Microarchitecture of Processing System Circuit

Ghassan Arnouk

SYSC 3006A
Summer 2020
Lab 3 Report
Group 1

Instructor: Michel Sayde
TA: Khalid Almahrog
Submitted: 2020/05/30

1 First Part

1.1 Fill in the Part 1 Instruction Encoding Table for the following instructions.

Table 1: Part 1 Instruction Encoding

OPR	Instruction	Encoding (Hex)
NOT	R5 ← NOT (R4)	0750 4000
SUB	R7 ← R6 - R5	0276 5000
NOP	NOP	0000 0000
MOV	R0 ← R7	0300 7000

1.2 Complete the FSM Output ROM for part 1.

Table 2: FSM Output ROM for Part 1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
State Hex encoding	Unused (0)	IRCE	PCOE	C1OE	AADD	MARCE	MAROE	MDRCE	MDROE	MDRget	MDRput	IBRead	IBWrite	AOP	ANOP	DR	SXR	SYR	RegSEL	RegLD	T1CE	T1OE	T2CE	T2OE	Q7+	Q6+	Q5+	Q4+	Q3+	Q2+	Q1+	Q0+	Hex Encoding
F0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0002 0001
F1 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0002 0002
F2 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0002 0003
Decode 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0002 0007
E0 4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	1	0	0	0	0	0	0	0	0	1	0	1	0002 B805
E1 5	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	1	1	0	0	0	0	0	0	1	1	0	0004 7606
E2 6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0003 2100
Dead 7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0000 0007

1.3 Complete the FSM Decode ROM Tables (this table is same for Part 1 and 2 of this lab).

Table 3: FSM Decode ROM

Instruction	Address (Hex)	Contents (Hex)
NOP	00	00
ADD	01	04
SUB	02	04
MOV	03	05
AND	04	04
OR	05	04
XOR	06	04
NOT	07	05

- The design requires that Decode state is 0x:03
- The content is **04** when two words are involved in the ALU operation.
- However, the content is **05** when a single word is involved in the ALU operation. In this case, the execution state, E0, is skipped and ignored.

1.4 Save your circuit as Lab-3-Part1.circ and submit it with your assignment for verification and to get the marks for this section.

