University of Glasgow

School of Computing Science

CSC1104 – Computer Organisation and Architecture

Assessed Coursework

Course Name	**CSC1104 – Computer Organisation and Architecture**			
Coursework No.	**Assignment**			
Deadline	**Time**	**23:59**	**Date**	**10 Nov 2023**
% Contribution to final course mark			**20%**	
Solo or Group	**Solo**		**Group**	**√**
Anticipated Hours	**12 hours per group member**			
Submission Instructions	**Submit your assignment report to the Dropbox of LMS Xsite**			
Please Note: This Coursework cannot be Re-Done				

Code of Assessment Rules for Coursework Submission

Deadlines for the submission of coursework which is to be formally assessed will be published in course documentation, and work which is submitted later than the deadline will be subject to penalty as set out below.

The primary grade and secondary band awarded for coursework which is submitted after the published deadline will be calculated as follows:

(i) in respect of work submitted not more than five working days after the deadline
  a. the work will be assessed in the usual way;
  b. the primary grade and secondary band so determined will then be reduced by two secondary bands for each working day (or part of a working day) the work was submitted late.

(ii) work submitted more than five working days after the deadline will be awarded Grade H.

Penalties for late submission of coursework will not be imposed if good cause is established for the late submission. You should submit documents supporting good cause to Admin-In-Charge

Penalty for non-adherence to Submission Instructions is 2 bands

1) Objectives

There are three objectives of the assignment. The first objective is to practice the knowledge of computer organisation and architecture that we learn in lectures in hands-on programming exercises. The programming exercises include questions both C programming and assembly language programming on your Raspberry Pi platform with ARM processor. You are also expected to perform self-study and explore the solutions based on your group discussions.

The second objective is to train you the teamwork and divide the workload equally among the group members. Everyone should do an equal share of contributions to the group assignment. Every group member needs to take turns to perform a short presentation and records in 5 minutes video. Every member needs to present your contributions in the group project in the video (which part is done by you). Each group member will conduct the peer assessment to all members in the same group, which derive the individual grade for each member.

The third objective is to learn how to use C and Assembly language programming codes running in the ARM processor of the Raspberry Pi board.

2) Learning outcomes

- Perform study and group discussions to understand the function of modules in computer organization and architecture.
- Create C programming code and assemble language programming codes according to the corresponding computer organisation and architecture.
- Practice the creation of assembly language programming codes for ARM processor in the Raspberry Pi platform with supporting user inputs and output for results displaying.
- Perform self-study how to use C programming codes on Raspberry Pi to control Inputs/Outputs (general purpose input/output GPIO), and transfer the generated waveforms to the PC through WiFi connections, finally display the waveforms on the PC for the visualization purpose using third party's plotting tools supported by Visual Studio Code.
- Learn how to write a good report to present the works you have done in the group assignments, and how to highlight key features, solutions and contributions in a PowerPoint presentation and demo video.

3) Group assignment questions

Please answer all questions, using a report in the Microsoft Word format, describe your solutions including texts, drawings, diagrams, figures, tables, flowcharts, etc. Please also attach the separate C programming codes .c files, .h files, assembly language codes .s files and generated example waveform files in the submission together with the **WORD format report**. A PowerPoint PPT deck and a demo video (maximum 5 minutes) are also needed for the submission.

Raspberry Pi comprises of a 40-pin connector connecting to the ARM processor. Some connector pins are connected the GPIO pins of the ARM processor. Please conduct the self study through Google search for relevant materials about the Raspberry Pi and GPIO pins. There are many relevant examples and documentations about the Raspberry Pi GPIO pins, and C to control GPIO pins.

In the previous years, the school POD have purchased sets of devices for students. Each set consist of a small breadboard, 2 LED lights, some connecting jump wires, and resistors. Each group borrowed one set of devices from the POD for their project assignments in previous years, such that they connected a GPIO pin on Raspberry Pi to the red LED light, and connect a PWM GPIO pin on Raspberry Pi to the green LED using the jump wires, resistors and the small breadboard.

