AVR I/O PORT PROGRAMMING

Hoda Roodaki

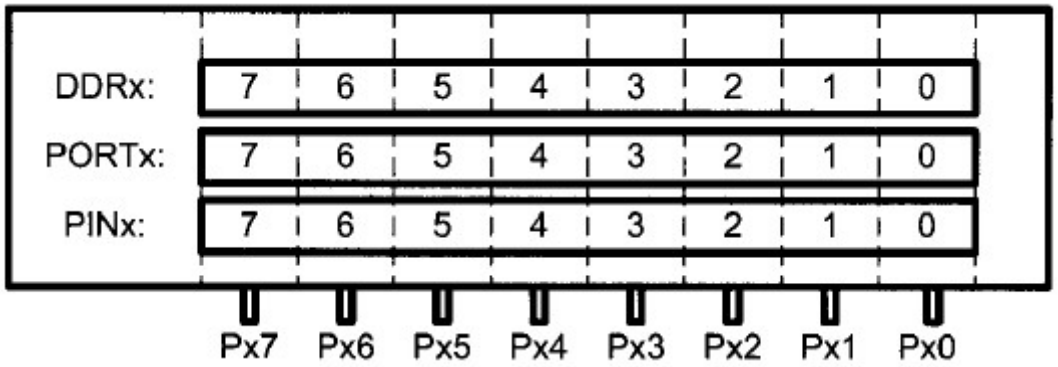
hroodaki@kntu.ac.ir

I/O port pins and their functions

- The 40-pin AVR has four ports. They are PORTA, PORTB, PORTC, and PORTD.
  - To use any of these ports as an input or output port, it must be programmed.
- Each port has three I/O registers associated with it.

I/O port pins and their functions

Port	Address	Usage
PORTA	$3B	output
DDRA	$3A	direction
PINA	$39	input
PORTB	$38	output
DDRB	$37	direction
PINB	$36	input
PORTC	$35	output
DDRC	$34	direction
PINC	$33	input
PORTD	$32	output
DDRD	$31	direction
PIND	$30	input



DDRx register role in outputting data

- Each of the ports A-D can be used for input or output.
- The DDRx I/O register is used solely for the purpose of making a given port an input or output port.
  - To make a port an output, we write 1 s to the DDRx register.
    - To output data to all of the pins of the PortB, we must first put 0b11111111 into the DDRB register to make all of the pins output.

DDRx register role in outputting data

- It must be noted that unless we set the DDRx bits to one, the data will not go from the port register to the pins of the AVR.
- This means that if we don't use

  LDI R16, 0xFF

  Out DDRB, R16

  - The 0x55 and 0xAA values will not get to the pins. They will be sitting in the I/O register of PortB inside the CPU.

Example

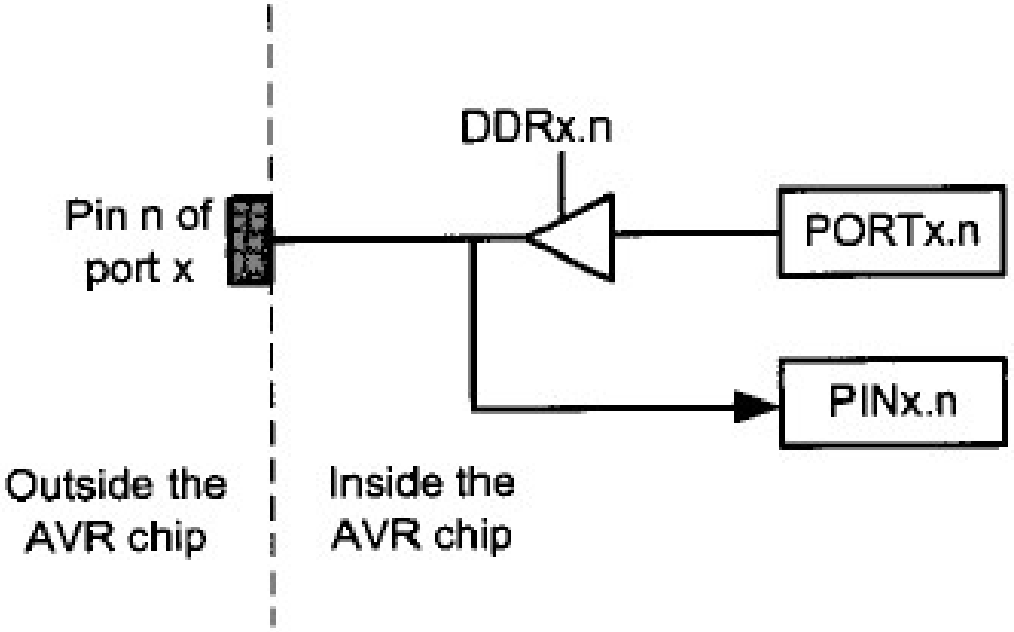
The following code will toggle all 8 bits of Port B forever with some time delay between "on" and "off" states:

```
        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
```

DDRx register role in inputting data

- To make a port an input port, we must first put 0s into the DDRx register for that port, and then bring in (read) the data present at the pins.

