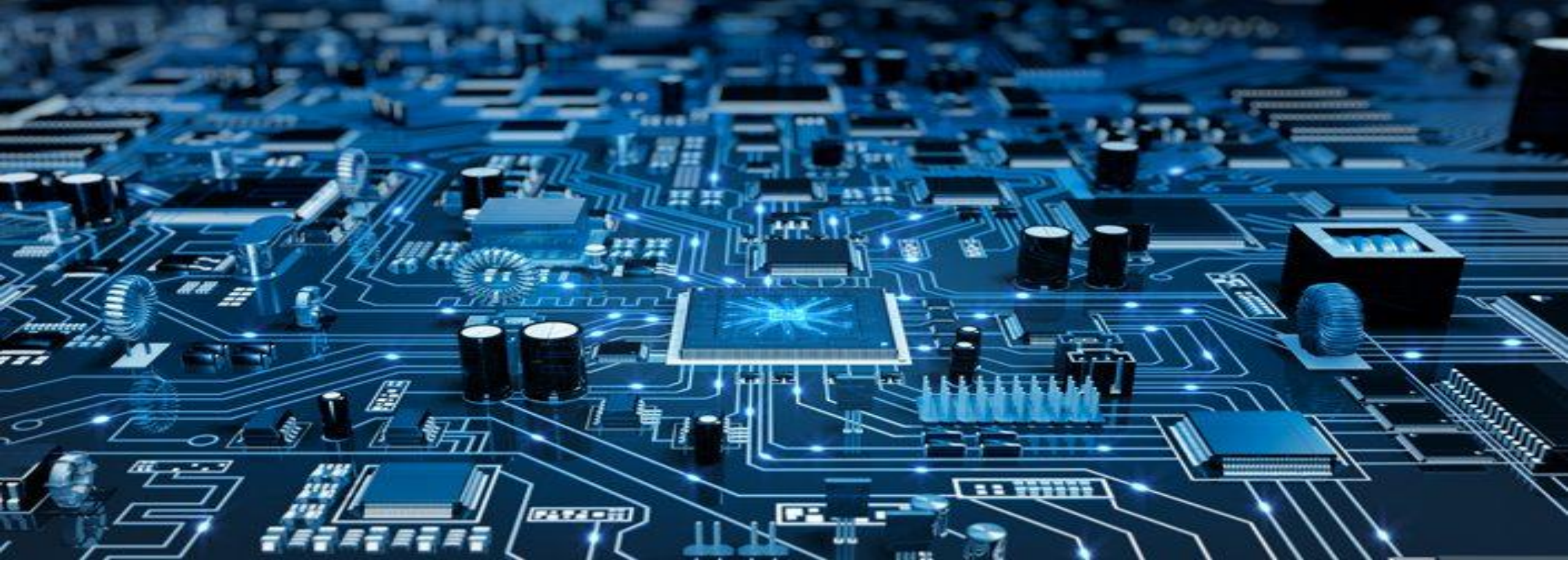


Programming ATMEGA328P in Atmel Studio

Eng. R. Saahith Ahamed.
Department of Electrical and Information Engineering
Faculty of Engineering,
University of Ruhuna.

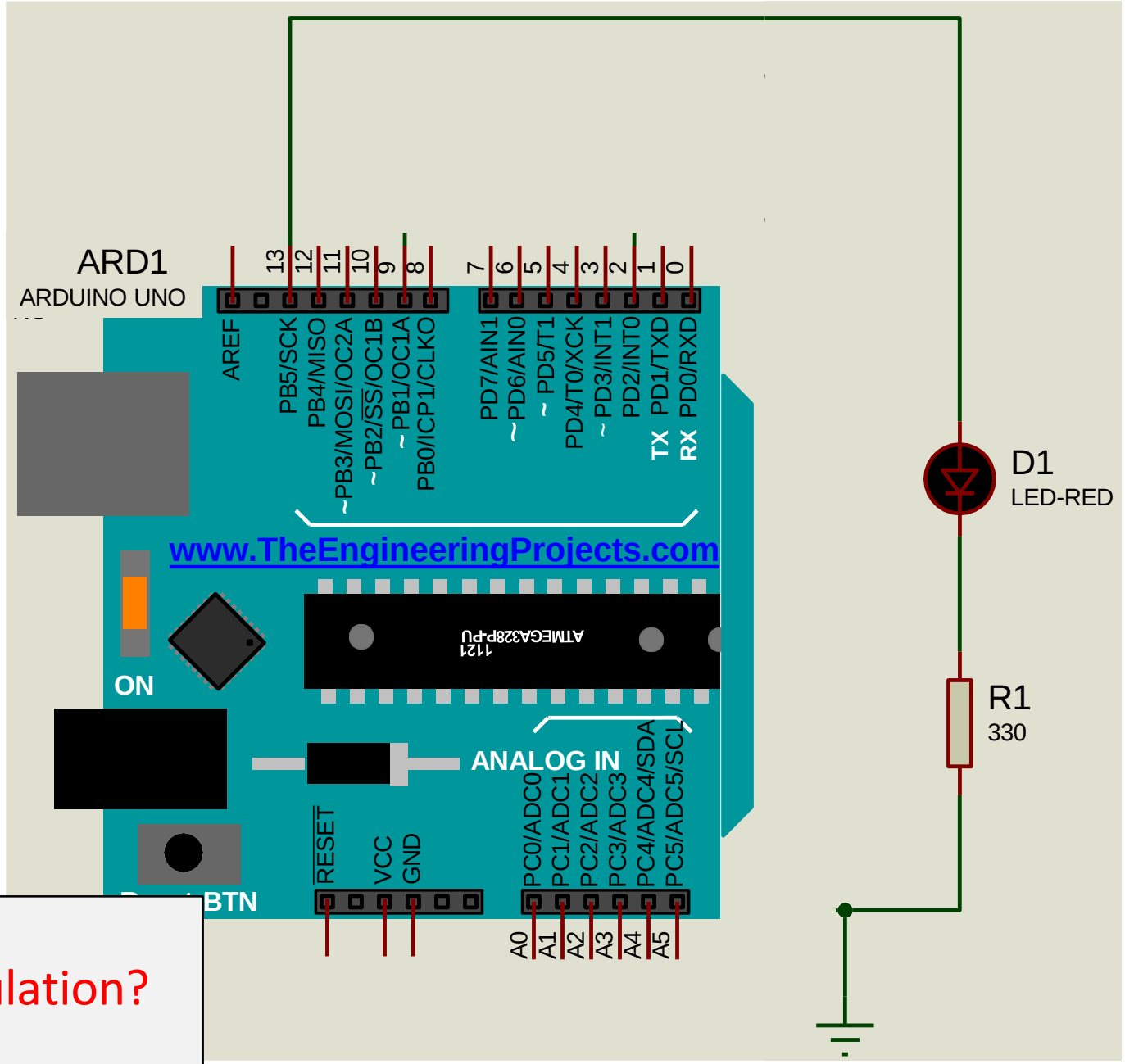
Proteus Simulation

Implement the Circuit illustrated in figure using Proteus.

Point to the hex file

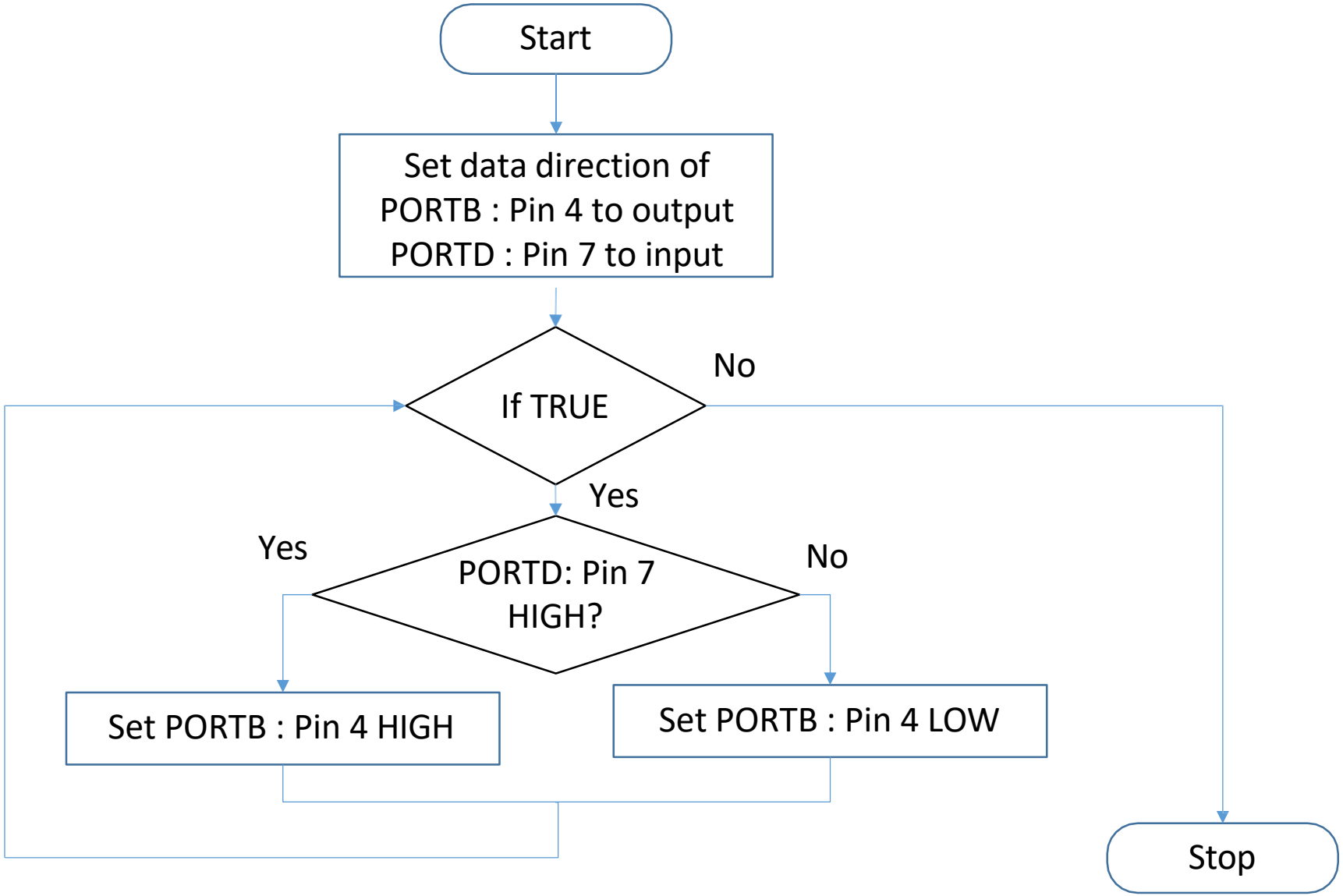
and run the simulation.

