**MICROCONTROLLER AND MICROPROCESSOR LAB**

**EXPERIMENT 9**

**AIM**: Write an embedded C program to toggle the port pin with time delay in interrupt mode (8-bit Mode).

**SOFTWARE USED**: Keil uVision5

**Question-1:** Write an Embedded C program to blink the LED connected to port P1 for a continuous 1 ms delay using the 8051 Micro Controller. Blink in delay interval should be generated using timer 0 in interrupt mode. (Timer 0) Crystal Frequency = 11.0592 MHz

**Code**:

#include<reg51.h>

void timer0\_isr (void) interrupt 1{

P1= ~P1;

TH0 = 0XFC;

TL0 = 0X67;

}

void main(){

TMOD = 0X01;

IE = 0X82;

TH0 = 0XFC;

TL0 = 0X67;

TR0 = 1;

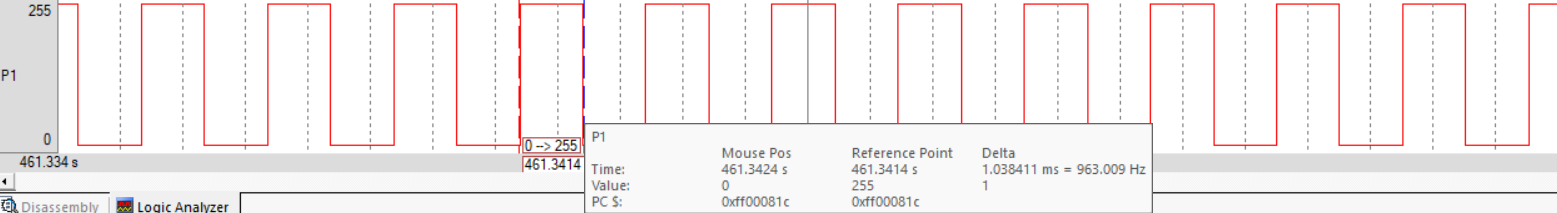
while(1);

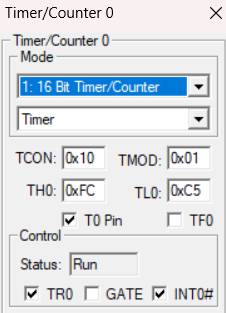
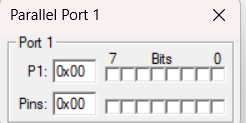
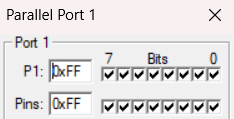
}

**Algorithm**:

1. Include the 8051-header file.
2. Define Timer 0 ISR to toggle the LED state.
3. Initialize Timer 0 in mode 1 (16-bit mode).
4. Enable Timer 0 interrupt.
5. Load initial values for Timer 0.
6. Start Timer 0.
7. Loop infinitely.

**Result**:

****

**  **

**Conclusion:**

The program uses Timer 0 interrupt to generate a 1 ms delay for LED blinking, achieving precise timing with an 11.0592 MHz crystal oscillator.

**Question-2:** Write an Embedded C program to blink the LED connected to port P1 for a continuous 1 ms delay using the 8051 Micro Controller. Blink in delay interval should be generated using timer 1 in interrupt mode. (Timer 1) Crystal Frequency = 11.0592 MHz

**Code**:

#include<reg51.h>

void timer1\_isr (void) interrupt 3{

P1= ~P1;

TH1 = 0XFC;

TL1 = 0X67;

}

void main(){

TMOD = 0X10;

IE = 0X88;

TH1 = 0XFC;

TL1 = 0X67;

TR1 = 1;

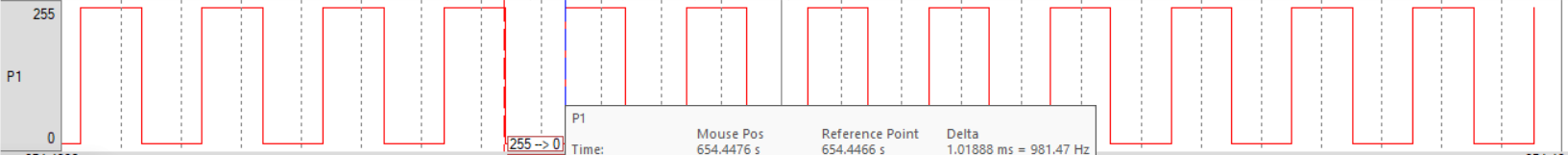
while(1);

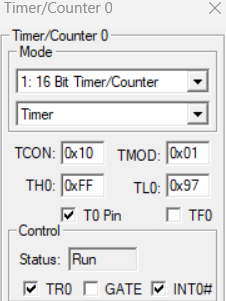
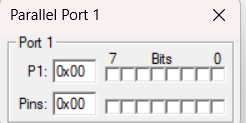
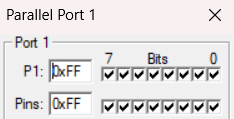
}

**Algorithm**:

1. Include the 8051-header file.
2. Define Timer 1 ISR to toggle the LED state.
3. Initialize Timer 1 in mode 1 (16-bit mode) with auto-reload.
4. Enable Timer 1 interrupt.
5. Load initial values for Timer 1.
6. Start Timer 1.
7. Loop infinitely.

**Result**:

****

**  **

**Conclusion:**

This program utilizes Timer 1 interrupt to generate a 1 ms delay for LED blinking, ensuring precise timing with an 11.0592 MHz crystal oscillator.

