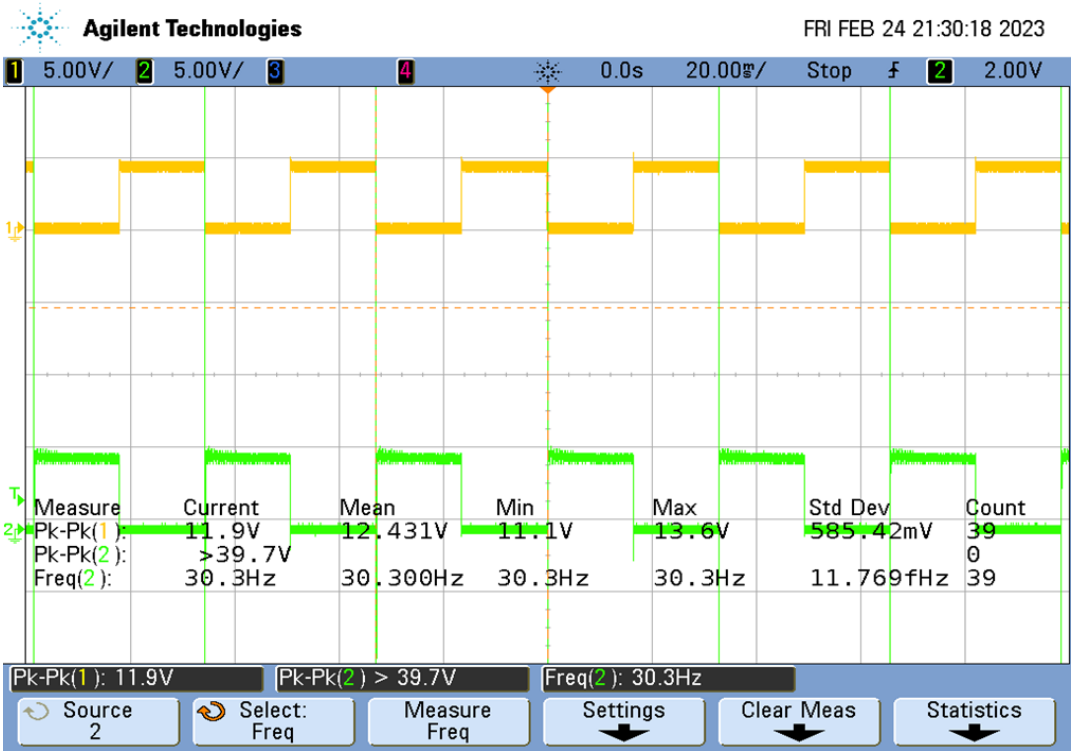
Lab 3: Theremin

ESE3500: Embedded Systems & Microcontroller Laboratory
University of Pennsylvania

In this document, you'll fill out your responses to the questions listed in the Lab 3 Manual. Please fill out your name and link your Github repository below to begin. Be sure that your code on the repo is up-to-date before submission!

Student Name: Erica Feehery
Pennkey: efeehery
GitHub Repository: https://github.com/ese3500/lab-3-efeehery