Show your output to the instructor and get your lab sheet signed.



What is the expected outcome of this simulation?

Making the LED Respond to a Button



Setting Pin as Input

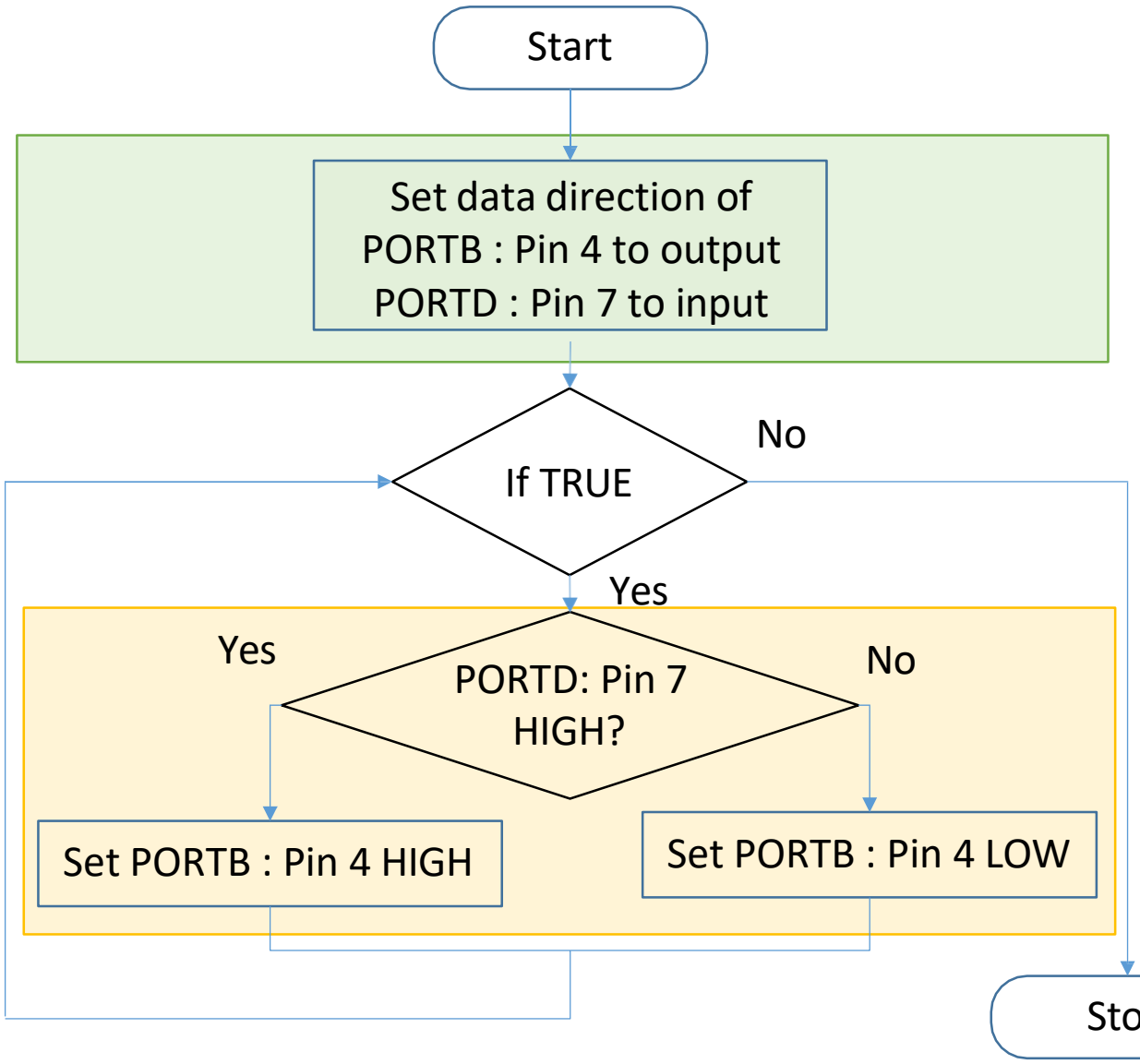
In order to set a pin as input : set to 0
set a pin as output : set to 1

Output register of port x : PORTx Input
register of a port x : PINx

Ex: Setting PORTD pin7 as input : DDRB &= 0B01111111;

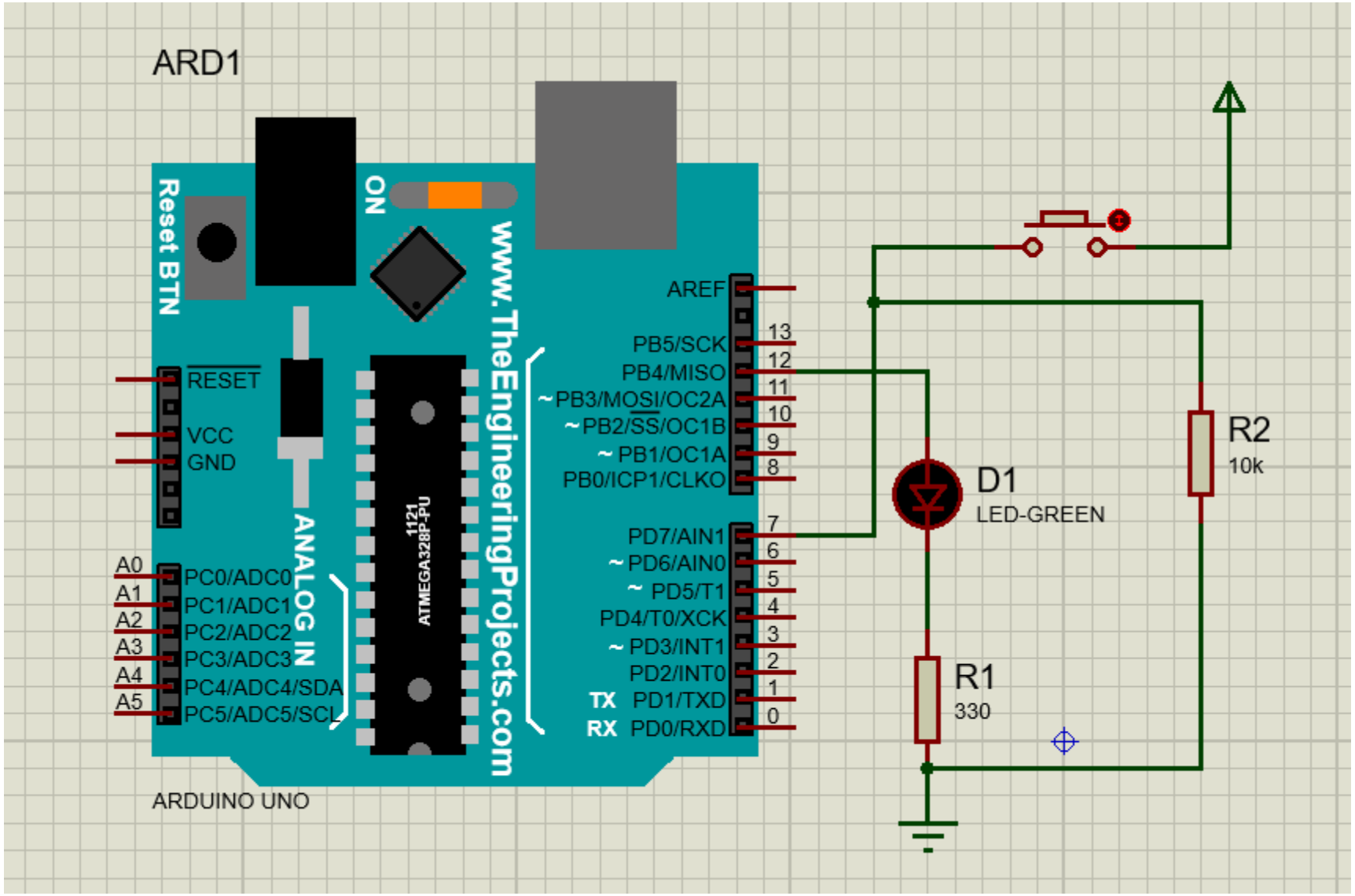
Reading from PORTD : use PIND

Coding to Make LEDs Respond to Push Button




```c
#include <avr/io.h>
#define F_CPU 16000000UL


int main(void)
{
    /* Replace with your application code */

    DDRD &= 0B01111111; // Port D's pin7 is set to input
    DDRB |= 0B00010000; //POrt B's pin4 is set to output

    while (1)
    {
        if (PIND & 0B10000000)
        {
            PORTB |= 0B00010000; //make the port B's pin4 to high
    }else
    {
        PORTB &= 0B11101111;//make the port B's pin4 to low
    }

    }
}
```

