

Lab 02

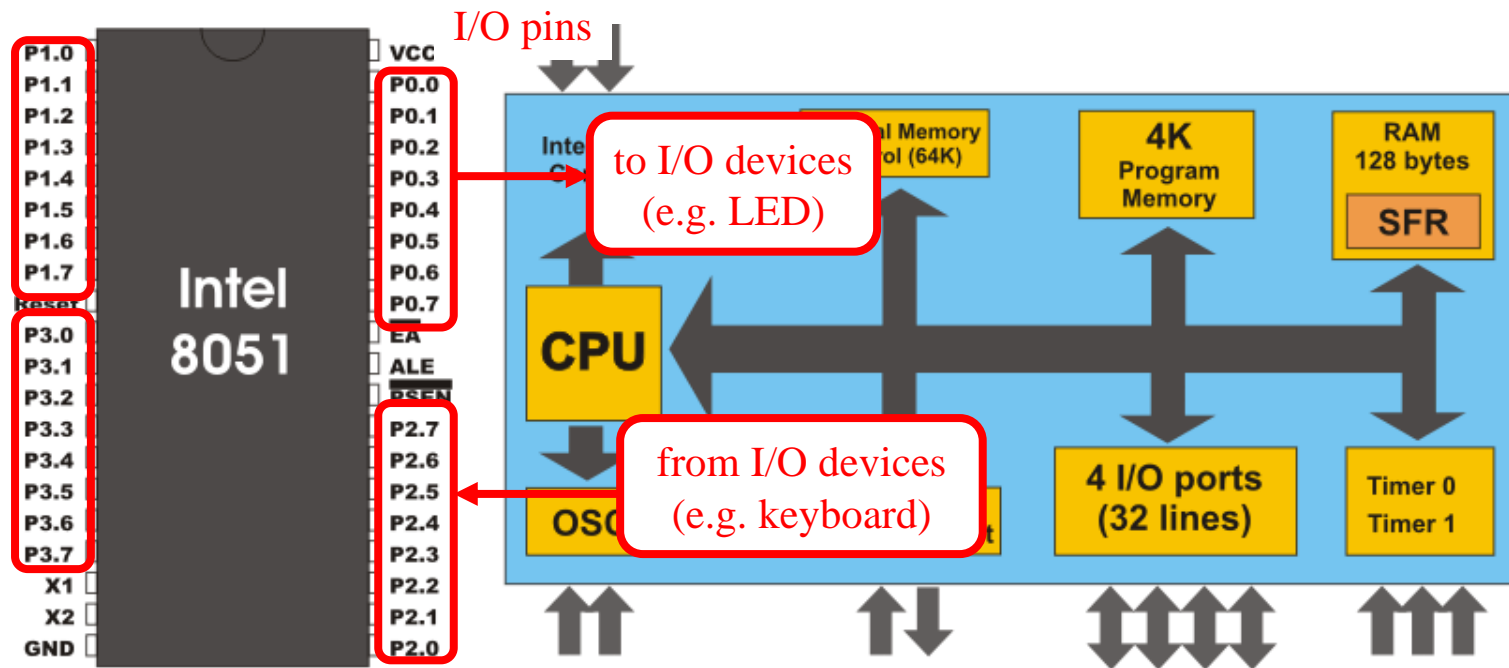
# General Purpose Digital I/O (GPIO)



---

# Objectives of this lab

- (1) to build up your imagination on how a program affects hardware signals
- (2) to learn how to send/receive signals from an application processor to external devices through I/O pads

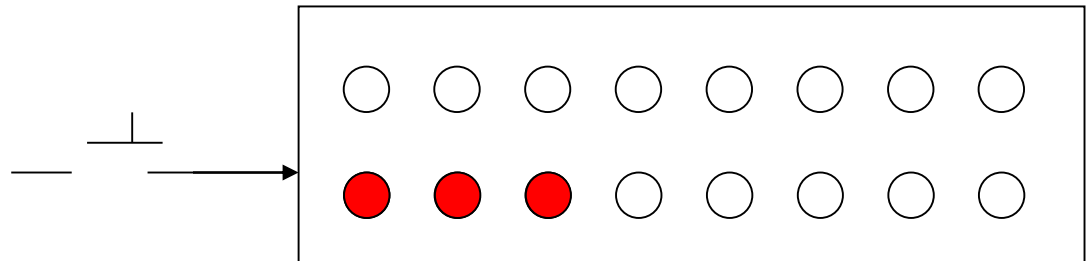




# Your work

---

- design a LED box
  - initial: all LED off
  - the LED runs some pattern after some button pressed
    - you can design your own pattern





# Outline

---

- Basic concepts of I/O control
- I/O model of legacy 8051 processor
- SiliconLab C8051F040 I/O control
- Simplified programming model



# Preparations before the Lab

---

- Read the data sheet of SiliconLab C8051F040 SoC
  - Chap. 17
- Read the schematics of the Big8051 experiment board
  - On LEDs



# Pre-Lab Report

---

- Q1: Explain what is watch-dog timer
- Q2: Explain what is memory-mapped I/O

# Pre-Lab Report (cont'd)

## ■ Q4:

- Read Figure 17.1 of C8051F040 spec and the schematics of Big8051
- List all control signal values to turn-on an LED at P0.0

Value of these  
Control signals

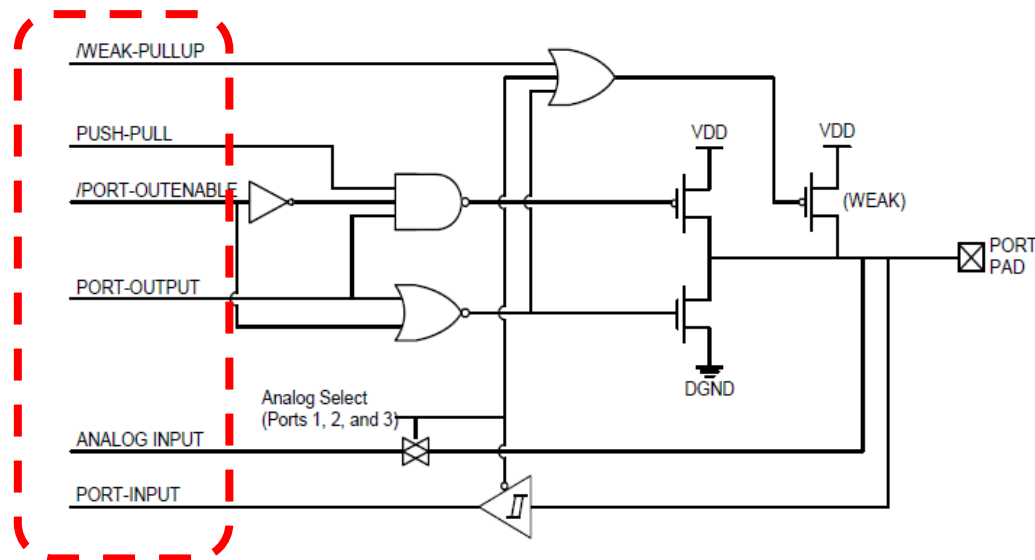


Figure 17.1. Port I/O Cell Block Diagram

# Pre-Lab Report (cont'd)

## ■ Q5:

- Read Figure 17.2 of C8051F040 spec
- List the values of all control registers to configure port P0 as a digital output port

