디지털 시스템 및 마이크로 컴퓨터 II

1 주차 강의

인제대학교 의용공학부

조 종만 교수

강의 시작에 앞서 (1)

1. 평가 방법
   - 출석: 15%
   - 중간고사: 40%
   - 기말고사: 45%

2. 수업 형태
   - 대면 강의: 3시간

강의 시작에 앞서 (2)

3. 교재 소개

- ATmega328PB 마이크로컨트롤러의 구조 및 프로그래밍
- 인제대학교 출판부, 조종만
- ISBN: 978-89-6620-105-1

4. 의용공학과 학생으로서 왜 이 교과목를 배우는가?

- 의료기기의 핵심 요소
- 의공학 관련 연구의 핵심 도구

Part 1

Review of Digital Systems

Review of Digital Systems (1)

- Number systems – Binary, decimal, hexadecimal
- Boolean algebra
  - Basic operations of Boolean algebra – AND, OR, NOT, NAND, NOR, XOR
  - Theorems and laws of Boolean algebra
- Building an equation from verbal expressions
- Truth table
- Minterm, maxterm

Review of Digital Systems (2)

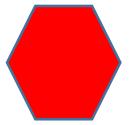
- Simplification of Boolean expression
- Karnaugh map, Quine-McCluskey method
- Commercialized ICs
  - Multiplexer, Encoder, Decoder
- Programmable logic devices (PLD)
- VHDL for combinational logic circuits

Review of Digital Systems (3)

- Latches
- Flip-Flops
  - D F/F, S-R F/F, J-K F/F, T F/F
- Registers
- Counters
  - Straight, Non-straight binary counter

Review of Digital Systems (4)

- Analysis of Clocked Sequential Circuits
  - Moore machine, Mealy machine
  - State graph, state table
- Design of Sequential Circuits
- VHDL for sequential logic circuits



Microcontroller

Wrap-Up

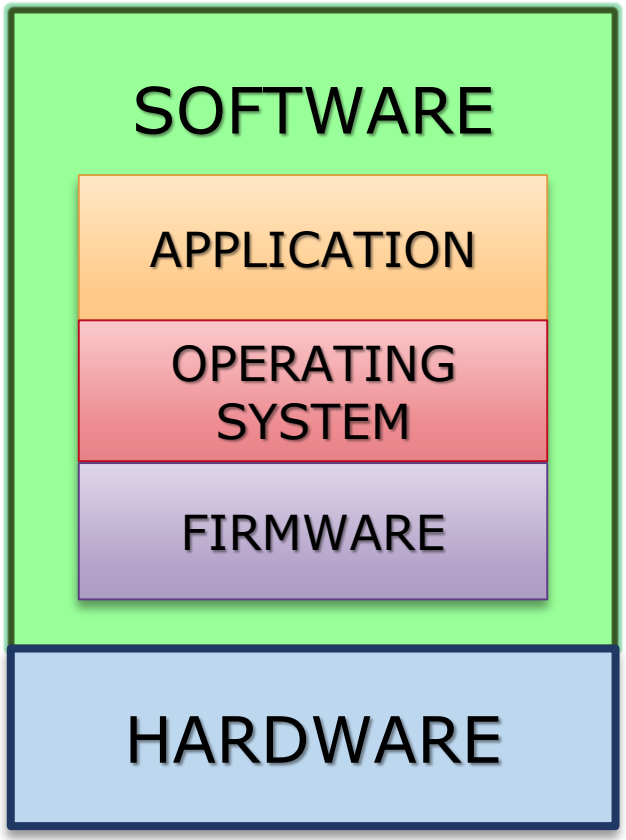
Part 2

Introduction to Microcontroller

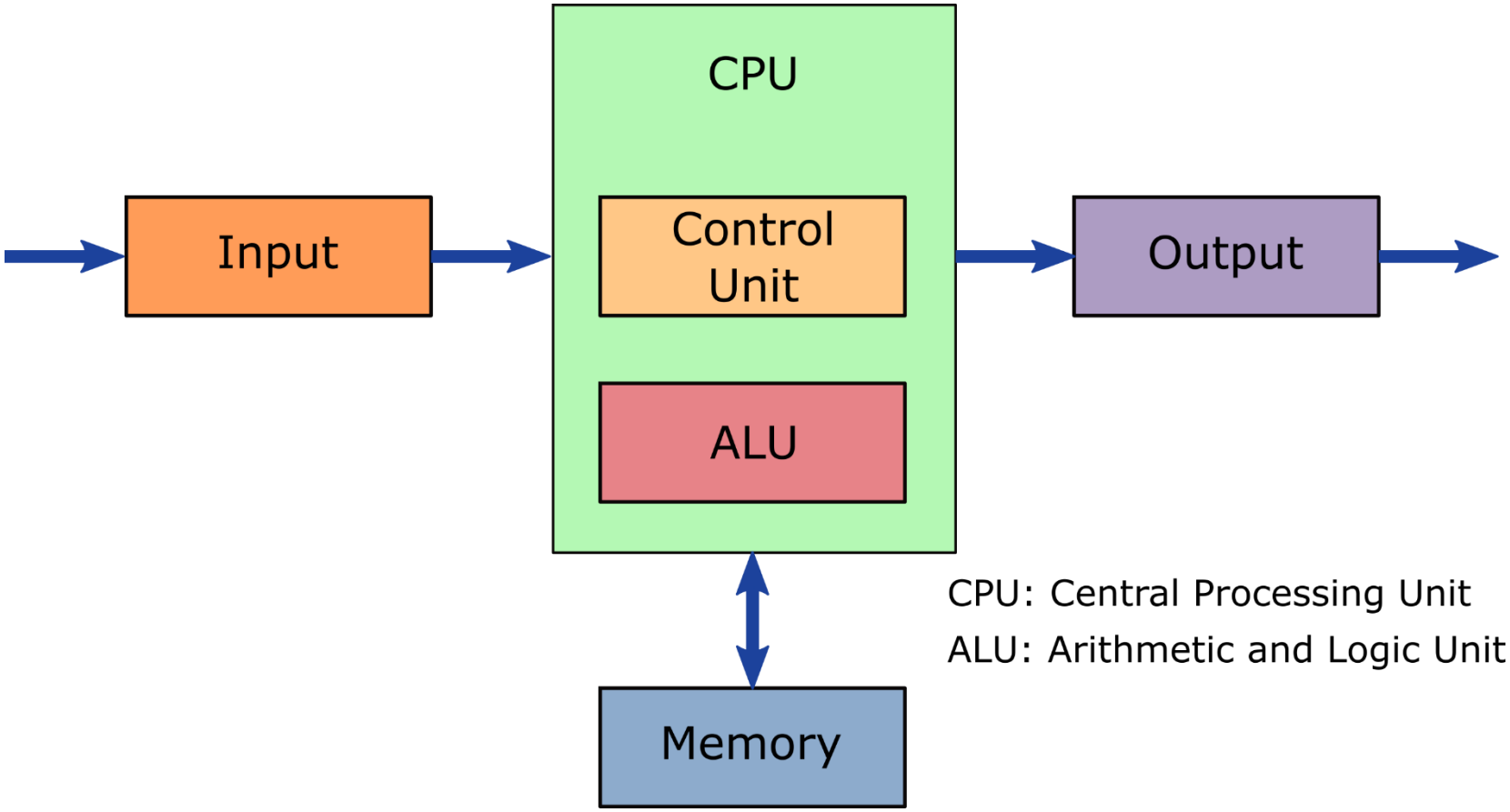
Objectives

- Computer Architecture
- CPU, Microprocessor and Microcontroller
- AVR Microcontrollers
- ATmega328PB Microcontroller Registers

Computer Organization



Computer Hardware



CPU: Central Processing Unit

ALU: Arithmetic and Logic Unit

Computer Software (1)

- Program
  - A set of instructions that the computer hardware can execute
- Machine Instruction / Machine Language
  - All programs are stored in the computer's memory in the form of binary number (machine instruction).
  - It is difficult to use and not productive.

- Ex: Machine language program
  - 1001 0100 0001 0011b

    stands for "increment the contents of R1 register by 1."
  - 0000 1100 0010 0001b

    stands for "add the contents of R1 register to the contents of R2 register."

Computer Software (2)

- Assembly Language
  - Invented to simplify the programming
  - Consists of assembly instructions
  - Mnemonic representation of a machine instruction

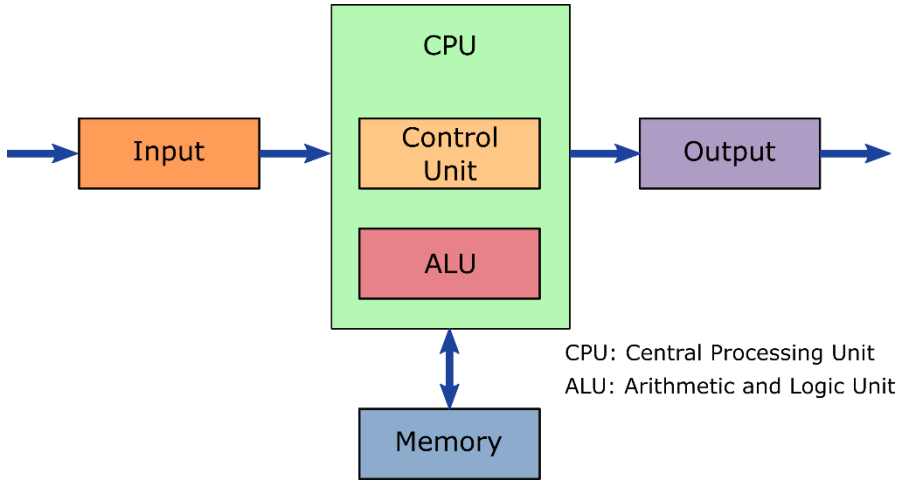
- Ex: Assembly language program
  - INC R1

