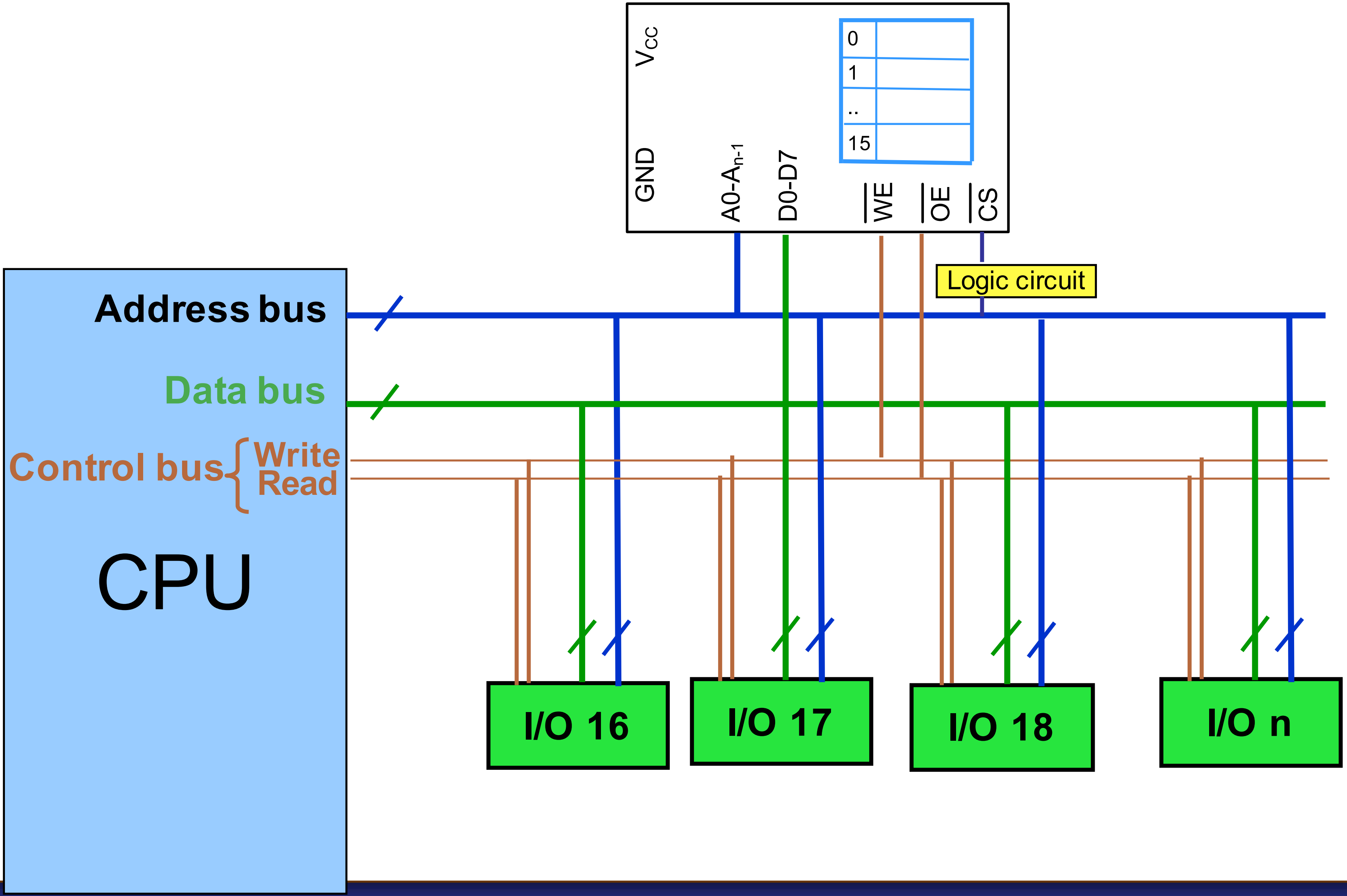
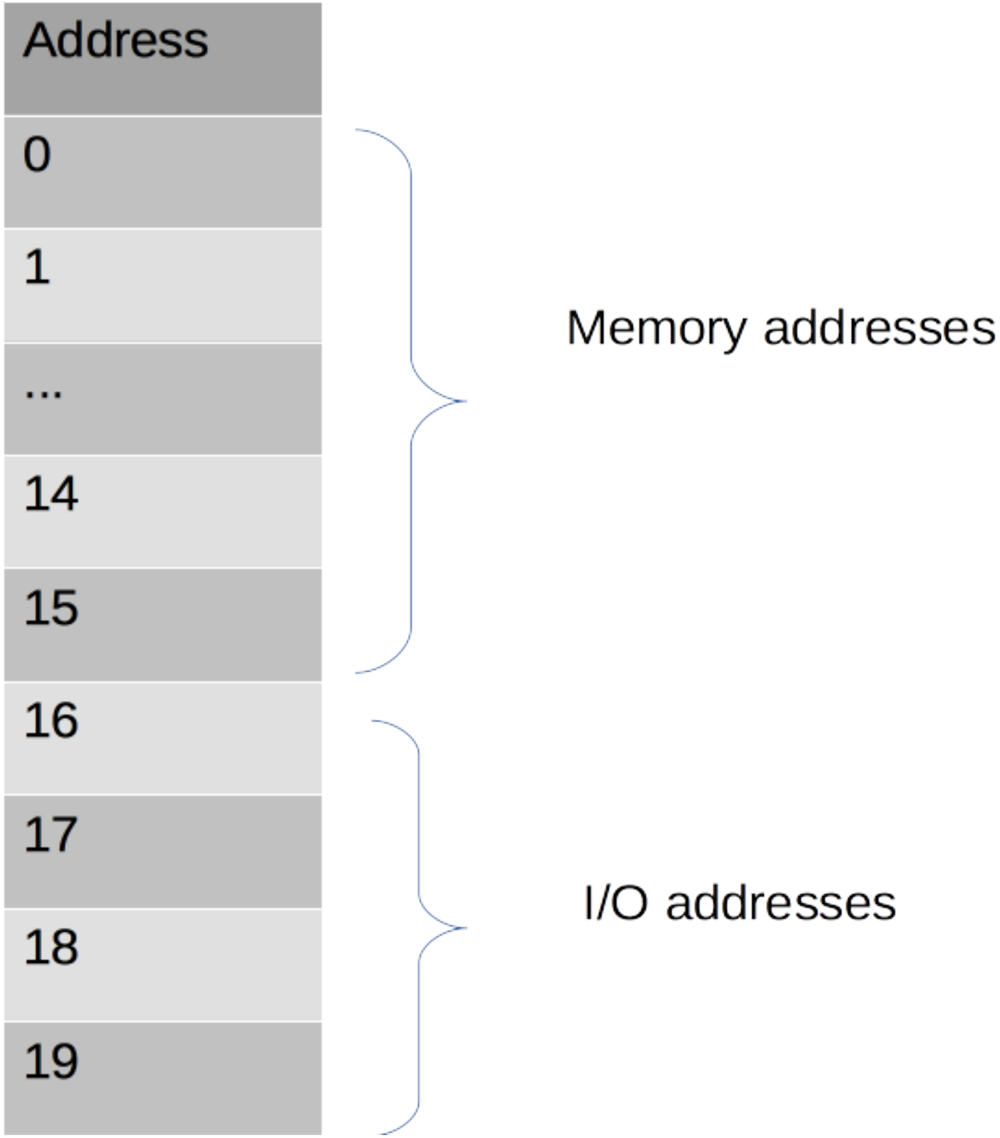
Remember this?