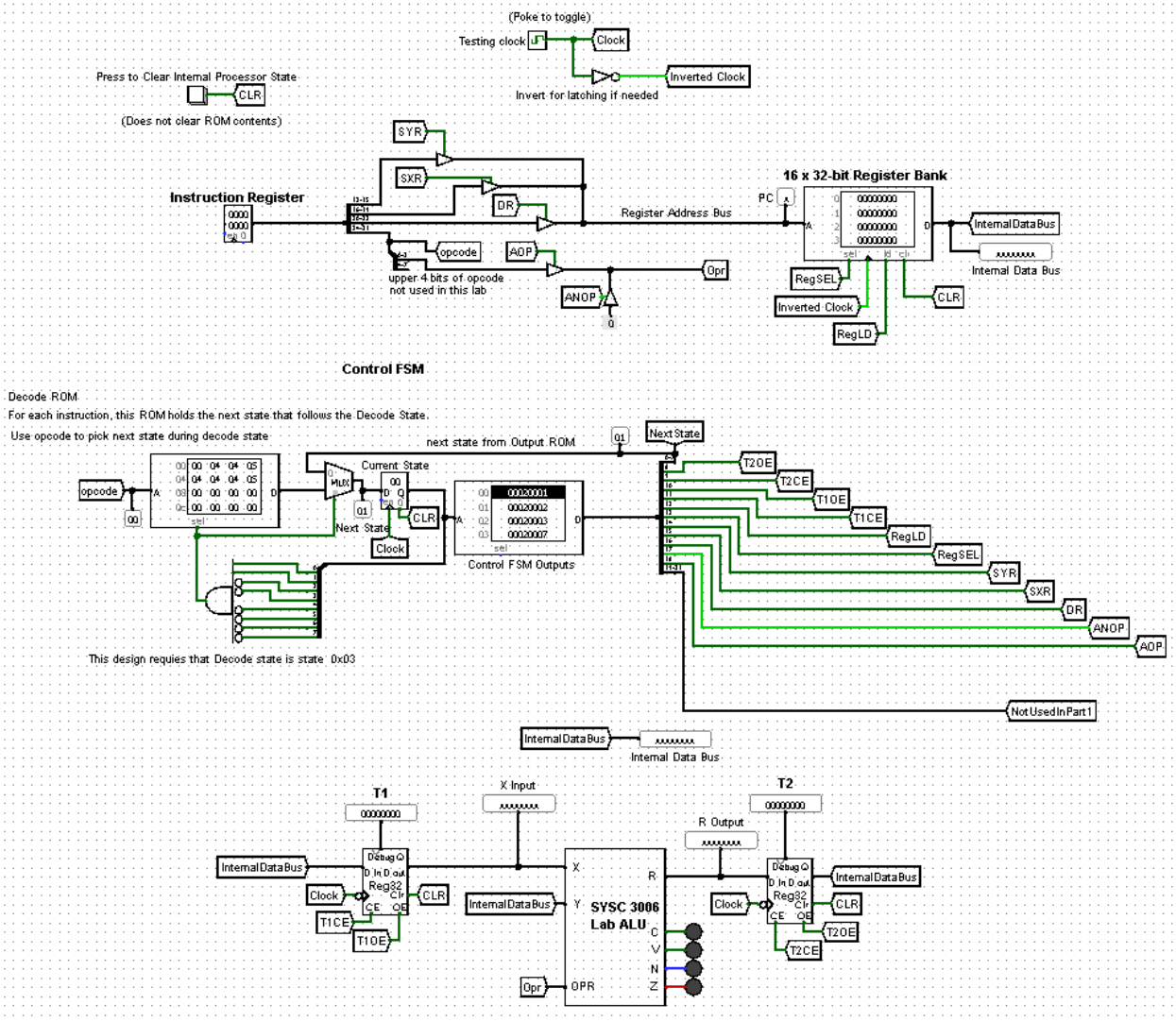

Figure 1: Logisim Circuit of Part 1

1.5 Same as you did in lab 2, execute the instructions in the table above in the given sequence with all registers initially containing 0x0. Log the execution of the sequence on your implementation to validate the execution of the required instructions and show the results here. (The simulation log should include: Current State, IR, PC, registers, and Next State. Set the log radix to hex).

Table 4: Simulation Log Table of the operation R5 ← NOT (R4)

Current State	Instruction Register	PC	T1	T2	Next State
00	07504000	x	00000000	00000000	01
01	07504000	x	00000000	00000000	02
02	07504000	x	00000000	00000000	03
03	07504000	x	00000000	00000000	05
05	07504000	4	00000000	00000000	06
05	07504000	4	00000000	ffffffff	06
06	07504000	5	00000000	ffffffff	00
00	07504000	x	00000000	ffffffff	01
01	07504000	x	00000000	ffffffff	02
02	07504000	x	00000000	ffffffff	03
03	07504000	x	00000000	ffffffff	05
05	07504000	4	00000000	ffffffff	06
06	07504000	5	00000000	ffffffff	00

Table 5: Simulation Log Table of the operation R7 ← R6 - R4

Current State	Instruction Register	PC	T1	T2	Next State
06	02765000	2	00000000	ffffffff	00
06	02765000	7	00000000	ffffffff	00
00	02765000	x	00000000	ffffffff	01
01	02765000	x	00000000	ffffffff	02
02	02765000	x	00000000	ffffffff	03
03	02765000	x	00000000	ffffffff	04
04	02765000	6	00000000	ffffffff	05
05	02765000	5	00000000	ffffffff	06
05	02765000	5	00000000	00000001	06
06	02765000	7	00000000	00000001	00

Table 6: Simulation Log Table of the operation NOP

Current State	Instruction Register	PC	T1	T2	Next State
06	00000000	0	00000000	00000001	00
00	00000000	x	00000000	00000001	01
01	00000000	x	00000000	00000001	02
02	00000000	x	00000000	00000001	03
03	00000000	x	00000000	00000001	00
00	00000000	x	00000000	00000001	01
01	00000000	x	00000000	00000001	02
02	00000000	x	00000000	00000001	03
03	00000000	x	00000000	00000001	00

Table 7: Simulation Log Table of the operation R0 ← R7

Current State	Instruction Register	PC	T1	T2	Next State
00	03007000	x	00000000	00000001	01
01	03007000	x	00000000	00000001	02
02	03007000	x	00000000	00000001	03
03	03007000	x	00000000	00000001	05
05	03007000	7	00000000	00000001	06
06	03007000	0	00000000	00000001	00
00	03007000	x	00000000	00000001	01

