ECEN 5803

Mastering Embedded Systems Architecture

Written Document: Technical Project Report

Date	10/28/2021
Supervisor	Timothy Scherr
Prepared by	Taher Ujjainwala Steve Christensen

Index

1) Executive Summary

According to RFS issued by Keithley Corporation, we performed Suitability analysis of STM32F401RET6 for Signal Analyzer application. The Nucleo-F401RE development board was used to evaluate STM32F401RET6 controller. STM32F401RET6 is interfaced with digital interfaces, analog sensors and user interface devices to test its hardware capabilities. Audio waves in the range of 20Hz – 20 KHz were generated using MCU whose volume and pitch could be adjusted using two potentiometer. Additionally, four push buttons are interfaced in digital mode and interrupt driven mode to change the state of internal and external LED. The software performance was tested on the basis of responsiveness, ability to execute multi-threading, DSP based mathematical operations, and accuracy. This performance is also compared with simulation tool MATLAB.

1) The STM32F401RET6 uses ST-LINK/V2.1 debugger to program the main target MCU. It is designed in such a way to make the debugging easy. This reduced the overall complexity of using JTAG to program the microcontroller
2) When a Dhrystone benchmark was run on the target MCU, it runs on 129.9 DMIP which is nearly 30% greater than the required processor performance that was mentioned by Keithley as 100DMIPS
3) Clock frequency of the MCU is 72MHz which makes it suitable to frequency driven applications like function generator and signal synthesizer
4) Having high clock frequency gives us better reliability and performance for digital signal processing applications
5) This processor supports floating point calculations that makes is advantageous for DSP application
6) Audio frequency was generated using STM32F401RET6 with the help of potentiometer. Known frequency was used to get the accuracy on the software end by generating equivalent proportional change in led brightness
7) Real time performance was tested to check the ability of system to work in interrupt driven, multi-threading environment. Shared resources and concurrency was tested using threads and mutex. Implementation was done in efficient manner with no human visible latency
8) This MCU have about 50 GPIO with external interrupt capability, 7 timer, 2 watchdog, 3 SPI and I2C bus interfaces, supports OTG features, and provides inbuilt RTC, 4 UARTS, and 12-bit ADC with total of 16 channels
9) The estimated cost for the system would be approximately $15-$17 if some of the components like shift register are removed and bare chip is used which is under the requirement given by Keithley

Therefore, based on the above results, performance and reliability the target MCU STM32F401RET6 satisfies all the functionalities and requirements as listed in RFS. Hence, we will GO for this design

2) Problem Statement and Objectives

To evaluate the hardware and software capabilities of STM32F401RET6 micro controller based on the requirement specification listed by Keithley. This MCU is based on ARM Cortex M4 architecture. The objective is to test DSP capabilities to generate particular frequency signal as well as interface various peripherals like push buttons, LED's, potentiometer, external I2C based temperature sensor - DS1631 and SPI based 16X2 LCD display. We also evaluate the MCU performance in interrupt, non-interrupt, and multiple threads RTOS mode.

Objectives:

Module 1: Getting started with Nucleo board using Mbed online compiler and Keil uVision IDE. Blinking an LED and displaying the current time.

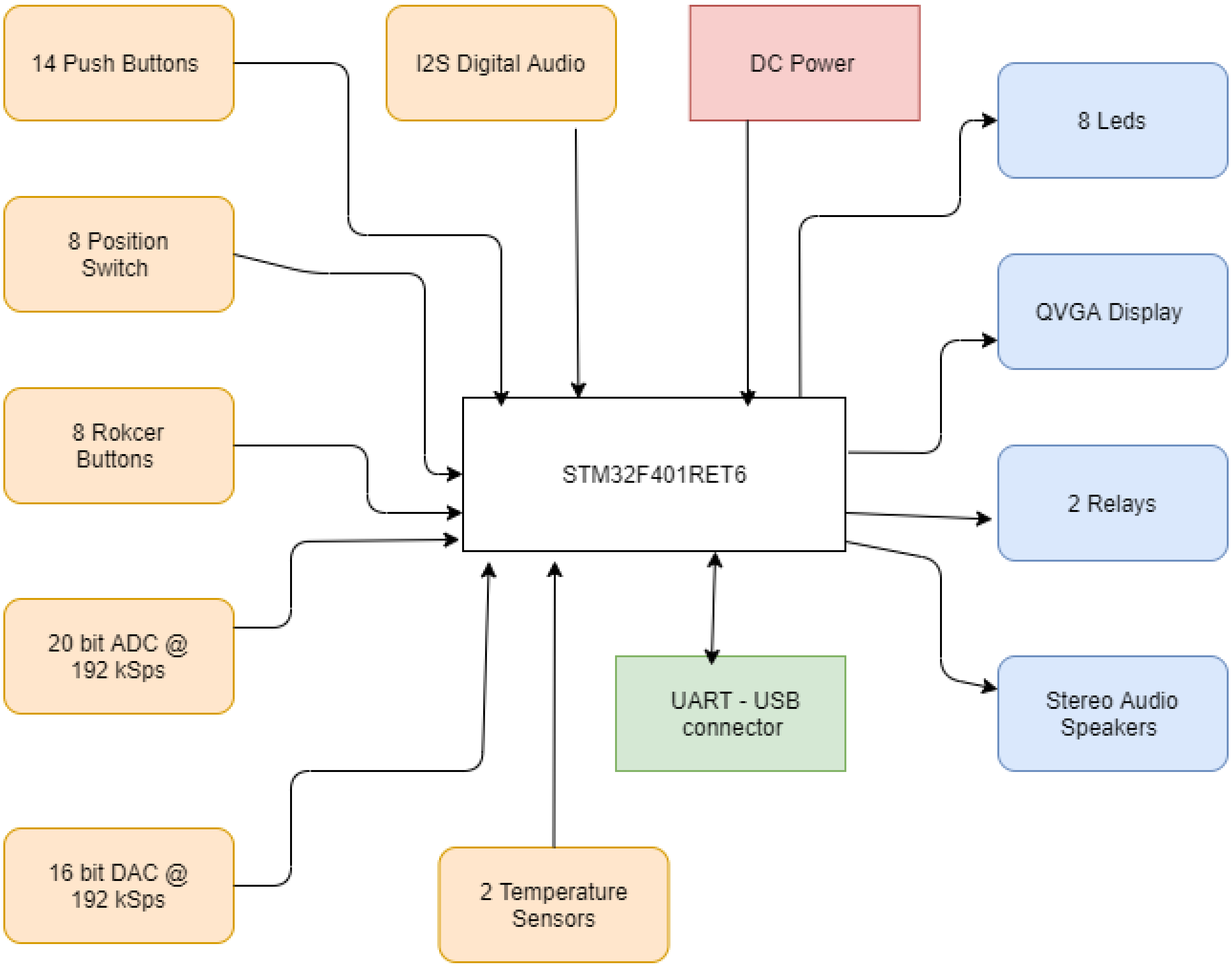
Module 2: a) Interfacing leds and push buttons on Nucleo Development board. Making using of MBED API's to read digital input from push button and writing it to led. Button were interfaces in normal polling mode as well as interrupt mode
b) Interfacing potentiometers and speaker using ADC and PWM. Values of potentiometer were used to control the pitch and volume of speaker
c) Interfacing DS1631 temperature sensor and 16X2 LCD using I2C and SPI. Printing the debugging results on UART. DS1631 temperature sensor used I2C protocol and the results were displayed on LCD. LCD was interfaced using SPI with the help of shift register 74HC595N

Module 3: RTOS capabilities were tested and evaluated for STM32F401RET6 MCU. Concepts of multi-threading, mutex, semaphores are used to interface multiple hardware components like displaying temperature and counter on LCD, changing the brightness of led based on potentiometer value and blinking the led.

