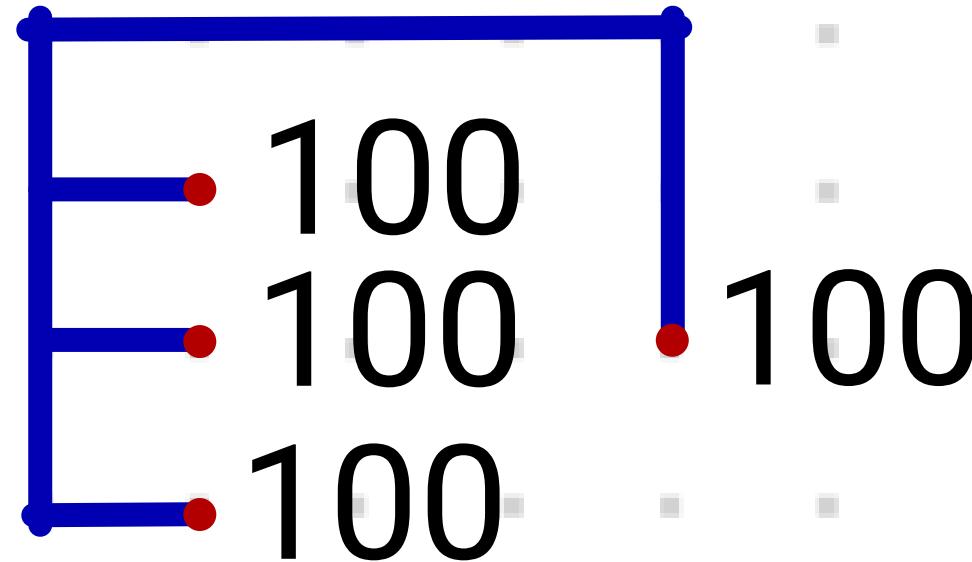


Final Project 2568  
Digital Logic 2110252

Withdraw

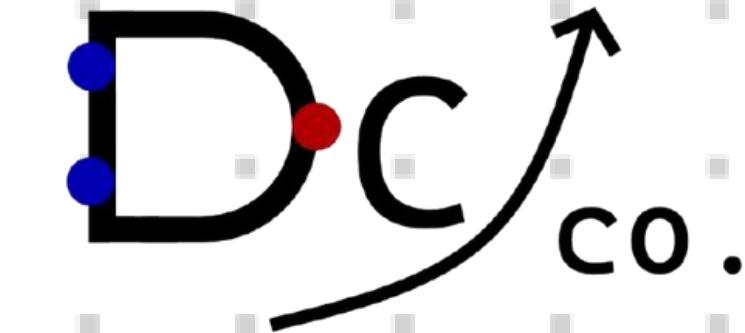
2025

All Class



Mini CPU

Drop Computing  
Co.



# Steps

- 1• Understand assignment & prepare a plan
- 2• Draw ASM Chart
- 3• Implement OpCode Process
- 4• Implement Control Unit & Data Path
- 5• Connect all circuit & Run Tests

# Design Concept

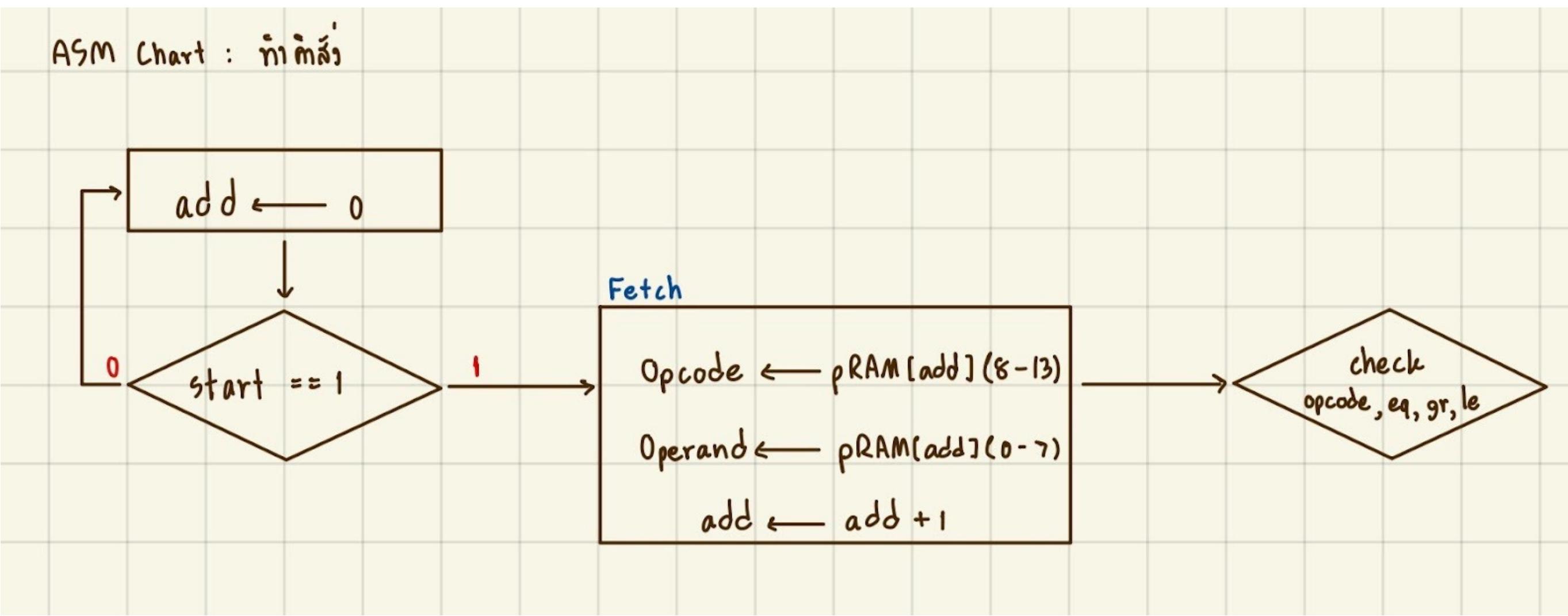
Phase :

