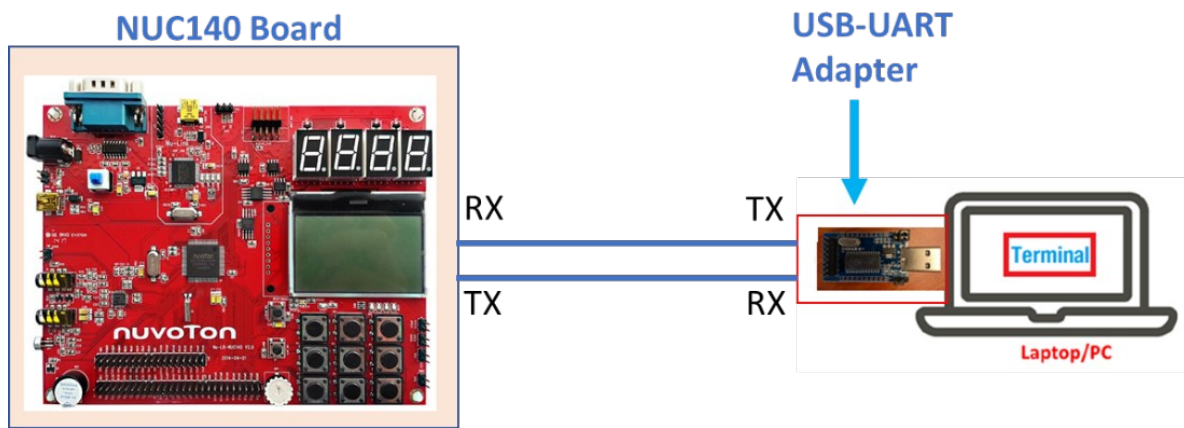


EEET2481 – Assessment 3 (40%)

Exercise 1 – UART (10 marks)

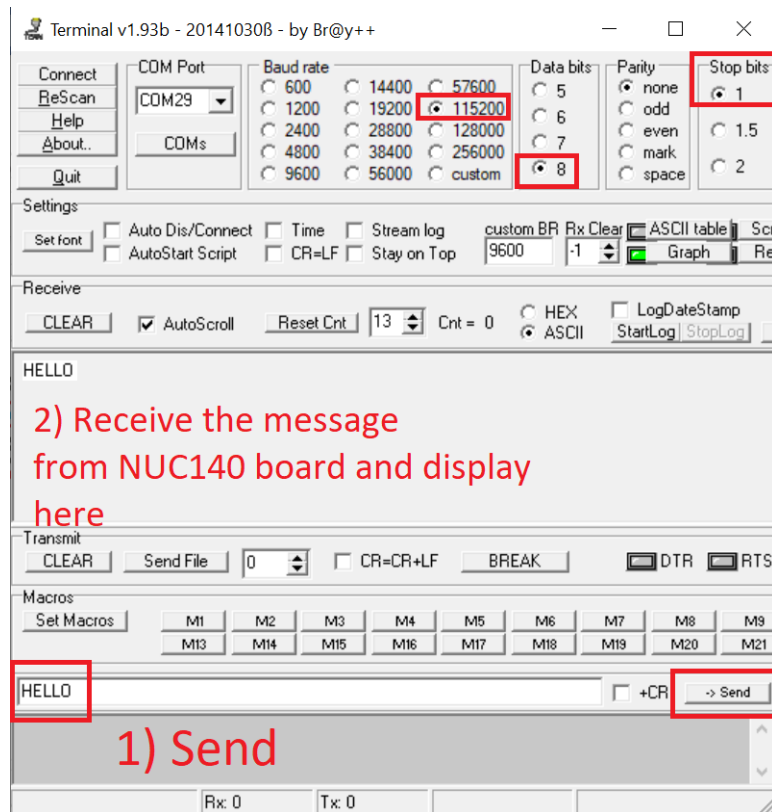
In this exercise you will be develop a program that will run on the NUC140 board, to send and receive data between the NUC140 Board and your laptop. From laptop side, you need an additional USB-UART adapter together with Terminal software to facilitate the communication. The diagram of the system can be simplified as follows:



The program has the following specifications:

- a. Clock frequency of the CPU is at 50 MHz
- b. UART channel 0 specification:
 - o UART clock source is 22.1184 MHz
 - o UART0 channel is used to transmit data and receive data.
 - o Data packet: 1 start bit + 8 data bit + no parity bit + 1 stop bit
 - o Baud rate: 115200 bps

After the reset, the board is standing by to wait for data to send from laptop. You will send the message **"HELLO, THIS IS <student 1 name>, <student 2 name>, <student 3 name>, FROM TEAM <team name>"** from Terminal to the NUC140 board. The board will receive the message and send it back to the laptop. If the setting is successful implemented, you can see the message re-appear in the RX box of Terminal.

