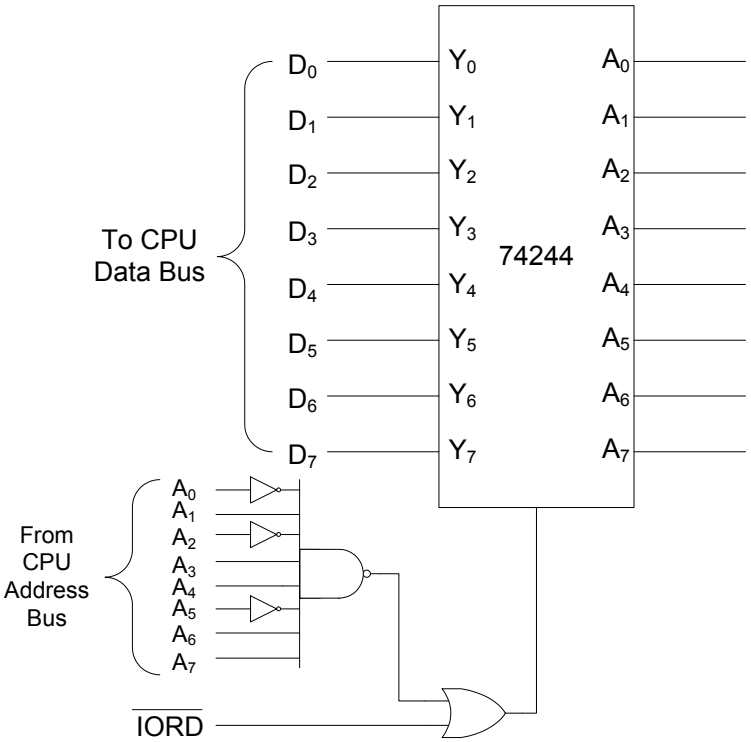
Microprocessors, Lecture 9

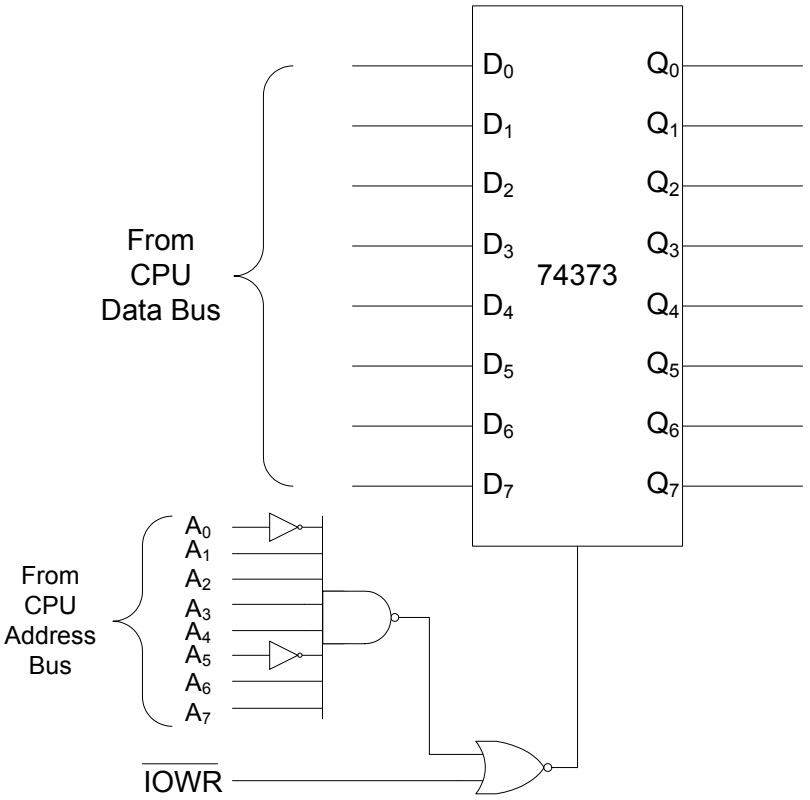
AVR Basic I/O

Hamid Fadishei
Assistant Professor, University of Bojnord
