Microprocessor Project

Barnabas, James, Rishi
and Olu

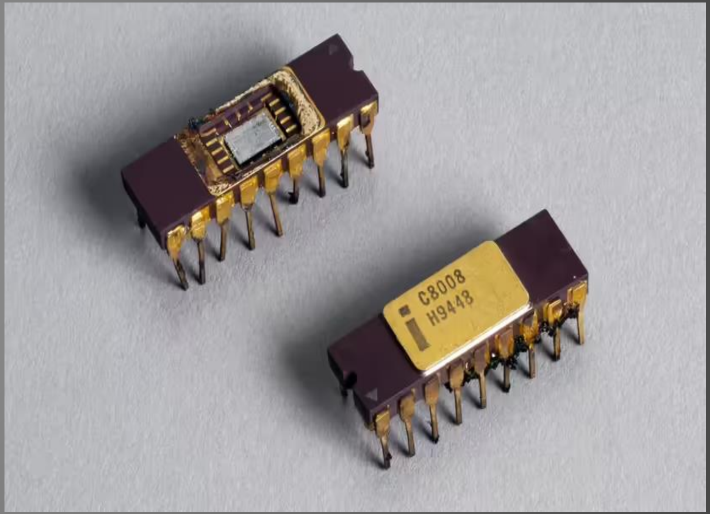
Digital Logic 2700

What is a microprocessor?

- CPU - Central Processing Unit
  - Perform computations.
  - Execute instructions.

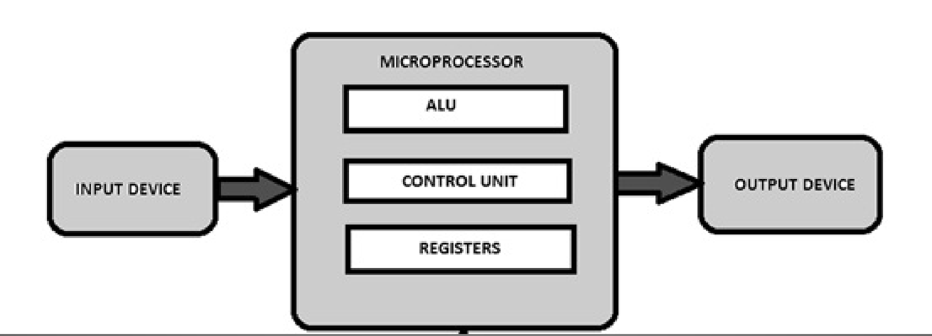
- 1st CPU: Intel 4004 (1971) - Add & Subtract