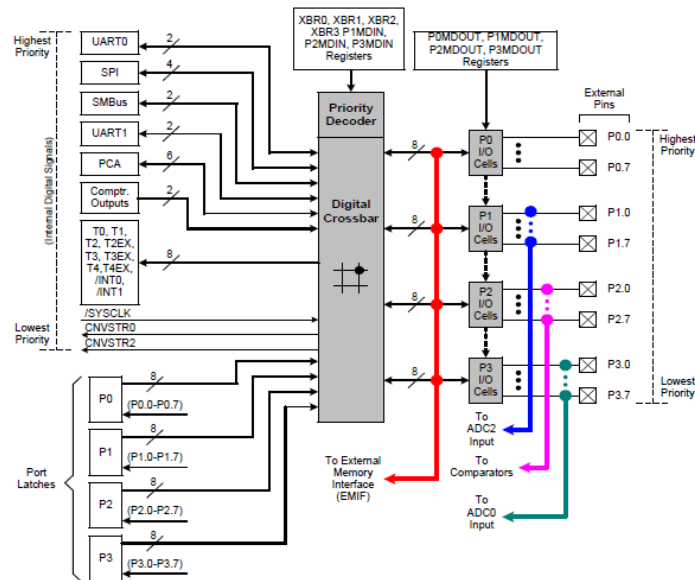


Figure 17.2. Port I/O Functional Block Diagram



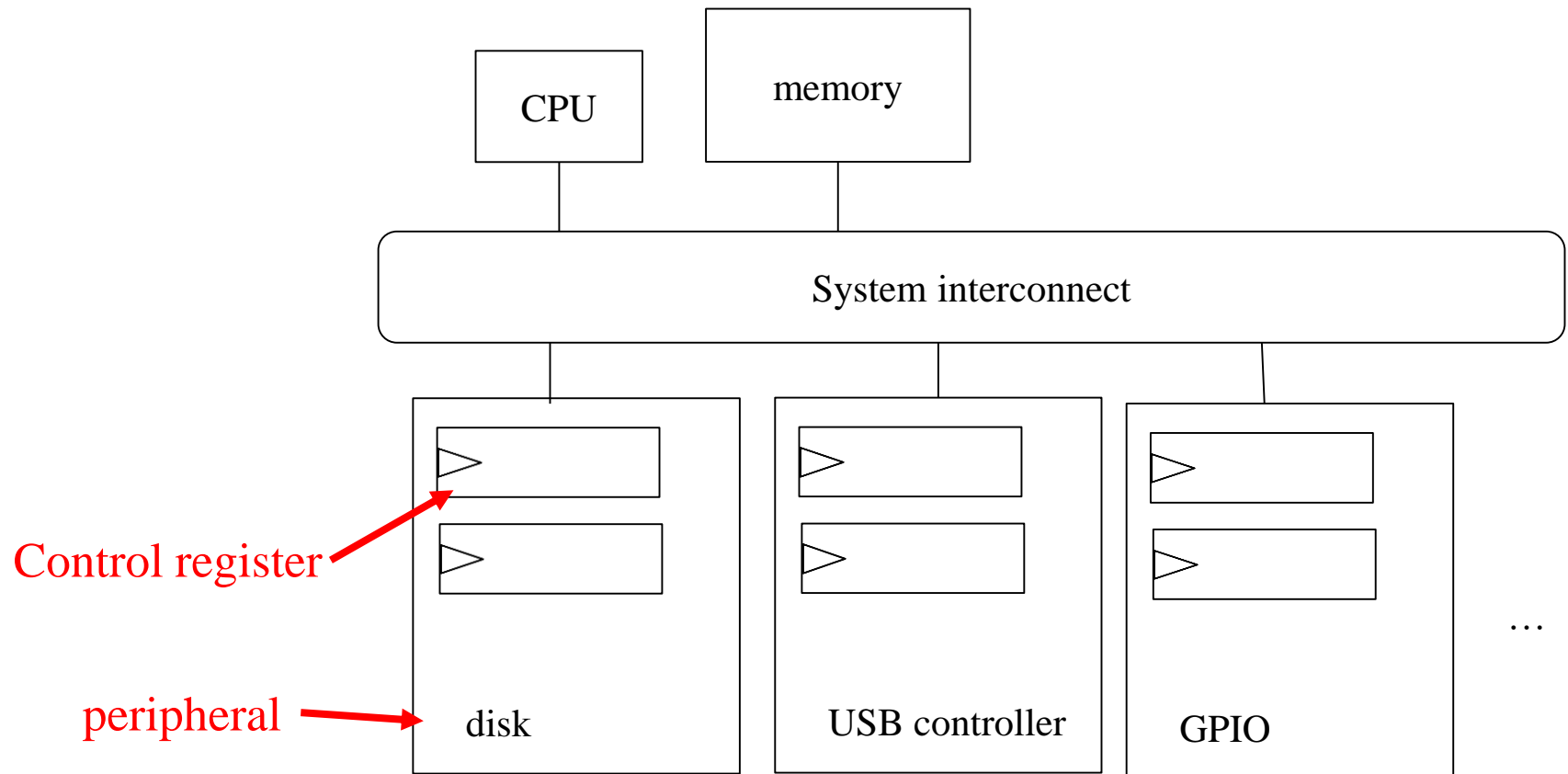


# General I/O Control Model

---

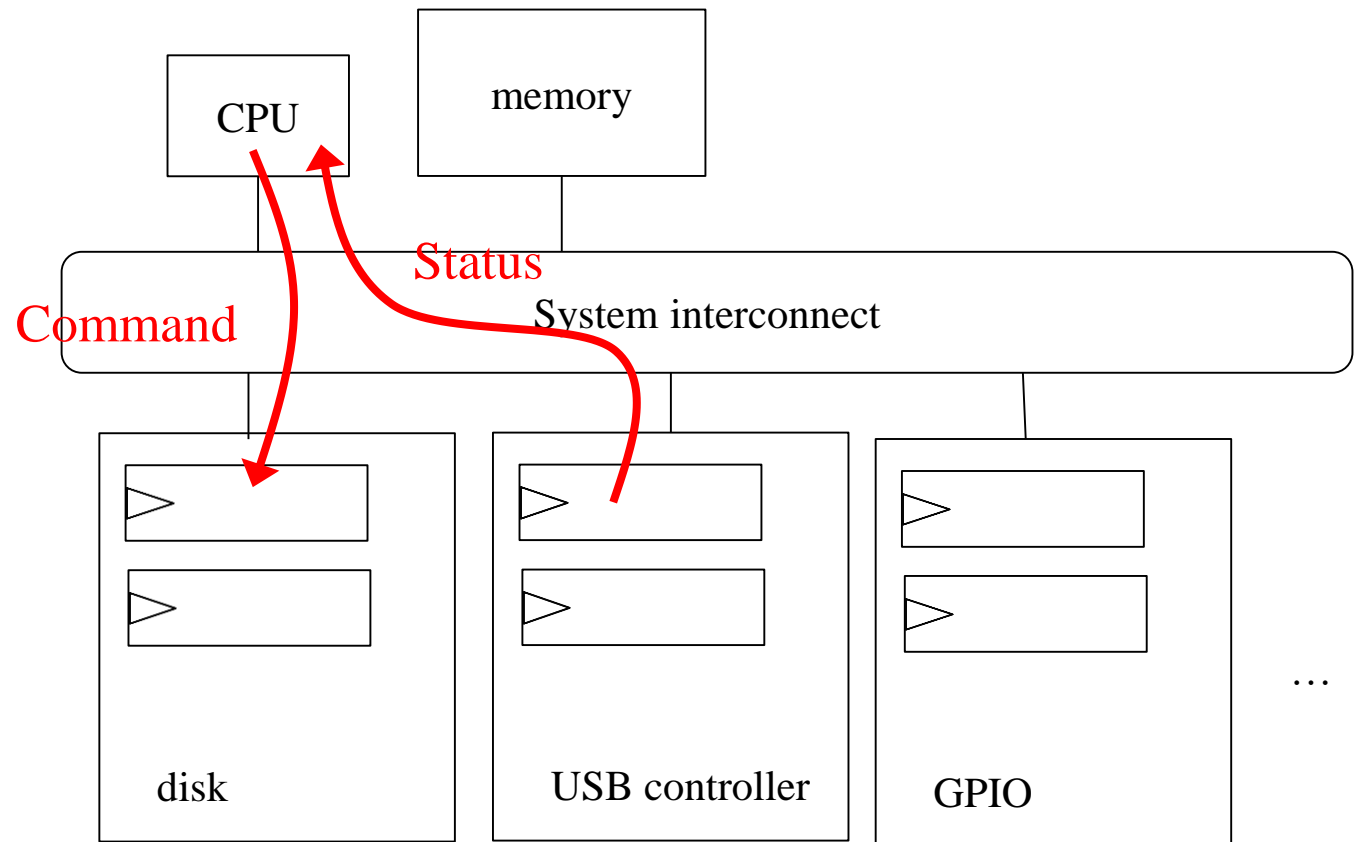
# How a processor commands an I/O peripheral

- Through access control registers



# How a processor commands an I/O peripheral

- Through access control registers



# How to access control registers: the memory-mapped I/O

- Part of the addressing space is assigned to control registers
- Each control register is mapped to some memory address

