**System Design Part 4**

**ALU Input list:**

| Name | Bit-size | Description |
|---|---|---|
| IN 1.16 | 16 | 16-bit input |
| FEEDBACK | 16 | 16-bit input (from register) |
| OP 1.4 | 4 | 4-bit operation code input |
| CLK | 1 | Clock |

**ALU Output list:**

| Name | Bit-size | Description |
|---|---|---|
| OUT 1.32 | 32 | 32-bit arithmetic output |
| OUT 2.2 | 2 | 2-bit error output |

---

**calculateGrid Input list**

| Name | Bit-size | Description |
|---|---|---|
| P1_action | 4 | 4-bit input action of Player 1 |
| P2_action | 4 | 4-bit input action of Player 2 |
| oldGrid | 400 | 10 by 10 by 4 grid representing old grid of game |
| CLK | 1 | 40-tick clock (20 high, 20 low) |

**calculateGrid Output list**

| Name | Bit-size | Description |
|---|---|---|
| newGrid | 400 | 10 by 10 by 4 grid representing new grid of game |

**Interfaces:**

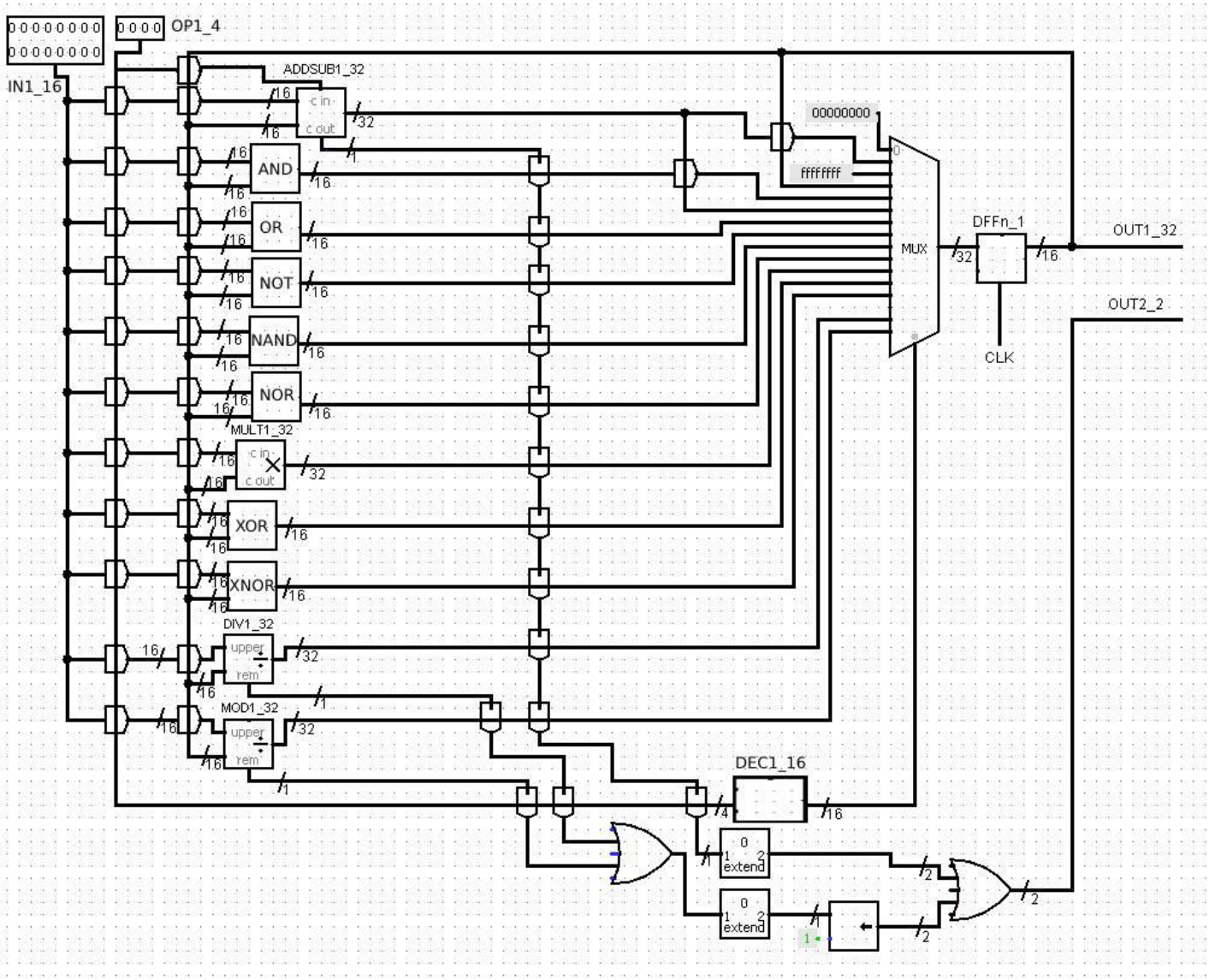
**For simplicity of wording, the value stored in the register is considered to be an input.**