Input

Execution

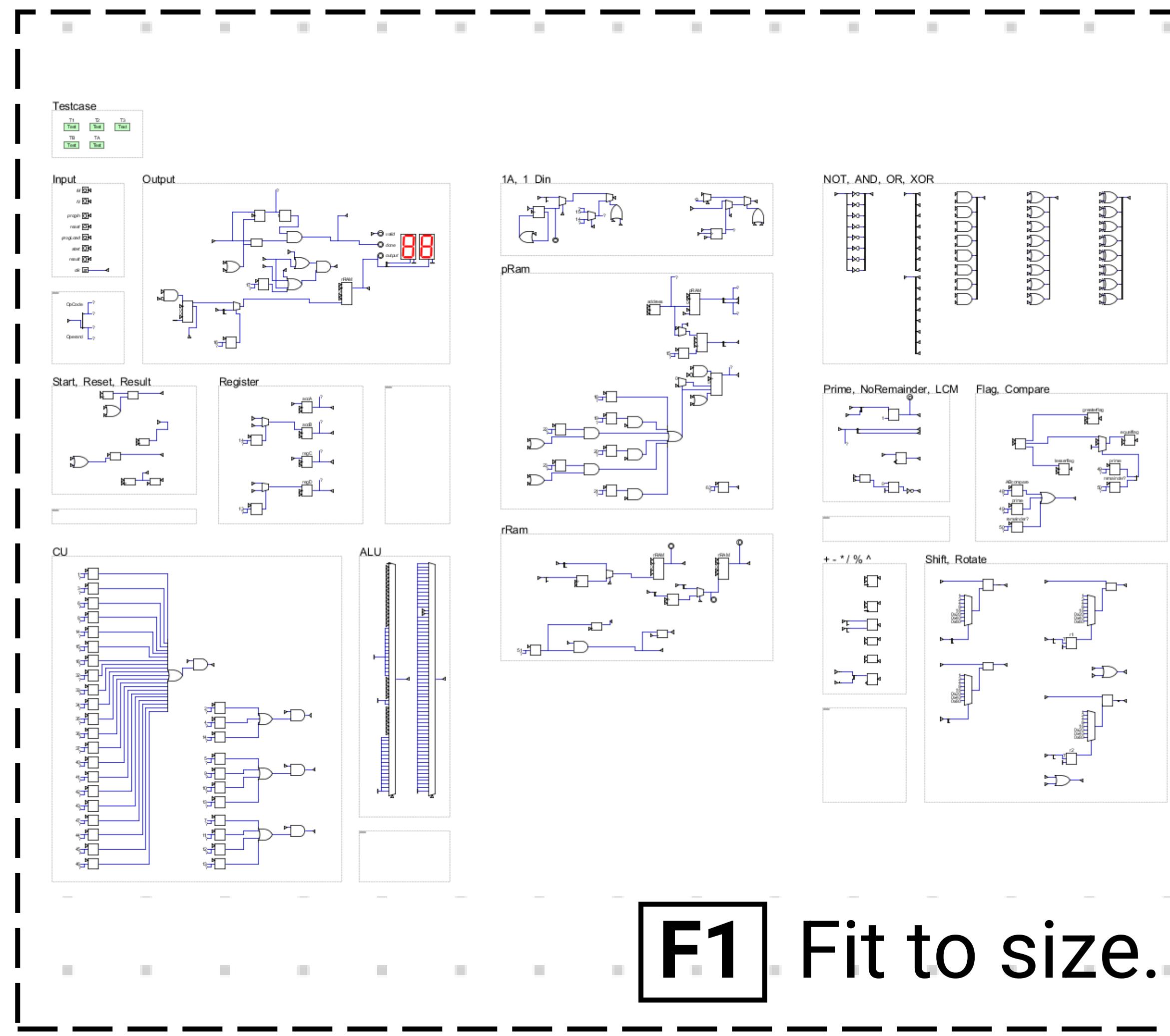
Output

ASM  
Chart  
Example



# Overview

Circuit Statistics				
Component	In...	Bits	Ad...	N...
Add		8		1
And	2	1		22
Comparator	2			1
Comparator	6			50
Comparator	8			1
Counter				3
CounterPreset				2
Div		8		5
FlipflopD	1			5
FlipflopRSAsync				4
Mul	4			1
Mul	8			2
Multiplexer	2	4	1	6
Multiplexer	2	8	1	7
Multiplexer	4	1	2	1
Multiplexer	8	8	3	4
Multiplexer	64	8	6	2
Not	1	1		9
Or	2	1		17
Or	2	8		2
Or	3	1		3
Or	4	1		2
Or	6	1		1
Or	21	1		1
RAMDualAccess	8	4		3
RAMDualAccess	14	8		1
RAMDualPort	14	8		1
ROM	1	8		1
ROM	8	6		1
ROM	16	16		1
Register	1			3
Register	4			2
Register	8			4
Sub	3			2
Sub	8			2
XOr	2	1		16