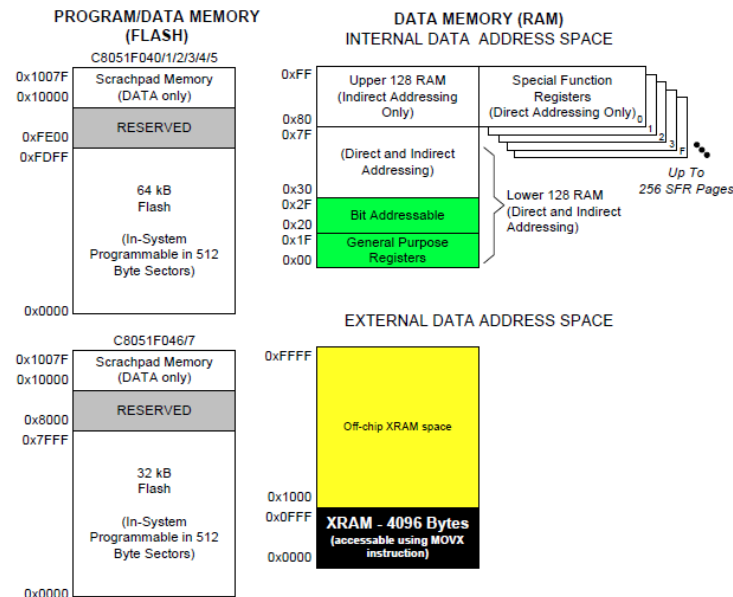
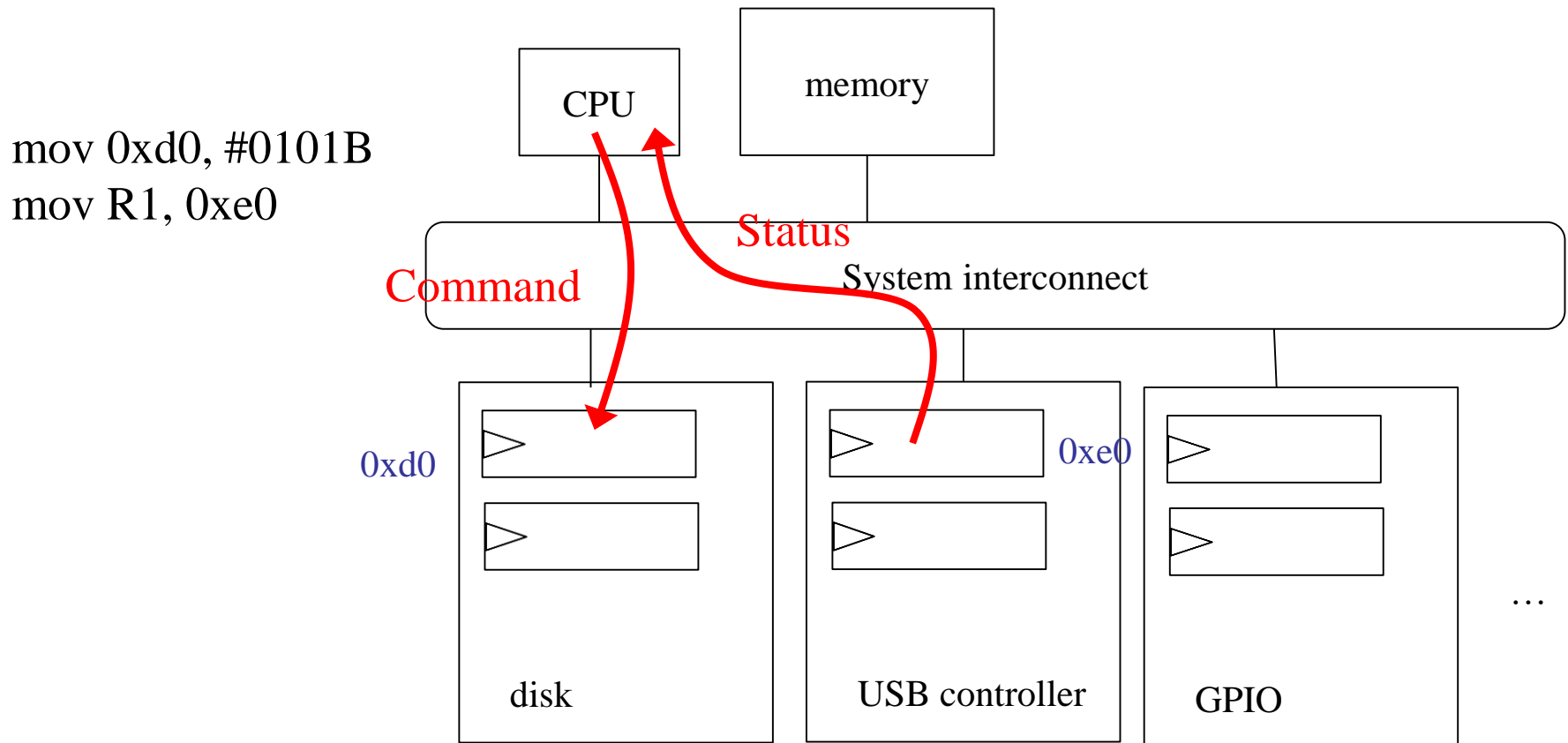


Figure 1.7. On-Chip Memory Map

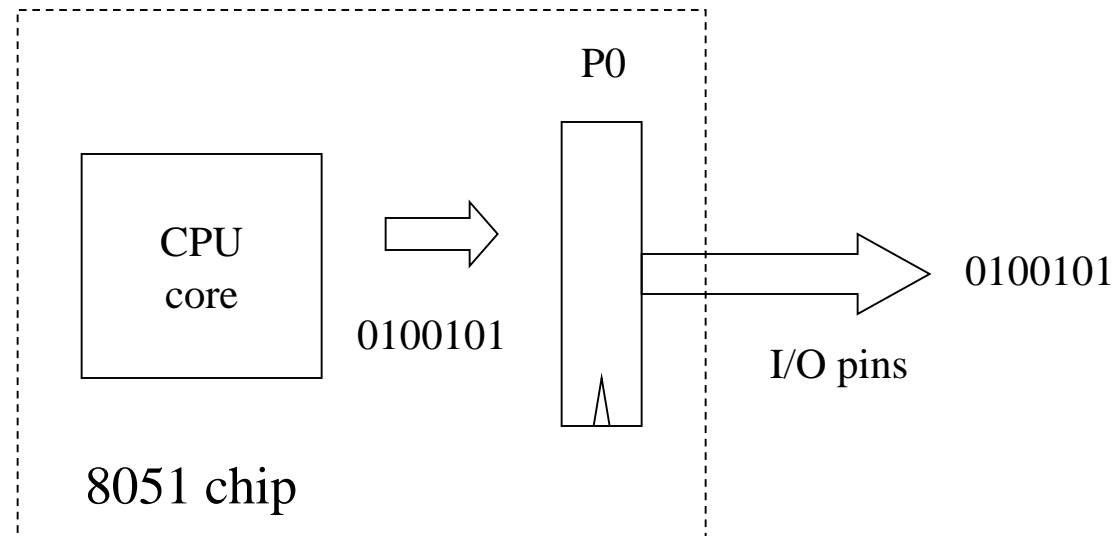
# How a processor commands an I/O peripheral

