

Skyler MacDougall, Matthew Gerace

Homework 4: Due 2/12/2020

1. There are 25 notes of the musical scale. They are each produced by a square wave with the frequency as defined by the following formula:

$$f_{note} = 440 \times 2^{\frac{P}{12}} \quad (1)$$
$$-12 \leq P \leq 12$$

The period of the square wave is as follows:

$$T_{note} = \frac{1}{f_{note}} \quad (2)$$

$$hint : (OCR1A + 1) = \frac{T_{note}}{2} = \frac{16MHz}{2 \times f_{note}}$$

1. For the 25 notes where $220 \leq f_{note} \leq 880$, determine the `OCR1A` values needed to output the square wave of each note using Timer1 in CTC mode.
2. Open another program and use `#define` to create 25 constants for the results of part 1. Also create time delay constants.
3. In the initialization section setup Timer1 to create a square wave.
4. Create a function called `playNote()`.
 1. It takes in an integer representing the max count for a given note and assigns that value to the `OCR1A` register.
 2. It also takes in a delay constant to hold the note for a given amount of time.
 3. it must then turn off CTC mode and `delay(50)` to put a break between notes.
5. In the main loop call the function with different notes and delays in between. You can look up simple songs on the internet and experiment with delays to actually play a song.
6. Use the speaker in your lab kit and drive one pin with the waveform output and the other to ground.

Simple Gifts

Brackett



© Copyright 2013 RK Deverich. All rights reserved.

The only authorized display of this sheet music is on the website **ViolinOnline.com**. This content is protected by international copyright law. It is illegal to copy, post, display or publish this content in any form, and displaying this music on other websites, blogs, feeds or social media is prohibited. Permission is given for individual users to print and perform this music for their personal, noncommercial use.

```
1 //Note-name/constant conversion
2 #define LA 36363
3 #define LAS 34323
4 #define LB 32396
5 #define LC 30578
6 #define LCS 28861
7 #define LD 27242
8 #define LDS 25713
9 #define LE 24270
10 #define LF 22908
11 #define LFS 21622
12 #define LG 20408
13 #define LGS 19263
```

```
14 #define MA 18182
15 #define MAS 17161
16 #define MB 16198
17 #define MC 15289
18 #define MCS 14430
19 #define MD 13620
20 #define MDS 12856
21 #define ME 12135
22 #define MF 11454
23 #define MFS 10811
24 #define MG 10204
25 #define MGS 9631
26 #define HA 9091
27
28 //Note Length definitions
29 #define QUARTER 400
30 #define EIGHTH 200
31 #define DOTQUARTER 600
32 #define HALF 800
33
34 void setup(){
35     TCCR1A=0x40;
36     TCCR1B=0x09;
37     TCCR1C=0;
38     TCNT1=0;
39     DDRB|=0x02;
40 }
41
42 void playNote(int note, int time){
43     OCR1A = note;
44     delay(time);
45     OCR1A = 0;
46     delay(50);
47 }
48
49 void loop(){
50     playNote(LD, EIGHTH);
51     playNote(LD, EIGHTH);
52     playNote(LG, QUARTER);
53     playNote(LG, EIGHTH);
54     playNote(MA, EIGHTH);
55     playNote(MB, EIGHTH);
56     playNote(LG, EIGHTH);
57     playNote(MB, EIGHTH);
58     playNote(MC, EIGHTH);
59     playNote(MD, QUARTER);
60     playNote(MD, EIGHTH);
61     playNote(MC, EIGHTH);
62     playNote(MB, QUARTER);
63     playNote(MA, EIGHTH);
64     playNote(LG, EIGHTH);
65     playNote(MA, QUARTER);
66     playNote(MA, QUARTER);
67     playNote(MA, QUARTER);
68     playNote(MA, QUARTER);
69     playNote(MA, EIGHTH);
70     playNote(MB, EIGHTH);
71     playNote(MA, EIGHTH);
```

```
72     playNote(LFS, EIGHTH);
73     playNote(LD, QUARTER);
74     playNote(LD, QUARTER);
75     playNote(LG, EIGHTH);
76     playNote(LFS, EIGHTH);
77     playNote(LG, EIGHTH);
78     playNote(MA, EIGHTH);
79     playNote(MB, QUARTER);
80     playNote(MA, EIGHTH);
81     playNote(MA, EIGHTH);
82     playNote(MB, QUARTER);
83     playNote(MC, QUARTER);
84     playNote(MD, DOTQUARTER);
85     playNote(MD, EIGHTH);
86     playNote(MA, QUARTER);
87     playNote(MA, EIGHTH);
88     playNote(MB, EIGHTH);
89     playNote(MA, QUARTER);
90     playNote(LG, EIGHTH);
91     playNote(LG, EIGHTH);
92     playNote(MA, QUARTER);
93     playNote(LG, EIGHTH);
94     playNote(LFS, EIGHTH);
95     playNote(LG, HALF);
96     playNote(MD, HALF);
97     playNote(MB, DOTQUARTER);
98     playNote(MA, EIGHTH);
99     playNote(MB, EIGHTH);
100    playNote(MC, EIGHTH);
101    playNote(MB, EIGHTH);
102    playNote(MA, EIGHTH);
103    playNote(LG, DOTQUARTER);
104    playNote(MA, EIGHTH);
105    playNote(MB, QUARTER);
106    playNote(MB, EIGHTH);
107    playNote(MC, EIGHTH);
108    playNote(MD, QUARTER);
109    playNote(MB, QUARTER);
110    playNote(MA, QUARTER);
111    playNote(MA, EIGHTH);
112    playNote(MB, EIGHTH);
113    playNote(MA, DOTQUARTER);
114    playNote(LD, EIGHTH);
115    playNote(LG, HALF);
116    playNote(LG, DOTQUARTER);
117    playNote(MA, EIGHTH);
118    playNote(MB, QUARTER);
119    playNote(MB, EIGHTH);
120    playNote(MC, EIGHTH);
121    playNote(MD, QUARTER);
122    playNote(MC, EIGHTH);
123    playNote(MB, EIGHTH);
124    playNote(MA, QUARTER);
125    playNote(MA, QUARTER);
126    playNote(MB, QUARTER);
127    playNote(MB, EIGHTH);
128    playNote(MA, EIGHTH);
129    playNote(LG, QUARTER);
```

```

130     playNote(LG, QUARTER);
131     playNote(LG, HALF);
132     delay(1000);
133 }

