COEN 6541 Project 1 Report

Winter 2021

Supervised by Dr. Otmane Ait Mohamed

Group 2:

Divyaa Mahalakshmi Guruswamy [40167923]

Chirag Jhamb [40169876]

Abdul Rahman Koleilat [40086025]

Table of Contents

1. Introduction

We are given a calculator design that can perform several operations like addition, subtraction, left shift, and right shift. The calculator has four input ports and can take four operation requests at the same time with equal priority for all ports. After each operation is done, the calculator will give an output response and the result for the corresponding port. However, this design has several bugs and to detect those bugs, we are required to do direct testing by writing a testbench that includes the appropriate test cases that test each operation with specific and appropriate input values for the commands and the operands.

2. Test Plan

Specification/Feature	Reference	Test Points	Test Scenarios	Expected Results
Operations(in all the 4 ports)				
Add				
Basic addition operation	1.1, 1.2	To check the basic addition operation without overflow	The input command is given as ‘0001’b to the req(x)_cmd_in bus and the operand1 value is provided to the req(x)_data_in input bus in the first cycle. In the second cycle, the operand2 value is given to the req(x)_data_in input bus and input command is given as no operation. In this case, the output computation value does not exceed the maximum output value range.	The out_res(x) is '01'b implying a successful response. The addition of operand1 and operand2 takes place and the output value is obtained in the out_data(x) bus.

addition operaration resulting in a overflow output value	1.1,1.3	To verify the addition operation with overflow	In the first cycle, The operand1 value and input command of '0001'b is given to the req(x)_data_in and req(x)_cmd_in bus, respectively. The operand2 input value is given to the req(x)_data_in in the second cycle. In this case, the output computation value exceeds the maximum output value range.	The out_res(x) is '10'b implying a overflow response.
Data dependent corner case with overflow	2.4.1	To verify the output of adding two numbers that results in a overflow by 1	In the first cycle, The req(x)_data_in is driven with a input value of “FFFFFFFF" as operand1 and req(x)_cmd_in is provided with the value of '0001'b for addition operation. In the second cycle, The req(x)_data_in is driven with a input value of 1 with no operation as input command.	The out_res(x) is '10'b implying a overflow response.
Data dependent corner case without overflow	2.4.2	To verify the output of adding two numbers that result in a output sum value of “FFFFFFFF"	The req(x)_cmd_in bus receives a value of ‘0001’b for addition operation. The operand1 and operand2 values are provided in such a way that	The out_res(x) is '01'b implying a successful response. The addition of operand1 and operand2 takes place and the output value of “FFFFFFFF" is

			the output sum is “FFFFFFFF".	obtained in the out_data(x) bus.
adding 2 maximum numbers		To check the additionesult when 2 operands have maximum value	the input command is addition and both the operands values are “FFFFFFFF"	The out_res(x) is '10'b implying a overflow response.
Adding max number with min number	2.4.2	To check the addition result when 1 operand has maximum value and other has minimum value	the input command is addition. The operand1 takes a value of “FFFFFFFF" and operand2 takes a value of "00000000" in the req(x)_data_in port, in their respective cycle.	The out_res(x) is '01'b implying a successful response. The addition of operand1 and operand2 takes place and the output value of “FFFFFFFF" is obtained in the out_data(x) bus.
adding 2 minimum numbers		To check the addition result when 2 operands have minimum values	the input command is addition and both the operands values are “00000000"	The out_res(x) is a success and the output result is "00000000"
Subtract				
Basic subtraction operation	1.1, 1.2	To check the basic subtraction operation without overflow	The input command is given as ‘0010’b to the req(x)_cmd_in bus and the operand1 value is provided to the req(x)_data_in input bus in the first cycle. In the second cycle, the operand2 value is given to the req(x)_data_in input bus. In this case, the value of	The out_res(x) is '01'b implying a successful response. The subtraction of operand2 from operand1 takes place and the output value is obtained in the out_data(x) bus.

			operand2 does not exceed the operand 1 value.	
Subtraction operaration resulting in a underflow output value	1.1, 1.3	To verify the subtraction operation with underflow	In the first cycle, The operand1 value and input command of '0010'b is given to the req(x)_data_in and req(x)_cmd_in bus, respectively. The operand2 input value is given to the req(x)_data_in bus in the second cycle. In this case, the operand2 value is more than operand1 value.	The out_res(x) is '10'b implying a underflow response.
Data dependent corner case without underflow	2.4.3	To verify the output of subtracting two equal numbers	The req(x)_cmd_in bus receives a value of '0010'b for subtraction operation. A equal value is provided to operand1 and operand2.	The out_res(x) is '01'b implying a successful response. The subtraction of operand2 from operand1 takes place and the output value is obtained as "00000000" in the out_data(x) bus.
Data dependent corner case with underflow	2.4.4	To verify the output of adding two numbers that results in a underflow by 1	The req(x)_cmd_in is driven with a input value of '0010'b for subtraction operation and req(x)_data_in is provided with the	The out_res(x) is '10'b implying a underflow response.

			operand 1 value in the first cycle. In the second cycle, The operand 2 value is given to the req(x)_data_in bus in such a way that it exceeds the operand1 value by 1 .	
subtraction of 2 minimum numbers		To check the subtraction result when 2 operands have minimum values	the input command is subtraction and both the operand values are “00000000"	The out_res(x) is a success and the output result is "00000000"
subtraction of 2 maximum numbers		To check the subtraction result when 2 operands have maximum values	the input command is subtraction and both the operand values are “FFFFFFFF"	The out_res(x) is a success and the output result is "00000000"
Subtracting a max number from a min number	2.4.2	To check the addition result when operand1 has minimum value and operand 2 has a maximum value and other has	the input command is subtraction(value of ‘0010’b) . The operand1 takes a value of "00000000" and operand2 takes a value of “FFFFFFFF" in the req(x)_data_in port, in their respective cycle.	The out_res(x) is '10'b implying a underflow response as operand2 is greater than operand1.
Shift left				
Basic shift operation	1.1, 1.2	To check the basic left shift operation	The req(x)_cmd_in bus is driven with a value of ‘0101' b for left shift operation and the operand1 value is provided at the req(x)_data_in input bus in the	The out_res(x) is '01'b implying a successful response. The out_data(x) has the result of operand1 left shifted by the number of bits

			first cycle. In the second cycle, the operand2 value is given to the req(x)_data_in input bus. In this case, the value of operand2 does not exceed the decimal value of 31.	mentioned in by operand2 value.
Data dependent corner case shifting 0 places	2.4.5	To verify the result when operand1 is left shifted by 0 places	The input command is given as '0101'b to the req(x)_cmd_in bus for left shift operation and the operand1 value is provided to the req(x)_data_in input bus in the first cycle. In the second cycle, req(x)_cmd_in receives a no operation command and the req(x)_data_in input bus receives a value of "00000000".	The out_res(x) is a successful response indicated by the value '01'b . The out_data(x) result is same as operand1 as it is left shifted by 0 bits.
Data dependent corner case shifting 31 places	2.4.6	To verify the result when operand1 is left shifted by 31 places	The input command is given as '0101'b to the req(x)_cmd_in bus for left shift operation and the operand1 value is provided to the req(x)_data_in input bus in the first cycle. In the second cycle, req(x)_data_in input bus receives a value of "0000001F".	The out_res(x) is a successful response indicated by the value '01'b . The out_data(x) results in "00000000" as the operand1 is left shifted by maximum allowable shifting places i.e 31 bits.

Ignoring high order 27 bits in operand2	2.3	To ensure that the MSB 27 bits are not considered in operand2 for left shifting	The req(x)_cmd_in is given as '0101'b for the left shift operation and the operand1 value is given in the req(x)_data_in port.The operand2 is driven with input value greater than 31(decimal). In this case, few of the high order 27 bits are also asserted with 1.	The out_res(x) is a successful response indicated by the value '01'b . In the result, the operand is only left shifted by the decimal value obtained from the lower 5 bits of operand2, rest of the higher bits are ignored.
Shift Right				
Basic shift operation	1.1, 1.2	To check the basic right shift operation	The req(x)_cmd_in bus is driven with a value of '0110' b for right shift operation and the operand1 value is provided at the req(x)_data_in input bus in the first cycle. In the second cycle, the operand2 value is given to the req(x)_data_in input bus. In this case, the value of operand2 does not exceed the decimal value of 31.	The out_res(x) is '01'b implying a successful response. The out_data(x) has the result of operand1 right shifted by the number of bits mentioned in by operand2 value.

Data dependent corner case shifting 0 places	2.4.5	To verify the result when operand1 is right shifted by 0 places	The input command is given as '0110' b to the req(x)_cmd_in bus for right shift operation and the operand1 value is provided to the req(x)_data_in input bus in the first cycle. In the second cycle, req(x)_data_in input bus receives a value of "00000000".	The out_res(x) is a successful response indicated by the value '01'b . The out_data(x) result is same as operand1 as it is right shifted by 0 bits.
Data dependent corner case shifting 31 places	2.4.6	To verify the result when operand1 is right shifted by 31 places	The input command is given as '0110'b to the req(x)_cmd_in bus for right shift operation and the operand1 value is provided to the req(x)_data_in input bus in the first cycle. In the second cycle, req(x)_data_in input bus receives a value of "0000001F".	The out_res(x) is a successful response indicated by the value '01'b . The out_data(x) results in "00000000" as the operand1 is right shifted by maximum allowable shifting places i.e 31 bits.
Ignoring high order 27 bits in operand2	2.3	To ensure that the MSB 27 bits are not considered in operand2 for right shifting	The req(x)_cmd_in is given as '0110'b for the right shifting operation and the operand1 value is given in the req(x)_data_in port.The operand2 is driven with input value greater than 31(decimal). In this case, few of the high order 27 bits are also asserted with 1.	The out_res(x) is a successful response indicated by the value '01'b . In the result, the operand is only right shifted by the decimal value obtained from the lower 5 bits of operand2, rest of the higher bits are ignored.

Dirty state				
addition followed by subtraction	2.1.1	To verify the result when a addition command is followed by a subtraction command	In the first cycle, the Input commad is given as addition and the operand1 value is provided to the req(x)_data_in port. In the second cycle, the input command is given as subtraction instead of no operation and operand2 value is provided to the data in port.	The out_res(x) should be a successful response and the result output data should be the addition of the two operands. The command given in the second cycle should not affect the command given in the first cycle.
Shift left followed by shift right	2.1.1	To verify the result when a shift left command is followed by a shift right command	In the first cycle, the Input commad is given as shift left and the operand1 value is provided to the req(x)_data_in port. In the second cycle, the input command is given as shift right instead of no operation and operand2 value is provided to the data in port.	The out_res(x) should be a successful response and the result output data should be the shifted left by the number of bits mentioned by the operand2. The command given in the second cycle should not affect the command given in the first cycle.
Across all ports, Addition followed by subtraction	2.1.2	To verify the result in all the four ports when a addition command is followed by a subtraction command	the input command of addition and input data as operand1 is given to all the ports concurrently, in the first cycle. The command is subtraction instead of no operation and operand2 data is provided in the second cycle.	The output response should be a success in all the ports and the addition of both the operands should be obtained at the output data result. The second cycle command does not affect the operation

Across all ports, Shift left followed by shift right	2.1.2	To verify the result in all the four ports when a addition command is followed by a subtraction command	the input command of shift left and input data as operand1 is given to all the ports concurrently, in the first cycle. In the second cycle, the input command is given as shift right instead of no operation and operand2 values is provided to the data in ports.	The out_res of all the ports should be a success. The result output data in all the port should be the shifted left by the number of bits mentioned by the operand2 values. The command given in the second cycle should not affect the command given in the first cycle.
Equal Port Priority	2.2	To check the priority when all the ports are given with the same command	The required operation is given in the input command to all the ports. And the operand1 and operand2 are values are given in the req_data_in bus at first and second cycle, respectivly.	The corresponding output response is obtained in the out_res bus and the evaluated out is obtained in the out-data of each port. All the ports are given equal priority.
invalid data	2.5	To check the response when invalid data is provided	One or both of the input command is given as invalid data	The data which is invalid is ignored.
invalid commands	3.1	To check when invalid commands are given in input	The input command is not given from the command table for operation.	The output response '10' showing that it is an invalid command.
Output response when operand is not mentioned	3.2	To check when the operand is not given	The input command in both the cycles is 4'b 0000	there should not be any output response produced

Reset	3.3	To check the reset function	Drive the reset input (0:7) but setting it to '11111111'b and hold it for seven cycles	All the input busses except the the clk is set to zero

3. Test Cases

Before starting the test cases, we initially set the values for the local parameters like (No_Op, Add, Sub, Shift_Left, Shift_Right No_Response , Success etc). Then, the design is set to the reset state for the first seven clock cycles and during this reset phase, all the other input buses are set to 0.

The following test cases are executed on all the ports and the expected outputs are mentioned:

- For the ADD operation, the input command is given as 0001’b and the output response obtained is 01’b which tells us that the operation has been successful and the output obtained after the addition of the two operands does not exceed the maximum limit.
- There is another ADD operation case that leads to an overflow by giving 10’b as the output response. In this case , the input command (0001’b) and the first input are given in the first cycle while the second input is given in the second cycle. When we give the input command as 0001’b and the first input value as “FFFFFFFF” in the first cycle and as ‘1’ in the second cycle. The output response is expected as 10’b telling us that there has been an overflow as this addition has led to an overflow by 1.
- In another ADD operation the inputs with the command as 0001’b are given in such a way that the output sum is “FFFFFFFF”, The output response obtained is 01’b telling us that the execution has been successful.
- In a basic subtraction operation we give the command a value of 0010’b and the first input value is provided to the bus in the first cycle while the second input value is provided in the second cycle , keeping in mind that the second input operand should never exceed the value of the first input operand. We should get the output response as 01’b which confirms that the implementation has been successful and the output value after the subtraction is given in the output data bus.
- In a SUBTRACTION UNDERFLOW case, we operate in the same way, giving the command as 0010’b and the first input in the first cycle and the second input in the next clock cycle. The only difference for this case is that the value of the second input, in this case, is greater than the value of the first input, and the output response obtained should be 10’b telling us that it’s an underflow condition.
- When we Subtract two equal numbers, we should obtain the output as “00000000” in the output data bus and a success response.
- In another case when we give the command as 0010’b and the first input in the first clock cycle and the second input in the second clock cycle in such a way that the second input exceeds the first input by a value of 1. This case represents an underflow condition by a value of 1. We should get an output response as 10’b confirming the execution of an underflow condition.

- For a basic shift operation, we give the command as 0101'b, The first input is given in the first clock cycle while the second input is given in the second cycle in such a way that the value of the second input is not more than the decimal value of 31. The output response obtained should be 01'b which tells us that the execution was successful and in the output, we get the result of input1 shifted by the number of bits in the second input. When we provide the second input as "00000000" in the second cycle what we should get the output value as the first input because it is shifted by 0 bits and therefore it's the same value.
- If we want to shift the input1 given in the first cycle by 31 bits then we provide the command as 0101'b and in the second cycle we give the second input value as "0000001F". In the output, we should get "00000000" because the first input is shifted by 31 bits and we also must get the output response as 01'b confirming the correct execution of the test case.
- In another case when we give the second input in the second cycle a value greater than 31, a few high order 27 bits are given a value 1. In this case, the obtained input1 should be shifted to the left only by the lower 5 bits of the second input, and all the other bits are discarded.
- To check the basic shift right operation we give the command value of "0110" and the first input is given in the first cycle while the second input is given in the second cycle and this input is not greater than 31. The output input1 should be shifted towards the right by the number of bits given in input2 and the output response value "01" tells us that the execution has been successful.
- To shift the first input towards the right by 0 places we follow the same process as we have done for shifting them to 0 places towards the left by giving the value of the second input as "00000000". We should get an output which is input1 as there has been no shifting.
- For shifting the input 31 places towards the right we follow the same procedure the only difference is in the command, In this case, we give the command as '0110' and we provide the first input in the first clock cycle and the second input in the second clock cycle as "0000001F" and when we check the output we should observe that we are getting "00000000" because the first input has been shifted 31 bits towards the right.
- In another case, we use the command as '0110' and give the first input in the first cycle and the second input in the second cycle with a value which is greater than 31 and we get the output response as 01'b telling us that the execution has been successful. When we check the output, the first input should only be shifted by the lowest 5 bits of the second input and all the other higher bits are ignored or discarded.
- When we give the input and command in the first cycle and the second cycle we give the command as subtraction and provide the second input. In the output, we should get the sum of the two inputs, and the second command given in the second cycle does not affect the case.
- Similarly, when we give the command as shift left in the first cycle and shift right in the second cycle, The output obtained should be the first input shifted left by the number of bits in the second input and the command given in the second cycle (shift right) is completely ignored.
- Another case is when we give the addition command to all the ports in the first cycle and then give the subtraction command in the second cycle to all the ports, In the output, we would get the sum of the first input and the second input in all the ports and the subtraction command is ignored for all of them.
- Similarly when we give the shift left command in the first cycle for all the ports and give shift right command in the second cycle for all the ports. In the output, we should get the first input shifted to the left by the number of bits mentioned in the second input, and the shift right command is completely ignored.

