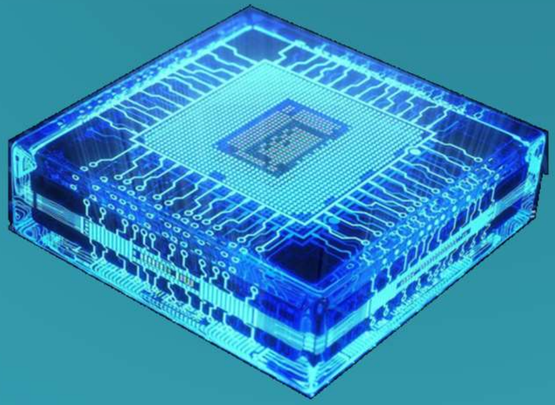
Microprocessors and Assembly language

Isfahan University of Technology (IUT)

1402



IO ports

Dr. Hamidreza Hakim

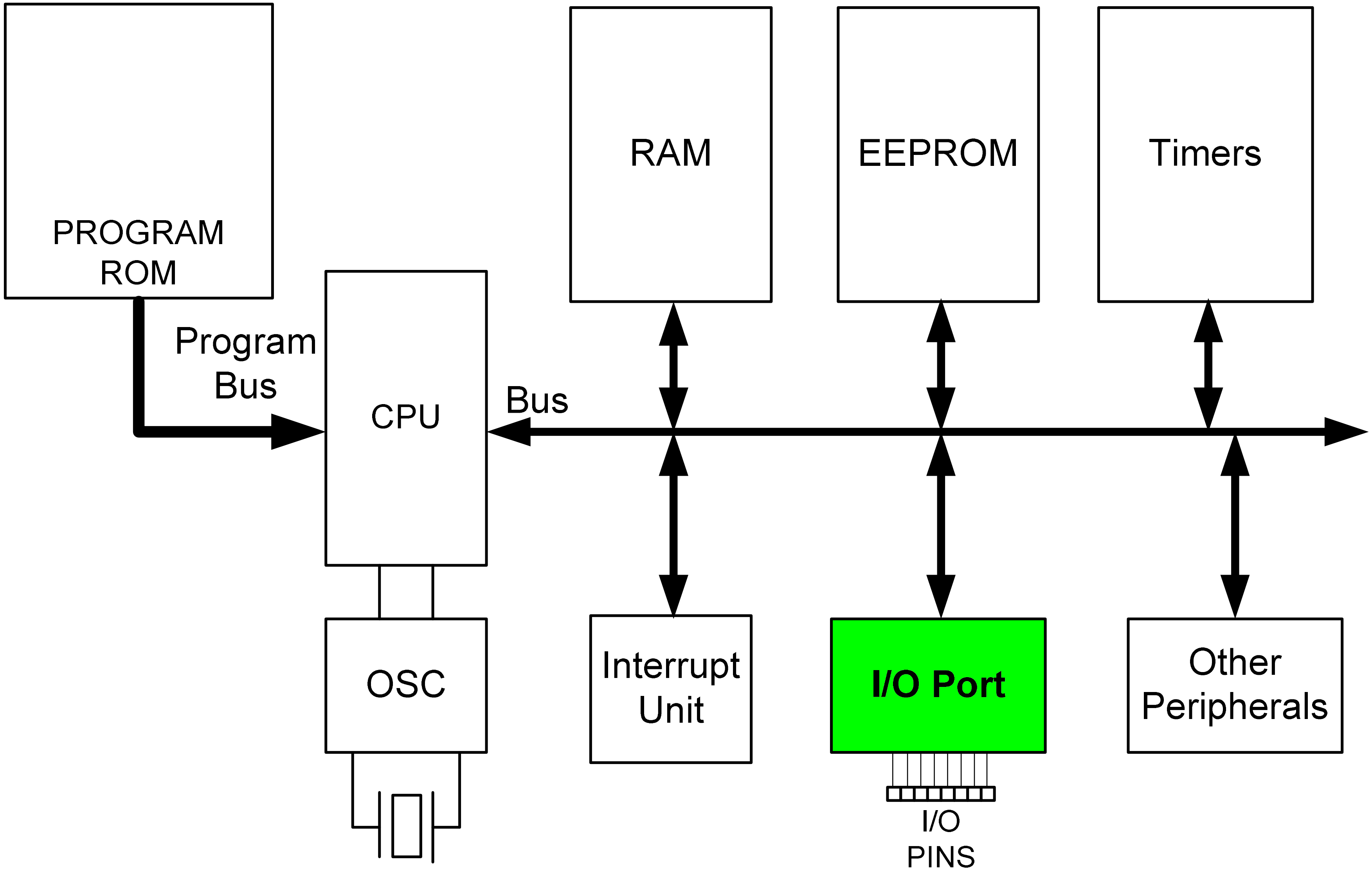
hakim@iut.ac.ir

بسم الله الرحمن الرحیم

Topics

- AVR pin out
- The structure of I/O pins
- I/O programming
- Bit manipulating

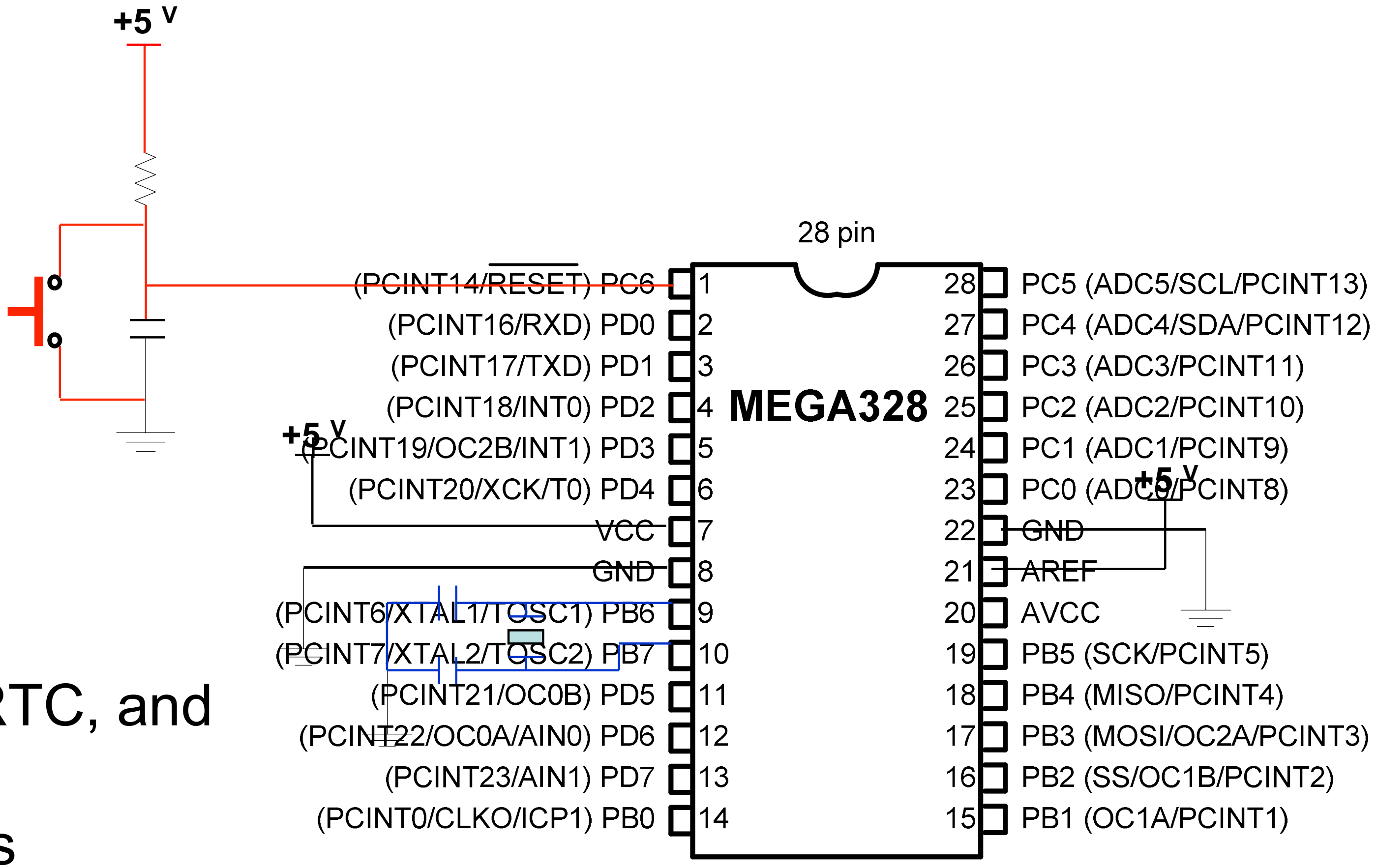
I/O unit in AVR



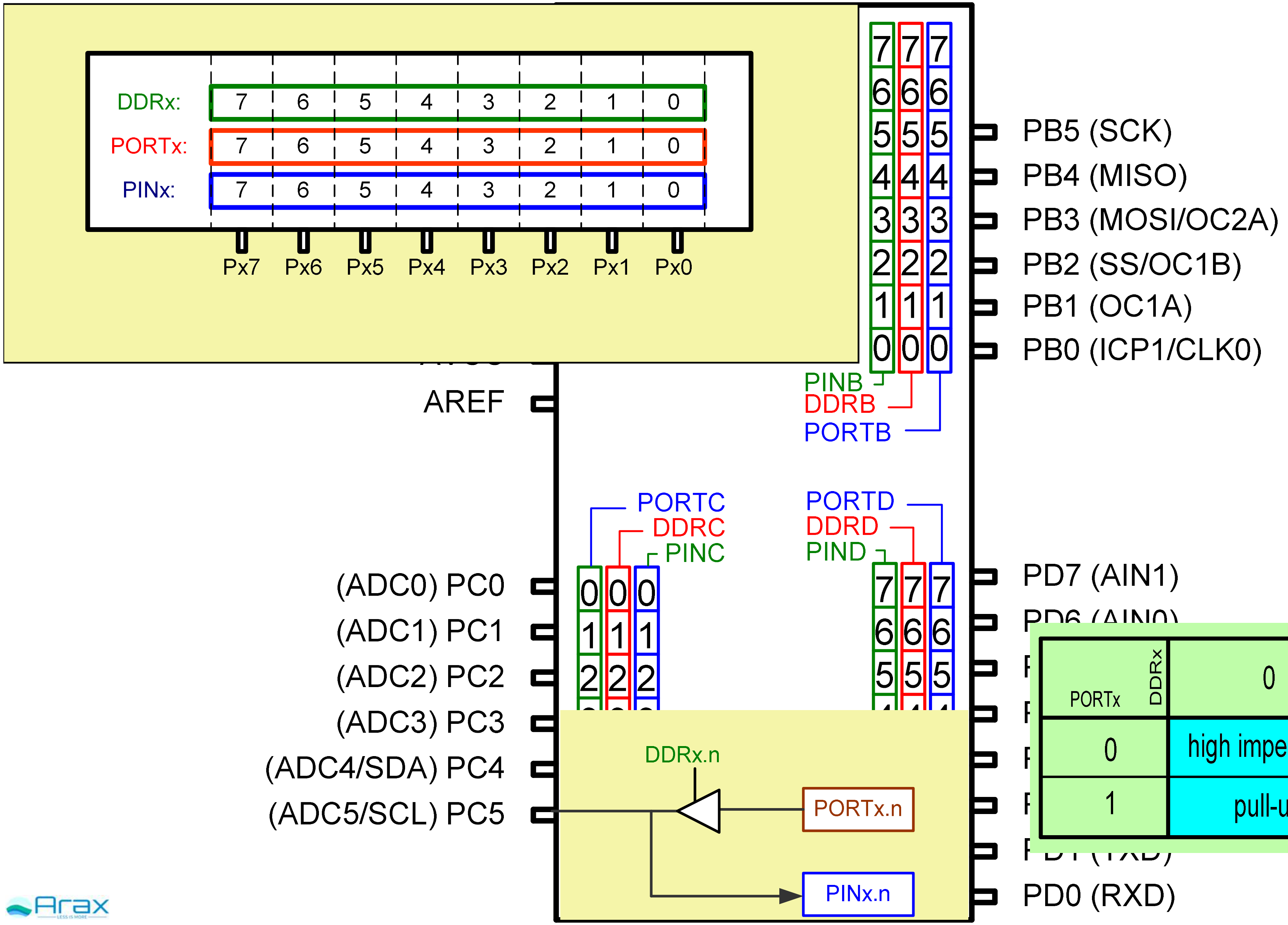