```

2. Use Timer1 in normal mode to create an output wave that has the following properties:

$$f = 1Hz \quad (3)$$

$$duty\ cycle = 25\%$$

Assume that this output is driving an LED that will cause it to blink. You can use a 2-state state machine.

(Hint: Use `T0V1` as the exit condition for the states. you will need a prescaler.)

DO NOT USE PWM.

```

1  #define OVERFLAG 0x02
2  #define OVERFLOW 0x01
3  #define LEDHIGH 0x02
4  #define LEDLOW 0xFD
5  enum{LED_on, LED_off, stop};
6  int state = LED_on, prevState = !LED_on;
7  int stateTimer = 0;
8  boolean isNewState;
9  void setup(){
10     TCCR1A = 0x80;
11     TCCR1B = 0x04;
12     TCCR1C = 0;
13     TCNT1 = 65536 - 23437;
14     OCR1A = 24520;
15     DDRB |= 0x02;
16     PORTB &= ~(0x02);
17 }
18
19 void loop(){
20     isNewState = (state != prevState);
21     prevState = state;
22     switch(state){
23         case LED_on:
24             if(isNewState)
25             {
26                 PORTB |= LEDHIGH;
27                 TIFR |= OVERFLOW; //Flag reset
28             }
29
30             if(TIFR & OVERFLAG) state = LED_off;
31             break;
32
33         case LED_off:
34             if(isNewState)
35             {
36                 PORTB &= LEDLOW;
37                 TIFR |= OVERFLAG; //Flag reset
38             }
39
40             if(TIFR & OVERFLOW) state = LED_on;
41             break;

```

```

42
43     case stop:
44         PORTB |= 0x02;
45         break;
46     default: state = stop;
47 }
48 }
49

```

3. For each group member, discuss the feasibility of the invention. Is it feasible? Why or why not? List the I/O devices that each invention would require.

1. Skyler MacDougall

one-way switch

- 2 servos
 - 3 pins each
 - 1 servo to move the arm
 - 1 servo to adjust the y-axis to flip the switch
- 1 switch
 - 1 pin

2. Matthew Gerace

Pool measurement device

Screen for temp and pH readout

I/O devices:

- pH sensor
 - 2 pins
- thermocouple
 - 2 pins
- switch
 - 1 pin
 - for changing between pH and temp
- 2 7seg displays
 - 7 pins per display