Aditi Tapariya
U21EE081

MICROCONTROLLER AND MICROPROCESSOR LAB

EXPERIMENT 8

AIM: Write an embedded C program to toggle the port pin with time delay (16-bit Mode).

SOFTWARE USED: Keil uVision5

Question-1: Blink all the LEDs connected to port pin P 1 with 1ms using timer 0.

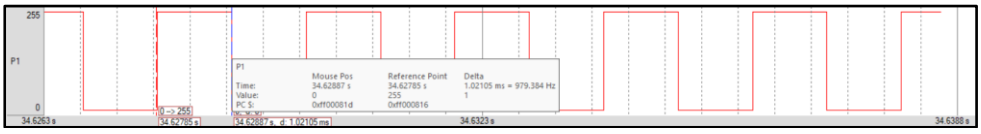
Code:

```
#include<reg51.h>
void timer_delay(void);
void main(){
	P1=0x00;
	while(1){
		P1=0xff;
		timer_delay();
		P1=0x00;
		timer_delay();
	}
}
void timer_delay(void){
	TMOD=0x01;
	TH0=0xfc;
	TL0=0x67;
	TR0=1;
	while (TF0==0) {
		;
	}
	TF0=0;
	TR0=0;
}
```

Algorithm:

1. Include the 8051-header file.
2. Define the function `timer_delay` to create a 1 ms delay using Timer 0.
3. In the main function, set port P1 to all LEDs off initially.
4. Enter an infinite loop.
5. Turn on all LEDs connected to port P1.
6. Call `timer_delay` to create a 1 ms delay.
7. Turn off all LEDs connected to port P1.
8. Call `timer_delay` to create a 1 ms delay.
9. Repeat steps 5-8 infinitely.

Result:



Conclusion:
The provided code blinks all the LEDs connected to port P1 with a 1 ms delay using Timer 0. It alternates between turning all LEDs on and off, creating a blinking effect.

Question-2: Blink the LED connected to port pin P1.1 with 1ms using timer 0.

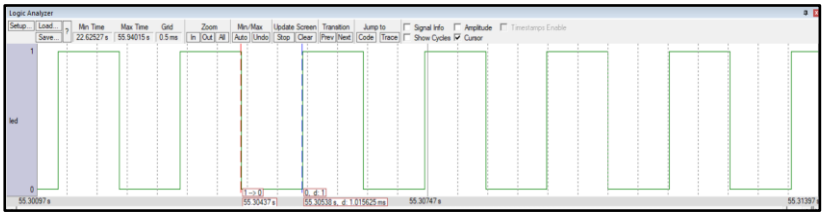
Code:

```
#include<reg51.h>
void delay(void);
sbit led = P1^1;
void main()
{
	while(1)
	{
		led=1;
		delay();
		led=0;
		delay();
	}
}
void delay(void)
{
	TMOD=0x01;
	TH0=0xFC;
	TL0=0x67;
	TR0=1;
	while(TF0 == 0);
	TF0=0;
	TR0=0;
}
```

Algorithm:

1. Include the 8051-header file.
2. Define a function `delay` to create a 1 ms delay using Timer 0.
3. Define a sbit `led` to represent the LED connected to port P1.1.
4. In the main function, enter an infinite loop.
5. Turn on the LED.
6. Call `delay` to create a 1 ms delay.
7. Turn off the LED.
8. Call `delay` to create a 1 ms delay.
9. Repeat steps 5-8 infinitely.

Result:



Conclusion:
The provided code blinks the LED connected to port P1.1 with a 1 ms delay using Timer 0. It alternates between turning the LED on and off, creating a blinking effect.

Question-3: Blink the LED connected to port pin P1.1 with a 1ms delay whenever the switch connected to port pin P2.0 is pressed using timer 0.

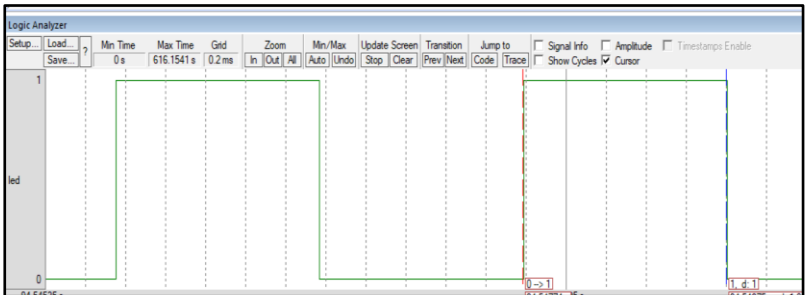
Code:

