General purpose data acquisition system using the STM32 microcontroller and FreeRTOS

LM118 – Bachelor of Engineering in Electronic and Computer Engineering

Project Interim Report

Qinyuan Liu
20137095

Dr Ian Grout
<Submission Date>

Abstract

Keywords: RTOS, STM32, Embedded system, data acquisition,

The rapid growing of complexity of embedded systems has made precise, lightweight, agile data acquisition increasingly important, especially in areas like industrial automation and robotics. Using the STM32 microcontroller combined with FreeRTOS, the project focuses on creating a data acquisition system that will have analogue and digital IO, with communication functionality.

STM32_CUB_IDE (C) and NUCLEO-L476RG (Testing platform) will be used for this project, along with Target 3001 Beta-Ver (Schematics and PCB layout).

The primary implementation of this project is the prototype system, which is expected to be done at the end of the autumn semester, The winter semester will be focused on PCB design,

Presentation and final report will be the main focus of the final semester.

~~This report will follow the standard ECE FYP template. The abstract will give the reader an overview of the project in both its technical and practical scope, while the introduction will provide a detailed technical overview and the motivation for the project.~~

Declaration

This interim report is presented in part fulfilment of the requirements for the LM118 Bachelor of Engineering in Electronic and Computer Engineering **Bachelors Project**.

It is entirely my own work and has not been submitted to any other University or Higher Education Institution or for any other academic award within the University of Limerick.

Where there has been made use of work of other people it has been fully acknowledged and referenced.

Name **Qinyuan Liu**

Signature _____

Date _____

Table of Contents

Chapter 1 Contents

List of Figures

Replace this text with a table showing figure numbers, captions and page numbers

Will add photo in appropriate place later

List of Equations

Replace this text with a table showing equation numbers, captions and page numbers.