| Name | Bit-size | Description |
|---|---|---|
| Mode | 1 | 1-bit command the controls Addsub |
| Select | 16 | 16-bit one hot selector |
| Channel 0 | 32 | Reset - returns all 0’s |
| Channel 1 | 32 | Result of summation between the two inputs to the multiplexor |
| Channel 2 | 32 | Preset- returns all 1’s |
| Channel 3 | 32 | No-op - Do nothing |
| Channel 4 | 32 | And - returns the bitwise AND of the two inputs |
| Channel 5 | 32 | Result of difference between the two inputs to the multiplexor |
| Channel 6 | 32 | Or - returns the bitwise OR of the two inputs |
| Channel 7 | 32 | Not - returns the bitwise NOT of the two inputs |
| Channel 8 | 32 | Nand - returns the bitwise NAND of the two inputs |
| Channel 9 | 32 | Nor - returns the bitwise NOR of the two inputs |
| Channel 10 | 32 | Result of multiplication between the two inputs to the multiplexor |
| Channel 11 | 32 | Xor- returns the bitwise XOR of the two inputs |
| Channel 12 | 32 | Xnor - returns the bitwise XNOR of the two inputs |
| Channel 14 | 32 | Result of division between the two inputs to the multiplexor |
| Channel 15 | 32 | Result of modulo between the two inputs to the multiplexor |

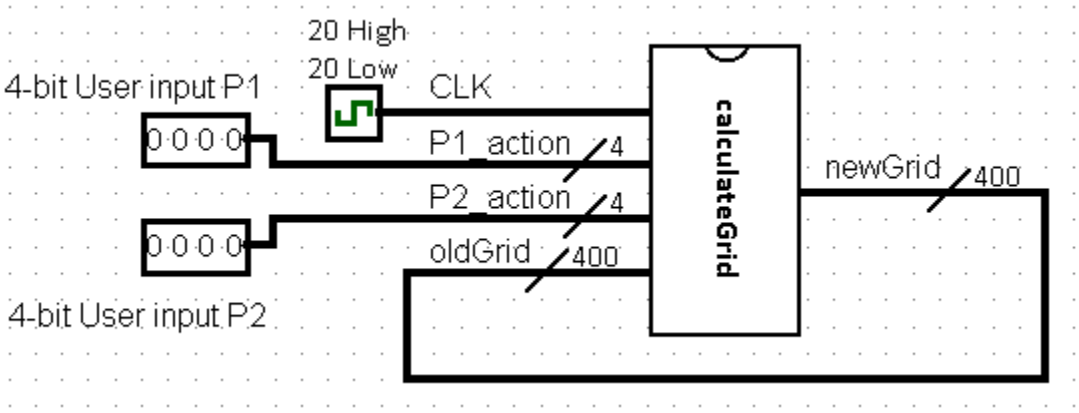
Part List:

- 1 comparator
    - Compares constant 0001 to opcode input to select the AddSub mode.
- 1 32-bit Adder-Subtractor
    - Takes two 16-bit input and adds them into a 32-bit output or subtracts them into a 32-bit output based on the mode.
    - Outputs a 1-bit error code that is 1 if there is an overflow or underflow, or 0 if there is none.
- 1 32-bit multiplier
    - Takes two 16-bit input and adds them into a 32-bit output.
- 1 32-bit divider
    - Takes two 16-bit input and adds them into a 32-bit output.
    - Outputs a 1-bit error code if the input is divided by zero.
- 1 32-bit modulo
    - Takes two 16-bit input and adds them into a 32-bit output.
    - Outputs a 1-bit error code if the input is divided by zero.
- 1 4-to-16 decoder
    - Takes the 4-bit opcode input and outputs a 16-bit one hot.
- 1 16 channel 32-bit with a one-hot selector multiplexer
    - Has 16 channels and outputs a 32-bit output.
- 2 OR gates
- 1 bit shifter
    - Shifts 1-bit input logically left by 1 bit and outputs a 2-bit output
- 2 bit extenders
- 1 32-bit register
- 1 clock
- 1 16-bit AND gate
- 1 16-bit OR gate
- 1 16-bit NOT gate
- 1 16-bit NAND gate
- 1 16-bit NOR gate
- 1 16-bit XOR gate
- 1 16-bit XNOR gate
- 1 calculateGrid Module
    - Constructed from several ALU and gates according to the system design
    - Calculates the next state of the grid with player actions as inputs and outputs the new state of the grid for the game
    - States are as followed: Empty: E , Player 1: X, Player 2: Y, Destructible Wall: D, Unbreakable Wall: W, Bombs : B, Bomb 2: B, Player 1 on a bomb: J, Player 2 on a bomb: K, Explosion: 9, Player dead: O

