Carleton University

Department of Systems and Computer Engineering

SYSC 3006 (Computer Organization) summer 2020

Lab / Assignment 4 – Answers file

Student Name: ID#:

Part 1 – [0.75-mark/5]

1-1 Control FSM Output Table

[0.75-mark] Complete the provided Control FSM Output Table for Part 1 for the Fetch, Decode, and Execution States for opcodes 0x01 (ADD) through 0x07 (NOT).

FSM Output ROM Table: **Fetch, Decode, and Execution States for opcodes 0x01 (ADD) through 0x07 (NOT)**

	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 000	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 01	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 02	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 03	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 04	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 05	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 06	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 07	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

Part 2 – [0.75-mark/5]

2.1 - Control FSM Output Table

[0.75-mark] Complete the provided Control FSM Output Table for Part 2 for NOP Instruction Execution 3 States. This table will extend the Control FSM Output Table for Part 1 (same FSM Output ROM).

FSM Output ROM Table: **NOP Instruction Execution States**

	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**
E3 08	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	0	0	1	0000 0009
E4 09	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	0	1	0	0000 000A
E4 0A	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	0	0	0	0000 0000
	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	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	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	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	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	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Part 3 – [2.0-mark/5]

3.1 - Control FSM Output Table

[0.75-mark] Complete the provided Control FSM Output Table for Part 3 for NEG Instruction Execution States. This table will extend the Control FSM Output Table for Part 1 and 2 (same FSM Output ROM).

FSM Output ROM Table: **NEG Instruction Execution States**

	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**
E5 0B	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	1	0	0 1	1	0	0	0	0	0	1	1	0	0	0005 320C 0005 360C
E6 0C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	1	0	0	1	0	0	0	0	1	1	0	1	0003 290D
E7 0D	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	1	1	0	1800 060E
E8 0E	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
	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	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	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	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	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

3.2 -

[0.50-mark] Describe how NEG instruction is executed at each execution state.

First execution state we do NOT operation over Regsiter content (ALU input Y and OPR 111) (first complement)

Second execution state we store back the result into same register and into T1 at same time

Third execution state we assert 1 C1OE into the Data bus (ALU input Y), and we assert 1 AADD (1) Opr and the result is stored at T2 (2^{nd} complement)

Fourth execution state we store back T2 (2^{nd} complement) into same register to complete the NEG operation of that register

3.3 – Decode ROM Table

[0.75-mark] Complete the provided FSM Decode ROM Table to show any entries that must be programmed (for all parts).

FSM Decode ROM Table

Instruction	Address (hex)	Contents (hex)
NOP	00	08
ADD	01	04
SUB	02	04
MOV	03	05
AND	04	04
OR	05	04
XOR	06	04
NOT	07	05
NEG	17	0B

Part 4 – Execution test [1.5-mark/5]

4.1 - Instructions Table

[0.75-mark] Complete the provided Main Memory Table to contain the encodings of the Test Program instructions as indicated. Then program the words of this table into the Main Memory. Be sure to include the Main Memory contents exactly as given in the table.

