

CHƯƠNG 1

TỔNG QUAN VỀ

VI XỬ LÝ

VI ĐIỀU KHIỂN

LỊCH SỬ PHÁT TRIỂN CỦA VI XỬ LÝ

➤ Thời kỳ đầu:

- 1971 **Intel 4004**, CPU 4 bit đầu tiên, $f_{CPU} = 740\text{KHz}$
- 1972 **Intel 8008**, CPU 8 bit đầu tiên, $f_{CPU} = 0.2 - 0.8\text{MHz}$.



Intel 4004



Intel 8008

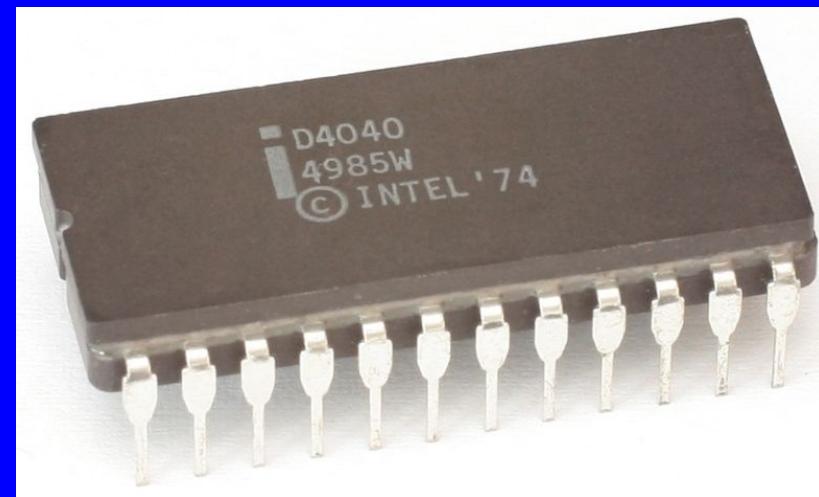
LỊCH SỬ PHÁT TRIỂN CỦA VI XỬ LÝ

➤ Thời kỳ đầu:

- 1971 **Intel 4040**, CPU 4 bit, $f_{CPU} = 500 - 740\text{KHz}$
- 1974 **Intel 8080**, CPU 8 bit, $f_{CPU} = 2\text{MHz}$

(Các sản phẩm cùng thời: Motorola 6800, TI TMS1000, Microchip PIC16, MOS 6502, AMD 2901, Zilog Z80).

Intel 8080



Intel 4040

LỊCH SỬ PHÁT TRIỂN CỦA VI XỬ LÝ

➤ Thời kỳ đầu:

- 1974 **MITS Altair 8800**, máy tính đầu tiên sử dụng vi xử lý **Intel 8080** được lập trình bằng BASIC, ngôn ngữ được phát triển bởi Bill Gates và Paul Allen.

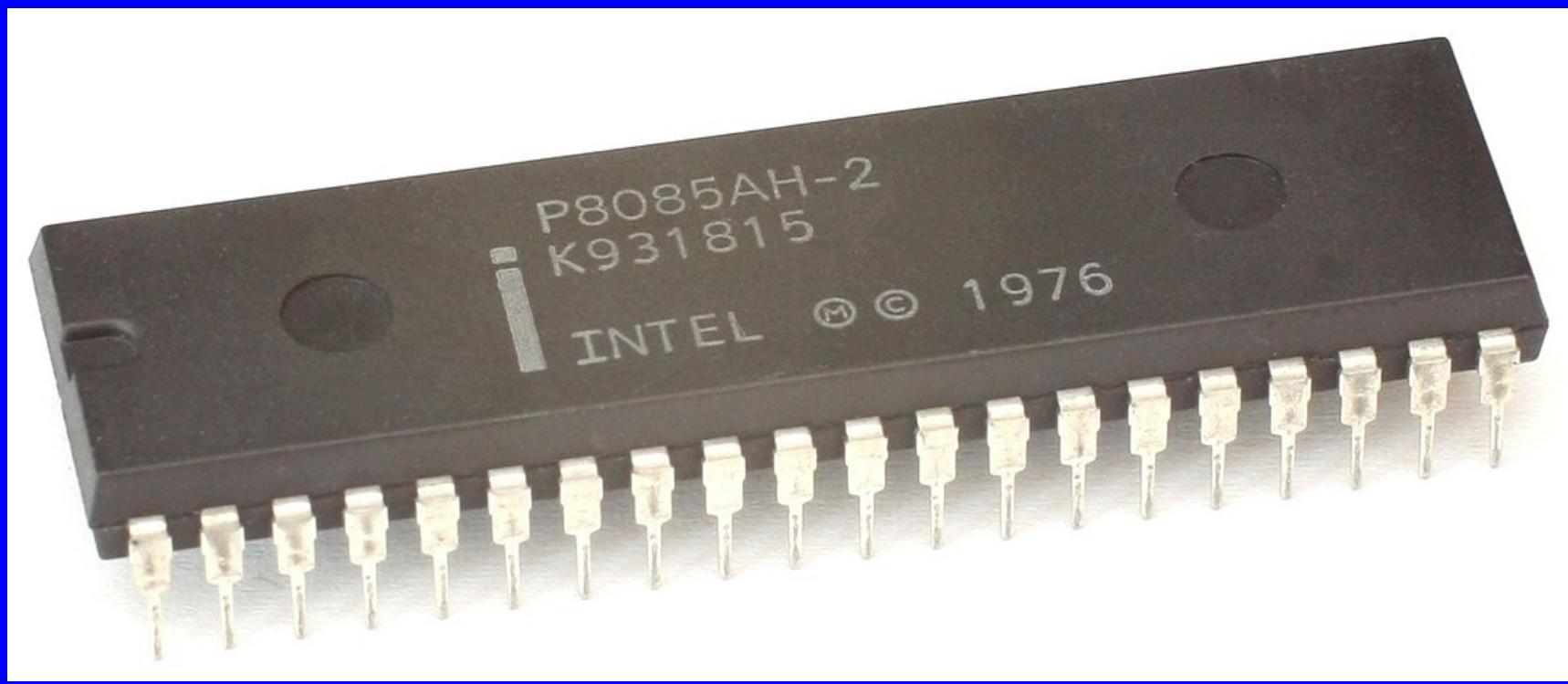


LỊCH SỬ PHÁT TRIỂN CỦA VI XỬ LÝ

➤ Thời kỳ đầu:

- 1976 **Intel 8085**, CPU 8 bit, $f_{CPU} = 3 - 6\text{MHz}$

(Các sản phẩm cùng thời: Fairchild 3850, Signetics 2650, Intel MCS-48, RCA 1802).



LỊCH SỬ PHÁT TRIỂN CỦA VI XỬ LÝ

➤ Thời kỳ đầu:

- 1977 **Apple II**, máy tính gia đình phổ cập đầu tiên, sử dụng vi xử lý **MOS Technology 6502**.



OS: Integer BASIC

CPU: MOS Technology 6502

Memory: 4KB – 64KB

Storage: 5-1/4", 140KB (Cassette)

Display: NTSC video out

Graphics: Lo-res (40×48, 16-color)

Hi-res (280×192, 6 color)

Sound: 1-bit speaker

1-bit microphone jack

1-bit headphone jack

Input: Upper-case keyboard, 52 keys

Controller input: Paddles

Connectivity: Parallel, Serial, SCSI

Predecessor: Apple I

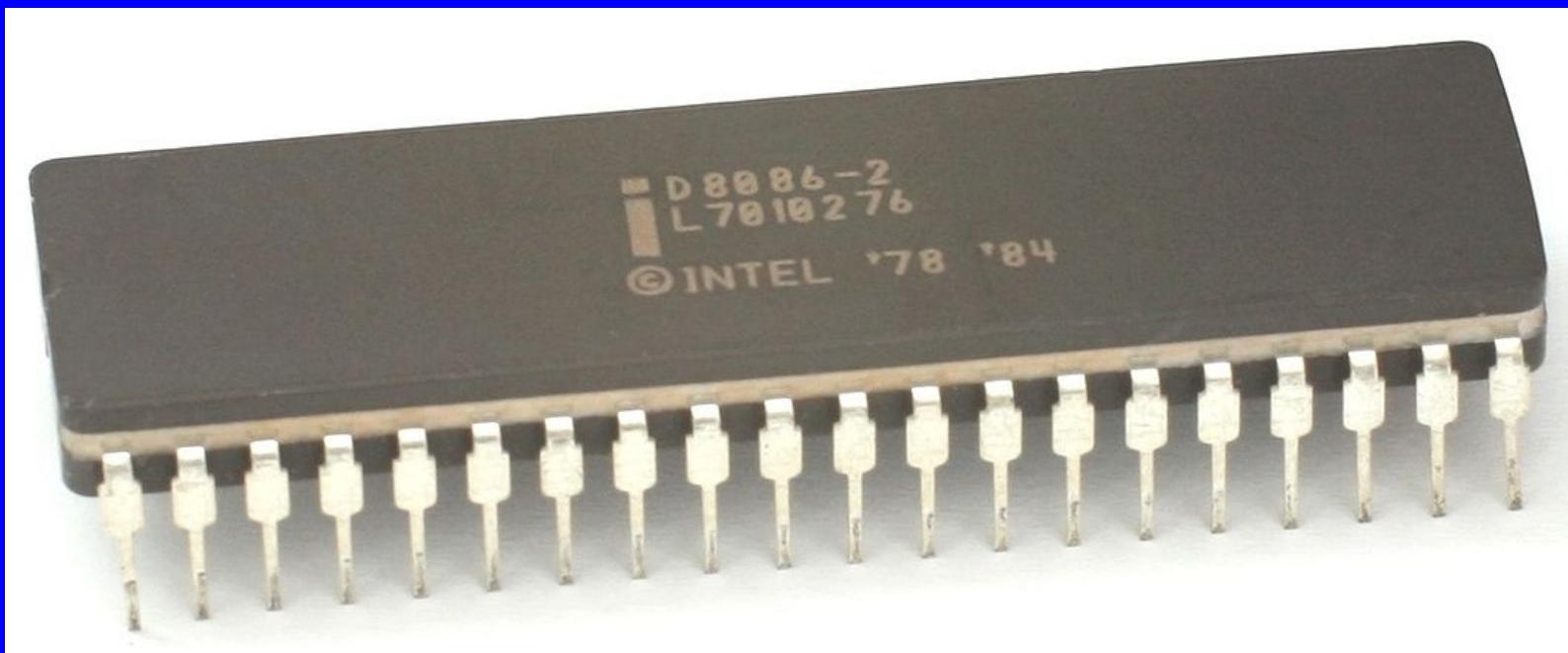
Successor: Apple II Plus

LỊCH SỬ PHÁT TRIỂN CỦA VI XỬ LÝ

➤ Thời kỳ đầu:

- 1978 Intel 8086, CPU 16 bit, $f_{CPU} = 5 - 10\text{MHz}$

(Các sản phẩm cùng thời: Motorola 68000, Intel MCS-51).



LỊCH SỬ PHÁT TRIỂN CỦA VI XỬ LÝ

➤ Thập niên 1980:

- 1981 **IBM PC 5150**, máy tính cá nhân (**Personal Computer**) của hãng IBM sử dụng vi xử lý **Intel 8088**.



OS: IBM BASIC / PC DOS 1.0

CPU: Intel 8088 4.77MHz

Memory: 16KB – 256KB

Storage: 5-1/4“, 720KB (Floppy)

Display: NTSC video out

Graphics: 720x350, Monochrome

Sound: 1-channel PWM

Input: IBM PC keyboard, 84 keys

Controller input: Joystick

Connectivity: Parallel, Serial, SCSI

Predecessor: IBM Datamaster

Successor: IBM PC/XT, IBM PC/AT

LỊCH SỬ PHÁT TRIỂN CỦA VI XỬ LÝ

➤ Thập niên 1980:

- 1982 **Intel 80186**, CPU 16 bit, $f_{CPU} = 6 - 25\text{MHz}$
- 1982 **Intel 80286**, CPU 16 bit, $f_{CPU} = 6 - 25\text{MHz}$.

Intel 80186



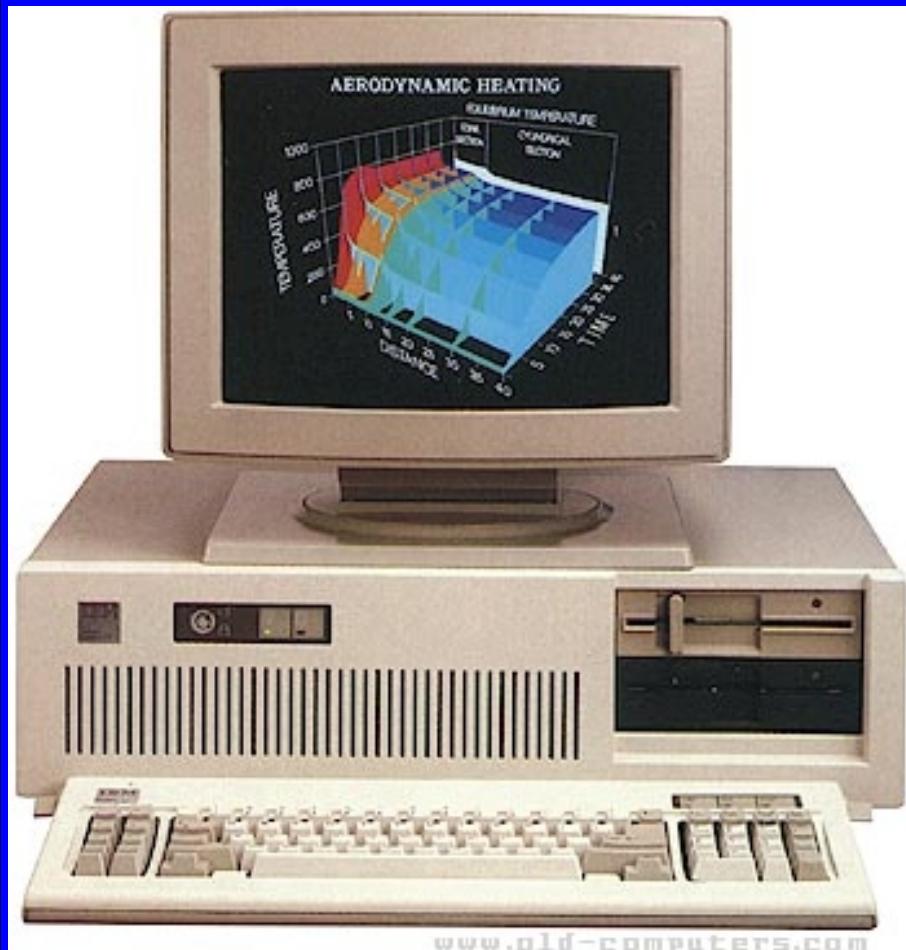
Intel 80286



LỊCH SỬ PHÁT TRIỂN CỦA VI XỬ LÝ

➤ Thập niên 1980:

- 1981 **IBM PC/AT 286**, máy tính cá nhân của hãng IBM sử dụng vi xử lý **Intel 80286**.



OS: PC DOS 3.0

CPU: Intel 80286 6MHz

Memory: 256KB – 16MB

Storage: 5-1/4“, 1.2MB (Floppy)
20MB and more (HDD)

Display: NTSC video out

Graphics: EGA 640x350, Color Screen

Sound: Beeper

Input: IBM PC keyboard, 84 keys

Controller input: Joystick

Connectivity: Parallel, Serial, ISA

Predecessor: IBM PC/XT

Successor: IBM Personal System/2

LỊCH SỬ PHÁT TRIỂN CỦA VI XỬ LÝ

➤ Thập niên 1980:

- 1985 Intel 80386, CPU 32 bit, $f_{CPU} = 12 - 40\text{MHz}$
- 1989 Intel 80486, CPU 32 bit, $f_{CPU} = 16 - 50\text{MHz}$

Intel 80386



Intel 80486

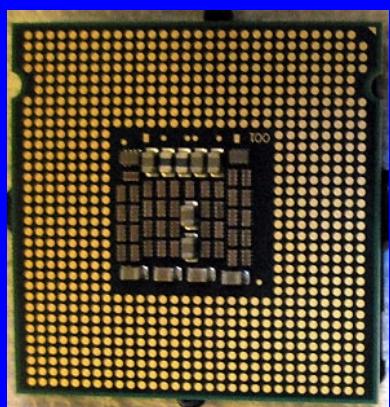


LỊCH SỬ PHÁT TRIỂN CỦA VI XỬ LÝ



➤ Thập niên 1990 – nay:

- 1993 **Intel Pentium**, CPU 32 bit, $f_{CPU} = 60 - 166\text{MHz}$
- 1995 **Intel Pentium Pro**, CPU 32 bit, $f_{CPU} = 150 - 200\text{MHz}$
- 1997 **Intel Pentium II**, CPU 32 bit, $f_{CPU} = 233 - 450\text{MHz}$
- 1999 **Intel Pentium III**, CPU 32 bit, $f_{CPU} = 450\text{MHz} - 1.4\text{GHz}$
- 2000 **Intel Pentium 4**, CPU 32/64 bit, $f_{CPU} = 1.3 - 3.8\text{GHz}$
- 2005 **Intel Pentium D**, CPU 64 bit, $f_{CPU} = 2.6 - 3.73\text{GHz}$
- 2006 **Intel Core 2** CPU đa nhân 64 bit, $f_{CPU} = 1.6 - 3.33\text{GHz}$
- 2010 – nay **Intel Core i3, i5, i7, i9**, CPU đa nhân 64 bit.



LỊCH SỬ PHÁT TRIỂN CỦA VI XỬ LÝ

➤ Hình ảnh các máy vi tính ngày nay:



LỊCH SỬ PHÁT TRIỂN CỦA VI XỬ LÝ

➤ Hình ảnh các máy vi tính ngày nay:

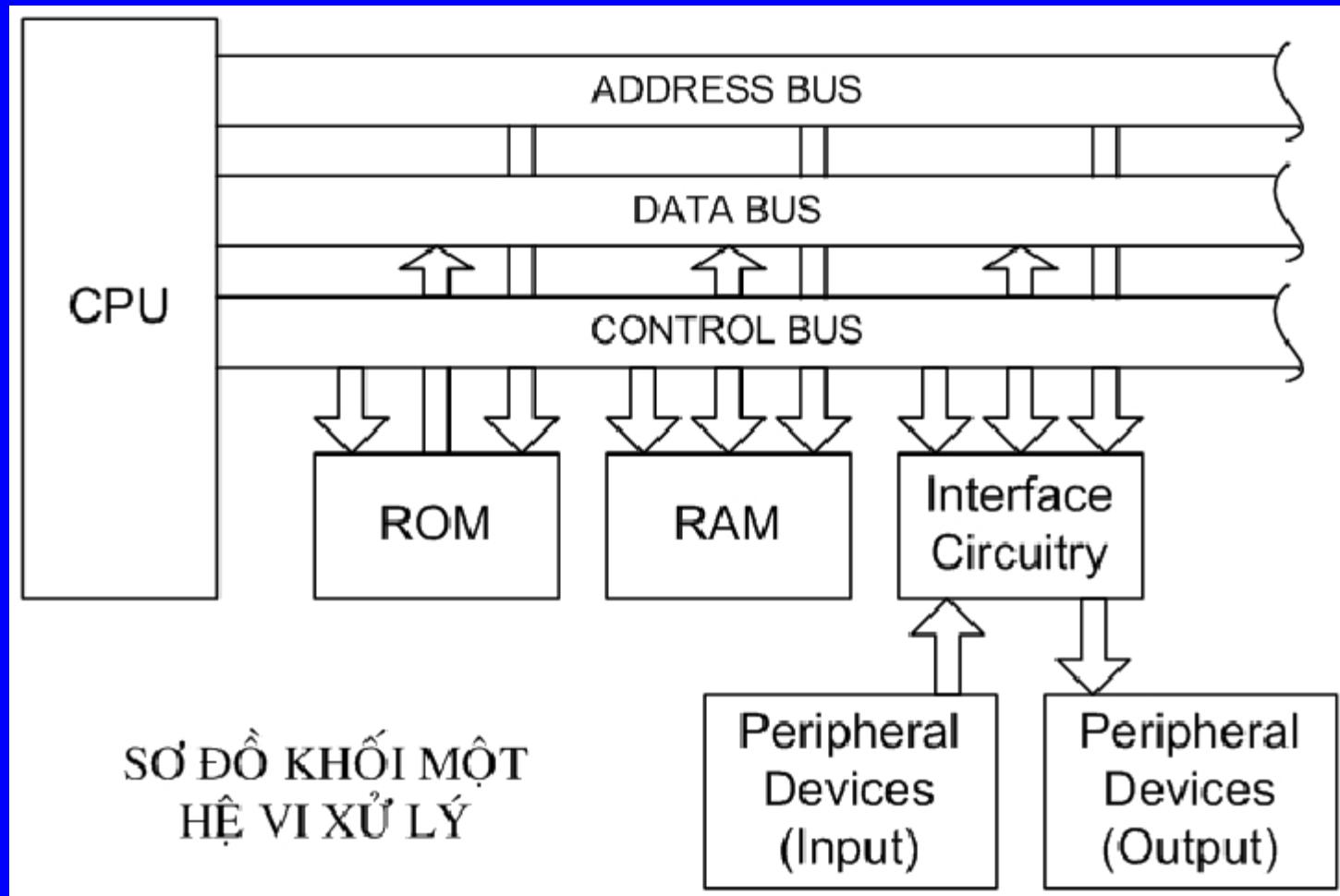


LỊCH SỬ PHÁT TRIỂN CỦA VI XỬ LÝ

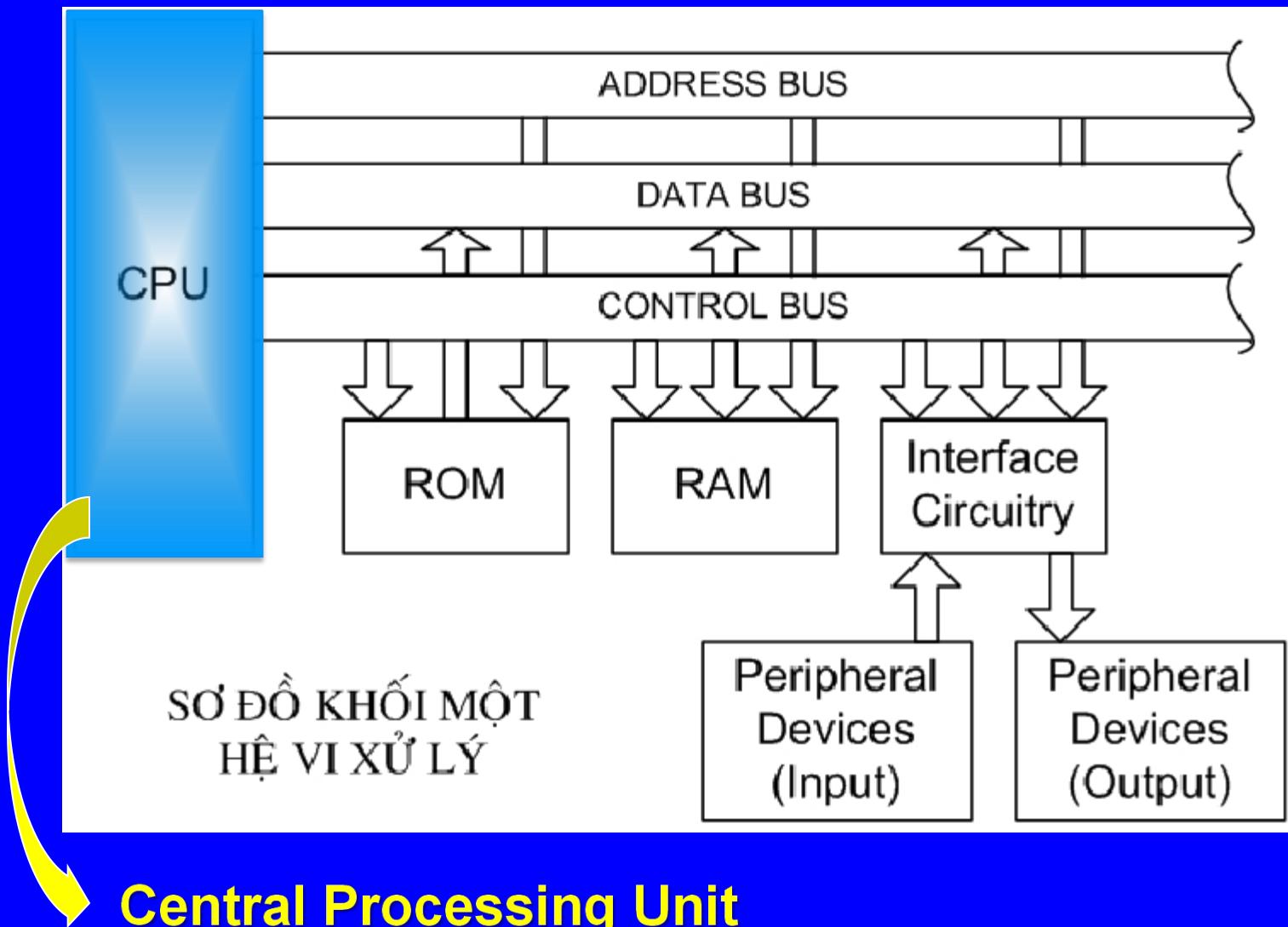
➤ Hình ảnh các máy vi tính ngày nay:



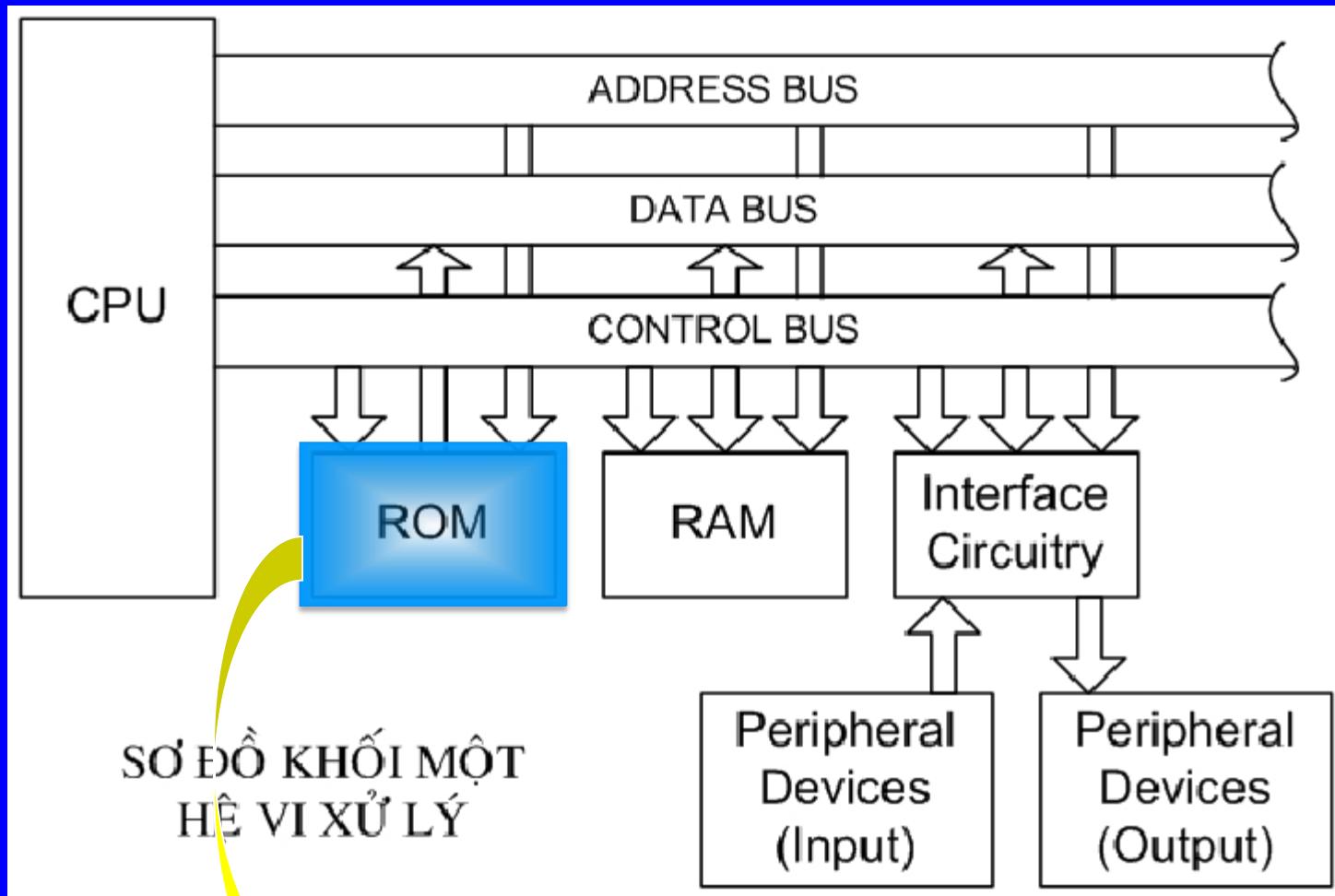
SƠ ĐỒ KHỐI CỦA HỆ VI XỬ LÝ



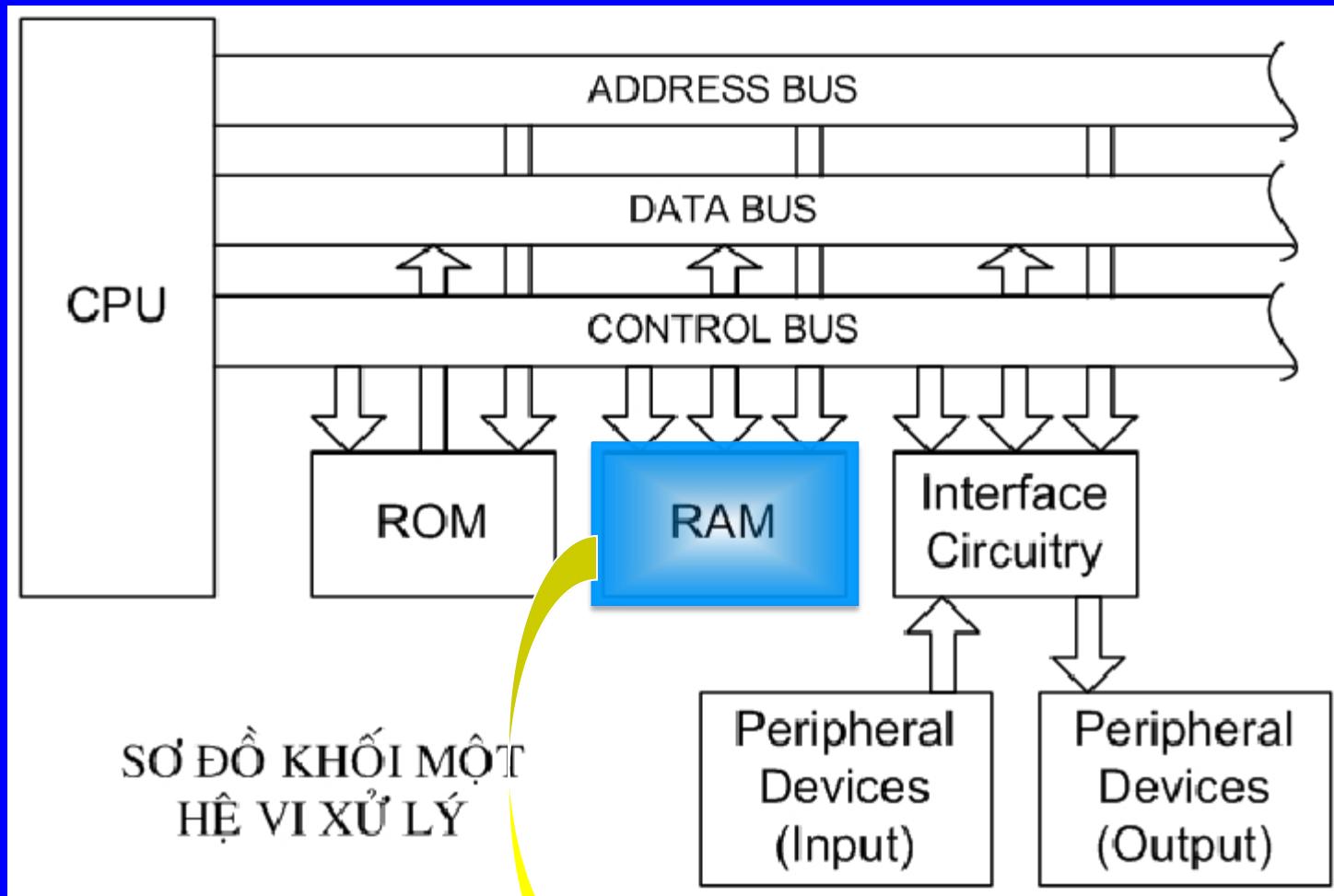
SƠ ĐỒ KHỐI CỦA HỆ VI XỬ LÝ



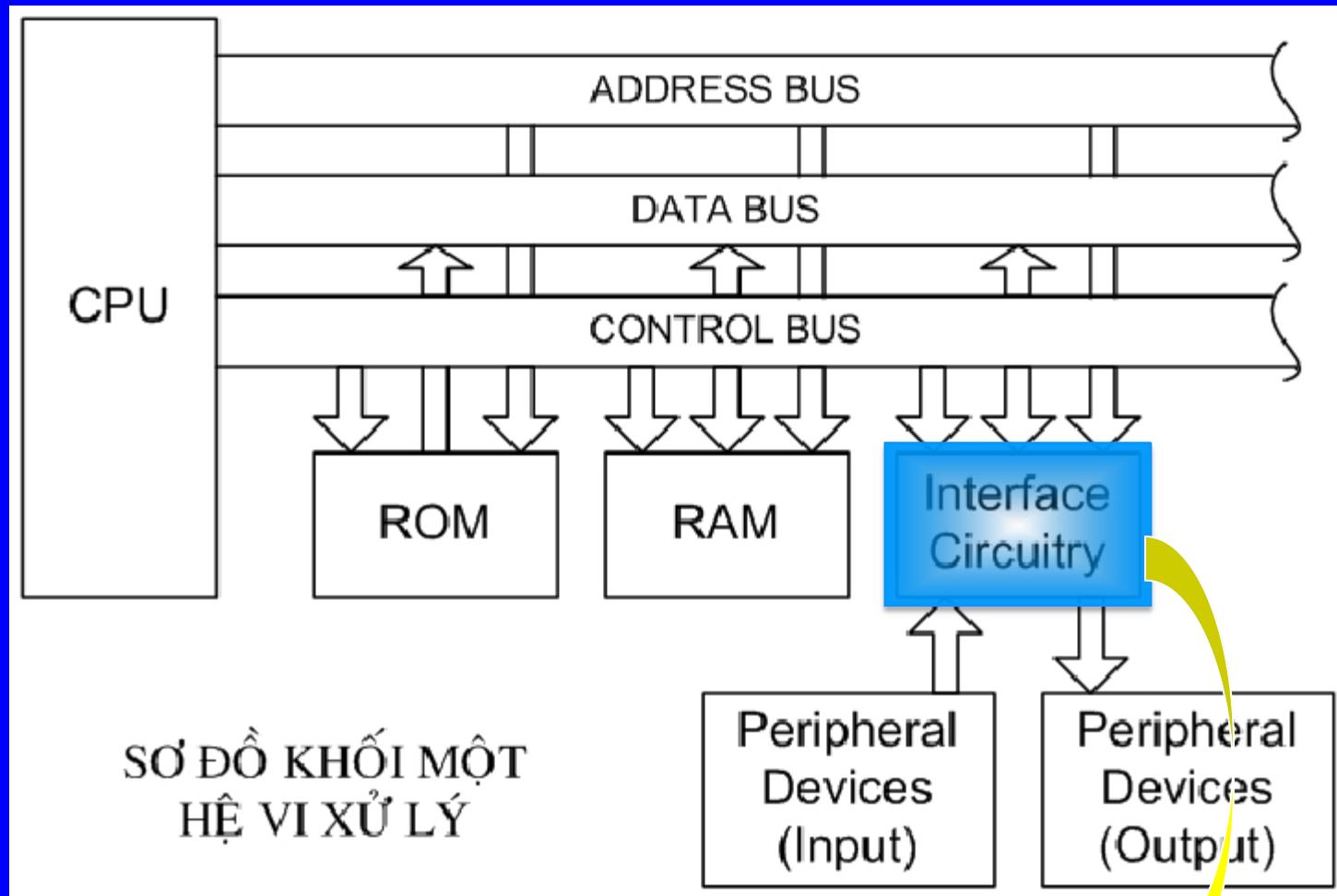
SƠ ĐỒ KHỐI CỦA HỆ VI XỬ LÝ



SƠ ĐỒ KHỐI CỦA HỆ VI XỬ LÝ

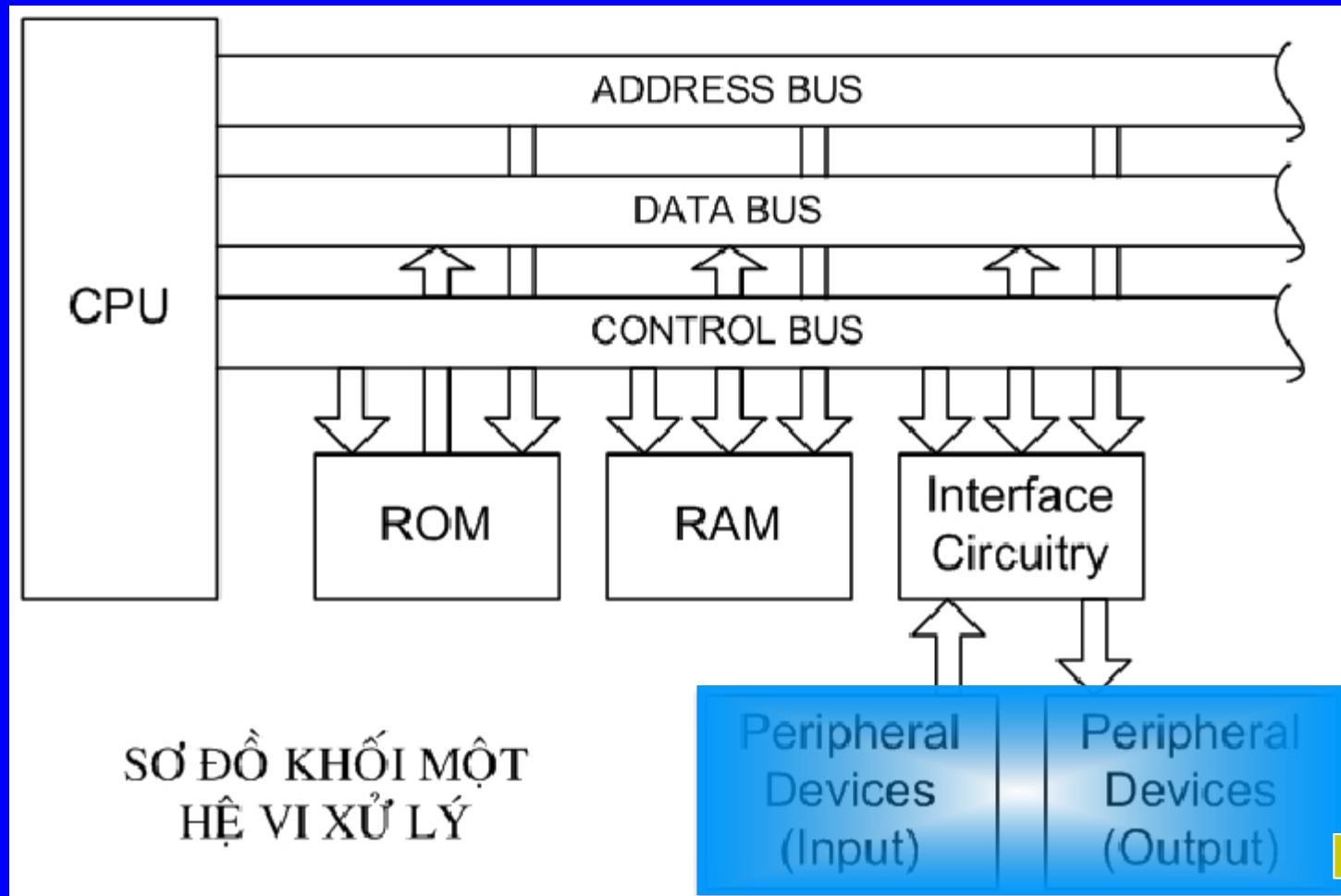


SƠ ĐỒ KHỐI CỦA HỆ VI XỬ LÝ



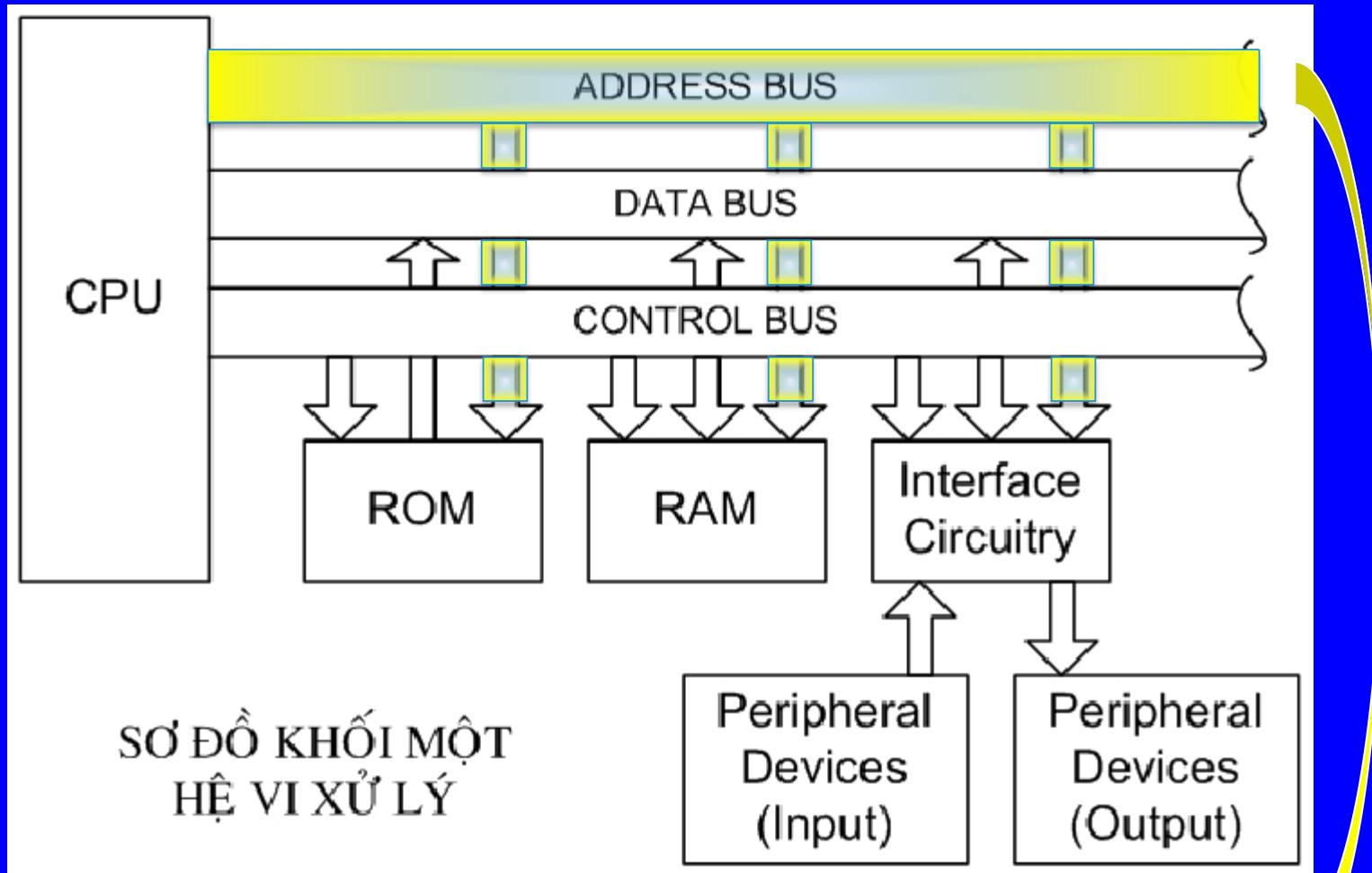
Mạch điện giao tiếp

SƠ ĐỒ KHỐI CỦA HỆ VI XỬ LÝ



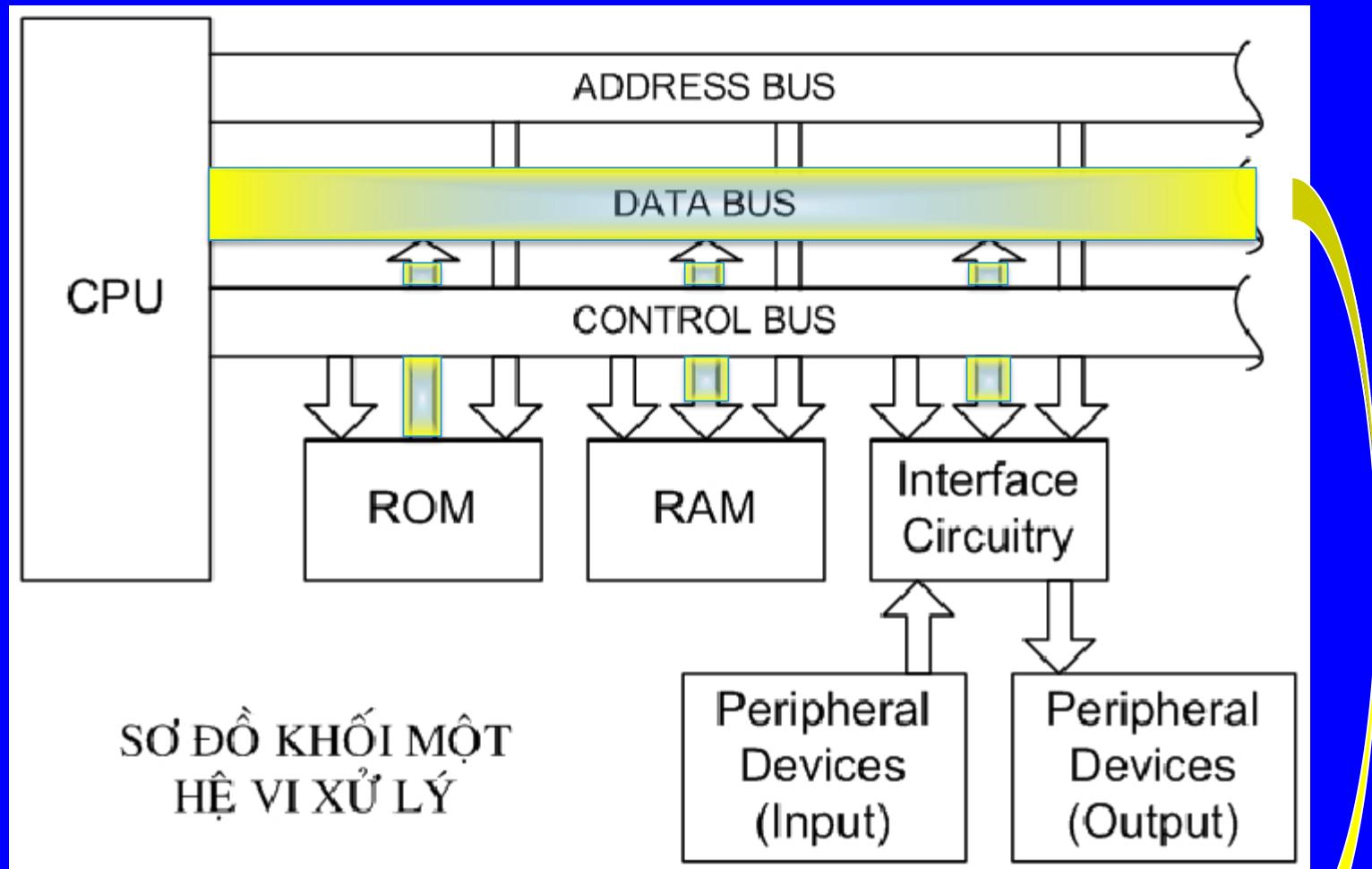
Thiết bị ngoại vi (xuất/nhập)

SƠ ĐỒ KHỐI CỦA HỆ VI XỬ LÝ

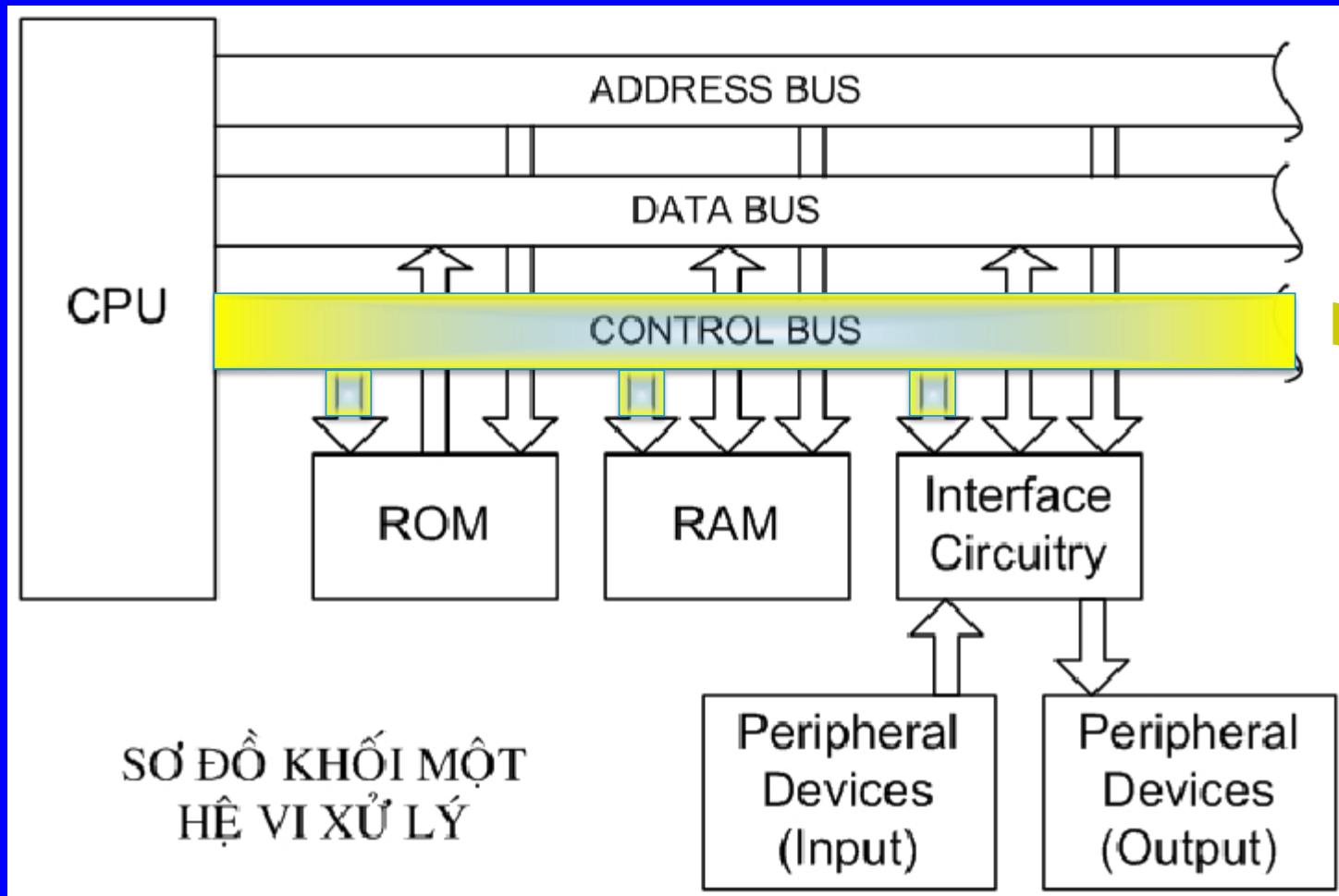


Bus địa chỉ

SƠ ĐỒ KHỐI CỦA HỆ VI XỬ LÝ

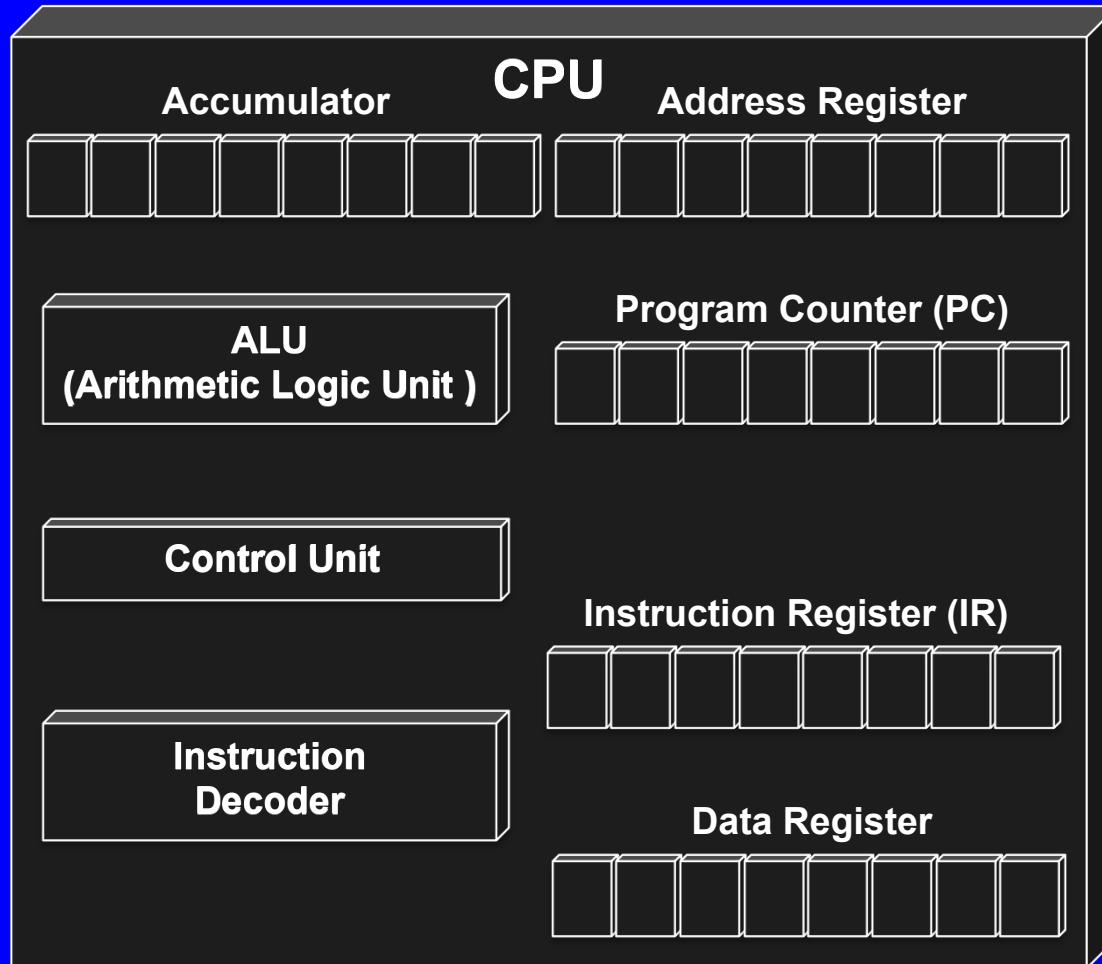


SƠ ĐỒ KHỐI CỦA HỆ VI XỬ LÝ



Bus điều khiển

SƠ ĐỒ KHỐI ĐƠN GIẢN CỦA CPU



QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

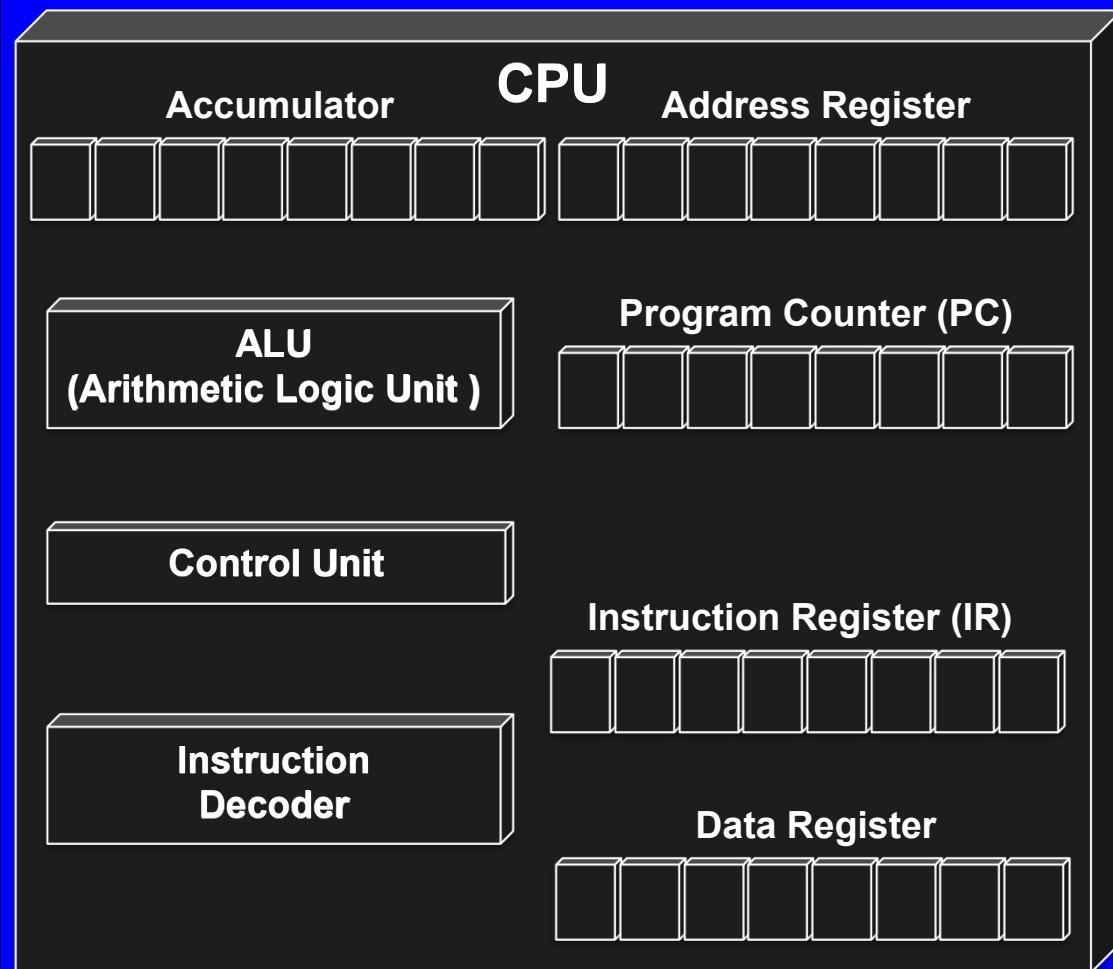
- ✓ Để hiểu CPU hoạt động như thế nào, ta phân tích ví dụ dưới đây của một chương trình.

✓ Ví dụ:

LDA 7

ADD 10

HLT



✓ Lệnh LDA 7:

1000 0110

0000 0111

Mã lệnh LDA

Toán hạng (7)

✓ Lệnh ADD 10:

1000 1011

0000 1010

Mã lệnh ADD

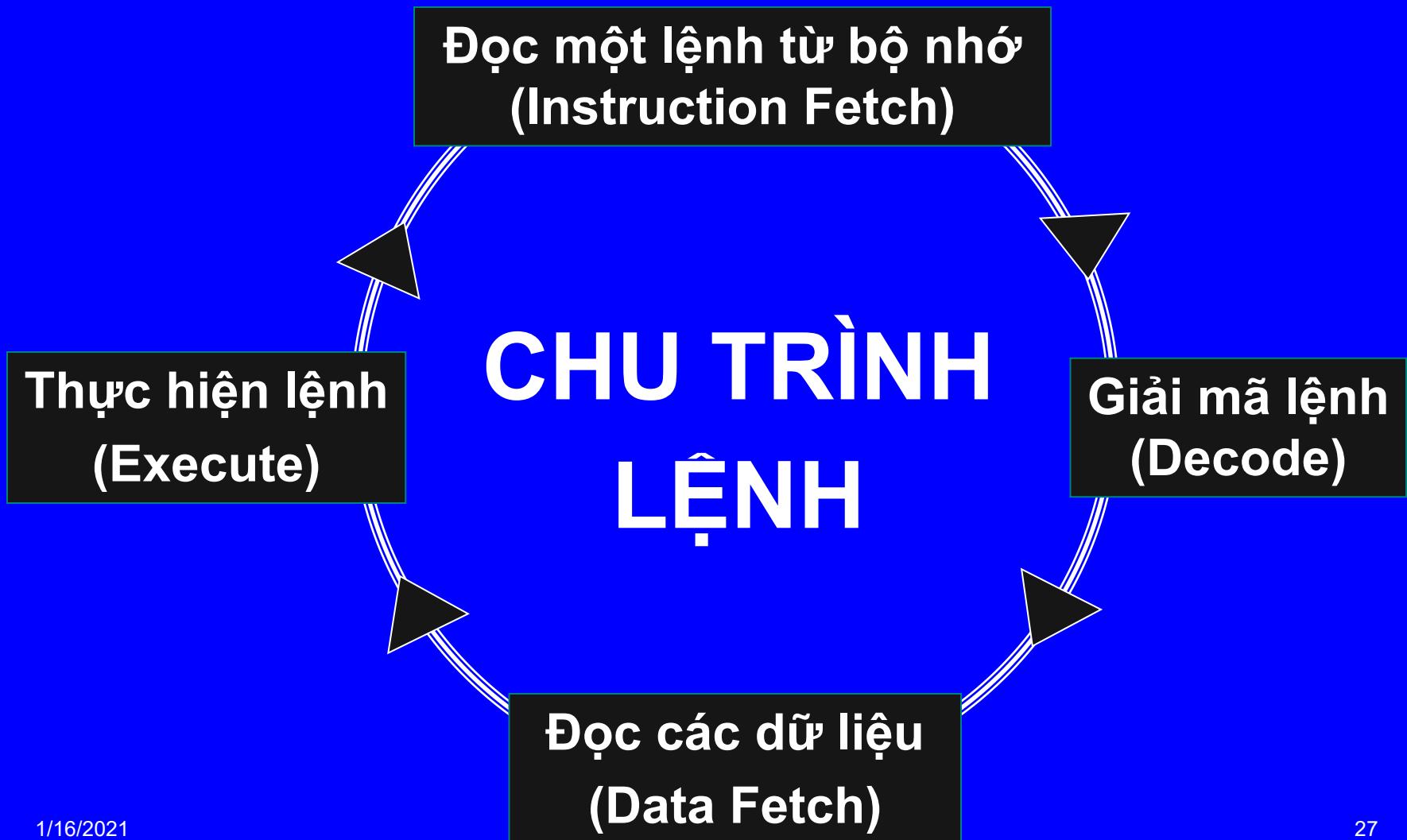
Toán hạng (10)

✓ Lệnh HLT:

0011 1110

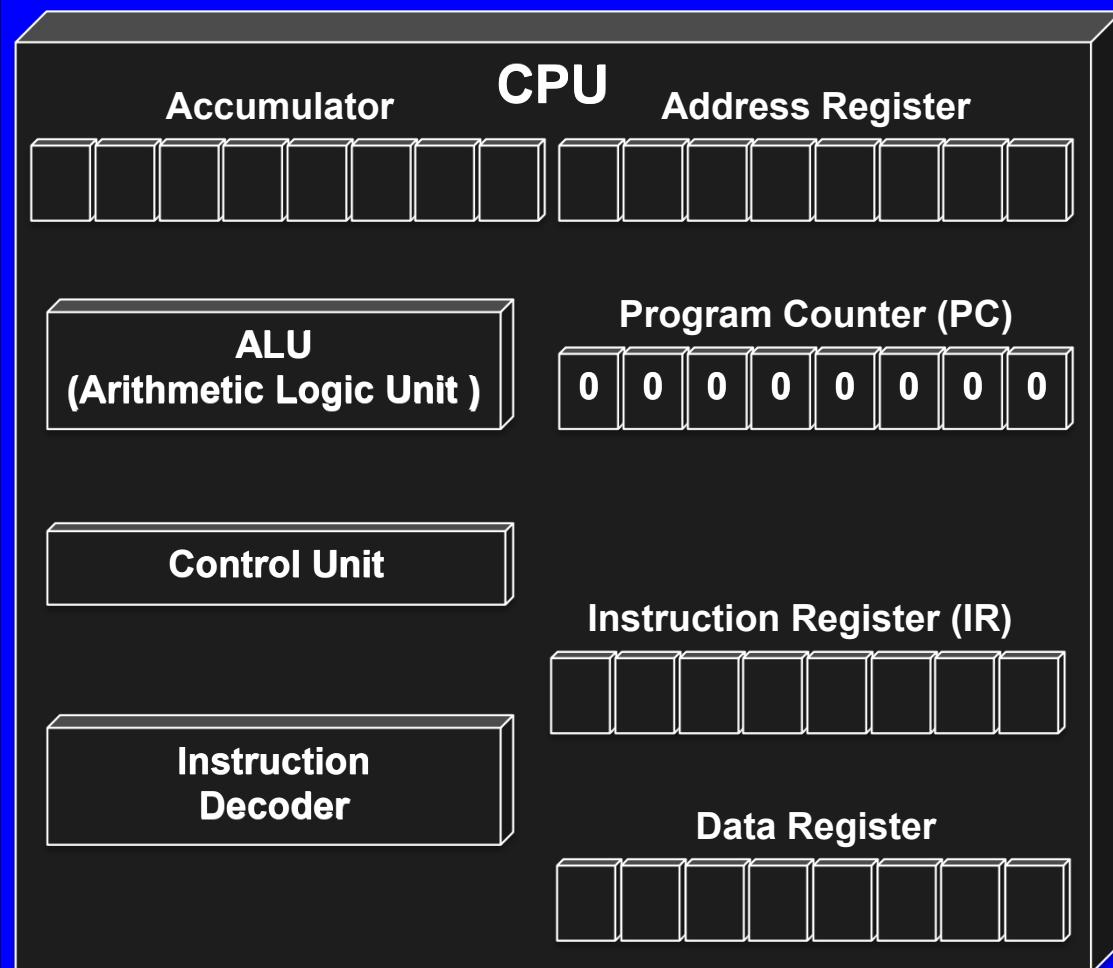
Mã lệnh HLT

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU



QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

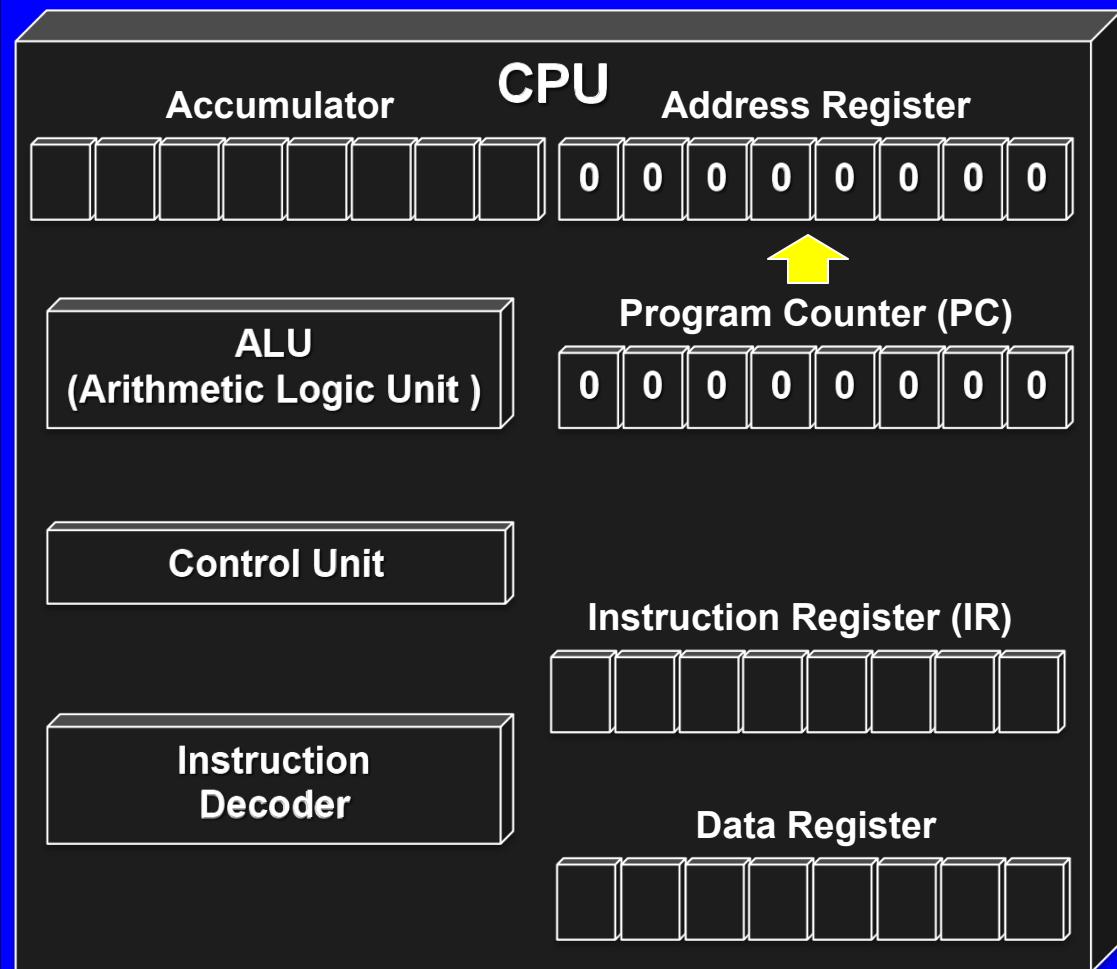
- ✓ Trước khi một chương trình có thể hoạt động, chương trình phải được đặt trong bộ nhớ
- ✓ Bắt đầu thực thi chương trình, PC phải được đặt địa chỉ của lệnh đầu tiên.



Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	28

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của PC được chuyển sang thanh ghi địa chỉ (Address Register).