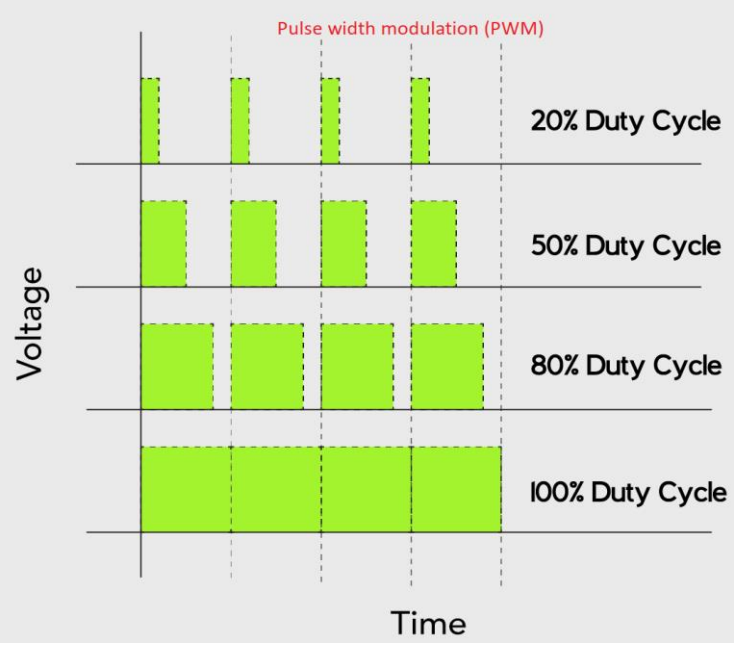


Figure 1. Illustration of duty cycles by pulse width modulation (PWM)

(PWM: pulse width modulation. Please check the web link here for more information: How to generate the PWM using the Raspberry Pi 4 (linuxhint.com); Can I use the GPIO for pulse width modulation (PWM)? - Raspberry Pi Stack Exchange). Please see Figure 1 to understand better on PWM.

This year, the project assignment will be for further developments based on the previous year's assignment outcomes. But there will be no devices needed (including a small breadboard, 2 LED lights, and resistors) this year. We will mainly do the software programming on the Raspberry Pi board, and send the generated waveforms data from the Raspberry Pi to your laptops, then display such waveforms using another C program in your laptops.

Please use the attached student.c code as your code base to further develop for this project assignment. Please also read the text descriptions below for better understanding the existing works done (which is the base for your development).

The scope of the existing works done consists of:

1. A C program (student.c) that controls the GPIO output pins of a Raspberry Pi (student device). The program will be able to change the frequency and duty cycle for the GPIO output pins.
2. The program should allow user to configure the frequency and duty cycle of the GPIO output pins when user run the C Program using SSH or Putty.
3. The program includes a handshake to another C program (monitor.c) on another Pi (monitor device). Once handshake is successful, it will send the frequency and duty cycle information via UART. (No needed for this new project assignment)
4. The monitor device will handshake with the student device and receive the information sent. (No needed for this new project assignment). The monitor device set up is only for the purpose of testing whether it can receive the information successfully via UART. (No needed for this new project assignment).

