Arbitrary Waveform Generator

Nicholas Antoniades

2020

Table of Contents

Requirements

Attempting to find a black box solution that will allow the CRIO to control the input signal frequency and amplitude through some digital logic.

System Requirements:

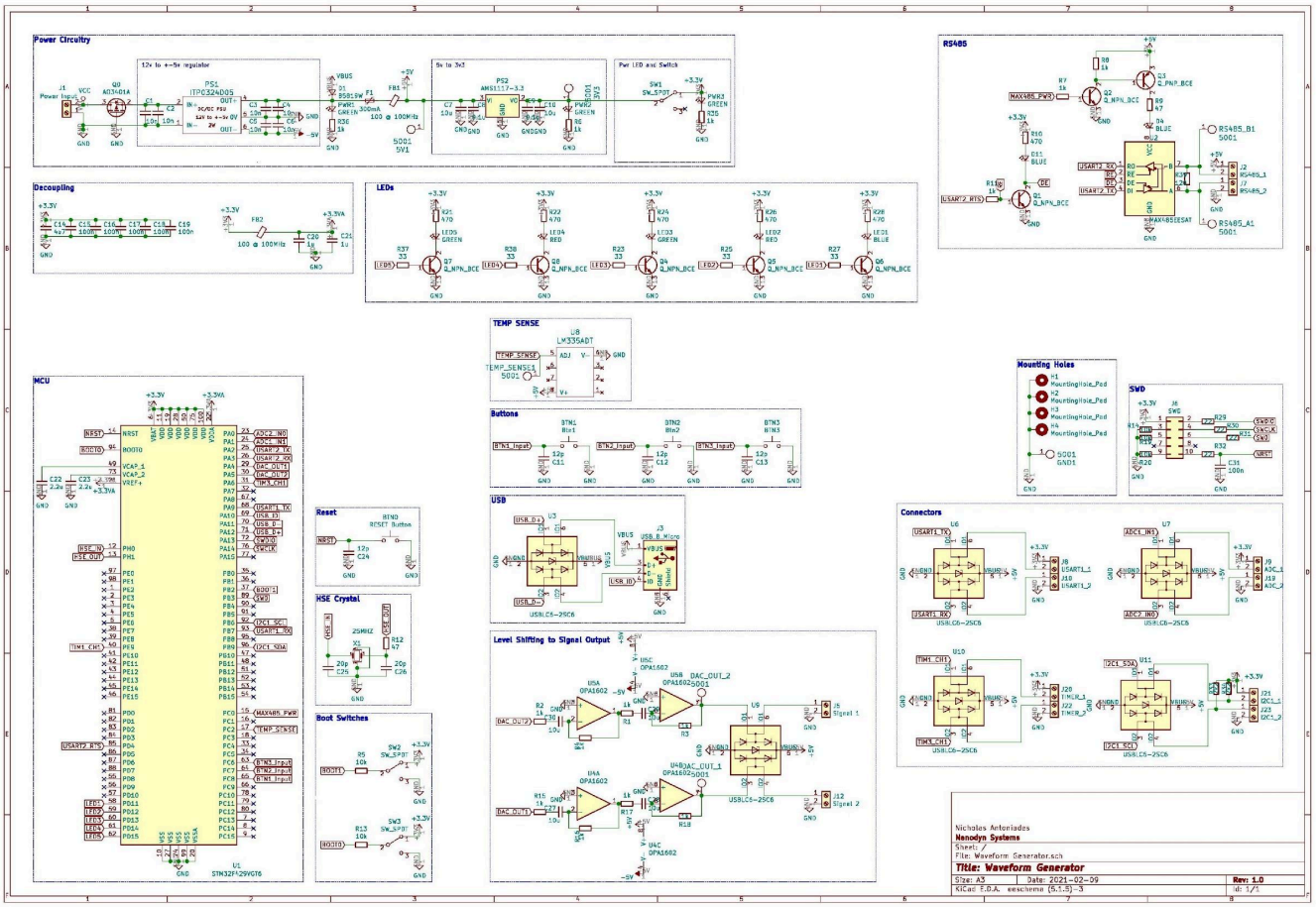
1. An output achieving at least a -2V to 2V peak to peak output signal.
2. 0-20kHz frequency range.
3. Programmable via TTL or other logic.