- There are few cases, where the priority of the operations are verified. It is expected for all the ports to have a fair priority.
- When no operand is mentioned, the output response should be no response and should not produce a superfluous outputs.
- In the cases, where invalid datas are provided to the operands. The data should be ignored. If illegal command is given the output response should be 2'b 01 implying that the command is invalid.

4. Bugs detected in the DUT

The above test cases were performed in the DUT to find the errors in the design under test (DUT). The below are the bugs detected and observation made using the direct test method:

- The basic addition and subtraction operation in Port4 gives an output response as no response.
- In the overflow test case, ports 1,2, and 3 produce an "xx" state, and port 4 gives a no response output instead of giving an overflow response in all the out_resp bus.
- In the underflow case, Ports 1,3 and 4 doesn't produce a correct response.
- In the dirty stage, in which addition is followed by a subtraction command. In this case the second subtraction command should not influence the adding of 2 operands. The port1 gives a success response, but the output data is not equal to the expected output data. In other ports, there is no output response which is an error and at the same time port1 produces a success response, when should not have any output response.
- In the dirty state of shifting left followed by shifting right, the left shift command should not be affected. The following is observed in the results-
  - In port1,2 and 3 the output data is not equal to expected data, but the response of success is obtained.
  - Port4 does not produce an out response.
  - Port1 gives a success response when the other ports are used, which is an error.
- In the dirty state across all ports when we do (addition followed by a subtraction) or Shift left followed by a shift right, there is an error at port 1 and the output is not equal to the expected output. The other ports do not give a response.
- For an addition priority, port1 produces the first result followed by port2 but we encounter an error as the output is not the same as the expected output.
- For an subtraction priority, port3 produces the first correct output.
- For shift operations, port1 produces the first correct outputs.
- While simultaneous executing all operation, all operations are getting executed the add and subtract output is obtained first. Later the shift operation outputs are obtained from the ports 3 and 4.
- When we subtract a number that underflows by 1, we get an error at port 1,2 and 3 telling us that there is an underflow, but the response is successful.
- When we shift the first input to the left or to the right by 0 places we get an error in port 1,2,3 and 4 as we did not get the expected output.
- When we give an invalid input we get an error in port 2 and 3 as our output does not match the expected output.

- In the data dependent corner cases which results in an overflow or underflow, gives a wrong response. But the cases which do not have overflow or underflow, the result of response and data is correct.
- For the illegal input command ‘0011’, ‘0100’, ‘0111’, ’1000’, ‘1001’, 1010’,’1011’, ‘1100’, ‘1101’, ‘1110’, ‘1111’, Port 1,2,3 and 4 gives us an error for an illegal command but the response was successful.

5. Conclusion

The direct testing method has been useful in detecting the bugs that were present in the design of the calculator. However, it is a very time-consuming process as we have to manually write all the possible test cases on our own and you have to think about every possible scenario as well. Therefore, direct testing isn’t quite a practical method and can’t be used for larger designs which is why random testing is much more preffered.

6. Appendix

6.1. Command Line Output

In showing the output in transcript of the Questasim. The output display of the previous of test case is shown before the input command and operand values of the current test case. (for test cases)

Compile of calc1_tb (7).sv was successful.

vsim -novopt work.calc1_tb

vsim -novopt work.calc1_tb

Refreshing /nfs/home/d/d_gurusw/COEN6541/cal/work.calc1_tb

Loading sv_std.std

Loading work.calc1_tb

add wave -position insertpoint sim:/calc1_tb/*

run -all

Add (Port 1)

time = 850 Port 1: Input Command = 0001, Operand 1 = 00000001, Operand 2 = 00000001

Add (Port 2)

time = 1000 At port 1 Input Command = 0001: Correct: Output (00000002) is equal to Expected Result (00000002)

time = 1050 Port 2: Input Command = 0001, Operand 1 = 00000001, Operand 2 = 00000001

Add (Port 3)

time = 1200 At port 2 Input Command = 0001: Correct: Output (00000002) is equal to Expected Result (00000002)

time = 1250 Port 3: Input Command = 0001, Operand 1 = 00000001, Operand 2 = 00000001

Add (Port 4)

time = 1400 At port 3 Input Command = 0001: Correct: Output (00000002) is equal to Expected Result (00000002)

time = 1450 Port 4: Input Command = 0001, Operand 1 = 00000001, Operand 2 = 00000001

Sub (Port 1)

time = 1650 Port 1: Input Command = 0010, Operand 1 = 00000002, Operand 2 = 00000001

Sub (Port 2)

time = 1800 At port 1 Input Command = 0010: Correct: Output (00000001) is equal to Expected Result (00000001)

time = 1850 Port 2: Input Command = 0010, Operand 1 = 00000002, Operand 2 = 00000001

Sub (Port 3)

time = 2000 At port 2 Input Command = 0010: Correct: Output (00000001) is equal to Expected Result (00000001)

time = 2050 Port 3: Input Command = 0010, Operand 1 = 00000002, Operand 2 = 00000001

Sub (Port 4)

time = 2200 At port 3 Input Command = 0010: Correct: Output (00000001) is equal to Expected Result (00000001)

time = 2250 Port 4: Input Command = 0010, Operand 1 = 00000002, Operand 2 = 00000001

Shift Left (Port 1)

time = 2450 Port 1: Input Command = 0101, Operand 1 = 00000001, Operand 2 = 00000002

Shift Left (Port 2)

time = 2600 At port 1 Input Command = 0101: Correct: Output (00000004) is equal to Expected Result (00000004)

time = 2650 Port 2: Input Command = 0101, Operand 1 = 00000001, Operand 2 = 00000002

Shift Left (Port 3)

time = 2800 At port 2 Input Command = 0101: Correct: Output (00000004) is equal to Expected Result (00000004)

time = 2850 Port 3: Input Command = 0101, Operand 1 = 00000001, Operand 2 = 00000002

Shift Left (Port 4)

time = 3000 At port 3 Input Command = 0101: Correct: Output (00000004) is equal to Expected Result (00000004)

time = 3050 Port 4: Input Command = 0101, Operand 1 = 00000001, Operand 2 = 00000002

Shift Right (Port 1)

time = 3200 At port 4 Input Command = 0101: Correct: Output (00000004) is equal to Expected Result (00000004)

time = 3250 Port 1: Input Command = 0110, Operand 1 = 80000000, Operand 2 = 00000002

Shift Right (Port 2)

time = 3400 At port 1 Input Command = 0110: Correct: Output (20000000) is equal to Expected Result (20000000)

time = 3450 Port 2: Input Command = 0110, Operand 1 = 80000000, Operand 2 = 00000002

Shift Right (Port 3)

time = 3600 At port 2 Input Command = 0110: Correct: Output (20000000) is equal to Expected Result (20000000)

time = 3650 Port 3: Input Command = 0110, Operand 1 = 80000000, Operand 2 = 00000002

Shift Right (Port 4)

time = 3800 At port 3 Input Command = 0110: Correct: Output (20000000) is equal to Expected Result (20000000)

time = 3850 Port 4: Input Command = 0110, Operand 1 = 80000000, Operand 2 = 00000002

Overflow (Port 1)

time = 4000 At port 4 Input Command = 0110: Correct: Output (20000000) is equal to Expected Result (20000000)

time = 4050 Port 1: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = 0000000f

Overflow (Port 2)

time = 4250 Port 2: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = 0000000f

Overflow (Port 3)

time = 4450 Port 3: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = 0000000f

Overflow (Port 4)

time = 4650 Port 4: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = 0000000f

Underflow (Port 1)

time = 4850 Port 1: Input Command = 0010, Operand 1 = 00000000, Operand 2 = 0000000f

Underflow (Port 2)

time = 5050 Port 2: Input Command = 0010, Operand 1 = 00000000, Operand 2 = 0000000f

time = 5100 Input Command = 0010: Error at Port 1: Overflow but the response is Success ('01'b)

Underflow (Port 3)

time = 5250 Port 3: Input Command = 0010, Operand 1 = 00000000, Operand 2 = 0000000f

time = 5300 Input Command = 0010: Error at Port 2: Underflow but the response is Success ('01'b)

Underflow (Port 4)

time = 5450 Port 4: Input Command = 0010, Operand 1 = 00000000, Operand 2 = 0000000f

time = 5500 Input Command = 0010: Error at Port 3: Underflow but the response is Success ('01'b)

Dirty State: Addition followed by Subtraction (Port 1)

time = 5650 Port 1: Input Command = 0001, Operand 1 = 00000fff, Operand 2 = 000f0f0f

Dirty State: Addition followed by Subtraction (Port 2)

time = 5800 Input Command = 0001: Error at Port 1 Output (001e1e1e) is not equal to Expected Result (000f1f0e)

time = 5850 Port 2: Input Command = 0001, Operand 1 = 00000fff, Operand 2 = 000f0f0f

Dirty State: Addition followed by Subtraction (Port 3)

time = 6000 Error at Port 1: out_resp1 should be No Response ('00'b)

time = 6050 Port 3: Input Command = 0001, Operand 1 = 00000fff, Operand 2 = 000f0f0f

Dirty State: Addition followed by Subtraction (Port 4)

time = 6200 Error at Port 1: out_resp1 should be No Response ('00'b)

time = 6250 Port 4: Input Command = 0001, Operand 1 = 00000fff, Operand 2 = 000f0f0f

time = 6400 Error at Port 1: out_resp1 should be No Response ('00'b)

Dirty State: Shift Left followed by Shift Right (Port 1)

time = 6550 Port 1: Input Command = 0101, Operand 1 = 00000fff, Operand 2 = 0000000f

time = 6600 Input Command = 0101: Error at Port 1 Output (00000000) is not equal to Expected Result (000f1f0e)

Dirty State: Shift Left followed by Shift Right (Port 2)

time = 6750 Port 2: Input Command = 0101, Operand 1 = 00000fff, Operand 2 = 0000000f

time = 6800 Input Command = 0101: Error at Port 2: Output (00000000) is not equal to Expected Result (000f1f0e)

Dirty State: Shift Left followed by Shift Right (Port 3)

time = 6900 Error at Port 1: out_resp1 should be No Response ('00'b)

time = 6950 Port 3: Input Command = 0101, Operand 1 = 00000fff, Operand 2 = 0000000f

time = 7000 Input Command = 0101: Error at Port 3: Output (00000000) is not equal to Expected Result (000f1f0e)

Dirty State: Shift Left followed by Shift Right (Port 4)

time = 7100 Error at Port 1: out_resp1 should be No Response ('00'b)

time = 7150 Port 4: Input Command = 0101, Operand 1 = 00000fff, Operand 2 = 0000000f

time = 7300 Error at Port 1: out_resp1 should be No Response ('00'b)

Dirty State: Across all ports, Addition followed by subtraction

time = 7450 All Ports: Input Command = 0001, Operand 1 = 00000fff, Operand 2 = 0000000f

time = 7500 Input Command = 0001: Error at Port 1 Output (00000000) is not equal to Expected Result (07ff8000)

Dirty State: Across all ports, Shift left followed by shift right

time = 7650 All Ports: Input Command = 0101, Operand 1 = 00000fff, Operand 2 = 0000000f

Addition Priority

time = 7800 Input Command = 0001: Error at Port 1 Output (80000000) is not equal to Expected Result (07ff8000)

time = 7850 All Ports: Input Command = 0001, Operand 1 = 00000fff, Operand 2 = 0000000f

time = 8000 Input Command = 0001: Error at Port 1 Output (0000100e) is not equal to Expected Result (07ff8000)

time = 8200 Input Command = 0001: Error at Port 2: Output (0000100e) is not equal to Expected Result (07ff8000)

Subtraction Priority

time = 8350 All Ports: Input Command = 0010, Operand 1 = 00000fff, Operand 2 = 0000000f

time = 8400 At port 3 Input Command = 0010: Correct: Output (0000100e) is equal to Expected Result (0000100e)

Shift left Priority

time = 8550 All Ports: Input Command = 0101, Operand 1 = 00000fff, Operand 2 = 0000000f

time = 8600 At port 1 Input Command = 0101: Correct: Output (00000ff0) is equal to Expected Result (00000ff0)

Shift right Priority

time = 8750 All Ports: Input Command = 0110, Operand 1 = 00000fff, Operand 2 = 0000000f

All operations at the same time

time = 8900 At port 1 Input Command = 0001: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 8950 Port 1: Input Command = 0001, Operand 1 = 00000fff, Operand 2 = 0000000f

time = 8950 Port 2: Input Command = 0010, Operand 1 = 00000fff, Operand 2 = 0000000f

time = 8950 Port 3: Input Command = 0101, Operand 1 = 00000fff, Operand 2 = 0000000f

time = 8950 Port 4: Input Command = 0110, Operand 1 = 00000fff, Operand 2 = 0000000f

time = 9100 At port 1 Input Command = 0001: Correct: Output (0000100e) is equal to Expected Result (0000100e)

time = 9100 At port 3 Input Command = 0101: Correct: Output (07ff8000) is equal to Expected Result (07ff8000)

time = 9300 At port 2 Input Command = 0010: Correct: Output (00000ff0) is equal to Expected Result (00000ff0)

time = 9300 At port 4 Input Command = 0110: Correct: Output (00000000) is equal to Expected Result (00000000)

Check that the high-order 27 bits are ignored in the second operand of shift left command. (Port 1)

time = 9450 Port 1: Input Command = 0101, Operand 1 = ffffffff, Operand 2 = 00ab0fff

time = 9600 At port 1 Input Command = 0101: Correct: Output (80000000) is equal to Expected Result (80000000)

Check that the high-order 27 bits are ignored in the second operand of shift left command. (Port 2)

time = 9750 Port 2: Input Command = 0101, Operand 1 = ffffffff, Operand 2 = 00ab0fff

Check that the high-order 27 bits are ignored in the second operand of shift left command. (Port 3)

time = 9900 At port 2 Input Command = 0101: Correct: Output (80000000) is equal to Expected Result (80000000)

time = 9950 Port 3: Input Command = 0101, Operand 1 = ffffffff, Operand 2 = 00ab0fff

Check that the high-order 27 bits are ignored in the second operand of shift left command. (Port 4)

time = 10100 At port 3 Input Command = 0101: Correct: Output (80000000) is equal to Expected Result (80000000)

time = 10150 Port 4: Input Command = 0101, Operand 1 = ffffffff, Operand 2 = 00ab0fff

Check that the high-order 27 bits are ignored in the second operand of shift right command. (Port 1)

time = 10300 At port 4 Input Command = 0101: Correct: Output (80000000) is equal to Expected Result (80000000)

time = 10350 Port 1: Input Command = 0110, Operand 1 = ffffffff, Operand 2 = 00ab0fff

Check that the high-order 27 bits are ignored in the second operand of shift right command. (Port 2)

time = 10500 At port 1 Input Command = 0110: Correct: Output (00000001) is equal to Expected Result (00000001)

time = 10550 Port 2: Input Command = 0110, Operand 1 = ffffffff, Operand 2 = 00ab0fff

