|  |
| --- |
| The University of York |
| Electronics Department |
| Design & Construction |
| Initial Report |
|  |
| **Jonathan Caine**  **Sam Beedell**  **Alexander Brown**  **Liyu Ding** |
| **11/02/2014** |

# Introduction

The project is to create an accurate, low distortion signal generator for sine, square and triangle waveforms, alongside a frequency meter for the same. This will be implemented using an ARM processor carrier board from ST Microelectronics (STM32F407VGT6) fitted with various peripherals.

The project will be split into two parts, software and hardware. From the combination of these two parts, the maximum possible frequency range for the waveforms needs to be achieved, and again for the frequency meter using the lowest power achievable. The input signal to the frequency meter on the board is provided by the hardware and displays the frequency on the LCD display. Further work on the product may include using the LCD screen, I2C connector, USB connector, and Direct Digital Synthesis (DDS) using an AD9850 chip to generate accurate pulse and arbitrary functions.

Sam Beedell and Jonathan Caine make up the software team. They have been focusing on creating achievable concepts to generate high frequency signals and measure waveforms from the DAC, also the most efficient way to output signals on specific GPIO pins. With solid concepts made they can implement each level confidently and then begin to achieve the further requirements, with consideration of their relevant limitations.

Alex Brown and Liyu Ding make up the hardware team. Their main focus so far has been experimentation with the hardware generation of the required signals, particularly investigating the limits of the various methods available. Future work will involve designing a switching circuit, the dc offset and amplifier circuit as well as working out the best method to convert an incoming signal into a useable input for the software team to measure the frequency.

# Specification

## Level 2 System Requirements

* Generate a sine wave from 0.01Hz to 100kHz with a maximum amplitude of +/- 12V.
* Generate a square wave from 0.01Hz to 100kHz with a maximum amplitude of +/- 12V.
* Generate a triangle wave from 0.01Hz to 100kHz with a maximum amplitude of +/- 12V.
* Measure the frequency of sine, square and triangle waves from 0.01Hz to 100MHz with a minimum signal of 0.1V rms.
* Control the amplitude of sine, square and triangle waves by sending control signals to a digital potentiometer, which will control the gain of the amplifier circuit.
* Create a pulse generator from 0.01Hz to 1MHz with a maximum amplitude of +12V.
* Measure the duty cycle for digital waveforms.
* Vary the duty cycle for digital waveforms.
* Amplitude modulate an input signal by using a sine wave to send control signals to a digital potentiometer, which will control the gain of the amplifier circuit.
* Frequency modulate an input signal by multiplying it with a sine wave to transpose the fundamental frequency to a much higher frequency for transmission, the frequency of the output signal is based on a sine wave with a fixed frequency.
* Supply 5V of power to the external board.
* Operational LCD on external board to hold up to 16x2 characters, displaying frequency and duty cycle of waveforms.

## Further Requirements

* Generate square waves using DDS on the AD9850 chip to increase maximum frequency.
* Generate sine waves using DDS on the AD9850 chip to increase maximum frequency.
* Create a random noise generator using the internal PR sequence generator.

# Software Design/Implementation

## Tasks so Far

After following a few brief example programs provided as part of lab script 1 [1], the software team started the wave generation section.

To start with they generated a square wave with a 50:50 duty cycle, by outputting a high for 5 milliseconds followed by a low for 5 milliseconds on GPIO pin PA0. This gave a nice clean square wave which was viewed on the oscilloscope, and the frequency and period can be measured accuratly. The theoretical period was 10 milliseconds giving a 100Hz frequency, however when measured on the oscilloscope the period was 32 milliseconds and the frequency at 31.25Hz. Consequently this method of setting the frequency is very poor in terms of accuracy, and therefore unsuitable given our specification.

Before progressing further, it was appropriate to be able to measure the frequency of the generated waves within the processor, the LCD screen must display the frequency of the current wave being generated according to labscript 1 [1], and be able to use the buttons to adjust the frequency and duty cycle. So in order to display the frequency and adjust it, the current frequency must be known, so the software team started investigating the input capture function of the timers. Initially this concept seemed to have problems relating to timing delays, as it appeared that there was a small delay between a rising edge being detected, and the counter starting. This would have been no good for measuring frequency as unless this delay was fixed and the same every time, then it’s extremely difficult to account for it. However after looking further at the reference manual [REF ME] it appears that it may be possible to use the input capture function with one of the advanced timers 1 or 8.

