Minecraft ALU Module

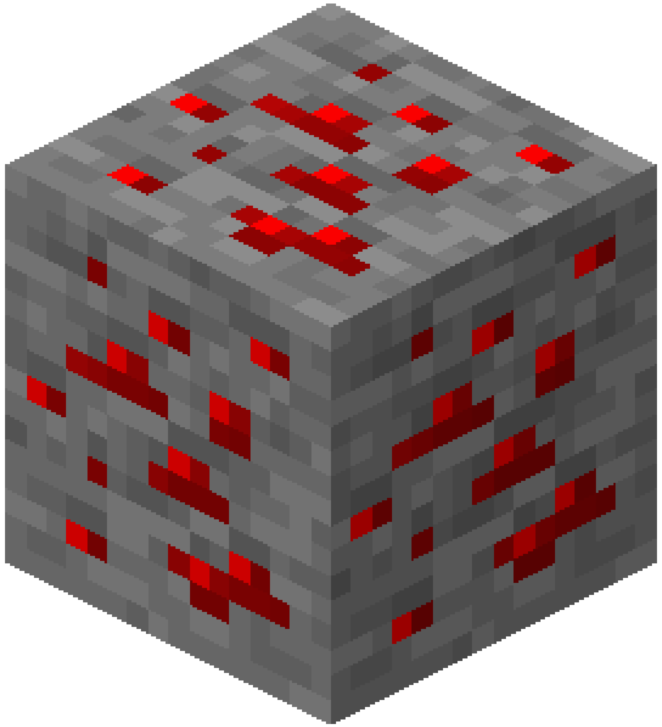
By: Brandon Esebag, Maria Aponte, Aaron Hoang, Jonathan Liu

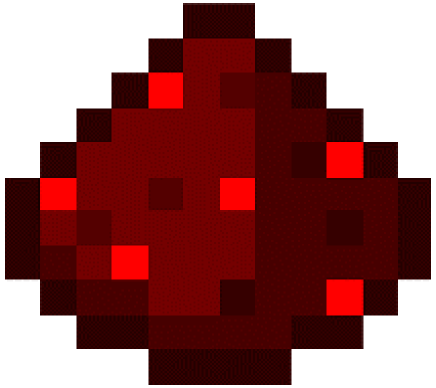
What is Minecraft?

- Minecraft is a kid's game released in 2009
- The game is an open world sandbox where the player can essentially do anything
- As per the name Minecraft, there is an emphasis on **mining** ores and **crafting** tools
- The world is voxel based and procedurally generated

What is Redstone?

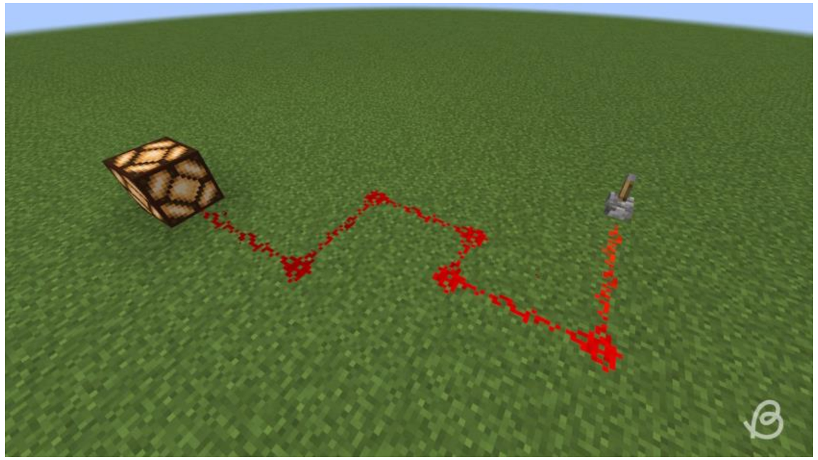
- Most ores in Minecraft (eg Diamond, Gold, Iron, etc) are real materials…
  - Redstone is a fictional ore that enables specific game mechanics!
- Redstone is an in-game resource that behaves as a conductor
- In combination with other items in the game (levers, repeaters, pistons, etc.) someone can use redstone to create primitive electronics in the game
- Normal redstone uses include:
  - Controlling doors
  - Controlling mine-carts
  - Using pistons to control water/lava flow

