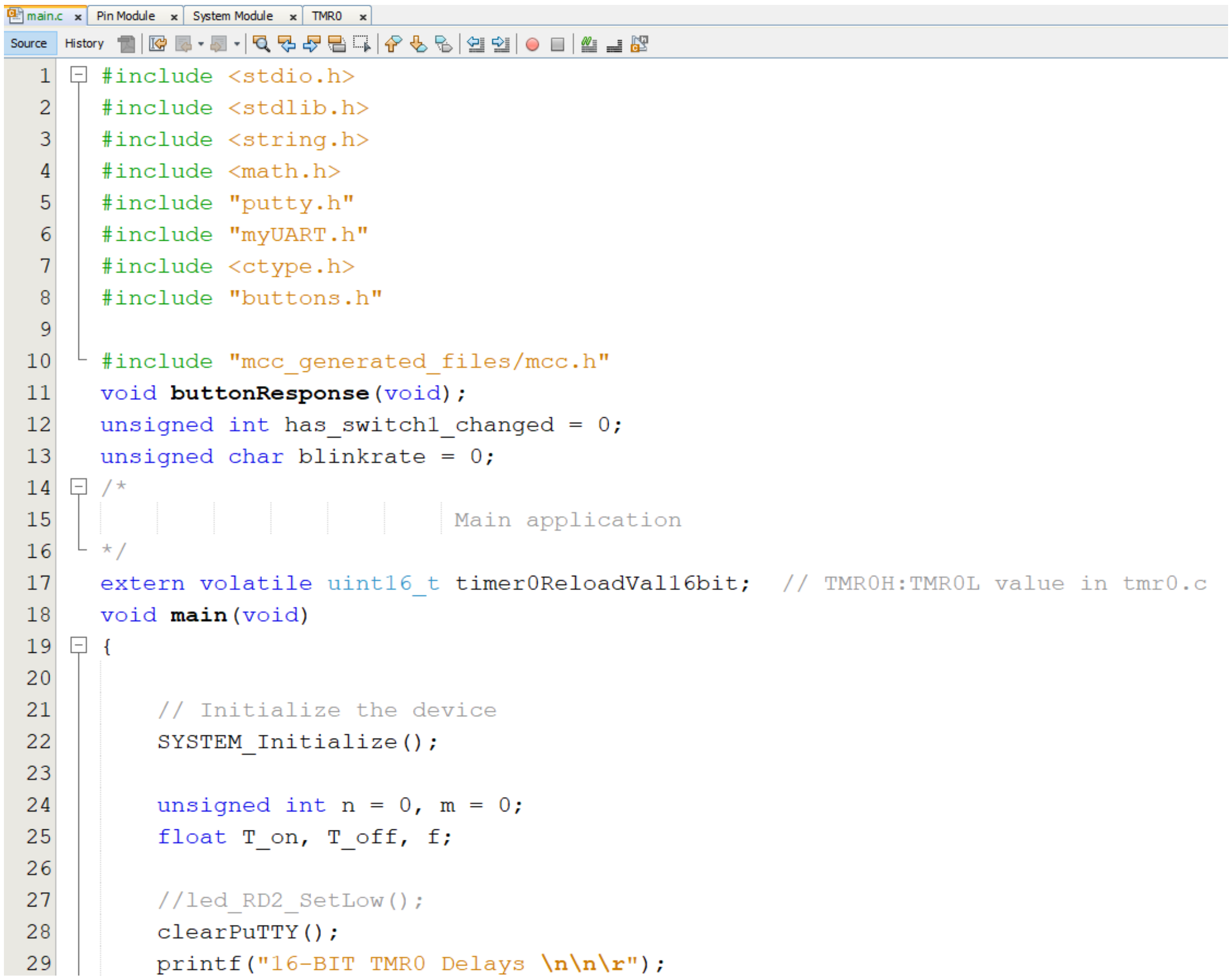
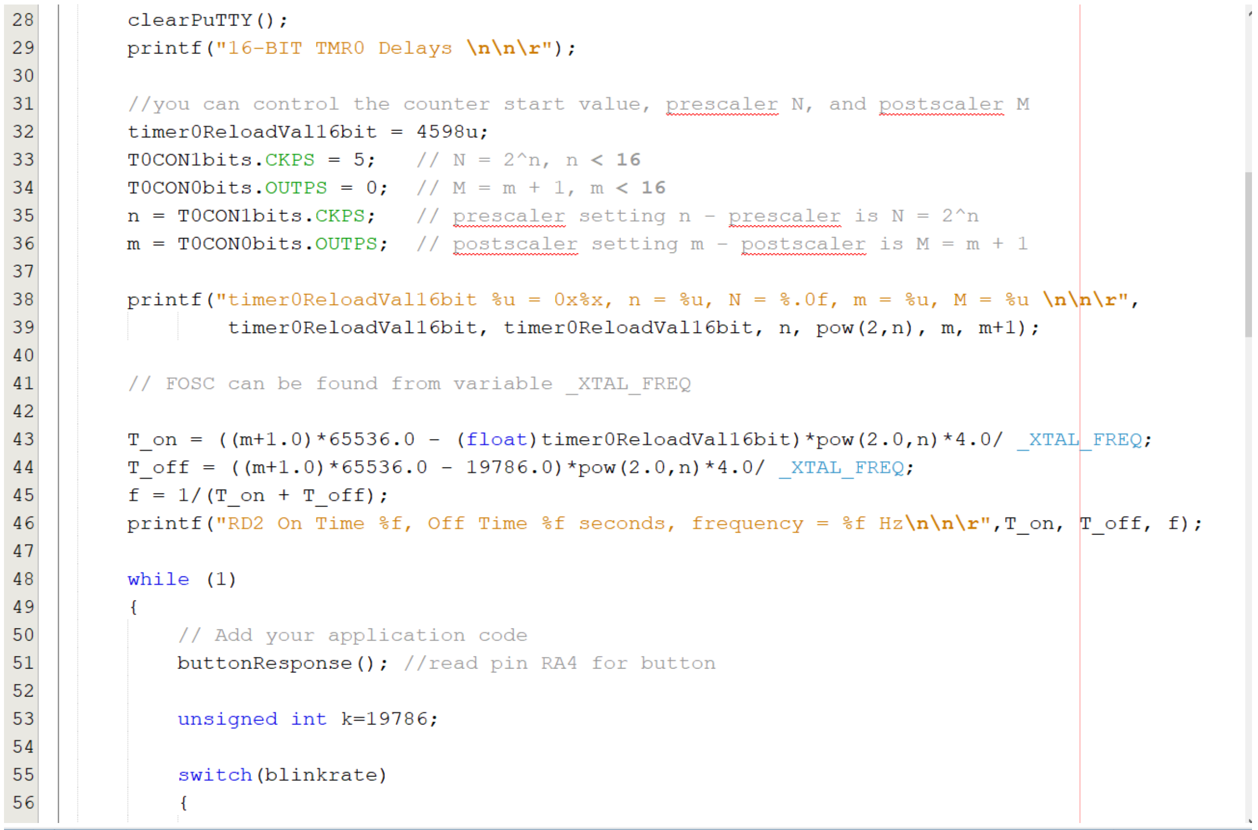
