ECEN-361 Lab-05:SPI & Logic Analyzer

NAME: _____

Introduction and Objectives of the Lab

This lab will require very little code development. The project as cloned from the GitHub-Classroom, runs without modification. You'll be asked to use a logic analyzer to capture traces of the formats and compare their utility.

- Part 1: Physical observation of different digital serial communication protocols: I2C, SPI, and UART
- Part 2: Become familiar with a logic analyzer, capture & decoding capabilities.

For each of the parts, follow the instructions, then fill in answers to the questions. Expected answers are indicated in the boxes with red text/spaces to fill in answers.

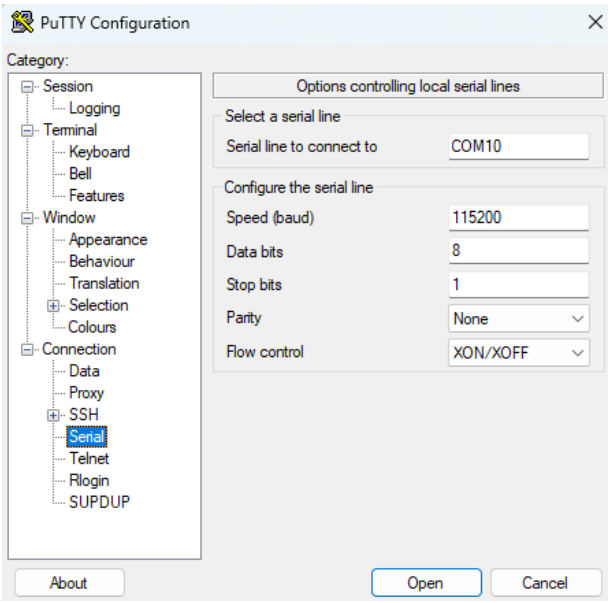
Lab Instructions

Part 1: Accept the Assignment, Download the repo, Run it

A. Accept the Assignment, Download the repo, Run it

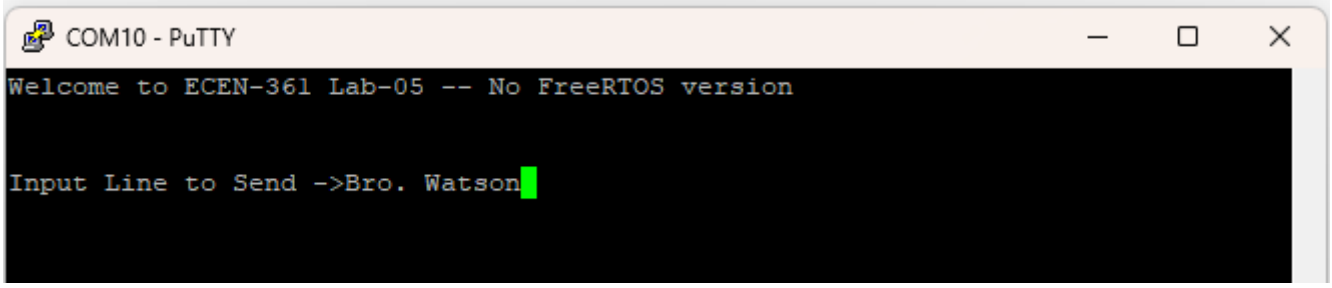
Same procedure as previous labs – get the laboratory into your STM32CubeIDE workspace, then clean/build/run it.

In order to interact with the program, you'll need to bring up a serial terminal emulator, like PuTTY (windows) or screen (MAC). Terminal emulator specs are (always the same for this class):



Your COM: port will be found with DeviceManager or on a MAC as (/dev/tty…)

With a terminal emulator running you'll see the opening banner and a prompt to enter a line of text:



B. Install the Saelae Logic Analyzer Software: Logic 2

C. Connect and label the probes for the SPI, UART and I2C:

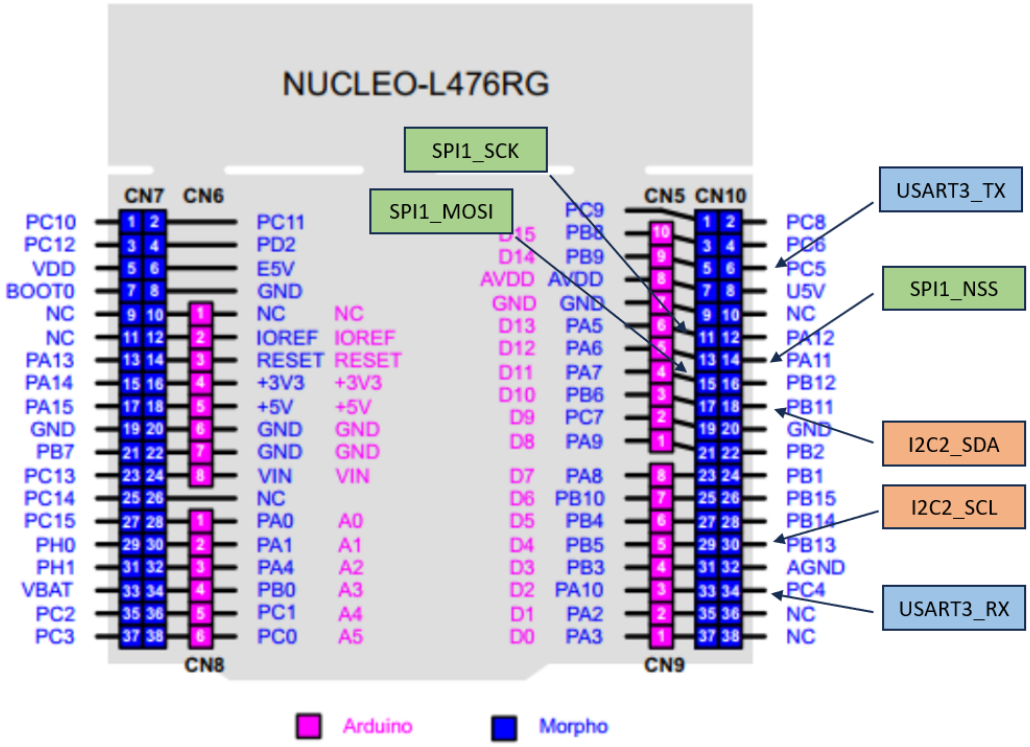
To probe the SPI and I2C pins, you'll use the following:

SPI-1 Transmit-only Master		
Name	*Chip Pin*	*Morpho*
SPI1_SCK	PA5	CN 10.11
SPI1_MOSI	PA7	CN 10.15
SPI1_NSS	PA11	CN 10.14

I2C-2 Master			
Name	*ChipPin*	*Morpho*	*Comment*
I2C2_SDA	PB11	CN 10.18	Note, this takes away LED_D1 on the board
I2C2_SCL	PB13	CN 10.30	Note, this takes away LED_D1 on the board
			Done manually with HAL_GPIO_WritePin

UART-3 Out		
Name	*Chip Pin*	*Morpho*
USART3_TX	PC4	CN 10.34
USART3_RX	PC5	CN 10.6

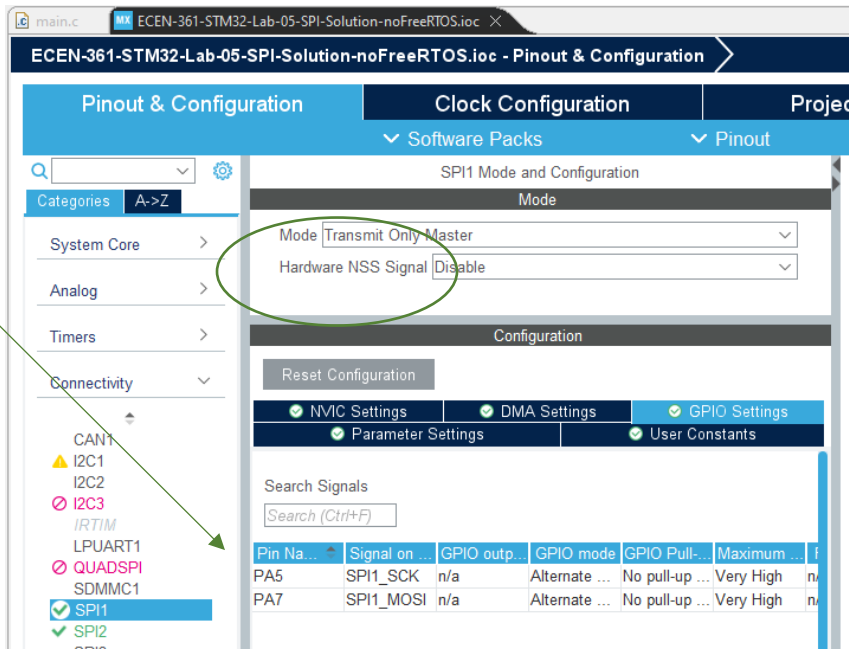
Note that we're connecting only to the MORPHO connectors (CN7 and CN10)



