Instituto Tecnológico de Estudios Superiores de Monterrey

Laboratorio Sistemas Embebidos

Práctica 7 - SPI and SD Interfacing with Arduino Zero

Prof. Matias Vázquez Piñón

Gpo 2

Héctor Javier Pequeño Chairez - A01246364

Stephanie Denisse Benítez Cabrera - A00820320

Fecha: 1/12/2021

Objectives

- Strengthen your knowledge and abilities about C programming and serial communication protocols, particularly, SPI.
- Learn how to write programs using the SPI protocol to interface an SD card.

Introduction

Explain what you did in this laboratory

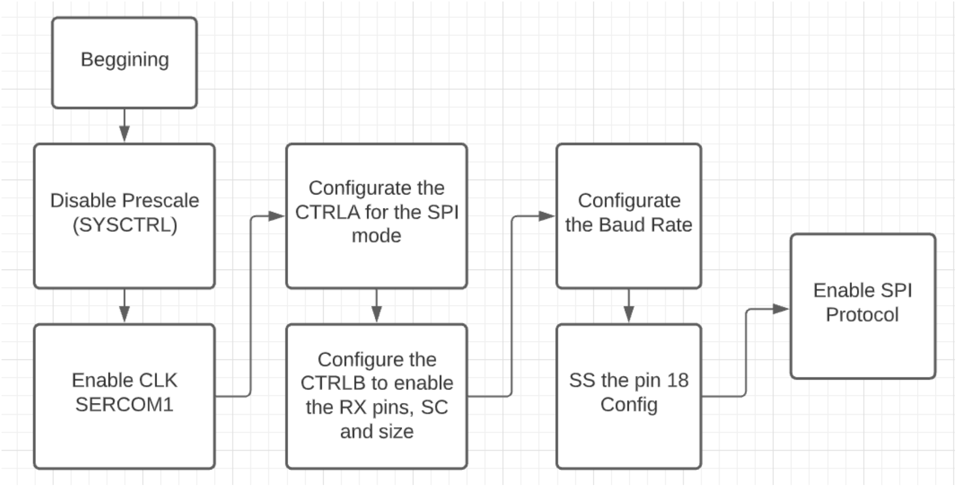
We use the SPI module innside the Arduino, in this case Arduino Zero. The SPI is a protocol of communication, in this lab we used to communicate with the SD Card to produce a signal and then display it on the oscilloscope. Also we configure the microcontroller with the instructions given by the professor at Github.

Include a brief explanation of each .C file written for your project

We use imported code from the laboratory like the UART configuration Codes. The main code was provided via Github, and the SPI code in the spi.c. Also we work with the datasheet to obtain the constants and registers to communicate the SD Card to the oscilloscope.

Part 1

Flowchart illustrating the initializing process for the SPI


Beggining
Disable Prescale (SYSCTRL)
Enable CLK SERCOM1
Configurate the CTRLA for the SPI mode
Configure the CTRLB to enable the RX pins, SC and size
Configurate the Baud Rate
SS the pin 18 Config
Enable SPI Protocol

