Assembly Programming through Arduino

G V V Sharma*

Contents

1 Display Control through Hardware

1.1 Powering the Display

The breadboard can be divided into 5 segments. In each of the green segements, the pins are internally connected so as to have the same voltage. Similarly, in the central segments, the pins in each column are internally connected in the same fashion as the blue columns.

Problem 1.1. Plug the display to the breadboard in Fig. 1.1

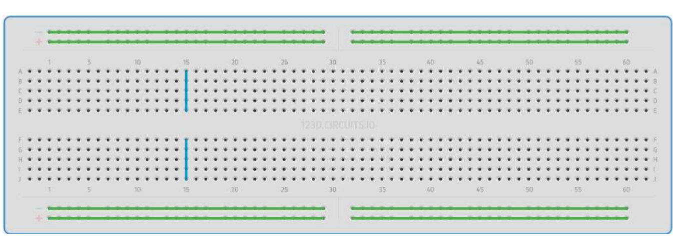

Fig. 1.1

*The author is with the Department of Electrical Engineering, Indian Institute of Technology, Hyderabad 502285 India e-mail: gadepall@iith.ac.in.

The seven segment display in Fig. 1.2 has eight pins, a, b, c, d, e, f, g and *dot* that take an active LOW input, i.e. the LED will glow only if the input is connected to ground. Each of these pins is connected to an LED segment. The *dot* pin is reserved for the · LED.

Problem 1.2. Connect one end of the 1K resistor to the COM pin of the display and the other end to an extreme pin of the breadboard.

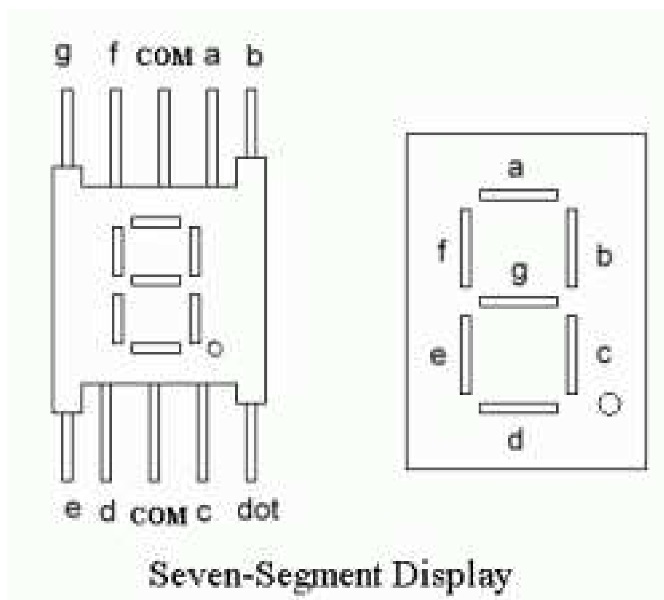

Fig. 1.2

The Arduino Uno has some ground pins, analog input pins A0-A3 and digital pins D1-D13 that can be used for both input as well as output. It also has two power pins that can generate $3.3V$ and $5V$. In the following exercises, only the GND, $5V$ and digital pins will be used.

Problem 1.3. Connect the 5V pin of the arduino to an extreme pin that is in the same segment as the 1K resistor pin.

Problem 1.4. Connect the GND pin of the arduino to the opposite extreme pin of the breadboard

Problem 1.5. Connect the Arduino to the computer.

Problem 1.6. Connect the *dot pin of the display to a pin in the same segment as the GND pin. What do you observe?*

1.2 Software Setup

Problem 1.7. Install AVRA and AVRDUDE on your Linux system through the following commands

```
sudo apt-get install avra avrdude
%Finding the port

sudo dmesg | grep tty
%The output will be something like
[    6.153362] cdc_acm 1-1.2:1.0:
    ttyACM0: USB ACM device
%and your port number is ttyACM0
```