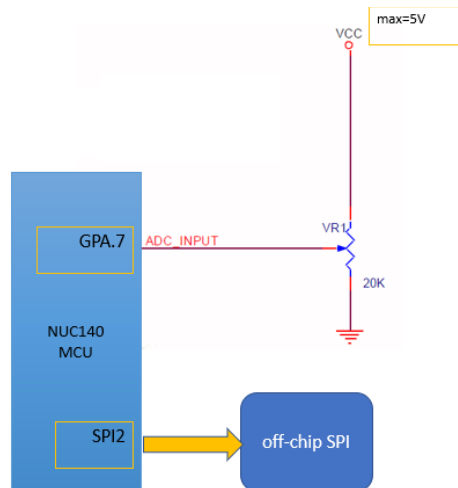


For this exercise 1, you must use **UART Interrupts** for both TX and RX.

Exercise 2 – SPI and ADC (10 marks)

2.1 Requirement

Write an embedded software program for the following hardware diagram. We will use the ADC and SPI of NUC140 MCU.



The program has the following specifications:

- Clock for the CPU is at 50 MHz
- ADC channel 7 (12-bit A/D converter with reference voltage, $V_{ref} = 3.3\text{ V}$) is used to continuously sample analog voltage at GPIO port A pin 7 and convert it into a corresponding digital value. The configuration for ADC:
 - o ADC clock frequency is 1 MHz
 - o Operating mode: continuous scan

As the program runs, as users change the potentiometer, the input voltage at GPIO port A pin 7 changes along. If the voltage is **greater than 2 V**, the program will be sending data out on the SPI interface to an off-chip slave device. In specific we will send out your **team name** continuously (if your team is Team 02, then you should display Team 02).

- SPI2 of the NUC140 MCU will be configured as a master device to transmit data to an off-chip slave device (in this Assignment, you can use the LCD to display the data). The configuration for the SPI2:
 - o SPI serial clock is 1 MHz
 - o SPI serial clock is IDLE at HIGH state
 - o Data bit is transmitted on the POSITIVE edge of serial clock
 - o Data is transferred from LSB first
 - o One BYTE of data to be transmitted/received in a transaction
 - o SPI2 pins: GPIOD.0 (SPI2_SS), GPIOD.1 (SPI2_CLK), GPIOD.3 (SPI2_MOSI)

2.2 Submission

1. Complete Keil MDK project which I can rebuild and re-run from my end.
2. A screenshot of your waveform showing the data coming out from SPI2

- Students will demo live the system to the Lecturer and prepare to answer questions relating to how they design the program.

Exercise 3 – Battleship Game (20 marks)

In this exercise you will design a simple battleship Game using NUC140 Board. If you are unfamiliar with the game, you can check out some videos on YouTube on how to play it. We will only implement the single-player mode. The game field is loaded in advance, and the player will play until the game ends. There will be indications if there are misses, hits and game overs. Technically, you are required to work with UART communication, GPIO, Timers and Interrupts.

3.1 Game Field

The game field is to be 8x8 locations. For example, the following game field has **five** ships (each ship consists of 2 dots). For example:

```

- - - - - - - -
-  X  X  - - - - -
-  -  -  - - -  X  -
-  -  -  - - -  X  -
-  -  X  X  - - - - -
-  -  -  - - - - -
X  X  -  -  X  - - - -
-  -  -  -  X  - - - -

```

To create the field, we will use Notepad to create a text file with the number 0 (represent water) and the number 1 (represent part of a ship). The text file version of the above looks as:

```

0 0 0 0 0 0 0 0
0 1 1 0 0 0 0 0
0 0 0 0 0 0 1 0
0 0 0 0 0 0 1 0
0 0 1 1 0 0 0 0
0 0 0 0 0 0 0 0
1 1 0 0 1 0 0 0
0 0 0 0 1 0 0 0

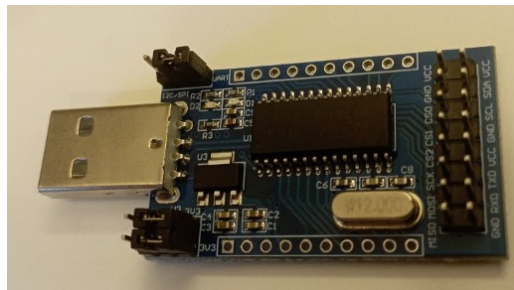
```

Two sample fields will be provided for your information. You can also create other files using Notepad if required.

3.2 System Description


- After reset, the LCD shows **a welcome screen**. You can flexibly to choose the texts and images to show on this screen.
- Then, if you send a map file from your laptop/PC to the board. You will need to find our Technical Officer and borrow a CH341A module (UART module looks like below photo) then connect TX and GND of the module to RX and GND of Nuvoton board. You need to go to UART lecture/tutorial on Canvas to download Terminal software to send the map.

Then you need to write an embedded program to capture and store this map temporarily. You only need to do this one time before the game starts. Once the sent complete, the screen should display **"Map Loaded Successfully"** right in the middle of the screen.



- If you press **SW_INT (GP15)** the game starts. We will use External Interrupt in this system.



