó)

- Đảo bit (XOR với 1)

❖VD: Xóa thanh ghi A

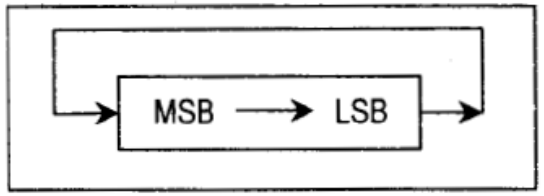
XRL A,A

Đảo các bit của thanh ghi A

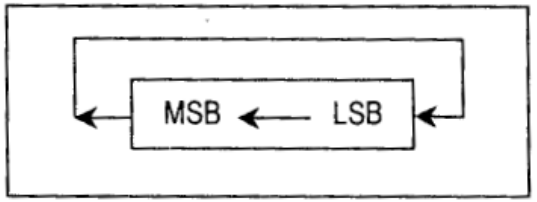
XRL A,#0FFh

Các lệnh logic, lệnh quay

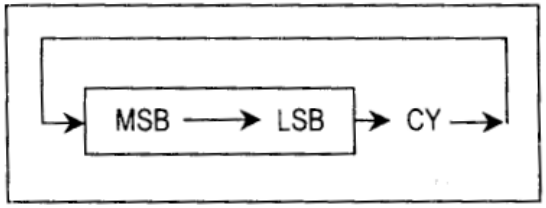
- Lệnh quay
  - Lệnh quay phải: RR A



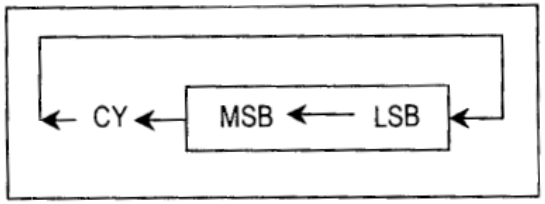
  - Lệnh quay trái: RL A



  - Lệnh quay phải qua cờ nhớ: RRC A



  - Lệnh quay trái qua cờ nhớ: RLC A



Lệnh thao tác với bit

Table 1-9. 8051 Boolean Instructions

Mnemonic	Operation	Execution Time @ 12MHz (µs)
ANL C,bit	C = C AND bit	2
ANL C,/bit	C = C AND (NOT bit)	2
ORL C,bit	C = C OR bit	2
ORL C,/bit	C = C OR (NOT bit)	2
MOV C,bit	C = bit	1
MOV bit,C	bit = C	2
CLR C	C = 0	1
CLR bit	bit = 0	1
SETB C	C = 1	1
SETB bit	bit = 1	1
CPL C	C = NOT C	1
CPL bit	bit = NOT bit	1
JC rel	Jump if C = 1	2
JNC rel	Jump if C = 0	2
JB bit,rel	Jump if bit = 1	2
JNB bit,rel	Jump if bit = 0	2
JBC bit,rel	Jump if bit = 1 ; CLR bit	2

Lệnh rẽ nhánh	
Lệnh	**Giải thích**
ACALL	Gọi chương trình con, địa chỉ 11 bit
LCALL	Gọi chương trình con, địa chỉ 16 bit
RET	Trở về từ chương trình con
JMP	Lệnh nhảy không điều kiện
JZ, JNZ, JB, JNB…	Lệnh nhảy có điều kiện (kiểm tra bit)

Lệnh nhảy, rẽ nhánh

Mnemonic	Operation	Execution Time @ 12MHz (µs)
JMP addr	Jump to addr	2
JMP @A + DPTR	Jump to A + DPTR	2
CALL addr	Call subroutine at addr	2
RET	Return from subroutine	2
RETI	Return from interrupt	2
NOP	No operation	1

Mnemonic	Operation	Addressing Modes				Execution Time @ 12MHz (µs)
		DIR	IND	REG	IMM	
JZ rel	Jump if A = 0	Accumulator only				2
JNZ rel	Jump if A ≠ 0	Accumulator only				2
DJNZ <byte>,rel	Decrement and jump if not zero	X		X		2
CJNZ A,<byte>,rel	Jump if A = <byte>	X			X	2
CJNE <byte>,#data,rel	Jump if <byte> = #data		X	X		2

Các lệnh rẽ nhánh

- Lệnh nhảy có điều kiện: JZ, JNZ, DJNZ, JC, JNC, JB, JNB
- Lệnh nhảy không điều kiện: SJMP (nhảy ngắn), LJMP (nhảy dài)
- Ví dụ:

```
        MOV     A,R5            ;A=R5
        JNZ     NEXT            ;Nhảy tới NEXT nếu A khác 0
        MOV     R5,#55h
NEXT:
        ...
```