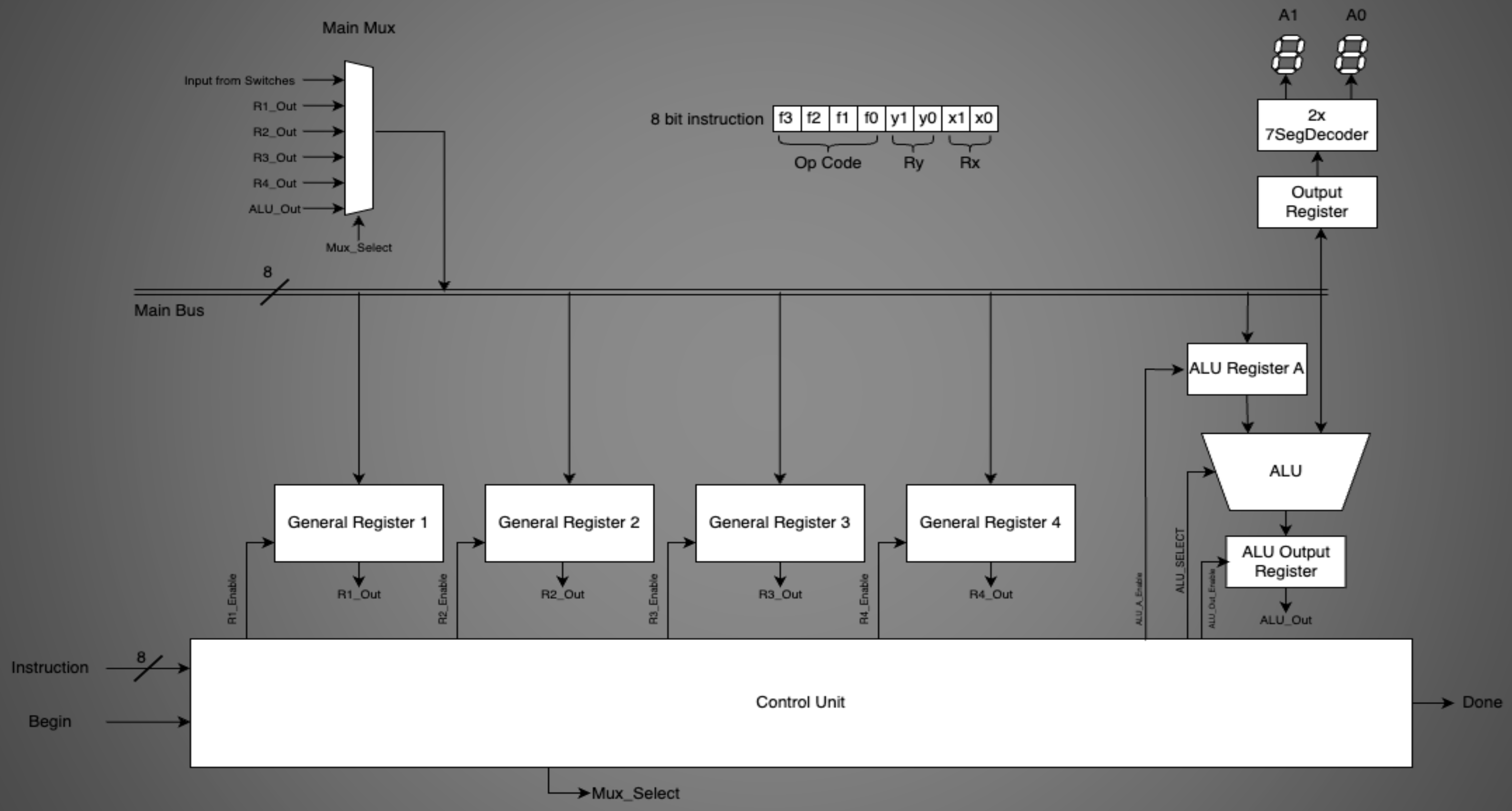
- Intel 8080 (1974)
  - Single semiconductor chip.
  - First CPU in home computers.
  - Logic, control, storage, input, and output.



How Does It Work?

- Fetch: The microprocessor retrieves instructions from the computer's memory
- Decode: The microprocessor interprets instructions and initiates a process or a computation
- Execute: It then performs the requested operation
- Store: The results of the execution are stored in the computer's memory