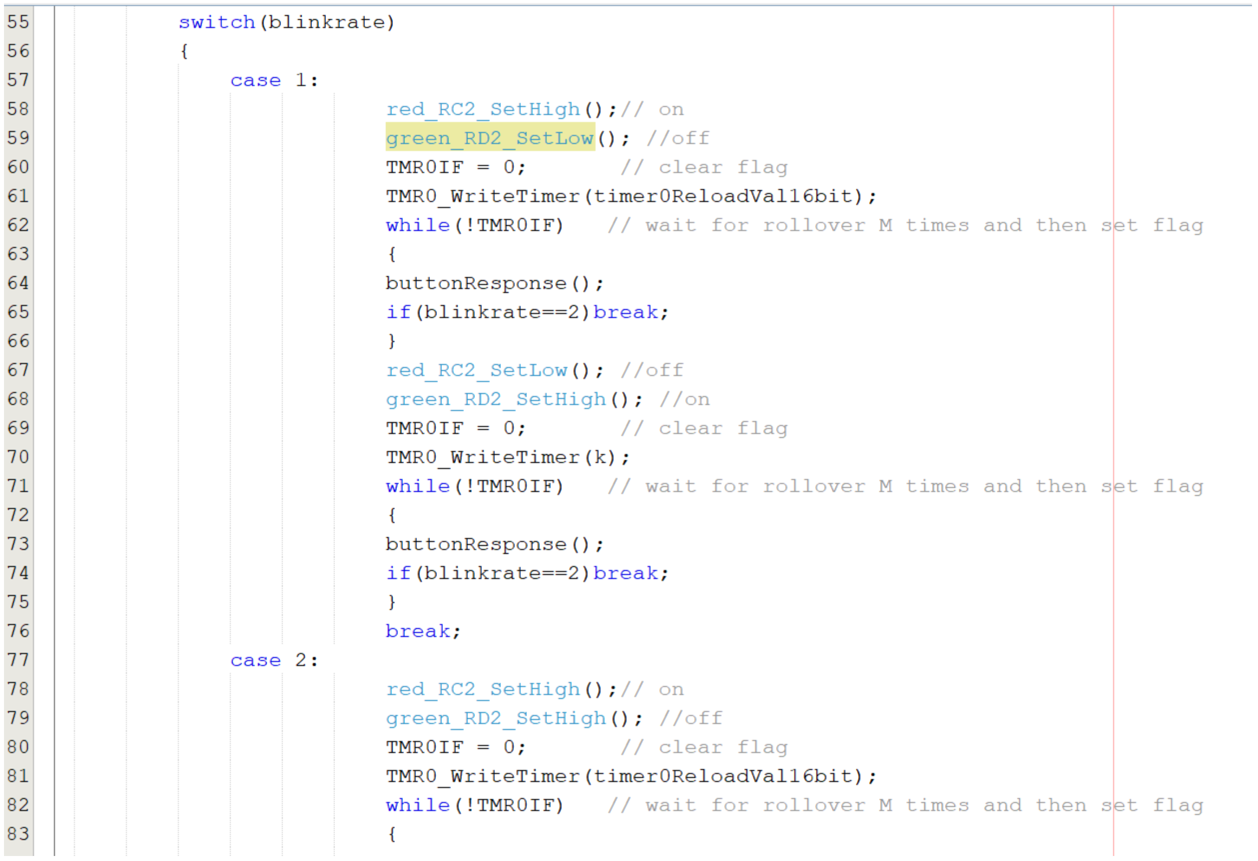


