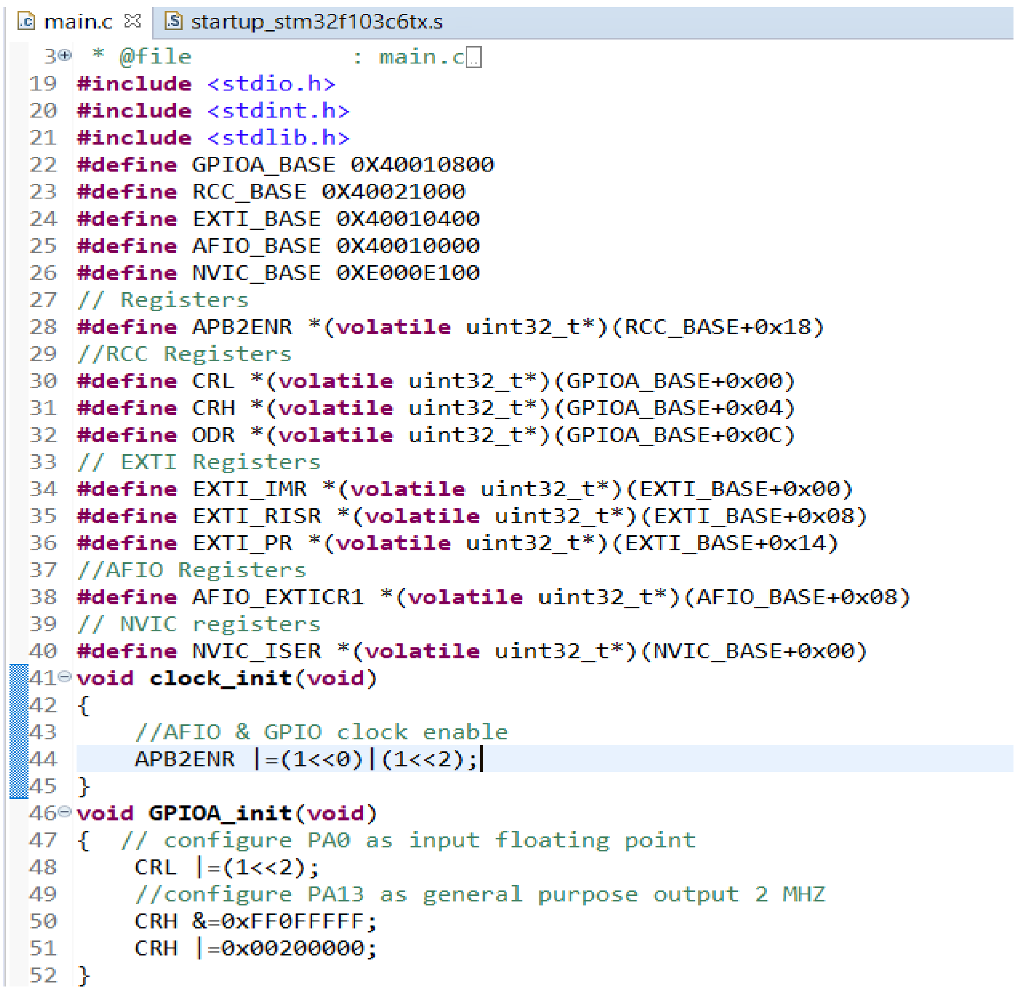
Report

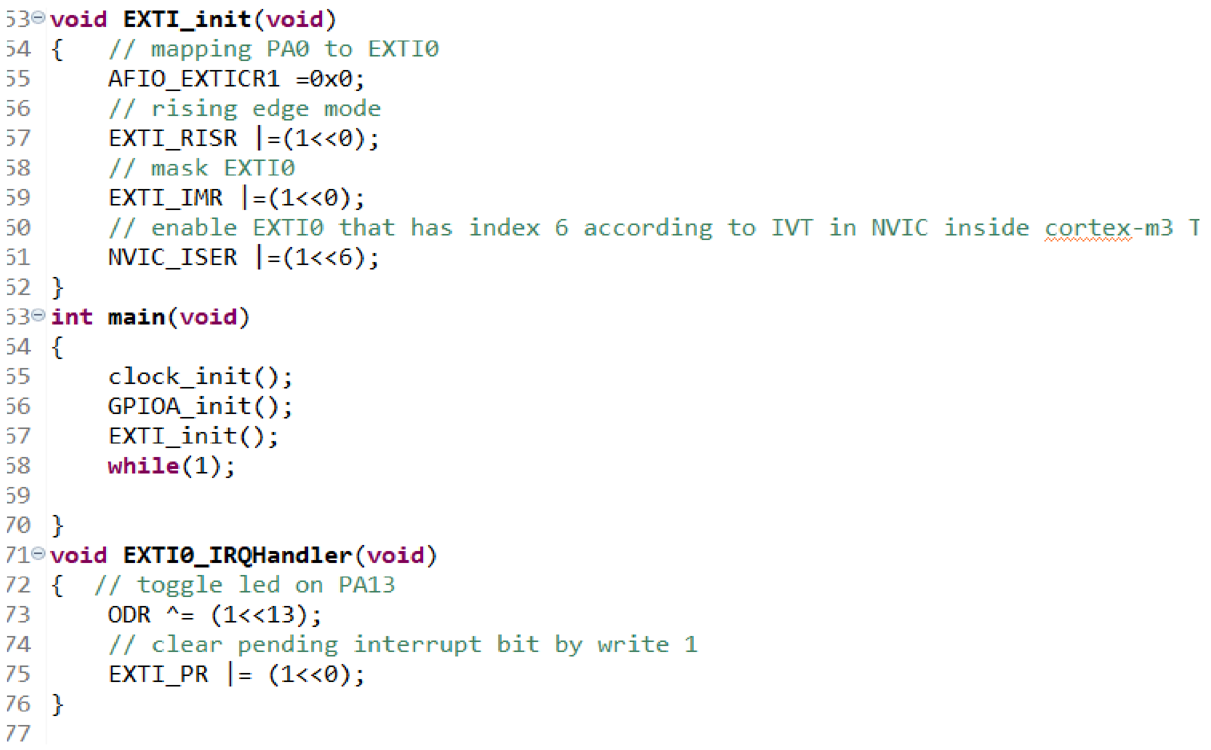
Lab1:

Use STM32F103C6 to enable external interrupt on PA0 rising edge mode to toggle led on PA13.

C code:

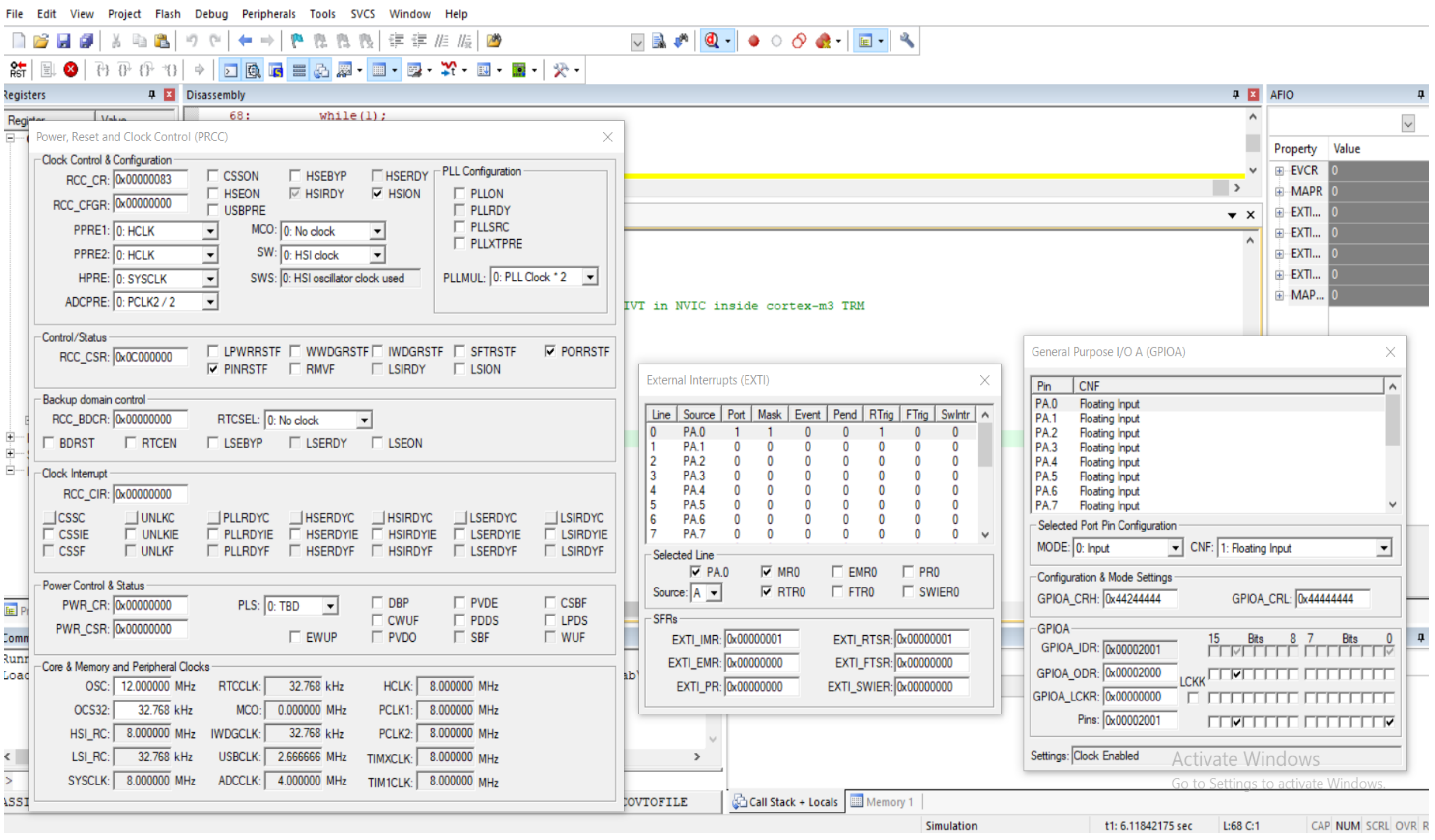


```c
 * @file             : main.c
#include <stdio.h>
#include <stdint.h>
#include <stdlib.h>
#define GPIOA_BASE 0X40010800
#define RCC_BASE 0X40021000
#define EXTI_BASE 0X40010400
#define AFIO_BASE 0X40010000
#define NVIC_BASE 0XE000E100
// Registers
#define APB2ENR *(volatile uint32_t*)(RCC_BASE+0x18)
//RCC Registers
#define CRL *(volatile uint32_t*)(GPIOA_BASE+0x00)
#define CRH *(volatile uint32_t*)(GPIOA_BASE+0x04)
#define ODR *(volatile uint32_t*)(GPIOA_BASE+0x0C)
// EXTI Registers
#define EXTI_IMR *(volatile uint32_t*)(EXTI_BASE+0x00)
#define EXTI_RISR *(volatile uint32_t*)(EXTI_BASE+0x08)
#define EXTI_PR *(volatile uint32_t*)(EXTI_BASE+0x14)
//AFIO Registers
#define AFIO_EXTICR1 *(volatile uint32_t*)(AFIO_BASE+0x08)
// NVIC registers
#define NVIC_ISER *(volatile uint32_t*)(NVIC_BASE+0x00)
void clock_init(void)
{
    //AFIO & GPIO clock enable
    APB2ENR |=(1<<0)|(1<<2);
}
void GPIOA_init(void)
{  // configure PA0 as input floating point
    CRL |=(1<<2);
    //configure PA13 as general purpose output 2 MHZ
    CRH &=0xFF0FFFFF;
    CRH |=0x00200000;
}
```



```c
void EXTI_init(void)
{   // mapping PA0 to EXTI0
    AFIO_EXTICR1 =0x0;
    // rising edge mode
    EXTI_RISR |=(1<<0);
    // mask EXTI0
    EXTI_IMR |=(1<<0);
    // enable EXTI0 that has index 6 according to IVT in NVIC inside cortex-m3 T
    NVIC_ISER |=(1<<6);
}
int main(void)
{
    clock_init();
    GPIOA_init();
    EXTI_init();
    while(1);

}
void EXTI0_IRQHandler(void)
{  // toggle led on PA13
    ODR ^= (1<<13);
    // clear pending interrupt bit by write 1
    EXTI_PR |= (1<<0);
}
```