- Through access control registers



# General Purpose Digital I/O

- the processor assigns/examines the logical status of some I/O pins directly



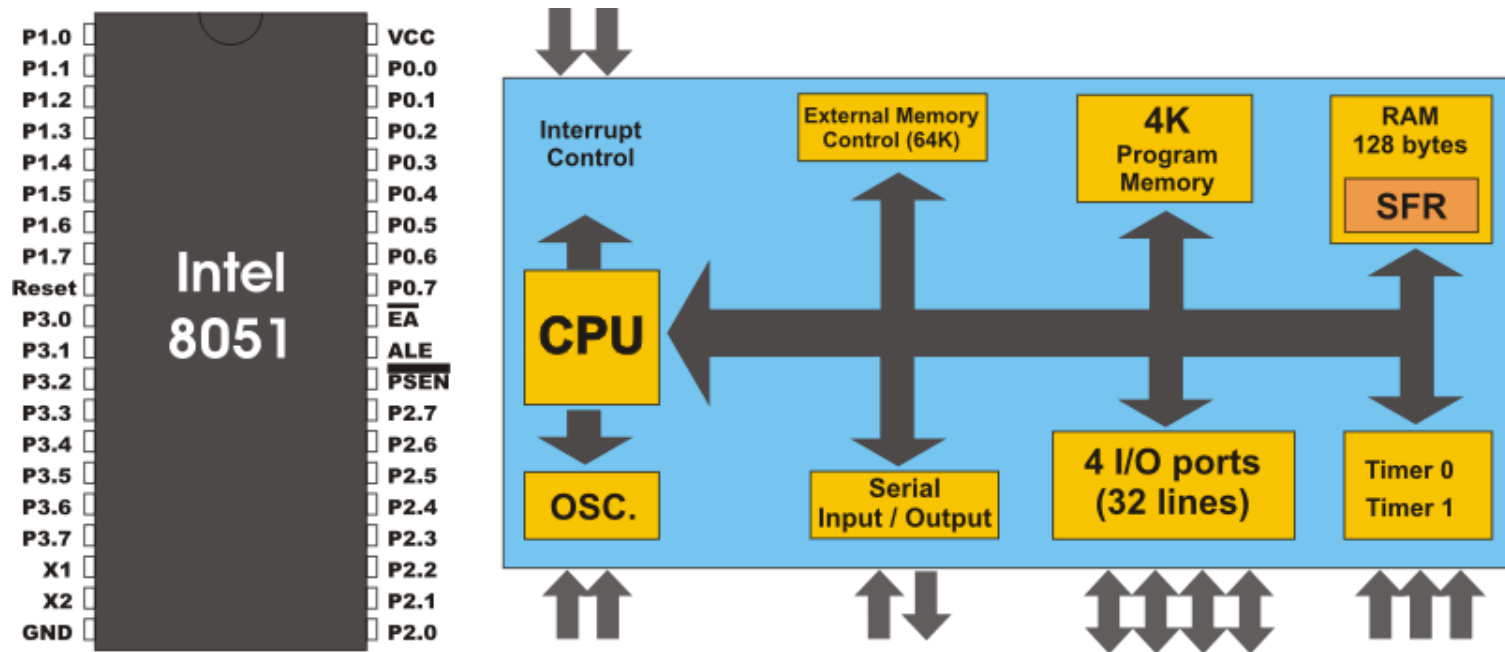
# I/O Model of Legacy 8051 Processor



---

# Features of 8051 I/O

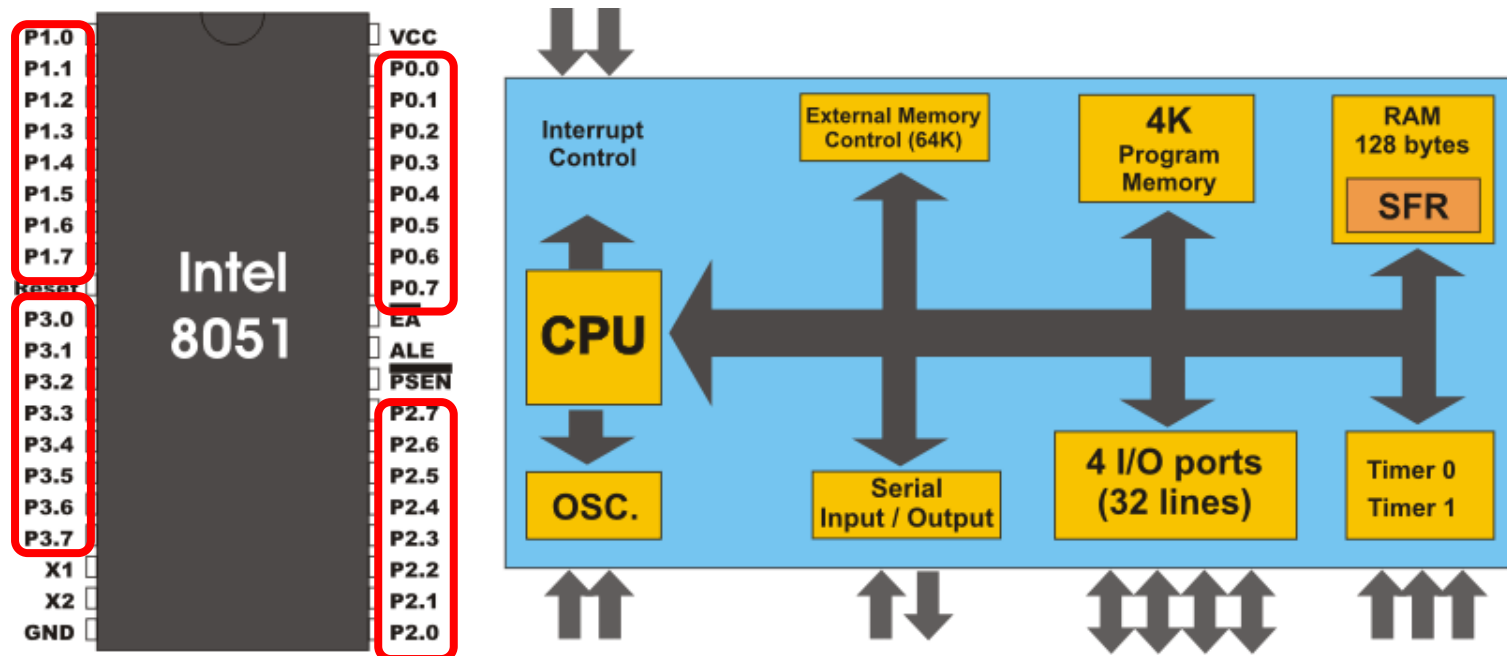
- (1) four 8-bit I/O ports P0-P3
- (2) each pin is bidirectional
  - sometimes input and sometimes output





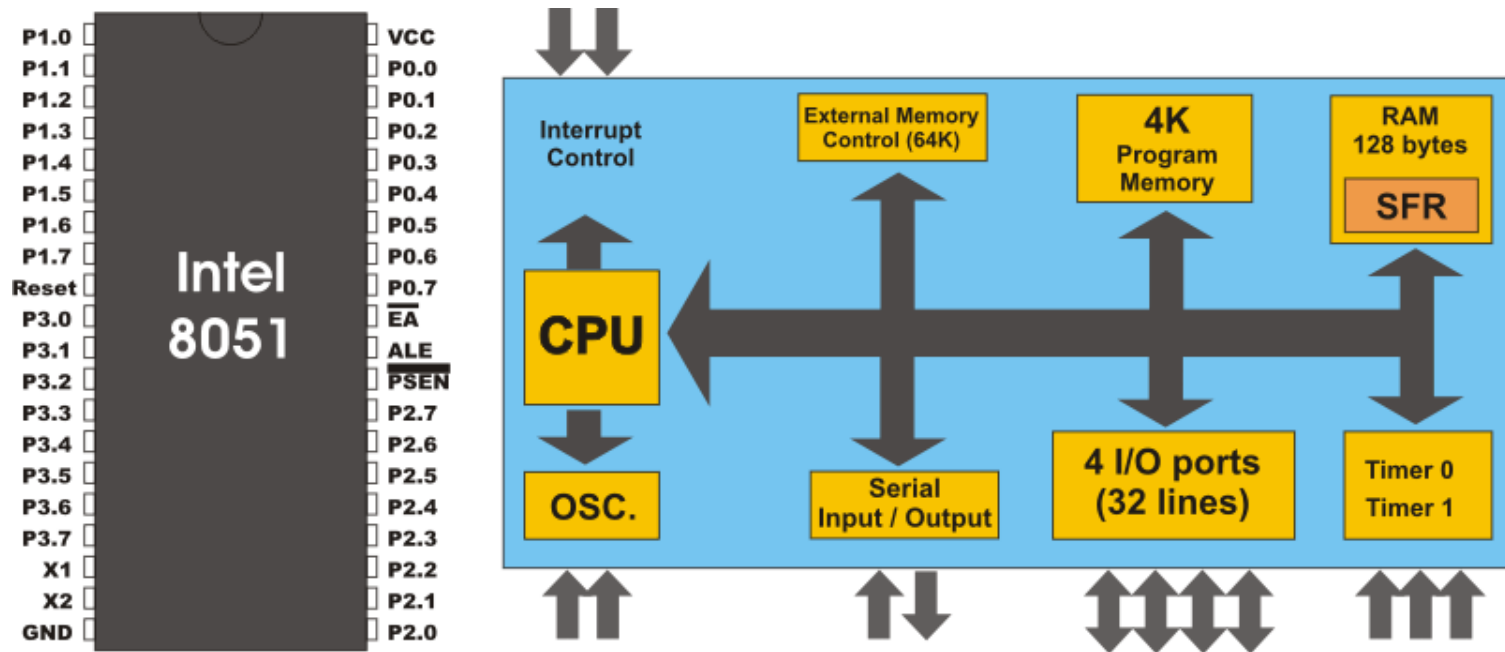
# Features of 8051 I/O

- (1) four 8-bit I/O ports P0-P3
- (2) each pin is bidirectional
  - sometimes input and sometimes output



