AVR: Digital input-output

Dr Asimakis Leros
University of the Aegean

Summary

- 23 pins available for digital I/O
  - Port B (PB0-PB7)
  - Port C (PC0-PC6)
  - Port D (PD0-PD7)
- Each pin can be individually used as input or output and be read or written independently of the rest. Five pins of the above are reserved on the Arduino.

Registers used by I/O ports

- Each pin individually programmed using three bits of registers DDRx, PORTx, PINx, where x = B, C, D.
- For example, pin PB5 is programmed by bit 5 of DDRB, PORTB, PINB.
- All pins have alternate functions, such as analog inputs, interrupt sources, reset etc. These functions are selected and programmed by other special registers.

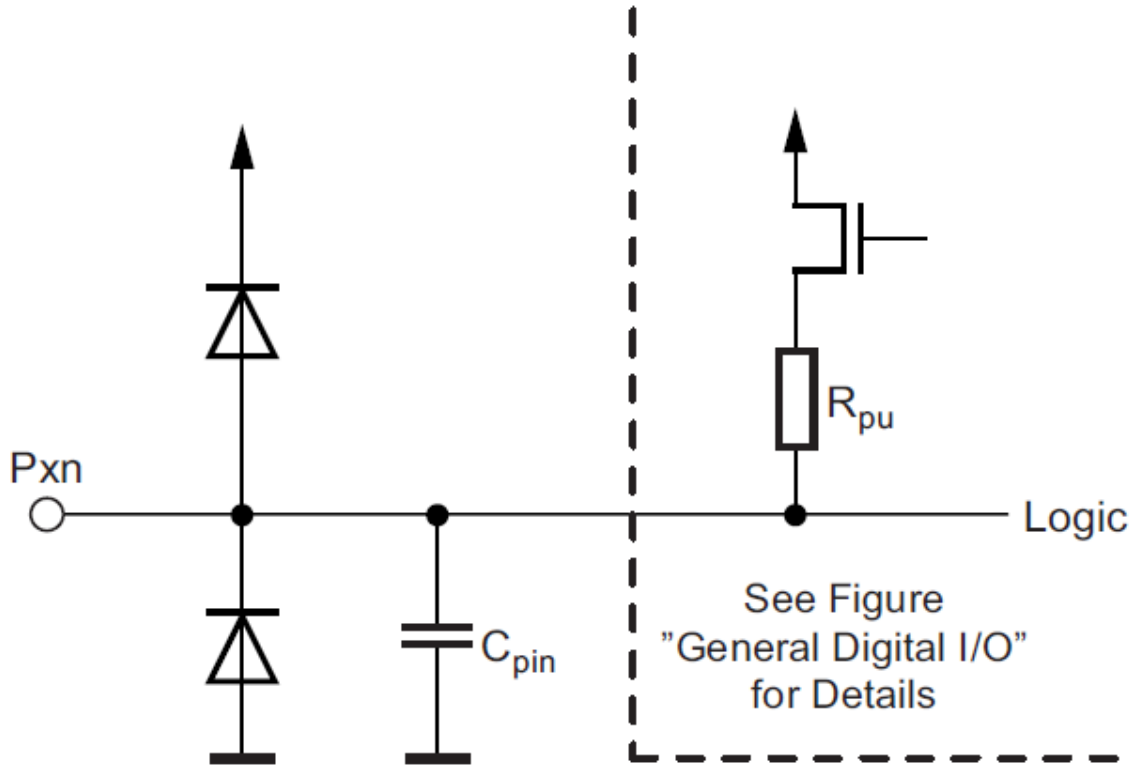
Data direction register: DDRx

- DDRB (data direction register B):
  - Writing bit DDRBn = 0, sets PBn as input.
  - Writing DDRBn = 1, sets PBn as output.
- Same for DDRC and DDRD.