## Tasks Still to Complete

The initialisation of the DDS chip (after the hardware team have created the appropriate breadboard circuit) using software needs to happen first, followed by the implementation of a square wave. This should be relatively simple from reading through the data sheet for the chip [REF ME], as it’s a case of writing values to resisters to control the output frequency and duty cycle directly.

Having implemented a square wave using the DDS chip, it should be simple to expand on this and generate a sine wave.

The pulse generator should also be able to be implemented using the DDS chip, as it is simply a square wave with a non 50:50 duty cycle.

Measuring the duty cycle of the waveforms and displaying it on the LCD should also be relatively easy if using the DDS chip because the duty cycle is controlled by configuring the DDS chip directly, the same value simply needs to be converted to a human readable form then displayed.

As the DDS chip is unable to generate triangle waves, this will have to be done using a counter that outputs its value to the Digital to Analogue Converter (DAC). Some investigation will be required as to which counter would be the best to use (16 or 32 bit depending on precision), and the limitations of the DAC such as maximum operating frequency.

Following this there is a need to implement a control interface via the digital potentiometer so that the hardware team can control the output amplitude of the waveforms generated on the ARM board. The ARM board can only output voltages in the range 0-3 volts, whereas +/- 12 volts is required from the specification, so the hardware team will need to amplify and level shift the output we provide.

After the generation of waveforms a frequency meter must be implemented. Two options have already been considered; either using the input capture functionality of one of the advanced timers that will allow the detection of a rising edge on a waveform, thus creating an event which triggers a counter to count up until the next rising edge is detected. This functionality can calculate the period of the waveform as the frequency of the system clock is known, and therefore the rate at which the counter counts. From the period the frequency can be calculated by doing 1/period. Alternatively a timer can start with interrupts enabled, and set to fire every 1 second for example. While the timer is running, a counter counts the number of highs its sees on its input pin (GPIO pins can be mapped to the inputs of counters). Knowing the number of highs during a fixed amount of time enables calculation of the period of the waveform, from which the frequency can again be calculated. As to which method is the most suitable will require further investigation.

Next the amplitude modulation functionality, which makes use of the fact a sine wave with a known frequency has been generated, and that an interface to the digital potentiometer is available. By mapping the output of the generated sine wave to the digital potentiometer, the digital potentiometer can use the sine wave as a carrier frequency to modulate the amplitude of the input signal.

Following this the frequency modulation functionality is to be implemented, which will involve simply multiplying the input waveform by a sine wave with a known frequency, allowing us to modulate the frequency of the input signal. Some investigation into this will be required as to whether the best performance can be achieved in software or hardware.

No psuedo code can be provided, as the software team has adopted an agile approach to the software side of the project, which means you do not consider the next piece of functionality until the one you are currently working on is fully implemented and tested. Otherwise many hours are spent upfront, planning and designing the product, often only to change dramatically during implementation. Therefore the team feels that they would be wasting their time trying to plan the implementation detail of the functionality up front (ie psuedo code) rather than just going ahead and implementing it straight away, and are more suited to start coding straight away and consider each piece of functionality in a discrete manner.

# Hardware Design/Implementation

## Tasks so Far

The hardware team so far has been working together to investigate a variety of circuits for generation of the square, triangle and sine waves. Their main focus has been trying to generate a 1MHz signal from each circuit, with 100 kHz being the absolute minimum requirement to achieve the level one design. They have managed to generate all three types of signal with varying degrees of success in terms of the highest frequency obtained.

**Square-waves**

Three circuits have been utilised for square wave generation so far: a Schmitt trigger configuration and two astable multivibrator circuits as described on the Intersil datasheet [1]. Figure 1 shows the Schmitt trigger circuit, with a LM318 op-amp; a TL071 was also used though it has a lower slew rate than the LM318 meaning it would be less likely to achieve the 1MHz goal. Figure 2, below, shows the output from this circuit set up as given in the lab script [2] to give a 400Hz frequency output.

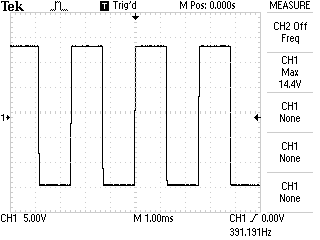


Figure 2 Oscilloscope display for the Schmitt trigger

As can been seen this configuration gives a very clear square wave output. The next task was to work out whether to change the capacitor or the resistor to vary the frequency as both components effect this. Figures 3 and 4 show the generation of a 100Hz wave, with the circuit represented by figure 3 having a different capacitor and that represented by figure 4 using a different resistor to the one in the initial circuit. It is quite clear that at low frequencies it doesn’t matter which component is changed as both give the same result.