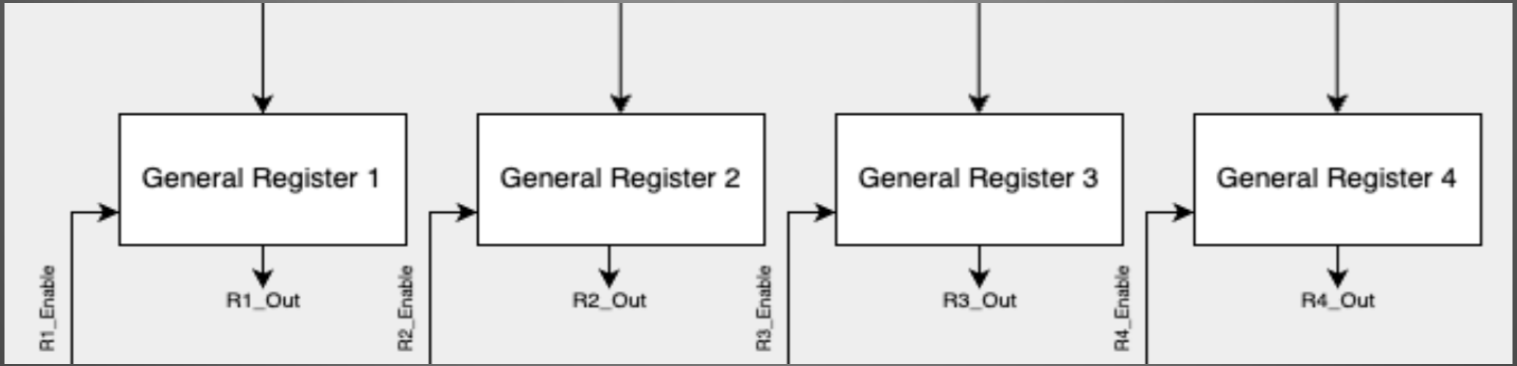

Main Mux
Input from Switches
R1_Out
R2_Out
R3_Out
R4_Out
ALU_Out
Mux_Select
8 bit instruction
f3
f2
f1
f0
y1
y0
x1
x0
Op Code
Ry
Rx
A1
A0
2x 7SegDecoder
Output Register
8
Main Bus
ALU Register A
ALU
ALU Output Register
ALU_SELECT
ALU_Out
General Register 1
General Register 2
General Register 3
General Register 4
R1_Enable
R2_Enable
R3_Enable
R4_Enable
R1_Out
R2_Out
R3_Out
R4_Out
Instruction
8
Begin
Control Unit
Done
Mux_Select

Registers

- Registers are small, high-speed storage locations and can provide fast access to data necessary for computations performed

- Purpose: Stores data temporarily while the microprocessor is executing an instruction set

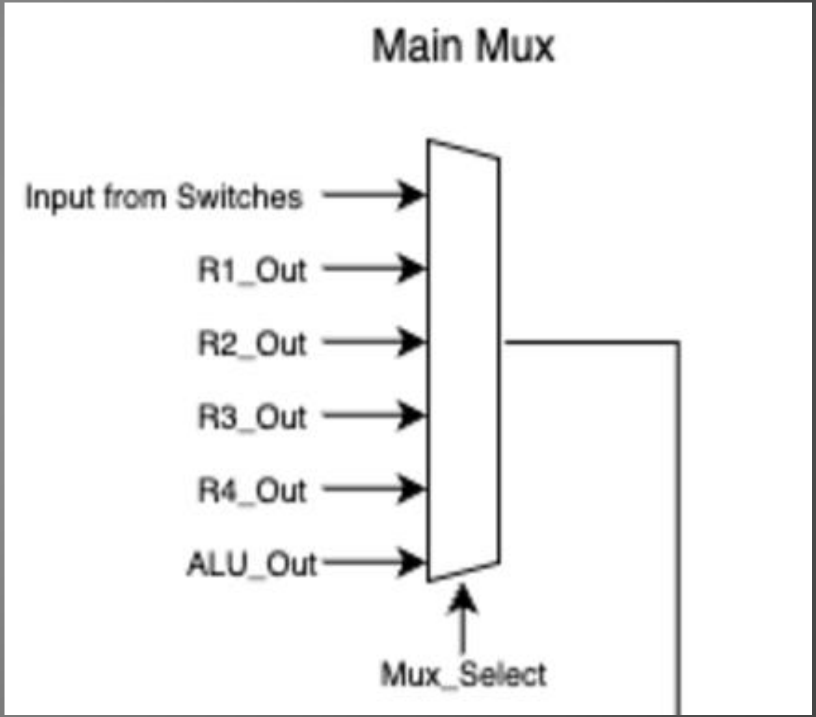
- Example Usage: Output of the ALU can be stored in one of these registers to complete instructions with several steps



MUX

- MUX (multiplexer) allows the selection of one of several input signals and forwards the selected signal to a single output
    - Essentially, it serves as a data selector

- Purpose: Controls the flow of data between different parts of the microprocessor by directing data from multiple sources to a single destination, and sets the value of the bus