Check that the high-order 27 bits are ignored in the second operand of shift right command. (Port 3)

time = 10700 At port 2 Input Command = 0110: Correct: Output (00000001) is equal to Expected Result (00000001)

time = 10750 Port 3: Input Command = 0110, Operand 1 = ffffffff, Operand 2 = 00ab0fff

Check that the high-order 27 bits are ignored in the second operand of shift right command. (Port 4)

time = 10900 At port 3 Input Command = 0110: Correct: Output (00000001) is equal to Expected Result (00000001)

time = 10950 Port 4: Input Command = 0110, Operand 1 = ffffffff, Operand 2 = 00ab0fff

Data dependent corner case: Add two numbers that overflow by 1 (â œFFFFFFFFâ X + 1). (Port 1)

time = 11100 At port 4 Input Command = 0110: Correct: Output (00000001) is equal to Expected Result (00000001)

time = 11150 Port 1: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = 00000001

Data dependent corner case: Add two numbers that overflow by 1 (â œFFFFFFFFâ X + 1). (Port 2)

time = 11350 Port 2: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = 00000001

Data dependent corner case: Add two numbers that overflow by 1 (â œFFFFFFFFâ X + 1). (Port 3)

time = 11550 Port 3: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = 00000001

Data dependent corner case: Add two numbers that overflow by 1 (â œFFFFFFFFâ X + 1). (Port 4)

time = 11750 Port 4: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = 00000001

Data dependent corner case: Add two numbers whose sum is â œFFFFFFFFâ X. (Port 1)

time = 11950 Port 1: Input Command = 0001, Operand 1 = eeeeffff, Operand 2 = 11110000

Data dependent corner case: Add two numbers whose sum is â œFFFFFFFFâ X. (Port 2)

time = 12100 Input Command = 0001: Error at Port 1: Overflow but the response is Success ('01'b)

time = 12150 Port 2: Input Command = 0001, Operand 1 = eeeeffff, Operand 2 = 11110000

Data dependent corner case: Add two numbers whose sum is â œFFFFFFFFâ X. (Port 3)

time = 12300 At port 2 Input Command = 0001: Correct: Output (ffffffff) is equal to Expected Result (ffffffff)

time = 12350 Port 3: Input Command = 0001, Operand 1 = eeeeffff, Operand 2 = 11110000

Data dependent corner case: Add two numbers whose sum is â œFFFFFFFFâ X. (Port 4)

time = 12500 At port 3 Input Command = 0001: Correct: Output (ffffffff) is equal to Expected Result (ffffffff)

time = 12550 Port 4: Input Command = 0001, Operand 1 = eeeeffff, Operand 2 = 11110000

Data dependent corner case: Subtract two equal numbers. (Port 1)

time = 12750 Port 1: Input Command = 0010, Operand 1 = 0000ffff, Operand 2 = 0000ffff

Data dependent corner case: Subtract two equal numbers. (Port 2)

time = 12900 At port 1 Input Command = 0010: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 12950 Port 2: Input Command = 0010, Operand 1 = 0000ffff, Operand 2 = 0000ffff

Data dependent corner case: Subtract two equal numbers. (Port 3)

time = 13100 At port 2 Input Command = 0010: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 13150 Port 3: Input Command = 0010, Operand 1 = 0000ffff, Operand 2 = 0000ffff

Data dependent corner case: Subtract two equal numbers. (Port 4)

time = 13300 At port 3 Input Command = 0010: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 13350 Port 4: Input Command = 0010, Operand 1 = 0000ffff, Operand 2 = 0000ffff

Data dependent corner case: Subtract a number that underflows by 1 (Operand2 is one greater than Operand1). (Port 1)

time = 13550 Port 1: Input Command = 0010, Operand 1 = 00000000, Operand 2 = 00000001

Data dependent corner case: Subtract a number that underflows by 1 (Operand2 is one greater than Operand1). (Port 2)

time = 13750 Port 2: Input Command = 0010, Operand 1 = 00000000, Operand 2 = 00000001

time = 13800 Input Command = 0010: Error at Port 1: Underflow but the response is Success ('01'b)

Data dependent corner case: Subtract a number that underflows by 1 (Operand2 is one greater than Operand1). (Port 3)

time = 13950 Port 3: Input Command = 0010, Operand 1 = 00000000, Operand 2 = 00000001

time = 14000 Input Command = 0010: Error at Port 2: Underflow but the response is Success ('01'b)

Data dependent corner case: Subtract a number that underflows by 1 (Operand2 is one greater than Operand1). (Port 4)

time = 14150 Port 4: Input Command = 0010, Operand 1 = 00000000, Operand 2 = 00000001

time = 14200 Input Command = 0010: Error at Port 3: Underflow but the response is Success ('01'b)

Data dependent corner case: Shift Left 0 places (should return Operand1 unchanged). (Port 1)

time = 14350 Port 1: Input Command = 0101, Operand 1 = ffffffff, Operand 2 = 00000000

Data dependent corner case: Shift Left 0 places (should return Operand1 unchanged). (Port 2)

time = 14500 Input Command = 0101: Error at Port 1 Output (xxxxxxxx) is not equal to Expected Result (ffffffff)

time = 14550 Port 2: Input Command = 0101, Operand 1 = ffffffff, Operand 2 = 00000000

Data dependent corner case: Shift Left 0 places (should return Operand1 unchanged). (Port 3)

time = 14700 Input Command = 0101: Error at Port 2: Output (xxxxxxxx) is not equal to Expected Result (ffffffff)

time = 14750 Port 3: Input Command = 0101, Operand 1 = ffffffff, Operand 2 = 00000000

Data dependent corner case: Shift Left 0 places (should return Operand1 unchanged). (Port 4)

time = 14900 Input Command = 0101: Error at Port 3: Output (xxxxxxxx) is not equal to Expected Result (ffffffff)

time = 14950 Port 4: Input Command = 0101, Operand 1 = ffffffff, Operand 2 = 00000000

Data dependent corner case: Shift Right 0 places (should return Operand1 unchanged). (Port 1)

time = 15100 Input Command = 0101: Error at Port 4: Output (xxxxxxxx) is not equal to Expected Result (ffffffff)

time = 15150 Port 1: Input Command = 0110, Operand 1 = ffffffff, Operand 2 = 00000000

Data dependent corner case: Shift Right 0 places (should return Operand1 unchanged). (Port 2)

time = 15300 Input Command = 0110: Error at Port 1 Output (00000000) is not equal to Expected Result (ffffffff)

time = 15350 Port 2: Input Command = 0110, Operand 1 = ffffffff, Operand 2 = 00000000

Data dependent corner case: Shift Right 0 places (should return Operand1 unchanged). (Port 3)

time = 15500 Input Command = 0110: Error at Port 2: Output (00000000) is not equal to Expected Result (ffffffff)

time = 15550 Port 3: Input Command = 0110, Operand 1 = ffffffff, Operand 2 = 00000000

Data dependent corner case: Shift Right 0 places (should return Operand1 unchanged). (Port 4)

time = 15700 Input Command = 0110: Error at Port 3: Output (00000000) is not equal to Expected Result (ffffffff)

time = 15750 Port 4: Input Command = 0110, Operand 1 = ffffffff, Operand 2 = 00000000

Data dependent corner case: Shift Left 31 places (the max allowable shift places). (Port 1)

time = 15900 Input Command = 0110: Error at Port 4: Output (00000000) is not equal to Expected Result (ffffffff)

time = 15950 Port 1: Input Command = 0101, Operand 1 = ffffffff, Operand 2 = 0000001f

Data dependent corner case: Shift Left 31 places (the max allowable shift places). (Port 2)

time = 16100 At port 1 Input Command = 0101: Correct: Output (80000000) is equal to Expected Result (80000000)

time = 16150 Port 2: Input Command = 0101, Operand 1 = ffffffff, Operand 2 = 0000001f

Data dependent corner case: Shift Left 31 places (the max allowable shift places). (Port 3)

time = 16300 At port 2 Input Command = 0101: Correct: Output (80000000) is equal to Expected Result (80000000)

time = 16350 Port 3: Input Command = 0101, Operand 1 = ffffffff, Operand 2 = 0000001f

Data dependent corner case: Shift Left 31 places (the max allowable shift places). (Port 4)

time = 16500 At port 3 Input Command = 0101: Correct: Output (80000000) is equal to Expected Result (80000000)

time = 16550 Port 4: Input Command = 0101, Operand 1 = ffffffff, Operand 2 = 0000001f

Data dependent corner case: Shift Right 31 places (the max allowable shift places). (Port 1)

time = 16700 At port 4 Input Command = 0101: Correct: Output (80000000) is equal to Expected Result (80000000)

time = 16750 Port 1: Input Command = 0110, Operand 1 = ffffffff, Operand 2 = 0000001f

Data dependent corner case: Shift Right 31 places (the max allowable shift places). (Port 2)

time = 16900 At port 1 Input Command = 0110: Correct: Output (00000001) is equal to Expected Result (00000001)

time = 16950 Port 2: Input Command = 0110, Operand 1 = ffffffff, Operand 2 = 0000001f

Data dependent corner case: Shift Right 31 places (the max allowable shift places). (Port 3)

time = 17100 At port 2 Input Command = 0110: Correct: Output (00000001) is equal to Expected Result (00000001)

time = 17150 Port 3: Input Command = 0110, Operand 1 = ffffffff, Operand 2 = 0000001f

Data dependent corner case: Shift Right 31 places (the max allowable shift places). (Port 4)

time = 17300 At port 3 Input Command = 0110: Correct: Output (00000001) is equal to Expected Result (00000001)

time = 17350 Port 4: Input Command = 0110, Operand 1 = ffffffff, Operand 2 = 0000001f

Data dependent corner case: Add max number â œFFFFFFFFâ X. (Port 1)

time = 17500 At port 4 Input Command = 0110: Correct: Output (00000001) is equal to Expected Result (00000001)

time = 17550 Port 1: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = ffffffff

Data dependent corner case: Add max number â œFFFFFFFFâ X. (Port 2)

time = 17750 Port 2: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = ffffffff

Data dependent corner case: Add max number â œFFFFFFFFâ X. (Port 3)

time = 17950 Port 3: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = ffffffff

Data dependent corner case: Add max number â œFFFFFFFFâ X. (Port 4)

time = 18150 Port 4: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = ffffffff

Data dependent corner case: Add max number â œFFFFFFFFâ X with min number. (Port 1)

time = 18350 Port 1: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = 00000000

Data dependent corner case: Add max number â œFFFFFFFFâ X with min number. (Port 2)

time = 18500 Input Command = 0001: Error at Port 1: Overflow but the response is Success ('01'b)

time = 18550 Port 2: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = ffffffff

Data dependent corner case: Add max number â œFFFFFFFFâ X with min number. (Port 3)

time = 18750 Port 3: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = 00000000

Data dependent corner case: Add max number â œFFFFFFFFâ X with min number. (Port 4)

time = 18950 Port 4: Input Command = 0001, Operand 1 = ffffffff, Operand 2 = 00000000

Data dependent corner case: Add min number. (Port 1)

time = 19150 Port 1: Input Command = 0001, Operand 1 = 00000000, Operand 2 = 00000000

Data dependent corner case: Add min number. (Port 2)

time = 19300 At port 1 Input Command = 0001: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 19350 Port 2: Input Command = 0001, Operand 1 = 00000000, Operand 2 = 00000000

Data dependent corner case: Add min number. (Port 3)

time = 19500 At port 2 Input Command = 0001: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 19550 Port 3: Input Command = 0001, Operand 1 = 00000000, Operand 2 = 00000000

Data dependent corner case: Add min number. (Port 4)

time = 19700 At port 2 Input Command = 0001: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 19750 Port 4: Input Command = 0001, Operand 1 = 00000000, Operand 2 = 00000000

Data dependent corner case: Subtract min number. (Port 1)

time = 19900 At port 2 Input Command = 0001: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 19950 Port 1: Input Command = 0010, Operand 1 = 00000000, Operand 2 = 00000000

Data dependent corner case: Subtract min number. (Port 2)

time = 20100 At port 1 Input Command = 0010: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 20150 Port 2: Input Command = 0010, Operand 1 = 00000000, Operand 2 = 00000000

Data dependent corner case: Subtract min number. (Port 3)

time = 20300 At port 2 Input Command = 0010: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 20350 Port 3: Input Command = 0010, Operand 1 = 00000000, Operand 2 = 00000000

Data dependent corner case: Subtract min number. (Port 4)

time = 20500 At port 2 Input Command = 0010: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 20550 Port 4: Input Command = 0010, Operand 1 = 00000000, Operand 2 = 00000000

Data dependent corner case: Subtract max number. (Port 1)

time = 20700 At port 2 Input Command = 0010: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 20750 Port 1: Input Command = 0010, Operand 1 = ffffffff, Operand 2 = ffffffff

Data dependent corner case: Subtract max number. (Port 2)

time = 20900 At port 1 Input Command = 0010: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 20950 Port 2: Input Command = 0010, Operand 1 = ffffffff, Operand 2 = ffffffff

Data dependent corner case: Subtract max number. (Port 3)

time = 21100 At port 2 Input Command = 0010: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 21150 Port 3: Input Command = 0010, Operand 1 = ffffffff, Operand 2 = ffffffff

Data dependent corner case: Subtract max number. (Port 4)

time = 21300 At port 2 Input Command = 0010: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 21350 Port 4: Input Command = 0010, Operand 1 = ffffffff, Operand 2 = ffffffff

Data dependent corner case: Subtract max and min numbers. (Port 1)

time = 21500 At port 2 Input Command = 0010: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 21550 Port 1: Input Command = 0010, Operand 1 = 00000000, Operand 2 = ffffffff

Data dependent corner case: Subtract max and min numbers. (Port 2)

time = 21750 Port 2: Input Command = 0010, Operand 1 = 00000000, Operand 2 = 00000000

time = 21800 Input Command = 0010: Error at Port 1: Underflow but the response is Success ('01'b)

Data dependent corner case: Subtract max and min numbers. (Port 3)

time = 21900 At port 2 Input Command = 0010: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 21950 Port 3: Input Command = 0010, Operand 1 = 00000000, Operand 2 = ffffffff

Data dependent corner case: Subtract max and min numbers. (Port 4)

time = 22100 At port 2 Input Command = 0010: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 22150 Port 4: Input Command = 0010, Operand 1 = 00000000, Operand 2 = ffffffff

Invalid Input Data (Port 1)

time = 22300 At port 2 Input Command = 0010: Correct: Output (00000000) is equal to Expected Result (00000000)

time = 22450 Port 1: Input Command = 0001, Operand 1 = 0000000f, Operand 2 = 0000000f

Invalid Input Data (Port 2)

time = 22750 Port 2: Input Command = 0001, Operand 1 = 0000000f, Operand 2 = 0000000f

time = 22800 Input Command = 0001: Error at Port 2: Output (0000000f) is not equal to Expected Result (00000000)

Invalid Input Data (Port 3)

time = 22900 Input Command = 0001: Error at Port 2: Output (0000000f) is not equal to Expected Result (0000001e)

time = 23050 Port 3: Input Command = 0001, Operand 1 = 0000000f, Operand 2 = 0000000f

Invalid Input Data (Port 4)

time = 23350 Port 4: Input Command = 0001, Operand 1 = 0000000f, Operand 2 = 0000000f

Invalid Input Data 2 (Port 1)

time = 23650 Port 1: Input Command = 0001, Operand 1 = 0000000f, Operand 2 = 0000000f

Invalid Input Data 2 (Port 2)

time = 23800 At port 1 Input Command = 0001: Correct: Output (0000001e) is equal to Expected Result (0000001e)

time = 23950 Port 2: Input Command = 0001, Operand 1 = 0000000f, Operand 2 = 0000000f

Invalid Input Data 2 (Port 3)

time = 24100 At port 2 Input Command = 0001: Correct: Output (0000001e) is equal to Expected Result (0000001e)

time = 24250 Port 3: Input Command = 0001, Operand 1 = 0000000f, Operand 2 = 0000000f

Invalid Input Data 2 (Port 4)

time = 24400 At port 3 Input Command = 0001: Correct: Output (0000001e) is equal to Expected Result (0000001e)

time = 24550 Port 4: Input Command = 0001, Operand 1 = 0000000f, Operand 2 = 0000000f

