Embedded System on AVR Microcontroller (ATMEGA32)

Exp2: Using Registers, SRAM, and Flash Memory to Control Numbers on a 7-Segment Display

Submitted by
Ronit Dutta, MS in IOT and Signal Processing
Department of Electrical Engineering, IIT Kharagpur

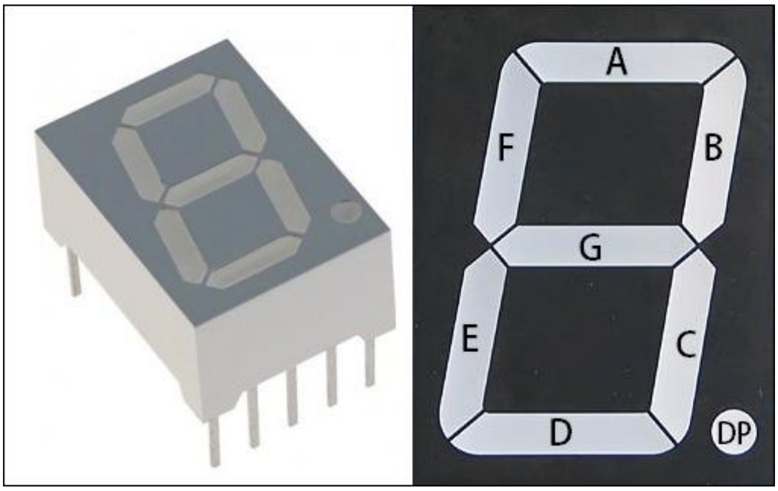
Under the guidance of
Aurobinda Routray, Professor, Electrical Engineering

Department of Electrical Engineering
Indian Institute of Technology Kharagpur

January, 2025

Seven Segment LED Display

Light Emitting Diode (LED) is the most widely used semiconductor which emits either visible light or invisible infrared light when forward biased. A Light-emitting diode (LED) is optical-electrical energy into light energy when voltage is applied. A seven-segment LED is a digital display module specialized to display numerical information.
Light-emitting diodes (LEDs) arranged in the shape of numbers offer an easily visible display. They are sometimes called "seven-segment displays" or "seven-segment indicators."
The 7-segment display consists of seven LEDs (hence its name) arranged in a rectangular fashion as shown. Each of the seven LEDs is called a segment because when illuminated the segment forms part of a numerical digit (both Decimal and Hex) to be displayed.
An additional 8th LED is sometimes used within the same package thus allowing the indication of a decimal point, (DP) when two or more 7-segment displays are connected together to display numbers greater than ten.