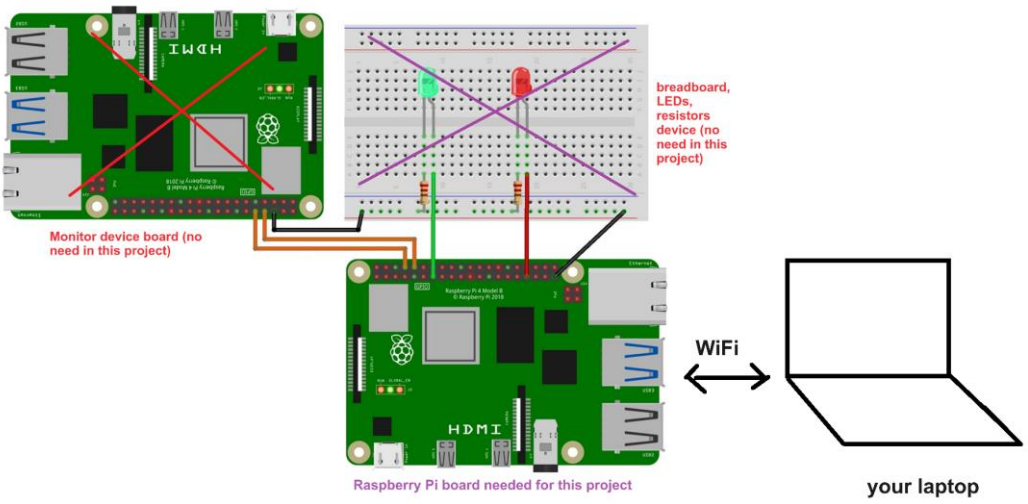


Figure 2. Illustration of hardware connections (only Raspberry Pi is needed to connect to your laptop)

Please take note that in this new project assignment, there is no need the monitor device board. It only needs the Raspberry Pi board to run the student.c program. As such, in your assignment, please comment out those code sections in the student.c program relevant to the monitor device board. Then you can modify the

C codes in the student.c code to achieve the requirements in this new project assignment.

student.c Program Explanation

This section will provide an explanation of how the C program configures the frequency and duty cycle of the output pins on a General Purpose Input Output (GPIO) interface. The rest of the code can be viewed in student.c code which is self-documenting codes with additional comments. Please read the student.c code on how to compile the C code, how to install the WiringPi Library, etc.

Frequency and Duty Cycle

Firstly, let us define what is frequency and duty cycle.

Frequency refers to the number of cycles per second that an electrical signal completes. In the context of GPIO pins, the frequency determines how many times the output voltage will switch between high and low states in one second.

Duty cycle is the percentage of time that the output signal remains in the high state during one cycle. For example, a signal with a duty cycle of 50% will spend an equal amount of time in the high and low states during one cycle.

Frequency is used to determine the blinking frequency and duty cycle is used to determine the LED brightness when it is turned on.

WiringPi Library

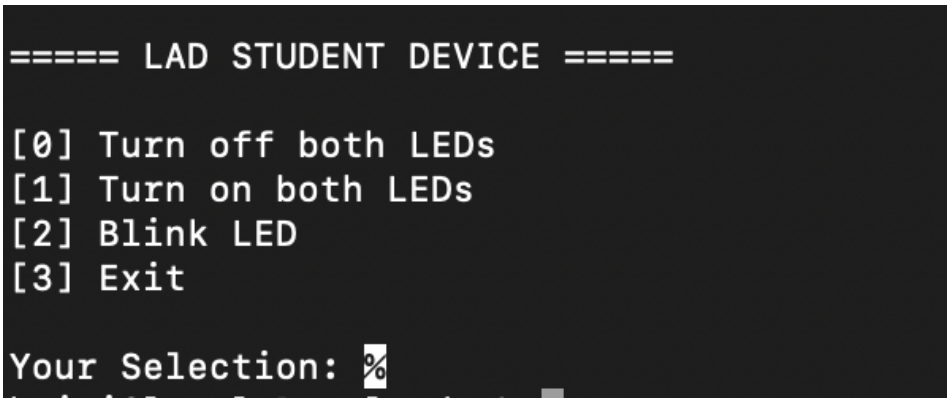
In our program, we configure duty cycle for the GPIO output pins by using the library WiringPi. Calling the softPwmCreate() function help us to configure the exact GPIO output pins to our desired PWM configuration. We set the maximum to be 100, which will act as 100% brightness.

Next, to achieve the blinking effect of GPIO output pins at a desired frequency, we will utilize the milis() function to obtain the current time in milliseconds. This will enable us to create non-blocking delays and set the voltage of the GPIO output pins to high within the desire timeframe, before setting it back to low. The process will repeat to achieve the blinking effect with the frequency that the user has configured.