List of Tables

Replace this text with a table showing table numbers, captions and page numbers

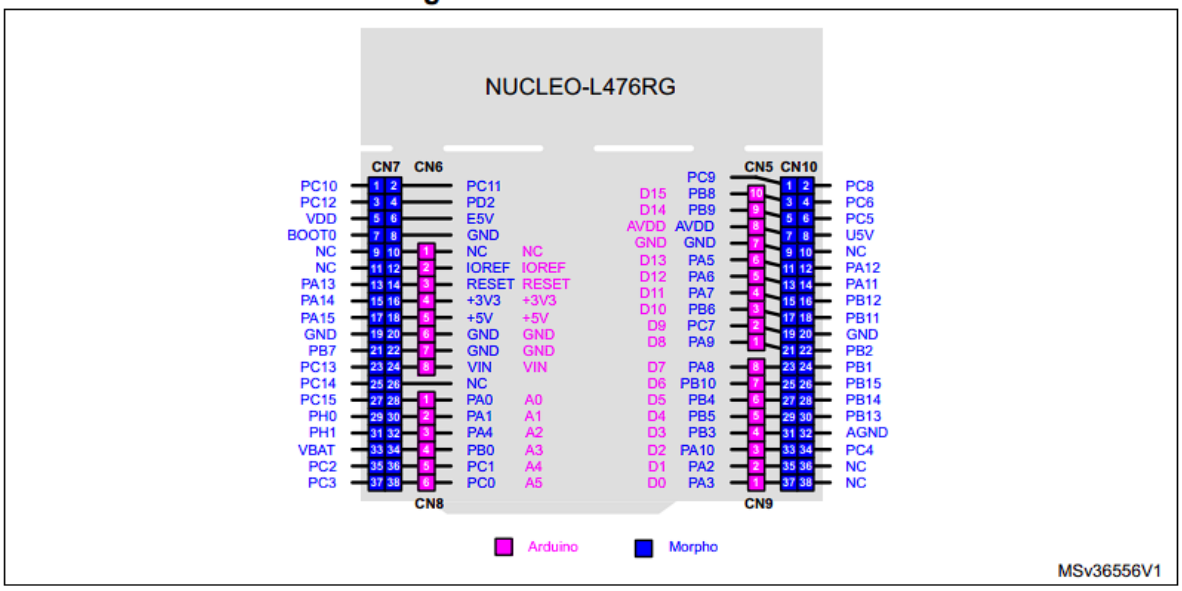
Chapter 1 Introduction

RTOS has wide applications in embedded system development, from robotic control in the Mars lander to data applications in DSP processing. This system enables an ARM/AVR chip to perform many more complex tasks in a Time accurate tasks scheduling system. The goal of the project is to develop a general-purpose data acquisition system with the STM32, based on FreeRTOS. RTOS will provide a stable platform for control over timing and communication, and control over an array of ADC/DAC and 2 digital IO chips, handling communication with a PC and potential Peripherals.

The project will be implemented in 3 steps: a breadboard prototype for proof of concept, a PCB prototype for advanced design integration, and a final STM32 chip on board design. The breadboard prototype will be first made as a proof of concept that will provide a solid foundation for the following steps: Following the breadboard prototype, A PCB prototype that connects with the NUCLEO board will be developed as a demonstration of the more advanced design capability of the student, while also providing a better integration of the electronic design of this project. The final step will be the implementation of a full STM32 chip-on-board design, to elevate this project into an industry implementation-ready, advanced electronic design project.

[1]

Figure 24. NUCLEO-L476RG



As a low-speed general purpose data acquisition system, this project has a wide range of applications in the industry. The project can be adapted into various uses in the industry, such as automation engineering, civil engineering, machinery, vehicle design, safety systems, and fields.

For example, this project can be put into the battery management function of the battery pack as an application. The ADC array provides the voltage reading of each battery, each DAC provides the control voltage for the power semiconductor of the charging circuit, the digital IO realizes a simple display battery status display, the PC serial port communication provides the error detection and control functions and calculates the battery algorithm based on this system to provide a highly integrated solution. In another example, in industrial vibration detection, many large rotating mechanical equipment require continuous vibration detection, such as wind turbines, thermal power turbines, and large mechanical engines. The resonant frequency of these systems is usually less than 1khz. This project can detect vibration signals through ADC sampling, use the relevant communication protocols to communicate with PLC, provide feedback loops, and realize real-time monitoring and analysis of mechanical equipment.

10*2 Header for output in DAC

Shield is too big

2PCB A shield,

Chapter 2 Background

RTOS (Realtime operating system) for embedded systems was developed in the 80s with the advancement of micro processer. The emergence of RTOS, in parallel of other none-time sensitive system, prove to be critical for the embedded system in latter decades of development. The time sensitivity of the RTOS provides a bedrock foundation for a majority of the embedded system design. With the growing complexity of embedded systems, more dedicated RTOS platform appeared. In the early 90s, RTOS has grown form a core kernel environment. To a matured development platform.

RTEMS (Real-Time Executive for Missile Systems) is the one if the first mature RTOS emerged in military industry, it was developed around the late 80s and provides support for [1] ADA[2] programming language and is still widely used today.

In the 1990s, the development of more powerful microprocessors and the growth of the internet led to the development of more sophisticated RTOS, such as VxWorks and QNX. These operating integrity systems were designed to handle more complex real-time applications, such as telecommunications, network routing, and multimedia [3].

[1] https://ftp.rtems.org/pub/rtems/releases/5/5.1/docs/html/doxygen/RTEMSPreface.html Because of this shortcoming in the Ada programming language, software developers in research and development and contractors for project managed systems, are mandated by technology to purchase and utilize off-the-shelf third party kernel code. The contractor, and eventually the Government, must pay a licensing fee for every copy of the kernel code used in an embedded system.

[2] ADA: A High-level programming language
https://intechhouse.com/blog/real-time-operating-system-im-embedded-systems/#:~:text=The%20RTOS%20History,-The%20history%20of&text=One%20of%20the%20earliest%20RTOS,industries%2C%20including%20aerospace%20and%20defence.

