Handheld Oscilloscope with Multimeter based on Cortex-M Microcontroller

BY

POONG KA VUI

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION TECHNOLOGY (HONS)

COMPUTER ENGINEERING

Faculty of Information Communication Technology

(Perak Campus)

MAY 2016

Handheld Oscilloscope with Multimeter based on Cortex-M Microcontroller

By

Poong Ka Vui

A REPORT
SUBMITTED TO
Universiti Tunku Abdul Rahman
in partial fulfillment of the requirements
for the degree of
BACHELOR OF INFORMATION TECHNOLOGY (HONS)
COMPUTER ENGINEERING
Faculty of Information Communication Technology
(Perak Campus)

MAY 2016

DECLARATION OF ORIGINALITY

I declare that this report entitled "**HANDHELD OSCILLOSCOPE WITH MULTIMETER BASED ON CORTEX-M MICROCONTROLLER**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : ______________________________

Name : ______________________________

Date : ______________________________

ACKNOWLEDGEMENTS

First of all, I would like to express my appreciation to my supervisor, Mr. Lee Wai Kong for his patient and knowledge when guide me for this challenging project. Mr Lee Wai Kong shares his experience about the embedded system. By this knowledge, I had able to implement the firmware of oscilloscope and multimeter.

Apart from that, thanks to my parents and family for their love, support and encouragement while I am developing this project.

ABSTRACT

In real life, oscilloscope acts as an important tool in various fields such as engineers, electronic technicians, physicists, and automobiles. An engineer uses an oscilloscope for designing electronic or electrical equipment. Sound engineers rely on oscilloscopes for observe sound equipment's frequency response. Scientist uses an oscilloscope for observe the nature moves in the nature such as earthquake or sonic boom.

As you see, oscilloscope is an important part of many modern professions and also important for a student to learn how to use the oscilloscope before they step out to work. While students are only able to use oscilloscope during university's laboratory due to oscilloscope is an expensive tool and come with huge size.

In this case, we propose to develop a handheld oscilloscope to create more chances for student to hands on some work with oscilloscope. Throughout this project, a handheld oscilloscope is developed and it also comes with multimeter features.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 INTRODUCTION

1-1 Problem Statement

Massive open online courses (MOOCs) is the recent trend in education. Websites such as Khan Academy, edX and Coursera have begun developing large online classes for a variety of technical fields such as electrical engineering, mathematics, and computer science. Where these courses have a problem is that they cannot provide hands-on experience for classes that would need lab component.

For electrical engineering and computer engineering classes, they often use some basic lab equipment such as oscilloscopes and function generator. These equipments are often easily cost thousands of Ringgit Malaysia. Most students cannot afford to purchase such expensive lab equipment. Furthermore, students that enrolled in online course may not have access to a lab or university that could provide such test equipment.

Traditional analog oscilloscopes are invariably bigger and heavier, this kind of oscilloscope is not suitable for hand carry and often put in a lab. Even though there are portable oscilloscopes exist commercially, but the portable oscilloscope still large and quite burden to carry it to everywhere.

Hence, there is a very high demand in producing a handheld oscilloscope with multimeter that is low production cost.

CHAPTER 1 INTRODUCTION

1-2 Background and Motivation

Nature moves in the form of a sine wave, such as sound through air, ocean wave, earth quake, or natural frequency of a body in motion. Even light also an example of wave, where it has fundamental frequency which is referred to as colour. Sensor such as ADC can convert these wave forces into electrical signals that we can study and observe these waves with an oscilloscope.

Oscilloscopes are important tools for those people who designing, repairing or manufacturing electronic equipment. As the eyes of the engineer, oscilloscopes are the key to meeting today's demanding measurement challenges. A computer engineer uses an oscilloscope to measure I^2C signals from ICs to processor or microcontroller.

Oscilloscope is actually a graph displaying device where it draws the electrical signal on a screen, this graph shows how signals change over time: the horizontal axis (X axis) represents time, and the vertical (Y) axis represents voltage.

Oscilloscope can tell us many things about signal, which are shown as below

- Voltage and time values of a signal
- The frequency of an oscillating signal and the frequency with which a particular portion of the signal is occurring relative to other portions
- Whether or not a malfunctioning component is distorting the signal
- How much of the signal is noise and whether the noise is changing with time

Digital multimeter is a tool which people used to measure electrical values. Basic functionality of a DMM is that it able to measure voltage, current, resistance, capacitance, and continuity. Some advance DMM is able to do more features such as measure temperature, diode and IC testing mode.

DMM is generally handheld where designed to be used while holding it everywhere. All the calculated electrical values is then display on the digital screen of the DMM.

CHAPTER 1 INTRODUCTION

The motivation of this project is to implement an oscilloscope with multimeter which is relatively affordable by students and design in smaller size so that students can use it in the comfort of their own homes or bring everywhere.

1-3 Project Objectives

In this project, the objective is to design, implement and build a handheld oscilloscope for students and hobbyist. There are several objectives need to be accomplishing in order to achieve our target which are as below:

1. Design and implement the firmware of oscilloscope
2. Design and implement the hardware for oscilloscope signal input attenuator
3. Design and implement the firmware of multimeter
4. Design and implement the circuit for multimeter

By the end of this project, a handheld oscilloscope with additional function of multimeter is developed. The oscilloscope is able to convert the wave signals into electrical signals that we can study and observe with this oscilloscope.

1-4 Proposed Study

1-4-1 Digital Signal Processing (DSP)

DSP knowledge is important in this implementation because all the sampled analog signal will converted to digital form for digital representation of signals and the use of digital processors to analyse or extract information from signals. On the other hand, all the signal sampled is time domain signal, by analyze the time domain signal it is required to use Fast Fourier Transform to convert the time domain signal to frequency domain signal where all these are knowledge from Digital Signal Processing.

1-4-2 Programming Concept & Data Structure

To program embedded system, it is common that using C language to program it. In order to know how to program it, knowledge of programming concept and data structure is important because embedded system has only limited resource. To utilize the resource, knowledge from data structure can be used and to prevent data race condition.

1-4-3 Circuit Theory

This project required to build extra circuit integrate with microcontroller. Hence knowledge such as voltage divider rule, operational amplifier, and Ohm's law are important.

1-5 Achievement

Hardware and firmware part of oscilloscope and multimeter are successfully designed and developed. All the hardware and firmware are also integrated become a fully functional system.

Oscilloscope in this design is able to sampling speed up to 2MHz with 4 different voltage divisions and 17 time divisions. There are 11 types of signal measurement being implemented and also having FFT feature. Cursor function also being develop to let user easily study the signal.

On the other hand, four features being developed in multimeter which is voltmeter, ammeter, resistance and continuity.

1-6 Report Organization

This report consists of eight chapters. In Chapter 1, it is mainly briefing about the project background and problem statement, objective, achievement and organization of the report. Chapter 2 is literature review, there are 3 real product and comparison between these product included in this chapter.

In Chapter 3, this chapter is the section of all hardware components for oscilloscope and multimeter used in this project. Chapter 4 talks about the firmware architecture of the design for oscilloscope and multimeter. Chapter 5 is about the function implemented for oscilloscope and multimeter.

In chapter 6, it is all about integration of whole system. Overall specification, block diagram, flowchart are included in this chapter. Testing of whole system and the results are included in chapter 7.

In the last chapter, which is chapter 8 is the conclusion of the whole design which consist of the summary of the project, project contributions, limitation and improvement.

CHAPTER 2 LITERATURE REVIEW

2-1 Product Study 1: Owon HDS1022M

Owon HDS1022M is a handheld unit combines of digital storage oscilloscope and digital multimeter (OWON HDS-N series digital oscilloscope, 2012). This device features a bandwidth of 20MHz, 2 input channel for measuring DC and AC voltage up to 400V. Besides that, it support sampling rate of 100MSPS and also able to store up to 6000 points per channel. It supports 20 automated measurements such as frequency, period, and peak-to-peak. It also supports math function such as multiply, add and Fast Fourier Transform(FFT) that enable user analysis the waveform. For the digital multimeter part, it support measuring of DC and AC current up to 10 ampere, resistance up to 40MΩ, and capacitance up to 100µF.

The screen is a 3.7 inch color TFT LCD for viewing the waveforms and measurements. It has USB plug-in functionality which allow user connects to the PC for download and prints the stored waveform data.

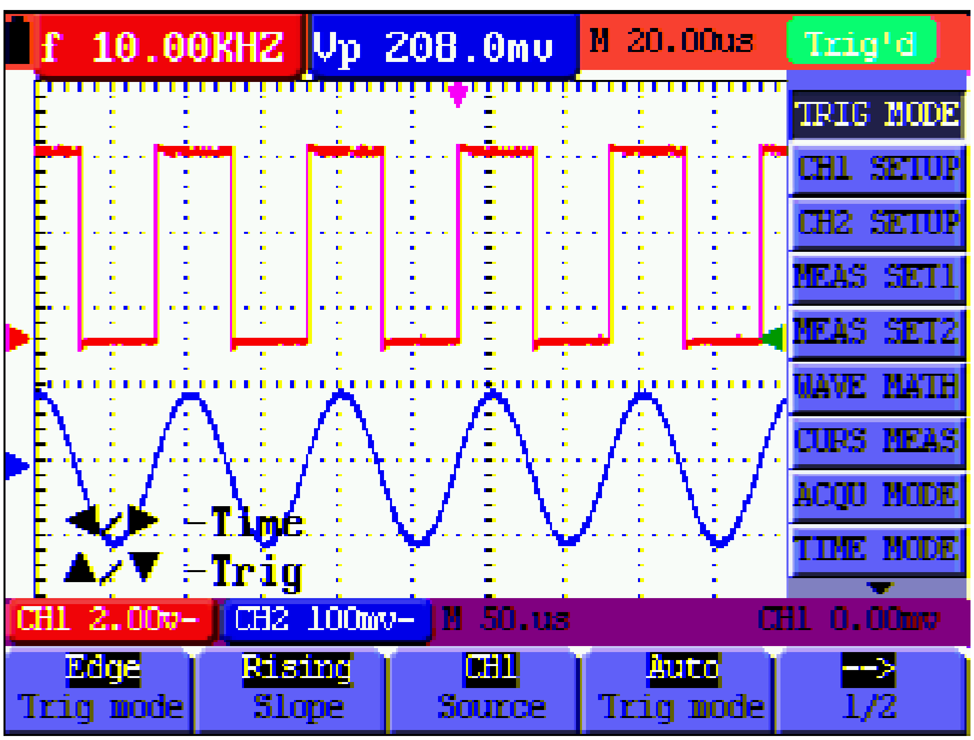

Figure 2-1-1: Interface of HDS1022M

2-2 Product Study 2: Velleman HPS140I

Velleman HPS140I is a tiny pocket oscilloscope. It features bandwidth of 10MHz and sampling rate of 40MSPS but 8-bit of analog to digital converter where this kind of resolution can't give user fine details of waveform. It support measurements such as peak-to-peak, weighted root mean square, and dBm (decibels) for audio/speaker impedance from 2 to 32 ohms. Where this oscilloscope is focus at audio work, but its bandwidth still enough for low-frequency pulse (Vellaman Inc, n.d).

The scope is quite high sensitivity where it support reading until 0.1mV. HPS140I has a special feature which is fully automatic setup system. Which this feature allows a signal quickly selects vertical (voltage per division) and horizontal (time per division) setting to show user the waveform. This auto range mode only work down to 5mV for the voltage per division section. The display used for HPS140I only 128x64 pixels where it only able to show a simple waveform.

HPS140I allows user to save waveform into memory. On the other hand, it does not support USB connectivity to pc which makes user cannot transfer or print the waveforms that stored in memory.

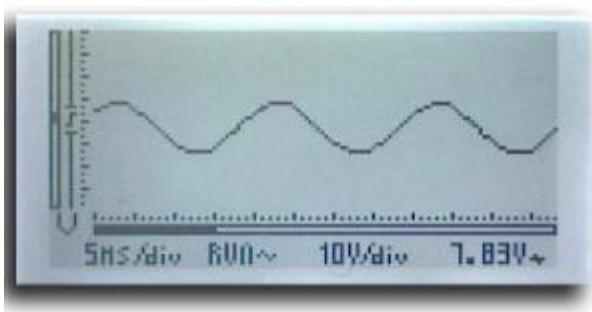

Figure 2-2-1: Interface of HPS140I

2-3 Product Study 3: Hantek DSO1060

Hantek DSO1060 is a handheld oscilloscope with digital multimeter function where it supports 2 channel of digital oscilloscope (Toolboom 2012). It features bandwidth of 60MHz and real-time sampling rate of 150MSPS. Where DSO1060 using a 8 bit ADC which allows the device shows input sensitivity from 10mV per division until 5V per division. This product allows maximum input of 400V.

DSO1060 using a 5.7 inch LCD to display waveform where allows user to observe the waveform clearly. It features 22 kinds of auto measurements such as rise time, fall time duty cycle and frequency. There are also 3 types of trigger modes edge, alternate and pulse. User can define own pulse width for the trigger.

DSO1060 allows user to store up to 1000 waveforms into internal memory or to external storage such as USB. The waveform is also can be save in .jpg or .bmp file formats.

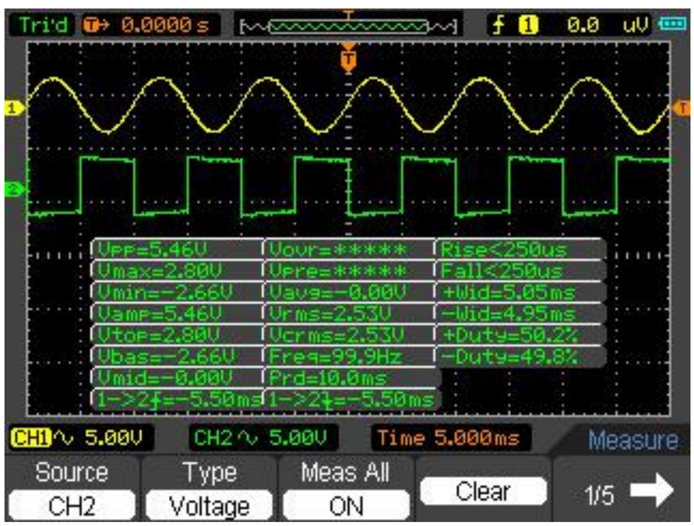

Figure 2-3-1: DSO1060 22 kinds of waveform measurement

2-4 Product Comparison

Specification	Owon HDS1022M	Vellaman HPS140i	Hantek DSO1060
Display / Display Resolution	3.7 inch TFT-LCD / 640 x 480	Mono Passive LCD/ 128 x 64	5.7 inch TFT LCD / 240 x 320
Sampling Rate	100 MSPS	40 MSPS	150MSPS
Vertical resolution	8-bit	8-bit	8-bit
Max Input Voltage	400V	100V	400V
Input Channel	2 channel	1 channel	2 channel
Vertical Sensitivity	5mV/div - 5V/div	0.1mV/div-20V/div	10mV/div - 5V/div
Horizontal Sensitivity	5ns/div - 100s/div	250ns/div - 1h/div	5ns/div - 1000s/div
Measurement	Vpp, Vavg, Vamp, Vrms, Freq, Period, Vmax, Vmin, Vtop, Vbase, Overshoot, Preshoot, Rise Time, Fall time, Duty Cycle	RMS, dBm, Vpp	Vpp, Vamp, Vmax, Vmin, Vtop, Vmid, Vbase, Vavg, Vrms,Vcrms, Preshoot, Frequency, Period, Rise time, Fall Time Duty Cycle
Waveform measurement	+, - , *, / , FFT	No	+, - , *, / , FFT
Multimeter Function	Voltage, Current, Impedance, Capacitance, Continuity	No	Voltage, Current, Resistance, Capacitance, Diode, Continuity
Download to PC	Yes	No	Yes

Table 2-4-1: Comparison between HDS1022M, HPS140i and DSO1060

As a result, most of the handheld oscilloscope are using 8 bit ADC to sampling the data. Where 8 bit ADC cannot give user fine details of waveform, but most of them is having a high sampling rate which enables to sample more data per second.

There are some similar function between the products such as most of them is able to measure the waveform's Vpp, Vavg, Vamp, Vrms, Freq, Period, Vmax, Vmin, Vtop, Vbase, Overshoot, Preshoot, Rise Time, Fall time and Duty Cycle. Besides that, they also have built in basic mathematic formulae such as add, subtract, multiply, divide and Fast Fourier Transform option.

CHAPTER 3 HARDWARE DEVELOPMENT

3-1 Microcontroller and development board

In this project, STM32F407VG microcontroller is used as the controlling and processing unit of the oscilloscope and multimeter. This microcontroller is manufactured by STMicroelectronics. STM32F407VG is based on the ARM Cortex M4 32-bit RISC core and operating at a frequency of up to 168 MHz. The Cortex M4 features a single precision, floating point unit (FPU) and also implements a full set of DSP instructions. Following table shows the specification of the STM32F407VG microcontroller.

Microcontroller	STM32F407VG
Core Processor	ARM® Cortex® -M4
Core Size	32-Bit
Speed	168MHz
Program Memory Size	1Mbyte
Program Memory Type	Flash memory
RAM Size	192+4 Kbytes of SRAM
ADC	3 ADCs
ADC Channels / Resolution	24 Channels / 12-bit
ADC Sampling Rate	2.4 MSPS and 7.2 MSPS in triple interleaved mode
General Purpose DMA	16-stream DMA controller with FIFOs

Table 3-2-1: Specification of STM32F407VG microcontroller

For ease of development, Open407V-D development board is used. Open407V-D is designed for the ST official tool STM32F4DISCOVERY, which features the STM32F407VG. This development board is manufactured by Waveshare Electronics. Following diagram shows the board layout of the Open407V-D and the table shows the layout description of the development board (Waveshare, n.d).

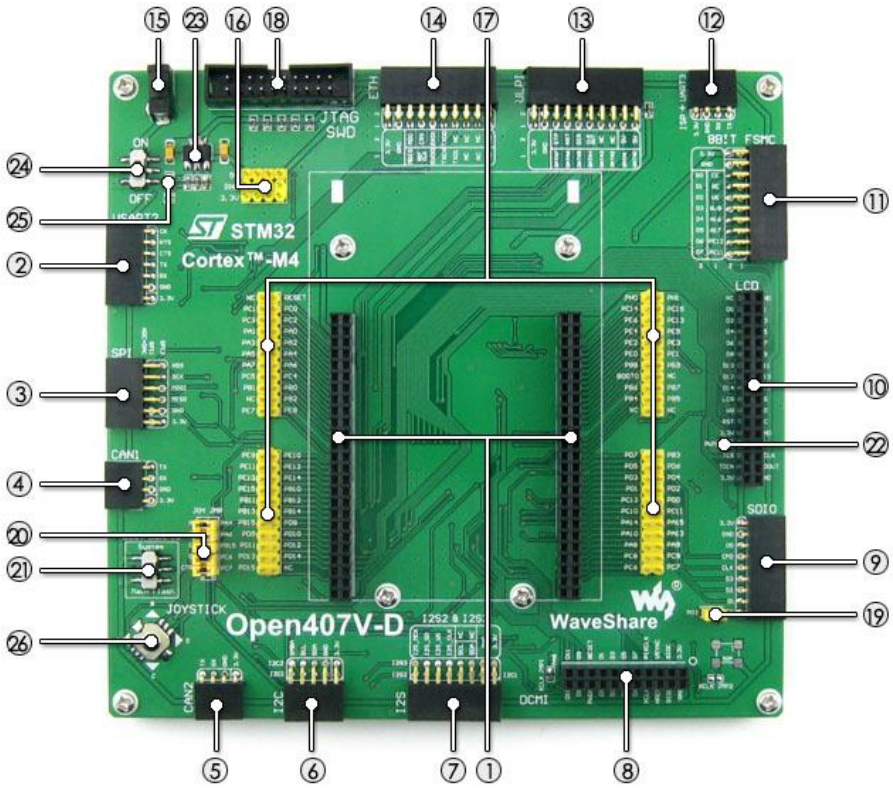

Figure 3-1-1: Layout view of the Open407V-D development board

Label	Function
1	STM32F4DISCOVERY socket
2	USART2 interface
3	SPI/SPI2 + AD/DA interface
4	CAN1 interface
5	CAN2 interface
6	I2C1/ I2C2 interface
7	I2S2 / I2S3 / I2C1
8	DCMI interface
9	SDIO interface
10	FSMC + SPI interface(16-bit FSMC + SPI)
11	FSMC interface (8-bit FSMC)
12	USART3 interface

13	ULPI interface
14	Ethernet interface
15	5V DC jack
16	5V/3.3V power input / output
17	MCU pins connector
18	JTAG/SWD interface
19	SD card detect jumper
20	Joystick jumper
21	Boot mode switch
22	LCD backlight adjustment enable jumper
23	AMS1117-3.3
24	Power supply switch
25	Power indicator
26	Joystick

Table 3-1-2-2: Layout Description of Open407V-D

3-2 Hardware Implementation for Oscilloscope

3-2-1 Hardware Block Diagram for Oscilloscope

There are 4 input attenuator circuits to let suitable voltage flow into ADC where ADC of this microcontroller only able to sample 0V to 3.3V. Where in between input attenuator and ADC, there will be an analogue multiplexer to pick input attenuator circuit and an op-amp circuit to convert positive to negative signal into a range suitable for ADC. Figure below is the hardware block diagram of oscilloscope.

