

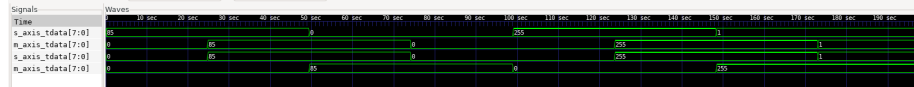
# FPGA UART ALU

[Link to GitHub Repo](#)

## Step 1 – UART Echo

### 1.3

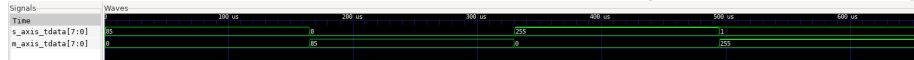
Waveform that verifies the UART modules are connected correctly



The first signal is the output of the testing `uart_tx` instance. The second and third signals are the `rx` and `tx` signals of the alu. The fourth signal is the input of the testing `uart_tx` instance. The `rx` and `tx` signals of the alu are connected to each other, which is shown in the waveform. The values 85, 0, 255, 1 being transmitted and received.

### 1.8

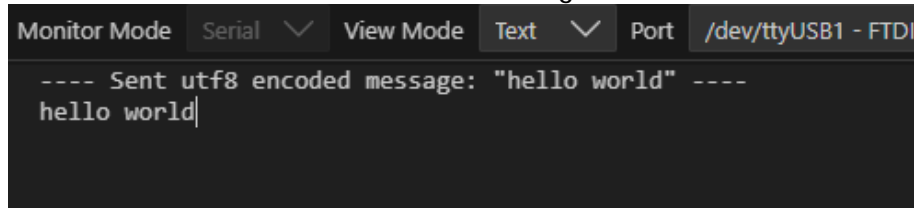
Waveform that verifies the UART modules synthesized correctly



This waveform shows the same testbench as 1.3, but after synthesis. It also only shows the signals from the testing `uart_tx` instance.

### 1.12

Screenshot of the terminal where the design echos back received data



## Difficulties

Some difficulties faced in this section included:

- Installing all of the necessary tools and software
- Getting the Verilog project structure set up
- Using a PLL to get a correct clock-frequency

## Step 2 – Packets Over UART

Python script connecting to the FPGA and sending packets

```
ericbreh@titan:~/school/verilog-project-to-silicon/hw1$ /bin/python3 /home/ericbreh/school/verilog-project-to-silicon/hw1/misc/packet_test.py
Sending packet: ec0006004869
Header: ec000600
Data: 4869
Received: 4869

Sending packet: ec00100048656c6cf20576f726c6421
Header: ec001000
Data: 48656c6cf20576f726c6421
Received: 48656c6cf20576f726c6421

Sending packet: ec00f0054657374205061636b6574
Header: ec00f000
Data: 54657374205061636b6574
Received: 54657374205061636b6574
```

### Difficulties

Some difficulties faced in this section included:

- Timing errors
- Debugging the Python script used to send packets and receive data

## Step 3 – Sending Operations Over UART

| Operation | Opcode |
|-----------|--------|
| ECHO      | 0xEC   |
| ADD       | 0xAD   |
| MUL       | 0x88   |
| DIV       | 0xD1   |

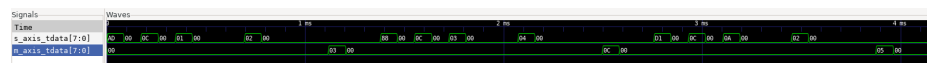
### Difficulties

Some difficulties faced in this section included:

- Debugging the Python script used for sending packets and testing

## Step 4 – ALU RTL

Waveform demonstrating that packets are transmitted, and replies are received



This shows 3 operations being sent

| Operation | Inputs (Dec) | Inputs (Hex) | Output (Dec) | Output (Hex) |
|-----------|--------------|--------------|--------------|--------------|
| ADD       | 1, 2         | 0x01, 0x02   | 3            | 0x03         |
| MUL       | 3, 4         | 0x03, 0x04   | 12           | 0x0C         |

| Operation | Inputs (Dec) | Inputs (Hex) | Output (Dec) | Output (Hex) |
|-----------|--------------|--------------|--------------|--------------|
| DIV       | 10, 2        | 0x0A, 0x02   | 5            | 0x05         |



This is zoomed in to only show the add operation

## Python script sending packets and receiving responses

```

import socket
import struct
import sys

# Create a socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Connect to the server
s.connect(('192.168.1.100', 8080))

# Send a packet
data = struct.pack('f', 1.0)
s.send(data)

# Receive a response
response = s.recv(1024)
print(response)

# Close the socket
s.close()

```

## Difficulties

Some difficulties faced in this section included:

- Not understanding the ready valid system used in the third party multiplication and division modules