    stands for “increment the contents of R1 register by 1.”
  - ADD R2, R1

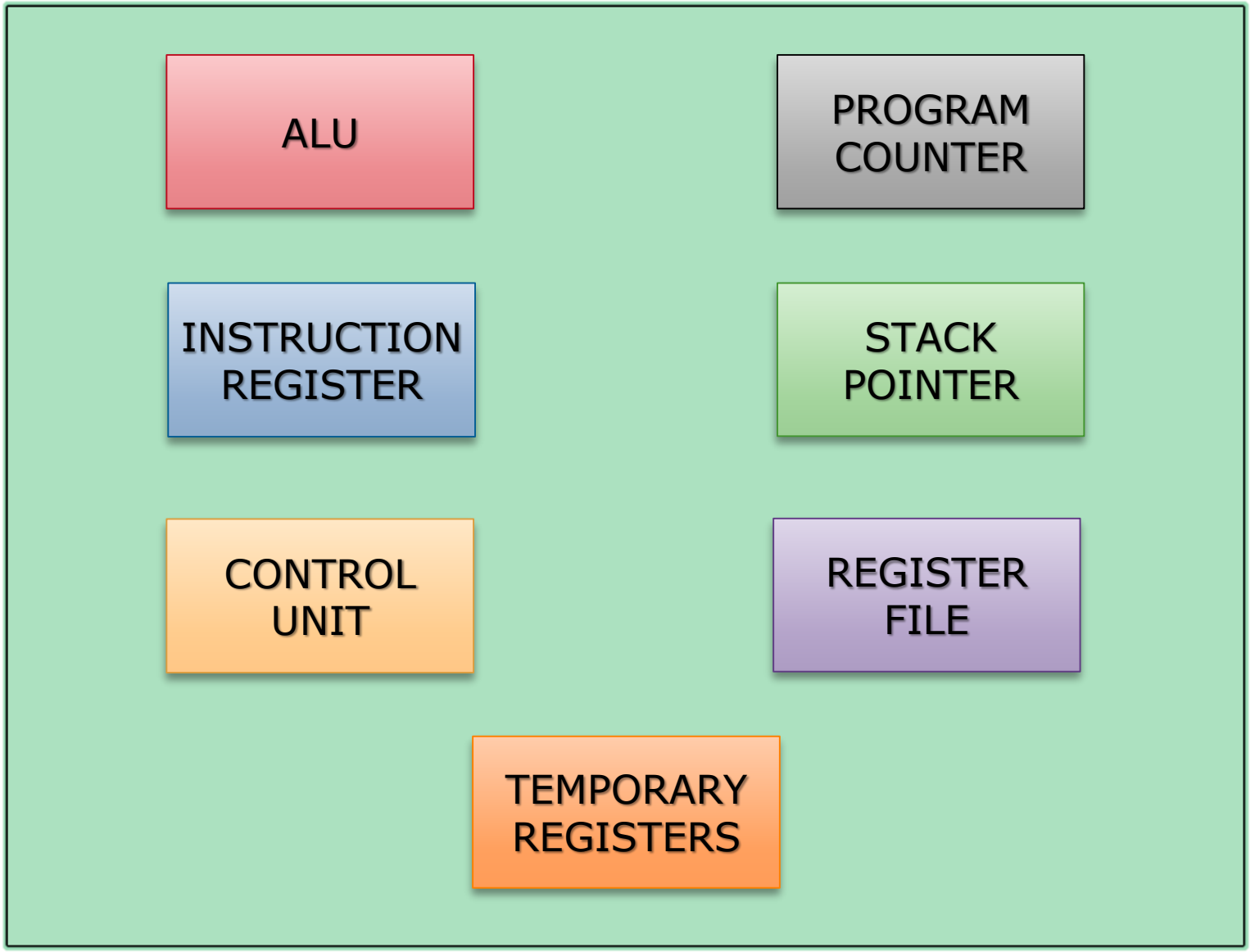
    stands for “add the contents of R1 register to the contents of R2 register.”

Central Processing Unit (CPU) (1)

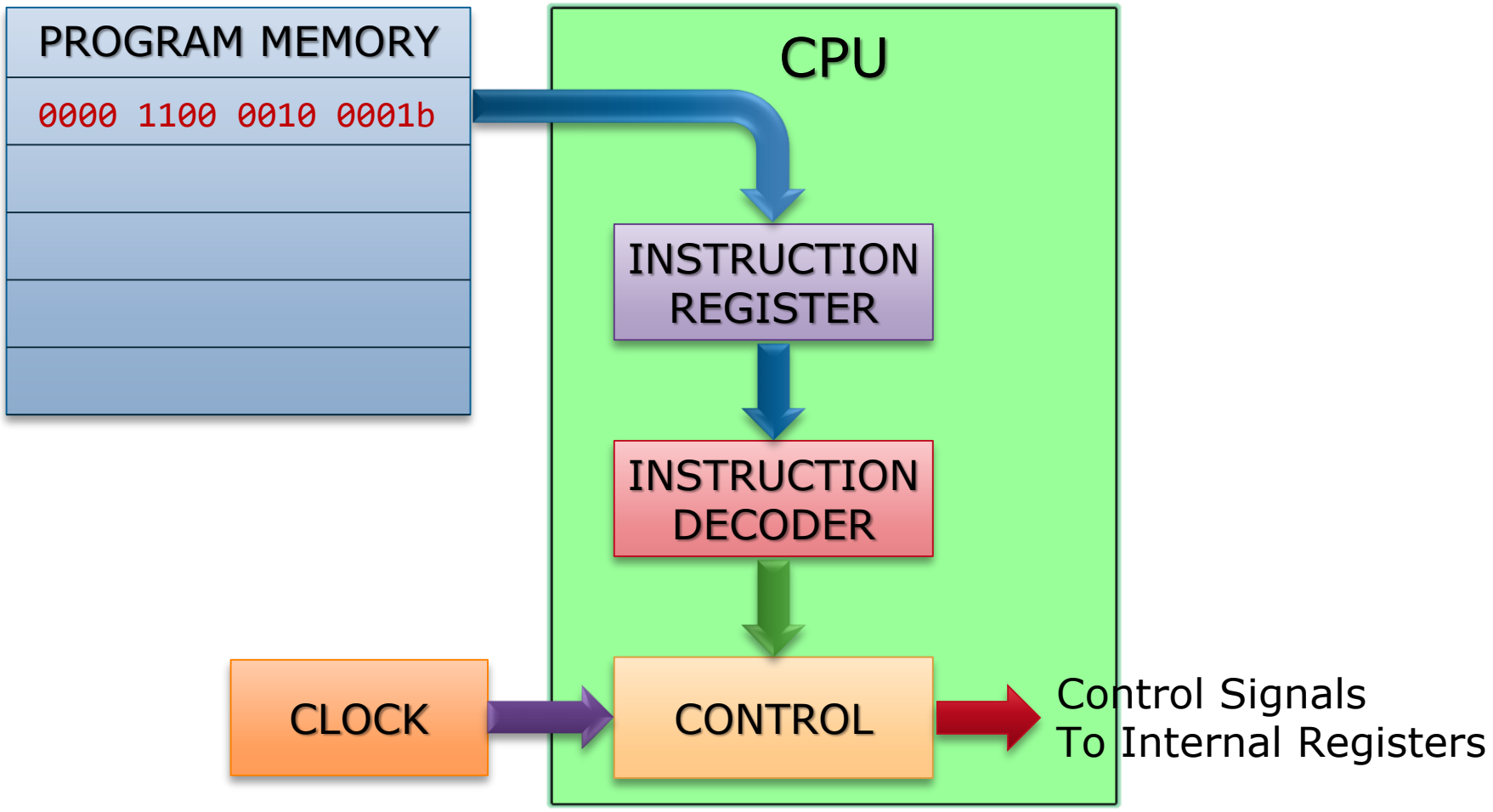
- Arithmetic Logic Unit (ALU)
  - Execute numerical and logical operation
- Register file
  - Storage location inside the CPU
  - Used to hold data / memory address
  - Access to data in register is much faster than memory
  - Number of registers varies depending on CPU
- Control unit
  - Hardware instruction logic
  - Decodes and monitors the execution of instructions
  - System clock synchronizes the activities of CPU



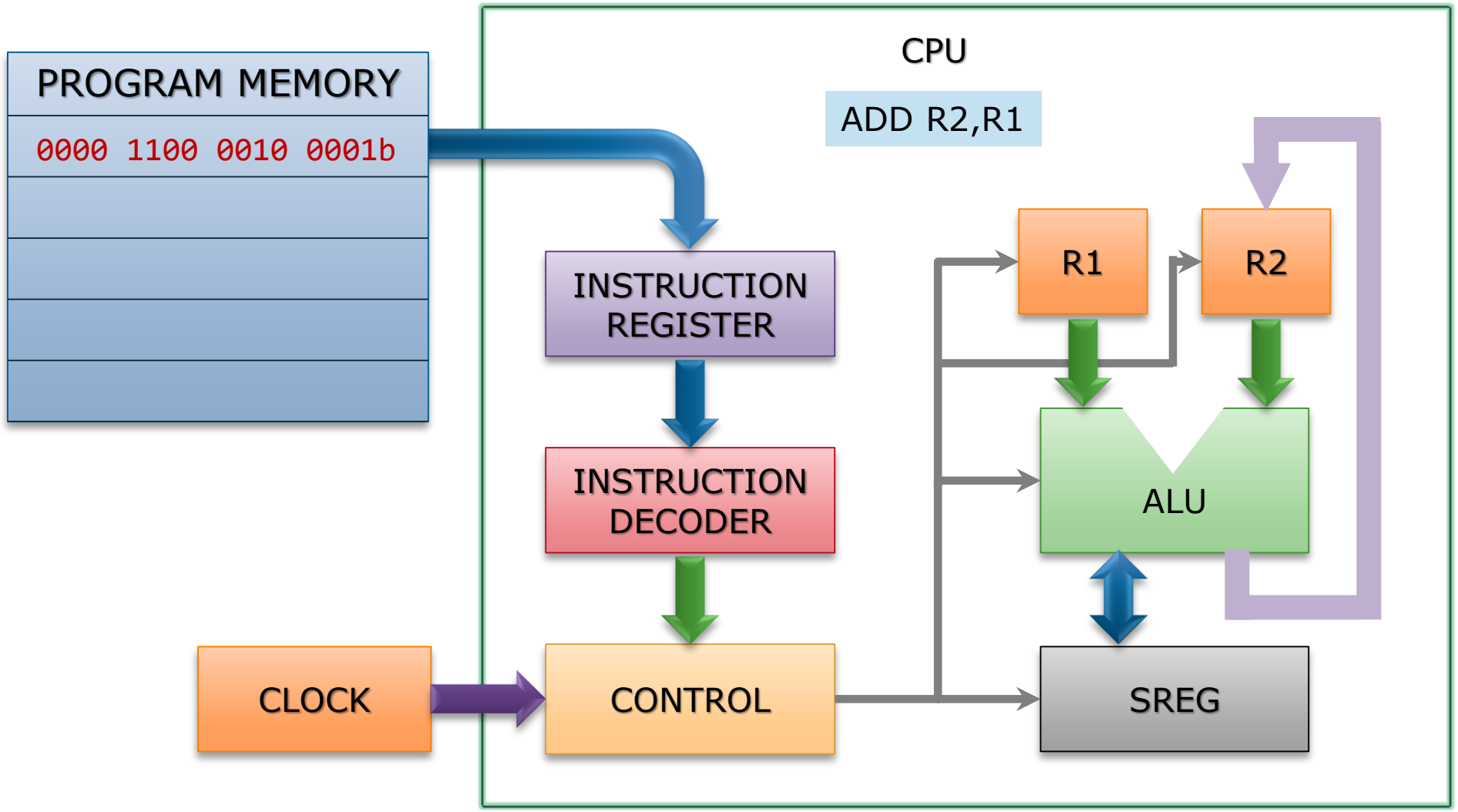
Central Processing Unit (CPU) (2)