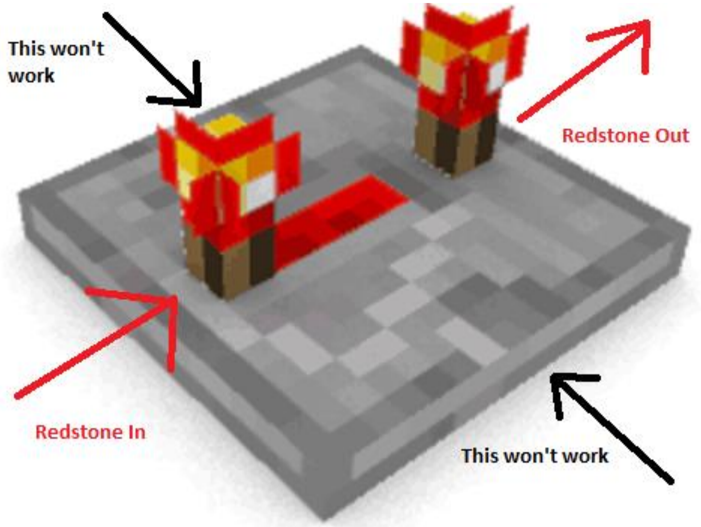




Redstone Dust Placement

- Redstone dust can be placed to form wires as shown below (top)
- Redstone wire signals attenuate from value 15 -> 0 over 16 blocks of travel
- A redstone repeater (bottom) amplifies any signal > 0 back to 15 with a time delay of 0.1 - 0.4 s