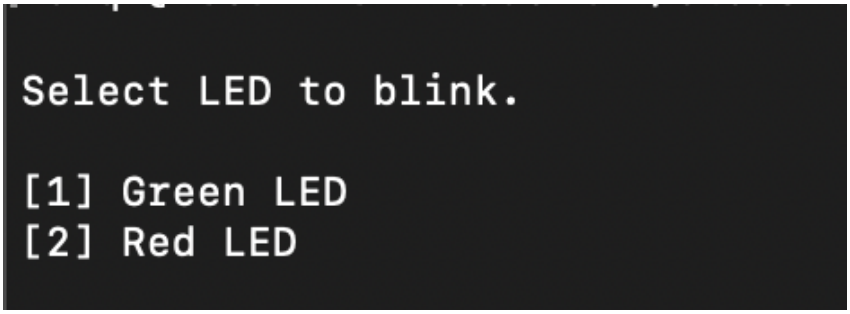
Using this method, we can blink at a desired frequency and brightness without any conflicts between frequencies of the blink and PWM.

Executing student.c code

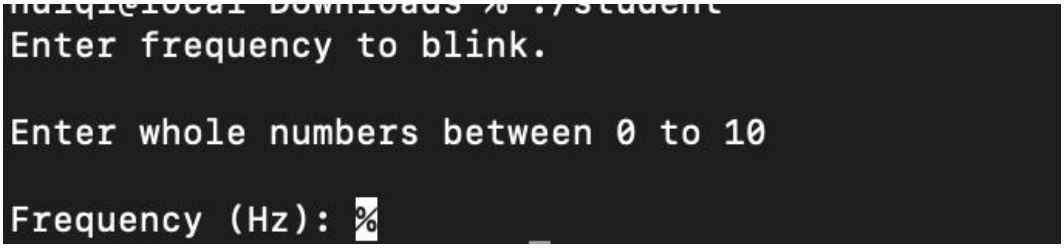
1. Once the code run for Student, a menu will pop up with selections. Select [3] "Blink LED" by typing 3 and hitting the "Enter" button.



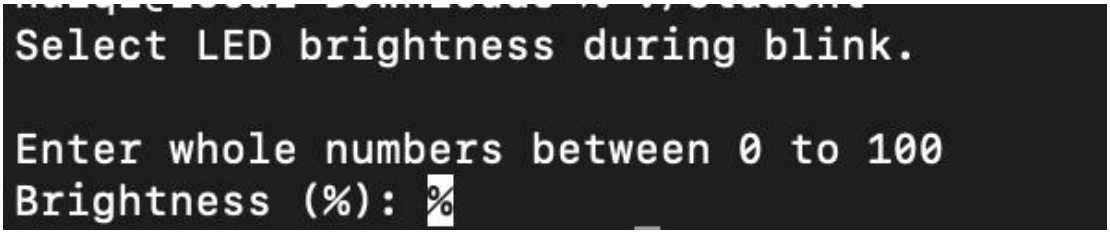
2. Select 2 to choose Red LED.



3. Enter 5 to blink Red LED at 5Hz.



4. Enter 50 for 50% brightness.



5. A confirmation message will appear. Enter 1 to confirm the configuration.

```
Confirm your blink configrations.

LED to blink: Red
Blink Frequency: 5Hz
Blink Brightness: 50%

[1] Confirm Configuration
[0] Return to Home

Your Selection: %
```

6. After your confirmation, the handshake will start. connectToMonitorDevice() function will be called to send UART messages from Student to Monitor. (No needed for this new project assignment)
   * Make sure the serial UART port name for the Pi is correct. The default serial port name for UART for Raspberry Pi 4V should be 'ttyAMA0'.
   * Make sure the baud rate for both Raspberry Pi is the same.

```
serial_port = serialOpen("/dev/ttyAMA0", 9600);
```

This function should send messages to the Pi Monitor of your configuration to the Pi Monitor. You will also see the acknowledgment messages on Student. (No needed for this new project assignment)
If the acknowledgement is received but LEDs do not blink, please use the "Turn on both LEDs" and "Turn off both LEDs" feature to troubleshoot your breadboard connection.

To try again with a different configuration, re-run monitor device code.

With the student.c code, it can change the LED lights' patterns. For example, keep blinking with both LED lights of 2 times per second. Both red LED and green LED utilize the square wave signal with frequency of 2 Hz (with 50% duty cycle) to blink both LED lights 2 times per second, shown in Figure 3.