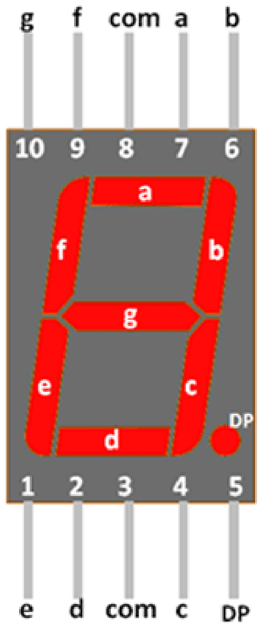
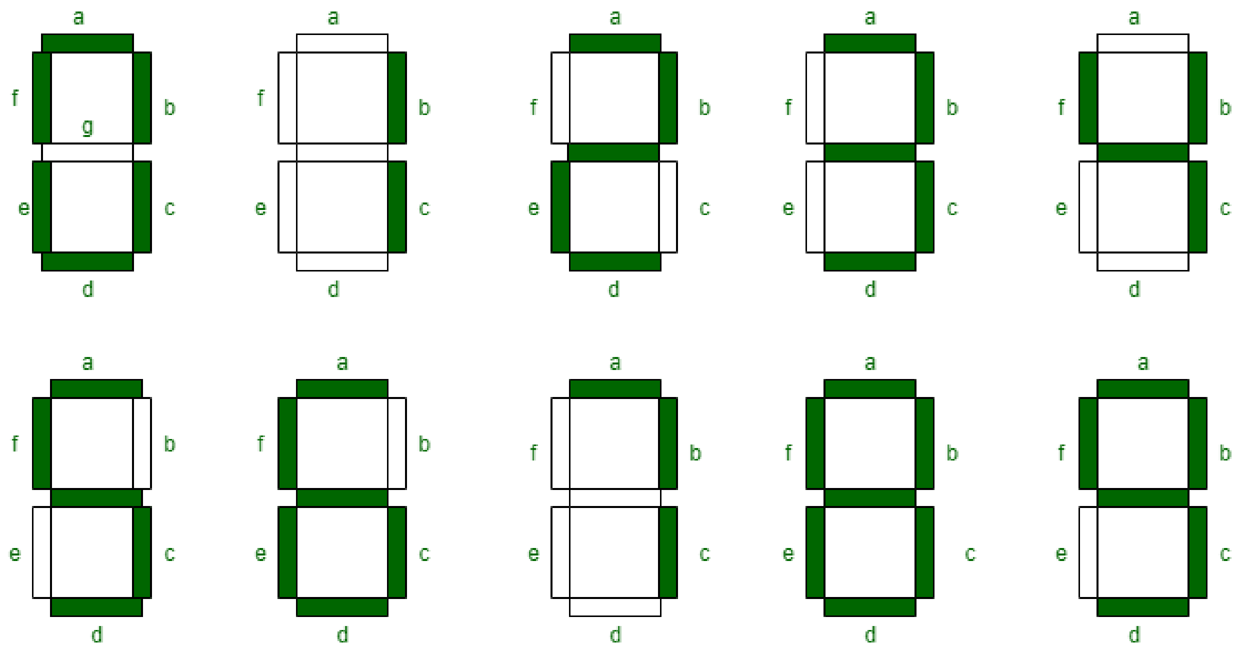


Fig.1: Seven Segment Display Package with pinout

The parts of the seven-segment LED are as follows:

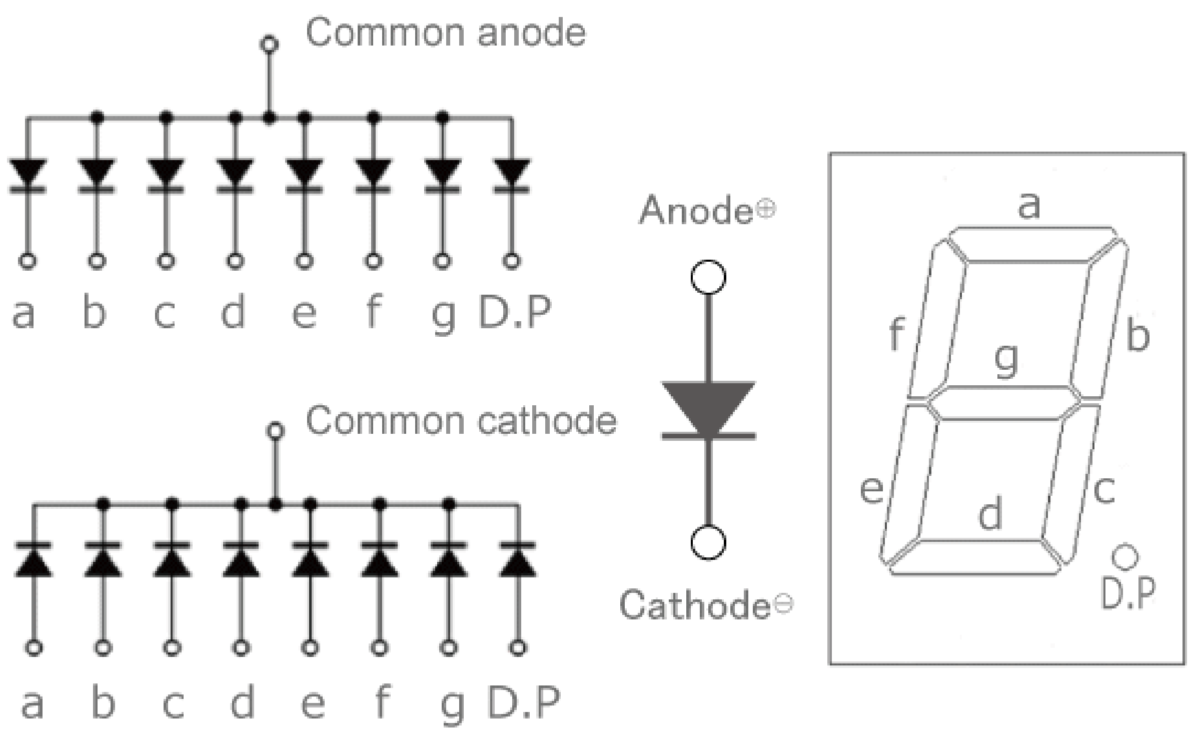
I. **Light-emitting components (a–g): 7 Segments (Seg)**
II. **Dot light-emitting component: Decimal point (DP)**
III. **Common LED Pins: 2 pins from same common node**