Illegal Input Command: 0011 (Port 1)

time = 24750 Port 1: Input Command = 0011, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0011 (Port 2)

time = 24900 Error at Port 1: Illegal command but the response is Success ('01'b)

time = 24950 Port 2: Input Command = 0011, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0011 (Port 3)

time = 25100 Error at Port 2: Illegal command but the response is Success ('01'b)

time = 25150 Port 3: Input Command = 0011, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0011 (Port 4)

time = 25300 Error at Port 3: Illegal command but the response is Success ('01'b)

time = 25350 Port 4: Input Command = 0011, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0100 (Port 1)

time = 25550 Port 1: Input Command = 0100, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0100 (Port 2)

time = 25700 Error at Port 1: Illegal command but the response is Success ('01'b)

time = 25750 Port 2: Input Command = 0100, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0100 (Port 3)

time = 25900 Error at Port 2: Illegal command but the response is Success ('01'b)

time = 25950 Port 3: Input Command = 0100, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0100 (Port 4)

time = 26100 Error at Port 3: Illegal command but the response is Success ('01'b)

time = 26150 Port 4: Input Command = 0100, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0100 (Port 1)

time = 26300 Error at Port 4: Illegal command but the response is Success ('01'b)

time = 26350 Port 1: Input Command = 0100, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0100 (Port 2)

time = 26500 Error at Port 1: Illegal command but the response is Success ('01'b)

time = 26550 Port 2: Input Command = 0100, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0100 (Port 3)

time = 26700 Error at Port 2: Illegal command but the response is Success ('01'b)

time = 26750 Port 3: Input Command = 0100, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0100 (Port 4)

time = 26900 Error at Port 3: Illegal command but the response is Success ('01'b)

time = 26950 Port 4: Input Command = 0100, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0111 (Port 1)

time = 27100 Error at Port 4: Illegal command but the response is Success ('01'b)

time = 27150 Port 1: Input Command = 0111, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0111 (Port 2)

time = 27300 Error at Port 1: Illegal command but the response is Success ('01'b)

time = 27350 Port 2: Input Command = 0111, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0111 (Port 3)

time = 27500 Error at Port 2: Illegal command but the response is Success ('01'b)

time = 27550 Port 3: Input Command = 0111, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 0111 (Port 4)

time = 27700 Error at Port 3: Illegal command but the response is Success ('01'b)

time = 27750 Port 4: Input Command = 0111, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1000 (Port 1)

time = 27900 Error at Port 4: Illegal command but the response is Success ('01'b)

time = 27950 Port 1: Input Command = 1000, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1000 (Port 2)

time = 28100 Error at Port 1: Illegal command but the response is Success ('01'b)

time = 28150 Port 2: Input Command = 1000, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1000 (Port 3)

time = 28300 Error at Port 2: Illegal command but the response is Success ('01'b)

time = 28350 Port 3: Input Command = 1000, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1000 (Port 4)

time = 28500 Error at Port 3: Illegal command but the response is Success ('01'b)

time = 28550 Port 4: Input Command = 1000, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1001 (Port 1)

time = 28700 Error at Port 4: Illegal command but the response is Success ('01'b)

time = 28750 Port 1: Input Command = 1001, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1001 (Port 2)

time = 28900 Error at Port 1: Illegal command but the response is Success ('01'b)

time = 28950 Port 2: Input Command = 1001, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1001 (Port 3)

time = 29100 Error at Port 2: Illegal command but the response is Success ('01'b)

time = 29150 Port 3: Input Command = 1001, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1001 (Port 4)

time = 29300 Error at Port 3: Illegal command but the response is Success ('01'b)

time = 29350 Port 4: Input Command = 1001, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1010 (Port 1)

time = 29500 Error at Port 4: Illegal command but the response is Success ('01'b)

time = 29550 Port 1: Input Command = 1010, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1010 (Port 2)

time = 29700 Error at Port 1: Illegal command but the response is Success ('01'b)

time = 29750 Port 2: Input Command = 1010, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1010 (Port 3)

time = 29900 Error at Port 2: Illegal command but the response is Success ('01'b)

time = 29950 Port 3: Input Command = 1010, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1010 (Port 4)

time = 30100 Error at Port 3: Illegal command but the response is Success ('01'b)

time = 30150 Port 4: Input Command = 1010, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1011 (Port 1)

time = 30300 Error at Port 4: Illegal command but the response is Success ('01'b)

time = 30350 Port 1: Input Command = 1011, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1011 (Port 2)

time = 30500 Error at Port 1: Illegal command but the response is Success ('01'b)

time = 30550 Port 2: Input Command = 1011, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1011 (Port 3)

time = 30700 Error at Port 2: Illegal command but the response is Success ('01'b)

time = 30750 Port 3: Input Command = 1011, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1011 (Port 4)

time = 30900 Error at Port 3: Illegal command but the response is Success ('01'b)

time = 30950 Port 4: Input Command = 1011, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1100 (Port 1)

time = 31100 Error at Port 4: Illegal command but the response is Success ('01'b)

time = 31150 Port 1: Input Command = 1100, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1100 (Port 2)

time = 31300 Error at Port 1: Illegal command but the response is Success ('01'b)

time = 31350 Port 2: Input Command = 1100, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1100 (Port 3)

time = 31500 Error at Port 2: Illegal command but the response is Success ('01'b)

time = 31550 Port 3: Input Command = 1100, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1100 (Port 4)

time = 31700 Error at Port 3: Illegal command but the response is Success ('01'b)

time = 31750 Port 4: Input Command = 1100, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1101 (Port 1)

time = 31900 Error at Port 4: Illegal command but the response is Success ('01'b)

time = 31950 Port 1: Input Command = 1101, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1101 (Port 2)

time = 32100 Error at Port 1: Illegal command but the response is Success ('01'b)

time = 32150 Port 2: Input Command = 1101, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1101 (Port 3)

time = 32300 Error at Port 2: Illegal command but the response is Success ('01'b)

time = 32350 Port 3: Input Command = 1101, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1101 (Port 4)

time = 32500 Error at Port 3: Illegal command but the response is Success ('01'b)

time = 32550 Port 4: Input Command = 1101, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1110 (Port 1)

time = 32700 Error at Port 4: Illegal command but the response is Success ('01'b)

time = 32750 Port 1: Input Command = 1110, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1110 (Port 2)

time = 32900 Error at Port 1: Illegal command but the response is Success ('01'b)

time = 32950 Port 2: Input Command = 1110, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1110 (Port 3)

time = 33100 Error at Port 2: Illegal command but the response is Success ('01'b)

time = 33150 Port 3: Input Command = 1110, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1110 (Port 4)

time = 33300 Error at Port 3: Illegal command but the response is Success ('01'b)

time = 33350 Port 4: Input Command = 1110, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1111 (Port 1)

time = 33500 Error at Port 4: Illegal command but the response is Success ('01'b)

time = 33550 Port 1: Input Command = 1111, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1111 (Port 2)

time = 33700 Error at Port 1: Illegal command but the response is Success ('01'b)

time = 33750 Port 2: Input Command = 1111, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1111 (Port 3)

time = 33900 Error at Port 2: Illegal command but the response is Success ('01'b)

time = 33950 Port 3: Input Command = 1111, Operand 1 = 0010f011, Operand 2 = 00a0c001

Illegal Input Command: 1111 (Port 4)

time = 34100 Error at Port 3: Illegal command but the response is Success ('01'b)

```
# time =              34150 Port 4: Input Command = 1111, Operand 1 =  0010f011, Operand 2 =
00a0c001

# Reset Check

# time =              34300 Error at Port 4: Illegal command but the response is Success ('01'b)

#              35150: At end of test error count is 80 and correct count = 60

# ** Note: $finish    : /nfs/home/d/d_gurusw/COEN6541/project/calc1_tb (7).sv(2240)

#    Time: 35150 ns  Iteration: 1  Instance: /calc1_tb
```