**Question-3:** Write an Embedded C program for blinking an LED connected to a port by external interrupt mode 0. Crystal Frequency = 11.0592 MHz

**Code**:

#include<reg51.h>

unsigned int i,j;

void Delay(int t){

for(i=0;i<t;i++){

for(j=0;j<t;j++){

;

}

}

}

void ex0\_isr (void) interrupt 0{

P1= ~P1;

Delay(500);

}

void main(){

IE=0X81;

while(1);

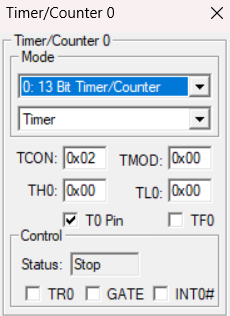
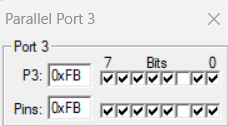
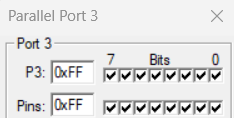
}

**Algorithm**:

* Include 8051 header file.
* Define a delay function `Delay` to create a delay.
* Define external interrupt 0 ISR to toggle LED state.
* Enable external interrupt 0.
* Infinite loop to keep the program running.

**Result**:



**  **

**Conclusion:**

The program utilizes external interrupt 0 to toggle an LED connected to port P1. A delay function is implemented to debounce the switch. The code ensures proper functionality with an 11.0592 MHz crystal oscillator.

**Question-4:** Write an Embedded C program for blinking an LED connected to a port by external interrupt mode 1. Crystal Frequency = 11.0592 MHz

**Code**:

#include<reg51.h>

unsigned int i,j;

void Delay(int t){

for(i=0;i<t;i++){

for(j=0;j<t;j++){

;

}

}

}

void ex0\_isr (void) interrupt 2{

P1= ~P1;

Delay(500);

}

void main(){

IE=0X84;

while(1);

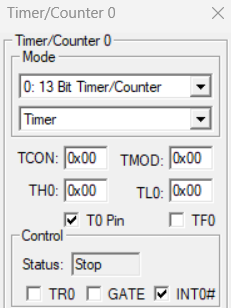
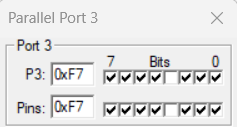
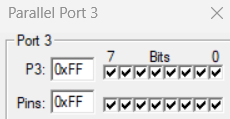
}

**Algorithm**:

1. Include the 8051-header file.
2. Define a delay function `Delay` to create a delay.
3. Define external interrupt 1 ISR to toggle LED state.
4. Enable external interrupt 1.
5. Infinite loop to keep the program running.

**Result**:



**Conclusion:**

The provided code configures external interrupt 1 to toggle an LED connected to port P1. A delay function is implemented to debounce the switch. The program operates under the assumption of an 11.0592 MHz crystal oscillator.

**Question-5:** Write an Embedded C program to generate a PWM of 1KHz 70% duty cycle using timer 0 in interrupt mode. Crystal Frequency = 11.0592 MHz, 16-bit Mode

**Code**:

#include<reg51.h>

unsigned int x=0;

void timer0\_isr (void) interrupt 1{

if(x==0){

P1= 0X00;

TH0 = 0XFE;

TL0 = 0XEC;

x=1;

}

else if(x==1){

P1=0XFF;

TH0=0XFD;

TL0=0X7B;

x=0;

}

}

void main(){

TMOD = 0X01;

IE = 0X82;

TH0 = 0XFD;

TL0 = 0X7B;

TR0 = 1;

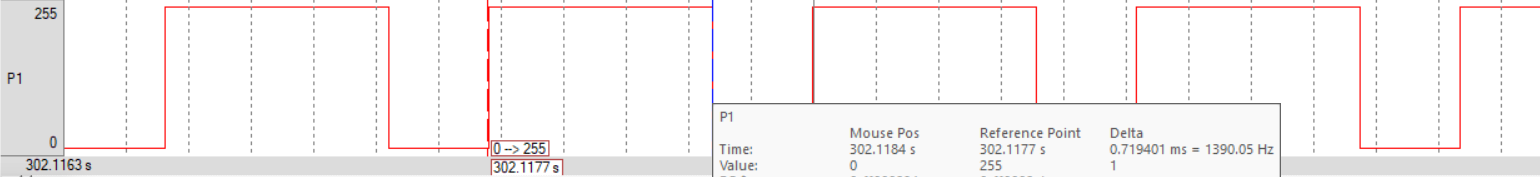
while(1);

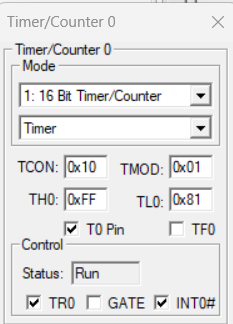
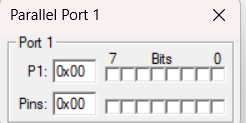
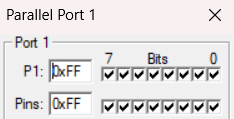
}

**Algorithm**:

1. Include the 8051-header file.
2. Define Timer 0 ISR to generate PWM with a 1 kHz frequency and 70% duty cycle.
3. Initialize Timer 0 in mode 1 (16-bit mode).
4. Enable Timer 0 interrupt.
5. Load initial values for Timer 0.
6. Start Timer 0.
7. Loop infinitely.

**Result**:

****

**  **

**Conclusion:**

The provided code configures Timer 0 to generate a PWM signal with a frequency of 1 kHz and a duty cycle of 70% using interrupt mode. The precise timing is achieved with an 11.0592 MHz crystal oscillator.