# **Testing and Results**

## **Data Extraction Block**

The components in the data extraction block were tested using VHDL testbenches and were tested for intended functionality. This means that the test signals generated by each testbench were imitated to represent signals from other parts of the system that would occur under normal operating conditions. A test table was generated for each component containing the descriptions of the purpose of the test, component specific test configurations, component configuration during testing and test pass conditions that the component needs to meet in order to be considered functional.

### **Data Extraction Controller**

|  |  |  |  |
| --- | --- | --- | --- |
| Purpose of Test | Test Configuration | Component Configuration | Test Pass Conditions |
| To test if the component functions as intended in normal operating conditions. | * + The input clock is generated by the user's code.   + The testing code has been written to follow the stages of the state machine according to which the component operates therefore the timing of the input signals is accurate relative to the timing of the actual component operation.   + The DONErx signal is asserted high for one clock cycle every 20 clock cycles to imitate the pulse from the SPI master that notifies the controller that extraction of data piece has been complete. This number would signify how long the actual SPI takes to extract the sample but for testing purposes it does not need to be accurate as it does not affect the actual operation of the component   + The bufferEmpty signal is asserted at the beginning to notify the controller that the buffer needs filling. After 12 repetitions of the DONErx signal, the bufferEmpty signal is asserted LOW after a brief 5 clock cycle delay.   + There is a 300 clock cycle delay in between the bufferEmpty signal assertions.   + Gate level simulation was used to simulate the component operation to obtain realistic response. | * + The communication system is configured to handle 12 pieces of data. Therefore the component has to send signals 12 times in order to extract 12 pieces of data. | * + The controller should generate 12 NEXT\_ADD, STORE and EXTRACT signals when 12 DONErx signals are generated.   + The controller should go into IDLE state after 12 data pieces are extracted.   + The component should start again once the bufferEmpty signal is asserted HIGH again. |

Table

Ⅱ|譬~ 
11111110 
Ⅱ|E夐흐 
IIEEEEQ 
Ⅱ|纛至 
Ⅱ爨爨至 
까까까까까까 
. 수수•챷才才수 
1 

Figure : A screenshot of the Data Extraction Controller testbench.

It can be seen from figure 1 that when 12 DONErx signals are generated, 12 NEXT\_ADD, STORE and EXTRACT signals are generated which results in the component transitioning to the IDLE state. After a delay of 300 clock cycles (or 300), the bufferEmpty signal is raised once again which and DONErx signals are generated which results in the component generating the desired signals once again and thereby proving continuous functionality and meeting all the pass conditions. For a complete set of screenshots for this testbench see Appendix A.

### **Data Transmission Controller**

|  |  |  |  |
| --- | --- | --- | --- |
| Purpose of Test | Test Configuration | Component Configuration | Test Pass Conditions |
| To test if the components functions as intended in normal working conditions. | * + The input clock is generated by the user's code.   + The testing code has been written to follow the stages of the state machine according to which the component operates therefore the timing of the input signals is accurate relative to the timing of the actual component operation.   + The DONEtx signal is asserted high for one clock cycle every 20 clock cycles to imitate the pulse from the SPI master that notifies the controller that transmission of data piece has been completed. This number would signify how long the actual SPI takes to extract the sample but for testing purposes it does not need to be accurate as it does not affect the actual operation of the component.   + The bufferEmpty signal is de-asserted at the beginning to notify the controller that the buffer is full. After 12 repetitions of the DONEtx signal, the bufferEmpty signal is asserted HIGH after a brief 5 clock cycle delay.   + There is a 300 clock cycle delay in between the bufferEmpty signal assertions.   + There is a brief 5 clock cycle delay between the R2S signal and the sTx signal to imitate the short latency that occurs when transmitting signals between components. This is not an accurate representation of latency but for testing purposes it does not matter.   + Gate level simulation was used to simulate the component operation to obtain realistic response. | * + The communication system is configured to handle 12 pieces of data. Therefore, the component has to send signals 12 times in order to extract 12 pieces of data. | * + The controller should generate 12 NEXT\_DAT and TRANSMIT signals.   + The controller should go into IDLE state after 12 data pieces are transmitted.   + The component should start again once the bufferEmpty signal is asserted LOW again.   + The controller should only start transmitting when it receives the sTx signal as HIGH.   + The R2S output signal should be asserted low once all data is transmitted. |