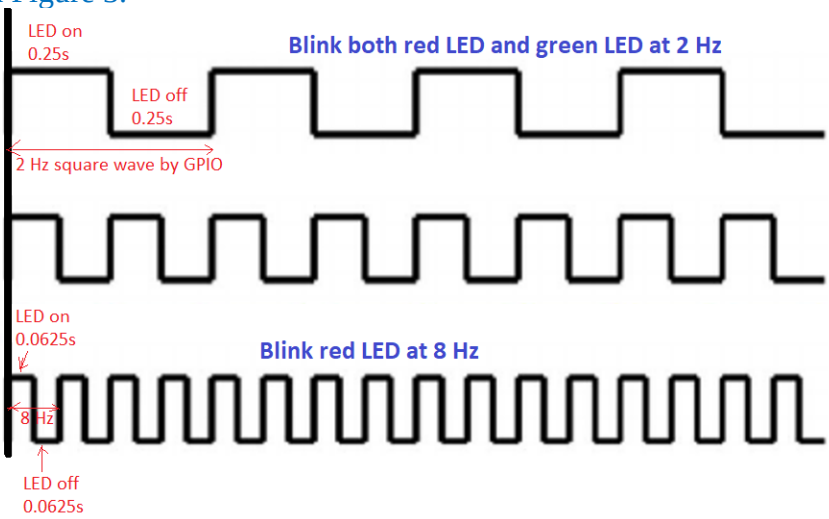


Figure 3. Illustration to blink the red LED light at 2 Hz and 8 Hz by the GPIO pins

For example, blink the red LED light with the pattern of 8 times per second (i.e., 8 Hz blinking frequency). It uses the square wave signal by the GPIO pin with frequency of 8 Hz to blink the red LED 8 times per second.

For example, for the green LED light, it still blinks at 2 times per second, but with the reduced brightness. It uses the PWM GPIO pin to reduce the brightness of the green LED by 4 times. It means, the duty cycle of the PWM GPIO pin will be reduced from 50% into 12.5% (i.e., 50% ÷ 4), as shown in Figure 4.

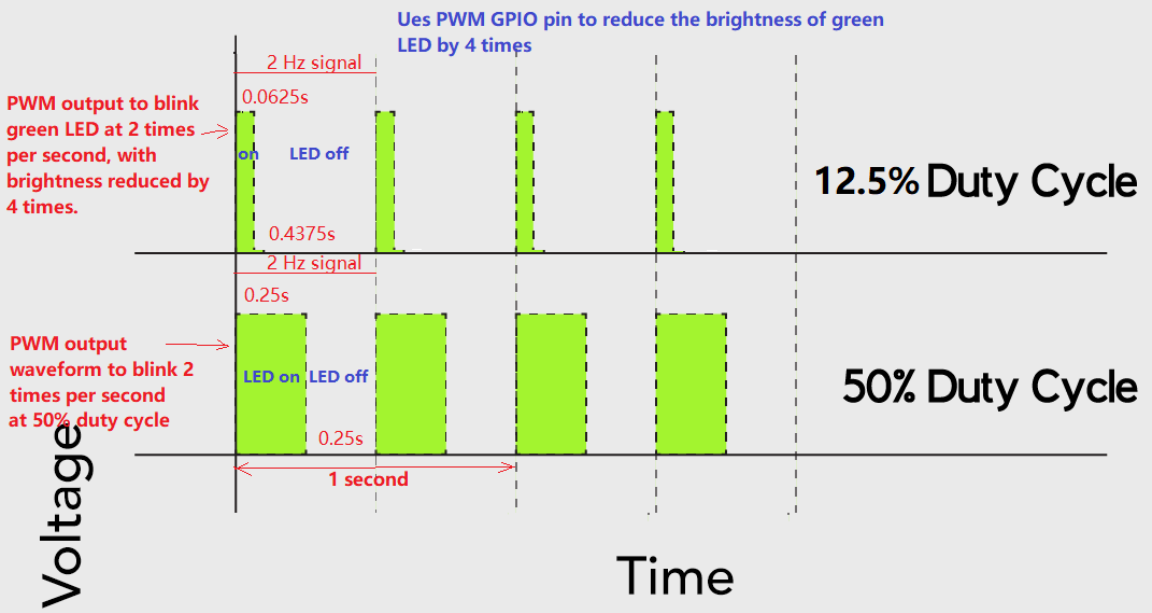


Figure 4. Illustration to reduce the brightness of the green LED light by 4 times by the PWM GPIO pin

For your information, all these above examples have been achieved by the student.c code already.

