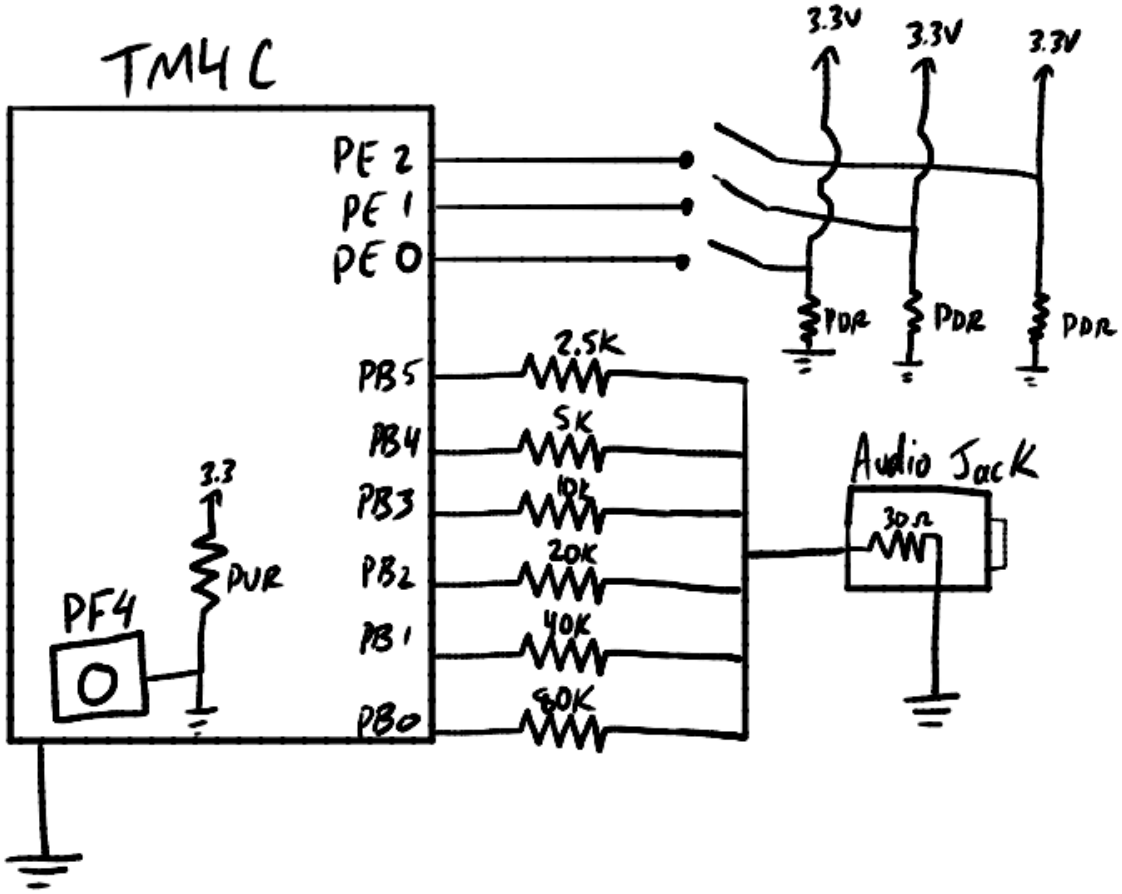
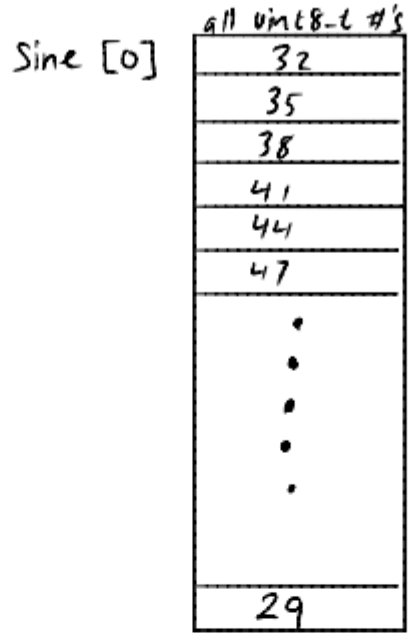
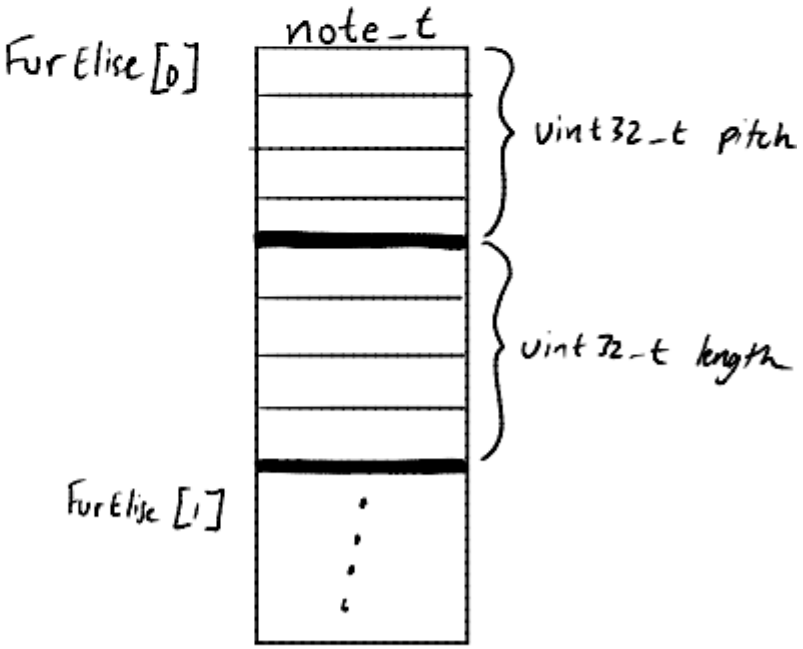
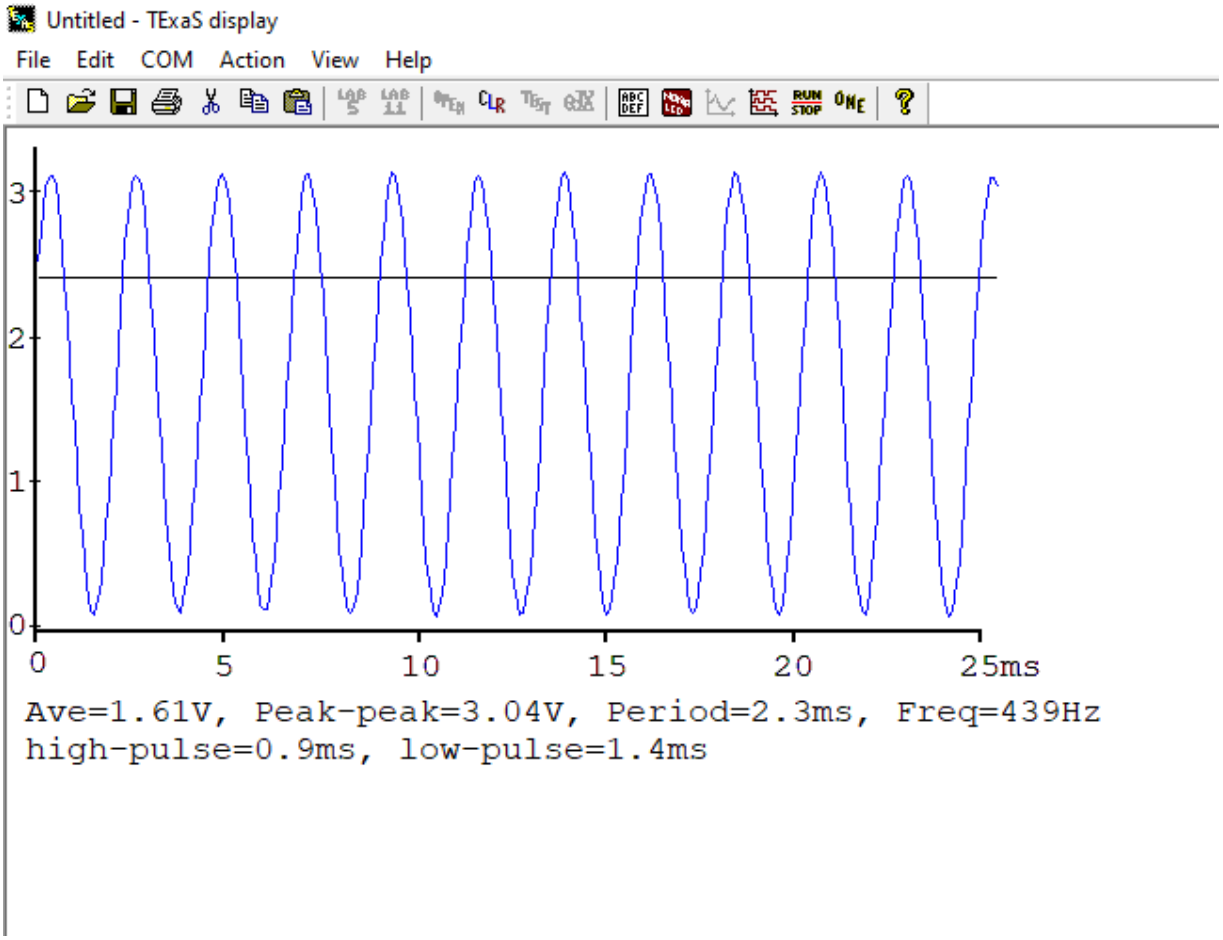