Memory Map

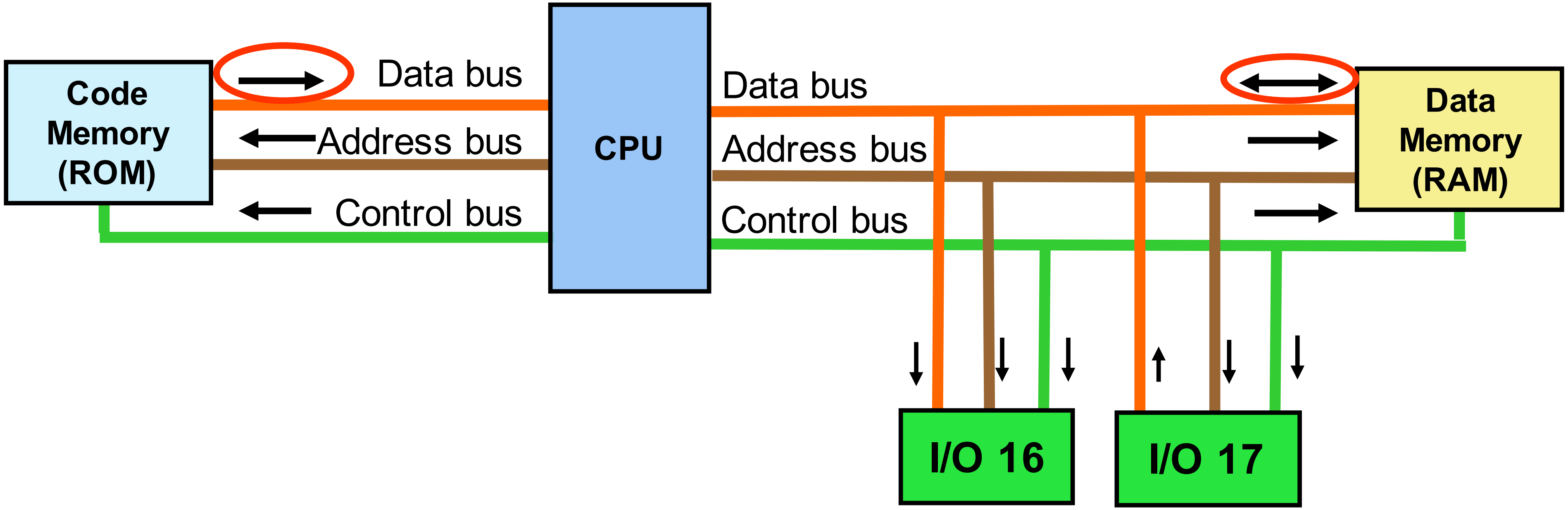


Address
0
1
...
14
15
16
17
18
19

Memory addresses

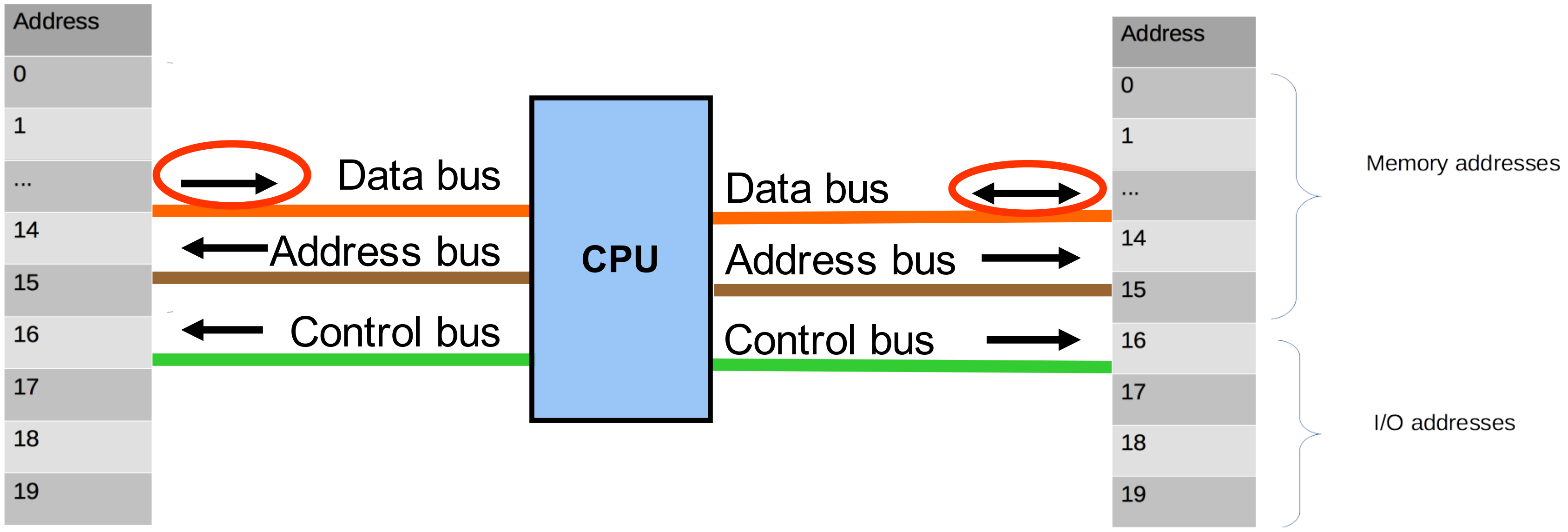
I/O addresses

Remember this? Harvard architecture



- Can fetch next instruction while executing previous one
- Simple pipelining
- Used in most small micros including the ATmega328p