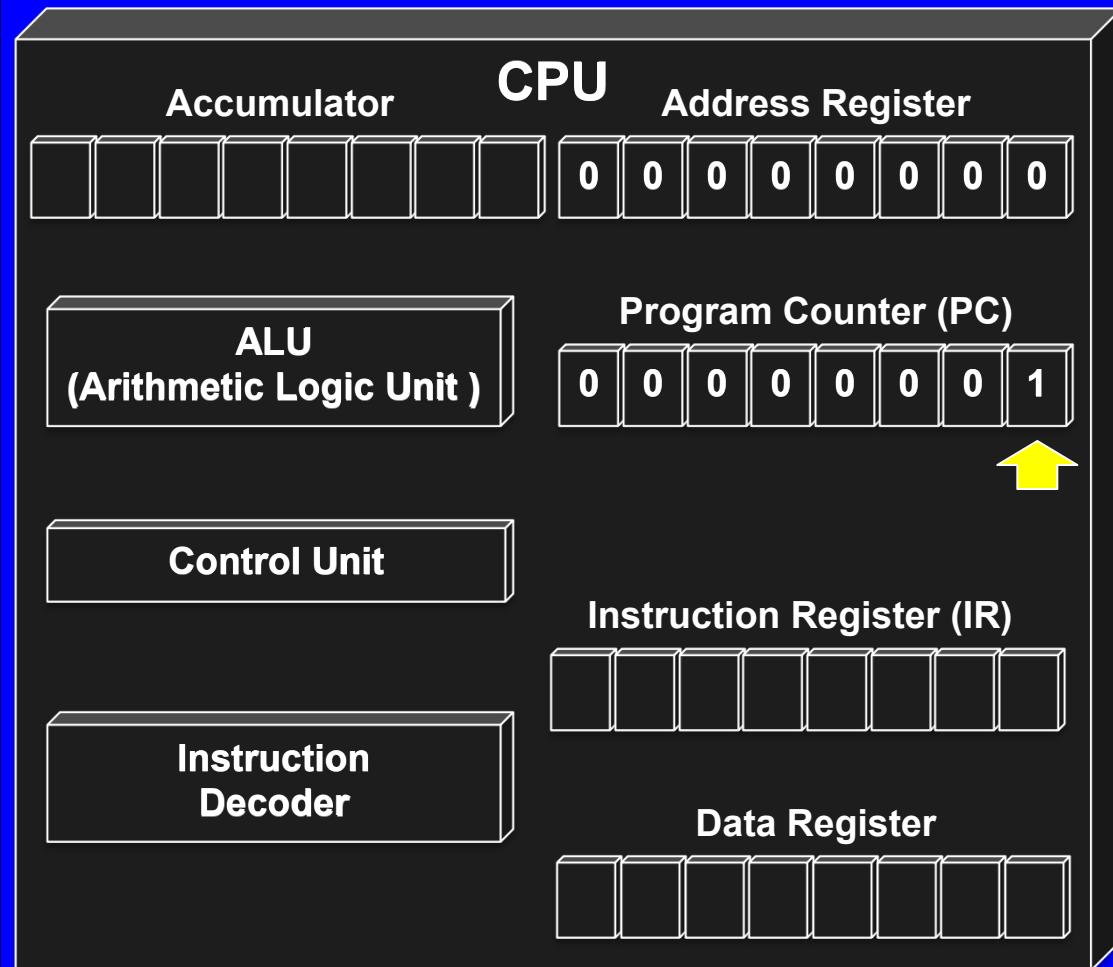


CHU KỲ TÌM NẠP LỆNH
LDA

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	29

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của PC được tăng lên 1.

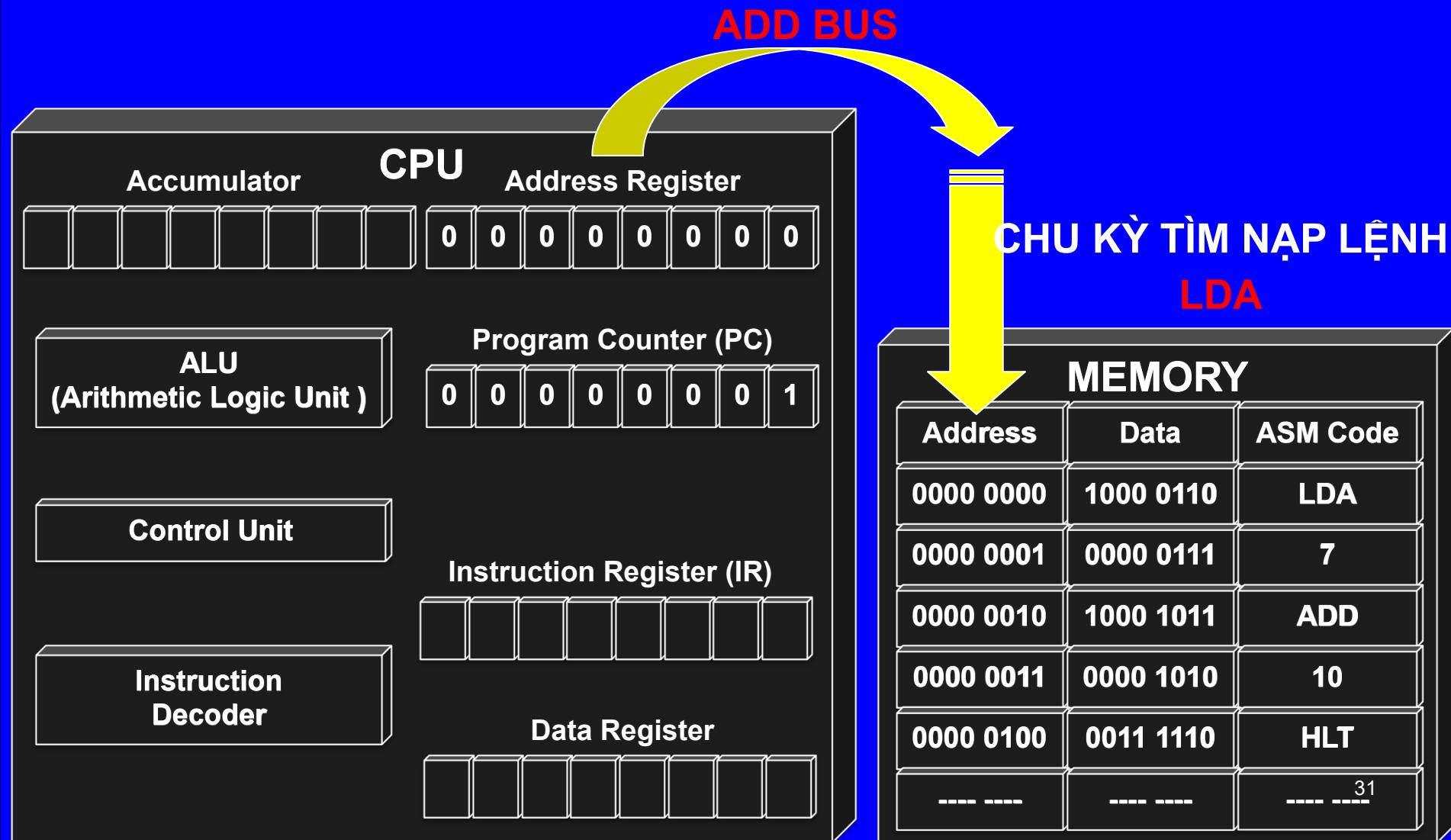


CHU KỲ TÌM NẠP LỆNH
LDA

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	-----

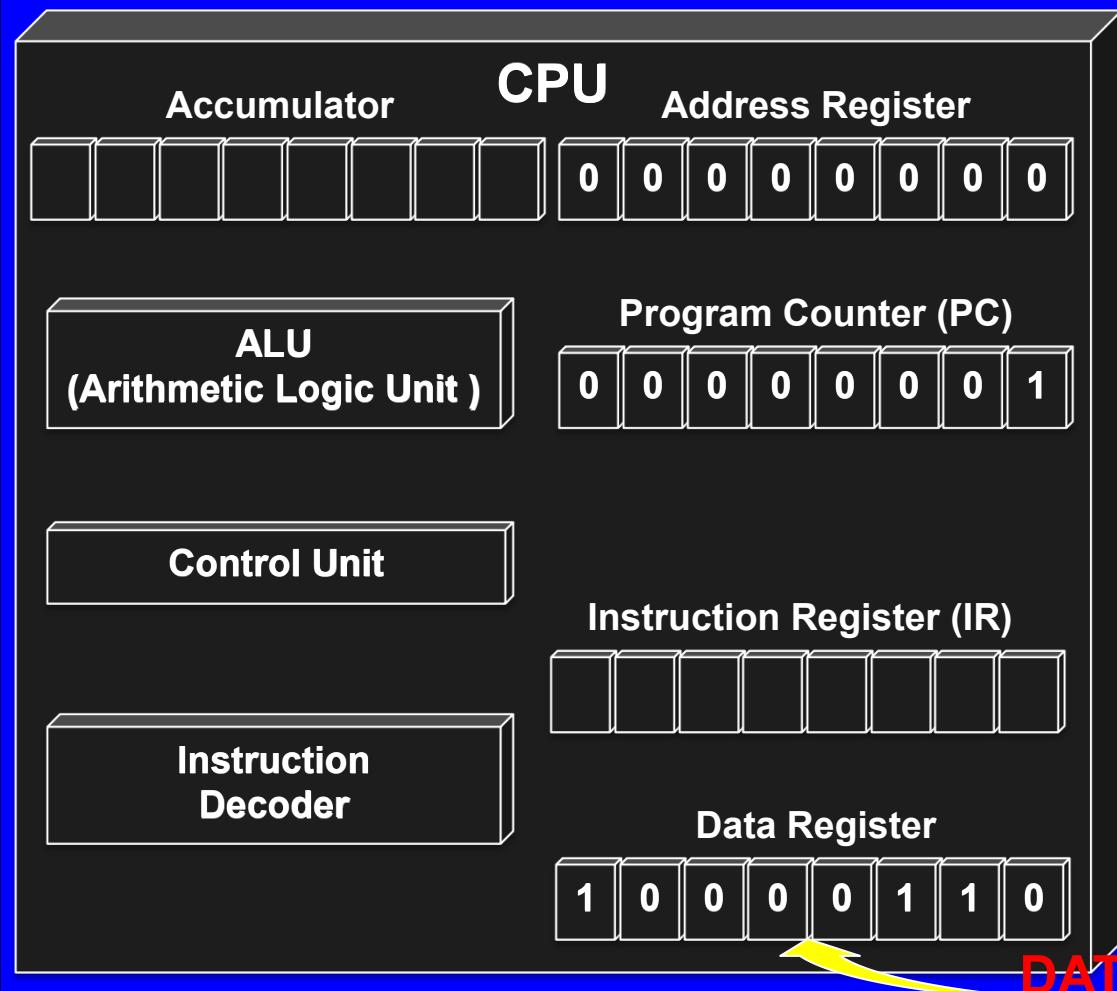
QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của thanh ghi địa chỉ được đặt lên bus địa chỉ.

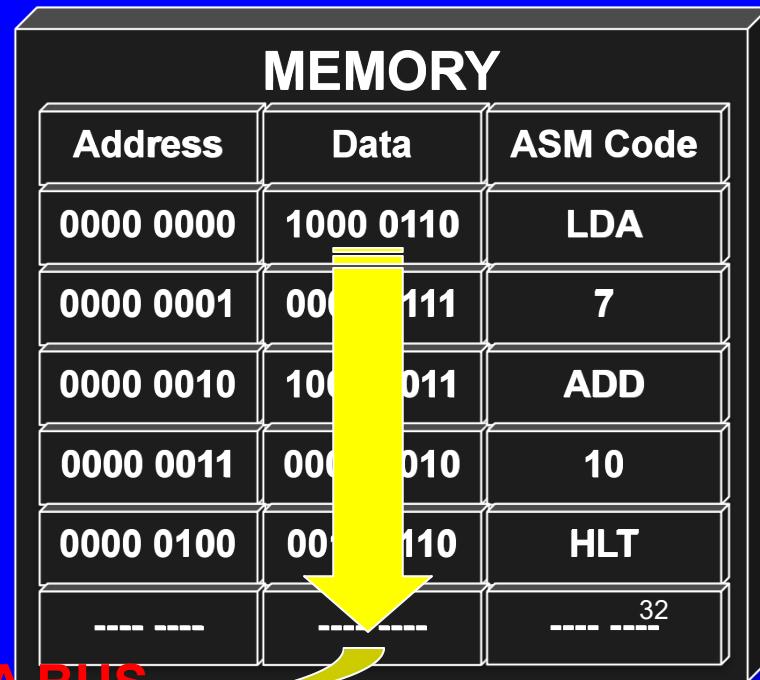


QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của ô nhớ được chọn truyền tới thanh ghi dữ liệu (Data Register).



**CHU KỲ TÌM NẠP LỆNH
LDA**



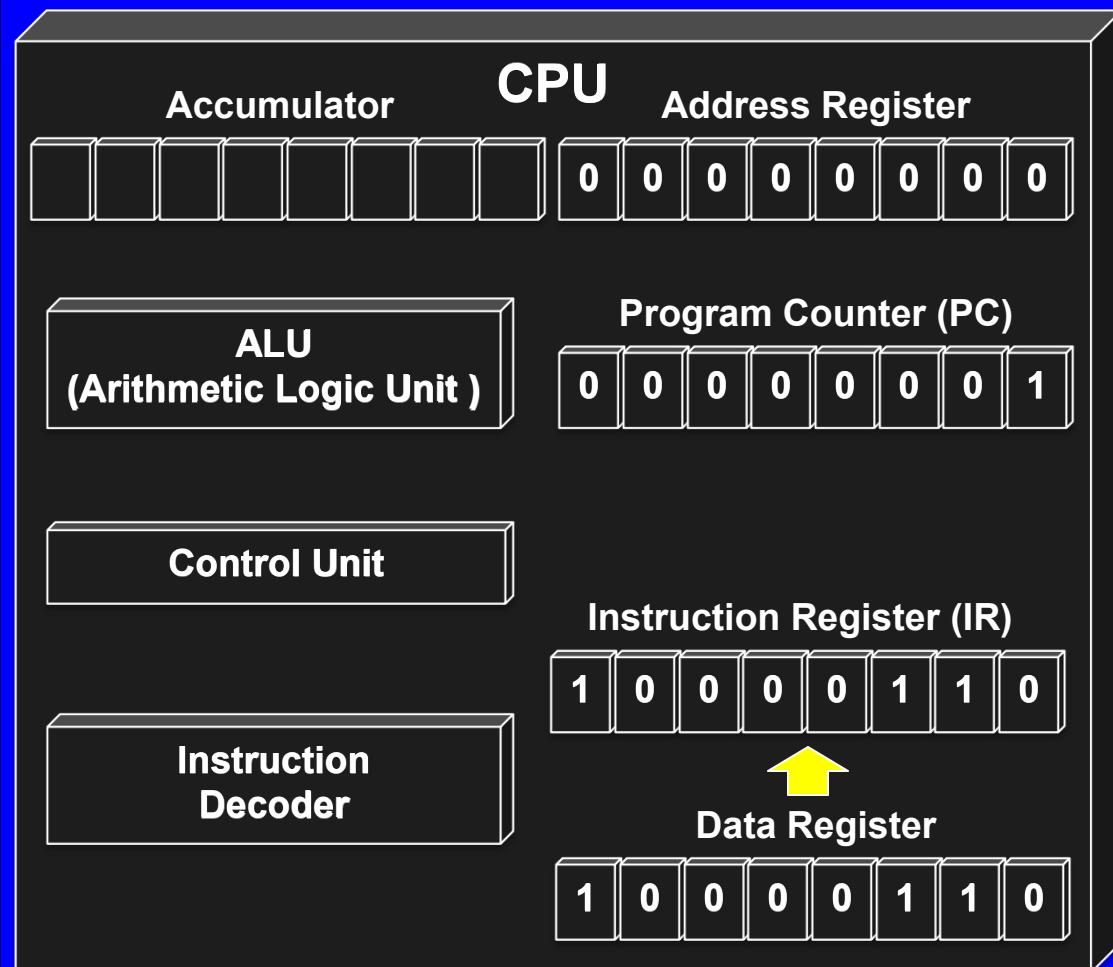
MEMORY:

Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 1111	7
0000 0010	1000 0111	ADD
0000 0011	0000 0100	10
0000 0100	0000 1110	HLT
-----	-----	32

A yellow arrow labeled "DATA BUS" points from the CPU's Data Register to the corresponding data row in the memory table (row 0000 0000, value 1000 0110).

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của thanh ghi dữ liệu được chuyển sang IR.

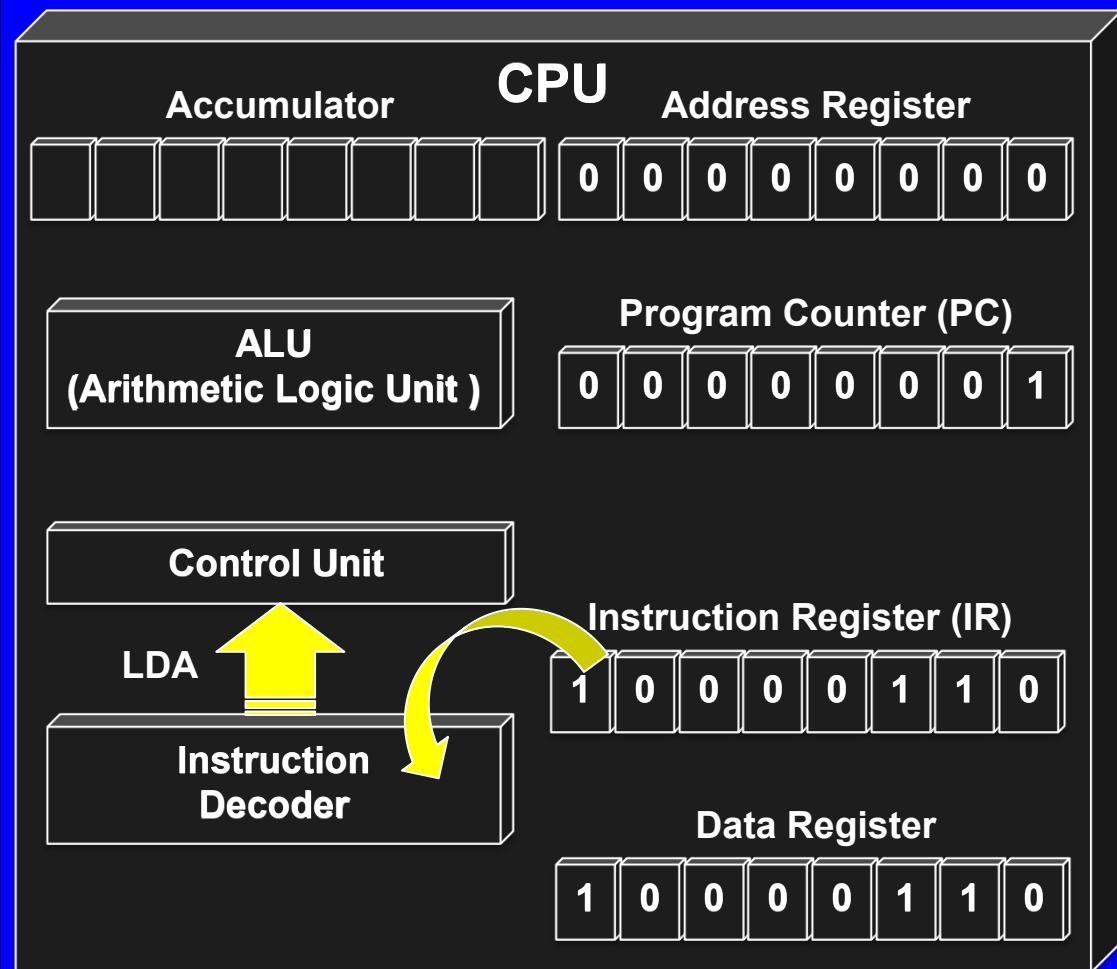


CHU KỲ TÌM NẠP LỆNH
LDA

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	33

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- Nội dung của IR được giải mã bởi bộ giải mã lệnh (Instruction Decoder). Bộ giải mã lệnh tác động đến đơn vị điều khiển (CU) để tạo ra tín hiệu điều khiển cần thiết.

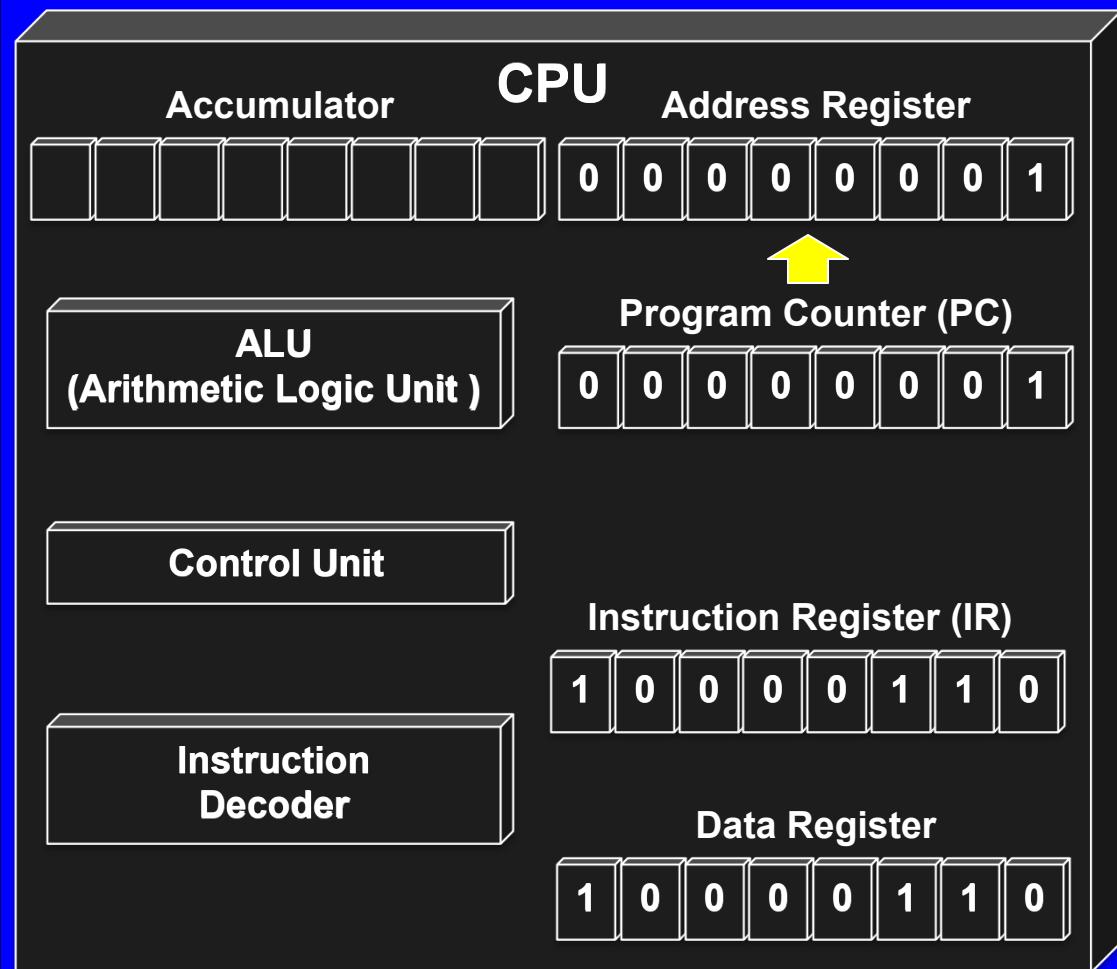


**CHU KỲ TÌM NẠP LỆNH
LDA**

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	-----

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của PC được chuyển sang thanh ghi địa chỉ (Address Register).

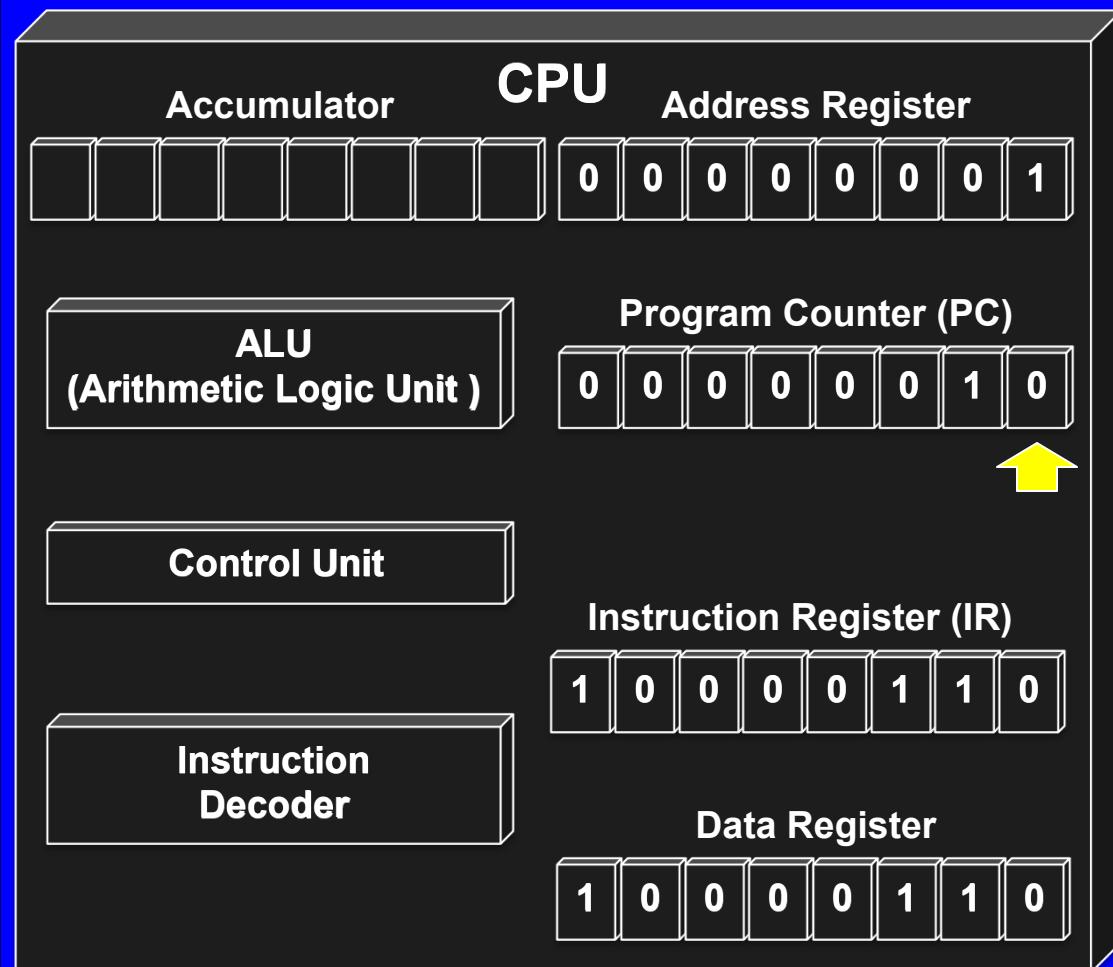


CHU KỲ THỰC THI LỆNH
LDA

Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	35

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của PC được tăng lên 1 cho chu kỳ tìm nạp kế tiếp.

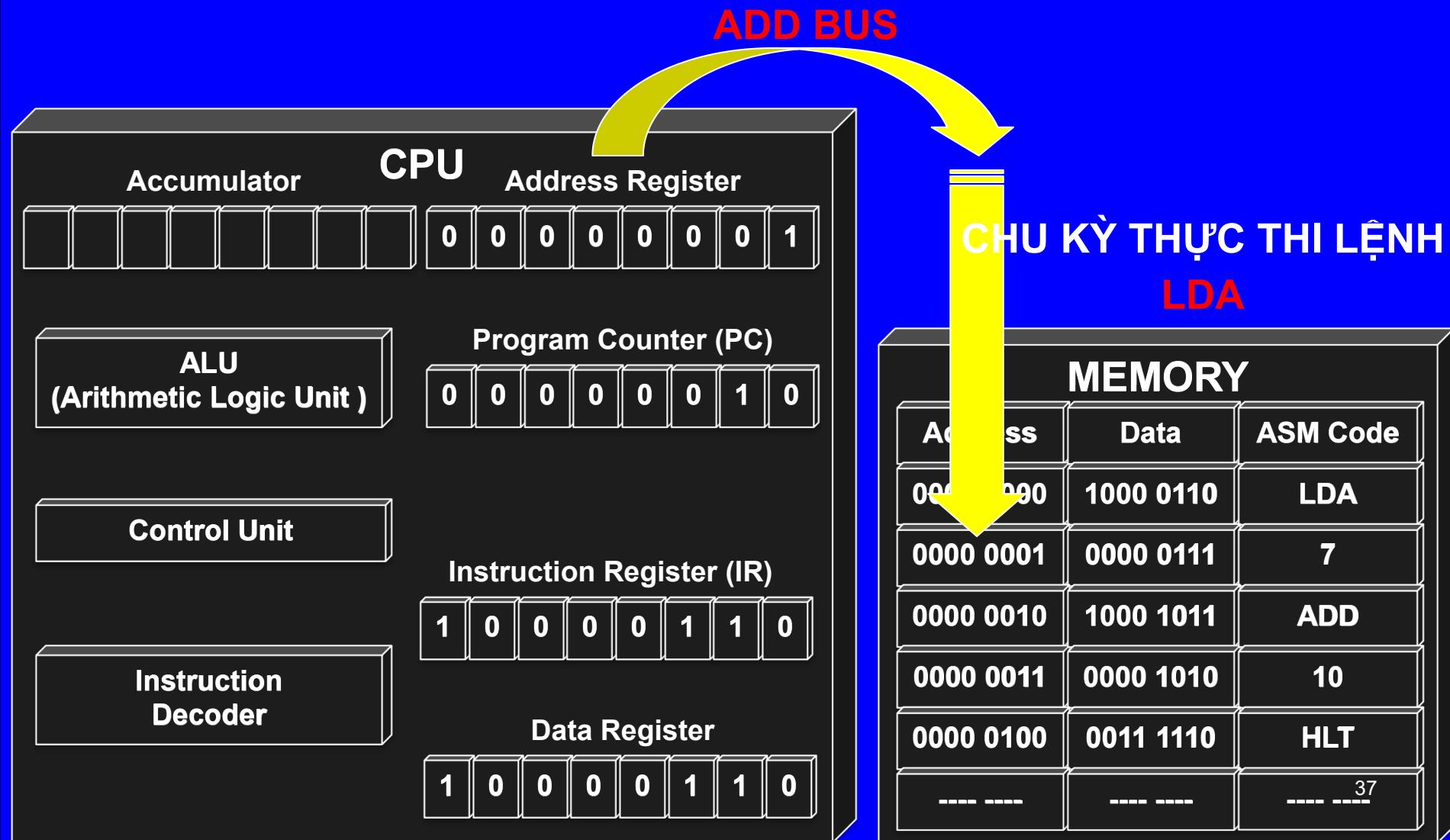


CHU KỲ THỰC THI LỆNH
LDA

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	-----

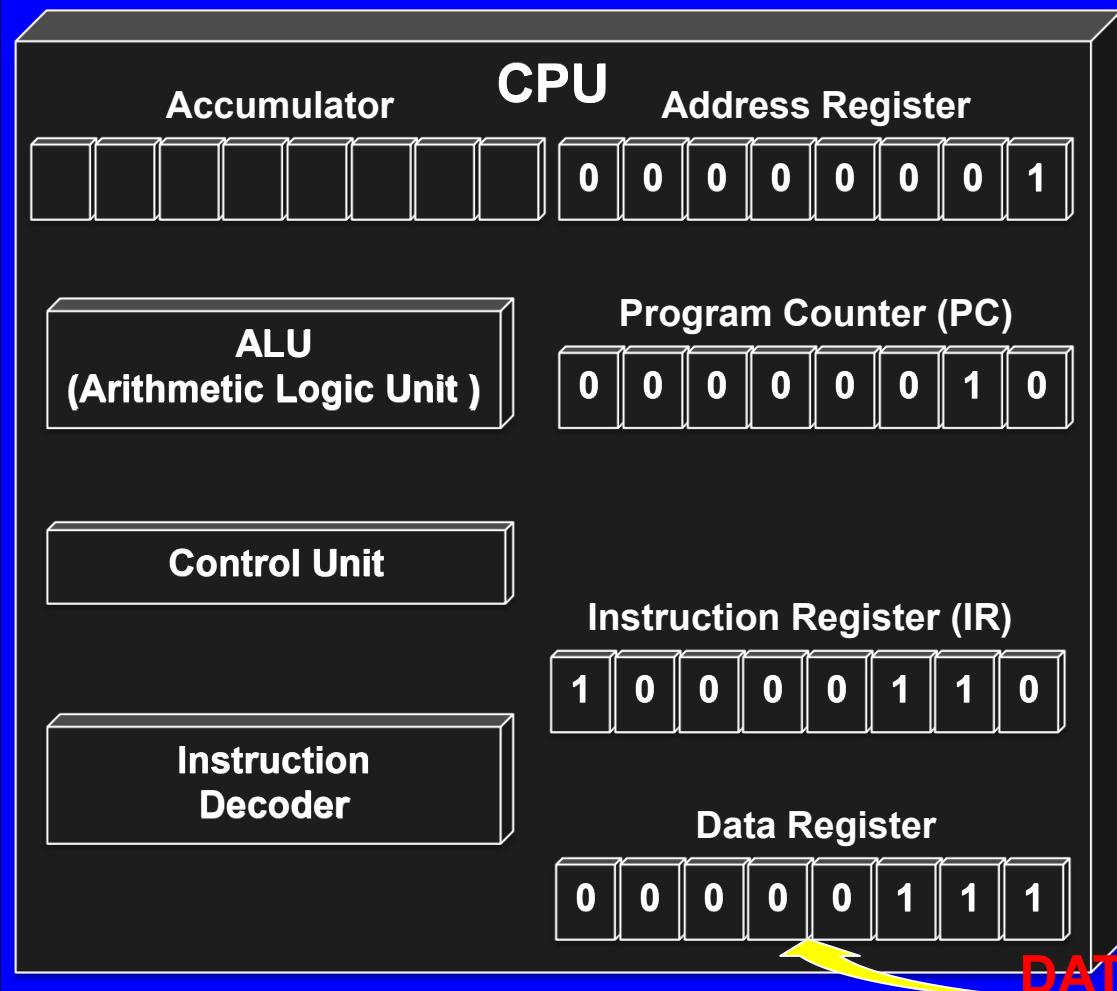
QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Địa chỉ của toán hạng được đặt lên bus địa chỉ.

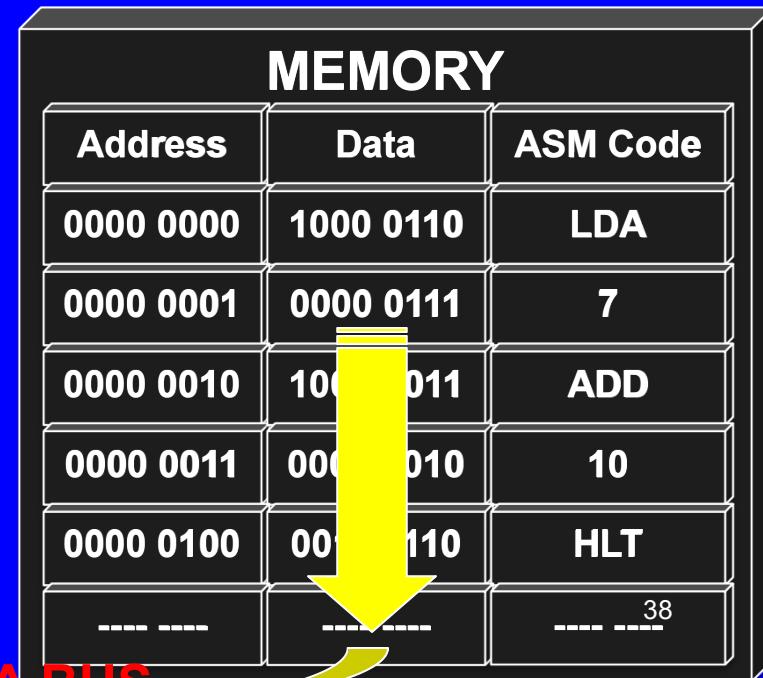


QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Toán hạng được chọn truyền tới thanh ghi dữ liệu (Data Register).



