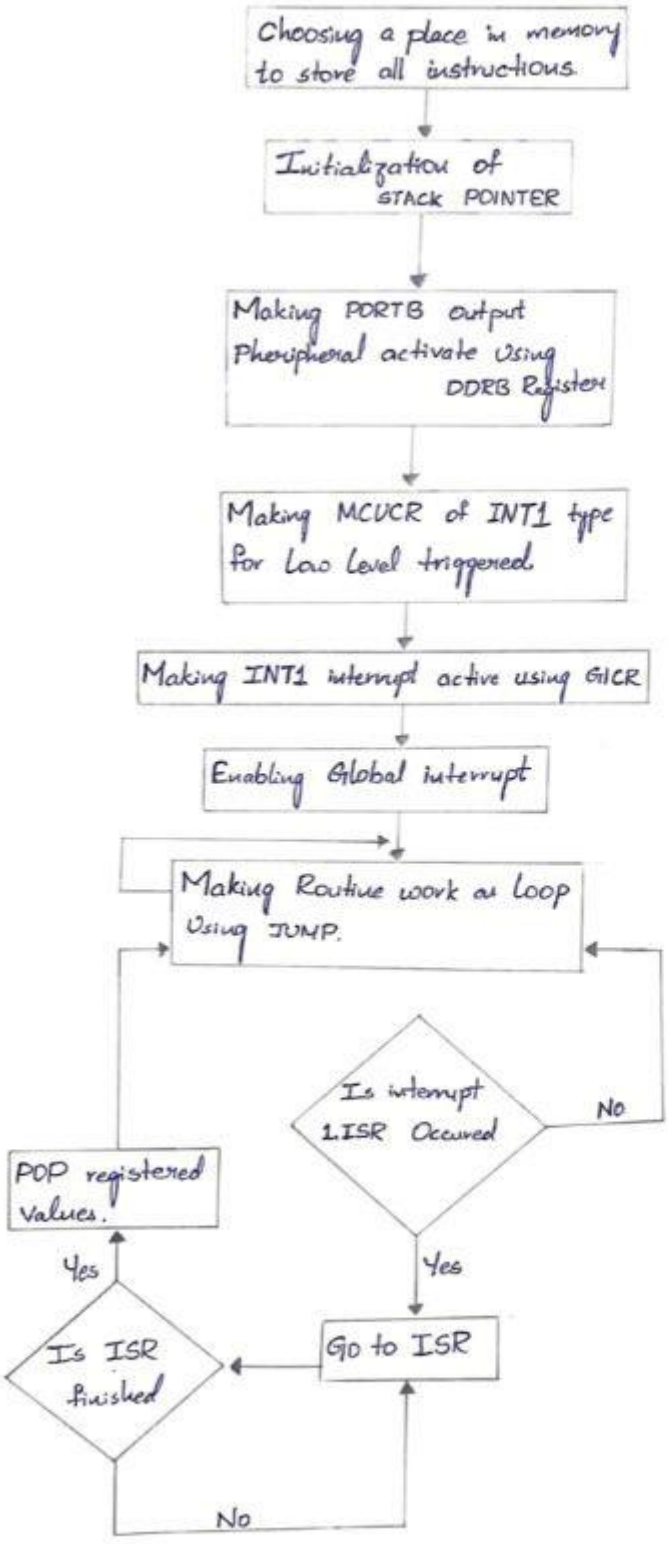
BLINKING OF LED USING INT1 AND INT0 INTERRUPTS