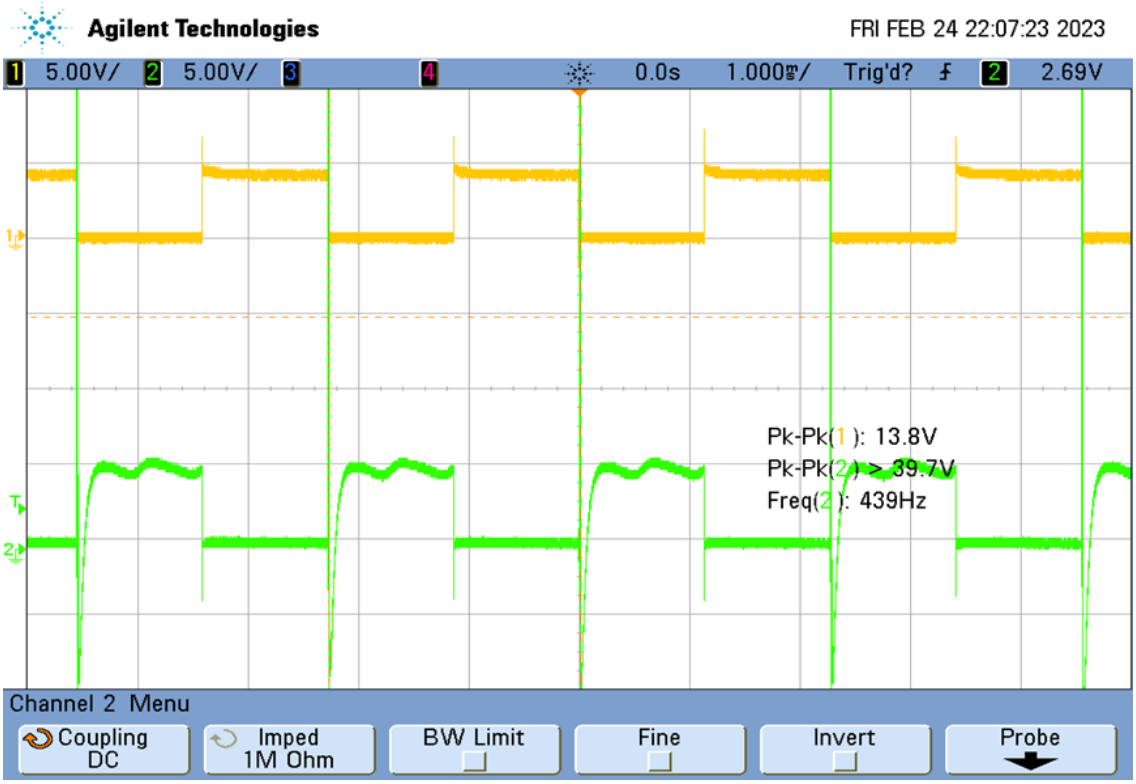
1. The clock sas a frequency of 16 MHz. The timer is prescaled by 1024. Timer 0 has an 8 bit counter, so it will overflow after 256 ticks.
    a. $\frac{16*10^6 ticks}{s} * \frac{1\ tick}{1024\ ticks} * \frac{1\ overflow}{256\ ticks} = 61.035 overflows/s$
    b. Each overflow toggles the output so one cycle is 2 overflows.
    c. The signal has a frequency of **30.5Hz**
2. Yellow is PD6 and green is attached to where the PN2222 and buzzer connect. The measured frequency of 30.3Hz is very close to the calculated value from (1).



3. Yes, the timer was prescaled by a factor of 256, but the system clock was not prescaled.
4. $OCR0A = 71 = 16 * 10^6/256/(440 * 2)$
5.

```
void Initialize() {
	cli(); //disable global interrupts for setup
	//PB1 pin 6 - buzzer output
	DDRD |= (1<<DDD6); //set to output pin
	PORTD |= (1<<PORTD6); // set to low
	//timer 0 setup
	TCCR0B &= ~(1<<CS00);
	TCCR0B &= ~(1<<CS01);
	TCCR0B |= (1<<CS02);
	//sets prescaler to 256
	//16*10^6 / 256 / (440 * 2) = 71
	TIMSK0 |= (1<<OCIE0A);//enables output compare
	OCR0A = 71;
	sei();//enable global interrupts
}
ISR(TIMER0_COMPA_vect) {
	PORTD ^= (1<<PORTD6);
	OCR0A = (OCR0A + 71) % 256;
}
```