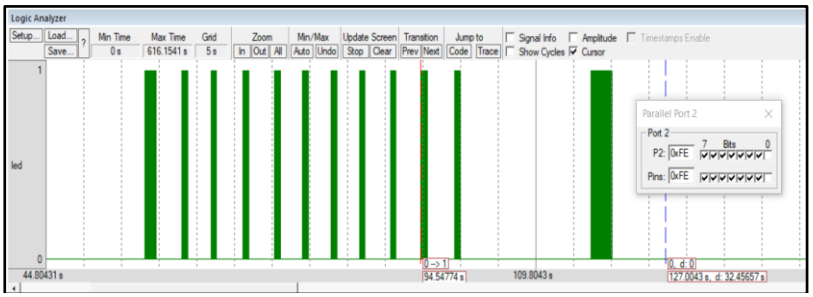
```
#include<reg51.h>
void delay(void);
sbit led = P1^1;
sbit toggle = P2^0;
void main()
{
	while(1)
	{
		if (toggle == 1)
		{
			led=1;
			delay();
			led=0;
			delay();
		}
	}
}
void delay(void)
{
	TMOD=0x01;
	TH0=0xFC;
	TL0=0x67;
	TR0=1;
	while(TF0 == 0);
	TF0=0;
	TR0=0;
}
```

Algorithm:

1. Include the 8051-header file.
2. Define a function `delay` to create a 1 ms delay using Timer 0.
3. Define sbit variables `led` and `toggle` to represent the LED connected to port P1.1 and the switch connected to port P2.0, respectively.
4. In the main function, enter an infinite loop.
5. Check if the switch (`toggle`) is pressed (logic 1).
6. If the switch is pressed, turn on the LED.
7. Call `delay` to create a 1 ms delay.
8. Turn off the LED.
9. Call `delay` to create a 1 ms delay.
10. Repeat steps 5-9 indefinitely.

Result:





Conclusion:
The provided code blinks the LED connected to port P1.1 with a 1 ms delay whenever the switch connected to port P2.0 is pressed, using Timer 0 for timing. It ensures the LED blinks only when the switch is pressed.

<u>Question-4</u>: Generate a square wave of frequency 500Hz in port pin P1.1 using timer 1 in 16-bit mode.

Code:

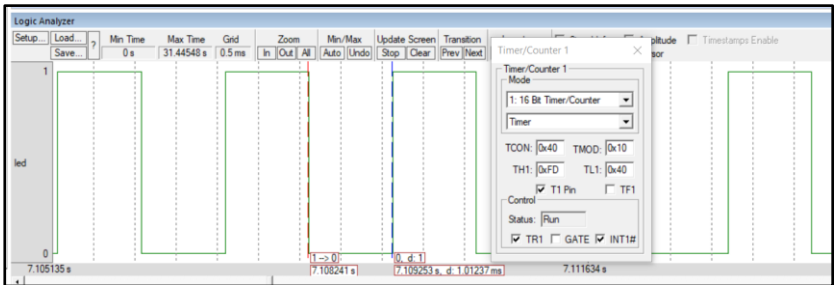
```
#include<reg51.h>
void delay(void);
sbit led = P1^1;
void main()
{
	while(1)
		{
			led=1;
			delay();
			led=0;
			delay();
		}
	}
void delay(void)
{
	TMOD=0x10;
	TH1=0xFC;
	TL1=0x67;
	TR1=1;
	while(TF1 == 0);
	TF1=0;
```

```
    TR1=0;
}
```

Algorithm:

1. Include the 8051-header file.
2. Define a function `delay` to create a delay using Timer 1 in 16-bit mode.
3. Define a sbit `led` to represent the LED connected to port P1.1.
4. In the main function, enter an infinite loop.
5. Turn on the LED.
6. Call `delay` to generate a square wave.
7. Turn off the LED.
8. Call `delay` to generate a square wave.
9. Repeat steps 5-8 indefinitely.

Result:



Conclusion:
The provided code generates a square wave of frequency 500 Hz on port pin P1.1 using Timer 1 in 16-bit mode. It turns the LED on and off alternately to create the square wave.

<u>Question-5</u>: Generate a square wave of frequency 500Hz in port pin P1.1 using timer 1 in 16-bit mode.
Switches are connected from P2.0 to P2.3. Use a timer in polling mode (Timer 1).
P2.0 =1, generate a PWM of 2KHz, 30% duty cycle.
P2.1 =1, generate a PWM of 1KHz, 70% duty cycle.
P2.2 =1, generate a PWM of 500Hz, 30% duty cycle.
P2.3 =1, generate a PWM of 1KHz, 90% duty cycle.