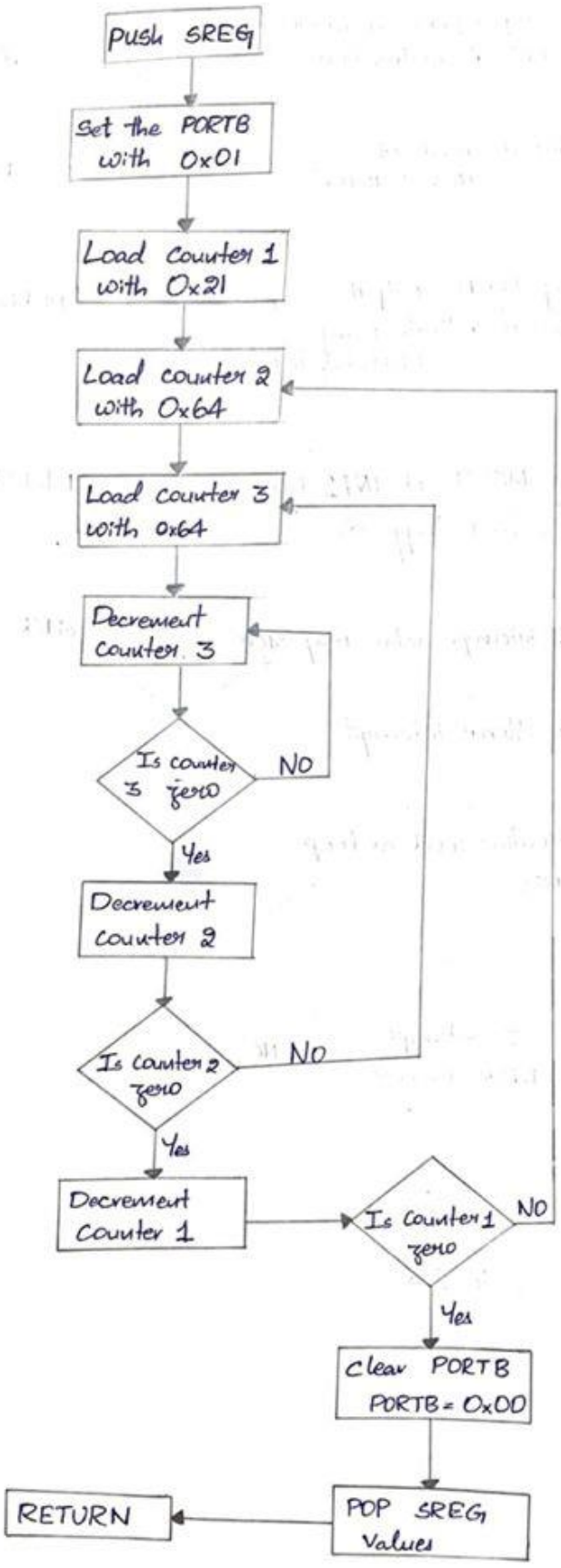
- The target of this experiment is using an external interrupt concept making LED to blink one second when the user gives the input and in normal case without interrupt the LED is in low mode.
- In this experiment the external signal given by the user causes the interrupt for the routine program of the processor. When interrupt occurs the PC points to the vector table of that particular interrupt, in that table the instruction directs to the other location where the ISR program stores and from there the ISR program starts executing. So, now here when interrupt occurs it moves to address 0x0002 as INT1 occurs from there it moves to the ISR address.

- Now to access the INT1 we need to enable the GICR register of that, and also we need to enable the global interrupt in status register. The MCUCR register tells about the mode of detection of interrupt here I am making it low level triggered.
- To take in input or give output the DDRX register plays a key role when the DDRX register is active then the output from the controller goes outside. If it is zero then input is activated.

Flow chart code-1



ISR Program


Push SREG
Set the PORTB with 0x01
Load Counter 1 with 0x21
Load counter 2 with 0x64
Load counter 3 with 0x64
Decrement counter 3
Is counter 3 zero
NO
Yes
Decrement counter 2
Is counter 2 zero
NO
Yes
Decrement Counter 1
Is counter 1 zero
NO
Yes
Clear PORTB PORTB = 0x00
POP SREG Values
RETURN

Using INT1

```
;
; EE2016F21Exp2int1.asm
;
; Created: 25-09-2021 11:11:00
; Author : Bandi Yaswanth

;


; Replace with your application code
start:
  #include "m8def.inc"

  .org 0;
  rjmp reset;

  .org 0x002;
  rjmp int1_ISR;

  .org 0x0100;

  reset:
              LDI R16,0x70;             initializing  the stack pointer address
              OUT SPL,R16,

              LDI R16,0x00;
              OUT SPH,R16;

              LDI R16,0x01;             PORTB as output
              OUT DDRB,R16;

              LDI R16,0x00;
              OUT DDRD,R16;

              IN R16,MCUCR;             load MCUCR register
              ORI R16,0x00;             logical OR with 00000000 and store it in
R16,this is to make int1 level triggered
              OUT MCUCR,R16;

              IN R16,GICR;        load GICR register
              ORI R16,0x80;             this is to make int1 high 10000000
              OUT GICR,R16;

              LDI R16,0x00;
              OUT PORTB,R16;

              SEI;
 ind_loop:rjmp IND_LOOP;          make cpu busy

 int1_ISR:IN R16,SREG;
              PUSH R16;           copy status register to R16 then push R16 to stack

              LDI R16,0x0A;
              MOV R0,R16;

         c1:LDI R17,0x01;
            OUT PORTB,R16;        make PORTB pins to high
```

```
            LDI R16,0x21;              delay for 1 second
          p:LDI R17,0x64;
          q:LDI R18,0x64;
          r:DEC R18;
            BRNE r;
               DEC R17
               BRNE q;
               DEC R16;
               BRNE p;

               LDI R16,0x00;
               OUT PORTB,R16;          make PORTB pins to low
                LDI R16,0x21;          delay for 1 sec
          a:LDI R17,0x64;
          b:LDI R18,0x64;
          c:DEC R18;
            BRNE c;
               DEC R17
               BRNE b;
               DEC R16;
               BRNE a;


               DEC R0;
               BRNE c1;
               POP R16;
               OUT SREG,R16;           pop R16 from stack

               RETI;
```