Arax
www.araxai.ir

ATmega328 pinout

1. Vital Pins:
   1. Power
      - VCC
      - Ground
   2. Crystal
      - XTAL1
      - XTAL2
   3. Reset
2. I/O pins
   - PORTB, PORTC, and PORTD
3. Internal ADC pins
   - AREF, AVCC, ADCn



The structure of I/O pins



DDRx PORTx	0	1
0	high impedance	Out 0
1	pull-up	Out 1

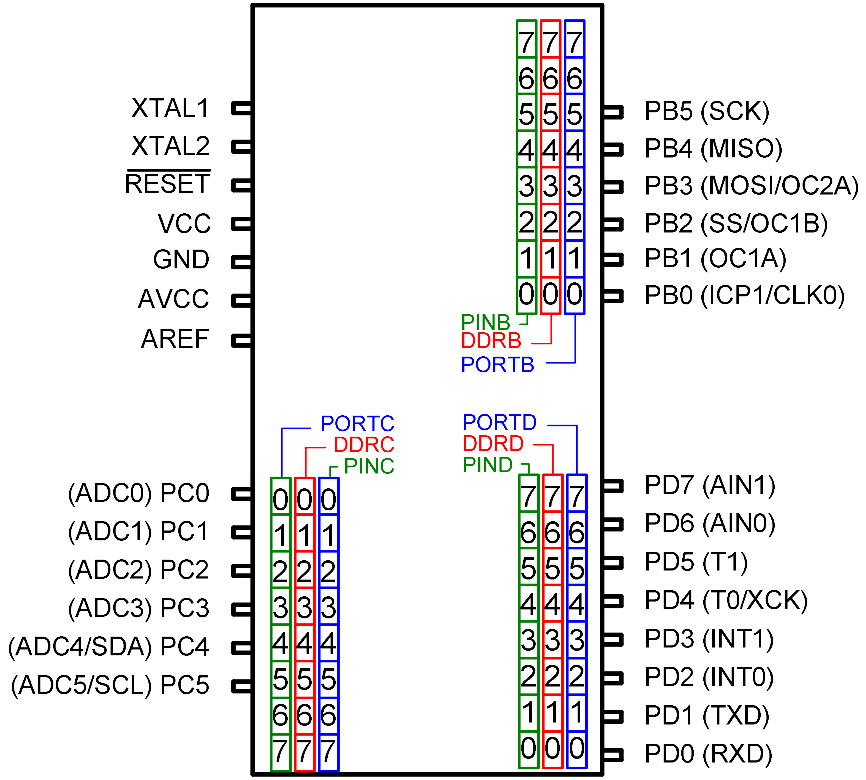
Example 1

- Write a program that makes all the pins of PORTB one.