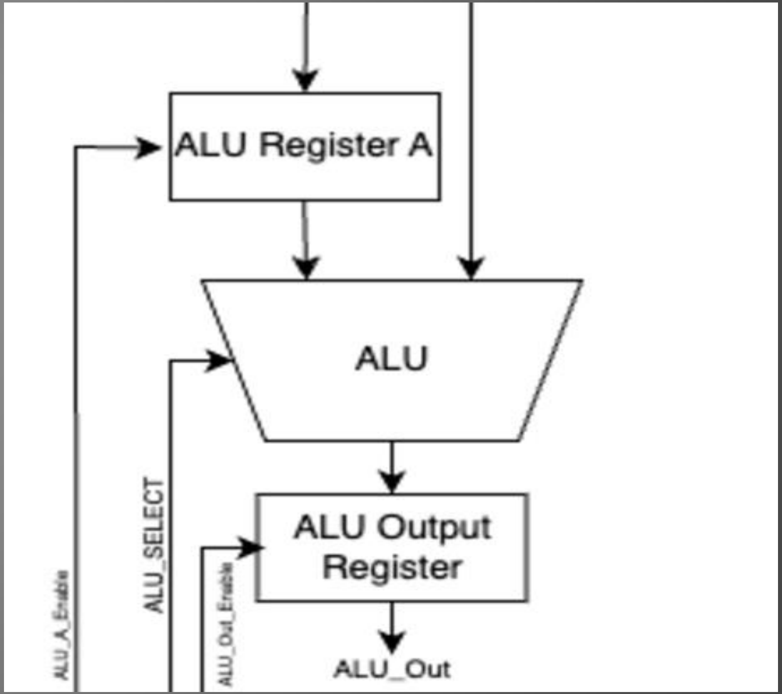
Arithmetic Logic Unit (ALU)

- Generic Function: Performs a variety of arithmetic and logic operations

- Directly handles the mathematical and logical tasks required for executing instructions given to the microprocessor

- Typical Functions: Logical Operations (i.e. AND, OR, XOR, NOT, XNOR) and mathematical operations (A+B, A-B, A+1, A-1, etc)

- Input 'A' Register and Output Register



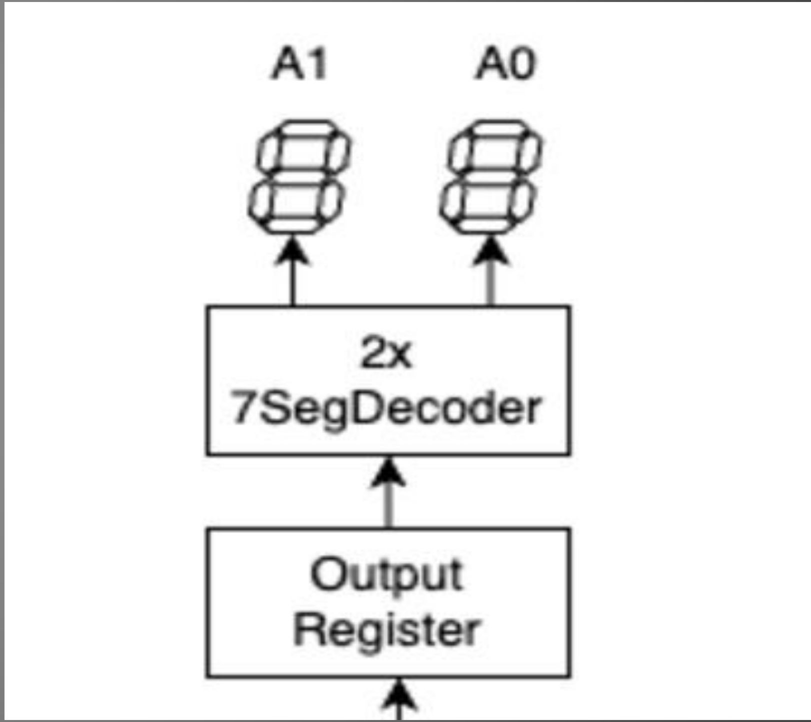
ALU Operations

IR Code	Function
0000	A + B
0001	|A - B|
0010	Shift A Left B Times
0011	Shift A Right B Times
0100	A AND B
0101	A OR B
0110	A XOR B
0111	NOT(A)
1000	Top 8 Bits of A * B
1001	Bottom 8 Bits of A * B
1010 to 1111	No Operation

- A+B, |A - B|: 9-bit adder(s)
- Shift Operation
- Logical Operations: AND, OR, XOR, NOT
- Multiplication: Generic Unsigned Array Multiplier
  - 2x 7-segment displays available to display the result
  - Max Output Size of A*B: 16 bits