Module 4: Implementing the use of DSP libraries to evaluate the FFT of an given waveform. STM32F401RET6 was used to generate waveform and drive the led using ADC and PWM.

3) Approach and methodology for evaluation

3.1 Block diagram



Block diagram of the NUCLEO-F401RE evaluation board

3.2 Methodology

Module 1: Getting Started, Hello World, What is the current time?

In this module, current time was displayed using Mbed API on the UART. Nucleo-64 board have inbuilt RTC module. MBED online compiler was used to set the current time in seconds since 1st Jan, 1970. Current time stamp in seconds was used to set time and print time and date on tera term.

Module 2: Reading button, driving leds, reading potentiometer, driving speaker, reading temperature sensor and displaying on LCD

2.8: Programming using Mbed API

In this section, four SPDT buttons were interfaced in pull down mode to read on MCU GPIO inputs. Based on the button press, internal and external led were controlled. Same setup was implemented using Interrupts. Mbed API like DigitalIn, DigitalOut, BusIn, BusOut and InterruptIn were used to implement the tasks.

2.9: Analog Input and Audio output

In this section, two potentiometers were interfaced using analog input pins of MCU. Based on the potentiometer value, pitch and volume were controlled for speaker using PWM. Mbed API like DigitalIn, AnalogIn, PwmOUT were used to implement the tasks.

2.11 Serial Communication – UART, I2C and SPI

In this section, temperature sensor DS1631 and NHD-0216HZ LCD 16X2 were interfaced. UART was used to print the debugging messages over serial terminal (Tera term). DS1631 was interfaced using I2C protocol to read the current temperature and display it on LCD. NHD-0216HZ LCD was interfaced using shift register 74HC595N via SPI. Temperature result was displayed on LCD and printed on serial terminal.

Module 3: RTOS Implementation

In this module, RTOS capabilities were evaluated for STM32F401RET6 by running multi-threading process. Display the temperature value on LCD, display the counter value on LCD, control the brightness of led based on potentiometer value and blink the on board led. Such thread were implemented using Osthreads, Mutex, semaphores.

Module 4: Signal generator and Analyzer

In this module, DSP libraries and vendor libraries were used to evaluate the DSP capabilities. Here target MCU was used to generate a sinusoidal waveform at fixed frequency as well as detect sinusoidal waveform frequency by performing Fast Fourier transform on the input signal.

3.3 Hardware evaluation

Hardware evaluation is done based on whether the necessary peripherals can be implemented on STM32F401RET6 for Keithley's Signal Analyzer product

Input Requirements:

1. **Temperature Sensor:** Since the MCU doesn't have any on board temperature sensor, we use an external temperature sensor which works on I2S protocol. DS1631 is a high precision temperature sensor, providing 9 to 12-bit temperature values. Ds1631 is capable to provide temperature in the range of -55 degree Celsius to 125 degree Celsius

2. **Potentiometer:** Two P103 high precision potentiometer are used as an input to MCU. The value of potentiometer are 10KOhms and have tolerance of ±10%. This potentiometer provides analog value which is used to adjust the pitch and volume of the audio signal.

3. **ADC:** There are 16 channels ADC on board which have 12 bit resolution. The output of signal generator is given to ADC to calculate the frequency using FFT.

4. **DAC:** There are is DAC capability provided on board. However, MCU have enough analog pins which can be used to read the data from external DAC module. Microchip MCP4921 is used in the system, which is a 12-bit low cost module. DAC is used to create a 1004 Hz tone which is used to perform further analysis by the software.

5. **Switches:** 4 SPDT type buttons are used to control the led state. This switches are push to connect type switches

Output Requirements:

1. **LCD:** STM32F401RET6 has no on-board LCD. Here, we make use of Newhaven NHD-0216HZ 16X2 LCD module. This module works on power supply 3.3V as well as 5V. We use shift register 74HC595N which connects all the control and data pins of LCD and communicates via SPI with the MCU

2. **LED:** The Nucleo development kit have single GREEN LED. We also use external RED 5mm pitch LED to control the brightness based on the potentiometer value.

3. **PWM:** This is used to generate the electrical signals. PWM drives the speaker module by adjusting the period that is duty cycle and also drives the speaker volume and pitch. To control the LED brightness using potentiometer PWM is used.

4. **Speaker:** An external speaker is driver based on the frequency generated

Result: The STM32F401RE MCU can be used to implement all the input and output subsystems given in the requirements in the RFS. The features required are Digital I/O, Analog I/O, ADC, DAC, SPI, UART and PWM output.

4) Module test results

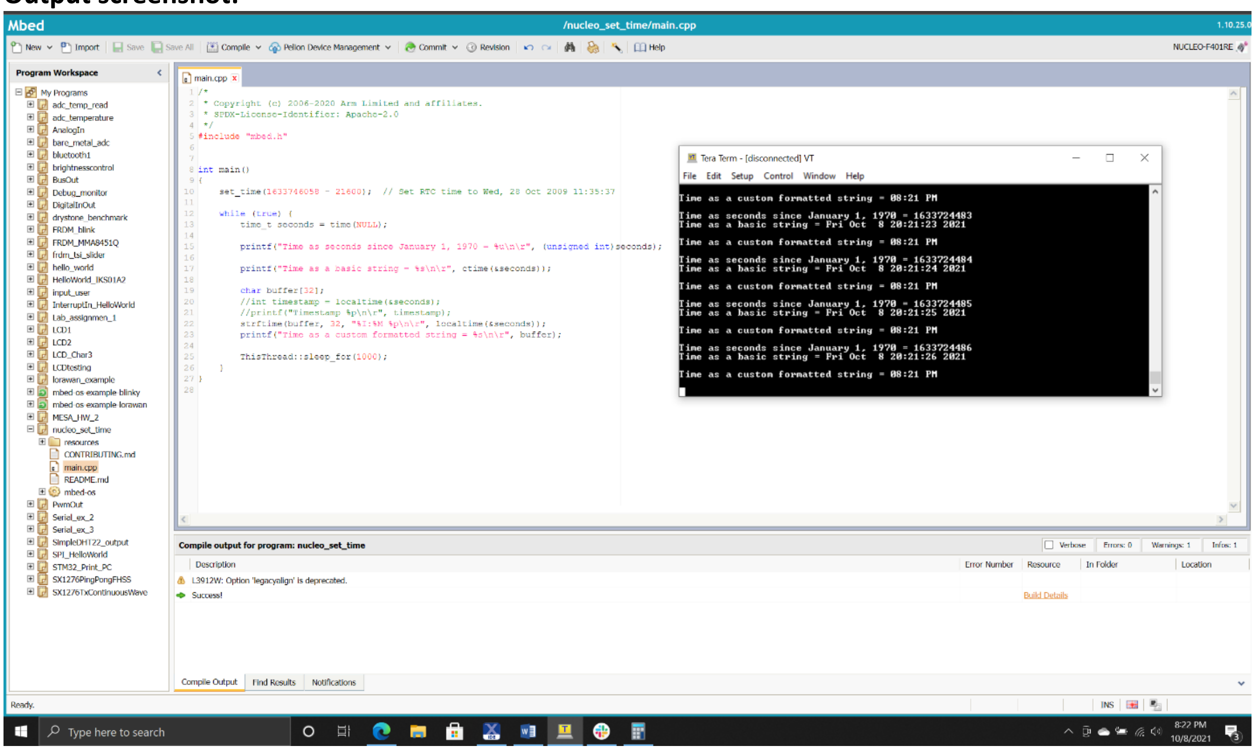
4.1 Module 1:

2. Where (at what address) does the Reset handler begin in the memory map?
Answer: **Reset handler is at 0x8000458 as shown by the disassembly window. Screenshot is attached in Appendix 1**

3. How much memory is used by the code (Led blinking code for the homework)?
Answer: **Memory used by the code is 36.340 KB as shown by the compiler after compiling the OS blinky code for Nucleo-F401RE. Screenshot is attached in Appendix 2**

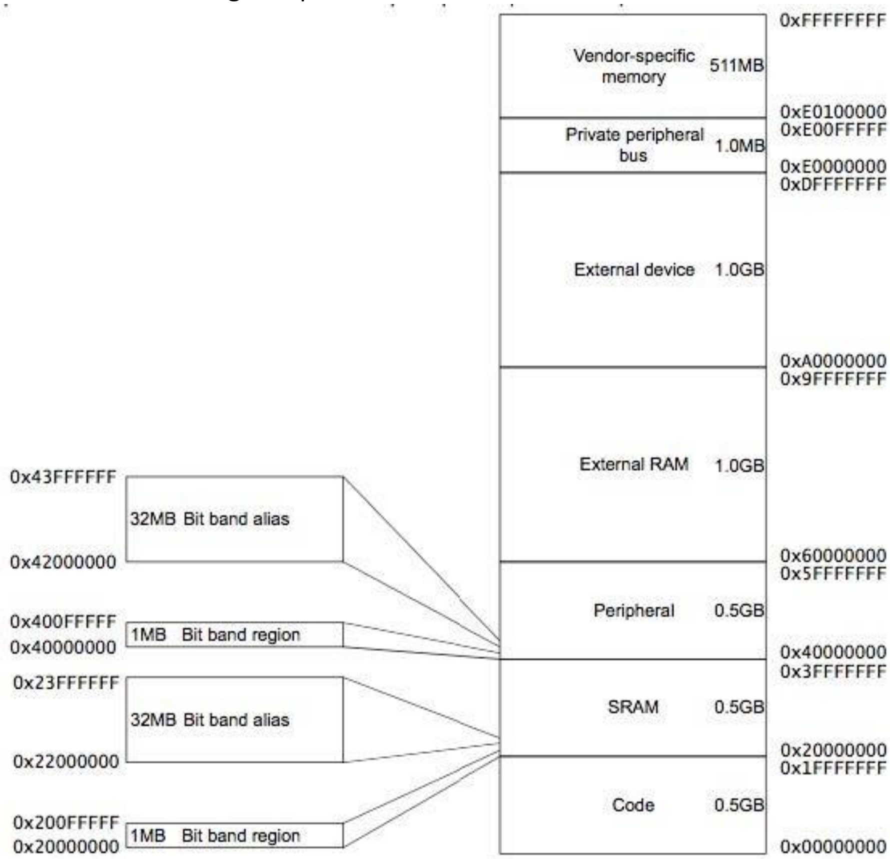
4. Run the mBed Nucleo Example (display time). Set the time to the current time, and combine this with your mBed Nucleo Example (printf) to print the current time to a terminal window on your PC. Capture a screenshot of the terminal window. How much memory is used by this code?
Answer: **35.1 kB flash memory is used by the code. Screenshot is attached in Appendix 3**
This screenshot was taken before the submission of Homework 3P

Output screenshot:



5. Explain the memory model of ARM Cortex-M4 with respect to the code memory, data memory, IRQ handlers and peripherals. Explain with the help of a diagram where required.
Answer: The below diagram is the memory model of the ARM Cortex M4. It has **4GB** of addressable memory (2^32).

- As shown in the figure above the Cortex-M4 has a predefined memory map, and the memory can be accessed by simple memory instructions by other built in peripherals.
- The Cortex-M4 has a fixed 4GB memory map, out of which 0.5GB is code memory which can be used to store program code,this region can also be used for data memory.
- The SRAM is 0.5GB which is used to store data which includes the stack. It can also be used to store program memory
- The peripheral memory is another 0.5GB and can be used to store peripherals and data memory.
- The RAM is of 1GB and can be used as a continuous memory block to store data.
- The Internal Private peripheral bus memory space is 1MB in size and is allocated to store NVIC and other debug components.



6. As a separate project, run either the Dhrystone or the Whetstone benchmark program on your target processor using the code provided, which may need to be modified. If running the Dhrystone, calculate the number of VAX DMIPS.

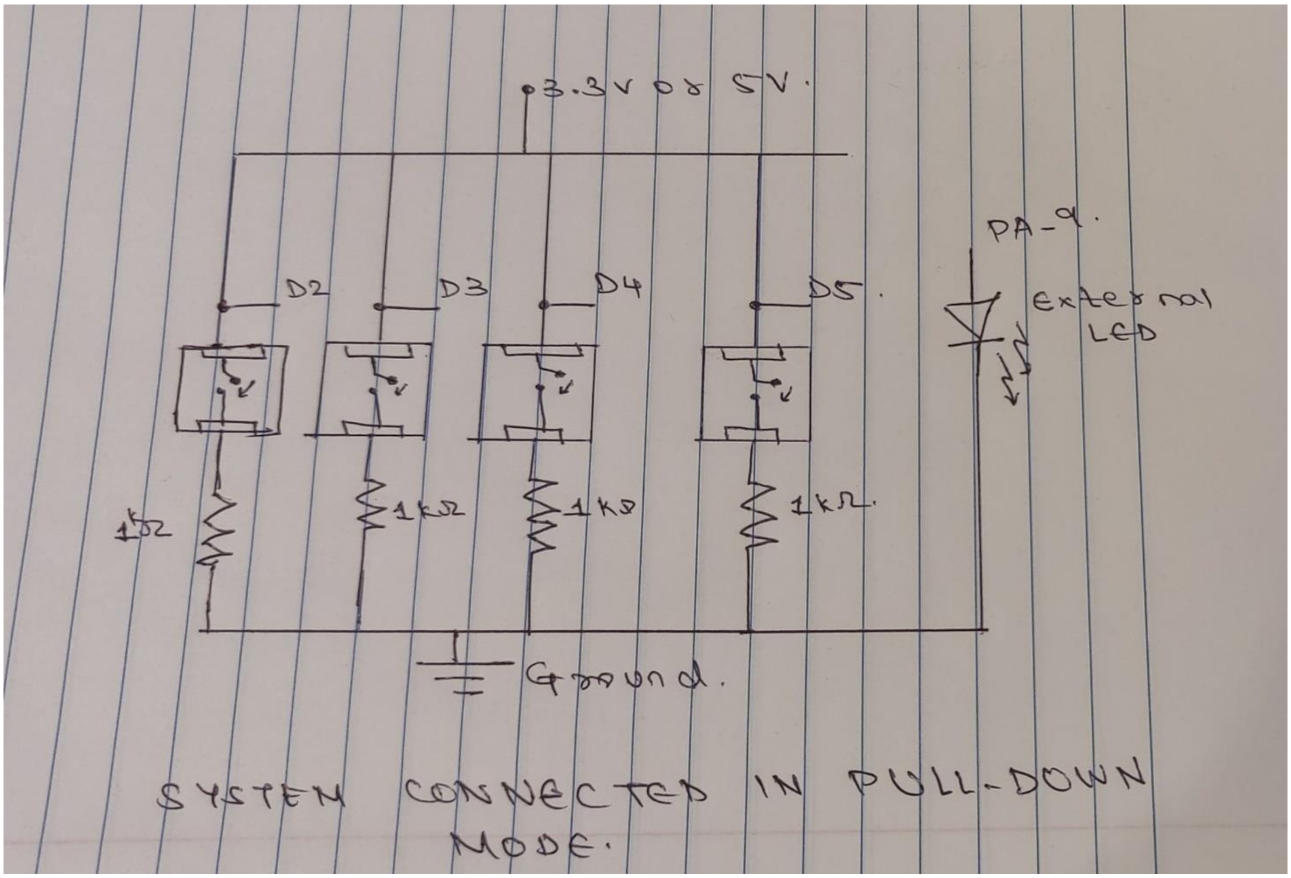
Answer: **We ran Dhrystone benchmark program on the target MCU and found 129.7DMIPS**. **Screenshot is attached in Appendix 4**