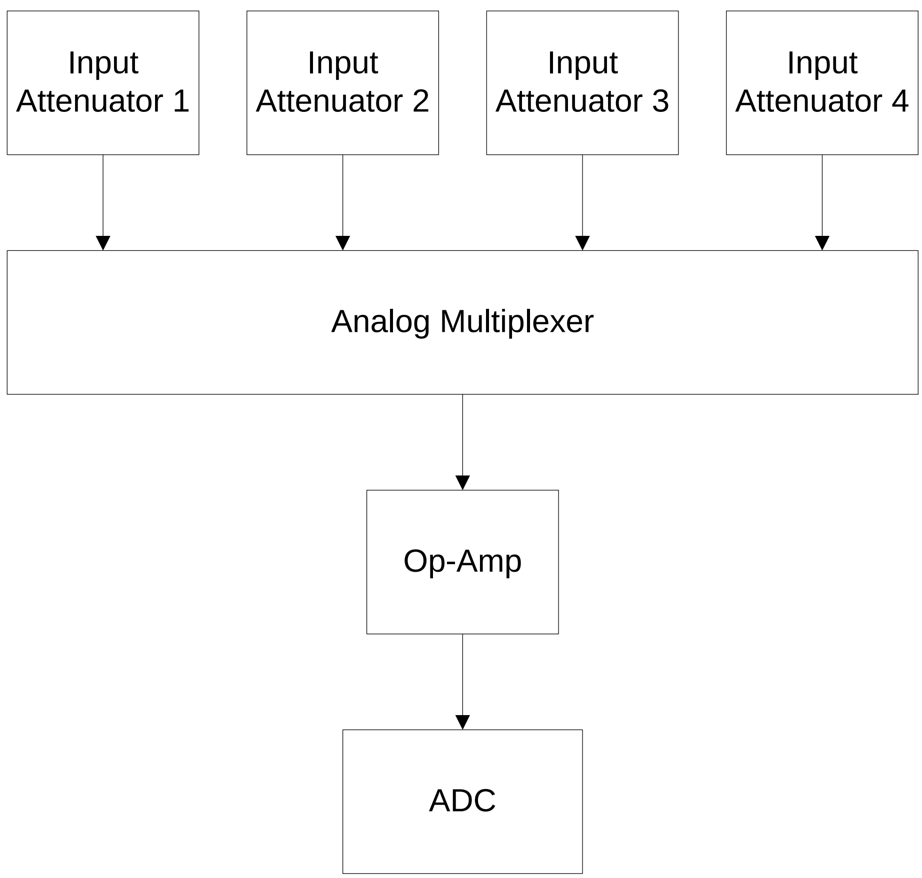

Figure 3-2-1: Hardware block diagram of oscilloscope

3-2-2 Input Attenuator

Input attenuator circuit is actually a voltage divider circuit which turns a large voltage into a smaller voltage. Voltage divider circuit is just using two series resistors and an input voltage, then able to create an output voltage which is a fraction of the input. Figure below shows the simple circuit of voltage divider circuit.

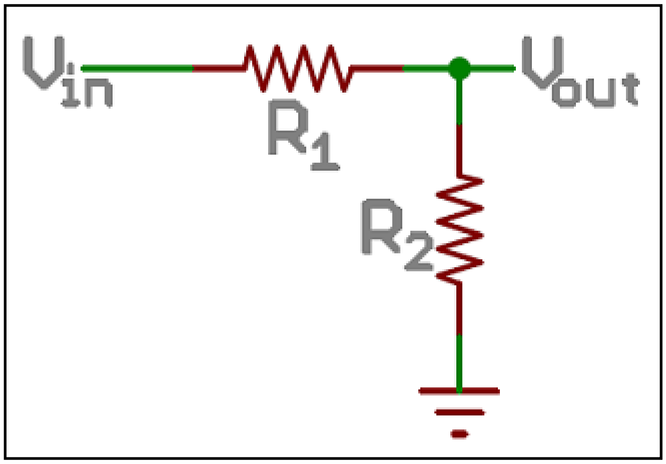

Figure 3-2-2-1: Simple Circuit of voltage divider

The voltage divider equation states that the output voltage of the signal is directly proportional to the input voltage and the ratio of R_1 and R_2. In order to get the value of V_{out}, we can use the equation below to calculate the V_{out}.

$$V_{out} = V_{in} \times \frac{R_2}{R_1 + R_2}.$$

3-2-2-1 Input Attenuator for 1V/Div

In this design, the oscilloscope design fixed 8 divisions for y-axis. Which means for 1V/Div, the scope have to show 4V to -4V. To design the input attenuator, 200kΩ resistor and 270kΩ resistor are used as R1 and R2. 4V will become 2.3V after pass through the circuit. Figure below shows the input attenuator for 1V/Div.

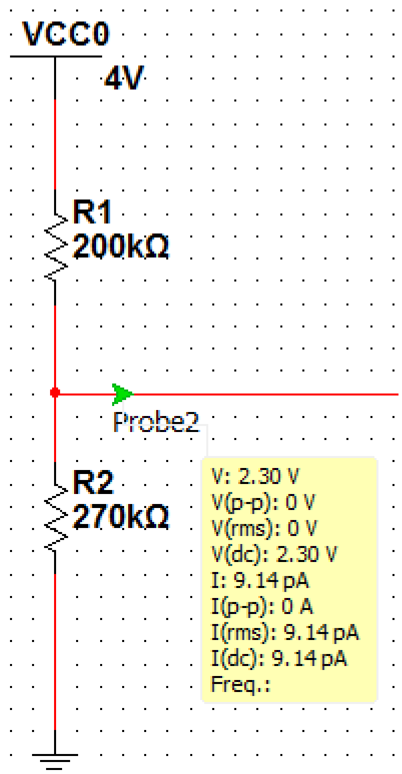

Figure 3-2-2-1-1: Input Attenuator for 1V/Div

3-2-2-2 Input Attenuator for 2V/Div

For 2V/Div, the scope has to show 8V to -8V. To design the input attenuator, 680kΩ resistor is used as R1 and 270kΩ resistor is used as R2. The 8V input will become 2.27V after pass through the circuit. Figure below shows the input attenuator for 2V/Div.

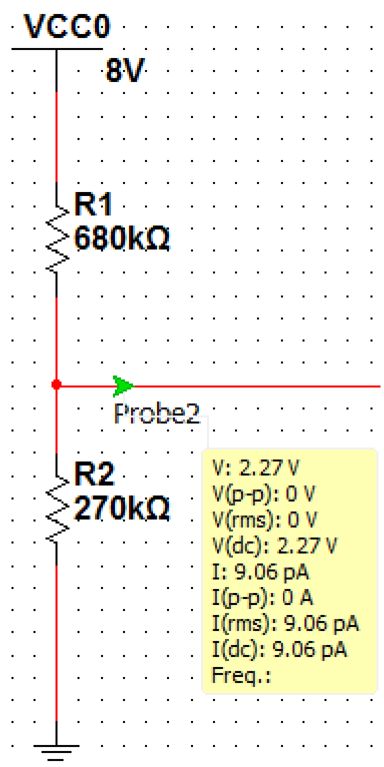

Figure 3-2-2-1-2: Input Attenuator for 2V/Div

3-2-2-3 Input Attenuator for 5V/Div

For 5V/Div, the scope has to support maximum 20V which may harm ADC and microcontroller. To design the input attenuator, 680kΩ resistor is used as R1 and 100kΩ resistor is used as R2. The 20V input will become 2.56V after pass through the circuit. Figure below shows the input attenuator for 5V/Div.

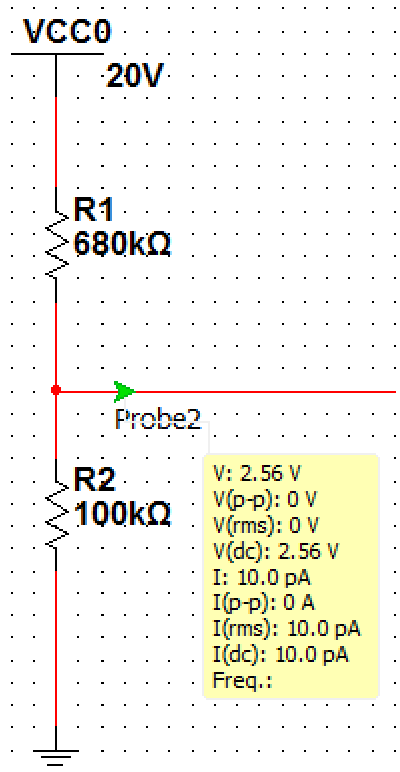

Figure 3-2-2-1-3: Input Attenuator for 5V/Div

3-2-2-3 Input Attenuator for 10V/Div

For 10V/Div, the scope has to support maximum 40V. To design the input attenuator, 680kΩ resistor is used as R1 and 51kΩ resistor is used as R2. The 40V input will become 2.79V after pass through the circuit. Figure below shows the input attenuator for 10V/Div.

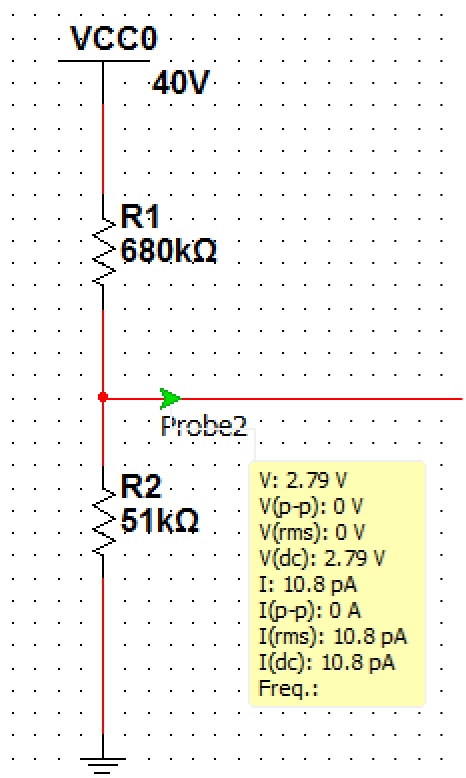

Figure 3-2-2-1-4: Input Attenuator for 10V/Div

3-2-3 Analog Multiplexer

Since this design has four different inputs which consist of different input range, a multiplexer is required in this design. A multiplexer is a device which enable user to select from several input signals and transmit either one of output signal. In this design, an analog multiplexer is used instead of digital multiplexer because the input signals are positive and negative voltage, where these signals can't be drive through digital multiplexer.

Analog multiplexer in this design is Intersil HI3-0508-5Z, this multiplexer consist of 8 analog switches and a digital decoder circuit for channel selection. To control the multiplexer supply at least 2.4V for Logic Level high and supply below 0.8V for Logic Level low. Supply minimum ±10V or Single 20V to power up this analog multiplexer (Texas Instruments, 2016). Figure below shows the top view of the HI3-0508-5Z IC.

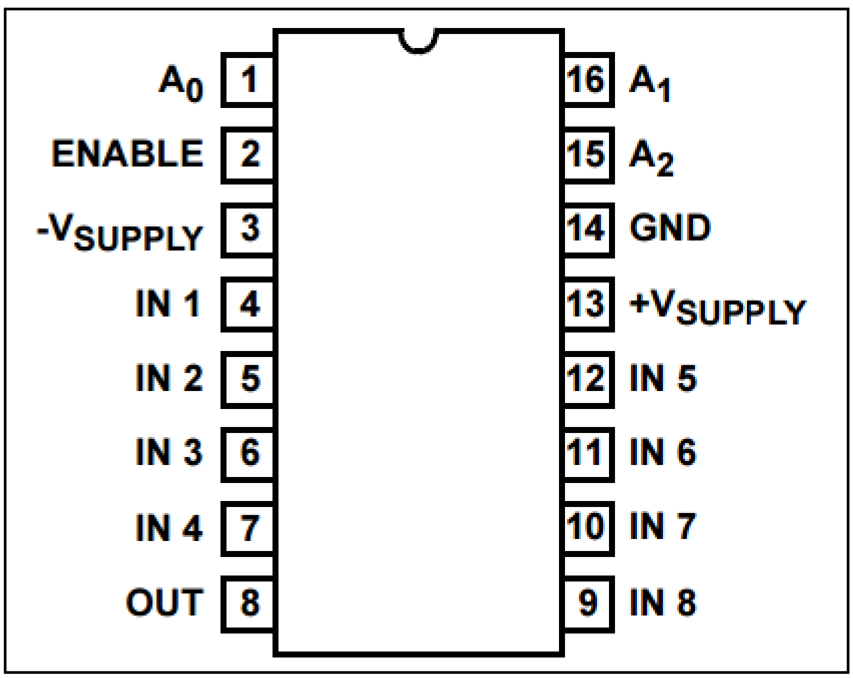

Figure 3-2-3-1: Top view of HI3-0508-5Z

To control which input channel as the output, functional diagram and truth table of HI3-0508-5Z are shown as below.

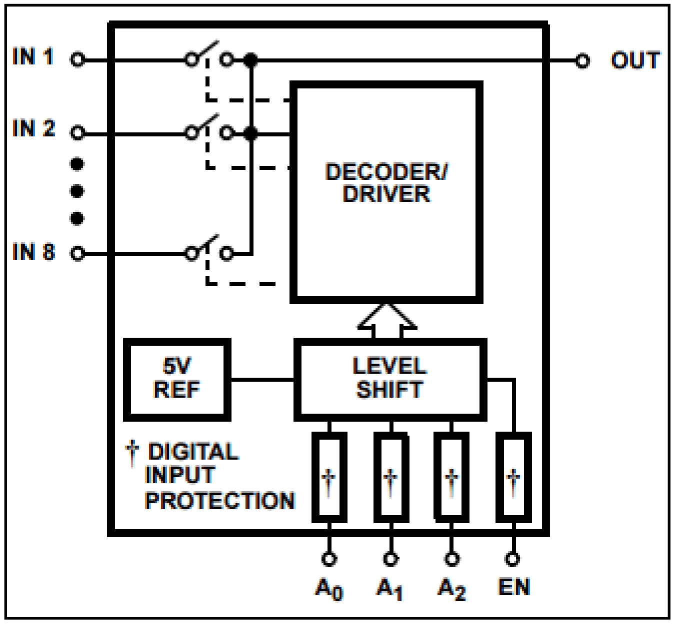

Figure 3-2-3-2: Functional Diagram of HI3-0508-5Z

A_2	A_1	A_0	EN	"ON" CHANNEL
X	X	X	L	None
L	L	L	H	1
L	L	H	H	2
L	H	L	H	3
L	H	H	H	4
H	L	L	H	5
H	L	H	H	6
H	H	L	H	7
H	H	H	H	8

Table 3-2-3-1: Truth table of HI3-0508-5Z

3-2-4 Voltage Level Shifter

As mention earlier, ADC only able to sample 0V to 3.3V which means it is not able to sample negative voltage. A non-inverting Op-Amp level shifter circuit is implemented convert a positive to negative signal into a range suitable for ADC.

Op-Amp used in this design is TLV314 which this is a low power and low noise operational amplifier. The Op-Amp has to be low noise to not affect the input signal into oscilloscope, if the input contains a lot of noise the sampling result will wrong. Figure below shows the circuit of level shifter.

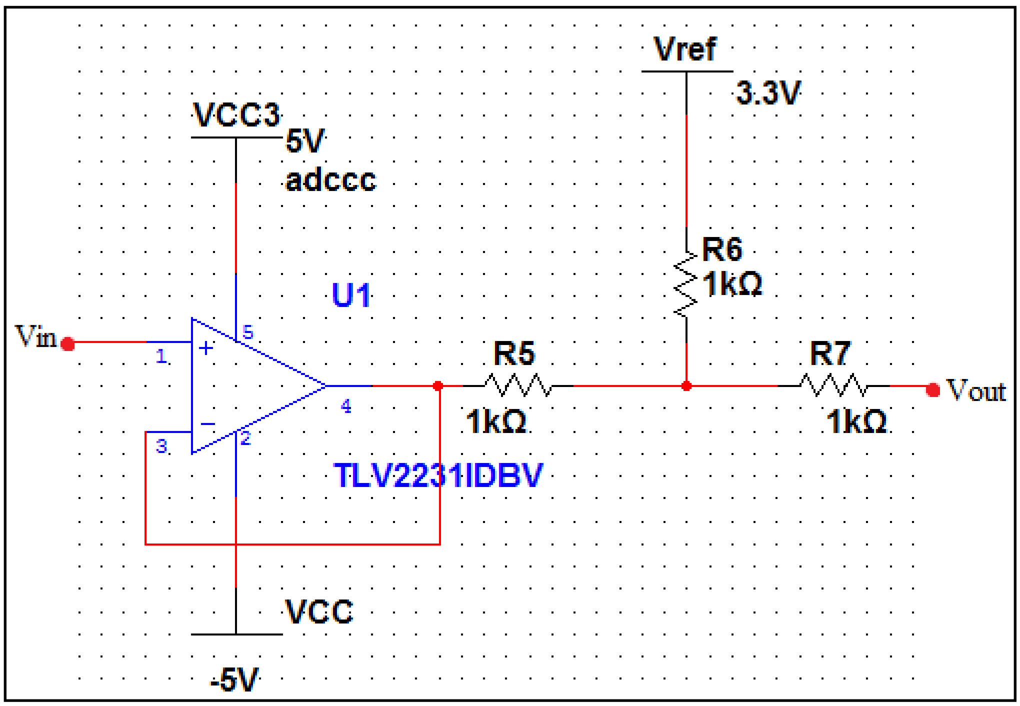

Figure 3-2-4-1: Circuit of level shifter

A simulation had been done to test the result of the level shifter when supply negative voltage and positive to it. Figures below show the result of apply -20V into the level shifter circuit and get output of 369mV. On the other hand, input 20V and get output of 2.97V, where these output is able to be sample by ADC.

