

SERDES – RTL VERIFICATION

6 MONTHS INTERN AT SYNOPSYS

Name: Pranes S

Roll no.: 108121091



Signature of Manager
Hanumantha Rao



Signature of Guide
Dr. Srinivasulu Jogi
Assistant Professor

PROBLEM STATEMENT ASSIGNED:

- To create Design Under Test in Verilog and create an testbench environment for I2C Protocol with Assertions and Functional Coverage.
- Write a Perl script for I2C Bus Log Analyzer using Perl using Perl Scripting Language.

INTRODUCTION

Overview:

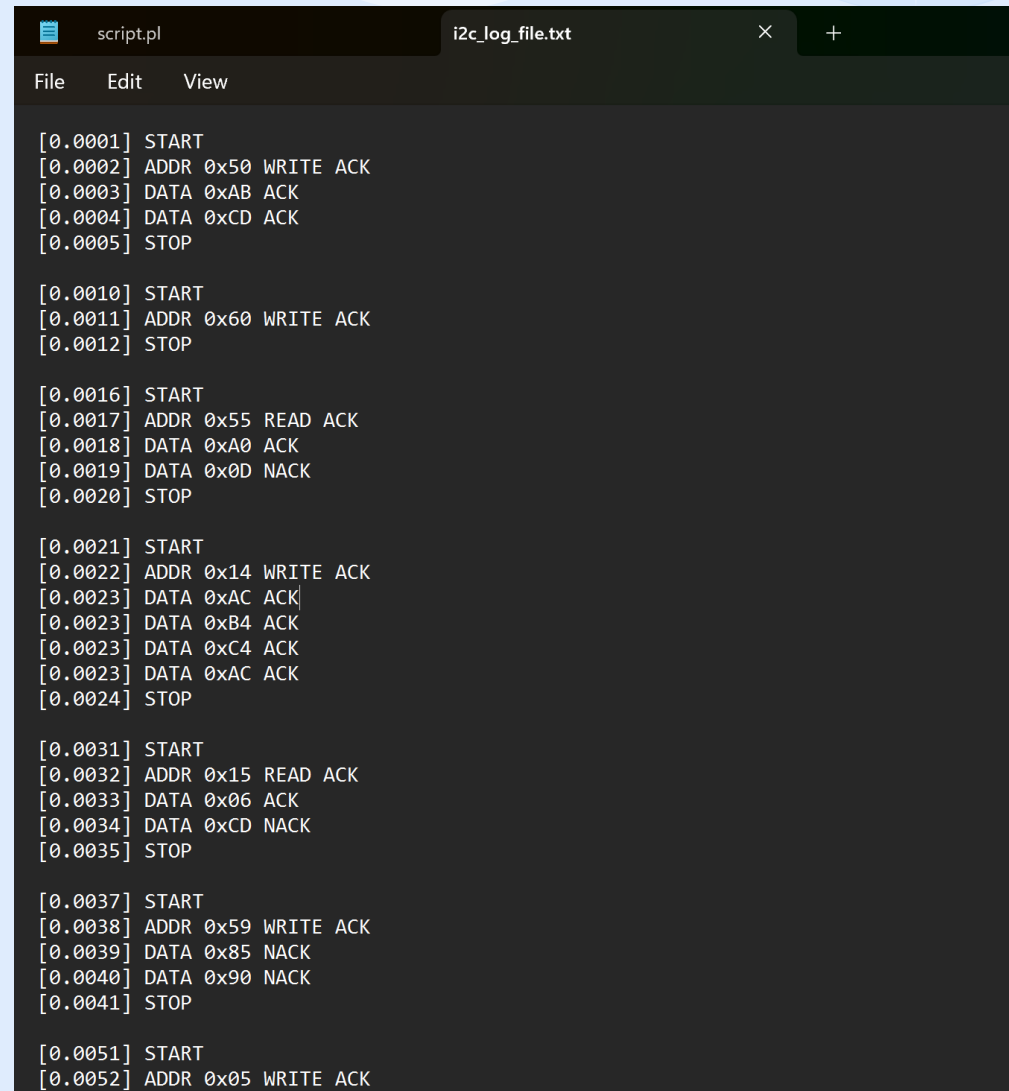
- The I2C (Inter-Integrated Circuit) protocol is a highly efficient communication protocol tailored for short-distance interactions within electronic devices. I2C facilitates the connection of microcontrollers to various peripheral devices, such as sensors, displays, and additional microcontrollers. This protocol is distinguished by its utilization of two wires—Serial Data (SDA) and Serial Clock (SCL)—with each device on the bus assigned a unique address.
- A **Perl script** is a program written in Perl, a high-level scripting language known for its powerful text processing, automation, and system administration capabilities. It is widely used for log parsing, file manipulation, web development, and automation tasks in domains like VLSI, bioinformatics, and networking.

