

# Signal Generation

N. Narasimhamurthi

## 1 Objective

To become familiar with generating square waves.

## 2 Background

In an earlier lab, you had to generate a square wave signal using various interrupts. This lab builds on this. For a general square wave signal, we define the on-time,  $T_{\text{on}}$ , the off-time,  $T_{\text{off}}$ , and the period  $T$  as shown in the figure 1. The goal of the lab is to generate such square waves. The ratio  $\frac{T_{\text{on}}}{T}$  is called the **duty cycle** and is expressed as a percentage. When dealing with time it is convenient to talk in terms of clock ticks. The HC11 has a 2 MHz e-clock, or you have  $2 \times 10^6$  ticks per second. Hence 1 clock tick is 0.5 microseconds.

## 3 Variable frequency signal generator

In this experiment, we will generate a square wave with frequency selected by the user. We will fix the duty cycle at 25%. To get started, we will first write the code for generating a single tone.

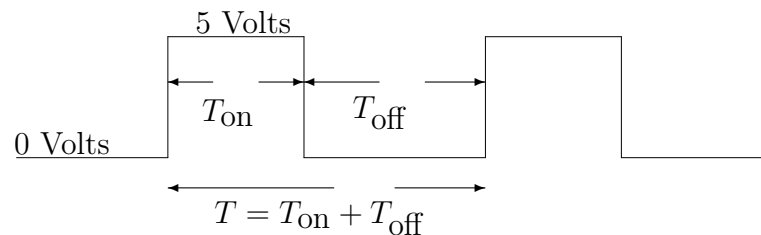


Figure 1: 0-5 volt square wave

### 3.1 500 Hz tone generator

Here is a typical calculation to generate a 500 Hz signal:

$$\begin{aligned}T &= 1/500 = 2 \times 10^{-3} \text{ seconds} = (2 \times 10^{-3}) \times (2 \times 10^6) = 4000 \text{ ticks} \\T_{\text{on}} &= 4000/4 = 1000 \text{ ticks} \\T_{\text{off}} &= 4000 - 1000 = 3000 \text{ ticks}\end{aligned}$$

To generate a 50 Hz, 25% duty cycle signal we use two variables called `ONTIME` and `OFFTIME`. Every time we get an `OC2` interrupt, we toggle the pin `PA4` as before. We check to see if we turned the pin `ON` or `OFF`. If we turned it on, then we schedule the next interrupt to occur `ONTIME` ticks later. However, If we turned it off, then we schedule the next interrupt to occur `OFFTIME` ticks later. Thus, the earlier code that was used to generate a square wave is modified as follows:

```
; Various defines go here ...
    ORG $3000 don't forget the $

ME    FCC /Your name/
      FCB 10
      FCC /ECE 372/
      FCB 10
      FCC /Date the program was last changed/
      FCB 10, 10, 4

ONTIME RMB 2
OFFTIME RMB 2

    ORG $2000 DONT FORGET THE $
    LDX #ME
    JSR OUTSTRG ; MAKE SURE YOU HAVE EQU FOR OUTSTRG

; Enable OC2 interrupt by setting OC2I (bit#6 in TMSK1)
; ALSO PERFORM ALL INITIALIZATION BETWEEN SEI/CLI

    SEI
    LDD #1000
    STD ONTIME
    LDD #3000
    STD OFFTIME
```



```
STAA TFLG1
```

```
RTI
```

```
; Connect the service to the interrupt ;  
ORG $00DC ; $00DC WHERE THE SERVICE STARTS  
JMP SERVICE ; JUMP TO WHERE THE SERVICE CODE ACTUALLY IS
```

Assemble and run the above program. Connect PA4 to an oscilloscope and verify that the duty cycle and the frequency are correct.

### 3.2 Variable frequency generator

We now modify the above code to create a variable frequency generator. The program will monitor the keyboard and depending on the number the user enters, it will change the frequency as shown in the table below:

Number	Frequency	Period seconds	Period ticks	On time ticks	Off time ticks
0	440	0.002273	4545	1136	3409
1	466	0.002145	4290	1073	3217
2	494	0.002025	4050	1013	3037
3	523	0.001911	3822	956	2866
4	554	0.001804	3608	902	2706
5	587	0.001703	3405	851	2554
6	622	0.001607	3214	804	2410
7	659	0.001517	3034	759	2275
8	698	0.001432	2863	716	2147
9	740	0.001351	2703	676	2027

As a programmer we are only interested in the on-time and off-time. We use FDB to create two tables in the data section as shown below:

```
ONTIMETBL
```

```
FDB 1136, 1073, 1013, 956, 902
```

```
FDB 851, 804, 759, 716, 676
```

```
OFFTIMETBL
```

```
FDB 3409, 3217, 3037, 2866, 2706
```

```
FDB 2554, 2410, 2275, 2147, 2027
```

Note that it makes sense to enter numbers in decimal notation. Each entry in the table requires two bytes (we use 16 bit numbers to measure time since the HC11 clock is a 16 bit quantity). Hence we use FDB instead of FCB. Note that this also

means that we have to index through memory in steps of **two bytes**. Suppose we want to access the element #4 in ONTIMETBL and the value #4 is in the **B** register (the number #4 is used only as an illustration. In the application the register **B** will have the value). Then we have to access the element we have to write

```
LDX #ONTIMETBL
ABX
ABX ; NEED TWO ABX'S
??? 0,X
```

We can now modify the single tone generator to a programmable square wave generator by making the following changes:

Before	After
<pre>LDD #1000 STD ONTIME LDD #3000 STD OFFTIME</pre>	<pre>LDX #ONTIMETBL LDY #OFFTIMETBL LDD 0,X STD ONTIME LDD 0,Y STD OFFTIME</pre>

Before	After
<pre> LDAA #'Z' LOOP JSR OUTA       BRA LOOP </pre>	<pre> LOOP       JSR INPUT       TSTA       BEQ LOOP        ANDA #\$0F       TAB ; B HAS INDEX        LDX #ONTIMETBL       ABX       ABX        LDY #OFFTIMETBL       ABY       ABY        LDD 0,X       STD ONTIME       LDD 0,Y       STD OFFTIME        BRA LOOP </pre>

### 3.3 Exercises

1. Make the changes shown above and run the program. Connect PA4 to an oscilloscope. Press any of the keys 0 to 9 and verify that the frequency changes.
2. Change the program so that when you press any of the keys 0 to 9, the frequency should be fixed at 100 Hz but the duty cycle changes. Pick 10 different duty cycles. Also, connect the output pin to a digital voltmeter. How does the voltage change with the duty cycle?