The I/O Port in AVR



Example

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:

```
.INCLUDE "M32DEF.INC"
      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           ;continue forever
```

If we want to make the pull-up resistors of port C active, we must put 1s into the PORTC register. The program becomes as follows:

```
.INCLUDE    "M32DEF.INC"
      LDI   R16,0xFF     ;R16 = 11111111 (binary)
      OUT   DDRB,R16     ;make Port B an output port
      OUT   PORTC,R16    ;make the pull-up resistors of C active
      LDI   R16,0x00     ;R16 = 00000000 (binary)
      OUT   DDRC,R16     ;Port C an input port (0 for I)
L2:   IN    R16,PINC     ;move data from Port C to R16
      LDI   R17,5
      ADD   R16,R17      ;add some value to it
      OUT   PORTB,R16    ;send it to Port B
      RJMP  L2           ;continue forever
```

Synchronizer delay

- The input circuit of the AVR has a delay of 1 clock cycle. In other words, the PIN register represents the data that was present at the pins one clock ago.

Example

```
.INCLUDE    "M32DEF.INC"
      .EQU  MYTEMP 0x100       ;save it here
      LDI   R16,0x00     ;R16 = 00000000 (binary)
      OUT   DDRA,R16     ;make Port A an input port (0 for In)
      NOP                ;synchronizer delay
      IN    R16,PINA     ;move from pins of Port A to R16
      STS   MYTEMP,R16   ;save it in MYTEMP
```

Example 4-1

Write a test program for the AVR chip to toggle all the bits of PORTB, PORTC, and PORTD every 1/4 of a second. Assume a crystal frequency of 1 MHz.

Solution:

```
;tested with AVR Studio for the ATmega32 and XTAL = 1 MHz
;to select the XTAL frequency in AVR Studio, press ALT+O
.INCLUDE "M32DEF.INC"
      LDI   R16, HIGH(RAMEND)
      OUT   SPH, R16
      LDI   R16, LOW(RAMEND)
      OUT   SPL, R16    ;initialize stack pointer

      LDI   R16, 0xFF
      OUT   DDRB, R16   ;make Port B an output port
      OUT   DDRC, R16   ;make Port C an output port
      OUT   DDRD, R16   ;make Port D an output port

      LDI   R16, 0x55   ;R16 = 0x55
L3:   OUT   PORTB, R16  ;put 0x55 on Port B pins
      OUT   PORTC, R16  ;put 0x55 on Port C pins
      OUT   PORTD, R16  ;put 0x55 on Port D pins
      CALL  QDELAY      ;quarter of a second delay
      COM   R16         ;complement R16
      RJMP  L3
```

Example 4-1

```
;----------------1/4 SECOND DELAY
QDELAY:
      LDI   R21, 200
D1:   LDI   R22, 250
D2:   NOP
      NOP
      DEC   R22
      BRNE  D2
      DEC   R21
      BRNE  D1
      RET
```

Calculations:

1 / 1 MHz = 1 μs
Delay = 200 × 250 × 5 MC × 1 μs = 250,000 μs (If we include the overhead, we will have 250,608 μs. See Example 3-18 in the previous chapter.)

I/O BIT MANIPULATION PROGRAMMING

- Sometimes we need to access only 1 or 2 bits of the port instead of the entire 8 bits.
- A powerful feature of AVR I/O ports is their capability to access individual bits of the port without altering the rest of the bits in that port.
- For all AVR ports, we can access either all 8 bits or any single bit without altering the rest.

I/O BIT MANIPULATION PROGRAMMING

Instruction		Function
SBI	ioReg,bit	Set Bit in I/O register (set the bit: bit = 1)
CBI	ioReg,bit	Clear Bit in I/O register (clear the bit: bit = 0)
SBIC	ioReg,bit	Skip if Bit in I/O register Cleared (skip next instruction if bit = 0)
SBIS	ioReg,bit	Skip if Bit in I/O register Set (skip next instruction if bit = 1)

SBI (set bit in I/O register)

- To set HIGH a single bit of a given I/O register, we use the following syntax:

  SBI ioReg, bit_num

  – where ioReg can be the lower 32 I/O registers (addresses 0 to 31) and bit_num is the desired bit number from 0 to 7.