6.2. Testbench Code

```
`default_nettype none
module calc1_tb ();

  reg       c_clk      ;
  reg [ 1:7] reset      ;
  reg [ 0:3] req1_cmd_in ;
  reg [0:31] req1_data_in;
  reg [ 0:3] req2_cmd_in ;
  reg [0:31] req2_data_in;
  reg [ 0:3] req3_cmd_in ;
  reg [0:31] req3_data_in;
  reg [ 0:3] req4_cmd_in ;
  reg [0:31] req4_data_in;

  wire [ 0:1] out_resp1;
  wire [0:31] out_data1;
  wire [ 0:1] out_resp2;
  wire [0:31] out_data2;
  wire [ 0:1] out_resp3;
  wire [0:31] out_data3;
  wire [ 0:1] out_resp4;
  wire [0:31] out_data4;

  reg [0:31] data1, data2, data3, data4;
  reg [ 0:3] cmd1, cmd2, cmd3, cmd4;
  reg [0:31] expected_data1, expected_data2, expected_data3, expected_data4;

  integer error_count = 0;

  integer correct_count = 0;
  integer t = 0;

  localparam
```

```
    No_Op = 4'b0000,
    Add = 4'b0001,
    Sub = 4'b0010,
    Shift_Left = 4'b0101,
    Shift_Right = 4'b0110;

  localparam
    No_Response = 2'b00,
    Success = 2'b01,
    Invalid_Command = 2'b10,
    Overflow = 2'b10,
    Underflow = 2'b10,
    Internal_Error = 2'b11;

  localparam
    Max = 32'hFFFFFFFF,
    Min = 32'h00000000;

  reg [0:3] Opcode;

  reg overflow_check1, overflow_check2, overflow_check3, overflow_check4, underflow_check1,
underflow_check2, underflow_check3, underflow_check4;

  calc1_top calc1_top (
    .c_clk      (c_clk      ),
    .reset      (reset      ),
    .req1_cmd_in (req1_cmd_in ),
    .req1_data_in(req1_data_in),
    .req2_cmd_in (req2_cmd_in ),
    .req2_data_in(req2_data_in),
    .req3_cmd_in (req3_cmd_in ),
    .req3_data_in(req3_data_in),
    .req4_cmd_in (req4_cmd_in ),
    .req4_data_in(req4_data_in),
    .out_resp1   (out_resp1   ),
    .out_data1   (out_data1   ),
    .out_resp2   (out_resp2   ),
    .out_data2   (out_data2   ),
    .out_resp3   (out_resp3   ),
    .out_data3   (out_data3   ),
    .out_resp4   (out_resp4   ),
    .out_data4   (out_data4   ),
    .scan_out    (          ),
    .a_clk       (          ),
    .b_clk       (          ),
    .error_found (          ),
    .scan_in     (          )
  );
```

```
  initial begin
    c_clk = 0;
    forever #50 c_clk = !c_clk;
  end

  initial begin
    reset[1:7]     =      7'b1111111;

    req1_cmd_in    = 4'b0000;
    req2_cmd_in    = 4'b0000;
    req3_cmd_in    =4'b0000;
    req4_cmd_in    = 4'b0000;
    req1_data_in   =      32'h00000000;
    req2_data_in   =      32'h00000000;
    req3_data_in   =      32'h00000000;
    req4_data_in   =      32'h00000000;

    repeat(7)@(posedge c_clk);
    reset[1:7] =    ~reset[1:7];

    repeat(1)@(posedge c_clk);
    //  reset[1:7]  <=     ~reset[1:7];



    Opcode = Add;

    $display("Add (Port 1)");
    req1_cmd_in                =      Opcode;
    req1_data_in         =      32'h00000001;
    data1                  =      req1_data_in;
    cmd1                   =      req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in                =      4'b0000;
    req1_data_in         =      32'h00000001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 + req1_data_in;

    $display("Add (Port 2)");
    req2_cmd_in                =      Opcode;
    req2_data_in         =      32'h00000001;
    data2                  =      req2_data_in;
    cmd2                   =      req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in                =      4'b0000;
```

```
    req2_data_in        =     32'h00000001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 + req2_data_in;

    $display("Add (Port 3)");
    req3_cmd_in             =     Opcode;
    req3_data_in        =     32'h00000001;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in             =     4'b0000;
    req3_data_in        =     32'h00000001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 + req3_data_in;

    $display("Add (Port 4)");
    req4_cmd_in             =     Opcode;
    req4_data_in        =     32'h00000001;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in             =     4'b0000;
    req4_data_in        =     32'h00000001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 + req4_data_in;

    Opcode = Sub;

    $display("Sub (Port 1)");
    req1_cmd_in         =     Opcode;
    req1_data_in        =     32'h00000002;
    data1               =     req1_data_in;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     4'b0000;
    req1_data_in        =     32'h00000001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 - req1_data_in;

    $display("Sub (Port 2)");
```

```
    req2_cmd_in          =      Opcode;
    req2_data_in         =      32'h00000002;
    data2                =      req2_data_in;
    cmd2                 =      req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in                 =      4'b0000;
    req2_data_in         =      32'h00000001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 - req2_data_in;

    $display("Sub (Port 3)");
    req3_cmd_in          =      Opcode;
    req3_data_in         =      32'h00000002;
    data3                =      req3_data_in;
    cmd3                 =      req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in                 =      4'b0000;
    req3_data_in         =      32'h00000001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 - req3_data_in;

    $display("Sub (Port 4)");
    req4_cmd_in          =      Opcode;
    req4_data_in         =      32'h00000002;
    data4                =      req4_data_in;
    cmd4                 =      req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in                 =      4'b0000;
    req4_data_in         =      32'h00000001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 - req4_data_in;

    Opcode = Shift_Left;

    $display("Shift Left (Port 1)");
    req1_cmd_in          =      Opcode;
    req1_data_in         =      32'h00000001;
    data1                =      req1_data_in;
    cmd1                 =      req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in                 =      4'b0000;
    req1_data_in         =      32'h00000002;
```

```
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 << req1_data_in[27:31];

    $display("Shift Left (Port 2)");
    req2_cmd_in          =      Opcode;
    req2_data_in         =      32'h00000001;
    data2                =      req2_data_in;
    cmd2                 =      req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in                 =      4'b0000;
    req2_data_in         =      32'h00000002;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 << req2_data_in[27:31];

    $display("Shift Left (Port 3)");
    req3_cmd_in          =      Opcode;
    req3_data_in         =      32'h00000001;
    data3                =      req3_data_in;
    cmd3                 =      req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in                 =      4'b0000;
    req3_data_in         =      32'h00000002;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 << req3_data_in[27:31];

    $display("Shift Left (Port 4)");
    req4_cmd_in          =      Opcode;
    req4_data_in         =      32'h00000001;
    data4                =      req4_data_in;
    cmd4                 =      req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in                 =      4'b0000;
    req4_data_in         =      32'h00000002;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 << req4_data_in[27:31];

    Opcode = Shift_Right;

    $display("Shift Right (Port 1)");
    req1_cmd_in          =      Opcode;
```

```
    req1_data_in        =     32'h80000000;
    data1               =     req1_data_in;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     4'b0000;
    req1_data_in        =     32'h00000002;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 >> req1_data_in[27:31];

    $display("Shift Right (Port 2)");
    req2_cmd_in         =     Opcode;
    req2_data_in        =     32'h80000000;
    data2               =     req2_data_in;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in             =     4'b0000;
    req2_data_in        =     32'h00000002;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 >> req2_data_in[27:31];

    $display("Shift Right (Port 3)");
    req3_cmd_in         =     Opcode;
    req3_data_in        =     32'h80000000;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in             =     4'b0000;
    req3_data_in        =     32'h00000002;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 >> req3_data_in[27:31];

    $display("Shift Right (Port 4)");
    req4_cmd_in         =     Opcode;
    req4_data_in        =     32'h80000000;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in             =     4'b0000;
    req4_data_in        =     32'h00000002;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
```

```
    expected_data4 = data4 >> req4_data_in[27:31];

    Opcode = Add;

    $display("Overflow (Port 1)");
    req1_cmd_in          =    Opcode;
    req1_data_in         =    32'hFFFFFFFF;
    data1                =    req1_data_in;
    cmd1                 =    req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in          =    4'b0000;
    req1_data_in         =    32'h0000000F;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 + req1_data_in;

    $display("Overflow (Port 2)");
    req2_cmd_in          =    Opcode;
    req2_data_in         =    32'hFFFFFFFF;
    data2                =    req2_data_in;
    cmd2                 =    req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in          =    4'b0000;
    req2_data_in         =    32'h0000000F;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 + req2_data_in;

    $display("Overflow (Port 3)");
    req3_cmd_in          =    Opcode;
    req3_data_in         =    32'hFFFFFFFF;
    data3                =    req3_data_in;
    cmd3                 =    req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in          =    4'b0000;
    req3_data_in         =    32'h0000000F;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 + req3_data_in;

    $display("Overflow (Port 4)");
    req4_cmd_in          =    Opcode;
    req4_data_in         =    32'hFFFFFFFF;
    data4                =    req4_data_in;
    cmd4                 =    req4_cmd_in;
```

```
    repeat(1)@(posedge c_clk);
    req4_cmd_in                =      4'b0000;
    req4_data_in         =      32'h0000000F;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 + req4_data_in;

    Opcode = Sub;

    $display("Underflow (Port 1)");
    req1_cmd_in                =      Opcode;
    req1_data_in         =      32'h00000000;
    data1                =      req1_data_in;
    cmd1                 =      req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in                =      4'b0000;
    req1_data_in         =      32'h0000000F;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 - req1_data_in;

    $display("Underflow (Port 2)");
    req2_cmd_in                =      Opcode;
    req2_data_in         =      32'h00000000;
    data2                =      req2_data_in;
    cmd2                 =      req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in                =      4'b0000;
    req2_data_in         =      32'h0000000F;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 - req2_data_in;

    $display("Underflow (Port 3)");
    req3_cmd_in                =      Opcode;
    req3_data_in         =      32'h00000000;
    data3                =      req3_data_in;
    cmd3                 =      req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in                =      4'b0000;
    req3_data_in         =      32'h0000000F;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 - req3_data_in;
```

```
    $display("Underflow (Port 4)");
    req4_cmd_in              =      Opcode;
    req4_data_in         =      32'h00000000;
    data4                =      req4_data_in;
    cmd4                 =      req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in              =      4'b0000;
    req4_data_in         =      32'h0000000F;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 - req4_data_in;


    Opcode = Add;

    $display("Dirty State: Addition followed by Subtraction (Port 1)");
    req1_cmd_in              =      Opcode;
    req1_data_in         =      32'h00000FFF;
    data1                =      req1_data_in;
    cmd1                 =      req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =      Sub;
    req1_data_in         =      32'h000F0F0F;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 + req1_data_in;

    $display("Dirty State: Addition followed by Subtraction (Port 2)");
    req2_cmd_in              =      Opcode;
    req2_data_in         =      32'h00000FFF;
    data2                =      req2_data_in;
    cmd2                 =      req2_cmd_in;
    repeat(1)@(posedge c_clk);
    cmd1=4'b0000;
    req2_cmd_in              =      Sub;
    req2_data_in         =      32'h000F0F0F;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 + req2_data_in;

    $display("Dirty State: Addition followed by Subtraction (Port 3)");
    req3_cmd_in              =      Opcode;
    req3_data_in         =      32'h00000FFF;
    data3                =      req3_data_in;
```

```
    cmd3                =      req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in              =      Sub;
    req3_data_in        =      32'h000F0F0F;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 + req3_data_in;

    $display("Dirty State: Addition followed by Subtraction (Port 4)");
    req4_cmd_in              =      Opcode;
    req4_data_in        =      32'h00000FFF;
    data4               =      req4_data_in;
    cmd4                =      req4_cmd_in;
    cmd1=4'b0000;
    repeat(1)@(posedge c_clk);
    req4_cmd_in              =      Sub;
    req4_data_in        =      32'h000F0F0F;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 + req4_data_in;

    repeat(1)@(posedge c_clk);

    Opcode = Shift_Left;
    cmd2=4'b0000;
    $display("Dirty State: Shift Left followed by Shift Right (Port 1)");
    req1_cmd_in              =      Opcode;
    req1_data_in        =      32'h00000FFF;
    data1               =      req1_data_in;
    cmd1                =      req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =      Shift_Right;
    req1_data_in        =      32'h0000000F;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 << req1_data_in[27:31];

    $display("Dirty State: Shift Left followed by Shift Right (Port 2)");
    req2_cmd_in              =      Opcode;
    req2_data_in        =      32'h00000FFF;
    data2               =      req2_data_in;
    cmd2                =      req2_cmd_in;
    cmd1=4'b0000;
    repeat(1)@(posedge c_clk);
    req2_cmd_in              =      Shift_Right;
```

```
    req2_data_in        =     32'h0000000F;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 << req2_data_in[27:31];

    $display("Dirty State: Shift Left followed by Shift Right (Port 3)");
    req3_cmd_in             =     Opcode;
    req3_data_in        =     32'h00000FFF;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in             =     Shift_Right;
    req3_data_in        =     32'h0000000F;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 << req3_data_in[27:31];

    $display("Dirty State: Shift Left followed by Shift Right (Port 4)");
    req4_cmd_in             =     Opcode;
    req4_data_in        =     32'h00000FFF;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in             =     Shift_Right;
    req4_data_in        =     32'h0000000F;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 << req4_data_in[27:31];
    repeat(1)@(posedge c_clk);

      Opcode = Add;


    $display("Dirty State: Across all ports, Addition followed by subtraction");
    req1_cmd_in             =     Opcode;
    req2_cmd_in             =     Opcode;
    req3_cmd_in             =     Opcode;
    req4_cmd_in             =     Opcode;
    req1_data_in        =     32'h00000FFF;
    req2_data_in        =     32'h00000FFF;
    req3_data_in        =     32'h00000FFF;
    req4_data_in        =     32'h00000FFF;
    data1               =     req1_data_in;
    data2               =     req2_data_in;
    data3               =     req3_data_in;
```

```
    data4               =     req4_data_in;
    cmd1                =     req1_cmd_in;
    cmd2                =     req2_cmd_in;
    cmd3                =     req3_cmd_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =     Sub;
    req2_cmd_in              =     Sub;
    req3_cmd_in              =     Sub;
    req4_cmd_in              =     Sub;
    req1_data_in       =     32'h0000000F;
    req2_data_in       =     32'h0000000F;
    req3_data_in       =     32'h0000000F;
    req4_data_in       =     32'h0000000F;
    $display("time = %t All Ports: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 + req1_data_in;
    expected_data2 = data2 + req2_data_in;
    expected_data3 = data3 + req3_data_in;
    expected_data4 = data4 + req4_data_in;

    Opcode = Shift_Left;

    $display("Dirty State: Across all ports, Shift left followed by shift right");
    req1_cmd_in              =     Opcode;
    req2_cmd_in              =     Opcode;
    req3_cmd_in              =     Opcode;
    req4_cmd_in              =     Opcode;
    req1_data_in       =     32'h00000FFF;
    req2_data_in       =     32'h00000FFF;
    req3_data_in       =     32'h00000FFF;
    req4_data_in       =     32'h00000FFF;
    data1               =     req1_data_in;
    data2               =     req2_data_in;
    data3               =     req3_data_in;
    data4               =     req4_data_in;
    cmd1                =     req1_cmd_in;
    cmd2                =     req2_cmd_in;
    cmd3                =     req3_cmd_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =     Shift_Right;
    req2_cmd_in              =     Shift_Right;
    req3_cmd_in              =     Shift_Right;
    req4_cmd_in              =     Shift_Right;
    req1_data_in       =     32'h0000000F;
    req2_data_in       =     32'h0000000F;
```

```
    req3_data_in        =     32'h0000000F;
    req4_data_in        =     32'h0000000F;
    $display("time = %t All Ports: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 << req1_data_in[27:31];
    expected_data2 = data2 << req2_data_in[27:31];
    expected_data3 = data3 << req3_data_in[27:31];
    expected_data4 = data4 << req4_data_in[27:31];

    Opcode = Add;

    $display("Addition Priority");
    req1_cmd_in             =     Opcode;
    req2_cmd_in             =     Opcode;
    req3_cmd_in             =     Opcode;
    req4_cmd_in             =     Opcode;
    req1_data_in        =     32'h00000FFF;
    req2_data_in        =     32'h00000FFF;
    req3_data_in        =     32'h00000FFF;
    req4_data_in        =     32'h00000FFF;
    data1               =     req1_data_in;
    data2               =     req2_data_in;
    data3               =     req3_data_in;
    data4               =     req4_data_in;
    cmd1                =     req1_cmd_in;
    cmd2                =     req2_cmd_in;
    cmd3                =     req3_cmd_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     No_Op;
    req2_cmd_in             =     No_Op;
    req3_cmd_in             =     No_Op;
    req4_cmd_in             =     No_Op;
    req1_data_in        =     32'h0000000F;
    req2_data_in        =     32'h0000000F;
    req3_data_in        =     32'h0000000F;
    req4_data_in        =     32'h0000000F;
    $display("time = %t All Ports: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(4)@(posedge c_clk);
    expected_data1 = data1 + req1_data_in;
    expected_data2 = data2 + req2_data_in;
    expected_data3 = data3 + req3_data_in;
    expected_data4 = data4 + req4_data_in;

    Opcode = Sub;
```

```
    $display("Subtraction Priority");
    req1_cmd_in             =     Opcode;
    req2_cmd_in             =     Opcode;
    req3_cmd_in             =     Opcode;
    req4_cmd_in             =     Opcode;
    req1_data_in       =     32'h00000FFF;
    req2_data_in       =     32'h00000FFF;
    req3_data_in       =     32'h00000FFF;
    req4_data_in       =     32'h00000FFF;
    data1              =     req1_data_in;
    data2              =     req2_data_in;
    data3              =     req3_data_in;
    data4              =     req4_data_in;
    cmd1               =     req1_cmd_in;
    cmd2               =     req2_cmd_in;
    cmd3               =     req3_cmd_in;
    cmd4               =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     No_Op;
    req2_cmd_in             =     No_Op;
    req3_cmd_in             =     No_Op;
    req4_cmd_in             =     No_Op;
    req1_data_in       =     32'h0000000F;
    req2_data_in       =     32'h0000000F;
    req3_data_in       =     32'h0000000F;
    req4_data_in       =     32'h0000000F;
    $display("time = %t All Ports: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 - req1_data_in;
    expected_data2 = data2 - req2_data_in;
    expected_data3 = data3 - req3_data_in;
    expected_data4 = data4 - req4_data_in;

        Opcode = Shift_Left;

    $display("Shift left Priority");
    req1_cmd_in             =     Opcode;
    req2_cmd_in             =     Opcode;
    req3_cmd_in             =     Opcode;
    req4_cmd_in             =     Opcode;
    req1_data_in       =     32'h00000FFF;
    req2_data_in       =     32'h00000FFF;
    req3_data_in       =     32'h00000FFF;
    req4_data_in       =     32'h00000FFF;
    data1              =     req1_data_in;
    data2              =     req2_data_in;
    data3              =     req3_data_in;
```

```
    data4               =     req4_data_in;
    cmd1                =     req1_cmd_in;
    cmd2                =     req2_cmd_in;
    cmd3                =     req3_cmd_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =     No_Op;
