

# Monophonic Digital Bass Synth

Madeline Abio s5259572

9/10/2025

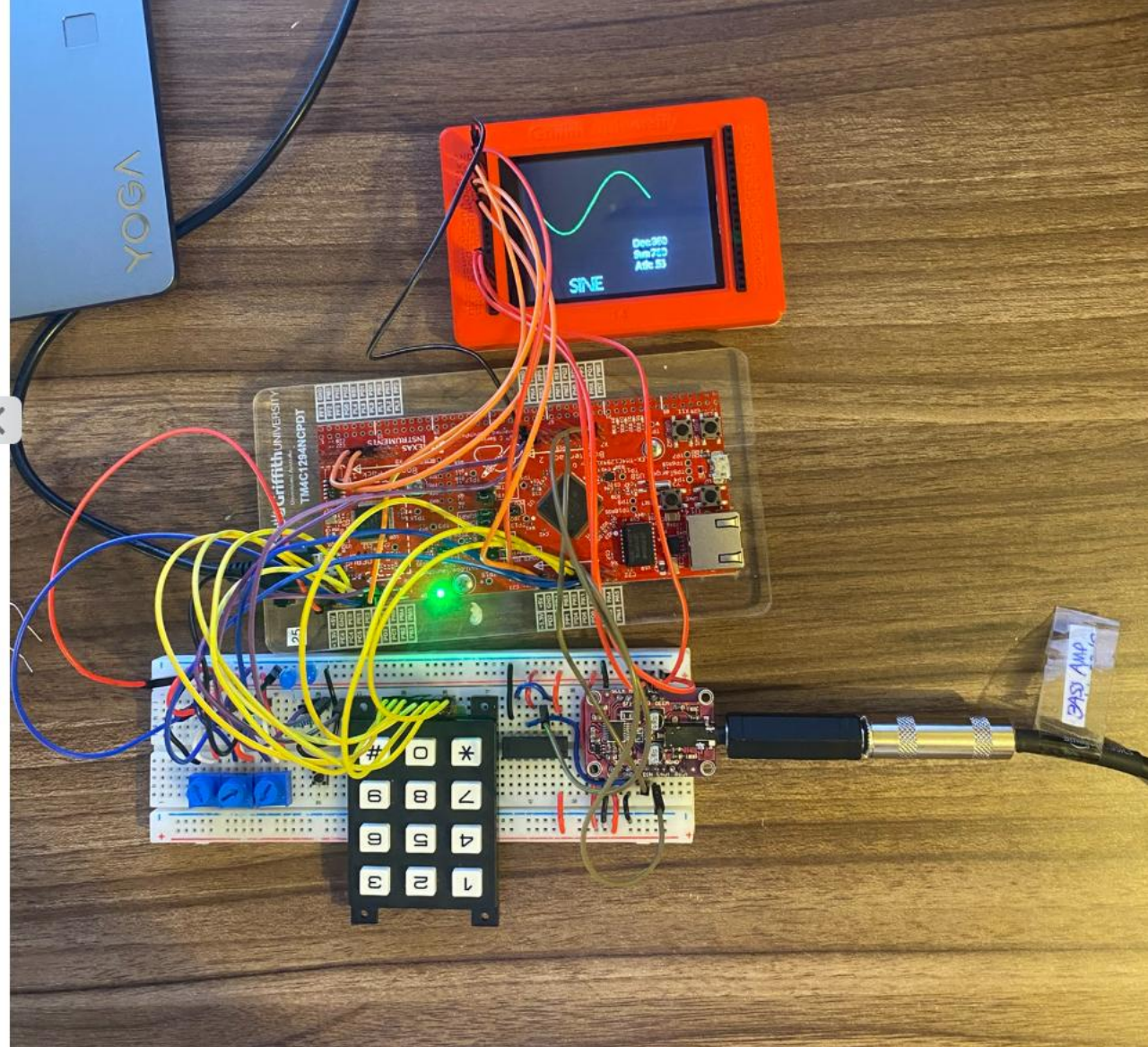
Embedded Systems  
(Student Defined Project)

# Introduction – Project Aim

- Design and implement a fully functioning waveform generator with adjustable envelope parameters, multiple waveform choices, and LCD visualization for the purpose of bass synthesis.