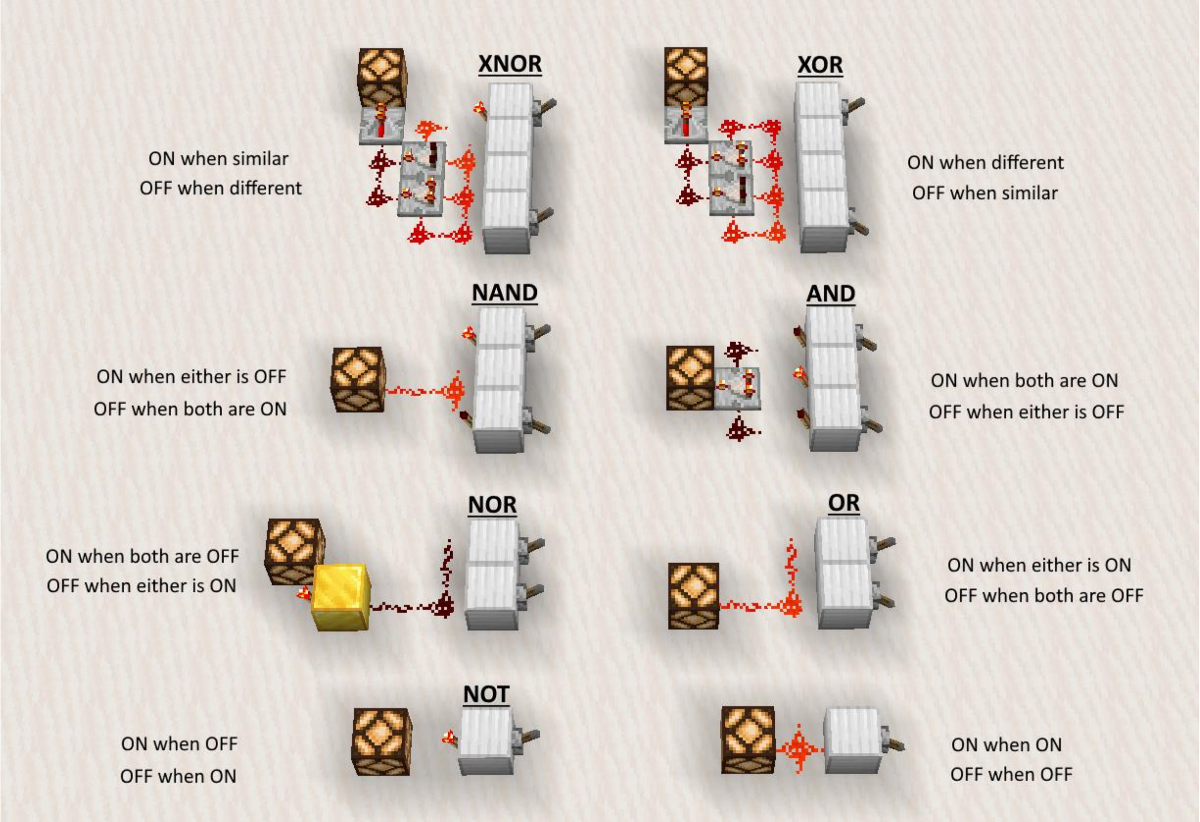




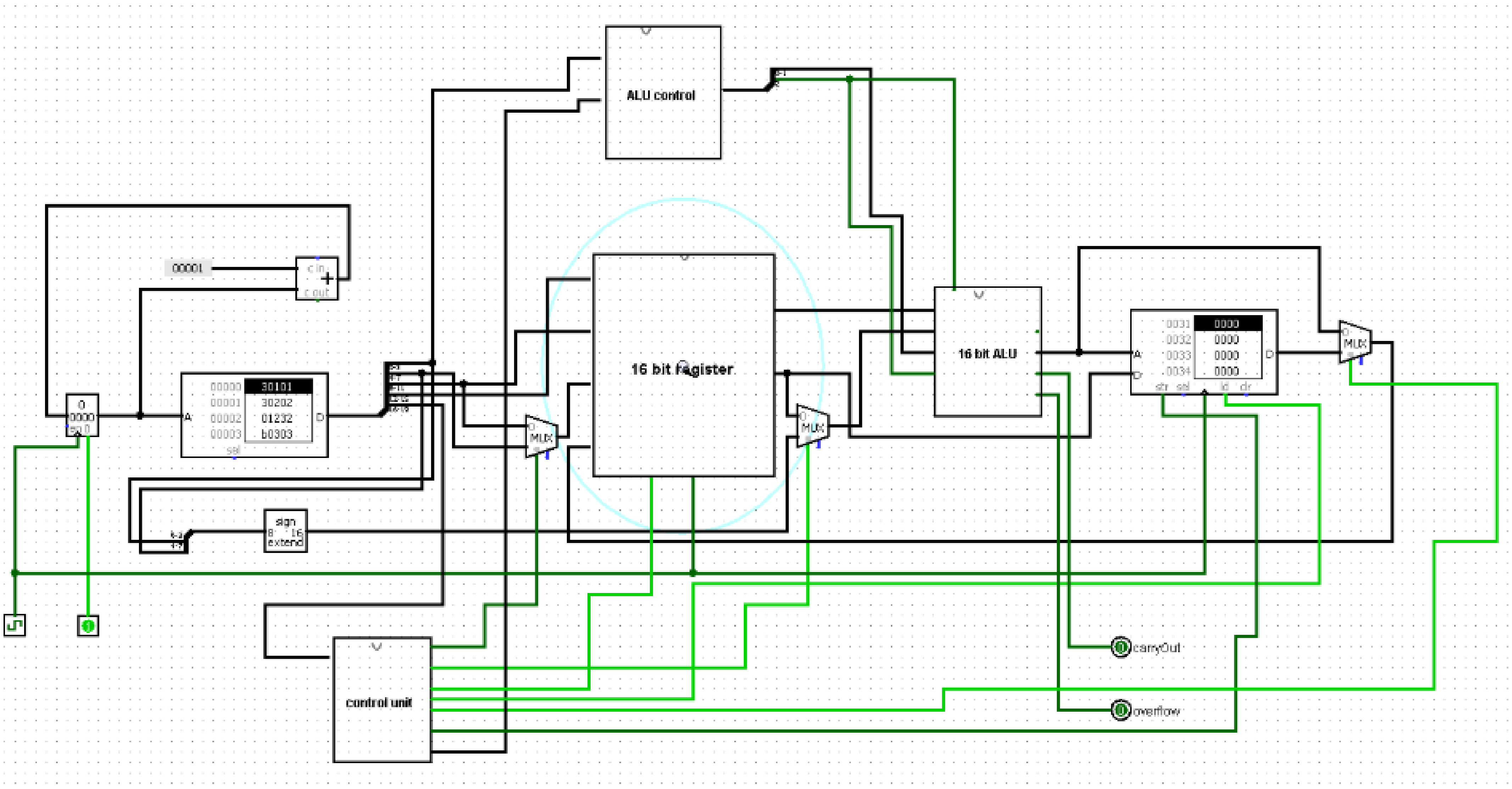
Combinational Logic in Redstone