Problem 1.8. Download the m328Pdef.inc file from

```
https://drive.google.com/file/d/0
    B_xAdct0mR1nd3JyT0R2eU1oWWs/
view?usp=sharing
```

and copy into the directory /usr/share/avra/ directory

Problem 1.9. In geany, go to Build→Set Build Commands→Compile and enter
avra "%f"
Then go to Build→Set Build Commands→Execute and enter
avrdude -p atmega328p -c arduino -P /dev/ttyACM0 -b 115200 -U flash:w:%e.hex

Problem 1.10. Now connect the dot pin of the display to pin 13 of the arduino. Execute the following program

```
;hello
;using assembly language for
    turning LED on

.include "/usr/share/avra/m328Pdef
    .inc"

   ldi r16,0b00100000
   out DDRB,r16
   ldi r17,0b00000000
   out PortB,r17
Start:
   rjmp Start
```

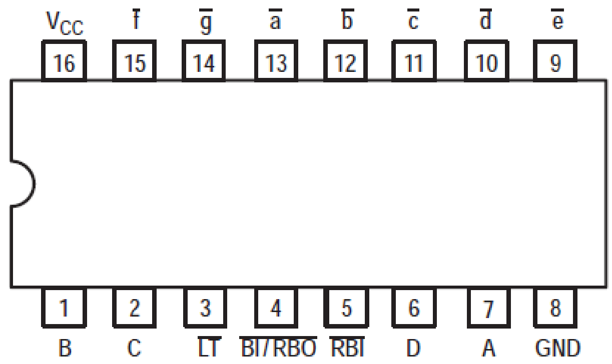

Fig. 1.14

Problem 1.11. Turn the LED off by modifying the above code.

1.3 Controlling the Display

Problem 1.12. Generate the number 1 on the display by connecting the pins $a-g$ to GND according to Table 1.12.

a	b	c	d	e	f	g	decimal
1	0	0	1	1	1	1	1

TABLE 1.12

Problem 1.13. Complete Table 1.12 for all numbers between 0-9.

Problem 1.14. Now generate the numbers from 1-9 on the display using the above table.

The 7447 IC helps in displaying decimal numbers on the seven segment display. The $\bar{a}-\bar{f}$, pins of the 7447 IC are connected to the $a-f$ pins of the display. V_cc should be connected to a 5V power source. The input pins of the decoder are A,B,C and D, with A being the lowest significant bit (LSB) and D being the most significant bit (MSB). For example, the number 5 is visible on the display when the A,B,C and D inputs are the following.

D	C	B	A	Decimal
0	1	0	1	5

Problem 1.15. Connect the 7447 IC decoder $\bar{a}-\bar{g}$ pins to the $a-g$ pins of the display respectively.

Problem 1.16. Connect the V_{cc} and GND pins of the decoder to the 5V supply and GND pins of the breadboard.

Problem 1.17. Connect the A,B,C,D pins to pins in the GND extreme segment of the breadboard. What do you observe.

Problem 1.18. Now remove the D pin from the breadboard and observe the display output.

Problem 1.19. Generate a table with A,B,C,D inputs and the equivalent decimal number output.

2 Display Control through Arduino Software

2.1 *Driving the Segments*

Problem 2.1. Connect the A-D pins of the 7447 IC to the pins D2-D5 of the Arduino.

Problem 2.2. Type the following code and execute. What do you observe? Explain through Fig. 2.4

```
;decoder.asm
;driving the display decoder


.include "/usr/share/avra/m328Pdef.inc"

        ldi r16, 0b00111100 ;identifying output pins 2,3,4,5
        out DDRD,r16        ;declaring pins as output
        ldi r16,0b00010100  ;loading the number 5 in binary
        out PORTD,r16       ;writing output to pins 2,3,4,5

Start:
        rjmp Start
```

Problem 2.3. Now generate the numbers 0-9 by modifying the above program.