4.2 Module 2:

Q1. Try to issue an interrupt on different signal edges (rising edge or falling edge). What changes?

Circuit diagram:

- The circuit diagram illustrates connections between the NUCLEOF401RE evaluation board and peripherals including 4 SPDT button switches and external LED.
- As it is clearly seen, the buttons are connected in pull down fashion which means when the button is pressed the controller will read logic high.



- When button is pressed in interrupt rising mode, the led state is switched as soon as the button is pressed. Whereas when the button is pressed in interrupt falling mode, the led state is switched the user releases the press event from the button.
- The amount by which the variable ‘i’ is incremented, the period of the resulting waveform changes, thereby causing a difference in the frequency (pitch) of the audio signal.
- Variable i is used to determine the duty cycle of PWM, thus determining the volume of the speaker. That is, when we increase the value of i, the volume of the speaker would be turned up. On the other hand, when we decrease the value of i, the volume of the speaker would be turned down.

4.3 Module 3:

We created 4 Threads to:-

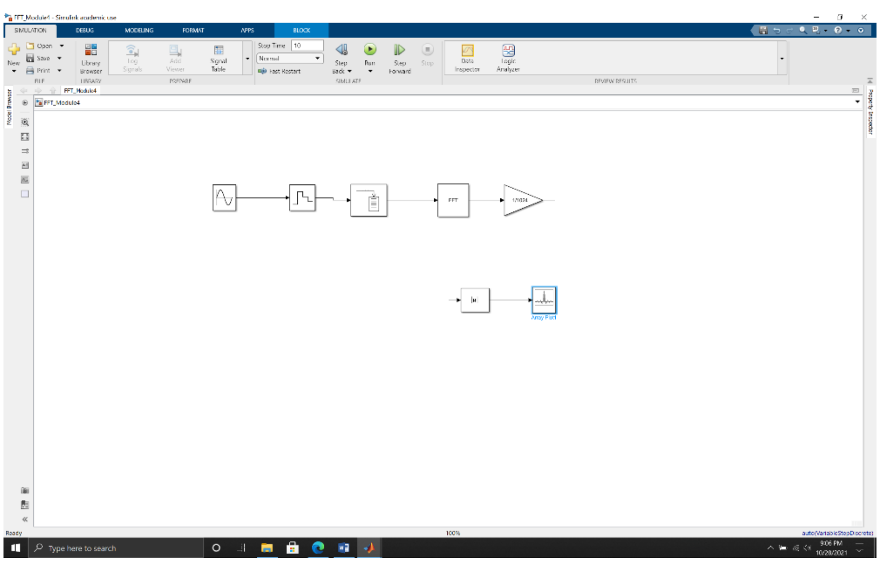
- Display the temperature on the LCD
- Adjust the brightness of the external LED using a potentiometer
- Display an incrementing counter on the LCD.
- Blink an internal LED

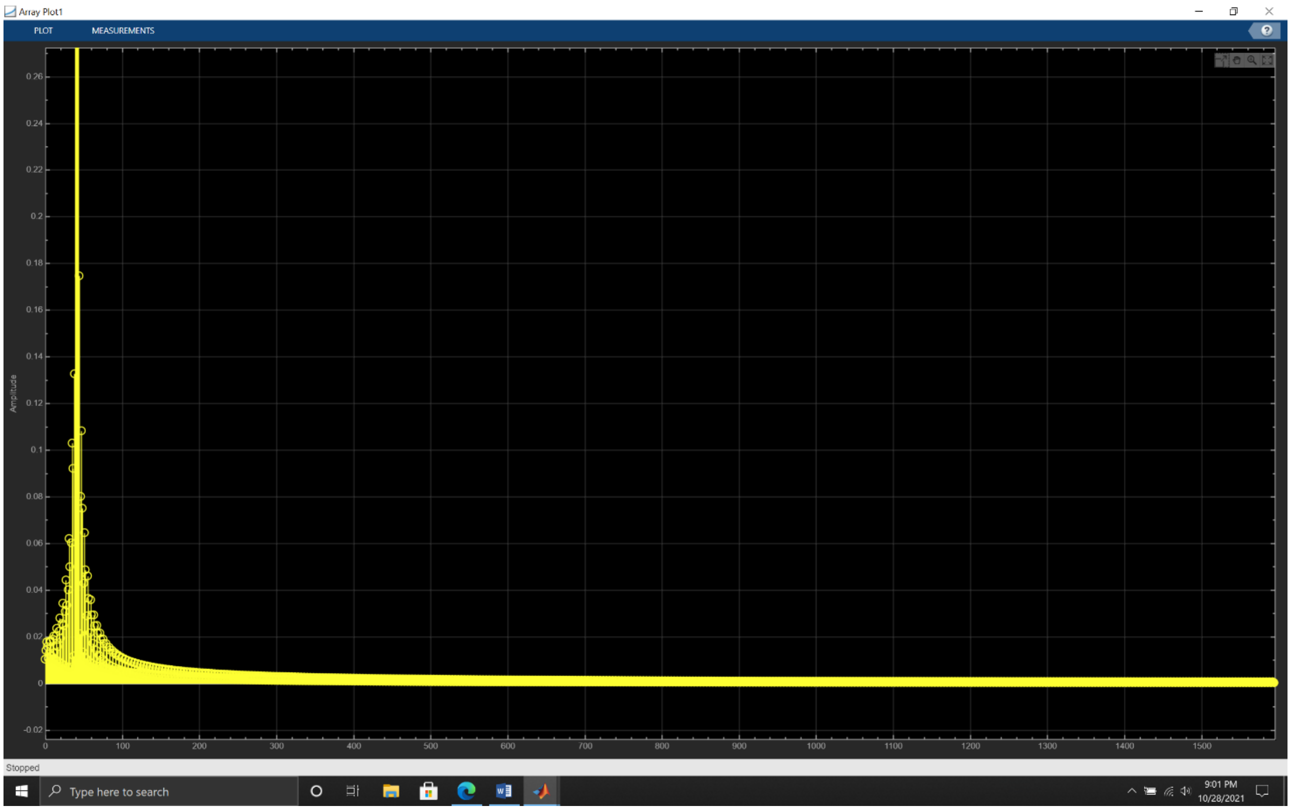
These threads were created using RTOS mbed-os and cmsis_os library. Four threads were created to implement each task as defined. Displaying the data on LCD is shared between two processes that is display the temperature sensor values and counter values. So LCD is shared resources which was prevented by each task during the execution. The mutex lock prevents both the threads to use the resource at the same time, i.e when one thread is writing on the LCD the other thread is waiting for the mutex to be unlocked. Some debugging messages are printed on UART.