The LCD now will show the full map where  representing water

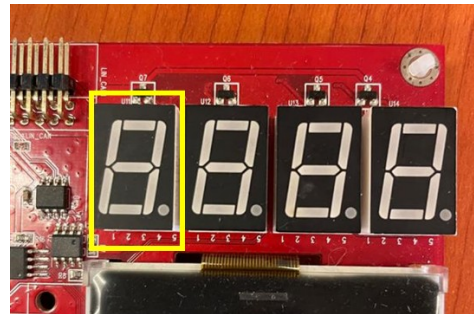
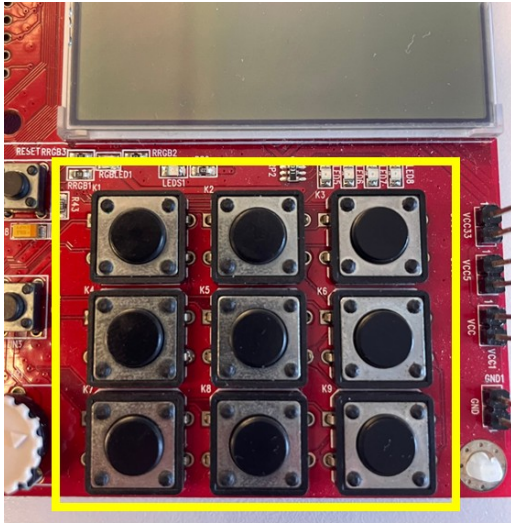
```

- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -

```

- - - - -

- d. Select **coordinate X** (ranging from 1 to 8) by using press the corresponding Key 1 to Key 8 from the Key matrix. This current coordinate X also displays on the 7segment U11.



- e. Press Key 9 will allow us to change **coordinate Y**
- f. Repeat step d to select **coordinate Y**. This current coordinate Y also displays on the 7segment U11. **If you press the Key 9 button again, it will change back to coordinate X selection.**
- g. Once XY is defined, we can press the **SW_INT (GP15)** to shoot



Your program then compares the entered coordinates vs the map earlier and then indicates if there is a Hit or Miss. If there is a **Hit**, we will flash **the LED5 (GPC12)** three times. You can choose the suitable flashing frequency here.

If there is a hit, the LCD updates to indicate the hit. For example, X display the hit:

- - - - -

- - X - - - -

- - - - -

- - - - -

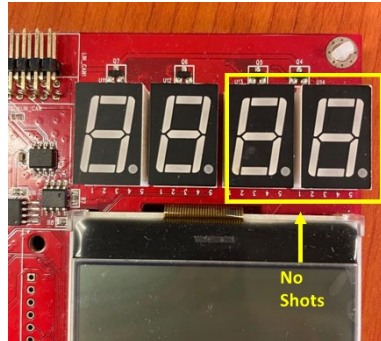
- - - - -

```

- - - - -
- - - - -
- - - - -

```

- h. The system also displays in real-time the number of shots fired via U13 and U14:



To control this compound 7segment-LED, we will need to apply the scanning technique to select and display one digit at a time, then change it to the next digit. The scanning rate must be quick enough, so all the numbers appear visible to our eyes. You must use a **Timer interrupt** to control the scanning rate.

- i. If the player successfully sinks **all five ships** on a map (Note that a ship is sunk if two hits are at two adjacent dots) or has fired more than **16 shots** but could not sink all ships, the game will be over. Couple indications for this:
- The LCD will display Game over message.
 - Buzzer will go off 5 times (only once).



- All the keys will stop responding.
- j. If you press the **SW_INT (GP15)** now, the game will be replay from the welcome screen

3.3 Other Requirements

The system must include the following game features:

- Check if a ship is shrunk when two adjacent dots are hit.
- Indicate if the same position is hit twice. For this, the number of shots will go up by one still.
- Count maximum shots allowed for a player; if he runs out of **16 shots**, then the game is over.

In terms of technical requirements, your system must use:

- Program is developed using **registered based method**. All configurations must be clearly explained (via code comments). Only standard drivers for LCD display (i.e., functions in LCD.c and Draw2D.c libraries) can be used in project code.
- Design with FSM
- UART Interrupt to receive text file sent from PC. You can decide the baud rate and UART format.
- Timer Interrupt for scanning rate. You can use this to define the delay time between the display of each digit.
- External Interrupt for **SW_INT (GP15)** button
- Use C functions to optimize the code.
- Put meaningful and clean comments to explain your design
- The entire system is built properly on a breadboard and ready to be checked.

3.4 Submission

1. Complete Keil MDK project which I can rebuild and re-run from my end.
2. Other supporting documents: State diagram , and other documents that help to explain your design understanding.

Demonstration

A project demonstration to present how the system works in practice and Q&A with the lecturer on project-related matters.

- Setup oscilloscope for measurement
- Introduce quickly about your setup
- Explain your algorithm and code
- Show what your code can do vs. requirement
- Majority of the demo time should be spent on Q&A: each student can be asked about any aspect of the project