CHU KỲ THỰC THI LỆNH
LDA



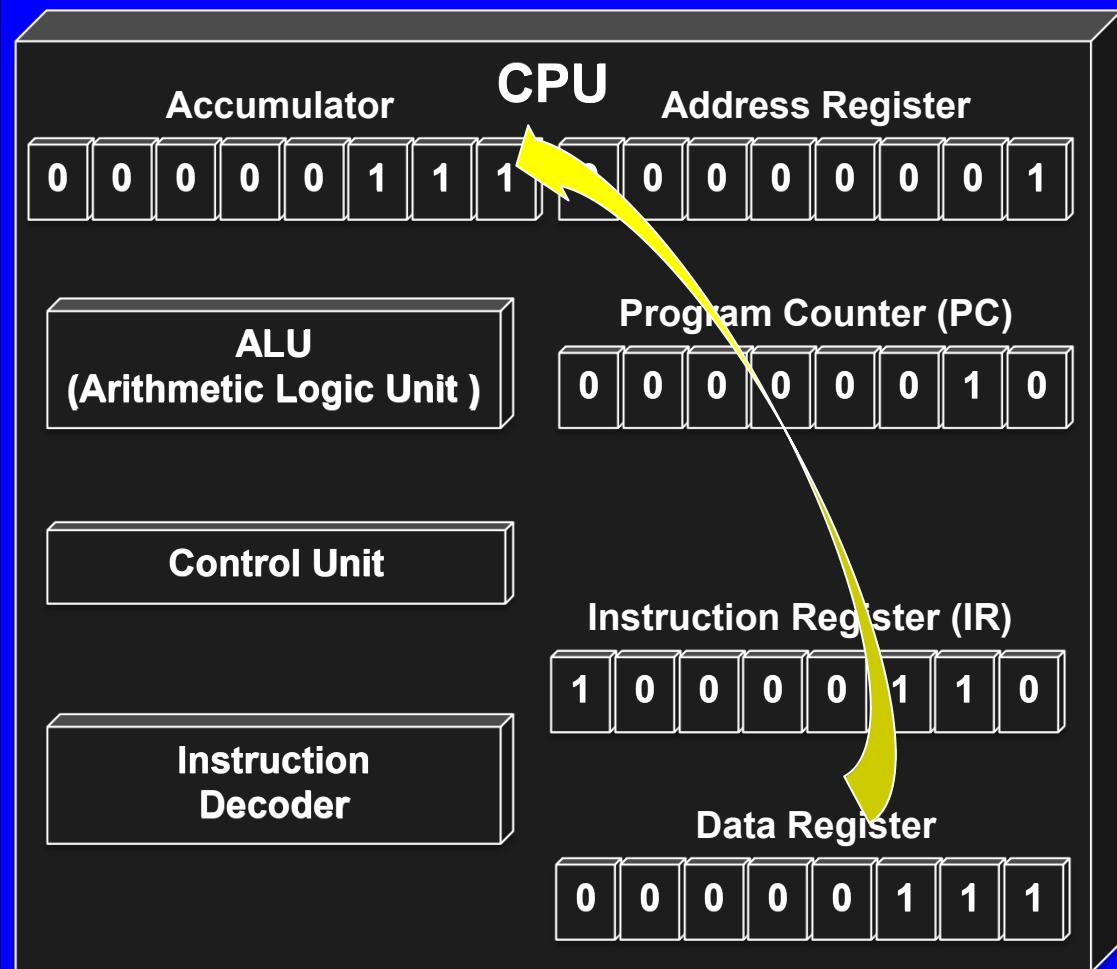
MEMORY

Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 0011	ADD
0000 0011	0000 0010	10
0000 0100	0000 1110	HLT
-----	-----	38

A yellow arrow labeled **DATA BUS** points from the CPU's Data Register to the memory location at address 0000 0000, where the value 1000 0110 is stored.

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của thanh ghi dữ liệu đồng thời cũng được truyền vào thanh ghi A.

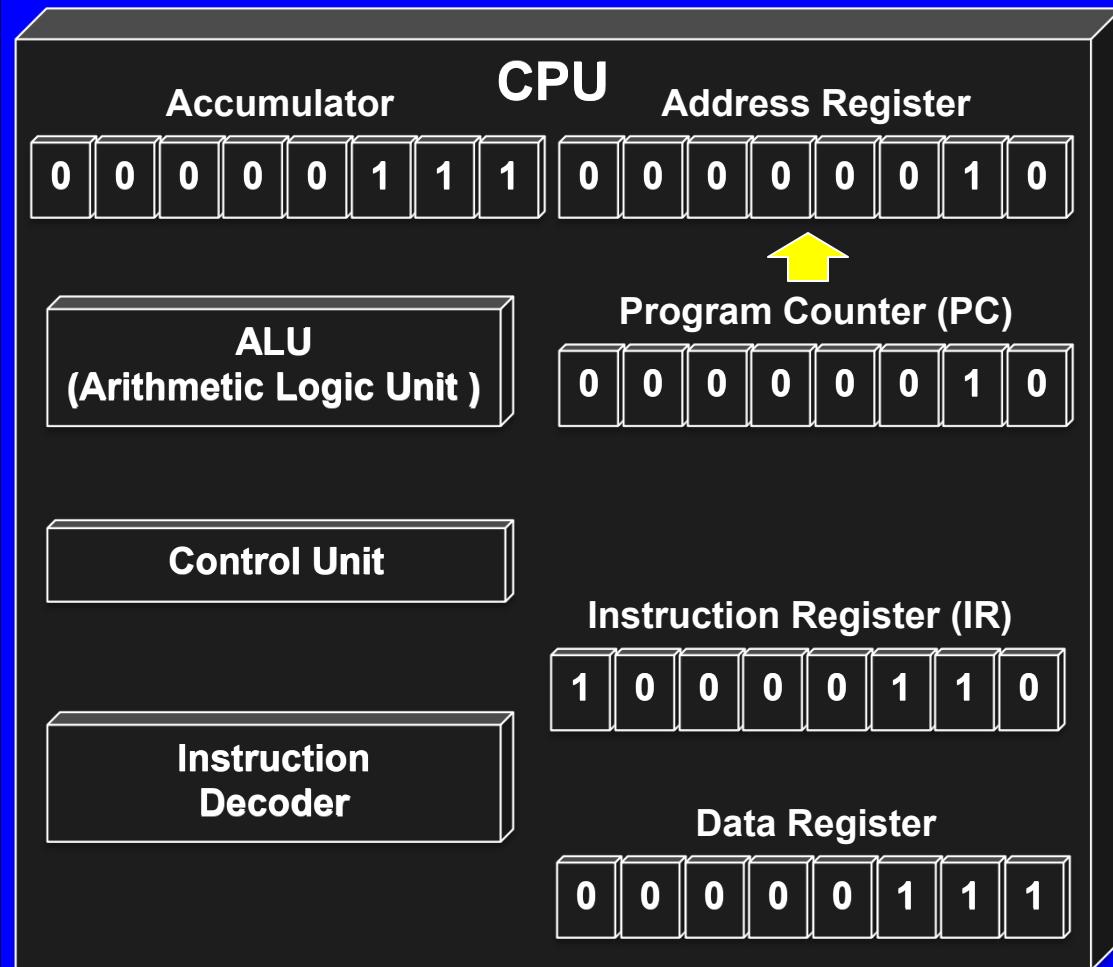


CHU KỲ THỰC THI LỆNH
LDA

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	39

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của PC được chuyển sang thanh ghi địa chỉ (Address Register).

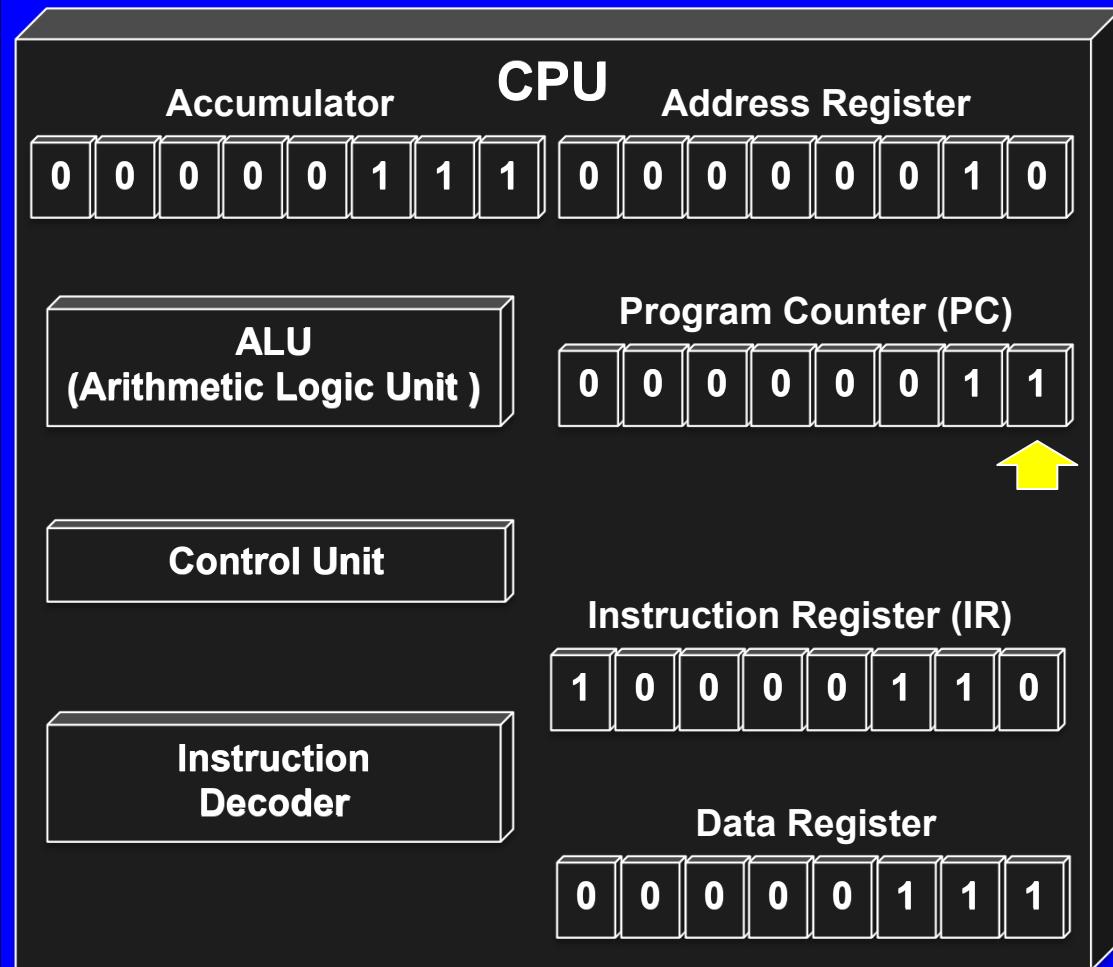


CHU KỲ TÌM NẠP LỆNH
ADD

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	40

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của PC được tăng lên 1.

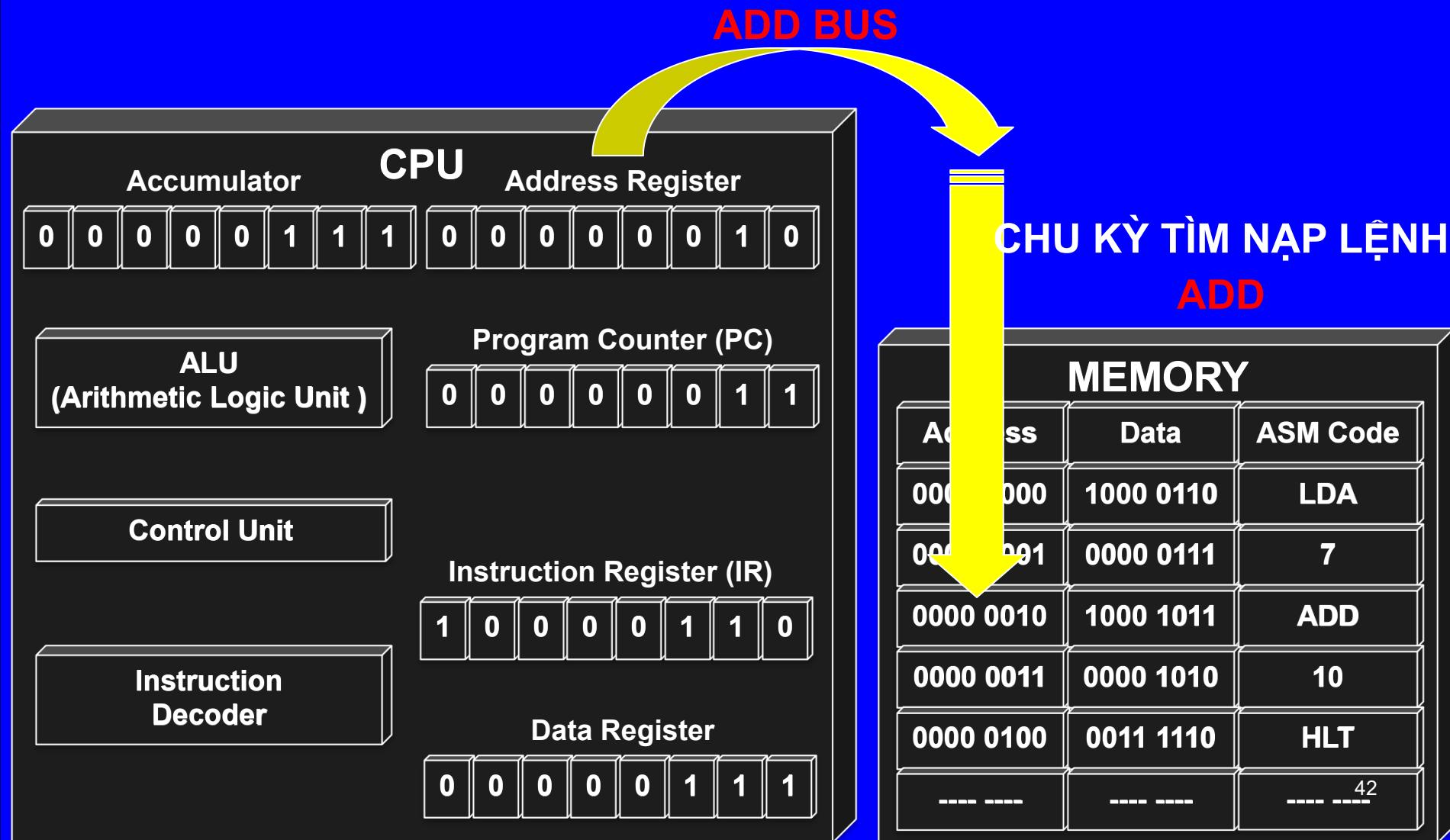


CHU KỲ TÌM NẠP LỆNH
ADD

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	41

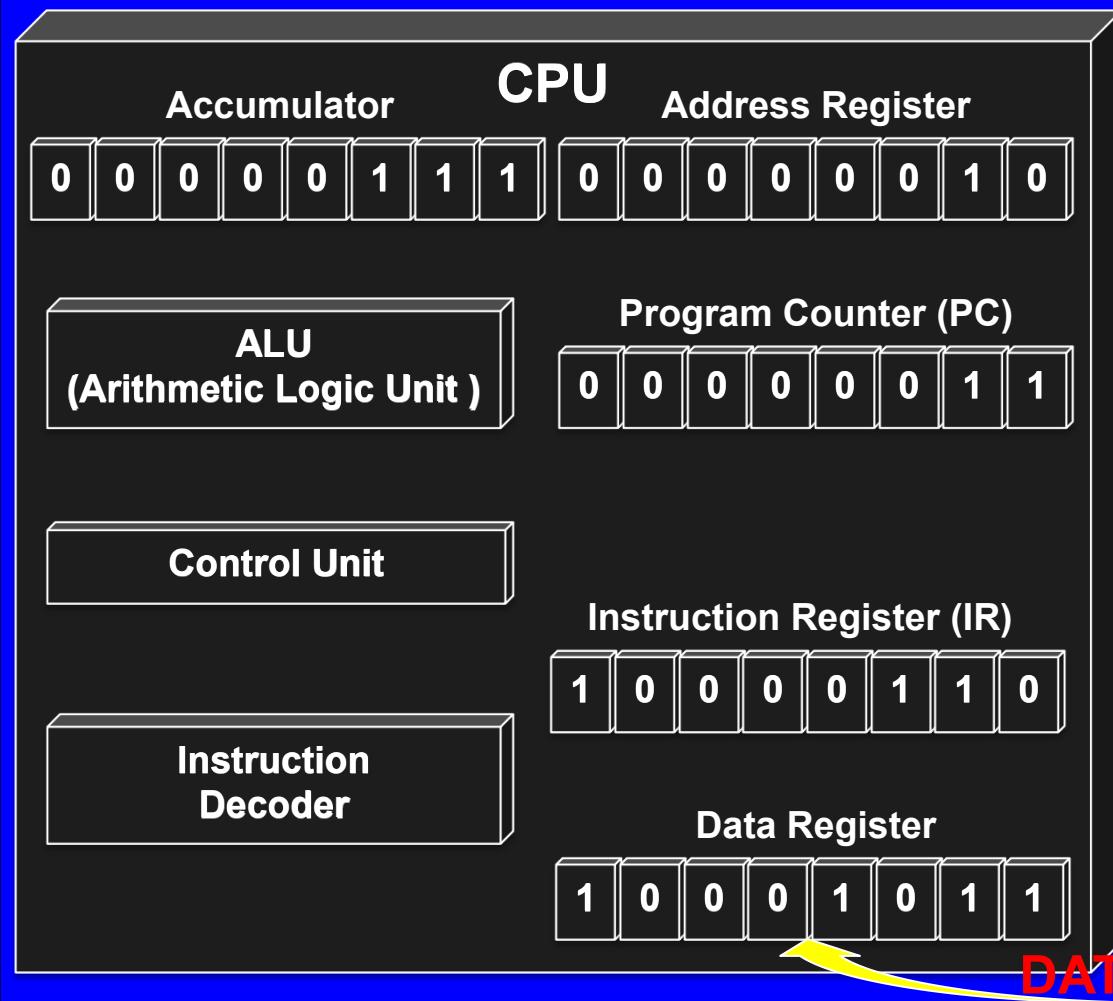
QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của thanh ghi địa chỉ được đặt lên bus địa chỉ.



QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của ô nhớ được chọn truyền tới thanh ghi dữ liệu (Data Register).

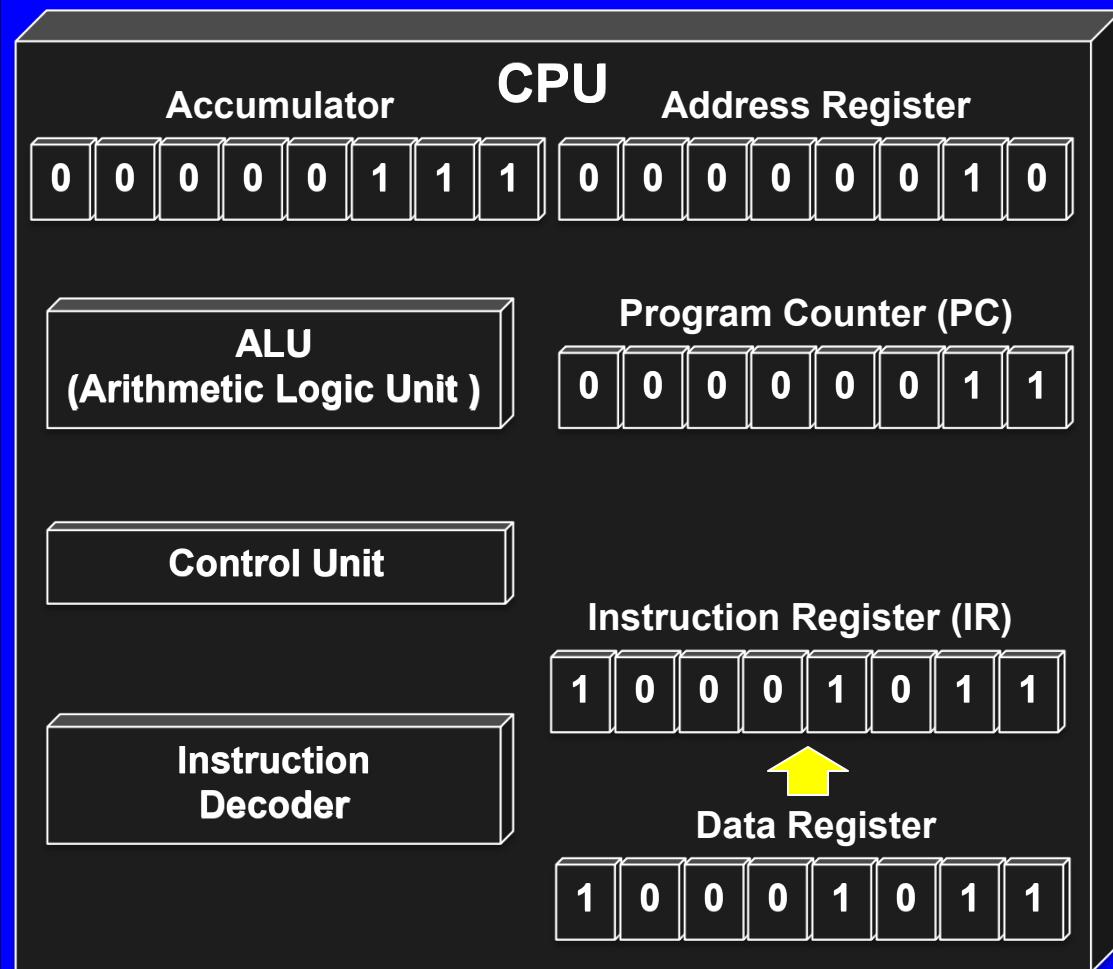


CHU KỲ TÌM NẠP LỆNH
ADD

Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 0010	10
0000 0100	00 110	HLT
-----	-----	43

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của thanh ghi dữ liệu được chuyển sang IR.

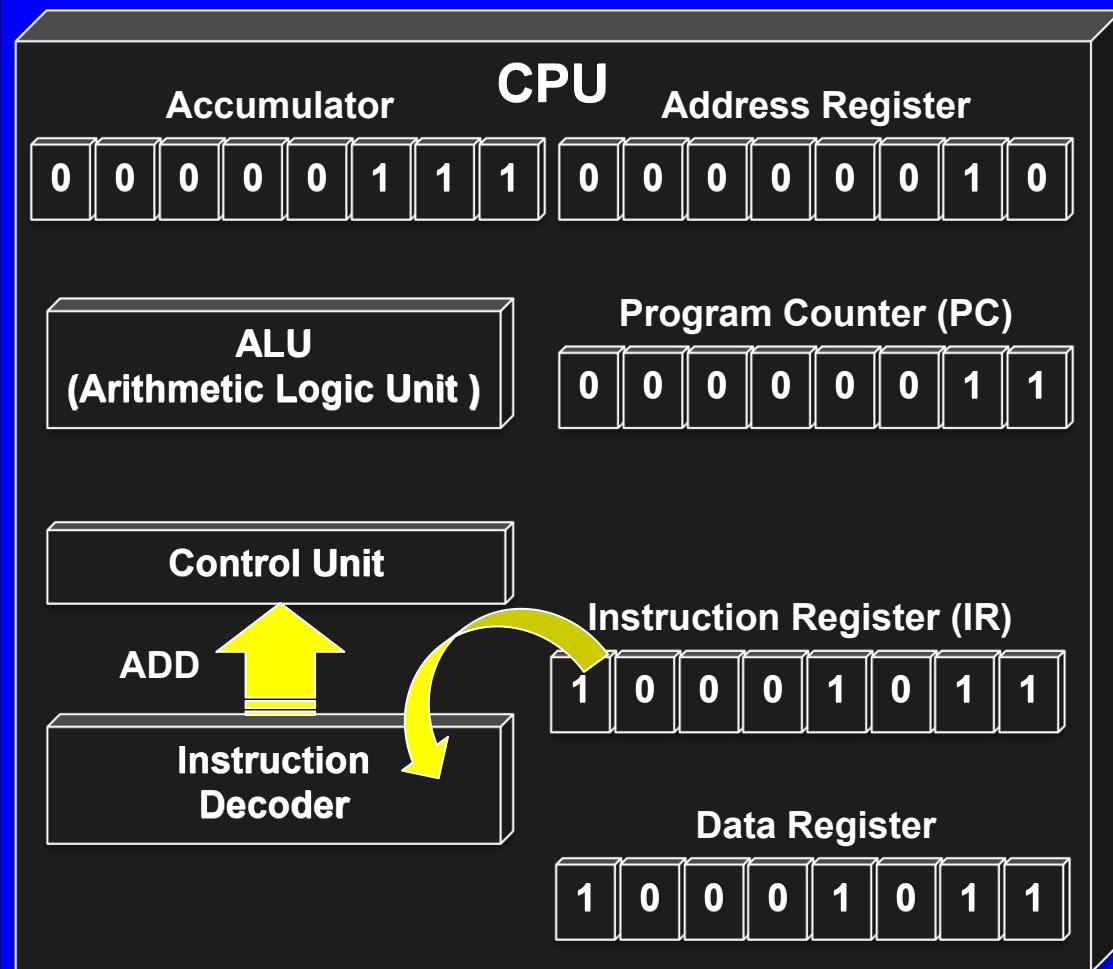


CHU KỲ TÌM NẠP LỆNH
ADD

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	-----

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- Nội dung của IR được giải mã bởi bộ giải mã lệnh (Instruction Decoder). Bộ giải mã lệnh tác động đến đơn vị điều khiển (CU) để tạo ra tín hiệu điều khiển cần thiết.

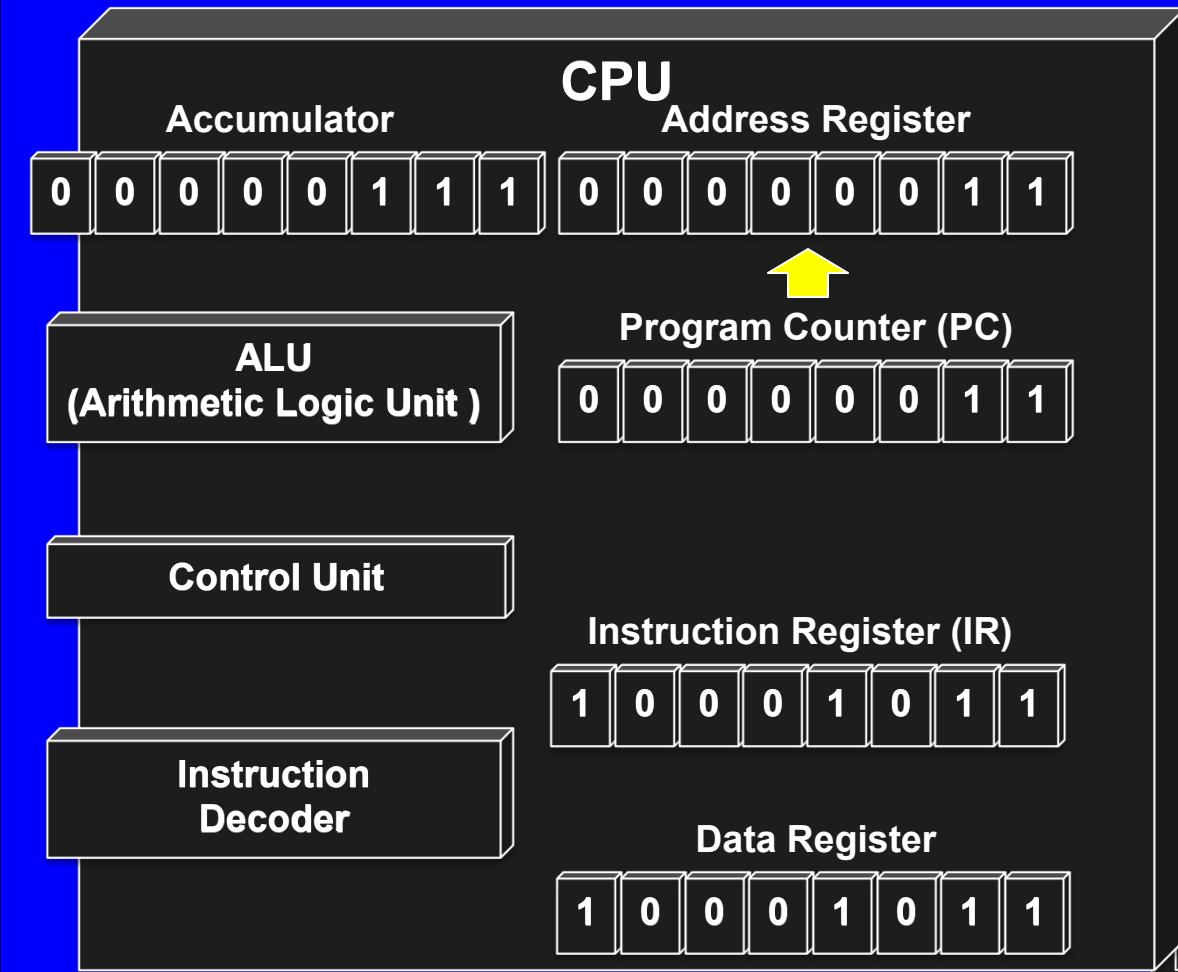


**CHU KỲ TÌM NẠP LỆNH
ADD**

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	45

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của PC được chuyển sang thanh ghi địa chỉ (Address Register).

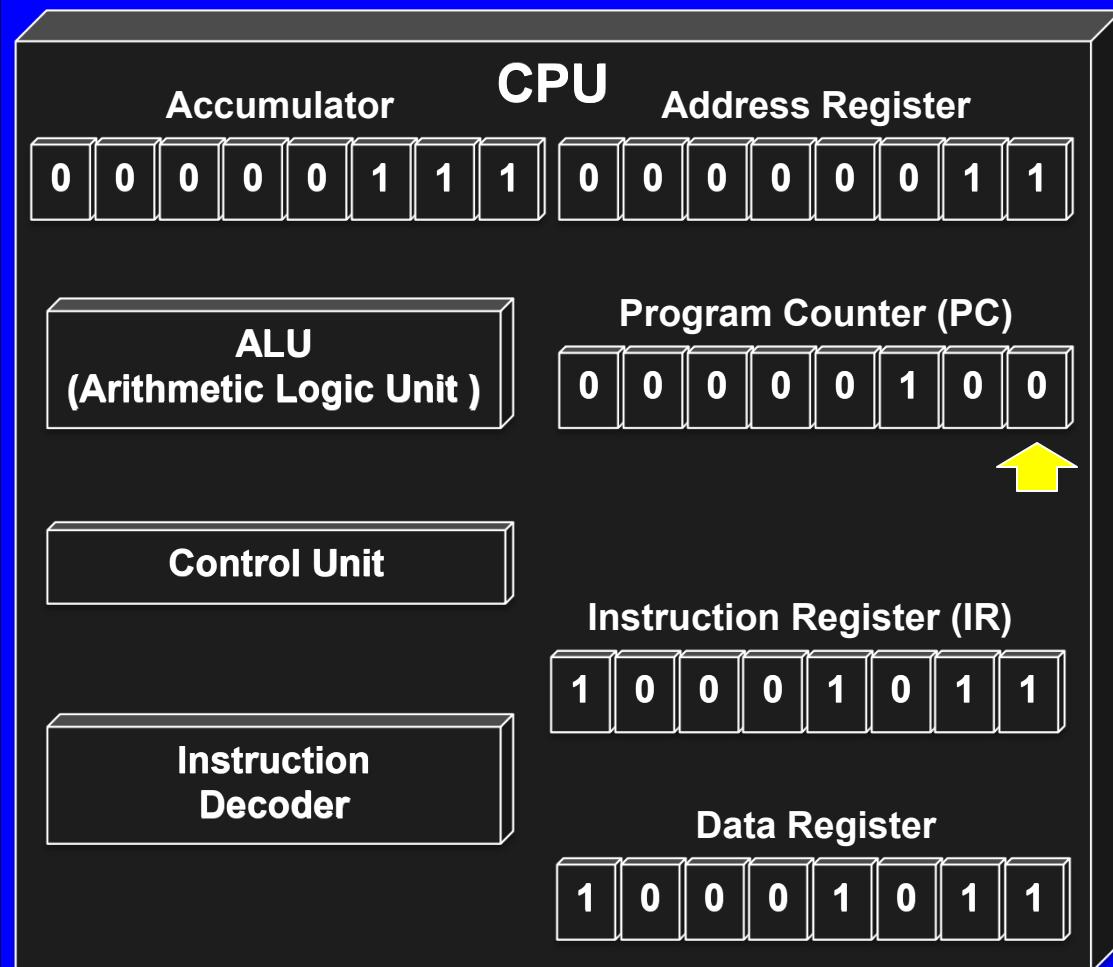


CHU KỲ THỰC THI LỆNH
ADD

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	46

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của PC được tăng lên 1 cho chu kỳ tìm nạp kế tiếp.