RTOS today has wide application in embedded systems with great promotion and adaptability. Its applications involve aerospace, defence, automobile, communication and multimedia. [site 3form last page]

In our application, we choose CMSIS-RTOS V2 provided by STM32_cube_IDE as our RTOS platform and develop on its basis. CMSIS-RTOS V2 provides generic RTOS interfaces for Arm® Cortex® processor-based devices. It provides a standardized API for software components that require RTOS functionality and it's well integrated into our IDE3.

RTOS is used in this project for two reasons: system time sensitivity and system complexity. As a general-purpose data acquisition system working at 1kHz scanning rate, the project needs to solve the problem of 'jitter' in digital signal processing (DSP). A key characteristic of an RTOS is the level of its consistency concerning the amount of time it takes to accept and complete an application's task; the variability is "jitter". The rigid RTOS Scheduling proved the perfect solution for the common problem of 'jitter' in DSP sampling. In terms of system complexity, this project needs to implement 16 ADC and 16DAC analog signal inputs, as well as 16 * 16 digital signal I/O. On top of this, the USART communication and control functions need to be implemented. Such complexity requires a rigorous multi-level priority task management system. Therefore, RTOS's Multitasking Scheduling, Timer Management, interrupt Management and Inter-Task Communication and Synchronization provide a suitable platform.

On top of RTOS, this project is developed based on STM32. STM32 is a common microcontroller architecture and is widely used in embedded development. We use the common STM32-nuclo development board as the experimental platform and use the mainstream STM32cube-IDE to develop this project. The project will also try to implement PCB printing technology, use Target 3001, further advance the student's capability of electronic drawing specifications, and PCB design.

[3] https://arm-software.github.io/CMSIS_5/RTOS2/html/index.html

Chapter 3 Technical details - Software

STM32 Background and Hardware Configuration

Introduction to STM32 Cube:

The STM32Cube IDE is a complete development platform, CUBE IDE offers intuitive hardware configuration in the 'ICO' interface and generates a skeleton code using the ICO configuration for the device. [screenshot of ICO] The CUBE IDE used a C-based scripting language, offering HAL (hardware abstraction layers) to simplify hardware interaction by providing an abstraction layer between the application and microcontroller peripherals, which greatly reduced the complexity of the software design of this project.

STM32 and RTOS Integration:

The STM32Cube IDE integrated CMSIS-RTOS in its environment, the project uses the CMSIS-RTOS V2 distribution that's offered in the IDE as the on-chip operating system of this project. [Screenshot of the CMSIS-RTOS], the project [make a block diagram of the task management of this project's instance, a hierarchy diagram] and we write more from the block diagram.

The syntax of the RTOS is relatively intuitive, RTOS skeleton code is generated by the STM32_cube_IDE, and it follows a similar syntax as the interrupt function (ISR) offered by the HAL. (insert code snibbit from the RTOS Example Project). For instance, a task-creating function can be declared inside #user_code_x eg:

```
   xTaskCreate creates a new task for the RTOS.

   Arguments:
   - TaskFunction: The function that implements the task.
   - "TaskName": A descriptive name for the task, useful for debugging.
   - 128: Stack size allocated for the task (in words, not bytes).
   - NULL: Pointer to parameters to pass to the task (NULL if no parameters
are needed).
   - 1: Priority of the task (higher numbers indicate higher priority).
   - NULL: Task handle (optional, can be used later to control or query the
task).
*/

xTaskCreate(Task_Read_ADC,    // Task function
            "ADC Reader",     // Name of the task for debugging
            128,              // Stack size (128 words)
```

```
            NULL,            // No parameters passed to the task
            1,               // Task priority
            NULL);           // No task handle used

/*
   This creates a task named "ADC Reader" with a stack size of 128 words,
priority 1.
   The task will run the function Task_Read_ADC, which reads the ADC values.
*/

// Example task function to read ADC data
void Task_Read_ADC(void *pvParameters) {
    while(1) {
        uint32_t adc_value = HAL_ADC_GetValue(&hadc1);  // Read ADC value
        xQueueSend(xQueue, &adc_value, portMAX_DELAY);  // Send value to a
queue
        vTaskDelay(pdMS_TO_TICKS(1000));  // Delay for 1000ms (1 second)
    }
}
/*
   In this function, the task reads from the ADC and sends the data to a
queue.
   vTaskDelay is used to make the task wait 1 second before repeating.
*/
```

Note: I am planning to use Jason in both the Synopsis UART communication and the internal data transfer in RTOS, need to look into

int data = 100;

xQueueSend(xQueue, &data, portMAX_DELAY);

Okay what's next?