Flowchart illustrating the process to send values using spiSend() in spi.c

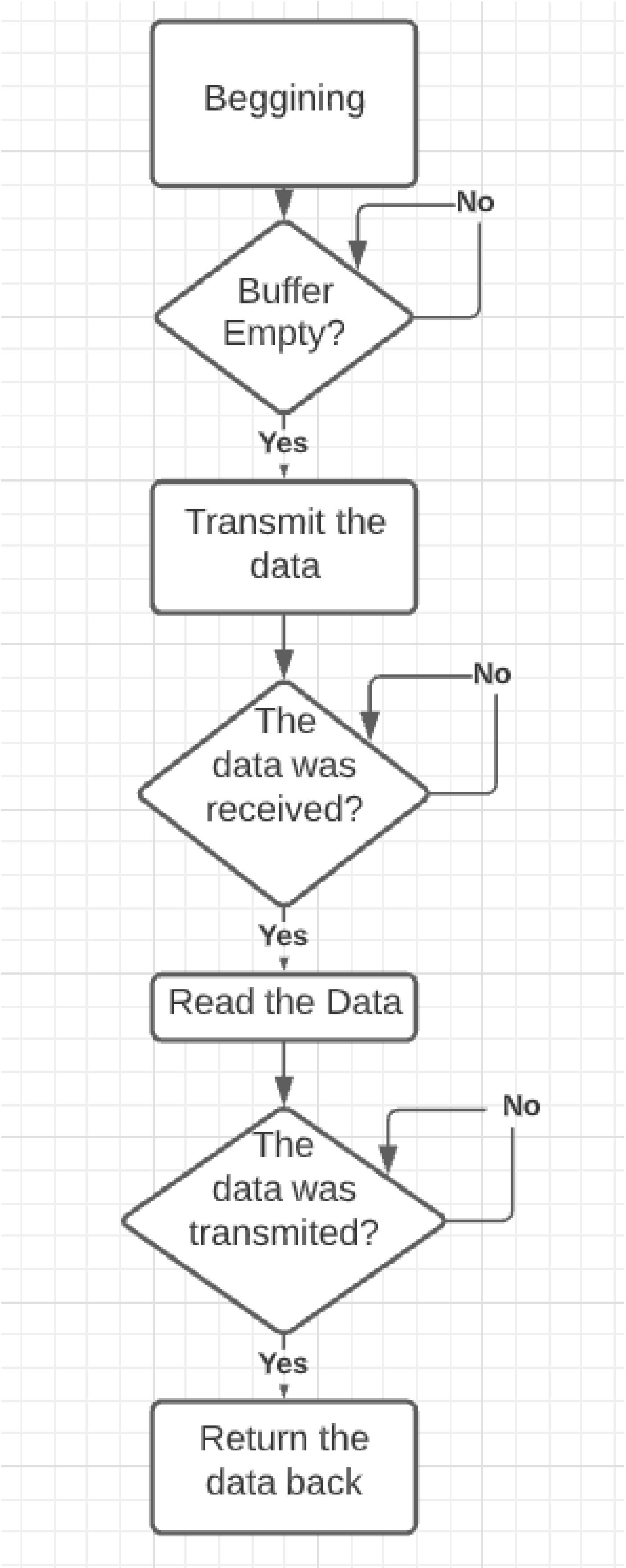
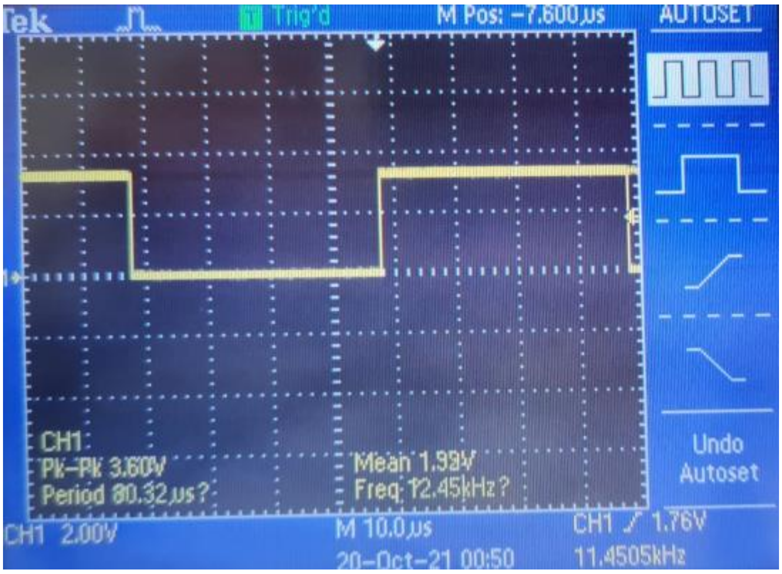

