Resource 4

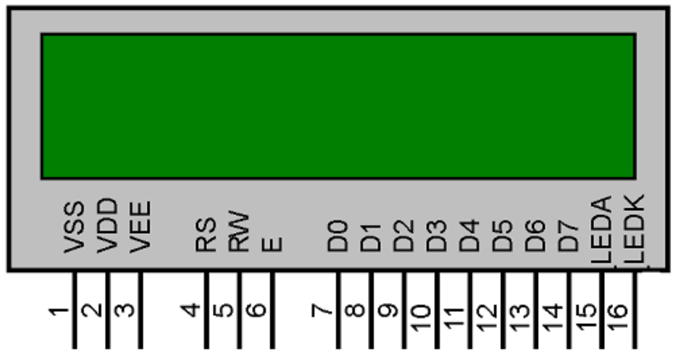
1. The microcontroller is much faster than the LCD.
2. The LCD can inform us of this busy status. So, we will create a function to wait until the LCD is not busy. For the LCD to accept information from the microcontroller, or let it give you information, we must turn its enable pin on and off while the information is present for the LCD to accept. **D7 pin will be ON if the LCD is busy and OFF if the LCD is not busy**
3. There is a **VDD** pin for 5 volts and a **VSS** pin for ground. There is a **V0** pin for the adjustment of the **LCD contrast**. Some LCDs even have an LED backlight and are generally the **last two pins**.
4. The LCD has a row of **8 pins to serve as its port**. The pins that serve as its port are D0-D7. These pins are generally used to pass information into the LCD, but it can also be set to pass information back to the microcontroller (Busy or not).
5. Two types of information can be passed through these pins:
    a. data to display on the LCD screen
    b. control information that is used to do things such as clearing the screen, controlling the cursor position, turning the display on or off, etc.
6. There are other pins on the LCD that help the LCD accept information, and tell the LCD to receive a character or control, or **control the read or write (input or output) function** of the pins.
    a. the read/write is **microcontroller centric**: the LCD "**read**" mode is the process of passing information from the LCD to the microcontroller (microcontroller port set as input or reading or listening).;
    b. The LCD "**write**" mode is passing information from the microcontroller to the LCD (microcontroller set to output or writing).
7. The pin on the LCD that is responsible for the read and write state is labeled **R/W**. The pin on the LCD that is responsible for whether the information sent is a character or a control, is the **RS** pin (Register Select). And the pin that helps the LCD accept information is called the **EN** pin (Enable).
    a. RW = 1 means Read mode else write mode
    b. RS = 0 means command mode
    c. EN = 1 means receive information (character)
8. Process in shortcut:
    a. To make sure the LCD is not busy
    b. Control the LCD's cursor, or display function;
    c. Write a character to the LCD for it to display.
9. Process in broad:
    a. Checking if the LCD is busy (If you try to display a character to the LCD while the LCD is busy, then the LCD will just ignore the character and it will not be displayed).
        i. We set the port to receive data on the microcontroller (Data direction as input).

    ii. We put the LCD in read mode (**RW on**).
    iii. We put the LCD in command mode (**RS off**).
    iv. And the port now magically contains the data from the LCD **(D7 pin will be ON if the LCD is busy and OFF if the LCD is not busy). [0b10000000 or 0b0000000]**

  b. **Send a command to the LCD**
    i. We check to see if the LCD is busy (Perform the steps in #1 above).
    ii. We set the **port direction as output so we can send information to the LCD.**
    iii. We turn RW off so we can write.
    iv. We turn RS off for command mode.
    v. We fire up the data lines with the command we want (simply making the port equal to a number that associates to a specific command).
    vi. We turn on the enable and then turn it off.
    vii. The LCD will magically perform the command.

  c. **Send a character to the LCD:** This is the same as sending a command except the RS is on and the port will equal the character corresponding to the ASCII code.

10. **The only catch is that they must be done in the correct sequence.**
11. **If the LCD is busy, the data will be 0b10000000 (0x80) and if not the data will be 0b00000000 (0x00).**

