ECSE 318 Lab 3 Report

Group 5

Nabeeh Daouk and Kyle Heston

Sunday, October 27, 2023

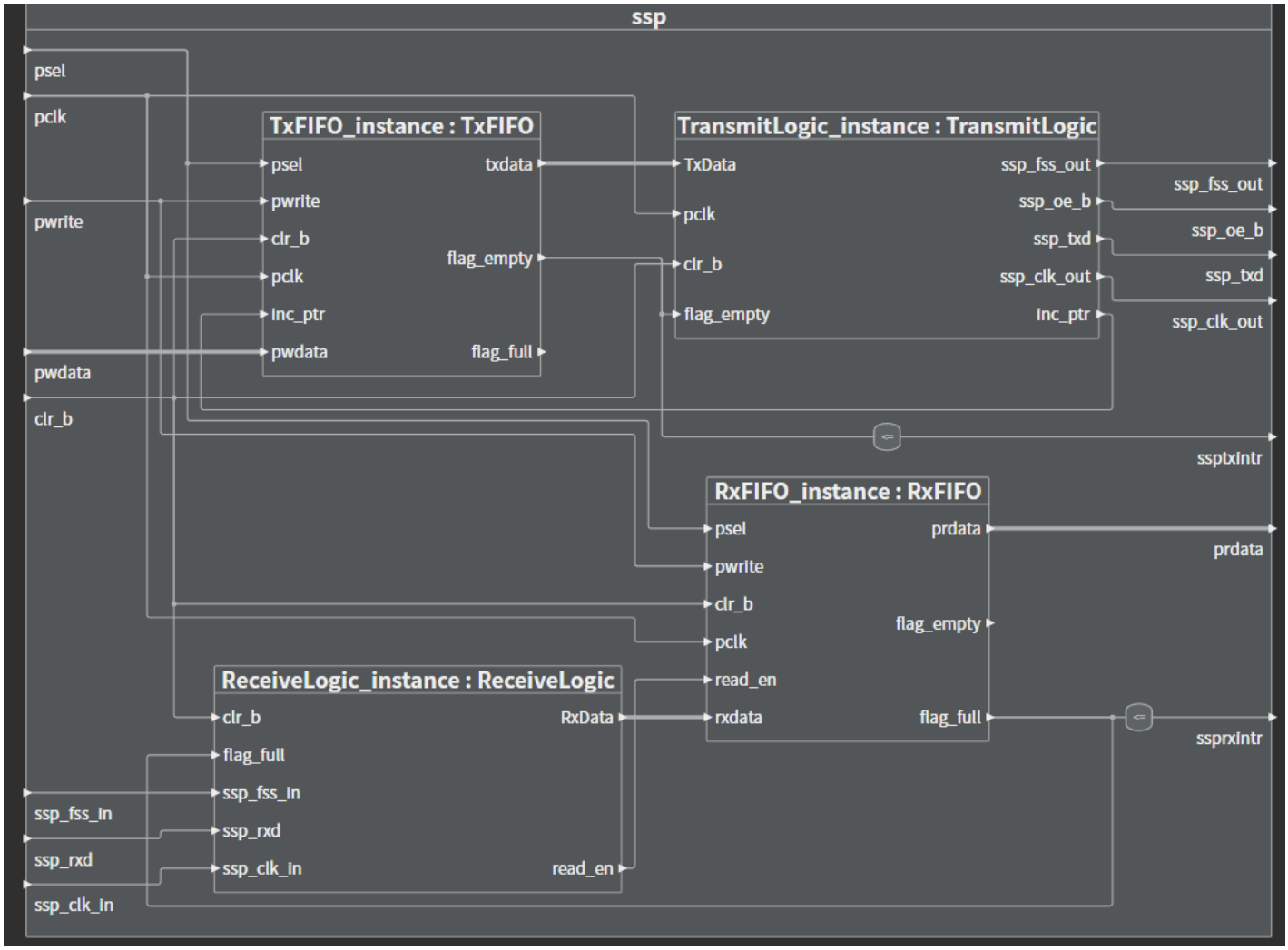
Problem 1:

```
#                   0  source: x           dest: x           illigal:x         win:0
#                   5  source: 0           dest:12           illigal:0         win:0
#                  10  source: 0           dest:12           illigal:0         win:0
#                  20  source: 0           dest:12           illigal:0         win:0
#                  30  source: 1           dest:12           illigal:0         win:0
#                  40  source: 2           dest:12           illigal:0         win:0
#                  50  source: 3           dest: 8           illigal:0         win:0
#                  60  source: 3           dest: 6           illigal:0         win:0
#                  70  source: 8           dest: 6           illigal:0         win:0
#                  80  source: 7           dest: 4           illigal:0         win:0
#                  90  source: 3           dest: 4           illigal:0         win:0
#                 100  source: 3           dest: 8           illigal:0         win:0
#                 110  source: 3           dest: 1           illigal:0         win:0
#                 120  source: 3           dest:12           illigal:0         win:0
#                 130  source: 5           dest:12           illigal:0         win:0
#                 140  source: 3           dest: 9           illigal:0         win:0
#                 150  source: 1           dest:10           illigal:0         win:0
#                 160  source: 1           dest: 3           illigal:0         win:0
#                 170  source:10           dest: 3           illigal:0         win:0
#                 180  source: 8           dest: 1           illigal:0         win:0
#                 190  source: 9           dest: 1           illigal:0         win:0
#                 200  source: 5           dest:12           illigal:0         win:0
#                 210  source: 7           dest: 6           illigal:0         win:0
#                 220  source: 7           dest: 5           illigal:0         win:0
#                 230  source: 7           dest: 8           illigal:0         win:0
#                 240  source: 1           dest:12           illigal:0         win:0
#                 250  source: 7           dest: 1           illigal:0         win:0
#                 260  source: 8           dest: 7           illigal:0         win:0
#                 270  source: 4           dest: 8           illigal:0         win:0
#                 280  source: 4           dest: 9           illigal:0         win:0
#                 290  source: 4           dest: 7           illigal:0         win:0
#                 300  source: 0           dest:12           illigal:0         win:0
#                 310  source: 4           dest:12           illigal:0         win:0
#                 320  source: 9           dest: 7           illigal:0         win:0
#                 330  source: 8           dest: 7           illigal:0         win:0
#                 340  source: 0           dest: 7           illigal:0         win:0
#                 345  source: 0           dest: 1           illigal:1         win:0
#                 350  source: 0           dest: 1           illigal:1         win:0
#                 355  source: 1           dest: 8           illigal:0         win:0
#                 360  source: 1           dest: 8           illigal:0         win:0
#                 370  source: 1           dest: 7           illigal:0         win:0
#                 380  source: 8           dest: 7           illigal:0         win:0
#                 390  source: 0           dest: 8           illigal:0         win:0
#                 400  source: 0           dest: 3           