2 Second Part

2.1 Tables and Figures

2.1.1 Complete the Part 2 FSM Output ROM Table.

Table 8: FSM Output ROM for Part 2

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
State Hex encoding	Unused (0)	IRCE	PCOE	C1OE	AADD	MARCE	MAROE	MDRCE	MDROE	MDRget	MDRput	IBRead	IBWrite	AOP	ANOP	DR	SXR	SYR	RegSEL	RegLD	T1CE	T1OE	T2CE	T2OE	Q7+	Q6+	Q5+	Q4+	Q3+	Q2+	Q1+	Q0+	Hex Encoding
F0 0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	2402 3801
F1 1	0	0	0	1	1	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	1b10 0602
F2 2	0	1	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	40C2 0003
Decode 3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	1	1	2002 2107
E0 4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	1	0	0	0	0	0	0	0	0	1	0	1	0002 B805
E1 5	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	1	1	0	0	0	0	0	0	1	1	0	0004 7606
E2 6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0003 2100
Dead 7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0000 0007

2.1.2 Complete the Part 2 Main Memory Instructions Table for the following instructions.

Table 9: Main Memory Instruction Table

Instruction	Address (Hex)	Contents (Hex)
R10 ← R1 OR R2	00	05A1 2000
R11 ← R2 - R10	01	02B2 A000
R12 ← NOT (R11)	02	07C0 B000
R13 ← R0 + R12	03	01D0 C000

2.2 Circuit Wiring

2.2.1 Show below a screenshot of the new control FMS outputs that you added in the Logisim circuits, and a short description about each output.

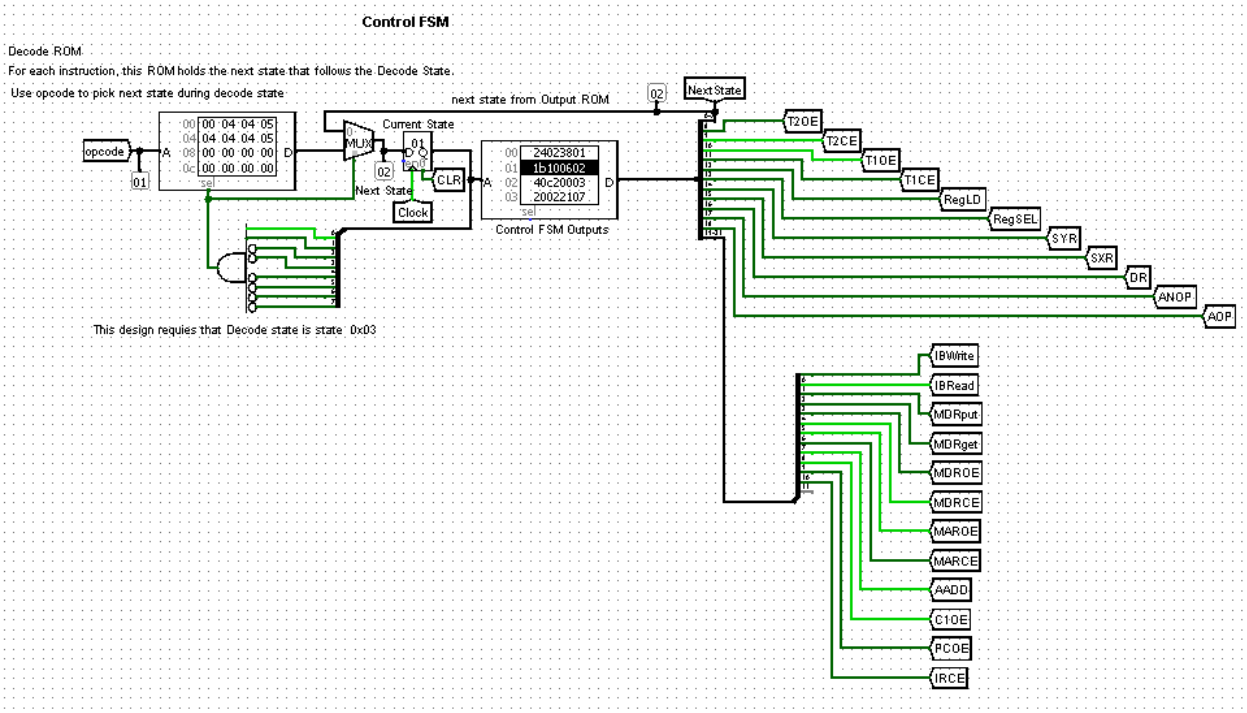

Figure 2: New Control FSM Circuit

- IBWrite: allows the register within the interconection bus to be written to
- IBRead: allows the register within the interconection bus to be read from
- MDRput: hold data exchanged on the interconnection data bus written to main memory
- MDRget: hold data exchanged on the interconnection data bus read from main memory
- MDROE: determines when the data is outputted from the MDR register
- MDRCE: determines when the data is inputted in the MDR register
- MAROE: determines when the data is outputted from the MAR register
- MARCE: determines when the data is outputted from the MAR register
- AADD: provides the ALU with the addition operation
- C1OE: inputs 0x00000001 to your internal data bus which then gets added to the PC value
- PCOE: introduces new constant to encode the PC's register address (address of R15)
- IRCE: introduces signal from internal data bus to the instruction register

2.2.2 Show here a screenshot of the new hardware components that you added in the Logisim circuits. Include a short description about each component function.