F1

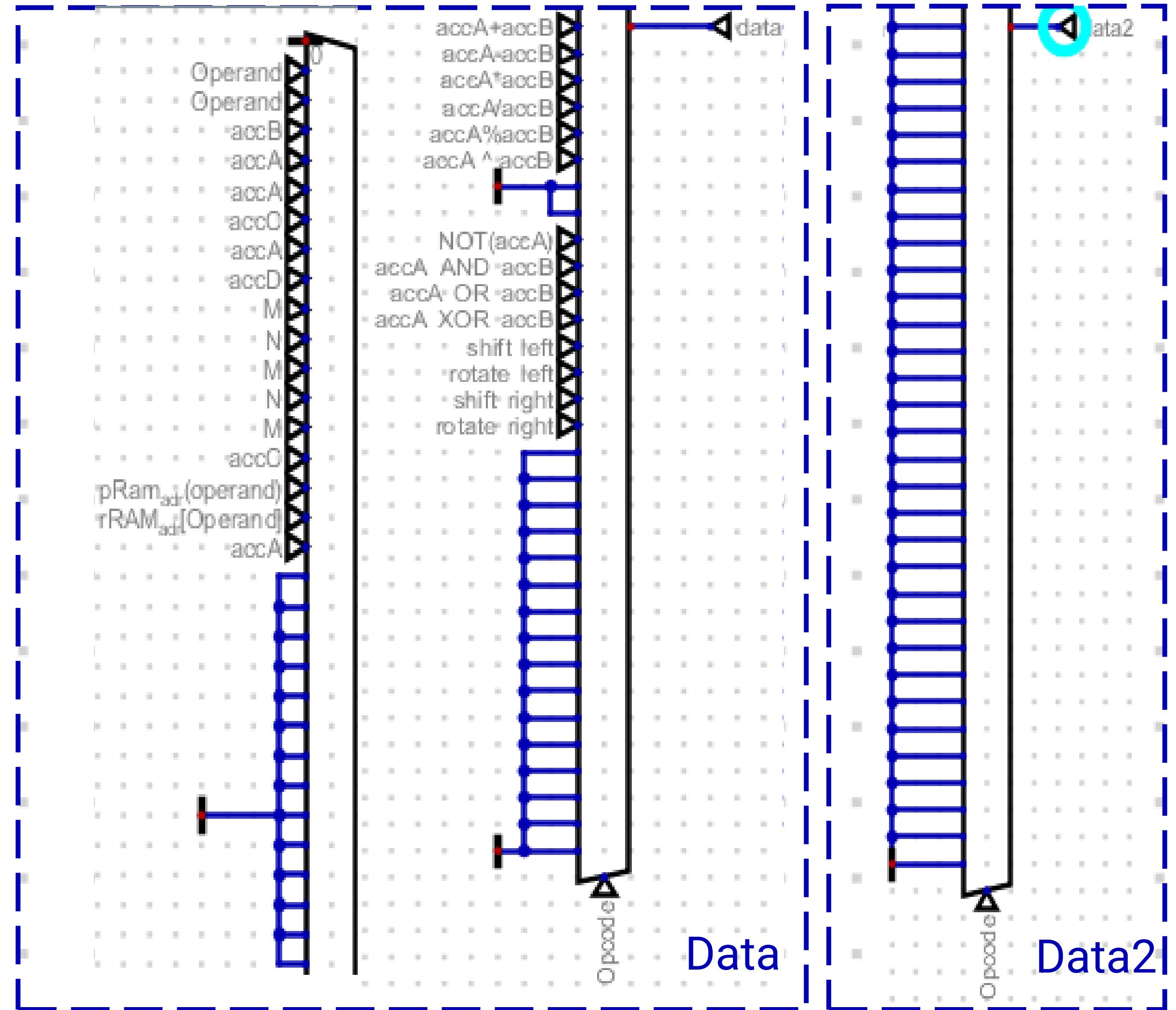
Fit to size.

# Data Path

## – Data Bus –

Select what data to write back with some of operation requiring to write back 2 data

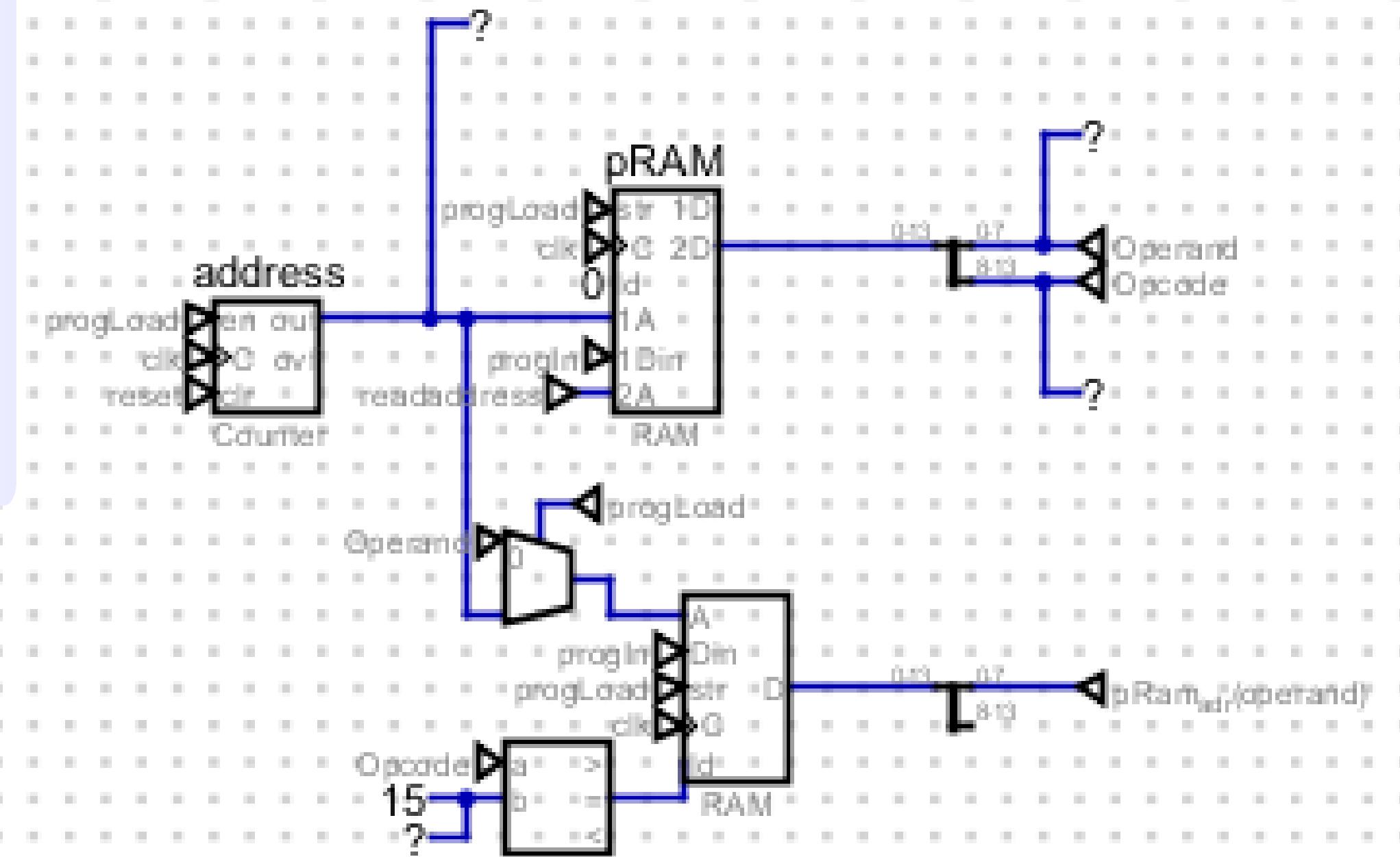
Ex. For OpCode 13,  
 $(accA \leftarrow regC, accB \leftarrow regD)$ , we select regC as data 1 and regD as data 2