Spring 2015

Microprocessor I/O

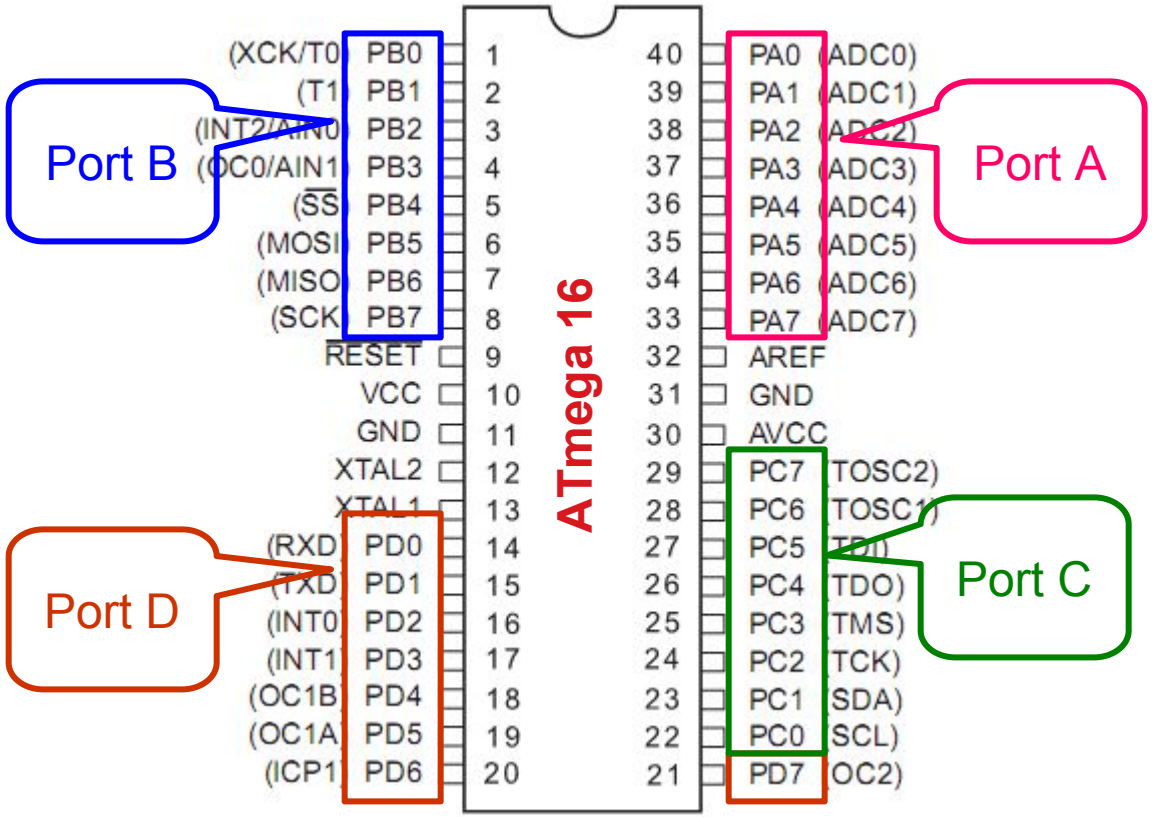
- In microprocessor systems, we had to add I/O ports separately