7 Segment Decoder

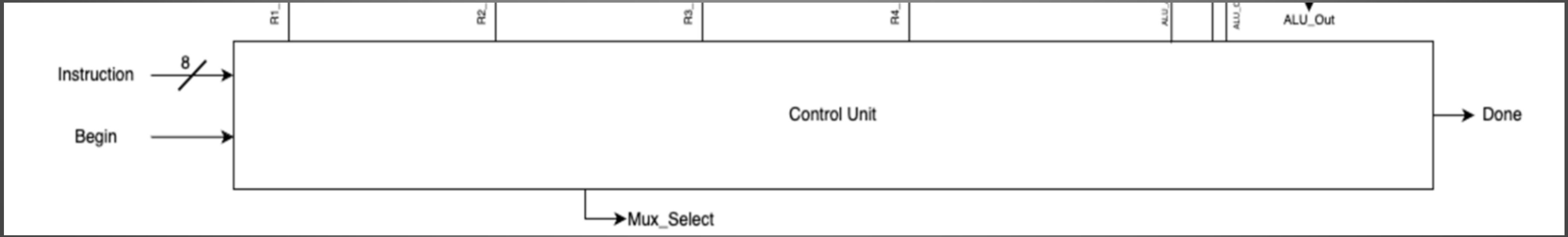
- 7 Segment decoder displays register output
- Takes 8 bit input and uses two 7 segment displays to show the value
- For debugging, it can also display additional information depending on whether a debug button is pressed or not



Control Unit

- Coordinates microprocessor actions
- Generates control signals

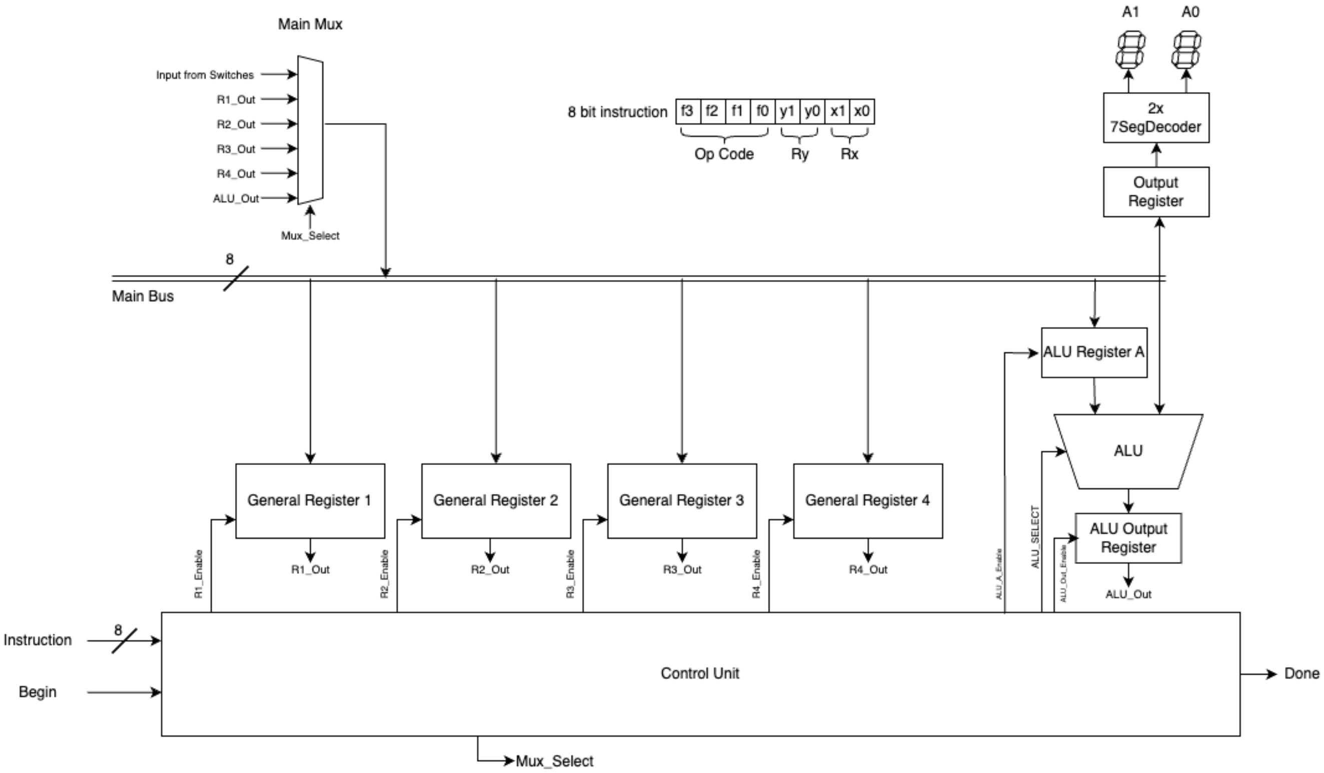
1. Instruction Locked
2. Instruction steps executed (registers, ALU, bus manipulated)
3. Done is placed "high"