illigal:0         win:0
#                 410  source: 0           dest:12           illigal:0         win:0
#                 420  source: 2           dest: 9           illigal:0         win:0
#                 430  source: 2           dest:12           illigal:0         win:0
#                 440  source: 2           dest: 6           illigal:0         win:0
#                 450  source: 8           dest: 0           illigal:0         win:0
#                 455  source:12           dest: 3           illigal:1         win:0
#                 460  source:12           dest: 3           illigal:1         win:0
#                 465  source: 2           dest: 0           illigal:0         win:0
#                 470  source: 2           dest: 0           illigal:0         win:0
#                 480  source: 2           dest: 8           illigal:0         win:0
#                 490  source: 0           dest:10           illigal:0         win:0
#                 500  source: 0           dest: 2           illigal:0         win:0
#                 510  source:10           dest: 2           illigal:0         win:0
#                 520  source: 1           dest:10           illigal:0         win:0
#                 525  source: 7           dest:10           illigal:1         win:0
#                 530  source: 7           dest:10           illigal:1         win:0
#                 535  source: 1           dest:11           illigal:0         win:0
#                 540  source: 1           dest:11           illigal:0         win:0
#                 550  source: 1           dest: 2           illigal:0         win:0
#                 560  source: 1           dest: 3           illigal:0         win:0
#                 570  source: 8           dest: 2           illigal:0         win:0
#                 580  source:10           dest: 2           illigal:0         win:0
#                 585  source:10           dest: 1           illigal:1         win:0
#                 590  source:10           dest: 1           illigal:1         win:0
#                 595  source: 1           dest:12           illigal:0         win:0
#                 600  source: 1           dest:12           illigal:0         win:0
```

```
#                 610  source: 5          dest: 8          illigal:0          win:0
#                 620  source: 5          dest: 3          illigal:0          win:0
#                 630  source: 5          dest:12          illigal:0          win:0
#                 640  source: 8          dest: 3          illigal:0          win:0
#                 650  source:11          dest:12          illigal:0          win:0
#                 660  source: 4          dest:12          illigal:0          win:0
#                 665  source: 8          dest:12          illigal:1          win:0
#                 670  source: 8          dest:12          illigal:1          win:0
#                 675  source: 5          dest: 1          illigal:0          win:0
#                 680  source: 5          dest: 1          illigal:0          win:0
#                 690  source: 4          dest: 1          illigal:0          win:0
#                 700  source: 4          dest: 1          illigal:0          win:0
#                 710  source: 6          dest: 0          illigal:0          win:0
#                 720  source: 6          dest: 3          illigal:0          win:0
#                 730  source: 0          dest: 3          illigal:0          win:0
#                 740  source: 5          dest: 3          illigal:0          win:0
#                 750  source: 9          dest: 5          illigal:0          win:0
#                 760  source: 4          dest: 5          illigal:0          win:0
#                 770  source: 6          dest: 8          illigal:0          win:0
#                 