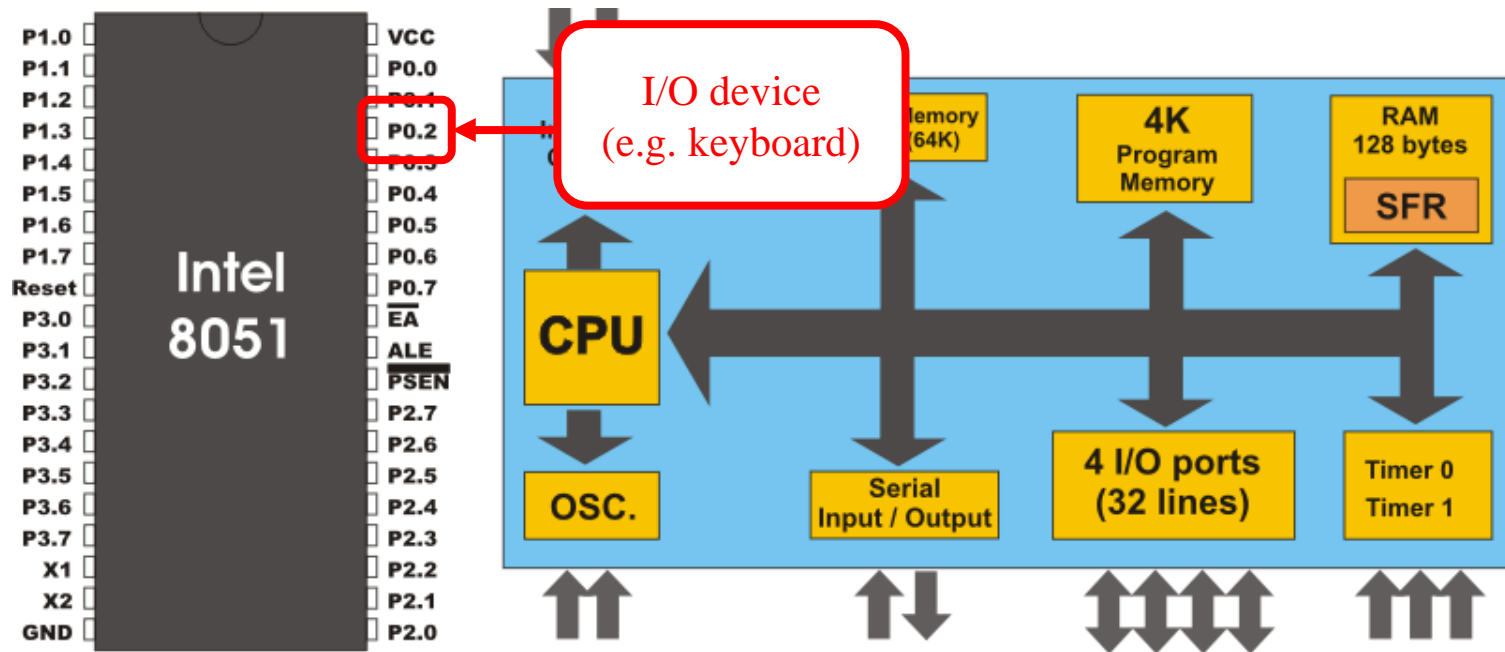
# Features of 8051 I/O

- (1) four 8-bit I/O ports P0-P3
- (2) **each pin is bidirectional**
  - sometimes input and sometimes output



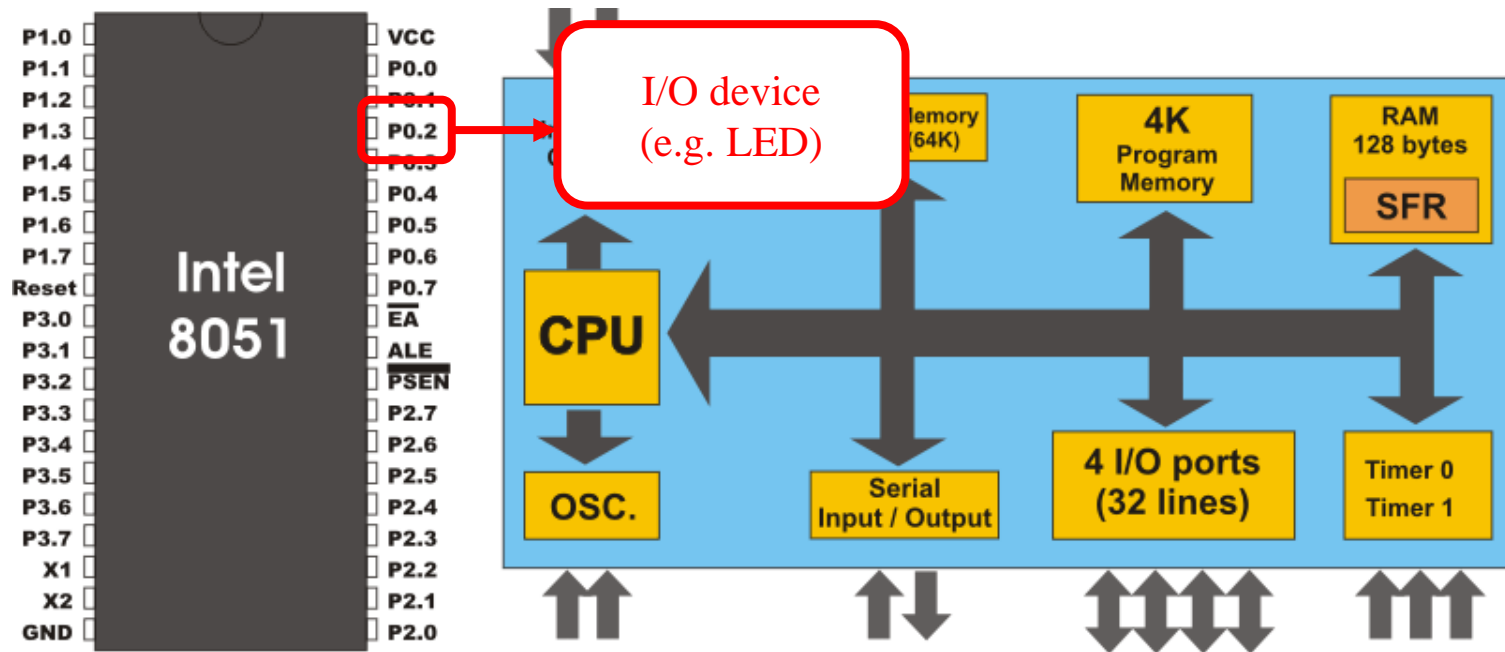
# Features of 8051 I/O

- (1) four 8-bit I/O ports P0-P3
- (2) each pin is bidirectional
  - sometimes input and sometimes output



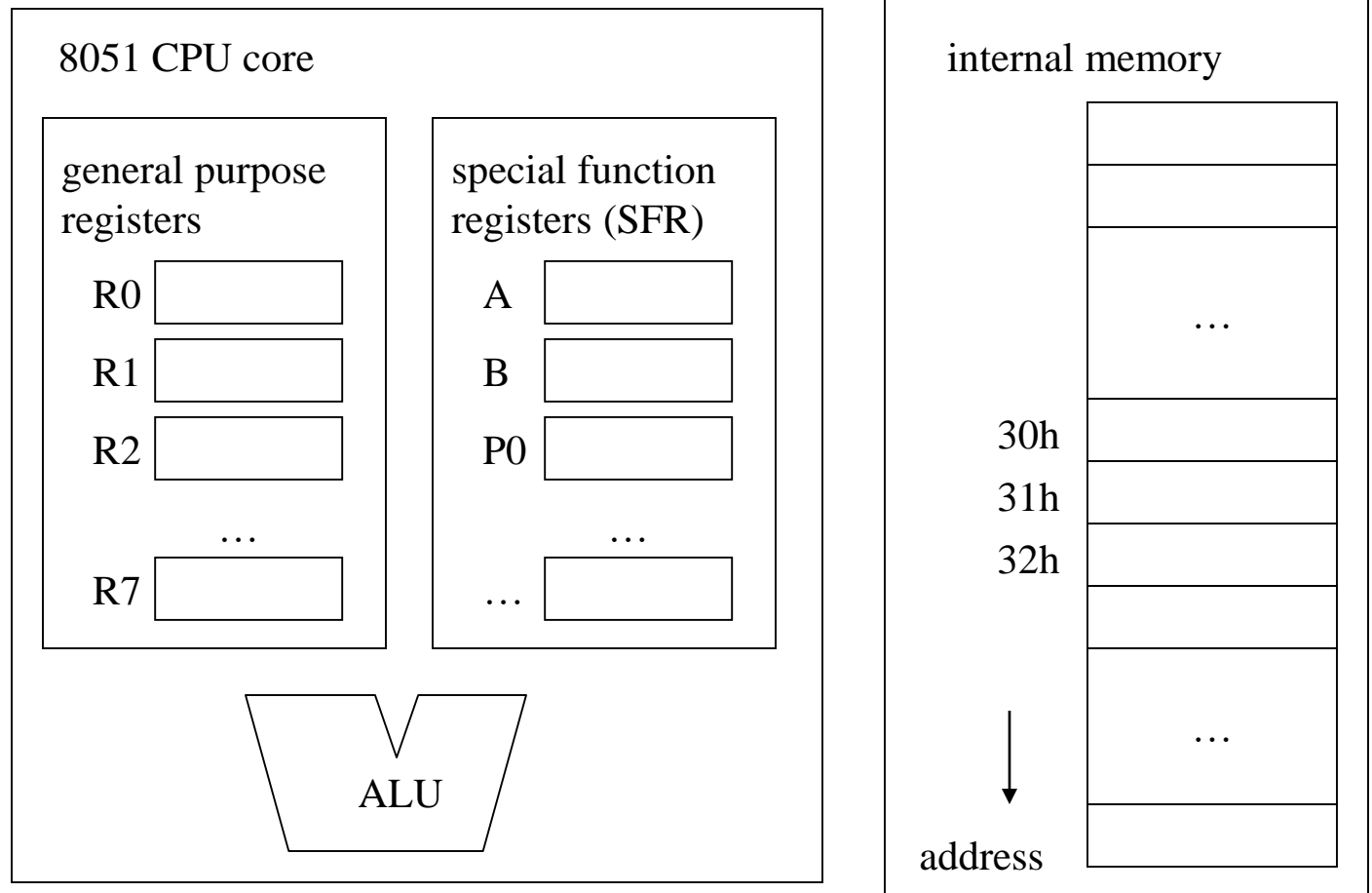
# Features of 8051 I/O

- (1) four 8-bit I/O ports P0-P3
- (2) each pin is bidirectional
  - sometimes input and sometimes output



# Imagination on 8051 architecture

- Imagine how data flow in the architecture!