Port register: PORTx

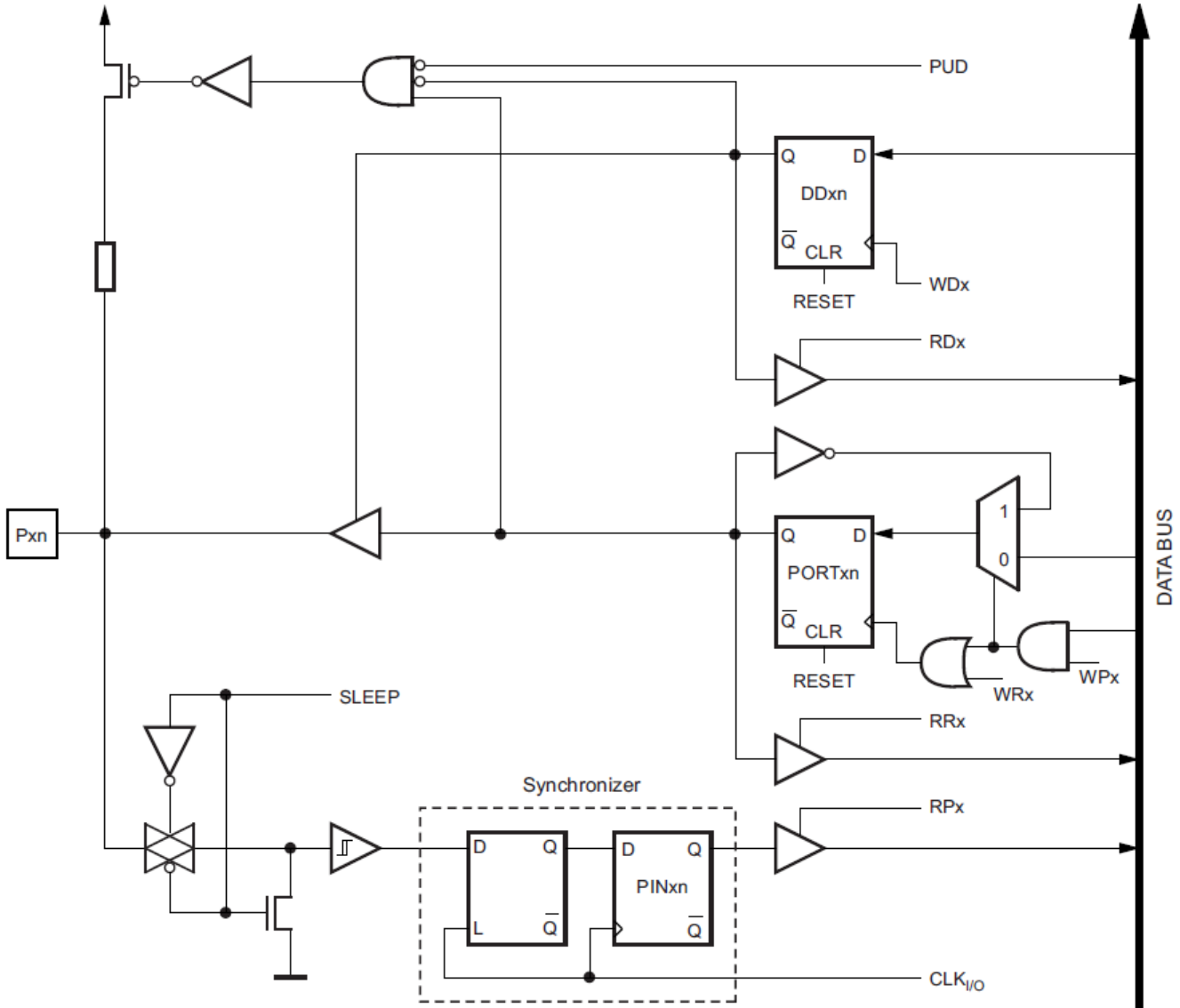
- PORTB (data register / buffer B): read/write
  - If DDRBn = 1 (output), then writing PORTBn = 0 brings PBn to low level (logical 0). Writing PORTBn = 1 brings PBn to logical 1.
  - If DDRBn = 0 (input), the buffer contents are preserved until they are overwritten.
  - While DDRBn = 0 (input), setting PORTBn = 1 activates the pull-up resistor for this pin, while setting PORTBn = 0 sets the pin in a high-impedance (tri-state) mode.

Pull-up resistors



Pin register: PINx

- PINB (pin input register B): read only
  - Double buffer for synchronization purpose
  - Receives directly the value of pin PBn
  - Can be read regardless if DDRBn = 0 or 1.
  - Delay of 0.5 to 1.5 clock cycles from PBn to PINBn.
  - Writing PINBn = 1, regardless of the value of DDRBn, has the effect of inverting the value of PORTBn.



PUD:	PULLUP DISABLE	WDx:	WRITE DDRx
SLEEP:	SLEEP CONTROL	RDx:	READ DDRx
$CLK_{I/O}$:	I/O CLOCK	WRx:	WRITE PORTx
		RRx:	READ PORTx REGISTER
		RPx:	READ PORTx PIN
		WPx:	WRITE PINx REGISTER

Accessing the registers

Register	Addr in I/O space	Addr in mem space
PINB	0x03	0x23
DDRB	0x04	0x24
PORTB	0x05	0x25
PINC	0x06	0x26
DDRC	0x07	0x27
PORTC	0x08	0x28
PIND	0x09	0x29
DDRD	0x0A	0x2A
PORTD	0x0B	0x2B

Accessing the registers

- IN and OUT instructions:

```
IN  r16, PORTC      ; copy value of PORTC …
OUT PORTB, r16      ; …to PORTB
```

- LD, LDS and ST, STS instructions:

```
LDS  r16, 0x28      ; PORTC = 0x28 (SRAM)
STS  0x25, r16
```

- Using the SBI και CBI instructions to set or clear an individual bit

SBI and CBI

- SBI: set bit b of I/O register A to 1
  ($0 \le b \le 7$, $0 \le A \le 31$)

  SBI A, b

- CBI: clear bit b of I/O register A to 0
  ($0 \le b \le 7$, $0 \le A \le 31$)

  CBI A, b

- They operate only on the first 32 out of 64 registers in I/O space.

Example: write a single pin

- The following code sets pin PB5 as output and writes a logical 1 to the pin.
- The rest bits of Port B are treated as inputs with pull-ups enabled (good practice for unused pins).

```
ldi r16, 0b00100000      ; PB5 output
out DDRB, r16
ldi r16, 0b11111111      ; PB5 high
out PORTB, r16
```

Blinking a pin: SBI, CBI

```
        ldi r16, 0b00100000
        out DDRB, r16
loop:
        sbi PORTB, 5 ; Set bit 5 of port B to 1
        nop
        cbi PORTB, 5 ; Set bit to 0
        nop
rjmp loop
```

Blinking a pin by using the PINx register

```
        ldi r16, 0b00100000
        out DDRB, r16
        ldi r16, 0b11111111
        out PORTB, r16
        ldi r16, 0b00100000
loop:
        out PINB, r16          ; Toggle bit 5 of port B
        nop
rjmp loop
```