Control Unit



ALU, Register File, and Control Unit



Microprocessor (μP)

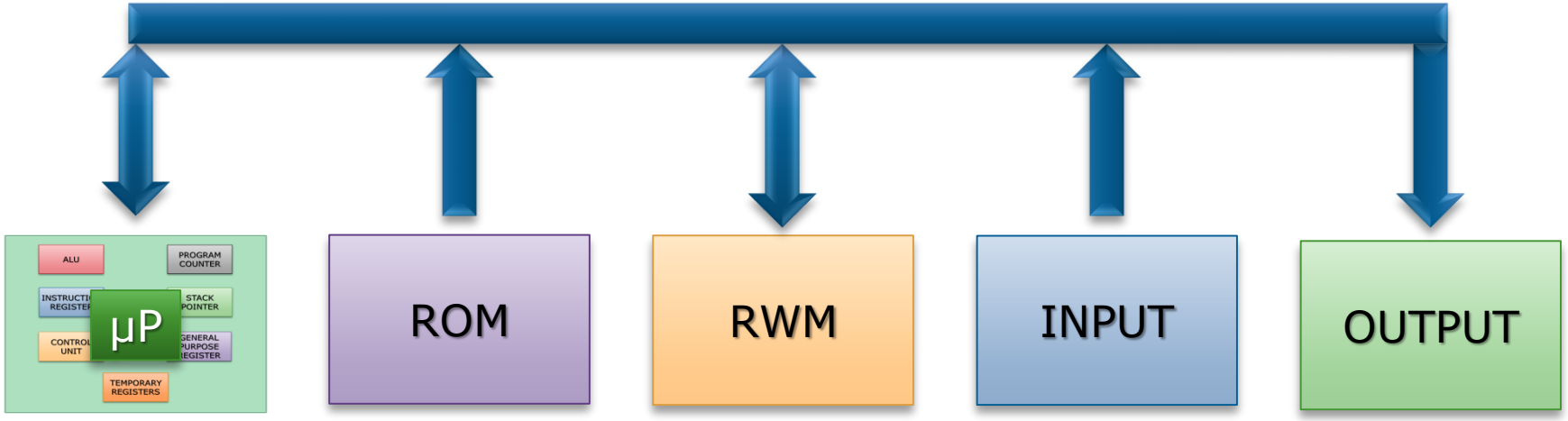
- A processor fabricated in a single IC
- The number of bits of μP refers to

  the number of bits that μP can manipulate in one operation
- Limitation of μP
  - Requires external memory to execute programs.
  - Cannot directly interface with I/O devices.

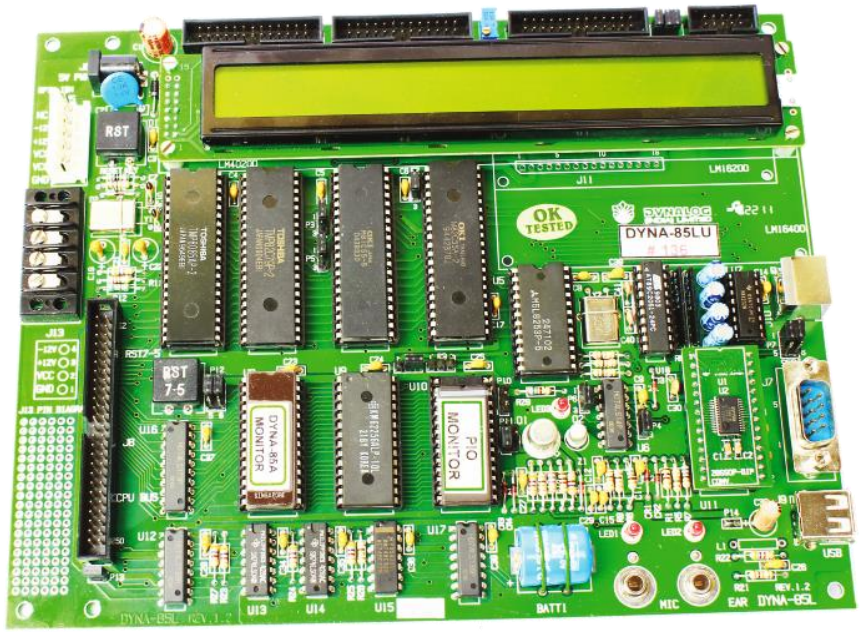
    Peripheral chips are needed.
  - Address decoders and buffers are needed.
  - Bigger system size

Microprocessor-Based System (1)

SYSTEM BUS
(Address, Data, Control)



Microprocessor-Based System (2)



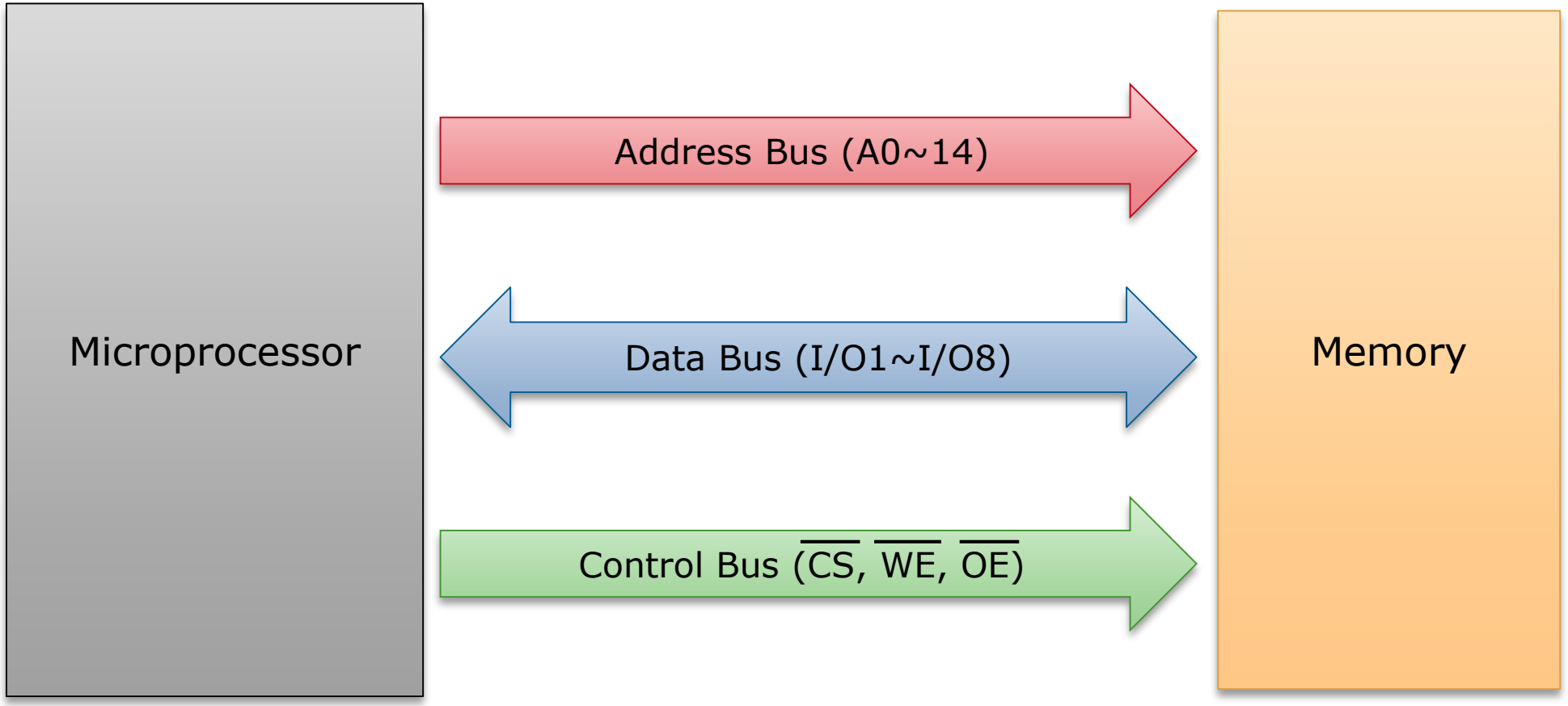
Dynalog, India에서 인용

Universidad Carlos IIIde Madris에서 인용

malinov.com에서 인용

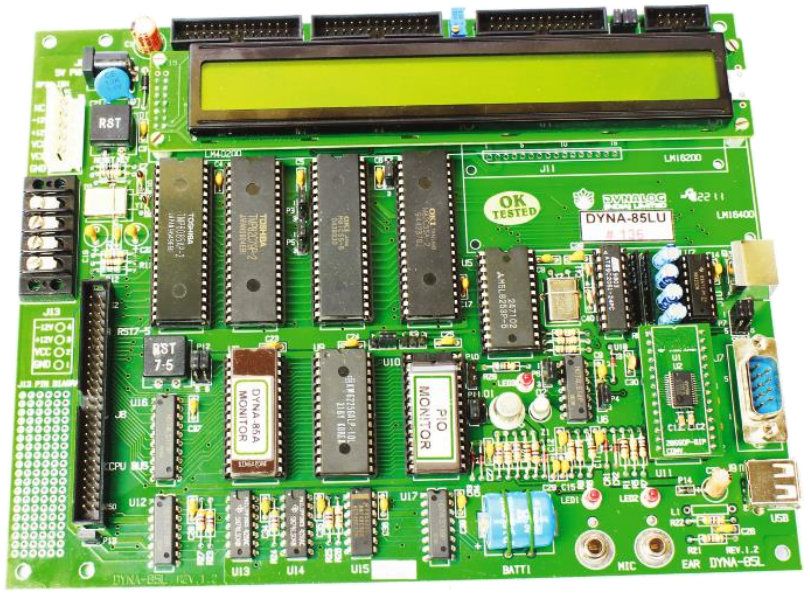
Memory Interface

Memory Interface Example (62256 - 32k x 8 bit Low Power CMOS Static RAM)