# Introduction - Design Requirements

- Monophonic digital synthesiser
  - 4 waveform modes, cycle via button
  - ADS (attack, decay, sustain) control via 3 potentiometers
  - 4x3 keypad for note selection
  - Audio output via I2S DAC
  - Waveform modes and parameters visualised on SPI display
- Audio amplifier is out of scope



# Design

- 1x I2S protocol implemented by configuring SSI3 and feeding Fss into a T-flipflop to act as I2S WSEL (UDA1334A breakout board)
  - 12-48kHz audio sample rate, currently set to 12kHz
- 1x uDMA configured to deliver the audio samples to SSI3 for DAC on SSI3 FIFO half empty (uses SSI3\_Handler interrupt on DMATXMIS)
  - Via Ping Pong buffer
  - Enables continuous audio transmission and minimises CPU overhead.
- 1x SPI display implemented by configuring SSI2 to display the current waveform mode and the ADS values.
- Digital function generator advancing through pre-calculated sample tables for sine, saw, square, triangle waves with note frequency configured by note->frequency table.
  - Values from this get sent to ping pong buffers as requested
- Digital envelope function to envelope wave samples in ADS modulation parameters.
- ADC0 configured with sequencer 0 to take measurements from the 3x potentiometers for ADS values.
  - Triggered by Timer0A
- 1x general purpose timer (Timer0A) configured in periodic mode. Triggers an interrupt every 4ms to check for numeric keypad updates, waveform mode updates, and potentiometer updates.
- 1x numeric matrix keypad set up via GPIO to act as a keyboard.
- 1x button configured with binary counter LEDs to cycle through waveform modes
- UART0 configured for debugging, but not much transmission during normal use as it can cause blocks.

# Testing – Methodology

- Most design components can be tested aurally. See the attached sound clips for recordings from the synth in different modes.
- The waveforms associated with each mode in the state it is in during the recording are displayed via an oscilloscope where the yellow channel shows the DAC output and the green channel shows the DIN (digital data in) signal.

# Test 1 – Clean Waveforms

- After the envelope settles, or if the sustain is low enough, the waveforms have a clean sound.
- The resulting waveforms are true to their name.
- The waveforms can be heard in the audio clip attached to this slide.

