Calculator Overview

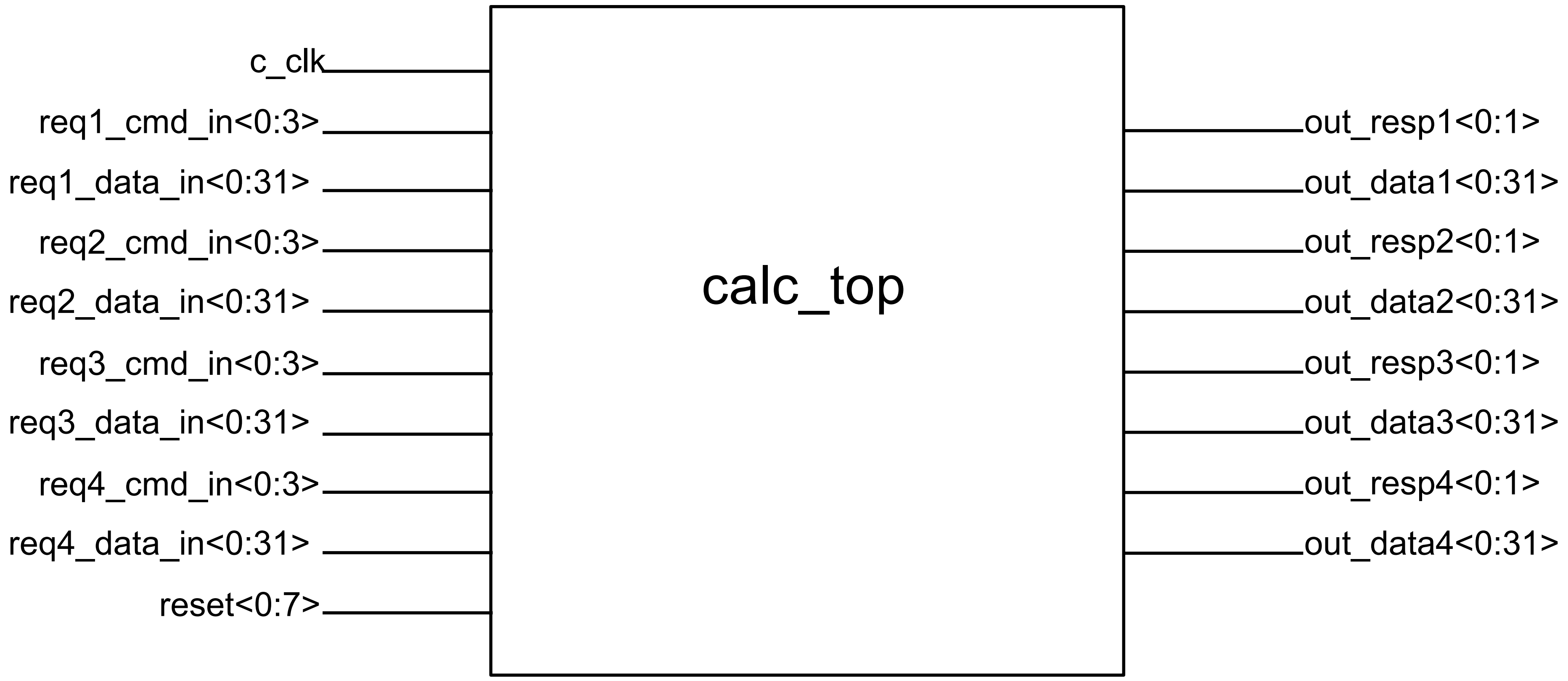
Functional Verification

Calculator Design

- Calculator has 4 functions:
  - Add
  - Subtract
  - Shift left
  - Shift right
- Calculator can handle 4 requests in parallel
  - All 4 requestors use separate input signals
  - All requestors have equal priority
  - Each port must wait for its response prior to sending the next command