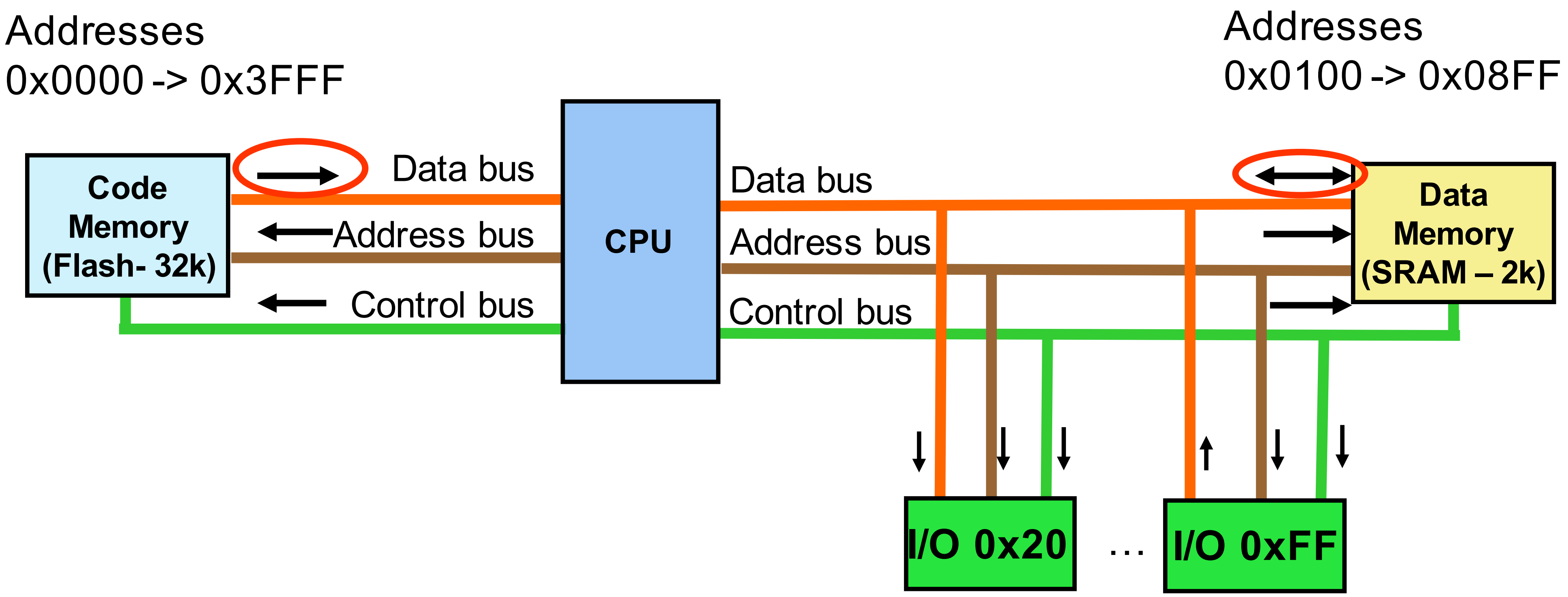
Memory Map Harvard architecture



Address
0
1
…
14
15
16
17
18
19

Address
0
1
…
14
15
16
17
18
19

AVRmega328p



Program Memory

- Atmel call the Code Memory space "Program Memory"
- Addresses 0x0000 -> 0x3FFF
  - Decimal: 0 -> 16383
  - But that is only 16kBytes?
  - How do we get to the 32kBytes?
    - It is an AVRmega***32***8p after all…

Program Memory is 16bits, 2 Byte, 1 Word

	15 8	7 0
0x3FFF		
…..		
0x000A		
0x0009		
0x0008		
0x0007		
0x0006		
0x0005		
0x0004		
0x0003		
0x0002		
0x0001		
0x0000	10101010	10101010

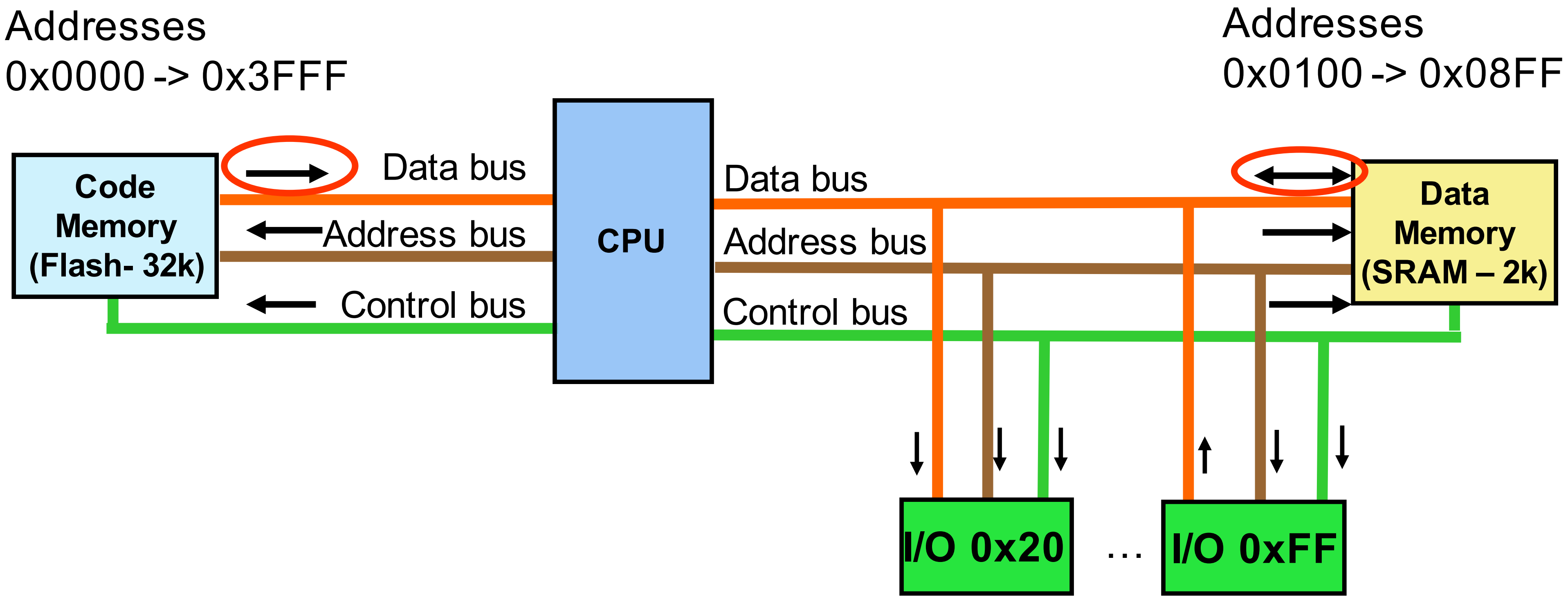
Program Memory

- Most instructions on the AVR are:
  - 16 bits, 2 bytes, 1 word

- Some instructions are:
  - 32 bits, 4 bytes, 2 words

- How big does the Instruction Register inside the AVR CPU have to be?

AVRmega328p



How wide does the Program Memory ***Data Bus*** have to be?

How wide does the Program Memory ***Address Bus*** have to be?

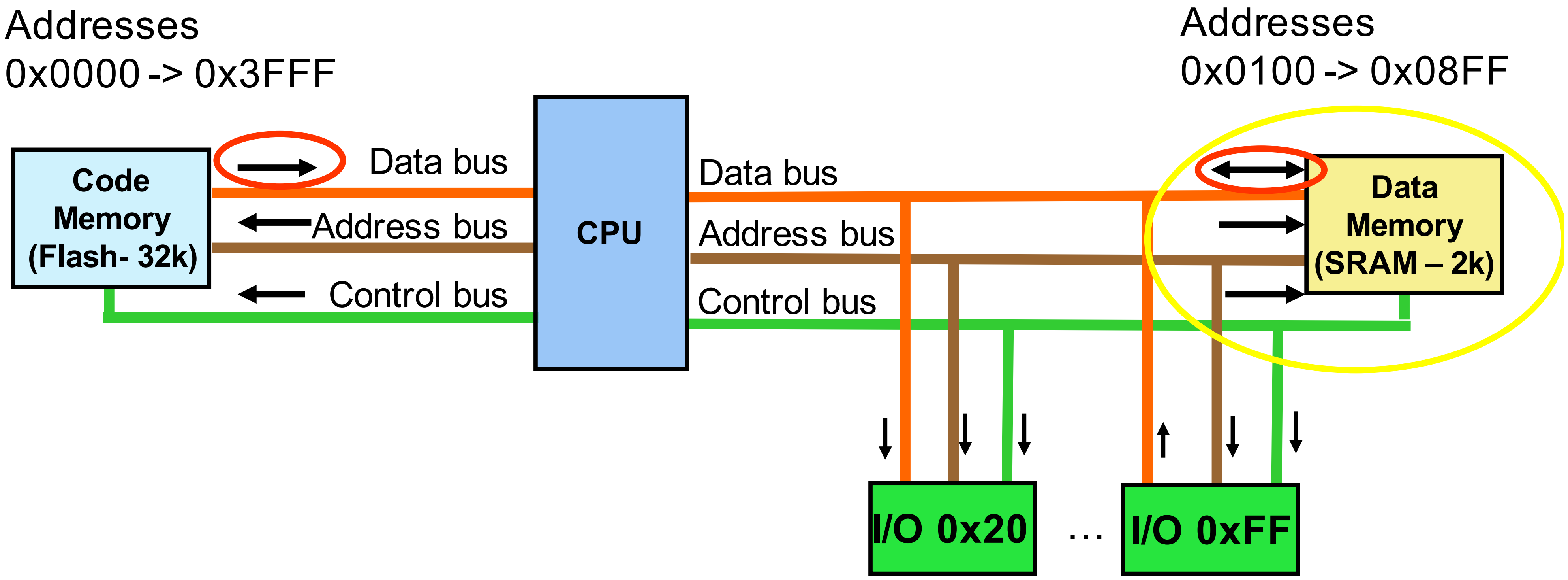
Data Memory

- Addresses 0x0000 -> 0x08FF
- Decimal: 0 -> 2303

- Data Memory ***Data Bus*** is 8 bits, 1 byte wide.