Block Diagram Explanation

The Block Diagram Shown in Appendicitis C

~~Fist introduce stm32 cube~~

~~And on RTOS~~

~~And on the sytex on stm32 on RTOS~~

Block diagram explanation

And chip explanation

And overall architecture

A diagram is needed here

And a sudo code

Chapter 4 Technical details – Hardware

Chapter 5 Action plan

~~The project will be implemented in 3 steps: a breadboard prototype for proof of concept, a PCB prototype for advanced design integration, and a final STM32 chip on board design. The breadboard prototype will be first made as a proof of concept that will provide a solid foundation for the following steps: Following the breadboard prototype, A PCB prototype that connects with the NUCLEO board will be developed as a demonstration of the more advanced design capability of the student, while also providing a better integration of the electronic design of this project. The final step will be the implementation of a full STM32 chip-on-board design, to elevate this project into an industry implementation-ready, advanced electronic design project.~~

The current development done is as follows: The preliminary work is done in the first 2 weeks, to ensure that the development is correctly installed, and the key concepts have been discussed with the supervisor. Week 2 to week 5 initiates the implementation, including using STM32_CUBE_IDE to familiarize with freeRTOS, produce schematics with Target 3001, and have the component (see schematic) and the NUCLAEO-L476RG ready. And week 5-7 is spent on preparing the intermedia report and finishing the first schematic.

The development will take place over two semesters, using the STM32_CUBE_IDE (C), NUCLEO-L476RG (for testing), and Target 3001 Beta Version for schematics and PCB layout. The initial semester will focus on coding and prototyping, with any additional

workload being addressed in the winter semester. The Prototype Breadboard and coding are aimed to be finished by the end of the autumn semester, and the first PCB design is to be sent for printing by the end of the fall. The following winter break will be focused on implementing the semester 2 mentioned above. The final semester will involve system verification and ensuring the project meets industry-level standards for robustness and versatility. In parallel, the presentation of the project will be a key focus, as clear technical communication is crucial both for grading and professional success.

Pin table on nucleo (as in the pin configuration that should be put in as a table)

Chapter title (Introduction and motivation shole be one)

Technical, 2 parts, software and hardware (Done)

Deliverables: Finish the Report

1. The Schematics of each block.
2. Complete the Gannt chat following the action plan.
3. Write with the reader in mind.
4. Finish the Miscellaneous.

Next meeting is this Tuesday at 4PM Online.

Chapter 6 Conclusions and future work

Significant progress has been made in understanding and developing the foundations of the project, a schematic has been made while the bread board testing prototype is under development.

The schematic design helped with the decision of chip selection and components wiring, which is critical for prototype design. The bread bord prototype, once complete, will help provide the test bench for the STM32 RTOS implementation, RTOS implementation will be challenging. The above is the current work content. In the next winter vacation, if the prototype is perfected smoothly, we will carry out the PCB design work, which involves from Schematics to PCB layout, PCB printing, assembly, among which PCB design will be the most challenging part of the project. If the PCB design goes well and there is time, the PCB

design of the onboard STM32 will be carried out during the winter vacation and the first to third weeks of the spring semester.

Chapter 7 References

[1] STMicroelectronics, "NUCLEO-XXXXCX, NUCLEO-XXXXRX, NUCLEO-XXXXRX-P, NUCLEO-XXXXRX-Q: STM32 Nucleo-64 boards," Data brief, Rev 19, June 2024. Available: https://www.st.com

[2]

[3] IntechHouse, "Real-Time Operating System in Embedded Systems," IntechHouse Blog, June 14, 2023. [Online]. Available: https://intechhouse.com/blog/real-time-operating-system-im-embedded-systems/. [Accessed: 20-Oct-2024].

