

IMPORTANT POINTS:

1. There are two functions – simulate () and display () which needs to be implemented as a part of project.
2. There should be second command line argument (simulate/display) to distinguish these two functions:
 - a. Second command line argument is “simulate” ~~Only~~ show State of Register File and Data Memory.
 - b. Second command line argument is “display” ~~Show~~ Instruction Flow as well as State of Register File ~~and~~ Data Memory.
3. There should be third command line argument as “number of cycles” means up to this number of cycles simulation should run and produce output.

Example with three command line arguments while running the program:

- make
- ./apex_sim input.asm simulate 50
 - Simulate for 50 cycles and then show State of Register File and Data Memory at the end of 50 cycles or at the end of program (whichever comes first).
- make
- ./apex_sim input.asm display 10
 - Simulate for 10 cycles and then show Instruction Flow as well as State of Register File and Data Memory at the end of 10 cycles or at the end of program (whichever comes first).

SAMPLE TEST CASE:

```
MOVC R0,#4000
MOVC R1,#1
MOVC R2,#2
MOVC R3,#3
MOVC R4,#1
ADD R5,R0,R1
SUB R3,R3,R4
CMP R3,R2
BZ #-12
MUL R7,R5,R2
MOVC R8,#0
AND R9,R7,R8
HALT
MOVC R10,#500
MOVC R11,#10
```

DISPLAY GUIDELINES: Below are output for first 5 clock cycles –

Note: The order of displaying the stages can be shown as given in the code template.

CLOCK CYCLE 1

1. Instruction at FETCH_____STAGE ---> (I0: 4000) MOVC R0,#4000
2. Instruction at DECODE_RF_STAGE ---> EMPTY
3. Instruction at EX_____STAGE ---> EMPTY
4. Instruction at MEMORY_____STAGE ---> EMPTY
5. Instruction at WRITEBACK_STAGE ---> EMPTY

CLOCK CYCLE 2

1. Instruction at FETCH_____STAGE ---> (I1: 4004) MOVC R1,#1
2. Instruction at DECODE_RF_STAGE ---> (I0: 4000) MOVC R0,#4000
3. Instruction at EX_____STAGE ---> EMPTY
4. Instruction at MEMORY_____STAGE ---> EMPTY
5. Instruction at WRITEBACK_STAGE ---> EMPTY

CLOCK CYCLE 3

- | | |
|--|--------------------------|
| 1. Instruction at FETCH_____STAGE --> | (I2: 4008) MOVC R2,#2 |
| 2. Instruction at DECODE_RF_STAGE --> | (I1: 4004) MOVC R1,#1 |
| 3. Instruction at EX_____STAGE --> | (I0: 4000) MOVC R0,#4000 |
| 4. Instruction at MEMORY_____STAGE --> | EMPTY |
| 5. Instruction at WRITEBACK_STAGE --> | EMPTY |

CLOCK CYCLE 4

- | | |
|--|--------------------------|
| 1. Instruction at FETCH_____STAGE --> | (I3: 4012) MOVC R3,#3 |
| 2. Instruction at DECODE_RF_STAGE --> | (I2: 4008) MOVC R2,#2 |
| 3. Instruction at EX_____STAGE --> | (I1: 4004) MOVC R1,#1 |
| 4. Instruction at MEMORY_____STAGE --> | (I0: 4000) MOVC R0,#4000 |
| 5. Instruction at WRITEBACK_STAGE --> | EMPTY |

CLOCK CYCLE 5

- | | |
|--|--------------------------|
| 1. Instruction at FETCH_____STAGE --> | (I4: 4016) MOVC R4,#1 |
| 2. Instruction at DECODE_RF_STAGE --> | (I3: 4012) MOVC R3,#3 |
| 3. Instruction at EX_____STAGE --> | (I2: 4008) MOVC R2,#2 |
| 4. Instruction at MEMORY_____STAGE --> | (I1: 4004) MOVC R1,#1 |
| 5. Instruction at WRITEBACK_STAGE --> | (I0: 4000) MOVC R0,#4000 |