It is even possible to create combination circuits in the game with redstone

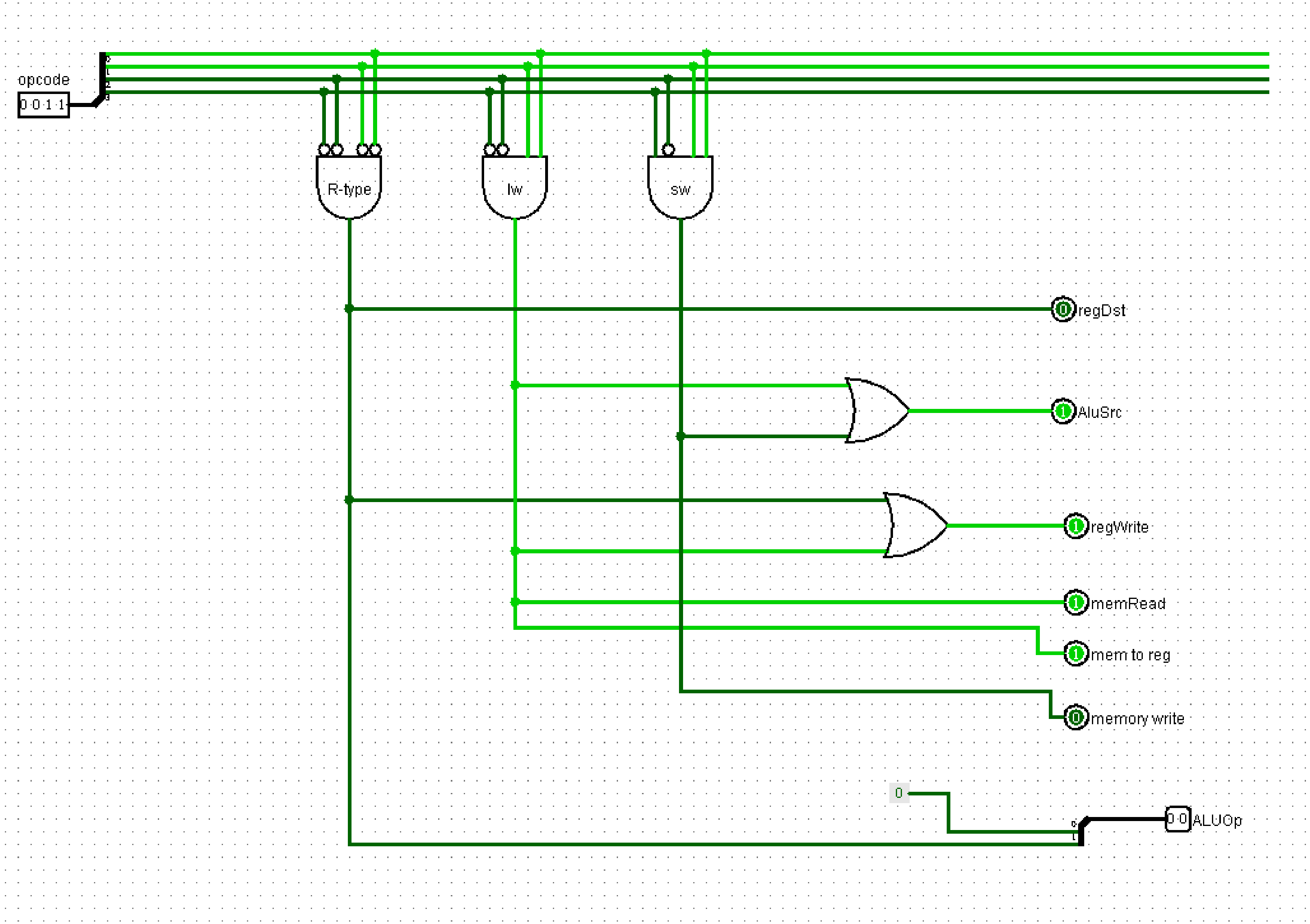
This leads us to what we want to do...