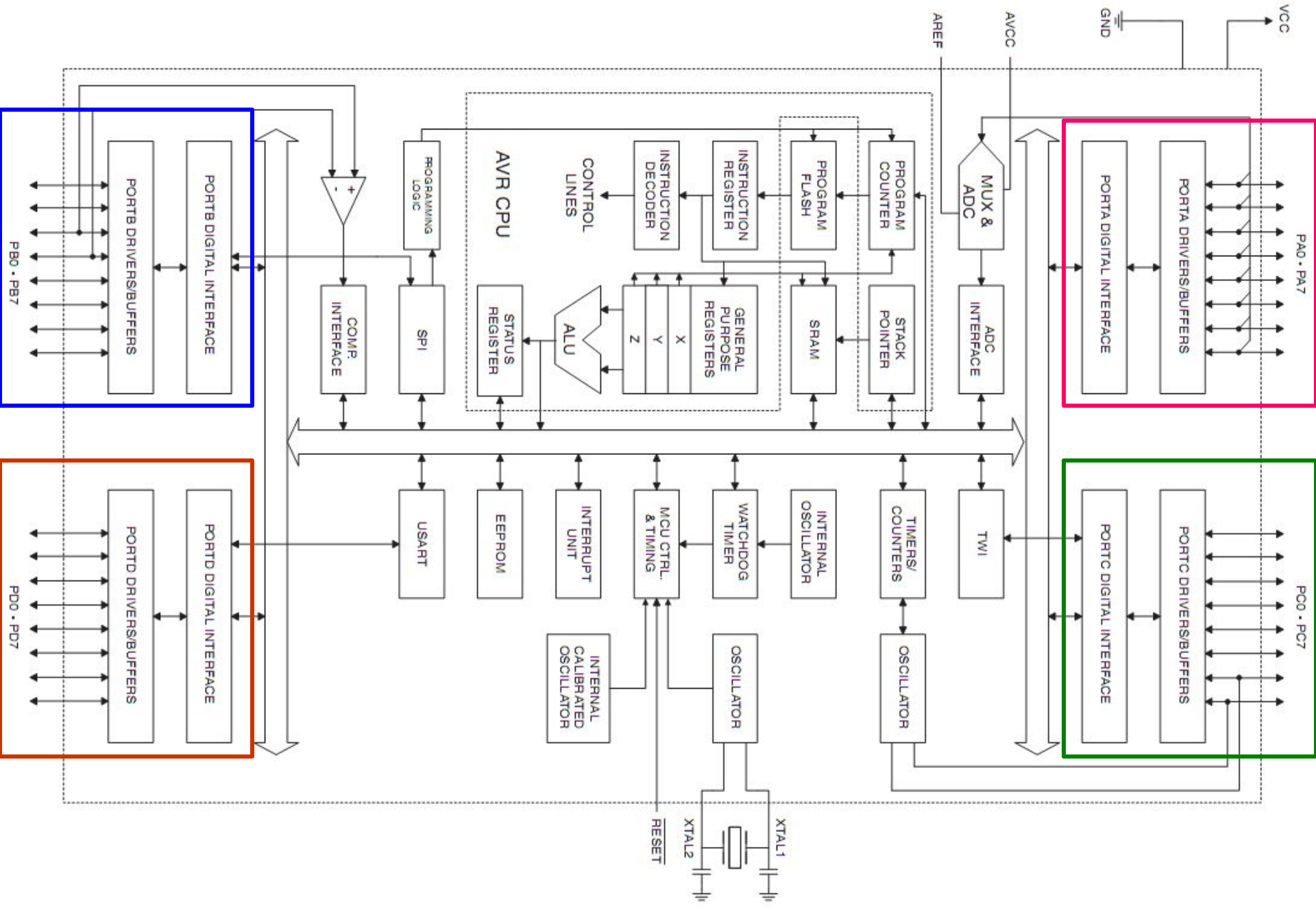


AVR I/O Ports

- A microcontroller already has some I/O ports
  - Example: ATmega16 has 4 ports
  - Refered to as GPIO (General-Purpose Input/Output)