/destx_vhd I/CLK 
Idestx_vhd _tst/i I/bu fferEmpty 
/destx_vhd I/R2S 
Idestx_vhd _tst/i I/DONEtx 
Idestx_vhd _tst/i I/sTx 
Idestx_vhd _tst/i I/sta te 
Cursor I 
Cursor 2 
Cursor 3 
Msgs 
5001000000 
388500000 
688500000 ps 
2452500000 ps 
4000 0000 s 
388500000 
800000000 
300000000 
688500000 
1200000000 
17640000 
1600000000 
o to ettngs to activate In ows. Table

Figure :

It can be seen that once the bufferEmpty signal is asserted low and the sTx signal is generated, the components beings to send the first TRANSMIT pulse. It then waits for the DONEtx signal to be transmitted before it generates NEXT\_DAT and TRANSMIT signals. The controller does generate 12 TRANSMIT and NEXT\_DAT signals and then asserts the R2S output LOW before transitioning to the IDLE state. When the bufferEmpty signal is asserted low once again, the entire process repeats without any problems. For a complete set of screenshots for this testbench see Appendix B.

### **FIFO Buffer**

|  |  |  |  |
| --- | --- | --- | --- |
| Purpose of Test | Test Configuration | Component Configuration | Test Pass Conditions |
| To test if the components functions as intended in normal working conditions. | * + The input clock is generated by the user's code.   + The testing code has been written to follow the stages of the state machine according to which the component operates therefore the timing of the input signals is accurate relative to the timing of the actual component operation.   + The storeData and sendData signals are asserted HIGH for one clock cycle every 20 clock cycles to imitate the pulses generated by the transmission and the extraction controllers that command the buffer to either send or store data during the extraction or transmission process. This number would signify how long the actual SPI takes to extract the sample and overall latency between signal transmission but for testing purposes it does not need to be accurate as it does not affect the actual operation of the component.   + There is a 300 clock cycle delay between the data extraction and data transmission process. This was done to imitate the delay between the data being ready for transmission and the accessControl block granting access to the specific IMU\_Extraction\_Block. At this point in time the actual delay has not been measured but is irrelevant in the functional test as it does not affect the operation of the component.   + There is a short 5 clock cycle delay between the end of the transmission process and the extraction process to imitate the delay between the end and the start of the two processes. The exact delay has not yet been measured but is irrelevant for functional testing as it does not affect the operation of the component.   + On each iteration of the storing process a different number is generated to be stored. This is done to be able to distinguish whether the data is being stored and asserted on the output correctly during the storing and transmission process.   + Gate level simulation was used to simulate the component operation to obtain realistic response. | * + The communication system is configured to handle 12 pieces of data. Therefore, the component has to send signals 12 times in order to extract 12 pieces of data.   + There will be two configurations to this component. The first configuration does not append the identification byte to each sample stored (like the FIFO buffer usually would). The second configuration will append the identification byte to each sample stored.   + Data width is 16-bits as this is the data width that will be used by default. | * + The FIFO buffer will assert the empty signal LOW when filled with 12 samples and HIGH when commanded to assert the 12 stored samples on the output.   + The FIFO buffer will output the data in a ‘first in, first out’ fashion.   + The FIFO buffer will append Identification bytes to the data correctly. |

Table

Wave - Default 
fifobuffer _ 
vhd tstf11/CLK 
fifobuffer _vhd_tstfi I/empty 
fifobuffer _vhd_tstflI/sendData 
Ifi fobu Fer _vhd _tst/i Ilda taOut 
Ifi fobu Fer _vhd _tst,fi l/sta te 
Ifi fobu Fer _ 
vhd_tstfllms 
Ifi fobu Fer _ 
vhd_tstf11Fs 
vhd_tstfll/os 
Non 
Msgs 
5001000000 
8 
8 
10 
10 
11 
12 
10 
2400000000 
2600000000 
2800000000 

