# Final Project



Introduction:
The overall system uses a main controller. The controller then takes in a command from the UART then interprets the input. Depending on the command the SPI will send. data from the block memory or the switches. The AD2 then takes the data from the ADC. The I2c module then takes the data and outputs 12 bits. This then goes into our encoder. The encoder splits the 12 bits into two 6 bit packets if a packet is ‘111111’then it will concatenaste ‘11’if the packet is not all 1’s then we concatenaste a ‘01’to the front. Th UART then sends the data back to the PC. Our python code is then able to interpret the output signal and output a sintaxe or a voltage depending on if we choose the block memory or the switches.

Methods

UART:

takes in data at a baud rate of 115200. The speed of the UART is required. If the UART is required. If the UART is slower than the I2C then we may lose data. The UART receives a command and transmits output back to the PC.

SPI:

The SPI has the option to switch between block memory or the switches on the board. The SPI sends data at 10Mhz. We chose this peed due to the previous design specifications.

I2C:

The I2C takes data that is input by the AD2. The clock speed on this 100kHz. We chose this speed because it is slower than the existing UART speed.

Encoder:

The encoder takes the 12 bits that is output by the I2C. It then splits the 12 bits into 2 6 bit packets. If the bits are ‘111111’then it will concatenate ‘11’to the front if anything else ‘01’concatenates to the front. This relies on the I2done and UART done signals.

I2C : 100kHz

SPI : 10 MHz

UART : 115200 baud rate

## FIFO Depth

Write frequency = 100kHz

Reading frequency = 10 MHz

Burst length = 3000

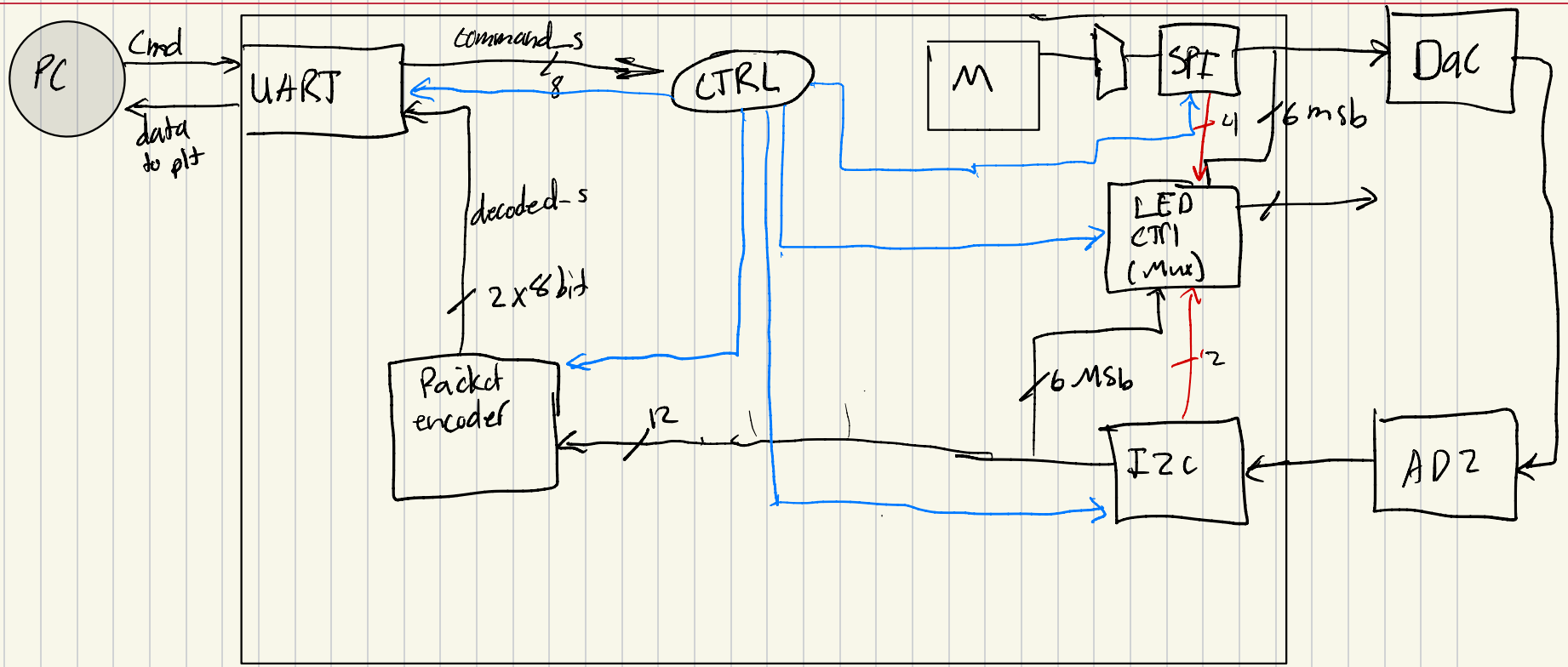
time to write = $\frac{1}{100kHz} = 10\mu s$

