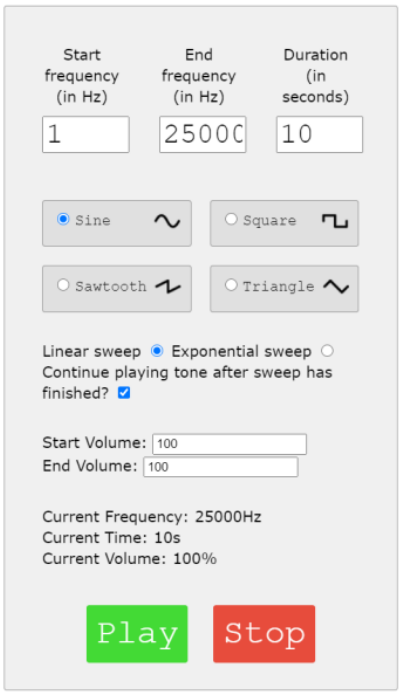
Thomas Smallarz ECE444 Homework 4

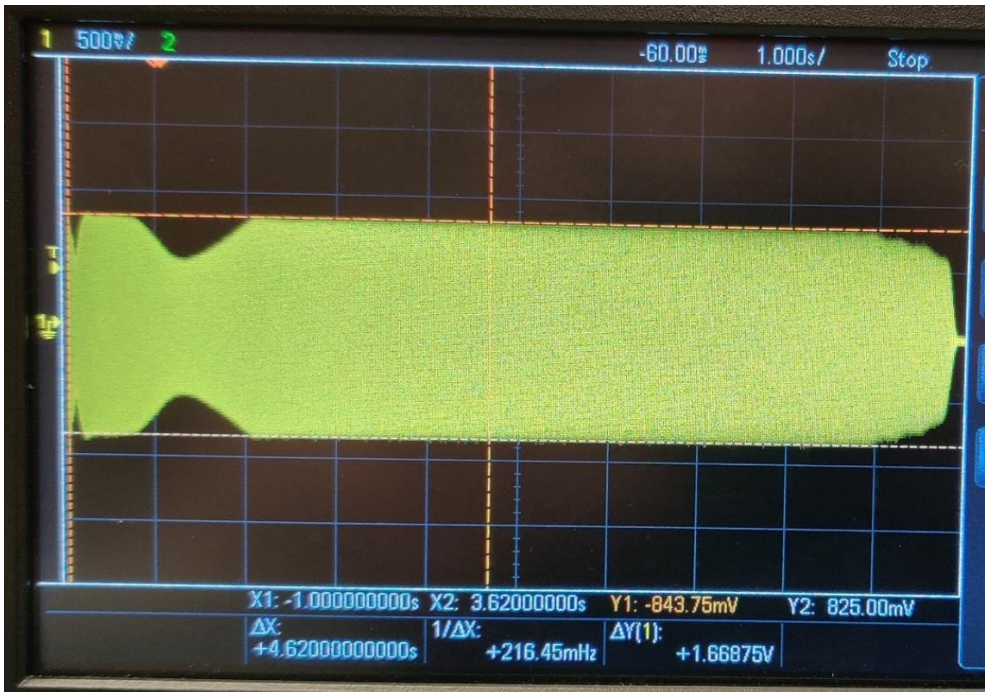
Mini Hardware Problem: Audio Pass-through

First, need to know what computer outputs through 3.5mm headphone jack. Using an online tone generator website and connecting a 3.5mm jack to my laptop with the two audio ends stripped down to copper:

https://onlinetonegenerator.com/frequency-sweep-generator.html

Generating a sweep from 1Hz to 25,000 over 10 seconds at 100% volume



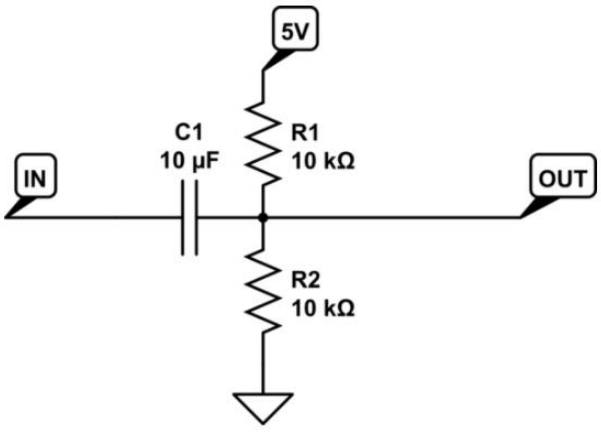


My laptop output is ~1.8Vpp with ~+-0.9V at most frequencies (dips down in lower frequencies). This is with “Windows Sonic for Headphones” turned off in my Windows sound settings.