Figure

Nllllhllillllllß 
급까까까까까까까까급 
. 수수 . 구才구才수수수수 

Figure

It can be seen from figure 3 that once 12 sendData input signals have been generated, the buffer outputs the stored numbers (1 to 12) on its output in a ‘first in, first out’ fashion and then asserts the empty output high. Then when 12 storeData input signals have been generated, the buffer stores the numbers from 1 to 12 and asserting the empty output low once again. Figure 4 shows the data on the dataOut output during the transmission process. It can be seen that the identification bytes that the identification bytes are being appended correctly and in the correct order **(use appendix n and appendix n to see order)**. For a complete set of screenshots for this testbench see Appendix C.

### **Address Loader**

|  |  |  |  |
| --- | --- | --- | --- |
| Purpose of Test | Test Configuration | Component Configuration | Test Pass Conditions |
| To test if the components functions as intended in normal working conditions. | * + The input clock is generated by the user's code.   + The NXTADD signal is asserted high for one clock cycle every 20 clock cycles to imitate the pulse from the extraction controller that commands the addressBlock the output the next address. The delay signifies the time between the completion of the transaction of the SPI and the next NXTADD signal. It is not accurate but for the testing purposes it is irrelevant as it does not change the operation of the component.   + Once 12 NXTADD signals are sent there is a 300 clock cycle delay between the next batch of NXTADD signals. | * + The communication system is configured to handle 12 pieces of data. Therefore, the component outputs 12 different addresses. | * + The addressBlock is to output 12 addresses in the right order. |

Table

Machine generated alternative text:
fad d r e ssblock_vhd _tst/ADDRESS 
'addr essblock_vhd _tst/CLK 
'addr essblock_vhd _tst/NXTADD 
Non 
Cursor I 
110... 
00 ns 
314 ns 
0000 0000000 000 
80000 
1011 
110000 
00 
10 
1110000 00000 
0000000 
160000 
10111 1000000 00 
000000 000000 
110000 
1000000 0 
1100 
1000000 000 
280000 
001010 
00000 
110001 
0000000 
11000 
110000 
00 
120000 
200000 
240000 
320000 
0100000 00000 
360000 

Figure

The testbench screenshot in figure 5 depicts the addresses that are output when requested. It can be seen that the addresses are output in the correct order when compared to the table in Appendix n (This is appendix A in Design – Comms network). For a complete set of screenshots for this testbench see Appendix D.

### **FPGA-STM32 SPI Access Controller**

|  |  |  |  |
| --- | --- | --- | --- |
| Purpose of Test | Test Configuration | Component Configuration | Test Pass Conditions |
| To test if the component functions as intended in normal working conditions. | * + The input clock is generated by the user's code.   + The testing code has been written to follow the stages of the state machine according to which the component operates therefore the timing of the input signals is accurate relative to the timing of the actual component operation.   + The r2sIMU signals are configured such that one goes LOW while another one goes HIGH after a 300 clock cycle delay.   + There is a 5 clock cycle delay between the r2sIMU signal going HIGH and the first TRANSMIT signal to give the component time to generate the strtTx(n).   + The txIMU(n) signal is asserted high for one clock cycle every 20 clock cycles to imitate the delay between the SPI transaction and the next pulse commanding it to start another transaction.   + 12 samples are sent ranging from 1 to 12 before the r2sIMU(n) signal is asserted LOW for consistency.   + Even though up to 6 IMUs can be connected to this component, signals are generated for only to lines. This number is enough to prove that this component can manage several IMU extraction blocks. | * + The data is set to 16 bits as this is the width that will be used by default. | * + The TRANSMIT output has to generate 12 output pulses as 12 pieces of data will be sent.   + The strtTx has to generate a pulse on the correct line. If r2sIMU(1) signals that it is ready to send data, strtTx has to generate a pulse on the strtTx(1) line to grant access.   + Data has to be asserted on the dataOut output in the same configuration it has been entered in. |