- How “wide” does the Data Memory ***Address Bus*** have to be?

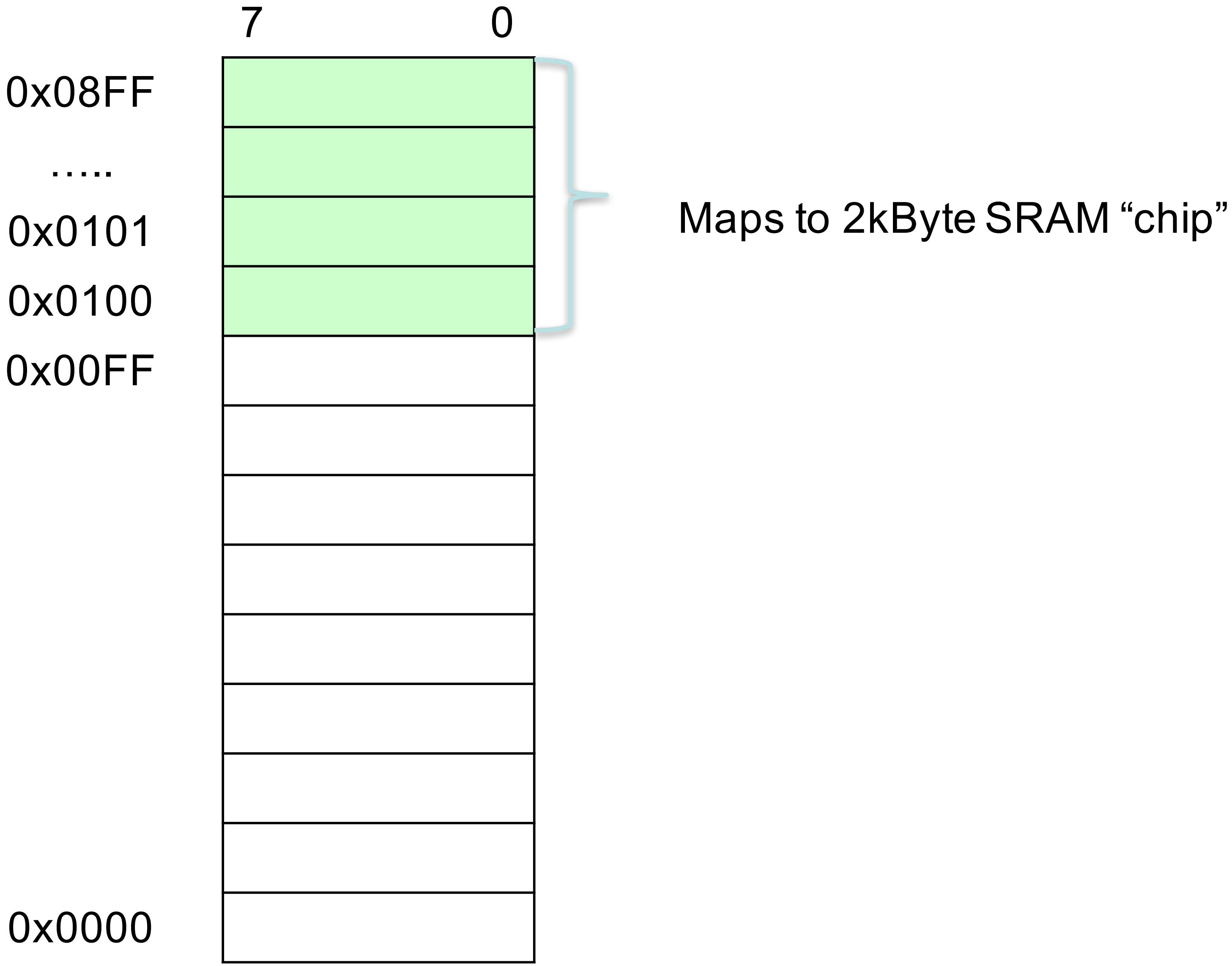
Data Memory SRAM “Chip”



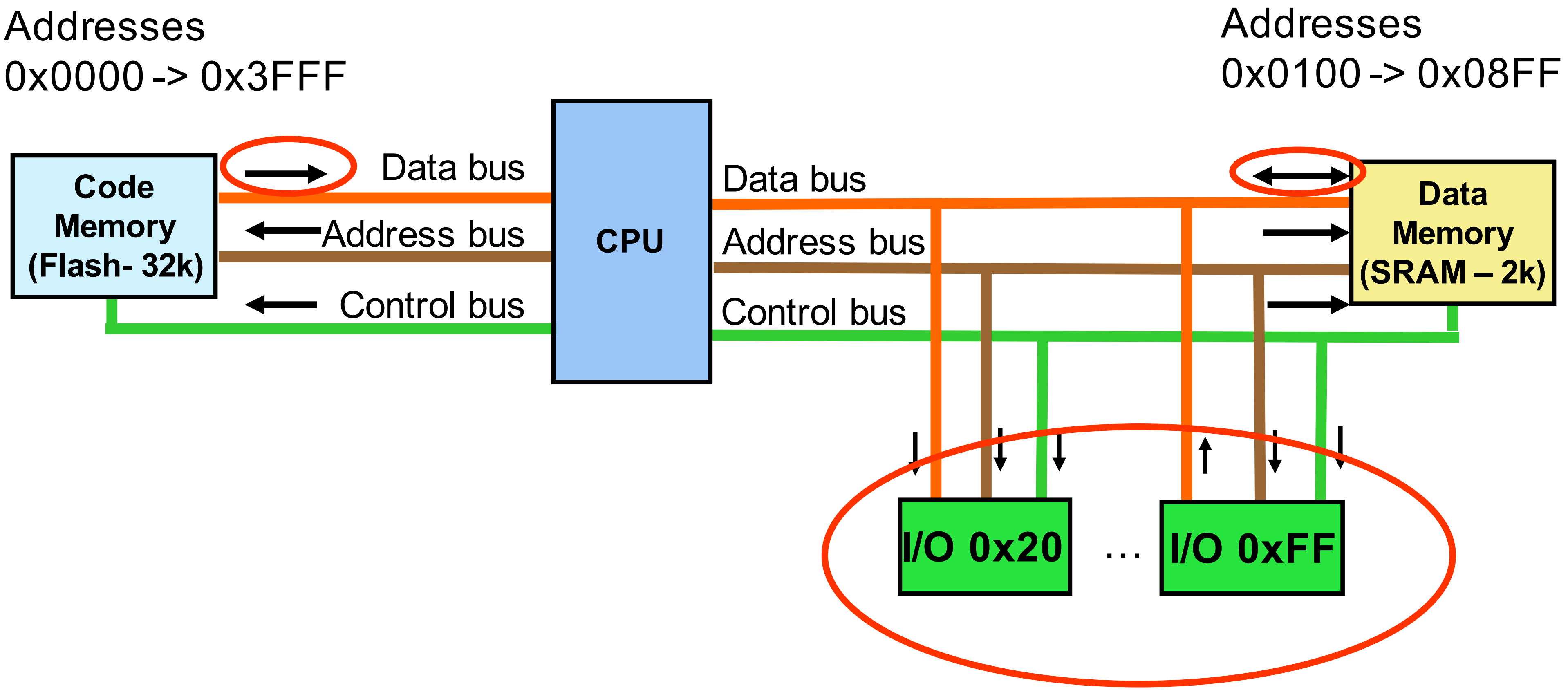
Data Memory SRAM “Chip”

- 2 kByte SRAM chip
  - – Addresses 0x0100 -> 0x08FF
  - – Decimal: 256 -> 2303
- 2303 – 256 = 2047
  - 2048 Locations = 2 KBytes. (starts at zero!)