DDRB: 1 1 1 1 1 1 1 1

PORTB: 1 1 1 1 1 1 1 1

```
LDI   R20,0xFF   ;R20 = 11111111 (binary)
OUT   PORTB,R20  ;PORTB = R20
OUT   DDRB,R20   ;DDRB = R20
```



PORTx \ DDRx	0	1
0	high impedance	Out 0
1	pull-up	Out 1

Example 2

- The following code will toggle all 8 bits of Port B forever with some time delay between “on” and “off” states:

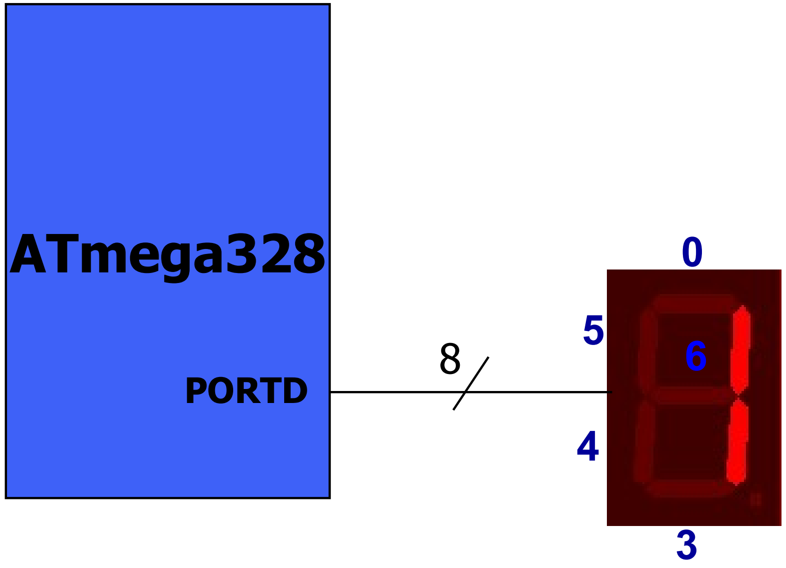
```
    LDI   R16,0xFF    ;R16 = 0xFF = 0b11111111
    OUT   DDRB,R16    ;make Port B an output port (1111 1111)
L1: LDI   R16,0x55    ;R16 = 0x55 = 0b01010101
    OUT   PORTB,R16   ;put 0x55 on port B pins
    CALL   DELAY
    LDI   R16,0xAA    ;R16 = 0xAA = 0b10101010
    OUT   PORTB,R16   ;put 0xAA on port B pins
    CALL DELAY
    RJMP L1
```

Example 3

- A 7-segment is connected to PORTD. Display 1 on the the 7-segment.

DDRD:	1	1	1	1	1	1	1	1
PORTD:	0	0	0	0	0	1	1	0

```
    LDI   R20,0x06   ;R20 = 00000110 (binary)
    OUT   PORTD,R20  ;PORTD = R20
    LDI   R20,0xFF   ;R20 = 11111111 (binary)
    OUT   DDRD,R20   ;DDRD = R20
L1: RJMP L1
```



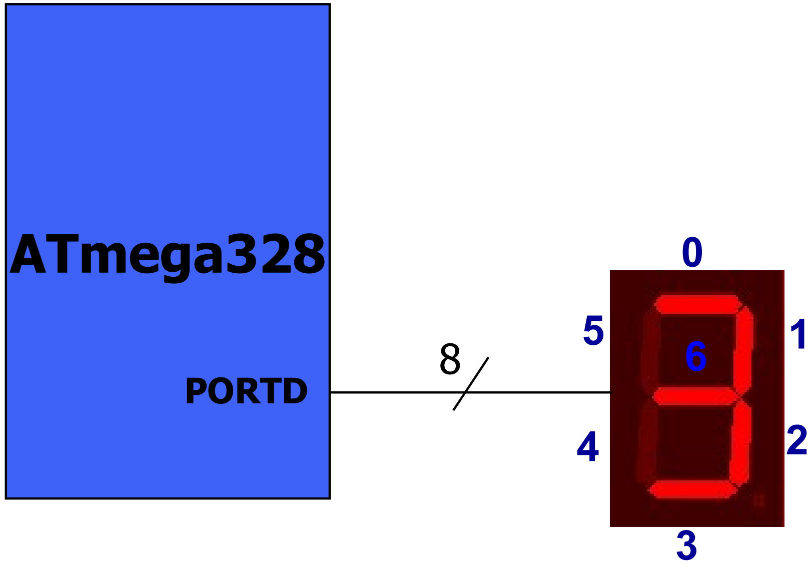
PORTx \ DDRx	0	1
0	high impedance	Out 0
1	pull-up	Out 1

Example 4

- A 7-segment is connected to PORTD. Display 3 on the 7-segment.

DDRD:	1	1	1	1	1	1	1	1
PORTD:	0	1	0	0	1	1	1	1

```
    LDI  R20,0x4F  ;R20 = 01001111 (binary)
    OUT  PORTD,R20 ;PORTD = R20
    LDI  R20,0xFF  ;R20 = 11111111 (binary)
    OUT  DDRD,R20  ;DDRD = R20
L1: RJMP L1
```

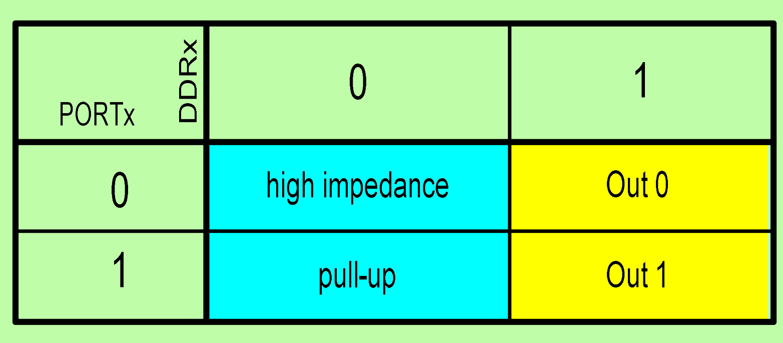


PORTx \ DDRx	0	1
0	high impedance	Out 0
1	pull-up	Out 1

Example 5: Input

- The following code gets the data present at the pins of port C and sends it to port B indefinitely, after adding the value 5 to it:

```
      LDI   R16,0x00    ;R16 = 00000000 (binary)
      OUT   DDRC,R16    ;make Port C an input port
      LDI   R16,0xFF    ;R16 = 11111111 (binary)
      OUT   DDRB,R16    ;make Port B an output port(1 for Out)
L2:   IN    R16,PINC    ;read data from Port C and put in R16
      LDI   R17,5
      ADD   R16,R17     ;add 5 to it
      OUT   PORTB,R16   ;send it to Port B
      RJMP  L2          ;jump L2
```



PORTx \ DDRx	0	1
0	high impedance	Out 0
1	pull-up	Out 1

Example 5: Input..

- The following code gets the data present at the pins of port C and sends it to port B indefinitely, after adding the value 5 to it:

What is wrong?