780  source: 6          dest: 9          illigal:0          win:0
#                 790  source: 6          dest: 1          illigal:0          win:0
#                 800  source: 9          dest: 1          illigal:0          win:0
#                 810  source: 8          dest: 1          illigal:0          win:0
#                 820  source: 6          dest: 4          illigal:0          win:0
#                 830  source: 6          dest: 1          illigal:0          win:0
#                 840  source: 6          dest: 4          illigal:0          win:0
#                 850  source: 6          dest:12          illigal:0          win:0
#                 860  source: 6          dest:12          illigal:0          win:0
#                 870  source: 7          dest:12          illigal:0          win:0
#                 880  source: 7          dest:12          illigal:0          win:0
#                 890  source: 3          dest:12          illigal:0          win:0
#                 900  source: 3          dest:12          illigal:0          win:0
#                 910  source: 2          dest:12          illigal:0          win:0
#                 920  source: 3          dest:12          illigal:0          win:0
#                 930  source: 7          dest:12          illigal:0          win:0
#                 940  source: 1          dest:12          illigal:0          win:0
#                 950  source: 2          dest:12          illigal:0          win:0
#                 960  source: 3          dest:12          illigal:0          win:0
#                 970  source: 2          dest:12          illigal:0          win:0
#                 980  source: 2          dest:12          illigal:0          win:0
#                 990  source: 7          dest:12          illigal:0          win:0
#                1000  source: 1          dest:12          illigal:0          win:0
#                1010  source: 7          dest:12          illigal:0          win:0
#                1020  source: 3          dest:12          illigal:0          win:0
#                1030  source: 1          dest:12          illigal:0          win:0
#                1040  source: 7          dest:12          illigal:0          win:0
#                1050  source: 1          dest:12          illigal:0          win:0
#                1060  source: 2          dest:12          illigal:0          win:0
#                1070  source: 3          dest:12          illigal:0          win:0
#                1080  source: 7          dest:12          illigal:0          win:0
#                1090  source: 1          dest:12          illigal:0          win:0
#                1100  source: 2          dest:12          illigal:0          win:0
#                1110  source: 3          dest:12          illigal:0          win:0
#                1120  source: 7          dest:12          illigal:0          win:0
#                1130  source: 1          dest:12          illigal:0          win:0
#                1140  source: 3          dest:12          illigal:0          win:0
#                1145  source: 7          dest:12          illigal:1          win:0
#                1150  source: 7          dest:12          illigal:1          win:0
#                1155  source: 4          dest:12          illigal:0          win:0
#                1160  source: 4          dest:12          illigal:0          win:0
#                1170  source: 5          dest:12          illigal:0          win:0
#                1180  source: 1          dest:12          illigal:0          win:0
#                1190  source: 3          dest:12          illigal:0          win:0
#                1200  source: 4          dest:12          illigal:0          win:0
#                1210  source: 5          dest:12          illigal:0          win:1
```