I googled “bipolar audio signal to unipolar adc” and found a Stack Exchange post asking how to convert +-3.5V to 0→5V:

https://electronics.stackexchange.com/questions/413058/bipolar-to-unipolar-voltage-converter

A guy answered the question with this solution:

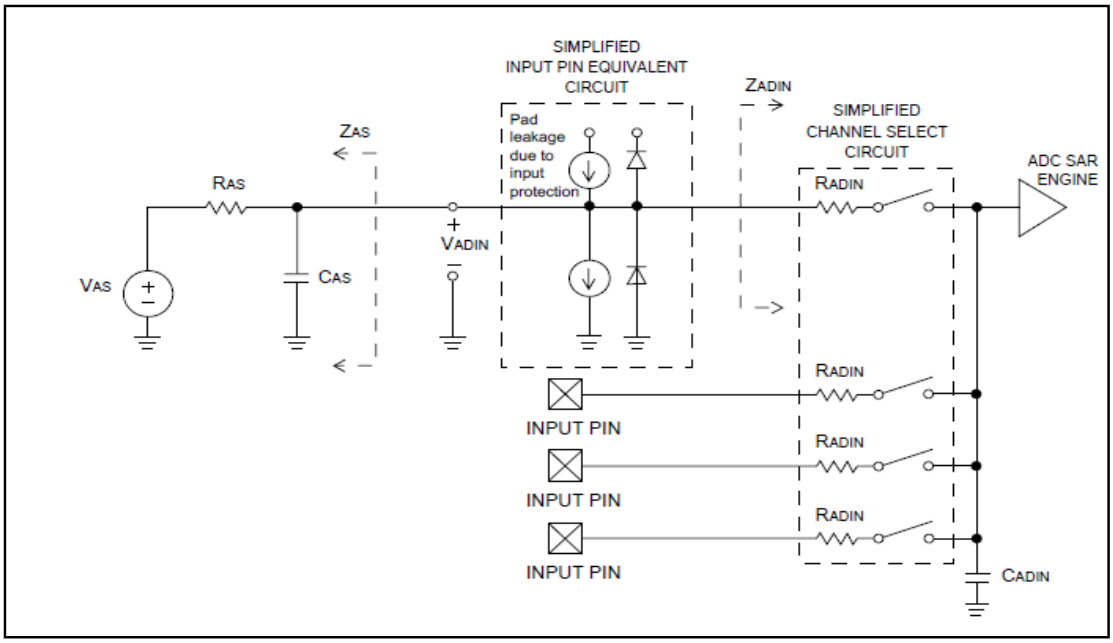


Using the Kinetis K22F 512KB Flash Technical Data Sheet I found the ADC electrical specifications