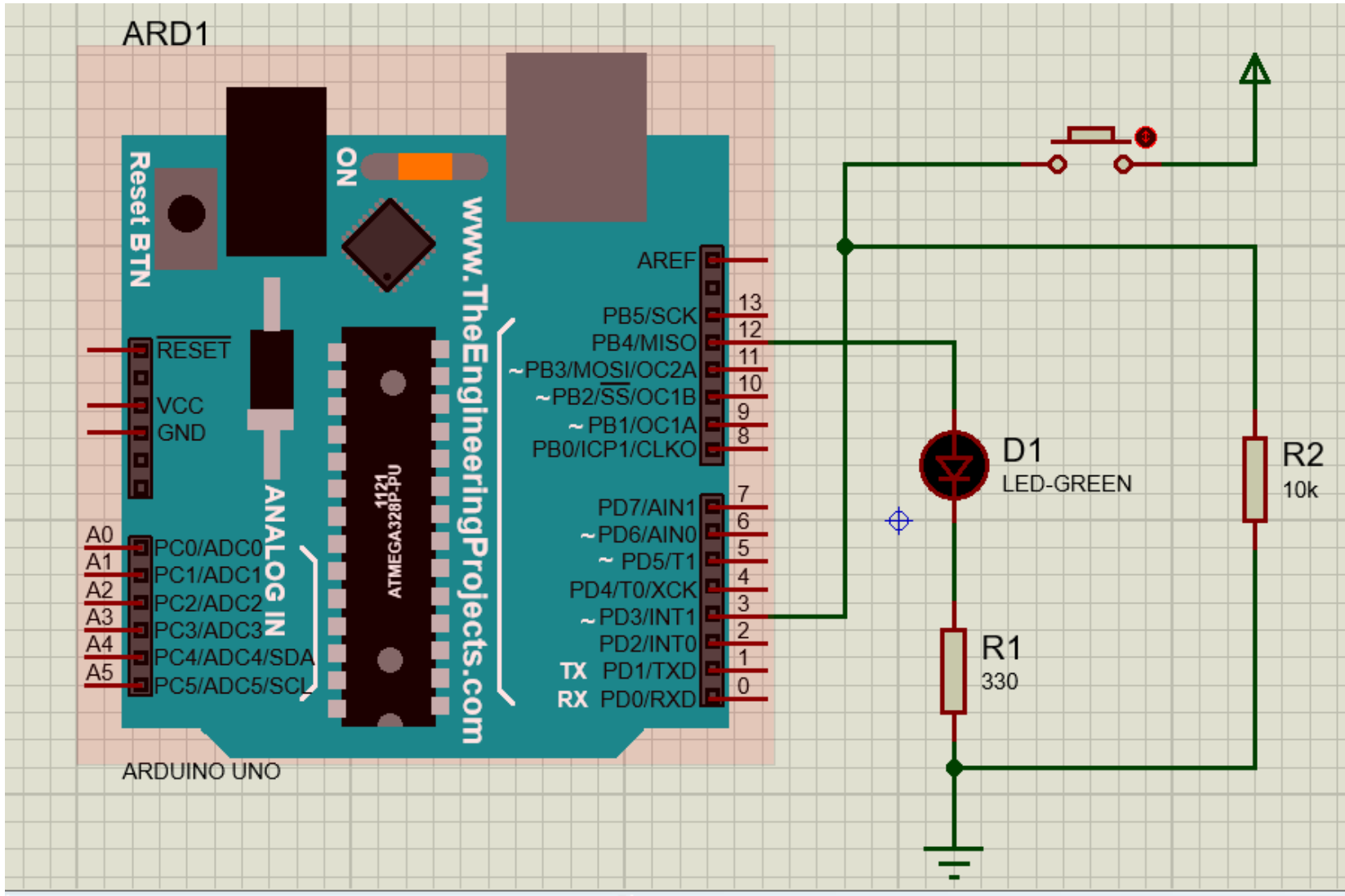
Proteus Simulation to Check Whether LEDs Respond to Push Button



Task

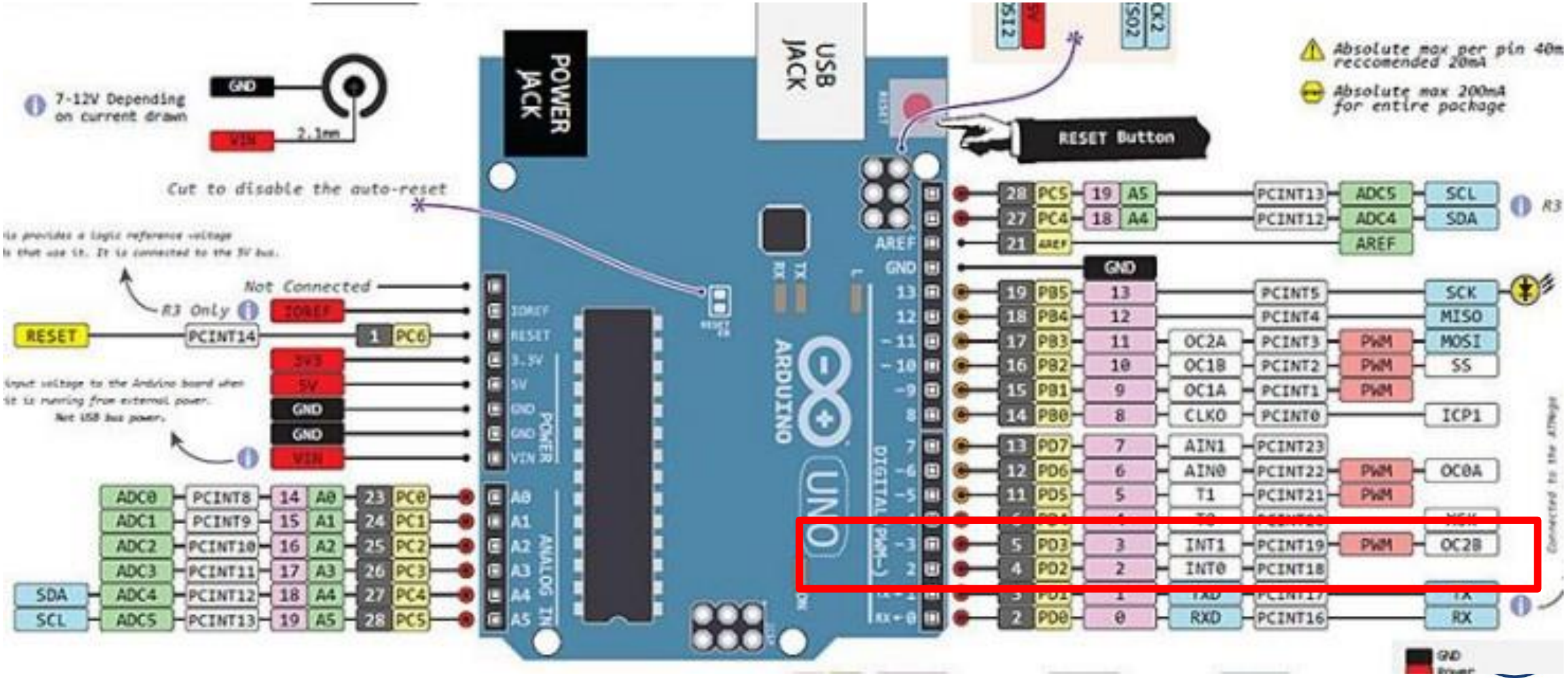
LED should toggle its state when the push button is pressed.

Use an External Interrupt.