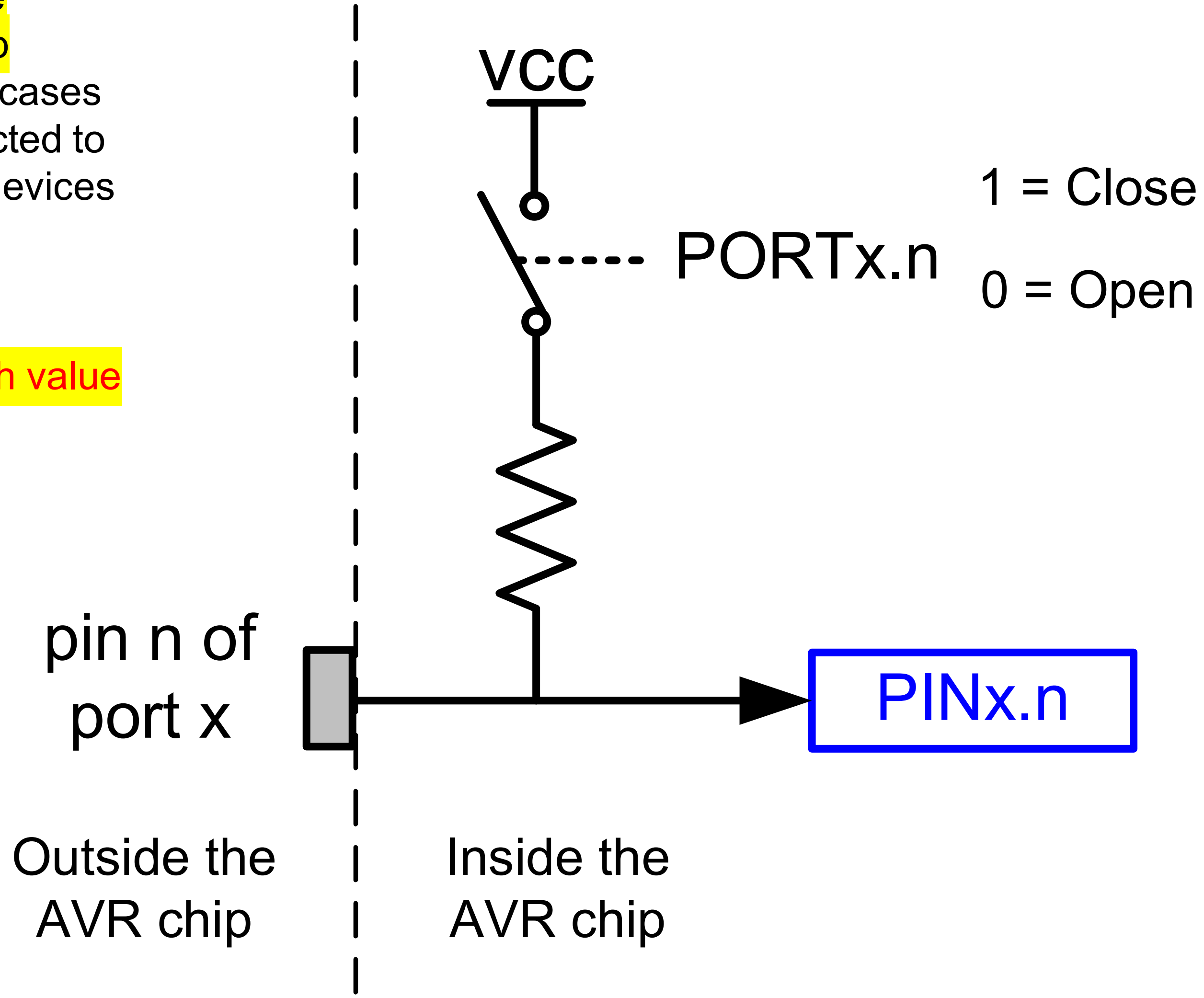
```
      LDI    R16,0xFF      ;R16 = 11111111 (binary)
      OUT    DDRB,R16      ;make Port B an output port(1 for Out)
      LDI    R16,0x00      ;R16 = 00000000 (binary)
      OUT    DDRC,R16      ;make Port C an input port
L2:   IN     R16,PINC      ;read data from Port C and put in R16
      LDI    R17,5
      ADD    R16,R17       ;add 5 to it
      OUT    PORTB,R16     ;send it to Port B
      RJMP   L2            ;jump L2
```

Bad Program-One cycle delay need NOP

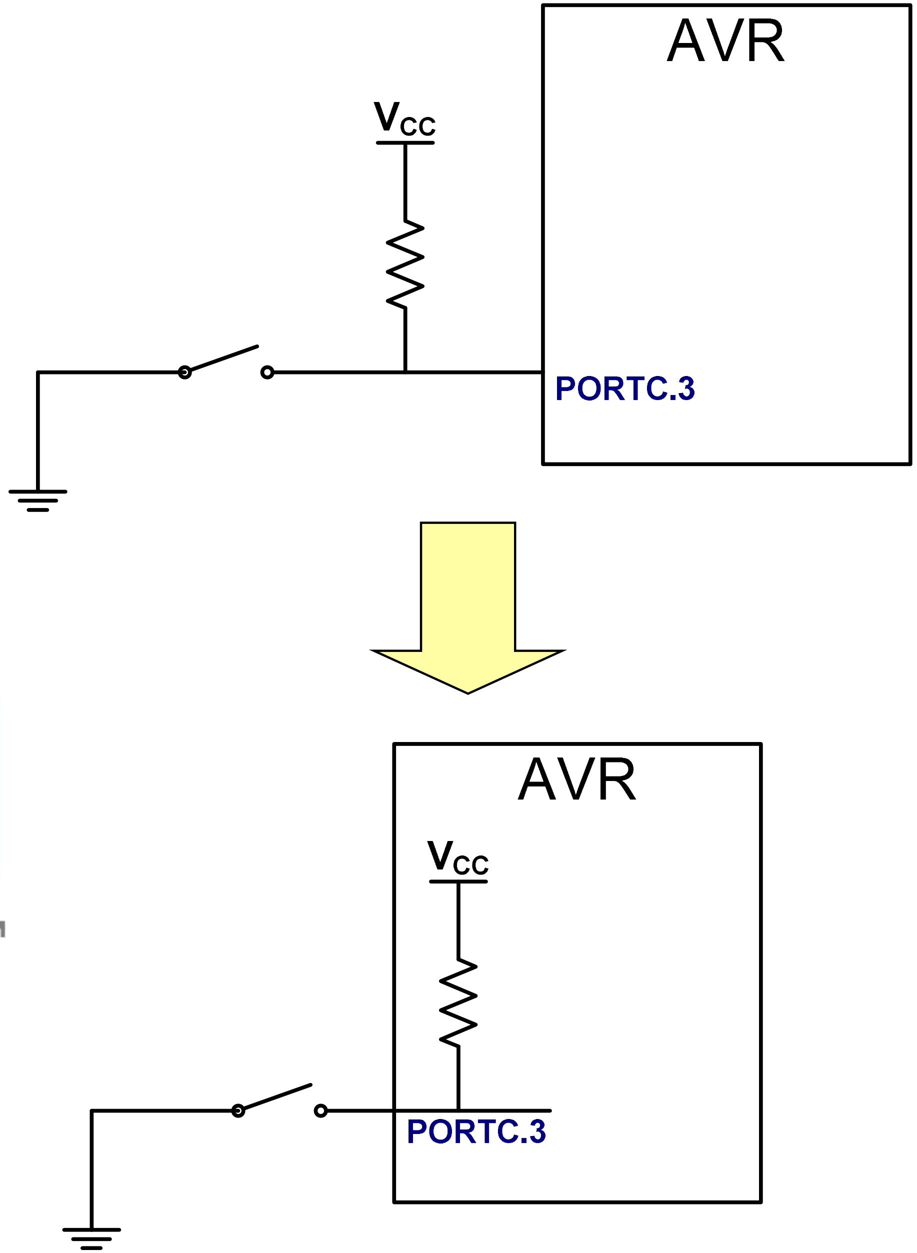
Pull-up resistor

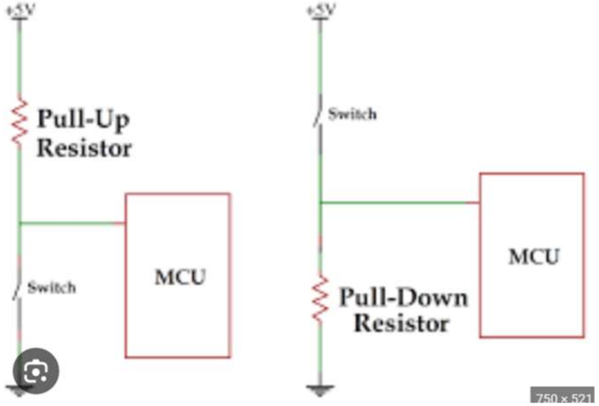
If we put 1s into bits of the PORTx register, the pullup resistors are activated. In cases in which nothing is connected to the pin or the connected devices have high impedance, the resistor pulls up the pin