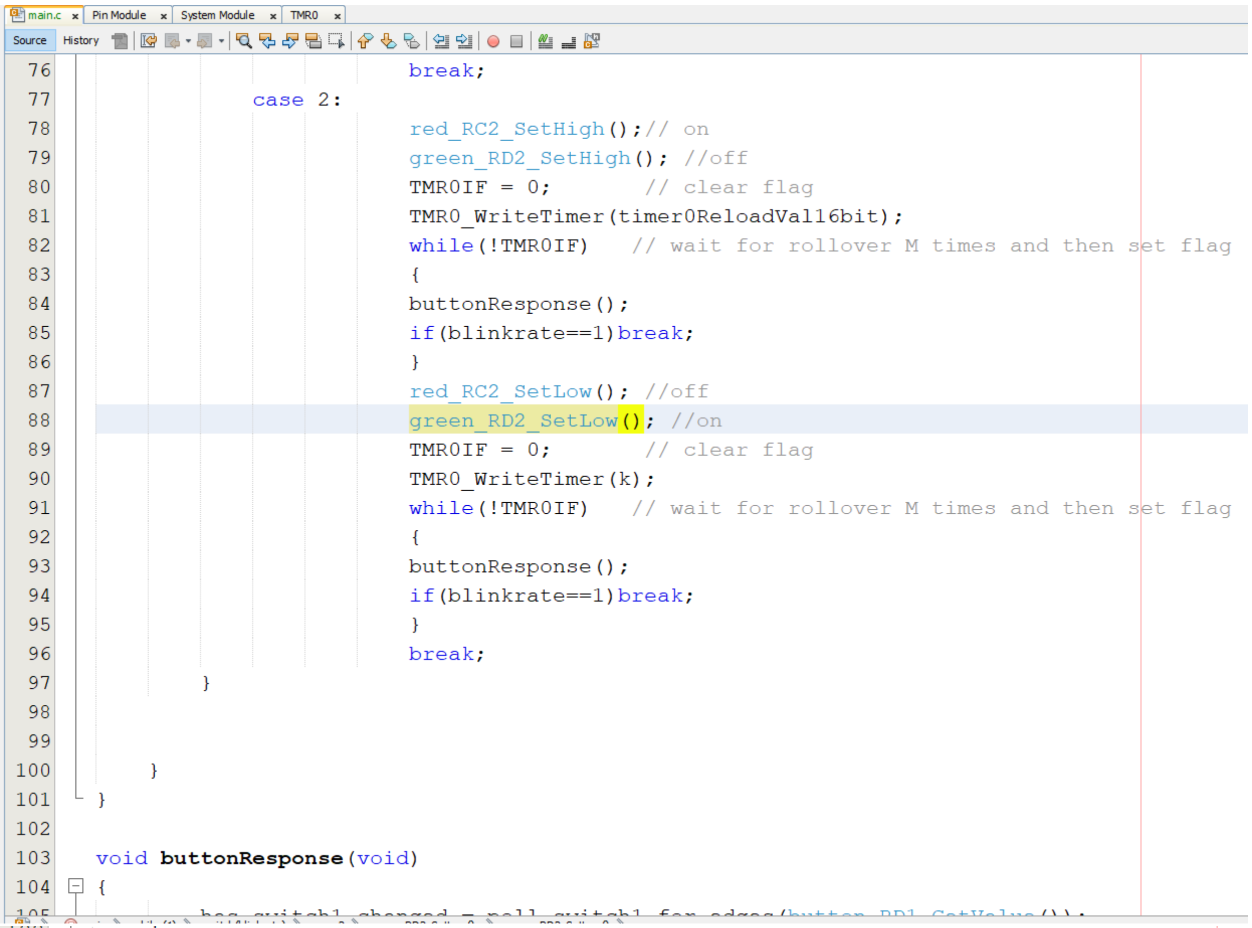
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include "putty.h"
#include "myUART.h"
#include <ctype.h>
#include "buttons.h"