Instruction Set

OP Name	OP Code					OP Name	OP Code			
MOVE	0	0	0	0		SHR	1	0	0	0
LOAD	0	0	0	1		SHL	1	0	0	1
INCR	0	0	1	0		AND	1	0	1	0
DECR	0	0	1	1		OR	1	0	1	1
ADD	0	1	0	0		XOR	1	1	0	0
SUB	0	1	0	1		MUL	1	1	0	1
COMP	0	1	1	0		MULH	1	1	1	0
MALU	0	1	1	1		DISP	1	1	1	1

Instruction Example: LOAD 0x5 R1



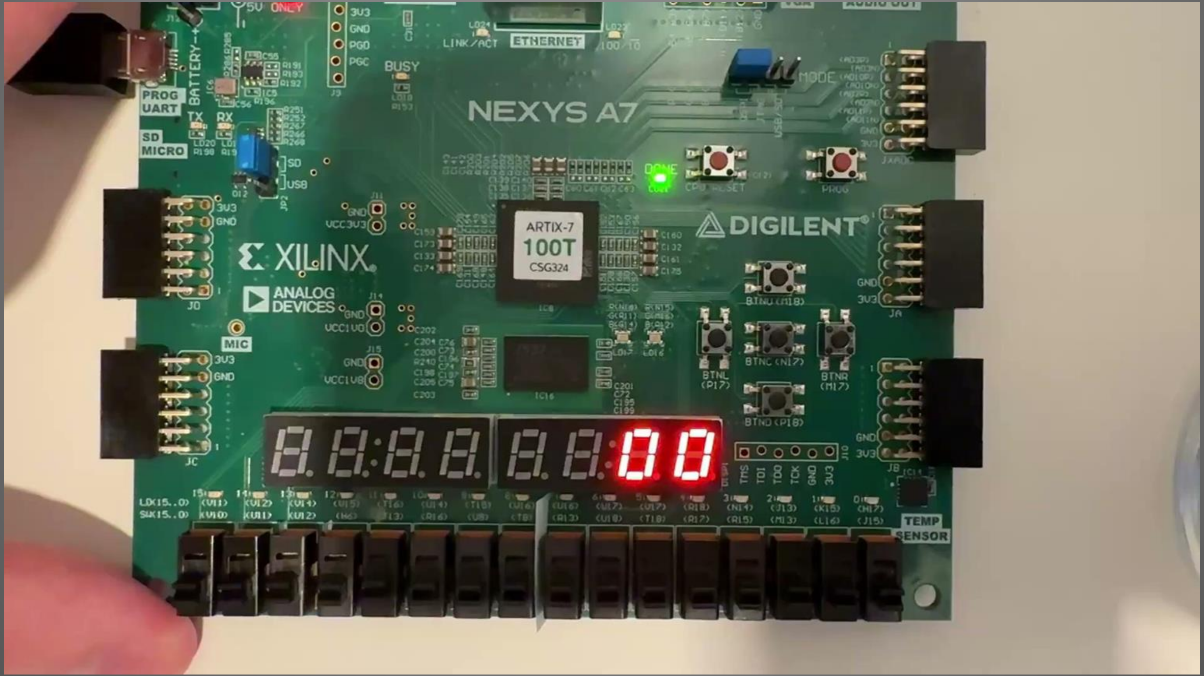
Our Microprocessor In Action

Live Demo Program

LOAD 0x4 R1 # Loads hex 4 into register 1
LOAD 0x4 R2 # Loads hex 4 into register 2
ADD R1 R2 # Adds the values in registers 1 & 2
MALU R2 # Places the output of the add operation in register 3
DISP R2 # Displays R2 on the output