Code:

```
#include<reg51.h>
void t_on(int, int);
void t_off(int, int);
sbit sw1 = P2^0;
sbit sw2 = P2^1;
sbit sw3 = P2^2;
sbit sw4 = P2^3;
sbit led1 = P1^0;
sbit led2 = P1^1;
sbit led3 = P1^2;
sbit led4 = P1^3;
void main()
{
```

```
	while(1)
		{
			if (sw1==1)
			{
				led1=1;
				t_on(0XFF,0X76);
				led1=0;
				t_off(0XFE,0XBD);
			}
			else if (sw2==1)
			{
				led2=1;
				t_on(0XFD,0X7B);
				led2=0;
				t_off(0XFE,0XEB);
			}
			else if (sw3==1)
			{
				led3=1;
				t_on(0XFD,0XD7);
				led3=0;
				t_off(0XFA,0XF6);
			}
			else if (sw4==1)
			{
				led4=1;
				t_on(0XFC,0XC2);
				led4=0;
				t_off(0XFF,0XA4);
			}
		}
	}
void t_on(int th_on, int tl_on)
{
	TMOD=0x10;
	TH1=th_on;
	TL1=tl_on;
	TR1=1;
	while(TF1 == 0);
	TF1=0;
	TR1=0;
}
void t_off(int th_off, int tl_off)
{
	TMOD=0x10;
	TH1=th_off;
	TL1=tl_off;
	TR1=1;
	while(TF1 == 0);
	TF1=0;
```

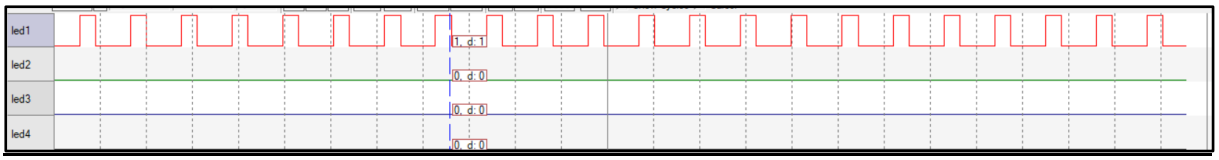
```
        TR1=0;
}
```

Algorithm:

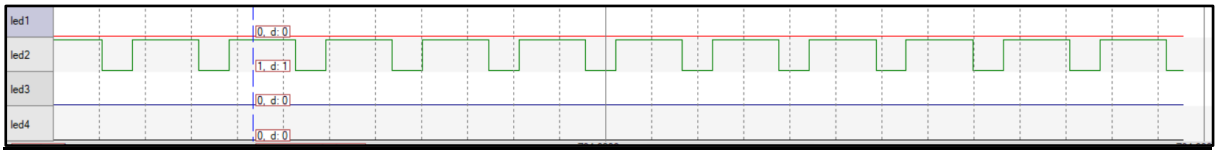
1. Include the 8051-header file.
2. Define functions `t_on` and `t_off` to control Timer 1 for generating PWM.
3. Define sbit variables `sw1`, `sw2`, `sw3`, and `sw4` to represent switches connected to pins P2.0 to P2.3.
4. Define sbit `led0, led1, led2, led3` to represent the LED connected to port P1.0 to P1.3.
5. In the main function, enter an infinite loop.
6. Check the state of switches.
7. If `sw1` is pressed, generate a PWM of 2 kHz with a 30% duty cycle.
8. If `sw2` is pressed, generate a PWM of 1 kHz with a 70% duty cycle.
9. If `sw3` is pressed, generate a PWM of 500 Hz with a 30% duty cycle.
10. If `sw4` is pressed, generate a PWM of 1 kHz with a 90% duty cycle.
11. Repeat steps 5-10 indefinitely.

Result:

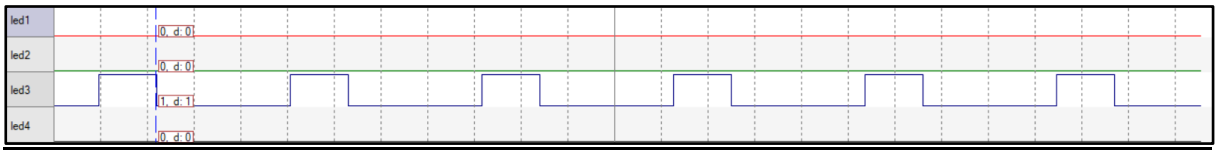
Switch 1:



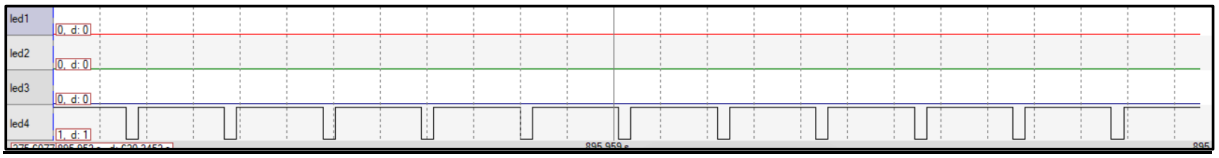
Switch 2:



Switch 3:



Switch 4:



Conclusion:

The provided code generates PWM signals of varying frequencies and duty cycles based on the states of switches connected to pins P2.0 to P2.3. It utilizes Timer 1 in polling mode to achieve this functionality. Additionally, it controls an LED connected to port P1.0 based on the switch states.