time to read = $\frac{1}{10MHz} = 1 \times 10^{-7} s$

$\frac{10\mu s}{1 \times 10^{-7} s} = 100$

FIFO = 3000 − 100

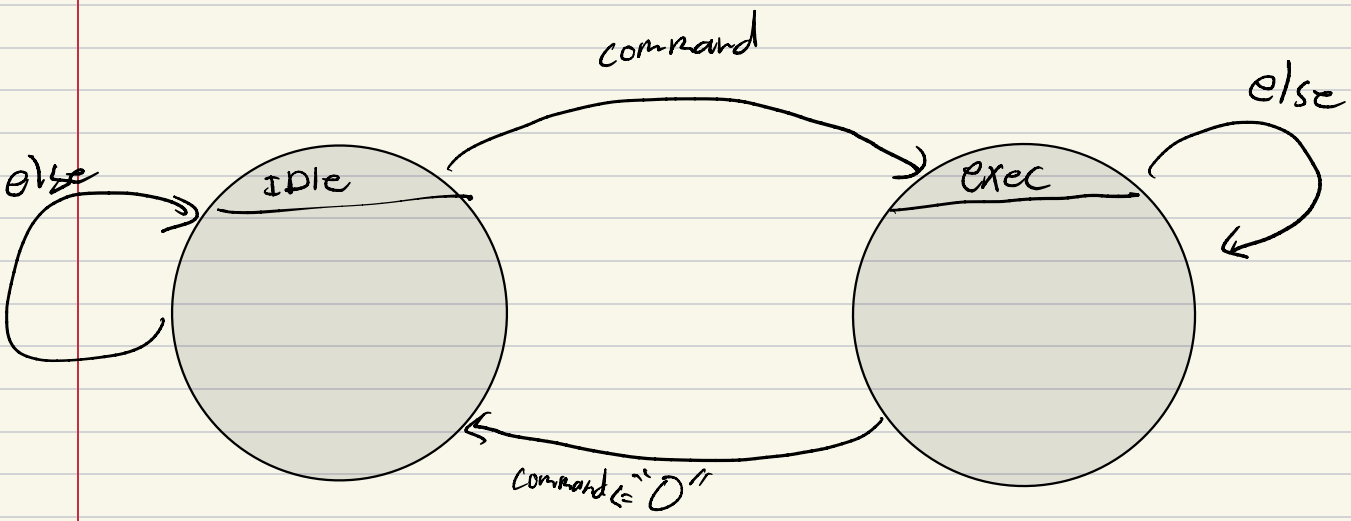
**Fifo Depth = 2900**

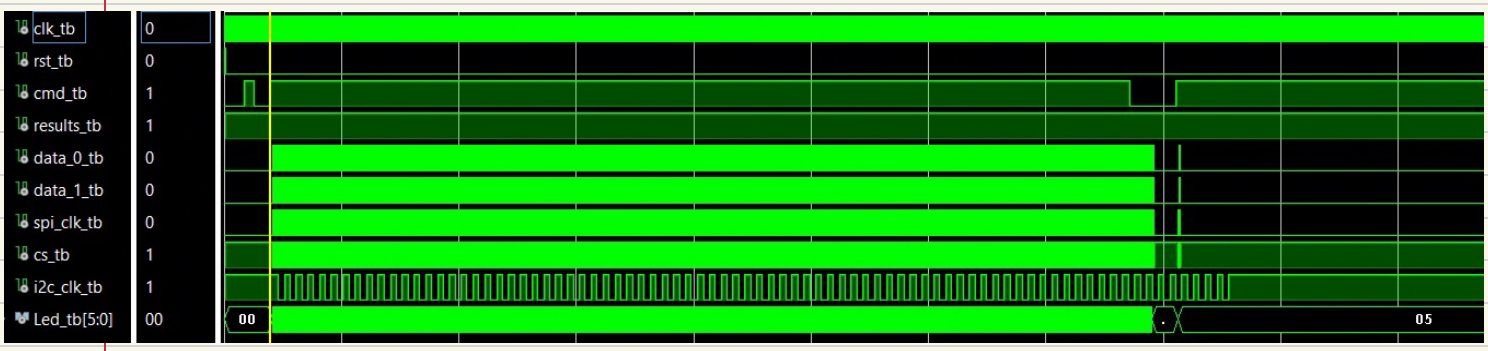



Use CS for SPI

Active Low

W


command
else
else
IDle
exec
command <= "0"