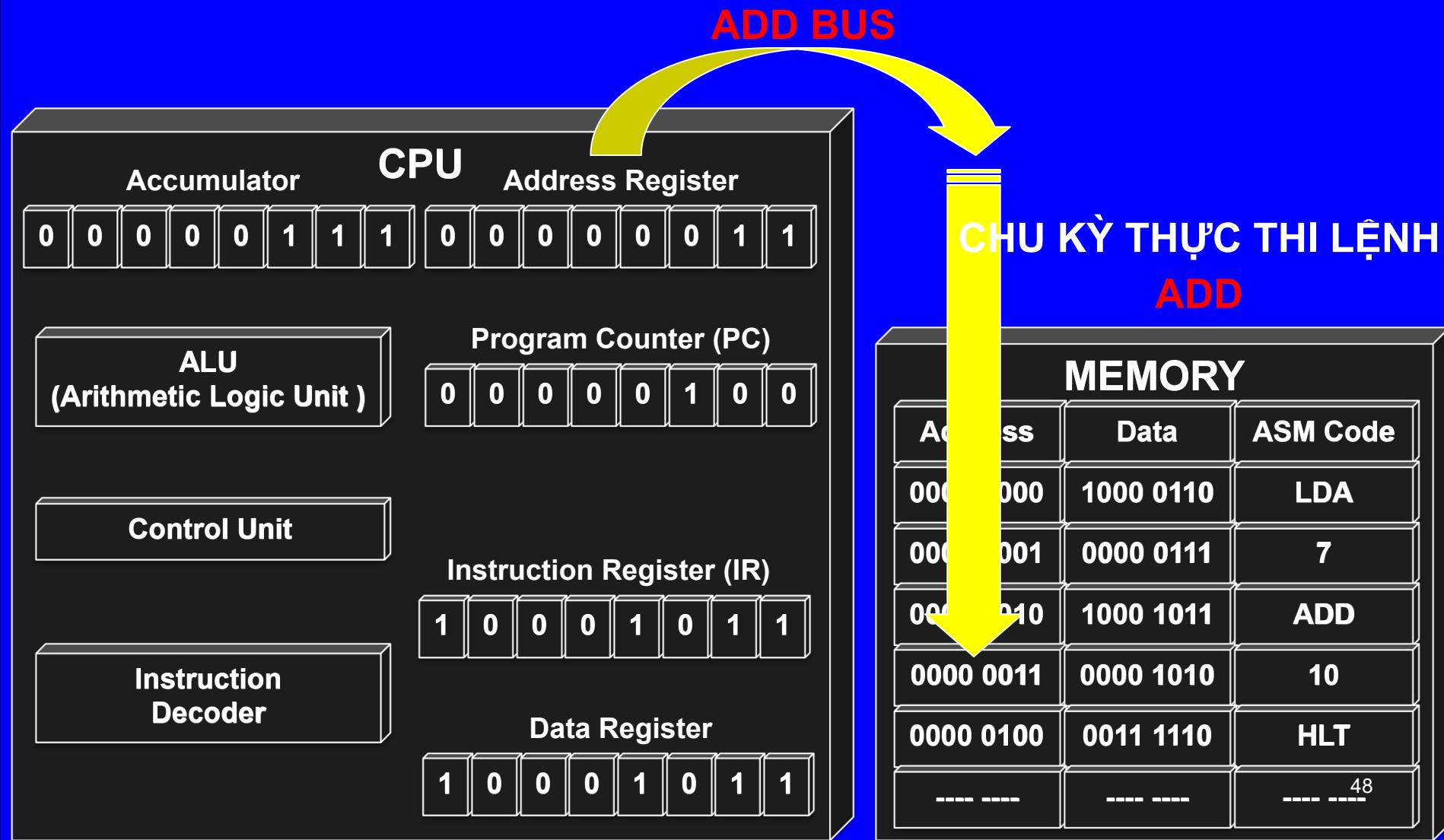


CHU KỲ THỰC THI LỆNH
ADD

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	47

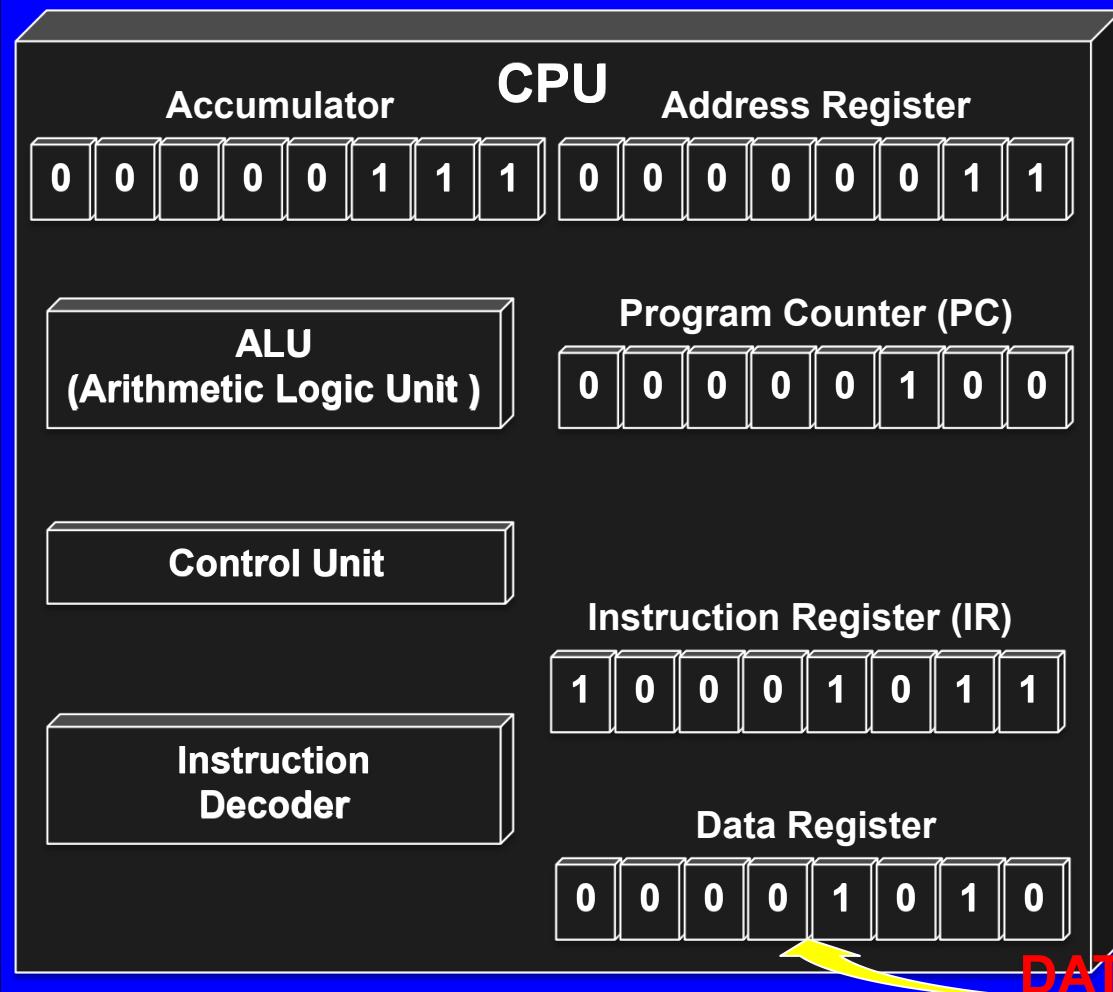
QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Địa chỉ của toán hạng được đặt lên bus địa chỉ.



QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Toán hạng được chọn truyền tới thanh ghi dữ liệu (Data Register).

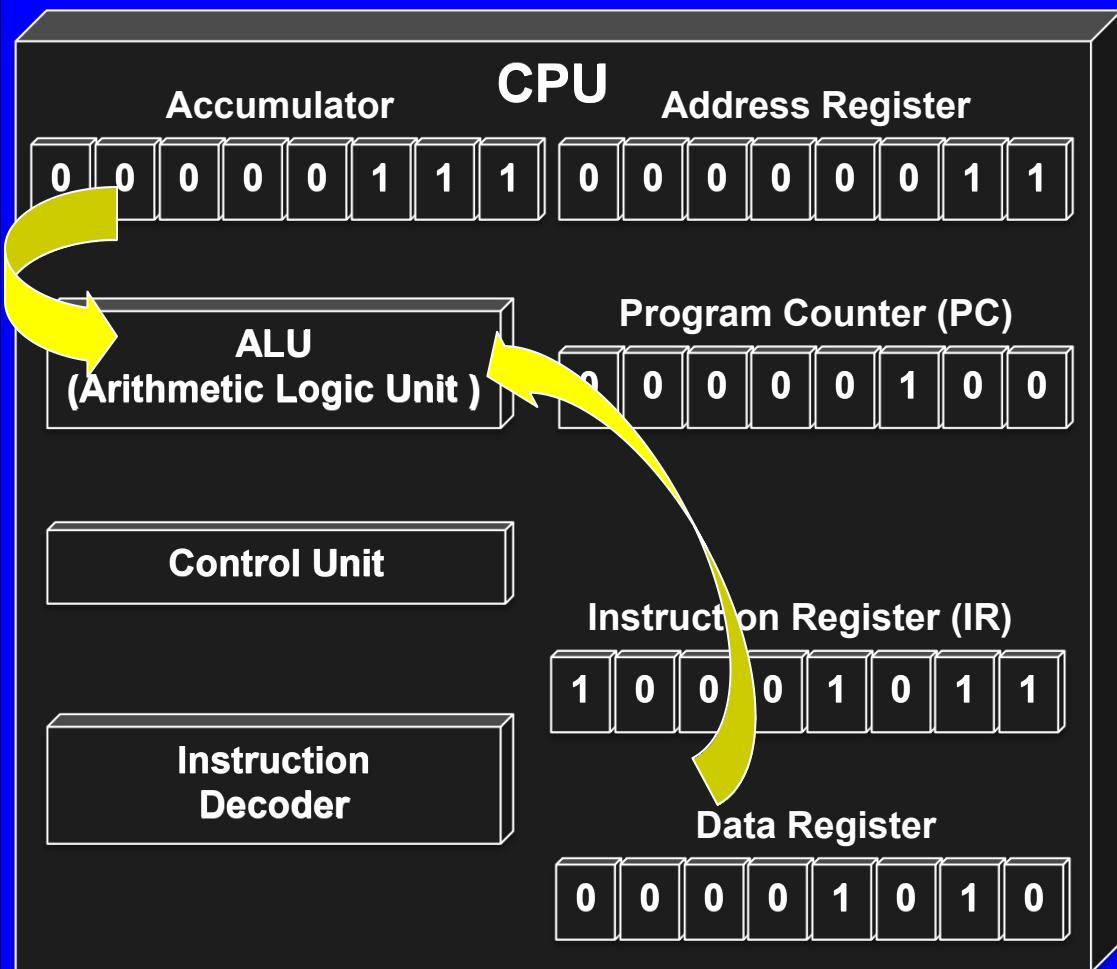


CHU KỲ THỰC THI LỆNH
ADD

Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	00 110	HLT
-----	-----	49

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Toán hạng (10) được chọn truyền tới một ngõ vào của ALU, đồng thời toán hạng (7) từ thanh ghi A cũng được truyền tới ngõ vào còn lại của ALU.

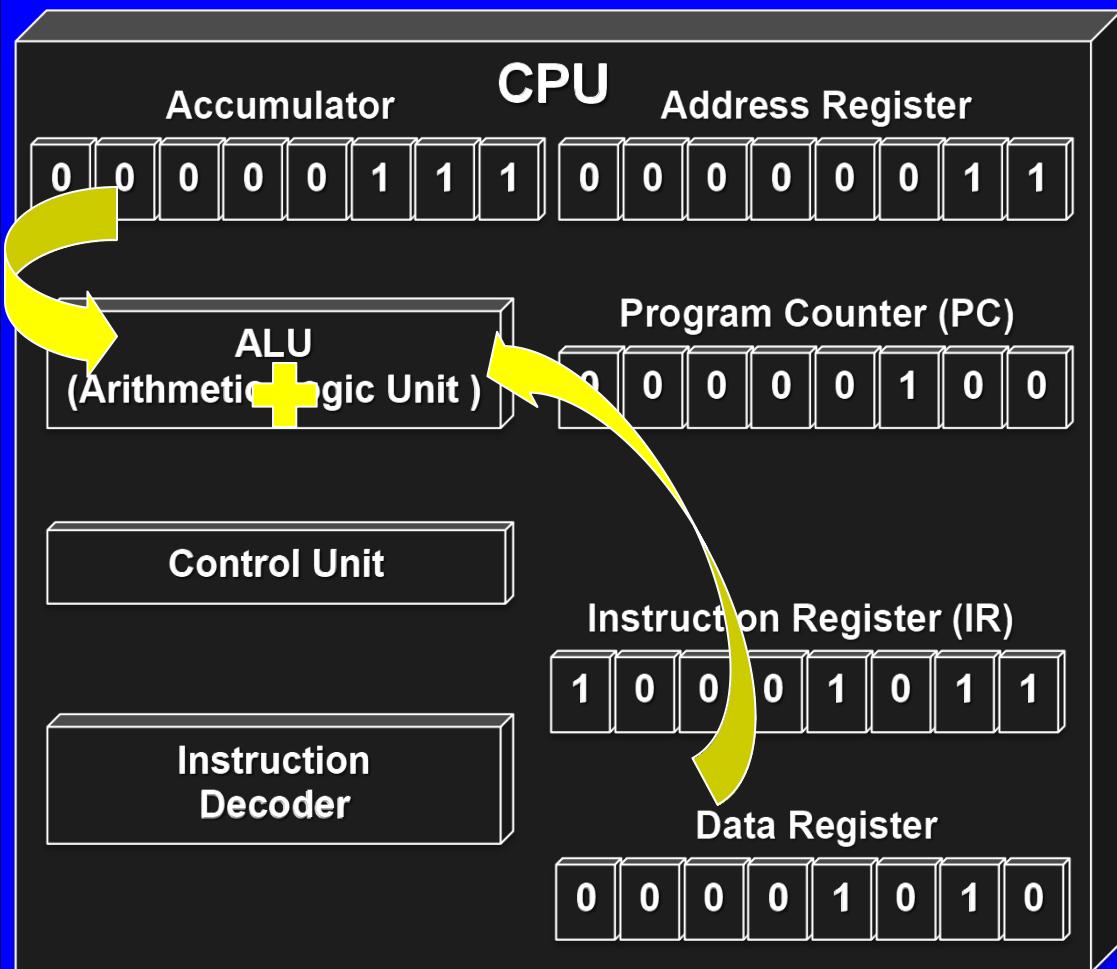


CHU KỲ THỰC THI LỆNH
ADD

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	50

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ ALU thực hiện việc cộng hai toán hạng này và trả kết quả (17) vào thanh ghi A.

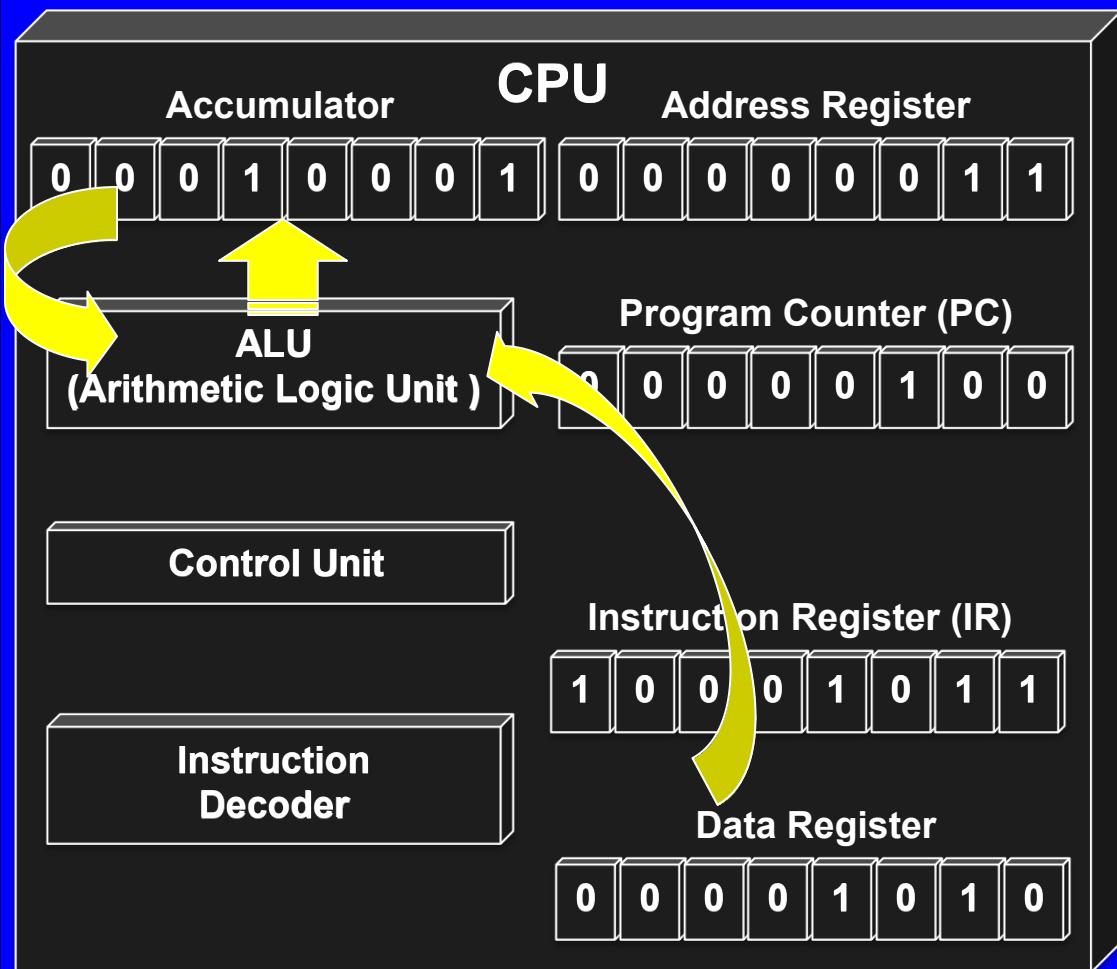


CHU KỲ THỰC THI LỆNH
ADD

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	51

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ ALU thực hiện việc cộng hai toán hạng này và trả kết quả (17) vào thanh ghi A.

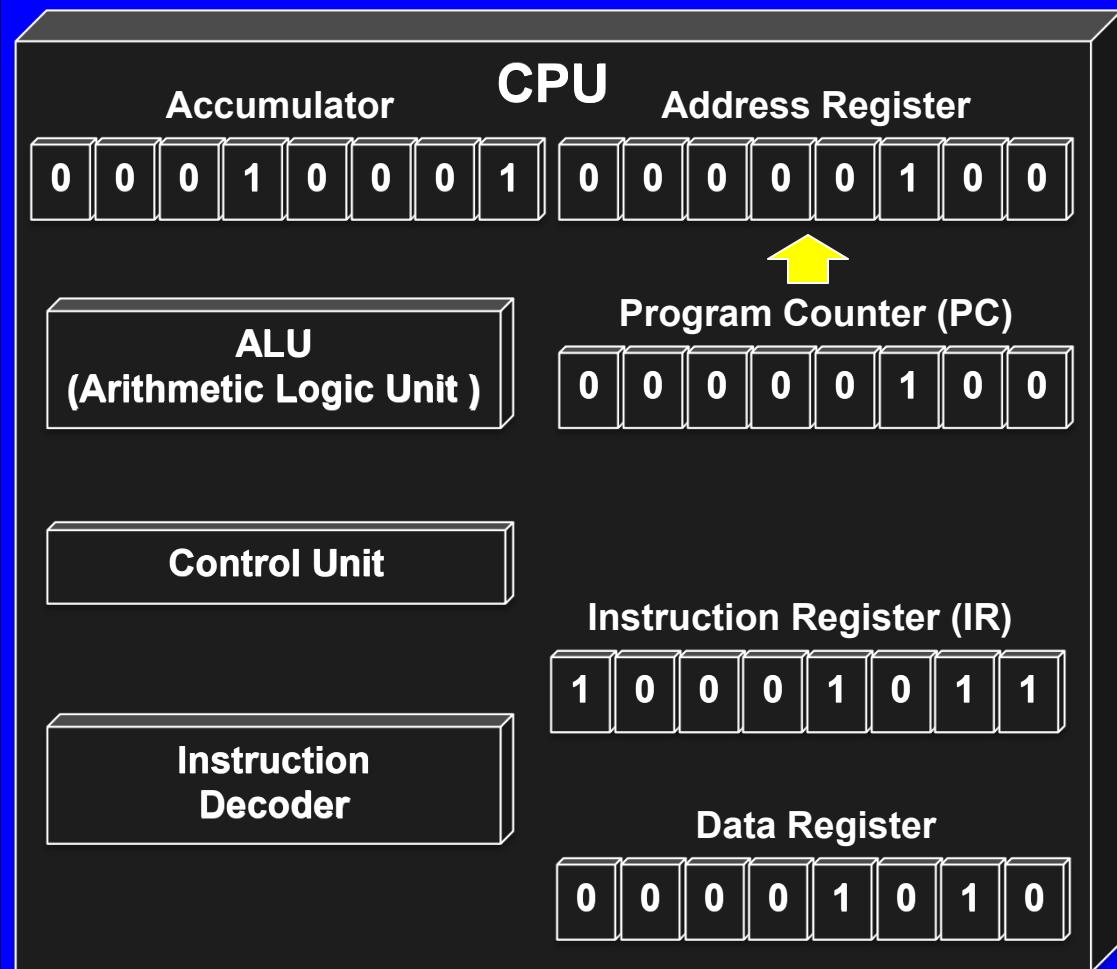


CHU KỲ THỰC THI LỆNH
ADD

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	52

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của PC được chuyển sang thanh ghi địa chỉ (Address Register).

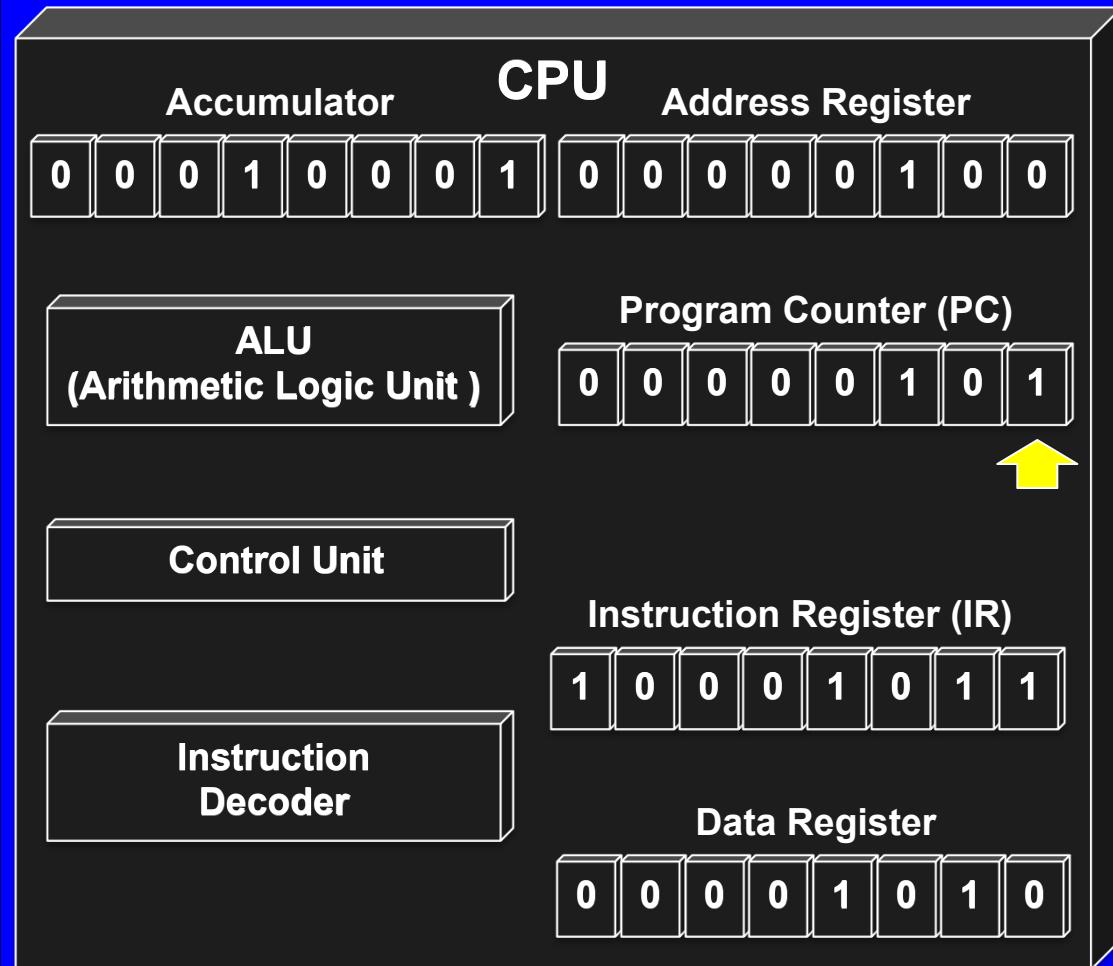


CHU KỲ TÌM NẠP LỆNH
HALT

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	53

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của PC được tăng lên 1.

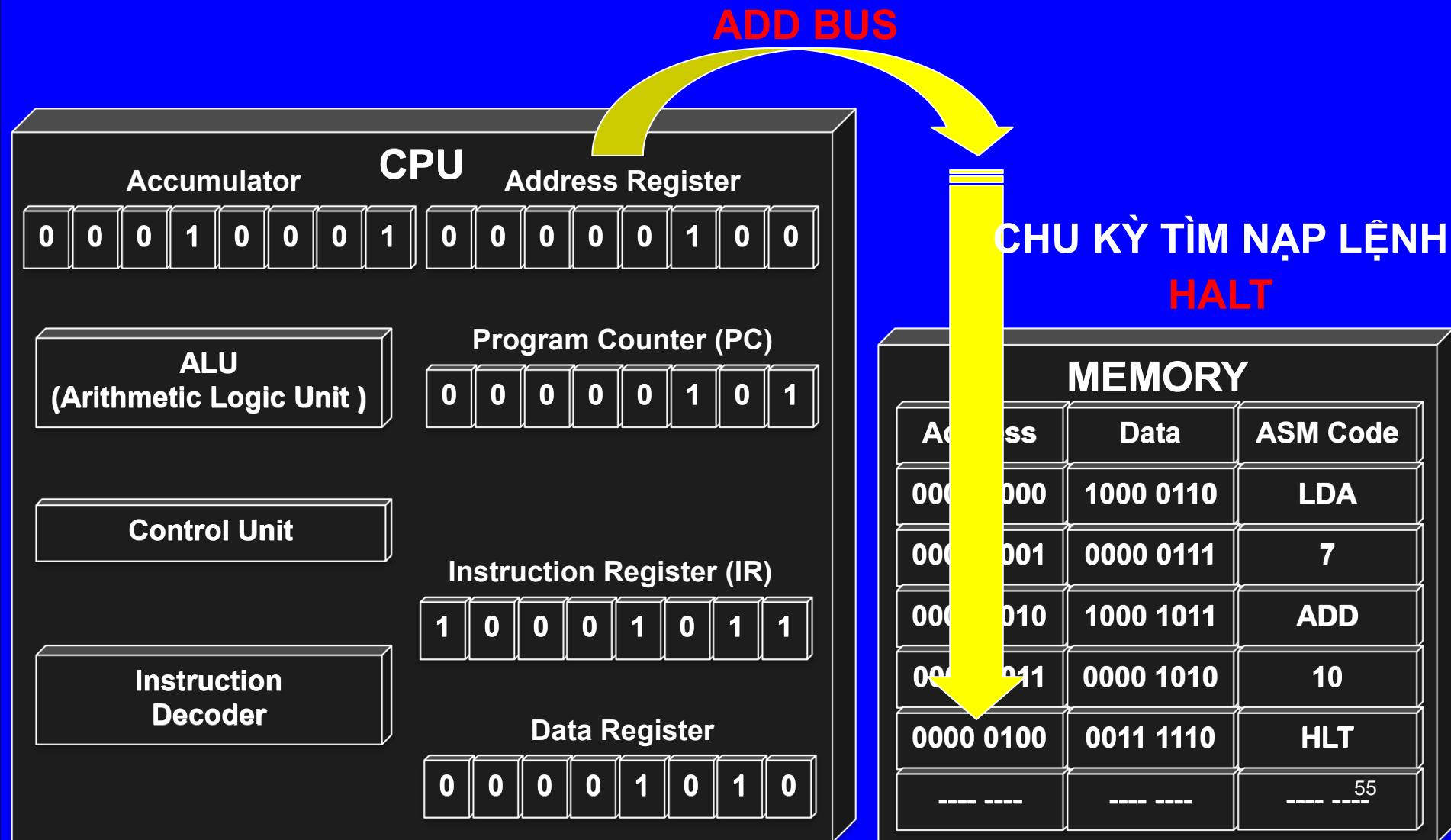


CHU KỲ TÌM NẠP LỆNH
HALT

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	-----

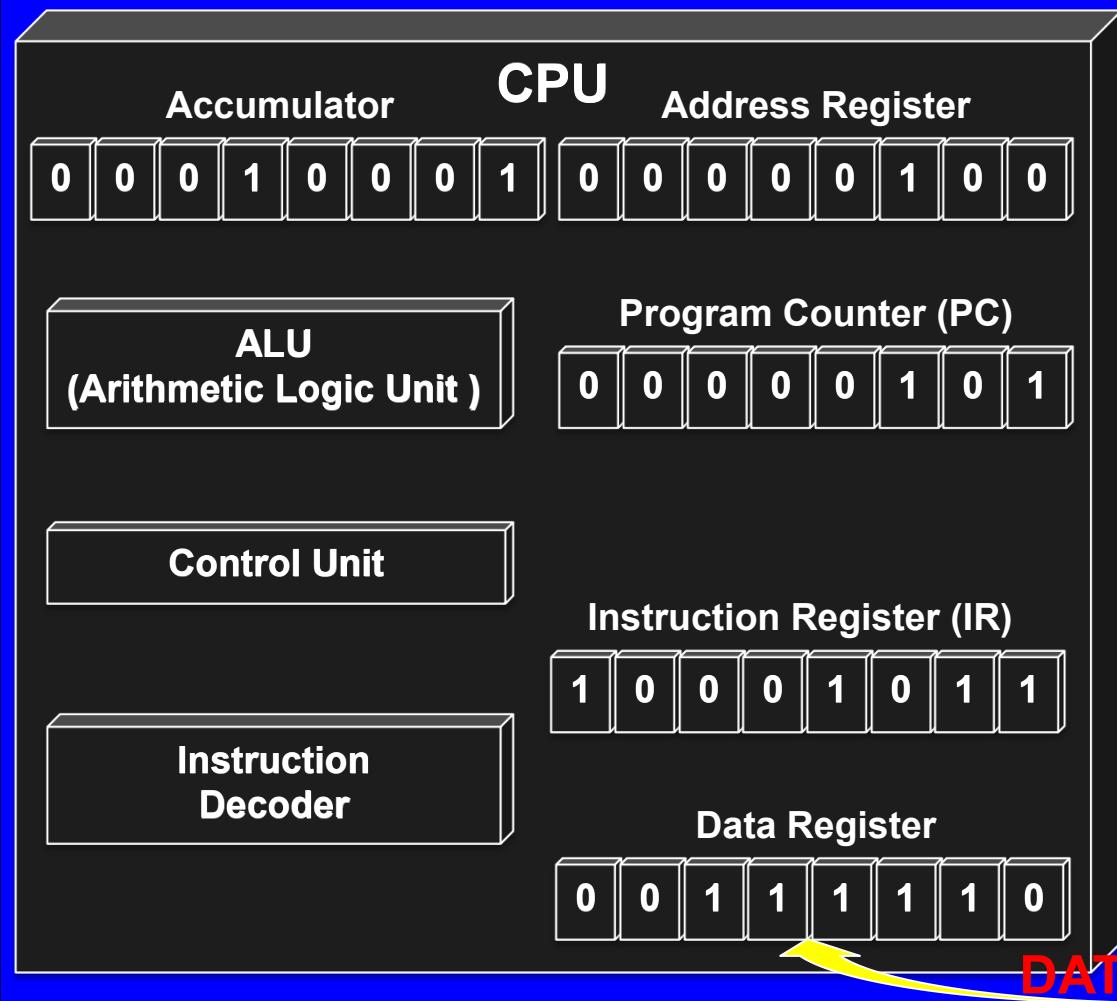
QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của thanh ghi địa chỉ được đặt lên bus địa chỉ.



QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của ô nhớ được chọn truyền tới thanh ghi dữ liệu (Data Register).

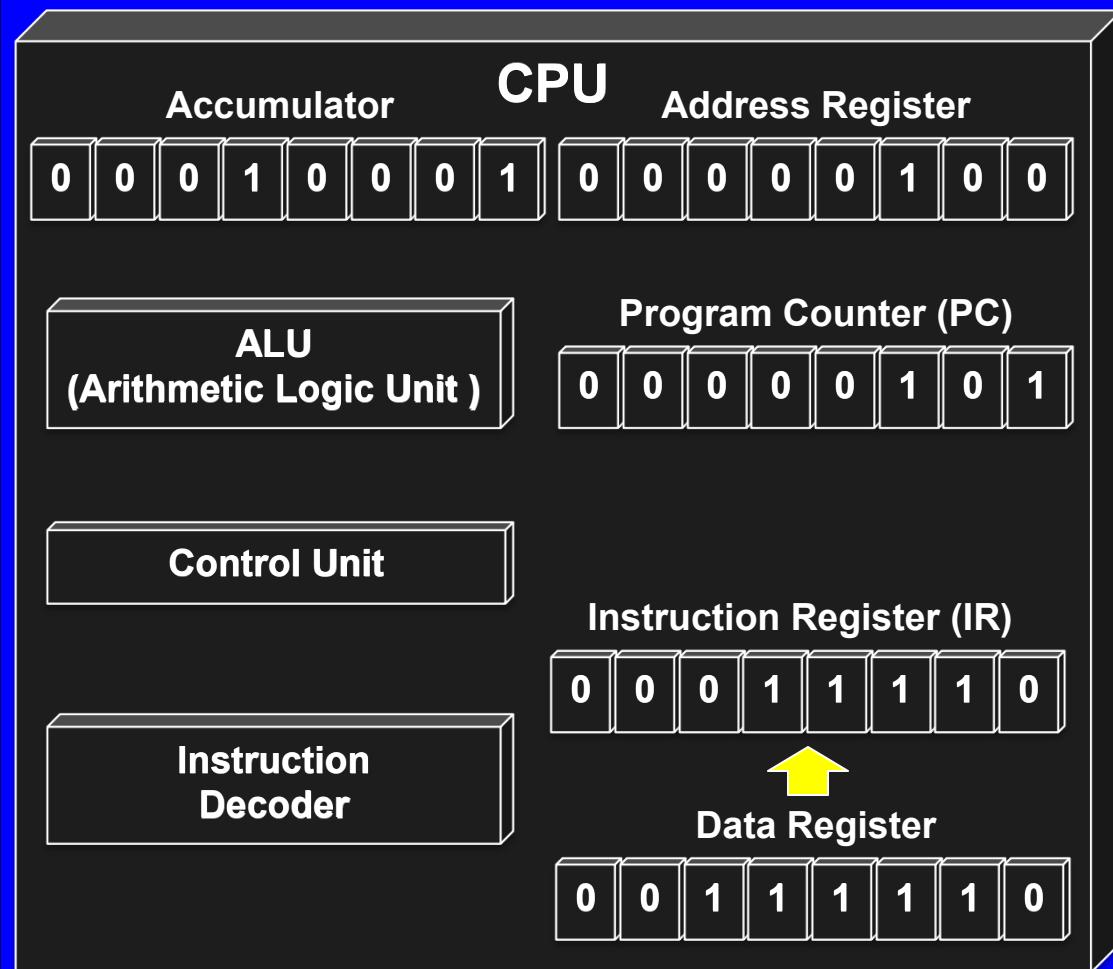


CHU KỲ TÌM NẠP LỆNH
HALT

Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	56

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Nội dung của thanh ghi dữ liệu được chuyển sang IR.

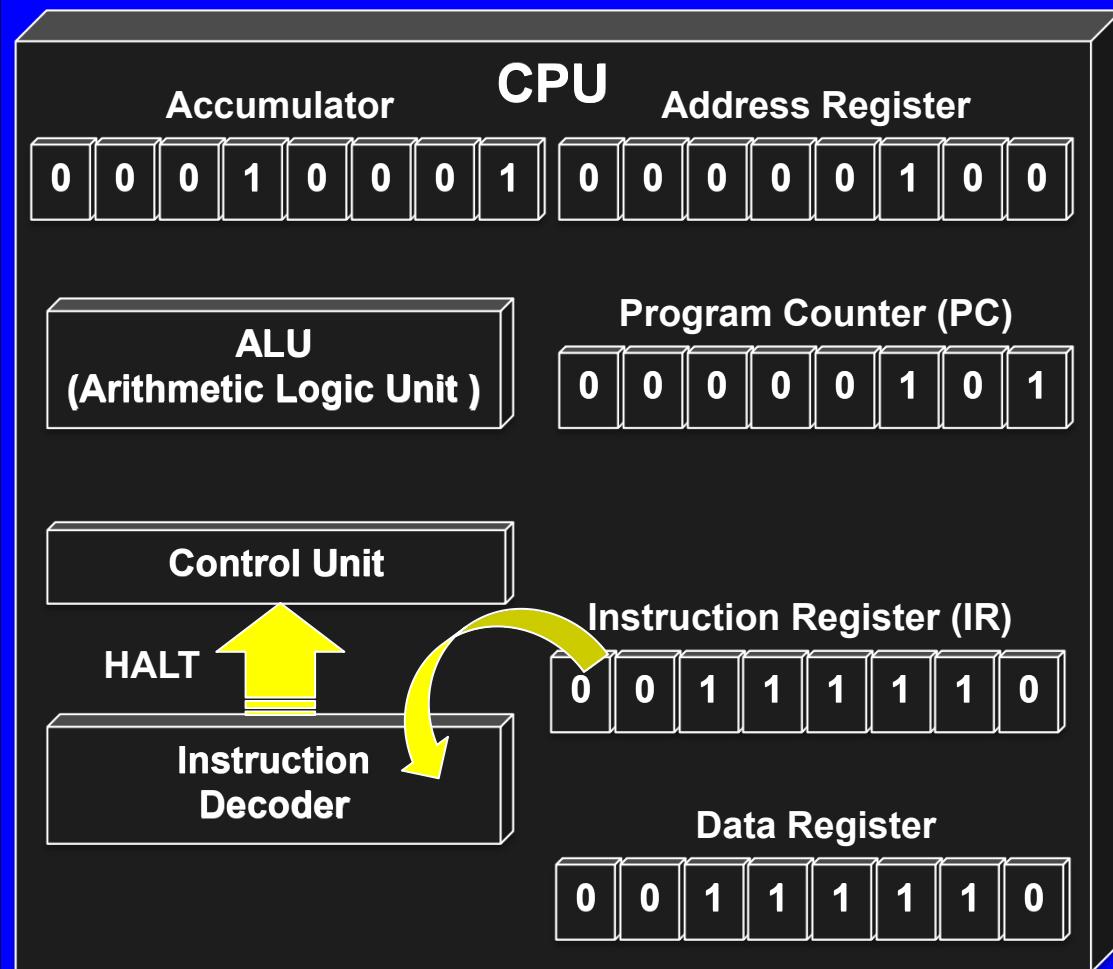


CHU KỲ TÌM NẠP LỆNH
HALT

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	57

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- Nội dung của IR được giải mã bởi bộ giải mã lệnh (Instruction Decoder). Bộ giải mã lệnh tác động đến đơn vị điều khiển (CU) để tạo ra tín hiệu điều khiển cần thiết.