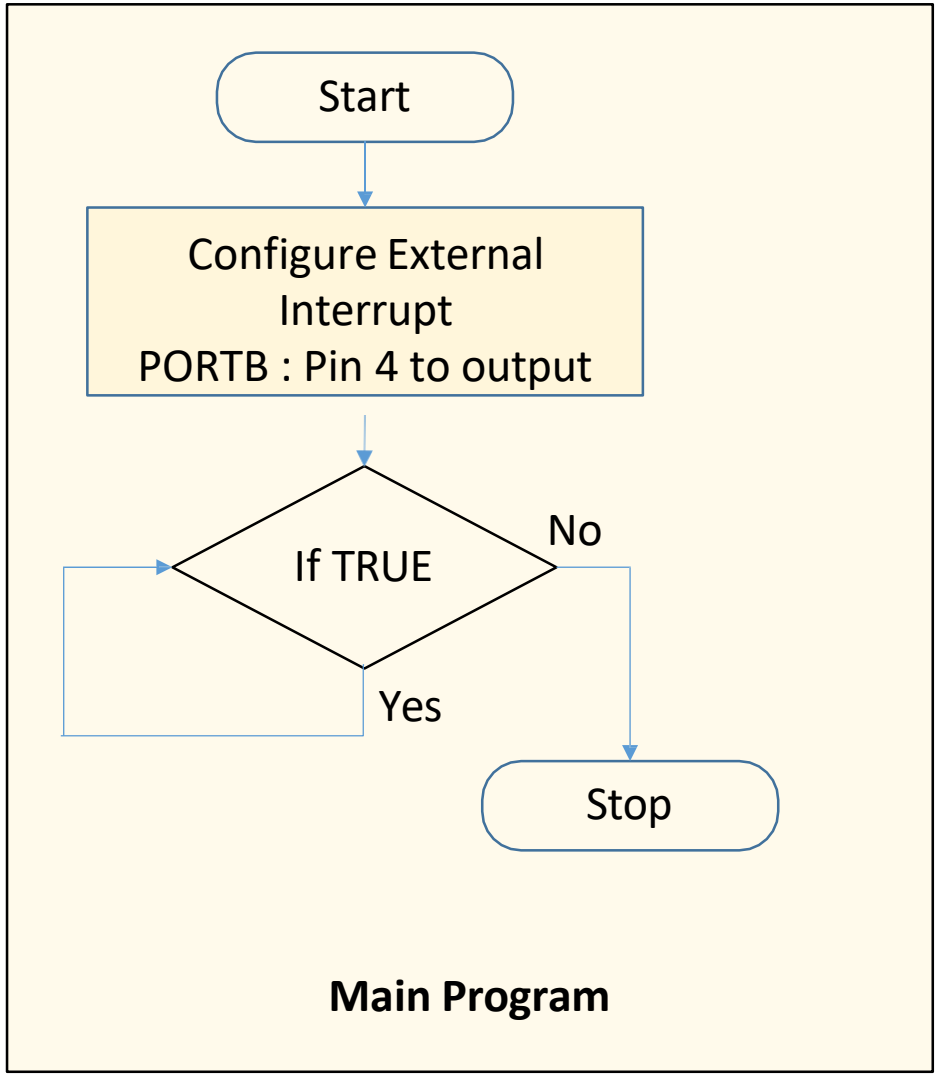


External Interrupt

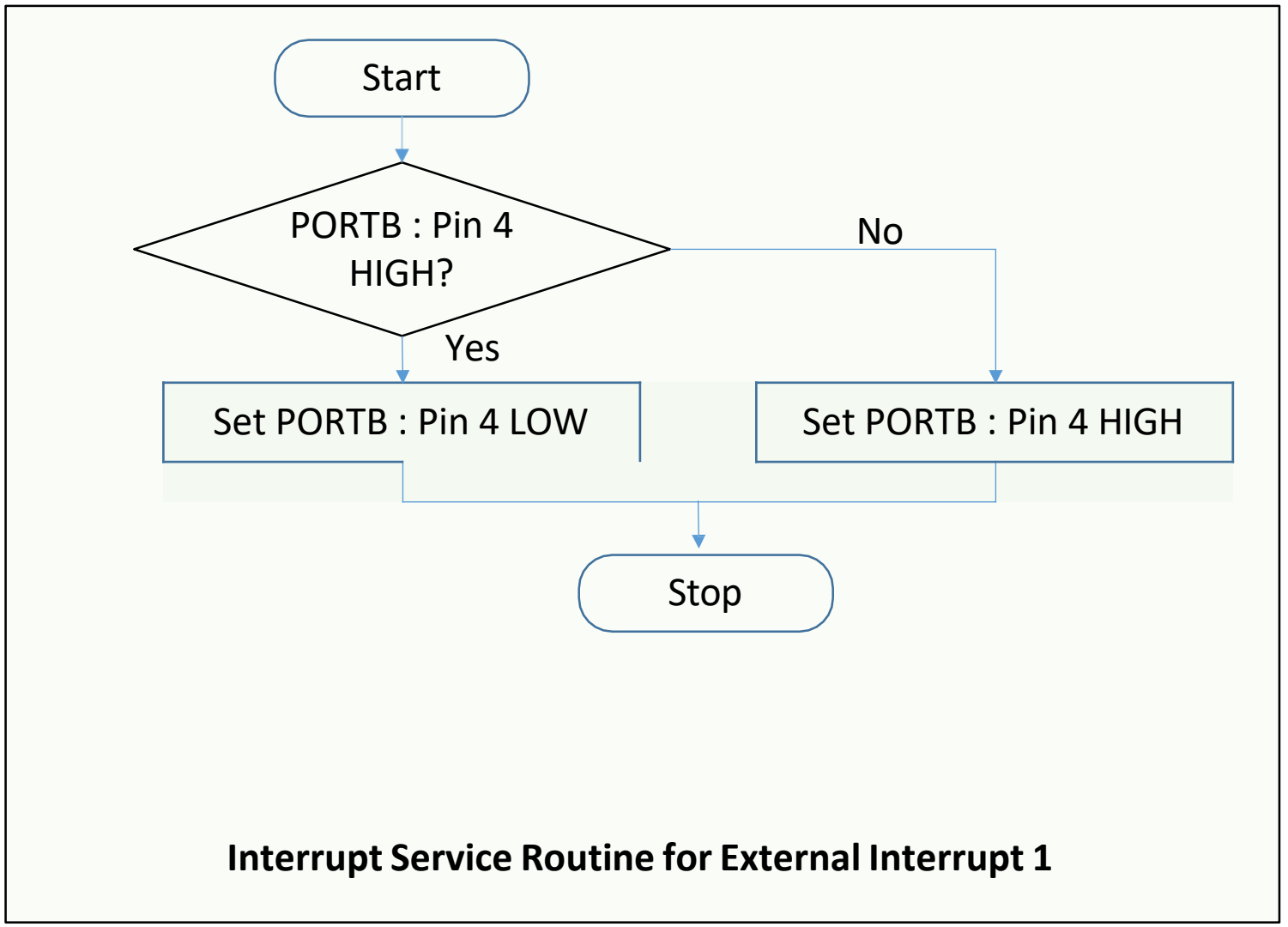
Two external interrupts INT0 and INT1



Making the LED Respond to a Pushbutton Using an Interrupt



Main Program



Interrupt Service Routine for External Interrupt 1

External Interrupt

Two external interrupts INT0 and INT1

Interrupt can be triggered by either rising edge or falling edge or both or the LOW level of the corresponding interrupt pin.

This is set using the EICRA register

Interrupt is enabled by setting the enable bit of EIMSK register

Name: EIMSK
Offset: 0x3D
Reset: 0x00
Property: When addressing as I/O Register: address offset is 0x1D

Bit	7	6	5	4	3	2	1	0
							INT1	INT0
Access							R/W	R/W
Reset							0	0

External Interrupt

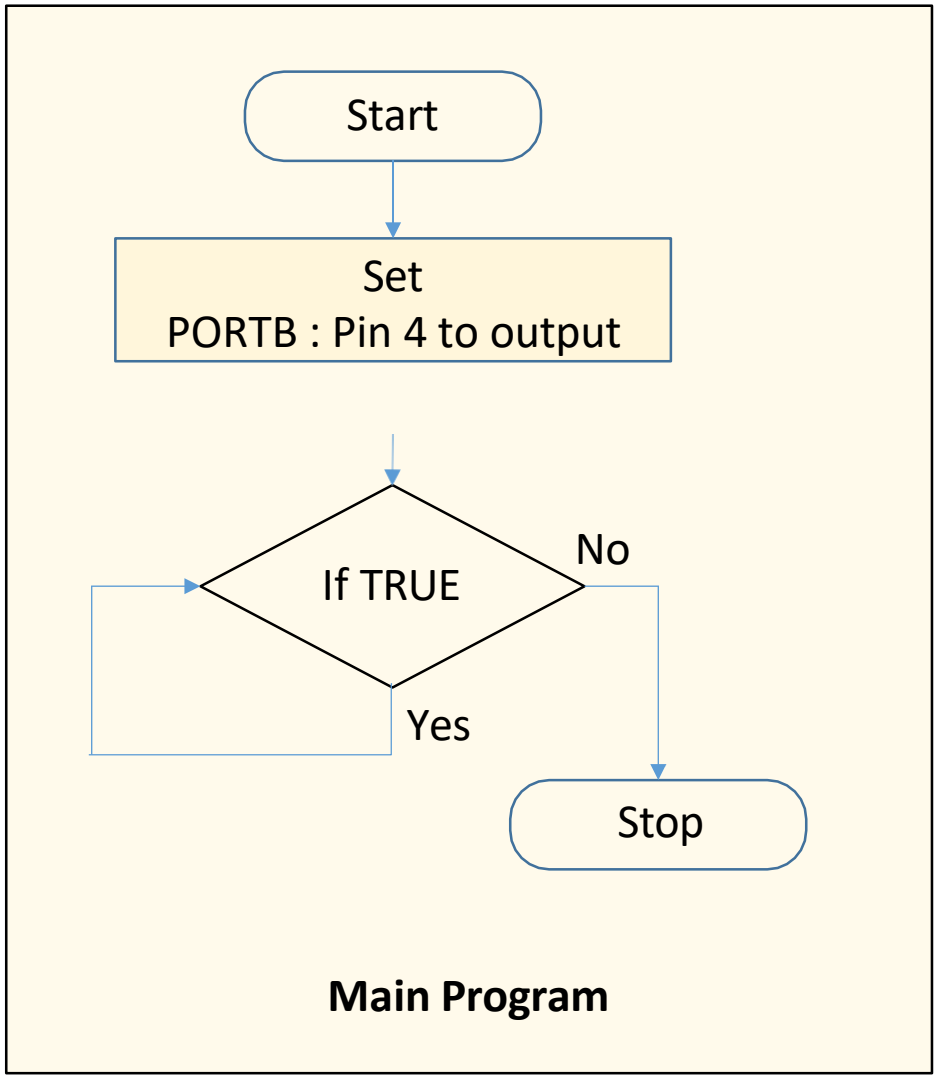
ISC11, ISC10 are two bits to configure the External Interrupt 1

Name: EICRA
Offset: 0x69
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
					ISC11	ISC10	ISC01	ISC00
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Value	Description
00	The low level of INT1 generates an interrupt request.
01	Any logical change on INT1 generates an interrupt request.
10	The falling edge of INT1 generates an interrupt request.
11	The rising edge of INT1 generates an interrupt request.