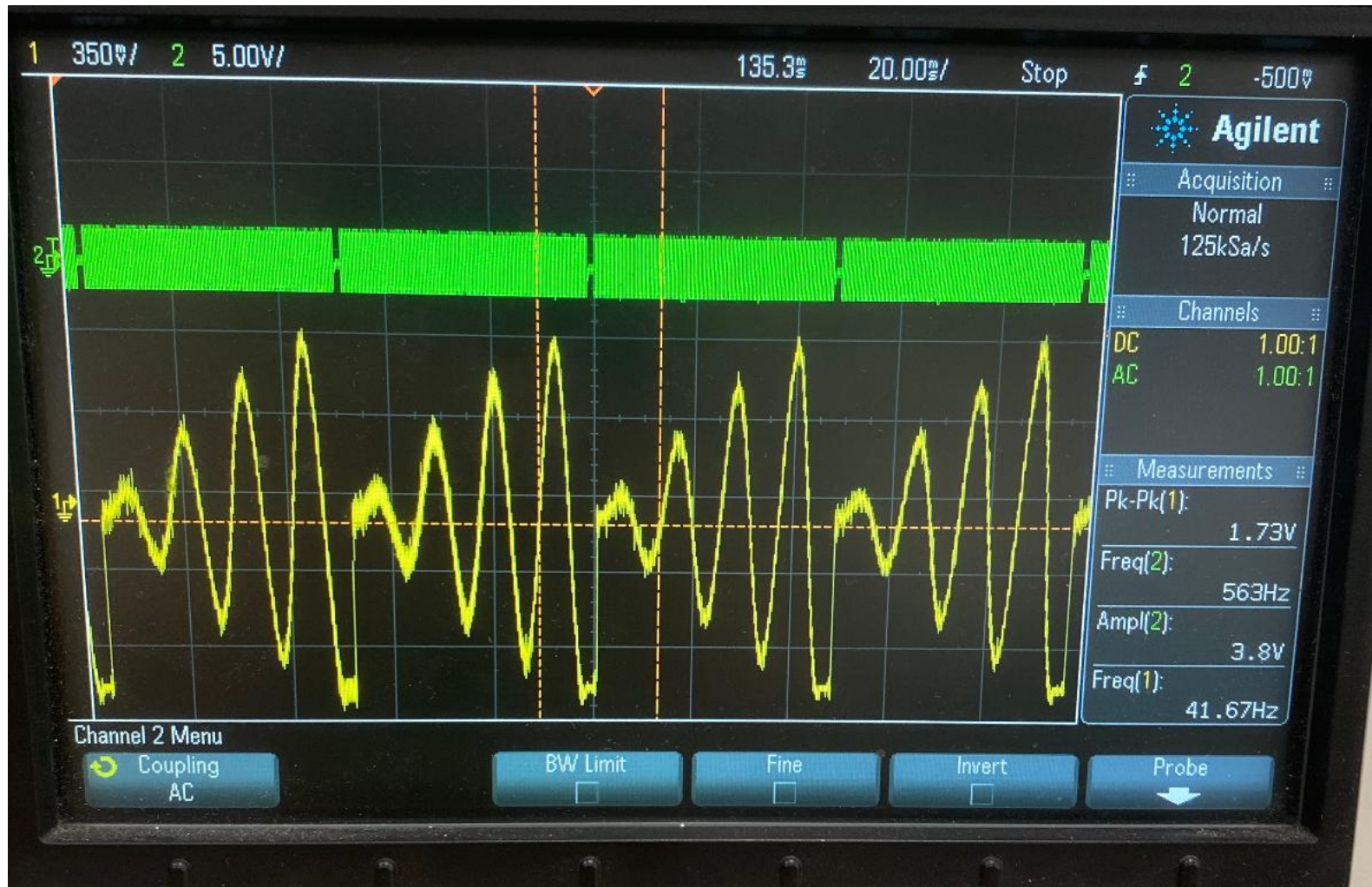
## **Order:**

1. Sine
2. Saw
3. Triangle
4. Square



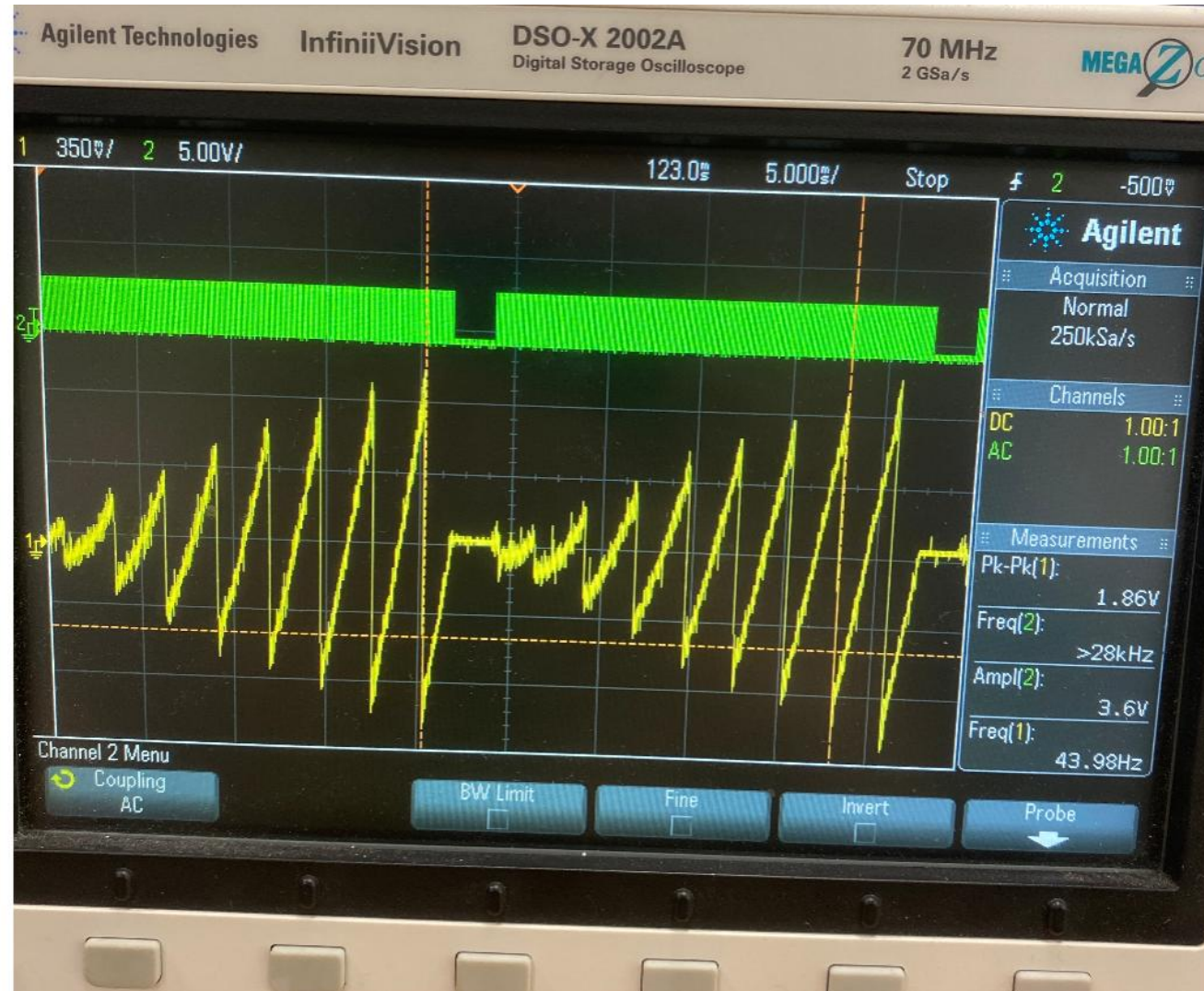