3.6.1.1 16-bit ADC operating conditions

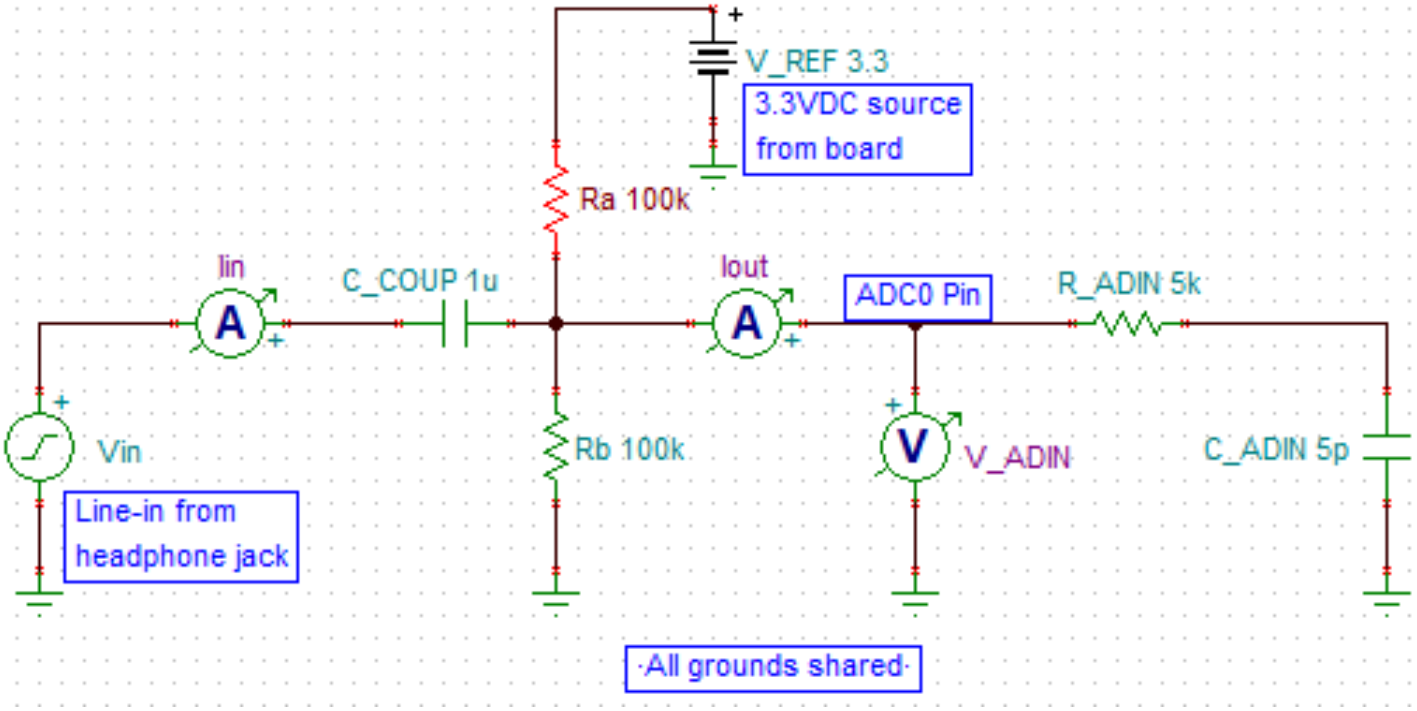
Table 29. 16-bit ADC operating conditions

Symbol	Description	Conditions	Min.	Typ.[1]	Max.	Unit	Notes
V_{DDA}	Supply voltage	Absolute	1.71	—	3.6	V	
ΔV_{DDA}	Supply voltage	Delta to V_{DD} (V_{DD} – V_{DDA})	-100	0	+100	mV	2
ΔV_{SSA}	Ground voltage	Delta to V_{SS} (V_{SS} – V_{SSA})	-100	0	+100	mV	2
V_{REFH}	ADC reference voltage high		1.13	V_{DDA}	V_{DDA}	V	
V_{REFL}	ADC reference voltage low		V_{SSA}	V_{SSA}	V_{SSA}	V	
V_{ADIN}	Input voltage	• 16-bit differential mode • All other modes	VREFL VREFL	— —	31/32 * VREFH VREFH	V	
C_{ADIN}	Input capacitance	• 16-bit mode • 8-bit / 10-bit / 12-bit modes	— —	8 4	10 5	pF	
R_{ADIN}	Input series resistance		—	2	5	kΩ	
R_{AS}	Analog source resistance (external)	13-bit / 12-bit modes f_{ADCK} < 4 MHz	—	—	5	kΩ	3
f_{ADCK}	ADC conversion clock frequency	≤ 13-bit mode	1.0	—	24.0	MHz	4
f_{ADCK}	ADC conversion clock frequency	16-bit mode	2.0	—	12.0	MHz	4
C_{rate}	ADC conversion rate	≤ 13-bit modes No ADC hardware averaging Continuous conversions enabled, subsequent conversion time	20	—	1200	Ksps	5
C_{rate}	ADC conversion rate	16-bit mode No ADC hardware averaging	37	—	461	Ksps	5