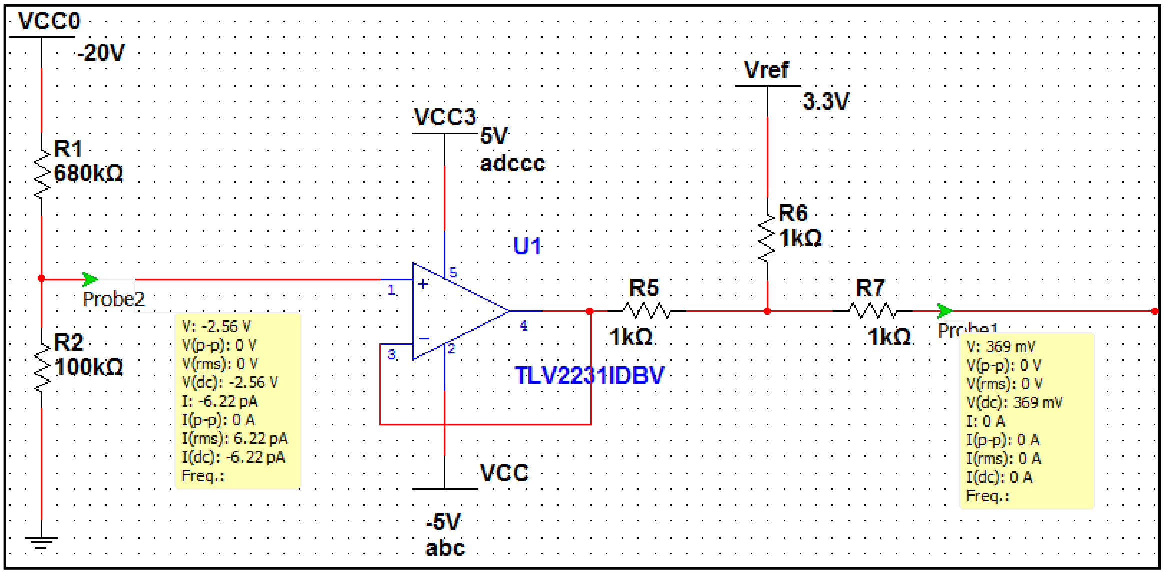

Figure 3-2-4-2: Simulation result of -20V to level shifter

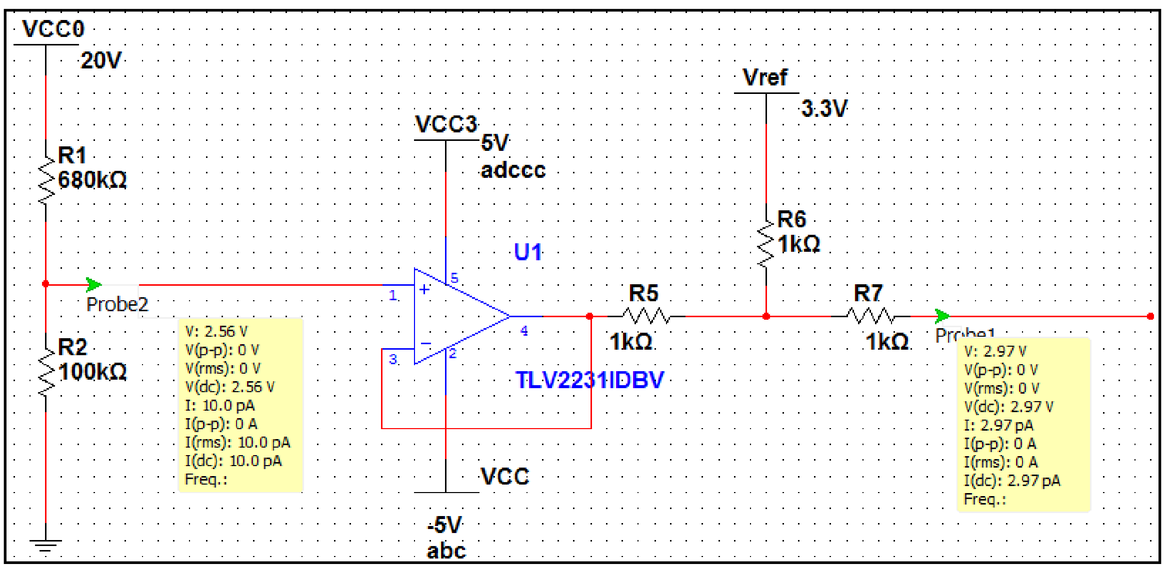

Figure 3-2-4-3: Simulation result of 20V to level shifter

3-2-5 Voltage Inverter Circuit

To power up operational amplifier TLV314 is required +5V and -5V. One of the thing about STM32F407VG microcontroller is that it provides a +5V power source. However, op-amp TLV314 require negative voltage source to function properly. Since the product in this project is focusing on handheld device, therefore implementation of voltage inverter circuit is required to avoid outsource to external power supply for-5V. The ICL7660 IC that acts as a voltage inverter is used to generate -5V.

All we need for this circuit is only ICL7660 and two 10uF capacitors. The circuit for this voltage inverter is shown as below.

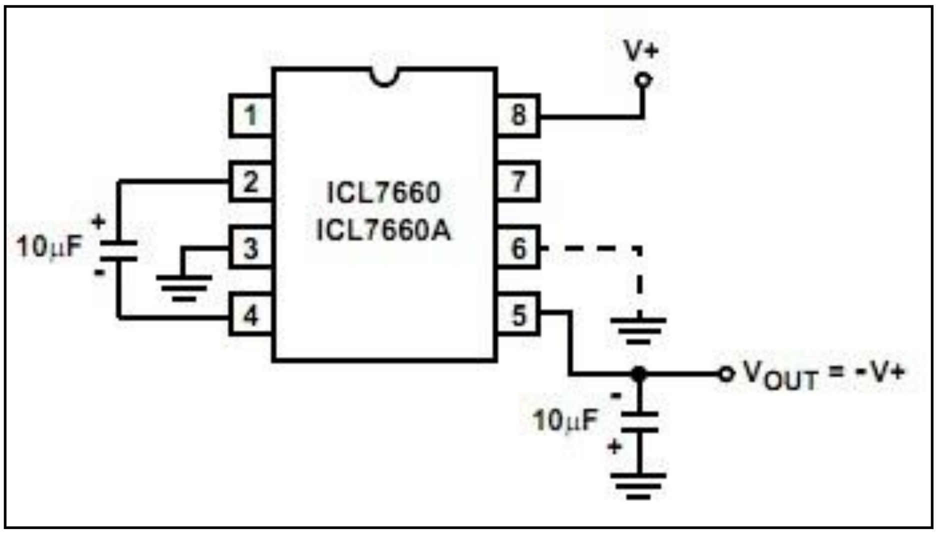

Figure 3-2-5-1: Voltage Inverter Circuit

3-3 Hardware Implementation for Multimeter

3-3-1 Current Measurement Circuit

A simple voltage divider circuit is used to measure current as shown in figure below.

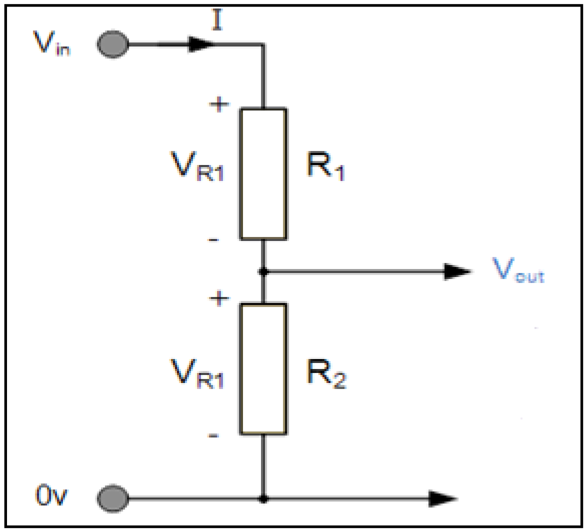

Figure 3-3-1-1: Current Measurement Circuit

Resistor R1 is the load connected to the circuit where resistor R2 represents the internal resistance of an ammeter. In this design, value of R2 is 10Ω. The section of the circuit after point Vout will be modelling as an ammeter.

3-3-2 Resistance Measurement Circuit

The Wheatstone bridge is build to measure the resistance. 3.3V is used for voltage source. When the bridge is fully balanced, the right side resistors identical to the left side resistor, which means R1=R3 and R2=Rx, the voltage across the bridge is zero (Daycounter, 2016). Hence value Rx can be calculated. Figure below shows the circuit of Wheatstone bridge.

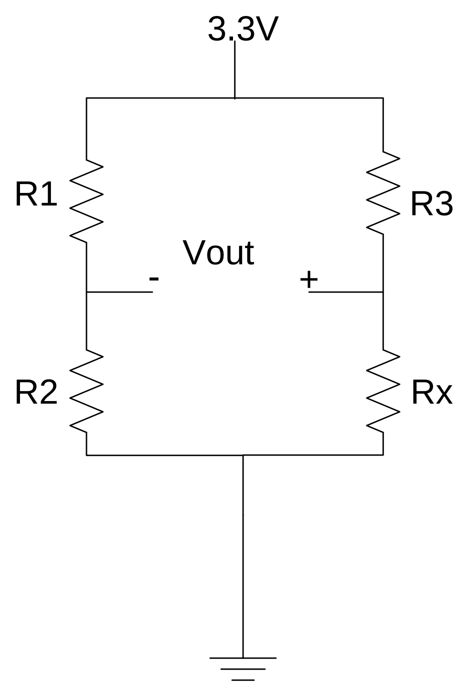

Figure 3-3-2-1: Wheatstone Bridge

CHAPTER 4 FIRMWARE ARCHITECTURE

4-1 Programming tools

The IDE used to write the program code and compile the code to machine code is Coocox CoIDE. CoIDE is a free integrated development environment focusing on ARM Cortex-M0/M0+/M3/M4 based microcontrollers. ARM GCC act as the compiler since it is from ARM official. All the program code will be written in C language where C language is more near to machine code and supported by CoIDE.

After compile the code, we can directly flash the program code into the microcontroller through CoIDE.

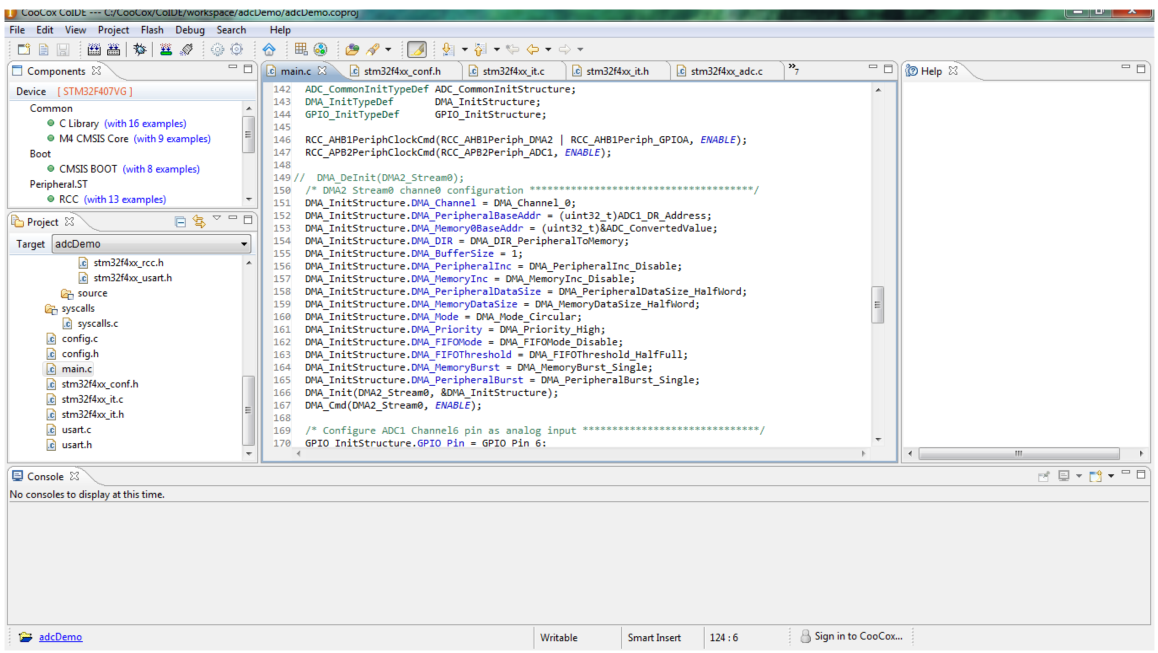

Figure 4-1-1: Coocox CoIDE

4-2 Microcontroller System Clock Configuration

System clock configuration is very important before start writing the firmware because slower clock speed may cause the firmware executes slower. ADC's clock for this microcontroller is generated from the APB2 clock and the general purpose timer (TIM2 to TIM5) clock source is generated from APB1 clock.

By default the configuration uses the internal RC oscillator which provides a system clock time of 16MHz, where this can be configured as high as 168MHz for STM32F407VG. To easily configure the clock configuration, ST had created a clock configuration tool in an excel file. The excel file will automatic generate the source file (system_stm32f4xx.c). This tool can be downloaded from ST website where the part number is stsw-stm32091. Figure below shows the default system clock setting in the excel file.

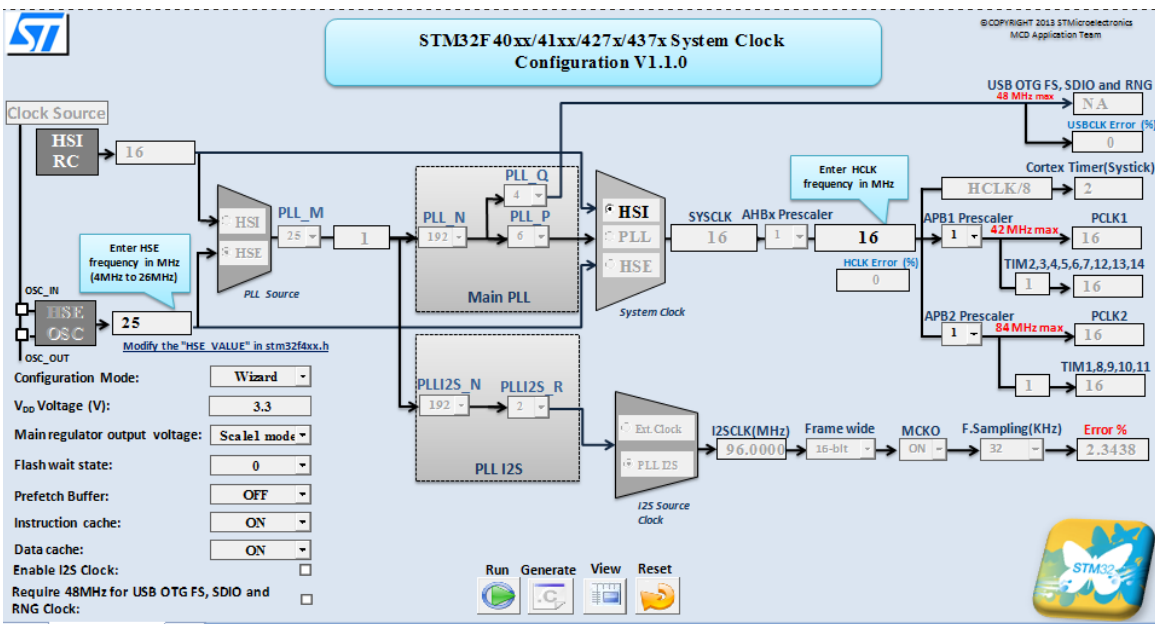

Figure 4-2-1: Before setup system clock configuration

Below are the steps to configure the system clock to 168MHz and set the APB1, APB2 clock to their maximum voltage.

In the bottom left corner set the following:

Configuration Mode = Expert

VDD = 3.3

Main Regulator Output Voltage = Scale1 Mode

Flash wait state = 5

Prefetch Buffer = Off

Instruction Cache = On

Data Cache = On

Uncheck Enable I2S Clock

In the graphical portion set the following:

HSE OSC = 8MHz

Check HSE as the PLL Source

PLL_M = 8

PLL_N = 336

PPL_Q = 7

PPL_P = 2

AHBx Prescaler = 1

APB1 Prescaler = 4

APB2 Prescaler = 2

After you done all the settings configured, press the "Generate" button at the bottom of the excel file then it will generate the system_stm32f4xx.c in the same folder of the excel file. Replace this system_stm32f4xx.c with your file in the project folder. Also have to change the HSE_VALUE in stm32f4xx.h, u can see it defaults as 25MHz (25000000). Change this value to 8MHz (8000000) or else USART will not work because USART clock is depends on HSE Oscillator. Figure 4-2-2 shows the clock speed after the configuration.

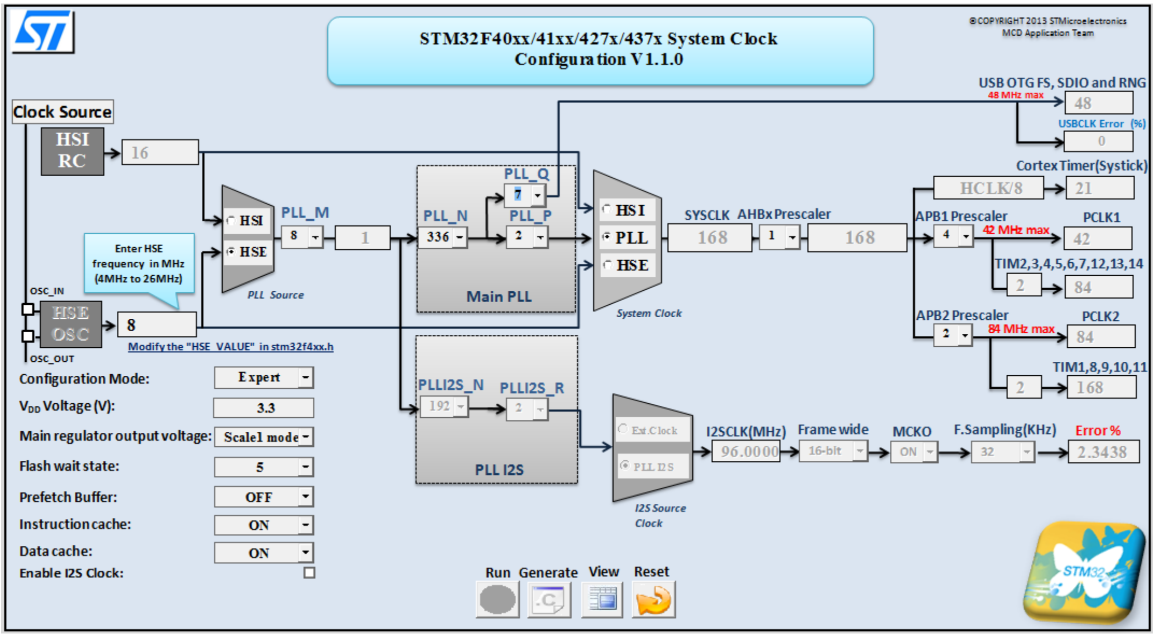

Figure 4-2-2: Clock Speed after Configuration

There is a function in the startup_stm32f4xx.c file defined as below:

//extern void SystemInit(void);

This *SystemInit()* function is commented out and never called. Uncomment this line of code to use the clock configuration that generated as mention above. Then add a line of code to call the *SystemInit()* under the *Default_Reset_Handler* function in the same start_stm32f4xx.c file. The system clock will be configured to 168MHz after all these changes.

4-3 Firmware Implementation of Oscilloscope

4-3-1 Flowchart of Oscilloscope Firmware

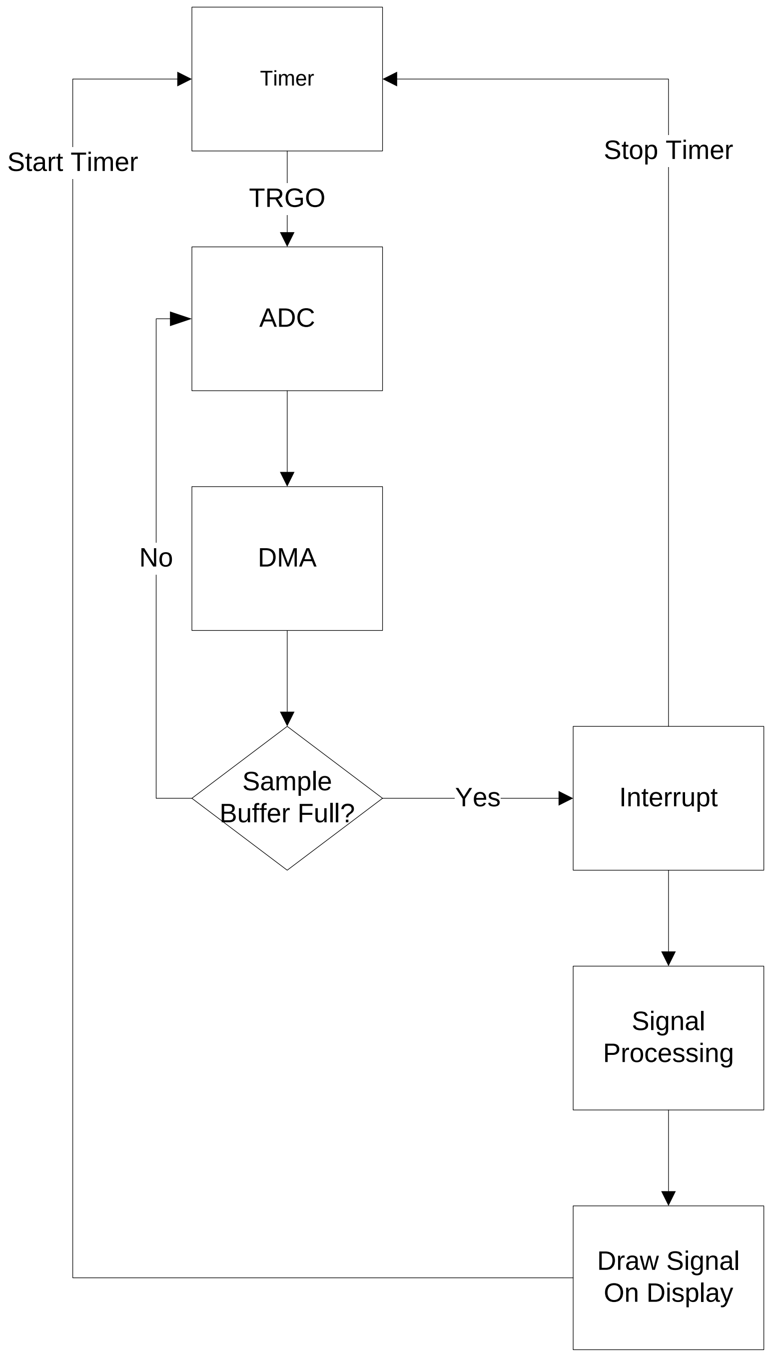

Figure 4-3-1-1: Flowchart of Oscilloscope Firmware

4-3-2 Signal Chain - Sequential Equivalent Time Sampling

Sequential equivalent time sampling means the system will acquires one sample per timer trigger, where timer trigger is dependent on the time/div setting. Sequential equivalent time sampling is illustrated in Figure 4-3-1.