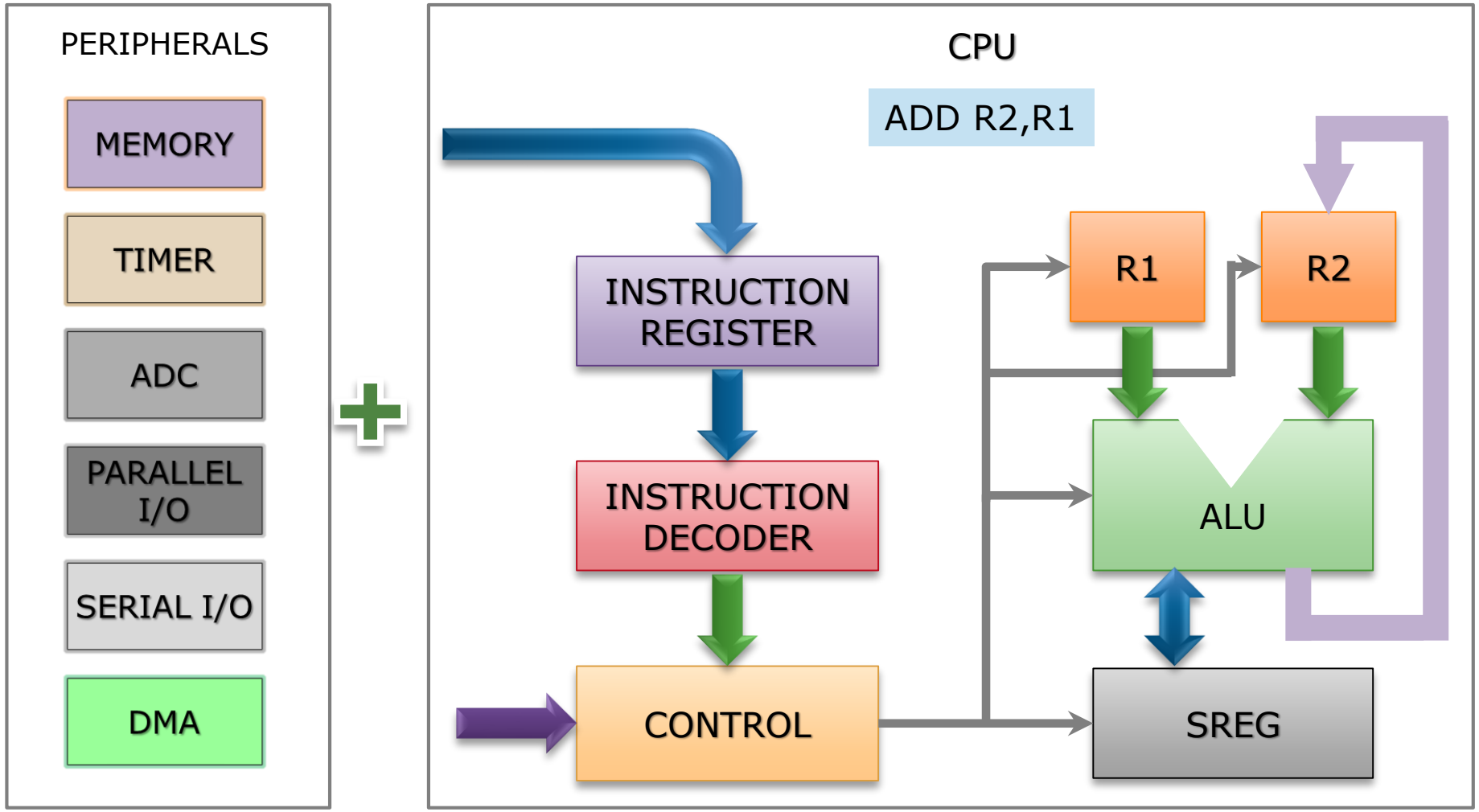


Microcontroller

- A computer implemented on a single VLSI chip
- Contains everything a µP contains plus
  - Memory
  - Timer
  - Analog-to-digital converter (ADC)
  - Digital-to-analog converter (DAC)
  - Parallel I/O interface
  - Serial I/O interface
  - Memory component interface circuitry
  - Direct memory access (DMA)



General Microcontroller



Summary

- Computer organization
  - hardware and software
- Hardware
  - CPU, input, output, and memory
- Software
  - Machine language, Assembly language
- CPU
  - ALU, register file, and control unit
- Microprocessor
- Microcontroller

Part 3

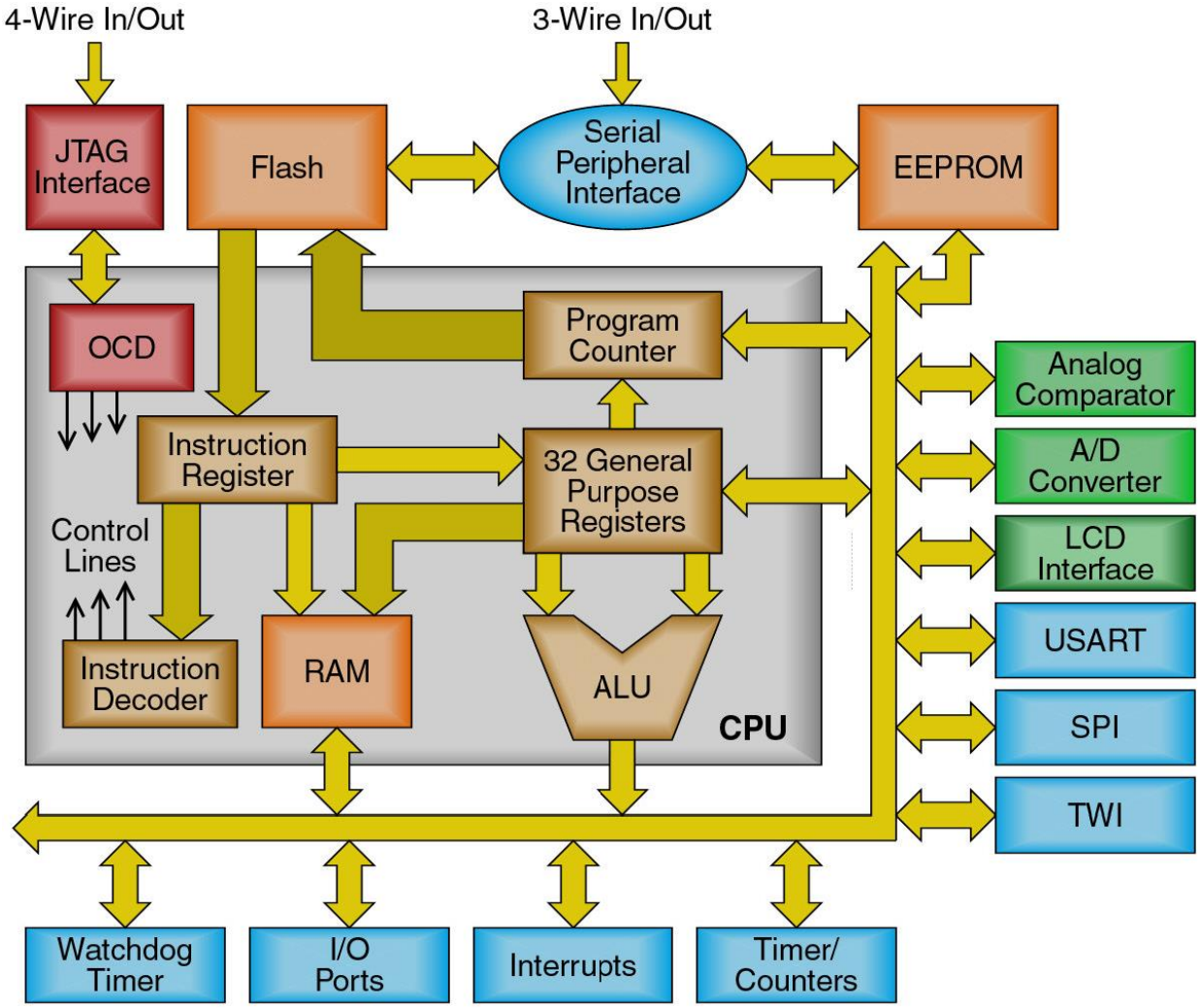
Introduction to AVR Microcontroller

Overview of 8-bit AVR Microcontroller (1)

It is commonly accepted that **AVR** stands for **A**lf-Egil Bogen and **V**egard Wollan's **R**ISC processor.