# Test 1 Waveforms – Sine

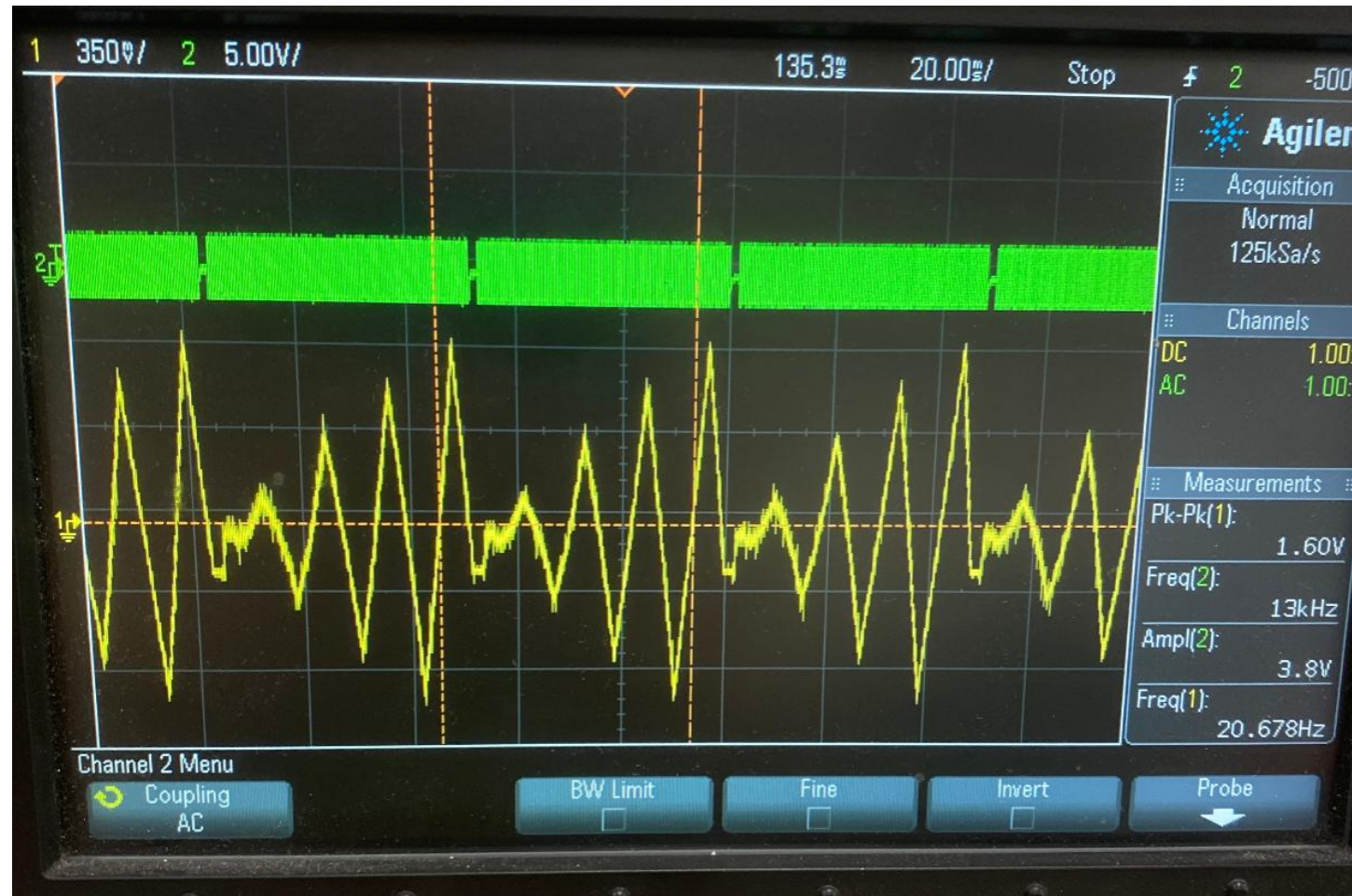




# Test 1 Waveforms - Saw



# Test 1 Waveforms - Triangle





# Test 1 Waveforms - Square



# Test 2 - Distortions

If the sustain is set high enough, distortions are introduced from the envelope. These can be heard here.