PCB Requirements:

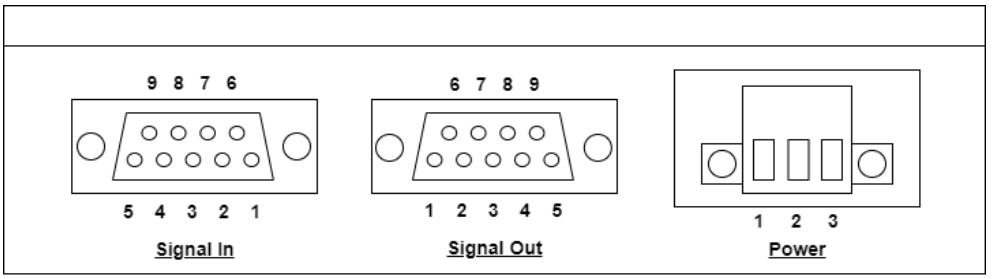
- USB DFU
- Power swich
- Level shifting circuitry
- DIP switch for boot mode settings
- DIP switch between power section and rest of circuit
- Power supply for +-5V
- Fuses
- Reset buttons
- LEDs
- ESD protection
- Reverse polarity protection
- SW with trace for debugging
- Temp sensor
- MAX485for RS485

Design

Schematics



AWG Enclosure wiring diagram

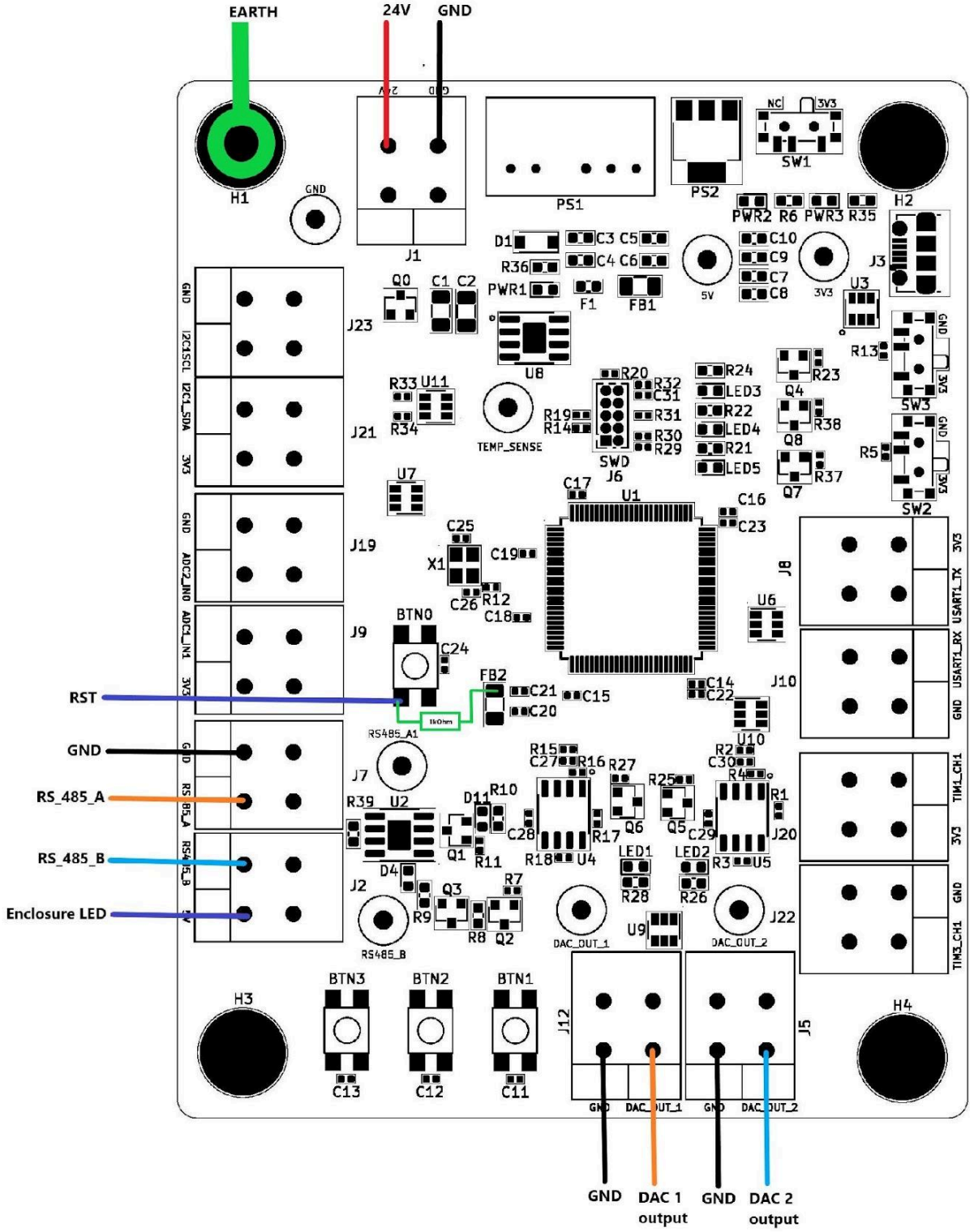


1 - RS485_A
2 - RS485_B
6 - GND
7 - RST
9 - GND

1 - Signal 1
2 - GND
3 - Signal 2
4 - GND

1 - 24v
2 - GND

PCB diagram



PCB Configuration

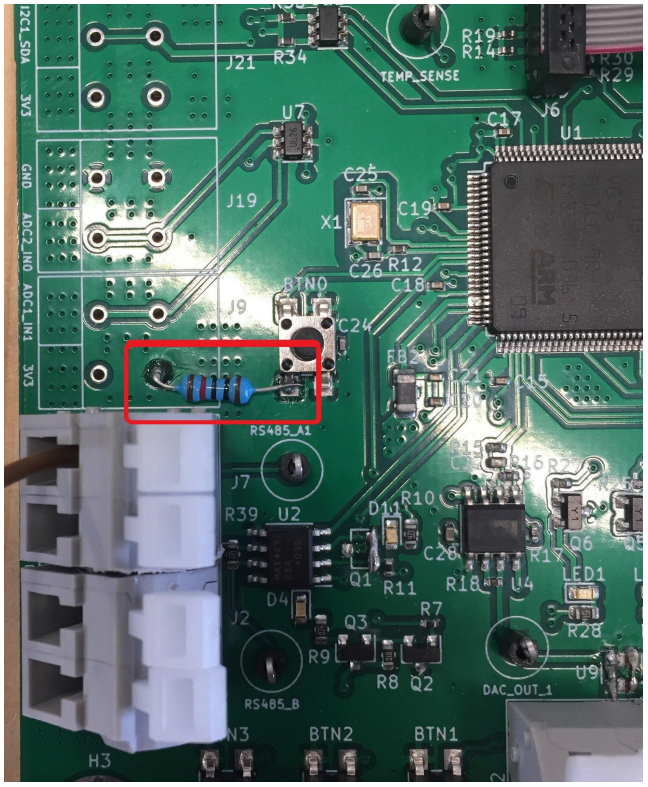
Steps for setting up the PCB:

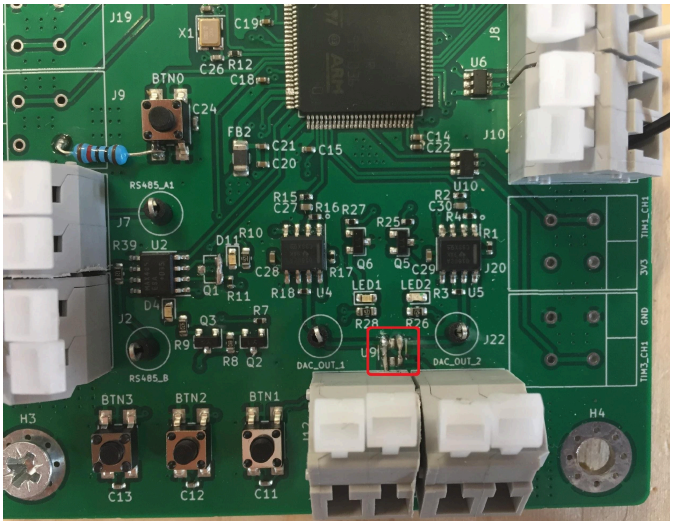
1. Add Fuse F1
2. Solder pull up resistor to NRST
3. Remove ESD protection U9. Bridge tracks as indicated
4. 10kOhm resistor was added as a pull-up resistor to bring the NRST pin to 3.3v
5. Removing LED D4 and bridging the pads as the voltage drop over the LED is such that the voltage over the MAX485 chip is too low.
6. Removing transistor Q1 and bridging its pads 1 and 2, then removing LED D11. This is so that DE and RE can be set at the same value. The original logic was so that they would be opposite values.

The 10kOhm pull-up resistor added can be seen in the figure below.

The ESD protection after the Op-Amp circuits were removed as the diode to the ground was clamping the circuit. Seen in the image below.

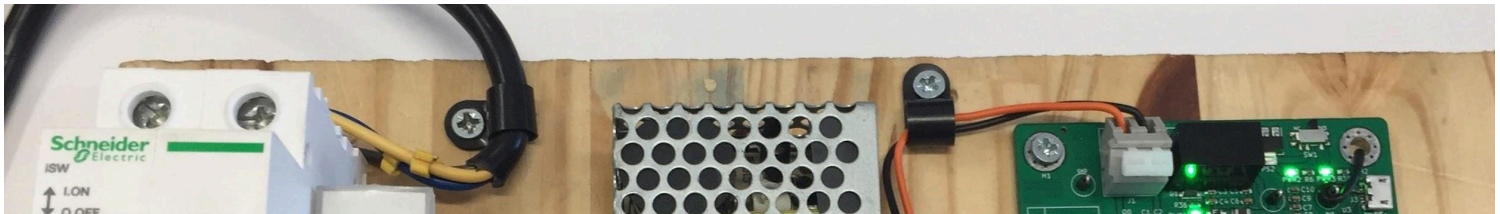
The inverting logic gate (The Npn diode) that was used to set the – pin and – pin which sets the MAX485 into transmit or receive mode was removed. This logic is not necessary as the pin is inverted already so both pins receive the same logic to switch between transmission and reception mode.





Power supply validation

A 24V supply was connected to the 9V - 36V Input. +-5v and 3.3v were measured at the correct locations and STM MCU started up correctly and began to run the code.