6. Yellow is PD6 and green is attached to where the PN2222 and buzzer connect.

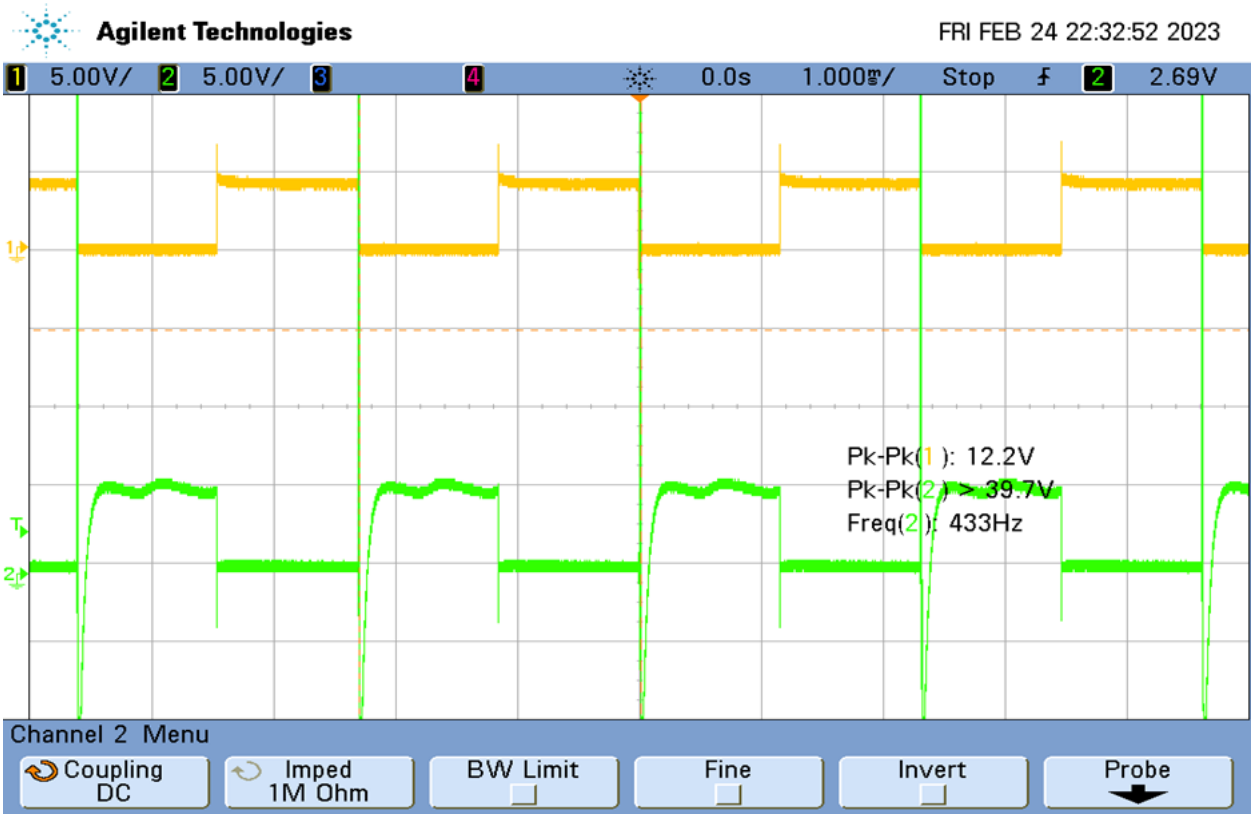


7. $OCR0A = 71 = 16 * 10^6/256/(440 * 2)$, same reasoning as for (4)

8.

```
void Initialize() {
	cli(); //disable global interrupts for setup
	DDRD |= (1<<DDD6); //set buzzer output to output pin (PD6 = OC0A)
	PORTD |= (1<<PORTD6); // set initially to low
	//timer 0 setup, sets pre scaler to 256
	TCCR0B &= ~(1<<CS00);
	TCCR0B &= ~(1<<CS01);
	TCCR0B |= (1<<CS02);
	//set to CTC
	TCCR0A &= ~(1<<WGM00);
	TCCR0A |= (1<<WGM01);
	TCCR0B &= ~(1<<WGM02);
	//toggle OC1A on compare match
	TCCR0A |= (1<<COM0A0);
	TCCR0A &= ~(1<<COM0A1);
	OCR0A = 71;//16*10^6 / 256 / (440 * 2) = 71
	sei();//enable global interrupts
}
```