TM4C
PE2
PE1
PE0
PB5
PB4
PB3
PB2
PB1
PB0
3.3V
3.3V
3.3V
PDR
PDR
PDR
2.5K
5K
10K
20K
40K
80K
Audio Jack
30Ω
3.3
PUR
PF4

Data Structures







	Theoretical	Measured		Percent Error		Precision	Range	Resolution
0	0					64	3.3	0.052381
1	0.052380952	0.056		7%				
7	0.366666667	0.379		3%				
8	0.419047619	0.432		3%				
15	0.785714286	0.81		3%				
16	0.838095238	0.848		1%				
17	0.89047619	0.904		2%				
18	0.942857143	0.955		1%				
31	1.623809524	1.654		2%				
32	1.676190476	1.651		2%				
33	1.728571429	1.704		1%				
47	2.461904762	2.453		0%				
48	2.514285714	2.491		1%				
49	2.566666667	2.546		1%				
62	3.247619048	3.24		0%				
63	3.3	3.294		0%				

When does the interrupt trigger occur?

In SysTick: when the CURRENT register counts from 1->0

In which file is the interrupt vector?

Startup.s

List the steps that occur after the trigger occurs and before the processor executes the handler.

1) Finish the current instruction

2) Push 8 registers on the stack (PSW,PC,LR,R12,R3,R2,R1,R0, with R0 on top)

3) Set LR to 0xFFFFFFF9

4) Set IPSR to the interrupt number

5) Set PC to the vector

It looks like **BX LR** instruction simply moves LR into PC, how does this return from interrupt?

The address 0xFFFFFFF9 is specifically for interrupts, so the program knows to pop the 8 registers we pushed from the stack, which loads PC with the main program.