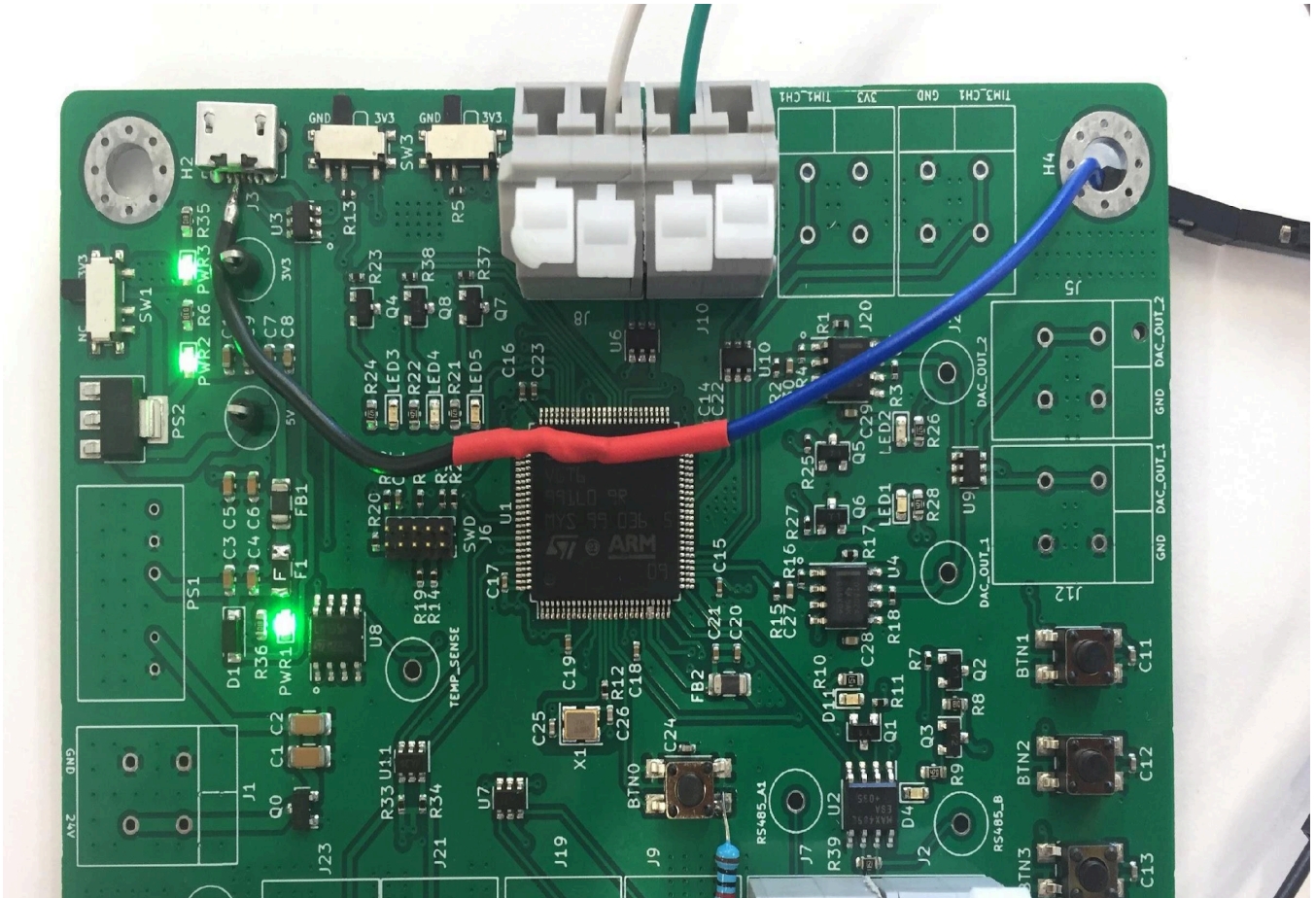


System Programming

UART Programming of the AWG

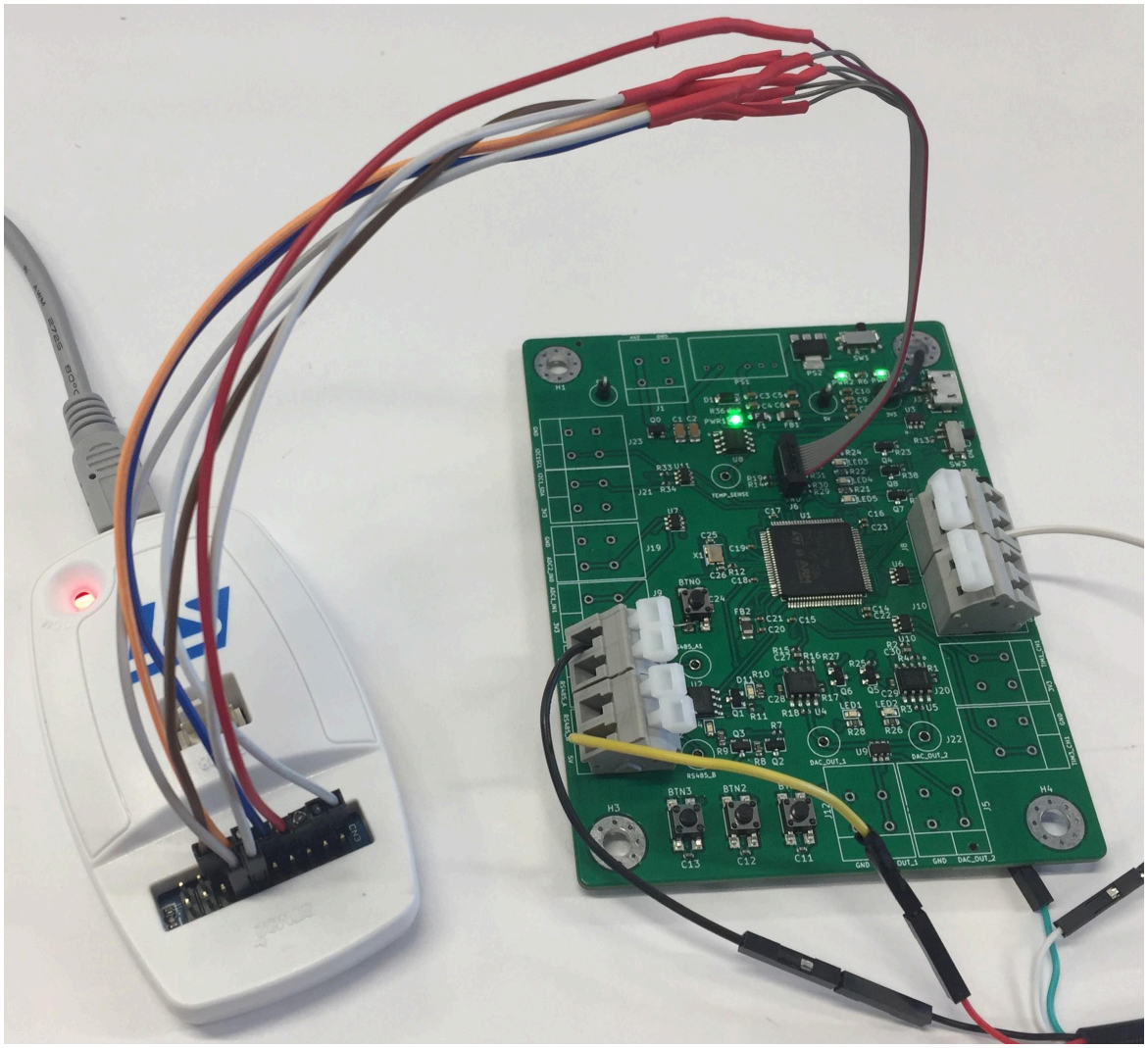
The AWG was first programmed via the USB to serial interface

1. An external wire was soldered to a USB ID pin to access PA10
2. Boot0 switch(sw3) was set to 3v3 and the device was reset to enter bootloading mode.
3. Start STM32CubeProgrammer and select UART programming
4. Select erasing & programming.
5. Select the .elf file to be uploaded, this can be found in the Debug folder of the build files from the STM32CubeIDE.



ST-Link V2 Programming of the AWG

1. Connect debugger and AWG PCB using the connector as seen in the figure below.
   - The footprint for the debugger on the AWG PCB is incorrect and inverted therefore the connector had to be manually wired to the correct orientation.
   - Add an extra wire from the ground of the debugger to the ground of the AWG PCB. Not seen in this image
2. Start STM32CubeProgrammer and select ST-Link programming
3. Select erasing & programming.
4. Select the .elf file to be uploaded, this can be found in the Debug folder of the build files from the STM32CubeIDE.



System Test

The max output frequency is dependent on Ns, the number of samples that form one full period of the sine wave. For lower output frequencies a higher Ns can be used to increase the resolution of the signal. The following table indicates the appropriate ns value for a desired output frequency range. These are based on the STM32F429VGT6 used on the AWG.

Based on the calculation:

PSC = - 1

There should be a function in the code that takes in the desired frequency and selects the appropriate Ns.