So by forward biasing the appropriate pins of the LED segments in a particular order, some segments will be light and others will be dark allowing the desired character pattern of the number to be generated on the display. This then allows us to display each of the ten decimal digits 0 through to 9 on the same 7-segment display.



As each LED has two connecting pins, one called the “Anode” and the other called the “Cathode”, there are therefore two types of LED 7-segment display called: Common Cathode (CC) and Common Anode (CA). The displays common pin is generally used to identify which type of 7-segment display it is.

Common Cathode (CC): In the common cathode display, all the cathode connections of the LED segments are joined together to logic “0” or ground. The individual segments are illuminated by application of a “HIGH”, or logic “1” signal via a current limiting resistor to forward bias the individual Anode terminals (a-g).



Common Anode (CA): In the common anode display, all the anode connections of the LED segments are joined together to logic “1”. The individual segments are illuminated by applying a ground, logic “0” or “LOW” signal via a suitable current limiting resistor to the Cathode of the particular segment (a-g).

In the LAB, the Common Cathode(CC) Seven Segment Display is used for the experiment purpose.

- **Truth Table for Common Cathode Display**

Decimal Digit	Individual Segments Illuminated						
	a	b	c	d	e	f	g
0	1	1	1	1	1	1	0
1	0	1	1	0	0	0	0
2	1	1	0	1	1	0	1
3	1	1	1	1	0	0	1
4	0	1	1	0	0	1	1
5	1	0	1	1	0	1	1
6	1	0	1	1	1	1	1
7	1	1	1	0	0	0	0
8	1	1	1	1	1	1	1
9	1	1	1	1	0	1	1

Therefore, Boolean expressions for each decimal digit that requires respective light-emitting diodes (LEDs) are ON or OFF. Seven segment displays must be controlled by other external devices where different types of microcontrollers are useful to communicate with these.

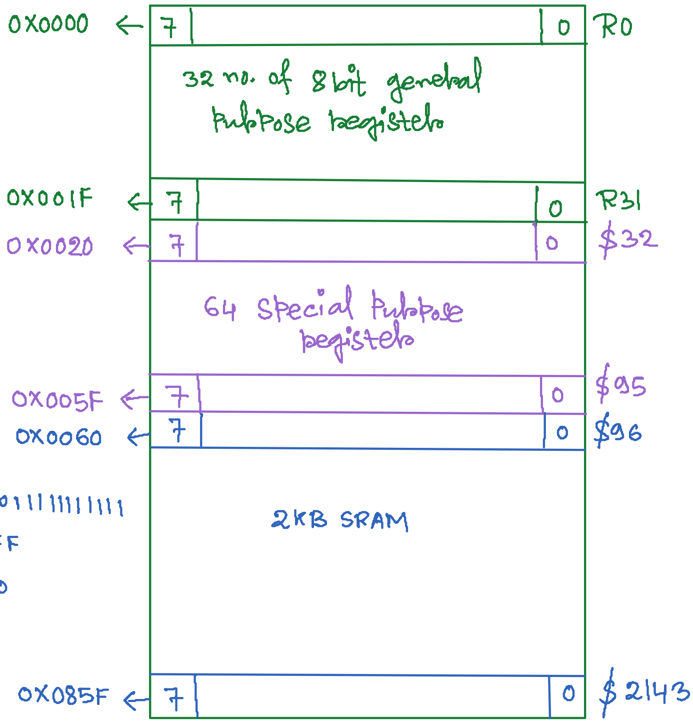
Memory Organization

Tuesday, May 7, 2024 10:51 AM

Data Memory

Registers Arrangement:—

- Each segment of these registers are 8bits/1byte wide.
- Thirty Two (R0 – R31) 8bit general purpose registers can be usualy accessed while doing assembly level coding.
- 64- Special purpose registers are associated with MCU peripherals.