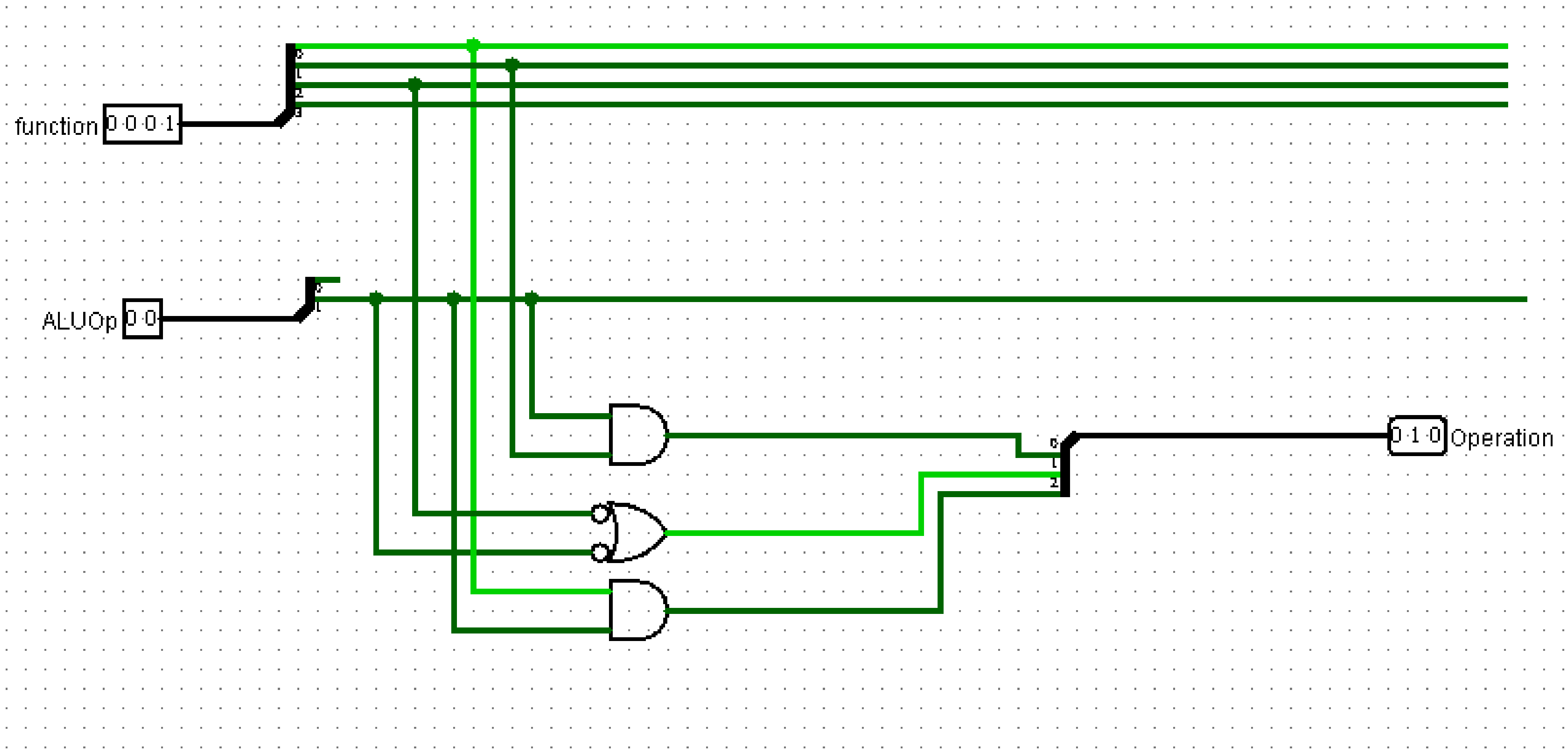
16 bit ALU Circuit Design



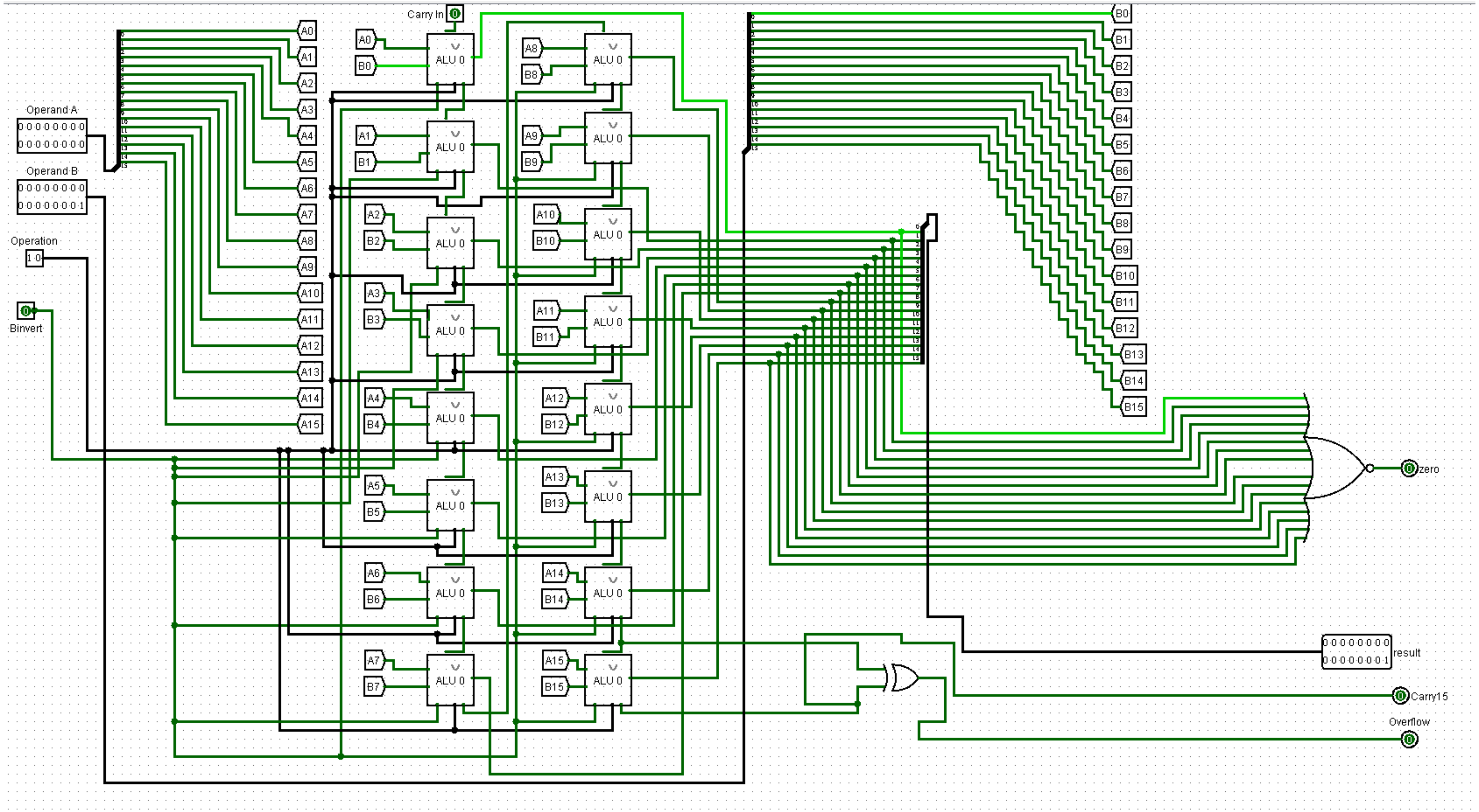
Control Unit

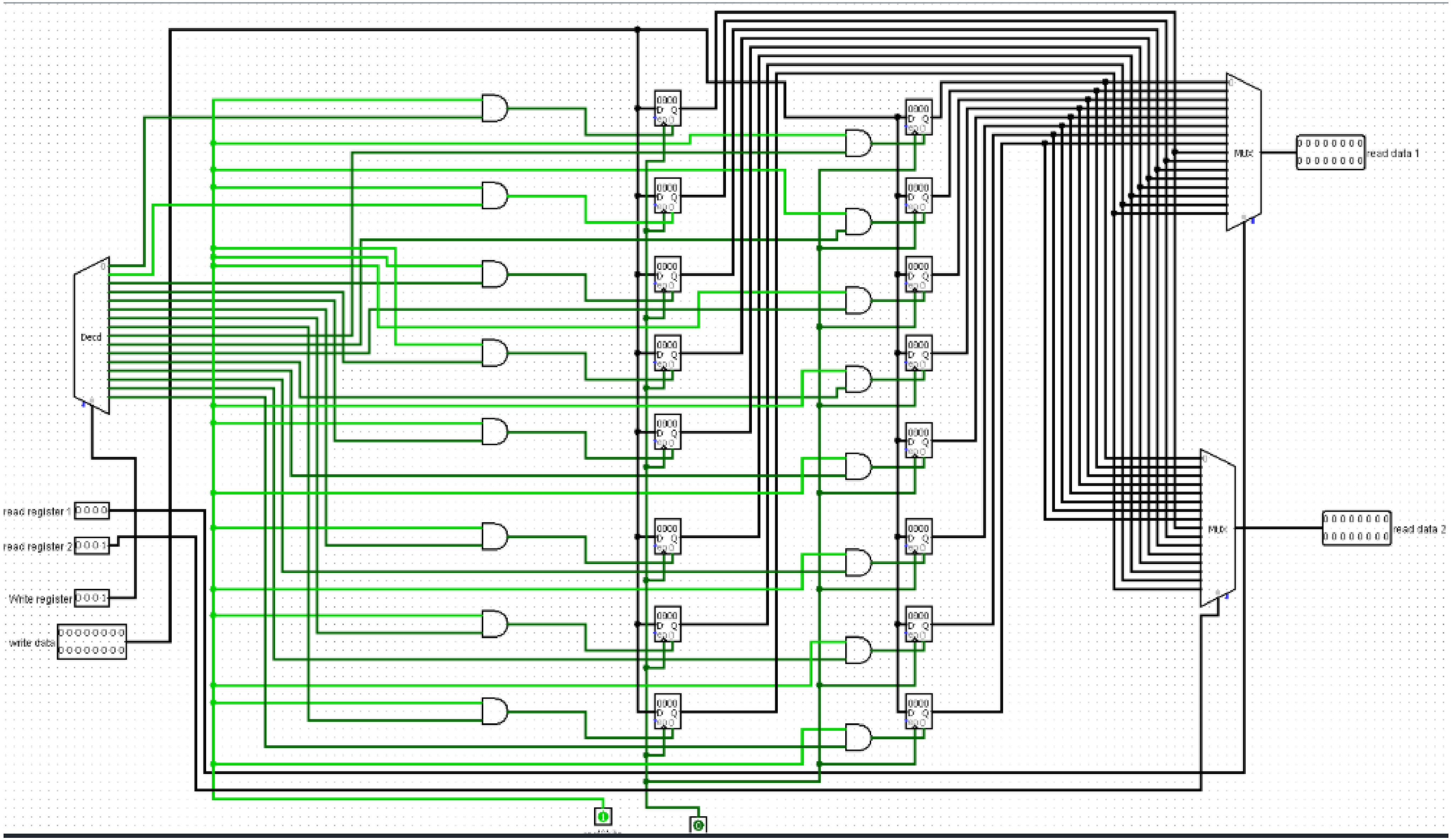


ALU Control



16-Bit ALU


Carry In
Operand A
00000000
00000000
Operand B
00000000
00000001
Operation
1 0
Binvert
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
B0
B1
B2
B3
B4
B5
B6
B7
B8
B9
B10
B11
B12
B13
B14
B15
ALU 0
zero
result
00000000
00000001
Carry15
Overflow


read register 1
0000
read register 2
0001
Write register
0001
write data
00000000
00000000
Decd
MUX
MUX
read data 1
00000000
00000000
read data 2
00000000
00000000
1
0