Concurrent execution of the threads was evaluated quantitatively, and the STM32F401RET6 MCU was found to perform satisfactorily without any visible latency.
Temperature value and counter values are displayed on LCD. **Screenshot is attached in Appendix 5**

4.4 Module 4:

Here we have developed an algorithm to read the data file which stores the DAC values and then feed that to the autocorrelation system to get the frequency. We have seen how it functions as a signal analyzer and to estimate the frequency of an analog input signal. We saved samples of an audio signal in floating point number format Screenshot attached in appendix 6. The audio signal was of 1004 Hz frequency, sampled with a 10 us resolution. We used an autocorrelation based peak detection algorithm to estimate the input signal frequency, which was found to be 1000 Hz.





In simulation we have seen a peak at 42.5ms which is verified to generate the 1 kHz signal. We observed that the error in frequency estimation is just 0.4% with respect to the sampled 1004 Hz signal. This is seen in the terminal during the execution. The frequency is computed using the inverse of time difference between peaks in the autocorrelation function and so the error persists. There is a lower limit on the achievable frequency resolution because of limitation in the autocorrelation function in time resolution by the sampling period of 10us. At last, we added a LED program to blink it at a rate proportional to the estimated frequency of 1000 Hz.

2. Run Splint or an equivalent code checker (like CppCheck or Cpplint) on your code in 1 of this module. Resolve any errors, explain the warnings.

```
Splint Console [ujjai@LAPTOP-303NE06E - Thu 10/28/2021]
C:\Windows>cd ..

C:\>cd Users

C:\Users>cd ujjai

C:\Users\ujjai>cd Desktop

C:\Users\ujjai\Desktop>cd "Project 2"

C:\Users\ujjai\Desktop\Project 2>cd "Module 2"

C:\Users\ujjai\Desktop\Project 2\Module 2>cd digital_io

C:\Users\ujjai\Desktop\Project 2\Module 2\digital_io>cd src

C:\Users\ujjai\Desktop\Project 2\Module 2\digital_io\src>splint main.cpp
Splint 3.1.2 --- 25 Aug 2010

Command Line: Unrecognized file extension:
                 main.cpp (assuming .cpp is C source code)
  warn when command line file does not have a recognized extension (Use
  -fileextensions to inhibit warning)
main.cpp(24,18): Cannot find include file mbed.h on search path:
                    C:/include;C:/local/include
  Preprocessing error. (Use -preproc to inhibit warning)
Preprocessing error for file: main.cpp
*** Cannot continue.

C:\Users\ujjai\Desktop\Project 2\Module 2\digital_io\src>
```

The above error is a parse error. BusOut and BusIn is a C++ class. Buttons and myleds is declared as an object of BusIn and BusOut. Buttons and myleds are defined previously which represents the specific pin where Buttons and Leds are located.
Splint FAQs says that they have limitation of current splint version which gives error and they will be fixing it in future releases.

4. Estimate the processor load in % of CPU cycles.

After running the elapsed time during execution, the overall CPU cycle usage comes out approximately to be around 33us that means the CPU load is 32-35%.

4.5 Module 5:

After comparing the startup used in module 1 with the auto generated code from STM32CubeMx these were the observations:

1) The code generated by STM32CubeMX is vendor code and it uses the Hardware Abstract layer HAL to generate all the required configurations.
2) It provides all .c and .h files where the function definition and prototyping and can be seen by the developer
3) It gives more flexibility to change the clock configuration or any peripherals configuration using the GUI. Screenshot is attached in Appendix 7

5) List of project deliverables

Files submitted are as follows:
1. Detailed Technical Report
2. Executive Summary Report
3. Bill of Materials(BOM) in excel
4. Module 1-4 design files
5. Dhrystone Benchmark test run result
6. Simulink data file
7. Doxygen output files for Module 3

6) Recommendations: GO/NO-GO

The STM32F401REY6 MCU was used to evaluate given requirements by Keithley Corporation in their RFS for Signal Analyzer would be **recommended-GO as a signal analyzer product**.

1) Hardware interfaces were done to check the MCU capability and performance. Interrupt driver I/O, multiple thread were executed.
2) This module doesn't have DAC functionality and inbuilt temperature sensor feature which can be managed at external cost and still won't exceed the budget of $20
3) While testing the ability to interface the LEDs and Switches, the MCU has good response for interrupt driven code and real time functionalities.
4) The controller can also be used to generate PWM in range of audio signal. Harmonic analysis capabilities were evaluated by implementing the peak detection algorithm. Based on these test results we can give the STM32F401RE a go, and can confirm it is capable of working as a signal analyzer.
5) The software development platform was Mbed online compiler which was easy to code in and available as open source. The required DMIPS was 100 and after running the Dhrystone benchmark, we found out it was 130 DMIPS which satisfies the requirement.

7) References

- STM32F401RE Reference Manual
- STM32F401RE Datasheet
- STM32 Nucleo-64 user manual
- STM32 HAL library user guide
- https://os.mbed.com/handbook/Homepage
- Full API list - API references and tutorials | Mbed OS 6 Documentation
- https://www.sciencedirect.com/topics/engineering/memory-space
- https://www.mathworks.com/matlabcentral/answers/780027-vector-scope-simulink-r2020b
- https://www.mathworks.com/help/dsp/ug/transform-time-domain-data-into-the-frequency-domain-using-the-fft-block.html
- How to Generate a Sine Wave, Use FFT and IFFT in Simulink - YouTube
- Simulink matlab FFT of a signal||4th semester experiment in DSP|| VTU experiment - YouTube