## **Audio Test**

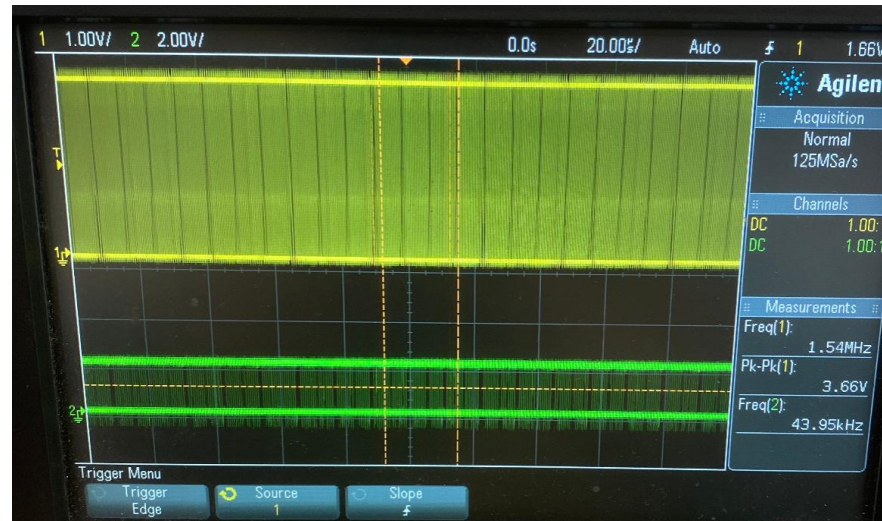
**(for waveform modes &  
ASR mod)**

1. Sine
2. Saw
3. Triangle
4. Square
5. Further ASR modulation



# Test 3I2S DAC

- Yellow signal = BCLK =  $16 * 2 * 48\text{kHz} \approx 1.54\text{MHz}$ ,
- Green signal = WSEL. Should be 48kHz, but is a bit less than 44kHz.
- Confirms that sample rate is essentially as expected.

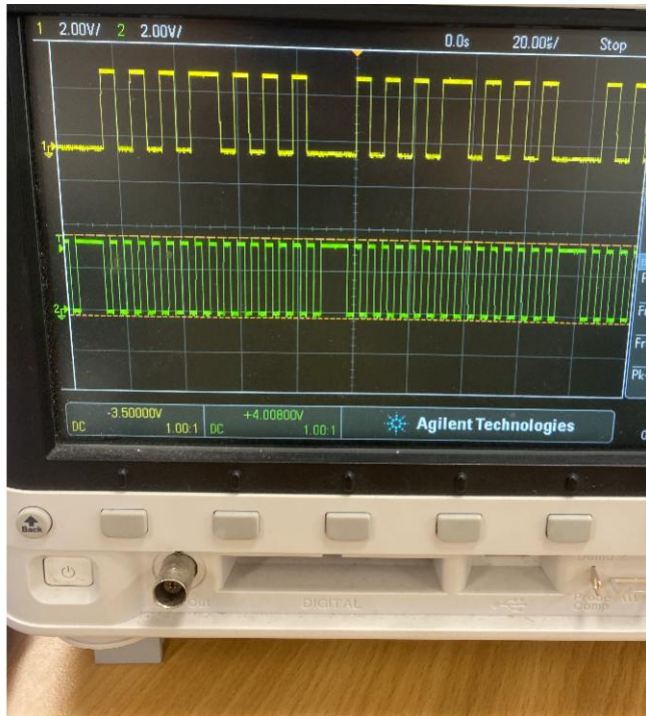


# Test 3 – I2S DAC

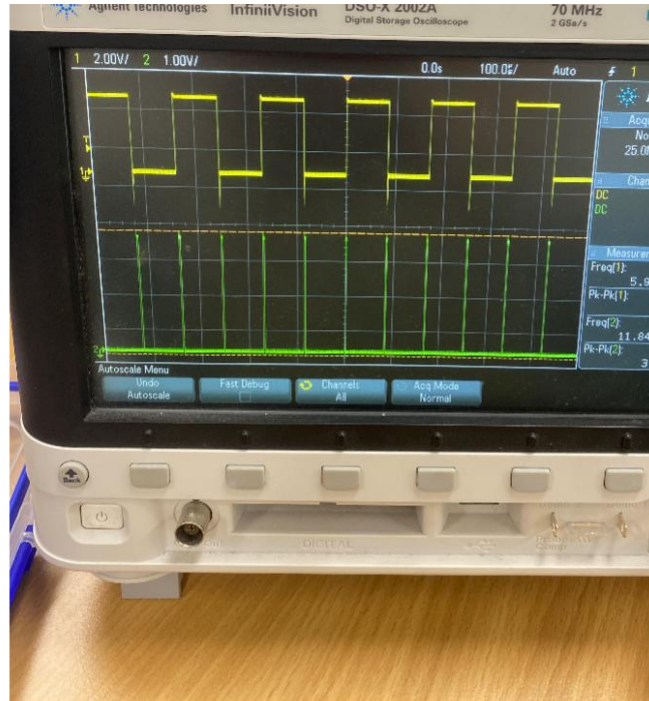
- The DAC I2S communication protocol requires three signals:
  - BCLK – clock signal determining the speed individual bits are pushed. This is also referred to as CLK here.
  - WSEL – a clock signal that flips at the end of each word to determine the channel.
    - This is also referred to as the T flip-flop (from a D-latch) circuit output here.
    - It operates by routing SPI's Fss value into a T flip flop.
  - DIN – the data being pushed in by the digital signal source. This is also referred to as TX here.
- The operation is confirmed in the following slide.