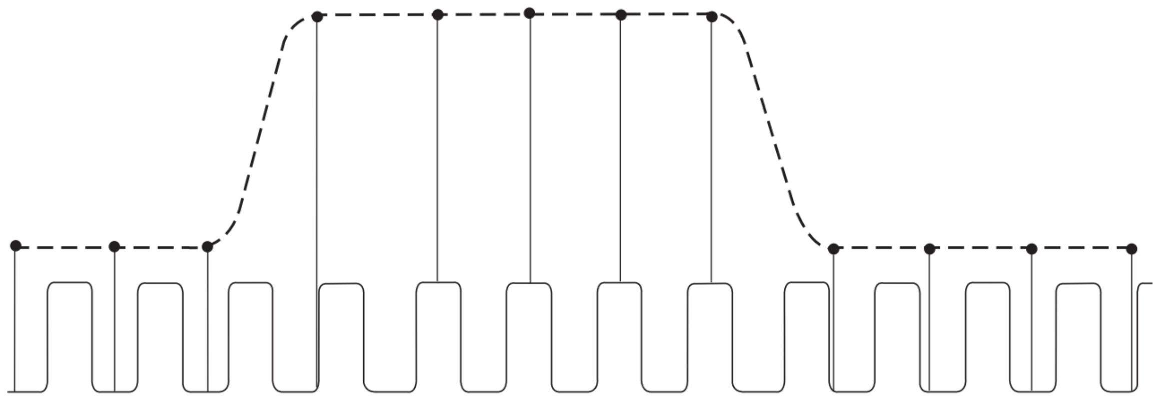

Figure 4-3-2-1 Sequential Equivalent time Sampling

In this project, the timer will trigger the ADC to sample the signal where ADC pre-fixed in continuous mode. Continuous mode allows the ADC to clock the DMA, DMA filling the buffer of sample. Due to RAM constraints onboard the microcontroller, fixed size of the signal sample buffer to 1000.

4-3-3 Timer Subsystem

General -purpose timer (TIM2) is used to producing the correct time for ADC to sample the signal. TIM2 clock speed is equal to the APB2 clock as illustrated in Figure 4-5-2 above, where it is 84MHz after system configuration. TIM2 is a 32-bit up, down, up/down auto-reloaded counter. TIM2 having 16-bit prescaler used to divide the counter clock frequency by any factor between 1 and 65536.

From the STM32F4 reference manual (STMicroelectronic, n.d.), TIM2 synchronize with ADC conversion, hence the ADCs could be triggered by TIM2 timer. To enable this timer synchronization, adding a line of code during initialize TIM2 to trigger TIM_TRGO. When TIM_TRGO event occurs, it will activates the ADC-DMA subsystem to fills the sample buffer. The code to trigger TRGO defined as below:

TIM_SelectOutputTrigger(TIM2, TIM_TRGOSource_Update);

The next step is to program the timer so that it will periodically generate TRGO signals to trigger ADC to sampling the signal. To program the timer, need to calculate the suitable prescaler and period for the timer configuration by following formula:

$$Timer\ TRGO\ event = \frac{TIM2\ CLOCK\ SPEED}{((Prescaler + 1)(Period + 1\ ))}$$

4-3-4 ADC-DMA Subsystem

This subsystem is to let ADC captures samples and DMA will places them in a buffer, later the signal buffer will go through signal processing.

4-3-4-1 ADC

Only one channel of ADC is being used in this design because the oscilloscope only single channel, where the ADC is ADC1 regular channel 6. ADC is set to 8-bit resolution where the ADC value is from 0 to 255 due to the 8-bit ADC can shorten the conversion time of ADC. The clock speed ADC is same as PCLK of the microcontroller which is 42MHz. During configuration, it is required to set the ADC prescaler to *ADC_Prescaler_Div2* which change the ADC clock speed to 21MHz.

One of the important things that need to mention is that the conversion time of ADC to convert analog signal to digital value. While configure the ADC, set the ADC sample time to 3 cycles as defined as below:

ADC_RegularChannelConfig(ADC1, ADC_Channel_6, 1, ADC_SampleTime_3Cycle);

Configuration about set the ADC conversion time become 11 cycles (3 cycle sample time plus 8 bit data). This configuration is important because it will affect the ADC conversion time for sampling data. With this setting the total conversion time for one ADC sample is 262ns calculated by this formula $\frac{1}{ADC\ ClockSpeed\ /11}$.

In order to let ADC synchronize with timer TRGO, following code have to be written during ADC initialization else ADC will trigger by ADC own clock speed.

```
ADC_InitStructure.ADC_ExternalTrigConv = ADC_ExternalTrigConv_T2_TRGO;
ADC_InitStructure.ADC_ExternalTrigConvEdge = ADC_ExternalTrigConvEdge_Rising;
```

Figure 4-3-4-1-1: Code to Synchronize ADC with Timer

Figure below shows the code to initialize the ADC, which start with initialize a GPIO pin as analog pin. Then initialize the ADC on that specific GPIO pin.

```
void ADC_GPIO_Init(void)
{
    ADC_InitTypeDef       ADC_InitStructure;
    ADC_CommonInitTypeDef ADC_CommonInitStructure;
    GPIO_InitTypeDef      GPIO_InitStructure;

    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOA, ENABLE);
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_ADC1, ENABLE);

    //Configure ADC1 Channel6 pin as analog input ******************************
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_6;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AN;
    GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL ;
    GPIO_Init(GPIOA, &GPIO_InitStructure);

    //ADC Common Init **********************************************************
    ADC_DeInit();
    ADC_CommonInitStructure.ADC_Mode = ADC_Mode_Independent;
    ADC_CommonInitStructure.ADC_Prescaler = ADC_Prescaler_Div2;
    ADC_CommonInitStructure.ADC_DMAAccessMode = ADC_DMAAccessMode_Disabled;
    ADC_CommonInitStructure.ADC_TwoSamplingDelay = ADC_TwoSamplingDelay_5Cycles;
    ADC_CommonInit(&ADC_CommonInitStructure);

    // ADC1 Init ****************************************************************
      ADC_InitStructure.ADC_Resolution = ADC_Resolution_8b;
      ADC_InitStructure.ADC_ScanConvMode = DISABLE;
      ADC_InitStructure.ADC_ContinuousConvMode = DISABLE;
      ADC_InitStructure.ADC_ExternalTrigConv = ADC_ExternalTrigConv_T2_TRGO;
      ADC_InitStructure.ADC_ExternalTrigConvEdge = ADC_ExternalTrigConvEdge_Rising;
      ADC_InitStructure.ADC_DataAlign = ADC_DataAlign_Right;
      ADC_InitStructure.ADC_NbrOfConversion = 1;
      ADC_Init(ADC1, &ADC_InitStructure);

      //ADC1 regular channe6 configuration *************************************
      ADC_RegularChannelConfig(ADC1, ADC_Channel_6, 1, ADC_SampleTime_3Cycles);
}
```

Figure 4-3-4-1-2: GPIO & ADC Initialization

4-3-4-2 DMA

In microcontroller project there are always a lot of read and write process. For example, reading data from peripheral unit just like ADC and writing values to RAM. This process normally is done by processor and causes loose a significant amount of processing time. Direct Memory Access (DMA) unit is use to avoid occupying processor by to do the process of data transfer between memory location without need of processor. DMA can do automated peripheral to memory transfer so it is used in this project.

In this project, DMA2 channel 0 is selected. To configure the DMA, first have specifies the DMA transfer data from peripheral to memory. Next is to get the memory address of ADC1. By referring to the STM32F4 datasheet, ADC1 memory address is 0x4001204C. Configure the ADC1 memory address in the *DMA_PeripheralBaseAddr* so that DMA know where to get the ADC value. Initialize the signal buffer, where the size of signal buffer in this project is 1000. Then specify the memory address as the address of the signal buffer and it is important to enable the *DMA_MemoryInc* which it specifies whether the memory address register should be incremented or not. If this option is not being enabled, DMA only will transfer the data from ADC to the first location of the signal buffer. Then configure DMA channel to transfer data in circular buffer, which means after 1000 data being transferred to the signal buffer, the DMA will point back to the base address of the signal buffer for the next 1000 data transfer. After that, load all these values using *DMA_Init(DMA2_Stream0, &DMA_InitStructure).* Using *DMA_Cmd(DMA2_Stream0, ENABLE)* command to fire the DMA. Figure 4-3-4-2-1 shows the code of configure DMA.

```
void ADC1_CH6_DMA_Init(void)
{
    DMA_InitTypeDef DMA_InitStructure;

    //Clock Enable
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_DMA2, ENABLE);

    //DMA Disable
    DMA_Cmd(DMA2_Stream0, DISABLE);

    DMA_DeInit(DMA2_Stream0);

    //DMA2 Stream0 Channel0 Config
    DMA_InitStructure.DMA_Channel = DMA_Channel_0;
    DMA_InitStructure.DMA_PeripheralBaseAddr = (uint32_t)ADC1_DR_Address;
    DMA_InitStructure.DMA_Memory0BaseAddr = (uint32_t)&ADC_DMA_Buffer;
    DMA_InitStructure.DMA_DIR = DMA_DIR_PeripheralToMemory;
    DMA_InitStructure.DMA_BufferSize = ADC_BUFFER_SIZE;
    DMA_InitStructure.DMA_PeripheralInc = DMA_PeripheralInc_Disable;
    DMA_InitStructure.DMA_MemoryInc = DMA_MemoryInc_Enable;
    DMA_InitStructure.DMA_PeripheralDataSize = DMA_PeripheralDataSize_Byte;
    DMA_InitStructure.DMA_MemoryDataSize = DMA_MemoryDataSize_Byte;
    DMA_InitStructure.DMA_Mode = DMA_Mode_Circular;
    DMA_InitStructure.DMA_Priority = DMA_Priority_High;
    DMA_InitStructure.DMA_FIFOMode = DMA_FIFOMode_Disable;
    DMA_InitStructure.DMA_FIFOThreshold = DMA_FIFOThreshold_HalfFull;
    DMA_InitStructure.DMA_MemoryBurst = DMA_MemoryBurst_Single;
    DMA_InitStructure.DMA_PeripheralBurst = DMA_PeripheralBurst_Single;
    DMA_Init(DMA2_Stream0, &DMA_InitStructure);

    DMA_Cmd(DMA2_Stream0, ENABLE);
}
```

Figure 4-3-4-2-1: Configuration of DMA2 Stream0 Channel0

In order to catch the end of DMA data transfer, initialize DMA transfer complete on channel0 interrupt. When the interrupt occur, then the interrupt handler will triggered and start the signal processing for the signal buffer.

4-3-4 Time base

Time base is the x-axis of the oscilloscope graph and represented as time-per-division (time/div). Time base acts as an important role in sampling the signal because every time/div setting will affect the speed of the ADC sampling. There are 17 time base being developed, which are 5s/Div, 2s/Div, 1s/Div, 500ms/Div, 200ms/Div, 100ms/Div, 50ms/Div, 20ms/Div, 10ms/Div, 5ms/Div, 2ms/Div, 1ms/Div, 500µs/Div, 250µs/Div, 100µs/Div, 50µs/Div and 25µs/Div.

There are 500 data will be drawn on the display into 10 division, and then each division will only be 50 data. Hence calculation of prescaler and period for the timer to trigger the ADC sampling is dependent on 50 data to be drawn in each division. Table below shows calculation of frequency of timer to sample signal.

Time/Div	**Time sample per data,** $\frac{Time}{50(pix\ per\ division)}$	**Frequency of timer to sample,** $F = \frac{1}{Time\ Sample\ per\ Data}$
5s/Div	100ms	10Hz
2s/Div	40ms	25Hz
1s/Div	20ms	50Hz
500ms/Div	10ms	100Hz
200ms/Div	4ms	250Hz
100ms/Div	2ms	500Hz
50ms/Div	1ms	1000Hz
20ms/Div	400µs	2500Hz
10ms/Div	200µs	5000Hz
5ms/Div	100µs	10000Hz
2ms/Div	40µs	25000Hz
1ms/Div	20µs	50000Hz
500µs/Div	10µs	100000Hz
250µs/Div	5µs	200000Hz
100µs/Div	2µs	500000Hz
50µs/Div	1µs	1000000Hz
25µs/Div	0.5µs	2000000Hz

Table 4-3-4-1: Frequency of timer to sample signal

From table 4-3-4-1 can shows that the smaller the time base, the faster the speed of timer to trigger ADC to sample and the highest speed of sampling is 2MHz. Table below shows that suitable prescaler and period for timer that being used in this project which fulfill the frequency of timer sample found in table 4-3-4-1.

Time/Div	**Prescaler**	**Period**	$\boldsymbol{Timer\ TRGO\ event} = \frac{\boldsymbol{TIM2\ CLOCK\ SPEED}}{((\boldsymbol{Prescaler} + \mathbf{1})(\boldsymbol{Period} + \mathbf{1}\ ))}$
5s/Div	41999	199	10Hz
2s/Div	41999	79	25Hz
1s/Div	41999	39	50Hz
500ms/Div	41999	19	100Hz
200ms/Div	20999	15	250Hz
100ms/Div	20999	7	500Hz
50ms/Div	4199	19	1000Hz
20ms/Div	8399	3	2500Hz
10ms/Div	839	19	5000Hz
5ms/Div	839	9	10000Hz
2ms/Div	419	7	25000Hz
1ms/Div	209	7	50000Hz
500µs/Div	83	9	100000Hz
250µs/Div	83	4	200000Hz
100µs/Div	41	3	500000Hz
50µs/Div	20	3	1000000Hz
25µs/Div	20	1	2000000Hz

Table 4-3-4-2: Prescaler and Period for Timer TRGO event

4-3-5 Utilize Memory Usage

There is a common problem of microcontroller based embedded system project will met which is the problem or extremely limited resource such as RAM. This microcontroller STM32F407VF has 2 block of SRAM of 128 Kbyte. For implementation of oscilloscope required a huge amount of memory due to huge size of sample buffer and signal processing array. In order to do Fast Fourier Transform feature, it required to use 768 of 32bit floating point array which that 2 block of SRAM is not sufficient.

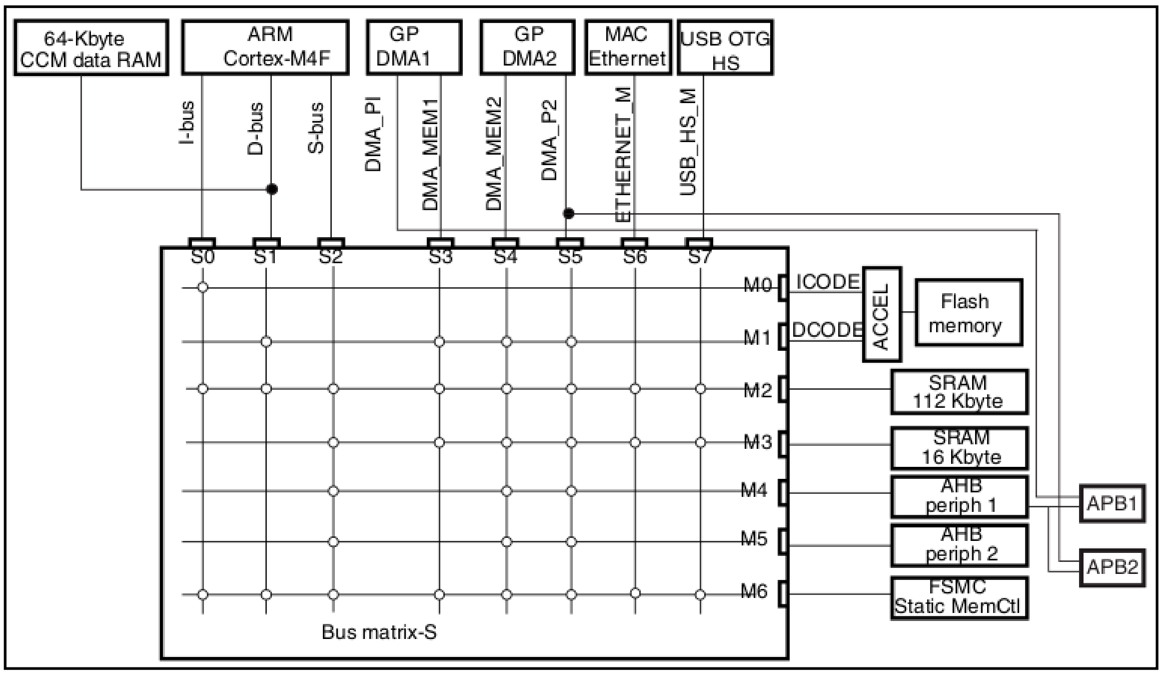

Figure 4-3-5-1: STM32F4 Memory Connection

However, inside the microcontroller has a 64Kbytes of Core Coupled Memory (CCM) block which is connected directly to the core. The CCM block is commonly used as the stack and handling critical OS data. Since this project is not using any OS, this CCM block could be used as an extra memory block to store data.

Following are the steps to load the initialization of the CCM memory board. First have to edit the linker script of the project. For Coocox project, linker script is named as arm-gcc-link.ld as default name which located in the top of the project folder. Figure below show the internal memory map of the linker script, where ram1 is the address of CCM.

```
MEMORY
{
    rom (rx)   : ORIGIN = 0x08000000, LENGTH = 0x00100000
    ram (rwx) : ORIGIN = 0x20000000, LENGTH = 0x00020000
    ram1 (rwx) : ORIGIN = 0x10000000, LENGTH = 0x00010000
}
```

Figure 4-3-5-2: Internal memory map of linker script

After the linker script is open, simply defining a section as highlighted in the linker script as figure below:

```
.ccm (NOLOAD) :
{
  . = ALIGN(4);
  *(.ccm)
  . = ALIGN(4);
} > ram1
```

Figure 4-3-5-3: Defining CCM in linker script

In order to let the variable allocate to CCM memory, a section attribute is used after declaration of the variable. Figure below shows the declaration of variable to allocate the variable into CCM memory.

```
float32_t fft_input[512] __attribute__ ((section(".ccm")));
float32_t fft_output[256] __attribute__ ((section(".ccm")));
```

Figure 4-3-5-4: Declaration of variable allocate to CCM memory

Declaration of variable has to be a global variable or else the compiler cannot compiler the project.

4-3-6 Digital Output Pin

As mention in earlier, to control the analog multiplexer required digital signal. In this microcontroller, GPIO is able to set to digital pin where user able to write output Low (0 volt) or High (3.3 volts) that able to control the analog multiplexer.