[]

Appendices

TBD

Put the weekly reports here??????????!!!!!!!!!!

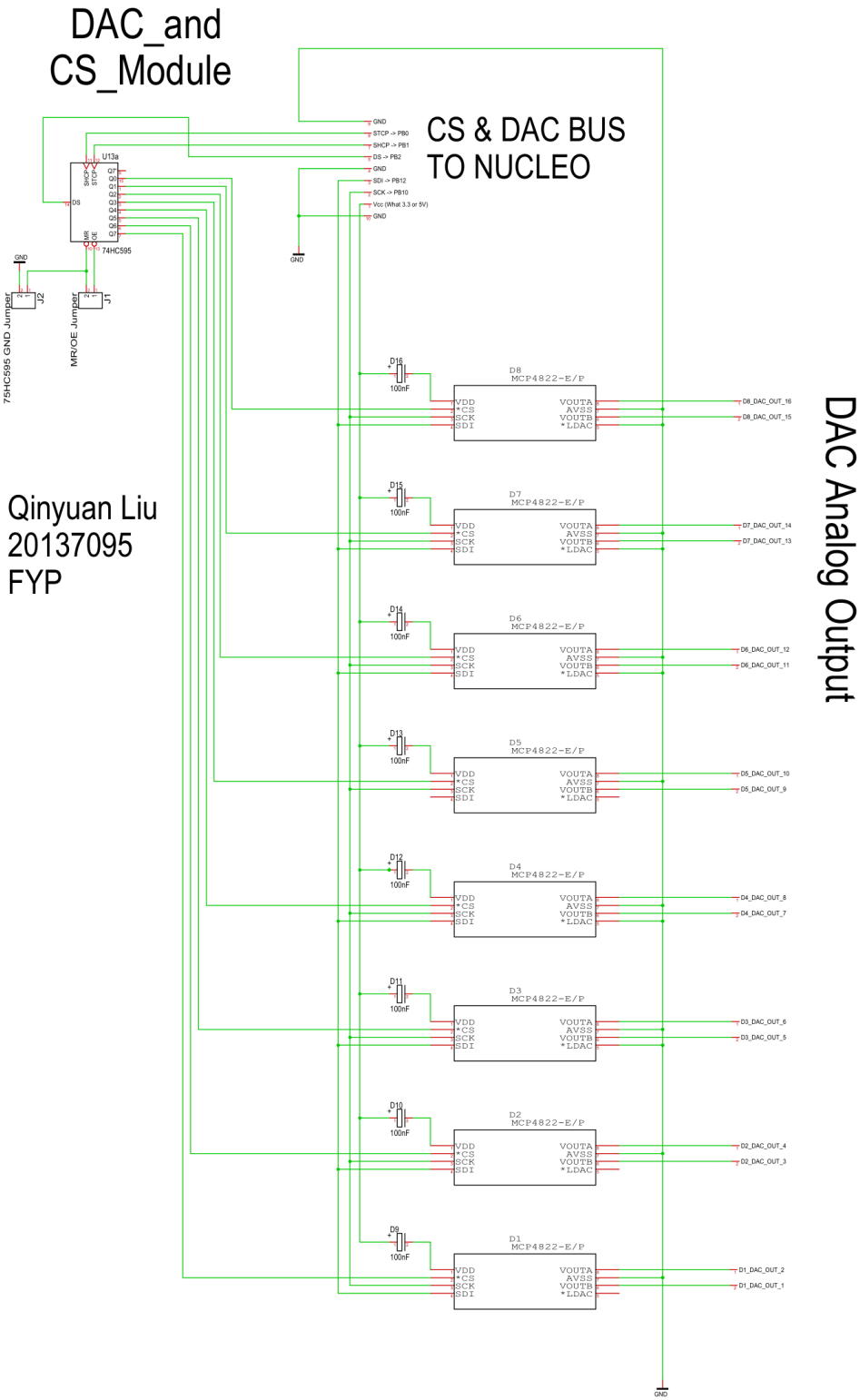
Appendix A: Project Gantt chart

TBD

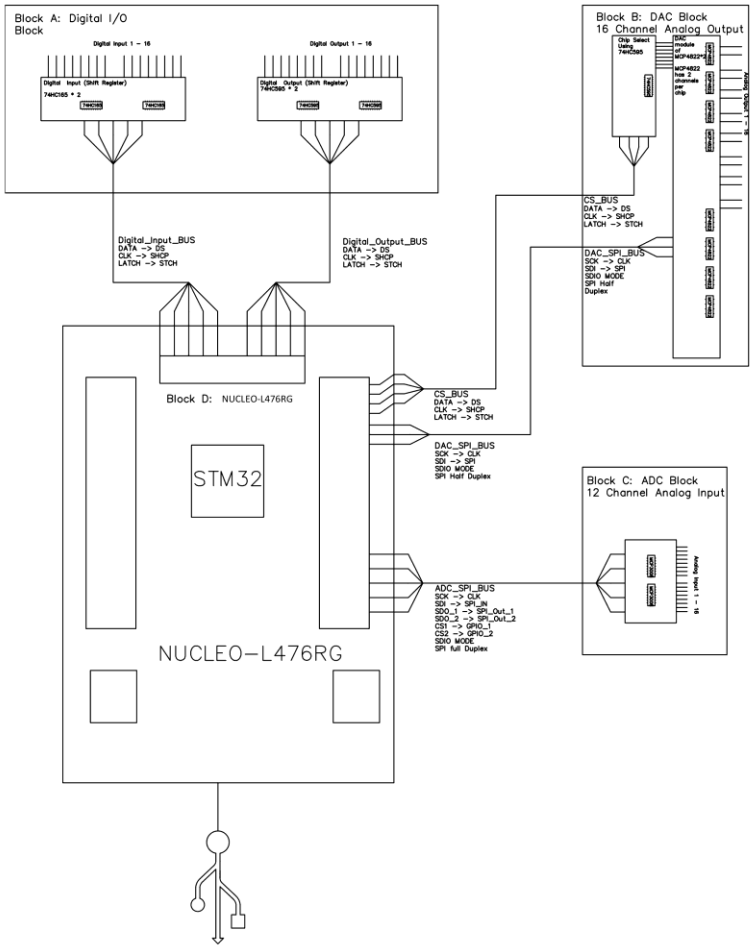
Appendix B: Interim presentation slides

Replace this text with the project interim presentation slides (2 slides per page format). Appendix to start on odd number page.

Appendix C: Drawing



FYP Block Diagram
Qinyuan Liu 20137095



Appendix D: Weekly Reports

This appendix contains a collection of weekly reports that document the progress and deliverables of the project, communicated through regular email updates. The reports were prepared on a weekly basis, typically submitted at the start of each week, summarizing progress made on the tasks from the previous week and setting deliverables for the following week.

****Week 1 (9/9)****

Hi Ian,

Good morning and happy Monday.

I’m sending a bit of material in preparation for the meeting today, feel free to give it a read if you got bit of time:

Last Meeting Notes:
1. **Key Deliverable for Next Monday:**
   - Implement a Timer ISR on STM32:
     - Use a similar approach to AVR Dude.
     - Combine HAL functions and direct register manipulation.
     - Watch the Digikey video on Timer ISR.
     - Refer to FreeRTOS and STM32 official documentation for guidance.

2. **Timing Sensitivity Regarding FreeRTOS:**
   - Understand Integration:
     - Study how Timer ISR works with FreeRTOS.
     - Use the standard/default FreeRTOS configuration.

3. **Next Meeting:**
   - Scheduled for Monday at 12 PM.
   - Discuss the first report and request a holistic view of the project.

To-Do List:
- **High Priority:**
  - Watch the Digikey Video on Timer ISR.
  - Implement Timer ISR on STM32: Begin with combining HAL functions and direct register tweaking.