output: using KEIL



```
68:          while(1);
IVT in NVIC inside cortex-m3 TRM
```

Power, Reset and Clock Control (PRCC)

Clock Control & Configuration
RCC_CR: 0x00000083   [ ] CSSON [ ] HSEBYP [ ] HSERDY
RCC_CFGR: 0x00000000   [ ] HSEON [✓] HSIRDY [✓] HSION
[ ] USBPRE
PPRE1: 0: HCLK   MCO: 0: No clock
PPRE2: 0: HCLK   SW: 0: HSI clock
HPRE: 0: SYSCLK   SWS: 0: HSI oscillator clock used
ADCPRE: 0: PCLK2 / 2

PLL Configuration
[ ] PLLON [ ] PLLRDY [ ] PLLSRC [ ] PLLXTPRE
PLLMUL: 0: PLL Clock * 2

Control/Status
RCC_CSR: 0x0C000000   [ ] LPWRRSTF [ ] WWDGRSTF [ ] IWDGRSTF [ ] SFTRSTF [✓] PORRSTF
[✓] PINRSTF [ ] RMVF [ ] LSIRDY [ ] LSION

Backup domain control
RCC_BDCR: 0x00000000   RTCSEL: 0: No clock
[ ] BDRST [ ] RTCEN [ ] LSEBYP [ ] LSERDY [ ] LSEON

Clock Interrupt
RCC_CIR: 0x00000000
[ ] CSSC [ ] UNLKC [ ] PLLRDYC [ ] HSERDYC [ ] HSIRDYC [ ] LSERDYC [ ] LSIRDYC
[ ] CSSIE [ ] UNLKIE [ ] PLLRDYIE [ ] HSERDYIE [ ] HSIRDYIE [ ] LSERDYIE [ ] LSIRDYIE
[ ] CSSF [ ] UNLKF [ ] PLLRDYF [ ] HSERDYF [ ] HSIRDYF [ ] LSERDYF [ ] LSIRDYF

Power Control & Status
PWR_CR: 0x00000000   PLS: 0: TBD   [ ] DBP [ ] PVDE [ ] CSBF
PWR_CSR: 0x00000000   [ ] CWUF [ ] PDDS [ ] LPDS
[ ] EWUP [ ] PVDO [ ] SBF [ ] WUF

Core & Memory and Peripheral Clocks

OSC:	12.000000 MHz	RTCCLK:	32.768 kHz	HCLK:	8.000000 MHz
OCS32:	32.768 kHz	MCO:	0.000000 MHz	PCLK1:	8.000000 MHz
HSI_RC:	8.000000 MHz	IWDGCLK:	32.768 kHz	PCLK2:	8.000000 MHz
LSI_RC:	32.768 kHz	USBCLK:	2.666666 MHz	TIMXCLK:	8.000000 MHz
SYSCLK:	8.000000 MHz	ADCCLK:	4.000000 MHz	TIM1CLK:	8.000000 MHz

External Interrupts (EXTI)