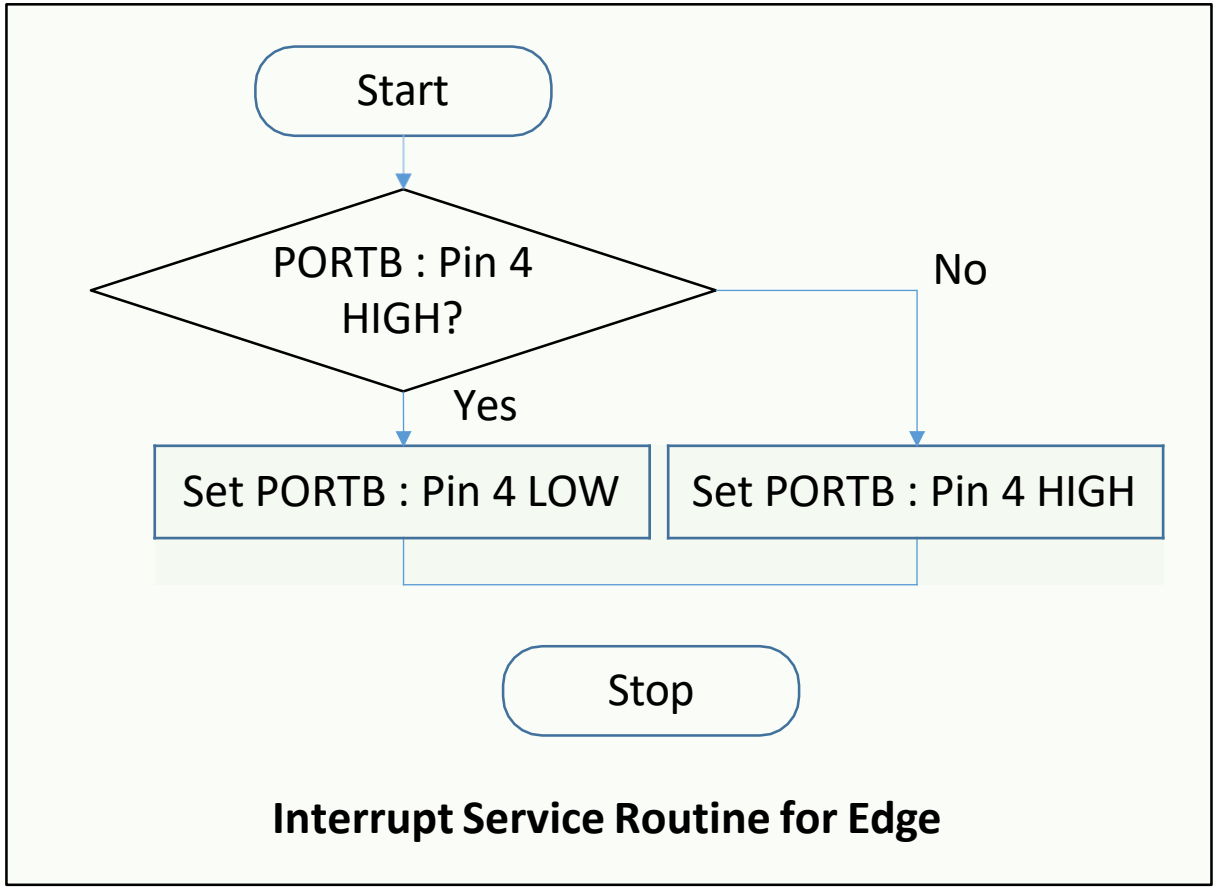
Implementation



Main Program

```
#define F_CPU 16000000UL
#include <avr/io.h>
#include <avr/interrupt.h>

volatile int toggle = 0;

int main(void)
{

    DDRB |= 0B00010000; // (1<<PB4) : Pin4 of PotB to output

    EIMSK |= 0B00000010; //(1<<INT1) Enabling INT1
    EICRA |= 0B00001100; //(1<<ISC11) | (1<<ISC10) Interrupt is set to Rising edge
    sei(); // Enabling Global Interrupt
    while (1)
    {
    }
}
```

Implementation



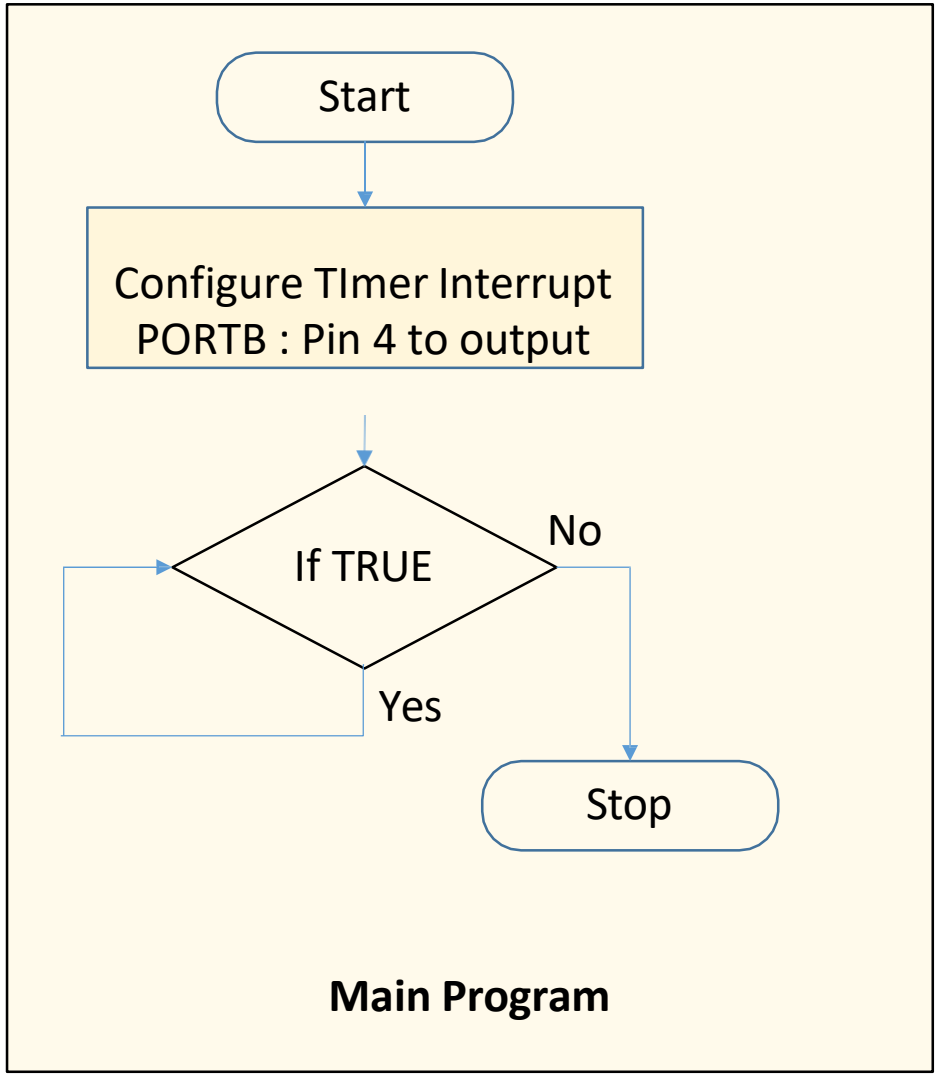
Interrupt Service Routine for Edge

```
ISR (INT1_vect)
{
    if (toggle)
    {
        PORTB |= 0B00010000; // (1<<PB4) Making Pin4 to High
    }else{
        PORTB &= 0B11101111; // ~(1<<PB4) Making Pin4 to Low
    }

    toggle = 1 - toggle;
}
```