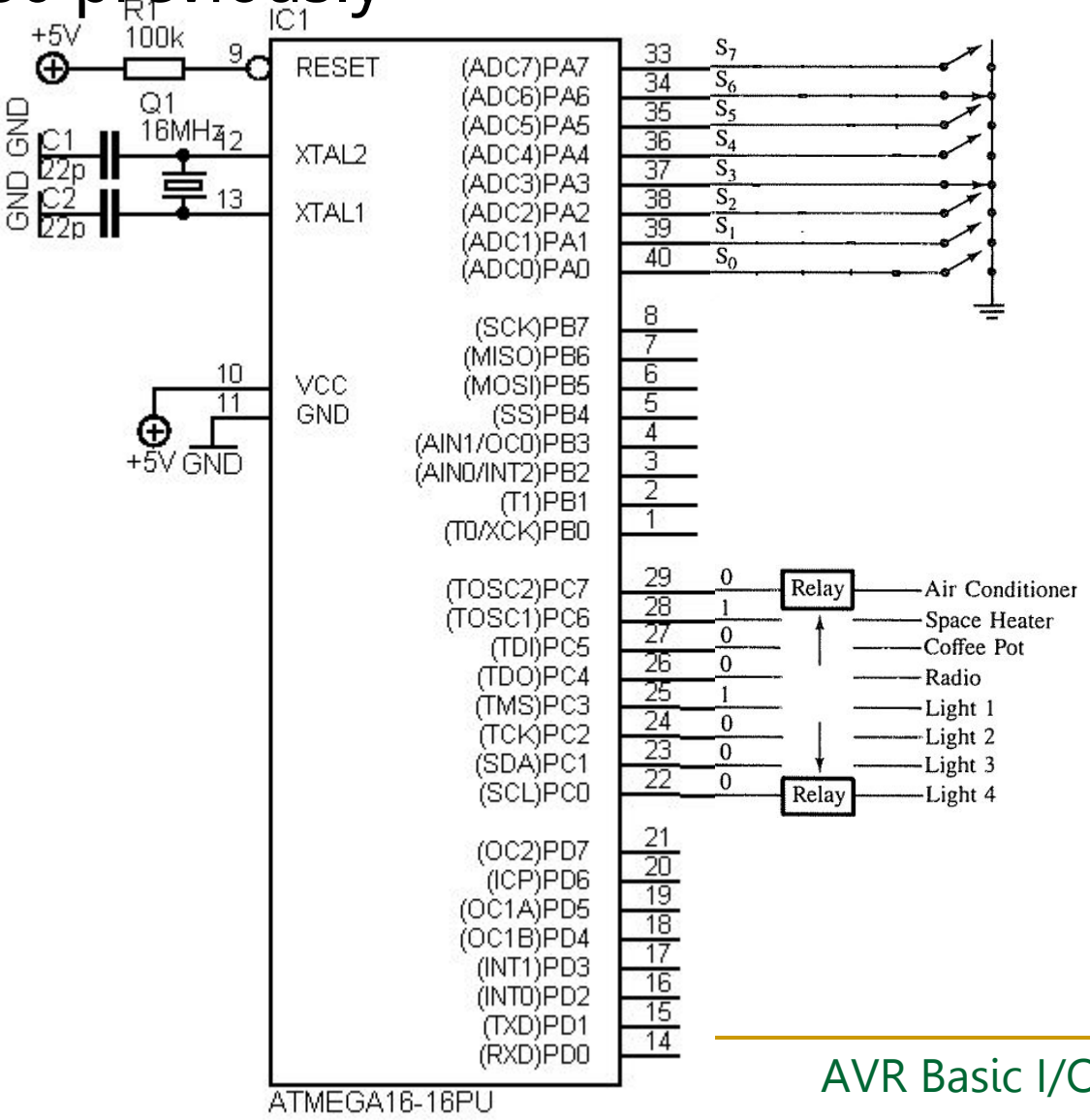


AVR I/O Ports


VCC
GND
AREF
AVCC
PA0 - PA7
PORTA DRIVERS/BUFFERS
PORTA DIGITAL INTERFACE
MUX & ADC
ADC INTERFACE
PROGRAM COUNTER
PROGRAM FLASH
INSTRUCTION REGISTER
INSTRUCTION DECODER
CONTROL LINES
AVR CPU
PROGRAMMING LOGIC
STACK POINTER
SRAM
GENERAL PURPOSE REGISTERS
X
Y
Z
ALU
STATUS REGISTER
COMP. INTERFACE
SPI
PB0 - PB7
PORTB DRIVERS/BUFFERS
PORTB DIGITAL INTERFACE
PC0 - PC7
PORTC DRIVERS/BUFFERS
PORTC DIGITAL INTERFACE
TWI
TIMERS/ COUNTERS
INTERNAL OSCILLATOR
WATCHDOG TIMER
MCU CTRL. & TIMING
INTERRUPT UNIT
EEPROM
USART
PD0 - PD7
PORTD DRIVERS/BUFFERS
PORTD DIGITAL INTERFACE
OSCILLATOR
OSCILLATOR
INTERNAL CALIBRATED OSCILLATOR
RESET
XTAL2
XTAL1

AVR I/O Simple Example - Hardware

- You saw it for Z80 previously



AVR I/O Simple Example - Software

- Each GPIO port has 3 I/O registers
  - Direction register (DDRA, DDRB, etc)
    - Used to configure port bits as input or output
    - Note: GPIO ports are designed to be bidirectional and their direction can be chosen by software
  - Port input register (PINA, PINB, etc)
    - Used to read the port when configured as input
  - Port output register (PORTA, PORTB, etc)
    - Used to write to port when configured as output

AVR I/O Simple Example - Software

- Pseudocode

```
1. Configure port A as input
2. Enable internal pullups on port A
3. Configure port C as output
4. Set port C to all 0's
5. While (true)
       read from port A
       complement the read value
       write the result to port C
```

AVR I/O Simple Example - Software

- Using Assembly

```
#include <m16def.inc>

.org 0

; define interrupt vectors
vects:
rjmp RESET

RESET:
ldi R16, 0X00    ; load register 16 with zero
ldi R17, 0XFF    ; load register 17 with all 1's
out DDRA, r16    ; set port A to input
out PORTA, r17   ; set pullups on port A inputs
out DDRC, R17    ; set port C to output
out PORTC, r16   ; write 0 to output port C

LOOP:            ; infinite loop
in r16, PINA     ; read port A
COM r16          ; complement the read value
out PORTC, r16   ; output to port C
rjmp LOOP        ; repeat loop
```

AVR I/O Simple Example - Software

- Using C

```
#include <avr/io.h>

int main(void)
{
    DDRA=0x00;  // set port A to input
    PORTA=0xFF; // set pullups
    DDRC=0xFF;  // set port C to output
    PORTC=0x00; // write 0 to port C
    while (1)
    {
        // read from port A and write
        // the complement of the read
        // value to port C
        PORTC = ~PINA;
    }
}
```

AVR I/O Simple Example - Simulation

- Using Atmel Studio