WORK DONE TILL REVIEW1

- Gained Knowledge in System Verilog
- Built a basic testbench environment for full adder
- For I2C Protocol, Wrote DUT Verilog code for Master and Slave
 - tried two methods for data transmission
- Completed the code for Interface in the testbench Components

INPUT

- The test result will be stored in a text file like this

A screenshot of a text editor window with a dark theme. The window has two tabs: 'script.pl' and 'i2c_log_file.txt'. The 'i2c_log_file.txt' tab is active. The editor shows a log of I2C transactions. Each transaction is a block of lines starting with a timestamp in brackets. The transactions include START, ADDR, WRITE, ACK, DATA, READ, NACK, and STOP commands. The data values are in hexadecimal. The transactions are separated by blank lines.

```
script.pl i2c_log_file.txt
File Edit View

[0.0001] START
[0.0002] ADDR 0x50 WRITE ACK
[0.0003] DATA 0xAB ACK
[0.0004] DATA 0xCD ACK
[0.0005] STOP

[0.0010] START
[0.0011] ADDR 0x60 WRITE ACK
[0.0012] STOP

[0.0016] START
[0.0017] ADDR 0x55 READ ACK
[0.0018] DATA 0xA0 ACK
[0.0019] DATA 0x0D NACK
[0.0020] STOP

[0.0021] START
[0.0022] ADDR 0x14 WRITE ACK
[0.0023] DATA 0xAC ACK
[0.0023] DATA 0xB4 ACK
[0.0023] DATA 0xC4 ACK
[0.0023] DATA 0xAC ACK
[0.0024] STOP

[0.0031] START
[0.0032] ADDR 0x15 READ ACK
[0.0033] DATA 0x06 ACK
[0.0034] DATA 0xCD NACK
[0.0035] STOP

[0.0037] START
[0.0038] ADDR 0x59 WRITE ACK
[0.0039] DATA 0x85 NACK
[0.0040] DATA 0x90 NACK
[0.0041] STOP

[0.0051] START
[0.0052] ADDR 0x05 WRITE ACK
```

THE REQUIRED OUTPUT SHOULD CONTAIN

- I2C Error Report which contains the NACK's got during the process
- The total number of Transactions made
- The total number of DATA_ACK
- The total number of RW_ACK
- The total number of NACK's
- Regular expression is the primary concept used in the Perl Script

RESULT OF PERL SCRIPT

```
all done
C:\Users\prane\Desktop\perl_sort>perl script.pl
New Transaction:1
- Address: 0x50, Mode: WRITE, Response: ACK
- Data: 0xAB, Response: ACK
- Data: 0xCD, Response: ACK
- STOP

New Transaction:2
- Address: 0x60, Mode: WRITE, Response: ACK
- STOP

New Transaction:3
- Address: 0x55, Mode: READ, Response: ACK
- Data: 0xA0, Response: ACK
- Data: 0x0D, Response: NACK
- STOP

New Transaction:4
- Address: 0x14, Mode: WRITE, Response: ACK
- Data: 0xAC, Response: ACK
- Data: 0xB4, Response: ACK
- Data: 0xC4, Response: ACK
- Data: 0xAC, Response: ACK
- STOP

New Transaction:5
- Address: 0x15, Mode: READ, Response: ACK
- Data: 0x06, Response: ACK
- Data: 0xCD, Response: NACK
- STOP
```

```
New Transaction:6
- Address: 0x59, Mode: WRITE, Response: ACK
- Data: 0x85, Response: NACK
- Data: 0x90, Response: NACK
- STOP

New Transaction:7
- Address: 0x05, Mode: WRITE, Response: ACK
- Data: 0x02, Response: ACK
- Data: 0x66, Response: NACK
- STOP

I2C Error Report:
Error: NACK received for data 0x0D!
Error: NACK received for data 0xCD!
Error: NACK received for data 0x85!
Error: NACK received for data 0x90!
Error: NACK received for data 0x66!

The total no.of Transactions made:7
The total number of DATA_ACK:14
The total number of RW_ACK:7
The total number of NACK's:5


#####all done#####
```