Problem 2.4. Type the following code and execute. Comment

```
;logic_xor.asm
;logical XOR

.include "/usr/share/avra/m328Pdef.inc"

          ldi r16,0              ;reset system status
          out SREG,r16           ;init stack pointer
          ldi r16,low(RAMEND)    ;0xff
          out SPL,r16
          ldi r16,high(RAMEND)   ;0x08
          out SPH,r16


        ldi r16, 0b00111100 ;identifying output pins 2,3,4,5
        out DDRD,r16        ;declaring pins as output

        ldi xl,0x00
                    ;loading memory address lower byte into r26
        ldi xh,0x01
                    ;loading memory address higher byte into r27

        ldi r16,0b00000000  ;initializing W
        st x,r16
                    ;storing W in 0x0100 address

        ldi r17, 0x09       ;initialising count

;loading numbers 0-9 into memory locations 0x0100-0x0109
loop_cnt:
        inc r16
                    ;increment register value
        inc xl
                    ;increment address
```

```
        st x,r16                ;store number into memory
        dec r17                 ;decrement count
        brne loop_cnt

;start printing numbers from 0-9
Start:
        ldi r17, 0x0A           ;load the number 10 in r17
        clr xl                  ;reset memory to 0x0100
loop_inc:
        ;writing W to pin 2
        ldi r16, 0b00000010     ;counter = 0
        ld r0,x                 ;load number from memory
loopw:
        lsl r0                  ;left shift
        dec r16                 ;counter --
        brne loopw              ;if counter != 0

        out PORTD,r0            ;writing output to pins 2,3,4,5

        call wait               ;calling delay

        inc xl                  ;incrementing address
        dec r17                 ;decrement decade count
        brne loop_inc           ;branch if decade count !=0

        rjmp Start

;delay routine
wait:
        push r16                ;save register contents
        push r17
        push r18

        ldi r16,0x20            ;loop 0x400000 times
        ldi r17,0x00            ;~12 million cycles
        ldi r18,0x00            ;~0.7s at 16Mhz
w0:
        dec r18
        brne w0
        dec r17
        brne w0
        dec r16
        brne w0

        pop r18                 ;restore register contents
        pop r17
        pop r16
        ret
```

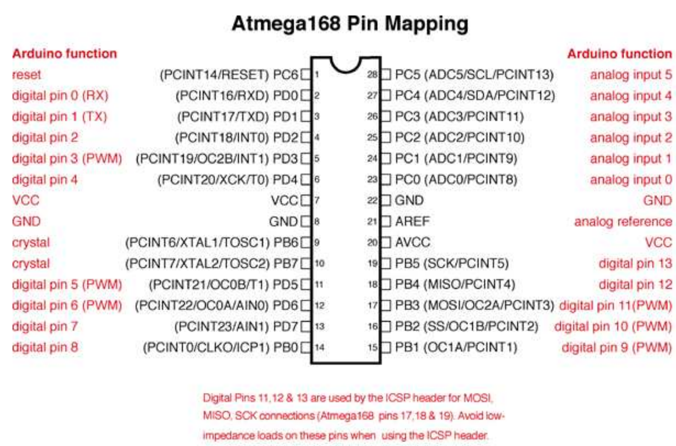

Fig. 2.4

3 Combinational Logic

Test all the following .using the 7447 IC

3.1 Counting Decoder

Problem 3.1. Write an assembly program for the AND operation.

Solution:

```
;logic_and.asm
;logical AND


.include "/usr/share/avra/m328Pdef
   .inc"

        ldi r31, 0b00111100 ;
           identifying output pins
           2,3,4,5
        out DDRD,r31            ;
           declaring pins as output

        ldi r16,0b00000000      ;
           initializing W
        ldi r17,0b00000000      ;
           initializing X
        and r16,r17
                   ;W AND X
        ;complementing W

        ;writing W to pin 2
        ldi r31, 0b00000010     ;
           counter = 0
loopw:  lsl r16                 ;
   left shift
                dec r31
                           ;counter
                   --
                brne    loopw   ;
                   if counter != 0

                out PORTD,r16
                           ;writing
                    output to pins
                   2,3,4,5

Start:
        rjmp Start
```

Problem 3.2. Write an assembly program for the OR operation.