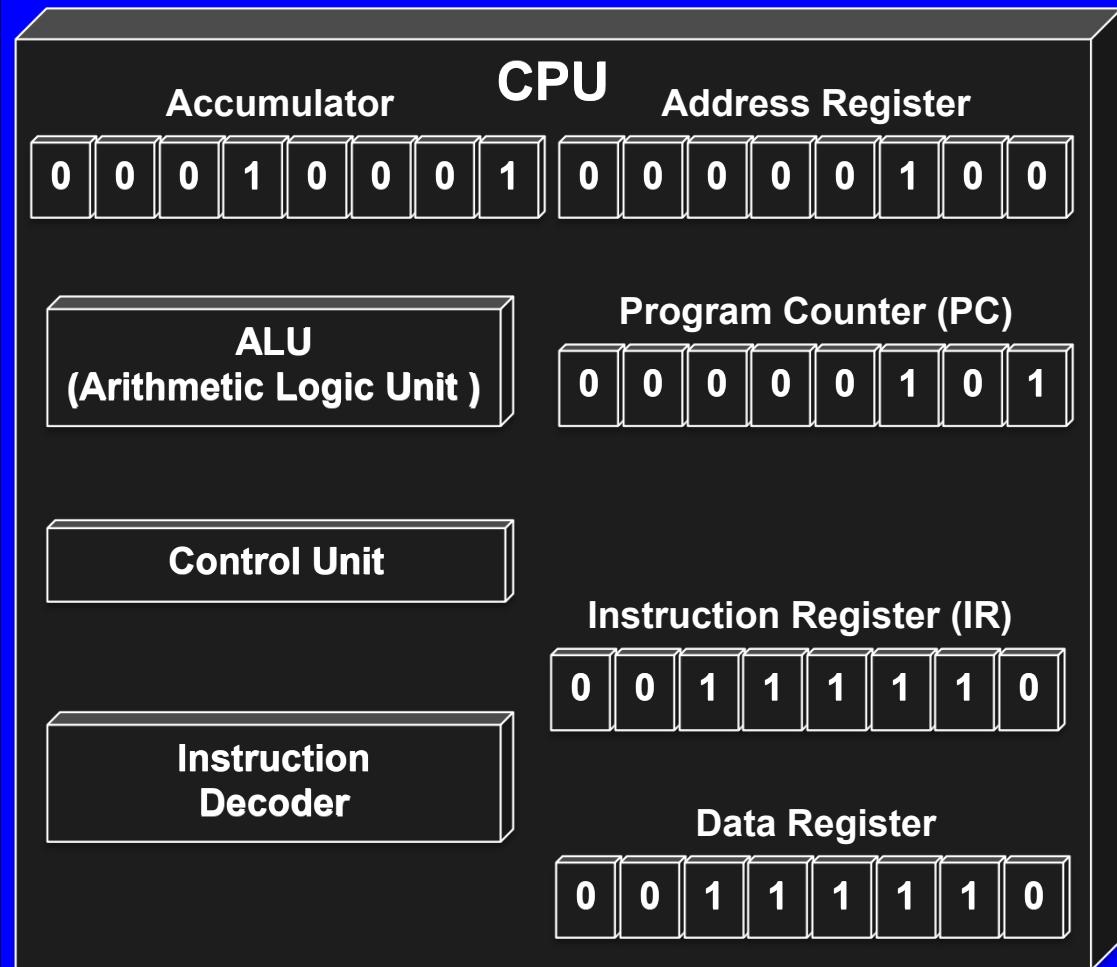


**CHU KỲ TÌM NẠP LỆNH
HALT**

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	58

QUÁ TRÌNH THỰC THI CHƯƠNG TRÌNH CỦA CPU

- ✓ Việc thực thi của lệnh HALT rất đơn giản. Đơn vị điều khiển sẽ dừng việc tạo ra các tín hiệu điều khiển, khi đó toàn bộ hệ thống sẽ bị dừng.



**CHU KỲ THỰC THI LỆNH
HALT**

MEMORY		
Address	Data	ASM Code
0000 0000	1000 0110	LDA
0000 0001	0000 0111	7
0000 0010	1000 1011	ADD
0000 0011	0000 1010	10
0000 0100	0011 1110	HLT
-----	-----	59

BỘ NHỚ BÁN DẪN

➤ **Bộ nhớ bán dẫn trong hệ vi xử lý gồm:**

Bộ nhớ chỉ đọc (ROM: Read Only Memory):

- Thông tin trong ROM không bị mất ngay cả khi nguồn điện cung cấp không còn
- Chỉ cho phép đọc thông tin ra từ ROM
- Lưu giữ chương trình điều khiển hoạt động của hệ thống

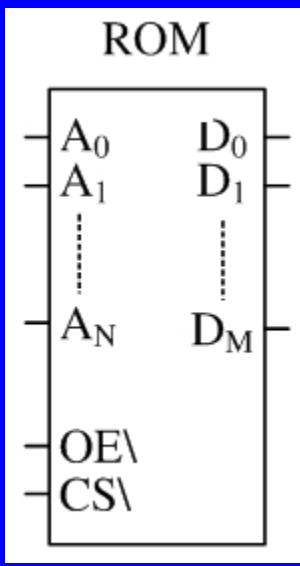
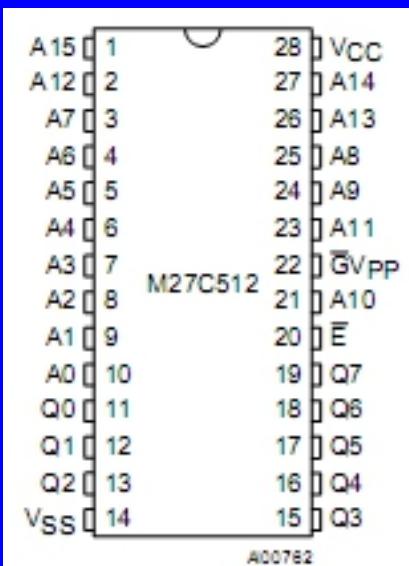
Bộ nhớ truy xuất ngẫu nhiên (RAM: Random Access Memory):

- Thông tin trong RAM sẽ bị mất ngay khi nguồn điện cung cấp không còn
- Cho phép ghi thông tin vào RAM và đọc thông tin ra từ RAM
- Lưu giữ dữ liệu, một phần chương trình điều khiển hệ thống, các ứng dụng và kết quả tính toán.

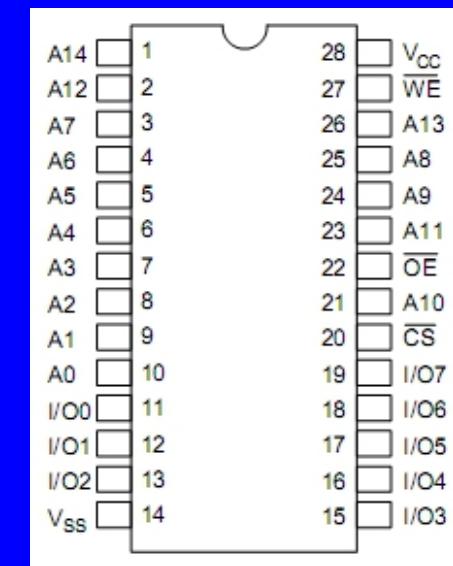
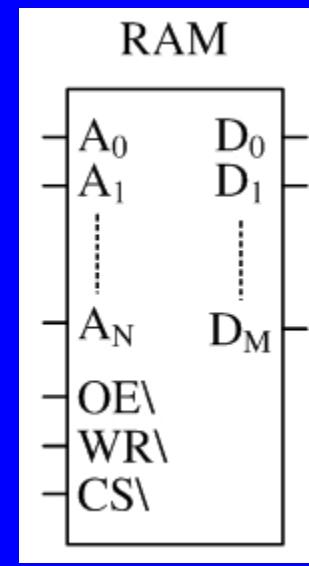
BỘ NHỚ BÁN DẪN

> Sơ lược về cấu trúc và phân loại ROM - RAM:

ROM



RAM

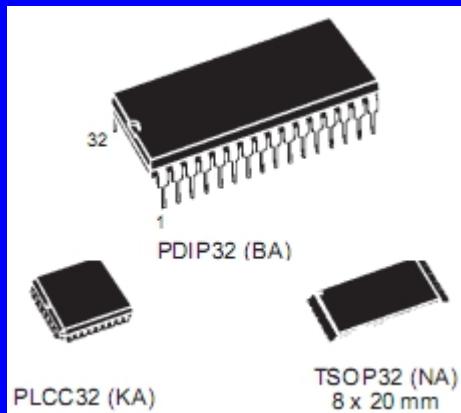
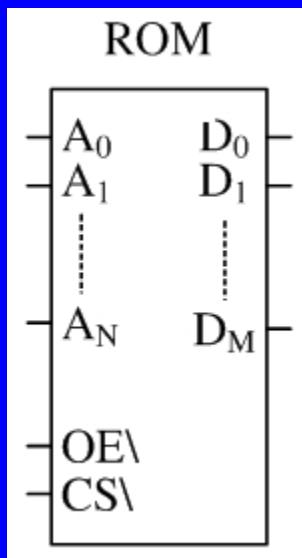
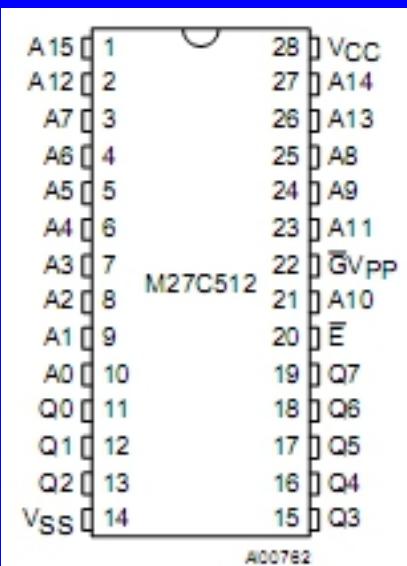


- ✓ $A_0 - A_N$: các chân địa chỉ (Address - N: số chân địa chỉ)
- ✓ $D_0 - D_M$: các chân dữ liệu (Data - M: số chân dữ liệu)
- ✓ OE: ngõ vào cho phép xuất (Output Enable)
- ✓ CS: ngõ vào cho phép IC hoạt động (Chip Select)
- ✓ WR: ngõ vào cho phép ghi (Write) – **chỉ có ở RAM.**

BỘ NHỚ BÁN DẪN

➤ Sơ lược về cấu trúc và phân loại ROM - RAM:

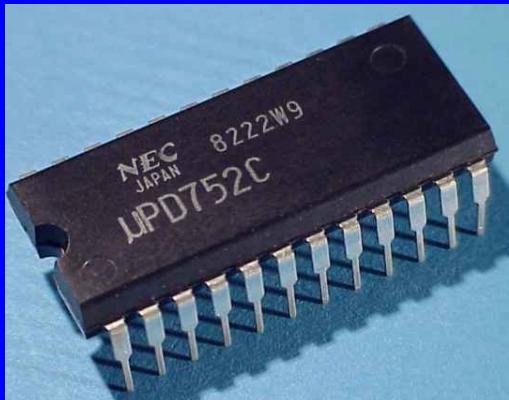
ROM



- ✓ **MROM (Mask ROM)**: ROM măt nạ
- ✓ **PROM (Programmable ROM)**: ROM lập trình được
- ✓ **EPROM (Eraseable PROM)**: ROM lập trình và xóa được
 - **UV-EPROM (Ultra Violet EPROM)**: ROM xóa bằng tia cực tím
 - **EEPROM (Electric EEPROM)**: ROM lập trình và xóa bằng tín hiệu điện
 - **Flash ROM**: ROM lập trình và xóa bằng tín hiệu điện.

BỘ NHỚ BÁN DẪN

> Sơ lược về cấu trúc và phân loại ROM - RAM:
MROM



PROM

UV-EPROM



EEPROM



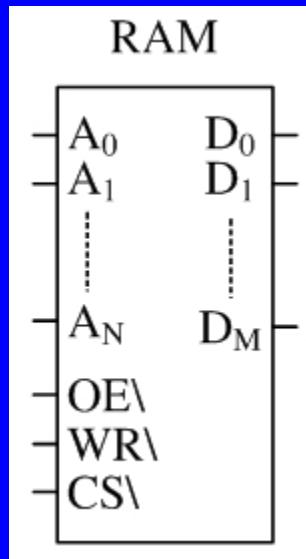
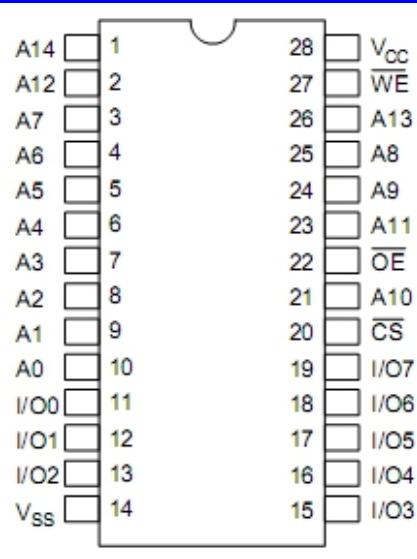
FLASH ROM



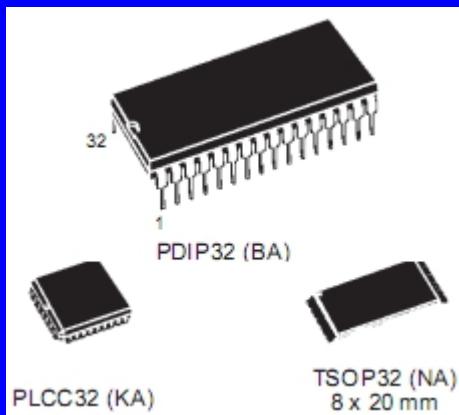
BỘ NHỚ BÁN DẪN

➤ Sơ lược về cấu trúc và phân loại ROM - RAM:

RAM



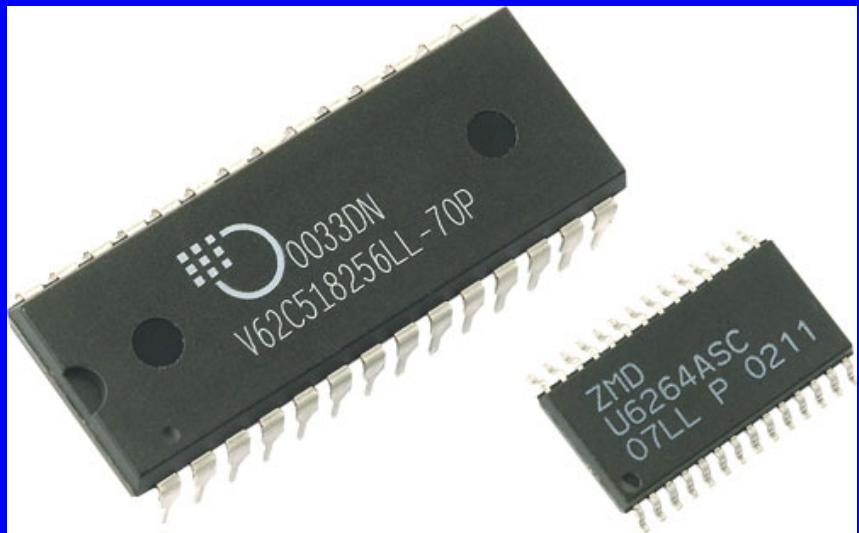
- ✓ **SRAM (Static RAM)**: RAM tĩnh
- ✓ **DRAM (Dynamic RAM)**: RAM động.



BỘ NHỚ BÁN DẪN

➤ Sơ lược về cấu trúc và phân loại ROM - RAM:
DRAM

SRAM



BỘ NHỚ BÁN DẪN

➤ Dung lượng của bộ nhớ:

- Dung lượng của bộ nhớ hay còn gọi là kích thước của bộ nhớ nói lên khả năng lưu trữ thông tin nhiều hay ít của bộ nhớ
- Đơn vị cơ bản đo lượng thông tin trong hệ vi xử lý là **BIT**. Đây là thuật ngữ chỉ phần nhỏ nhất của bộ nhớ máy tính có thể lưu trữ một trong hai trạng thái thông tin là 0 hoặc 1
- Một dãy 8 BIT được gọi là 1 **BYTE**. Thuật ngữ **BYTE** để chỉ một đơn vị lưu trữ dữ liệu trên hệ vi xử lý. Ngoài ra, người ta còn dùng các đơn vị bội của byte như sau:

Kilobyte (KB) $1 \text{ (KB)} = 2^{10} \text{ (B)} = 1,024 \text{ (B)}$

Megabyte (MB) $1 \text{ (MB)} = 2^{10} \text{ (KB)} = 1,024 \text{ (KB)}$

Gigabyte (GB) $1 \text{ (GB)} = 2^{10} \text{ (MB)} = 1,024 \text{ (MB)}$

Terabyte (TB) $1 \text{ (TB)} = 2^{10} \text{ (GB)} = 1,024 \text{ (GB)}$

Petabyte (PB) $1 \text{ (PB)} = 2^{10} \text{ (TB)} = 1,024 \text{ (TB)}$.

BỘ NHỚ BÁN DẪN

> Cách xác định dung lượng bộ nhớ bán dẫn 8 bit dùng cho vi xử lý:

Căn cứ vào số chân địa chỉ:

$$\text{DUNG LƯỢNG} = 2^N \times M \text{ (bit)}$$

- N: số chân (bit) địa chỉ
- M: số chân (bit) dữ liệu

Ví dụ: Bộ nhớ bán dẫn 8 bit có 15 đường địa chỉ.
Cho biết dung lượng của bộ nhớ là bao nhiêu?

Giải

Số chân (bit) địa chỉ: 15 chân

$$\rightarrow N = 15$$

Số chân (bit) dữ liệu: 8 chân

$$\rightarrow M = 8$$

$$\text{Dung lượng} = 2^{15} \times 8 \text{ (bit)} = 32.768 \times 8 \text{ (bit)} = 32 \text{ (KB)}.$$

SMJ27C256	
V _{PP}	1
A ₁₂	2
A ₇	3
A ₆	4
A ₅	5
A ₄	6
A ₃	7
A ₂	8
A ₁	9
A ₀	10
DQ ₀	11
DQ ₁	12
DQ ₂	13
GND	14
	28
	27
	26
	25
	24
	23
	22
	21
	20
	19
	18
	17
	16
	15
	V _{CC}
	A ₁₄
	A ₁₃
	A ₈
	A ₉
	A ₁₁
	G
	A ₁₀
	E
	DQ ₇
	DQ ₆
	DQ ₅
	DQ ₄
	DQ ₃

BỘ NHỚ BÁN DẪN

> **Cách xác định dung lượng bộ nhớ bán dẫn 8 bit dùng cho vi xử lý:**

Căn cứ vào mã số:

MÃ SỐ = XXYYYY

- XX: xác định loại bộ nhớ

✓ 27: UV-EPROM	28: EEPROM
✓ 61, 62: SRAM	40, 41: DRAM
- YYYY: xác định dung lượng.

DUNG LƯỢNG = YYYY (Kbit)

Ví dụ: Bộ nhớ bán dẫn 8 bit có mã số 27256.

Cho biết dung lượng của bộ nhớ là bao nhiêu?

Giải

Bộ nhớ thuộc loại UV-EPROM \rightarrow XX = 27

SMJ27C256	
V _{PP}	1
A12	2
A7	3
A6	4
A5	5
A4	6
A3	7
A2	8
A1	9
A0	10
DQ0	11
DQ1	12
DQ2	13
GND	14
	28
	27
	26
	25
	24
	23
	22
	21
	20
	19
	18
	17
	16
	15
	V _{CC}
	A14
	A13
	A8
	A9
	A11
	G
	A10
	E
	DQ7
	DQ6
	DQ5
	DQ4
	DQ3

BỘ NHỚ BÁN DẪN

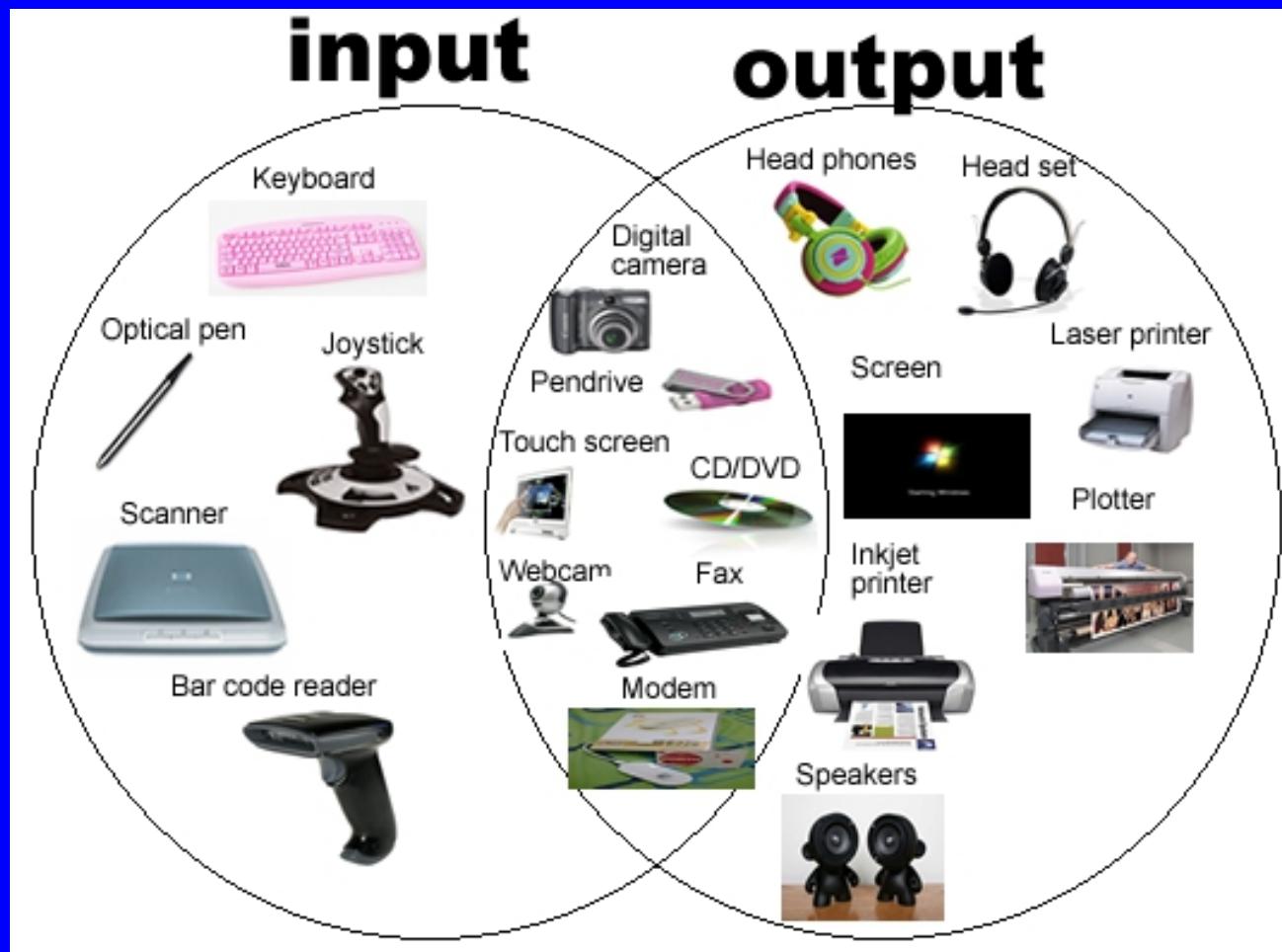
> Một số thiết bị lưu trữ khác dùng trong hệ vi xử lý:



THIẾT BỊ NGOẠI VI

➤ Phân loại thiết bị ngoại vi:

- Thiết bị nhập (Input)
- Thiết bị xuất (Output).



THIẾT BỊ NGOẠI VI

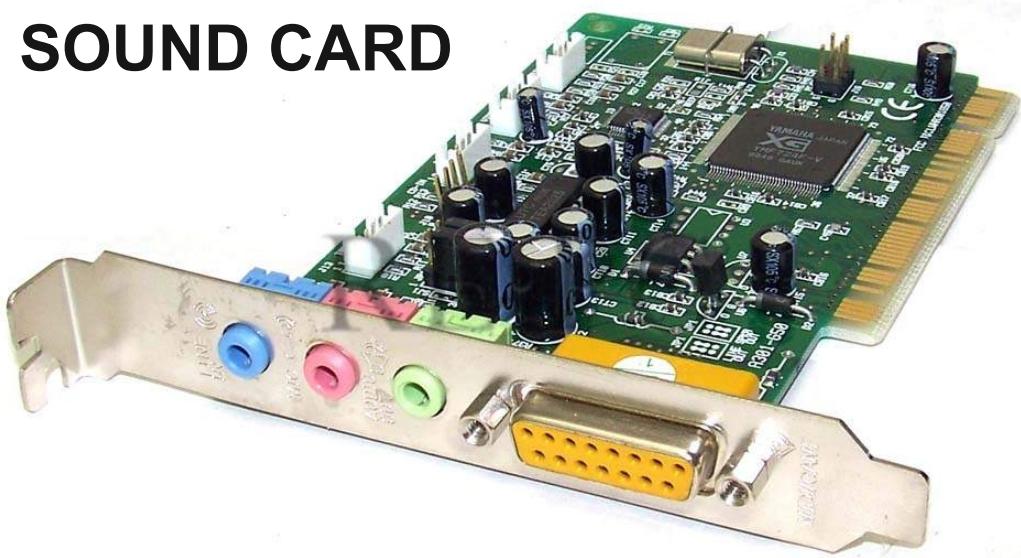
➤ **Mạch điện giao tiếp:**

- **Ghép nối tương thích các thiết bị ngoại vi vào hệ thống vi xử lý**
- **Giải mã địa chỉ cho thiết bị ngoại vi**
- **Cung cấp các tín hiệu cho bộ điều khiển ngoại vi**
- **Đồng bộ hóa luồng dữ liệu và giám sát tốc độ chuyển dữ liệu giữa thiết bị ngoại vi với CPU hoặc bộ nhớ.**

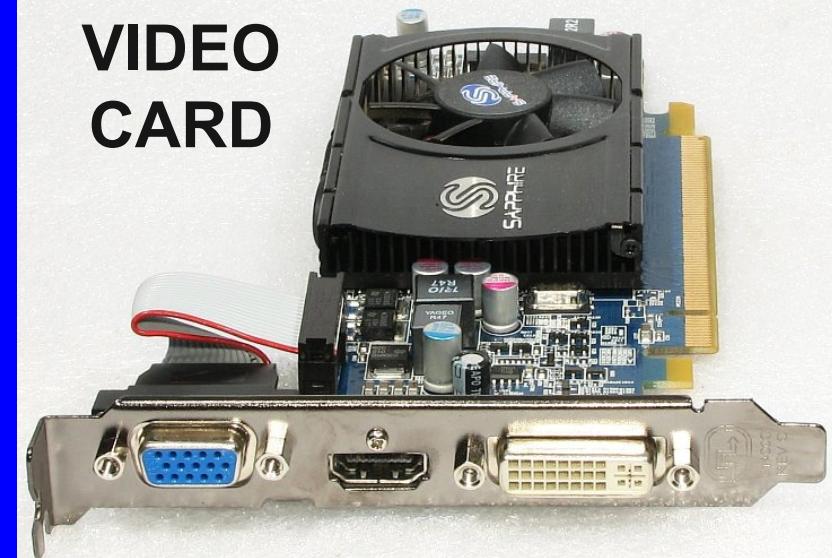
THIẾT BỊ NGOẠI VI

➤ Mạch điện giao tiếp:

SOUND CARD



VIDEO
CARD



DATA ACQUISITION
CARD



WIFI
CARD



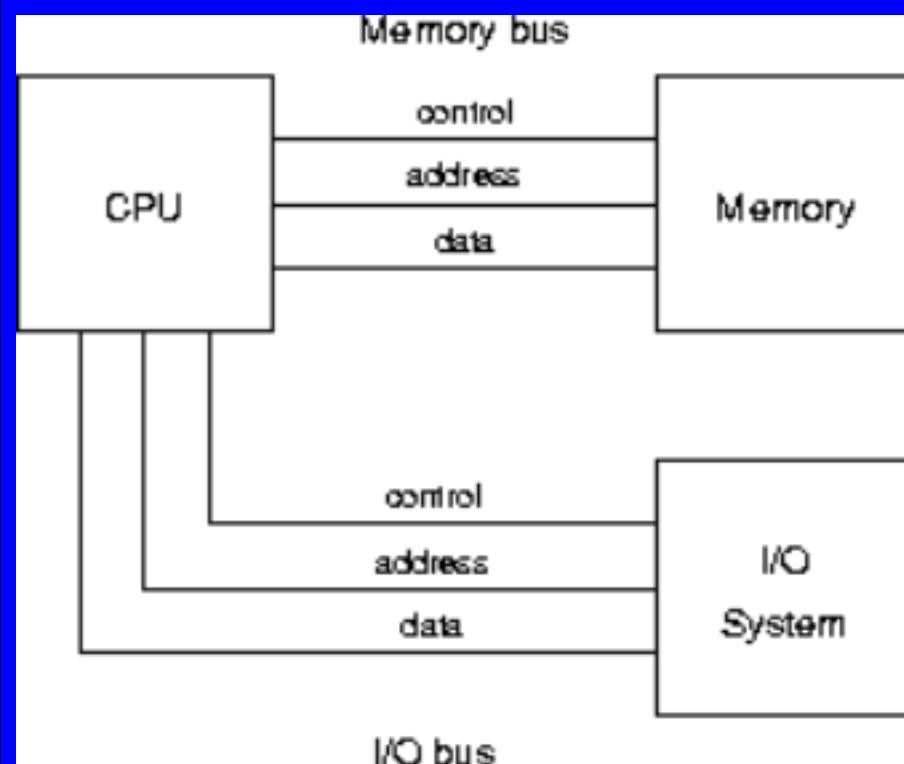
TUNER CARD



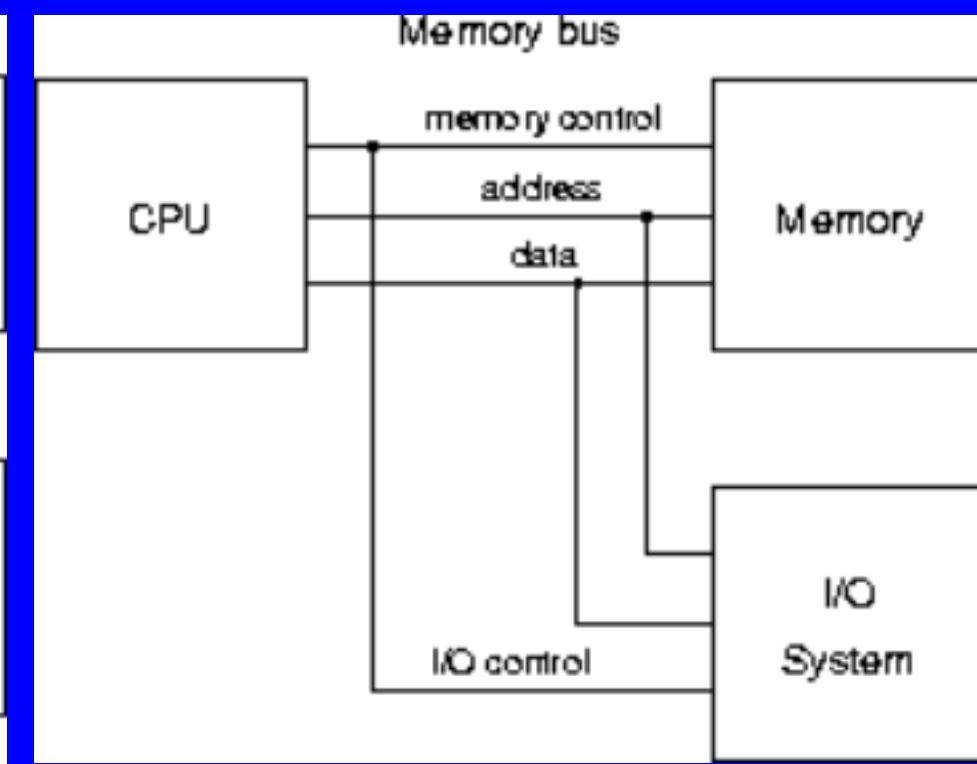
THIẾT BỊ NGOẠI VI

➤ Phương pháp giao tiếp ngoại vi:

- I/O trực tiếp hay I/O cách ly (Isolated I/O)
- I/O ánh xạ bộ nhớ (Memory mapped I/O).