To configure the digital pin, first have to enable the GPIO clock where all the GPIO in this controller are on AHB1 bus. GPIOE port pin 3, 4, 5 and 6 are used due to no other peripheral is using them. Figure below shows the initialization code of digital pin.

```
void digital_pin_init(void)
{
    GPIO_InitTypeDef GPIO_InitStructure;
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOE, ENABLE);
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_3 | GPIO_Pin_4 | GPIO_Pin_5 | GPIO_Pin_6;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;
    GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
    GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
    GPIO_Init(GPIOE, &GPIO_InitStructure);
}
```

Figure 4-3-6-1: Digital Pin initialization

Call function *GPIO_SetBits(GPIOx, GPIO_Pin_x)* to write a output HIGH and function *GPIO_ResetBits(GPIOx, GPIO_Pin_x)* to write a output LOW.

4-4 Firmware Implementation of Multimeter

4-4-1 ADC

Accuracy of a digital multimeter is very important. Precision of the measure will be affect by resolution of ADC, so in this design 12-bit of ADC resolution is used to measure a better result. ADC2 is used to avoid conflict between ADC used in oscilloscope which is ADC1. Figure below shows the ADC initialization code for ADC2.

```
//Config ADC2 Channel5,7 pin as analog input
GPIO_InitStructure.GPIO_Pin = GPIO_Pin_5 | GPIO_Pin_7;
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AN;
GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL ;
GPIO_Init(GPIOA, &GPIO_InitStructure);

//ADC common Init
ADC_DeInit();
ADC_CommonInitStructure.ADC_Mode = ADC_Mode_Independent;
ADC_CommonInitStructure.ADC_Prescaler = ADC_Prescaler_Div2;
ADC_CommonInitStructure.ADC_DMAAccessMode = ADC_DMAAccessMode_Disabled;
ADC_CommonInitStructure.ADC_TwoSamplingDelay = ADC_TwoSamplingDelay_5Cycles;
ADC_CommonInit(&ADC_CommonInitStructure);

//ADC1 Init
ADC_InitStructure.ADC_Resolution = ADC_Resolution_12b;
ADC_InitStructure.ADC_ScanConvMode = ENABLE;
ADC_InitStructure.ADC_ContinuousConvMode = ENABLE;
ADC_InitStructure.ADC_ExternalTrigConvEdge = ADC_ExternalTrigConvEdge_Rising;
ADC_InitStructure.ADC_ExternalTrigConv = ADC_ExternalTrigConv_T3_TRGO;
ADC_InitStructure.ADC_DataAlign = ADC_DataAlign_Right;
ADC_InitStructure.ADC_NbrOfConversion = 2;
ADC_Init(ADC2, &ADC_InitStructure);

ADC_RegularChannelConfig(ADC2, ADC_Channel_5, 1, ADC_SampleTime_3Cycles);
ADC_RegularChannelConfig(ADC2, ADC_Channel_7, 2, ADC_SampleTime_3Cycles);
```

Figure 4-4-1-1: Initialization of ADC2 for multimeter

Two ADC channel have been initialized to measure the voltage output from Wheatstone bridge.

4-4-2 DMA

As mention above ADC for multimeter is different from ADC for oscilloscope, which means the DMA stream to handle the ADC-DMA request also different. Table below shows that DMA2 request mapping.

Peripheral requests	Stream 0	Stream 1	Stream 2	Stream 3	Stream 4	Stream 5	Stream 6	Stream 7
Channel 0	ADC1		TIM8_CH1 TIM8_CH2 TIM8_CH3		ADC1		TIM1_CH1 TIM1_CH2 TIM1_CH3	
Channel 1		DCMI	ADC2	ADC2				DCMI
Channel 2	ADC3	ADC3				CRYP_OUT	CRYP_IN	HASH_IN
Channel 3	SPI1_RX		SPI1_RX	SPI1_TX		SPI1_TX		
Channel 4			USART1_RX	SDIO		USART1_RX	SDIO	USART1_TX
Channel 5		USART6_RX	USART6_RX				USART6_TX	USART6_TX
Channel 6	TIM1_TRIG	TIM1_CH1	TIM1_CH2	TIM1_CH1	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_UP	TIM1_CH3	
Channel 7		TIM8_UP	TIM8_CH1	TIM8_CH2	TIM8_CH3			TIM8_CH4 TIM8_TRIG TIM8_COM

Table 4-4-2-1: DMA2 request mapping

As shown as above, DMA2 stream2 channel 1 is able to handle DMA request from ADC2, hence it is used for transfer ADC2 value to buffer.

4-6 Interrupt

Interrupts play an important role in microcontroller because interrupt is the only way to stop executing program and jump to execute code in the interrupt handler. The STM32 has a powerful interrupt controller which is Nested Vector Interrupt Controller (NVIC) to handle any types of interrupt. NVIC in this microcontroller supports up to 256 different interrupt vectors. In this design, there is an interrupt called each time when DMA finish fills the buffer.

CHAPTER 5 FUNCTION IMPLEMENTATION

5-1 Function for Oscilloscope

5-1-1 Voltage Division

A voltage waveform shows on the vertical axis. Vertical axis of this design is having four volts/div control to adjust suitable amplitude of the signal to the desired measurement range. The four volts/div are 1V/Div, 2V/Div, 5V/Div and 10V/Div. This oscilloscope having 4 positive vertical division and 4 negative vertical division. Hence, the total input supply of the signal can go up to positive/negative 40V. The input signal will pass through the attenuation and then signal will hold in an analog multiplexer. When 1V/Div is selected, the microcontroller will send a 3 bit digital signal to control the multiplexer let the V/Div signal into ADC to prevent the input signal damaging the ADC by feeding input signal more than 3.3V. Every volts/div also has different calculation of voltage due to different input attenuator.

5-1-1-1 Calculation for 1V/Div Division and Pixel to Draw Signal

Input	ADC Value
0V	131
5V	174
-5V	88

Table 5-1-1-1-1: ADC Value for 1V/Div

Table above shows the ADC value from different input voltage. There is a different of 43 between 0V to 5V and also 0V to -5V. With that information, an equation can be made to calculate the voltage as defined as:

$$V = \frac{ADC\ Value - 131}{43} * 5$$

After calculation of the Voltage, calculate which pixel to draw into screen. Figure below shows the information about the division and pixel in this project.

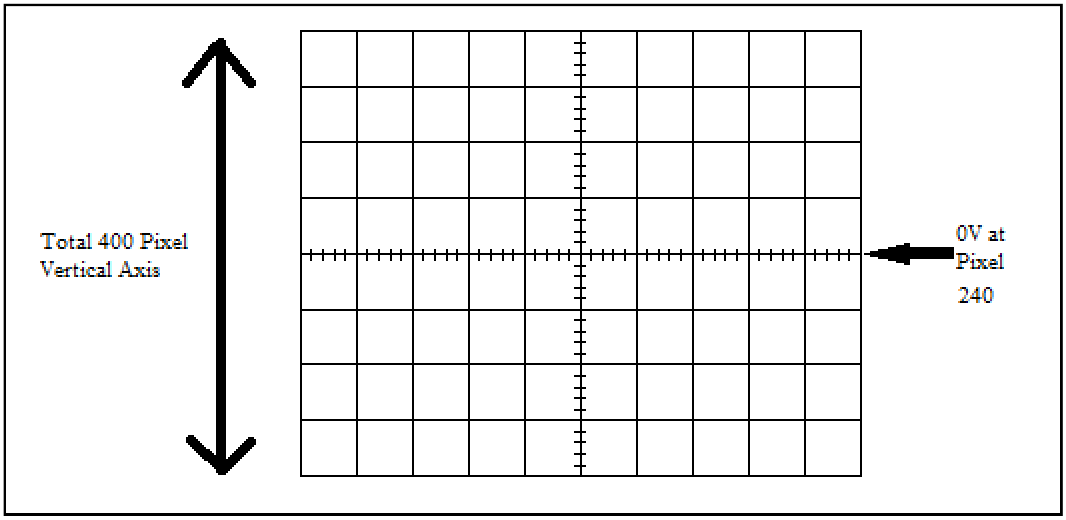

Figure 5-1-1-1-1: Relationship between Division box and Pixel User Interface

As mention as above that 0V is fixed at pixel 220. Each division has 50 pixels and one division represents 1V, so each pixel represent 0.02V. To plot the signal on the screen, calculate the pixel value through this formula:

$$Pixel\ to\ plot = 240 - \frac{Voltage}{0.02}$$

5-1-2 Software Trigger

5-1-2-1 Rising and Falling edge Trigger

Software trigger function of oscilloscope is devoted to stabilizing and focusing the repetitive waveform signal. This trigger function can be manipulated to keep the display of waveform static and unflinching as illustrated as figure below.

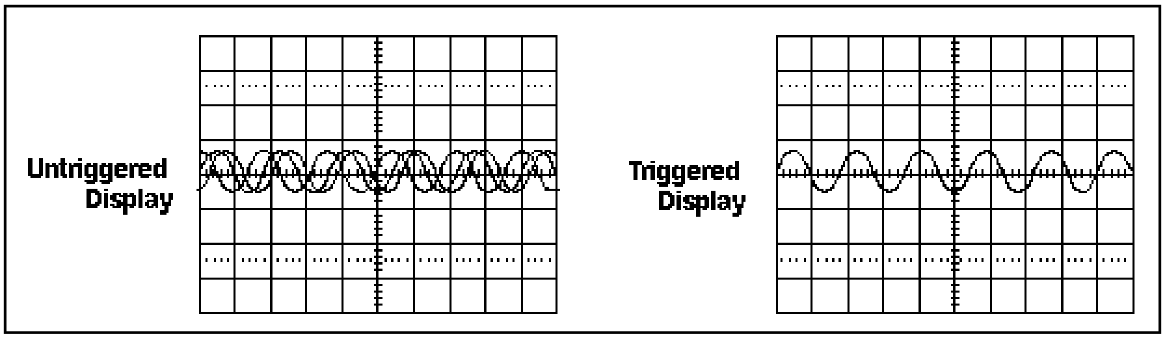

Figure 5-1-2-1: Comparison between Untriggered and Triggered Signal

A simple trigger function is developed which enable user to trigger the rising edge and falling edge of the signal. As mention above that this oscilloscope will sample 1000 signal buffer but only draw 500 data. This is due to avoid insufficient data to draw while implement trigger function because triggered signal could be happen in any time.

First a trigger line is implemented in the oscilloscope which allow user to choose any vertical axis to trigger the signal. The algorithm will check for the signal data before and after the trigger line to find out the triggered signal in between signal buffer 250 to 750. Take a rising edge trigger as an example, the algorithm check the data with condition of:

if (data before trigger line <= trigger line && data after trigger line > trigger line){

Found rising edge trigger signal

}

When trigger signal is found, the triggered signal buffer location is then recorded down as used as the starting point to draw the trigger signal. +-250 data from the triggered signal buffer is then drawn on the display. Figure below illustrated how the algorithm found the trigger signal by trigger line.

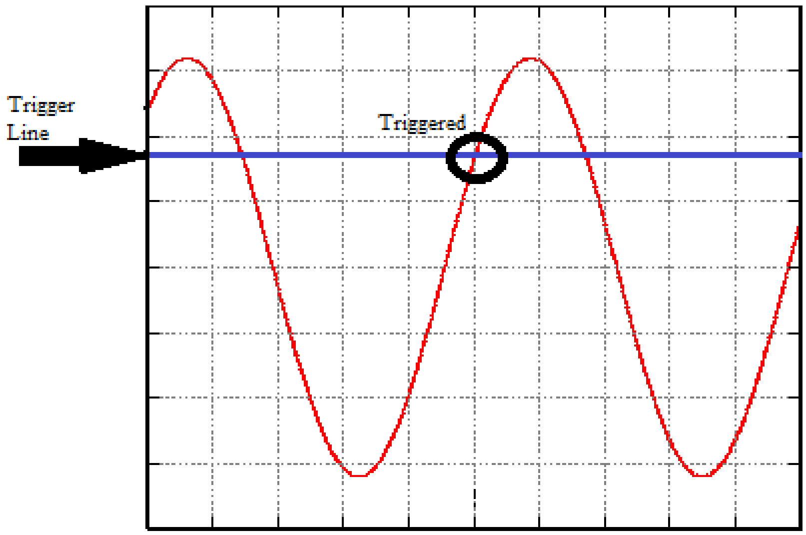

Figure 5-1-2-1-1: Rising edge trigger signal found

For falling edge trigger, the condition to check the trigger signal is:

if (data before trigger line >= trigger line && data after trigger line < trigger line){

Found falling edge trigger signal

}

5-1-2-2 Trigger Mode

The trigger mode determines whether or not the oscilloscope draws the waveform if it does not detect a trigger. In this project, there are 3 trigger modes implemented which is auto mode, normal mode and single mode.

Auto mode causes the oscilloscope to draw the signal even without a trigger. However, in normal mode the oscilloscope will not draw any signal if didn't find any trigger signal. Normal mode only draw triggered signal. Single mode is sampling the single in a single shot then it will not sampling again until user rerun it.

5-1-3 Period

Period is the total time for a wave to make one complete cycle. To calculate period, we detect the rising edge and falling edge of the waveform.

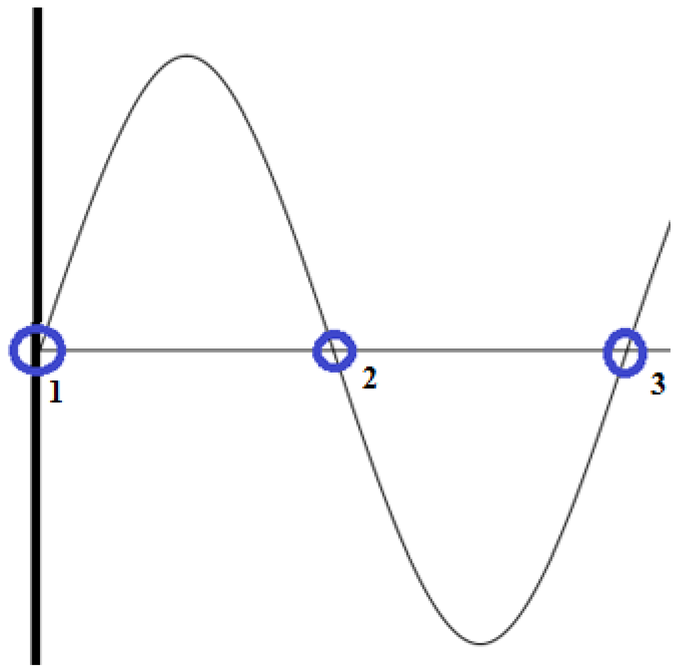

Figure 5-1-3-1: Detect Rising and falling edge

Edge counter is used to record down the rising edge and falling edge of the waveform. When a rising edge or falling edge is detected, counter++ and the x-axis pixel value will be record down. X-axis value will record down when counter = 1 and counter = 3. When counter = 3 mean that waveform had made one complete cycle and will calculate the period of the waveform by formula below:

$$Period = (third\ x\ value - first\ x\ value) \times time\ represent\ in\ single\ pixel\ of\ horizontal\ axis$$

Time represent in single pixel of horizontal axis is always change due to different time/div will affect the value of time in each pixel of horizontal axis.

5-1-4 Frequency

Frequency is the number of cycles per second. The standard unit of frequency is hertz (Hz). To calculate frequency of the signal, apply the formula $F = \frac{1}{Period}$, where period is as measured as mention in Chapter 5-1-3.

5-1-5 Duty Cycle

Duty cycle is the proportion of time which a signal is active. To calculate duty cycle, first check the wave is rising edge or falling edge. Start point and end point of the rising edge, and also start and end point of the falling edge will be record down. Next is to calculate the length of rising edge and length falling edge. With these values, calculate duty cycle by $DutyCycle = \frac{Lenth\ of\ Rising\ Edge}{Length\ of\ Rising\ Edge + Length\ of\ Falling\ Edge} \times 100.$

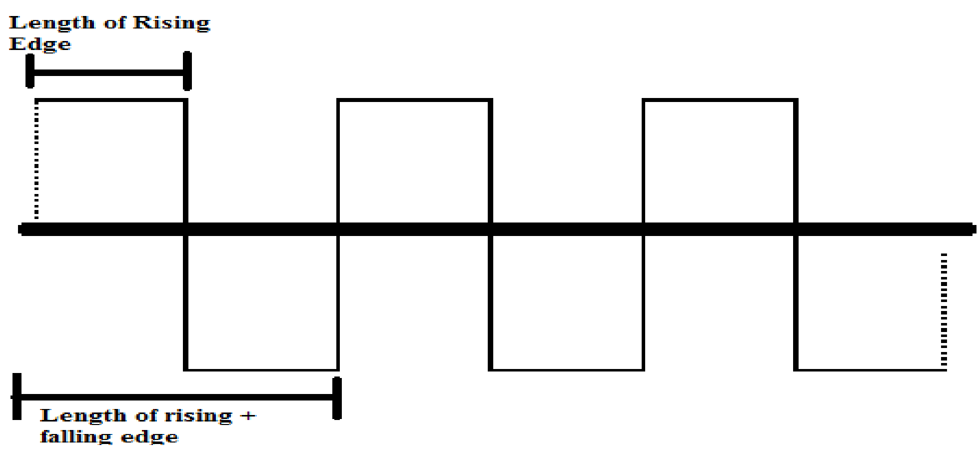

Figure 5-1-5-1: Length of Rising edge and falling edge

5-1-6 Maximum, Minimum, Peak-to-Peak Voltage

Maximum voltage (Vmax) is also called as peak voltage, is the highest voltage of the signal and minimum voltage (Vmin) is the lowest voltage of the signal.

Peak to peak voltage (Vpp) is the difference between maximum voltage and minimum voltage. The algorithm to calculate Vpp is Vpp = Vmax-Vmin. Therefore, calculation of Vmin and Vmax have to be done before calculate Vpp. Figure below shows the explanation of Vmax, Vmin, and Vpp.

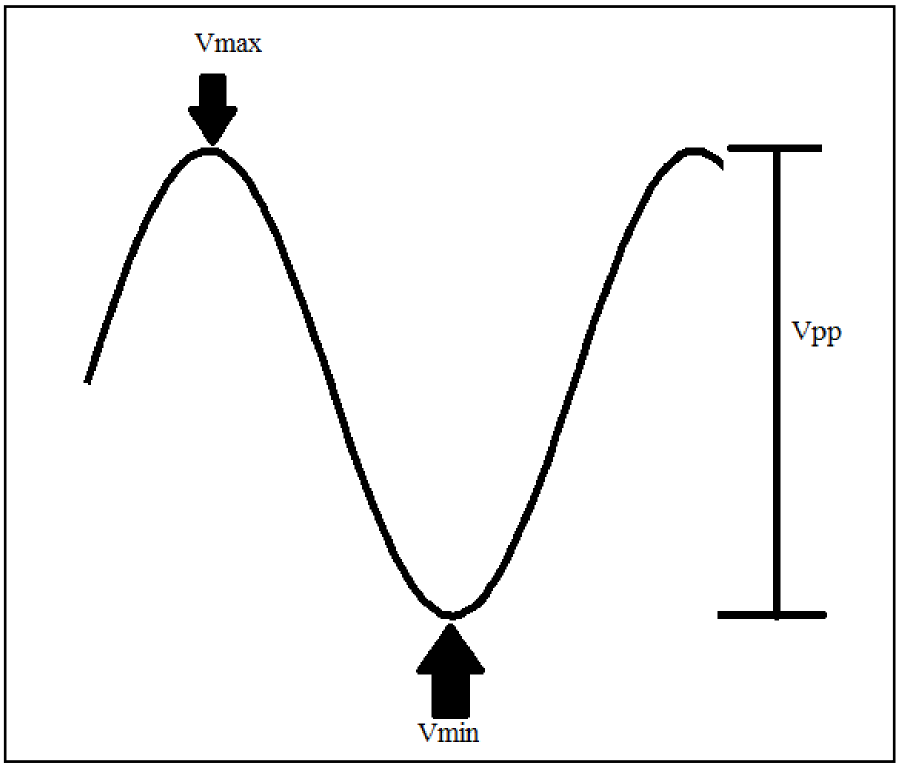

Figure 5-1-6-1: Explanation of Vmax, Vmin, and Vpp

5-1-7 Root-Mean-Square Voltage (Vrms)

Root-mean-square voltage is the value that indicated by the vast majority of AC voltmeter. Vrms is that value that when the voltage applied across a resistance, it produces the same amount of heat that a DC voltage of the same magnitude would produce.