8) Project Staffing

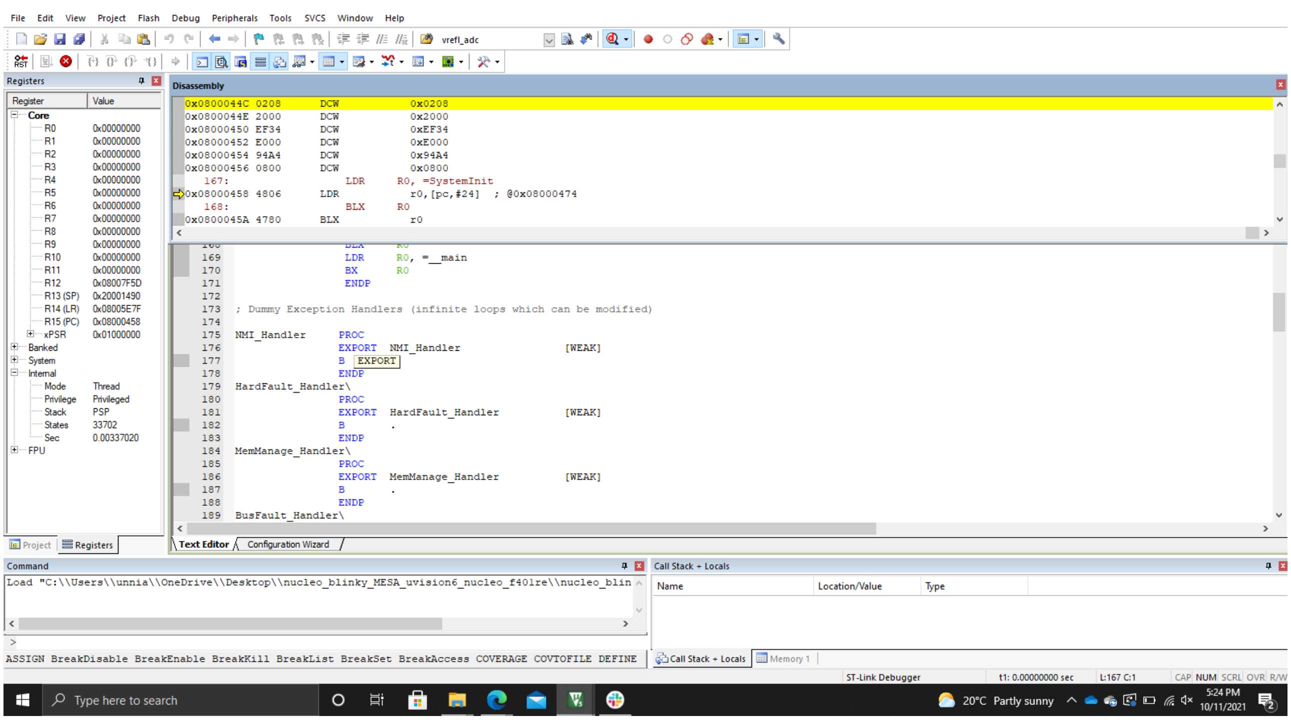
Instructor:

Prof. Timothy Scherr
Teaching Associate Professor
EMBEDDED SYSTEMS ENGINEERING
University of Colorado Boulder
Email: Timothy.Scherr@colorado.edu

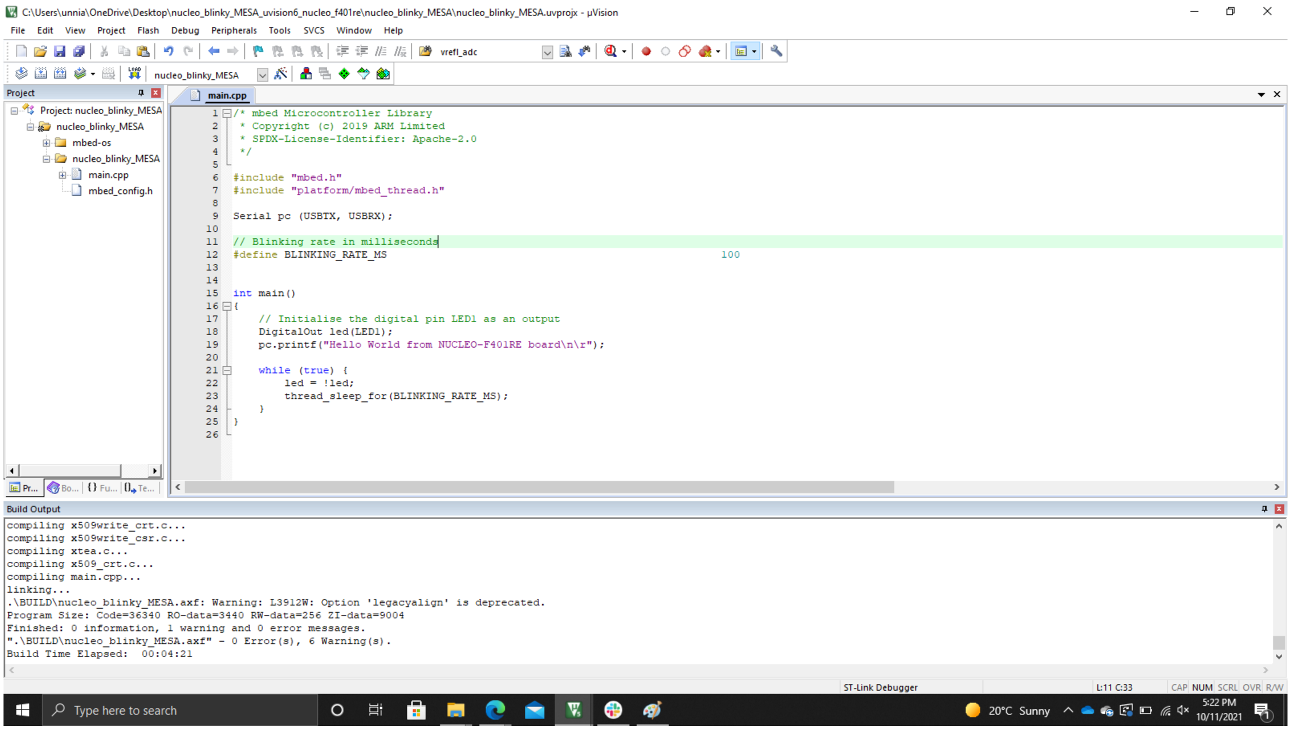
Taher Ujjainwala
Graduate Student University of Colorado Boulder
Email: tauj5361@colorado.edu
Phone: +1 (720)-808-6561

Steve Christensen
Graduate Student University of Colorado Boulder
Email: stch2306@colorado.edu
Phone: +1 (859)-321-5539