(2KB – 1) = 0b0000011111111111
= 0X07FF

0X07FF + 0X0060
= 0X 085F

$2KB = 2 \times 2^{10}$
= 2048

2048 + 95 = 2143

SRAM:—

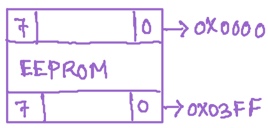
SRAM is 2KB in size and its starting address is 96 in decimal and 0X0060 in hexadecimal.

Final address of SRAM = 96 + (2 × 1024 – 1)
= 2143 in decimal
or
= 0X0060 + 0X07FF
= 0X085F in hexadecimal

(Not for ATMEGA32)
Note:— Extra registers (like another UART, SPI, I2C etc.) will be after SRAM and can go up to address of 65535 in decimal or 0XFFFF in hexadecimal.

EEPROM:— Electrically Erasable Programmable Read-only Memory

- It is kept as a separate memory space for permanent data storage.
- Byte addressable i.e. each segment is in 8bit/1byte format.
- 1KB size ⇒ starting address = 0X0000
  Ending address = 0X03FF



- ## Seven Segment LED Display to Increment the Number Memory Mapped (General Purpose Register)

  Disadvantage: The general-purpose registers, which are currently being filled with values, possess calculation capabilities, but we are losing access to these essential computational functions.

```
// Seven Segment Display Increment

.INCLUDE "M32DEF.INC"
.ORG 0X0000

LDI R16,HIGH(RAMEND)
OUT SPH,R16
LDI R16,LOW(RAMEND)
OUT SPL,R16

LDI R16,0x3F //Seven Segment Bits for 0
MOV R0,R16

LDI R16,0x06 //Seven Segment Bits for 1
MOV R1,R16

LDI R16,0x5B //Seven Segment Bits for 2
MOV R2,R16

LDI R16,0x4F //Seven Segment Bits for 3
MOV R3,R16

LDI R16,0x66 //Seven Segment Bits for 4
MOV R4,R16

LDI R16,0x6D //Seven Segment Bits for 5
MOV R5,R16

LDI R16,0x7D //Seven Segment Bits for 6
MOV R6,R16

LDI R16,0x07 //Seven Segment Bits for 7
MOV R7,R16

LDI R16,0x7F //Seven Segment Bits for 8
```

```
MOV R8,R16

LDI R16,0x6F //Seven Segment Bits for 9
MOV R9,R16

LDI R16,0XFF
OUT DDRA,R16

LDI R27,0x00; // XH of the register pair X
LDI R26,0x00; // XL of the register pair X

MAIN:    LD R16,X+
         OUT PORTA,R16
         CALL Delay
         CPI R26,0x0A
         BRNE MAIN
         LDI R26,0x00
         JMP MAIN

Delay:   LDI R17,0xFF
         L1:  LDI R18,0xFF
         L2:  LDI R19,0x04
         L3:  NOP
              DEC R19
              BRNE L3
              DEC R18
              BRNE L2
              DEC R17
              BRNE L1
              RET
```