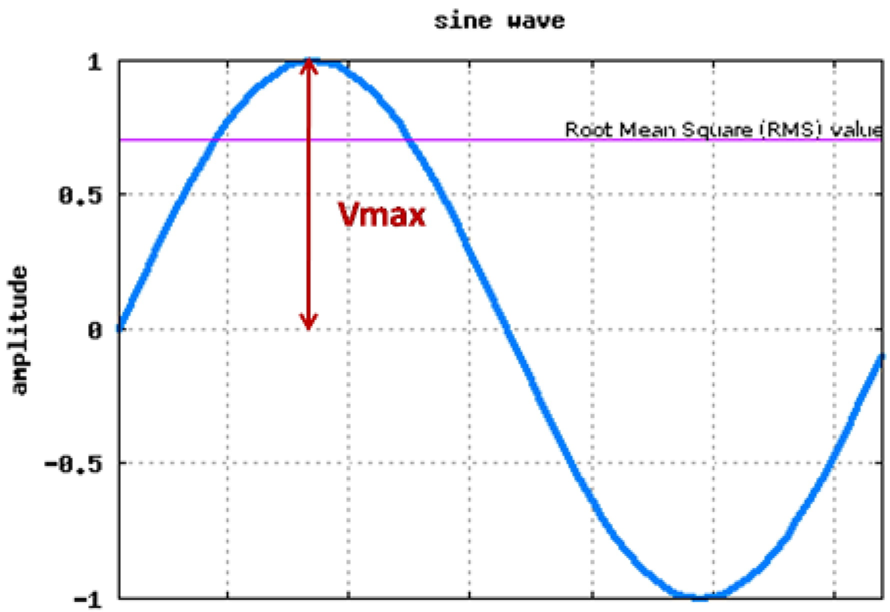

Figure 5-1-7: Explanation of Vrms

Following is the formula for Vmax to Vrms conversion.

$$Vrms = 0.707 \times Vmax$$

5-1-8 Average Voltage (Vavg)

Average voltage is calculated by the average of voltage in an appropriately chosen interval. Which also defined that is the quotient of the area under the waveform with respect to time. The formula to calculate average voltage is:

$$Vavg = \frac{\sqrt{V_1 + V_2 + \cdots + V_n}}{n}$$

5-1-9 Rise Time & Fall Time

Rise time is the time a signal takes to rise from 10 percent to 90 percent of the voltage between the low level and high level. On the other hand, fall time is the time a signal takes to fall from 90 percent to 10 percent of the voltage between high level and low level. Figure below shows the rise time and fall time of a signal.

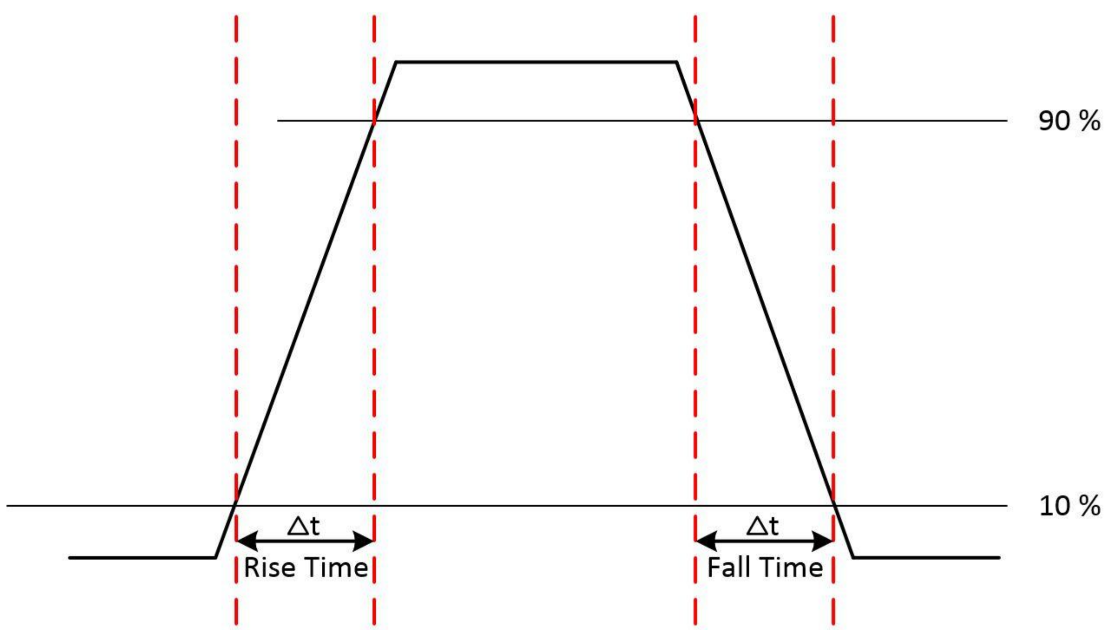

Figure 5-1-9-1: Rise time and fall time of a signal

To calculate the rise time and fall time, first use the peak to peak voltage divided by 10 to get the 10% voltage of the Vpp. To get the 10% rising voltage by adding 10% of Vpp with Vmin, and get the 90% rising voltage by Vmax minus the 10% of the Vpp.

Then get the x-axis pixel value of 10% voltage level and x-axis pixel value of 90% voltage level. Using this two x-axis values to get the total length of pixel from 10% to 90%. After getting the x-axis value, then rise time and fall time is then calculate by

$$Rise\ time = total\ length\ of\ pixel\ from\ 10\%\ to\ 90\% \times time\ represent\ of\ xaxis\ per\ pixel$$

$$Fall\ time = total\ length\ of\ pixel\ from\ 90\%\ to\ 10\% \times time\ represent\ of\ xaxis\ per\ pixel$$

5-1-10 Mean Voltage

Mean voltage is calculate by sum up the entire signal sampled, then the total sum of the signal divide by the sample size. The formula to calculate mean voltage is as below:

$$V_{mean} = \frac{Sum\ of\ Signals}{500}$$

5-1-11 Fast Fourier Transform

Fast Fourier Transform (FFT) is an algorithm to convert time domain signal into frequency domain signal. It is good to have FFT function in oscilloscope because engineer is able to observe distortion in signals using the FFT. Apart from that, FFT is also able to detect frequency of the signal.

STM32F4 is a microcontroller with DSP instructions. To implement FFT, first have to download the DSP libraries which contain mathematic and FFT libraries. There are only few libraries required to include instead of the whole DSP libraries, the required libraries are: arm_cfft_radix4_f32.c, arm_cfft_radix4_init_f32.c, arm_cmplx_mag_f32.c, arm_common_tables.c, arm_common_tables.h, arm_math.h, arm_max_f32.c, arm_rfft_f32.c and arm_rfft_init_f32.c. There is also another method to include the library which is link the arm_cortexM4lf_math.lib to the ARM compiler, which this .lib files contain all the function of the libraries mentioned above.

To use the FFT, first have to take number of samples which are power of 2. In this project, 512 samples are used as the input of FFT. After calculate the FFT of the 512 samples, only 256 samples will be valid to display which from result 0-255 are valid, result 256-511 are the same result as first half but in reverse order.

The number of samples will affect how good the FFT detect frequency; the more sample the more accurate. Let's take an example on the 5ms/Div in this project. To get proper frequency from signal, need at least 2 times sample from one period of highest frequency to detect. Hence 5ms/Div is sample with 10000Hz, then the largest frequency able to sample correct is 5000Hz. In this case, having 10000Hz sampling frequency and 512 samples, the resolution of the x-axis of FFT result is $10000Hz \div 512 = 19.53Hz$. For better understanding of the output, refer table 5-1-10-1 that show the interpret results from FFT output.

FFT Output	X-Axis(Frequency)	Description
Output[0]	0Hz	First parameter is always DC voltage in signal
Output[1]	1 × 19.53Hz	Amplitude of 19.53Hz frequency in signal
Output[n]	n × 19.53Hz	n-th value of frequency result
Output[255]	255 × 19.53Hz = ~5000Hz	Largest frequency able to sample

Table 5-1-11-1: Interpret FFT result

Figure below shows the code to calculate the FFT of the signals.

```
void calculate_fft()
{
    arm_cfft_radix4_instance_f32 S;
    uint16_t count = 0;
    for(uint32_t i = 500-256; i < 500+256; i++)
    {
        fft_input[(uint16_t)count] = (float32_t)((float32_t)ADC_DMA_Buffer[i] - 128);
        count++;
    }
    for(uint32_t i = 0; i < 512; i+=2)
    {
        fft_input[(uint16_t)(i+1)] = 0;
    }

    //Initialize CFFT module, intFlag = 0, doBitReverse =1
    arm_cfft_radix4_init_f32(&S, 256, 0, 1);

    //Process data through CFFT module
    arm_cfft_radix4_f32(&S, fft_input);

    //Process the data through the Complex Magniture Module for calculating the magnitude at each bin
    arm_cmplx_mag_f32(fft_input, fft_output, 256);

    //Calculates maxValue and returns corresponding value
    arm_max_f32(fft_output, 256, &fft_maxValue, &fft_maxIndex);

    fft_output[0] = 0;
    //Display on LCD
    for(uint16_t i = 0; i < 256/2; i++)
    {
        draw_fft_bar(31 + 2 * i, 439, 120, (uint16_t)fft_maxValue,(float32_t)fft_output[i]);
    }
```

Figure 5-1-11-1: Code to calculate FFT

5-2 Function for Multimeter

5-2-1 Voltage Measurement

Precise measurement is important for multimeter, so 12-bit ADC is used. The total level of the ADC value that can be mapped are 2^{12}=4096(0~4095). The voltage measurement can be done by dividing the ADC value read with its maximum level, and multiplied with 3.3V, since the STM32F4 ADC only support up to 3.3V input.

$$Voltage = \frac{ADC\ Value}{4096} x3.3V$$

5-2-2 Current Measurement

To measure current of a circuit, an ammeter must be connected in series to a resistor in a circuit. For current measurement, simple voltage divider rule is used.

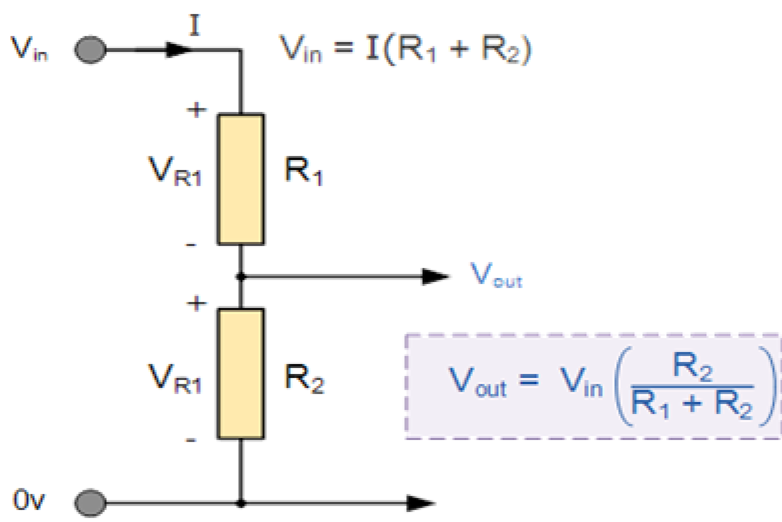

Figure 5-2-2-1: Circuit to measure current

The above figure shows the circuit used to measure current. Resistor R1 will represent the load connected to the circuit. Resistor R2 will represent the internal resistance of an ammeter. By obeying voltage divider rule, Vout will be the voltage across resistor R2. The section of the circuit after point Vout will be modelling an ammeter. Since current flow across a series circuit remains constant at any point, current for circuit can be obtained by dividing Vout with the value of R2 since Ohm's Law stat that $I = \frac{V}{R}$, The internal resistance R2 will be fixed at 10Ω. To calculate the current, Ohm's Law will be applied again the verify current of the overall circuit.

$$\mathrm{I} = \frac{Vin}{(R_1 + 10\Omega)}$$

The value of Vout will be sampling by ADC of the microcontroller, and the Vout obtained will divide by 10Ω.

5-3-3 Resistance Measurement

To calculate resistance, Wheatstone Bridge are used.

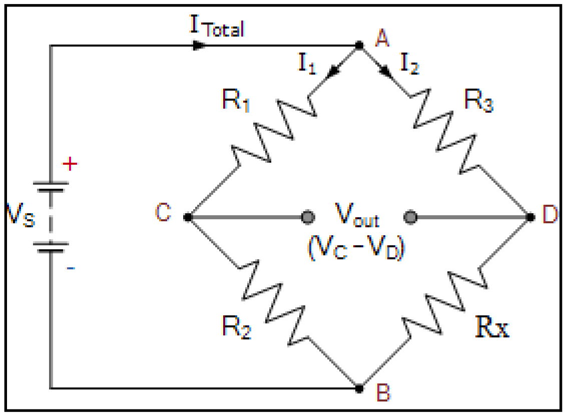

Figure 5-3-3-1: Wheatstone Bridge circuit

The figure above shows a Wheatstone Bridge and the formula below are used to find the voltage between node C and node D, V_{out}.

$$V_{out} = \left(\frac{R_2}{R_{1+R_2}} - \frac{R_x}{R_{x+R_3}}\right) V_s$$

To measure the unknown resistance Rx, the formula can be rearranged again to the equation below.

$$R_x = \frac{V_s . R_2 - V_{out}\,(R_1 + R_2)}{V_s . R_1 - V_{out}\,(R_1 + R_2)} \times R_3$$

Value of R_1, R_2 and R_3 will be predetermined. V_{out} can be obtained by finding the difference between the voltage of node C (V_c) and node D (V_d). V_s is fixed as 3.3V. The predetermined value for R_1 = 1kΩ, R_2= 750Ω and R_3 = 2kΩ. The resistance measurement range of this design is from 0Ω to 100kΩ.

5-3-4 Continuity

Continuity testing is a test to insure that connections are made correctly between two points. Continuity testing is done by testing the resistance between two points. If there is a low resistance such as less than a few Ωs that means the two points are connected electrically. The Wheatstone Bridge will be reused again for continuity measurement purpose. Measurement of continuity is show as table below.

Resistance	**Connection**
<3Ω	Two points are connected
>= 3Ω	Two points are not connected

Table 5-3-4-1: Continuity measurement

CHAPTER 6 SYSTEM INTEGRATION

6-1 Product Specification

Display	7.0 inch TFT Capacitive Touch LCD
Display Resolution	800 x 480
Oscilloscope Input Channel	1 Channel
Oscilloscope Sampling Resolution	256 (8-bit)
Oscilloscope Maximum Sampling speed	2MHz
Input Voltage Range	+40V to -40V
Oscilloscope Waveform Measurement	Vmin, Vmax, Vavg, Vpp, Vrms, Vmean, Frequency, Period, Duty Cycle, Rise Time, Fall Time, FFT
Oscilloscope System & Control	Time Base, Volt/Div, Trigger Position
Oscilloscope Trigger Mode	Auto, Normal, Single
Data Logger Mode	Yes
Storage Waveform	SD Card
Multimeter Sampling Resolution	4096(12-bit)
Multimeter Measurement	Voltage, Current, Resistance, Continuity

Table 6-1-1: Product Specification

CHAPTER 6 SYSTEM INTEGRATION

6-2 Hardware Block Diagram

With the hardware parts been discuss in previous chapter, all the hardware components need to be combine to become a fully functional handheld oscilloscope and multimeter. The figure below shows the block diagram of the hardware block diagram of all hardware connect to microcontroller.

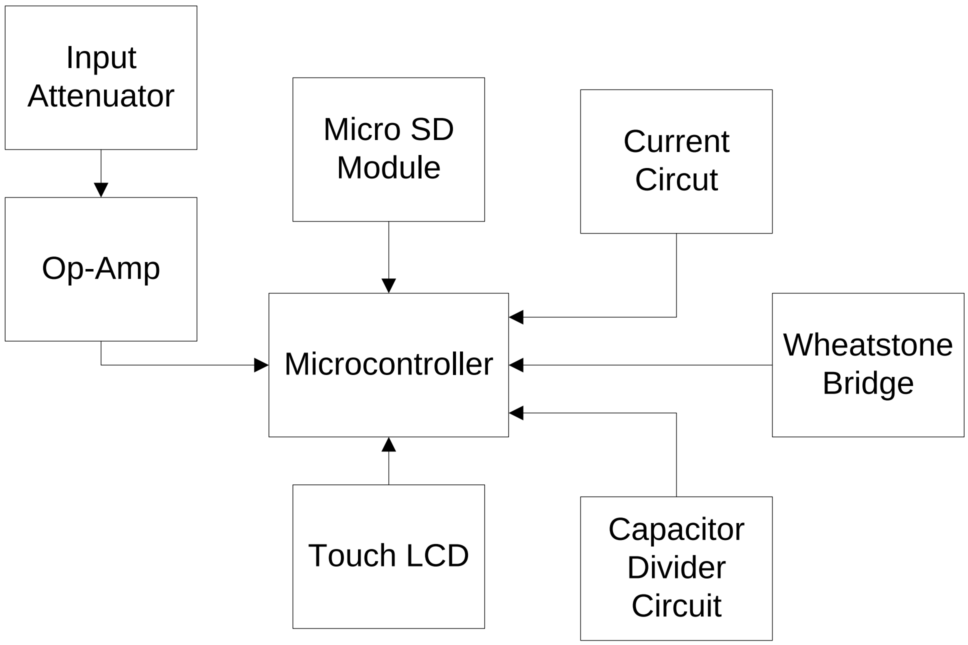

Figure 6-2-1: Hardware block Diagram

6-3 Firmware Flowchart

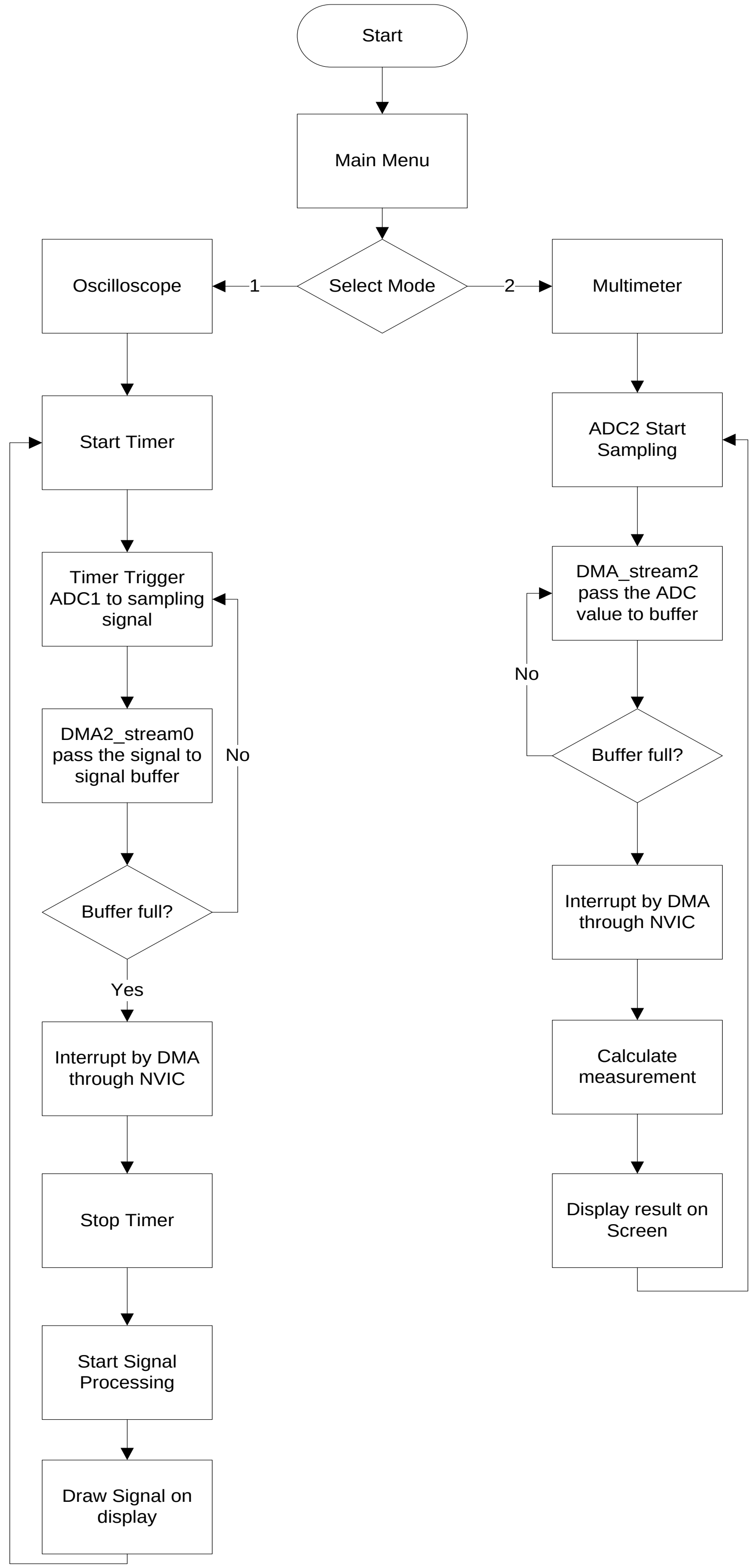

Figure 6-3-1: Firmware flowchart

6-4 Description of Firmware

Refer to the firmware flowchart in section 6-3, the timer is being stopped after DMA interrupt in oscilloscope part. This action is to prevent data hazard problem for the signal buffer in the oscilloscope. If the timer not being stopped, it still will continue trigger the ADC to sample and overwrite the current buffer. After the signal buffer finish being processing, then just start the timer again.