Other requirements for AWG:

1. The output signal must be scaled by o.5.
2. The output signal must be offset by 500.
3. DAC resolution = 4096
4. Fclock = 90 MHz
5. Period = 1.
6. PSC > 50

Setting up the RS485

1. Work on the correct response for uart
2. Set the MAX485 chip on

Waveform Generation Calculations:

PSC = - 1, Fsine = - 1, Fsine = = 13.636 kHz

Notes:

- When using op-amps it is important to consider the frequency limitations caused by the limited slew rate and bandwidth.
- Managing to get a working sine wave in the range of 0Hz and +- 50kHz.

Research

Waveform Generation Calculations:

PSC = - 1, Fsine = - 1, Fsine = = 13.636 kHz

Notes:

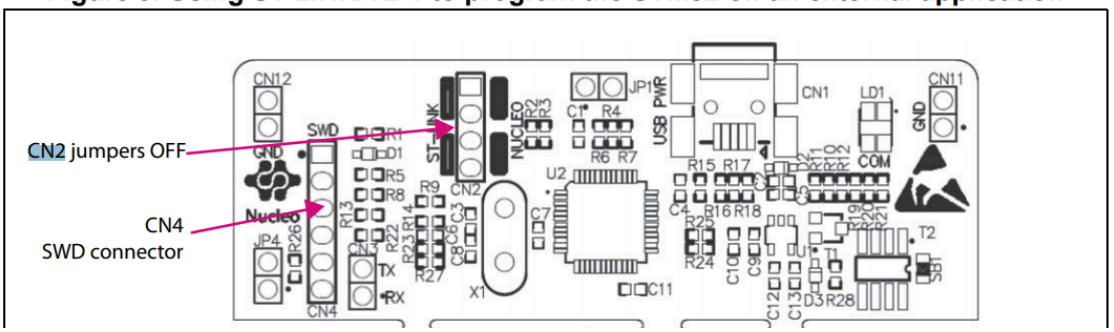
- When using op-amps it is important to consider the frequency limitations caused by the limited slew rate and bandwidth.
- Managing to get a working sine wave in the range of 0Hz and +- 50kHz.

STM32 Nucleo 144 SWD Pinout

Using this pinout proved difficult and I was not able to programme the development board this way.

Pin	CN4	Designation
1	VDD_TARGET	VDD from application
2	SWCLK	SWD clock
3	GND	ground
4	SWDIO	SWD data input/output
5	NRST	RESET of target STM32
6	SWO	Reserved

Figure 9. Using ST-LINK/V2-1 to program the STM32 on an external application



UART bootloader

Boot pin configurations: Refer to the chip reference manual for the correct configuration of pins Boot0 and Boot1 to put the device into the desired bootloader mode. For the STM32F439VGT6 the following pin configurations set the desired memory location to refer to for the boot mode.

USART bootloader protocol: Refer to AN2026 to activate the specific peripheral bootloader pins and settings. For the USART1 bootloader protocol commands and process refer to AN3155.

All communication from the programming tool to the device is verified by:

1. Checksum: received blocks of data bytes are XOR-ed. A byte containing the computed XOR of all previous bytes is added to the end of each communication (checksum byte). By XOR-ing all received bytes, data plus checksum, the result at the end of the packet must be 0x00.
2. For each command, the host sends a byte and its complement (XOR = 0x00).
3. UART: 1 start bit, even parity and 1 stop bit.
4. Each packet is either accepted (ACK answer) or discarded (NACK answer):
   - ACK = 0x79
   - NACK = 0x1F

UART message protocol

8 bytes message structure:

|<|(60/3C) : Start of message byte. |ADDR|() : Device Address byte. |CMD|() : Command byte. |DATA1| : Data byte 1. |DATA2| : Data byte 2. |DATA3| : Data byte 3. |DATA4| : Data byte 4. |>|(62/3E) : End of message byte.

List of commands from CRIO to STM:

1. On/off command.
2. Change - the frequency of DAC channel 1.
3. Change - the frequency of DAC channel 2.
4. Change - the amplitude of DAC channel 1.
5. Change - the amplitude of DAC channel 2.
6. Request - Voltage and Current measurement of channel 1 output.
7. Request - Voltage and Current measurement of channel 2 output.
8. Request - Temperature sensor 1 and 2 output.
9. Acknowledge message received.
10. Bad message received.
11. Request current system state.