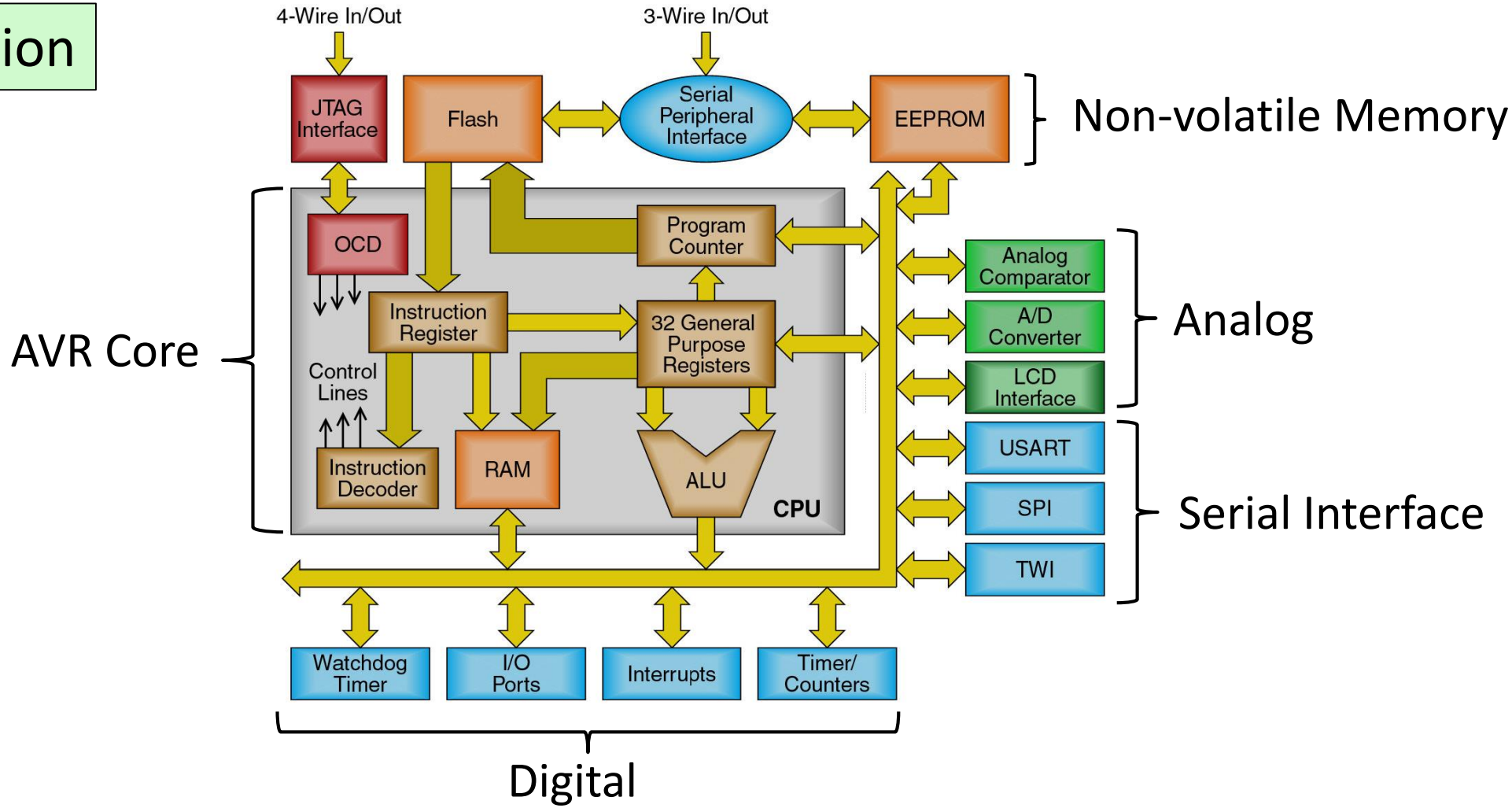
Overview of 8-bit AVR Microcontroller (2)

- RISC architecture with CISC instruction set
  - ✓ Powerful instruction set for C and Assembly
- Scalable
  - ✓ Same powerful AVR core in all devices
- Single cycle execution
  - ✓ One instruction per external clock
  - ✓ Low power consumption
- 32 working Registers
  - ✓ All directly connected to ALU!
- Very efficient core
  - ✓ 20 MIPS @ 20MHz
- High System Level Integration
  - ✓ Lowest total system cost



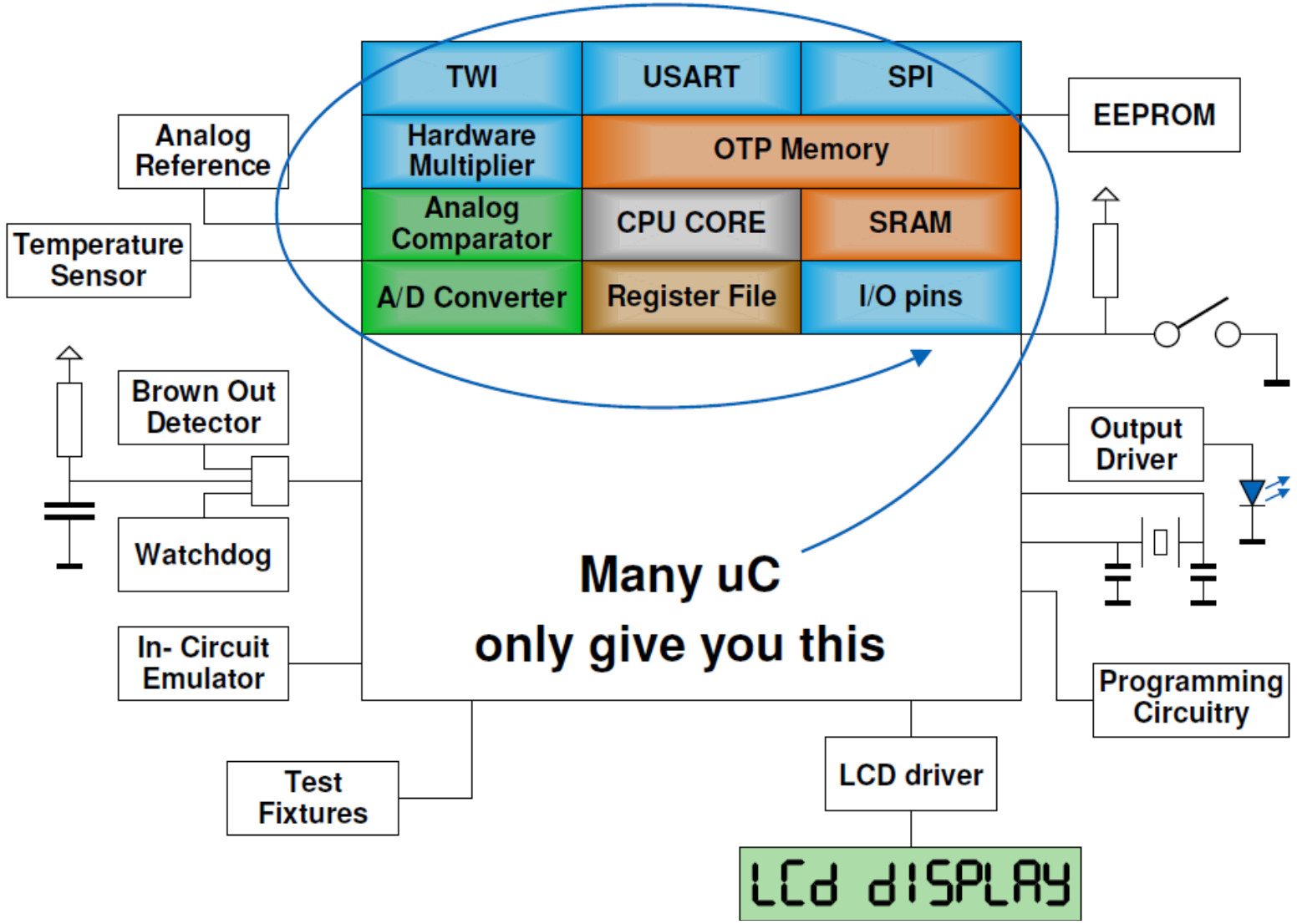
Overview of 8-bit AVR Microcontroller (3)

High Integration



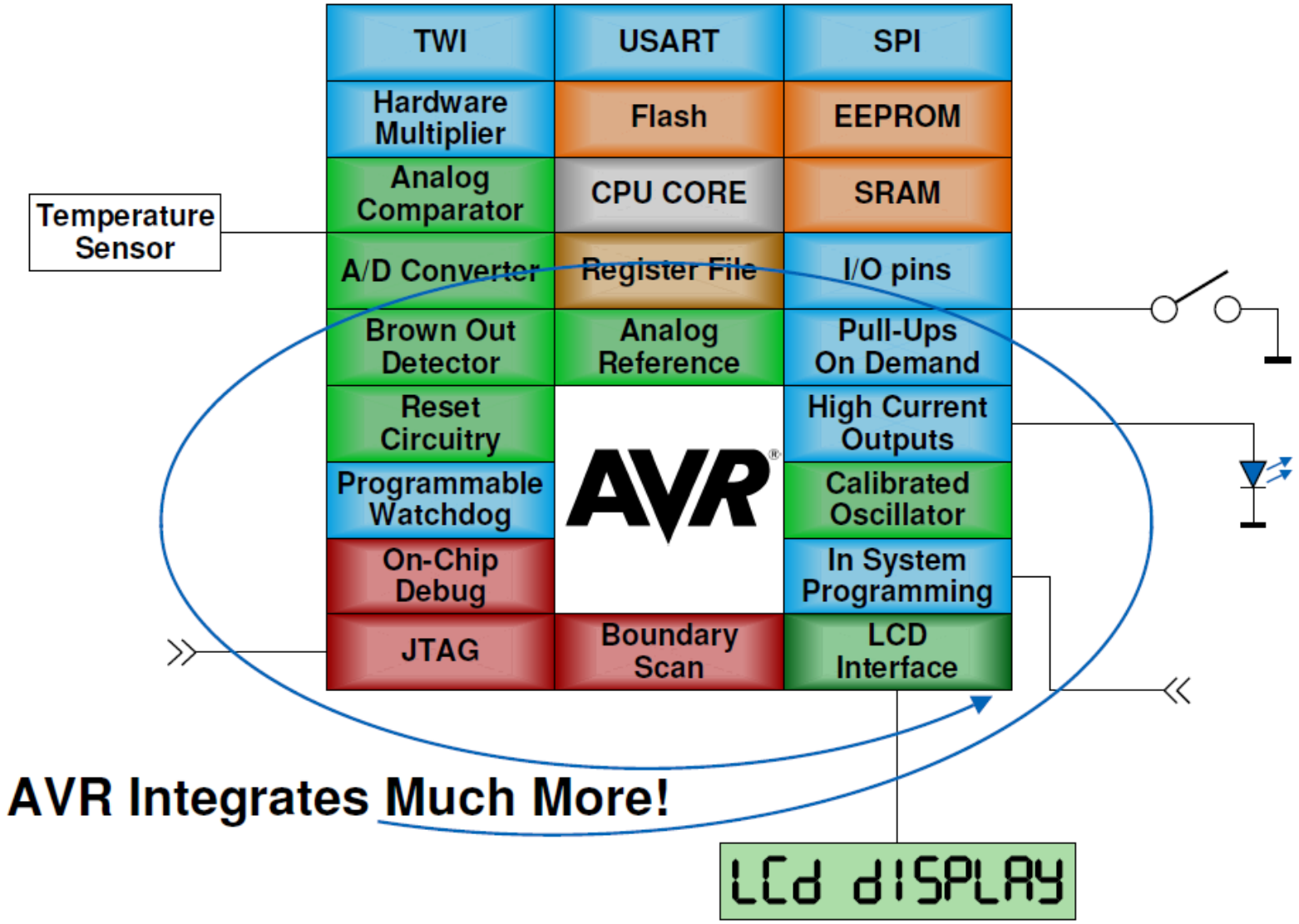
Overview of 8-bit AVR Microcontroller (4)