Furthermore, user's touch screen input is set to be the highest priority of all interrupt. The reason to do this is to provide a better user experience when using the product. No one else will like a laggy device. By implementing this, causes a tradeoff between user experience and data sampling features. This is because when touch screen interrupt, microcontroller will handling user's touch screen input and stop the sampling process due to limitation of single microcontroller.

6-5 User Interface

6-5-1 Main Menu

Main menu of this design is very simple, it is only a menu which allow user to choose between oscilloscope mode or multimeter mode. Figure below shows the main menu of this design.

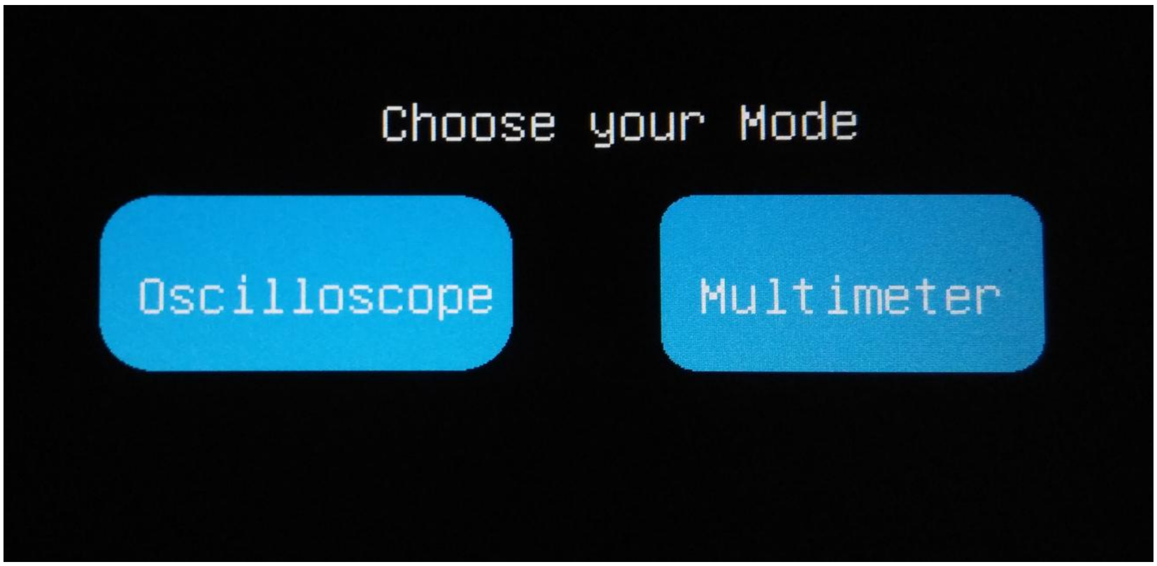

Figure 6-5-1-1: Main Menu of the product

6-5-2 User Interface of Oscilloscope

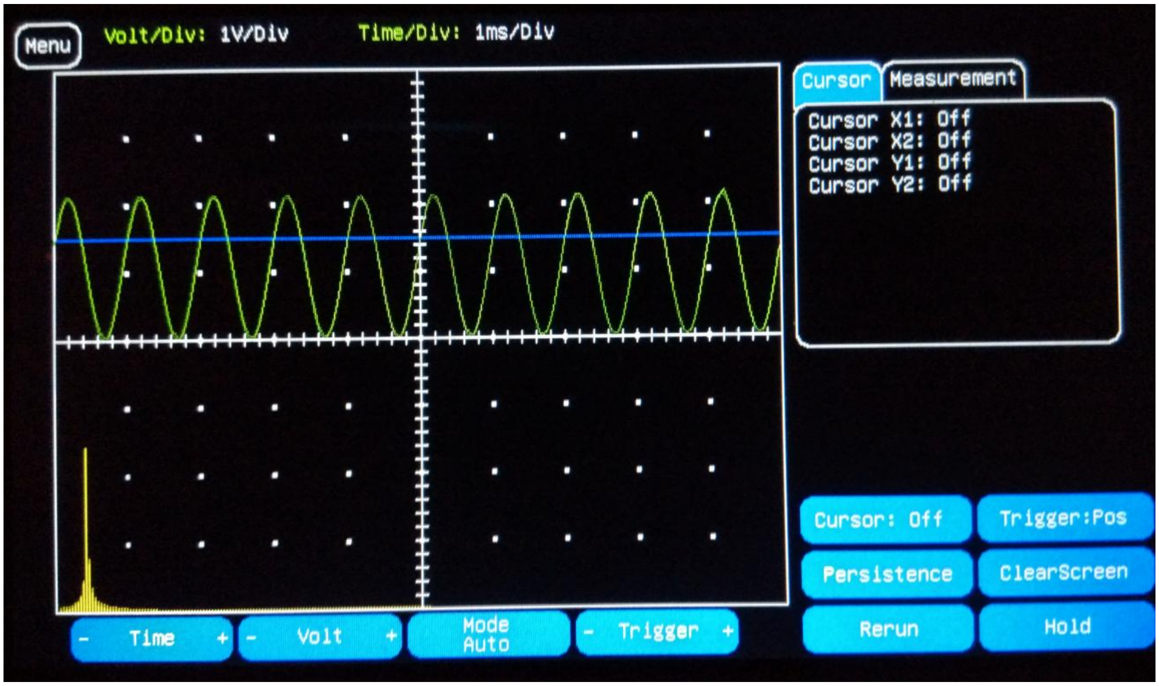

Figure 6-5-2-1: Overview user interface of oscilloscope

Top left of the user interface is the button return to main menu. Top part shows the current setting of volt/div and time/div. Green signal drawn in the signal box is the input signal. Blue line is the trigger line to trigger the signal. Yellow signal is the FFT result of the input signal also known as the frequency domain of the input signals. On the top right part is the tabs to show the cursor time and measurement result. User is able to switch between the two tabs.

For the button "Time", click on "+" to go for larger time/div and "-" for smaller time/div. Button "Volt" is to change the volt/div which this button also doing job of sending suitable digital signal to control the analog multiplexer. Button "Mode" is to switch between auto trigger mode, single trigger mode and normal trigger mode. Button "trigger" is to adjust the trigger line. Button "cursor" is to select which cursor to turn on. On the right side of "cursor" button is the button to change positive or negative trigger, the word "pos" means positive trigger and "neg" for negative trigger. Click on "persistence" button will turn on persistence mode and button "ClearScreen" will clear the signal that draw during persistence mode. Button "rerun" is used when user required rerunning sampling once during single trigger mode. Lastly, "Hold" button use to pause sampling signal while user requires to study the signal.

Figure below shows the measurement tab which displays all the measurement result of the signal.

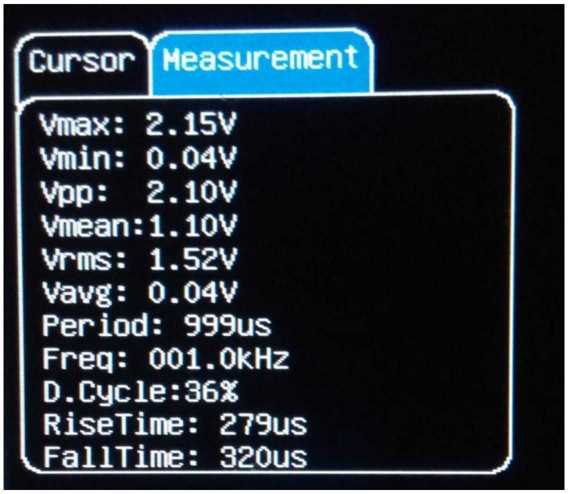

Figure 6-5-2-2: Measurement tab of Oscilloscope

6-5-3 User Interface of Multimeter

Design of user interface of multimeter is simple and clean which as shown as figure below.

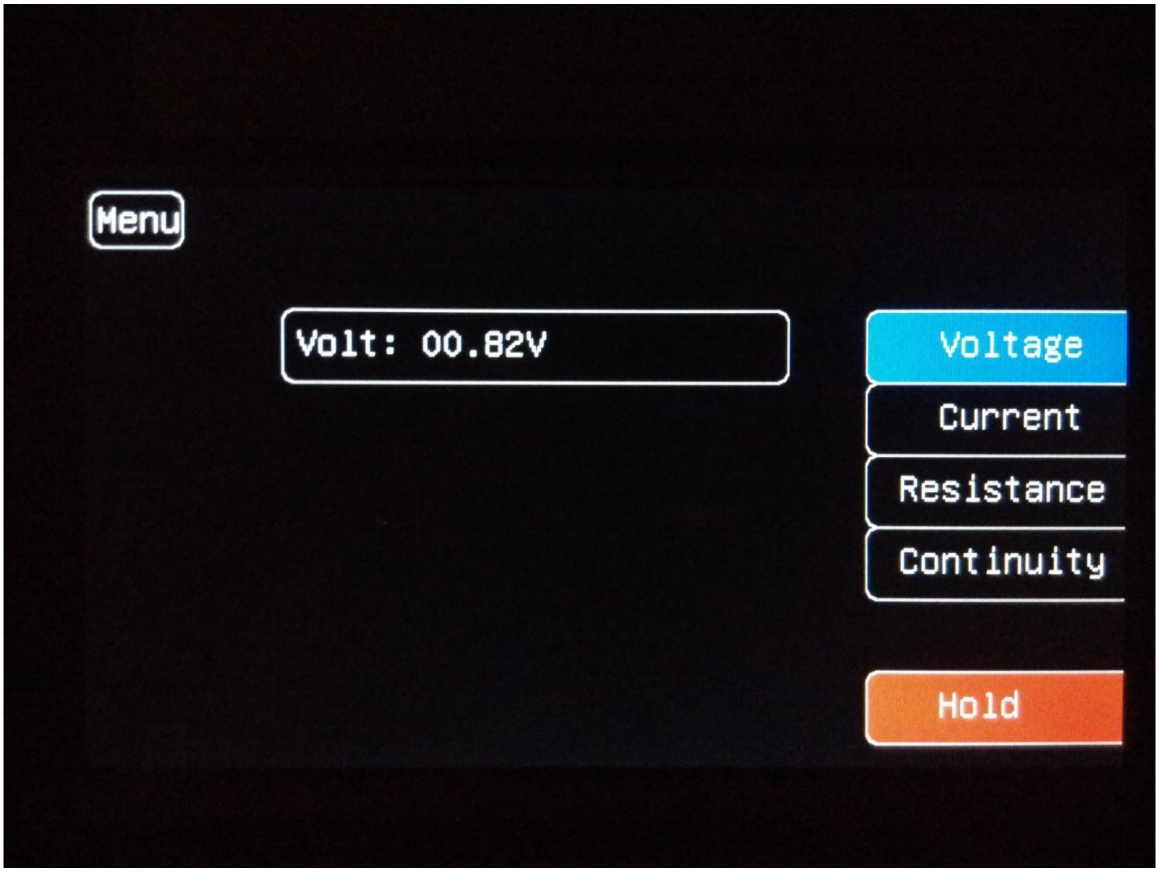

Figure 6-5-3-1: User Interface of Multimeter

Top left button is the button let user back to main menu. Middle part of the UI is the part shows measurement result and also right side is the selection of what value user wan to measure.

CHAPTER 7 TESTING AND RESULT DISCUSSION

7-1 Oscilloscope Testing

In order to test the accuracy of signal processing of the oscilloscope, a function generator is required to generate different amplitude and different frequency of AC voltage into ADC.

7-1-1 Testing Tools

7-1-1-1 GK101 function generator

GK101 is a direct digital synthesizer which used for creating arbitrary waveforms from a single, fixed-frequency reference clock. It is able to generate standard waveform such as sine, square, triangle, sawtooth rise, sawtooth fall, sinc, noise, exponential rise, exponential fall, and DC with frequency range between 1mHz to 10MHz.

Figure 7-1-1-1-1 GINGKO GK101

7-1-1-2 Multisim

Multisim is powerful circuit design software which allow user to build virtual circuit and do simulation based on the circuit. In the initial design of project, multisim is used to testing the circuit of input attenuator and level shifter before go to the stage of real implementation. By this simulation, it saves our time to testing on real components. Figure below shows one of the simulations done before build real circuit.

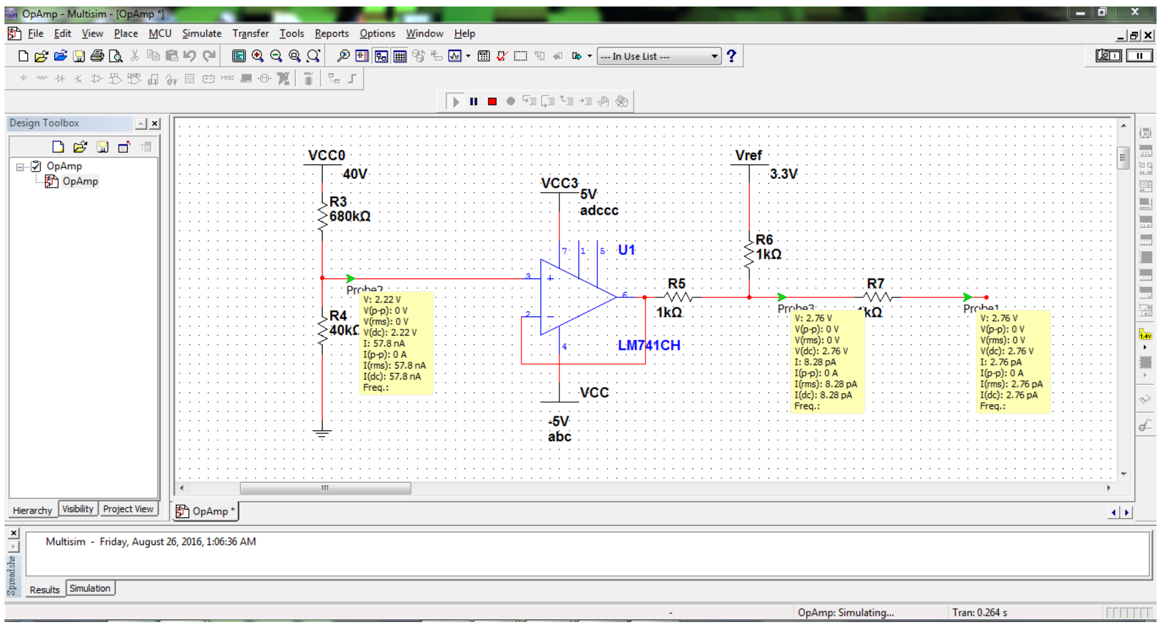

Figure 7-1-1-2-1: Simulation in Multisim

7-1-2 Oscilloscope Testing and Results

As mention earlier different time/div of the oscilloscope will change the sampling speed, hence each of time/div will have different sampling limit for signal frequency. We will test oscilloscope frequency range of each division by pumping different frequency input signal into oscilloscope. Constant input signal of sine wave and voltage range between 0 to 2V to test the frequency range of each division.

7-1-2-1 Sampling Frequency and Voltage Measurement of 5s/Div

Input Signal type: Sine wave

Input Vpp: 2V (0V to 2V)

Input Frequency	**Measurement Output**							
	Vmax	**Vmin**	**Vpp**	**Vmean**	**Vrms**	**Vavg**	**Period**	**Frequency**
0.5Hz	2.05V	0.03V	2.02V	1.05V	1.45V	0.04V	2s	0.5Hz
1Hz	2.07V	0.03V	2.03V	1.05V	1.46V	0.04V	1s	1Hz
2Hz	1.99V	0.03V	1.96V	1.05V	1.39V	0.04V	500ms	2Hz
3Hz	2.07V	0.03V	2.03V	1.05V	1.46V	0.04V	300ms	3Hz
4Hz	1.95V	0.02V	1.92V	1.05V	1.38V	0.04V	300ms	3Hz

Table 7-1-2-1-1: Measurement Testing for 5s/Div

Refer to the test result above, sampling frequency range for 5s/div is from 0.5Hz until 3Hz. Signals that from 4Hz onward required another time/div to display the correct result.

7-1-2-2 Sampling Frequency and Voltage Measurement of 2s/Div

Input	Measurement Output							
Frequency	**Vmax**	**Vmin**	**Vpp**	**Vmean**	**Vrms**	**Vavg**	**Period**	**Frequency**
1Hz	2.08V	0.03V	2.04V	1.05V	1.47V	0.04V	1s	1Hz
5Hz	2.00V	0.04V	1.95V	1.05V	1.41V	0.04V	199ms	5Hz
8Hz	2.09V	2.03V	2.05V	1.06V	1.48V	0.04V	120ms	8Hz
9Hz	2.08V	0.04V	2.04V	1.06V	1.47V	0.04V	120ms	8Hz
10Hz	2.08V	0.04V	2.03V	1.06V	1.42V	0.04V	80ms	12Hz

Table 7-1-2-2-1: Measurement output of 2s/Div

Hence, frequency range for 2s/Div is from 1Hz to 8Hz. Signal measurement result for more than 8Hz will be wrong for this division.

7-1-2-3 Sampling Frequency and Voltage Measurement of 1s/Div

Input	Measurement Output							
Frequency	**Vmax**	**Vmin**	**Vpp**	**Vmean**	**Vrms**	**Vavg**	**Period**	**Frequency**
1Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	1s	1Hz
2Hz	2.05V	0.07V	1.98V	1.06V	1.45V	0.04V	500ms	2Hz
5Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	199ms	5Hz
10Hz	2.05V	0.04V	2.01V	1.06V	1.35V	0.04V	99ms	10Hz
12Hz	2.09V	0.03V	2.05V	1.06V	1.47V	0.04V	80ms	12Hz
13Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	80ms	12Hz
14Hz	2.08V	0.04V	2.04V	1.16V	1.47V	0.04V	60ms	16Hz
15Hz	2.05V	0.07V	1.98V	1.06V	1.45V	0.04V	60ms	16Hz

Table 7-1-2-3-1: Measurement output of 1s/Div

Frequency range for 1s/div is from 1Hz to 12Hz where signal more than 12Hz will not accurate. Vpp is about ±0.05V difference with the input signal setting.