List of messages from STM to CRIO:

1. Return - the frequency of DAC channel 1.
2. Return - the frequency of DAC channel 2.
3. Return - the amplitude of DAC channel 1.
4. Return - the amplitude of DAC channel 2.

5. Return - Voltage and Current measurement of channel 1 output.
6. Return - Voltage and Current measurement of channel 2 output.
7. Return - Temperature sensors 1 and 2 output.
8. Acknowledge message received.
9. Bad message received.
10. Low power mode.

List of Bootloader commands This is a snippet from AN3155. Each command that is sent through to the STM must be sent with its compliment.

- The STM will return with a “79” to acknowledge a correct message.

Command(1)	Code	Description
Get(2)	0x00	Gets the version and the allowed commands supported by the current version of the bootloader.
Get Version & Read Protection Status(2)	0x01	Gets the bootloader version and the Read protection status of the Flash memory.
Get ID(2)	0x02	Gets the chip ID.
Read Memory(3)	0x11	Reads up to 256 bytes of memory starting from an address specified by the application.
Go(3)	0x21	Jumps to user application code located in the internal Flash memory or in the SRAM.
Write Memory(3)	0x31	Writes up to 256 bytes to the RAM or Flash memory starting from an address specified by the application.
Erase(3)(4)	0x43	Erases from one to all the Flash memory pages.
Extended Erase(3)(4)	0x44	Erases from one to all the Flash memory pages using two byte addressing mode (available only for v3.0 USART bootloader versions and above).
Write Protect	0x63	Enables the write protection for some sectors.
Write Unprotect	0x73	Disables the write protection for all Flash memory sectors.
Readout Protect	0x82	Enables the read protection.
Readout Unprotect(2)	0x92	Disables the read protection.
Get Checksum	0xA1	Computes a CRC value on a given memory area with a size multiple of 4 bytes.

1. If a denied command is received or an error occurs during the command execution, the bootloader sends NACK byte and goes back to command checking.
2. Read protection. When the RDP (Read protection) option is active, only this limited subset of commands is available. All other commands are NACK-ed and have no effect on the device. Once the RDP has been removed, the other commands become active.
3. Refer to STM32 product datasheets and to AN2606 to know the valid memory areas for these commands.
4. Erase (x043) and Extended Erase (0x44) are exclusive. A device can support either the Erase command or the Extended Erase command, but not both.

Once the system memory boot mode is entered and the STM32 has been configured:

1. The DFU command “7F 00”, No compliment needed
2. The GET command “00 FF”
3. The GO command “21 DE”

- Send, the start address, this is usually 0x8000000, wait for ACK
- Might require a checksum to be sent with this address, AN315

4. The EXTENDED ERASE command “44 BB” (This seems to work inconsistently).
   - First, send the READ PROTECT command “82 7D”
   - Then send the READ UNPROTECT “92 6D”
   - The WRITE UNPROTECT command “73 8C”
   - Then send EXTENDED ERASE COMMAND “44 BB”
5. The WRITE MEMORY command “31 CE”
   - Send, 31 CE, wait for ACK
   - Send, 08 00 00 00 08, the start address which is usually 0x08000000 and the CRC, wait for ACK
   - Send, the number of bytes to be written (1 byte), the data (N+1 bytes), the checksum, wait for ACK
     - Checksum byte: XOR (N, N+1 data bytes)
     - N+1 must be a multiple of 4
6. The WRITE PROTECT command “63 63”
7. The WRITE UNPROTECT command “73 8C”
8. The READ PROTECT command “82 7D”
9. The READ UNPROTECT command “92 6D”

Software changes

Problem

The max485 chip is physically connected to UART2 Tx and Rx pins. UART2 uses the same DMA channels as the DAC 1 and 2 outputs. So using the DMA method for receiving data on the UART2 RX line won’t work.

Solution : - Using the UART interrupt (IT) method to handle the received messages. - Using the UART polling method for sending messages