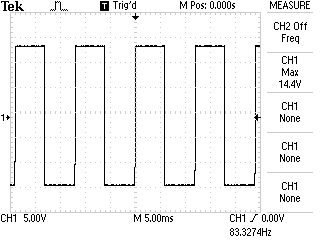
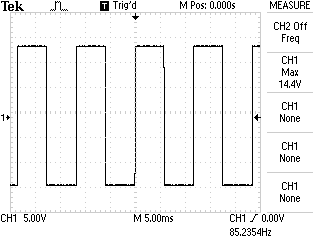


Figure Output due to changing the capacitor Figure Output due to changing the resistor

It was then decided to try a frequency of 10 kHz again varying just one of the components. Again both produced the same results however, as can be seen in Figures 5 and 6, the slew rate of the op-amp became a limiting factor and thus the rise and fall times for the square wave have increased significantly. This circuit is clearly unsuitable for use in the function generator as it is unable to produce precise square waves at even moderately high frequencies.

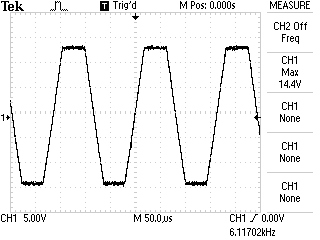
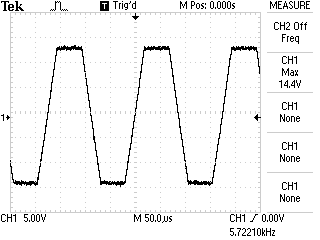


Figure 5 Output with fixed resistor value Figure 6 Output with a fixed capacitor value

The next circuit investigated utilised the 7555 timer IC as part of a multivibrator circuit, set up following figure 2B from the datasheet [1]. This configuration allows for the duty cycle to be varied as well as the frequency, however it was quickly found that trying to adjust one would also cause the other to change and that it would be difficult to achieve a fixed 50/50 duty cycle while also being able to change the frequency. Fortunately the datasheet provided another circuit (Figure 2A from [1]) that had a fixed 50/50 duty cycle; this circuit has two outputs, both of which have been considered. Figure 7 shows both of these outputs for a 1 kHz square wave, each one giving a very clean signal.

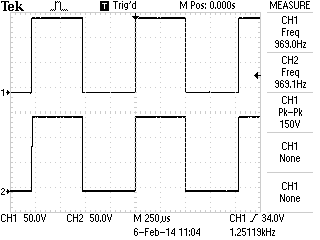


Figure 7 Output from the 7555 circuit set up to give a fixed 50/50 duty cycle

The next step was to attempt to output a 100 kHz signal; this is show in figure 8.

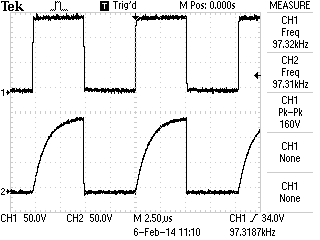


Figure 8 The 7555 circuit outputting an approximate 100kHz wave

Here the outputs have become much noisier and the alternate output’s rise time is affected drastically by the capacitor. Ultimately a 700kHz output was obtained, though the signal at this point had a noticeable rise time and was fairly noisy, the other output at this frequency was giving a saw tooth wave, figure 9 shows the scope display.

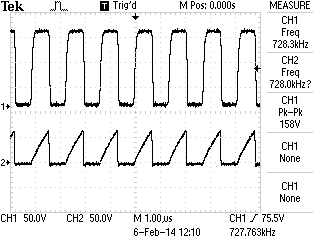


Figure 9 The maximum possible value from the 7555 circuit

The team found that a frequency much higher than this would be hard to obtain as both the resistor (100Ω) and the capacitor (100pF) were heading towards being dominated by the internal resistance and capacitance of the wiring used to create the circuit. It was also decided to check that this circuit operated correctly at very low frequencies. Figure 10 shows a 0.01Hz signal. The scope was unable to measure such low frequencies however a very clean signal was obtained meaning that this circuit could be used if the software team struggled to generate lower frequency signals.