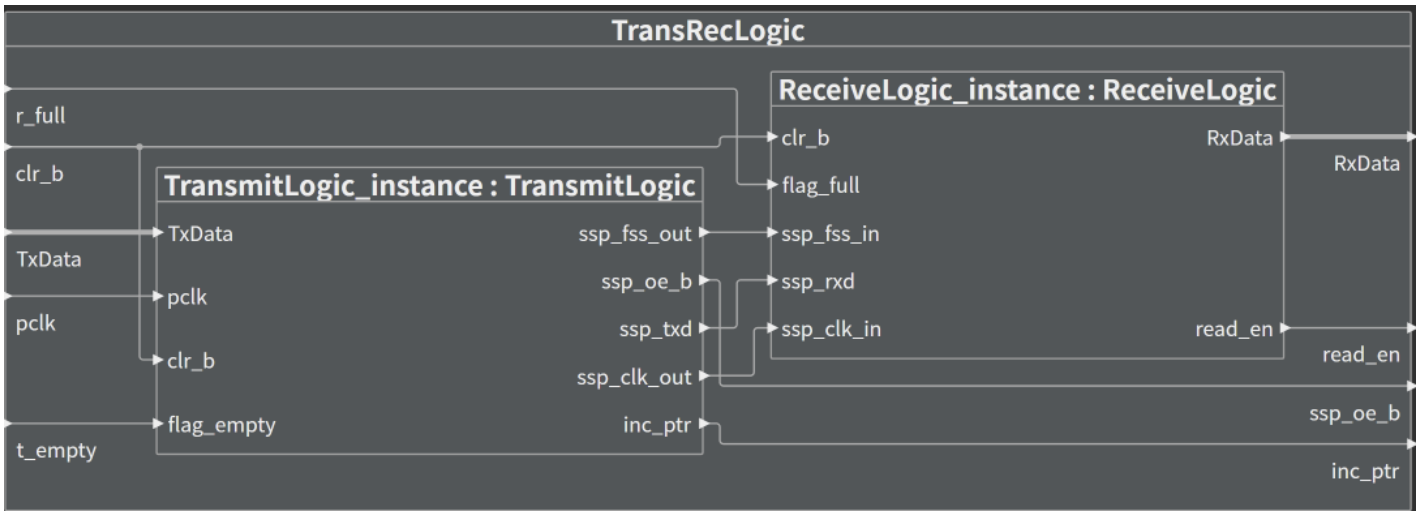
FreeCell Testbench Results

Problem 2:



Block Diagram at SSP level

Shows implementation of TxFIFO, RxFIFO, Transmit and Receive Logic blocks with appropriate I/O connections.