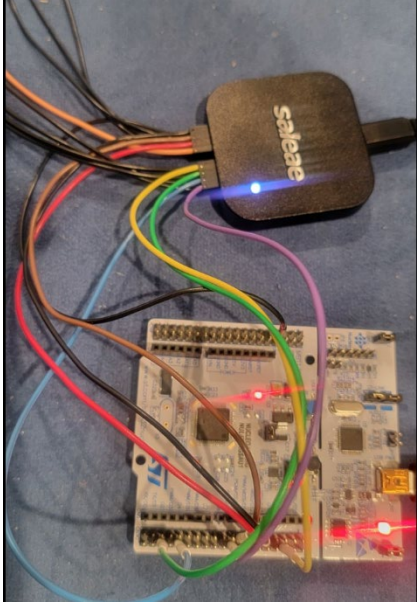
Note that these pins are found from the STMCubeIDE configuration file. Opening will show (SPI1 for example):

Connectivity / GPIO Settings:

You'll manually change, configure I/O pins later and in other labs.



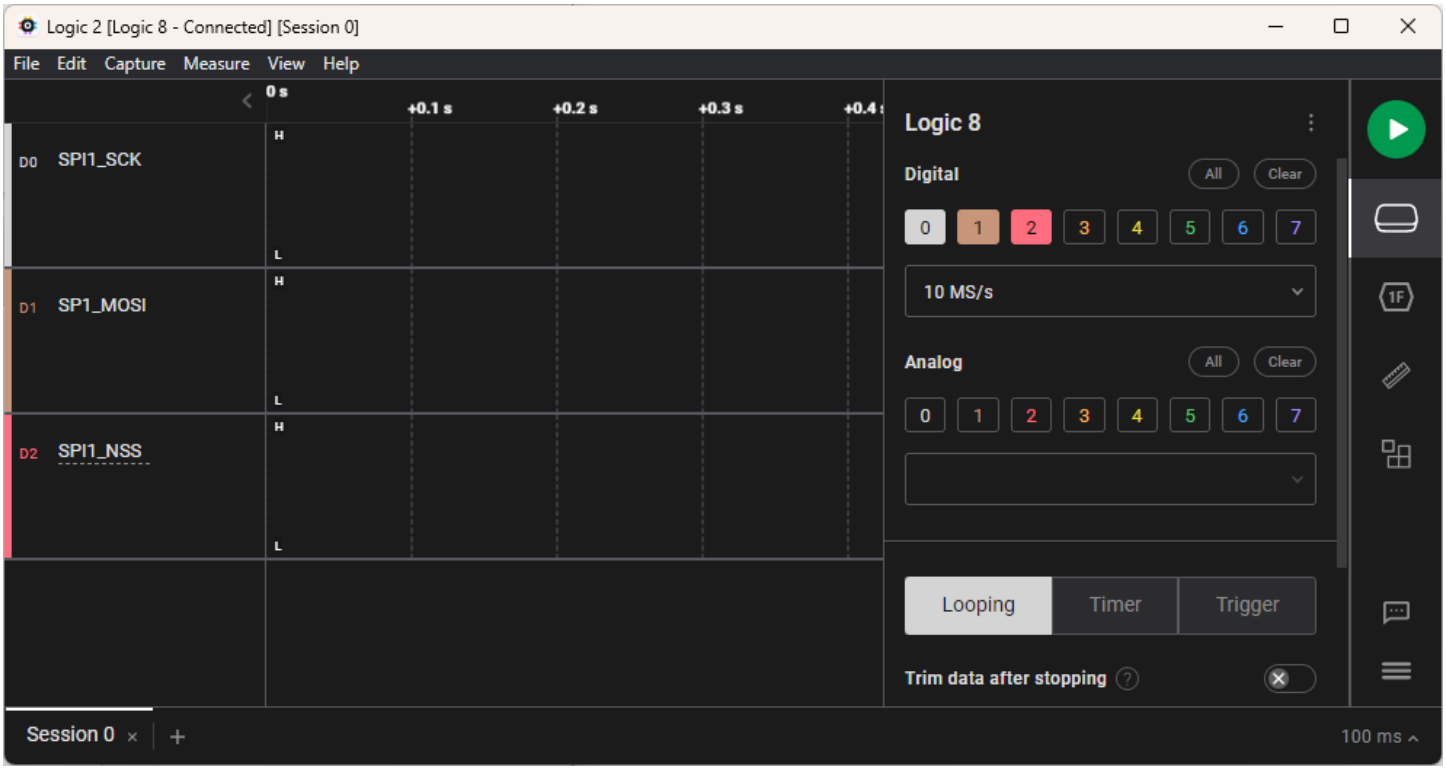
Plug in the logic analyzer and run the “Logic 2” program.