Beggining
No
Buffer Empty?
Yes
Transmit the data
No
The data was received?
Yes
Read the Data
No
The data was transmited?
Yes
Return the data back

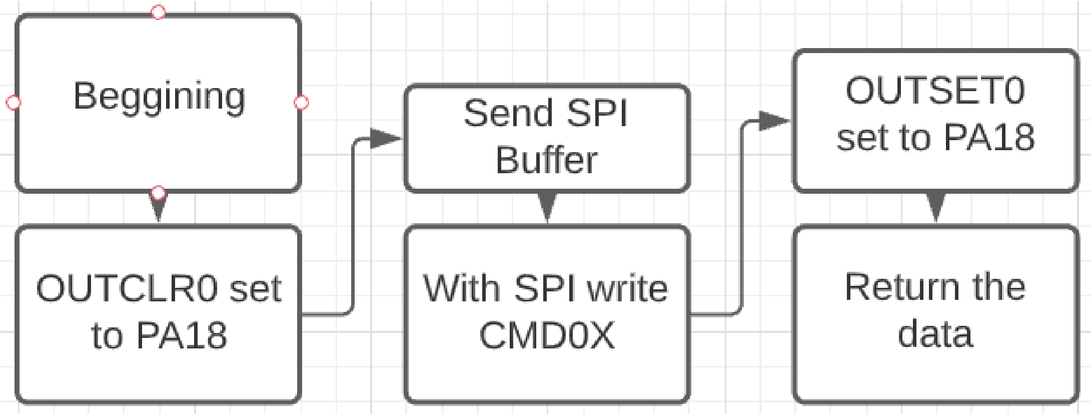
Image of the generated SPI signal over the MOSI and SS lines, including justification of the observed waveform. Was the signal observed on the oscilloscope the expected waveform?



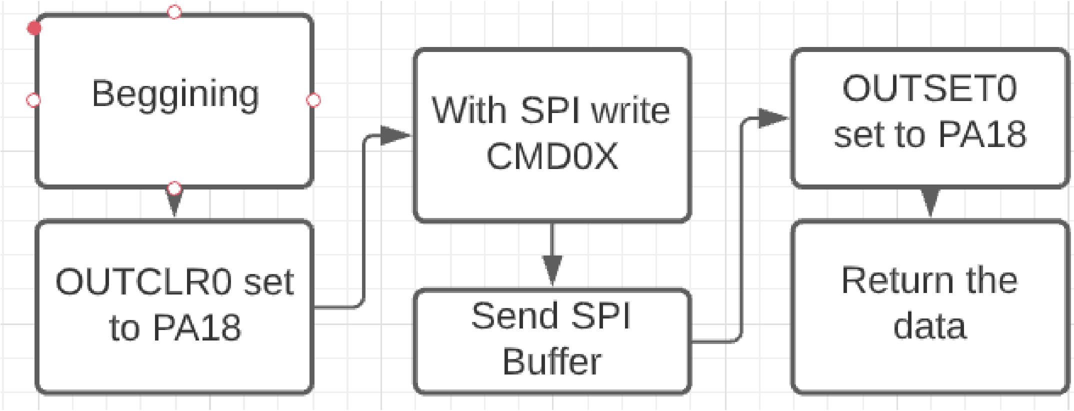
The result was as expected, a square signal, in this case with a frequency of 12.4505 KHz, and a slope of 1.76 V of 3.60 Vpp, and a period of 80.32 us.