# Data Path

## - Memory --

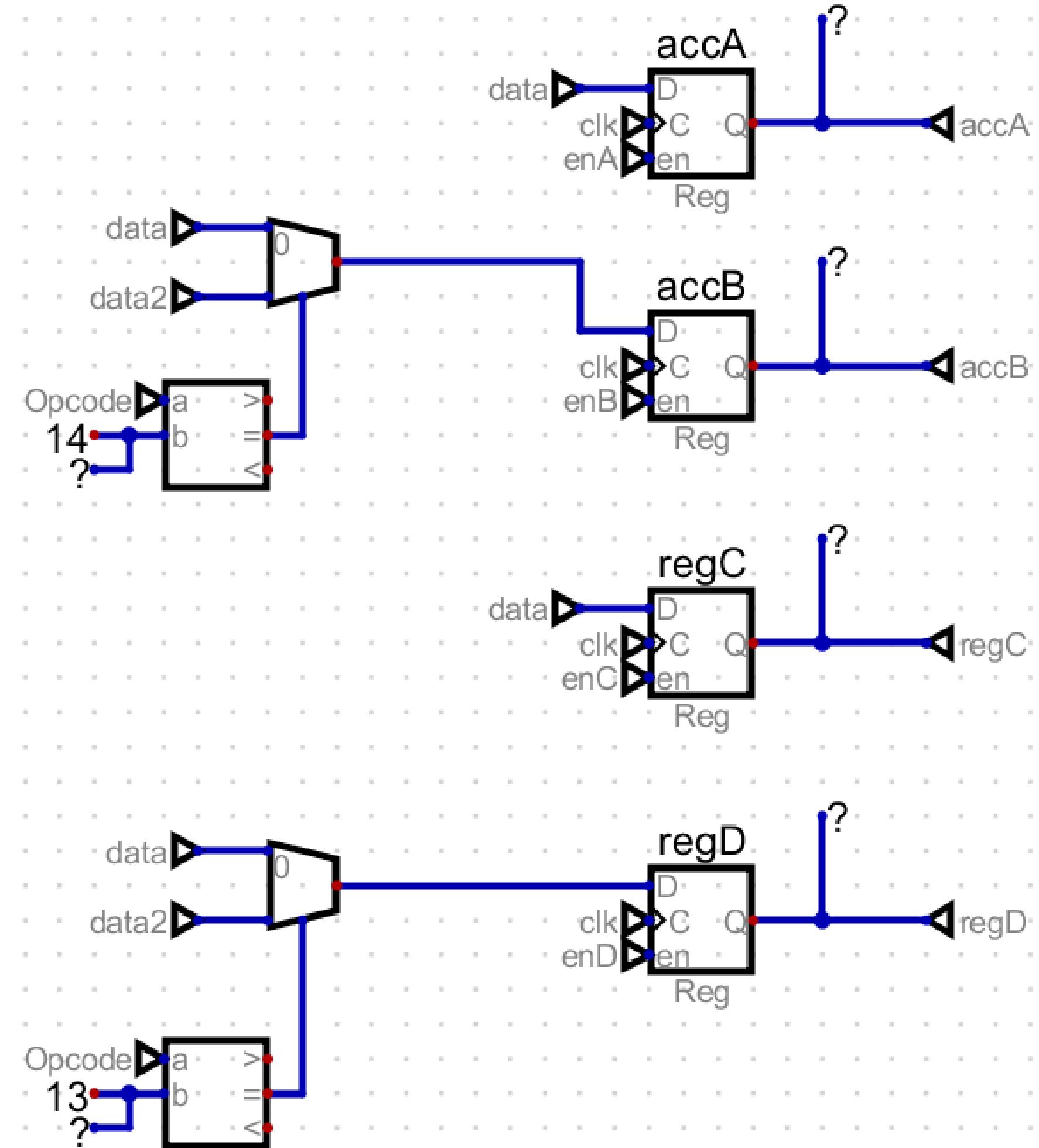
Save each progIn in pRam  
when progLoad is on while  
start, pRam will send out the  
code to be split into  
Opcode,Operand



