Kai(Hsuankai) Chang
Name of the Course: 168584 Embedded Systems Hardware Design
Course Number: ECE-40292
Student ID: U09475562

Date: 02/28/2023

FRAM Challenge (Extra Work)

- **Project Description:**
  - I use SPI1 on the B-L475E-IOT01A Discovery kit, the connection has been specified on my HW7
  - In the project, I use STM32Cube IDE to develop and enable the USART1 and SPI1 peripheral. The reason I choose USART1 is because it is linked to the ST-LINK, so I don't need extra USART-USB connector to send/receive command between pc and uc.
  - For the SPI Tx/Rx, I use polling mode to simplify the overall process since the code size and task is fairly limited, no need to worry about other interrupt priority issue or task overrun issue.
  - Then I use RealTerm software to send and receive command, I choose to use this one rather than famous Tera Term is because for me, it is easier to send hex and ascii data. In the video you can also see I have written some CR or LF, which initially it is used for testing on other terminal software.
  - The result is successful, I am managed to write to entire memory array, and also read it back correctly.
- Note:
  - When I am debating on whether to use GPIO or SSI bit to control the NSS signal, I found out that SSI bit configuration is not in the SPI HAL driver layer, and based on the RM, SSI bit should be able to replace the NSS pin value if SSM bit is 1 (software slave management enabled). However, when I set SSI bit to 1, NSS signal isn't go low. I tried the same setup on my STM32F4-DISC board, and SSI bit can control the NSS pin to low, I am wondering what is the cause of it on our IoT board, or maybe I do something wrong on the setup?
  - I tried to use logic analyzer to analyze my SPI bus performance, I found out that the time between setting the GPIO NSS pin to low and SCK sends out time is pretty big, I think it is because the HAL layer does a lot of checking and settings inside that cause the time delay.