# How to program I/O ports?

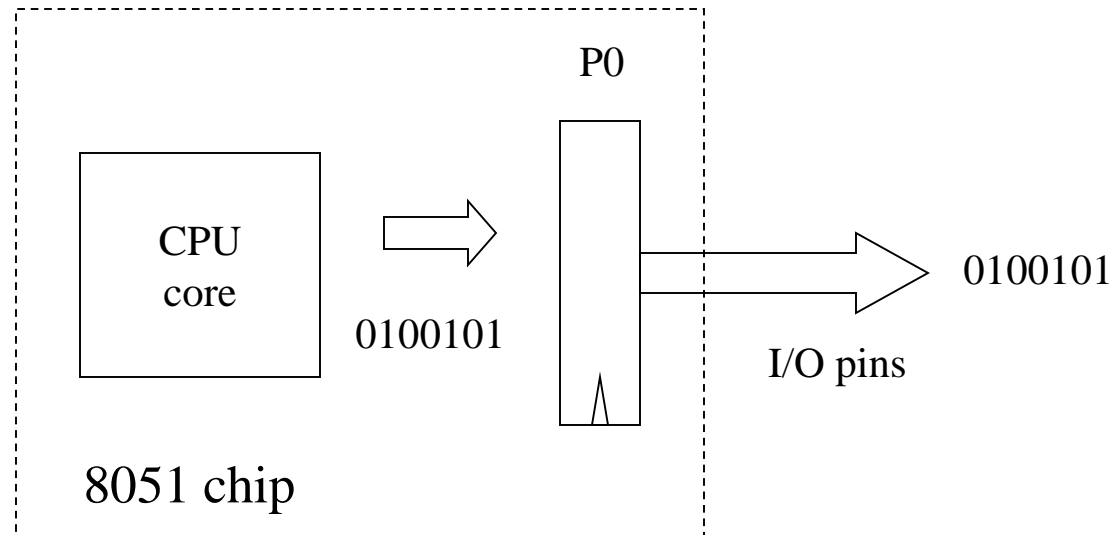
- through SFRs P0-P3

F8									FF
F0	B								F7
E8									EF
E0	ACC								E7
D8									DF
D0	PSW								D7
C8									CF
C0									C7
B8	IP								BF
B0	P3								B7
A8	IE								AF
A0	P2								A7
98	SCON	SBUF							9F
90	P1								97
88	TCON	TMOD	TL0	TL1	TH0	TH1			8F
80	P0	SP	DPL	DPH				PCON	87

Bit-addressable Registers

# How 8051 send out dedicated control signals

```
MOV R0, #01001101B  
MOV P0, R0
```



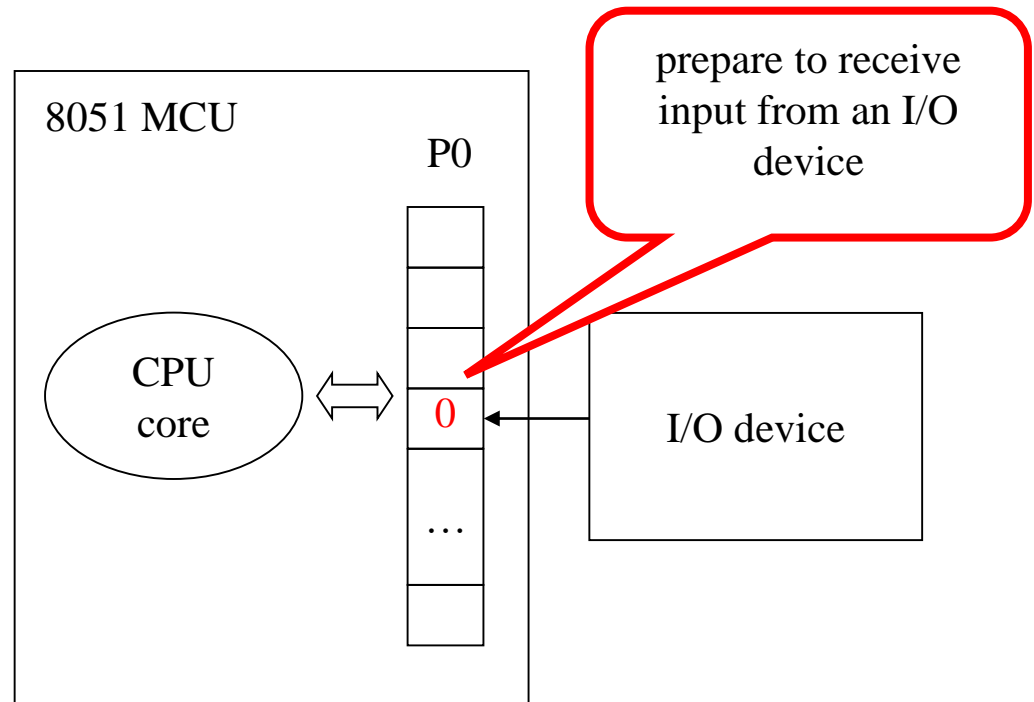
# The case of input (receive)

- initial: set a bit (pin) with value 0
- receive (input): wait for the bit to be toggled to be 1

P0.3 = 0

```
//wait unit P0.3 been set to 1  
while (P0.3==0);
```

```
//action for the I/O event
```





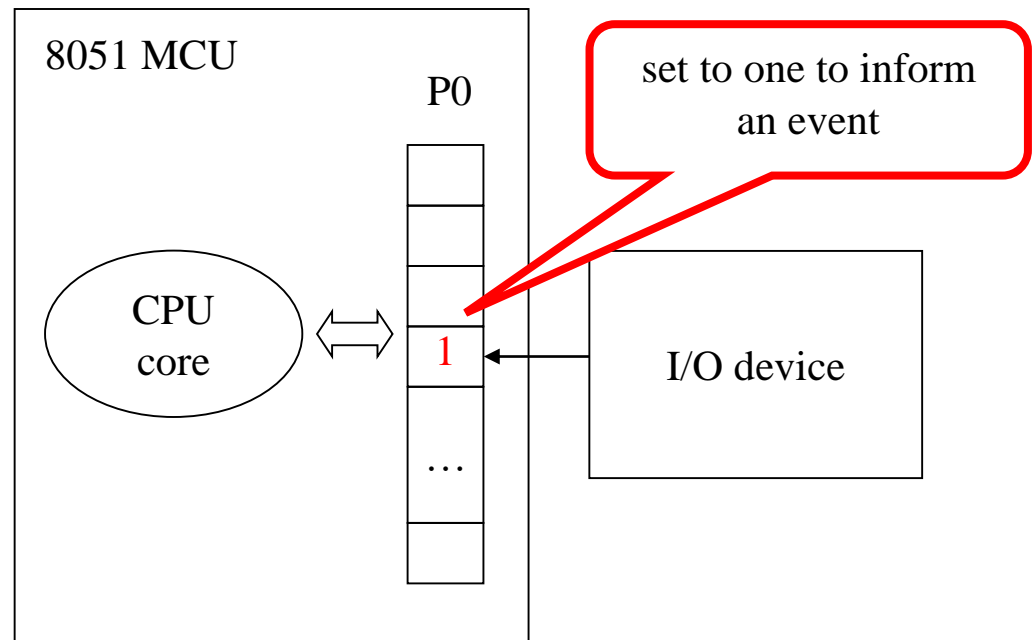
# The case of input (receive)

- initial: set a bit (pin) with value 0
- receive (input): wait for the bit to be toggled to be 1

P0.3 = 0

```
//wait unit P0.3 been set to 1  
while (P0.3==0);
```

```
//action for the I/O event
```





# The GPIO of C8051F040 SoC

---



# Overview of SFR

---

- Extension from legacy 8051
- Divided into 3 pages
- Page 145-149 of the C8051F040 data sheet