Line	Source	Port	Mask	Event	Pend	RTrig	FTrig	SwIntr
0	PA.0	1	1	0	0	1	0	0
1	PA.1	0	0	0	0	0	0	0
2	PA.2	0	0	0	0	0	0	0
3	PA.3	0	0	0	0	0	0	0
4	PA.4	0	0	0	0	0	0	0
5	PA.5	0	0	0	0	0	0	0
6	PA.6	0	0	0	0	0	0	0
7	PA.7	0	0	0	0	0	0	0

Selected Line
[✓] PA.0 [✓] MR0 [ ] EMR0 [ ] PR0
Source: A [✓] RTR0 [ ] FTR0 [ ] SWIER0

SFRs
EXTI_IMR: 0x00000001   EXTI_RTSR: 0x00000001
EXTI_EMR: 0x00000000   EXTI_FTSR: 0x00000000
EXTI_PR: 0x00000000   EXTI_SWIER: 0x00000000

General Purpose I/O A (GPIOA)

Pin	CNF
PA.0	Floating Input
PA.1	Floating Input
PA.2	Floating Input
PA.3	Floating Input
PA.4	Floating Input
PA.5	Floating Input
PA.6	Floating Input
PA.7	Floating Input

Selected Port Pin Configuration
MODE: 0: Input   CNF: 1: Floating Input

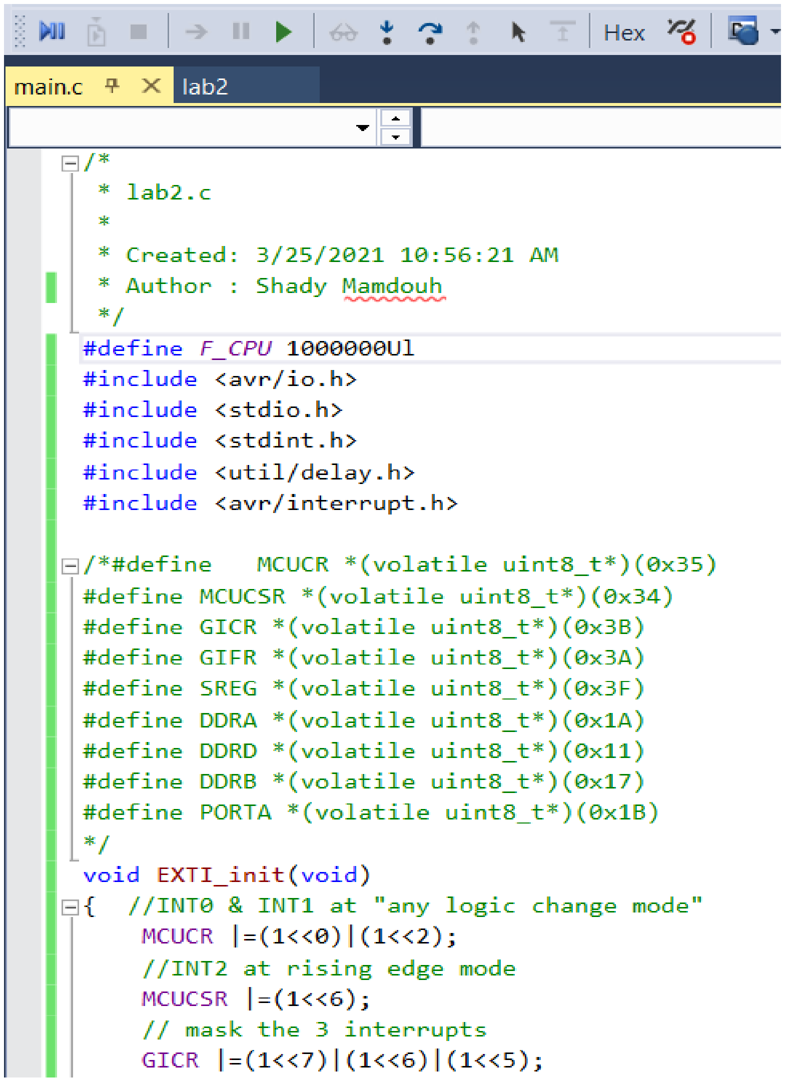
Configuration & Mode Settings
GPIOA_CRH: 0x44244444   GPIOA_CRL: 0x44444444