#include "mcc_generated_files/mcc.h"
void buttonResponse(void);
unsigned int has_switch1_changed = 0;
unsigned char blinkrate = 0;
/*
                         Main application
 */
extern volatile uint16_t timer0ReloadVal16bit;  // TMR0H:TMR0L value in tmr0.c
void main(void)
{

    // Initialize the device
    SYSTEM_Initialize();

    unsigned int n = 0, m = 0;
    float T_on, T_off, f;

    //led_RD2_SetLow();
    clearPuTTY();
    printf("16-BIT TMR0 Delays \n\n\r");
```



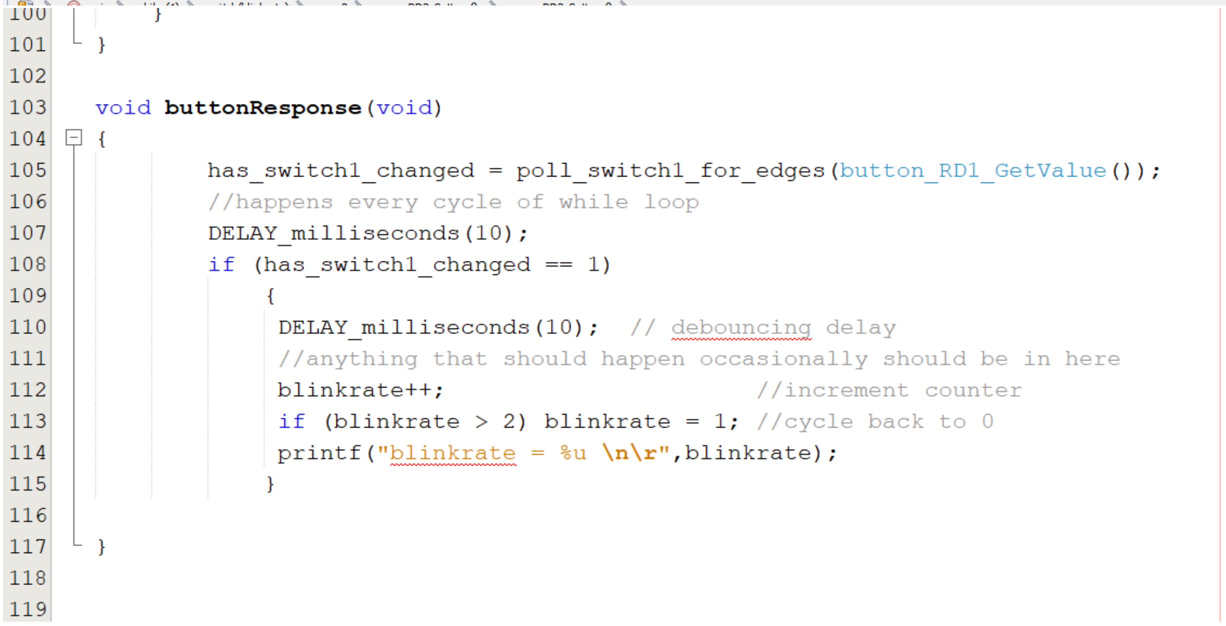
```c
    clearPuTTY();
    printf("16-BIT TMR0 Delays \n\n\r");