Single Chip Solution

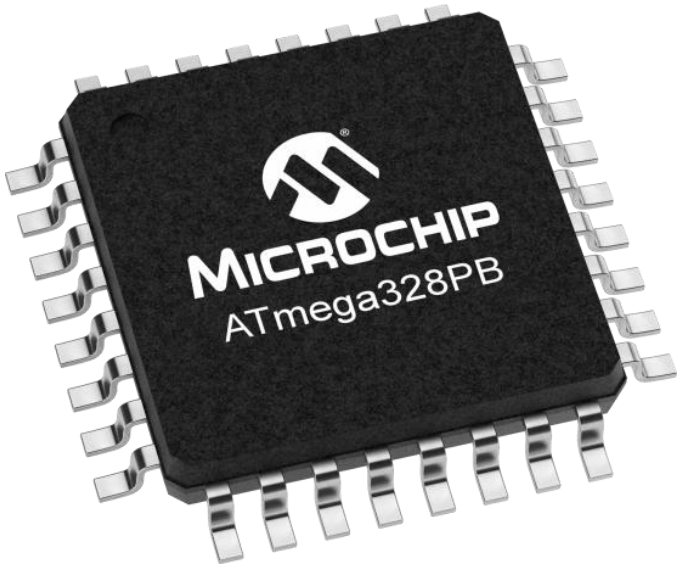


Overview of 8-bit AVR Microcontroller (5)

Single Chip Solution

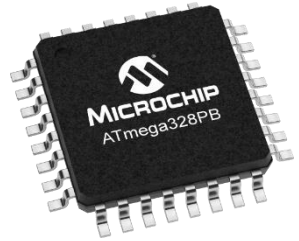


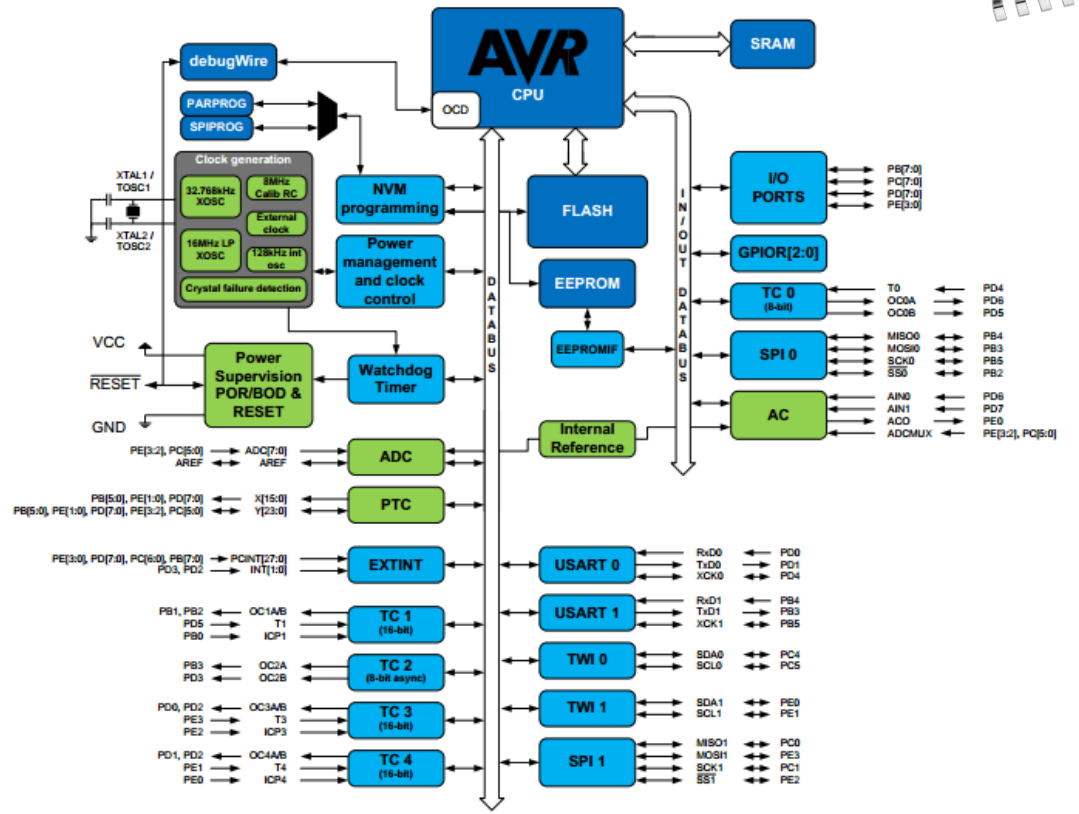
Introduction to ATmega328PB



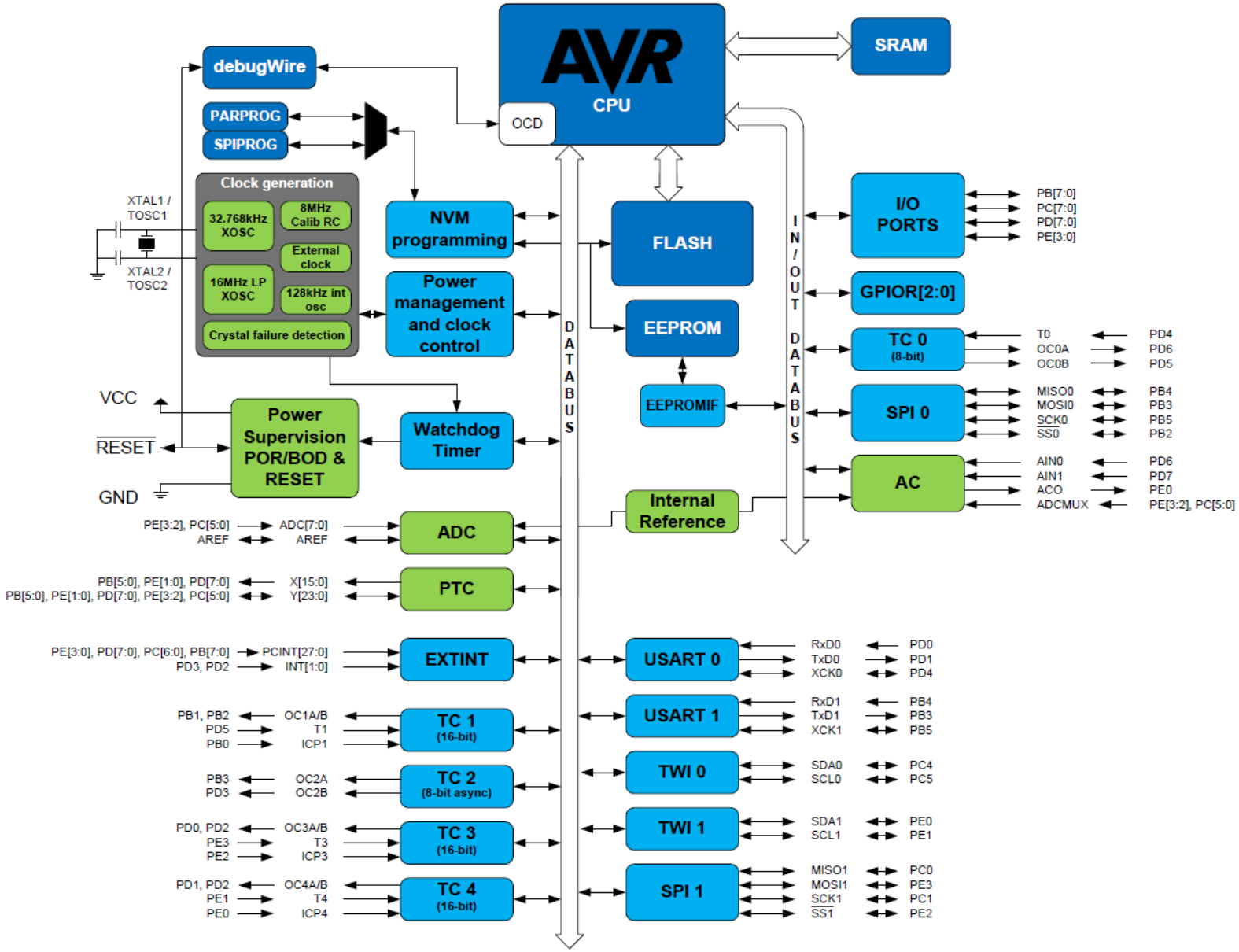
ATmega328PB Features

- Memories
  - 32 kBytes of In-System Self-Programmable Flash program memory
  - 1 kBytes EEPROM
  - 2 kBytes Internal SRAM
- Peripherals
  - GPIO
  - Timer/Counters
  - 10-bit ADC
  - USART
  - SPI
  - TWI
- Operating Voltage: 1.8 - 5.5V
- Speed Grade:
  - 0 - 4MHz @ 1.8 - 5.5V
  - 0 - 10MHz @ 2.7 - 5.5V
  - 0 - 20MHz @ 4.5 - 5.5V