Solution:

```
;logic_or.asm
;logical OR


.include "/usr/share/avra/m328Pdef
   .inc"

        ldi r31, 0b00111100 ;
           identifying output pins
           2,3,4,5
        out DDRD,r31            ;
           declaring pins as output

        ldi r16,0b00000001      ;
           initializing W
       mov r0,r16
        ldi r16,0b00000000      ;
           initializing X
        or r0,r16
                   ;W OR X


        ;writing W to pin 2
        ldi r31, 0b00000010     ;
           counter = 0
loopw:  lsl r0                  ;
   left shift
                dec r31
                           ;counter
                   --
                brne    loopw   ;
                   if counter != 0

                out PORTD,r0
                           ;writing
                    output to pins
                   2,3,4,5

Start:
        rjmp Start
```

Problem 3.3. Write an assembly program for the XOR operation.

Solution:

```
;logic_xor.asm
;logical XOR


.include "/usr/share/avra/m328Pdef
   .inc"

        ldi r16, 0b00111100 ;
           identifying output pins
```

```
        2,3,4,5
    out DDRD,r16                ;
        declaring pins as output

    ldi r16,0b00000001          ;
        initializing W
    mov r0,r16

    ldi r16,0b00000001          ;
        initializing X
    eor r0,r16
                ;W XOR X


    ;writing W to pin 2
    ldi r16, 0b00000010         ;
        counter = 0
loopw:  lsl r0                  ;
   left shift
            dec r16
                        ;counter
                --
            brne    loopw       ;
                if counter != 0

            out PORTD,r0
                        ;writing
                output to pins
                2,3,4,5


Start:
    rjmp Start
```

Problem 3.4. Write an assembly program to complement a bit using the XOR operation.

Problem 3.5. Write an assembly routine to complement a bit. Call this routine by using RCALL.

Problem 3.6. In the truth table in Table 3.7, W, X, Y, Z are the inputs and A, B, C, D are the outputs. This table represents the system that increments the numbers 0-8 by 1 and resets the number 9 to 0 Note that $D = 1$ for the inputs 0111 and 1000. Using *boolean* logic,

$$D = WXYZ' + W'X'Y'Z \tag{3.6.1}$$

Note that 0111 results in the expression $WXYZ'$ and 1000 yields $W'X'Y'Z$. Write the boolean logic functions for A, B, C in terms of W, X, Y, Z.

Z	Y	X	W	**D**	**C**	**B**	**A**
0	0	0	0	**0**	**0**	**0**	**1**
0	0	0	1	**0**	**0**	**1**	**0**
0	0	1	0	**0**	**0**	**1**	**1**
0	0	1	1	**0**	**1**	**0**	**0**
0	1	0	0	**0**	**1**	**0**	**1**
0	1	0	1	**0**	**1**	**1**	**0**
0	1	1	0	**0**	**1**	**1**	**1**
0	1	1	1	**1**	**0**	**0**	**0**
1	0	0	0	**1**	**0**	**0**	**1**
1	0	0	1	**0**	**0**	**0**	**0**

TABLE 3.7

Solution: The desired equations are

$$A = W' \tag{3.6.2}$$
$$B = WX'Z' + W'X \tag{3.6.3}$$
$$C = WXY' + X'Y + W'Y \tag{3.6.4}$$
$$D = WXY + W'Z \tag{3.6.5}$$

Problem 3.7. Write an assembly program for implementing Table 3.7 and verify if your logic is correct by observing the output on the seven segment display.

3.2 Display Decoder

Problem 3.8. Now write the truth table for the seven segment display decoder (IC 7447). The inputs will be A, B, C, D and the outputs will be a, b, c, d, e, f, g.

Problem 3.9. Obtain the logic functions for outputs a, b, c, d, e, f, g in terms of the inputs A, B, C, D.

Problem 3.10. Disconnect the arduino from IC 7447 and connect the pins D2-D8 in the Arduino directly to the seven segment display.

Problem 3.11. Write a new program to implement the logic in Problem 3.9 and observe the output in the display. You have designed the logic for IC 7447!

Problem 3.12. Now include your counting decoder program in the display decoder program and see if the display shows the consecutive number.