# The port configuration

- Set XBR2, PxMDIN and PxMDOUT to set port Px as general purpose I/O

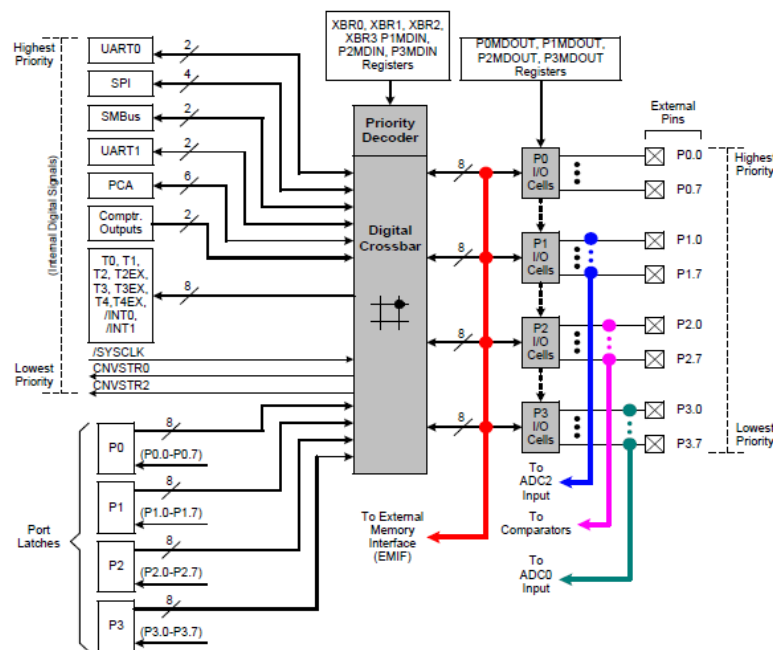


Figure 17.2. Port I/O Functional Block Diagram

# The I/O pad

- To send output 1

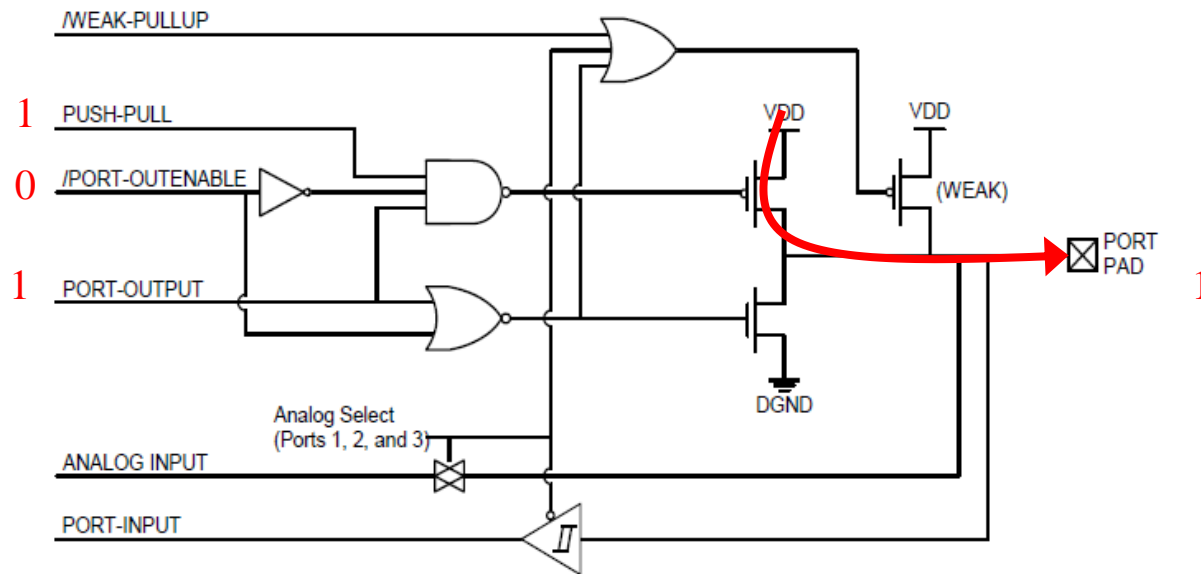
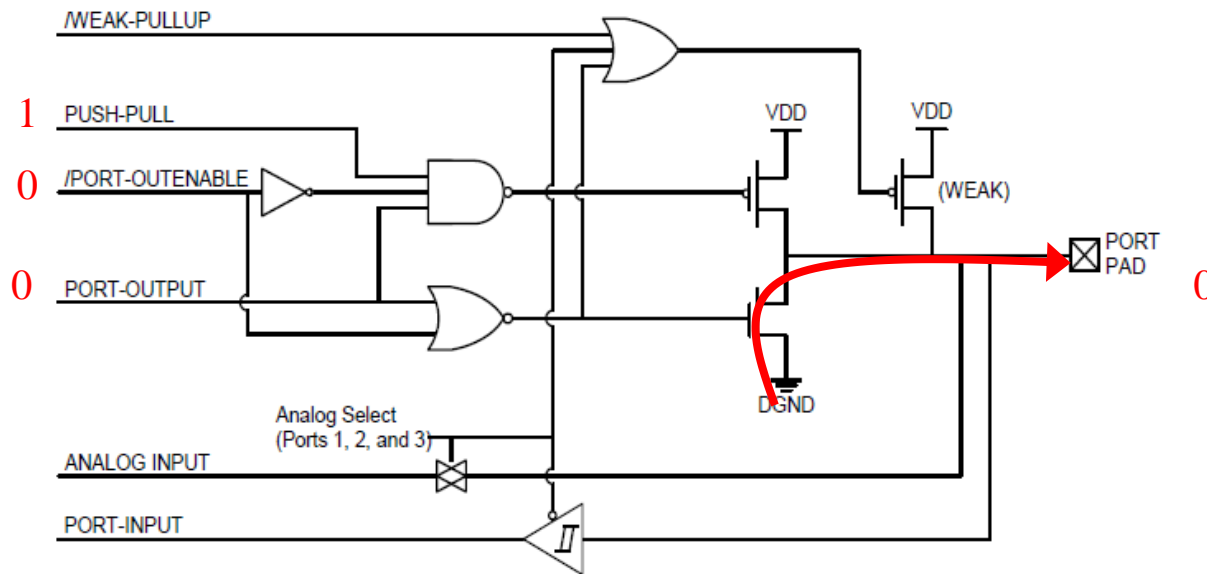


Figure 17.1. Port I/O Cell Block Diagram

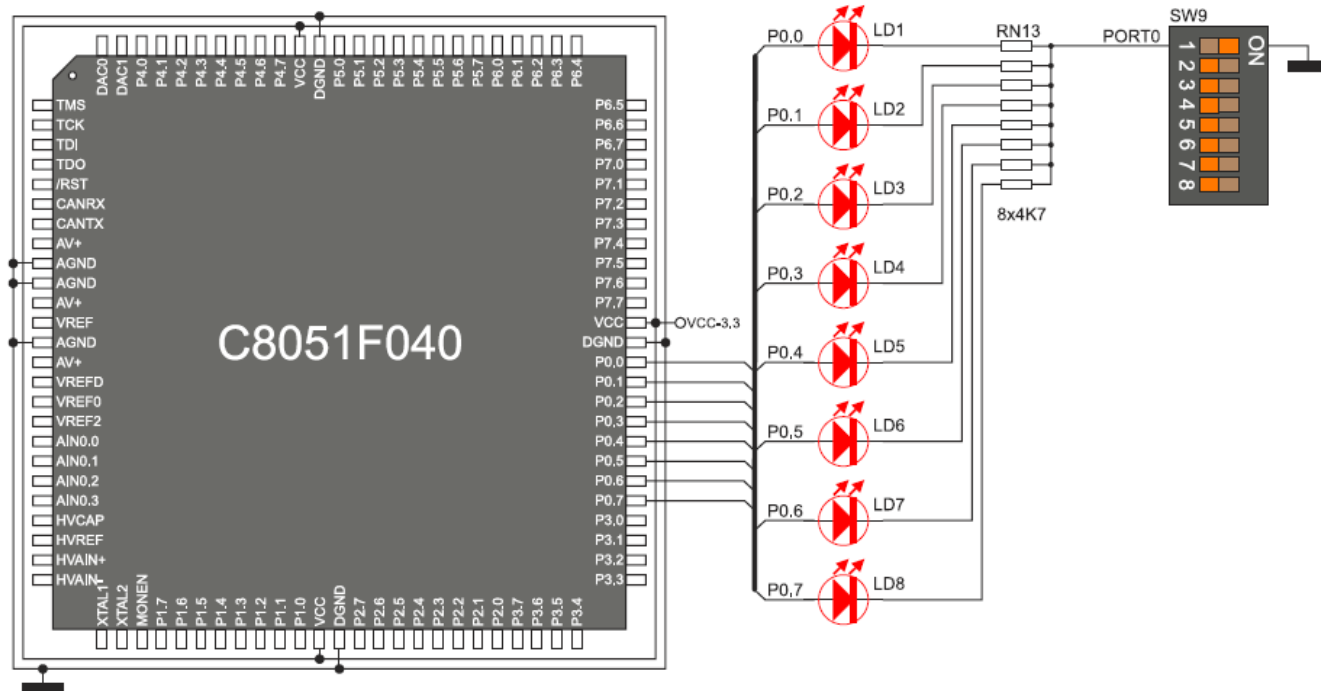
- To send output 0

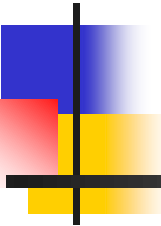


### Figure 17.1. Port I/O Cell Block Diagram

# Schematics of the LED

- `mov P0, #100000000B` to turn on LD8





# Example 1

---

Detect button press and display on  
LED





# Example Code

```
;define control registers (with address)
XBR2      equ      0e3h
P1MDIN    equ      0adh
P2MDOUT    equ      0a6h
WDTCN     equ      0ffh
SFRPAGE    equ      084h
P1         equ      090h
P2         equ      0a0h

;define control words
CONFIG_PAGE equ      0fh
LEGACY_PAGE equ      00h

;turn-off the watch-dog timer
mov        WDTCN, #0deh
mov        WDTCN, #0adh

;setup port configuration
mov        SFRPAGE, #CONFIG_PAGE
mov        XBR2, #0c0h
mov        P1MDIN, #0ffh
mov        P2MDOUT, #0ffh
mov        SFRPAGE, #LEGACY_PAGE

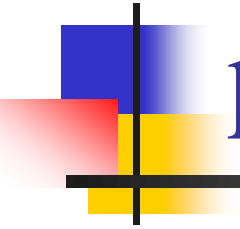
mov        R0, #0

;detect button and display
Loop_Begin:
mov        R0, P1
mov        P2, R0
sjmp      Loop_Begin

end
```