- **Medium Priority:**
  - Study FreeRTOS and Timer ISR Integration: Understand how Timer ISR impacts FreeRTOS tasks.
  - Refer to Official Documentation: FreeRTOS and STM32 docs for accurate implementation details.

- **Low Priority:**
  - Prepare for Next Meeting: Focus on the first report and a holistic view of the project.

Subject: Follow-up on FYP Project and Key Meeting Notes

Hi Ian,

I hope you're doing well. I'm sending this email to open a thread regarding the FYP project, in the hope of easing your inbox for future updates. Below are the key notes from our meeting today. Please let me know if you have any additions or corrections.

Key Notes:
- **Key Deliverables:**
  - ISR Timer Implementation
  - FreeRTOS LED Toggling Demonstration

Work to be Done:
1. Study the `static` keyword in C and its implications on ISR using "C How to Program."

2. Implement the timer interrupt routine and understand its relationship with FreeRTOS.
3. Review the real-time system book for insights into system timing and interrupts.
4. Browse Mars Lander FreeRTOS for additional relevant examples and resources.
5. Reflect on the implications of using FreeRTOS and why it's the chosen solution for this project.
6. Review the Analog Development Kit MK1 for relevant insights.

I will do my best to ensure that the deliverables are sent to you before the end of Wednesday, and I'll reach out for any necessary updates. Looking forward to seeing you next Monday.

Thank you again for your time and for this morning's meeting.

Best regards,
Qinyuan

****Week 2 (9/16)****

Hi Ian,

Minutes from Last Meeting:

Key Notes:
- **Key Deliverables:**
  - ISR Timer Implementation
  - FreeRTOS LED Toggling Demonstration

Work to be Done:
1. Study the `static` keyword in C and its implications on ISR using "C How to Program."
2. Implement the timer interrupt routine and understand its relationship with FreeRTOS.
3. Review the real-time system book for insights into system timing and interrupts.

4. Browse Mars Lander FreeRTOS for additional relevant examples and resources.
5. Reflect on the implications of using FreeRTOS and why it's the chosen solution for this project.
6. Review the Analog Development Kit MK1 for relevant insights.

Best regards,
Qinyuan

****Week 3 (9/23)****

Hi Ian,

I’m sending you the list of deliverables for next Monday to review, just in case I’ve missed or misunderstood anything on my end.

Deliverables:
1. **Produce Schematics:**
   - Requirements: SPI for MCP3008 * 2, MCP4822 * 8 (what resolution do we need?)
   - Use multiplex, demultiplex, and shift register to reduce pin count.
   - Software: Torgit PCB—Beta Layout.
   - Potentially I2C capability.

2. **Learn SPI & I2C Protocols:**
   - Digikey has great content for this.

3. **Gantt Chart:**
   - Review the Gantt chart from the Leap Motion project and use it as a template.
   - Keep fewer than 10 tasks per semester, and ensure the rest are sub-tasks.

4. **System Tick/Timing Architecture in FreeRTOS:**

- Focus on FreeRTOS and STM32's documentation page.
- Reference: Real-Time Systems by Jane W.S. Liu.

5. **Mars Lander Archives: **
- Review how the Mars Lander Viking 1 & 2 handled OS scheduling and priority inversion.

6. **Miscellaneous: **
- We're using the MK2 as an oscilloscope.

Thank you for your time! I hope you have a great week, and I’m looking forward to our next meeting.

Best regards,
Qinyuan

****Week 4 (9/29)****

Hi Ian,

Happy Monday! The following is the update on the Deliverables for this meeting!

1. **Produce Schematics: **
- Done, 74HC959 selected, Shift register solution for CS SPI, more detail needed.

2. **Learn SPI & I2C Protocols: **
- Done, reviewed resources:
  - [STM32 I2C Example] (https://www.digikey.ie/en/maker/projects/getting-started-with-stm32-i2c-example/ba8c2bfef2024654b5dd10012425fa23)

3. **Gantt Chart: **

- Done, but I believe it needs improvement.