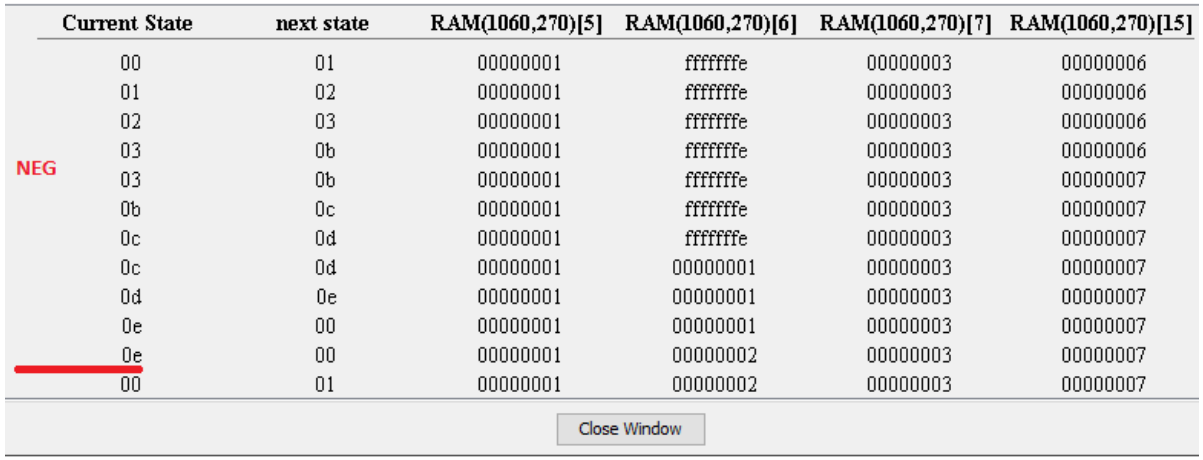
Main Memory Table

Address (hex)	Instruction	Encoding (hex)	Results
0	MOV R5 ← [R15]	**0350F000**	**R5 = 1**
1	NOT R6 ← NOT [R5]	**07605000**	**R6 = -2**
2	MOV R7 ← [R15]	**0370F000**	**R7 = 3**
3	SUB R15 ←[R7] – [R6]	**02F76000**	**R15 = 5**
4	EEBB FFFF	Illegal instruction	skipped
5	NOP	**00000000**	**NOP**
6	NEG R6 ← – [R6]	**17600000**	**R6 = 2**

4.2 – Test Results

[0.75-mark] Cycle the System Clock through the execution of your Test program and show your logs here.

	Current State	next state	RAM(1060,270)[5]	RAM(1060,270)[6]	RAM(1060,270)[7]	RAM(1060,270)
	00	01	00000000	00000000	00000000	00000000
	01	02	00000000	00000000	00000000	00000000
MOV	02	03	00000000	00000000	00000000	00000000
	03	05	00000000	00000000	00000000	00000000
	03	05	00000000	00000000	00000000	00000001
	05	06	00000000	00000000	00000000	00000001
	06	00	00000000	00000000	00000000	00000001
	06	00	00000001	00000000	00000000	00000001
	00	01	00000001	00000000	00000000	00000001
	01	02	00000001	00000000	00000000	00000001
	02	03	00000001	00000000	00000000	00000001
NOT	03	05	00000001	00000000	00000000	00000001
	03	05	00000001	00000000	00000000	00000002
	05	06	00000001	00000000	00000000	00000002
	06	00	00000001	00000000	00000000	00000002
	06	00	00000001	fffffffe	00000000	00000002
	00	01	00000001	fffffffe	00000000	00000002
	01	02	00000001	fffffffe	00000000	00000002
	02	03	00000001	fffffffe	00000000	00000002
	03	05	00000001	fffffffe	00000000	00000002
MOV	03	05	00000001	fffffffe	00000000	00000003
	05	06	00000001	fffffffe	00000000	00000003
	06	00	00000001	fffffffe	00000000	00000003
	06	00	00000001	fffffffe	00000003	00000003
	00	01	00000001	fffffffe	00000003	00000003
	01	02	00000001	fffffffe	00000003	00000003
	02	03	00000001	fffffffe	00000003	00000003
	03	04	00000001	fffffffe	00000003	00000003
SUB	03	04	00000001	fffffffe	00000003	00000004
	04	05	00000001	fffffffe	00000003	00000004
	05	06	00000001	fffffffe	00000003	00000004
	06	00	00000001	fffffffe	00000003	00000004
	06	00	00000001	fffffffe	00000003	00000005
	00	01	00000001	fffffffe	00000003	00000005
	01	02	00000001	fffffffe	00000003	00000005
	02	03	00000001	fffffffe	00000003	00000005
	03	08	00000001	fffffffe	00000003	00000005
NOP	03	08	00000001	fffffffe	00000003	00000006
	08	09	00000001	fffffffe	00000003	00000006
	09	0a	00000001	fffffffe	00000003	00000006
	0a	00	00000001	fffffffe	00000003	00000006



Current State	next state	RAM(1060,270)[5]	RAM(1060,270)[6]	RAM(1060,270)[7]	RAM(1060,270)[15]
00	01	00000001	fffffffe	00000003	00000006
01	02	00000001	fffffffe	00000003	00000006
02	03	00000001	fffffffe	00000003	00000006
03	0b	00000001	fffffffe	00000003	00000006
03	0b	00000001	fffffffe	00000003	00000007
0b	0c	00000001	fffffffe	00000003	00000007
0c	0d	00000001	fffffffe	00000003	00000007
0c	0d	00000001	00000001	00000003	00000007
0d	0e	00000001	00000001	00000003	00000007
0e	00	00000001	00000001	00000003	00000007
0e	00	00000001	00000002	00000003	00000007
00	01	00000001	00000002	00000003	00000007

Close Window

Submission deadline

Must be submitted on cuLearn, locate (Assignment 4 submission) and follow instructions. Submission exact deadline (date and time) is displayed clearly within the Assignment 4 submission on cuLearn.

Note: If you have any question please contact your respective group TA (see TA / group information posted on cuLearn) or use Discord class server.

Good Luck