RESULTS OF I2C PROTOCOL

```
Support of the MGLS_LICENSE_FILE and LM_LICENSE_FILE licensing environment variables will be discontinued starting with the 2025.1 release. Please update to using the SALT_LICENSE_SERVER variable.
Please contact Siemens EDA Customer Support (https://support.sw.siemens.com/) for assistance.
QuestaSim-64 qrun 2024.3.1 Utility 2024.10 Oct 17 2024
Start time: 14:23:45 on Mar 20,2025
qrun -batch -access=rw+/. -timescale 1ns/1ns -mfcu design.sv testbench.sv -voptargs="+acc=npr" -do " run -all; exit"
Creating library 'qrun.out/work'.
QuestaSim-64 vlog 2024.3.1 Compiler 2024.10 Oct 17 2024
Start time: 14:23:45 on Mar 20,2025
vlog -timescale 1ns/1ns -mfcu design.sv testbench.sv -work qrun.out/work -statslog qrun.out/stats_log -writesessionid "+qrun.out/top_dus" -csession=incr
-- Compiling package design_sv_unit
-- Compiling interface master_interface
-- Compiling interface slave_interface
-- Compiling module i2c_slave
-- Compiling module i2c_master
-- Compiling program testcase
** Warning: testcase.sv(9): (vlog-2240) Treating stand-alone use of function 'build' as an implicit VOID cast.
** Warning: testcase.sv(19): (vlog-2240) Treating stand-alone use of function 'build' as an implicit VOID cast.
-- Compiling module top

Top level modules:
    top
End time: 14:23:45 on Mar 20,2025, Elapsed time: 0:00:00
Errors: 0, Warnings: 2
QuestaSim-64 vopt 2024.3.1 Compiler 2024.10 Oct 17 2024
** Warning: (vopt-10587) Some optimizations are turned off because the +acc switch is in effect. This will cause your simulation to run slowly. Please use -access/-debug to maintain needed visibility. The +acc switch would be deprecated in a future release.
Start time: 14:23:45 on Mar 20,2025
vopt -access=rw+/. -timescale 1ns/1ns -mfcu "+acc=npr" -findtoplevels qrun.out/work+1+ -work qrun.out/work -statslog qrun.out/stats_log -csession=incr -o qrun_opt -csessionid=2
```


RESULTS

Brought to you by
 DOULOS

Doulos does not endorse training material
from other suppliers on EDA Playground.

► Languages & Libraries

▼ Tools & Simulators ?

Synopsys VCS 2023.03

Compile Options ?

-timescale=1ns/1ns +vcs+flush+

Run Options ?

Run Options

- ☐ Use **run.do** Tcl file
- ☐ Use **run.bash** shell script
- ☒ Open **EPWave** after run
- ☐ Show output file after run
- ☐ Download files after run

▼ Examples

- using EDA Playground
- VHDL
- Verilog/SystemVerilog
- UVM
- EasierUVM
- SVAUnit
- SVUnit
- VUnit (Verilog/SV)
- VUnit (VHDL)

Log

Share

```
# SCOREBOARD Start_Task
# Slave sample  data = 24          3240
# Master sample data = 24          3800
# COMPARE TASK
# PASS [24 , 24]
# -----
# Slave sample  data = 81          4680
# Master sample data = 81          5240
# COMPARE TASK
# PASS [81 , 81]
# -----
# Slave sample  data = 09          6120
# Master sample data = 09          6680
# COMPARE TASK
# PASS [09 , 09]
# -----
# Slave sample  data = 63          7560
# Master sample data = 63          8120
# COMPARE TASK
# PASS [63 , 63]
# -----
# Slave sample  data = 0d          9000
# Master sample data = 0d          9560
# COMPARE TASK
# PASS [0d , 0d]
# -----
# SB REPORT Errors =          0
# Master sample data = 0d        11000
# ENVIRONMENT Build
```