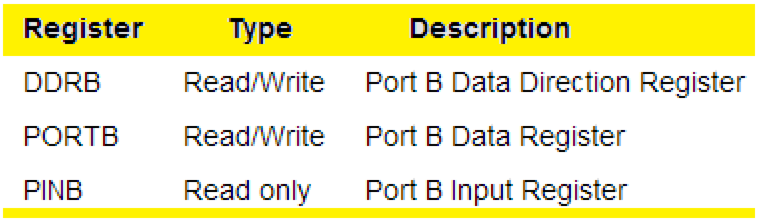


Resource 2

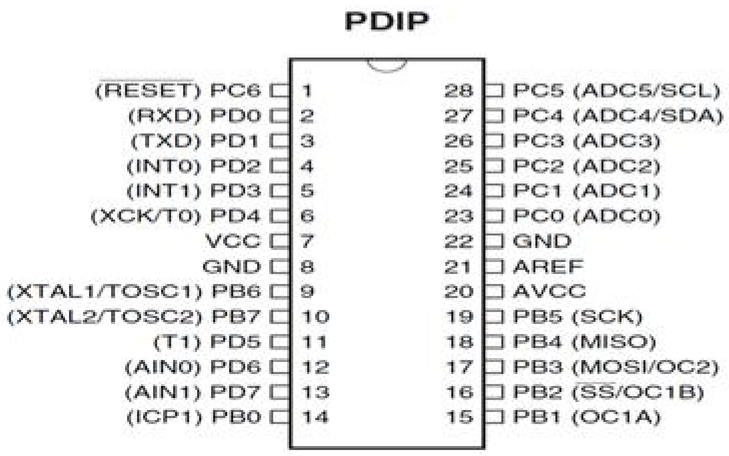
1. ATmega8 microcontroller has 23 programmable input/output (I/O).
2. Atmega8 is available in 2 versions; ATmega8 and Atmega8L
    a. Port B (PB0 to PB7) - 8 pins
    b. Port C (PC0 to PC6) - 7 pins
    c. Port D (PD0 to PD7) - 8 pins
3. **PC6- most often used as the RESET pin, not an I/O**
4. **P1-P2-P3 are Inputs , and others (P0-P4-P5-P6-P7) are Outputs. [for PORTB = 0b11110001]**
    a. **1 for write [output]**

   b. **0 for read [input]**

5. DDRC |= (1<<5) → Shifts binary representation of 1 to left 5 times (i.e. 00000001 to 00100000), which means pin 5 of PORTC (PC5) is configured as an output port!
6. DDRB is the Data Direction register for port "B". This means that if you set this register to 0xFF (by running DDRB |= 0xFF), all ports or pins in the "B" I/O port act as outputs. If you set DDRB to 0x00 (it's initialized to 0x00 by default), then ports or pins in the "B" I/O port act as inputs.
7. After setting the DDRB register,
   a. If it's configured as an output you can then set individual pins of PORTB to high or low voltages. (digitalWrite(8,HIGH) can also be written as PORTB |= (1<<PB0)). To output a low voltage -> PORTB &= ~(1<<PB0). This method clears or sets only the ports that we explicitly want to manipulate.
   b. If it's configured as an input, then you can read a certain pin/port(s) state (high/low).
      i. if (PINB & 0x01 == 0x00) means PB0 is high
      ii. If (PINB & 0x03 == 0x00) means both PB0 and PB1 are high/triggered.
   c. If it's configured as an input and if you set PORTB register bits to 1, then the corresponding pin/port's input pull-up resistor will be enabled.



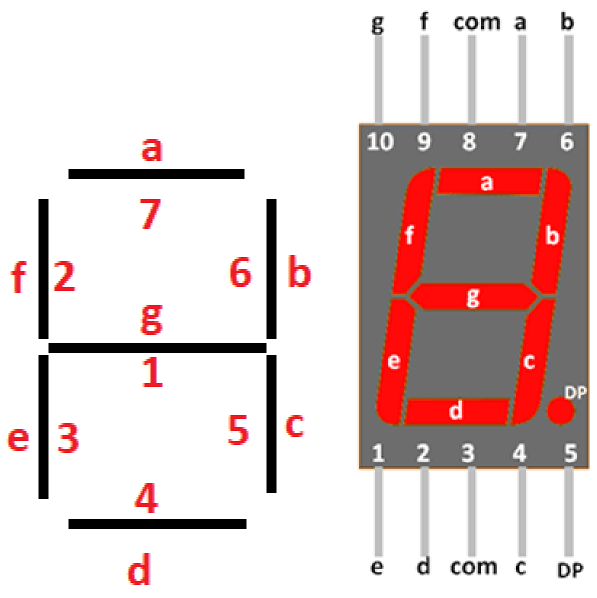
Register	Type	Description
DDRB	Read/Write	Port B Data Direction Register
PORTB	Read/Write	Port B Data Register
PINB	Read only	Port B Input Register



Resource 3

1. The segments in this display are normally single LEDs or liquid crystals
2. **A 7 segment IC comprises of LED or a LCD components**