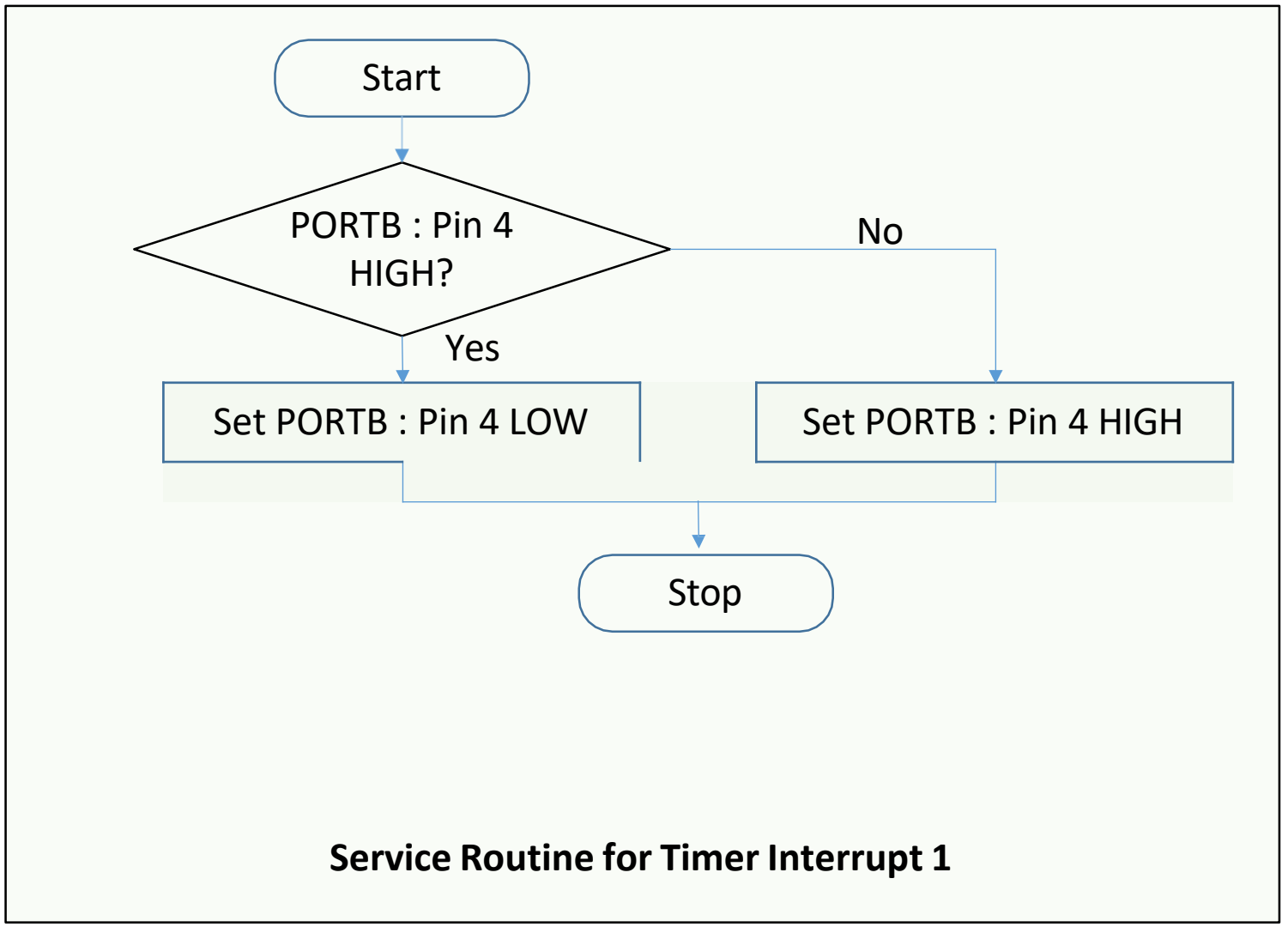
Implementation

Press the pushbutton over and over and observe the behavior of LED

Making the LED Respond to a Timer Interrupt



Main Program



Service Routine for Timer Interrupt 1

Timers in Atmel 328P

Two timers : one 8-bit and one 16-bit

Two interrupt modes : Overflow mode and Compare mode

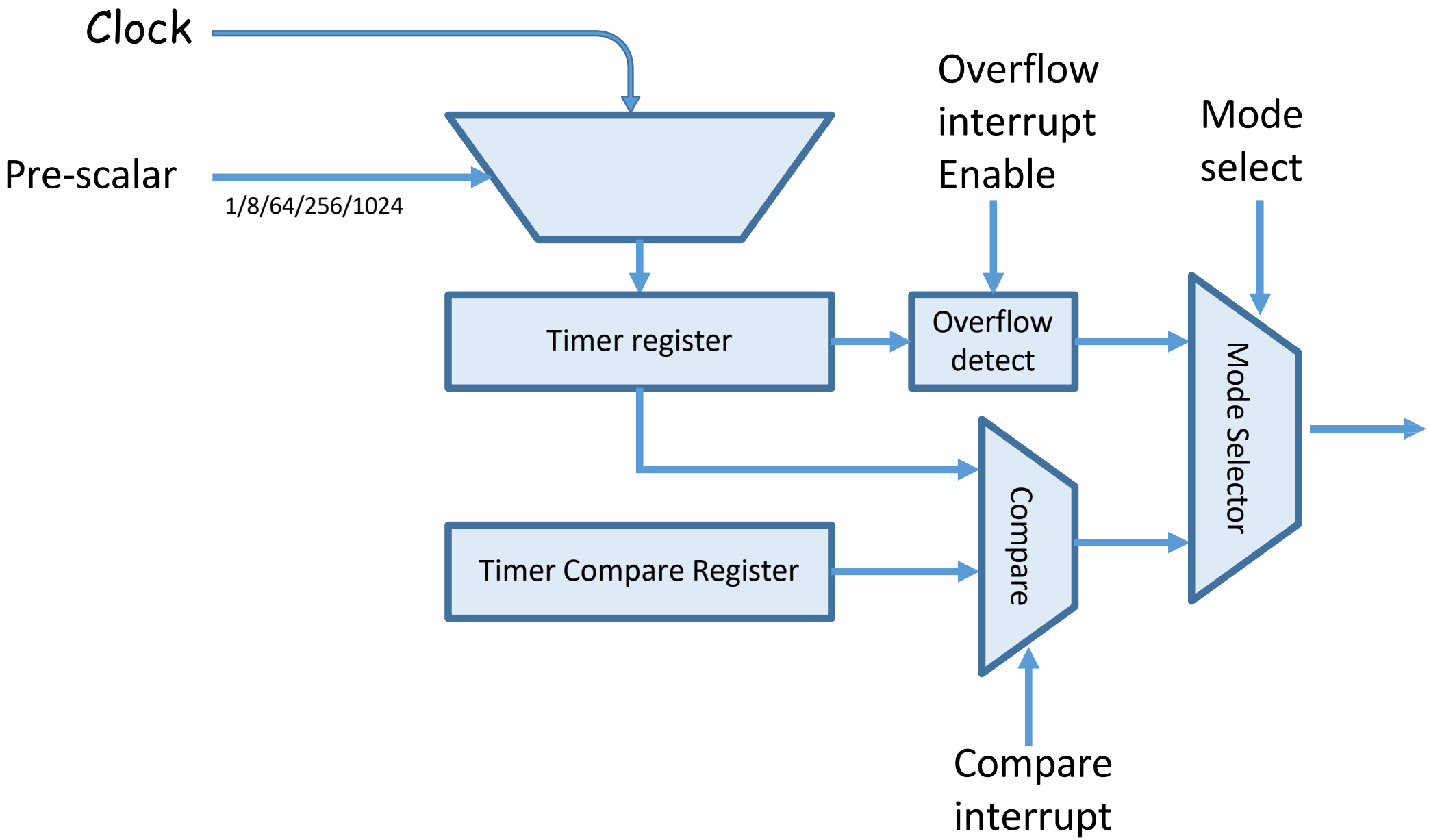
Overflow mode : Interrupt occurs when timer register overflows

Compare mode : Interrupt occurs when timer register matches the compare register

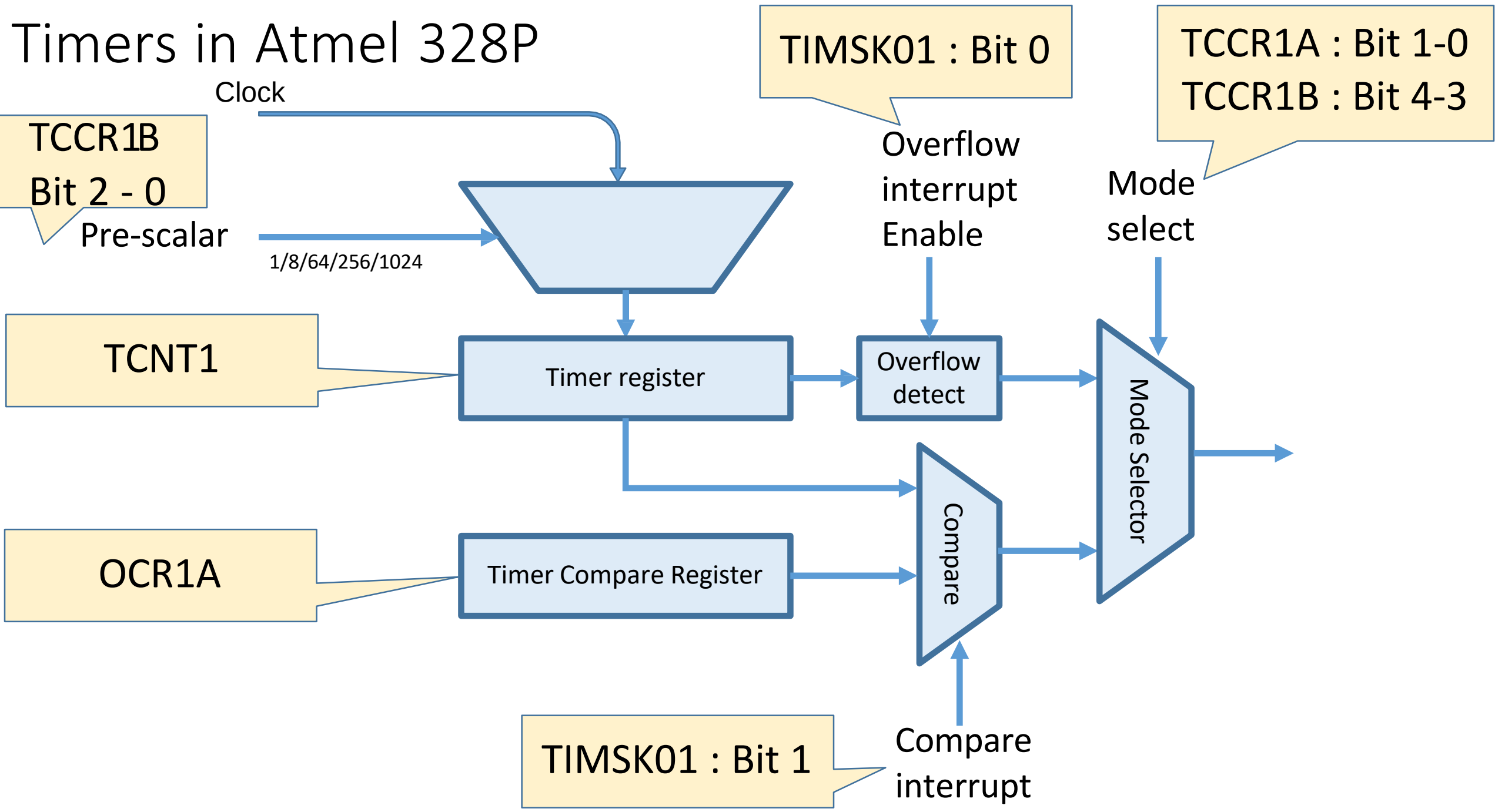
5 pre-scale options : 1, 8, 64, 256, 1024

Pre-scalar is to divide the main clock to make the clock pulse longer

Timers in Atmel 328P



Timers in Atmel 328P



Timer Interrupt 0 : Important Registers

TCNT1 – Timer/Counter Register 1 (16-bit)

OCR1A – Output Compare Register A

OCR1B – Output Compare Register B

TCCR1A, TCCR1B – Timer/Counter Control Registers A & B

TIMSK01– Timer Interrupt Mask Register 1

TOV interrupt (Timer overflow interrupt) [NORMAL MODE]

Compare interrupt A [COMPARE MODE]

Compare interrupt B [COMPARE MODE]

TIFR1 – Timer/Counter Interrupt Flag Register

Timer Interrupt 1 : Mode of Operation

Mode	WGM13	WGM12 (CTC1)[1]	WGM11 (PWM11)[1]	WGM10 (PWM10)[1]	Timer/ Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, Phase	OCR1A	BOTTOM	BOTTOM

Name: TCCR1A
Offset: 0x80
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	COM1	COM1	COM1	COM1			WGM11	WGM10
Access	R/W	R/W	R/W	R/W			R/W	R/W
Reset	0	0	0	0			0	0

Name: TCCR1B
Offset: 0x81
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	ICNC1	ICES1		WGM13	WGM12	CS12	CS11	CS10
Access	R/W	R/W		R/W	R/W	R/W	R/W	R/W
Reset	0	0		0	0	0	0	0

Timer Interrupt 1 : Pre-scalar

CS12	CS11	CS10	Description
0	0	0	No clock source (Timer/Counter stopped).
0		1	$clk_{I/O}$/1 (No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	clkI/O/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)

Name: TCCR1B
Offset: 0x81
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	ICNC1	ICES1		WGM13	WGM12	CS12	CS11	CS10
Access	R/W	R/W		R/W	R/W	R/W	R/W	R/W
Reset	0	0		0	0	0	0	0

Timer Interrupt 1 : Compare Interrupt Enable

There are 2 timer overflow interrupts, namely, A and B

We can use either of them and corresponding registers to set it up.