Lệnh lặp

- Lệnh DJNZ
  - Cú pháp

DJNZ thanh_ghi, nhãn

+ Sau mỗi lần nhảy, giá trị thanh ghi bị giảm đi 1

+ Nếu giá trị thanh ghi vẫn khác 0 thì nhảy tới nhãn

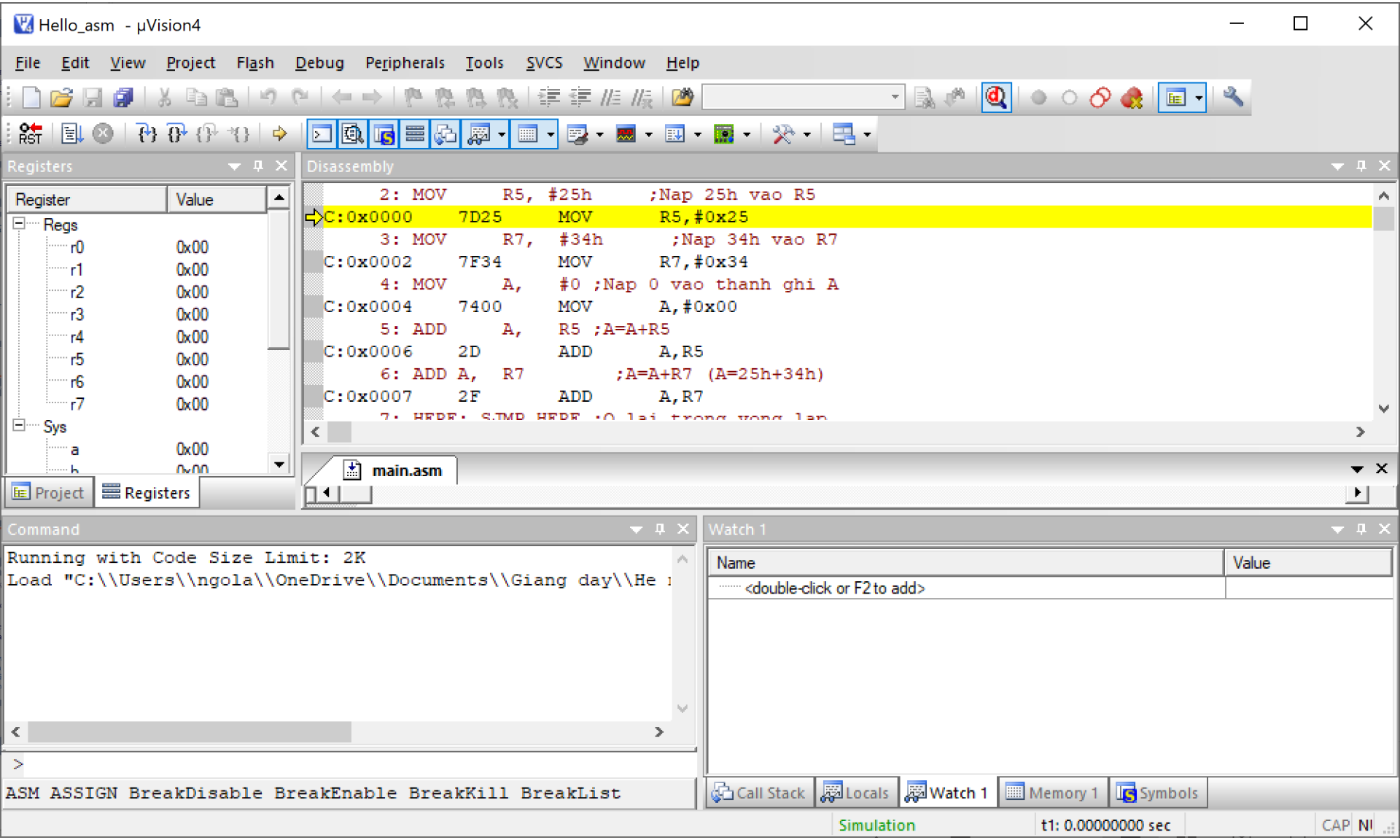
- Ví dụ: Xóa thanh ghi A, cộng 3 vào thanh ghi A 10 lần

```
        MOV   A,#0
        MOV   R2,#10
BACK:   ADD   A,#3
        DJNZ  R2,BACK     ;Lặp 10 lần
        MOV   R5,A        ;Cất A vào R5
```

Ví dụ: chương trình đầy đủ

```
ORG 000                 ;Dia chi bat dau cua chuong trinh
MOV   R5, #25h          ;Nap 25h vao R5
MOV   R7,     #34h      ;Nap 34h vao R7
MOV   A,      #0        ;Nap 0 vao thanh ghi A
ADD   A,      R5        ;A=A+R5
ADD A,        R7        ;A=A+R7 (A=25h+34h)
HERE: SJMP HERE         ;loop forever. WHY?
END
```