3. **A 7 segment device is assembled in form of 8 structure**
4. **A 7-segment display comprises 8 numbers of segments.**
5. Each segment or a '**6 sided box**' consists of two thin pieces of metal, usually aluminum.
6. The individual numerals of a 7-segment display are made up of light-emitting diodes.
7. **Pins:**
    a. Pin1 (e): This pin is used to control the left bottom LED of the display
    b. Pin2 (d): This pin is used to control the bottom-most LED of the display
    c. Pin3 (Com): This pin is used to connect to Vcc or Ground-based on display type
    d. Pin4 (c): This pin controls the right bottom LED of the display
    e. Pin5 (DP): This pin controls the decimal point LED of the display
    f. Pin6 (b): This pin controls the top right LED of the display
    g. Pin7 (a): This pin is used to control the topmost LED of the display
    h. Pin8 (Com): This pin is connected to Vcc/GND based on display type
    i. Pin9 (f): This pin controls the top left LED of the display
    j. Pin10 (g): This pin is used to control the middle LED of the display
8. If common anode, the COM should be connected to the VCC (+5V).
9. Other segment displays:
    a. The fourteen-segment display has four additional diagonal bars between the top horizontal and middle vertical segments; it was used on some early pocket calculators.
    b. The sixteen-segment display adds four diagonal bars between each set of two adjacent vertical segments; it is used on some calculators and watches.
10. Seven segment displays are mostly used in electronic meters, digital calculators, clock radios, digital clocks, digital clocks, odometers, etc.
11. https://www.watelectronics.com/mcq/7-segment-display/
12. The advantages of LEDs include small size, compact package and minimum power supply.
13. **8th pin in 7-segment display displays dot. (Don't know why!)**
14. **The first pin of a 7-segment display IC is Vcc. (Don't know why!)**
15. '2' outlines are deployed in a seven segment display namely common anode and common cathode.
16. A 7-segment display can display **A-G, L, T, O, S. [A-J]**
17. The characters a segment display finds **difficult** in displaying are **H, X, 2 and Z.**
18. LEDs blink only on **forward** bias conditions.
19. The forward voltage drop of a red LED is 2 to 2.2 V.
20. '3.6V' is the forward voltage drop of a **blue/white** LED.
21. In common anode based 7-segment LED display low logic is applied to corresponding cathode.
22. A CMOS 4511 is used to drive a 7-segment display.
23. A 7-segment display is replaced with an LCD component.
24. F.W.Wood scientist discovered 8-segment display.
25. The techniques integrated along with a 7-segment display includes **Panaplex, VFD and Numitron.**



Resource 1

1. AVR microcontrollers from ATMEL are 8 Bit RISC controllers. They come in many varieties and most of them give 16 MIPS (Mega Instructions Per Seconds) throughput at 16MHz .
2. **C cross-compilers** provide you the facility to write programs in C and then they convert it to equivalent assembly code.
3. **Hex file actually contains machine code**, and the microcontroller will execute.
4. Three types of registers:(x can be replaced by A,B,C,D as per the AVR you are using)
   a. DDRx register
   b. PORTx register
   c. PINx register
5. **Writing 0 to a bit in DDRx makes the corresponding port pin as input, while writing 1 to a bit in DDRx makes corresponding port pin as output.**
6. PINx (Port IN) used to read data from port pins. In order to read the data from the port pin, first you have to change the port's data direction to input. [DDRB = 0x00]

My understanding:

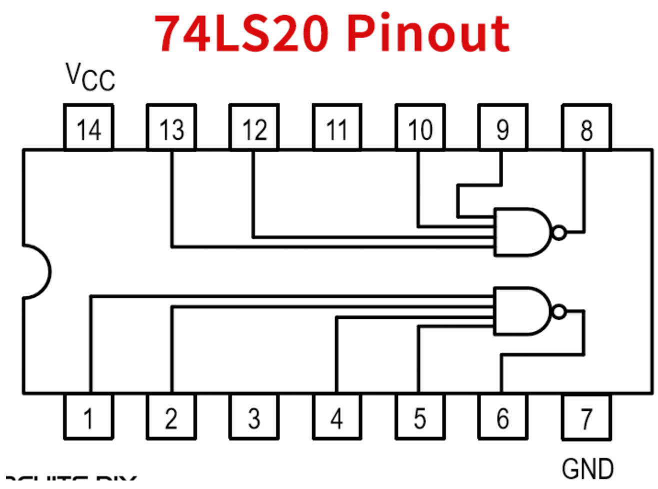
1. First you need to be specific, what are you going to do? Read data via a pin status? Then set DDRx = 0x00. You should connect the Pins of x port with switches. Or if you want to write to the port, then set DDRx = 0xFF. This enables all the pins of port x to be written. So connect all the pins of port x with LEDs.
2. Now if you want to write something to a pin, let's say you want to turn on all the LEDs, then just write PORTx = 0xFF. This line changes the status of pins of port x as 1.

3. Or if you want to read the switch value from a pin, then read like this, y = PINx.

Let's take an example. Suppose you want to read the 0th pin of PORTB which is connected to the switch and based on the switch status you want to turn on one LED which is connected to the 1th pin of PORTB.

```
DDRB = 0b00000001; ///Data Direction as input for bit 1 and
output for bit 0. Rest is not to be considered.
Y = PINB;
if(Y & 0b00000010 == 0b00000010) {    //if 1th pin has ON status
      PORTB = 0b00000001; //
}
```

NAND:



1. How Computer know when a keyboard is pressed?
   Ans: Each key has a scan code number, which corresponds to its position on the keyboard. The keyboard transmits this number as binary data to the computer's CPU. The CPU is running the operating system, which constantly checks for key presses.
2. The bus controller (PCIe or whatever) knows which address ranges go to which peripheral.