# Data Path

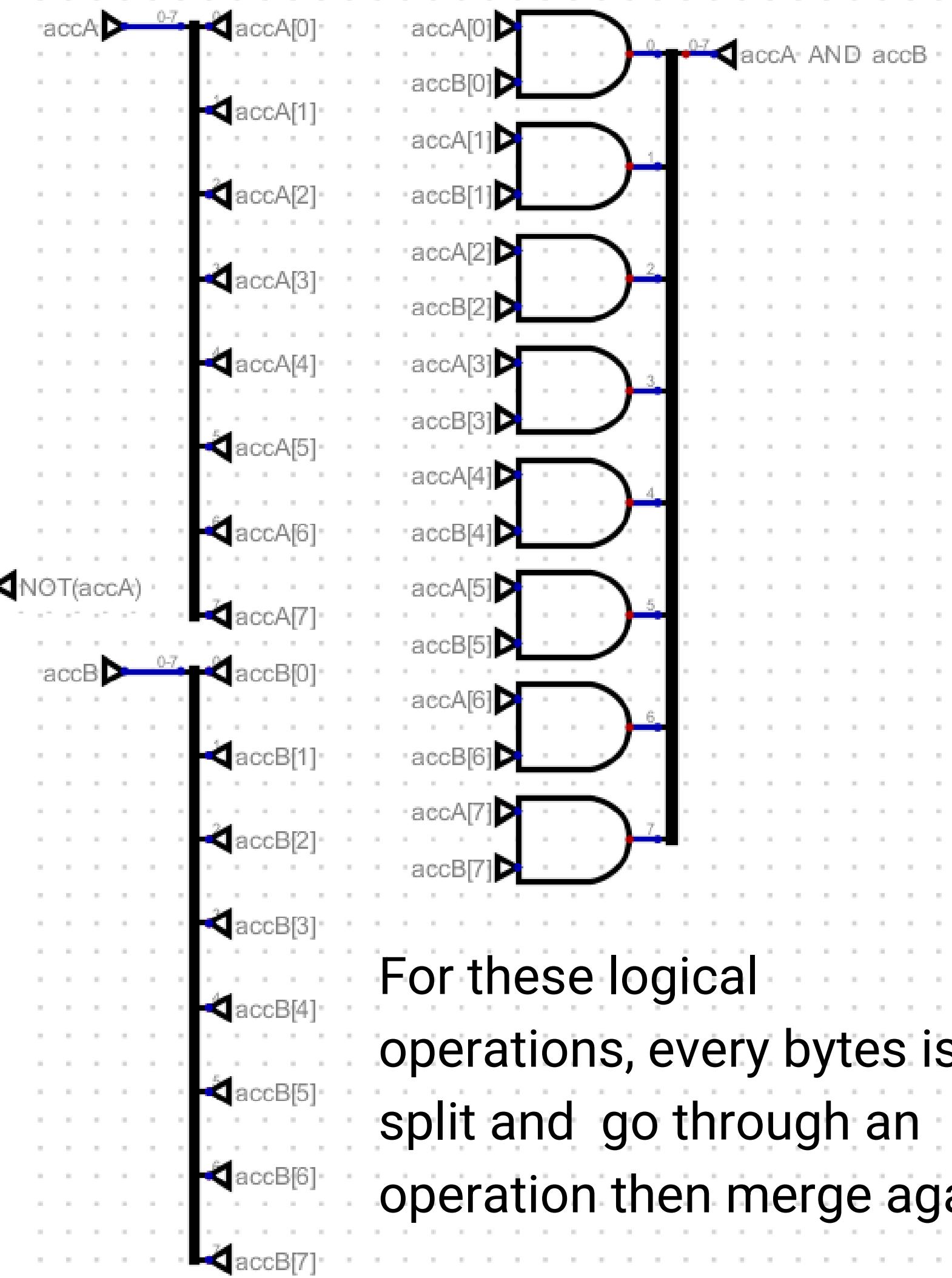
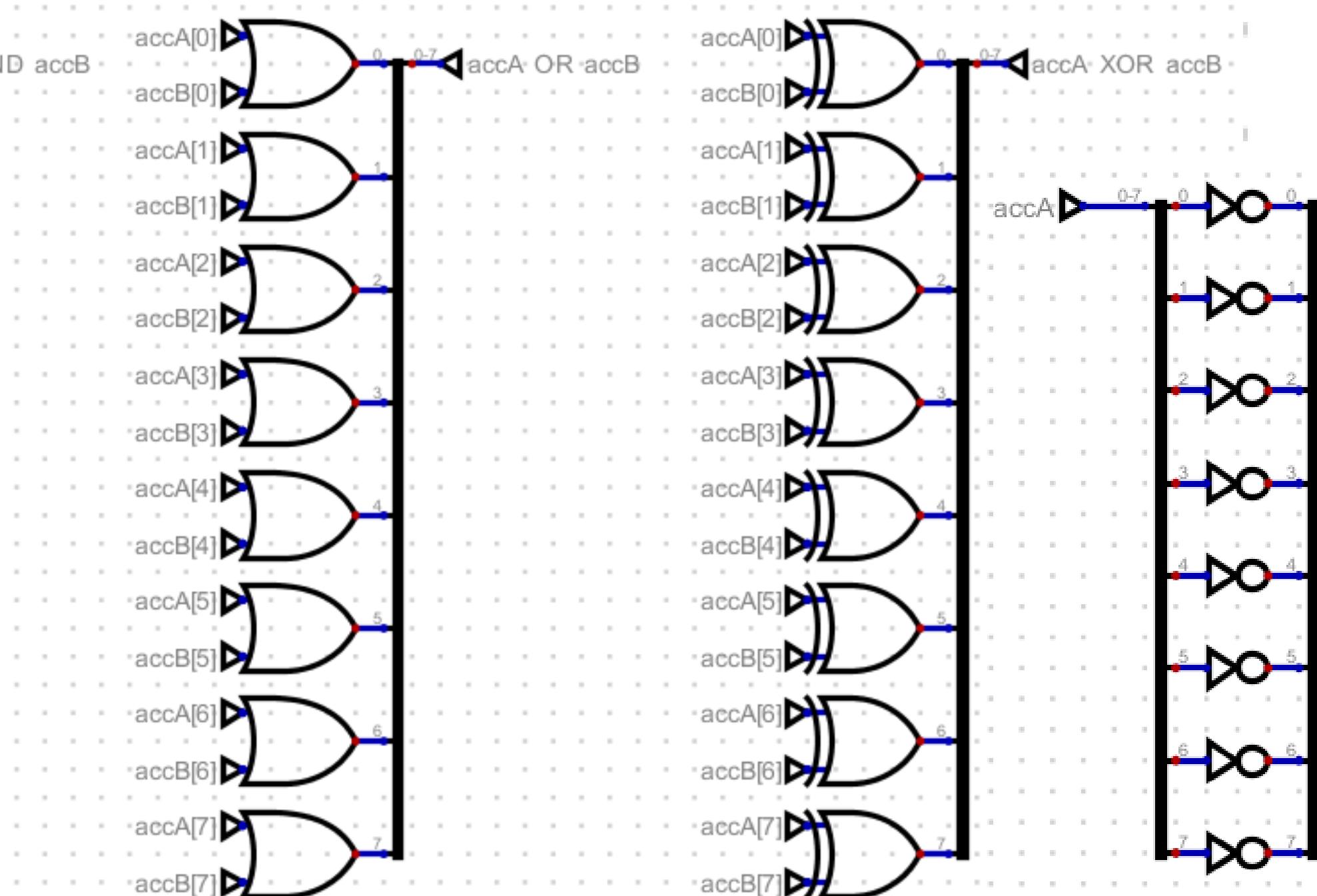
## - Data Register --

For OpCode 13,14, which two input are written back, there's a selector that will choose data2 (N, regD) instead