New Project Assignments to do this year:

Question 1: (70 Marks)

According to the descriptions of the student.c program, please modify it into the **NewStudent.c** code to achieve the further features.

1. When a user chooses to turn on, or turn off, or blink the two LED lights at the same frequency/duty cycle, or different frequency/duty cycle separately with your **NewStudent.c** code, please write the waveforms data into a file (could be a .CSV or .txt file format) in the Raspberry Pi folder using the **NewStudent.c** code.

2. The value of the data point is either '1' or '0', representing the HIGH or LOW state of the waveforms. The number of data points per second should be about 50 (about 20 ms time interval per data point). If the printf function is not fast enough to write 50 data points per second into the file, you can save the data points in the memory first. At the end of the waveform length, then write these data points into the file.

3. The configurations of the waveform need to also to written into the same file, including what is its frequency and what is its duty cycle. The reason is that you need to display the waveforms in your laptop. The information of the frequency and duty cycle is required.

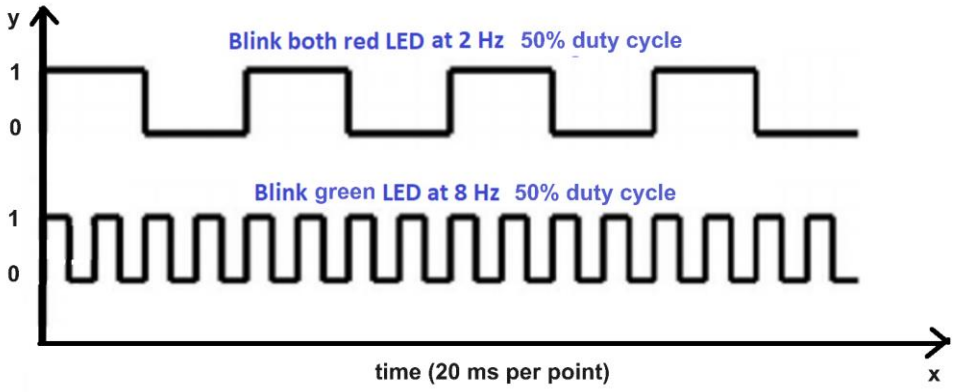
4. The same file consists of two waveforms data, one waveform is for the red LED light, the other waveform is for the green LED light. The reason for recording two waveforms is that we can display one on the top, the other on the bottom, to compare these two waveforms on your laptops.

5. You can decide how long of the waveform data to be written into this file. Ideally should not be shorted than 60 seconds.

6. Next, transfer this file from your Raspberry Pi to your laptop. There is no restriction on the method. You can decide which method to transfer the file (for example: scp, MQTT, or other methods https://forums.raspberrypi.com/viewtopic.php?t=281259).

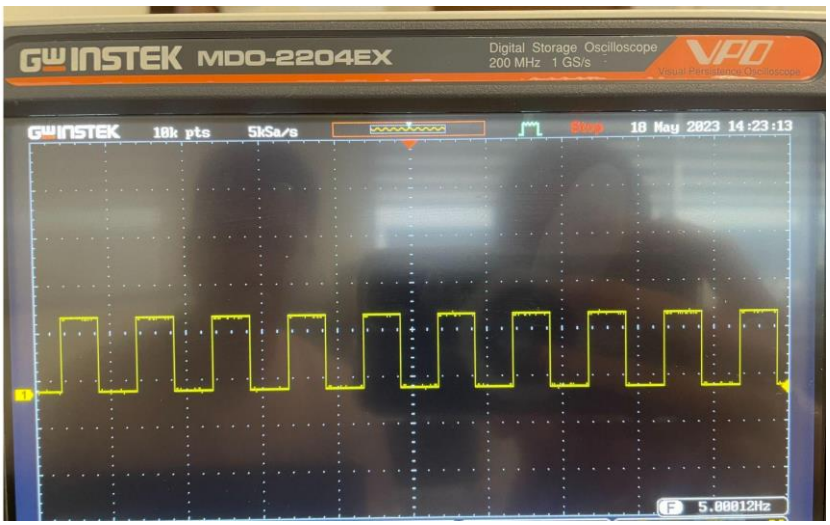
7. Once the waveform file has been transferred to your laptop, please develop another C code in your laptop, named as **DisplayPlot.c** program, to read this waveform file, and display the two waveforms using the drawing tool

supported by Visual Studio Code, e.g., koolplot, GNUplot, etc. One waveform is on the top, the other is at the bottom, similar to the example output below:

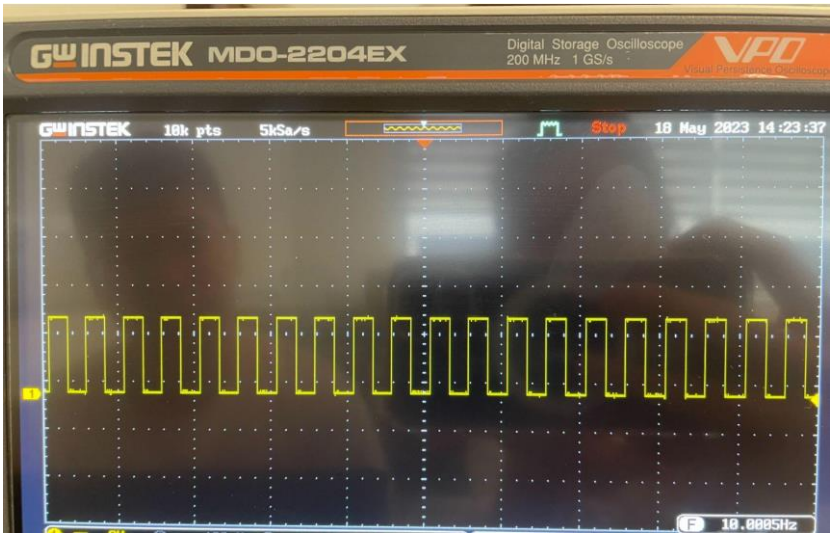


The x-axis of the plot will be the time (every time point is about 20 ms, or other value as the same as the time interval of your waveforms recorded by the Raspberry Pi), the y-axis will be either '1' or '0', representing the HIGH or LOW state of the waveforms. Your plots will also indicate the frequency and duty cycle of the waveforms (such information is obtained from the file generated by the Raspberry Pi).

8. Check if the two waveforms plotted by the **DisplayPlot.c** program on your laptop are the same as the waveforms generated according to the configurations keyed in by the user input at **NewStudent.c** code.
   For example, if the user chooses to generate the waveform of the red LED light will be 5 Hz with 50% duty cycle at the **NewStudent.c** code at the Raspberry Pi, the waveform is plotted by your **DisplayPlot.c** program on your laptop should be similar to the following waveform:



For example, if the user chooses to generate the waveform of the red LED light will be 10 Hz with 50% duty cycle at the **NewStudent.c** code at the Raspberry Pi, the waveform is plotted by your **DisplayPlot.c** program on your laptop should be similar to the following waveform:



Significance of the outcomes of Question (1)

The purpose of this project is to reduce the hardware setup to connect the GPIO pins to LED lights using the breadboard, jumping wires, and resistors. With the c programs, **NewStudent.c and DisplayPlot.c** developed by you, there will be no need to use the oscilloscope to measure the GPIO pins for the waveforms generated. We can achieve the waveforms generation and plots using software approach. As such, it is a quite useful method if your project can achieve these functions in Question (1).

Please google search and reference the example codes or documentations on internet. Then adapt the reference codes or example codes from internet into your C programming codes. But please do not copy the C programming codes from other groups in CSC1104 – Computer Org Arch. Otherwise, your group will receive zero mark for this assignment project (not worth to take the risks).

In your C programming codes, please add clear and detailed comments for every line of the C codes. Please provide the high level descriptions, including block diagrams of your code module, high level structures and flowcharts to descript your code work flows in the documentations in the report how you design your C programming codes (**NewStudent.c and DisplayPlot.c**).