- ## Instruction Sets of ATmega32

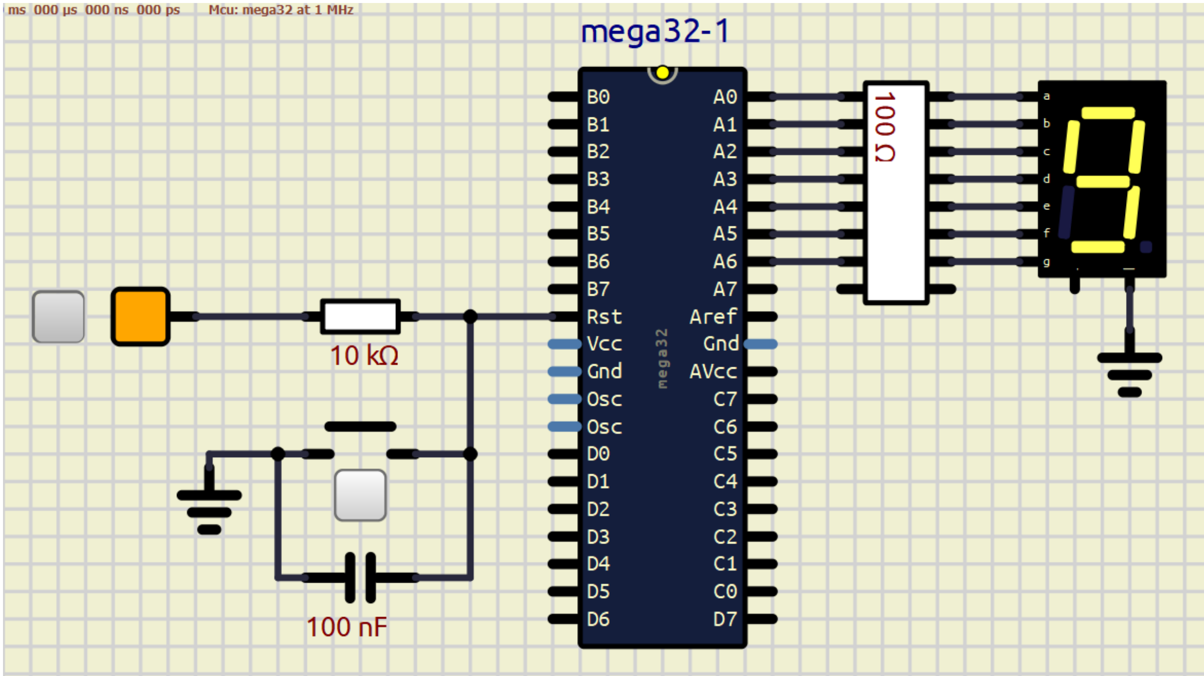
  https://onlinedocs.microchip.com/oxy/GUID-0B644D8F-67E7-49E6-82C9-1B2B9ABE6A0D-en-US-23/GUID-5E958E2F-B101-43C6-8047-C1868B292921.html

- ## Make the below Circuit on SimulIDE to simulate

  Components Required:

  1. ATmega32
  2. Seven Segment LED Display
  3. 100 Ohm DIP Resistor

4. Fixed Voltage
5. Push Button
6. 10KOhm Resistor
7. 100nF Capacitor



After uploading the HEX file, verify the simulation by TA.

Assignment1: Write the C code for the above Experiment

- ## Seven Segment LED Display to Decrement the Number Memory Mapped (General Purpose Register)

  Disadvantage: The general-purpose registers, which are currently being filled with values, possess calculation capabilities, but we are losing access to these essential computational functions.

```
// Seven Segment Display Decrement

.INCLUDE "M32DEF.INC"
.ORG 0X0000

LDI R16,HIGH(RAMEND)
OUT SPH,R16
LDI R16,LOW(RAMEND)
OUT SPL,R16
```

```
LDI R16,0x3F //Seven Segment Bits for 0
MOV R0,R16

LDI R16,0x06 //Seven Segment Bits for 1
MOV R1,R16

LDI R16,0x5B //Seven Segment Bits for 2
MOV R2,R16

LDI R16,0x4F //Seven Segment Bits for 3
MOV R3,R16

LDI R16,0x66 //Seven Segment Bits for 4
MOV R4,R16

LDI R16,0x6D //Seven Segment Bits for 5
MOV R5,R16

LDI R16,0x7D //Seven Segment Bits for 6
MOV R6,R16

LDI R16,0x07 //Seven Segment Bits for 7
MOV R7,R16

LDI R16,0x7F //Seven Segment Bits for 8
MOV R8,R16

LDI R16,0x6F //Seven Segment Bits for 9
MOV R9,R16

LDI R16,0XFF
OUT DDRA,R16

LDI R27,0x00; // XH of the register pair X
LDI R26,0x0A; // XL of the register pair X

MAIN:     LD R16,-X
          OUT PORTA,R16
          CALL Delay
          CPI R26,0X00
          BRNE MAIN
```

```
            LDI R26,0x0A
            JMP MAIN

Delay:      LDI R17,0xFF
            L1:  LDI R18,0xFF
            L2:  LDI R19,0x04
            L3:  NOP
                 DEC R19
                 BRNE L3
                 DEC R18
                 BRNE L2
                 DEC R17
                 BRNE L1
                 RET
```

Upload the HEX file in the same circuit on SimulIDE discussed above and verify the simulation by TA.