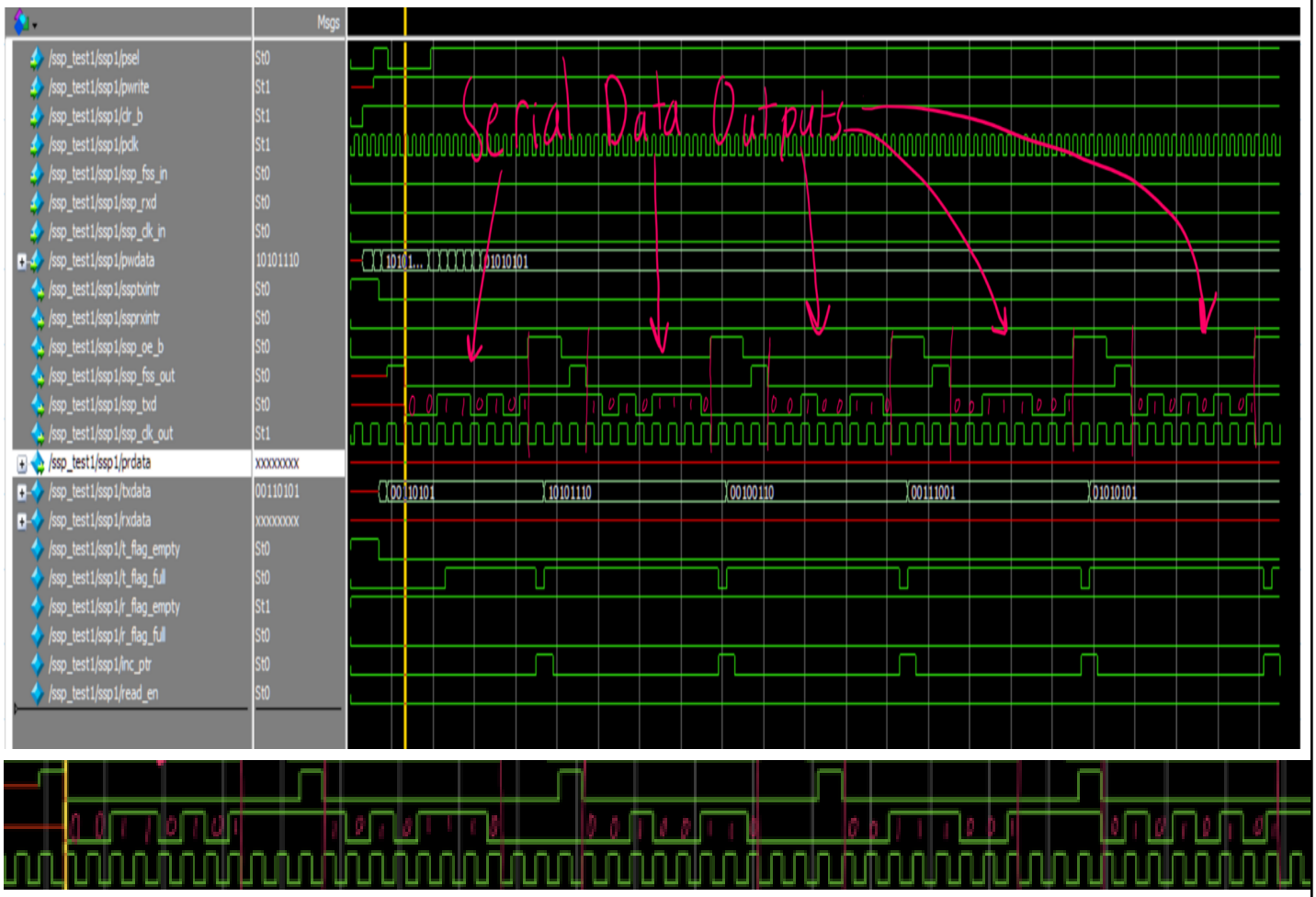
Block Diagram at TransRec level

Shows implementation of TransmitLogic and ReceiveLogic. With appropriate connections, these blocks function as the larger TransRec Logic block. We used this to test the transmit receive functions, although not instantiated in final ssp design.

```
# run -all
# This testbench will load 4 bytes of data into the SSP
# The transmit output of the SSP is connected to the
#  serial input of the recieve side
# By toggeleing the correct signals, we can send data in, and
#  expect paralell data to be returned to the processor
# This testbench will also monitor the interupt signals
#  alert the signal trigger via the console
#                    0          SSPTX_INTR triggered (AKA: TxFIFO is EMPTY)
# TRANSMITING DATA...