    req2_cmd_in              =     No_Op;
    req3_cmd_in              =     No_Op;
    req4_cmd_in              =     No_Op;
    req1_data_in       =     32'h0000000F;
    req2_data_in       =     32'h0000000F;
    req3_data_in       =     32'h0000000F;
    req4_data_in       =     32'h0000000F;
    $display("time = %t All Ports: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 << req1_data_in[27:31];
    expected_data2 = data2 << req2_data_in[27:31];
    expected_data3 = data3 << req3_data_in[27:31];
    expected_data4 = data4 << req4_data_in[27:31];

    Opcode = Shift_Right;

    $display("Shift right Priority");
    req1_cmd_in              =     Opcode;
    req2_cmd_in              =     Opcode;
    req3_cmd_in              =     Opcode;
    req4_cmd_in              =     Opcode;
    req1_data_in       =     32'h00000FFF;
    req2_data_in       =     32'h00000FFF;
    req3_data_in       =     32'h00000FFF;
    req4_data_in       =     32'h00000FFF;
    data1               =     req1_data_in;
    data2               =     req2_data_in;
    data3               =     req3_data_in;
    data4               =     req4_data_in;
    cmd1                =     req1_cmd_in;
    cmd2                =     req2_cmd_in;
    cmd3                =     req3_cmd_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =     No_Op;
    req2_cmd_in              =     No_Op;
    req3_cmd_in              =     No_Op;
    req4_cmd_in              =     No_Op;
    req1_data_in       =     32'h0000000F;
    req2_data_in       =     32'h0000000F;
```

```
    req3_data_in        =     32'h0000000F;
    req4_data_in        =     32'h0000000F;
    $display("time = %t All Ports: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 >> req1_data_in[27:31];
    expected_data2 = data2 >> req2_data_in[27:31];
    expected_data3 = data3 >> req3_data_in[27:31];
    expected_data4 = data4 >> req4_data_in[27:31];

  $display("All operations at the same time");
    req1_cmd_in             =     Add;
    req2_cmd_in             =     Sub;
    req3_cmd_in             =     Shift_Left;
    req4_cmd_in             =     Shift_Right;
    req1_data_in        =     32'h00000FFF;
    req2_data_in        =     32'h00000FFF;
    req3_data_in        =     32'h00000FFF;
    req4_data_in        =     32'h00000FFF;
    data1               =     req1_data_in;
    data2               =     req2_data_in;
    data3               =     req3_data_in;
    data4               =     req4_data_in;
    cmd1                =     req1_cmd_in;
    cmd2                =     req2_cmd_in;
    cmd3                =     req3_cmd_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     No_Op;
    req2_cmd_in             =     No_Op;
    req3_cmd_in             =     No_Op;
    req4_cmd_in             =     No_Op;
    req1_data_in        =     32'h0000000F;
    req2_data_in        =     32'h0000000F;
    req3_data_in        =     32'h0000000F;
    req4_data_in        =     32'h0000000F;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Add, data1, req1_data_in);
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Sub, data2, req2_data_in);
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Shift_Left, data3, req3_data_in);
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Shift_Right, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 + req1_data_in;
    expected_data2 = data2 - req2_data_in;
    expected_data3 = data3 << req3_data_in[27:31];
```

```
    expected_data4 = data4 >> req4_data_in[27:31];

    repeat(3)@(posedge c_clk);


    Opcode = Shift_Left;

    $display("Check that the high-order 27 bits are ignored in the second operand of shift left
command. (Port 1)");
    req1_cmd_in              =      Opcode;
    req1_data_in        =      32'hFFFFFFFF;
    data1                =      req1_data_in;
    cmd1                 =      req1_cmd_in;
    cmd2       =4'b0000;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =      4'b0000;
    req1_data_in        =      32'h00AB0FFF;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 << req1_data_in[27:31];
    repeat(1)@(posedge c_clk);
    cmd1       =4'b0000;
    $display("Check that the high-order 27 bits are ignored in the second operand of shift left
command. (Port 2)");
    req2_cmd_in              =      Opcode;
    req2_data_in        =      32'hFFFFFFFF;
    data2                =      req2_data_in;
    cmd2                 =      req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in              =      4'b0000;
    req2_data_in        =      32'h00AB0FFF;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 << req2_data_in[27:31];

    $display("Check that the high-order 27 bits are ignored in the second operand of shift left
command. (Port 3)");
    req3_cmd_in              =      Opcode;
    req3_data_in        =      32'hFFFFFFFF;
    data3                =      req3_data_in;
    cmd3                 =      req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in              =      4'b0000;
    req3_data_in        =      32'h00AB0FFF;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
```

```
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 << req3_data_in[27:31];

    $display("Check that the high-order 27 bits are ignored in the second operand of shift left
command. (Port 4)");
    req4_cmd_in              =      Opcode;
    req4_data_in        =      32'hFFFFFFFF;
    data4               =      req4_data_in;
    cmd4                =      req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in              =      4'b0000;
    req4_data_in        =      32'h00AB0FFF;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 << req4_data_in[27:31];



    Opcode = Shift_Right;

    $display("Check that the high-order 27 bits are ignored in the second operand of shift right
command. (Port 1)");
    req1_cmd_in              =      Opcode;
    req1_data_in        =      32'hFFFFFFFF;
    data1               =      req1_data_in;
    cmd1                =      req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =      4'b0000;
    req1_data_in        =      32'h00AB0FFF;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 >> req1_data_in[27:31];

    $display("Check that the high-order 27 bits are ignored in the second operand of shift right
command. (Port 2)");
    req2_cmd_in              =      Opcode;
    req2_data_in        =      32'hFFFFFFFF;
    data2               =      req2_data_in;
    cmd2                =      req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in              =      4'b0000;
    req2_data_in        =      32'h00AB0FFF;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 >> req2_data_in[27:31];
```

```
    $display("Check that the high-order 27 bits are ignored in the second operand of shift right
command. (Port 3)");
    req3_cmd_in          =    Opcode;
    req3_data_in         =    32'hFFFFFFFF;
    data3                =    req3_data_in;
    cmd3                 =    req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in          =    4'b0000;
    req3_data_in         =    32'h00AB0FFF;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 >> req3_data_in[27:31];

    $display("Check that the high-order 27 bits are ignored in the second operand of shift right
command. (Port 4)");
    req4_cmd_in          =    Opcode;
    req4_data_in         =    32'hFFFFFFFF;
    data4                =    req4_data_in;
    cmd4                 =    req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in          =    4'b0000;
    req4_data_in         =    32'h00AB0FFF;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 >> req4_data_in[27:31];

  Opcode = Add;

    $display("Data dependent corner case: Add two numbers that overflow by 1 (â  œFFFFFFFFâ    X
+ 1). (Port 1)");
    req1_cmd_in          =    Opcode;
    req1_data_in         =    32'hFFFFFFFF;
    data1                =    req1_data_in;
    cmd1                 =    req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in          =    4'b0000;
    req1_data_in         =    32'h00000001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 + req1_data_in;

    $display("Data dependent corner case: Add two numbers that overflow by 1 (â  œFFFFFFFFâ    X
+ 1). (Port 2)");
    req2_cmd_in          =    Opcode;
    req2_data_in         =    32'hFFFFFFFF;
```

```
    data2               =     req2_data_in;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in              =     4'b0000;
    req2_data_in       =     32'h00000001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 + req2_data_in;

    $display("Data dependent corner case: Add two numbers that overflow by 1 (â  œFFFFFFFFâ    X
+ 1). (Port 3)");
    req3_cmd_in              =     Opcode;
    req3_data_in       =     32'hFFFFFFFF;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in              =     4'b0000;
    req3_data_in       =     32'h00000001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 + req3_data_in;

    $display("Data dependent corner case: Add two numbers that overflow by 1 (â  œFFFFFFFFâ    X
+ 1). (Port 4)");
    req4_cmd_in              =     Opcode;
    req4_data_in       =     32'hFFFFFFFF;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in              =     4'b0000;
    req4_data_in       =     32'h00000001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 + req4_data_in;


    $display("Data dependent corner case: Add two numbers whose sum is â  œFFFFFFFFâ    X. (Port
1)");
    req1_cmd_in              =     Opcode;
    req1_data_in       =     32'hEEEEFFFF;
    data1               =     req1_data_in;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =     4'b0000;
    req1_data_in       =     32'h11110000;
```

```
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 + req1_data_in;

    $display("Data dependent corner case: Add two numbers whose sum is â  œFFFFFFFFâ     X. (Port
2)");
    req2_cmd_in              =      Opcode;
    req2_data_in         =      32'hEEEEFFFF;
    data2                =      req2_data_in;
    cmd2                 =      req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in              =      4'b0000;
    req2_data_in         =      32'h11110000;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 + req2_data_in;

    $display("Data dependent corner case: Add two numbers whose sum is â  œFFFFFFFFâ     X. (Port
3)");
    req3_cmd_in              =      Opcode;
    req3_data_in         =      32'hEEEEFFFF;
    data3                =      req3_data_in;
    cmd3                 =      req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in              =      4'b0000;
    req3_data_in         =      32'h11110000;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 + req3_data_in;

    $display("Data dependent corner case: Add two numbers whose sum is â  œFFFFFFFFâ     X. (Port
4)");
    req4_cmd_in              =      Opcode;
    req4_data_in         =      32'hEEEEFFFF;
    data4                =      req4_data_in;
    cmd4                 =      req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in              =      4'b0000;
    req4_data_in         =      32'h11110000;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 + req4_data_in;
```

```
    Opcode = Sub;

    $display("Data dependent corner case: Subtract two equal numbers. (Port 1)");
    req1_cmd_in          =    Opcode;
    req1_data_in         =    32'h0000FFFF;
    data1                =    req1_data_in;
    cmd1                 =    req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in          =    4'b0000;
    req1_data_in         =    32'h0000FFFF;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 - req1_data_in;

    $display("Data dependent corner case: Subtract two equal numbers. (Port 2)");
    req2_cmd_in          =    Opcode;
    req2_data_in         =    32'h0000FFFF;
    data2                =    req2_data_in;
    cmd2                 =    req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in          =    4'b0000;
    req2_data_in         =    32'h0000FFFF;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 - req2_data_in;

    $display("Data dependent corner case: Subtract two equal numbers. (Port 3)");
    req3_cmd_in          =    Opcode;
    req3_data_in         =    32'h0000FFFF;
    data3                =    req3_data_in;
    cmd3                 =    req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in          =    4'b0000;
    req3_data_in         =    32'h0000FFFF;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 - req3_data_in;

    $display("Data dependent corner case: Subtract two equal numbers. (Port 4)");
    req4_cmd_in          =    Opcode;
    req4_data_in         =    32'h0000FFFF;
    data4                =    req4_data_in;
```

```
    cmd4                =      req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in             =      4'b0000;
    req4_data_in        =      32'h0000FFFF;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 - req4_data_in;

    $display("Data dependent corner case: Subtract a number that underflows by 1 (Operand2 is one
greater than Operand1). (Port 1)");
    req1_cmd_in             =      Opcode;
    req1_data_in        =      32'h00000000;
    data1               =      req1_data_in;
    cmd1                =      req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =      4'b0000;
    req1_data_in        =      32'h00000001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 - req1_data_in;

    $display("Data dependent corner case: Subtract a number that underflows by 1 (Operand2 is one
greater than Operand1). (Port 2)");
    req2_cmd_in             =      Opcode;
    req2_data_in        =      32'h00000000;
    data2               =      req2_data_in;
    cmd2                =      req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in             =      4'b0000;
    req2_data_in        =      32'h00000001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 - req2_data_in;

    $display("Data dependent corner case: Subtract a number that underflows by 1 (Operand2 is one
greater than Operand1). (Port 3)");
    req3_cmd_in             =      Opcode;
    req3_data_in        =      32'h00000000;
    data3               =      req3_data_in;
    cmd3                =      req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in             =      4'b0000;
    req3_data_in        =      32'h00000001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
```

```
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 - req3_data_in;

    $display("Data dependent corner case: Subtract a number that underflows by 1 (Operand2 is one
greater than Operand1). (Port 4)");
    req4_cmd_in          =    Opcode;
    req4_data_in         =    32'h00000000;
    data4                =    req4_data_in;
    cmd4                 =    req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in          =    4'b0000;
    req4_data_in         =    32'h00000001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 - req4_data_in;
    Opcode = Shift_Left;

    $display("Data dependent corner case: Shift Left 0 places (should return Operand1 unchanged).
(Port 1)");
    req1_cmd_in          =    Opcode;
    req1_data_in         =    32'hFFFFFFFF;
    data1                =    req1_data_in;
    cmd1                 =    req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in          =    4'b0000;
    req1_data_in         =    32'h00000000;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1;

    $display("Data dependent corner case: Shift Left 0 places (should return Operand1 unchanged).
(Port 2)");
    req2_cmd_in          =    Opcode;
    req2_data_in         =    32'hFFFFFFFF;
    data2                =    req2_data_in;
    cmd2                 =    req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in          =    4'b0000;
    req2_data_in         =    32'h00000000;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2;

    $display("Data dependent corner case: Shift Left 0 places (should return Operand1 unchanged).
(Port 3)");
```

```
    req3_cmd_in             =     Opcode;
    req3_data_in        =     32'hFFFFFFFF;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in             =     4'b0000;
    req3_data_in        =     32'h00000000;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3;

    $display("Data dependent corner case: Shift Left 0 places (should return Operand1 unchanged).
(Port 4)");
    req4_cmd_in             =     Opcode;
    req4_data_in        =     32'hFFFFFFFF;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in             =     4'b0000;
    req4_data_in        =     32'h00000000;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4;



    Opcode = Shift_Right;

    $display("Data dependent corner case: Shift Right 0 places (should return Operand1 unchanged).
(Port 1)");
    req1_cmd_in             =     Opcode;
    req1_data_in        =     32'hFFFFFFFF;
    data1               =     req1_data_in;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     4'b0000;
    req1_data_in        =     32'h00000000;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1;

    $display("Data dependent corner case: Shift Right 0 places (should return Operand1 unchanged).
(Port 2)");
    req2_cmd_in             =     Opcode;
    req2_data_in        =     32'hFFFFFFFF;
```

```
    data2               =     req2_data_in;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in              =     4'b0000;
    req2_data_in       =     32'h00000000;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2;

    $display("Data dependent corner case: Shift Right 0 places (should return Operand1 unchanged).
(Port 3)");
    req3_cmd_in              =     Opcode;
    req3_data_in       =     32'hFFFFFFFF;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in              =     4'b0000;
    req3_data_in       =     32'h00000000;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3;

    $display("Data dependent corner case: Shift Right 0 places (should return Operand1 unchanged).
(Port 4)");
    req4_cmd_in              =     Opcode;
    req4_data_in       =     32'hFFFFFFFF;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in              =     4'b0000;
    req4_data_in       =     32'h00000000;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4;

    Opcode = Shift_Left;

    $display("Data dependent corner case: Shift Left 31 places (the max allowable shift places). (Port
1)");
    req1_cmd_in              =     Opcode;
    req1_data_in       =     32'hFFFFFFFF;
    data1               =     req1_data_in;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =     4'b0000;
```

```
    req1_data_in        =     32'h0000001F;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 << 31;

    $display("Data dependent corner case: Shift Left 31 places (the max allowable shift places). (Port
2)");
    req2_cmd_in             =     Opcode;
    req2_data_in        =     32'hFFFFFFFF;
    data2               =     req2_data_in;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in             =     4'b0000;
    req2_data_in        =     32'h0000001F;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 << 31;

    $display("Data dependent corner case: Shift Left 31 places (the max allowable shift places). (Port
3)");
    req3_cmd_in             =     Opcode;
    req3_data_in        =     32'hFFFFFFFF;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in             =     4'b0000;
    req3_data_in        =     32'h0000001F;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 << 31;

    $display("Data dependent corner case: Shift Left 31 places (the max allowable shift places). (Port
4)");
    req4_cmd_in             =     Opcode;
    req4_data_in        =     32'hFFFFFFFF;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in             =     4'b0000;
    req4_data_in        =     32'h0000001F;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 << 31;
```

```
    Opcode = Shift_Right;

    $display("Data dependent corner case: Shift Right 31 places (the max allowable shift places).
(Port 1)");
    req1_cmd_in             =     Opcode;
    req1_data_in        =     32'hFFFFFFFF;
    data1               =     req1_data_in;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     4'b0000;
    req1_data_in        =     32'h0000001F;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 >> 31;

    $display("Data dependent corner case: Shift Right 31 places (the max allowable shift places).
(Port 2)");
    req2_cmd_in             =     Opcode;
    req2_data_in        =     32'hFFFFFFFF;
    data2               =     req2_data_in;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in             =     4'b0000;
    req2_data_in        =     32'h0000001F;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 >> 31;

    $display("Data dependent corner case: Shift Right 31 places (the max allowable shift places).
(Port 3)");
    req3_cmd_in             =     Opcode;
    req3_data_in        =     32'hFFFFFFFF;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in             =     4'b0000;
    req3_data_in        =     32'h0000001F;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 >> 31;

    $display("Data dependent corner case: Shift Right 31 places (the max allowable shift places).
(Port 4)");
    req4_cmd_in             =     Opcode;
    req4_data_in        =     32'hFFFFFFFF;
```

```
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in              =     4'b0000;
    req4_data_in       =     32'h0000001F;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 >> 31;


    Opcode = Add;

    $display("Data dependent corner case: Add max number â  œFFFFFFFFâ    X. (Port 1)");