Name: TIMSK1
Offset: 0x6F
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
			ICIE			OCIEB	OCIEA	TOIE
Access			R/W			R/W	R/W	R/W
Reset			0			0	0	0

Timer Interrupt 1 : Compare Interrupt Enable

There are 2 timer overflow interrupts, namely, A and B

We can use either of them and corresponding registers to set it up.

Name: OCR1AH
Offset: 0x89
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	OCR1AH[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Name: OCR1AL
Offset: 0x88
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	OCR1AL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Maximum Interval for Each Pre-scalar

Overflow register is 16 bits. Therefore, the maximum count it can support if 2^{16} clock pulses

Main clock pulse duration is 1/16,000,000 seconds are 16 MhZ clock frequency.

Pre-scalar value	Timer Overflow interrupt duration	Setting for CS12,CS11,CS10	Setting at TCCR1B
1	(1/16,000,000) * 1 * 2^{16} = 4.096 mS	001	TCCR1B |= 0b00000001 TCCR1B &= 0b11111001
8	(1/16,000,000) * 8 * 2^{16} = 32.768 mS	010	TCCR1B |= 0b00000010 TCCR1B &= 0b11111010
64	(1/16,000,000) * 64 * 2^{16} = 262.144 mS	011	TCCR1B |= 0b00000011 TCCR1B &= 0b11111011
256	(1/16,000,000) * 256 * 2^{16} = 1.048576 S	100	TCCR1B |= 0b00000100 TCCR1B &= 0b11111100
1024	(1/16,000,000) * 1024 * 2^{16} = 4.194304 S	101	TCCR1B |= 0b00000101 TCCR1B &= 0b11111101

Timer Interrupt 1 : Mode of Operation

Mode	WGM13	WGM12 (CTC1)[1]	WGM11 (PWM11)[1]	WGM10 (PWM10)[1]	Timer/ Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1		
5	0	1	0	1	Fast PWM, 8-bit	0x00F		
6	0	1	1	0	Fast PWM, 9-bit	0x01F		
7	0	1	1	1	Fast PWM, 10-bit	0x03F		
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1		
9	1	0	0	1	PWM, Phase	OCR1A	BOTTOM	BOTTOM

To enable timer 1 compare interrupt A
Setting for TMISK1

TCCR1A &= 0b11111100
TCCR1B &= 0b11101111
TCCR1B |= 0b00001000

Name: TCCR1A
Offset: 0x80
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	COM1	COM1	COM1	COM1			WGM11	WGM10
Access	R/W	R/W	R/W	R/W			R/W	R/W
Reset	0	0	0	0			0	0

Name: TCCR1B
Offset: 0x81
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	ICNC1	ICES1		WGM13	WGM12	CS12	CS11	CS10
Access	R/W	R/W		R/W	R/W	R/W	R/W	R/W
Reset	0	0		0	0	0	0	0

Selecting Pre-scalar

From the previous table you could see the maximum durations.

Suppose you have to generate a interrupt every 1 second.

Go through the maximum durations in ascending order and find the pre-scalar value that can support 1 second

Pre-scalar value	Timer Overflow interrupt duration	
1	4.096 mS	< 1 Second
8	32.768 mS	< 1 Second
64	262.144 mS	< 1 Second
256	1.048576 S	> 1 Second
1024	4.194304 S	

Choose pre-scalar 256

To choose pre-scalar 256 Pre-scalar value at TCCR1B
TCCR1B |= 0b00000100 TCCR1B &= 0b11111100

Timer Interrupt 1 : Compare Interrupt Enable

There are 2 timer overflow interrupts, namely, A and B

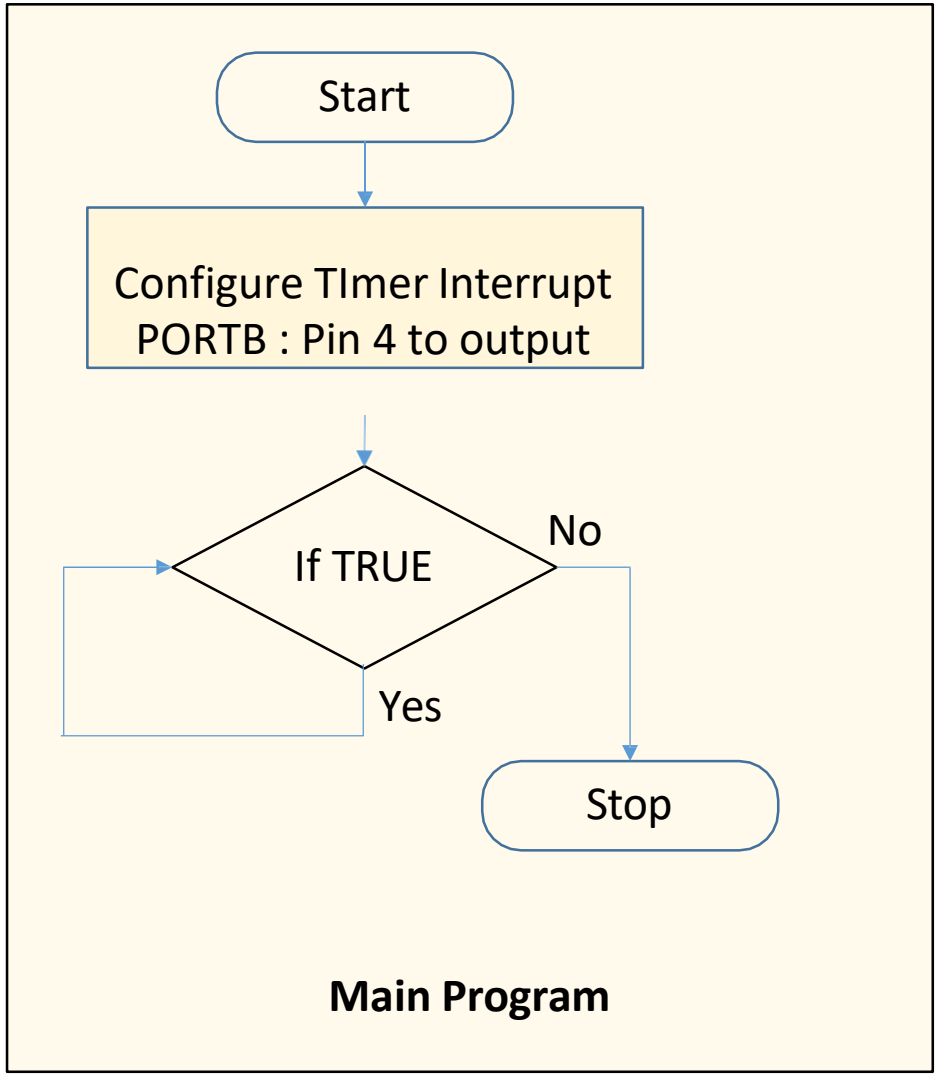
We can use either of them and corresponding registers to set it up.