Data Memory Mapping



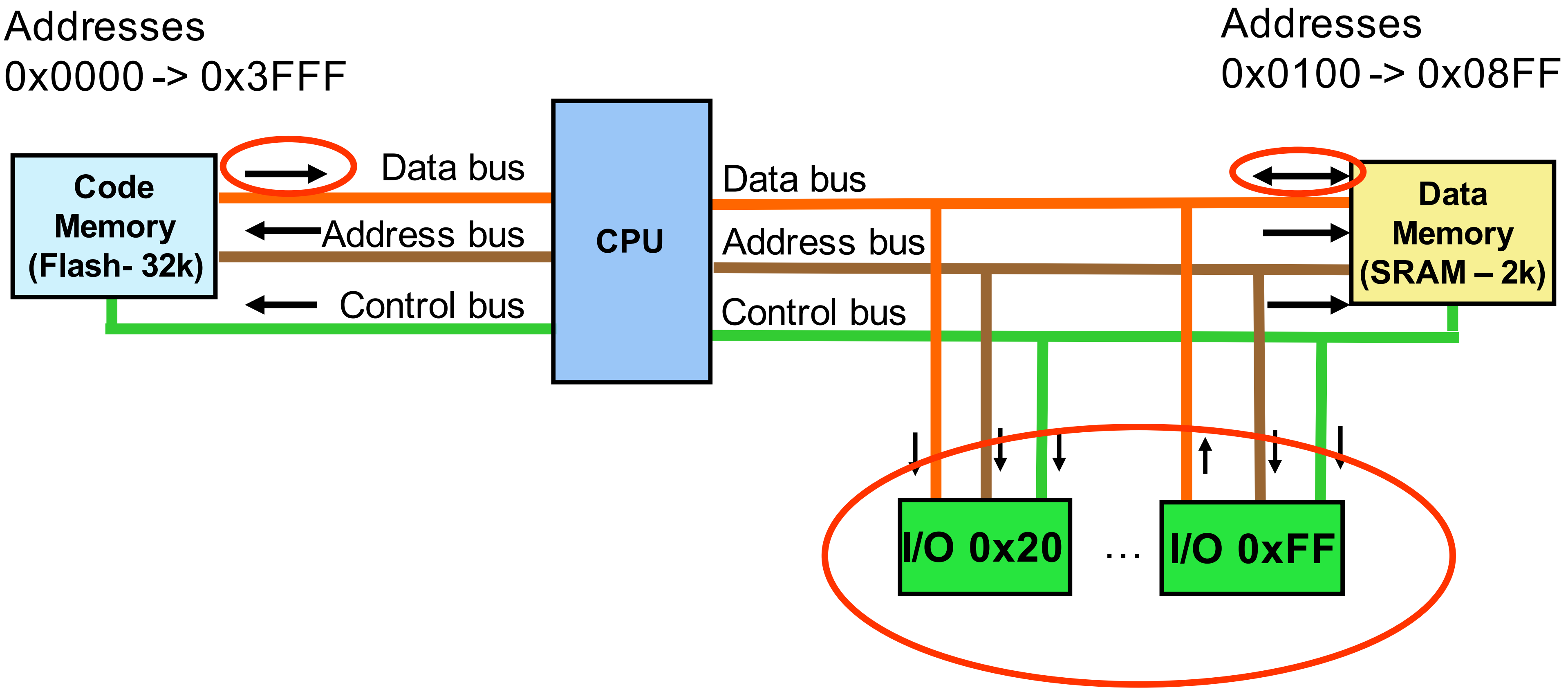
Data Memory I/O Mapping



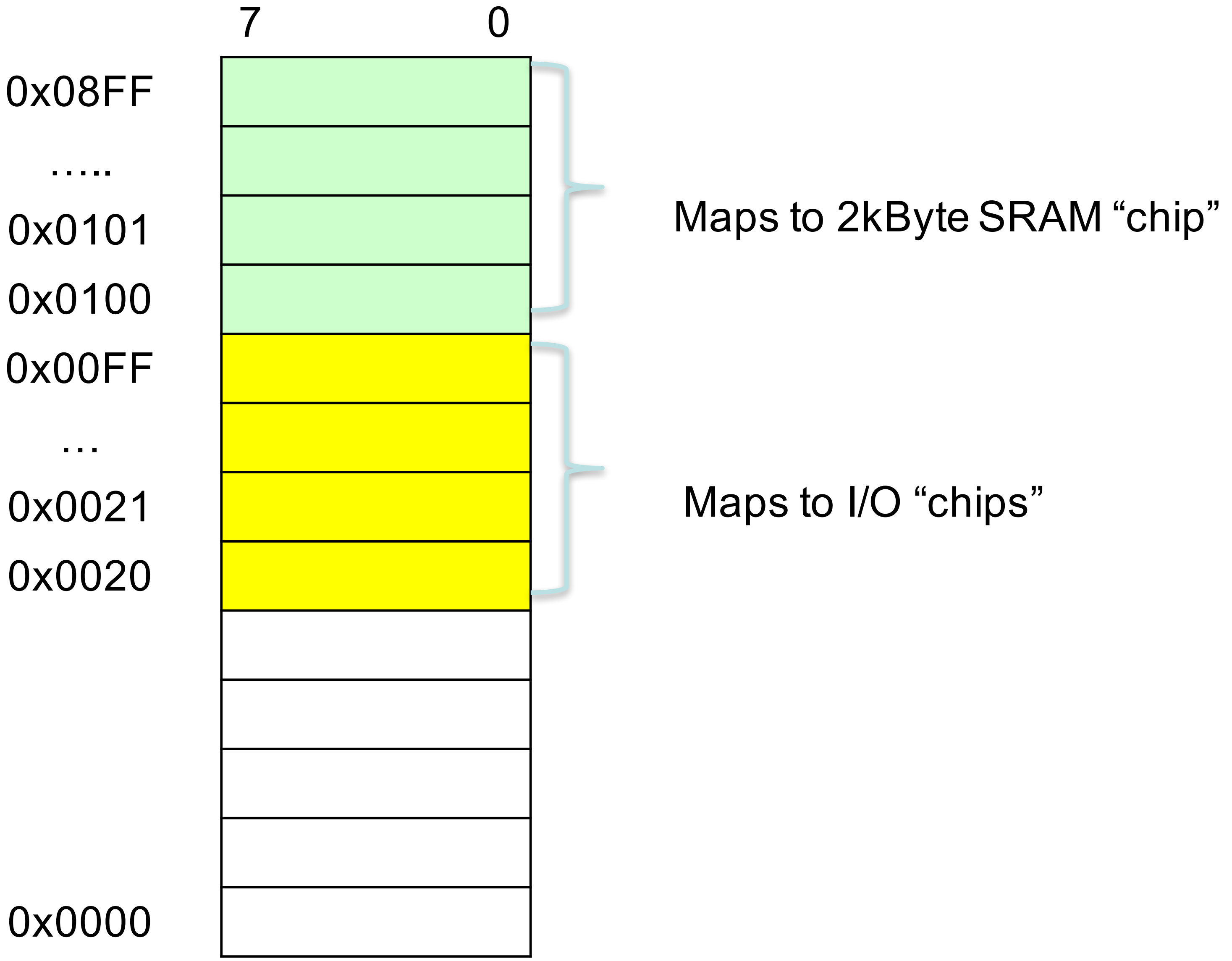
Data Memory I/O Mapping

- All I/O devices are "Memory Mapped" into the Data Memory address space.
- Basically means they all "hang" off the same address and data bus as the memory "chip"
- Have to share the "memory space" (address range) with the memory "chip"