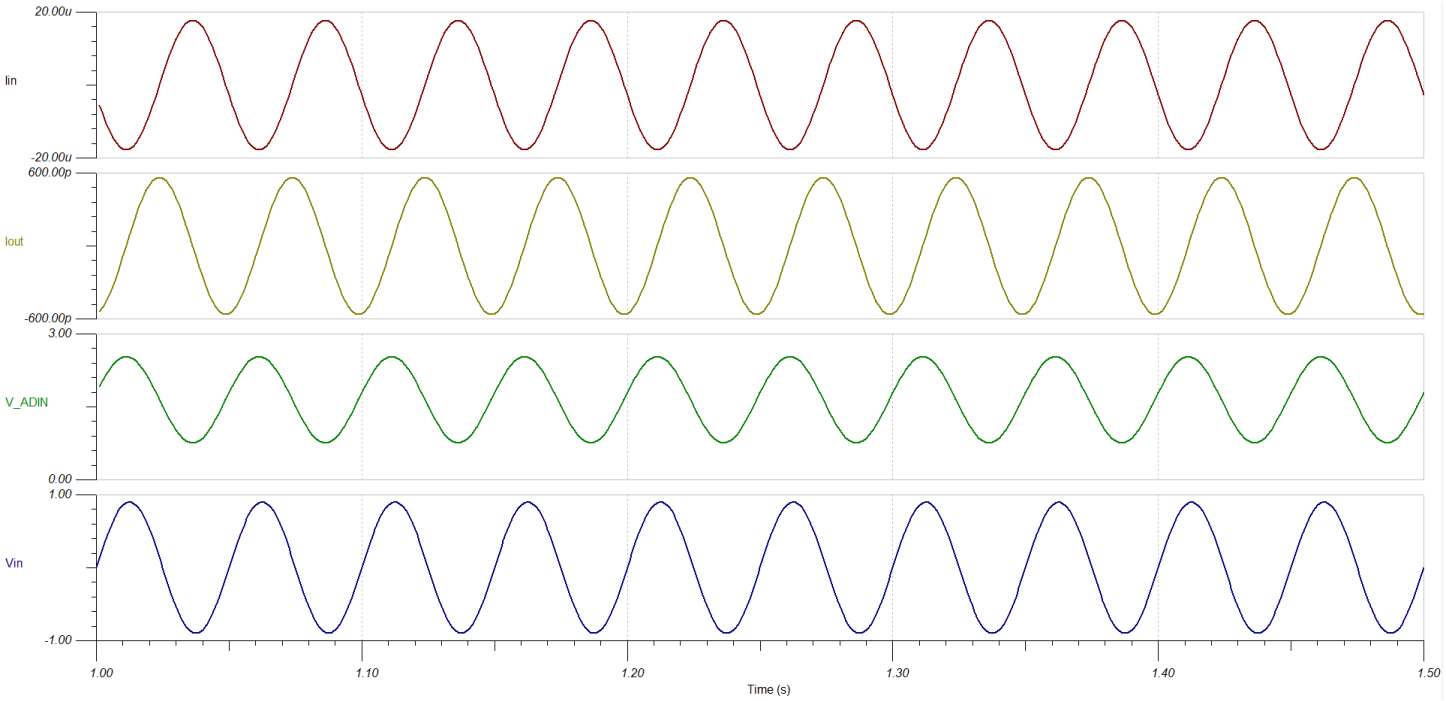


Thomas Smallarz ECE444 Homework 4

Used “Tina-TI” software to simulate circuit with the input voltage being 20Hz 1Vpp Sine wave, 20,000Hz 1Vpp Sine wave, and example .wav file of jazz music included in Tina-TI software:

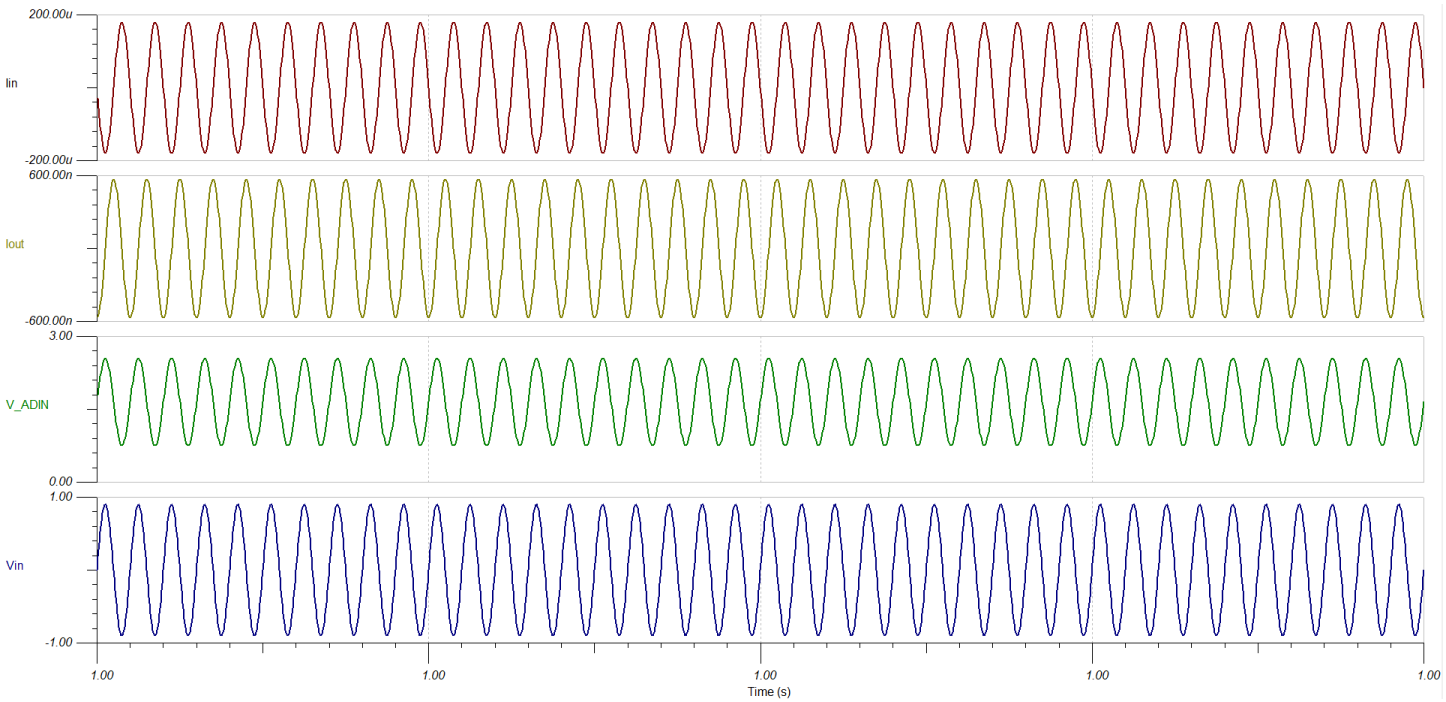


20Hz Sine wave with 1.8Vpp (max = 0.9V)