Part 2

Flowchart of the process to *send* commands to the SD card in the spiXchg() function

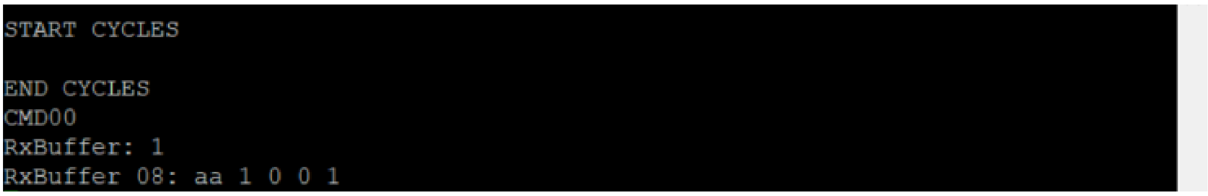


Flowchart of the process to *receive* responses from the SD card in the spiXchg() function



Interpretation of responses received from the SD card by sending commands CMD00 and CMD08.

The first response shows the connection status with the SD Card, and the second response shows that the connection is working correctly.



```
START CYCLES

END CYCLES
CMD00
RxBuffer: 1
RxBuffer 08: aa 1 0 0 1
```

Answers to questions:

When sending each byte of the command, what is the value being received from the SD?

0x01

When receiving the response from the SD, what is the value being sent to the SD?

0xFF

Part 3

What is the purpose of the initCycles() function?

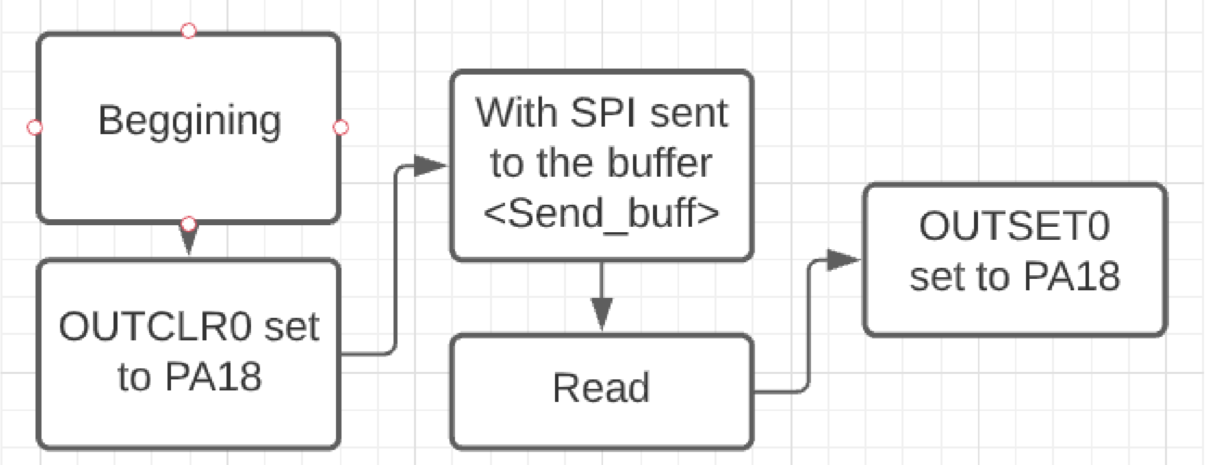
Help us to set the clock cycle, in this case with a cycle of 76

Modifications to the main program to additionally send commands CMD55 and CMD41.

We can Add to the CMD08: 0x48, 0x00, 0x00, 0x01, 0xAA, 0x87

Part 4

Flowchart of the process to send commands to the SD card in function rcvr_datablock().



Answer to the following questions:

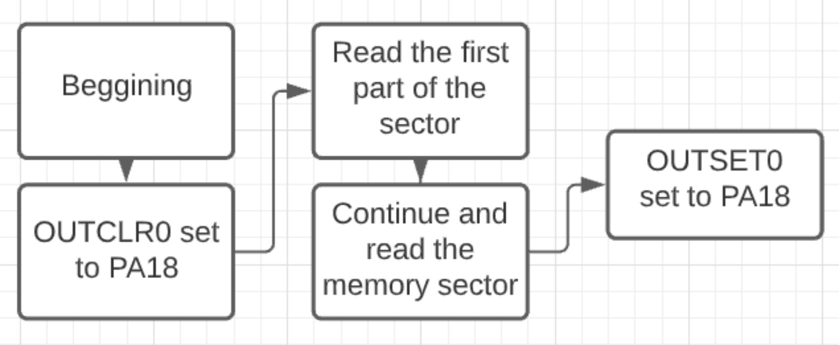
What arguments of the function are related to this part of the code? What information are these arguments providing?