Data Memory I/O Mapping



Data Memory Map

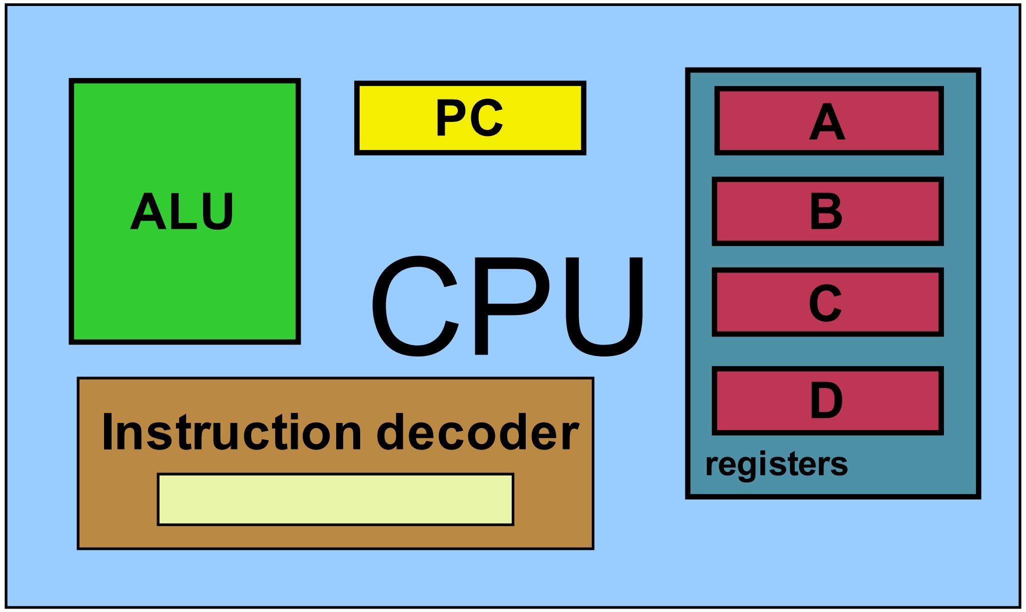


Registers…

- What about a
  - Addresses: 0x0000 -> 0x001F
  - Decimal: 0 -> 31
- This type of CPU is a RISC architecture Register File based architecture – More in Computer Architecture Module!
- Basically means there are no General Purpose registers “inside” the CPU!

Inside the CPU

- PC (Program Counter)
- Instruction decoder
- ALU (Arithmetic Logic Unit)
- Registers??