9. Yellow is PD6 and green is attached to where the PN2222 and buzzer connect.



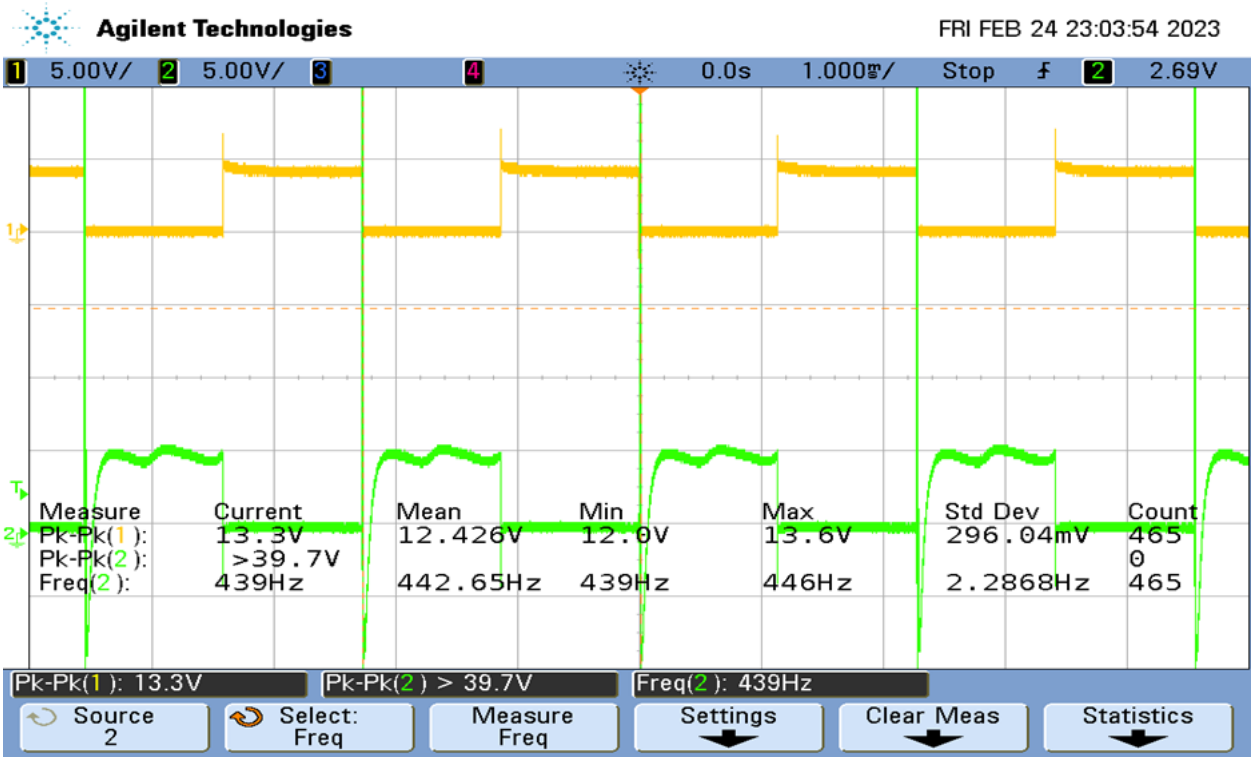
10. I used the Phase-Correct PWM mode that sets the TOP value to OCR0A. The frequency needs to be doubled again because the up/down of the phase-correct halves the frequency. In this code, the timer prescaler is 64.

    $TOP = OCR0A = 142 = 16 * 10^6/64/(440 * 2 * 2)$

11.