    //you can control the counter start value, prescaler N, and postscaler M
    timer0ReloadVal16bit = 4598u;
    T0CON1bits.CKPS = 5;    // N = 2^n, n < 16
    T0CON0bits.OUTPS = 0;   // M = m + 1, m < 16
    n = T0CON1bits.CKPS;    // prescaler setting n - prescaler is N = 2^n
    m = T0CON0bits.OUTPS;   // postscaler setting m - postscaler is M = m + 1

    printf("timer0ReloadVal16bit %u = 0x%x, n = %u, N = %.0f, m = %u, M = %u \n\n\r",
            timer0ReloadVal16bit, timer0ReloadVal16bit, n, pow(2,n), m, m+1);

    // FOSC can be found from variable _XTAL_FREQ

    T_on = ((m+1.0)*65536.0 - (float)timer0ReloadVal16bit)*pow(2.0,n)*4.0/ _XTAL_FREQ;
    T_off = ((m+1.0)*65536.0 - 19786.0)*pow(2.0,n)*4.0/ _XTAL_FREQ;
    f = 1/(T_on + T_off);
    printf("RD2 On Time %f, Off Time %f seconds, frequency = %f Hz\n\n\r",T_on, T_off, f);

    while (1)
    {
        // Add your application code
        buttonResponse(); //read pin RA4 for button

        unsigned int k=19786;