Isolated I/O

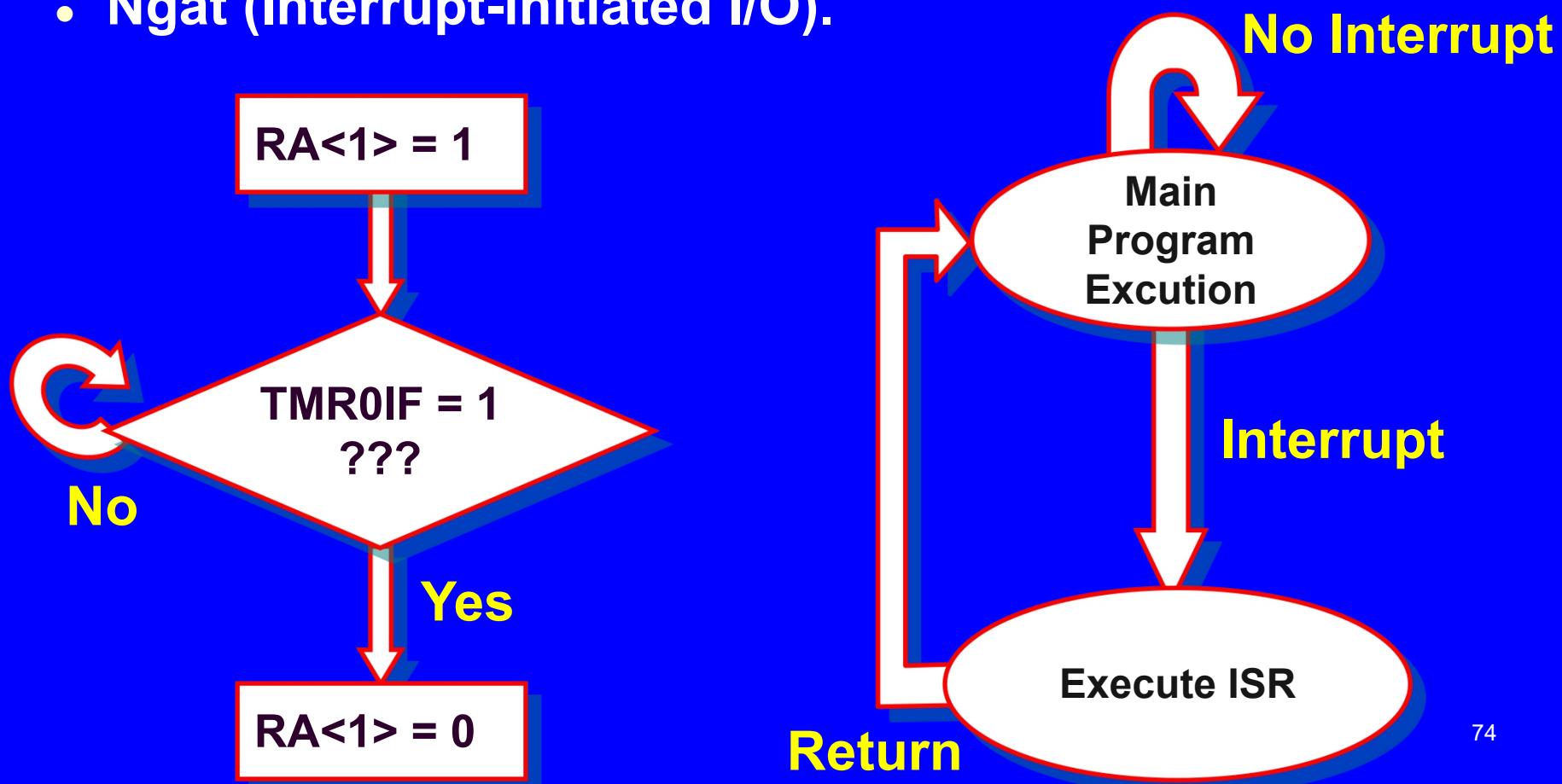


Memory mapped I/O

THIẾT BỊ NGOẠI VI

➤ Phương pháp điều khiển ngoại vi:

- Hỏi vòng (Polling I/O) hay còn gọi là I/O được điều khiển bằng chương trình (Program-controlled I/O)
- Ngắt (Interrupt-initiated I/O).

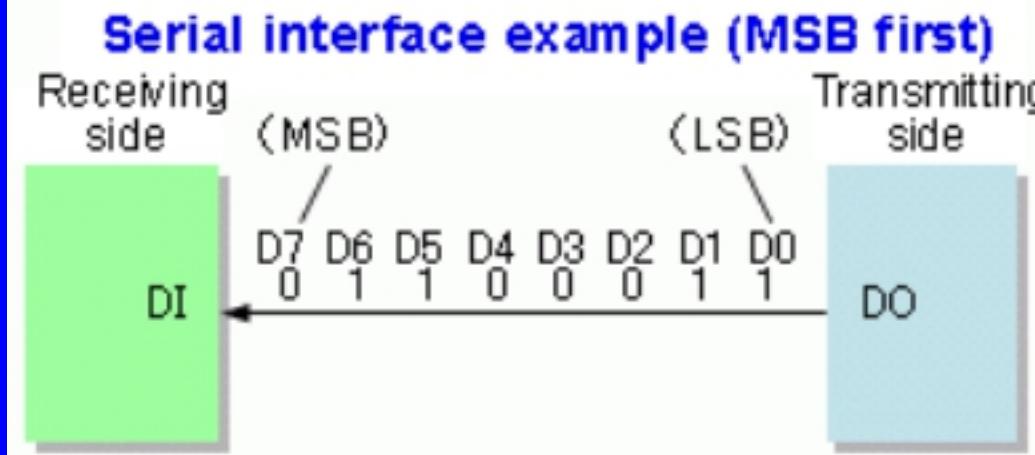
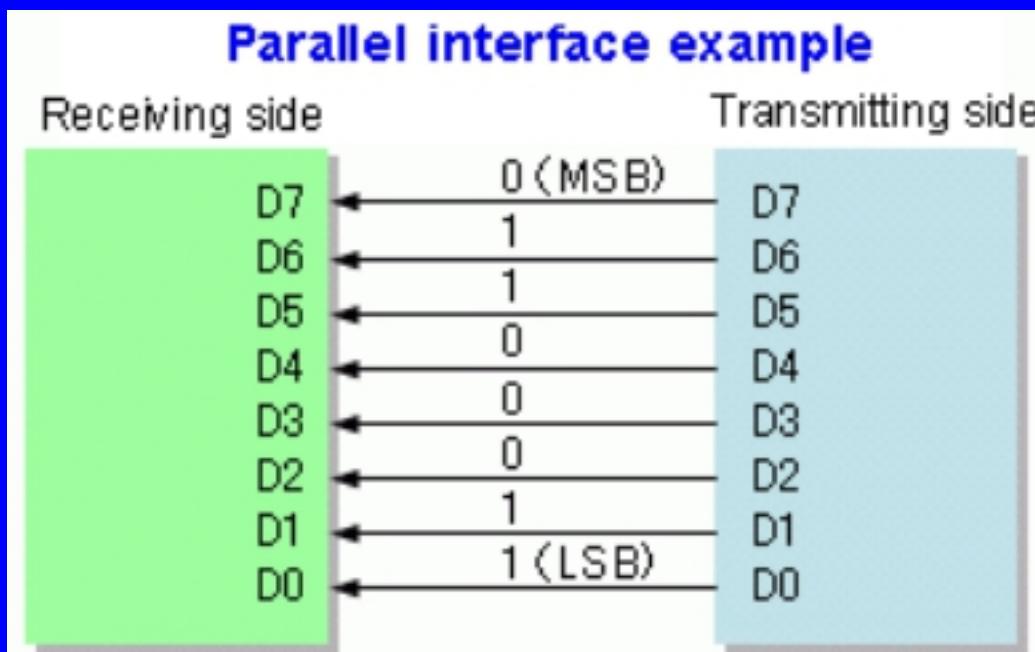


THIẾT BỊ NGOẠI VI

- Phương pháp truyền dữ liệu giữa CPU và thiết bị ngoại vi:
 - Truyền song song (Parallel Transfer)
 - Truyền nối tiếp (Serial Transfer):
 - Nối tiếp bất đồng bộ (Asynchronous Serial)
 - Nối tiếp đồng bộ (Synchronous Serial).

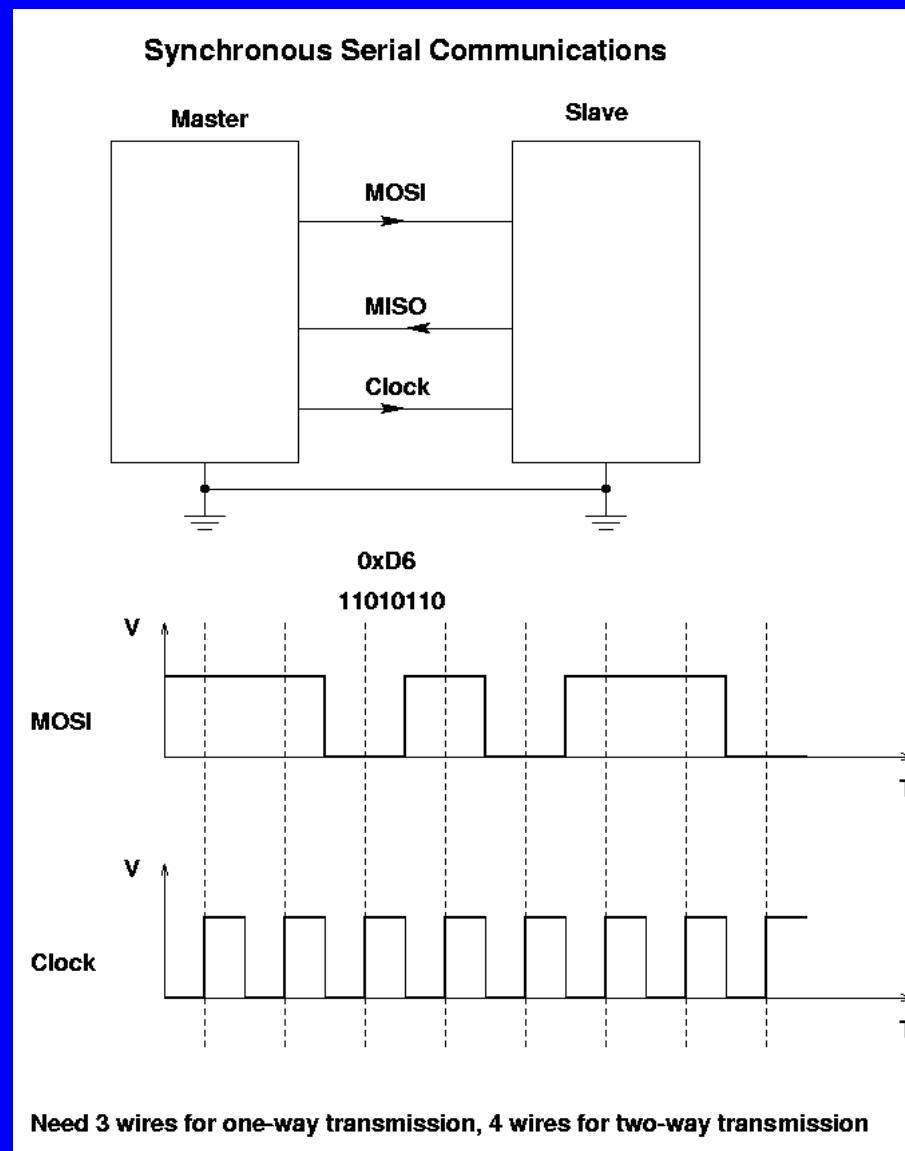
THIẾT BỊ NGOẠI VI

➤ Phương pháp truyền dữ liệu giữa CPU và thiết bị ngoại vi:



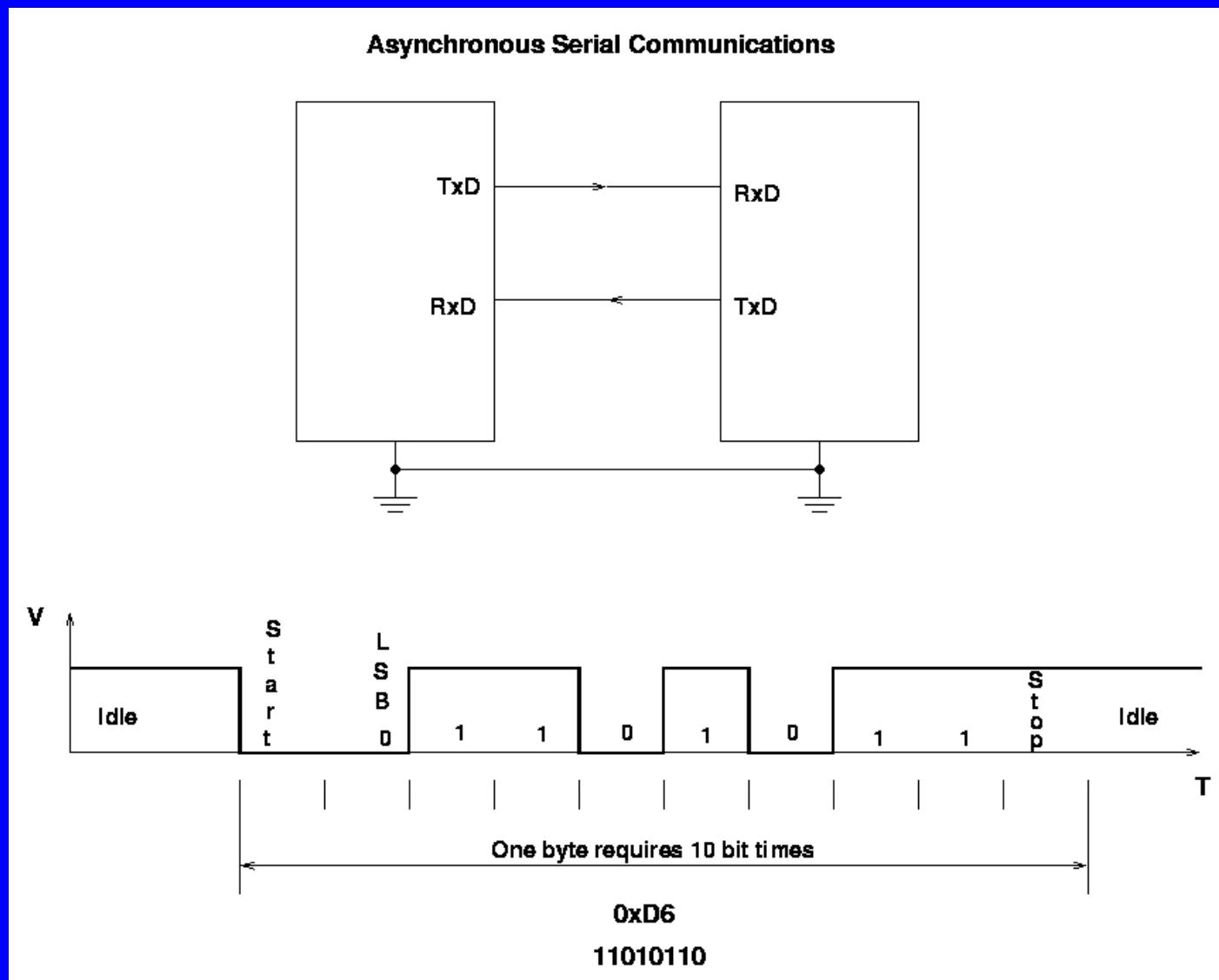
THIẾT BỊ NGOẠI VI

➤ Phương pháp truyền dữ liệu giữa CPU và thiết bị ngoại vi:



THIẾT BỊ NGOẠI VI

➤ Phương pháp truyền dữ liệu giữa CPU và thiết bị ngoại vi:



VI XỬ LÝ – VI ĐIỀU KHIỂN

➤ **Cấu trúc phần cứng (Hardware architecture):**

Vi xử lý (Microprocessor):

- Đơn vị xử lý trung tâm (CPU)

Vi điều khiển (Microcontroller):

- Đơn vị xử lý trung tâm (CPU)
- Bộ nhớ chương trình (ROM)
- Bộ nhớ dữ liệu (RAM)
- Mạch giao tiếp nối tiếp
- Mạch giao tiếp song song
- Mạch điều khiển ngắt
- Các mạch điều khiển khác.

VI XỬ LÝ – VI ĐIỀU KHIỂN

➤ Các ứng dụng (Applications):

Vi xử lý (Microprocessor):

- Ứng dụng lớn, tính toán phức tạp

Vi điều khiển (Microcontroller):

- Ứng dụng nhỏ, tính toán đơn giản.

VI XỬ LÝ – VI ĐIỀU KHIỂN

➤ Các đặc trưng của tập lệnh (Instruction set feature):

Vì xử lý (Microprocessor):

- Có nhiều kiểu định địa chỉ
- Độ dài từ dữ liệu xử lý: Byte, Word, Double word,...

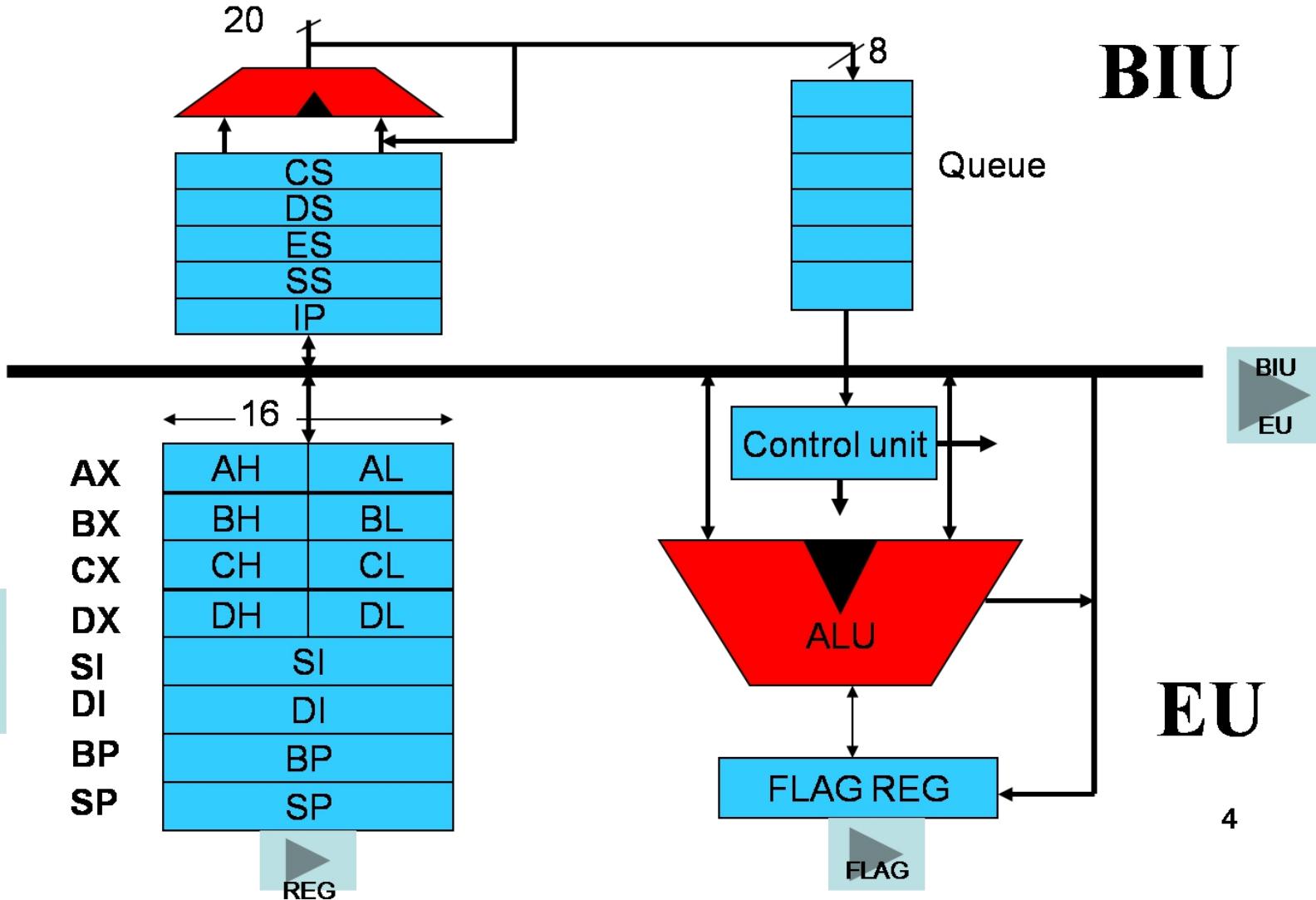
Vì điều khiển (Microcontroller):

- Có ít kiểu định địa chỉ
- Độ dài từ dữ liệu xử lý: Bit, Byte.

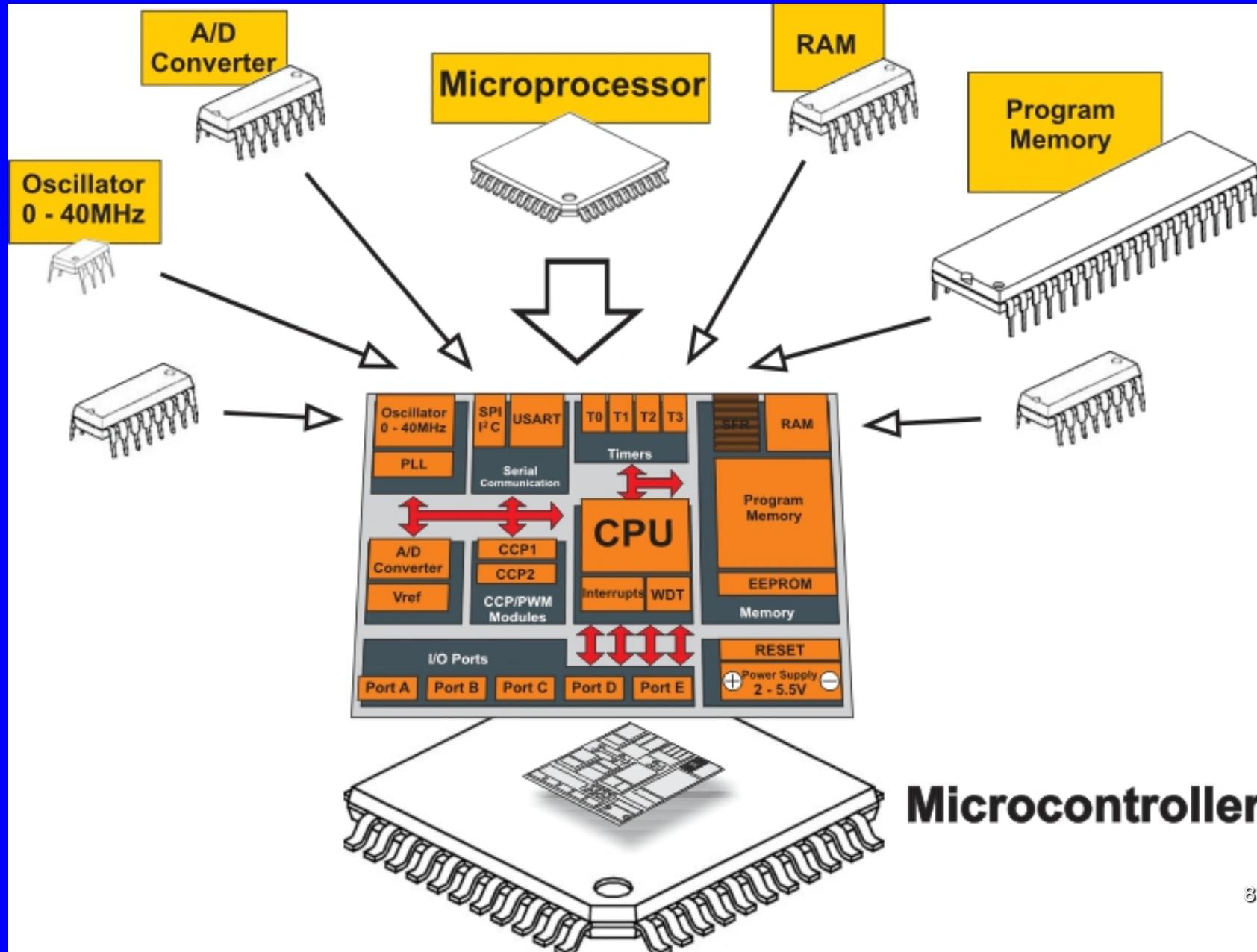
KIẾN TRÚC CỦA MỘT VI XỬ LÝ (8086)



8086 Architecture

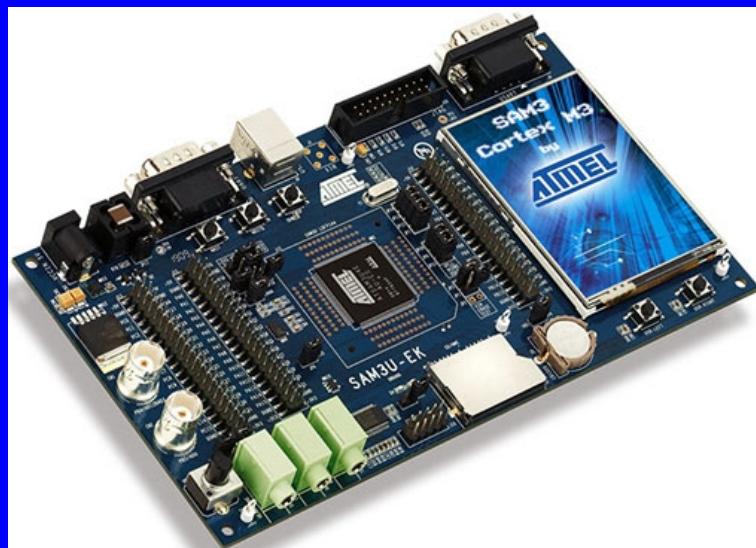


KIẾN TRÚC CỦA MỘT VI ĐIỀU KHIỂN



VI XỬ LÝ – VI ĐIỀU KHIỂN

➤ Một số loại vi điều khiển:



➤ **Các loại vi điều khiển thông dụng:**

- 68xxx của Motorola
- 80xxx, AVR, ARM của Intel
- Z8xx của Zilog
- PIC16xxx, PIC18xxx,... của Microchip.

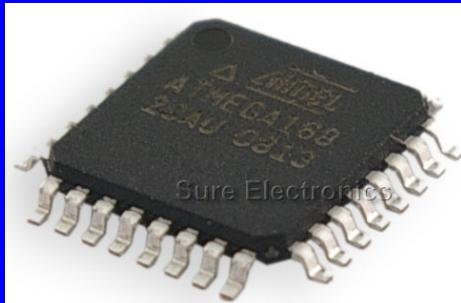
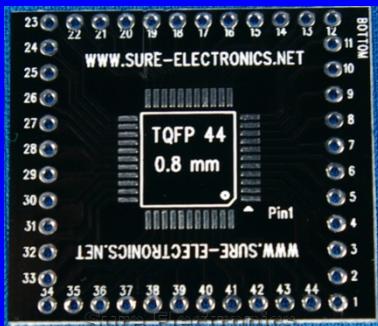
➤ **Tiêu chuẩn cơ bản khi chọn bộ vi điều khiển:**

- Đáp ứng yêu cầu tính toán một cách hiệu quả và kinh tế.
- Có sẵn các công cụ phát triển phần mềm (chương trình mô phỏng, trình biên dịch, trình hợp dịch và gỡ rối)
- Khả năng đáp ứng về số lượng ở hiện tại cũng như ở tương lai.

LỰA CHỌN BỘ VI ĐIỀU KHIỂN KHI THIẾT KẾ

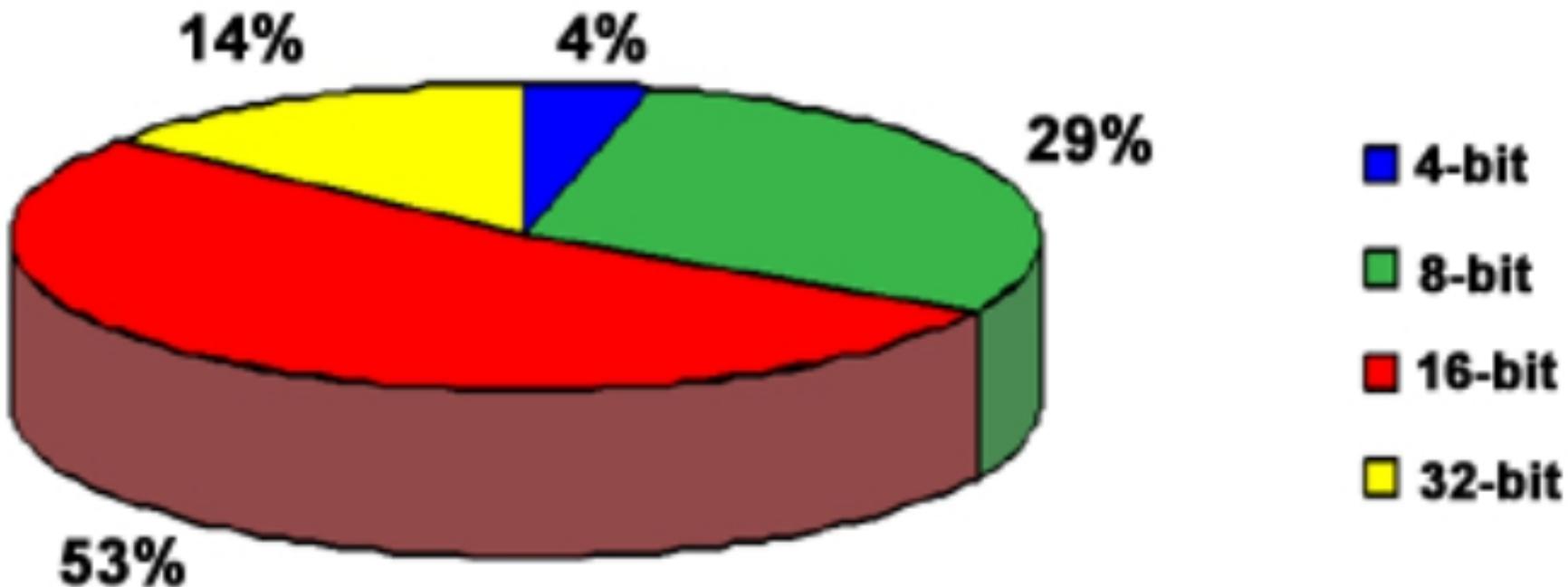
➤ **Một số tham số kỹ thuật cần chú ý:**

- Tốc độ
- Kiểu IC: DIP, QFP,... (DIP: vỏ dạng hai hàng chân, QFP: vỏ vuông dẹt)
- Công suất tiêu thụ
- Dung lượng ROM và RAM tích hợp sẵn trên chip
- Số chân vào/ra và bộ định thời trên chip
- Khả năng dễ dàng nâng cao hiệu suất hoặc giảm công suất tiêu thụ
- Giá thành trên một đơn vị khi mua số lượng lớn.



LỰA CHỌN BỘ VI ĐIỀU KHIỂN KHI THIẾT KẾ

- Thị phần của các loại vi điều khiển hiện nay.

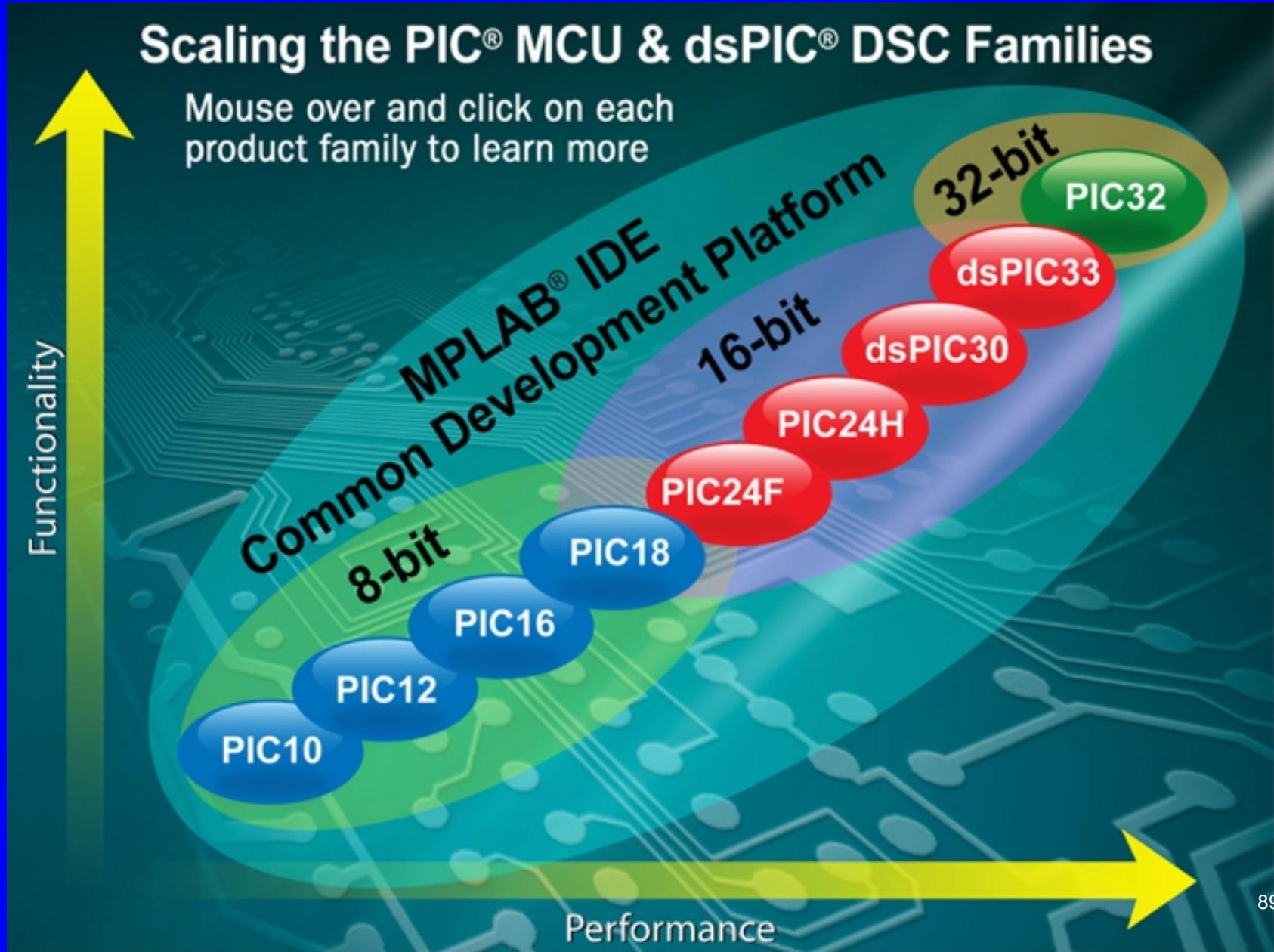


Nguồn: www.eetimes.com

GIỚI THIỆU VỀ VI ĐIỀU KHIỂN PIC

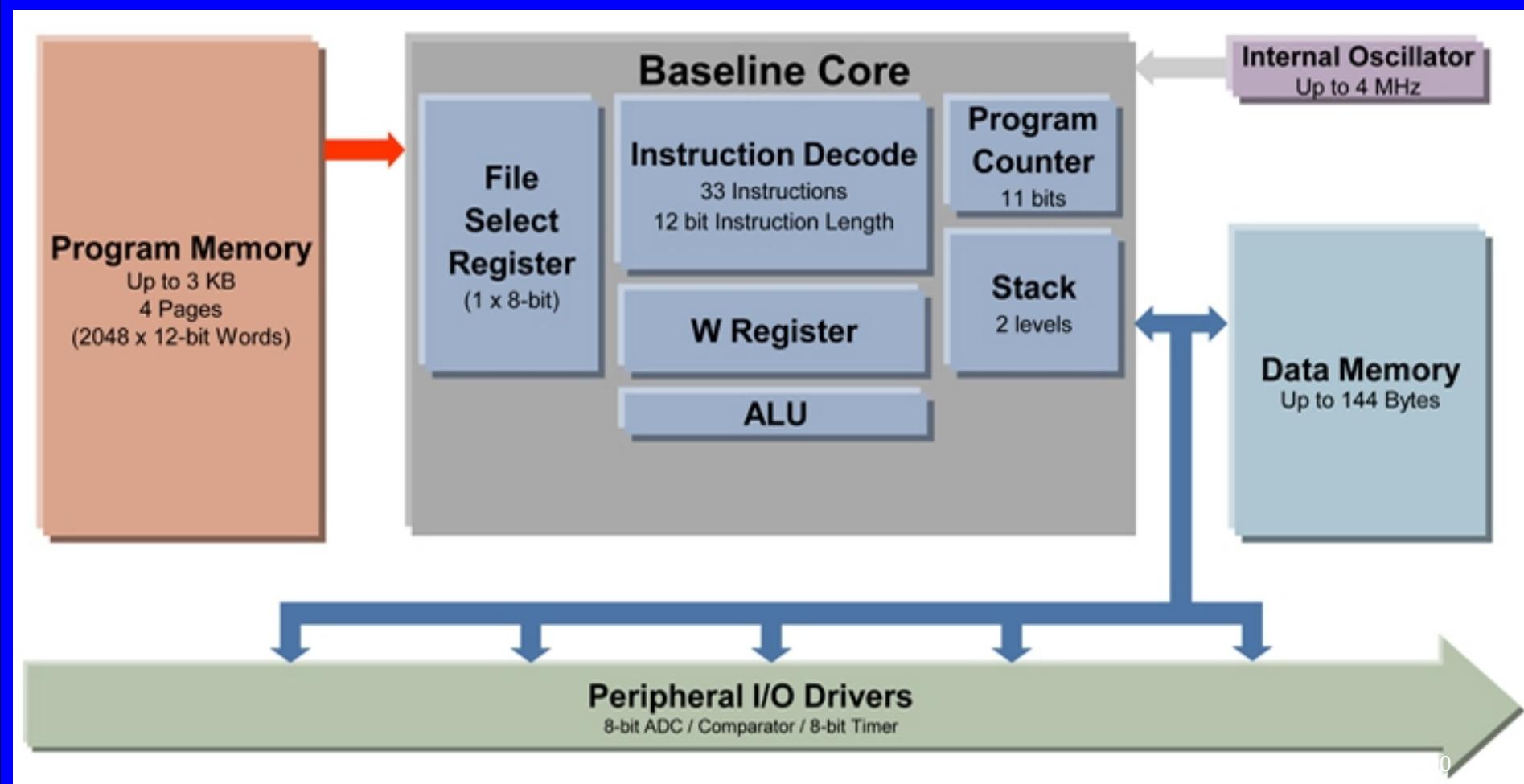
- PIC (**P**rogrammable **I**ntelligent **C**omputer, **P**eripheral **I**nterface **C**ontroller) nghĩa là “Máy tính thông minh khả trìn” xuất phát từ vi điều khiển PIC đầu tiên là PIC1650, do hãng General Instrument đặt tên
- Sau đó hãng Microchip tiếp tục phát triển loại PIC này và cho ra đời gần 100 loại PIC đến nay
- Các dòng PIC hiện nay của hãng Microchip:
 - 8-bit PIC® MCU (Baseline, Mid-Range, Enhanced Mid-Range, PIC18): PIC10, PIC12, PIC16, PIC18
 - 16-bit PIC® MCU: PIC24F, PIC24H, PIC24E
 - 16-bit dsPIC® DSC: dsPIC30F, dsPIC33F, dsPIC33E
 - 32-bit PIC® MCU: PIC32.