4. **System Tick/Timing Architecture in FreeRTOS:**
   - Chapter 12 Operating Systems, time services, and scheduling mechanisms.

5. **Mars Lander Archives:**
   - Done, VxWorks RTOS in Mars Pathfinder Mission had task priority set up wrong and was reprogrammed remotely.

6. **Miscellaneous:**
   - Using the MK2 as an oscilloscope.

Best regards,
Qinyuan

****Week 5 (10/7)****

Hi Ian,

Happy Monday! Here's a summary of today's meeting. The deliverables are listed at the end of the update.

30/09/2024 [FYP] Meeting Week 4 Sem1:

Agenda:
1. Gantt Chart: In progress
2. Schematics: In progress

Minutes:
1. **Schematics:**

- USB-USART connection
- 16 * 16 digital parallel IO (Shift register & Parallel to serial converter)
- Reference voltage and decoupling capacitor for the DAC chips and check the requirements for other chips.
- Ensure understanding of all analog configurations/requirements for all chips.
- Consider the design of pin connectors.
- Ensure plenty of power and ground (especially ground) connections.
- If approved, order components for breadboard testing.
- Do we need to design isolation and protection for all IO?

2. **Gantt Chart: **
- Refer to the FYP guide, translate weekly tasks from [ECE Final Year Project] to the Gantt Chart (Sem_1; Winter Breaks; Sem_2).

3. **FYP Report: **
- Develop a high-level structure for the report.
- Follow the guidelines carefully and ensure that the report format is tailored to the specific project.

4. **Miscellaneous: **
- Have a bit of think on how the task priority will be set.
- Look deeper into the clock configuration of both internal and external oscillators.
- Pay special attention to the oscillators used for communication and sys-tick.

Deliverables (Due next Monday):
1. **Schematics:** Complete the overall layout for review. If approved, order breadboard testing chips.
2. **Gantt Chart:** Complete the planning for Sem1 at the very least.
3. **FYP Report:** Complete the layout and introduction for review.

Let me know if there are any mistakes or if you have anything to add.

Best regards,
Qinyuan

****Week 6 (10/7 – 10/17)****

Hi Ian,

I hope this email finds you well. Below are the updates for this week:

Minutes:
1. **Review of Schematics:**
  - Current schematics provide an overview of system functionality but need further refinement.
  - Focus on two layout designs: Nucleo-based PCB and custom PCB featuring STM32 on board.
  - Key considerations include:
    - Modular Design: Modularize each block for easier testing.
    - Pin Connectors: Incorporate appropriate connectors for interfacing with Nucleo.
    - IC Component Review: Ensure all IC components meet manufacturer specs.
    - STM32 Integration: Begin designing the STM32-on-board PCB after Nucleo testing.
    - Shield Design: Consider designing the PCB as a shield for easier interfacing with Nucleo.

2. **Gantt Chart:**
  - Current format is acceptable, and I'll have it finalized by next Monday.

3. **FYP Interim Report:**
  - Abstract is nearly done. I can modify the chapter titles to better fit the project structure.

Deliverables/To-Do for Next Week:
1. Order components based on schematics.
2. Redraw schematics to meet professional standards for PCB design.

3. Review component specifications, ensuring compatibility with power and signal integrity requirements.
4. Assess clock configuration for communication and SysTick timing.
5. Review STM32-based RTOS projects focusing on ADC/DAC integration.
6. Complete the introduction section of the FYP report for review.
7. Finalize and submit the Gantt Chart.

Please let me know if there are any corrections or additional suggestions.

Best regards,
Qinyuan

****Week 7 (10/17)****

Hi Ian,

Pin Table on Nucleo:
- Create a pin configuration table for the Nucleo board.

Chapter Title:
- Combine Introduction and Motivation into one section.

Technical Details:
- The report should have two sections: Software and Hardware (already completed).

Deliverables:
1. Finalize schematics for each block.
2. Complete the Gantt chart based on the action plan.
3. Write with the reader in mind.
4. Complete miscellaneous tasks.

Next Meeting:

- Tuesday at 4 PM, online.

Best regards,

Qinyuan