    req1_cmd_in              =     Opcode;
    req1_data_in       =     32'hFFFFFFFF;
    data1               =     req1_data_in;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =     4'b0000;
    req1_data_in       =     32'hFFFFFFFF;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 + req1_data_in;

    $display("Data dependent corner case: Add max number â  œFFFFFFFFâ    X. (Port 2)");
    req2_cmd_in              =     Opcode;
    req2_data_in       =     32'hFFFFFFFF;
    data2               =     req2_data_in;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =     4'b0000;
    req1_data_in       =     32'hFFFFFFFF;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 + req2_data_in;

    $display("Data dependent corner case: Add max number â  œFFFFFFFFâ    X. (Port 3)");
    req3_cmd_in              =     Opcode;
    req3_data_in       =     32'hFFFFFFFF;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in              =     4'b0000;
    req3_data_in       =     32'hFFFFFFFF;
```

```
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 + req3_data_in;

    $display("Data dependent corner case: Add max number â  œFFFFFFFFâ    X. (Port 4)");
    req4_cmd_in              =      Opcode;
    req4_data_in        =      32'hFFFFFFFF;
    data4               =      req4_data_in;
    cmd4                =      req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in              =      4'b0000;
    req4_data_in        =      32'hFFFFFFFF;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 + req4_data_in;

    $display("Data dependent corner case: Add max number â  œFFFFFFFFâ    X with min number.
(Port 1)");
    req1_cmd_in              =      Opcode;
    req1_data_in        =      32'hFFFFFFFF;
    data1               =      req1_data_in;
    cmd1                =      req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =      4'b0000;
    req1_data_in        =      32'h00000000;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 + req1_data_in;

    $display("Data dependent corner case: Add max number â  œFFFFFFFFâ    X with min number.
(Port 2)");
    req2_cmd_in              =      Opcode;
    req2_data_in        =      32'hFFFFFFFF;
    data2               =      req2_data_in;
    cmd2                =      req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =      4'b0000;
    req1_data_in        =      32'h00000000;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 + req2_data_in;

    $display("Data dependent corner case: Add max number â  œFFFFFFFFâ    X with min number.
(Port 3)");
```

```
    req3_cmd_in             =     Opcode;
    req3_data_in        =     32'hFFFFFFFF;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in             =     4'b0000;
    req3_data_in        =     32'h00000000;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 + req3_data_in;

    $display("Data dependent corner case: Add max number â  œFFFFFFFFâ     X with min number.
(Port 4)");
    req4_cmd_in             =     Opcode;
    req4_data_in        =     32'hFFFFFFFF;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in             =     4'b0000;
    req4_data_in        =     32'h00000000;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 + req4_data_in;

    $display("Data dependent corner case: Add min number. (Port 1)");
    req1_cmd_in             =     Opcode;
    req1_data_in        =     32'h00000000;
    data1               =     req1_data_in;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     4'b0000;
    req1_data_in        =     32'h00000000;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 + req1_data_in;

    $display("Data dependent corner case: Add min number. (Port 2)");
    req2_cmd_in             =     Opcode;
    req2_data_in        =     32'h00000000;
    data2               =     req2_data_in;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     4'b0000;
    req1_data_in        =     32'h00000000;
```

```
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 + req2_data_in;

    $display("Data dependent corner case: Add min number. (Port 3)");
    req3_cmd_in          =    Opcode;
    req3_data_in     =    32'h00000000;
    data3            =    req3_data_in;
    cmd3             =    req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in          =    4'b0000;
    req3_data_in     =    32'h00000000;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 + req3_data_in;

    $display("Data dependent corner case: Add min number. (Port 4)");
    req4_cmd_in          =    Opcode;
    req4_data_in     =    32'h00000000;
    data4            =    req4_data_in;
    cmd4             =    req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in          =    4'b0000;
    req4_data_in     =    32'h00000000;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 + req4_data_in;

    Opcode = Sub;

    $display("Data dependent corner case: Subtract min number. (Port 1)");
    req1_cmd_in          =    Opcode;
    req1_data_in     =    32'h00000000;
    data1            =    req1_data_in;
    cmd1             =    req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in          =    4'b0000;
    req1_data_in     =    32'h00000000;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 - req1_data_in;

    $display("Data dependent corner case: Subtract min number. (Port 2)");
    req2_cmd_in          =    Opcode;
```

```
    req2_data_in        =     32'h00000000;
    data2               =     req2_data_in;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     4'b0000;
    req1_data_in        =     32'h00000000;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 - req2_data_in;

    $display("Data dependent corner case: Subtract min number. (Port 3)");
    req3_cmd_in             =     Opcode;
    req3_data_in        =     32'h00000000;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in             =     4'b0000;
    req3_data_in        =     32'h00000000;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 - req3_data_in;

    $display("Data dependent corner case: Subtract min number. (Port 4)");
    req4_cmd_in             =     Opcode;
    req4_data_in        =     32'h00000000;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in             =     4'b0000;
    req4_data_in        =     32'h00000000;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 - req4_data_in;

    $display("Data dependent corner case: Subtract max number. (Port 1)");
    req1_cmd_in             =     Opcode;
    req1_data_in        =     32'hFFFFFFFF;
    data1               =     req1_data_in;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     4'b0000;
    req1_data_in        =     32'hFFFFFFFF;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
```

```
    expected_data1 = data1 - req1_data_in;

    $display("Data dependent corner case: Subtract max number. (Port 2)");
    req2_cmd_in             =     Opcode;
    req2_data_in       =     32'hFFFFFFFF;
    data2              =     req2_data_in;
    cmd2               =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     4'b0000;
    req1_data_in       =     32'hFFFFFFFF;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 - req2_data_in;

    $display("Data dependent corner case: Subtract max number. (Port 3)");
    req3_cmd_in             =     Opcode;
    req3_data_in       =     32'hFFFFFFFF;
    data3              =     req3_data_in;
    cmd3               =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in             =     4'b0000;
    req3_data_in       =     32'hFFFFFFFF;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 - req3_data_in;

    $display("Data dependent corner case: Subtract max number. (Port 4)");
    req4_cmd_in             =     Opcode;
    req4_data_in       =     32'hFFFFFFFF;
    data4              =     req4_data_in;
    cmd4               =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in             =     4'b0000;
    req4_data_in       =     32'hFFFFFFFF;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 - req4_data_in;

    $display("Data dependent corner case: Subtract max and min numbers. (Port 1)");
    req1_cmd_in             =     Opcode;
    req1_data_in       =     32'h00000000;
    data1              =     req1_data_in;
    cmd1               =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     4'b0000;
```

```
    req1_data_in        =     32'hFFFFFFFF;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 - req1_data_in;

    $display("Data dependent corner case: Subtract max and min numbers. (Port 2)");
    req2_cmd_in             =     Opcode;
    req2_data_in        =     32'h00000000;
    data2               =     req2_data_in;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     4'b0000;
    req1_data_in        =     32'hFFFFFFFF;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 - req2_data_in;

    $display("Data dependent corner case: Subtract max and min numbers. (Port 3)");
    req3_cmd_in             =     Opcode;
    req3_data_in        =     32'h00000000;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in             =     4'b0000;
    req3_data_in        =     32'hFFFFFFFF;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 - req3_data_in;

    $display("Data dependent corner case: Subtract max and min numbers. (Port 4)");
    req4_cmd_in             =     Opcode;
    req4_data_in        =     32'h00000000;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in             =     4'b0000;
    req4_data_in        =     32'hFFFFFFFF;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 - req4_data_in;

    Opcode = Add;

    $display("Invalid Input Data (Port 1)");
```

```
    req1_cmd_in             =     Opcode;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     4'b0000;
    req1_data_in        =     32'h0000000F;
    data1               =     req1_data_in;
    repeat(1)@(posedge c_clk);
    req1_data_in        =     32'h0000000F;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 + req1_data_in;

    $display("Invalid Input Data (Port 2)");
    req2_cmd_in             =     Opcode;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in             =     4'b0000;
    req2_data_in        =     32'h0000000F;
    data2               =     req2_data_in;
    repeat(1)@(posedge c_clk);
    req2_data_in        =     32'h0000000F;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 + req2_data_in;

    $display("Invalid Input Data (Port 3)");
    req3_cmd_in             =     Opcode;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in             =     4'b0000;
    req3_data_in        =     32'h0000000F;
    data3               =     req3_data_in;
    repeat(1)@(posedge c_clk);
    req3_data_in        =     32'h0000000F;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 + req3_data_in;

    $display("Invalid Input Data (Port 4)");
    req4_cmd_in             =     Opcode;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in             =     4'b0000;
    req4_data_in        =     32'h0000000F;
    data4               =     req4_data_in;
```

```
    repeat(1)@(posedge c_clk);
    req4_data_in        =      32'h0000000F;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 + req4_data_in;

    $display("Invalid Input Data 2 (Port 1)");
    req1_data_in = 32'h0000000F;
    data1               =      req1_data_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =      Opcode;
    cmd1                =      req1_cmd_in;
    req1_data_in        =      32'h0000000F;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =      4'b0000;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
    expected_data1 = data1 + req1_data_in;

    $display("Invalid Input Data 2 (Port 2)");
    req2_data_in = 32'h0000000F;
    data2               =      req2_data_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in              =      Opcode;
    cmd2                =      req2_cmd_in;
    req2_data_in        =      32'h0000000F;
    repeat(1)@(posedge c_clk);
    req2_cmd_in              =      4'b0000;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
    expected_data2 = data2 + req2_data_in;

    $display("Invalid Input Data 2 (Port 3)");
    req3_data_in = 32'h0000000F;
    data3               =      req3_data_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in              =      Opcode;
    cmd3                =      req3_cmd_in;
    req3_data_in        =      32'h0000000F;
    repeat(1)@(posedge c_clk);
    req3_cmd_in              =      4'b0000;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);
    expected_data3 = data3 + req3_data_in;
```

```
    $display("Invalid Input Data 2 (Port 4)");
    req4_data_in = 32'h0000000F;
    data4               =     req4_data_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in              =     Opcode;
    cmd4               =     req4_cmd_in;
    req4_data_in       =     32'h0000000F;
    repeat(1)@(posedge c_clk);
    req4_cmd_in              =     4'b0000;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);
    expected_data4 = data4 + req4_data_in;

    Opcode = 4'b0011;

    $display("Illegal Input Command: 0011 (Port 1)");
    req1_cmd_in              =     Opcode;
    req1_data_in       =     32'h0010F011;
    data1              =     req1_data_in;
    cmd1               =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =     4'b0000;
    req1_data_in       =     32'h00A0C001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 0011 (Port 2)");
    req2_cmd_in              =     Opcode;
    req2_data_in       =     32'h0010F011;
    data2              =     req2_data_in;
    cmd2               =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in              =     4'b0000;
    req2_data_in       =     32'h00A0C001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 0011 (Port 3)");
    req3_cmd_in              =     Opcode;
    req3_data_in       =     32'h0010F011;
    data3              =     req3_data_in;
    cmd3               =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in              =     4'b0000;
```

```
    req3_data_in        =     32'h00A0C001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 0011 (Port 4)");
    req4_cmd_in              =     Opcode;
    req4_data_in        =     32'h0010F011;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in              =     4'b0000;
    req4_data_in        =     32'h00A0C001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);

    Opcode = 4'b0100;

    $display("Illegal Input Command: 0100 (Port 1)");
    req1_cmd_in              =     Opcode;
    req1_data_in        =     32'h0010F011;
    data1               =     req1_data_in;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =     4'b0000;
    req1_data_in        =     32'h00A0C001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 0100 (Port 2)");
    req2_cmd_in              =     Opcode;
    req2_data_in        =     32'h0010F011;
    data2               =     req2_data_in;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in              =     4'b0000;
    req2_data_in        =     32'h00A0C001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 0100 (Port 3)");
    req3_cmd_in              =     Opcode;
    req3_data_in        =     32'h0010F011;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
```

```
    repeat(1)@(posedge c_clk);
    req3_cmd_in             =     4'b0000;
    req3_data_in        =     32'h00A0C001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 0100 (Port 4)");
    req4_cmd_in             =     Opcode;
    req4_data_in        =     32'h0010F011;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in             =     4'b0000;
    req4_data_in        =     32'h00A0C001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);

    Opcode = 4'b0100;

    $display("Illegal Input Command: 0100 (Port 1)");
    req1_cmd_in             =     Opcode;
    req1_data_in        =     32'h0010F011;
    data1               =     req1_data_in;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     4'b0000;
    req1_data_in        =     32'h00A0C001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 0100 (Port 2)");
    req2_cmd_in             =     Opcode;
    req2_data_in        =     32'h0010F011;
    data2               =     req2_data_in;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in             =     4'b0000;
    req2_data_in        =     32'h00A0C001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 0100 (Port 3)");
    req3_cmd_in             =     Opcode;
    req3_data_in        =     32'h0010F011;
```

```
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in              =     4'b0000;
    req3_data_in        =     32'h00A0C001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 0100 (Port 4)");
    req4_cmd_in              =     Opcode;
    req4_data_in        =     32'h0010F011;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in              =     4'b0000;
    req4_data_in        =     32'h00A0C001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);

    Opcode = 4'b0111;

    $display("Illegal Input Command: 0111 (Port 1)");
    req1_cmd_in              =     Opcode;
    req1_data_in        =     32'h0010F011;
    data1               =     req1_data_in;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in              =     4'b0000;
    req1_data_in        =     32'h00A0C001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 0111 (Port 2)");
    req2_cmd_in              =     Opcode;
    req2_data_in        =     32'h0010F011;
    data2               =     req2_data_in;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in              =     4'b0000;
    req2_data_in        =     32'h00A0C001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 0111 (Port 3)");
```

```
    req3_cmd_in             =     Opcode;
    req3_data_in        =     32'h0010F011;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in             =     4'b0000;
    req3_data_in        =     32'h00A0C001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 0111 (Port 4)");
    req4_cmd_in             =     Opcode;
    req4_data_in        =     32'h0010F011;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in             =     4'b0000;
    req4_data_in        =     32'h00A0C001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);

    Opcode = 4'b1000;

    $display("Illegal Input Command: 1000 (Port 1)");
    req1_cmd_in             =     Opcode;
    req1_data_in        =     32'h0010F011;
    data1               =     req1_data_in;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in             =     4'b0000;
    req1_data_in        =     32'h00A0C001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1000 (Port 2)");
    req2_cmd_in             =     Opcode;
    req2_data_in        =     32'h0010F011;
    data2               =     req2_data_in;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in             =     4'b0000;
    req2_data_in        =     32'h00A0C001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);
```

```
    $display("Illegal Input Command: 1000 (Port 3)");
    req3_cmd_in         =   Opcode;
    req3_data_in        =   32'h0010F011;
    data3               =   req3_data_in;
    cmd3                =   req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in         =   4'b0000;
    req3_data_in        =   32'h00A0C001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1000 (Port 4)");
    req4_cmd_in         =   Opcode;
    req4_data_in        =   32'h0010F011;
    data4               =   req4_data_in;
    cmd4                =   req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in         =   4'b0000;
    req4_data_in        =   32'h00A0C001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);

    Opcode = 4'b1001;

    $display("Illegal Input Command: 1001 (Port 1)");
    req1_cmd_in         =   Opcode;
    req1_data_in        =   32'h0010F011;
    data1               =   req1_data_in;
    cmd1                =   req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in         =   4'b0000;
    req1_data_in        =   32'h00A0C001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1001 (Port 2)");
    req2_cmd_in         =   Opcode;
    req2_data_in        =   32'h0010F011;
    data2               =   req2_data_in;
    cmd2                =   req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in         =   4'b0000;
    req2_data_in        =   32'h00A0C001;
```

```
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1001 (Port 3)");
    req3_cmd_in          =    Opcode;
    req3_data_in     =    32'h0010F011;
    data3            =    req3_data_in;
    cmd3             =    req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in          =    4'b0000;
    req3_data_in     =    32'h00A0C001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1001 (Port 4)");
    req4_cmd_in          =    Opcode;
    req4_data_in     =    32'h0010F011;
    data4            =    req4_data_in;
    cmd4             =    req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in          =    4'b0000;
    req4_data_in     =    32'h00A0C001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);

    Opcode = 4'b1010;

    $display("Illegal Input Command: 1010 (Port 1)");
    req1_cmd_in          =    Opcode;
    req1_data_in     =    32'h0010F011;
    data1            =    req1_data_in;
    cmd1             =    req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in          =    4'b0000;
    req1_data_in     =    32'h00A0C001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1010 (Port 2)");
    req2_cmd_in          =    Opcode;
    req2_data_in     =    32'h0010F011;
    data2            =    req2_data_in;
    cmd2             =    req2_cmd_in;
    repeat(1)@(posedge c_clk);