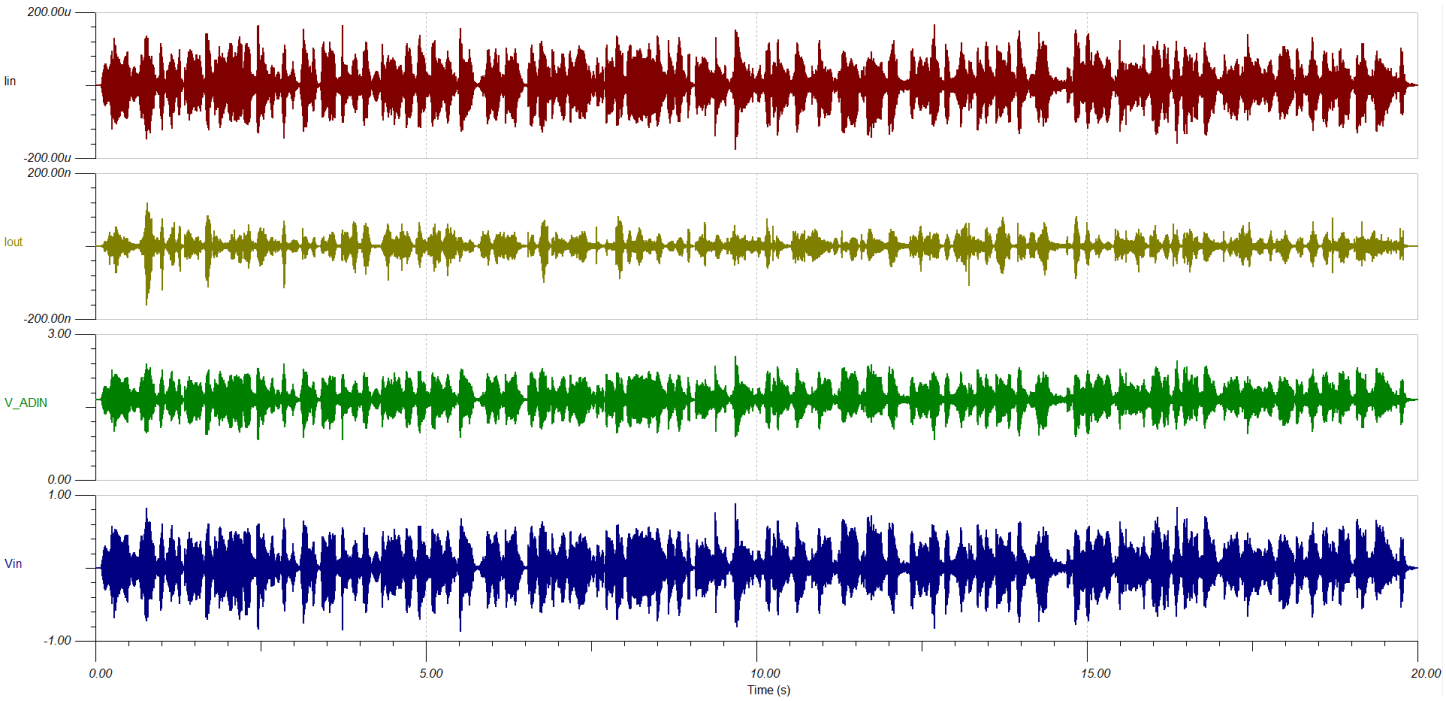


Thomas Smallarz ECE444 Homework 4

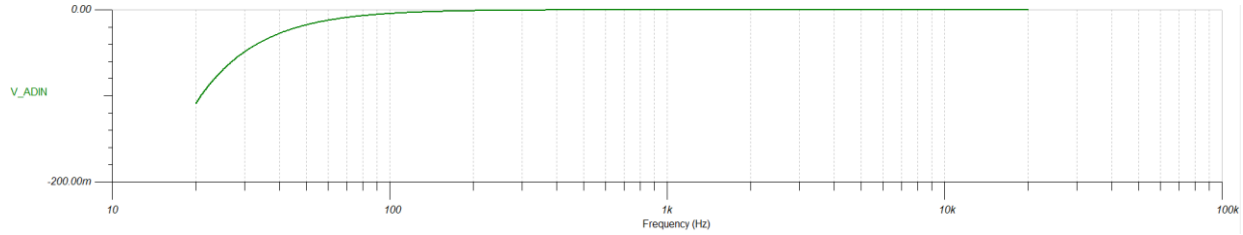
20,000Hz Sine wave with 1.8Vpp (max = 0.9V)



Example .wav file included with Tina-TI software of someone rapping with 1.8Vpp (max of 0.9V)