# **Appendix A**

A complete set of Data Extraction Controller testbench screenshots.Machine generated alternative text:
Wave - Default 
/desrx_ 
vhd tstf11/CLK 
e sr x_vhd _tst/i I/bu fferEmpty 
/desrx_ 
vhd_tst/i 
/desrx_ 
vhd _tst,'i 1mEXT_ADD 
/desrx_ 
vhd_tstf11/STORE 
/desrx_ 
vhd_tstf11/DONErx 
Idesr x_vhd _tst/i I/sta te 
Msgs 

Machine generated alternative text:
Wave - Default 
/desrx_ 
vhd tstf11/CLK 
Ide sr x_vhd _tst/i I/bu fferEmpty 
Idesrx_ 
vhd_tst/i 
Idesrx_ 
vhd _tst,'i 1mEXT_ADD 
Idesrx_ 
vhd tstf11/STORE 
/desrx_ 
vhd_tstf11/DONEr x 
Idesr x_vhd _tst/i I/sta te 
tate 
NXT 
STRTrx 

Machine generated alternative text:
Wave - Default 
/desrx_ 
vhd tstf11/CLK 
Ide sr x_vhd _tst/i I/bu fferEmpty 
Idesrx_ 
vhd_tst/i 
Idesrx_ 
vhd _tst,'i 1mEXT_ADD 
Idesrx_ 
vhd_tstf11/STORE 
/desrx_ 
vhd_tstf11/DONErx 
Idesr x_vhd _tst/i I/sta te 
Msgs 

Machine generated alternative text:
Wave - Default 
/desrx_vhd _tst/i I/CLK 
Idesrx_vhd _tst/i I/bu fferEmpty 
Idesrx_vhd _tst/i I/STORE 
Idesrx_vhd _tst/i IIDONEr x 
Idesr x_vhd _tst/i I/sta te 
Msgs 

# **Appendix B**

Wave - Default 
Msgs 
ii•iii" - 
/destx_vhd _ts t/ I/CLK 
Idestx_vhd _tst/i I/bu fferEmpty 
Idestx_vhd I/R2S 
Idestx_vhd _tst/i I/DONEtx 
Idestx_vhd _tst/i I/sTx 
Idestx_vhd _tst/i I/sta te A complete set of Data Transmission Controller testbench screenshots.

Machine generated alternative text:
Wave - Default 
/destx_vhd tstflI/CLK 
Ide stx_vhd _tst/i I/bu fferEmpty 
Idestx_vhd 
_tst/i 
/destx_vhd 
_tst,'i 1/DONEtx 
Idestx_vhd 
_tstf11mEXT_DAT 
Idestx_vhd 
_tst,'i I/TRANSMIT 
e stx_vhd _tst/i I/sTx 
Idestx_vhd _tst/i I/sta te 
Msgs 
WAITINGtx 

Machine generated alternative text:
Wave - Default 
/destx_vhd _tst/i I/CLK 
Ide stx_vhd _tst/i I/bu fferEmpty 
Idestx_vhd _tst/i I/R2S 
Ide stx_vhd _tst/i IIDONEtx 
Ide stx_vhd _tst/i I/sTx 
Idestx_vhd _tst/i I/sta te 
WAITINGtx 

Wave -Default 
Idestx_vhd _tst/i I/CLK 
Idestx_vhd _tst/i I/bu fferEmpty 
/destx_vhd 
_tst/i 
Idestx_vhd 
_tst,'i 1/DONEtx 
Idestx_vhd 
_tstf11/NEXT_DAT 
/destx_vhd 
_tst,'i I/TRANSMIT 
Idestx_vhd _tst/i I/sTx 
Idestx_vhd _tst/i I/sta te 
Msgs 
WAITINGtx 

# **Appendix C**

Wave -Default 
Ifi fobu Fer _ 
vhd tstf11/CLK 
fifobuffer _vhd _tstfi I/empty 
Ifi fobu Fer _vhd _tst,fi l/stor eDa ta 
fifobuffer _vhd_tstflI/sendData 
Ifi fobu Fer _vhd _tst,fi l/sta te 
Ifi fobu Fer _ 
vhd_tstfllms 
Ifi fobu Fer _ 
vhd_tstf11Fs 
vhd_tstfll/os 
Msgs 
5001000000 
28ûûûûûûûû s A complete set of FIFO Buffer testbench screenshots.