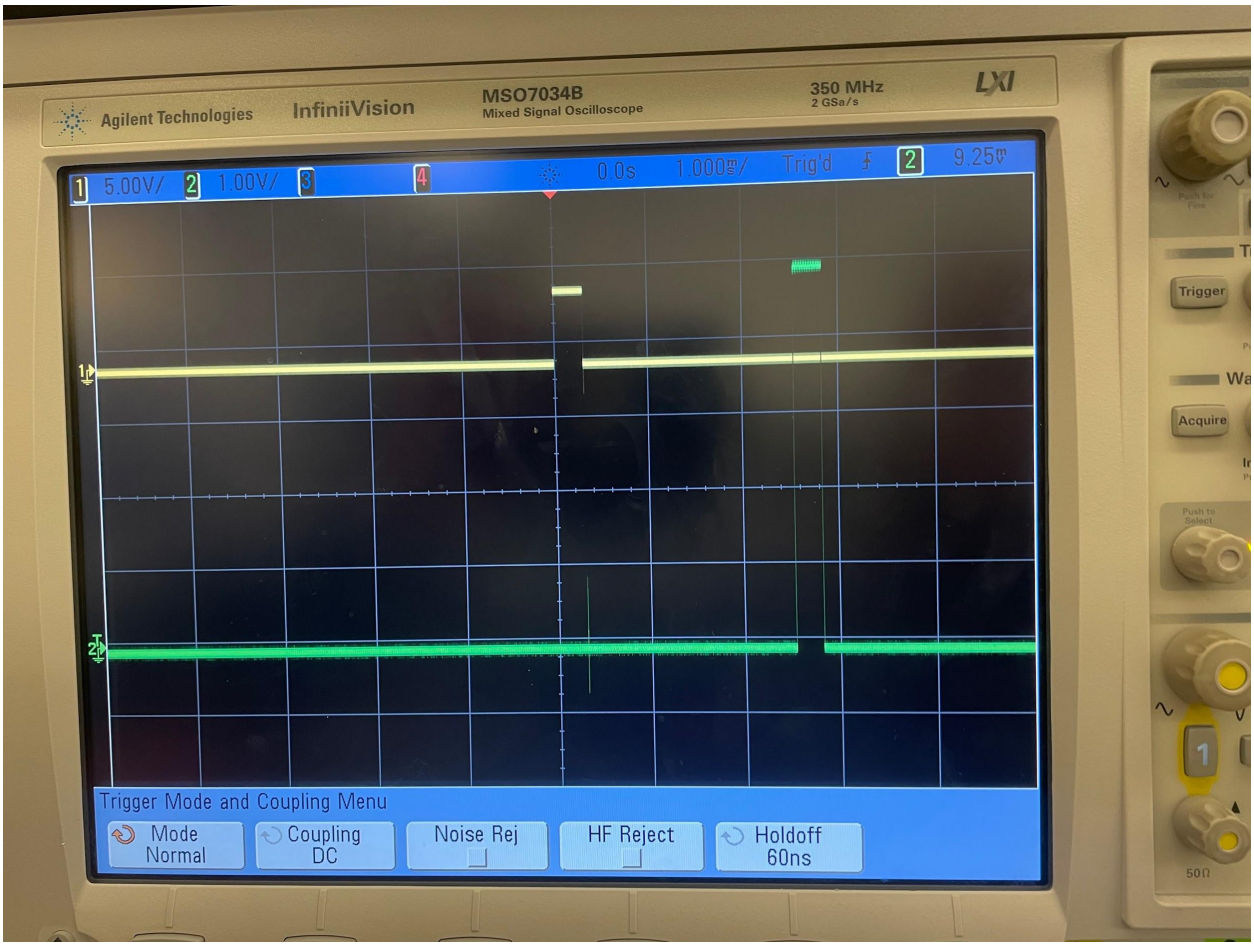
```
void Initialize() {
	cli(); //disable global interrupts for setup

	DDRD |= (1<<DDD6); //set buzzer to output pin (PD6 =
OC0A)
	PORTD |= (1<<PORTD6); // set initially to low

	//timer 0 setup, sets pre scaler to 64
	TCCR0B |= (1<<CS00);
	TCCR0B |= (1<<CS01);
	TCCR0B &= ~(1<<CS02);

	//set to phase correct PWM with settable TOP
	TCCR0A |= (1<<WGM00);
	TCCR0A &= ~(1<<WGM01);
	TCCR0B |= (1<<WGM02);

	//toggle OC1A on compare match
	TCCR0A |= (1<<COM0A0);
	TCCR0A &= ~(1<<COM0A1);

	OCR0A = 142;//16*10^6 / 64 / (440*2*2) = 142
	sei();//enable global interrupts
}
```

12. Yellow is PD6 and green is attached to where the PN2222 and buzzer connect.



13. The pulse that needs to be supplied for at least 10 us in order to start the ranging.
14. The Trig pin is used by the microcontroller to send a “trigger” pulse to initiate a distance measurement. The Echo pin is used for the device to return its distance measurement, where the width of the pulse is proportional to the measured distance.
15. The largest distance observed was 74 cm.
16. Yellow is the trigger pin and green in the echo pin.

17. The shortest distance observed was 4cm.
18. Yellow is the trigger pin and green in the echo pin.



19. Timer frequency: 250000 Hz; Waveform generation mode: phase correct PWM, and Output compare mode: toggle on compare match