Question 2: (30 Marks)

Compare and illustrate the execution speed differences between the C program and assembly language programming codes to achieve the same task. To achieve some tasks, the assembly language programming code has the advantages to be executed faster than that of the C program code (i.e., with the shorter execute time).

The Question (2) is an open question. Please Google search for relevant information or example solutions or codes (both C and assembly language codes), on how to measure, compare, and illustrate the execution time differences between the C and assembly language codes to perform the same tasks. Through such literature review, **please cite the references where you found the information, solutions, or example codes properly.** Such that other readers can get more detailed information from such relevant references. Please write your findings into the report of the project assignment. Please also describe clearly which task you choose to compare the execution time differences, and what are the configurations or setup steps to conduct your experiment. Such that readers can duplicate your methods and obtain the same results for the verification purposes.

With such references or example codes, develop two of your own codes: one is the assembly language programming code (named **Q2.s**), the other is the C code (named **Q2.c**) on the Raspberry Pi platform to achieve the same tasks. With two codes (**Q2.s** and **Q2.c**) are executed on the same Raspberry Pi platform, for the same tasks, please illustrate how you measure the execution time, how to illustrate the execution time difference. Please attach the evidence and codes execution results to show the execute time difference.

In your assembly language programming code **Q2.s** and C code **Q2.c**, please add clear and detailed comments for every line of the assembly language codes. Please provide the high level descriptions, including block diagrams of your code module, high level structures and flowcharts to descript your code work flows in the documentations in the report how you design your assembly programming code.

4) Assessment criteria

Assignments will be assessed according to the criteria for the Group Assessment and the Individual Assessment. Note that the criteria are subject to change depending on the progress throughout the trimester. Following timely submission, the exercise will be given a numerical mark between 0 (no submission) and 100 (perfect in every way). The numerical marks will then be converted to a grade, that is the **Group assessment Grade** for your group.

Each student will be asked to give the peer evaluation scores to each of your group member (**0% as no contributions - 100% as full contributions**), according to his/her contributions in the group project. Your peer evaluation scores given to other members will be kept strictly confidential.

Your individual grade for this assignment will be computed as follows:

Each group member individual grade = Group assessment * (weighted peer review score by each group member and his/her presentation performance in presentation video)

*Table 1. Group assessment**

*Group assessment will be weighted by peer review

Criteria	Weight
Codes implementation: • Code quality • Code structure • Functionalities • Neat variable and function names	20
Quality of detailed comments and documentation in the source codes, to help readers better understand your source codes.	15
• Solutions that produce the correct results, according to the requirements given in questions.	35
Report and PowerPoint presentation video: • Professional writing of the report with a clear logic and structure of your solutions.	30

• A clear and smooth demonstration and presentation of your solutions in the 5 minutes presentation video. • Every group member must give the speech to describe on his/her respective portion and his/her contributions in the assignment. • Highlight the strength and limitation of your solutions. • List down every group member's individual workloads and contributions in the report and presentation. • Reflection from each group member regarding their learning journey throughout the assignment.	

Table 2. Individual contribution assessment

Each member needs to write at least 250 words self-reflections in the learning of CSC1104 – Computer Org Arch and learning in the assignment project in the submitted report. Each individual team member is required to implement a portion of the programming codes, documentation and report. Every group member must give the speech on his/her respective portion in the assignment in the presentation demo video. Therefore, a clear task distribution is required. Peer evaluation score will be used as the individual contribution assessment factor.

5) Timeline and deliverables

Final submission: **Friday, 10 Nov 2023, 23:59 pm**.

Each group submits a Px-Groupxx-CSC1104.zip file to your group's LMS xSITE Dropbox ('CSC1104 Group Assignment'). Px means labs P1, P2, or P3. Groupxx means the group number in the corresponding Px. One submission per group.

The Px-Groupxx-CSC1004.zip file should contain files as follows:

- a **WORD format** of the final report named Px-Groupxx-report-CSC1104.doc,

- all the individual source codes (.c, .h, .s),
- the generated example waveform file,
- a group power point presentation video Px-Groupxx-CSC1104.mp4, maximum 5 minutes, where all group members must give their speech on their respective portion of works and contributions in this group assignment.