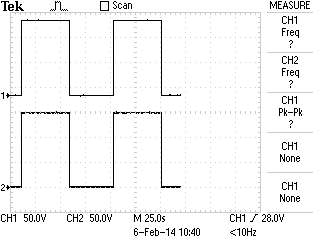


Figure 10 Using the 7555 to generate a very low frequency signal

After some research on the internet the team came across a circuit designed to generate a triangle wave [3]. Changing R1 to a variable potentiometer allowed the frequency to be adjusted. Currently the team have achieved a top frequency of about 500Hz; they have been limited by the fact that as the frequency increases the amplitude of the output waveform decreases. Beyond 500Hz the amplitude was so small that the signal barely registered on the oscilloscope. The team plan further research to find a circuit capable of achieving a higher frequency without the amplitude drop witnessed with the current setup.

The final task the team have so far accomplished is to build a simple Wien Bridge sine wave generator based on the circuit diagram from [2]. Due to the slew rate limitations a LM318 op-amp was used in this circuit. So far this circuit has been used to generate a 400kHz sine wave although at this frequency the sine wave was starting to look more like a triangle wave. The team again preferred to switch the capacitor for each multiple of ten frequency jump, with the idea that this set up is relatively easy to achieve with a simple switching circuit along with a digital potentiometer. Further experimentation is still needed to try and generate a clearer sine wave at high frequencies.

Overall the team feels that due to the limitations of the components the higher frequency generation should be done via software, with potential for the low frequency generation to be carried out in hardware. Their main focus going forward will be on implementing the digital potentiometer in the dc offset and amplification circuit.

## Tasks Still to Complete

**Switching circuit**

Switching circuits are supposed to be used for designing an integrated circuit for changing between three kinds of waveforms and vary their frequencies. Multiple switch or analogue switches controlled by processer such as DG201 quad SPST Switch and DG409 dual 4--‐1 mul0plexer will be used. This task will base on the final specific design of the oscillator circuits and the relationship between frequency of waveforms and components in oscillator circuit. For example, change the value capacitors is one of the solutions of changing sine wave frequency for Wien Bridge oscillator. In this case, circuit like figure [1] might be used and resistors connected with Half DG409 will be replaced by different value capacitors.

Limitation of this solution is that frequencies could not be varied into specific values, only several fixed value are available. In order to solve this, team will try to figure out the way of using other designs like variable resistors or using digital potentiometer.

**Compare CMOS square wave oscillator with 7555**

4093 CMOS is another square wave generator which was suggested to use in lab script. Although team already got a working 7555 square wave oscillator, some limitations of that circuit still exist. It is worth to build a CMOS oscillator and see what will happen and compare with 7555 oscillator.

**Better triangle wave generator**

Current triangle wave generator is just able to generate a fixed-frequency triangle wave. Team haven’t found the relationship between its frequency and components in the circuit.

Team planning to try to build a Schmitt trigger circuit and add an inverter at the output of trigger circuit for generating triangle wave. Circuit diagram figure [2] is added in appendix.

**Pulse generator**

This is a level 2 requirement. Team planning to use 7555 timer to build a pulse generator, circuit diagram figure [3] is added in appendix.

**Investigate other sine wave generators**

Wien bridge oscillator is the only circuit team have tried for sine wave generator. It works but has limitations when team tried to get high frequency waveform. In order to get higher frequency sine wave, it is necessary to try other kinds of sine wave oscillators such as RC Phase Shift Oscillator and Digital Shift register based Sine Wave Oscillator.

**Create amplifier and dc offset stage**

In order to control waveforms’ amplitude, team planning to design an amplifier circuits with variable resistors to change gain of the circuit. Circuit diagram figure [4] is added in appendix.

**Comparator with and without hysteresis**

Comparator would be used for convert waves produced by oscillators into digital signals. Team plan to build a circuit as figure [5] shows. Output signal of LM 319 will be connected into an N bit counter. Counter will count for N seconds and give output signals to GPIO. Hardware team will build both comparator circuits that with and without hysteresis and compare difference between them and decide which one should be used as final design.

**Detail of design**

*This product will be for indoor use and have an operational temperature of -20°C to 30°C. Protection against water resistance/humidity and dust will be considered at a later date. The housing of the board doesn’t currently exist so it is susceptible to being dropped and large vibrations. Operational lifetime will be made as long as possible. Minimal power consumption will optimize battery life but ideal power will be supplied by an external power supply.*