clk_tb
rst_tb
cmd_tb
results_tb
data_0_tb
data_1_tb
spi_clk_tb
cs_tb
i2c_clk_tb
Led_tb[5:0]
0
0
1
1
0
0
0
1
1
00
00
05

SPI

00 Idle
01 transmit 1/transmit 2

I2C

000 Waiting
001 config/conack
010 idle
011 transmit 1/transmit 2/ack
100 receive/masterack/stop

## Commands

| X | X | X | X | state | SPI | ADC | SW rem |
|---|---|---|---|---|---|---|---|

| Key | bin | lights | in |
|---|---|---|---|
| 0 | 0000 | X | X |
| 2 | 0010 | ADC | SW |
| 3 | 0011 | ADC | mem/cont |
| 4 | 0100 | SPI | SW |
| 5 | 0101 | SPI | mem/cont |
| 6 | 1000 | state | SW |
| 9 | 1001 | state | mem/cont |

State lights

0 0 0 0 XX Waiting
0 0 0 1 XX configuration
0 0 1 0 XX idle
0 1 0 0 XX transmitting
1 0 0 0 XX receiving

} I2C 4mSB

XXXX 01 Iple
XXXX 10 transmit

} SPI

## FIFO size

write with 12 read width is 6

Uart full count - 1736.11

Uart half count - 868

## I2C clk speed to 3ksps

$\frac{100e6}{3000} \neq 33\,333.3$

## Block mem

16 bit · 2000 = 3200

## SPI clk

$\frac{100e6}{50e3} = 2000$

## FIFO Depth

Write frequency = 100kHz

Reading frequency = 10MHz

Burst length = 3000

time to write = $\frac{1}{100kHz} = 10\mu s$

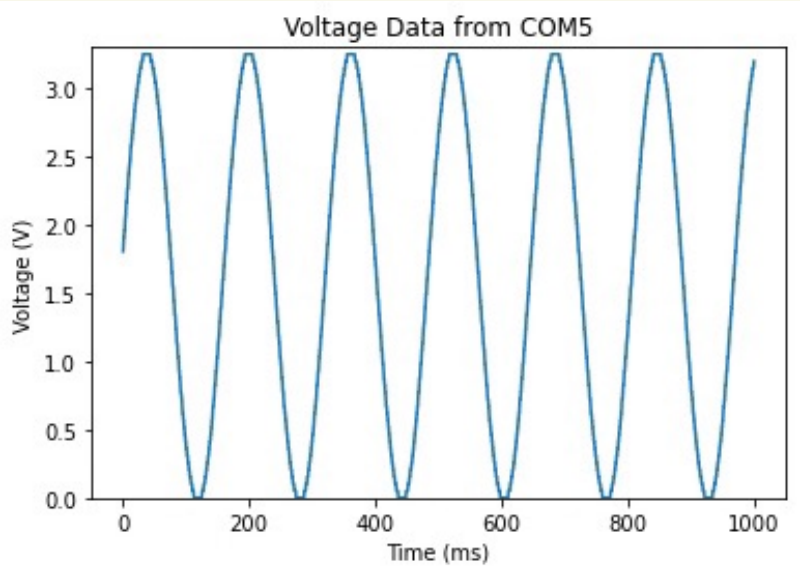
time to read = $\frac{1}{10MHz} = 1 \times 10^{-7} s$

$\frac{10\mu s}{1 \times 10^{-7} s} = 100$

$FIFO = 3000 - 100$

Fifo Depth = 2900

## Conclusion




We were able to successfully implement the final design. We learned that our I2C's speed was causing our sine wave to be warped. Once we corrected our clock speed this caused our sine wave to correctly show up. We were not able to implement the FIFO. In order to successfully implement the FIFO. We need to connect the output of the I2C to the FIFO then we take the output of the FIFO and connect it to the encoder.