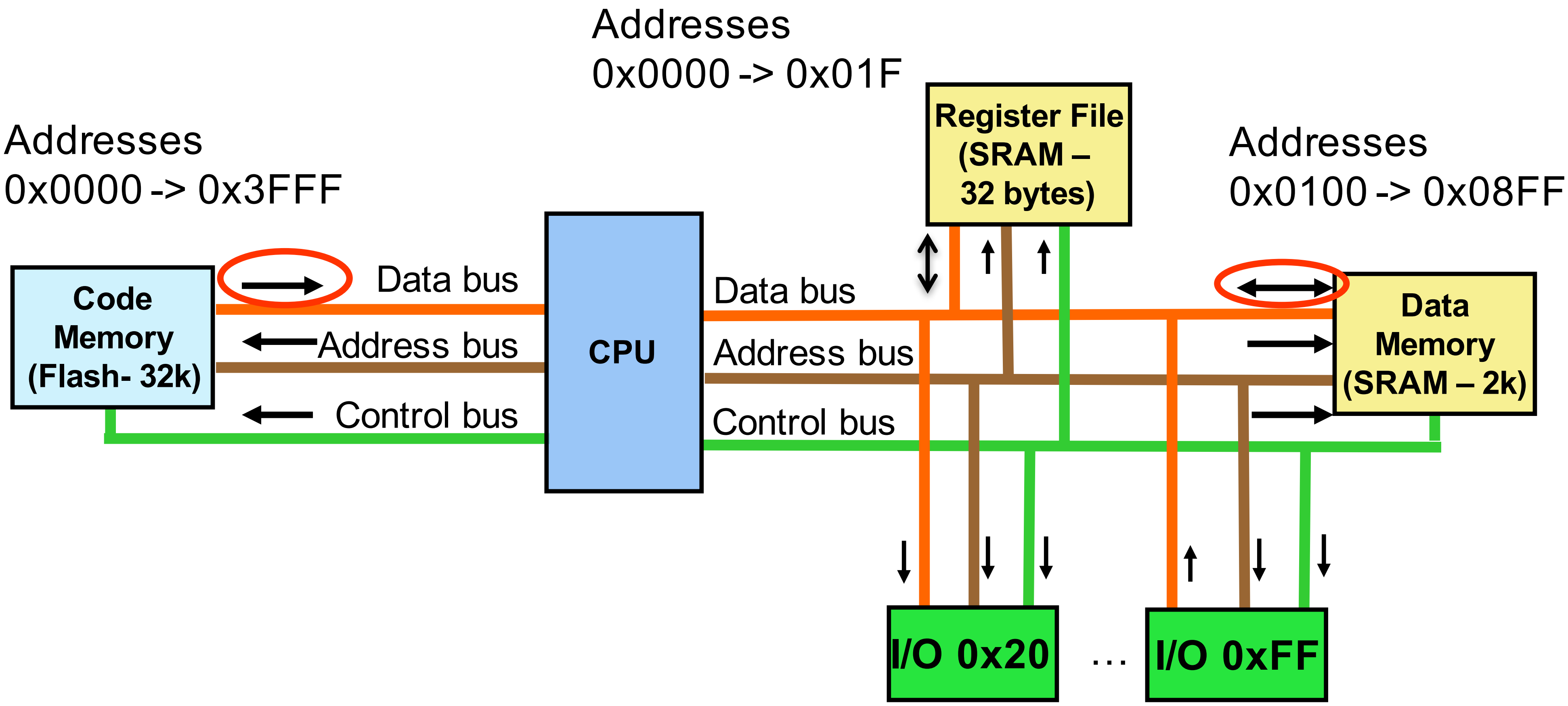


Registers

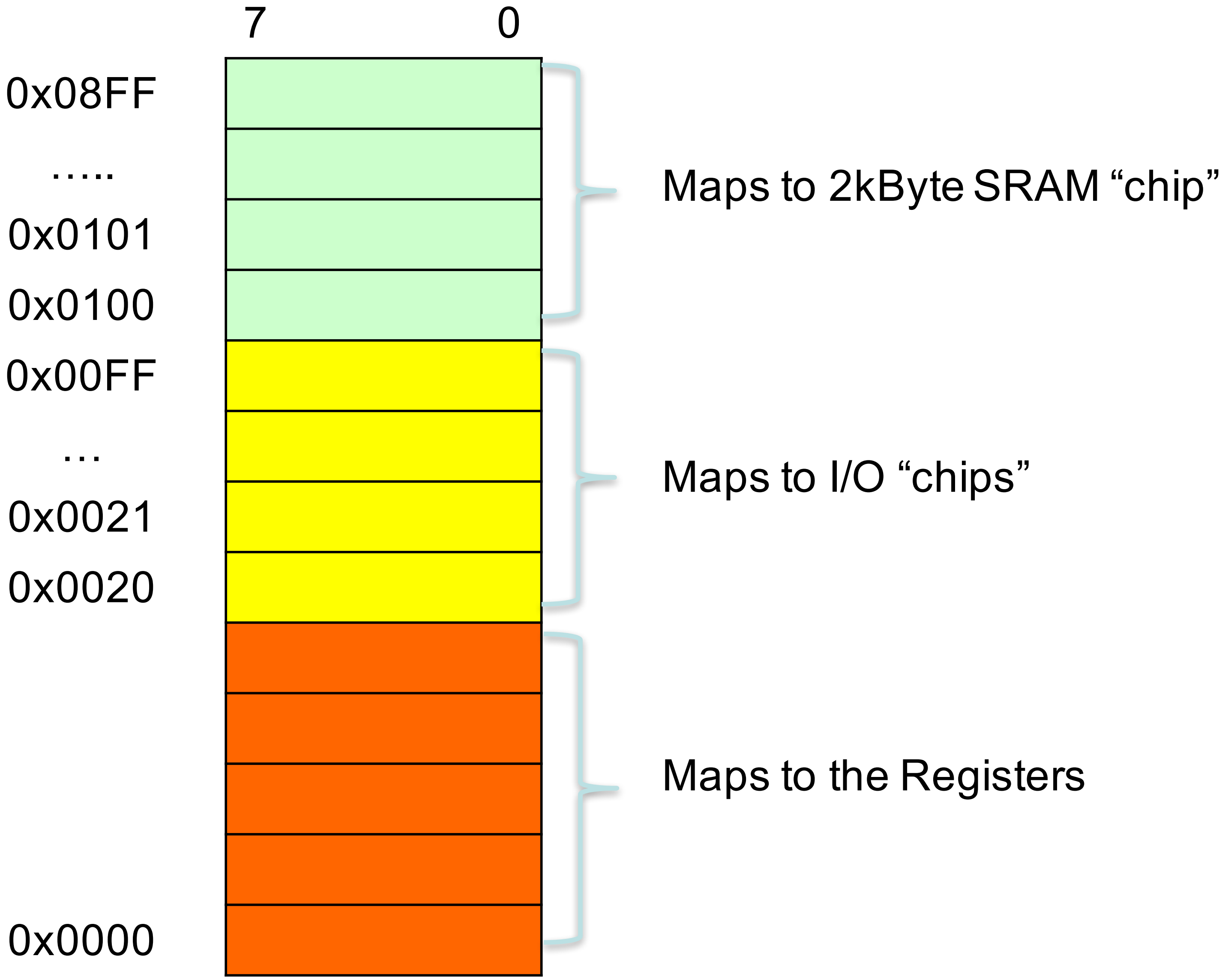
- The CPU registers are in their own little block of super high-speed SRAM called the:
  - "Register File"

- You can think of the registers as actually being another little memory "chip" on the address and data busses in Data memory.

Register File



Data Memory Map



7

0

0x08FF

…..

0x0101

0x0100

0x00FF

…

0x0021

0x0020

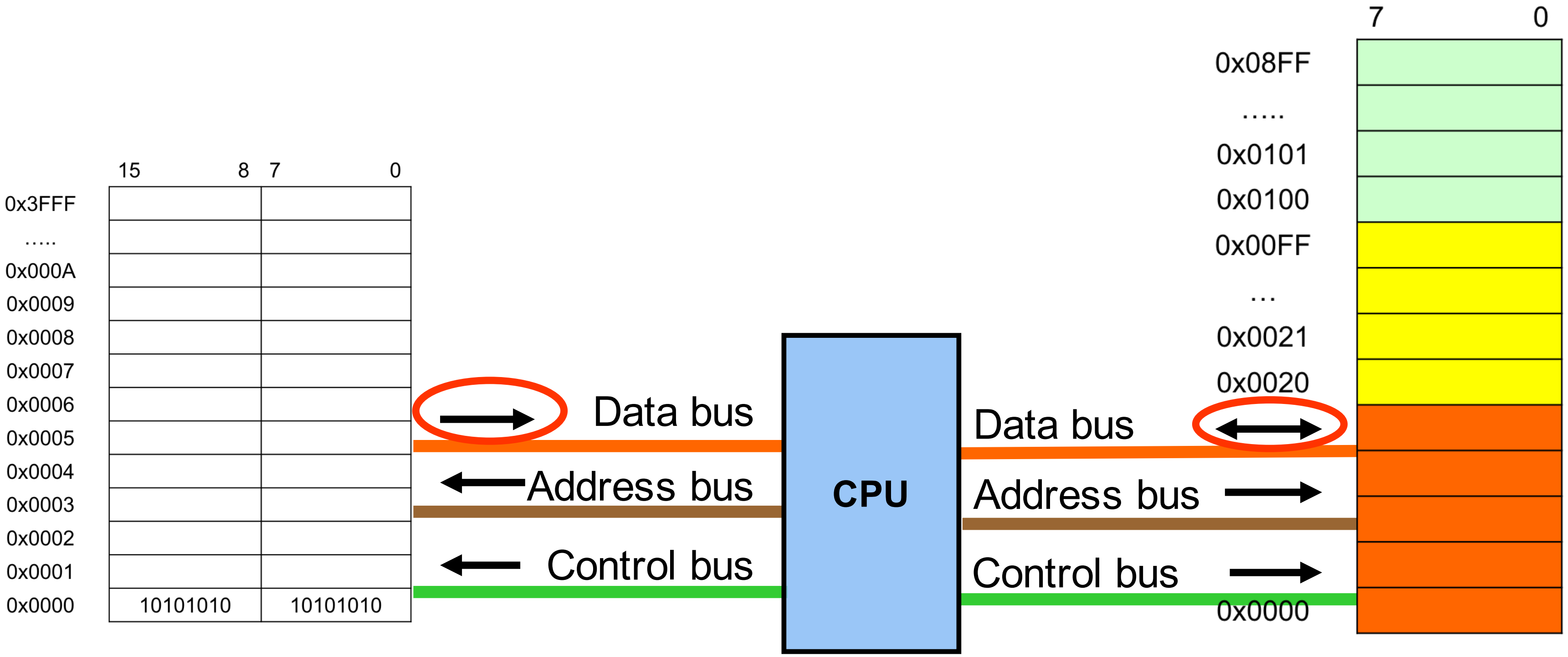
0x0000

Maps to 2kByte SRAM “chip”