Assignment2: Write the C code for the above Experiment

- ## A Real-Time Example: Road Traffic Signaling

```
// Road Traffic  Signaling

.INCLUDE "M32DEF.INC"
.ORG 0X0000

LDI R16,HIGH(RAMEND)
OUT SPH,R16
LDI R16,LOW(RAMEND)
OUT SPL,R16

LDI R16,0x3F //Seven Segment Bits for 0
MOV R0,R16

LDI R16,0x06 //Seven Segment Bits for 1
MOV R1,R16

LDI R16,0x5B //Seven Segment Bits for 2
MOV R2,R16

LDI R16,0x4F //Seven Segment Bits for 3
MOV R3,R16
```

```
LDI R16,0x66 //Seven Segment Bits for 4
MOV R4,R16

LDI R16,0x6D //Seven Segment Bits for 5
MOV R5,R16

LDI R16,0x7D //Seven Segment Bits for 6
MOV R6,R16

LDI R16,0x07 //Seven Segment Bits for 7
MOV R7,R16

LDI R16,0x7F //Seven Segment Bits for 8
MOV R8,R16

LDI R16,0x6F //Seven Segment Bits for 9
MOV R9,R16

LDI R16,0XFF
OUT DDRA,R16  //For 7 Segment Display

LDI R16,0x03
OUT DDRC,R16  //For Signaling LEDs

LDI R27,0x00; // XH of the register pair X
LDI R26,0x0A; // XL of the register pair X

MAIN:    SBI PORTC,PINC0
         CBI PORTC,PINC1
MLOOP1:  LD R16,-X
         OUT PORTA,R16
         CALL Delay
         CPI R26,0X00
         BRNE MLOOP1
         LDI R26,0x0A

         CBI PORTC,PINC0
         SBI PORTC,PINC1
MLOOP2:  LD R16,-X
         OUT PORTA,R16
         CALL Delay
```

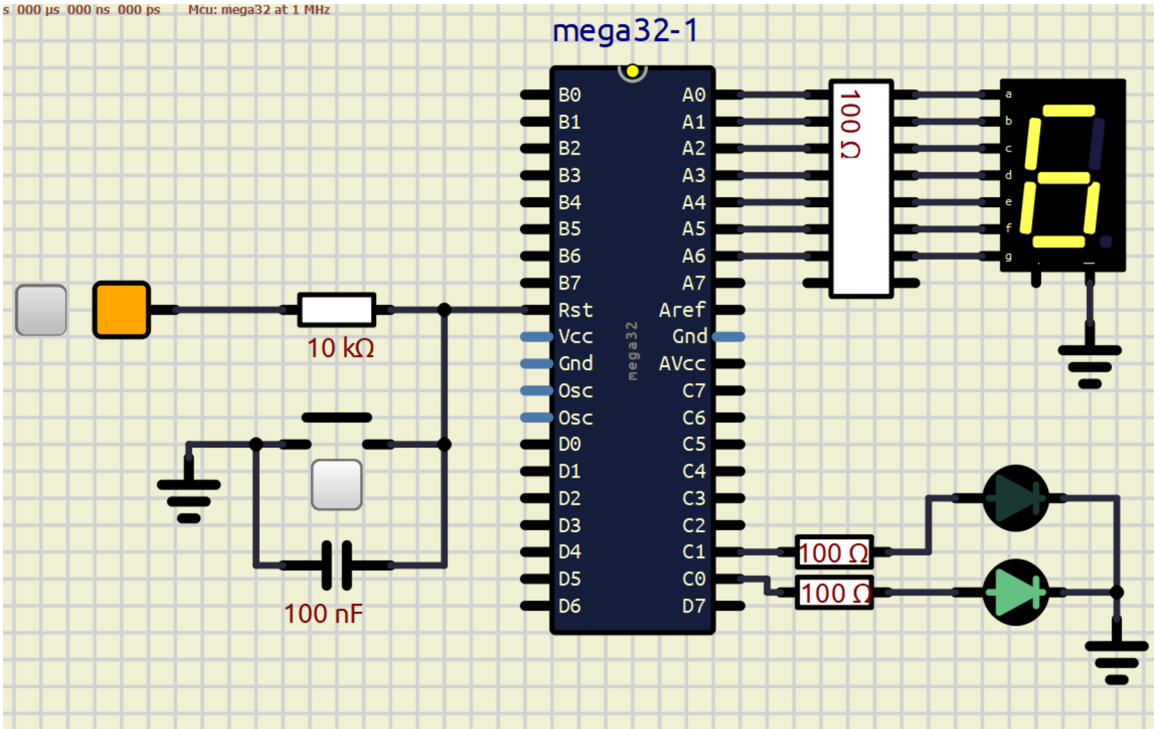
```
        CPI R26,0X00
        BRNE MLOOP2
        LDI R26,0x0A
        JMP MAIN

Delay:  LDI R17,0xFF
        L1:  LDI R18,0xFF
        L2:  LDI R19,0x04
        L3:  NOP
             DEC R19
             BRNE L3
             DEC R18
             BRNE L2
             DEC R17
             BRNE L1
             RET
```