Components Overview

- **1. 16-bit Register**
  - Acts as a storage unit for intermediate or temporary data.
  - Data from this register is sent to the ALU for computations.
- **2. 16-bit ALU**
  - Performs arithmetic and logic operations such as addition, subtraction, AND, OR, and NOT.
  - Accepts inputs from the 16-bit register and another source (multiplexer-controlled).
  - Outputs the result of computations to either RAM or another component.
- **3. RAM (Data Memory)**
  - Stores and retrieves data using the address generated by the ALU.
  - Data input (D) is sourced from the ALU output, while data output is sent back to the system through another multiplexer.
- **4. Multiplexers (MUX)**
  - Allow selection of data from multiple sources based on control signals.
  - Control the flow of data between the ALU, RAM, and registers.
- **5. Immediate Values**
  - The immediate values (from instruction encoding) are used as direct inputs for operations or memory addressing.
  - Extended to 16 bits to align with the width of the system.

This project is a comprehensive implementation of a 16-bit Arithmetic Logic Unit (ALU) integrated with registers and RAM. It demonstrates the fundamental structure and functionality of a simple processor system, focusing on memory access and data manipulation through arithmetic and logic operations.

Deliverables

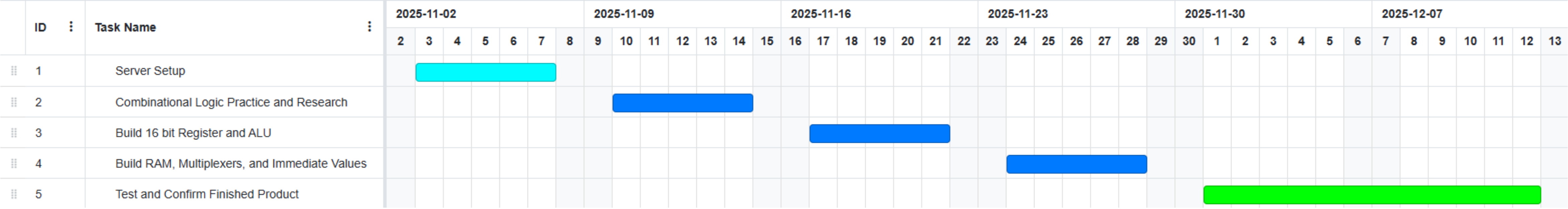
Core Features

- **Arithmetic Operations**: The ALU can add or subtract data from the register and other inputs.
- **Logic Operations**: Perform bitwise operations like AND, OR, and NOT.
- **Memory Access**: The ALU generates addresses for RAM to perform read/write operations.
- **Data Routing**: Multiplexers route the correct data to the required components based on control signals.

Example Operation

- **Input Data**:
  - Data is loaded into the 16-bit register and RAM.
  - Immediate values are prepared (if needed).
- **ALU Computation**:
  - Register 1 sends its data to the ALU.
  - Register 2 provides data for a secondary ALU input via the multiplexer.
  - The ALU computes the result and outputs it.
- **Memory Write**:
  - ALU output serves as an address for RAM.
  - Data from Register 2 is written to the specified address in RAM.
- **Memory Read**:
  - ALU generates an address.
  - RAM sends the data stored at that address to the next stage (e.g., a register).

Gantt Chart



ID	Task Name
1	Server Setup
2	Combinational Logic Practice and Research
3	Build 16 bit Register and ALU
4	Build RAM, Multiplexers, and Immediate Values
5	Test and Confirm Finished Product

- Week One: Server Setup
- Week Two: Build and Test All Combinational Logic Elements Using Redstone
- Week Three: Build 16-bit Register and ALU
- Week Four: Build RAM, Multiplexers, and Immediate Values
- Week Five: Combine Together and Test Finished Product