CSARCH2: Case Study 2

S12 GRP5: BRILLANTES, CHU, INFANTE, ONO, YOUNG

Things to be implemented in the minecraft world template:

Notes:

- Implement the Execution cycle for the other Assembly codes (the one above)

Notes from the masterclass lecture:

- Do not touch the fetch cycle/signals (I think these are the red blocks), changing this will not make the functions work)
- Control signals should be implemented properly in connecting with the ALU operations
- Basically connecting each microcode based on their opcode?
- Swap Out - swap input and output then we can output content of register

Implement decoder based with the ALU operations by the group

Specific Specs:

S12 - Group 5

Opcode	Instruction	Example Instruction	Example Opcode
0	HALT	HALT	0000 0000 0000 0000
1	MOV REG, REG	MOV R1, R2	0001 0001 0000 0010
2	MOV REG, POINTER	MOV R1, [R2]	0010 0001 0000 0010
3	XOR REG, REG	XOR R4, R5	0011 0100 0000 0101
4	MOV REG, IMMEDIATE	MOV R0, 0x05	0100 0000 0000 0101
5	MOV REG, ADDRESS	MOV R3, [0x02]	0101 0011 0000 0010
6	INC REG	INC R1	0110 0001 0000 0000
7	AND REG, REG	AND R4, R5	0111 0100 0000 0101
8	—	—	—
9	NOT REG	NOT R1	1001 0001 0000 0000
10	SUB REG, REG	SUB R3, R1	1010 0011 0000 0001
11	ADD REG, REG	ADD R3, R1	1011 0011 0000 0001
12	MOV ADDRESS, REG	MOV [0x02], R3	1100 0000 0010 0011
13	—	—	—
14	MOV POINTER, REG	MOV [R1], R2	1110 0001 0000 0010
15	ADD POINTER, REG	ADD [R4], R5	1111 0100 0000 0101

ALU Operations (S12 - GRP5):ee
LEGEND: Operations in our specs

ALU Operations	
000	ADD
001	SUB
010	NOR
011	XNOR
100	OR
101	XOR
110	AND
111	NAND

Microcode: Limit is 5 lines (According to PTS)

HALT, fetch cycle, WMFC, and END is already implemented

OP	Instructions	Minecraft
0	**HALT**	**1. RST_PC**
1	**MOV R1, R2** 1. R2_out, R1_in, END	**MOV REG REG** 1. REG_out, REG_in
2	**MOV R1, [R2]** 1. R2_out, MAR_in, READ, WMFC 2. MDR_out, R1_in, END	**MOV REG, PTR** 1. REG_out, MAR_in, READ 2. MDR_out, REG_in
3	**XOR R4, R5** 1. R5_out, Y_in 2. R4_out, Select Y, XOR, Z_in 3. Z_out, R4_in, END	**XOR REG, REG** 1. REG_out, Y_in 2. REG_out (SWP_out), Select_Y, XOR (101), Z_in 3. Z_out, REG_in
4	**MOV R0, 0x05**	**MOV REG, IMMEDIATE**

	1. IRDF_out, R0_in, END	1. IRDF_out, REG_in
5	**MOV R3, [0x02]** 1. IRAF_out, MAR_in, READ, WMFC 2. MDR_out, R3_in, END	**MOV REG, ADDRESS** 1. IRAF_out, MAR_in, READ 2. MDR_out, REG_in
6	**INC R1** 1. R1_out, Set carry in, ADD, Z_in 2. Z_out, R1_in, END	**INC REG** 1. REG_out, Set carry-in, ADD (000), Z_in 2. Z_out, REG_in
7	**AND R4, R5** 1. R5_out, Y_in 2. R4_out, Select Y, AND, Z_in 3. Z_out, R4_in, END	**AND REG, REG** 1. REG_out, Y_in 2. REG_out Select Y, AND (110), Z_in 3. Z_out, REG_in
8		
9	**NOT R1** 1. R1_out, NOT, Z_in 2. Z_out, R_in, END	**NOT REG** 1. REG_out, NOT/NAND(111), Z_in 2. Z_out, REG_in
10	**SUB R3, R1** 1. R1_out, Y_in 2. R3_out, Select Y, SUB, Z_in 3. Z_out, R3_in, END	**SUB REG, REG** 1. REG_out, Y_in 2. REG_out (SWP_out), Select_Y, SUB (001), Z_in 3. Z_out, REG_in
11	**ADD R3, R1** 1. R1_out, Y_in 2. R3_out, Select Y, ADD, Z_in 3. Z_out, R3_in, END	**ADD REG, REG** 1. REG_out, Y_in 2. REG_out (SWP_out), Select_Y, ADD (000), Z_in 3. Z_out, REG_in
12	**MOV [0x02], R3** 1. R3_out, MDR_in 2. IRAF_out, MAR_in, WRITE, WMFC, END	**MOV ADDRESS, REG** 1. REG_out, MDR_in 2. IRAF_out, MAR_in, WRITE
13		
14	**MOV [R1], R2** 1. R2_out, MDR_in 2. R1_out, MAR_in, WRITE, WMFC, END	**MOV PTR, REG** 1. REG_out, MDR_in 2. REG_out, MAR_in, WRITE, WMFC, END
15	**ADD [R4], R5** 1. R5_out, Y_in	**ADD PTR, REG** 1. REG_out, Yin