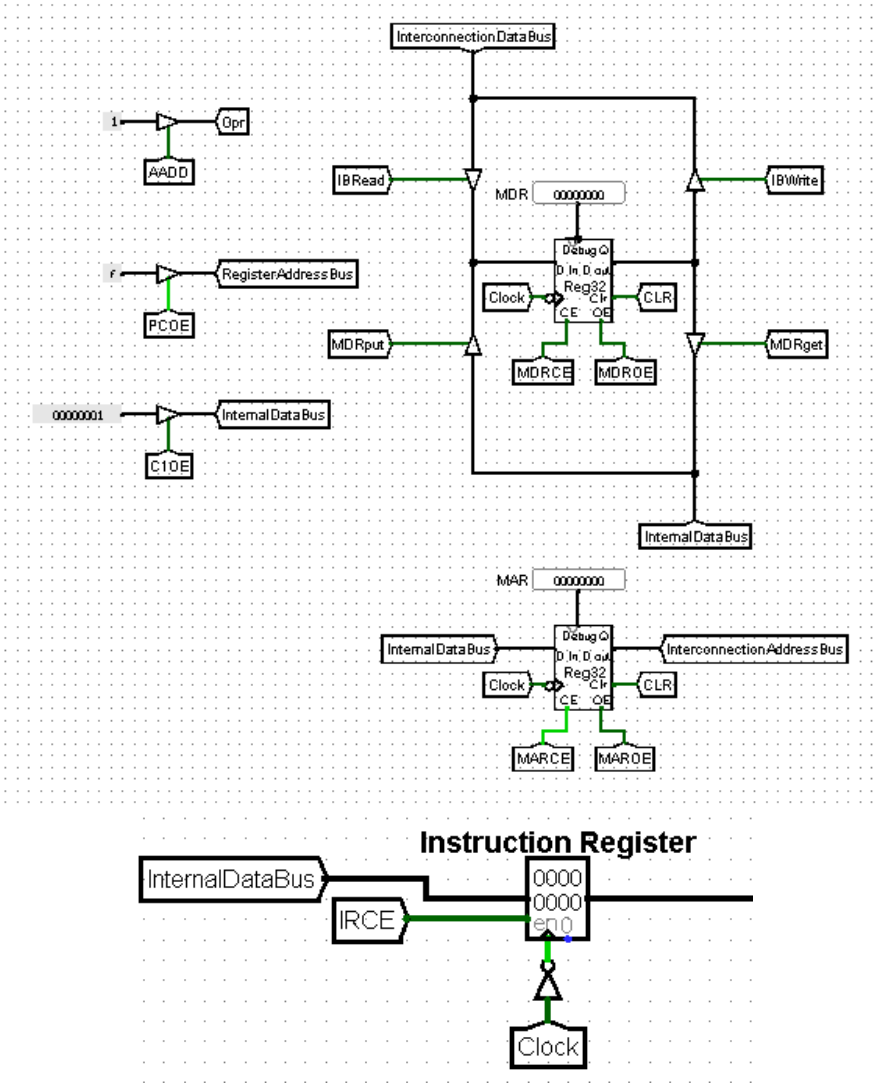

Figure 3: New Hardware Components added to the Logisim Circuit

A value is inputted from the internal data bus into the MDR and then outputted to the interconnection data bus. A value can also be read from the interconnection data bus to the register and then passed to the internal data bus.

The internal data bus goes to the MAR and then passed to the interconnection address bus. The register holds the address that is accessed by the interconnection address bus when outputted.

AADD provides the ALU with the addition operation which replaces the operation code that is inputted from the instruction register data. This allows the ALU to add 1 to the PC value.

PCOE introduces new constant to encode the PC's register address (address of R15). C1OE inputs 0x00000001 to your internal data bus which then gets added to the PC value.

Connecting IR to the internal data bus and adding IRCE control and Inverted Clock to allow fetching the Instruction from the internal data into IR at the falling edge of the clock.