7-1-2-4 Sampling Frequency and Voltage Measurement of 500ms/Div

Input	**Measurement Output**							
Frequency	**Vmax**	**Vmin**	**Vpp**	**Vmean**	**Vrms**	**Vavg**	**Period**	**Frequency**
1Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	1s	1Hz
5Hz	2.08V	0.03V	2.05V	1.06V	1.47V	0.04V	199ms	5Hz
10Hz	2.05V	0.05V	2V	1.06V	1.45V	0.04V	99ms	10Hz
15Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	66ms	15Hz
20Hz	2.05V	0.08V	1.97V	1.06V	1.46V	0.04V	49ms	20Hz
25Hz	2.04V	0.03V	2.04V	1.06V	1.47V	0.04V	40ms	25Hz
26Hz	2.08V	0.03V	2.04V	1.06V	1.47V	0.04V	40ms	25Hz
30Hz	2.08V	0.03V	2.04V	1.06V	1.47V	0.04V	30ms	33Hz

Table 7-1-2-4-1: Measurement output of 500ms/Div

Frequency range for 1s/div is from 1Hz to 25Hz where signal more than 25Hz is not accurate.

7-1-2-5 Sampling Frequency and Voltage Measurement of 200ms/Div

Input	**Measurement Output**							
Frequency	**Vmax**	**Vmin**	**Vpp**	**Vmean**	**Vrms**	**Vavg**	**Period**	**Frequency**
1Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	1s	1Hz
5Hz	2.08V	0.03V	2.05V	1.06V	1.47V	0.04V	199ms	5Hz
10Hz	2.05V	0.05V	2V	1.06V	1.45V	0.04V	99ms	10Hz
15Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	66ms	15Hz
20Hz	2.05V	0.08V	1.97V	1.06V	1.46V	0.04V	49ms	20Hz
25Hz	2.04V	0.03V	2.04V	1.06V	1.47V	0.04V	40ms	25Hz
50Hz	2.03V	0.03V	2V	1.06V	1.45V	0.04V	20ms	50Hz

Table 7-1-2-5-1: Measurement output of 200ms/Div

Frequency range for 200ms/Div is from 1Hz to 50Hz for accurate results.

7-1-2-6 Sampling Frequency and Voltage Measurement of 100ms/Div

Input	Measurement Output							
Frequency	Vmax	Vmin	Vpp	Vmean	Vrms	Vavg	Period	Frequency
2Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	500ms	2Hz
5Hz	2.08V	0.03V	2.05V	1.06V	1.47V	0.04V	199ms	5Hz
10Hz	2.05V	0.05V	2V	1.06V	1.45V	0.04V	99ms	10Hz
15Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	66ms	15Hz
20Hz	2.05V	0.08V	1.97V	1.06V	1.46V	0.04V	49ms	20Hz
25Hz	2.04V	0.03V	2.04V	1.06V	1.47V	0.04V	40ms	25Hz
50Hz	2.03V	0.03V	2V	1.06V	1.45V	0.04V	20ms	50Hz
100Hz	2.02V	0.04V	1.97V	1.06V	1.43V	0.04V	10ms	99Hz
125Hz	2.09V	0.02V	2.07V	1.06V	1.48V	0.04V	8ms	124Hz
130Hz	2.09V	0.02V	2.07V	1.06V	1.48V	0.04V	6ms	166Hz

Table 7-1-2-6-1: Measurement output of 100ms/Div

Frequency range for 100ms/Div is from 2Hz to 125Hz for accurate results.

7-1-2-7 Sampling Frequency and Voltage Measurement of 50ms/Div

Input	Measurement Output							
Frequency	Vmax	Vmin	Vpp	Vmean	Vrms	Vavg	Period	Frequency
4Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	250ms	4Hz
15Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	66ms	15Hz
20Hz	2.05V	0.08V	1.97V	1.06V	1.46V	0.04V	49ms	20Hz
25Hz	2.04V	0.03V	2.04V	1.06V	1.47V	0.04V	40ms	25Hz
50Hz	2.03V	0.03V	2V	1.06V	1.45V	0.04V	20ms	50Hz
100Hz	2.02V	0.04V	1.97V	1.06V	1.43V	0.04V	10ms	99Hz
125Hz	2.09V	0.02V	2.07V	1.06V	1.48V	0.04V	8ms	124Hz
150Hz	2.09V	0.02V	2.07V	1.06V	1.48V	0.04V	7ms	142Hz
200Hz	2.04V	0.07V	1.97V	1.06V	1.44V	0.04V	5ms	199Hz
250Hz	2.09V	0.15V	1.94V	1.06V	1.48V	0.04V	4ms	249Hz

Table 7-1-2-7-1: Measurement output of 50ms/Div

Frequency range for 50ms/Div is from 4Hz to 250Hz for accurate results.

7-1-2-8 Sampling Frequency and Voltage Measurement of 20ms/Div

Input	Measurement Output							
Frequency	Vmax	Vmin	Vpp	Vmean	Vrms	Vavg	Period	Frequency
6Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	166ms	6Hz
10Hz	2.08V	0.03V	2.04V	1.06V	1.47V	0.04V	99ms	10Hz
25Hz	2.04V	0.03V	2.04V	1.06V	1.47V	0.04V	40ms	25Hz
50Hz	2.03V	0.03V	2V	1.06V	1.45V	0.04V	20ms	50Hz
100Hz	2.02V	0.04V	1.97V	1.06V	1.43V	0.04V	10ms	100Hz
150Hz	2.09V	0.02V	2.07V	1.06V	1.48V	0.04V	6ms	147Hz
200Hz	2.04V	0.07V	1.97V	1.06V	1.44V	0.04V	4ms	208Hz
250Hz	2.10V	0.02V	2.08V	1.06V	1.48V	0.04V	3ms	250Hz
280Hz	2.10V	0.02V	2.08V	1.06V	1.48V	0.04V	3ms	277Hz
300Hz	2.10V	0.01V	2.09V	1.06V	1.48V	0.04V	3ms	312Hz

Table 7-1-2-8-1: Measurement output of 20ms/Div

7-1-2-9 Sampling Frequency and Voltage Measurement of 10ms/Div

Input	Measurement Output							
Frequency	Vmax	Vmin	Vpp	Vmean	Vrms	Vavg	Period	Frequency
20Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	49ms	20Hz
50Hz	2.09V	0.04V	2.04V	1.06V	1.48V	0.04V	20ms	50Hz
100Hz	2.09V	0.04V	2.04V	1.06V	1.48V	0.04V	10ms	100Hz
150Hz	2.10V	0.03V	2.07V	1.06V	1.48V	0.04V	6ms	151Hz
200Hz	2.10V	0.03V	2.07V	1.06V	1.48V	0.04V	5ms	200Hz
250Hz	2.10V	0.02V	2.08V	1.07V	1.48V	0.04V	3ms	250Hz
300Hz	2.10V	0.02V	2.08V	1.06V	1.48V	0.04V	3ms	294Hz
400Hz	2.11V	0.02V	2.09V	1.06V	1.49V	0.04V	2ms	416Hz
500Hz	2.07V	0.05V	2.01V	1.06V	1.46V	0.04V	1ms	500Hz
600Hz	2.12V	0.02V	2.10V	1.07V	1.50V	0.04V	1ms	625Hz
700Hz	2.12V	0.02V	2.10V	1.07V	1.50V	0.04V	1ms	714Hz

Table 7-1-2-9-1: Measurement output of 10ms/Div

7-1-2-10 Sampling Frequency and Voltage Measurement of 5ms/Div

Input	**Measurement Output**							
Frequency	**Vmax**	**Vmin**	**Vpp**	**Vmean**	**Vrms**	**Vavg**	**Period**	**Frequency**
40Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	24ms	40Hz
100Hz	2.10V	0.07V	2.03V	1.08V	1.48V	0.04V	10ms	100Hz
200Hz	2.11V	0.05V	2.05V	1.08V	1.49V	0.04V	5ms	200Hz
300Hz	2.10V	0.03V	2.07V	1.07V	1.48V	0.04V	3ms	303Hz
400Hz	2.10V	0.02V	2.08V	1.07V	1.48V	0.04V	2ms	400Hz
500Hz	2.10V	0.03V	2.07V	1.07V	1.48V	0.04V	1ms	500Hz
600Hz	2.11V	0.02V	2.09V	1.07V	1.49V	0.04V	1ms	588Hz
700Hz	2.11V	0.02V	2.09V	1.07V	1.49V	0.04V	1ms	714Hz
800Hz	2.12V	0.04V	2.08V	1.08V	1.50V	0.04V	1ms	833Hz
900Hz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	1ms	892Hz
1kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	999us	1.0kHz
1.1kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	900us	1.1kHz
1.2kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	800us	1.2kHz
1.3kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	759us	1.3kHz
1.4kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	680us	1.4kHz
1.5kHz	2.12V	0.01V	2.11V	1.08V	1.50V	0.04V	640us	1.5kHz
2kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	499us	2.0kHz

Table 7-1-2-10-1: Measurement output of 5ms/Div

7-1-2-11 Sampling Frequency and Voltage Measurement of 2ms/Div

Input	Measurement Output							
Frequency	**Vmax**	**Vmin**	**Vpp**	**Vmean**	**Vrms**	**Vavg**	**Period**	**Frequency**
100Hz	2.08V	0.04V	2.03V	1.06V	1.47V	0.04V	10ms	100Hz
500Hz	2.11V	0.02V	2.09V	1.07V	1.49V	0.04V	1ms	500Hz
1kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	999us	1kHz
1.5kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	640us	1.5kHz
2.5kHz	2.10V	0.03V	2.07V	1.08V	1.48V	0.04V	400us	2.5kHz
3kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	320us	3.1kHz
3.5kHz	2.12V	0.01V	2.11V	1.08V	1.50V	0.04V	279us	3.5kHz
4kHz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	240us	4.1kHz
5kHz	1.95V	0.02V	1.92V	1.08V	1.38V	0.04V	200us	5kHz
6kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	160us	6.2kHz

Table 7-1-2-11-1: Measurement output of 2ms/Div

7-1-2-12 Sampling Frequency and Voltage Measurement of 1ms/Div

Input	Measurement Output							
Frequency	**Vmax**	**Vmin**	**Vpp**	**Vmean**	**Vrms**	**Vavg**	**Period**	**Frequency**
200Hz	2.11V	0.02V	2.09V	1.07V	1.49V	0.04V	5ms	200Hz
1kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	999us	1kHz
2kHz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	499us	2kHz
3kHz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	340us	2.9kHz
4kHz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	240us	4.1kHz
5kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	200us	5kHz
6kHz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	160us	6.2kHz
7kHz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	139us	7.1kHz
8kHz	2.11V	0.02V	2.09V	1.07V	1.49V	0.04V	120us	8.3kHz
9kHz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	100us	10kHz

Table 7-1-2-12-1: Measurement output of 1ms/Div

7-1-2-13 Sampling Frequency and Voltage Measurement of 500us/Div

Input Frequency	Measurement Output							
	Vmax	**Vmin**	**Vpp**	**Vmean**	**Vrms**	**Vavg**	**Period**	**Frequency**
1kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	999us	1kHz
5kHz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	200us	5kHz
10kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	100us	10kHz
15kHz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	69us	14.2kHz
16kHz	2.15V	0.03V	2.11V	1.08V	1.52V	0.04V	60us	16.6kHz
17kHz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	60us	16.6kHz

Table 7-1-2-13-1: Measurement output of 500us/Div

7-1-2-14 Sampling Frequency and Voltage Measurement of 250us/Div

Input Frequency	Measurement Output							
	Vmax	**Vmin**	**Vpp**	**Vmean**	**Vrms**	**Vavg**	**Period**	**Frequency**
1kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	999us	1kHz
5kHz	2.14V	0.02V	2.11V	1.07V	1.51V	0.04V	200us	5kHz
10kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	100us	10kHz
15kHz	2.14V	0.02V	2.11V	1.07V	1.51V	0.04V	65us	15.3kHz
20kHz	2.10V	0.05V	2.04V	1.08V	1.48V	0.04V	50us	20kHz
25kHz	2.14V	0.03V	2.10V	1.08V	1.51V	0.04V	40us	25kHz
30kHz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	30us	33.3kHz
50kHz	2.10V	0.07V	2.03V	1.08V	1.48V	0.04V	20us	50kHz

Table 7-1-2-14-1: Measurement output of 250us/Div

7-1-2-15 Sampling Frequency and Voltage Measurement of 100us/Div

Input	**Measurement Output**							
Frequency	**Vmax**	**Vmin**	**Vpp**	**Vmean**	**Vrms**	**Vavg**	**Period**	**Frequency**
5kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	200us	5kHz
12.5kHz	2.14V	0.03V	2.10V	1.08V	1.51V	0.04V	80us	12.5kHz
25kHz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	40us	25kHz
37.5kHz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	26us	38.4kHz
50kHz	2.12V	0.02V	2.09V	10.8V	1.50V	0.04V	20us	50kHz
62.5kHz	2.15V	0.03V	2.11V	1.09V	1.52V	0.04V	16us	62.5kHz
75kHz	2.14V	0.03V	2.10V	1.08V	1.51V	0.04V	12us	83.3kHz

Table 7-1-2-15-1: Measurement output of 100us/Div

7-1-2-16 Sampling Frequency and Voltage Measurement of 50us/Div

Input	**Measurement Output**							
Frequency	**Vmax**	**Vmin**	**Vpp**	**Vmean**	**Vrms**	**Vavg**	**Period**	**Frequency**
10kHz	2.15V	0.05V	2.09V	1.10V	1.52V	0.04V	100us	10kHz
20kHz	2.15V	0.03V	2.11V	1.09V	1.52V	0.04V	50us	20kHz
30kHz	2.15V	0.03V	2.11V	1.09V	1.52V	0.04V	33us	30.3kHz
40kHz	2.16V	0.04V	2.11V	1.10V	1.53V	0.04V	25us	40kHz
50kHz	2.17V	0.04V	2.12V	1.11V	1.53V	0.04V	20us	50kHz
60kHz	2.15V	0.03V	2.11V	1.09V	1.52V	0.04V	17us	58.8kHz
70kHz	2.14V	0.02V	2.11V	1.08V	1.51V	0.04V	14us	71.4kHz
80kHz	2.11V	0.01V	2.10V	1.06V	1.49V	0.04V	12us	83.3kHz
90kHz	2.12V	0.01V	2.11V	1.07V	1.50V	0.04V	11us	90.9kHz
100kHz	2.11V	0.01V	2.10V	1.06V	1.49V	0.04V	10us	100kHz
110kHz	2.11V	0.01V	2.10V	1.06V	1.49V	0.04V	9us	111.1kHz
120kHz	2.12V	0.01V	2.11V	1.07V	1.50V	0.04V	8us	125kHz

Table 7-1-2-16-1: Measurement output of 50us/Div

7-1-2-17 Sampling Frequency and Voltage Measurement of 50us/Div

Input Frequency	Measurement Output							
	Vmax	**Vmin**	**Vpp**	**Vmean**	**Vrms**	**Vavg**	**Period**	**Frequency**
10kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	100us	10kHz
50kHz	2.12V	0.01V	2.11V	1.07V	1.50V	0.04V	20us	50kHz
100kHz	2.12V	0.01V	2.11V	1.07V	1.50V	0.04V	10us	100kHz
150kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	6us	153.8kHz
200kHz	2.10V	0.03V	2.07V	1.07V	1.48V	0.04V	5us	200kHz
250kHz	2.11V	0.02V	2.09V	1.07V	1.49V	0.04V	4us	250kHz
300kHz	2.12V	0.02V	2.10V	1.08V	1.50V	0.04V	3us	285.7kHz
350kHz	2.10V	0.03V	2.07V	1.07V	1.48V	0.04V	3us	333.3kHz
400kHz	2.03V	0.02V	2.01V	1.07V	1.43V	0.04V	2us	400kHz
500kHz	2.10V	0.03V	2.07V	1.07V	1.48V	0.04V	2us	500kHz

Table 7-1-2-17-1: Measurement output of 25us/Div

7-1-2-18 Duty Cycle Test

Square wave will be used as the input signal and different value of duty cycle will be used in order to test the accuracy of the duty cycle measurement. Table below shows the result of the duty cycle test.

Input Signal Duty Cycle	Oscilloscope Duty Cycle Result
10%	10%
15%	16%
20%	20%
31%	30%
36%	36%
50%	50%
75%	76%
80%	80%
90%	90%
98%	98%
99%	98%

Table 7-1-2-18: Duty Cycle Test

7-2 Multimeter Testing and Result

7-2-1 Voltmeter Test

To testing the function of voltmeter, a DC voltage generator is used to act as input voltage. Table below shows the input voltage and the result of testing.

Input Voltage	**Voltmeter Result**
0V	0.00V
0.1V	0.11V
0.5V	0.56V
1V	1.13V
1.3V	1.29V
1.7V	1.72V
2.5V	2.44V
3V	3.03V

Table 7-2-1-1: Result of Voltmeter Testing

7-2-2 Ammeter Test

To verify the result of current measured in this design, we compare the result on our design with the DC power supply used where the DC power supply has its build in ammeter. The result of ammeter in this design is same as the result shown on the DC power supply.

7-2-3 Resistance Test

Different resistors are used to test the resistance calculation for this design. Table below shows the result of resistance in this design.

Resistor Used	**Resistance Result**
10Ohm	12Ohm
100Ohm	96Ohm
1000Ohm	979Ohm
10kOhm	7.3kOhm

Table 7-2-3-1: Result of Resistance Test

Result above shows that the predefined resistor of Wheatstone bridge circuit designed in this project will get more accurate result when measure resistor from 10Ohm to 1kOhm.

7-2-4 Continuity Test

Continuity test is a simple test where we plug a wire on the Wheatstone bridge, if there is a wire without short circuit then display will show a connected result. When unplug the wire, display will show no continuity.

CHAPTER 8 CONCLUSION

8-1 Summary

The main objective of this project has been achieved where the hardware and firmware of oscilloscope and multimeter are built and developed. All the components are also integrated into a fully functional system.

In this project, a handheld oscilloscope with multimeter prototype is designed and built. The sampling speed of oscilloscope is up to 2MHz. The oscilloscope sampling resolution is 8-bit to ensure fast sampling and input signal ranges of the oscilloscope is +40V to -40V. The oscilloscope consists of FFT feature and 11 signal measurements. Besides that, 17 time/div are developed in the oscilloscope.

On the other hand, a multimeter is developed with features of voltmeter, ammeter, resistance measurement and continuity.

8-2 Encountered Problem

The first problem encountered is that lack of memory in the microcontroller which limits the firmware to add more features such as waveform storage and dual channel oscilloscope. The microcontroller doesn't have a huge memory allow the design to store more signal buffer.

On the other hand, the display used in the design is only a display controller which limits the design cannot implement print screen feature. Furthermore, refresh rate of the display is also slow which causes data loss when drawing of fast signal.

8-3 Future Implementation

There are still huge room of improvement for this design such as change the ADC sampling mode to interleaved mode which may improve 3 times speed of oscilloscope sampling.

Furthermore, build an external trigger hardware circuit instead of using software trigger which can improve the quality of signal sampling. Software edge trigger sometimes may fail when the signals contain a lot of noise, where these noises will trigger the software trigger function.

Reference

Daycounter , 2016. *Wheatstone bridge calculator*. Available from: <http://www.daycounter.com/Calculators/WheatStone-Bridge/WheatStone-Bridge-Calculator.phtml>. [3 August 2016]

OWON HDS-N series digital oscilloscope, 2012. Available from: <http://www.owon.com.hk/products_info.asp?ProID=170#sthash.MoueAcxs.dpbs>. [27 March 2016]

Texas Instruments, 2016. *TLVx314.* Available from: <http://www.ti.com/lit/ds/symlink/tlv314.pdf>. [17 August 2016]

Toolboom, 2012, *Review of Hantek DSO1060 Handheld Oscilloscope.* Available from : <http://toolboom.com/en/Articles-and-Video/Review-of-Hantek- DSO1060-Handheld-Oscilloscope-Oscilloscope-plus-Multimeter.html> [27 March 2015]

STMicroeletronic, n.d. *STM32F407xx Datasheet*. Available from: <http://udel.edu/~furkan/cpeg423/13summer/stm32f4/DM00037051.pdf>. [26 July 2016]

Vellaman Inc, n.d, *Handheld Pocket Scope 40MS/s.* Available from: <http://www.vellemanusa.com/products/view/?country=us&lang=enu&id=525377> [27 March 2016]

Waveshare, n.d, *STM32 Cortex-M4 F4Development Board.* Available from: < http://www.waveshare.com/open407v-d-standard.htm> [28 March 2016]