# SSP set in transmit mode
# TRANSMITING BYTE: 00 = 00000000
# TRANSMITING BYTE: d3 = 11010011
# TRANSMITING BYTE: a7 = 10100111
# TRANSMITING BYTE: ff = 11111111
# TRANSMITING BYTE: 12 = 00010010
#
# RECIEVING DATA...
# SSP set in recieve mode
#                  905          SSPTX_INTR triggered (AKA: TxFIFO is EMPTY)
#                  935          SSPrX_INTR triggered (AKA: RxFIFO is FULL)
# BYTE RECIEVED: 00 = 00000000
# BYTE RECIEVED: d3 = 11010011
# BYTE RECIEVED: a7 = 10100111
# BYTE RECIEVED: ff = 11111111
```

SSP Testbench (ssp_tb) *Note: we wrote this TB*

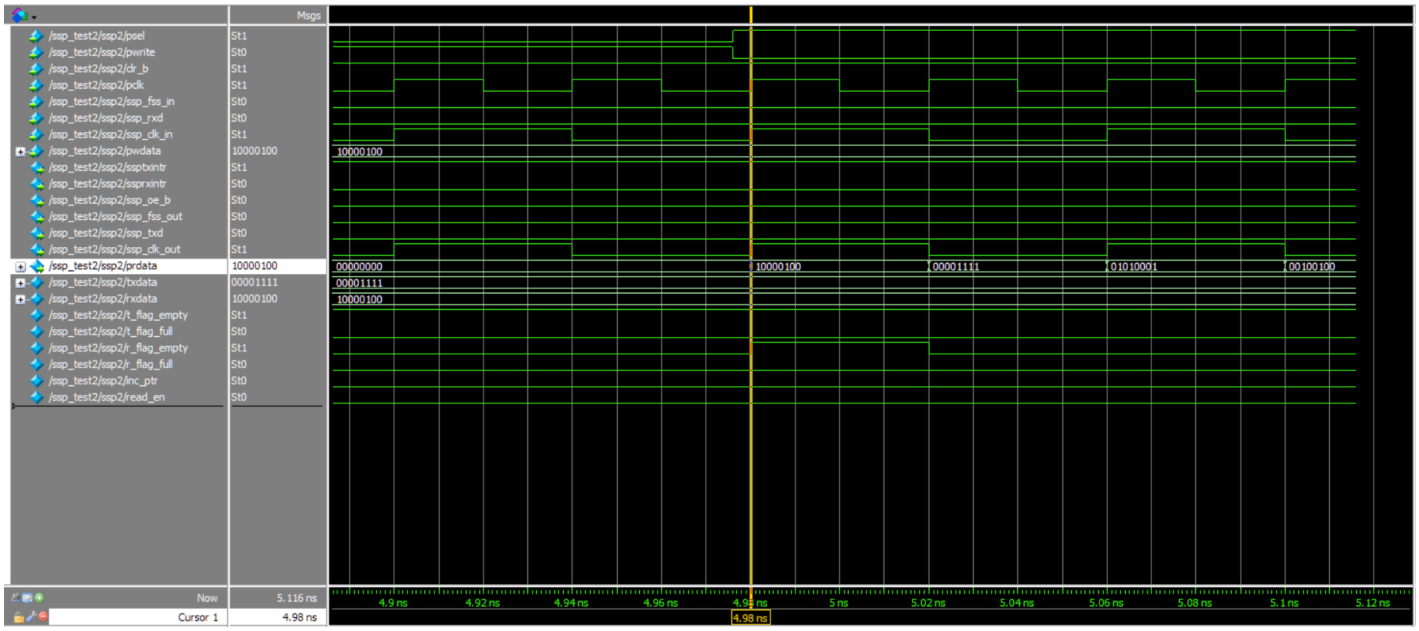
Note: In this TB, we connect the output serial data to the input of the receive side of ssp.
Note: the input data 12 appears as output on waveform at a later time. In this print out, we are showing that a full TxFIFO will not be loaded with additional data until the first data is output. The output data shown in this print out are therefore correct and in order.



