# 2 SYSTEM SPECS

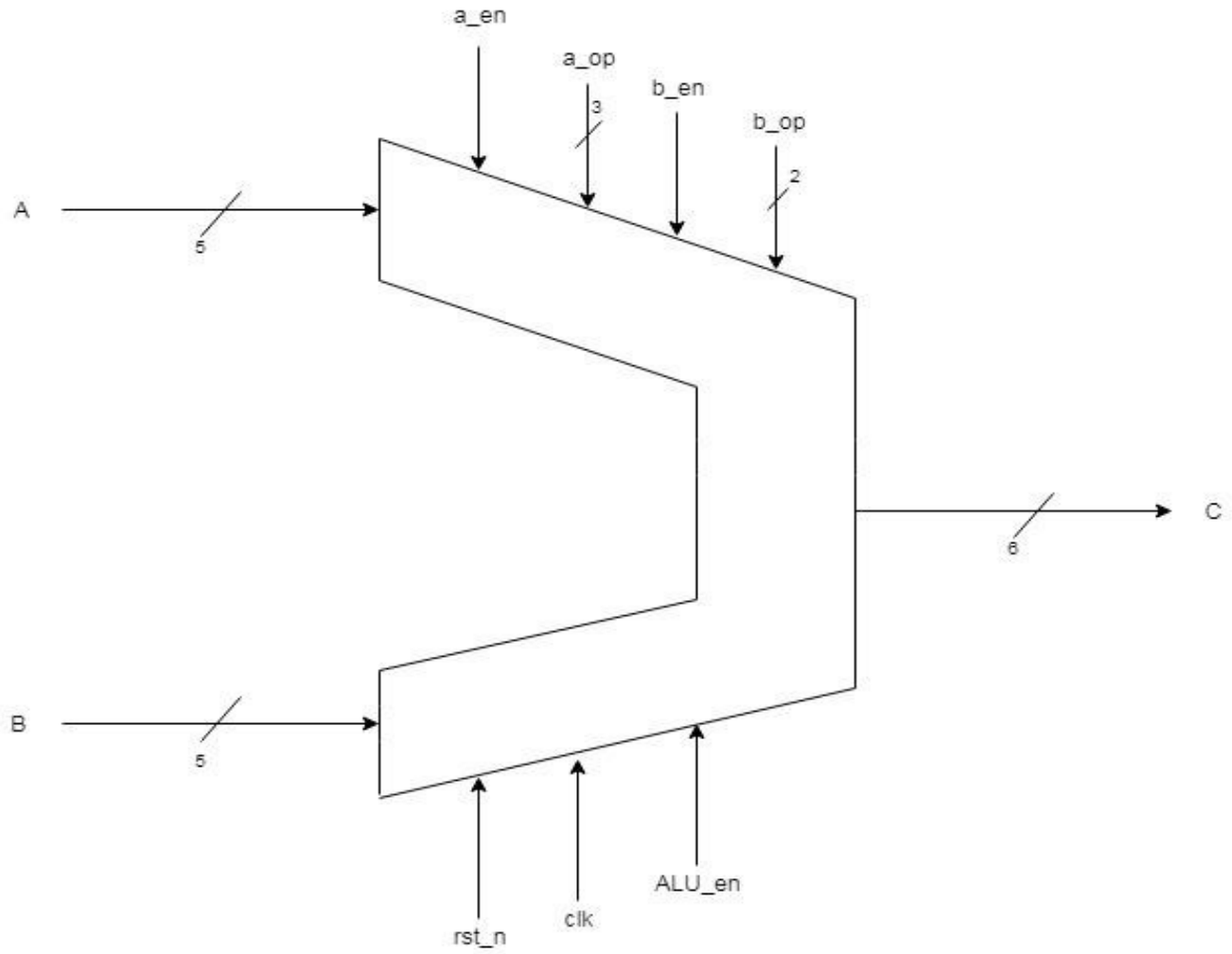


Figure 1: ALU Block

This Section discusses the different inputs and outputs of the ALU block (Figure 1) with a brief description of each. The following table will contain the overall system I/O:

Table 1: ALU I/O

| Signal Name | Type | Width (Bits) | Description |
|---|---|---|---|
| A | Input | 5 | Signed 5 bits input data |
| B | Input | 5 | Signed 5 bits input data |
| a_en | Input | 1 | Enable signal for a_op operations |
| a_op | Input | 3 | 3 bits defining a set of operations to be performed |
| b_en | Input | 1 | Enable signal for b_op operations |
| b_op | Input | 2 | 2 bits defining a set of operations to be performed |
| rst_n | Input | 1 | Asynchronous Active-Low reset signal |
| clk | Input | 1 | System input Clock signal |
| ALU_en | Input | 1 | The overall system Enable signal |
| C | Output | 6 | Signed 6 bits output data |

1. **Rst_n :** a **reset** signal that clears any pending errors or events and brings a system to normal condition or an initial state.