```

```
    req2_cmd_in                =      4'b0000;
    req2_data_in         =      32'h00A0C001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1010 (Port 3)");
    req3_cmd_in                =      Opcode;
    req3_data_in         =      32'h0010F011;
    data3                =      req3_data_in;
    cmd3                 =      req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in                =      4'b0000;
    req3_data_in         =      32'h00A0C001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1010 (Port 4)");
    req4_cmd_in                =      Opcode;
    req4_data_in         =      32'h0010F011;
    data4                =      req4_data_in;
    cmd4                 =      req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in                =      4'b0000;
    req4_data_in         =      32'h00A0C001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);

    Opcode = 4'b1011;

    $display("Illegal Input Command: 1011 (Port 1)");
    req1_cmd_in                =      Opcode;
    req1_data_in         =      32'h0010F011;
    data1                =      req1_data_in;
    cmd1                 =      req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in                =      4'b0000;
    req1_data_in         =      32'h00A0C001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1011 (Port 2)");
    req2_cmd_in                =      Opcode;
    req2_data_in         =      32'h0010F011;
    data2                =      req2_data_in;
```

```
    cmd2                =      req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in         =      4'b0000;
    req2_data_in        =      32'h00A0C001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1011 (Port 3)");
    req3_cmd_in         =      Opcode;
    req3_data_in        =      32'h0010F011;
    data3               =      req3_data_in;
    cmd3                =      req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in         =      4'b0000;
    req3_data_in        =      32'h00A0C001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1011 (Port 4)");
    req4_cmd_in         =      Opcode;
    req4_data_in        =      32'h0010F011;
    data4               =      req4_data_in;
    cmd4                =      req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in         =      4'b0000;
    req4_data_in        =      32'h00A0C001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);

    Opcode = 4'b1100;

    $display("Illegal Input Command: 1100 (Port 1)");
    req1_cmd_in         =      Opcode;
    req1_data_in        =      32'h0010F011;
    data1               =      req1_data_in;
    cmd1                =      req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in         =      4'b0000;
    req1_data_in        =      32'h00A0C001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1100 (Port 2)");
    req2_cmd_in         =    Opcode;
```

```
    req2_data_in        =     32'h0010F011;
    data2               =     req2_data_in;
    cmd2                =     req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in         =     4'b0000;
    req2_data_in        =     32'h00A0C001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1100 (Port 3)");
    req3_cmd_in         =     Opcode;
    req3_data_in        =     32'h0010F011;
    data3               =     req3_data_in;
    cmd3                =     req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in         =     4'b0000;
    req3_data_in        =     32'h00A0C001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1100 (Port 4)");
    req4_cmd_in         =     Opcode;
    req4_data_in        =     32'h0010F011;
    data4               =     req4_data_in;
    cmd4                =     req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in         =     4'b0000;
    req4_data_in        =     32'h00A0C001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);

    Opcode = 4'b1101;

    $display("Illegal Input Command: 1101 (Port 1)");
    req1_cmd_in         =     Opcode;
    req1_data_in        =     32'h0010F011;
    data1               =     req1_data_in;
    cmd1                =     req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in         =     4'b0000;
    req1_data_in        =     32'h00A0C001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);
```

```
    $display("Illegal Input Command: 1101 (Port 2)");
    req2_cmd_in          =    Opcode;
    req2_data_in         =    32'h0010F011;
    data2                =    req2_data_in;
    cmd2                 =    req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in          =    4'b0000;
    req2_data_in         =    32'h00A0C001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1101 (Port 3)");
    req3_cmd_in          =    Opcode;
    req3_data_in         =    32'h0010F011;
    data3                =    req3_data_in;
    cmd3                 =    req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in          =    4'b0000;
    req3_data_in         =    32'h00A0C001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1101 (Port 4)");
    req4_cmd_in          =    Opcode;
    req4_data_in         =    32'h0010F011;
    data4                =    req4_data_in;
    cmd4                 =    req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in          =    4'b0000;
    req4_data_in         =    32'h00A0C001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);

    Opcode = 4'b1110;

    $display("Illegal Input Command: 1110 (Port 1)");
    req1_cmd_in          =    Opcode;
    req1_data_in         =    32'h0010F011;
    data1                =    req1_data_in;
    cmd1                 =    req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in          =    4'b0000;
    req1_data_in         =    32'h00A0C001;
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
```

```
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1110 (Port 2)");
    req2_cmd_in          =    Opcode;
    req2_data_in         =    32'h0010F011;
    data2                =    req2_data_in;
    cmd2                 =    req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in          =    4'b0000;
    req2_data_in         =    32'h00A0C001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1110 (Port 3)");
    req3_cmd_in          =    Opcode;
    req3_data_in         =    32'h0010F011;
    data3                =    req3_data_in;
    cmd3                 =    req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in          =    4'b0000;
    req3_data_in         =    32'h00A0C001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1110 (Port 4)");
    req4_cmd_in          =    Opcode;
    req4_data_in         =    32'h0010F011;
    data4                =    req4_data_in;
    cmd4                 =    req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in          =    4'b0000;
    req4_data_in         =    32'h00A0C001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);

    Opcode = 4'b1111;

    $display("Illegal Input Command: 1111 (Port 1)");
    req1_cmd_in          =    Opcode;
    req1_data_in         =    32'h0010F011;
    data1                =    req1_data_in;
    cmd1                 =    req1_cmd_in;
    repeat(1)@(posedge c_clk);
    req1_cmd_in          =    4'b0000;
    req1_data_in         =    32'h00A0C001;
```

```
    $display("time = %t Port 1: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data1, req1_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1111 (Port 2)");
    req2_cmd_in              =      Opcode;
    req2_data_in        =      32'h0010F011;
    data2                =      req2_data_in;
    cmd2                 =      req2_cmd_in;
    repeat(1)@(posedge c_clk);
    req2_cmd_in              =      4'b0000;
    req2_data_in        =      32'h00A0C001;
    $display("time = %t Port 2: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data2, req2_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1111 (Port 3)");
    req3_cmd_in              =      Opcode;
    req3_data_in        =      32'h0010F011;
    data3                =      req3_data_in;
    cmd3                 =      req3_cmd_in;
    repeat(1)@(posedge c_clk);
    req3_cmd_in              =      4'b0000;
    req3_data_in        =      32'h00A0C001;
    $display("time = %t Port 3: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data3, req3_data_in);
    repeat(1)@(posedge c_clk);

    $display("Illegal Input Command: 1111 (Port 4)");
    req4_cmd_in              =      Opcode;
    req4_data_in        =      32'h0010F011;
    data4                =      req4_data_in;
    cmd4                 =      req4_cmd_in;
    repeat(1)@(posedge c_clk);
    req4_cmd_in              =      4'b0000;
    req4_data_in        =      32'h00A0C001;
    $display("time = %t Port 4: Input Command = %b, Operand 1 =  %h, Operand 2 =  %h", $time,
Opcode, data4, req4_data_in);
    repeat(1)@(posedge c_clk);

    $display("Reset Check");
    repeat(1)@(posedge c_clk);
    reset[1:7]     =      7'b1111111;

    req1_cmd_in    = 4'b0000;
    req2_cmd_in    = 4'b0000;
    req3_cmd_in    =4'b0000;
    req4_cmd_in    = 4'b0000;
```

```
    req1_data_in  =     32'h00000000;
    req2_data_in  =     32'h00000000;
    req3_data_in  =     32'h00000000;
    req4_data_in  =     32'h00000000;

    repeat(7)@(posedge c_clk);
    reset[1:7] =   ~reset[1:7];

    repeat(1)@(posedge c_clk);


    $display("%t: At end of test error count is %0d and correct count = %0d", $time, error_count,
correct_count);
    $finish;

  end

  always @(out_resp1, out_resp2, out_resp3, out_resp4) begin

    if((cmd1 == Add) && (req1_data_in > (Max - data1))) begin
      overflow_check1 = 1;
    end
    else begin
      overflow_check1 = 0;
    end
    if((cmd2 == Add) && (req2_data_in > (Max - data2))) begin
      overflow_check2 = 1;
    end
    else begin
      overflow_check2 = 0;
    end
    if((cmd3 == Add) && (req3_data_in > (Max - data3))) begin
      overflow_check3 = 1;
    end
    else begin
      overflow_check3 = 0;
    end
    if((cmd4 == Add) && (req4_data_in > (Max - data4))) begin
      overflow_check4 = 1;
    end
    else begin
      overflow_check4 = 0;
    end
    if((cmd1 == Sub) && (req1_data_in > data1)) begin
      underflow_check1 = 1;
    end
    else begin
      underflow_check1 = 0;
```

```
    end
    if((cmd2 == Sub) && (req2_data_in > data2)) begin
      underflow_check2 = 1;
    end
    else begin
      underflow_check2 = 0;
    end
    if((cmd3 == Sub) && (req3_data_in > data3)) begin
      underflow_check3 = 1;
    end
    else begin
      underflow_check3 = 0;
    end
    if((cmd4 == Sub) && (req4_data_in > data4)) begin
      underflow_check4 = 1;
    end
    else begin
      underflow_check4 = 0;
    end

    if (out_resp1 == Success) begin
       if (cmd1 == No_Op) begin
        error_count = error_count + 1;
        $display("time = %t Error at Port 1: out_resp1 should be No Response ('00'b)", $time);
       end
       else if (cmd1 == Add || cmd1 == Sub || cmd1 == Shift_Left || cmd1 == Shift_Right) begin
            if (overflow_check3 == 1) begin
           error_count = error_count + 1;
           $display("time = %t Input Command = %b: Error at Port 1: Overflow but the response is
Success ('01'b)", $time, cmd1);
            end
            else if (underflow_check1 == 1) begin
            error_count = error_count + 1;
           $display("time = %t Input Command = %b: Error at Port 1: Underflow but the response is
Success ('01'b)", $time, cmd1);
            end
            else if (out_data1 == expected_data1) begin
           correct_count = correct_count + 1;
           $display("time = %t At port 1 Input Command = %b: Correct: Output (%h) is equal to
Expected Result (%h)", $time, cmd1, out_data1, expected_data1);
            end
            else begin
           error_count = error_count + 1;
           $display("time = %t Input Command = %b: Error at Port 1 Output (%h) is not equal to
Expected Result (%h)", $time, cmd1, out_data1, expected_data1);
            end
         end
       else begin
```

```
        error_count = error_count + 1;
        $display("time = %t Error at Port 1: Illegal command but the response is Success ('01'b)",
$time);
      end
    end
    else if (out_resp1 == 2'b10) begin
      if (out_data3 != Min) begin
        error_count = error_count + 1;
        $display("time = %t Error at Port 1: Output data is not zero", $time, cmd1);
      end
      if (cmd1 == Add) begin
        if (overflow_check1 == 1) begin
          correct_count = correct_count + 1;
          $display("time = %t Overflow at Port 1", $time);
        end
        else begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 1: No overflow but the response is
'10'b", $time, cmd1);
        end
      end
      else if (cmd1 == Sub) begin
        if (underflow_check1 == 1) begin
          correct_count = correct_count + 1;
          $display("time = %t Underflow at Port 1", $time);
        end
        else begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 1: No underflow but the response is
'10'b", $time, cmd1);
        end
      end
      else if (cmd1 == Shift_Left || cmd1 == Shift_Right) begin
        error_count = error_count + 1;
        $display("time = %t Input Command = %b: Error at Port 1: Shift command but the response is
'10'b", $time, cmd1);
      end
      else if (cmd1 != No_Op && cmd1 != Add && cmd1 != Sub && cmd1 != Shift_Left && cmd1 !=
Shift_Right) begin
        correct_count = correct_count + 1;
        $display("time = %t Input Command = %b: Illegal input command at Port 3", $time, cmd1);
      end
     end


    if (out_resp2 == Success) begin
      if (cmd2 == No_Op) begin
        error_count = error_count + 1;
```

```
        $display("time = %t Error at Port 2: out_resp2 should be No Response ('00'b)", $time);
      end
      else if (cmd2 == Add || cmd2 == Sub || cmd2 == Shift_Left || cmd2 == Shift_Right) begin
        if (overflow_check2 == 1) begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 2: Overflow but the response is
Success ('01'b)", $time, cmd2);
        end
        else if (underflow_check2 == 1) begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 2: Underflow but the response is
Success ('01'b)", $time, cmd2);
        end
        else if (out_data2 == expected_data2) begin
          correct_count = correct_count + 1;
          $display("time = %t At port 2 Input Command = %b: Correct: Output (%h) is equal to
Expected Result (%h)", $time, cmd2, out_data2, expected_data2);
        end
        else begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 2: Output (%h) is not equal to
Expected Result (%h)", $time, cmd2, out_data2, expected_data2);
        end
      end
      else begin
        error_count = error_count + 1;
        $display("time = %t Error at Port 2: Illegal command but the response is Success ('01'b)",
$time);
      end
    end
    else if (out_resp2 == 2'b10) begin
      if (out_data2 != Min) begin
        error_count = error_count + 1;
        $display("time = %t Error at Port 2: Output data is not zero", $time, cmd2);
      end
      if (cmd2 == Add) begin
        if (overflow_check2 == 1) begin
          correct_count = correct_count + 1;
          $display("time = %t Overflow at Port 2", $time);
        end
        else
          error_count = error_count + 1;
        $display("time = %t Input Command = %b: Error at Port 2: No overflow but the response is
'10'b", $time, cmd2);
      end
      else if (cmd2 == Sub) begin
        if (underflow_check2 == 1) begin
          correct_count = correct_count + 1;
```

```
          $display("time = %t Underflow at Port 2", $time);
        end
        else begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 2: No underflow but the response is
'10'b", $time, cmd2);
        end
      end
      else if (cmd2 == Shift_Left || cmd2 == Shift_Right) begin
        error_count = error_count + 1;
        $display("time = %t Input Command = %b: Error at Port 2: Shift command but the response is
'10'b", $time, cmd2);
      end
      else if (cmd2 != No_Op && cmd2 != Add && cmd2 != Sub && cmd2 != Shift_Left && cmd2 !=
Shift_Right) begin
        correct_count = correct_count + 1;
        $display("time = %t Input Command = %b: Illegal input command at Port 2", $time, cmd2);
      end
    end

    if (out_resp3 == Success) begin
      if (cmd3 == No_Op) begin
        error_count = error_count + 1;
        $display("time = %t Error at Port 3: out_resp3 should be No Response ('00'b)", $time);
      end
      else if (cmd3 == Add || cmd3 == Sub || cmd3 == Shift_Left || cmd3 == Shift_Right) begin
        if (overflow_check3 == 1) begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 3: Overflow but the response is
Success ('01'b)", $time, cmd3);
        end
        else if (underflow_check3 == 1) begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 3: Underflow but the response is
Success ('01'b)", $time, cmd3);
        end
        else if (out_data3 == expected_data3) begin
          correct_count = correct_count + 1;
          $display("time = %t At port 3 Input Command = %b: Correct: Output (%h) is equal to
Expected Result (%h)", $time, cmd3, out_data3, expected_data3);
        end
        else begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 3: Output (%h) is not equal to
Expected Result (%h)", $time, cmd3, out_data3, expected_data3);
        end
      end
      else begin
```

```
        error_count = error_count + 1;
        $display("time = %t Error at Port 3: Illegal command but the response is Success ('01'b)",
$time);
      end
    end
    else if (out_resp3 == 2'b10) begin
      if (out_data3 != Min) begin
        error_count = error_count + 1;
        $display("time = %t Error at Port 3: Output data is not zero", $time, cmd3);
      end
      if (cmd3 == Add) begin
        if (overflow_check3 == 1) begin
          correct_count = correct_count + 1;
          $display("time = %t Overflow at Port 3", $time);
        end
        else begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 3: No overflow but the response is
'10'b", $time, cmd3);
        end
      end
      else if (cmd3 == Sub) begin
        if (underflow_check3 == 1) begin
          correct_count = correct_count + 1;
          $display("time = %t Underflow at Port 3", $time);
        end
        else begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 3: No underflow but the response is
'10'b", $time, cmd3);
        end
      end
      else if (cmd3 == Shift_Left || cmd3 == Shift_Right) begin
        error_count = error_count + 1;
        $display("time = %t Input Command = %b: Error at Port 3: Shift command but the response is
'10'b", $time, cmd3);
      end
      else if (cmd3 != No_Op && cmd3 != Add && cmd3 != Sub && cmd3 != Shift_Left && cmd3 !=
Shift_Right) begin
        correct_count = correct_count + 1;
        $display("time = %t Input Command = %b: Illegal input command at Port 3", $time, cmd3);
      end
    end

    if (out_resp4 == Success) begin
      if (cmd4 == No_Op) begin
        error_count = error_count + 1;
        $display("time = %t Error at Port 4: out_resp4 should be No Response ('00'b)", $time);
```

```
      end
      else if (cmd4 == Add || cmd4 == Sub || cmd4 == Shift_Left || cmd4 == Shift_Right) begin
        if (overflow_check4 == 1) begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 4: Overflow but the response is
Success ('01'b)", $time, cmd4);
        end
        else if (underflow_check4 == 1) begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 4: Underflow but the response is
Success ('01'b)", $time, cmd4);
        end
        else if (out_data4 == expected_data4) begin
          correct_count = correct_count + 1;
          $display("time = %t At port 4 Input Command = %b: Correct: Output (%h) is equal to
Expected Result (%h)", $time, cmd4, out_data4, expected_data4);
        end
        else begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 4: Output (%h) is not equal to
Expected Result (%h)", $time, cmd4, out_data4, expected_data4);
        end
      end
      else begin
        error_count = error_count + 1;
        $display("time = %t Error at Port 4: Illegal command but the response is Success ('01'b)",
$time);
      end
    end
    else if (out_resp4 == 2'b10) begin
      if (out_data4 != Min) begin
        error_count = error_count + 1;
        $display("time = %t Error at Port 4: Output data is not zero", $time, cmd4);
      end
      if (cmd4 == Add) begin
        if (overflow_check4 == 1) begin
          correct_count = correct_count + 1;
          $display("time = %t Overflow at Port 4", $time);
        end
        else begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 4: No overflow but the response is
'10'b", $time, cmd4);
        end
      end
      else if (cmd4 == Sub) begin
        if (underflow_check4 == 1) begin
          correct_count = correct_count + 1;
```

```
          $display("time = %t Underflow at Port 4", $time);
        end
        else begin
          error_count = error_count + 1;
          $display("time = %t Input Command = %b: Error at Port 4: No underflow but the response is
'10'b", $time, cmd4);
        end
      end
      else if (cmd4 == Shift_Left || cmd4 == Shift_Right) begin
        error_count = error_count + 1;
        $display("time = %t Input Command = %b: Error at Port 4: Shift command but the response is
'10'b", $time, cmd4);
      end
      else if (cmd4 != No_Op && cmd4 != Add && cmd4 != Sub && cmd4 != Shift_Left && cmd4 !=
Shift_Right) begin
        correct_count = correct_count + 1;
        $display("time = %t Input Command = %b: Illegal input command at Port 4", $time, cmd4);
      end
    end
  end
endmodule
```