Set value to predefine high value
Prevent Noise



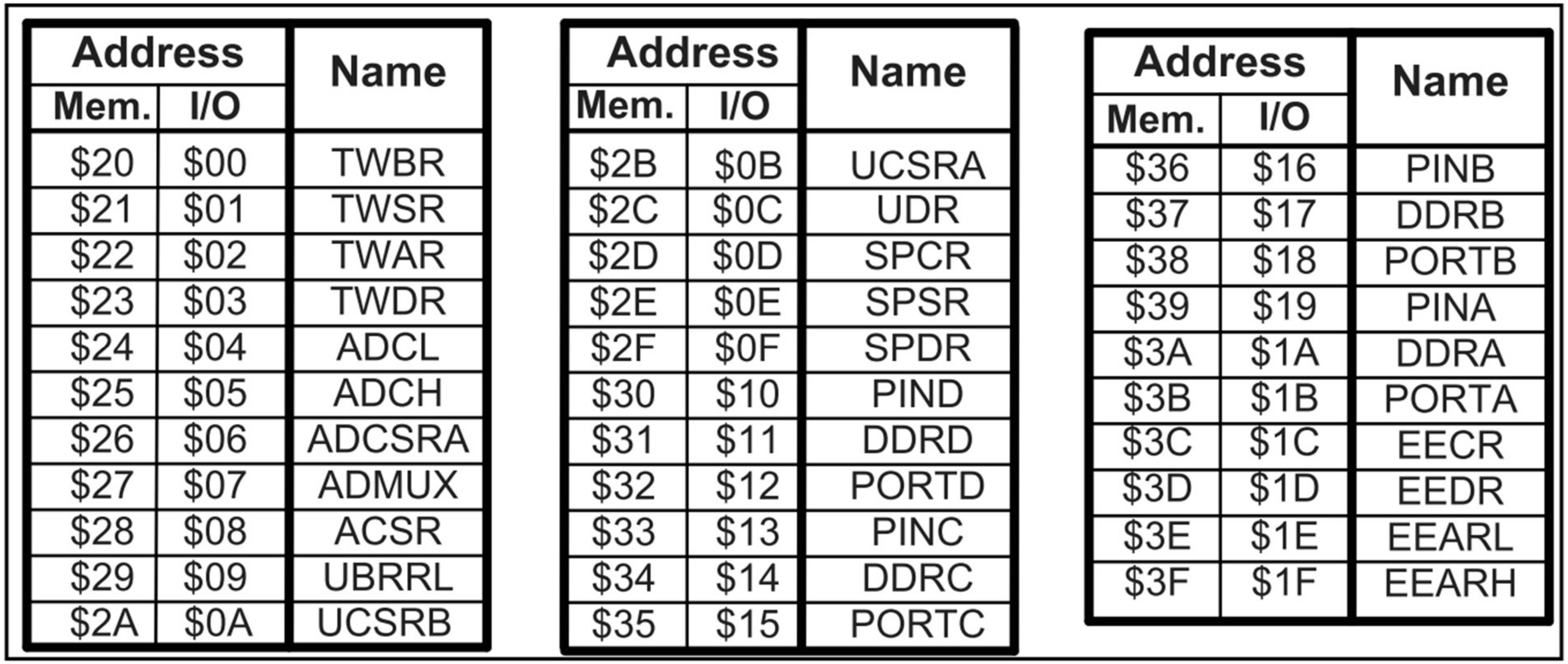
Example


AVR
V_{CC}
PORTC.3
AVR
V_{CC}
PORTC.3


+5V
Pull-Up Resistor
Switch
MCU
+5V
Switch
MCU
Pull-Down Resistor
750 x 521

I/O bit manipulation programming

- Manipulate just one bit



Address		Name
Mem.	I/O	
$20	$00	TWBR
$21	$01	TWSR
$22	$02	TWAR
$23	$03	TWDR
$24	$04	ADCL
$25	$05	ADCH
$26	$06	ADCSRA
$27	$07	ADMUX
$28	$08	ACSR
$29	$09	UBRRL
$2A	$0A	UCSRB
$2B	$0B	UCSRA
$2C	$0C	UDR
$2D	$0D	SPCR
$2E	$0E	SPSR
$2F	$0F	SPDR
$30	$10	PIND
$31	$11	DDRD
$32	$12	PORTD
$33	$13	PINC
$34	$14	DDRC
$35	$15	PORTC
$36	$16	PINB
$37	$17	DDRB
$38	$18	PORTB
$39	$19	PINA
$3A	$1A	DDRA
$3B	$1B	PORTA
$3C	$1C	EECR
$3D	$1D	EEDR
$3E	$1E	EEARL
$3F	$1F	EEARH

Table 8: The Lower 32 I/O Registers

SBI and CBI instructions

- SBI (Set Bit in IO register)
  - SBI ioReg, bit ;ioReg.bit = 1
  - Examples:
    - SBI PORTD,0 ;PORTD.0 = 1
    - SBI DDRC,5 ;DDRC.5 = 1

- CBI (Clear Bit in IO register)
  - CBI ioReg, bit ;ioReg.bit = 0
  - Examples:
    - CBI PORTD,0 ;PORTD.0 = 0
    - CBI DDRC,5 ;DDRC.5 = 0

Example

- Write a program that toggles PORTB.4 continuously.

```
    SBI   DDRB,4
L1: SBI   PORTB,4
    CBI   PORTB,4
    RJMP  L1
```

Example

- An LED is connected to each pin of Port D. Write a program to turn on each LED from pin D0 to pin D7. Call a delay module before turning on the next LED.

```
         LDI      R20, 0xFF
         OUT      DDRD, R20                ;make PORTD an output port
         SBI      PORTD,0                  ;set bit PD0
         CALL     DELAY                    ;delay before next one
         SBI      PORTD,1                  ;turn on PD1
         CALL     DELAY                    ;delay before next one
         SBI      PORTD,2                  ;turn on PD2
         CALL     DELAY
         SBI      PORTD,3
         CALL     DELAY
         SBI      PORTD,4
         CALL     DELAY
         SBI      PORTD,5
         CALL     DELAY
         SBI      PORTD,6
         CALL     DELAY
         SBI      PORTD,7
         CALL     DELAY
```

SBIC and SBIS

- SBIC (Skip if Bit in IO register Cleared)
  - SBIC ioReg, bit ; if (ioReg.bit = 0) skip next instruction
  - Example:

```
SBIC   PORTD,0   ;skip next instruction if PORTD.0=0
INC    R20
LDI    R19,0x23
```

- SBIS (Skip if Bit in IO register Set)
  - SBIS ioReg, bit ; if (ioReg.bit = 1) skip next instruction
  - Example:

```
SBIS   PORTD,0   ;skip next instruction if PORTD.0=1
INC    R20
LDI    R19,0x23
```

Example

- Write a program to perform the following:
- (a) Keep monitoring the PB2 bit until it becomes HIGH;
- (b) When PB2 becomes HIGH, write value $45 to Port C, and also send a HIGH-to-LOW pulse to PD3.

```
       CBI  DDRB, 2        ;make PB2 an input
       SBI  PORTB,2
       LDI  R16, 0xFF
       OUT  DDRC, R16      ;make Port C an output port
       SBI  DDRD, 3        ;make PD3 an output
AGAIN: SBIS PINB, 2        ;Skip if Bit PB2 is HIGH
       RJMP AGAIN          ;keep checking if LOW
       LDI  R16, 0x45
       OUT  PORTC, R16     ;write 0x45 to port C
       SBI  PORTD, 3       ;set bit PD3 (H-to-L)
       CBI  PORTD, 3       ;clear bit PD3
HERE:  RJMP HERE
```

Example

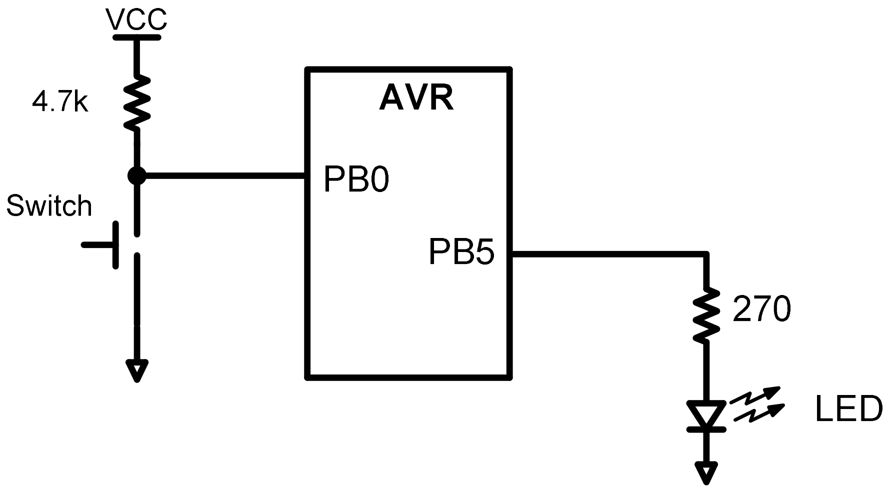
Write a program to perform the following:

- (a) Keep monitoring the PB2 bit until it becomes LOW;
- (b) When PB2 becomes LOW, write value $45 to Port C, and also send a HIGH-to-LOW pulse to PD3.

```
       CBI  DDRB, 2        ;make PB2 an input
       SBI  PORTB,2
       LDI  R16, 0xFF
       OUT  DDRC, R16      ;make Port C an output port
       SBI  DDRD, 3        ;make PD3 an output
AGAIN: SBIC PINB, 2        ;Skip if Bit PB2 is LOW
       RJMP AGAIN          ;keep checking if High
       LDI  R16, 0x45
       OUT  PORTC, R16     ;write 0x45 to port C
       SBI  PORTD, 3       ;set bit PD3 (H-to-L)
       CBI  PORTD, 3       ;clear bit PD3
HERE:  RJMP HERE
```

Example

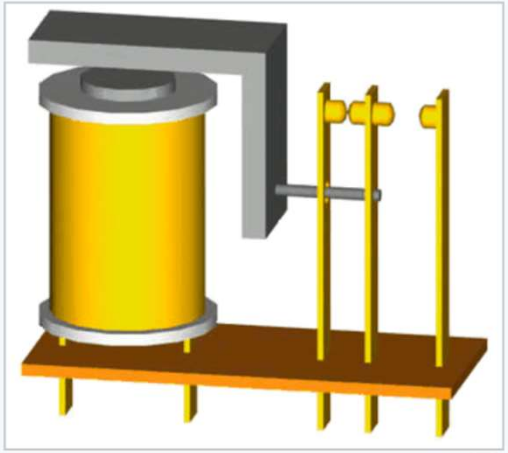
- A switch is connected to pin PB0 and an LED to pin PB5. Write a program to get the status of SW and send it to the LED.



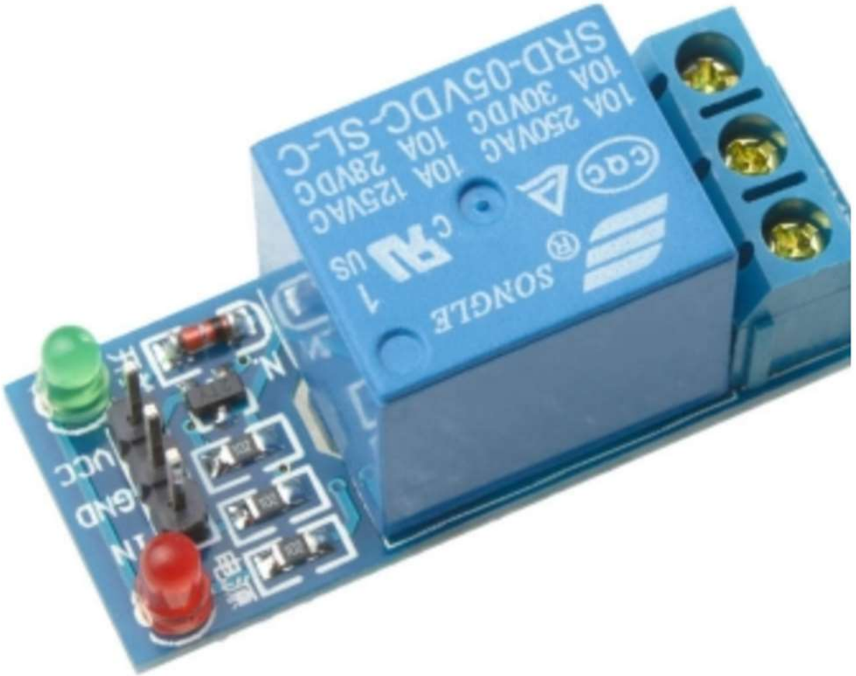
```
        CBI  DDRB,0         ;make PB0 an input
        SBI  DDRB,5         ;make PB5 an output
AGAIN:  SBIC PINB,0         ;skip next if PB0 is clear
        RJMP OVER           ;(JMP is OK too)
        CBI  PORTB,5
        RJMP AGAIN          ;we can use JMP too
OVER:   SBI  PORTB,5
        RJMP AGAIN          ;we can use JMP too
```

relay

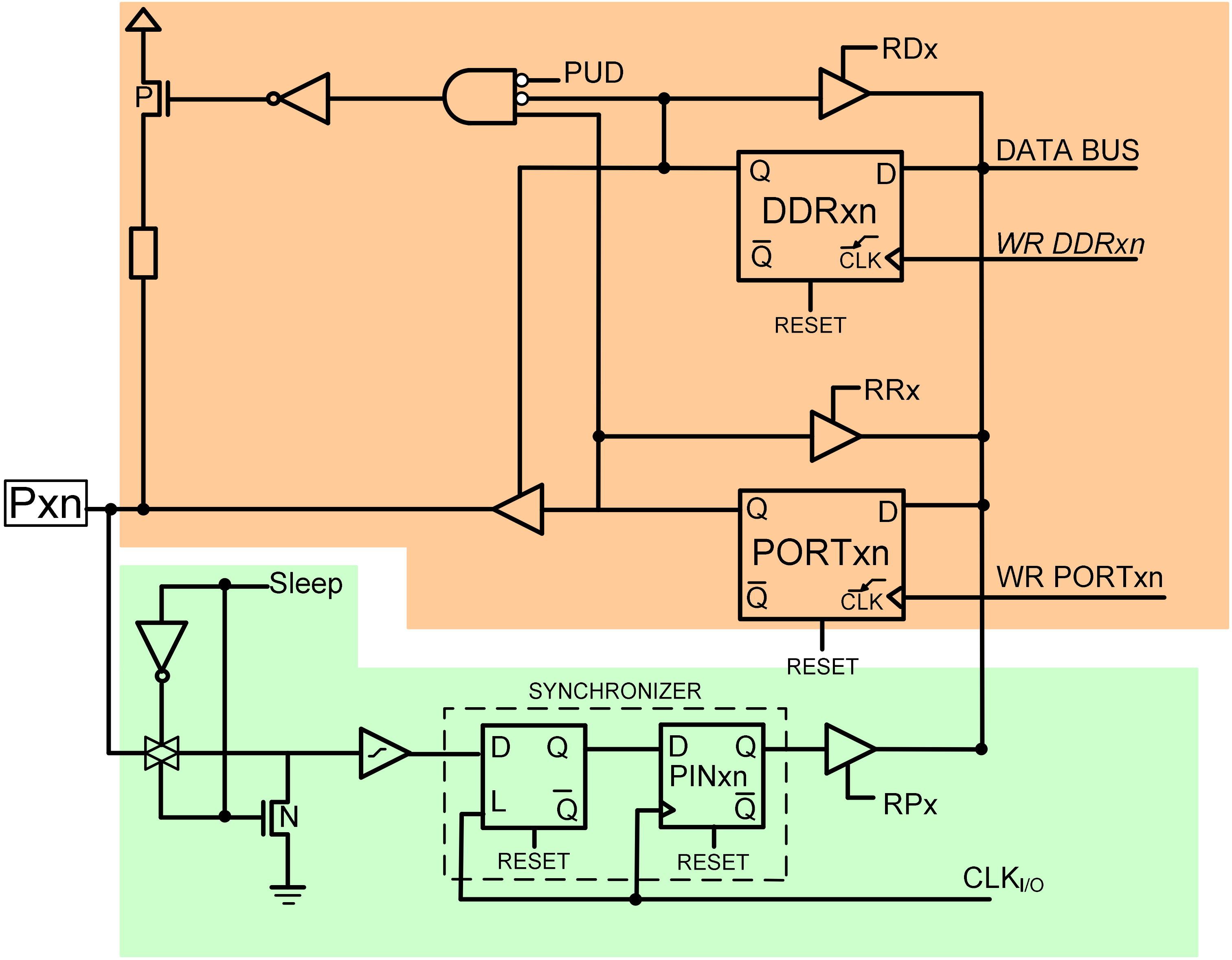
- A **relay** is an electrically operated switch.
- a high power or high voltage circuit with a low power circuit