Calculator design

- Input/Output description



Calculator Design

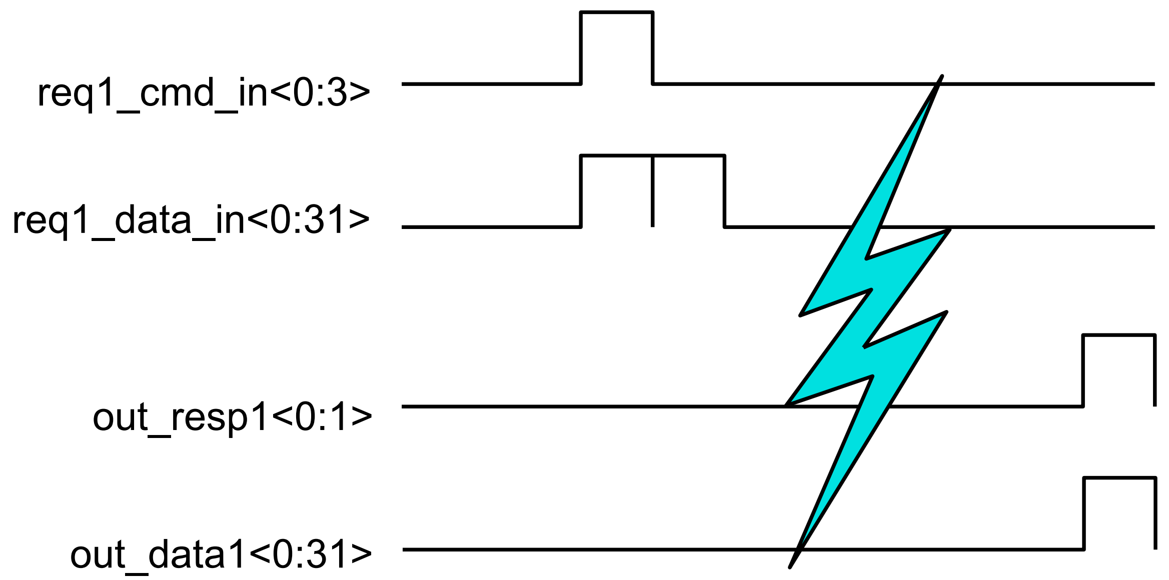
- I/O Description
  - Input commands:
    - 0 - No-op
    - 1 - Add operand1 and operand2
    - 2 - Subtract operand2 from operand1
    - 5 - Shift left operand1 by operand2 places
    - 6 - Shift right operand1 by operand2 places
  - Input Data
    - Operand1 data arrives with command
    - Operand2 data arrives on the following cycle

Calculator Design

- Outputs
  - Response line definition
    - 0 - no response
    - 1 - successful operation completion
    - 2 - invalid command or overflow/underflow error
    - 3 - Internal error
  - Data
    - Valid result data on output lines accompanies response (same cycle)

Calculator Design

- Input/Output timing



Each port must wait for its response prior to sending the next command!

Calculator Design

- Other information
  - Clocking
    - When using a cycle simulator, the clock should be held high (c_clk in the calculator model)
    - The clock should be toggled when using an event simulator
  - Calculator priority logic
    - Priority logic works on first come first serve algorithm
    - Priority logic allows for 1 add or subtract at a time and one shift operation at a time

Calculator Design

- Other information (con't)
  - Resets
    - Hold reset(1:7) to '1111111'b at start of testcase for seven cycles.
    - During the reset period, outputs of the calculator should be ignored
  - Shift operation
    - Only the low order 5 bits of the second operand are used
  - Arithmetic operations are unsigned

Creating the Verification Plan for Calc1

Calc1 basic function tests

Test reference number	Test description
1.1	Check the basic command-response protocol on each of the four ports.
1.2	Check the basic operation of each command on each port.
1.3	Check overflow and underflow cases for add and subtract commands.

Creating the Verification Plan for Calc1

Calc1 advanced function tests

Test reference Number	Test Description
2.1.1	For each port, check that each command can have any command follow it without leaving the state of the design dirty, such that the following command is corrupted.
2.1.2	Across all ports (e.g., four concurrent adds do not interfere with each other), check that each command can have any command follow it without leaving the state of the design dirty, such that the following command is corrupted.
2.2	Check that there is fairness across all four ports such that no port has higher priority than the others.
2.3	Check that the high-order 27 bits are ignored in the second operand of both shift commands.

Creating the Verification Plan for Calc1

Calc1 advanced function tests

Test reference Number	Test Description
2.4.1	Data dependent corner case: Add two numbers that overflow by 1 (“FFFFFFFF”X + 1).
2.4.2	Data dependent corner case: Add two numbers whose sum is “FFFFFFFF”X.
2.4.3	Data dependent corner case: Subtract two equal numbers.
2.4.4	Data dependent corner case: Subtract a number that underflows by 1 (Operand2 is one greater than Operand1).
2.4.5	Data dependent corner case: Shift 0 places (should return Operand1 unchanged).
2.4.6	Data dependent corner case: Shift 31 places (the max allowable shift places).
2.5	Check that the design ignores data inputs unless the data are supposed to be valid (concurrent with the command and the following cycle). Remember that “00000000”X is a data value just as any other 32- bit combination. Here, the check must include verifying that the design latches the data only when appropriate, and does not key off nonzero data.

Creating the Verification Plan for Calc1

Calc1 Generic tests and checks

Test reference number	**Test description**
3.1	Check that the design correctly handles illegal commands.
3.2	Check all outputs all of the time. Calc1 should not generate superfluous output values.
3.3	Check that the reset function correctly resets the design.