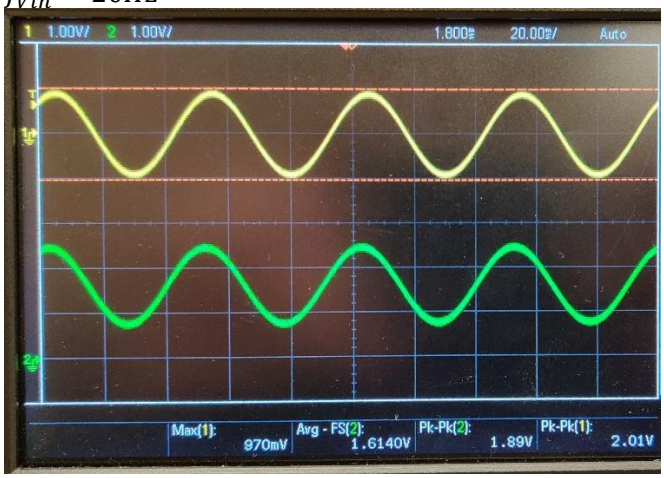
From these simulations we can see that at lower frequencies, the output amplitude is slightly (~-100mdB) smaller in amplitude, but we have an offset of 1.65VDC which is what matters.. We can see this dip in output amplitude by doing an AC sweep from 20Hz→20kHz



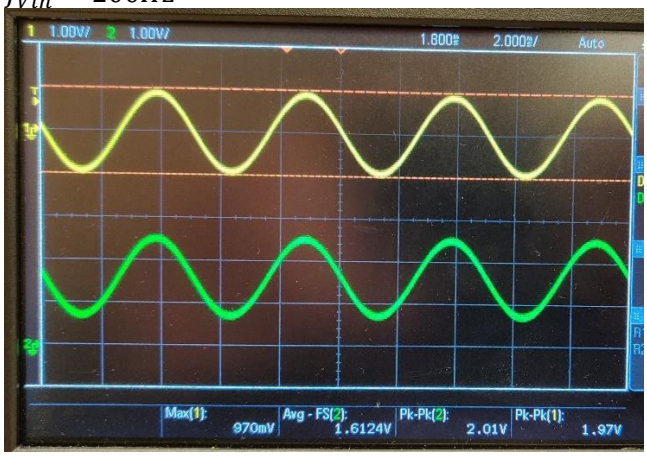
Build in lab and use waveform generator in place of laptop for input.

For all scope shots below 1 (Yellow) is audio input, 2 (Green) is output to ADC0

$f_{Vin} = 20Hz$

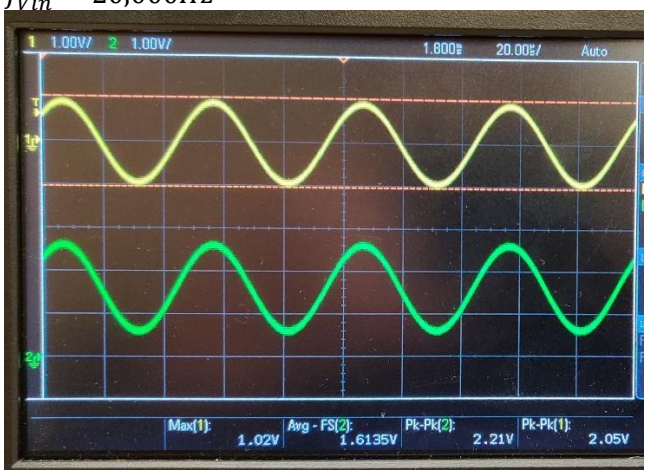


$f_{Vin} = 200Hz$



$f_{Vin} = 2{,}000Hz$

$f_{Vin} = 20{,}000Hz$



This looks like it is working correctly. Software for this is super easy:

```
void PIT0_IRQHandler(void){ //This function is called when the timer interrupt expires
  //Place Interrupt Service Routine Here
  ADC0->SC1[0] &= 0xE0;   //Start conversion of channel 1

  DAC0->DAT[0].DATL = DAC_DATL_DATA0(ADC0->R[0] & 0xFF);     //Set Lower 8 bits of Output
  DAC0->DAT[0].DATH = DAC_DATH_DATA1(ADC0->R[0] >> 0x8);     //Set Higher 8 bits of Output

  NVIC_ClearPendingIRQ(PIT0_IRQn);              //Clears interrupt flag in NVIC Register
  PIT->CHANNEL[0].TFLG  = PIT_TFLG_TIF_MASK;    //Clears interrupt flag in PIT Register
}
```