GPIOA
GPIOA_IDR: 0x00002001
GPIOA_ODR: 0x00002000
GPIOA_LCKR: 0x00000000   LCKK [ ]
Pins: 0x00002001

Settings: Clock Enabled

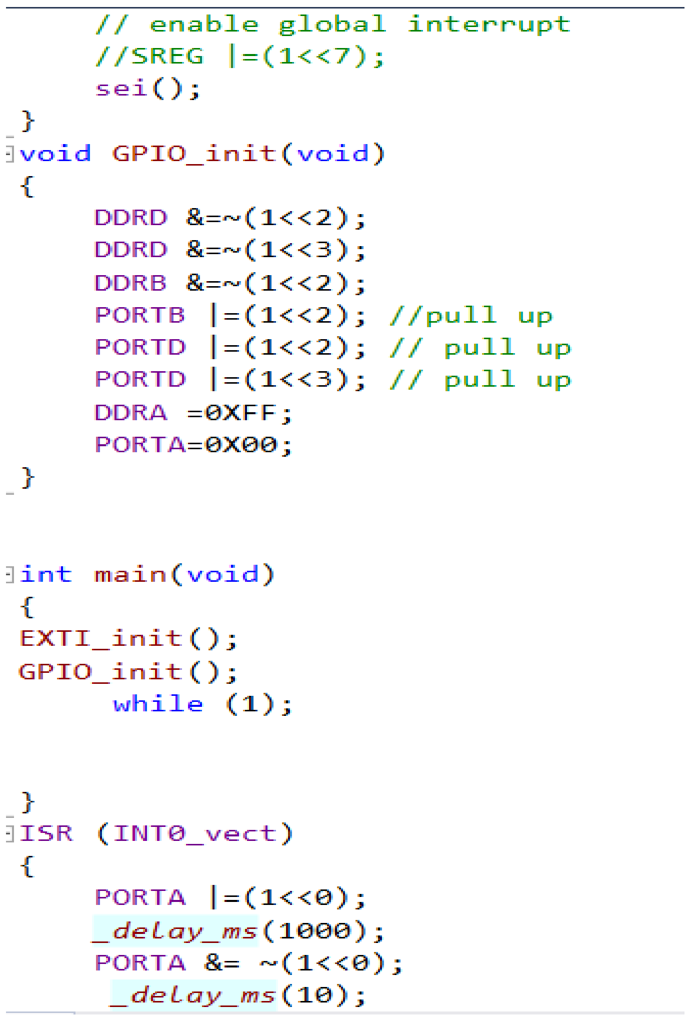
Simulation   t1: 6.11842175 sec   L:68 C:1   CAP NUM SCRL OVR R lab2: use Atmega32 to enable 3 external interrupts to control 3 LEDS to be on for 1 second using (any logic change mode) for INT0 And INT1, and using (rising edge mode) for INT2

C code:

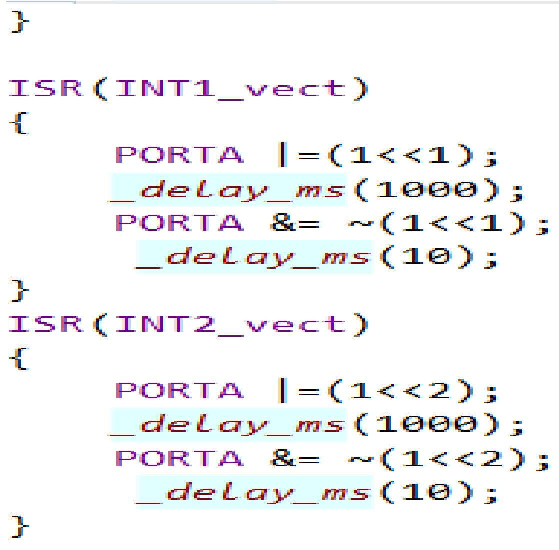


main.c   lab2

```c
/*
 * lab2.c
 *
 * Created: 3/25/2021 10:56:21 AM
 * Author : Shady Mamdouh
 */
#define F_CPU 1000000Ul
#include <avr/io.h>
#include <stdio.h>
#include <stdint.h>
#include <util/delay.h>
#include <avr/interrupt.h>

/*#define    MCUCR *(volatile uint8_t*)(0x35)
#define MCUCSR *(volatile uint8_t*)(0x34)
#define GICR *(volatile uint8_t*)(0x3B)
#define GIFR *(volatile uint8_t*)(0x3A)
#define SREG *(volatile uint8_t*)(0x3F)
#define DDRA *(volatile uint8_t*)(0x1A)
#define DDRD *(volatile uint8_t*)(0x11)
#define DDRB *(volatile uint8_t*)(0x17)
#define PORTA *(volatile uint8_t*)(0x1B)
*/
void EXTI_init(void)
{   //INT0 & INT1 at "any logic change mode"
    MCUCR |=(1<<0)|(1<<2);
    //INT2 at rising edge mode
    MCUCSR |=(1<<6);
    // mask the 3 interrupts
    GICR |=(1<<7)|(1<<6)|(1<<5);
```



```
     // enable global interrupt
     //SREG |=(1<<7);
     sei();
}
void GPIO_init(void)
{
     DDRD &=~(1<<2);
     DDRD &=~(1<<3);
     DDRB &=~(1<<2);
     PORTB |=(1<<2); //pull up
     PORTD |=(1<<2); // pull up
     PORTD |=(1<<3); // pull up
     DDRA =0XFF;
     PORTA=0X00;
}


int main(void)
{
EXTI_init();
GPIO_init();
      while (1);


}
ISR (INT0_vect)
{
     PORTA |=(1<<0);
     _delay_ms(1000);
     PORTA &= ~(1<<0);
      _delay_ms(10);
```