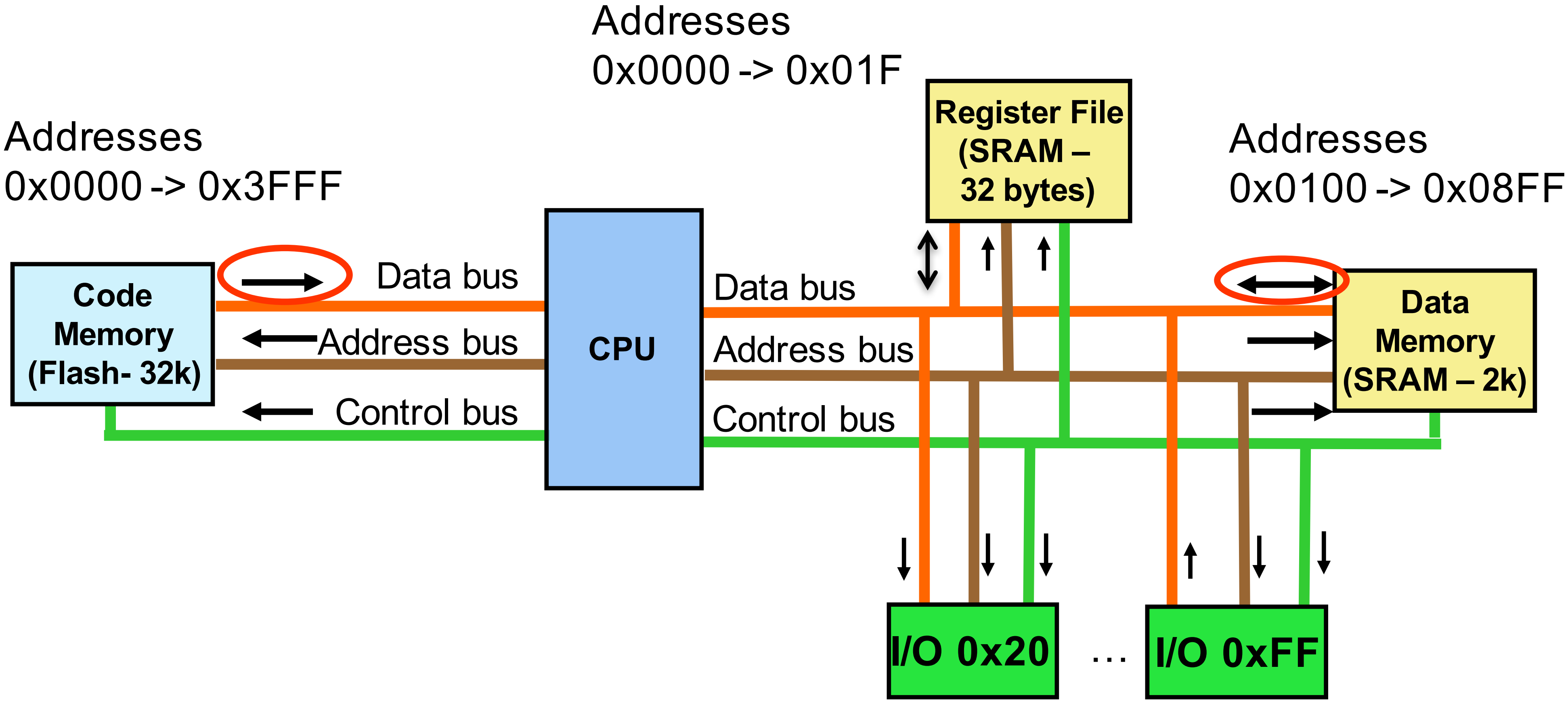
Maps to I/O “chips”

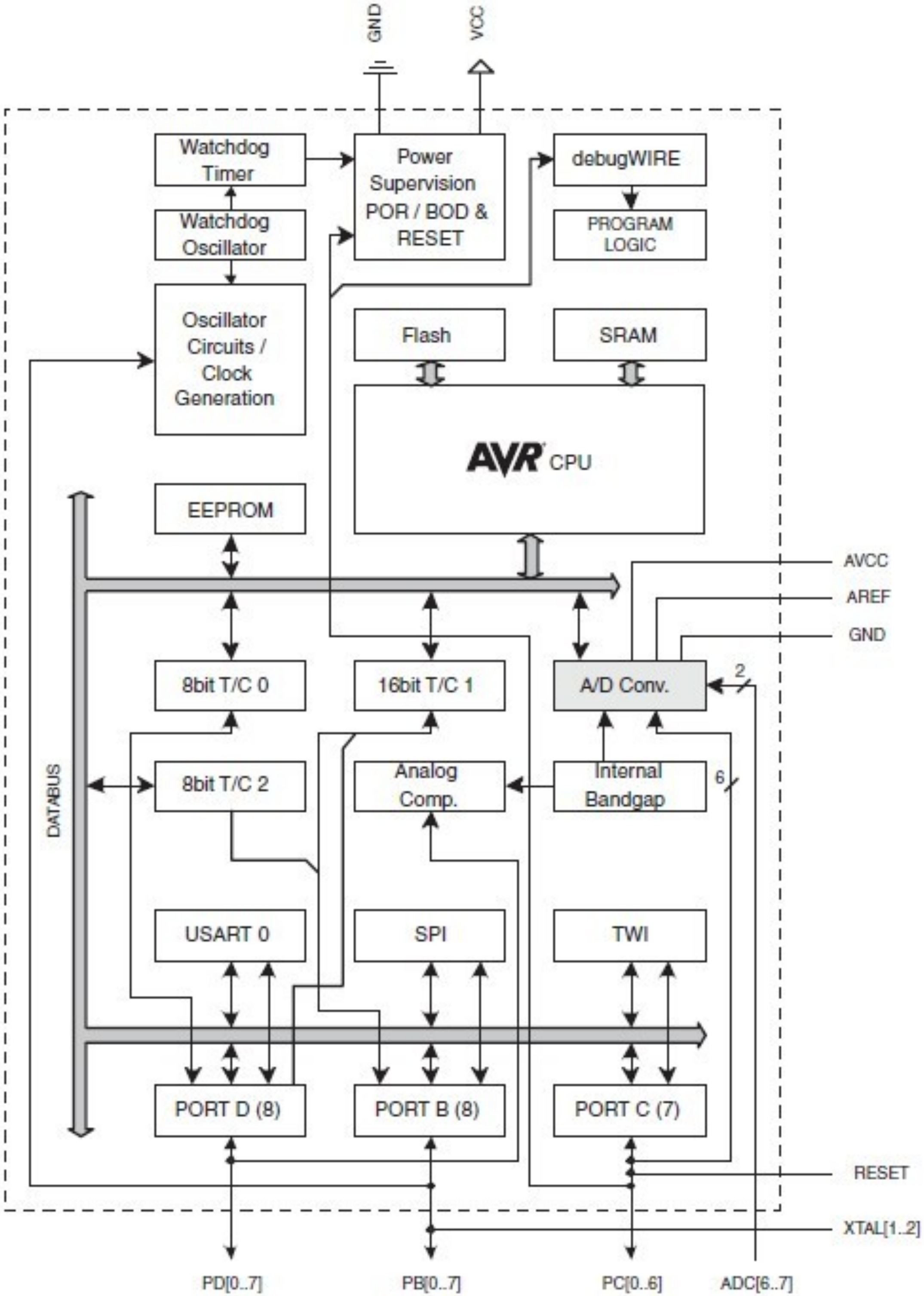
Maps to the Registers

atmega328p architecture



Atmega328p simplified Block Diagram




GND
VCC
Watchdog Timer
Power Supervision POR / BOD & RESET
debugWIRE
Watchdog Oscillator
PROGRAM LOGIC
Oscillator Circuits / Clock Generation
Flash
SRAM
AVR CPU
EEPROM
AVCC
AREF
GND
DATABUS
8bit T/C 0
16bit T/C 1
A/D Conv.
2
8bit T/C 2
Analog Comp.
Internal Bandgap
6
USART 0
SPI
TWI
PORT D (8)
PORT B (8)
PORT C (7)
RESET
XTAL[1..2]
PD[0..7]
PB[0..7]
PC[0..6]
ADC[6..7]