        switch(blinkrate)
        {
```



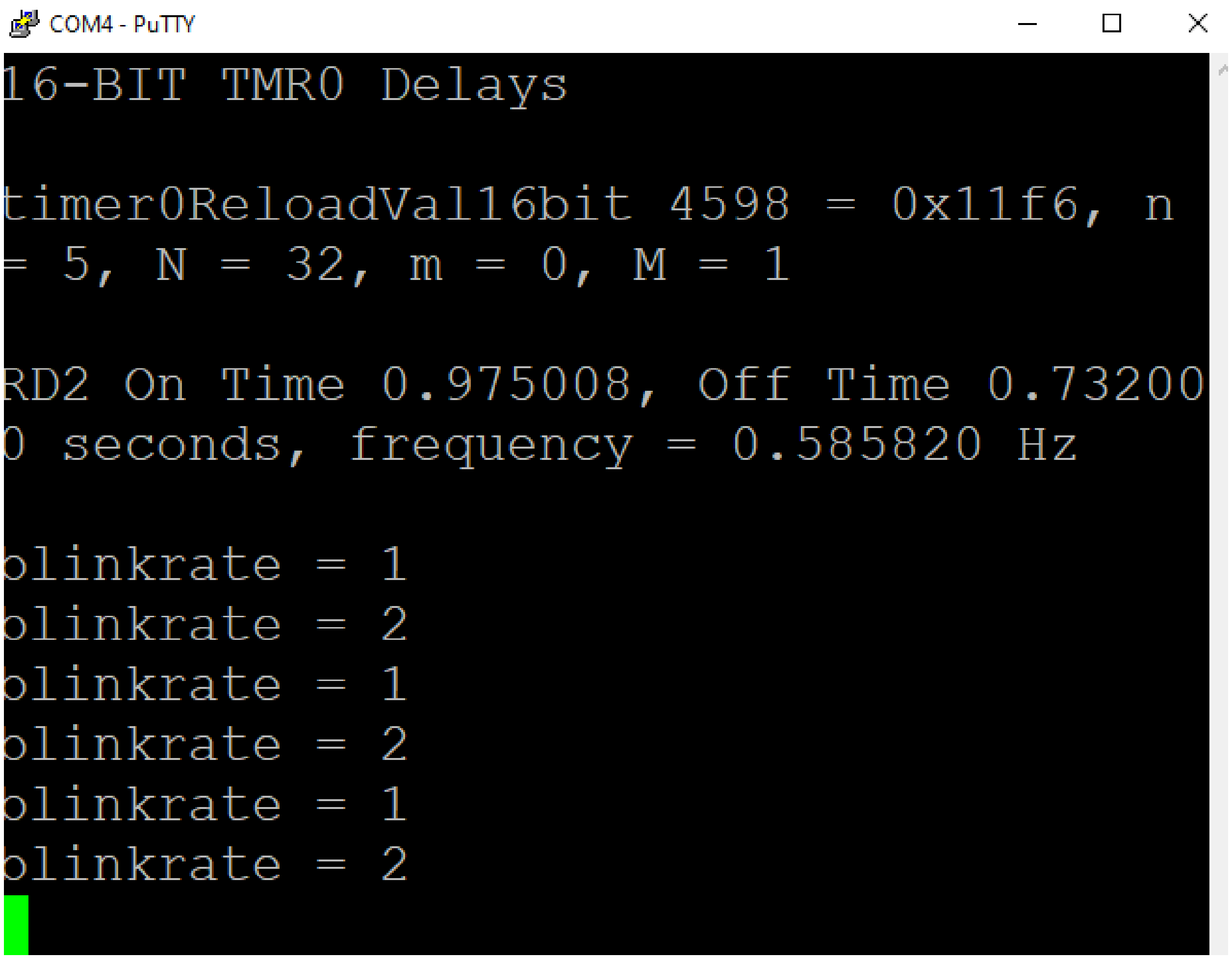
```c
        switch(blinkrate)
        {
            case 1:
                        red_RC2_SetHigh();// on
                        green_RD2_SetLow(); //off
                        TMR0IF = 0;         // clear flag
                        TMR0_WriteTimer(timer0ReloadVal16bit);
                        while(!TMR0IF)    // wait for rollover M times and then set flag
                        {
                        buttonResponse();
                        if(blinkrate==2)break;
                        }
                        red_RC2_SetLow(); //off
                        green_RD2_SetHigh(); //on
                        TMR0IF = 0;         // clear flag
                        TMR0_WriteTimer(k);
                        while(!TMR0IF)    // wait for rollover M times and then set flag
                        {
                        buttonResponse();
                        if(blinkrate==2)break;
                        }
                        break;
            case 2:
                        red_RC2_SetHigh();// on
                        green_RD2_SetHigh(); //off
                        TMR0IF = 0;         // clear flag
                        TMR0_WriteTimer(timer0ReloadVal16bit);
                        while(!TMR0IF)    // wait for rollover M times and then set flag
                        {
```



```
                        break;
            case 2:
                        red_RC2_SetHigh();// on
                        green_RD2_SetHigh(); //off
                        TMR0IF = 0;       // clear flag
                        TMR0_WriteTimer(timer0ReloadVal16bit);
                        while(!TMR0IF)   // wait for rollover M times and then set flag
                        {
                        buttonResponse();
                        if(blinkrate==1)break;
                        }
                        red_RC2_SetLow(); //off
                        green_RD2_SetLow(); //on
                        TMR0IF = 0;       // clear flag
                        TMR0_WriteTimer(k);
                        while(!TMR0IF)   // wait for rollover M times and then set flag
                        {
                        buttonResponse();
                        if(blinkrate==1)break;
                        }
                        break;
        }


    }
}

void buttonResponse(void)
{
```



```
    }
}

void buttonResponse(void)
{
        has_switch1_changed = poll_switch1_for_edges(button_RD1_GetValue());
        //happens every cycle of while loop
        DELAY_milliseconds(10);
        if (has_switch1_changed == 1)
            {
             DELAY_milliseconds(10);  // debouncing delay
             //anything that should happen occasionally should be in here
             blinkrate++;                      //increment counter
             if (blinkrate > 2) blinkrate = 1; //cycle back to 0
             printf("blinkrate = %u \n\r",blinkrate);
            }

}
```



```
16-BIT TMR0 Delays

timer0ReloadVal16bit 4598 = 0x11f6, n
= 5, N = 32, m = 0, M = 1

RD2 On Time 0.975008, Off Time 0.73200
0 seconds, frequency = 0.585820 Hz

blinkrate = 1
blinkrate = 2
blinkrate = 1
blinkrate = 2
blinkrate = 1
blinkrate = 2
```