

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <math.h>
5 #include "putty.h"
6 #include "myUART.h"
7 #include <ctype.h>
8 #include "buttons.h"
9
10 #include "mcc_generated_files/mcc.h"
11 void buttonResponse(void);
12 unsigned int has_switch1_changed = 0;
13 unsigned char blinkrate = 0;
14 /*
15 | | | | | | | Main application
16 */
17 extern volatile uint16_t timer0ReloadVal16bit; // TMR0H:TMR0L value in tmr0.c
18 void main(void)
19 {
20
21     // Initialize the device
22     SYSTEM_Initialize();
23
24     unsigned int n = 0, m = 0;
25     float T_on, T_off, f;
26
27     //led_RD2_SetLow();
28     clearPuTTY();
29     printf("16-BIT TMR0 Delays \n\n\r");
```

```

28 clearPuTTY();
29 printf("16-BIT TMR0 Delays \n\n\r");
30
31 //you can control the counter start value, prescaler N, and postscaler M
32 timer0ReloadVall6bit = 4598u;
33 T0CON1bits.CKPS = 5; // N = 2^n, n < 16
34 T0CON0bits.OUTPS = 0; // M = m + 1, m < 16
35 n = T0CON1bits.CKPS; // prescaler setting n - prescaler is N = 2^n
36 m = T0CON0bits.OUTPS; // postscaler setting m - postscaler is M = m + 1
37
38 printf("timer0ReloadVall6bit %u = 0x%x, n = %u, N = %.0f, m = %u, M = %u \n\n\r",
39 timer0ReloadVall6bit, timer0ReloadVall6bit, n, pow(2,n), m, m+1);
40
41 // FOSC can be found from variable _XTAL_FREQ
42
43 T_on = ((m+1.0)*65536.0 - (float)timer0ReloadVall6bit)*pow(2.0,n)*4.0/_XTAL_FREQ;
44 T_off = ((m+1.0)*65536.0 - 19786.0)*pow(2.0,n)*4.0/_XTAL_FREQ;
45 f = 1/(T_on + T_off);
46 printf("RD2 On Time %f, Off Time %f seconds, frequency = %f Hz\n\n\r",T_on, T_off, f);
47
48 while (1)
49 {
50 // Add your application code
51 buttonResponse(); //read pin RA4 for button
52
53 unsigned int k=19786;
54
55 switch(blinkrate)
56 {
57
58 case 1:
59 red_RC2_SetHigh();// on
60 green_RD2_SetLow(); //off
61 TMR0IF = 0; // clear flag
62 TMR0_WriteTimer(timer0ReloadVall6bit);
63 while(!TMR0IF) // wait for rollover M times and then set flag
64 {
65 buttonResponse();
66 if(blinkrate==2)break;
67 }
68 red_RC2_SetLow(); //off
69 green_RD2_SetHigh(); //on
70 TMR0IF = 0; // clear flag
71 TMR0_WriteTimer(k);
72 while(!TMR0IF) // wait for rollover M times and then set flag
73 {
74 buttonResponse();
75 if(blinkrate==2)break;
76 }
77 break;
78
79 case 2:
80 red_RC2_SetHigh();// on
81 green_RD2_SetHigh(); //off
82 TMR0IF = 0; // clear flag
83 TMR0_WriteTimer(timer0ReloadVall6bit);
84 while(!TMR0IF) // wait for rollover M times and then set flag
85 {

```

```

76         break;
77     case 2:
78         red_RC2_SetHigh(); // on
79         green_RD2_SetHigh(); //off
80         TMR0IF = 0; // clear flag
81         TMR0_WriteTimer(timer0ReloadVal16bit);
82         while(!TMR0IF) // wait for rollover M times and then set flag
83         {
84             buttonResponse();
85             if(blinkrate==1)break;
86         }
87         red_RC2_SetLow(); //off
88         green_RD2_SetLow(); //on
89         TMR0IF = 0; // clear flag
90         TMR0_WriteTimer(k);
91         while(!TMR0IF) // wait for rollover M times and then set flag
92         {
93             buttonResponse();
94             if(blinkrate==1)break;
95         }
96         break;
97     }
98
99
100 }
101
102
103 void buttonResponse(void)
104 {
105     has_switch1_changed = poll_switch1_for_edges(button_RD1_GetValue());
106     //happens every cycle of while loop
107     DELAY_milliseconds(10);
108     if (has_switch1_changed == 1)
109     {
110         DELAY_milliseconds(10); // debouncing delay
111         //anything that should happen occasionally should be in here
112         blinkrate++; //increment counter
113         if (blinkrate > 2) blinkrate = 1; //cycle back to 0
114         printf("blinkrate = %u \n\r",blinkrate);
115     }
116
117 }
118
119

```

## 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