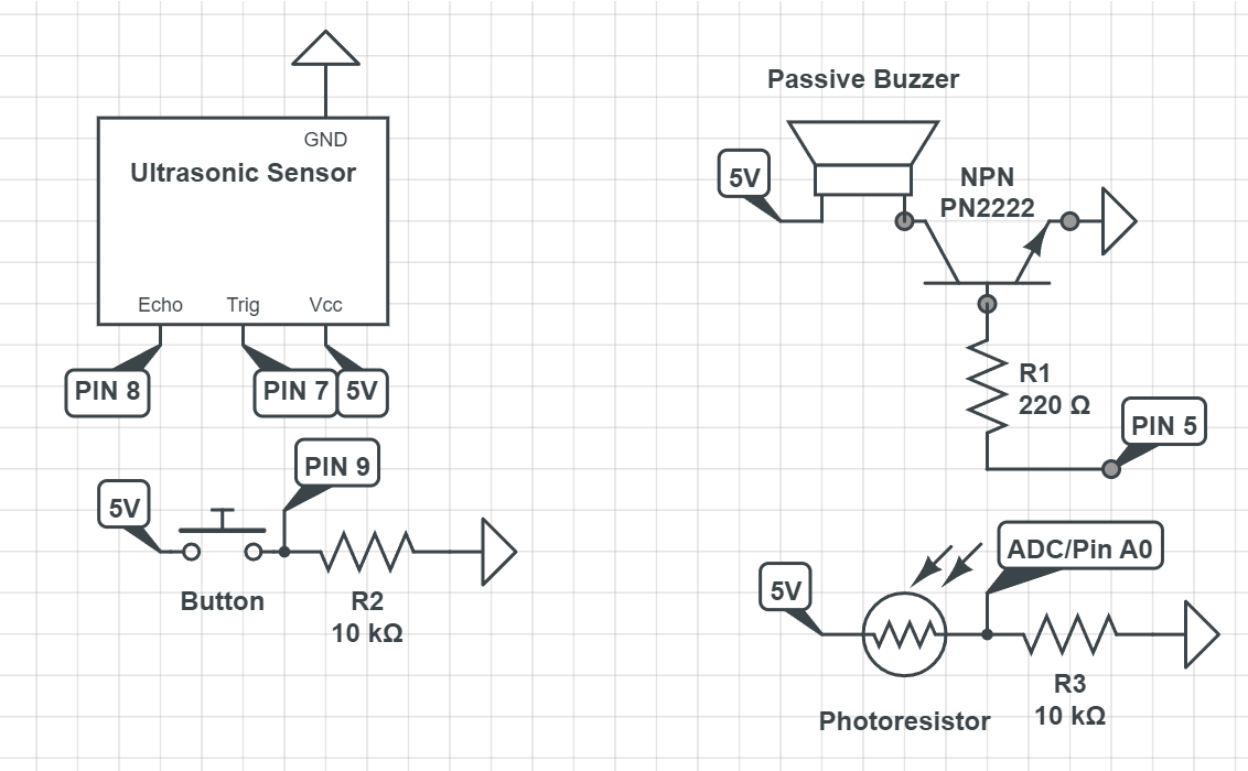
Note	**C6**	**D6**	**E6**	**F6**	**G6**	**A6**	**B6**	**C7**
Freq (Hz)	1046	1174	1318	1397	1568	1760	1975	2093
OCR0A	60	53	47	45	40	35	32	30

20. Equation: **OCR0A = DISTANCE * 0.4285 + 31.71**
    a. (d-dist1)(O2-O1)/(dist2-dist1)+O1) = OCR0A
    b. (d-4)(30)/(70)+30) = OCR0A
    c. OCR0A = (3/7)d+31.71
21. Maximum ADC Value Read: 985; Minimum ADC Value Read: 240
22.

ADC Ranges	Duty Cycle

240 (and below) - 314	5%
315-389	10%
390-464	15%
465-539	20%
540-614	25%
615-689	30%
690-764	35%
765-839	40%
840-914	45%
915-985 (and above)	50%

23.



24. Videos:
    a. Other View: https://youtu.be/b4x0WWVy40I
    b. First View: https://youtu.be/U2AhW7-eURo
    c. General Info:

i. The volume control is done in the discrete manner described in part e.

25. There is a resistor at the base of the transistor because a BJT uses the current at the base to control current through the emitter, and a resistor will limit the amount of current into the BJT's base and thus the amount of output current, as to perform within power limitations and to not damage other circuit elements like the buzzer
26.
27.
28. Link to code in github and video on youtube:
    a. https://github.com/ese3500/lab-3-efeehery/blob/main/song_ec.c
    b. https://youtu.be/7SJH3CdQf3I
29.
30.