GIỚI THIỆU VỀ VI ĐIỀU KHIỂN PIC



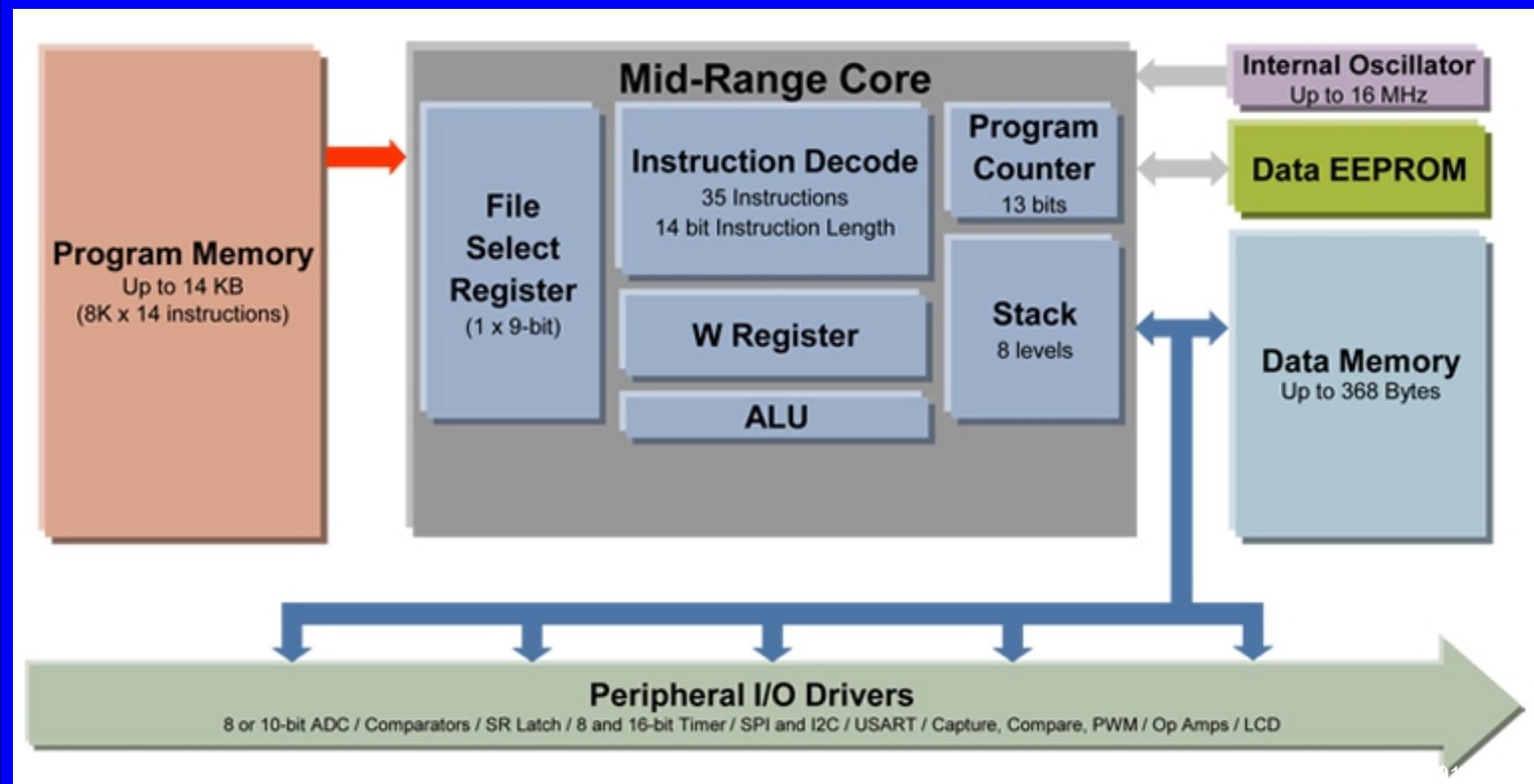
GIỚI THIỆU VỀ VI ĐIỀU KHIỂN PIC

- Kiến trúc 8-bit PIC® MCU (**Baseline**, Mid-Range, Enhanced Mid-Range, PIC18)



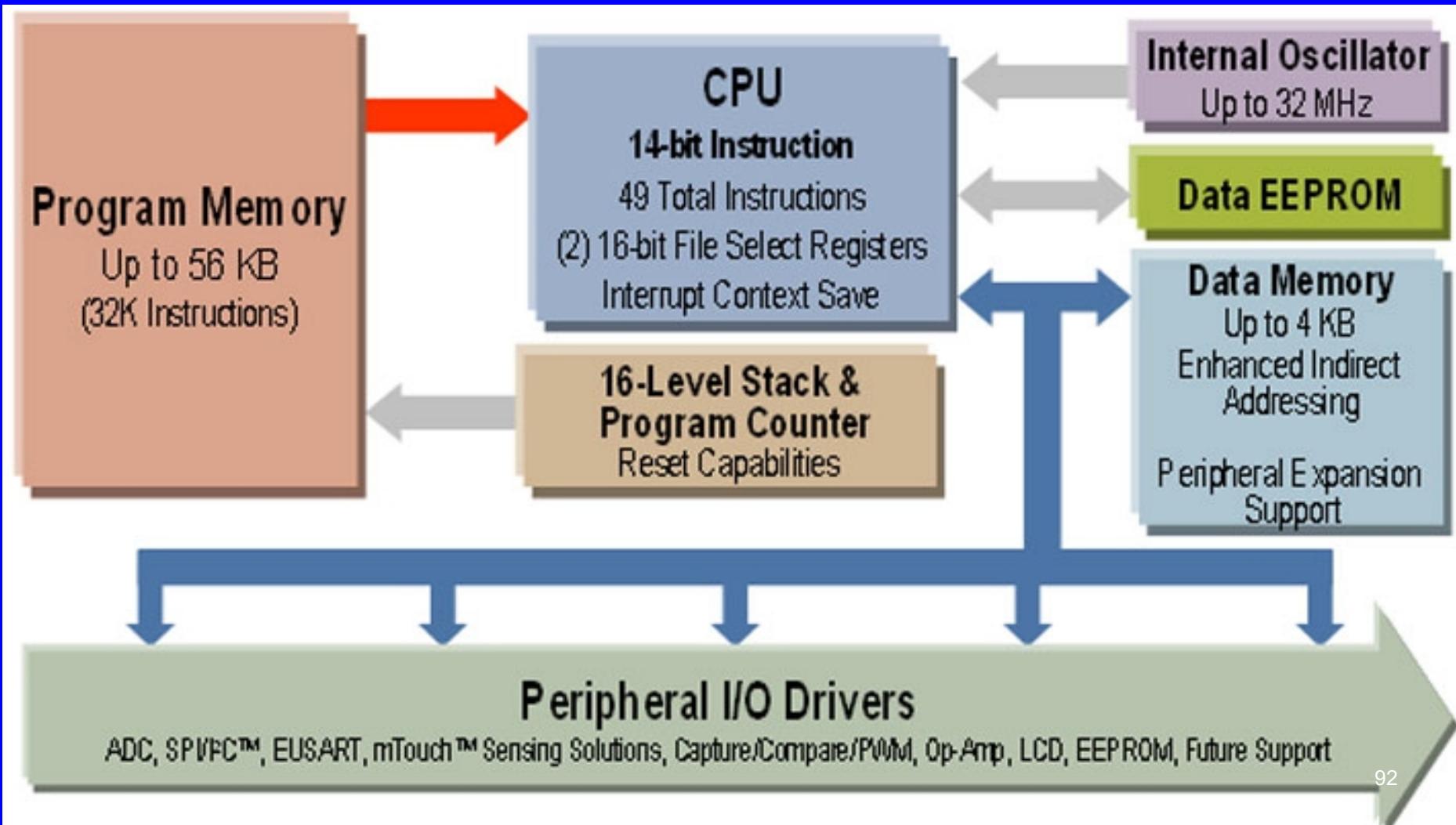
GIỚI THIỆU VỀ VI ĐIỀU KHIỂN PIC

- Kiến trúc 8-bit PIC® MCU (Baseline, Mid-Range, Enhanced Mid-Range, PIC18)



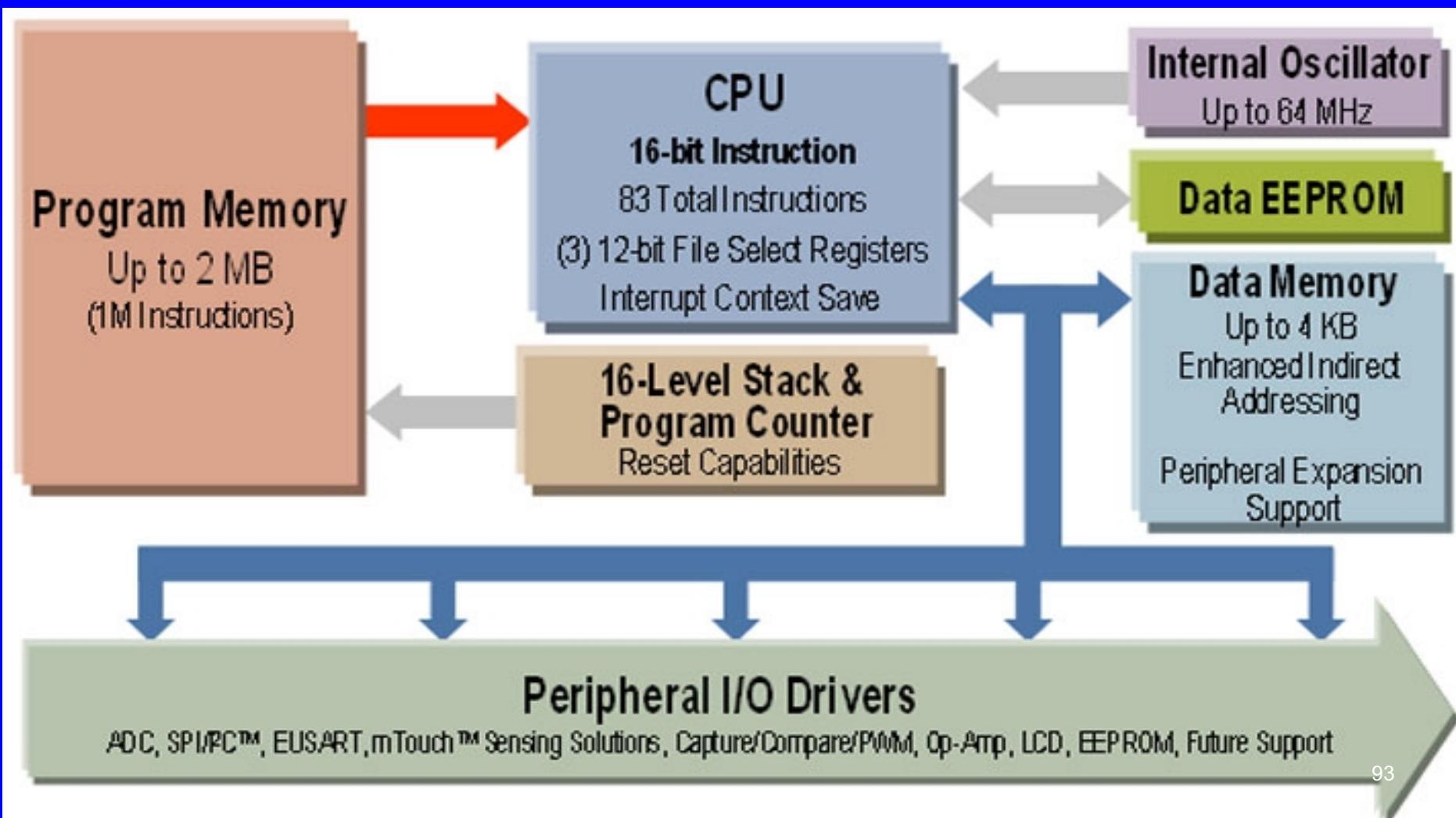
GIỚI THIỆU VỀ VI ĐIỀU KHIỂN PIC

- Kiến trúc 8-bit PIC® MCU (Baseline, Mid-Range, Enhanced Mid-Range, PIC18)



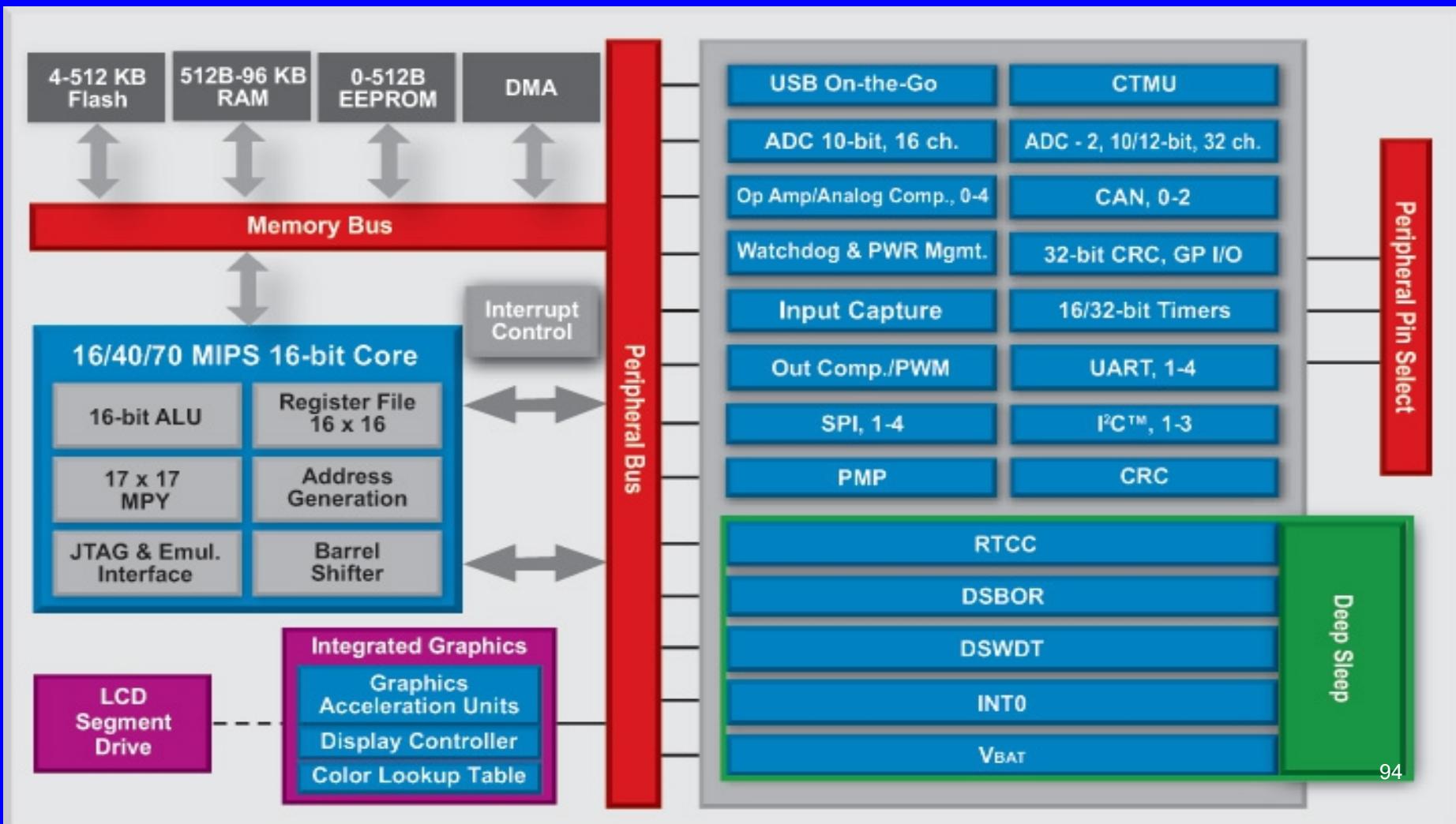
GIỚI THIỆU VỀ VI ĐIỀU KHIỂN PIC

- Kiến trúc 8-bit PIC® MCU (Baseline, Mid-Range, Enhanced Mid-Range, PIC18)



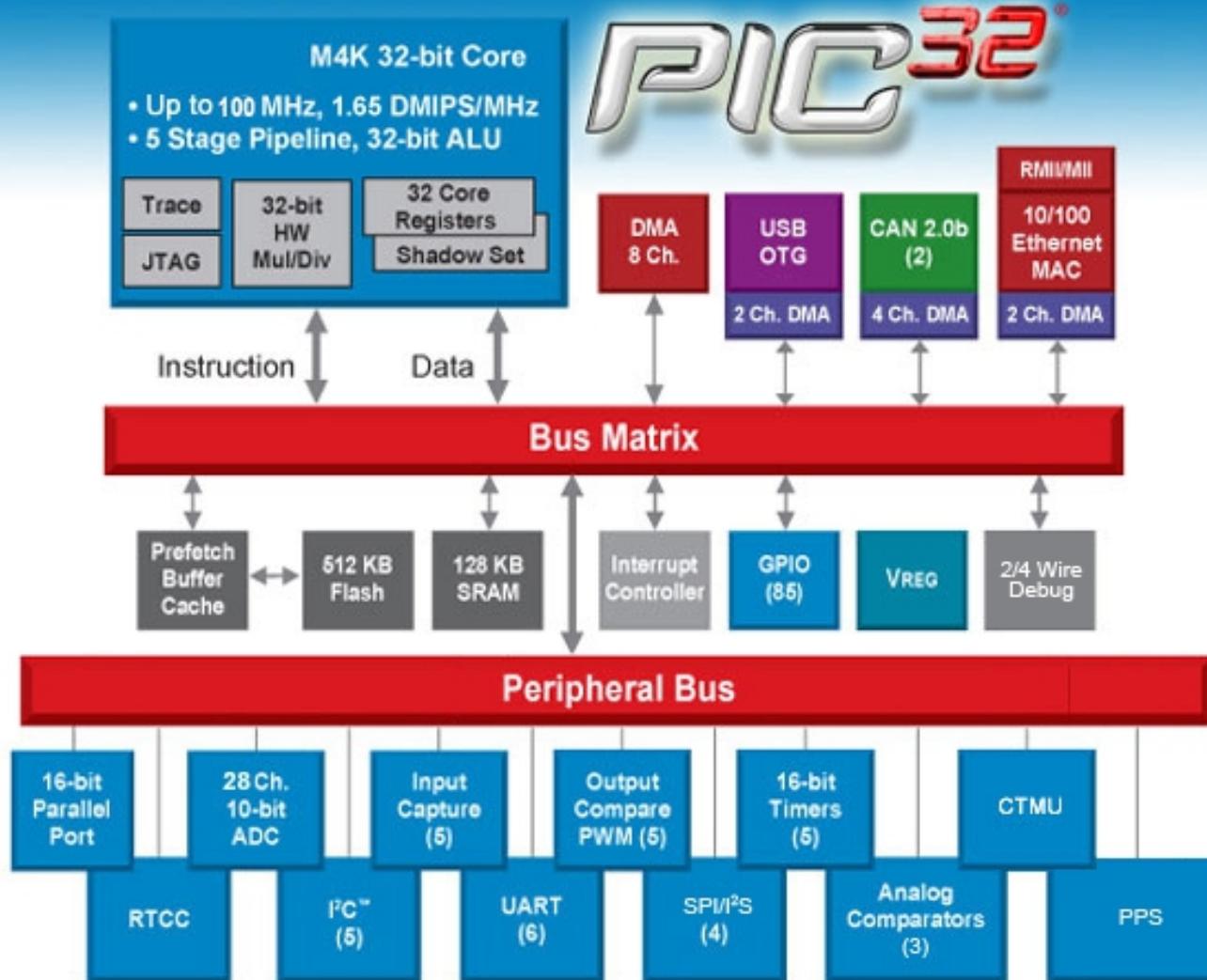
GIỚI THIỆU VỀ VI ĐIỀU KHIỂN PIC

- 16-bit PIC® MCU: PIC24F, PIC24H, PIC24E
- 16-bit dsPIC® DSC: dsPIC30F, dsPIC33F, dsPIC33E.



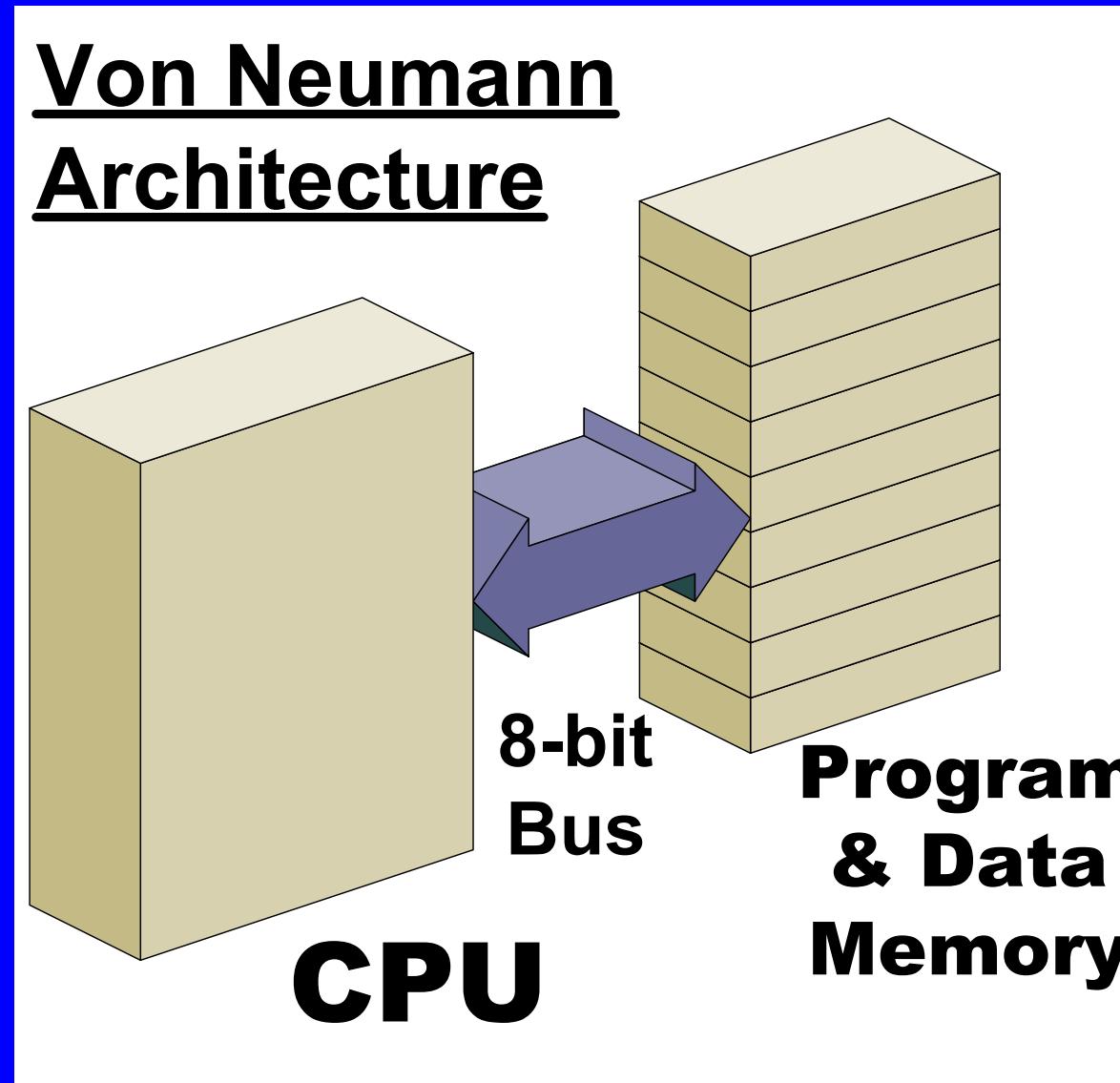
GIỚI THIỆU VỀ VI ĐIỀU KHIỂN PIC

➤ 32-bit PIC® MCU: PIC32

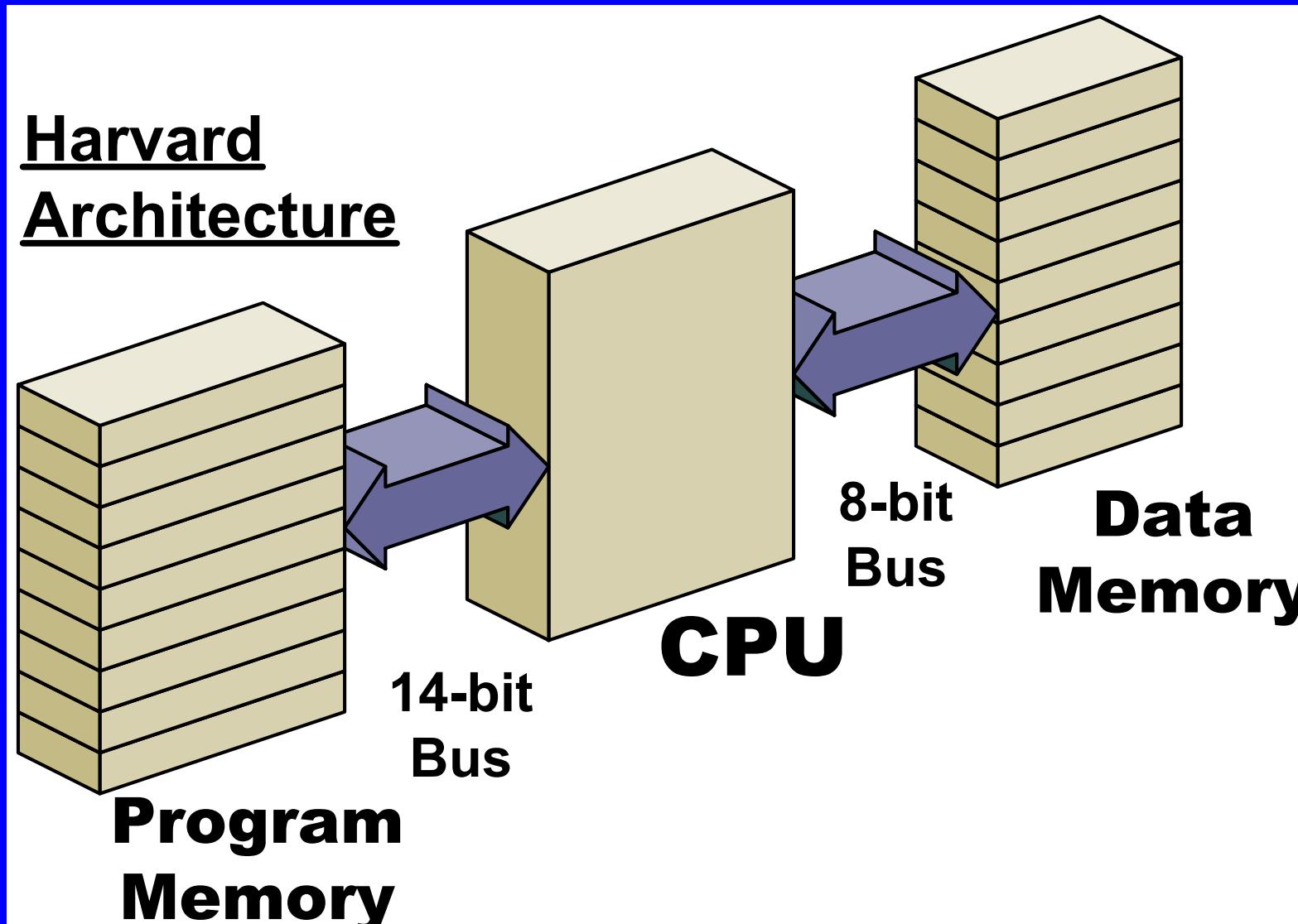


- Có hai kiểu kiến trúc vi điều khiển là **Von Neumann** và **Harvard**
- Kiến trúc **Von Neumann**: Có chung một không gian bộ nhớ cho cả chương trình và dữ liệu, sử dụng chung hệ thống bus nên tốc độ xử lý thông tin sẽ có nhiều hạn chế
- Kiến trúc **Harvard**: Có hai không gian bộ nhớ tách biệt nhau (một cho chương trình và một cho dữ liệu), sử dụng hai hệ thống bus do đó có thể hoạt động cả hai bộ nhớ trong cùng một thời điểm, nên tốc độ xử lý thông tin được cải thiện rất nhiều.

➤ Kiến trúc **Von Neumann**:



➤ Kiến trúc Harvard:

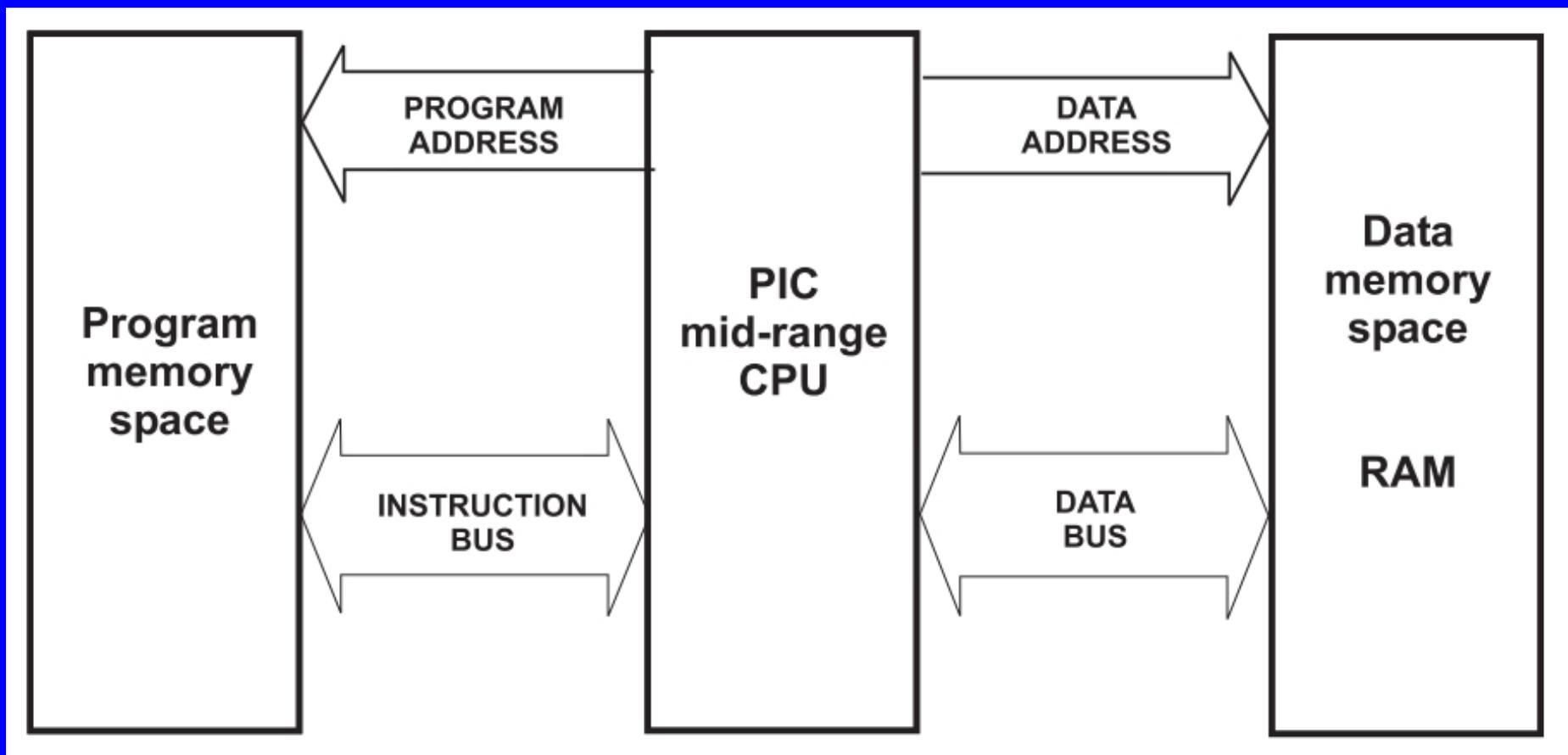


KIẾN TRÚC VI ĐIỀU KHIỂN PIC

- Ngoài ra, đối với kiến trúc Harvard thì cấu trúc tập lệnh có thể được tối ưu tùy theo từng họ vi điều khiển mà không phụ thuộc vào cấu trúc bộ nhớ dữ liệu
- Ví dụ:
 - Kiến trúc **Harvard**: PIC16F có độ dài lệnh là 14bit (trong khi cấu trúc bộ nhớ dữ liệu được chia theo từng Byte)
 - Kiến trúc **Von Neumann**: độ dài lệnh luôn là bội số của Byte (vì cấu trúc bộ nhớ dữ liệu được chia theo từng Byte).

KIẾN TRÚC VI ĐIỀU KHIỂN PIC

- Vi điều khiển PIC chúng ta tìm hiểu trong môn học này (Hệ 8-bit PIC® MCU Mid-Range) sử dụng kiến trúc Harvard.



RISC – CISC

- **Vi điều khiển được tổ chức theo kiến trúc Harvard** còn được gọi là **Vi điều khiển RISC (Reduced Instruction Set Computer)** hay **vi điều khiển có tập lệnh rút gọn**. Là một phương pháp thiết kế vi điều khiển theo hướng đơn giản hóa tập lệnh, thời gian thực thi tất cả các lệnh đều như nhau. Phổ biến là: ARM, AVR, SuperH, MIPS, SPARC, DEC Alpha, PA-RISC, PIC và PowerPC của IBM

- **Vi điều khiển được tổ chức theo kiến trúc Von Neumann** còn được gọi là **Vi điều khiển CISC (Complex Instruction Set Computer)** hay **vi điều khiển có tập lệnh phức tạp**, với độ dài mã lệnh luôn là bội số của Byte. Là một phương pháp thiết kế vi điều khiển theo hướng phức tạp hóa tập lệnh, thời gian thực thi của các lệnh khác nhau. Phổ biến là: Motorola 68x, Intel x86, MCS-51.