# Gantt Chart

# References

|  |  |
| --- | --- |
| [1] | Intersil, “ICM7555 datasheet,” 24 August 2006. [Online]. Available: http://www.elec.york.ac.uk/internal\_web/meng/yr3/modules/Design\_Construction/ICM7555%20better%20data%20sheet.pdf. [Accessed 8 February 2014]. |
| [2] | D. Chesmore, “Design and Constuction,” January 2014. [Online]. Available: http://www.elec.york.ac.uk/internal\_web/meng/yr3/modules/Design\_Construction/Design%20and%20Construction%20Lab%20Script%202014.pdf. [Accessed 8 February 2014]. |
| [3] | P. Marian, “555 Triangle Waveform Generator Circuit,” Electro Schematics, May 2010. [Online]. Available: http://www.electroschematics.com/5004/555-triangle-waveform-generator/. [Accessed 10 February 2014]. |

# Appendix

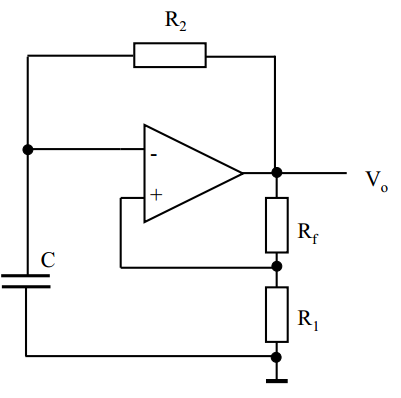


Figure 1 Schmitt trigger oscillator circuit diagram

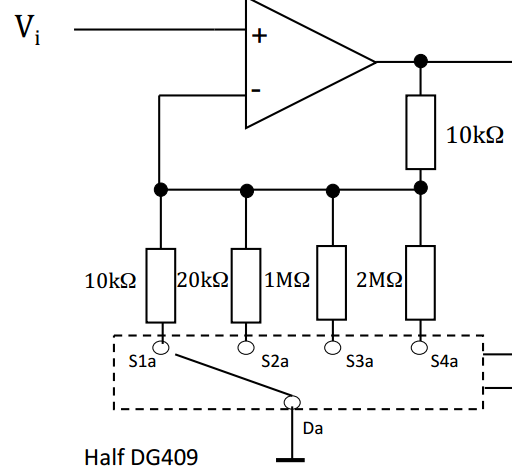


Figure. 11 <http://www.elec.york.ac.uk/internal_web/meng/yr3/modules/Design_Construction/Generating%20Signals.pdf>

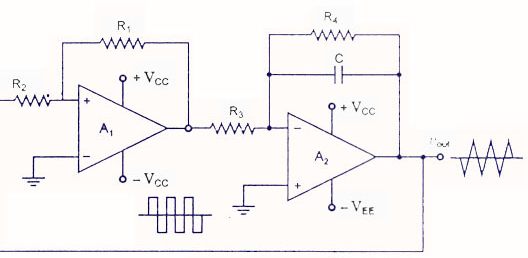


Figure. 12 <http://www.circuitstoday.com/wp-content/uploads/2009/09/triangular-waveform-using-schmitt-trigger-and-integrator.jpg>

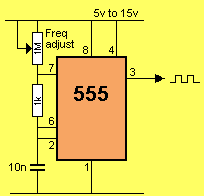


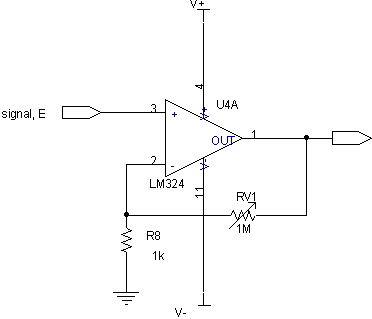
Figure. 13 <http://www.talkingelectronics.com/projects/555/Page2-555.html> 

Figure. 14 <http://homepages.which.net/~paul.hills/Circuits/PowerServo/PowerServo.html>

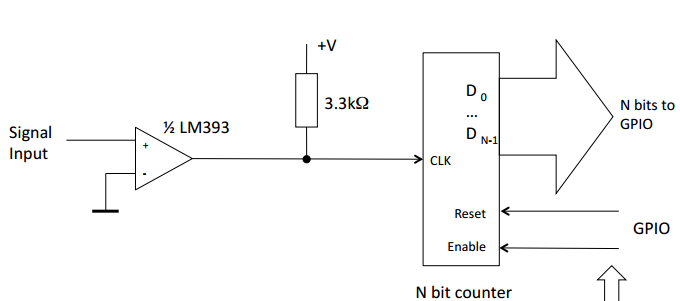


Figure .15 <http://www.elec.york.ac.uk/internal_web/meng/yr3/modules/Design_Construction/Measuring%20Frequency.pdf>