# ALU

- NOT, AND, OR, XOR --

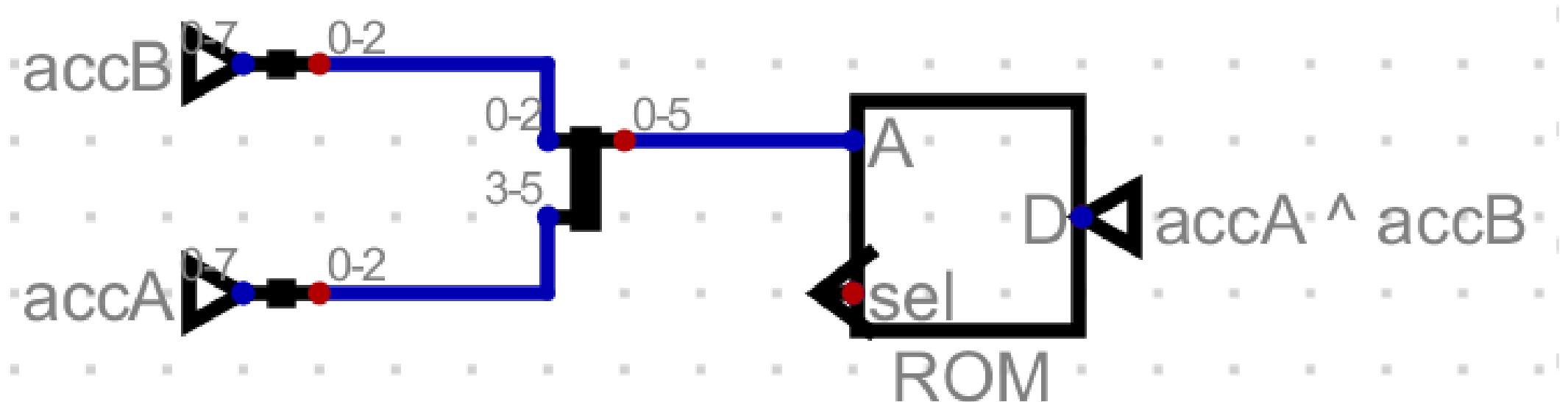
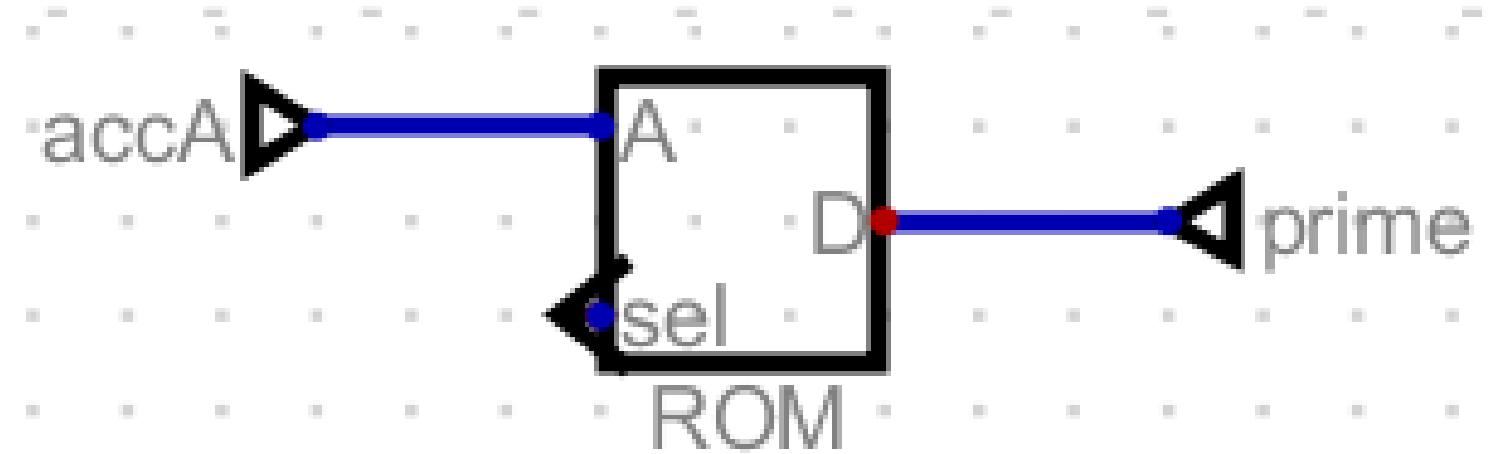
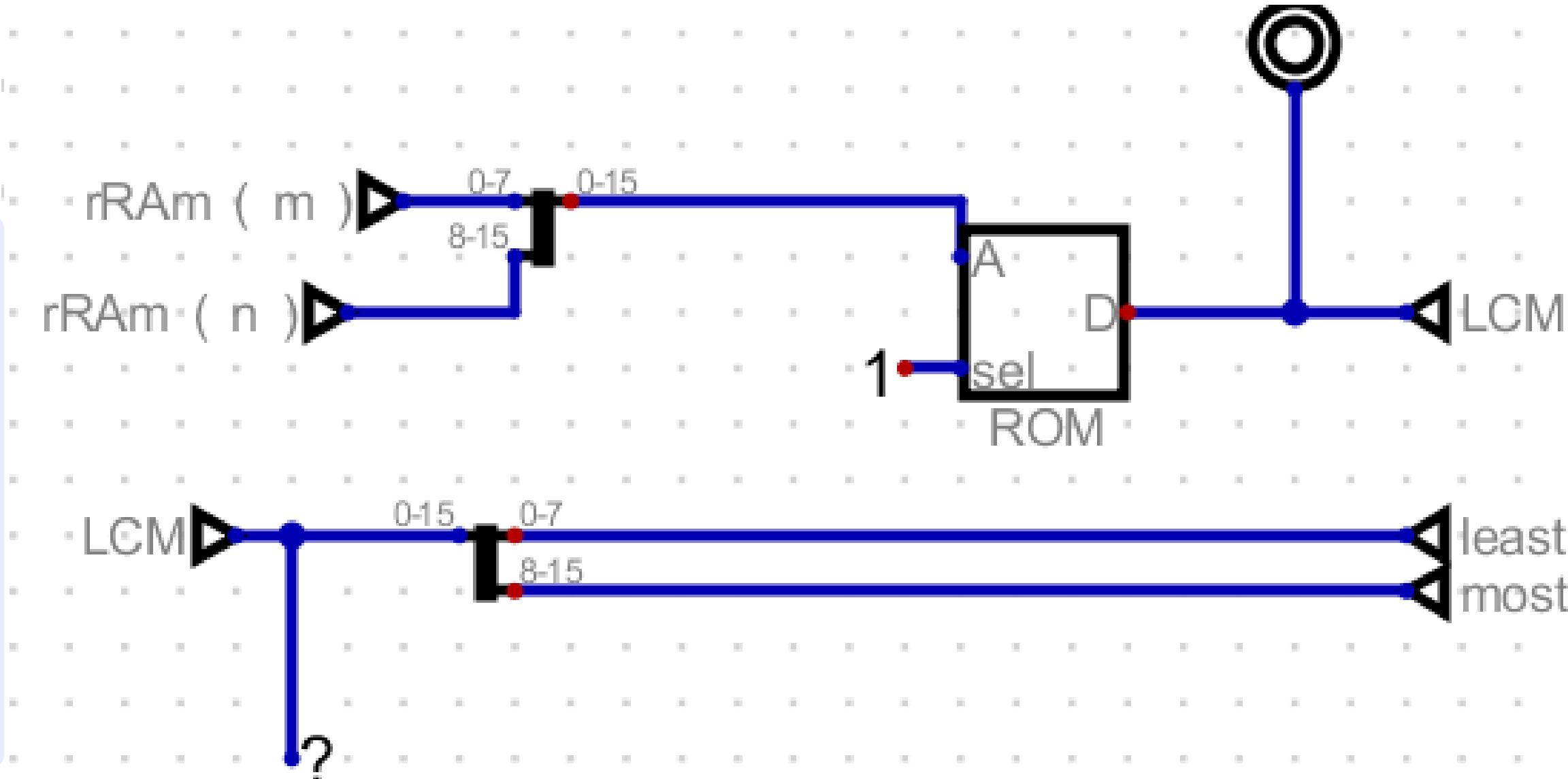