    - For example the following instruction sets HIGH bit 5 of Port B:

      SBI PORTB,5

SBI (set bit in I/O register)

SBI a, b

$0 \leq a \leq 31$

$0 \leq b \leq 7$

CBI (Clear Bit in I/O register)

- To clear a single bit of a given I/O register, we use the following syntax:

  CBI ioReg, bit_num

  – For example, the following code toggles pin PB2 continuously:

```
      SBI   DDRB, 2          ;bit = 1, make PB2 an output pin
AGAIN:SBI   PORTB, 2         ;bit set (PB2 = high)
      CALL  DELAY
      CBI   PORTB, 2         ;bit clear (PB2 = low)
      CALL  DELAY
      RJMP  AGAIN
```

  – Remember that for I/O ports, we must set the appropriate bit in the DDRx register if we want the pin to be output.

  – The unused portions of PortC are undisturbed.

CBI (Clear Bit in I/O register)

CBI a, b

$0 \leq a \leq 31$

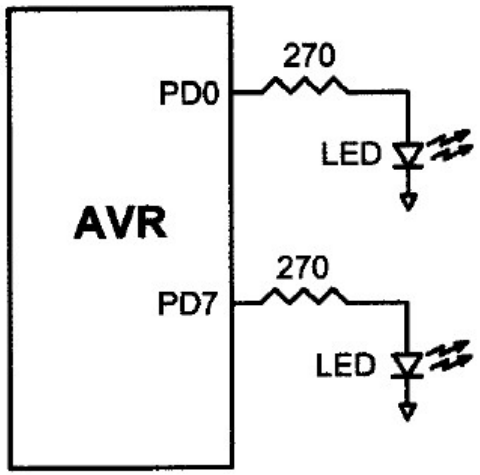
$0 \leq b \leq 7$

Example 4-2

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 subroutine before turning on the next LED.

Solution:

```
.INCLUDE "M32DEF.INC"
LDI   R20, HIGH(RAMEND)
OUT   SPH, R20
LDI   R20, LOW(RAMEND)
OUT   SPL, R20     ;initialize stack pointer
LDI   R20, 0xFF
OUT   PORTD, 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
```



Example 4-3

Write the following programs:
(a) Create a square wave of 50% duty cycle on bit 0 of Port C.
(b) Create a square wave of 66% duty cycle on bit 3 of Port C.

Solution:

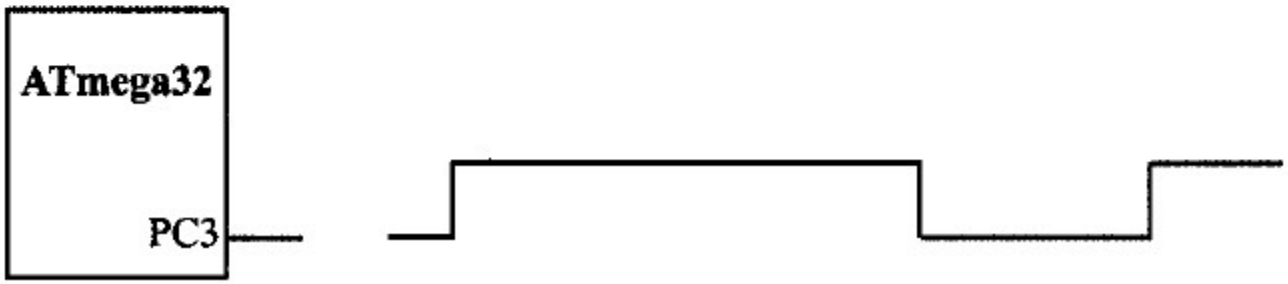
(a) The 50% duty cycle means that the "on" and "off" states (or the high and low portions of the pulse) have the same length. Therefore, we toggle PC0 with a time delay between each state.

```
.INCLUDE "M32DEF.INC"

       LDI   R20, HIGH(RAMEND)
       OUT   SPH, R20
       LDI   R20, LOW(RAMEND)
       OUT   SPL, R20     ;initialize stack pointer

       SBI   DDRC, 0      ;set bit 0 of DDRC (PC0 = out)
HERE:  SBI   PORTC, 0     ;set to HIGH PC0 (PC0 = 1)
       CALL  DELAY        ;call the delay subroutine
       CBI   PORTC, 0     ;PC0 = 0
       CALL  DELAY
       RJMP  HERE         ;keep doing it
```

Example 4-3

(b) A 66% duty cycle means that the "on" state is twice the "off" state.

```
        ....
        SBI    DDRC, 3      ;set bit 3 of DDRC (PC3 = out)
HERE:   SBI    PORTC, 3     ;set to HIGH PC3 (PC3 = 1)
        CALL   DELAY        ;call the delay subroutine
        CALL   DELAY        ;call the delay subroutine
        CBI    PORTC, 3     ;PC3 = 0
        CALL   DELAY
        RJMP   HERE         ;keep doing it
```



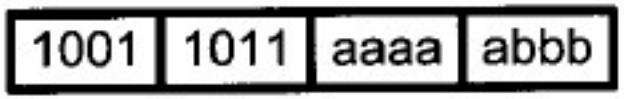
Checking an input pin

- To make decisions based on the status of a given bit in the file register:
  - SBIC (Skip if Bit in I/O register Cleared)
  - SBIS (Skip if Bit in I/O register Set)
- They allow you to monitor a single pin and make a decision depending on whether it is 0 or 1.
- Again it must be noted that the SBIC and SBIS instructions can be used for any bits of the lower 32 I/O registers, including the I/O ports A, B, C, D.