2. **ALU_en:** Enable signal to the system.

3. **A_en:** When this enable signal is active high, **only** the set of **a_op (Spec #7)** operations are performed.

4. **B_en:** When this enable signal is active high, **only** a set of **b_op (Spec #8)** operations are performed.

5. **Input A:** This is a 5 bits signed input bus to the ALU ranging from (-15 to +15) on which operations will be performed.

6. **Input B:** This is a 5 bits signed input bus to the ALU ranging from (-15 to +15) on which operations will be performed.

7. **Output C:** This is a 6 bits signed Output bus to the ALU ranging from (-31 to +31) which contains the result of ALU operations.

8. **a_op:** It is 3 bit unsigned bus representing an encoding for a set of operations. Each value refers to a specific function for the ALU to perform. The following table (Table 2) illustrates these functions with a brief description of each, putting into consideration the following 2 constraints:
   - These functions are performed **if and only if** the a_en is active-high and b_en is active low.
   - The values with a "Null" operation refer to an illegal input for the system.

Table 2: a_op functions

| Sub_Specs | A_op | C (Function) | Description |
|---|---|---|---|
| 8.a | 0 | A + B | The Summation of the inputs |
| 8.b | 1 | A – B | Subtracting Input B from A |
| 8.c | 2 | A "XOR" B | Bitwise "XOR" logic operation |
| 8.d | 3 | A & B | Bitwise "AND" logic operation |
| 8.e | 4 | A & B | Bitwise "AND" logic operation |
| 8.e | 5 | A \|\| B | Bitwise "OR" logic operation |
| 8.f | 6 | A "XNOR" B | Bitwise "XNOR" logic operation |
| | 7 | Null | The system is not permitted to take this value |

9. **b_op (SET 1):** It is 3 bit unsigned bus representing an encoding for a set of operations. Each value refers to a specific function for the ALU to perform. There are 2 different sets of these operations depending on the enable signals (a_en & b_en). The following table (Table 3) illustrates the first set of functions with a brief description of each, putting into consideration the following 2 constraints:
   - These functions are performed **if and only if** the b_en is active-high and a_en is active low.
   - The values with a "Null" operation refer to an illegal input for the system.

Table 3: Set 1 b_operations

| Sub_Specs | b_op(set 1) | C (Function) | Description |
|---|---|---|---|
| 9.a | 0 | A “NAND” B | The Summation compliment of the inputs |
| 9.b | 1 | A + B | The summation of the Inputs |
| 9.c | 2 | A + B | The summation of the Inputs |
| | 3 | Null | The system is not permitted to take this value |

**10.b_op (SET 2):** These are the second set of b_ operations .The following table (Table 3) illustrates these functions with a brief description of each, putting into consideration the following 2 constraints:

- These functions are performed **if and only if both** the b_en and a_en are active high.

Table 4: Set 2 b_operations

| Sub_Specs | b_op(set 2) | C (Function) | Description |
|---|---|---|---|
| 10.a | 0 | A “XOR” B | Bitwise “XOR” logic operation |
| 10.b | 1 | A “XNOR” B | Bitwise “XNOR” logic operation |
| 10.c | 2 | A – 1 | Subtracting ‘1’ from input A |
| 10.d | 3 | B + 2 | Adding ‘2’ to input B |

# 3 LOGIC REFERENCES

As observed in section 2, various operations are based on standard logic gates such as XOR, XNOR, etc…. The logical truth table of these gates are stated in table 5, as a reference during the output evaluation stage.

Table 5: LOGIC gates references

| INPUTS | | GATES | | | | |
|---|---|---|---|---|---|---|
| A(1bit) | B(1 bit) | AND | NAND | OR | XOR | XNOR |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |

# 4 TEST CASES

In this section we are going to handle the functional verification using selected test cases to ensure that the system is functioning as per required in all cases within the specified specs extracted in the section above. In each test case the following aspects are considered:

1. Derivation of Inputs to create the needed environment for testing.
2. Monitoring the output as a result of the tested environment.
3. Checking the output is as expected.