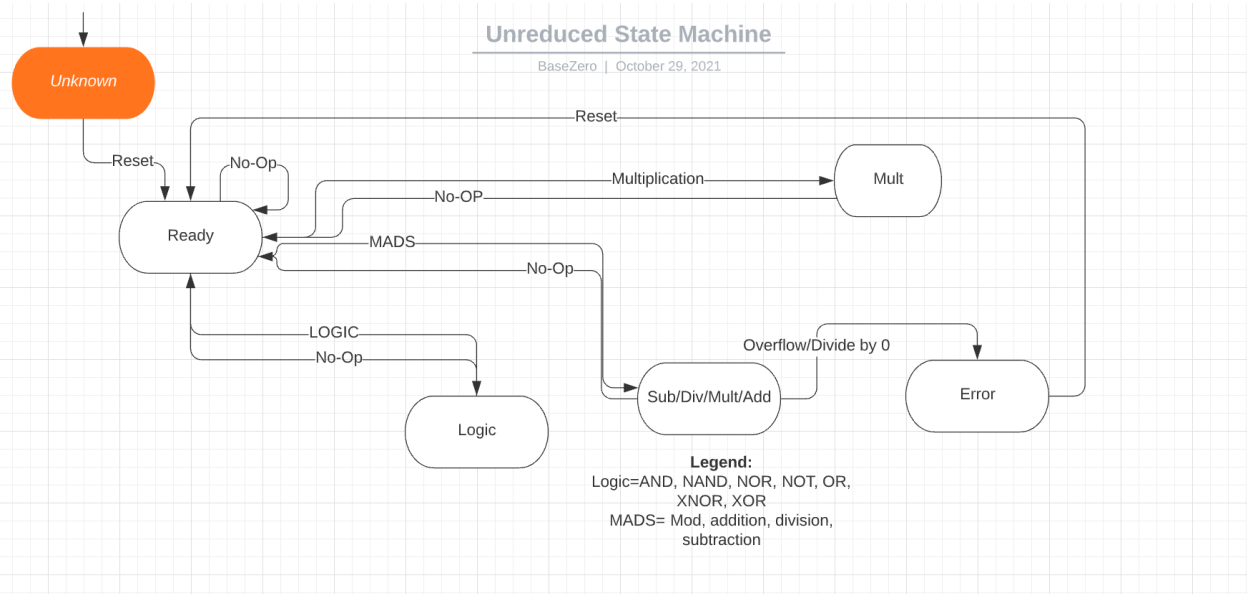
**ALU Circuit Diagram:**



**calculateGrid Circuit diagram:**



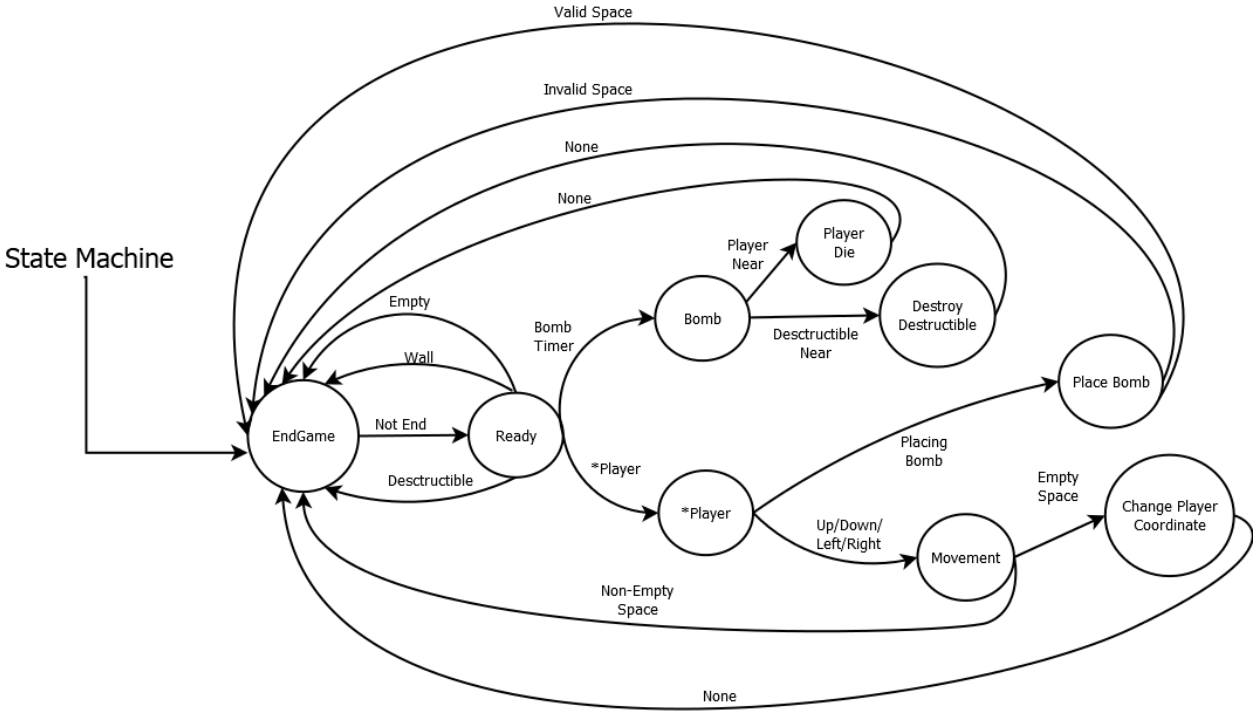
**ALU State Machine:**




**ALU State Table:**

| Current State | Trigger | Next State |
|---|---|---|
| Unknown | Reset | Ready |
| Ready | No-Op | Ready |
| Ready | Multiplication | Mult |
| Ready | MOD/ADD/DIV/SUB | MADS |
| Ready | Logic | Logic |
| Math(&Mod) | No-Op | Ready |
| Add | Carry | Error |
| MOD/ADD/DIV/SUB | Divide by 0/Overflow | Error |
| Logic | No-Op | Ready |
| Error | Reset | Ready |

**calculateGrid State Machine:**




*Player represents Player 1 and Player 2

**calculateGrid State Table:**

| **Current State** | **Trigger** | **State Diagram** |
|---|---|---|
| EndGame | Not End | Ready |
| Ready | Empty | EndGame |
| Ready | Player | Player |
| Ready | Destructible | EndGame |
| Ready | Wall | EndGame |
| Ready | Bomb Timer | Bomb |
| Player | None | EndGame |
| Player | Up/Down/Left/Right | Movement |
| Player | Placing Bomb | Place Bomb |
| Movement | Empty Space | Change Player Coordinate |
| Movement | Non-Empty Space | EndGame |
| Change Player Coordinate | None | EndGame |
| Place Bomb | Valid Space | EndGame |
| Place Bomb | invalid Space | EndGame |
| Bomb | Player Near | Player Die |
| Bomb | Destructible Near | Destroy Destructible |
| Player Die | None | EndGame |
| Destroy Destructible | None | EndGame |