SBIS (Skip if Bit in I/O register Set)

- To monitor the status of a single bit for HIGH, we use the SBIS instruction.
- This instruction tests the bit and skips the next instruction if it is HIGH.

SBIS a,b



$0 \leq a \leq 31$

$0 \leq b \leq 7$

Example 4-4

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

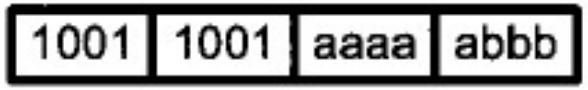
Solution:

```
.INCLUDE "M32DEF.INC"

             CBI   DDRB, 2      ;make PB2 an input
             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
```

SBIC (Skip if Bit in I/O register Cleared)

- To monitor the status of a single bit for LOW, we use the SBIC instruction.
- This instruction tests the bit and skips the instruction right below it if the bit is LOW.

SBIC a,b



$0 \leq a \leq 31$

$0 \leq b \leq 7$

Example 4-5

Assume that bit PB3 is an input and represents the condition of a door alarm. If it goes LOW, it means that the door is open. Monitor the bit continuously. Whenever it goes LOW, send a HIGH-to-LOW pulse to port PC5 to turn on a buzzer.

Solution:

```
.INCLUDE "M32DEF.INC"

      CBI   DDRB, 3     ;make PB3 an input
      SBI   DDRC, 5     ;make PC5 an output
HERE: SBIC  PINB, 3     ;keep monitoring PB3 for HIGH
      RJMP  HERE        ;stay in the loop
      SBI   PORTC,5     ;make PC5 HIGH
      CBI   PORTC,5     ;make PC5 LOW for H-to-L
      RJMP  HERE
```

Example 4-6

A switch is connected to pin PB2. Write a program to check the status of SW and perform the following:
(a) If SW = 0, send the letter 'N' to PORTD.
(b) If SW = 1, send the letter 'Y' to PORTD.

Solution:

```
        .INCLUDE "M32DEF.INC"

        CBI    DDRB, 2      ;make PB2 an input
        LDI    R16, 0xFF
        OUT    DDRD, R16    ;make PORTD an output port

AGAIN:SBIS   PINB, 2       ;skip next if PB bit is HIGH
        RJMP   OVER         ;SW is LOW
        LDI    R16, 'Y'     ;R16 = 'Y' (ASCII letter Y)
        OUT    PORTD, R16   ;PORTD = 'Y'
        RJMP   AGAIN
OVER:   LDI    R16, 'N'     ;R16 = 'N' (ASCII letter Y)
        OUT    PORTD, R16   ;PORTD = 'N'
        RJMP   AGAIN
```

Example 4-8

A switch is connected to pin PB0 and an LED to pin PB7. Write a program to get the status of SW and send it to the LED.

Solution:

```
.INCLUDE "M32DEF.INC"
      CBI   DDRB, 0      ;make PB0 an input
      SBI   DDRB, 7      ;make PB7 an output
AGAIN:SBIC  PINB, 0      ;skip next if PB0 is clear
      RJMP  OVER         ;(JMP is OK too)
      CBI   PORTB, 7
      RJMP  AGAIN        ;we can use JMP too
OVER: SBI   PORTB, 7
      RJMP  AGAIN        ;we can use JMP too
```

Example 4-9

A switch is connected to pin PB0. Write a program to get the status of SW and save it in location 0x200.

Solution:

```
.EQU MYTEMP = 0x200      ;set aside location 0x200
.INCLUDE "M32DEF.INC"
       CBI   DDRB, 0     ;make PB0 an input
AGAIN:SBIC   PINB, 0     ;skip next if PB0 is clear
       RJMP  OVER        ;(JMP is OK too)
       LDI   R16, 0
       STS   MYTEMP,R16  ;save it in MYTEMP
       RJMP  AGAIN       ;we can use JMP too
OVER:  LDI   R16,0x1     ;move 1 to R16
       STS   MYTEMP,R16  ;save it in MYTEMP
       RJMP  AGAIN       ;we can use JMP too
```