Electromechanical relay principle

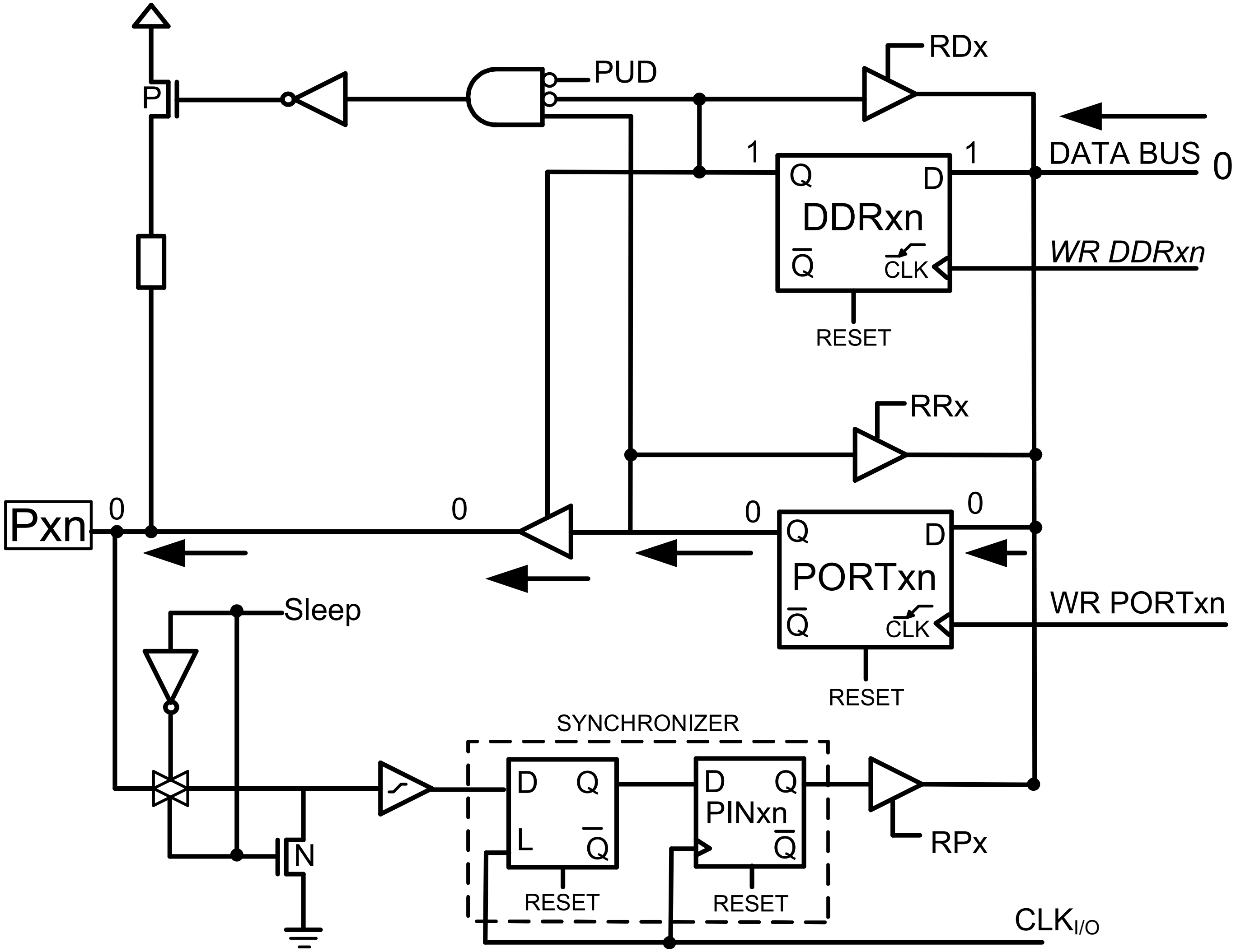


The structure of I/O pins



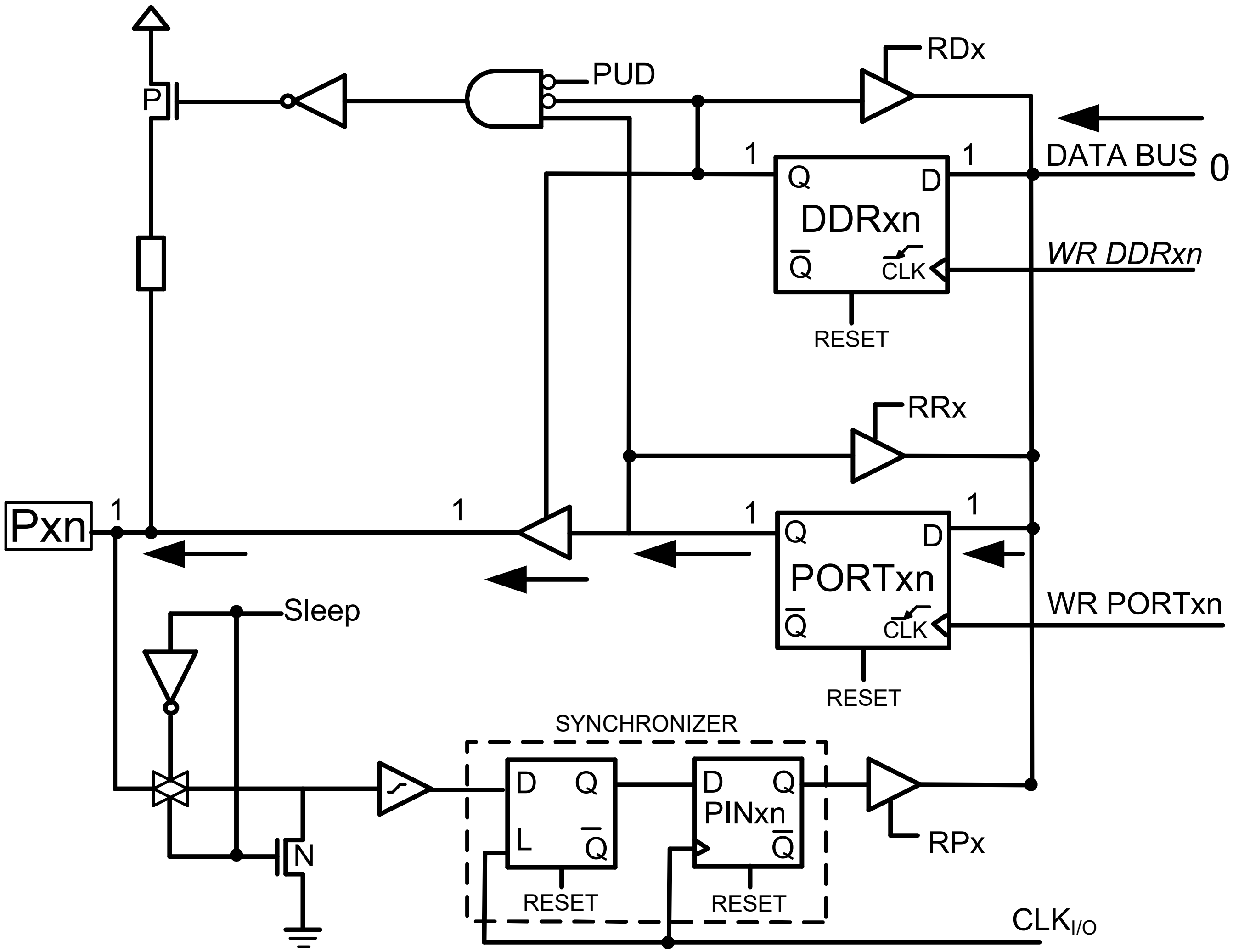
www.araxai.ir

Out 0



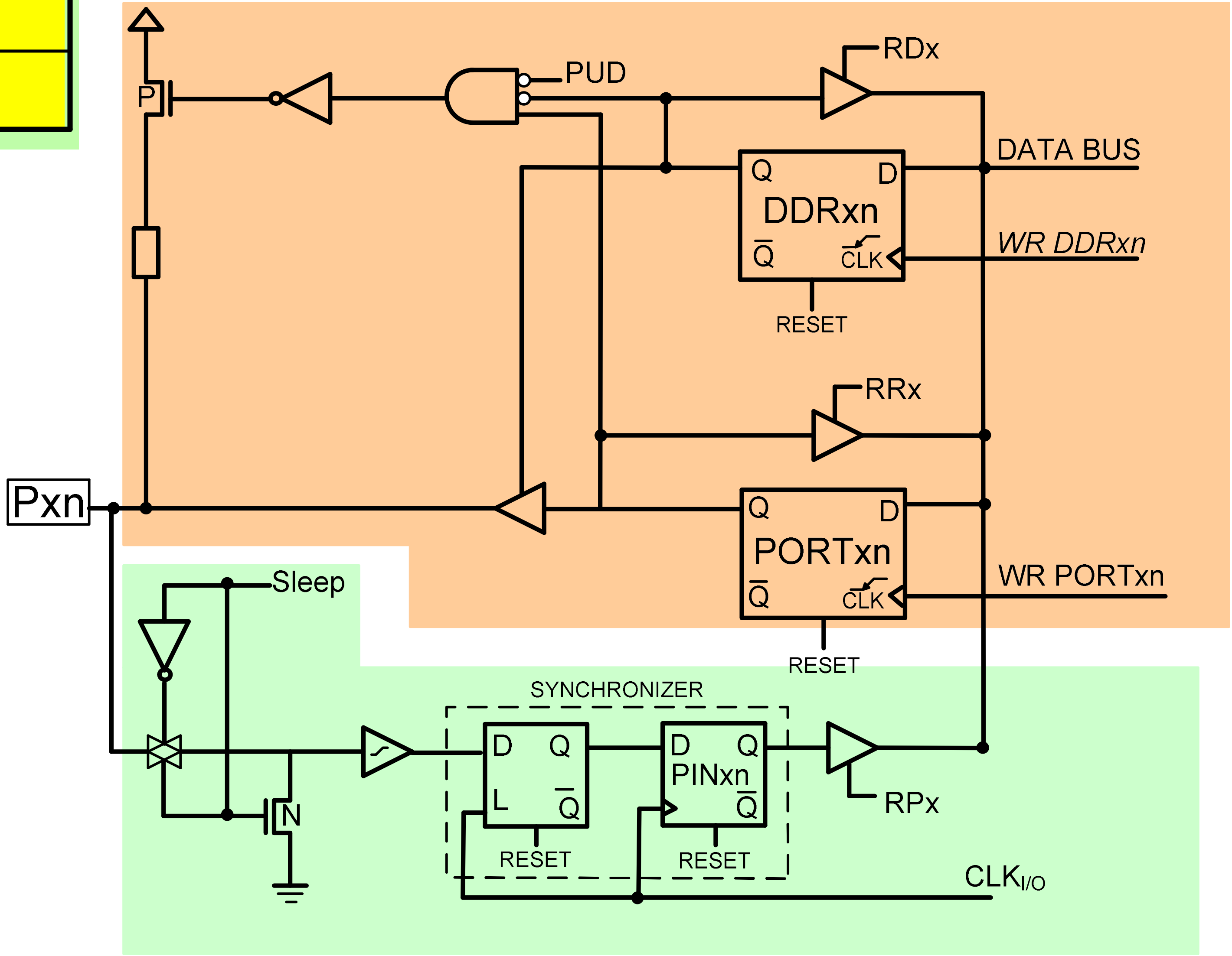
www.araxai.ir

Out 1


PUD
RDx
DATA BUS
0
1
1
Q
D
DDRxn
Q
CLK
WR DDRxn
RESET
RRx
P
Pxn
1
1
1
1
Q
D
PORTxn
Q
CLK
WR PORTxn
RESET
Sleep
N
SYNCHRONIZER