INT1 in C

```
/*
 * EE2016F21Exp2.int1.c
 *
 * Created: 27-09-2021 12:19:22
 * Author : Bandi Yaswanth
 */
#define F_CPU 1000000       //clock frequency
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

ISR (INT1_vect)
{
       int i;
       for (i=1;i<=10;i++)

       {
              PORTB=0x01;
              _delay_ms(1000);
              PORTB=0x00;
              _delay_ms(1000);
       }
}
int main(void)
{
       //set the input/output pins appropriately
       //to enable interrupt and port interfacing
       //for LED to blink
```

```
        DDRD=0x00;    //set appropriate data direction for D
        DDRB=0x01;    //make PB0 as output
        MCUCR=0x00;   //set MCUCR to level triggered
        GICR=0x80;    //enable interrupt 1
        PORTB=0x00;
        sei();        //global interrupt flag

        while (1)     //wait
        {

        }
}
```

Using INT0

```
;
; EE2016F21Exp2int0.asm
;
; Created: 27-09-2021 12:11:44
; Author : Bandi Yaswanth
;


; Replace with your application code
start:
     #include "m8def.inc"

  .org 0;
  rjmp reset;

  .org 0x001;
  rjmp int0_ISR;

  .org 0x0100;

  reset:
              LDI R16,0x70;             initializing  the stack pointer address
              OUT SPL,R16,

              LDI R16,0x00;
              OUT SPH,R16;

              LDI R16,0x01;             PORTB as output
              OUT DDRB,R16;

              LDI R16,0x00;
              OUT DDRD,R16;             PORTD as input

              IN R16,MCUCR;             load MCUCR register
              ORI R16,0x00;             logical OR with 00000000 and store it in
R16,this is to make int1 level triggered
              OUT MCUCR,R16;

              IN R16,GICR;        load GICR register
              ORI R16,0x40;             this is to make int1 high 01000000
              OUT GICR,R16;

              LDI R16,0x00;
              OUT PORTB,R16;

              SEI;
```

```
ind_loop:rjmp IND_LOOP;          make cpu busy

int1_ISR:IN R16,SREG;
              PUSH R16;          copy status register to R16 then push R16 to stack

              LDI R16,0x0A;
              MOV R0,R16;

        c1:LDI R17,0x01;
           OUT PORTB,R16;        make PORTB pins to high

              LDI R16,0x21;             delay for 1 second
         p:LDI R17,0x64;
         q:LDI R18,0x64;
         r:DEC R18;
           BRNE r;
              DEC R17
              BRNE q;
              DEC R16;
              BRNE p;

              LDI R16,0x00;
              OUT PORTB,R16;            make PORTB pins to low
               LDI R16,0x21;            delay for 1 sec
         a:LDI R17,0x64;
         b:LDI R18,0x64;
         c:DEC R18;
           BRNE c;
              DEC R17
              BRNE b;
              DEC R16;
              BRNE a;


              DEC R0;
              BRNE c1;
              POP R16;
              OUT SREG,R16;             pop R16 from stack

              RETI;
```

INT0 in C

```
/*
 * EE2016F21Exp2.int0.c
 *
 * Created: 27-09-2021 12:33:36
 * Author : Bandi Yaswanth
 */

#define F_CPU 1000000       //clock frequency
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

ISR (INT0_vect)
{
       int i;
       for (i=1;i<=10;i++)

       {
```

```
                PORTB=0x01;
                _delay_ms(1000);
                PORTB=0x00;
                _delay_ms(1000);
        }
}
int main(void)
{
        //set the input/output pins appropriately
        //to enable interrupt and port interfacing
        //for LED to blink
        DDRD=0x00;    //set appropriate data direction for D
        DDRB=0x01;    //make PB0 as output
        MCUCR=0x00;   //set MCUCR to level triggered
        GICR=0x40;    //enable interrupt 1
        PORTB=0x00;
        sei();        //global interrupt flag

        while (1)     //wait
        {

        }
}
```

Learning from this experiment –

1. Learnt about the loops , timers and interrupts in the AVR Atmega
2. Learnt to code in C
3. Running the C code in microchip