## Example 2: wait for a button pressed

---



# Demo: wait for a button pressed

wait:

```
A = P1;  
if (A==0) goto wait;
```

exit:

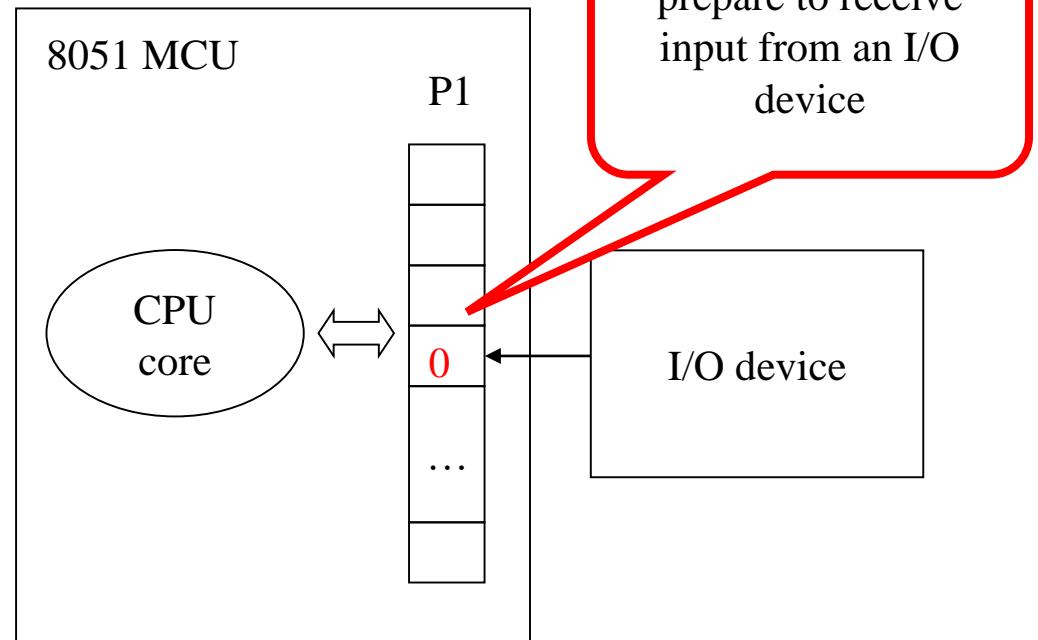
//something after button pressed

wait:

```
mov A, P1  
JZ wait
```

exit:

//something after button pressed



# Demo: wait for a button pressed

wait:

```
A = P1;  
if (A==0) goto wait;
```

exit:

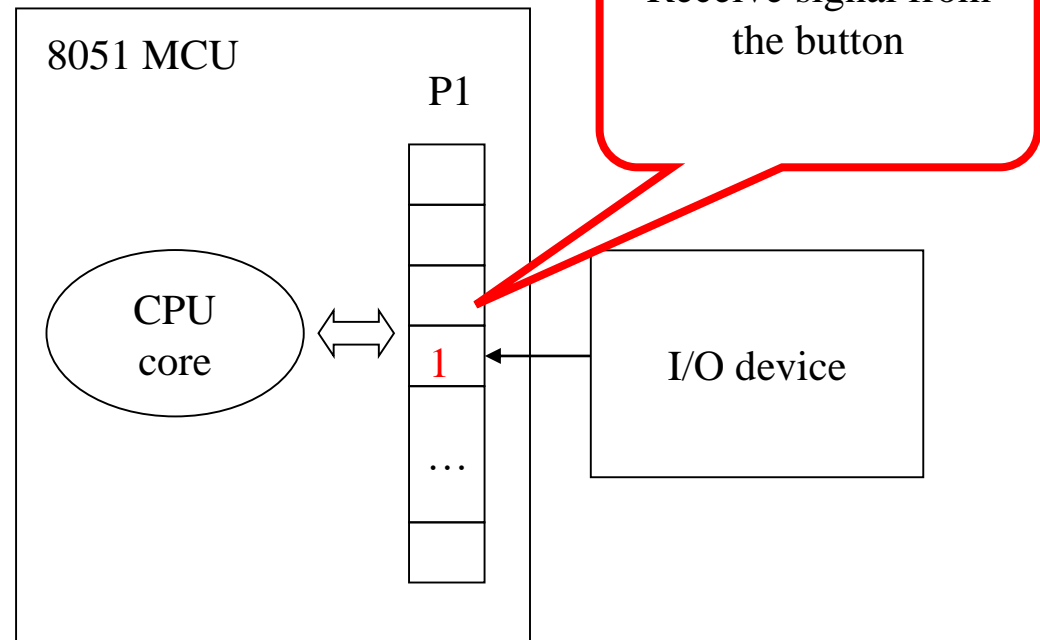
//something after button pressed

wait:

```
mov A, P1  
JZ wait
```

exit:

//something after button pressed





## Example 3: make LED run

---

show how to output signal



# Demo: rotate the LED light

---

MAIN:

```
MOV A, #00000001B
MOV PSW, #00H
```

Loop:

```
MOV    P0, A
LCALL  Delay
RR     A
LJMP   Loop
```

```
                MOV    R0, #50
Delay:          MOV    R1, #40
Delay1:         MOV    R2, #249
Delay2:         DJNZ   R2, Delay2
                DJNZ   R1, Delay1
                DJNZ   R0, Delay
                RET
```



# Demo: rotate the LED light

MAIN:

```
MOV A, #00000001B
MOV PSW, #00H
```

Loop:

```
MOV    P0, A
LCALL  Delay
RR      A
LJMP   Loop
```

control the LED through  
content of A

```
                MOV    R0, #50
Delay:          MOV    R1, #40
Delay1:         MOV    R2, #249
Delay2:         DJNZ   R2, Delay2
                DJNZ   R1, Delay1
                DJNZ   R0, Delay
                RET
```

# Demo: rotate the LED light

MAIN:

```
MOV A, #00000001B
MOV PSW, #00H
```

Loop:

```
MOV    P0, A
LCALL  Delay
RR      A
LJMP   Loop
```

```
                MOV    R0, #50
Delay:          MOV    R1, #40
Delay1:         MOV    R2, #249
Delay2:         DJNZ   R2, Delay2
                DJNZ   R1, Delay1
                DJNZ   R0, Delay
                RET
```

- rotate right (RR) A

00000001



1000000



01000000





# Demo: rotate the LED light

MAIN:

```
MOV A, #00000001B
MOV PSW, #00H
```

Loop:

```
MOV    P0, A
LCALL  Delay
RR     A
LJMP   Loop
```

call a function at label  
“Delay”

```
MOV    R0, #50
```

```
Delay:  MOV    R1, #40
Delay1: MOV    R2, #249
Delay2: DJNZ   R2, Delay2
        DJNZ   R1, Delay1
        DJNZ   R0, Delay
        RET
```

a nested loop to delay some  
time



# Lab02 Study Report

---

- File name: Bxxxxxxx-MCE-Lab2-Study
- File type: PDF only
- The requirements of report
  - Summarize the content of this slide set
  - Provide your plan for this lab exercise
  - No more than one A4 page
  - Grading:  $80 \pm 15$
- Deadline: 2021/10/27 23:00 (不收遲交)
- Upload to e-learning system
- Bonus: 0~10 points
  - Q6: Explain what is SFRPAGE in 8051



# Lab02 Lab Exercise Report

---

- File name: Bxxxxxxx-MCE-Lab2-Result
- File type: PDF only
- The requirements of report
  - Summarize the problems and results you have in this exercise
  - Some screen shots or some code explanation can be provided
  - No more than two A4 pages
  - Grading:  $80 \pm 15$
- Deadline: 2021/11/3 23:00 (不收遲交)
- Upload to e-learning system
- Bonus: 10 points
  - Provide two different LED patterns