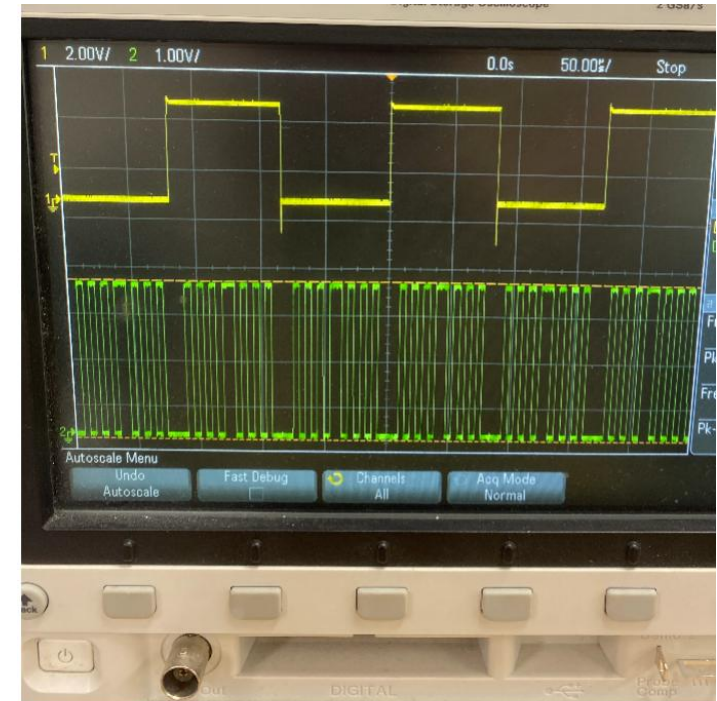
# Test 3 – I2S DAC



yellow = TX, green = CLK (snapshot)



green = Fss, yellow = output of clock circuit thinf with the Dlatc as a tf



green = tx, yellow = latch circuit output



# Test 5 - Display



# Testing – Other

- uDMA was tested via debugger by confirming the handler was being called and memory was being transferred into the ping pong buffers.
- Display was visually tested.
- Everything else can be tested with the oscilloscope and aurally.

# Conclusions



- Everything works as expected except the display has some glitches.
- Changing SSI3 CPSR and SCR had unexpected results at times. Subjectively the synth sounds a little better when they are sampling at 12kHz and the waveform tables are calculated at 48kHz of resolution. Is.
- There is a low frequency hum, potentially from power supply noise.
- There is a bug that happens when decay is set to 0 that makes the system need to reset. This should be fixed by adding a safety mechanism or analysing the underlying causes.
- Clean sounds can be challenging to get, potentially due to quantisation.
- The output seems to saturate at times causing distortion, this could be a resolution problem.
  - Maybe decreasing the max amplitude of the table values would allow for more headroom.
- Sounds very cool through a delay pedal.

# Improvements

- Add MIDI input over USB for external control.
- Expand to polyphonic synthesis and layering.
- Integrate a sequencer and/or arpeggiator.
- Improve grounding and shielding to reduce hum.
- Add EEPROM preset storage for saving user parameters.
- Add buttons to allow changing octaves without effecting code.
- Extend to 32 or 64 bit DAC for audio samples.
- Make ADC triggered by its own interrupt.
- Experiment with dynamic range/headroom by changing max amplitude of digital signal.
- Explore better DTSP algorithms.
- Improve display (fix bugs, make neater).
  - Add live waveform display to view effects of envelope on signal.