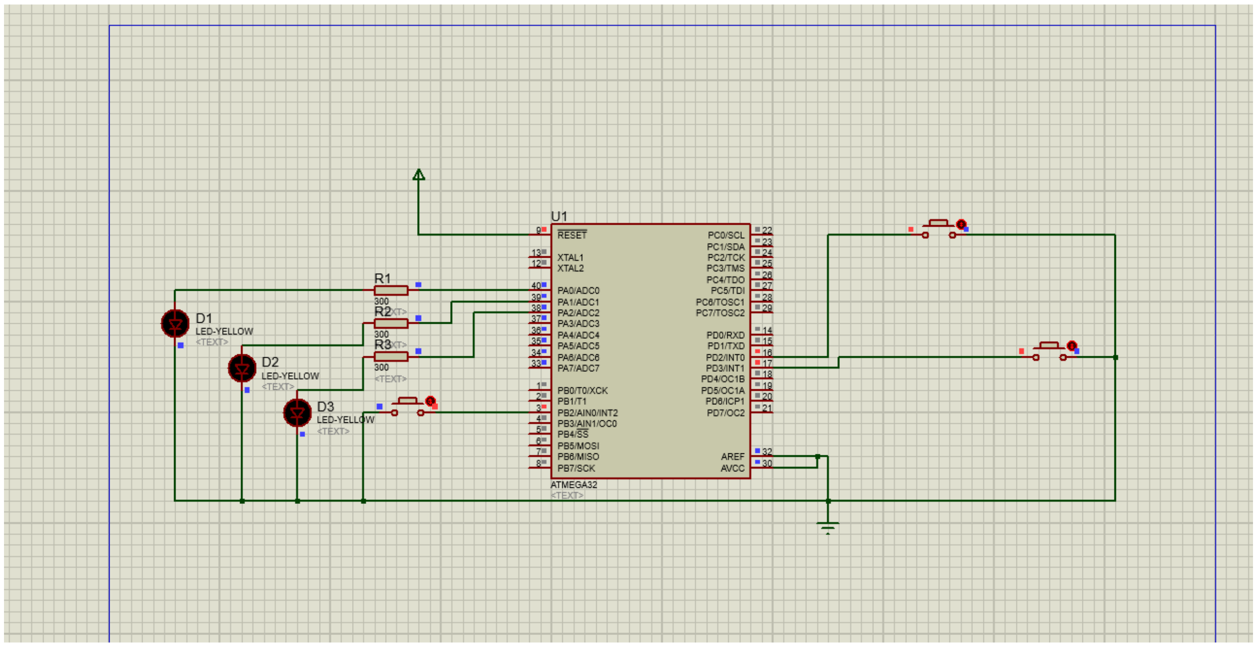


```
}

ISR(INT1_vect)
{
     PORTA |=(1<<1);
     _delay_ms(1000);
     PORTA &= ~(1<<1);
      _delay_ms(10);
}
ISR(INT2_vect)
{
     PORTA |=(1<<2);
     _delay_ms(1000);
     PORTA &= ~(1<<2);
      _delay_ms(10);
}
```