SSP Test 1 (ssp_test1) *Testbench from website*

Note: In this TB, we input “byte” sized data in the input and are able to visualize the serial data as an output of the testbench. This shows the transmit side of the ssp as functional.

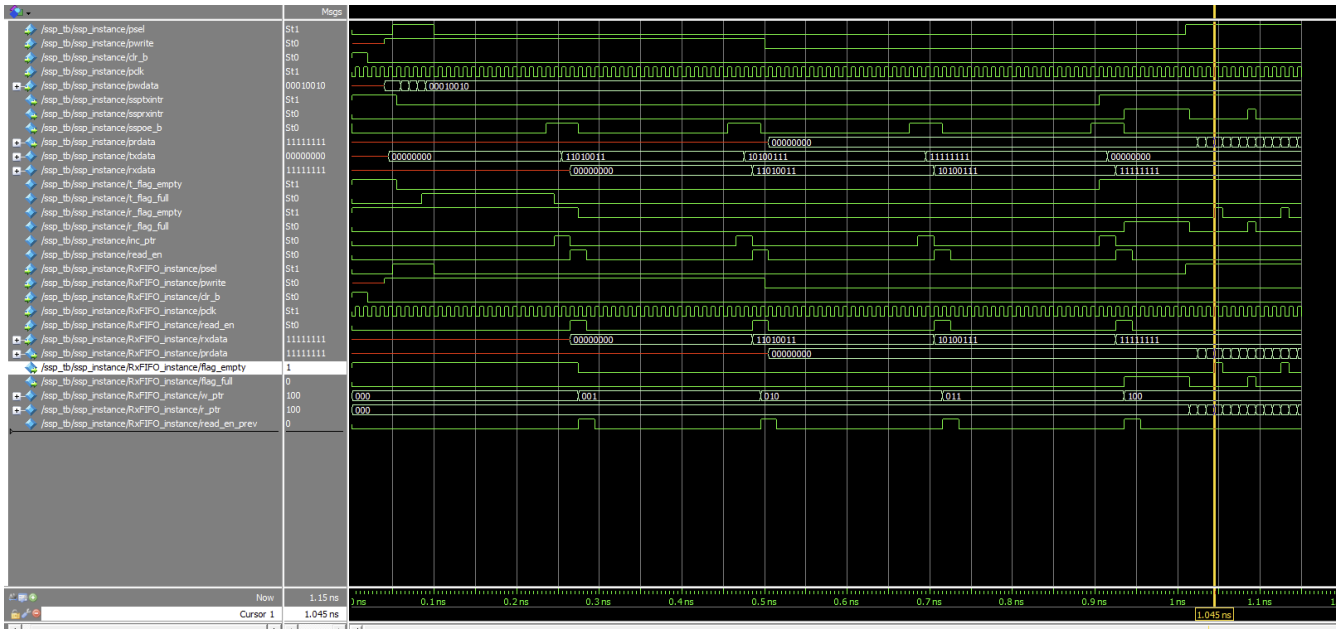
Note: Serial Data depicted and transcribed in PINK above




```
# Break in Module ssp_test2 at C:/Users/hesto/Source Code/Repositories/ECSE318/Lab3/SSP/ssp_test2.v line 39
VSIM 9> restart
# Loading work.ssp_test2
VSIM 10> run -all
# data_out: 10000100
# data_out: 00001111
# data_out: 01010001
# data_out: 00100100
# ** Note: $stop    : C:/Users/hesto/Source Code/Repositories/ECSE318/Lab3/SSP/ssp_test2.v(39)
#    Time: 5116 ps  Iteration: 0  Instance: /ssp_test2
# Break in Module ssp test2 at C:/Users/hesto/Source Code/Repositories/ECSE318/Lab3/SSP/ssp test2.v line 39
```