For these logical operations, every byte is split and go through an operation then merge again

# ALU

- Prime, Power, LCM --

Putting all result in ROM is easier with the time left than making and correcting these operations

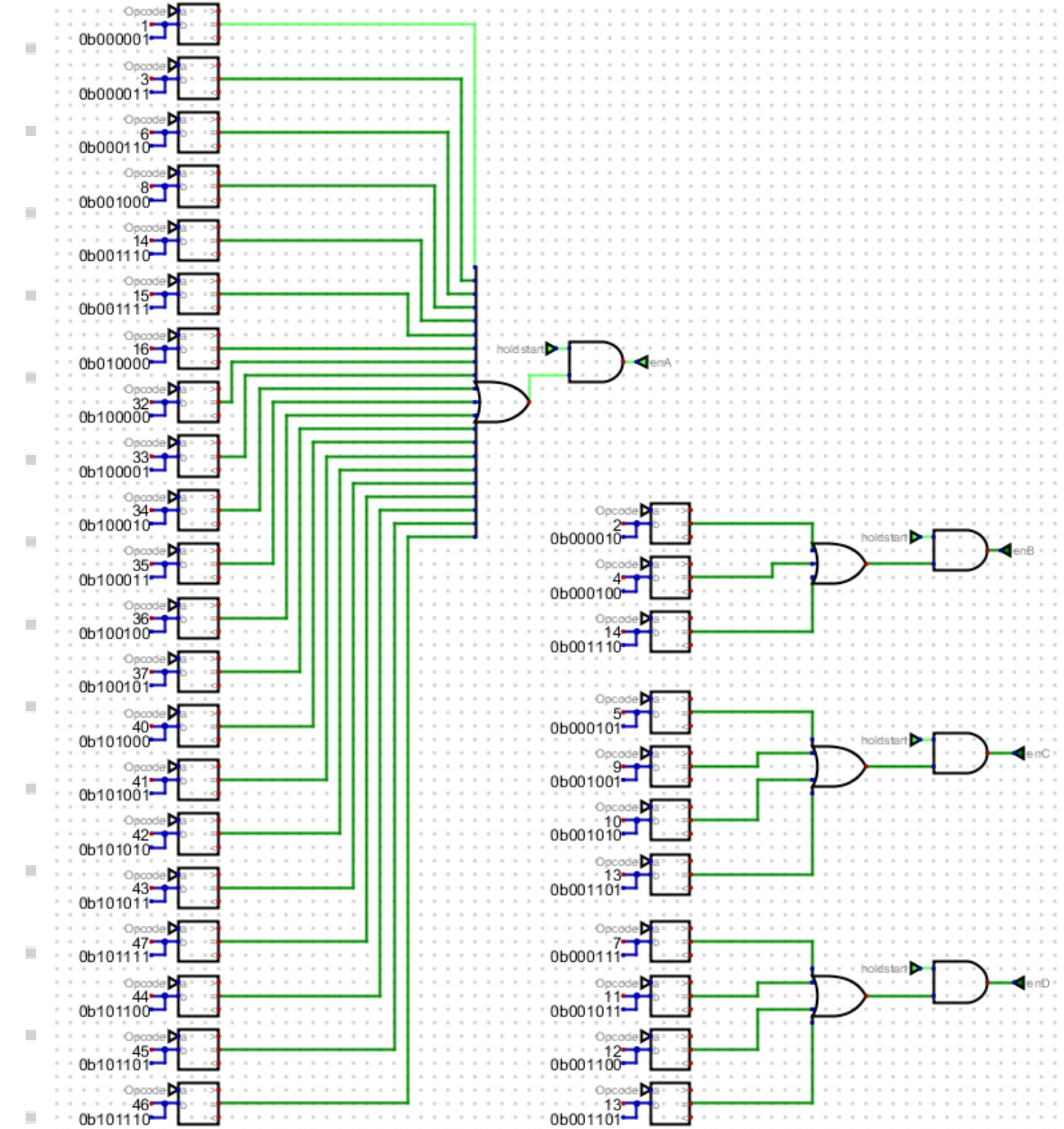
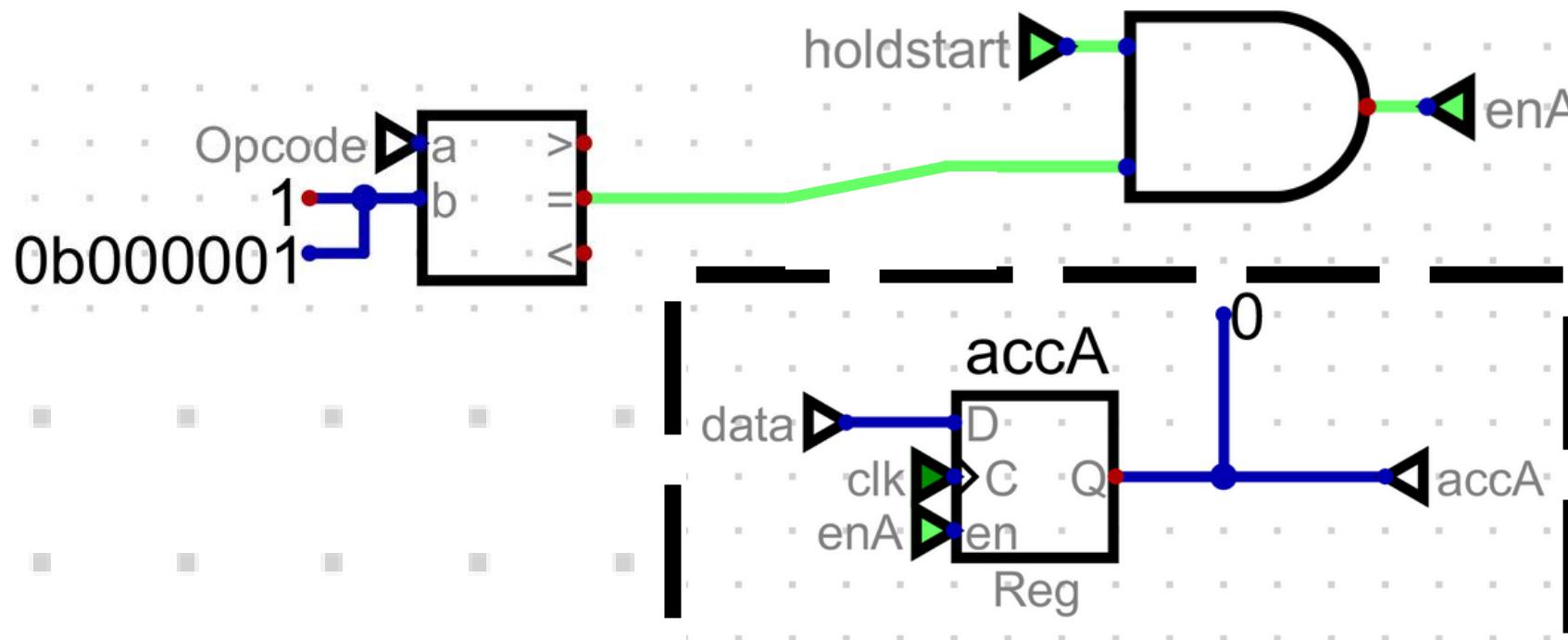