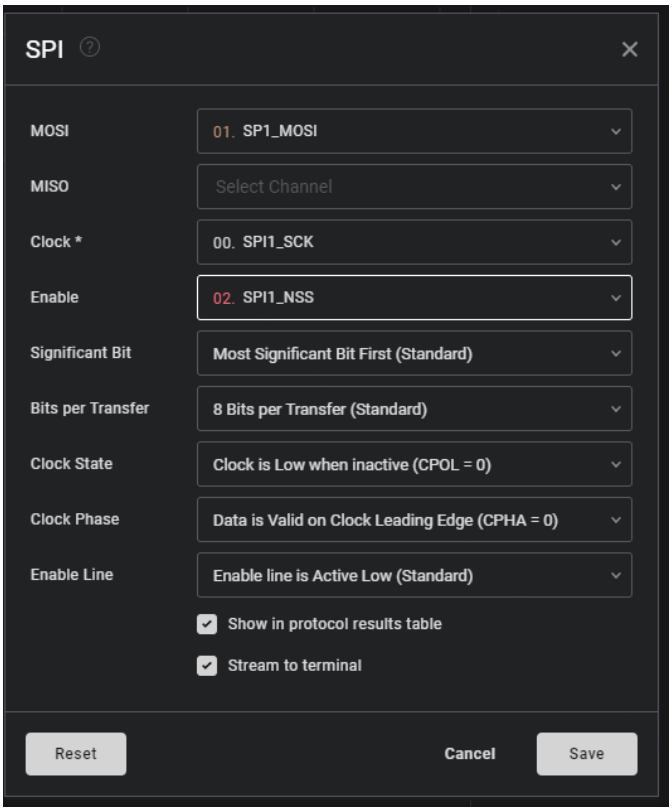
Connect probes from the Saelae Logic, using:

- GND (Black pins are on bottom)
- SPI1_SCK
- SPI1_MOSI
- SP1_NSS

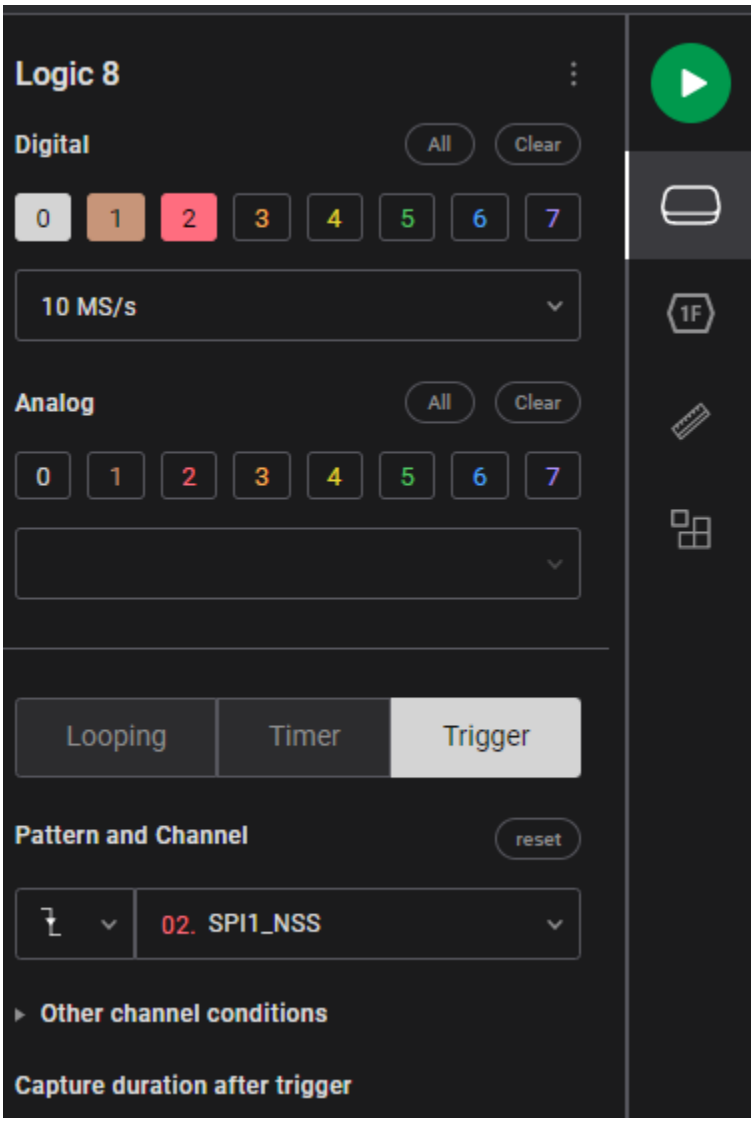
Label them in the software. Make sure the inputs are Digital



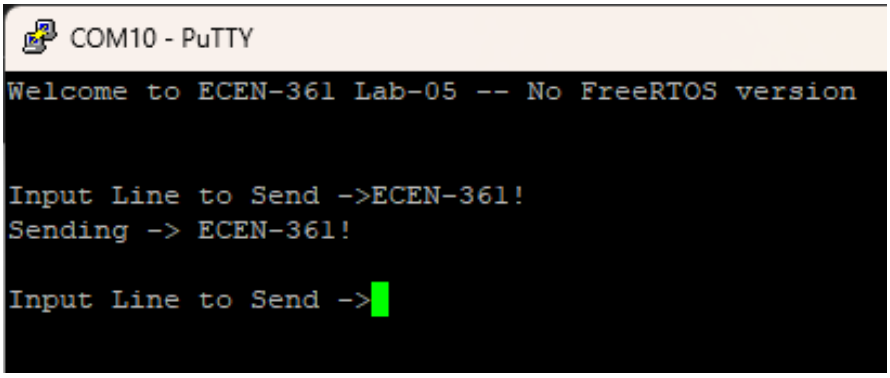
Program the Analyzer to decode SPI:

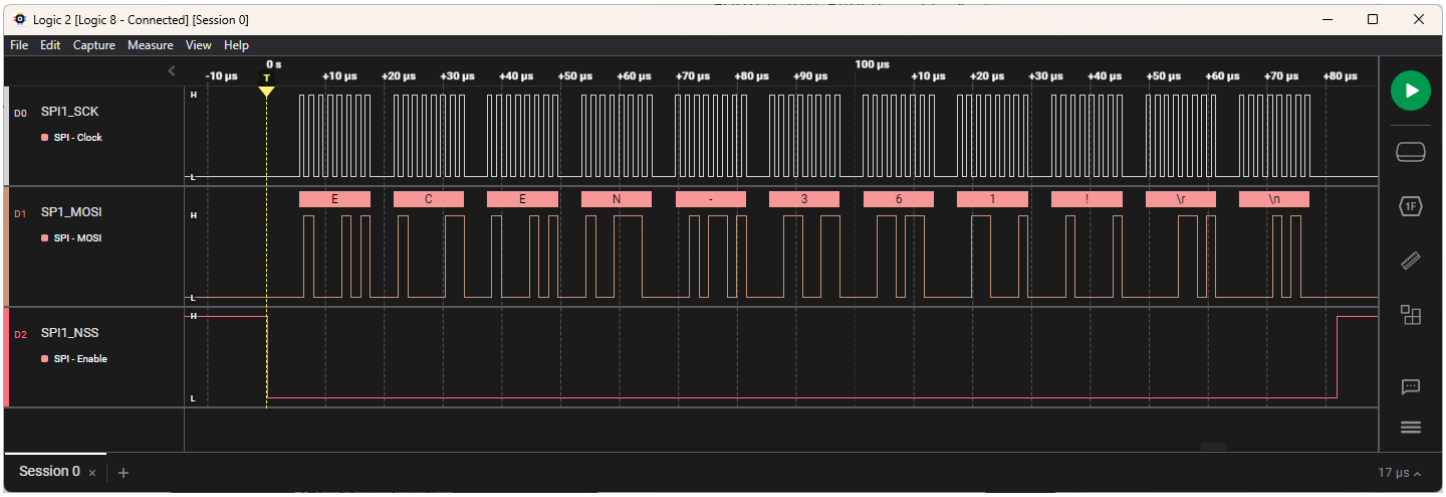


Set the mode to trigger on the falling edge of SPI1_NSS:



- Run a capture on the Logic2 (big green button)
- Enter some TXT into the TTY emulator
- Look at the results. Change the output formatting to be ASCII





Experiment with this, send different codes, learn to use the tool, then answer the following questions:

SPI-PART-1 3 Pts.

1.) What is the default bitrate? (time per bit) -- Use the measurement tool (looks like a ruler)

_____

2.) How much time between each byte? _____

3.) Which has more overhead: USART or SPI? And why?

_____

The transfer rate is a function of a clock divider on the main clock (80Mhz). Default in your code is 128. Find the line that defines this: `hspi1.Init.BaudRatePrescaler = SPI_BAUDRATEPRESCALER_128;` rate in **`main.c`**. Change this line to make the SPI transfer as fast as possible, compile, run, and capture the results and note what you see.

SPI_Par6t

1.) What is fastest bit rate possible with this processor? Equation?

_____

2.) Is there any problem capturing the fastest data of a USART channel?

_____

Part 2: Doing the same with the UART2 –

Attach the I2C Signals to look at the data coming out, then answer:

3 Pts.

1.) What is the default bitrate? (time per bit) -- Use the measurement tool (looks like a ruler)

_____

2.) How much time between each byte? _____

3.) What is the value of the data coming out first? It's not like the others.

_____

Part 3: Doing the same with the UART2 –

Attach the USART3_TX and RX and sample again, looking at the data coming out, then answer:

2 Pts.

4.) What is the default bitrate? (time per bit) -- Use the measurement tool (looks like a ruler)

_____

5.) What is the max bitrate easily supported? _____

Extra Credit Fun Ideas

5 Pts. Max, for completing any of these (or thinking of your own)

1.) Add on the MultiFunction Board, and have it display the number of characters sent each time.

2.) Enable SPI2, as shown in the table above (thru the .IOC configuration file), and read code that gets sent from SPI1->SPI2

3.) Currently SPI is running in "blocking" (polled) mode. Change it to run via an interrupt so the processor could be more efficient.

4.) Enable a 'smart' trigger on the Logic Analyzer so it doesn't actually begin capturing data until specific data is decoded/found.

Reference

To Finish

When completed with everything, commit/push the repo, and submit the URL of the checked-in repo to the iLearn Lab submission assignment. For any extra-credit, document it as appropriate, a small paragraph, a video clip, etc.

The following is a capture of all three protocols.
Notes that you'll need:

- The LogicAnalyzer can apply different protocols to different signals. In this case all three are shown
- I2C in master mode first transmits the address (here it's 0x11), then waits for the ACK to send the data. Without a SLAVE no data is actually sent.
- The UART is much slower and has overhead of start/stop bits.