The first three arguments are related to this part of the code, providing instructions, sending buffer and the size respectively.

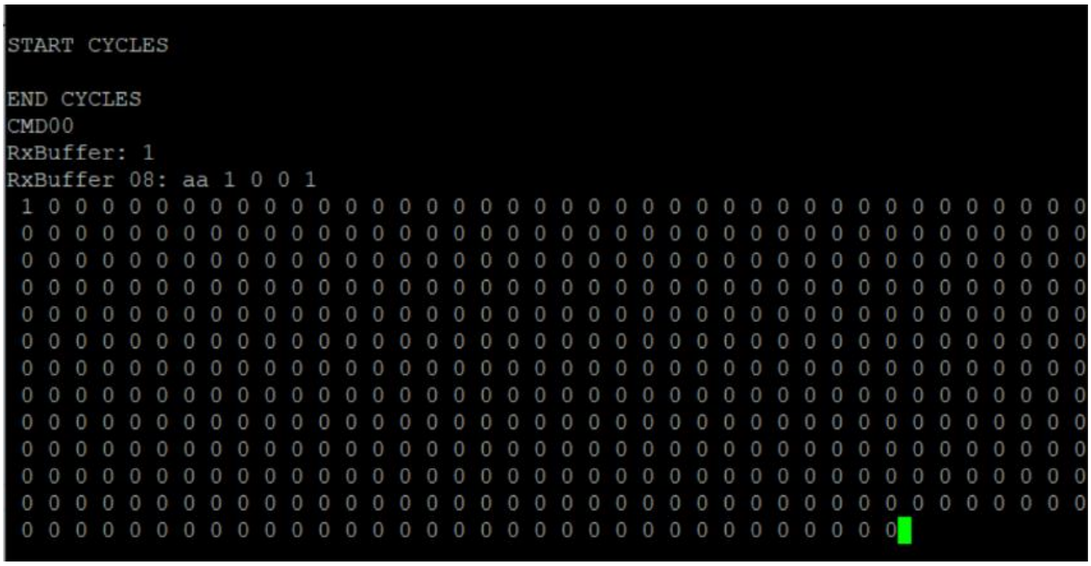
Explain how the CMD17 command argument is being passed to the SD. What is this argument for?

This argument reads the information in the block, it works like the CMD08.

Flowchart of the process to receive response from the SD card in function rcvr_datablock().


Beggining
OUTCLR0 set to PA18
Read the first part of the sector
Continue and read the memory sector
OUTSET0 set to PA18

What arguments of the rcvr_datablock() function are related to this part of the code?
In this case the last argument and the first two arguments are related to the rcvr_datablock().
What information are these arguments providing?
We can see the buffer size and the type of instruction that the SD needs to do.
Interpretation of the response received from the SD card by reading the first 512-byte block of the card.
In this case we have a recently formatted SD Card, for this reason we can see a lot of zeros.
Content of the memory locations of the first 512-byte block of the SD card



```
START CYCLES

END CYCLES
CMD00
RxBuffer: 1
RxBuffer 08: aa 1 0 0 1
 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Conclusion
My conclusion in this case, is that the SPI is a very important protocol of communication in the world, it is not just used to communicate arduino with other components, this is one of the standards used all around the world. At least in my experience at Tec Racing, we use this protocol to establish communication with a lot of different sensors, so for this reason it is very important to understand correctly how this protocol works.

Github:
https://github.com/stephyybc/Sistemas-Embebidos