SSP Test 2 (ssp_test2) *Testbench from website*

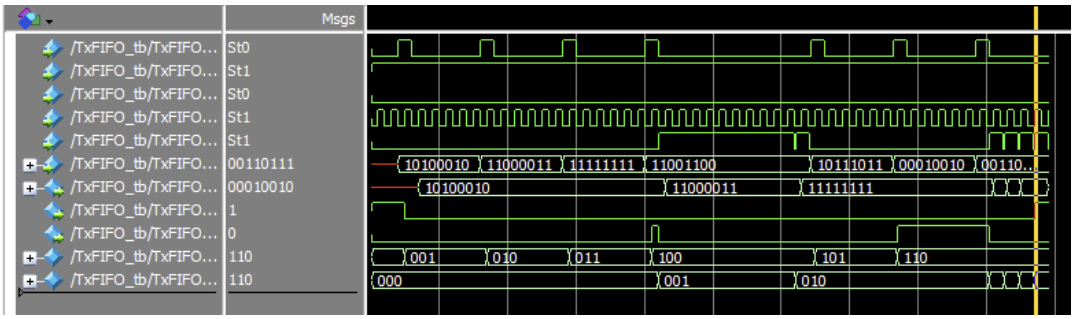
Note: In this TB, we connect the output serial data to the input of the receive side of ssp. We can see the data returned through the receive path on the right of the yellow cursor above (prdata). This validates BOTH the transmit and receive paths of the ssp.



Empty Flag for pulling from empty TxFIFO

Waveform shows the empty flag set high to indicate that data should *not* be pulled from TxFIFO into trans/rec since it will be old data that has already been pulled once. Though not part of the assignment, we could use this flag and a connection to the processor to create an invalid data request.

```
# run -all
# tx_data: xx
# full: 0 | empty: 1
# START
# tx_data: xx
# full: 0 | empty: 0
#
# tx_data: a2
# full: 0 | empty: 0
#
# tx_data: a2
# full: 0 | empty: 0
#
# tx_data: a2
# full: 1 | empty: 0
#
# tx_data: c3
# full: 0 | empty: 0
#
# tx_data: ff
# full: 0 | empty: 0
#
# tx_data: ff
# full: 0 | empty: 0
#
# tx_data: ff
# full: 1 | empty: 0
#
# tx_data: ff
# full: 1 | empty: 0
#
# tx_data: cc
# full: 0 | empty: 0
#
# tx_data: bb
# full: 0 | empty: 0
#
# tx_data: 12
# full: 0 | empty: 0
#
# tx_data: 12
# full: 0 | empty: 1
```



Example Intermediate Testbench Results for TxFIFO

Each module has its own tb for verification and targeted troubleshooting and module level verification.

NOTE on FIFO Modules:

Both the Rx and Tx FIFOs use a 4-byte memory array to store data values. In order to indicate which location in mem to store new data or read existing data, we have write and read pointers. TxFIFO write pointer is incremented when transmit path is active(pwrite = 1), TxFIFO is *not* full and new data must be written. TxFIFO read pointer is incremented based on the transrec logic output inc_ptr, which is set high for one clock cycle when a local counter indicates that the data has been fully serialized. RxFIFO write pointer is incremented when read path is active(pwrite = 0), RxFIFO is *not* full and newly parallelized data must be written from SSP port. RxFIFO read pointer is incremented on positive edge of psel and pclk when read path is active(pwrite = 0). In order to set the full and empty flags, an additional pointer bit is used. This ‘phase’ bit tracks what cycle through the FIFO the pointer is on. When the read and write phases are the same *and* the pointer locations are the same, we can set a flag(empty for TxFIFO or full for RxFIFO), which then triggers the interrupt signal.