Lập trình và mô phỏng với Keil C51



Lập trình cổng vào ra

- 8051 có 4 cổng vào ra GPIO (mỗi cổng 8 bit): P0, P1, P2, P3
- Sau khi reset, các cổng ở chế độ mặc định là cổng ra (output)
- Để các cổng/chân làm việc ở chế độ cổng/chân vào (input) phải tiến hành ghi các bit 1 ra các cổng/chân tương ứng
  - Ví dụ:

```
MOV   P1,#0FF ; Cổng 1 thành cổng vào
SETB  P1.0    ; Chân P1.0 làm chân vào
MOV   P1,#03  ; Chân P1.0 và P1.1 làm chân vào
              ; các chân còn lại làm chân ra
```

Xuất dữ liệu ra cổng/chân

- Xuất dữ liệu ra cổng ra

  MOV tên_cổng, giá trị

  - Ví dụ:
    - ✓MOV P1, #55h
- Xuất dữ liệu ra từng chân
  - Đưa chân lên mức cao:

    SETB bit

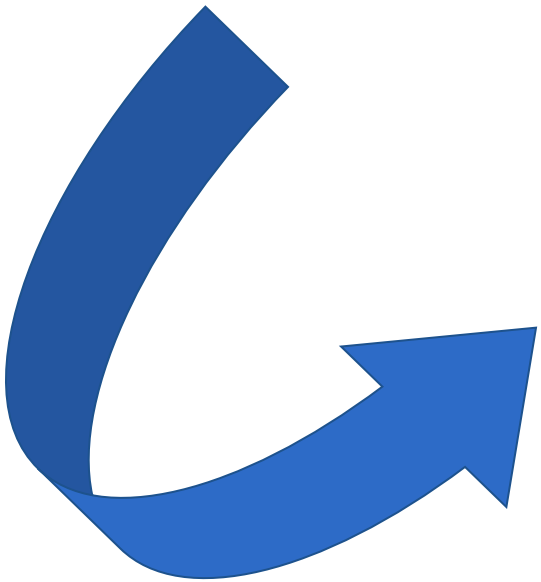
    Ví dụ: SETB P1.0
  - Đưa chân xuống mức thấp:

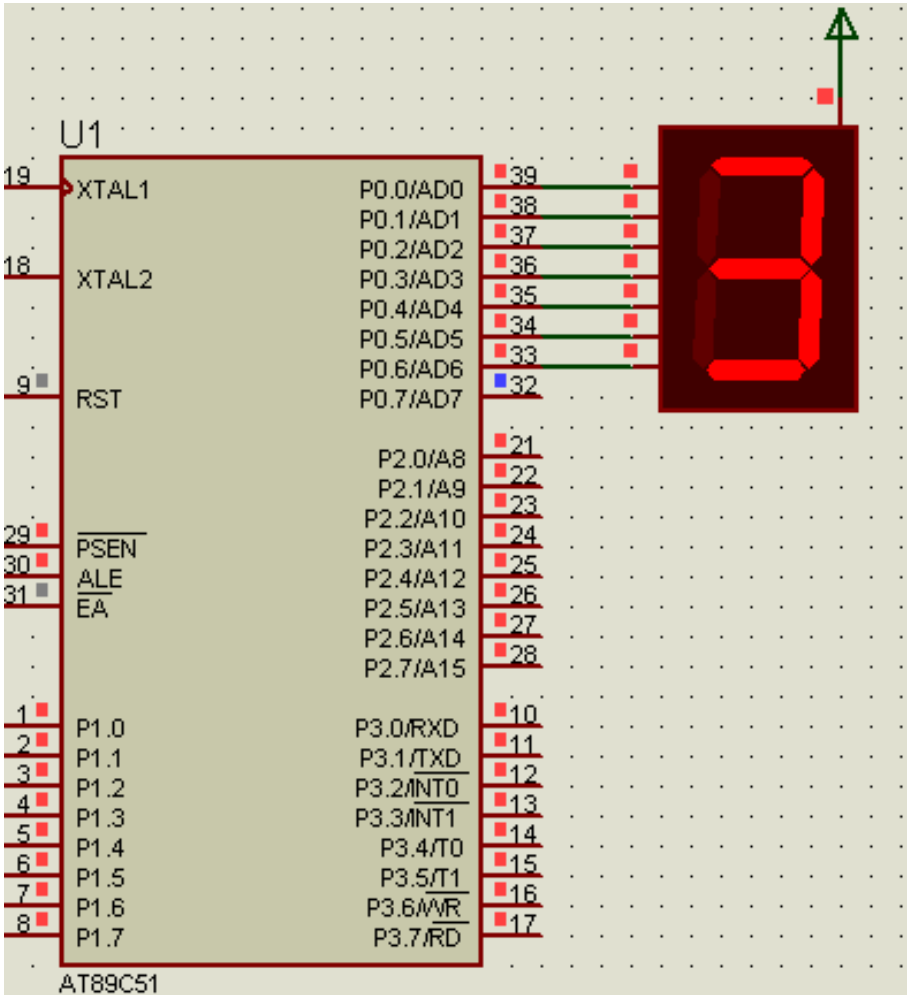
    CLR bit

    Ví dụ: CLR P1.0

Ví dụ xuất dữ liệu ra cổng ra

```
MOV     P0,#30h
```





Đọc dữ liệu từ cổng vào

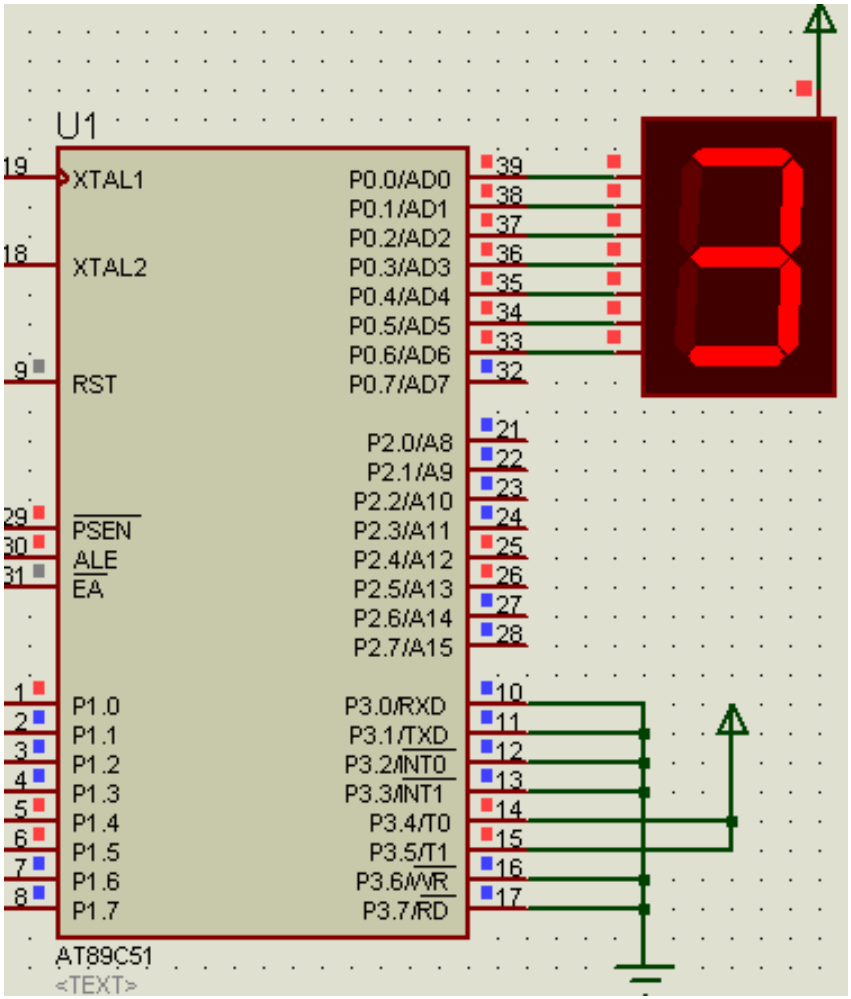
- Bước 1: Thiết lập cổng làm việc ở chế độ cổng vào
- Bước 2: Đọc dữ liệu trên cổng

Ví dụ:

```
MOV     P1, #0FFh
MOV     A, P1 ; Đọc giá trị tại
                ; cổng P1, lưu vào A
MOV     P2, A ; Xuất ra cổng P2
```

Ví dụ đọc dữ liệu từ cổng vào

```
MOV   P3,#0FFh
MOV   A,P3
MOV   P0,A
MOV   P2,A
```



Ví dụ nhấp nháy led

```
ORG 000                 ;Dia chi bat dau cua chuong trinh
AGAIN:
    SETB P1.0           ;Nhap nhay led o chan P1.0
    ACALL DELAY
    CLR P1.0
    ACALL DELAY
    SJMP AGAIN
DELAY:                  ;Tao tre
    MOV R1,#255
LOOP:
    DJNZ R1,LOOP
RET
END
```