- **Make the below Circuit on SimulIDE to simulate**

Components Required:

1. ATmega32
2. Seven Segment LED Display
3. 100 Ohm DIP Resistor
4. Fixed Voltage
5. Push Button
6. 10KOhm Resistor
7. 100nF Capacitor
8. Two 100 Ohm Resistor
9. One Green LED
10. One Red LED



Upload the HEX file in the same circuit on SimulIDE discussed above and verify the simulation by TA.

Assignment3: Write the C code for the above Experiment

- ## Seven Segment LED Display to Increment the Number Memory Mapped (SRAM)

Disadvantage: The same information is stored in two places: SRAM and Program Memory (Flash Memory).

```
// Write on SRAM Memory then Read
// Address 0x0080 to 0x0089
// Seven Segment Increment

.INCLUDE "M32DEF.INC"
.ORG 0x0000

LDI R16,HIGH(RAMEND)
OUT SPH,R16
LDI R16,LOW(RAMEND)
OUT SPL,R16

LDI R16,0xFF
OUT DDRA,R16
```

```
LDI R27,0x00  // For higher byte of register pair X
LDI R26,0x80  // For lower byte of register pair X

LDI R16,0x3F //Seven Segment Bits for 0
ST X+,R16

LDI R16,0x06 //Seven Segment Bits for 1
ST X+,R16

LDI R16,0x5B //Seven Segment Bits for 2
ST X+,R16

LDI R16,0x4F //Seven Segment Bits for 3
ST X+,R16

LDI R16,0x66 //Seven Segment Bits for 4
ST X+,R16

LDI R16,0x6D //Seven Segment Bits for 5
ST X+,R16

LDI R16,0x7D //Seven Segment Bits for 6
ST X+,R16

LDI R16,0x07 //Seven Segment Bits for 7
ST X+,R16

LDI R16,0x7F //Seven Segment Bits for 8
ST X+,R16

LDI R16,0x6F //Seven Segment Bits for 9
ST X,R16

LDI R26,0x80; // Make again XL=0x80

MAIN:    LD R16,X+
         OUT PORTA,R16
         CALL Delay
         CPI R26,0x8A
         BRNE MAIN
         LDI R26,0x80
```

```
        JMP MAIN

Delay:      LDI R17,0xFF
        L1: LDI R18,0xFF
        L2: LDI R19,0x04
        L3: NOP
            DEC R19
            BRNE L3
            DEC R18
            BRNE L2
            DEC R17
            BRNE L1
            RET
```

Upload the HEX file in the increment counter circuit on SimulIDE discussed above and verify the simulation by TA.