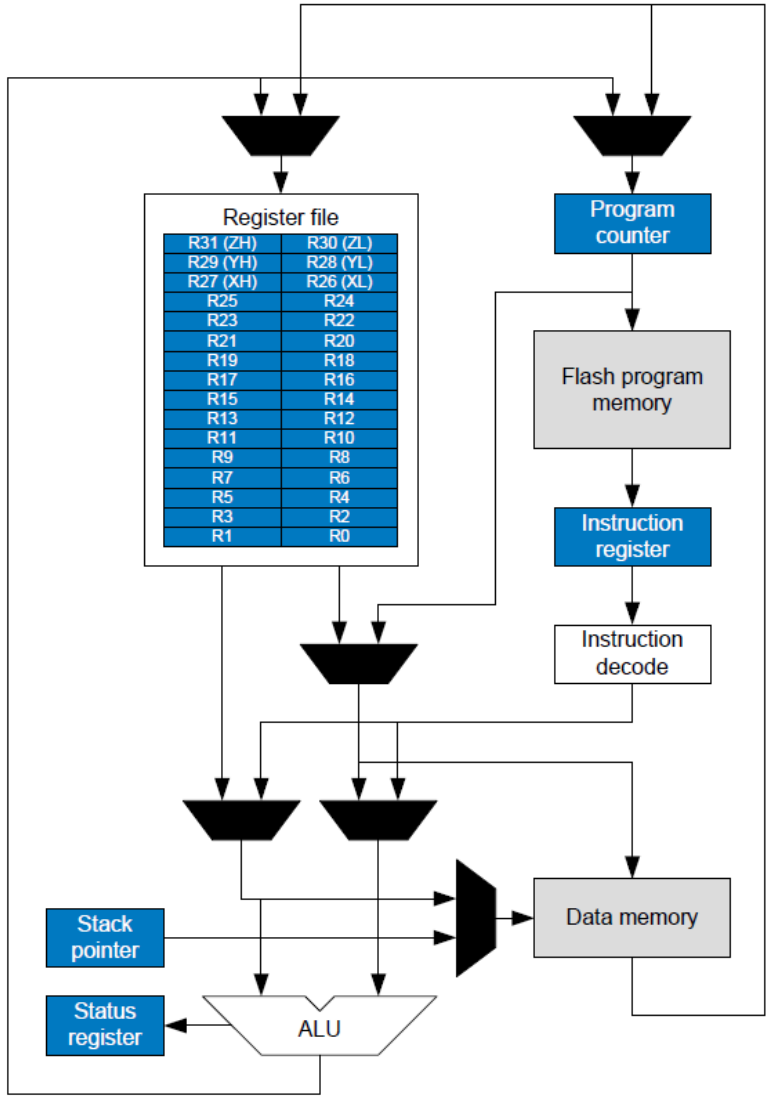




ATmega328PB Block Diagram



AVR CPU Core



- AVR uses Harvard architecture
  - Separate memories and buses for program and data
- Single level pipelining for instruction
- Register File
  - 32 x 8-bit general purpose working registers with a single clock cycle access time
- ALU (Arithmetic Logic Unit)
- Status Register
  - Contains information about the result of the most recently executed arithmetic instruction.
  - This information can be used for altering program flow in order to perform conditional operations.
  - The Status Register is updated after all ALU operations.

AVR Status Register

Bit No.	7	6	5	4	3	2	1	0
Name	I	T	H	S	V	N	Z	C
Reset	0	0	0	0	0	0	0	0

- Bit 7 – I: Global Interrupt Enable
- Bit 6 – T: Copy Storage
- Bit 5 – H: Half Carry Flag
- Bit 4 – S: Sign Flag. $S = N \oplus V$
- Bit 3 – V: Two's Complement Overflow Flag
- Bit 2 – N: Negative Flag
- Bit 1 – Z: Zero Flag
- Bit 0 – C: Carry Flag

AVR General Purpose Register File

	Address	
R0	0x00	
R1	0x01	
R2	0x02	
…		
R13	0x0D	
R14	0x0E	
R15	0x0F	
R16	0x10	
R17	0x11	
…		
R26	0x1A	X-register Low Byte
R27	0x1B	X-register High Byte
R28	0x1C	Y-register Low Byte
R29	0x1D	Y-register High Byte
R30	0x1E	Z-register Low Byte
R31	0x1F	Z-register High Byte

Registers are special storages with 8 bits capacity.
They are connected directly to the CPU → fast access time.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Additional Function:
These registers are 16-bit address pointers for indirect addressing of the memory space.
X, Y, Z: for Data Memory
Z: Program Memory

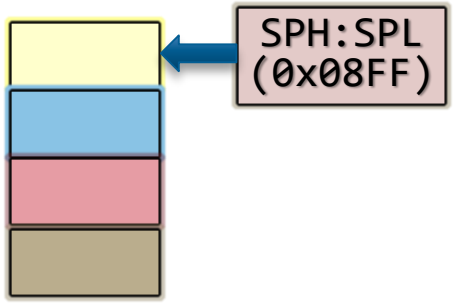
Stack Pointer for AVR

- Stack Pointer, SPL and SPH (0x3D and 0x3E)
  - A stack is a last-in first-out data structure
  - AVR stack is implemented as growing from higher to lower memory locations
  - 16-bit stack pointer points to the top of stack

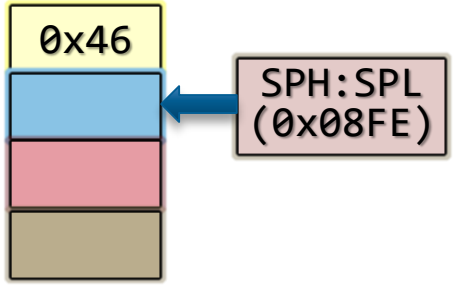
Stack and Stack Pointer for AVR

```
1.   LDI   R16,0x08
2.   OUT   SPH,R16
3.   LDI   R16,0xFF
4.   OUT   SPL,R16

5.   LDI   R16,0x46
6.   LDI   R17,0x47

7.   PUSH  R16
8.   PUSH  R17
9.   POP   R17
10.  POP   R19
```