	2. R4_out, MAR_in, READ, WMFC 3. MDR_out, Select Y, ADD, Z_in 4. Z_out, MDR_in, WRITE, WMFC, END	2. REG_out (SWP_out), MAR_in, READ, WMFC 3. MDR_out, Select_Y, ADD (000), Z_in 4. Z_out, MDR_in, WRITE, WMFC, END

Test Cases:

1.1 AND REG, REG

initialize:

DATA	ADDR
0x1234	0x00AB
0x1357	0x00AC

0001001000110100
0001001101010111

0000000010101011
0000000010101100

Code	
MOV R0, 0xAB	0x?0AB
MOV R1, 0xAC	0x?1AC
MOV R2, [R0]	0x?200
MOV R3, [R1]	0x?301
AND R3, R2	0x?302

MOV REG, IMM
MOV REG, IMM
MOV REG, PTR
MOV REG, PTR
AND REG, REG

4 = 0x40AB
4 = 0x41AC
2 = 0x2200
2 = 0x2301
7 = 0x7302
HALT

DATA
0100000010101011
0100000110101100
0010001000000000
0010001100000001
0111001100000010
0000000000000000

ADDR
0000000000000000
0000000000000001
0000000000000010
0000000000000011
0000000000000100
0000000000000101

Results	
R0	0x00AB
R1	0x00AC
R2	0x1234
R3	0x1214
R4	0x0000
R5	0x0000
R6	0x0000
R7	0x0000
R8	0x0000
R9	0x0000
RA	0x0000
RB	0x0000
RC	0x0000
RD	0x0000
RE	0x0000
RF	0x0000

1.3 ADD PTR, REG

initial:

0001111101001101

DATA	ADDR
0x1F4D	0x00DA

0000000011011010

Code	
MOV R0, 0xDA	0x?0DA
MOV R1, [R0]	0x?100
MOV R2, R1	0x?201
ADD R2, R0	0x?200
XOR R1, R1	0x?101
MOV [0xEB], R2	0x?EB2
MOV R3, R0	0x?300
ADD [R3], R2	0x?302
MOV R5, [R3]	0x?503

MOV REG, IMM
MOV REG, ADDR
MOV REG, REG
ADD REG, REG
XOR REG, REG
MOV ADDR, REG
MOV REG, REG
ADD PTR, REG
MOV REG, PTR

4 = 0x40DA
2 = 0x2100
1 = 0x1201
11 (B) = 0xB200
3 = 0x3101
12 (C) = 0xCEB2
1 = 0x1300
15 (F) = 0xF302
2 = 0x2503
HALT

DATA	ADDR
0100000011011010	0000000000000000
0010000100000000	0000000000000001
0001001000000001	0000000000000010
1011001000000000	0000000000000011
0011000100000001	0000000000000100
1100111010110010	0000000000000101
0001001100000000	0000000000000110
1111001100000010	0000000000000111
0010010100000011	0000000000001000
0000000000000000	0000000000001001

Results	
R0	0x00DA
R1	0x0000
R2	0x2027
R3	0x00DA
R4	0x0000
R5	0x3F74
R6	0x0000
R7	0x0000
R8	0x0000
R9	0x0000
RA	0x0000
RB	0x0000
RC	0x0000
RD	0x0000
RE	0x0000
RF	0x0000

1.4 ADD REG, PTR

initial:

0001111101001101

DATA	ADDR
0x1F4D	0x00DA

0000000011011010

	Code	
MOV REG, IMM	MOV R0, 0xDA	0x?0DA
MOV REG, ADDR	MOV R1, [R0]	0x?100
MOV REG, REG	MOV R2, R1	0x?201
ADD REG, REG	ADD R2, R0	0x?200
XOR REG, REG	XOR R1, R1	0x?101
MOV ADDR, REG	MOV [0xEB], R2	0x?EB2
MOV REG, REG	MOV R3, R0	0x?300
MOV REG, REG	MOV R5, R2	0x?502
ADD REG, PTR	ADD R2, [R3]	0x?203

4 = 0x40DA
2 = 0x2100
1 = 0x1201
11 (B) = 0xB200
3 = 0x3101
12 (C) = 0xCEB2
1 = 0x1300
1 = 0x1502
15 (F) = 0xF203
HALT

DATA	ADDR
0100000011011010	0000000000000000
0010000100000000	0000000000000001
0001001000000001	0000000000000010
1011001000000000	0000000000000011
0011000100000001	0000000000000100
1100111010110010	0000000000000101
0001001100000000	0000000000000110
0001010100000010	0000000000000111
1111001000000011	0000000000001000
0000000000000000	0000000000001001

Results	
R0	0x00DA
R1	0x0000
R2	0x3F74
R3	0x00DA
R4	0x0000
R5	0x2027
R6	0x0000
R7	0x0000
R8	0x0000
R9	0x0000
RA	0x0000
RB	0x0000
RC	0x0000
RD	0x0000
RE	0x0000
RF	0x0000