**Lab:9**

**Implementation of minutes counter using Timer**



**MBSD Lab**

**Spring 2023**

**Submitted by:**

**Muhammad Fahad Khan [20 Pwcse1940]**

**Semester: 6**

“On my honor, as a student of University of Engineering and Technology Peshawar, I have neither nor received unauthorized assistance on this academic work”

**Submitted to:**

**Dr:Amad Khalil**

**Task1: Write a program to generate 500 KHz signal with 25% duty cycle.**

**Answer:**

To generate a 500 kHz signal with a 25% duty cycle using an 8051 microcontroller, you can utilize Timer 1 in mode 2 and adjust the timer settings accordingly. Here's a sample program in C that achieves this:

**Source Code:**

#include <8051.h>

void main() {

TMOD = (TMOD & 0xF0) | 0x02; // Set Timer 1 in mode 2 (8-bit autoreload mode)

TH1 = 0xFA; // Set the initial value of Timer 1 to achieve a 500 kHz frequency

TL1 = TH1; // Set the same initial value for Timer 1's low byte

while(1) {

// Toggle the output pin to generate the square wave

P1 ^= (1 << 5);

// Wait for the specified time to achieve the desired frequency and duty cycle

// In this case, we are using a constant value for simplicity, but you may need to fine-tune it

// based on the actual clock frequency and specific requirements of your microcontroller

// For a 500 kHz signal with a 25% duty cycle, a delay of approximately 1 microsecond is needed

// You can experiment with different delay values to achieve the desired frequency and duty cycle accurately

for (unsigned int i = 0; i < 10; i++) {

for (unsigned int j = 0; j < 30; j++) {

// A loop to introduce a delay

}

}

}

}

```

In this program, Timer 1 is configured in mode 2, which is an 8-bit autoreload mode. We set the initial value of Timer 1 (TH1) to achieve a timer overflow frequency of approximately 500 kHz. The same initial value is copied to Timer 1's low byte (TL1) to synchronize the timer.

The delay loop in the program introduces a delay to achieve the desired frequency and duty cycle. In this example, we use a constant delay value for simplicity, but you may need to adjust it based on the actual clock frequency of your microcontroller and specific requirements.

Please note that the actual frequency and duty cycle achieved may vary based on the clock frequency and timing characteristics of your microcontroller. Adjustments may be necessary to fine-tune the program for your specific hardware setup.