2.2.3 Include a copy of your Lab3-Part-2.circ circuit with your submission. We must verify your circuit functionality in order to assign you marks for this part (c).

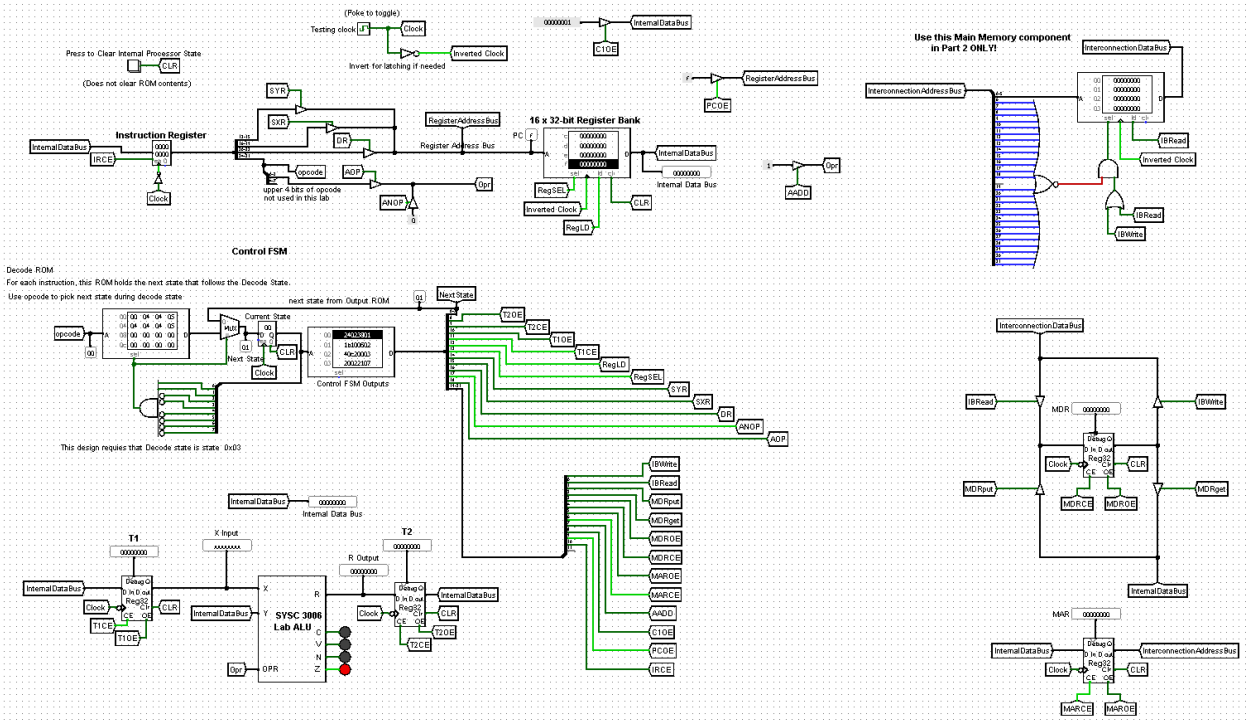

Figure 4: Logism Circuit of Part 2

2.3 Execution Test

2.3.1 Log the execution of the sequence on your implementation to validate the execution of the required instructions and show the results here.

Table 10: Simulation Log Table of the Part 2 Circuit

RAM(1050,270)[10]	RAM(1050,270)[11]	RAM(1050,270)[12]	RAM(1050,270)[13]	RAM(1050,270)[15]
00000000	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000001
f000000f	00000000	00000000	00000000	00000001
f000000f	00000000	00000000	00000000	00000002
f000000f	ffffffff	00000000	00000000	00000002
f000000f	ffffffff	00000000	00000000	00000003
f000000f	ffffffff	0000000e	00000000	00000003
f000000f	ffffffff	0000000e	00000000	00000004
f000000f	ffffffff	0000000e	0000000f	00000004

2.3.2 Compare the concept used here to your lab 2-part 2, briefly describe here what is the advantage of the concept here over lab2-part2?

The advantage of this concept is that it allows the user to not change the values in the instruction register. The data is read from main memeory and the circuit will automatically performs the instructions on the data. This is much quicker to complete the exact same instructions in comparison to lab2-part 2. This setup also allows us to see whenever an operation occurs. Due to the fact that there is a counter at memory location R15, it increments by one every time we go through all the states. This helps to keep track of the total number of operations that have occurred and lets stop at a desired operation.