Name: TIMSK1
Offset: 0x6F
Reset: 0x00
Property: -

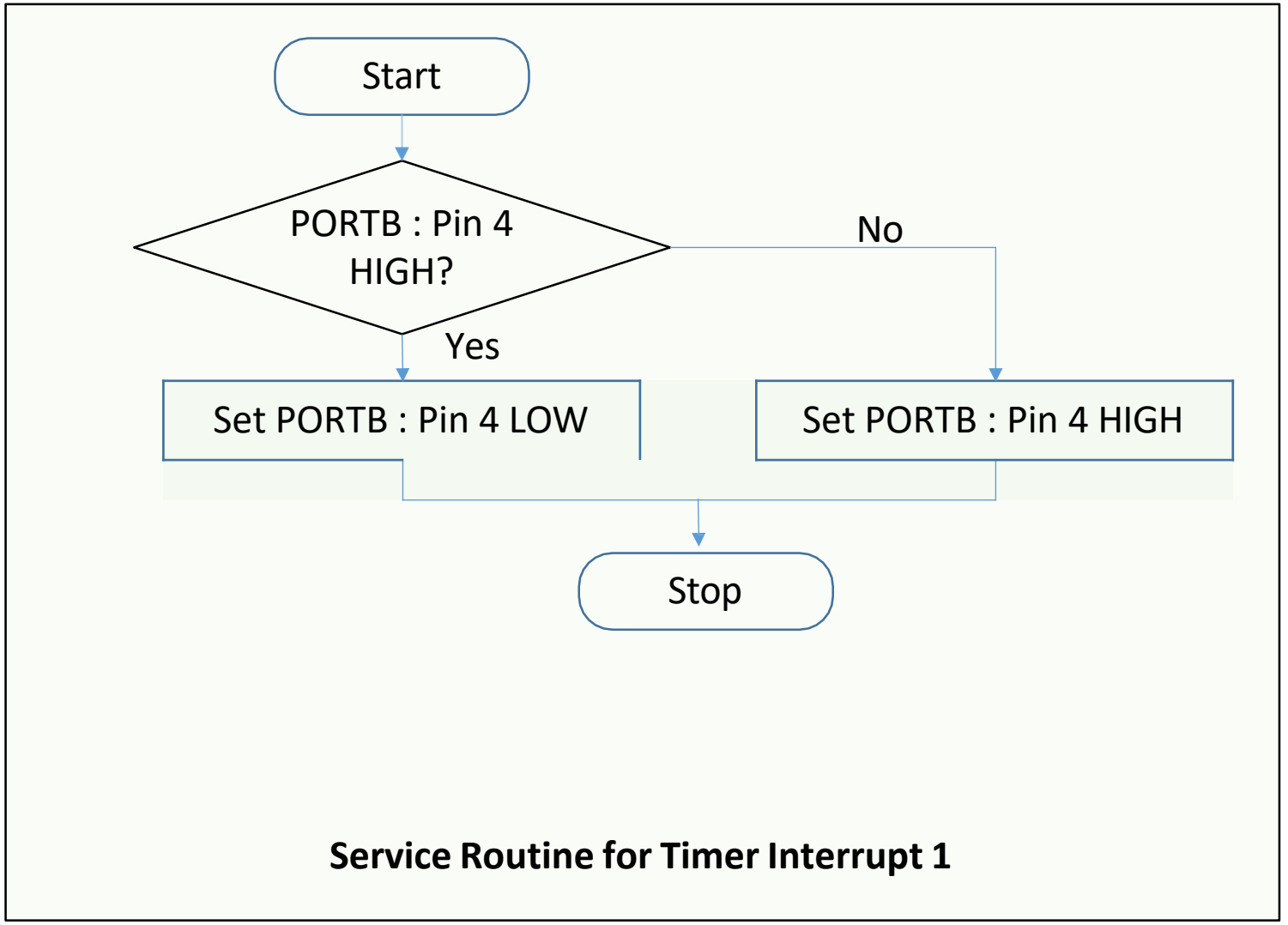
Bit	7	6	5	4	3	2	1	0
			ICIE			OCIEB	OCIEA	TOIE
Access			R/W			R/W	R/W	R/W
Reset			0			0	0	0

To enable timer 1 compare interrupt A Setting for TMISK1
TIMSK1 |= 0b00000010

Making the LED Respond to a Pushbutton Using an Interrupt

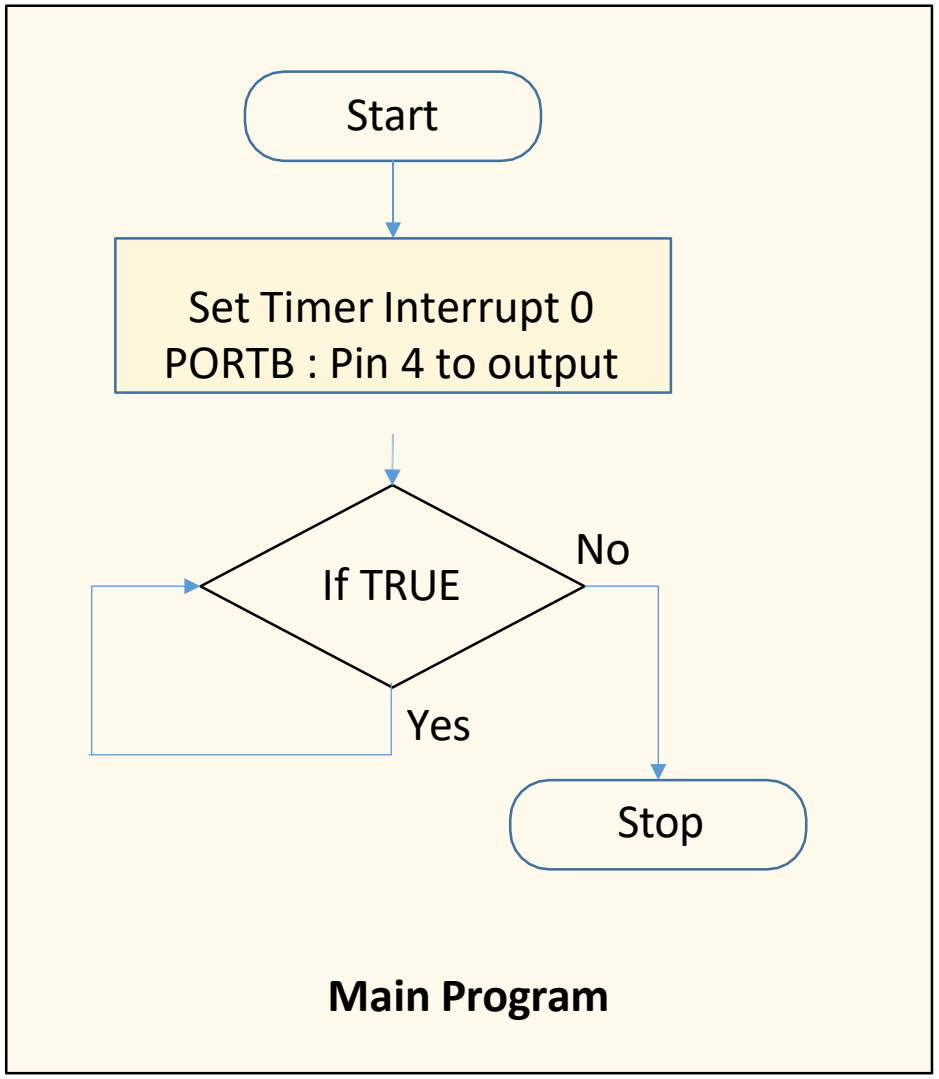


Main Program

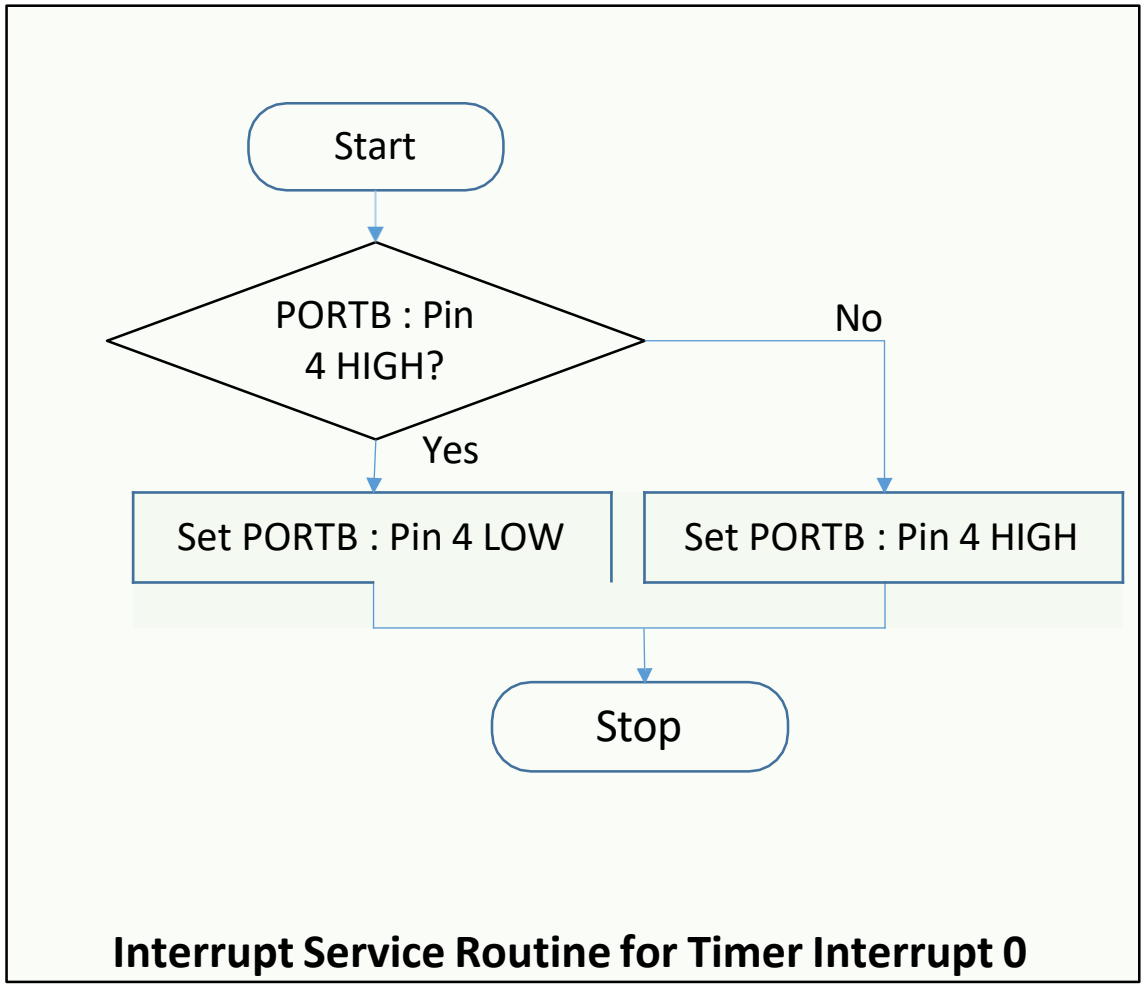


Service Routine for Timer Interrupt 1

Initializing the Timer 1 Compare A Interrupt



Coding the Timer 1 Compare A Interrupt ISR



Interrupt Service Routine for Timer Interrupt 0

Making the LED to toggle when the interrupt trigger

Build the program and run Proteus simulation. Observe behavior of the LED

Simulate Timer 1 Compare Interrupt A : Functions to turn LEDs ON and OFF