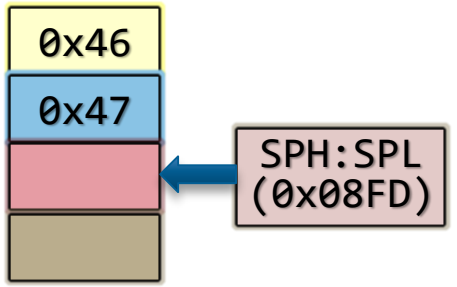
ATmega328PB RAM ADDRESS: 0x0100~0x08FF



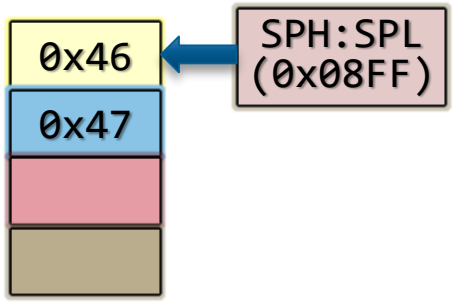
After Line #4



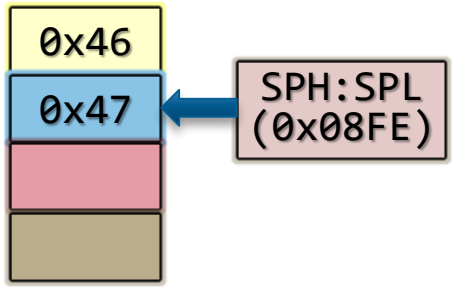
After Line #7



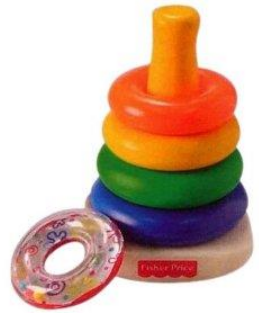
After Line #8



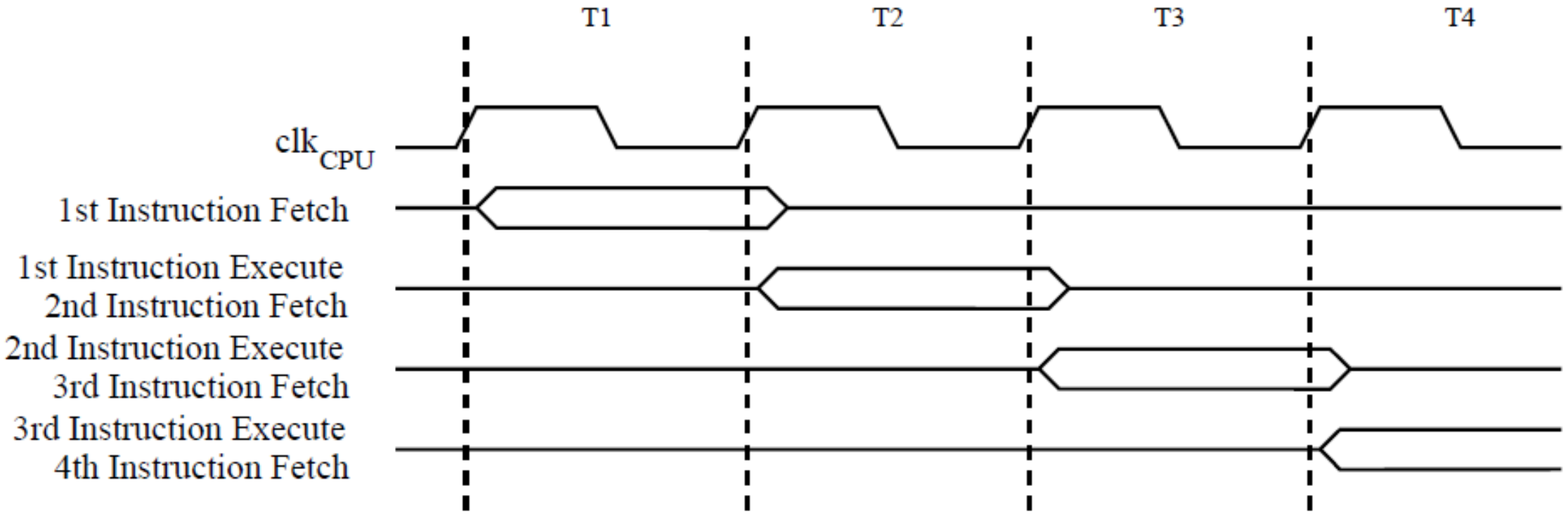
After Line #10



After Line #9

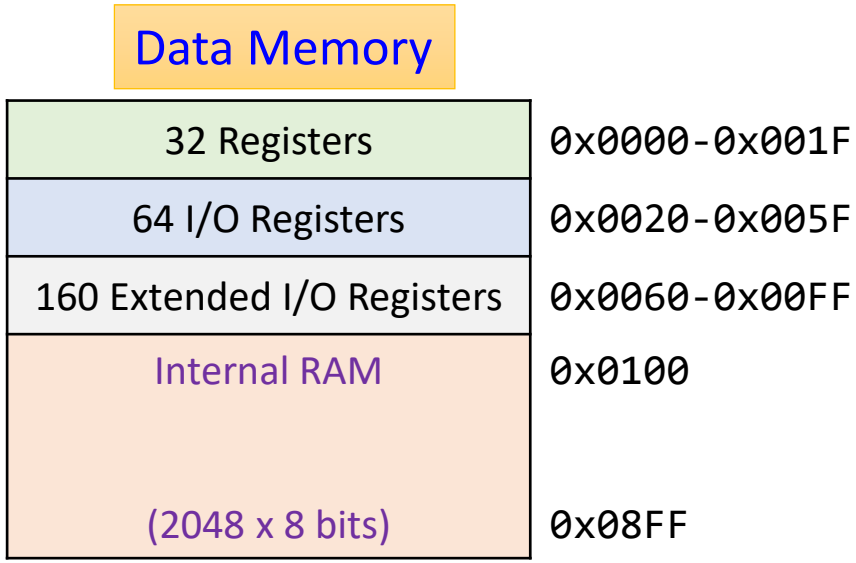


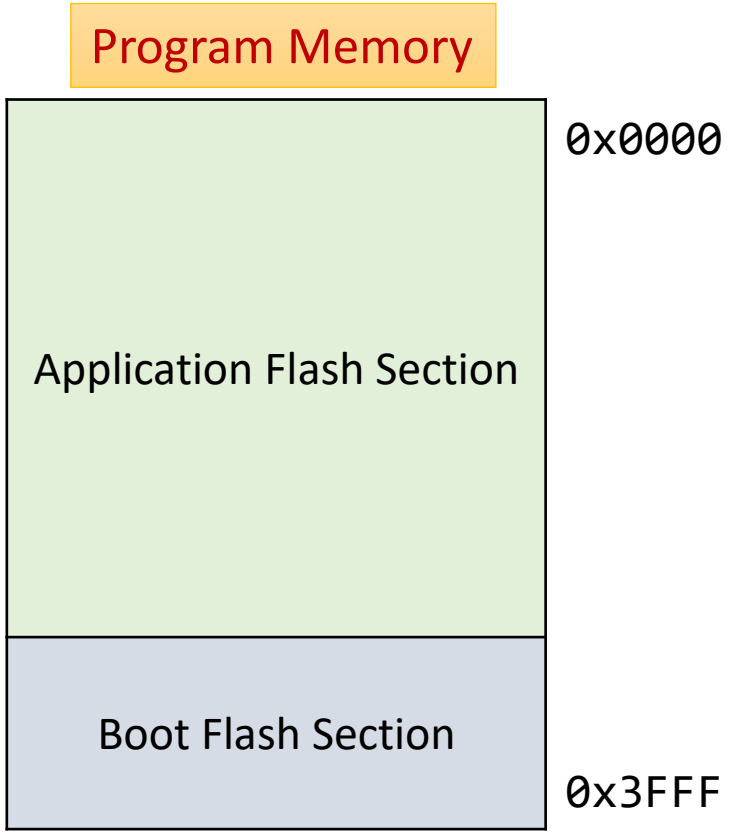
AVR Instruction Execution Timing (Pipeline)



ATmega328PB Memories (1)

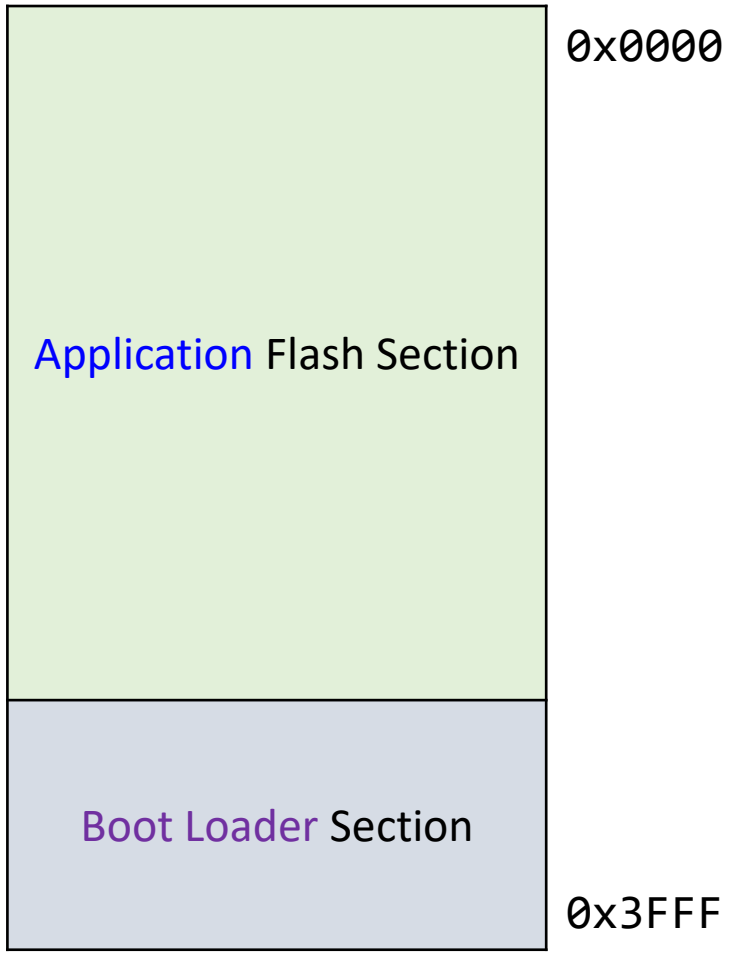
- Two memory spaces of ATmega328PB
  - Data memory
  - Program memory





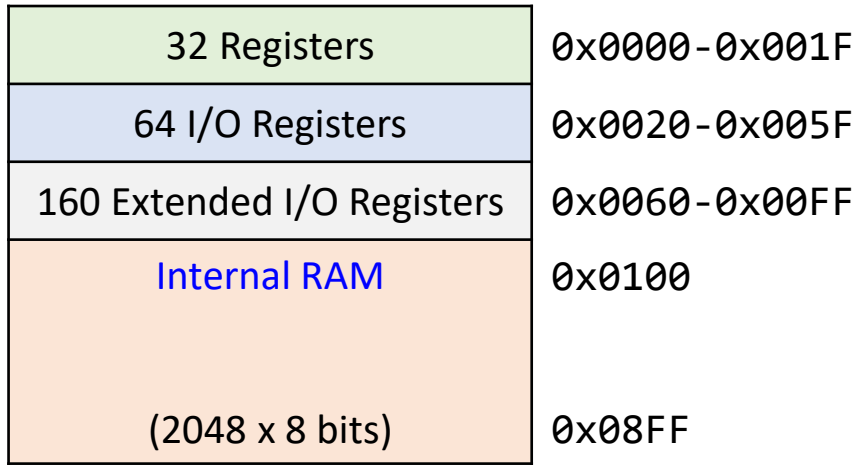
ATmega328PB Memories (2)

- In-System Reprogrammable Flash Program Memory
  - Boot Loader Section
  - Application Program Section



ATmega328PB Memories (3)

- SRAM Data Memory Space
  - Register File: 32
  - I/O Registers: 64
  - Extended I/O Registers: 160
  - Internal data SRAM: 2048



ATmega328PB Memories (4)

- EEPROM Data Memory
  - Electrically Erasable Programmable Read Only Memory
  - 1,024 Bytes of data EEPROM
  - Can be accessible by byte unit.
  - Endurance of at least 100,000 write/erase cycles.

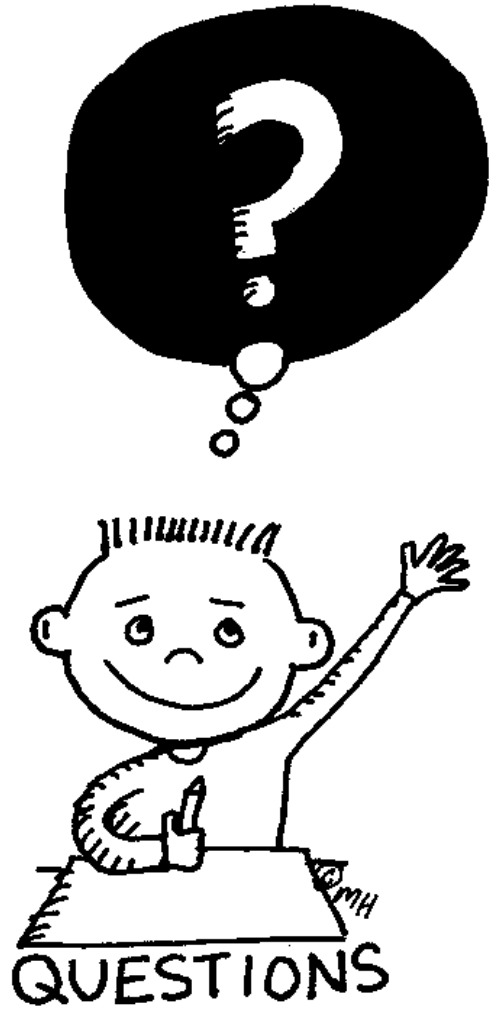
Units for Memory Size

Specific units of IEC 60027-2 A.2 and ISO/IEC 80000
(International Electrotechnical Commission)

IEC prefix		Representations				Customary prefix	
Name	**Symbol**	**Base 2**	**Base 1024**	**Value**	**Base 10**	**Name**	**Symbol**
kibi	Ki	2^{10}	1024^{1}	1024	≈1.02×10^3	kilo	k, K
mebi	Mi	2^{20}	1024^{2}	1,048,576	≈1.05×10^6	mega	M
gibi	Gi	2^{30}	1024^{3}	1,073,741,824	≈1.07×10^9	giga	G
tebi	Ti	2^{40}	1024^{4}	1,099,511,627,776	≈1.10×10^12	tera	T
pebi	Pi	2^{50}	1024^{5}	1,125,899,906,842,624	≈1.13×10^15	peta	P
exbi	Ei	2^{60}	1024^{6}	1,152,921,504,606,846,976	≈1.15×10^18	exa	E
zebi	Zi	2^{70}	1024^{7}	1,180,591,620,717,411,303,424	≈1.18×10^21	zetta	Z
yobi	Yi	2^{80}	1024^{8}	1,208,925,819,614,629,174,706,176	≈1.21×10^24	yotta	Y

Summary

- Number system
  - Decimal, binary, and hexadecimal numbers
- Computer organization
  - hardware and software
- Hardware
  - CPU, input, output, and memory
- Software
  - Machine language, Assembly language
- CPU
  - ALU, register file, and control unit
- Microprocessor
- Microcontroller
- AVR microcontroller


QUESTIONS


WHAT'S
NEXT?