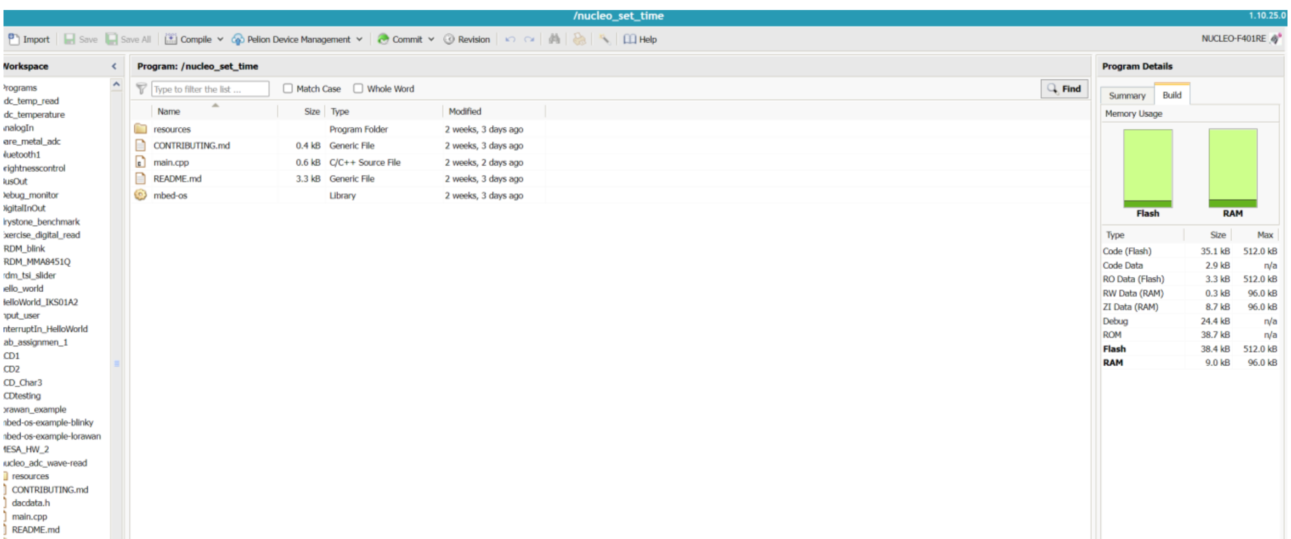
Appendix 1: Reset Handler starting address



Appendix 2: Code size for Blinky project

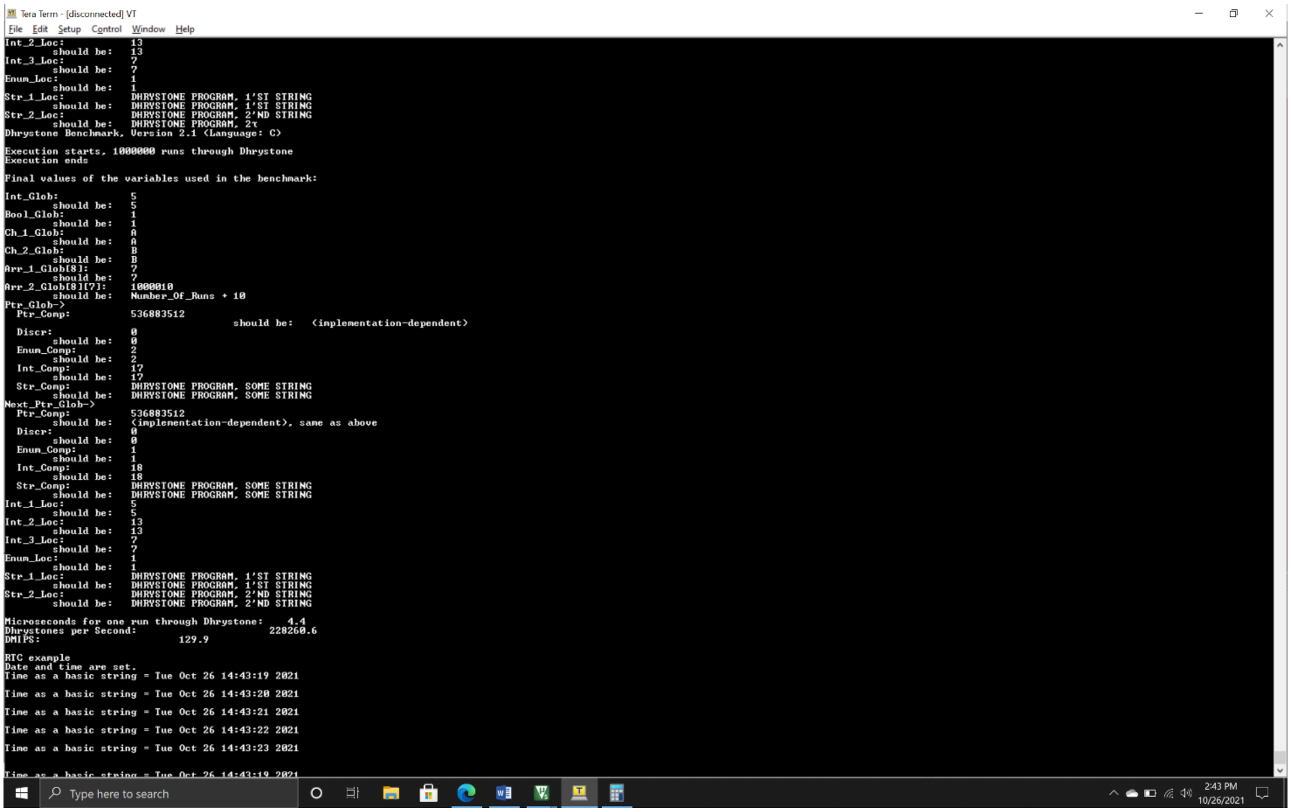


Appendix 3: Code size

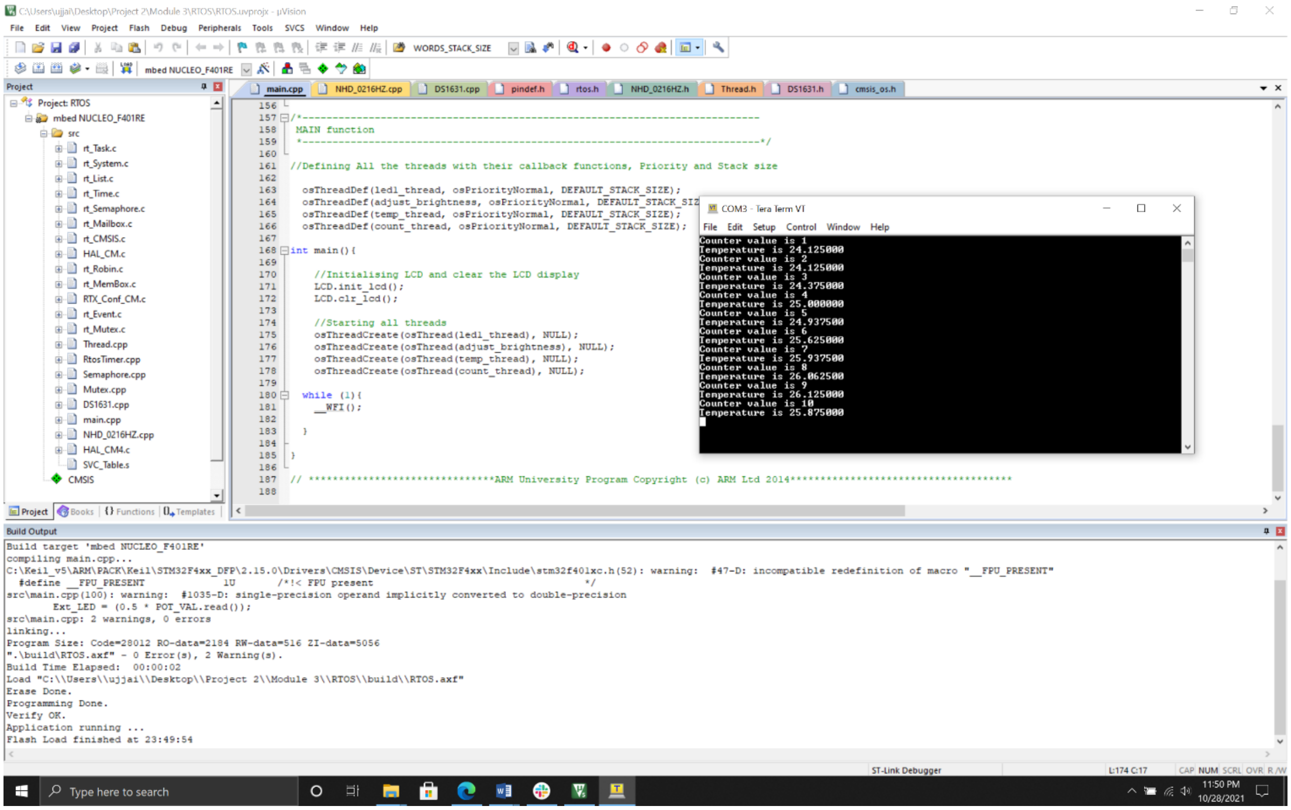


Appendix 4:

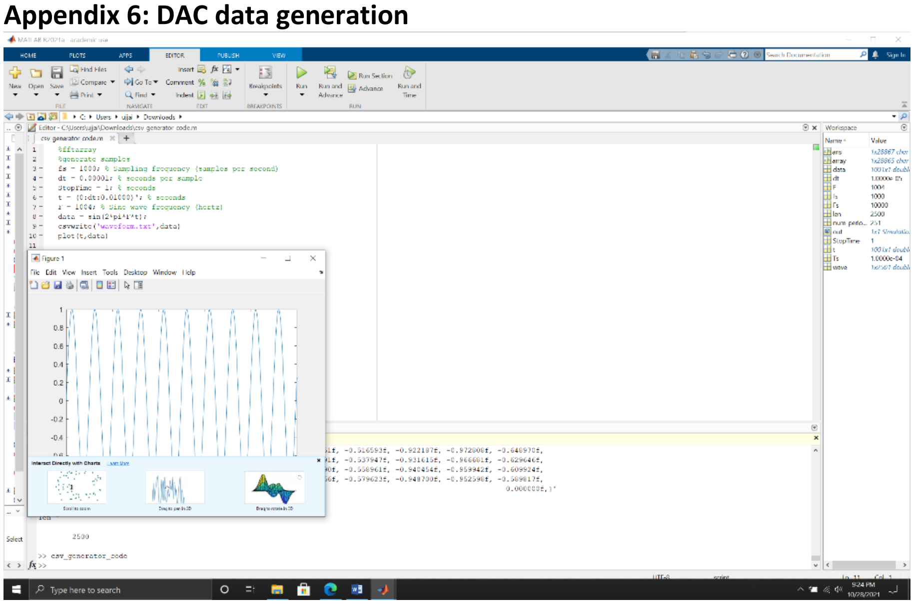
Evaluation of STM32F401RET6 with Dhrystone:



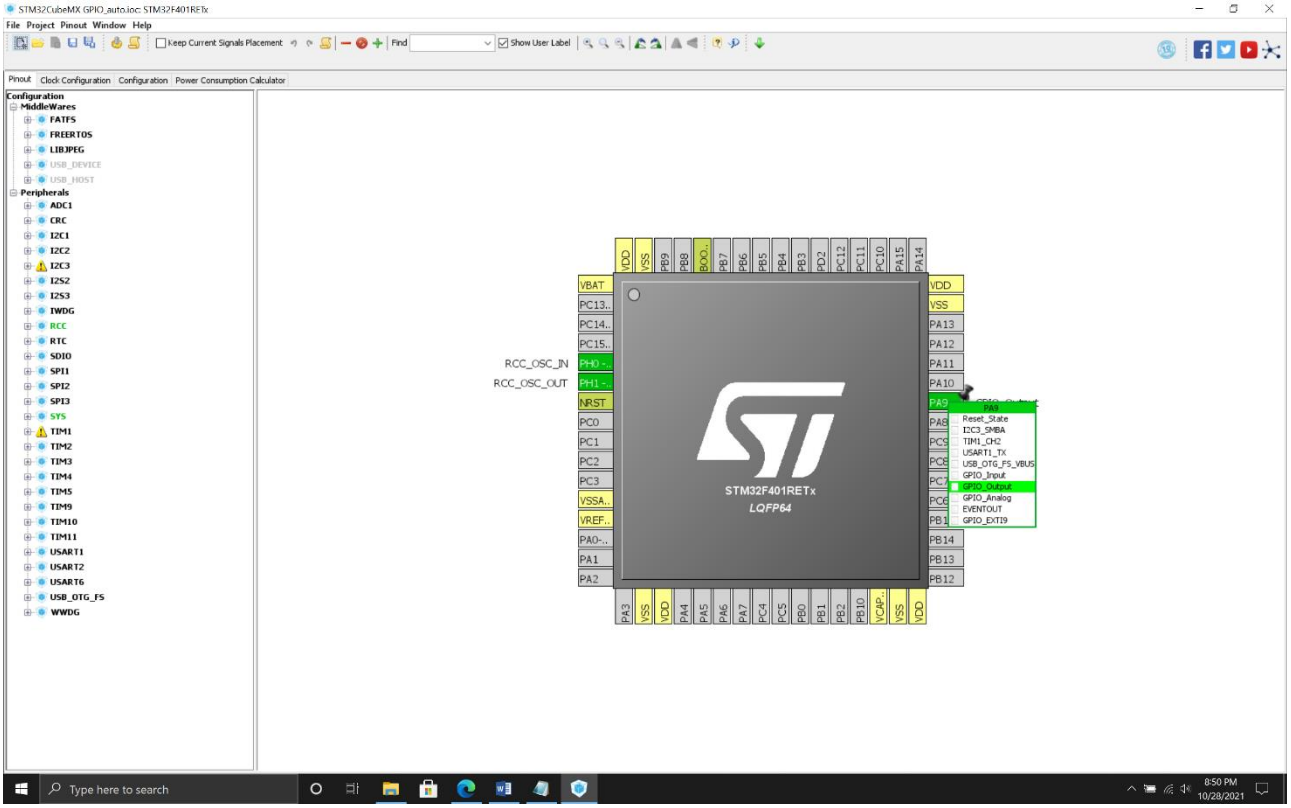
Appendix 5: Temperature and Counter value on UART for Debugging



Appendix 6: DAC data generation



Appendix 7:


STM32CubeMX GPIO_auto.ioc: STM32F401RETx
File Project Pinout Window Help
Keep Current Signals Placement
Find
Show User Label
Pinout
Clock Configuration
Configuration
Power Consumption Calculator
Configuration
MiddleWares
FATFS
FREERTOS
LIBJPEG
USB_DEVICE
USB_HOST
Peripherals
ADC1
CRC
I2C1
I2C2
I2C3
I2S2
I2S3
IWDG
RCC
RTC
SDIO
SPI1
SPI2
SPI3
SYS
TIM1
TIM2
TIM3
TIM4
TIM5
TIM9
TIM10
TIM11
USART1
USART2
USART6
USB_OTG_FS
WWDG
RCC_OSC_IN
RCC_OSC_OUT
STM32F401RETx
LQFP64
PA9
Reset_State
I2C3_SMBA
TIM1_CH2
USART1_TX
USB_OTG_FS_VBUS
GPIO_Input
GPIO_Output
GPIO_Analog
EVENTOUT
GPIO_EXTI9
Type here to search
8:50 PM
10/28/2021