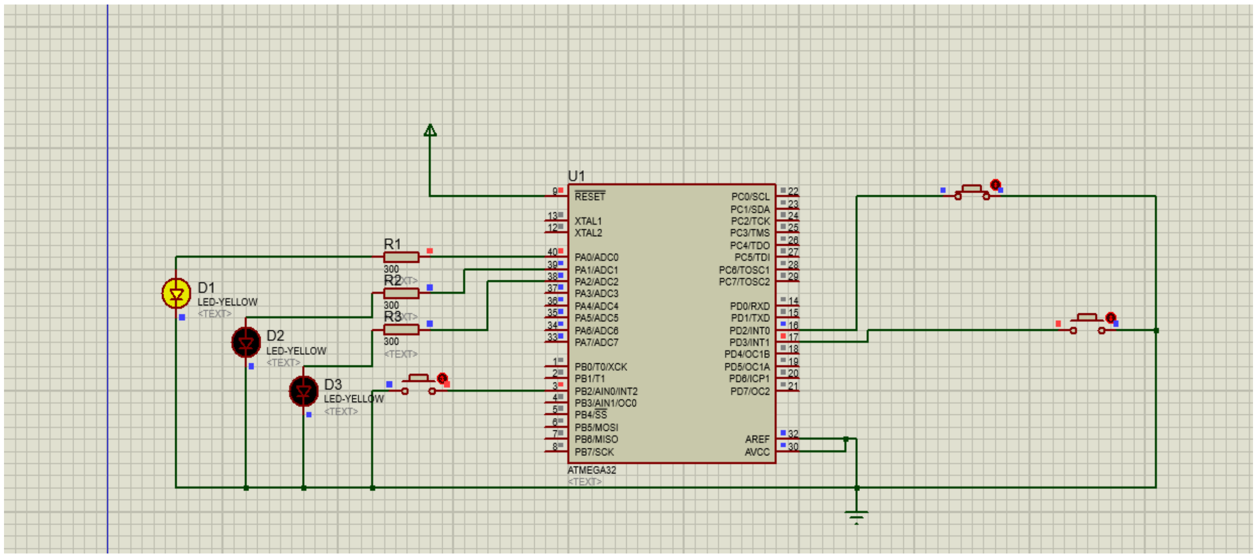
Output: on proteus

We are using internal pull up so the switches will be connected to ground

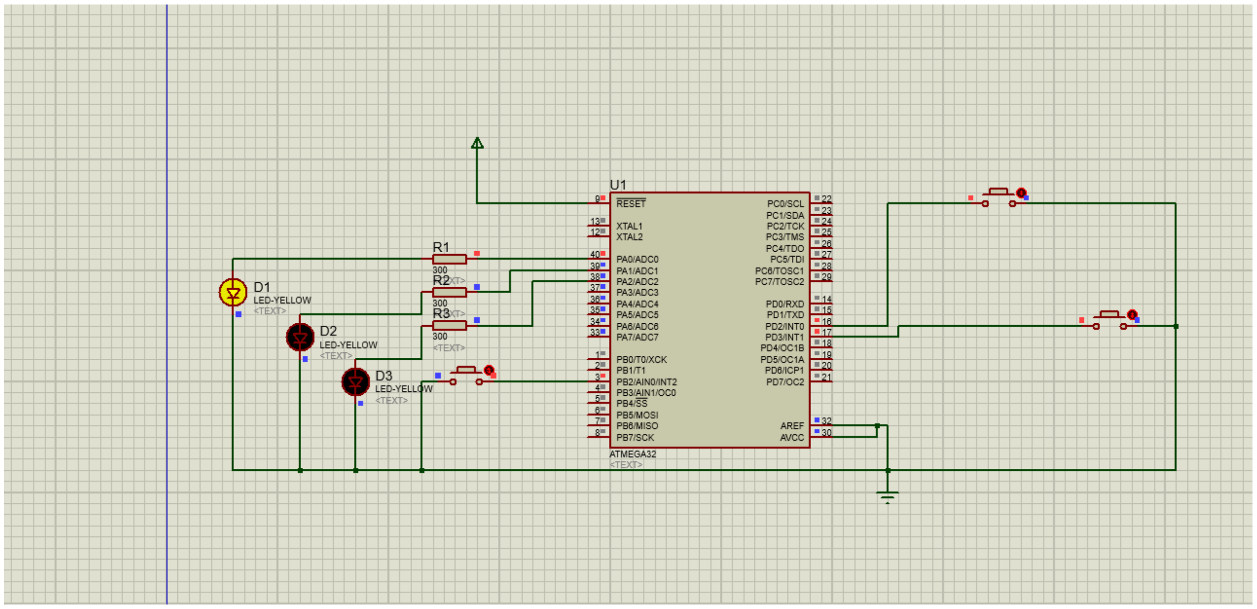
1- Before interrupt



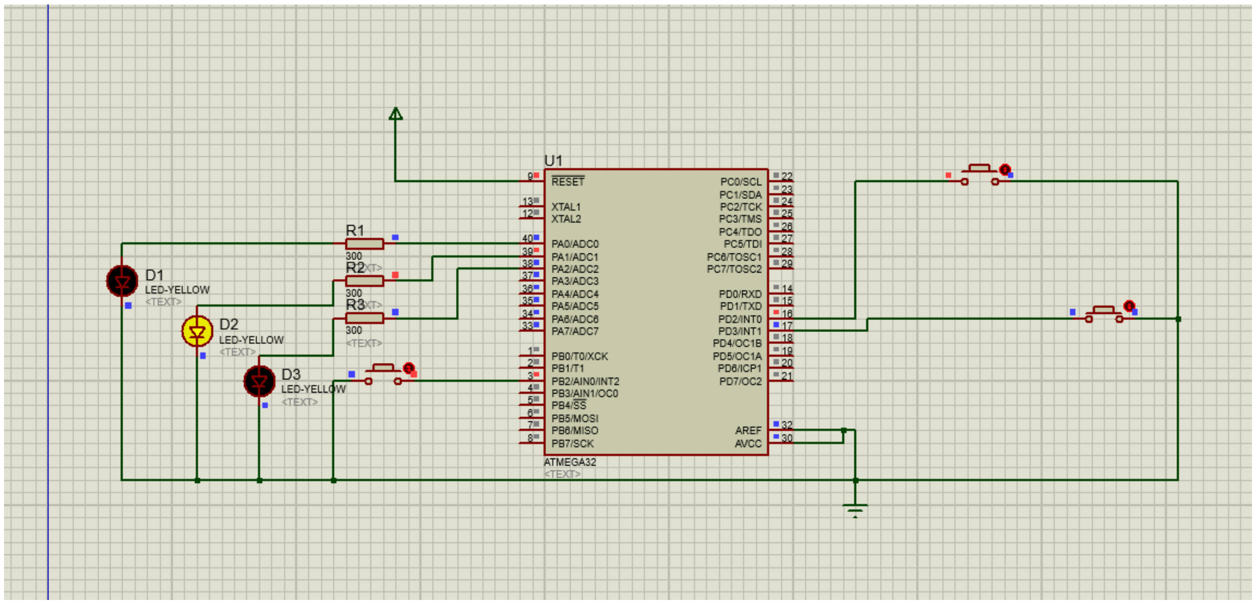
2- INT0 change logic to low



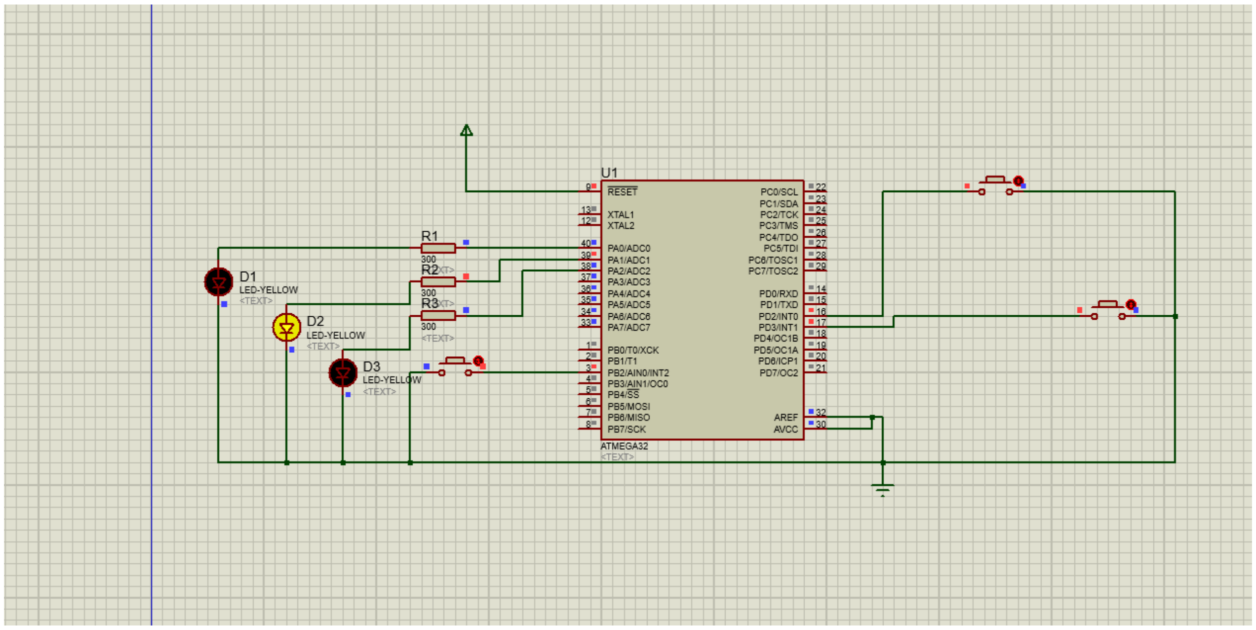
3- INT0 change logic to high (release switch)



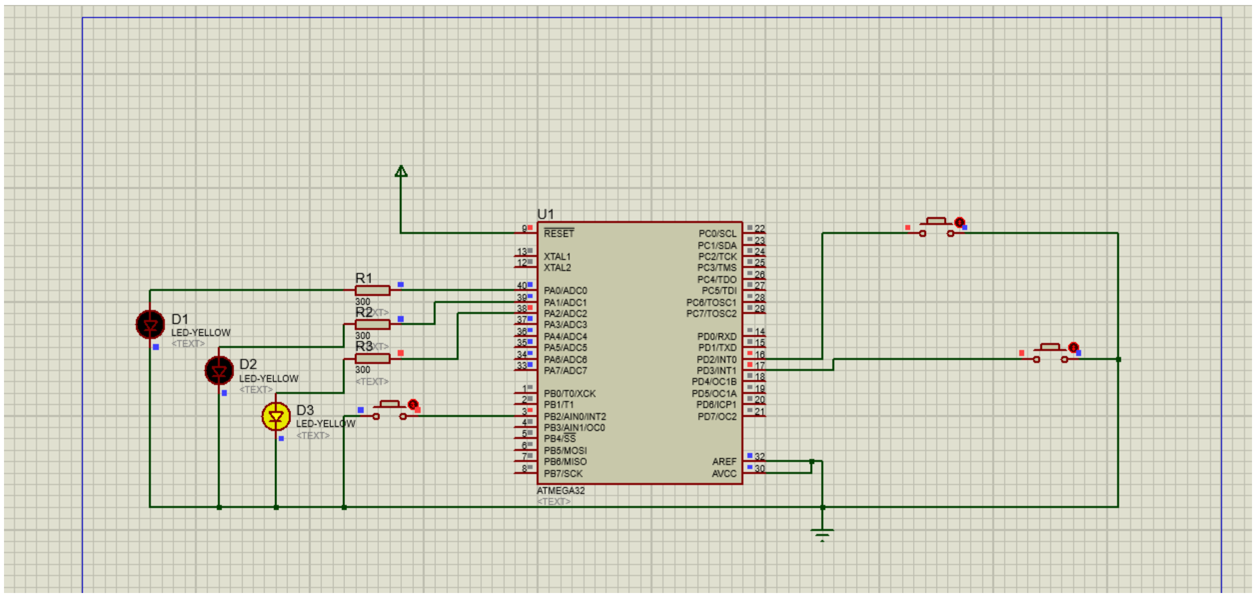
4- INT1 change logic to low



5- INT1 change logic to high (release switch)



6- INT2 on rising edge mode


U1
ATMEGA32
R1
R2
R3
300
D1
D2
D3
LED-YELLOW
RESET
XTAL1
XTAL2
PA0/ADC0
PA1/ADC1
PA2/ADC2
PA3/ADC3
PA4/ADC4
PA5/ADC5
PA6/ADC6
PA7/ADC7
PB0/T0/XCK
PB1/T1
PB2/AIN0/INT2
PB3/AIN1/OC0
PB4/SS
PB5/MOSI
PB6/MISO
PB7/SCK
PC0/SCL
PC1/SDA
PC2/TCK
PC3/TMS
PC4/TDO
PC5/TDI
PC6/TOSC1
PC7/TOSC2
PD0/RXD
PD1/TXD
PD2/INT0
PD3/INT1
PD4/OC1B
PD5/OC1A
PD6/ICP1
PD7/OC2
AREF
AVCC