D
Q
L
Q
RESET
D
Q
PINxn
Q
RESET
RPx
CLK$_{I/O}$

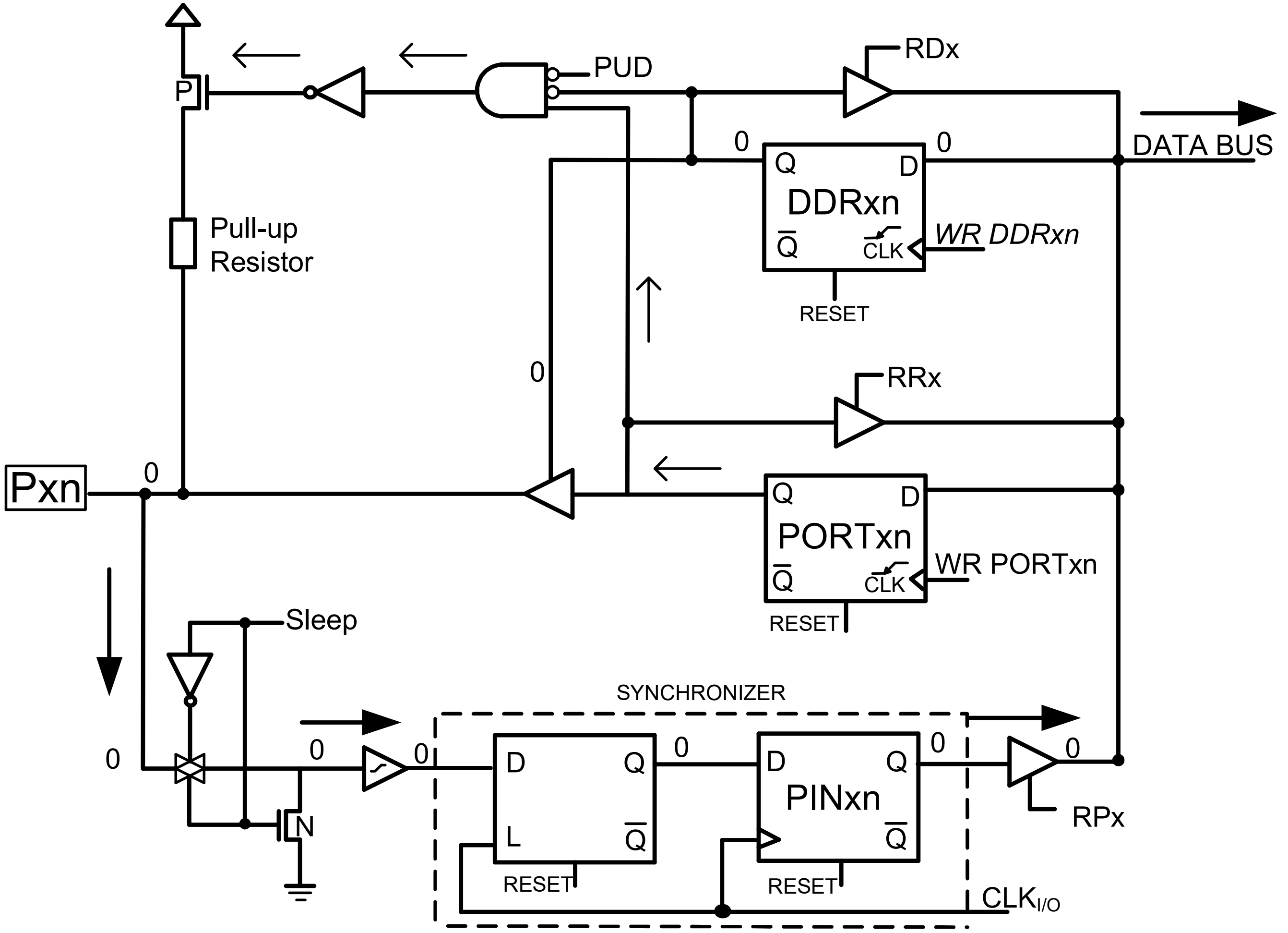
The structure of I/O pins

PORTx \ DDRx	0	1
0	high impedance	Out 0
1	pull-up	Out 1



www.araxai.ir

Input (Tri-state vs. pull up)



The ⟵ represents how the content of PORTx register affects the pull-up resistor;
while the ⟶ shows how a data can be read from a pin