Memory Organization

Thursday, May 2, 2024 8:24 PM

Atmega32 has Harvard architecture. It has separate memory location for program and data.

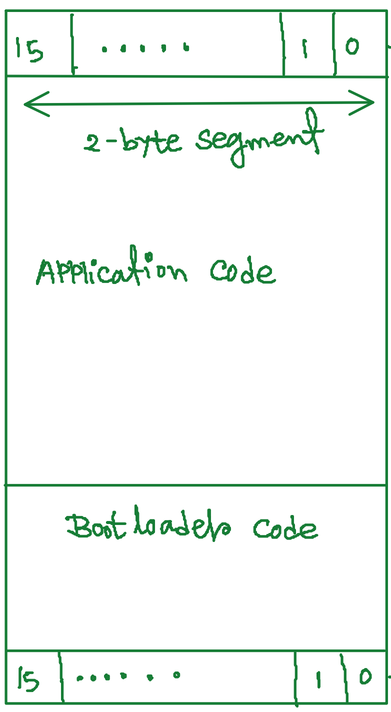
ATMEGA32 Memory
- Program memory
  - Flash Memory
- Data memory
  - Registers
    - General Purpose
    - Special Purpose
  - SRAM
  - EEPROM

Flash Memory :- (32 KB size)

Each segment of flash memory is arranged in the form of two 8 bit segments i.e. 2 bytes or 16 bits.

Flash memory is word addressable (1 word = 16 bits)

→ Starting Address = 0X0000



→ Ending Address = 0X3FFF

The program counter (PC) is 14-bit wide to accumulate flash memory address.

0X3FFF → Ending Address

$= (2^{14}-1)$ → only 14-bit wide is sufficient instead of 16bit.

Starting Address of flash memory = 0X0000

Since each segment is arranged in word format, therefore the final segment address of flash memory $= \frac{32k}{2} - 1$

$= 16k - 1$

$= (16 \times 1024) - 1$

$= 16383$ in decimal

$= 0X3FFF$ in Hexadecimal

* If bootloader is not being used to feed program into the controller, then entire flash memory can be used for storing application code.

- ## Seven Segment LED Display to Increment the Number Memory Mapped (Flash Memory)

  Advantage: The same information is stored only in one place: Program Memory (Flash Memory).

```
// Flash Memory Read
// Address 0x0400 to 0x0404
// Seven Segment Increment

.INCLUDE "M32DEF.INC"
.ORG 0x0000

LDI R16,HIGH(RAMEND)
OUT SPH,R16
LDI R16,LOW(RAMEND)
OUT SPL,R16

LDI R16,0xFF
OUT DDRA,R16

LDI R31,HIGH(seven_segment<<1)     // For higher byte of register pair Z
LDI R30,LOW(seven_segment<<1)      // For lower byte of register pair Z

MAIN:           LPM R16,Z+
                OUT PORTA,R16
                CALL Delay
                CPI R30,0x0A
                BRNE MAIN
                LDI R30,LOW(seven_segment<<1)
                JMP MAIN

Delay:          LDI R17,0xFF
                L1:     LDI R18,0xFF
                L2:     LDI R19,0x04
                L3:     NOP
                        DEC R19
                        BRNE L3
                        DEC R18
                        BRNE L2
                        DEC R17
                        BRNE L1
                        RET

.ORG 0x0400
seven_segment: .dB 0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07, 0x7F, 0x6F
```

Upload the HEX file in the increment counter circuit on SimulIDE discussed above and verify the simulation by TA.

- ## Hardware Simulation of Experiments

  Components Required:

  1. ATmega32
  2. AVR development board
  3. USBasp

4. Seven Segment LED Display (Common Cathode)
5. 100 Ohm Resistors: 9Pcs
6. One Green LED
7. One Red LED
8. Bread Board
9. Jumper Wires Female to Male: 10Pcs
10. Single Stand Wires as Required

Make the circuit on bread board as the Last Experiment on SimulIDE and Flash the Microcontroller with respective HEX file.

Assignment4: Experiment on SimulIDE with Double Digit Multiplexed Display for the Road Traffic Signalling to decrement the digit value from 30 to 00.