4 Decade Counter through Arduino

4.1 Through a Loop

Problem 4.1. Use the following code for LED blinking. You will have to connect pin 13 to the LED on the seven segment display. The blinking delay that you obtain is approximately 0.7 seconds.

```
.include "/usr/share/avra/m328Pdef
   .inc"

;works as 1 second delay on
   arduino: 16 MHz (16x10^6)

sbi DDRB, 1 ;set pin 9 as output
   pin (DDRB pin 1)
ldi r16, 0b00000101 ;the last 3
   bits define the prescaler, 101
   => division by 1024
out TCCR0B, r16
;prescalar used = 1024. So new
   freq. of clock cycle = (16 MHz /
    1024) = 16 kHz

clr r18 ;outpit bits. we are only
   interested in bit 2 from the
   right.

        ldi xl,0x00
                   ;loading memory
           address lower byte into
           r26
        ldi xh,0x01
                   ;loading memory
           address higher byte into
            r27

        ldi r16,0b00000010        ;
           initializing W
        st x,r16
                   ;storing W in 0
           x0100 address


loop:
        ld r16,x
                   ;load number
           from memory
        eor r18, r16              ;
           change the output of LED
        out PORTB, r18
        ldi r19, 0b01000000 ;times
            to run the loop = 64
           for 1 second delay
        rcall PAUSE               ;
           call the PAUSE label
        rjmp loop

PAUSE:  ;this is delay (function)
lp2:    ;loop runs 64 times
                IN r16, TIFR0 ;
                   tifr is timer
                   interupt flag (8
                    bit timer runs
                   256 times)
                ldi r17, 0
                   b00000010
                AND r16, r17 ;need
                    second bit
                BREQ PAUSE
                OUT TIFR0, r17    ;
                   set tifr flag
                   high
        dec r19
        brne lp2
        ret

;prescalar * loop_iterations *
   timer_duration =~ 16 million
   cycles
;16 million cycles at 16 MHz = 1
   second
```

Problem 4.2. Modify the above code to double the blinking delay.

Problem 4.3. A decade counter counts the numbers from 0-9 and then resets to 0. Implement a decade counter using the delay routine in Problem 4.1.

Problem 4.4. Find the exact blinking delay in Problem 4.1.

Problem 4.5. Verify your result through an oscilloscope.

4.2 Through Flip-Flops

The 7474 IC in Fig. 4.6 has two D flip flops. The D pins denote the input and the Q pins denote the output. CLK denotes the clock input.

Problem 4.6. Connect the D2-D5 pins of the arduino to the Q pins of the two 7474 ICs. Use the D2-D5 pins as Arduino input.

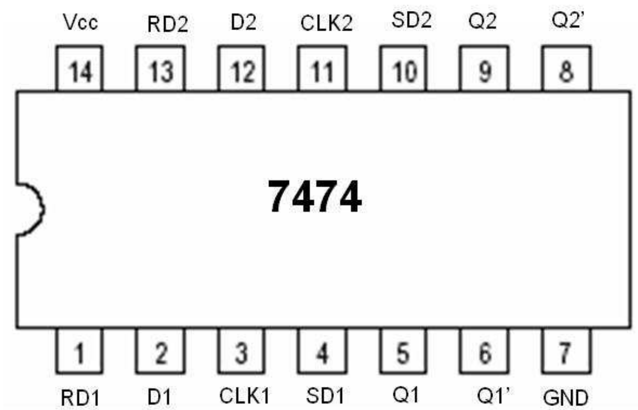

Fig. 4.6

Problem 4.7. Connect the Q pins to IC 7447 Decoder as input pins. Connect the 7447 IC to the seven segment display.

Problem 4.8. Connect the D6-D9 pins of the arduino to the D input pins of two 7474 ICs. Use the D6-D9 pins as Arduino output.

Problem 4.9. Connect pin 13 of the Arduino to the CLK inputs of both the 7474 ICs.

Problem 4.10. Using the logic for the counting decoder in assembly, implement the decade counter using flip-flops.