Demo Video

https://drive.google.com/file/d/1Qg_sPso8u_3EUJRID-ZDKCj5DVGtO3fQ/view?usp=sharing



Our Microprocessor In Action (2)

Calculating 13th Fibonacci Number
Fib(13)=233

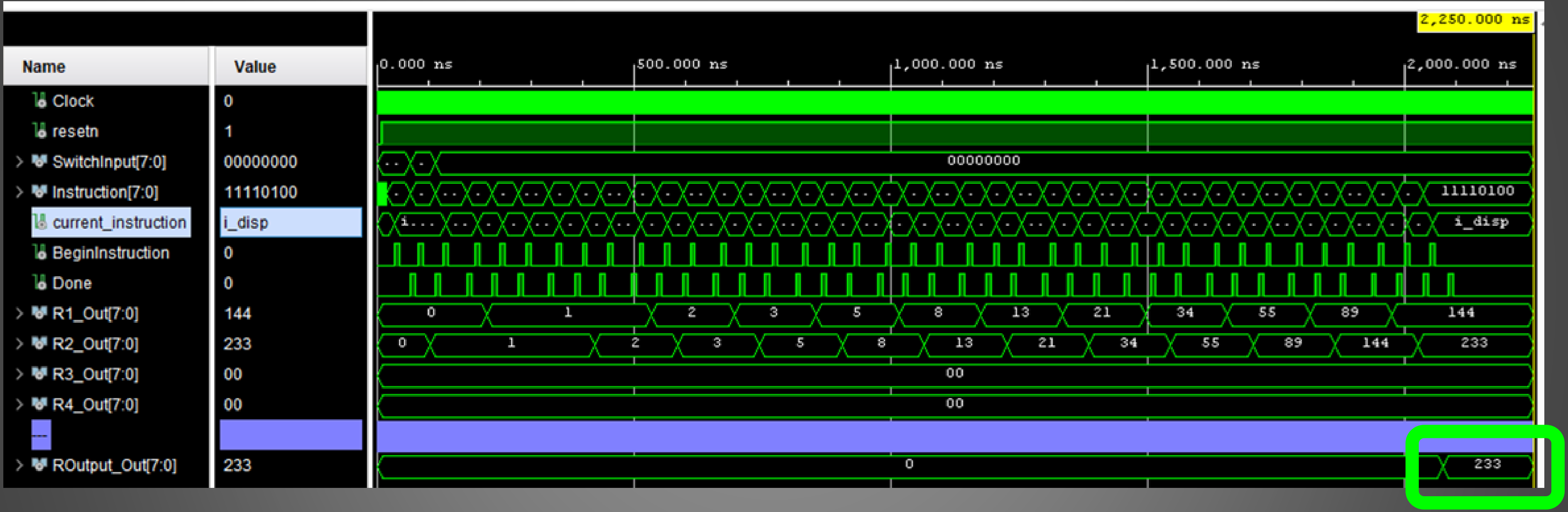
$$F_n = F_{n-1} + F_{n-2}$$

```
# Load initial values
# F(0)
load $0 r1
# F(1)
load $1 r2

# Repeat the following 13 times

# Perform add, result goes into ALU output
add r1 r2
# Move r2 to r1
move r1 r2
# Move ALU output to r2
malu r2
```

Our Microprocessor In Action (2)



13th Fibonacci Number ^

Thank You, Any Questions?

References

- D. Llamocca, “Laboratory 1, ECE-4710/5710.” Electrical and Computer Engineering Department, Oakland University, 2024
- [2] D. Llamocca, Reconfigurable Computing Research Laboratory, https://www.secs.oakland.edu/~llamocca/index.html (accessed Oct. 1, 2024).
- J. J. Jensen, “How to create a clocked process in VHDL,” VHDLwhiz, https://vhdlwhiz.com/clocked-process/ (accessed Nov. 20, 2024).
- In One Lesson, “How a CPU Works,” YouTube, https://www.youtube.com/watch?v=cNN_tTXABUA (accessed Nov. 2, 2024).