# Control Unit

## - OpCode Decoder -

Select what register to write back to by enable their signal (enA, enB, enC, enD,)

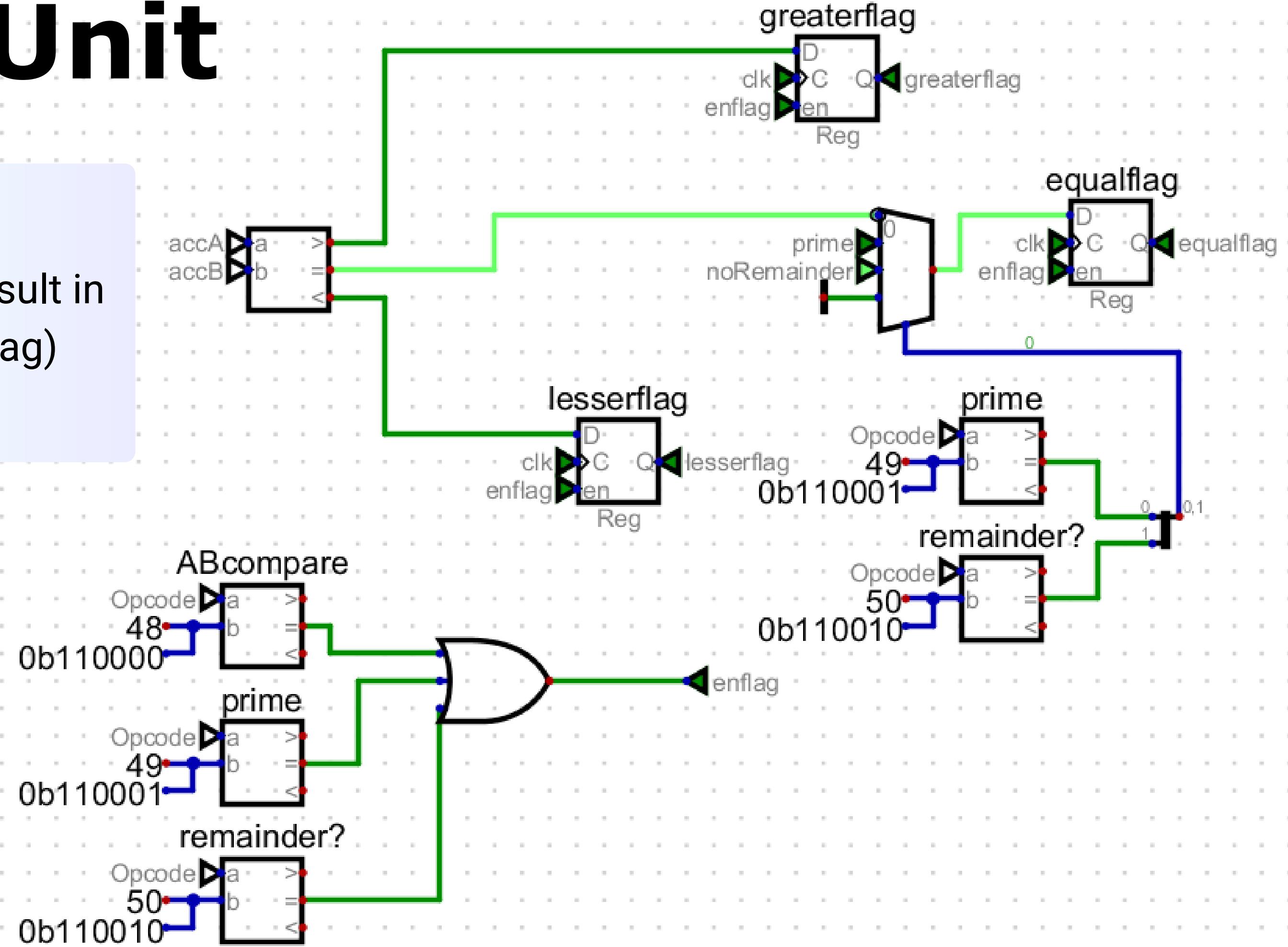
### Ex. For OpCode 1 (accA $\leftarrow$ Operand)

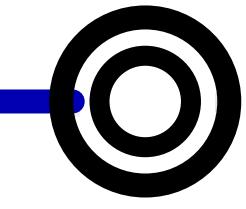


# Control Unit

## - Flag Checker --

Identify if given opcode result in comparison flag first (enflag) then assign a flag value





**END**