Wave - Default 
fifobuffer _ 
vhd tstf11/CLK 
fifobuffer _vhd_tstfi I/empty 
fifobuffer _vhd_tstflI/sendData 
Ifi fobu Fer _vhd _tst/i Ilda taOut 
Ifi fobu Fer _vhd _tst,fi l/sta te 
Ifi fobu Fer _ 
vhd_tstfllms 
Ifi fobu Fer _ 
vhd_tstf11Fs 
vhd_tstfll/os 
Non 
Msgs 
5001000000 
8 
8 
10 
10 
11 
12 
10 
2400000000 
2600000000 
2800000000 

Wave - Default 
/ffobuffer _ 
vhd tstf11/CLK 
Iffobuffer _vhd_tst/i I/empty 
/ffobuffer _vhd_tstflI/sendData 
If fobu ffer _vhd _tst/i I'd a taOu t 
fobu ffer _vhd _tst/i I/sta te 
fobu ffer _ 
vhd tstfllms 
fobu ffer _ 
vhd_tstf11/Fs 
_vhd 
_tst/i I/os 
Msgs 
5001000000 ps 
2448000000 s 
2452000000 s 
2456000000 s 
2460000000 s 

1 Iffobuffer_vhd_tstf11/dataOut 
• /ffobuffer tstfll/dataln 
/ffobuffer vhd_tstf11/sendData 0 
/ffobuffer vhd_tstf11/storeData O 
fi 。 bu 乛 hd till LK 
Iffobuffer_vhd_tstfll/empty 
vhd_tstf11/OS 
Iffobuffer_vhd_tstf11/FS 
Iffobuffer_vhd_tstf11/NS 
Iffobuffer_vhd_tstfll/state 
5001000000 ps 
2740000000 S 
2744000000 S 
2748000000 S 
2752000000 S 
2756000000 S 

Nllllhllillllllß 
급까까까까까까까까급 
. 수수 . 구才구才수수수수 

IllWlIIilIlilIA 
,IIIIWIWIIII\Ä 
,IIIIIIIIIIIIIO 
까까까까까까까까까까 
. 수수 . 구才구才수수수수 

# **Appendix D**

A complete set of Address Loader testbench screenshots.

Machine generated alternative text:
lad d r essblock_vhd _tst/ADDRESS 
'addr essblock_vhd _tst/CLK 
'addressblock_vhd _tst/NXTADD 
Non 
Cursor I 
Msgs 
110010000000000 
5001000 
519554.814ns 
110010 0.. 
1200000 
1100100 0000000 
1600000 
10010000 000000 
2000000 
2400000 
01000000 0000 
2800000 

Machine generated alternative text:
fad d r e ssblock_vhd _tst/ADDRESS 
'addr essblock_vhd _tst/CLK 
'addr essblock_vhd _tst/NXTADD 
Non 
Cursor I 
110... 
00 ns 
314 ns 
0000 0000000 000 
80000 
1011 
110000 
00 
10 
1110000 00000 
0000000 
160000 
10111 1000000 00 
000000 000000 
110000 
1000000 0 
1100 
1000000 000 
280000 
001010 
00000 
110001 
0000000 
11000 
110000 
00 
120000 
200000 
240000 
320000 
0100000 00000 
360000 

Machine generated alternative text:
'addr essblock_vhd _tst/ADDRESS 
'addr essblock_vhd _tst/CLK 
'addressblock_vhd _tst/NXTADD 
Novv 
Cursor I 
Msgs 
110... 
00 ns 
314 ns 
0100000 
00000 
101110 
000000 
1011 
1000000 
000 
10 
1110100 
00000 
0000000 
1920000 
10111 
1100000 00 
0000000 0000 
1960000 
100001 
0000000 
11000 
0000000 0 
110 01010000 000 
001100 000000 
110001 
000000 
2080000 
1100 
000000 
00 
1840000 
1880000 
2000000 
2040000 