===== STATE OF ARCHITECTURAL REGISTER FILE =====

REG[00]	Value = 4000	Status = VALID
REG[01]	Value = 1	Status = VALID
REG[02]	Value = 2	Status = VALID
REG[03]	Value = 2	Status = VALID
REG[04]	Value = 1	Status = VALID
REG[05]	Value = 4001	Status = VALID
REG[06]	Value = 00	Status = VALID
REG[07]	Value = 8002	Status = VALID
REG[08]	Value = 00	Status = VALID
REG[09]	Value = 00	Status = VALID
REG[10]	Value = 00	Status = VALID
REG[11]	Value = 00	Status = VALID
REG[12]	Value = 00	Status = VALID
REG[13]	Value = 00	Status = VALID
REG[14]	Value = 00	Status = VALID
REG[15]	Value = 00	Status = VALID

===== STATE OF DATA MEMORY =====

MEM[00]	Data Value = 00
MEM[01]	Data Value = 00
MEM[02]	Data Value = 00
.	
.	
.	
MEM[99]	Data Value = 00

Solution Without Forwarding:

4000	I0	MOVC R0,#4000
4004	I1	MOVC R1,#1
4008	I2	MOVC R2,#2
4012	I3	MOVC R3,#3
4016	I4	MOVC R4,#1
4020	I5	ADD R5,R0,R1
4024	I6	SUB R3,R3,R4
4028	I7	CMP R3,R2
4032	I8	BZ #=12
4036	I9	MUL R7,R5,R2
4040	I10	MOVC R8,#0
4044	I11	AND R9,R7,R8
4048	I12	HALT
4052	I13	MOVC R10,#500
4056	I14	MOVC R11,#10

ST/CY	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	
F	10	I1	I2	I3	I4	I5	I6	I7	I8	I8	I8	I9	I10	I5	I6	I7	I8	I8	I8	I9	I10	I11	I12	I12	I12	I13					
D/RF	I0	I1	I2	I3	I4	I5	I6	I6	I7	I7	I7	I8	I9		I5	I6	I7	I7	I7	I8	I8	I9	I10	I11	I11	I11	I12				
EX		I0	I1	I2	I3	I4	I5		I6			I7	I8			I5	I6			I7	I8	I8	I9	I10			I11	I12			
MEM		I0	I1	I2	I3	I4	I5		I6			I7	I8			I5	I6			I7	I8	I9	I10				I11	I12			
WB			I0	I1	I2	I3	I4	I5		I6			I7	I8			I5	I6			I7	I8	I9	I10				I11	I12		

Stalling

Flushed

Branch taken

Branch not taken

Solution With Forwarding:

4000	I0	MOVC R0,#4000
4004	I1	MOVC R1,#1
4008	I2	MOVC R2,#2
4012	I3	MOVC R3,#3
4016	I4	MOVC R4,#1
4020	I5	ADD R5,R0,R1
4024	I6	SUB R3,R3,R4
4028	I7	CMP R3,R2
4032	I8	BZ #=12
4036	I9	MUL R7,R5,R2
4040	I10	MOVC R8,#0
4044	I11	AND R9,R7,R8
4048	I12	HALT
4052	I13	MOVC R10,#500
4056	I14	MOVC R11,#10

ST/CY	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	
F	I0	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10	I5	I6	I7	I8	I9	I10	I11	I12	I13				
D/RF	I0	I1	I2	I3	I4	I5	I6	I7	I8	I9		I5	I6	I7	I8	I9	I10	I11	I11	I12				
EX		I0	I1	I2	I3	I4	I5	I6	I7	I8			I5	I6	I7	I8	I8	I9	I10	I11	I12			
MEM			I0	I1	I2	I3	I4	I5	I6	I7	I8			I5	I6	I7	I8	I9	I10	I11	I12			
WB				I0	I1	I2	I3	I4	I5	I6	I7	I8			I5	I6	I7	I8	I9	I10	I11	I12		

Stalling

Flushed

Branch taken

Branch not taken