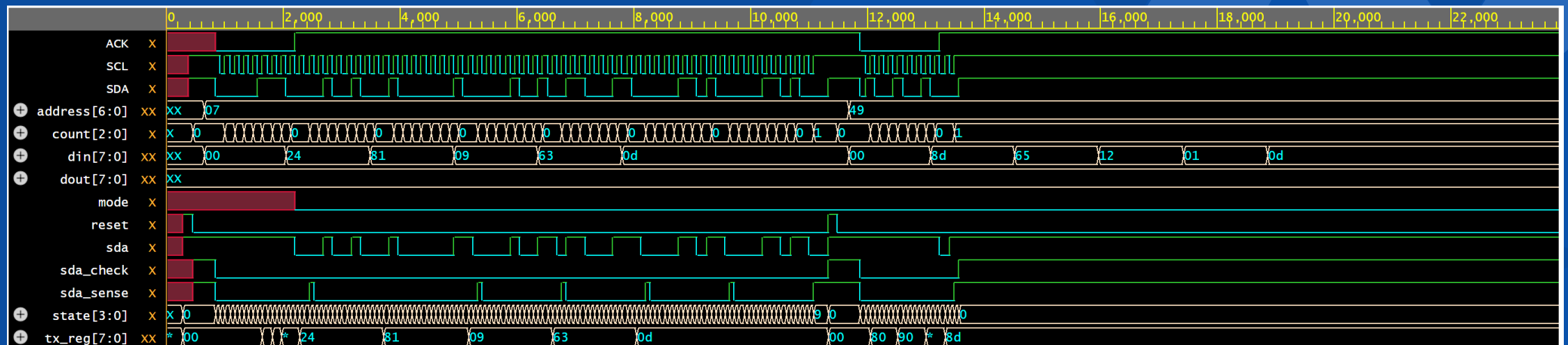
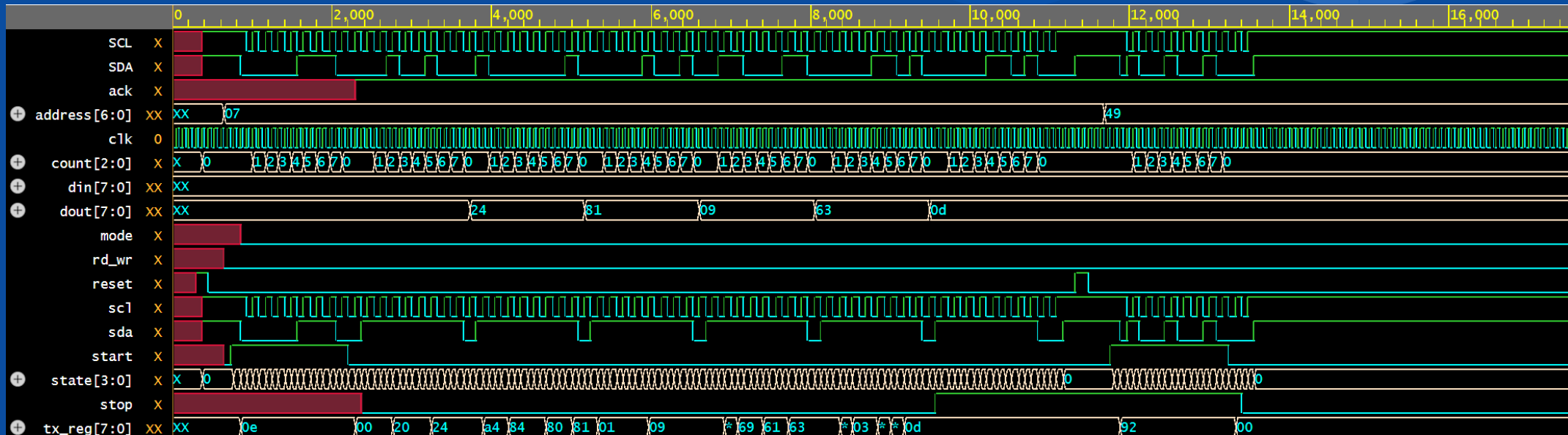
RESULTS

- These are all the results after the transaction
- Code has to be optimized with more assertions to reduce the number of fail transaction
- Functional coverage is yet to be done

```
# -----  
# Slave sample    data = 65                15720  
# Master sample   data = 0d                16280  
# COMPARE TASK  
# COMPARE TASK - Fail####  
# FAIL [0d , 65]  
# -----  
# Slave sample    data = 12                17160  
# Master sample   data = 0d                17720  
# COMPARE TASK  
# COMPARE TASK - Fail####  
# FAIL [0d , 12]
```

```
# -----  
# Slave sample    data = 01                18600  
# Master sample   data = 0d                19160  
# COMPARE TASK  
# COMPARE TASK - Fail####  
# FAIL [0d , 01]  
# -----  
# Slave sample    data = 0d                20040  
# Master sample   data = 0d                20600  
# COMPARE TASK  
# PASS [0d , 0d]  
# -----  
# SB REPORT Errors =                      4  
# Master sample   data = 0d                22040  
# ** Note: $finish      : testcase.sv(30)  
#   Time: 1022040 ns  Iteration: 1  Instance: /top/t1  
# End time: 14:23:46 on Mar 20,2025, Elapsed time: 0:00:01
```

SIMULATION RESULTS OF MASTER AND SLAVE



FUTURE WORK

- Write code for functional coverage and analyse code coverage whether all parts of the code is functioning properly
- Write more assertions and checkers to reduce the failed transition

REFERENCES

- Datasheets for I2C given by Synopsys
- **Sowmyashree B. G. and Arun Raj S. R.**, "Designing of I2C Master Based on Speed Modes," *Proc. Int. Conf. Recent Trends in Technology (ICRTT)*, Int. J. Eng. Res. Technol. (IJERT), vol. 6, no. 15, pp. 1-5, 2018.
- **Y. Zhang, X. Li, and Z. Wang**, "Implementation of I²C Communication Protocol in FPGA," in *Proceedings of the 2018 IEEE International Conference on Information and Automation (ICIA)*, Wuyishan, China, 2018, pp. 987-991. doi: 10.1109/ICInfA.2018.8812495.
- Smitha A and P. A. Vijaya, "Design of I²C Protocol in Verilog—A New Approach," *PiCES Journal*, vol. 9, Dec. 2020.