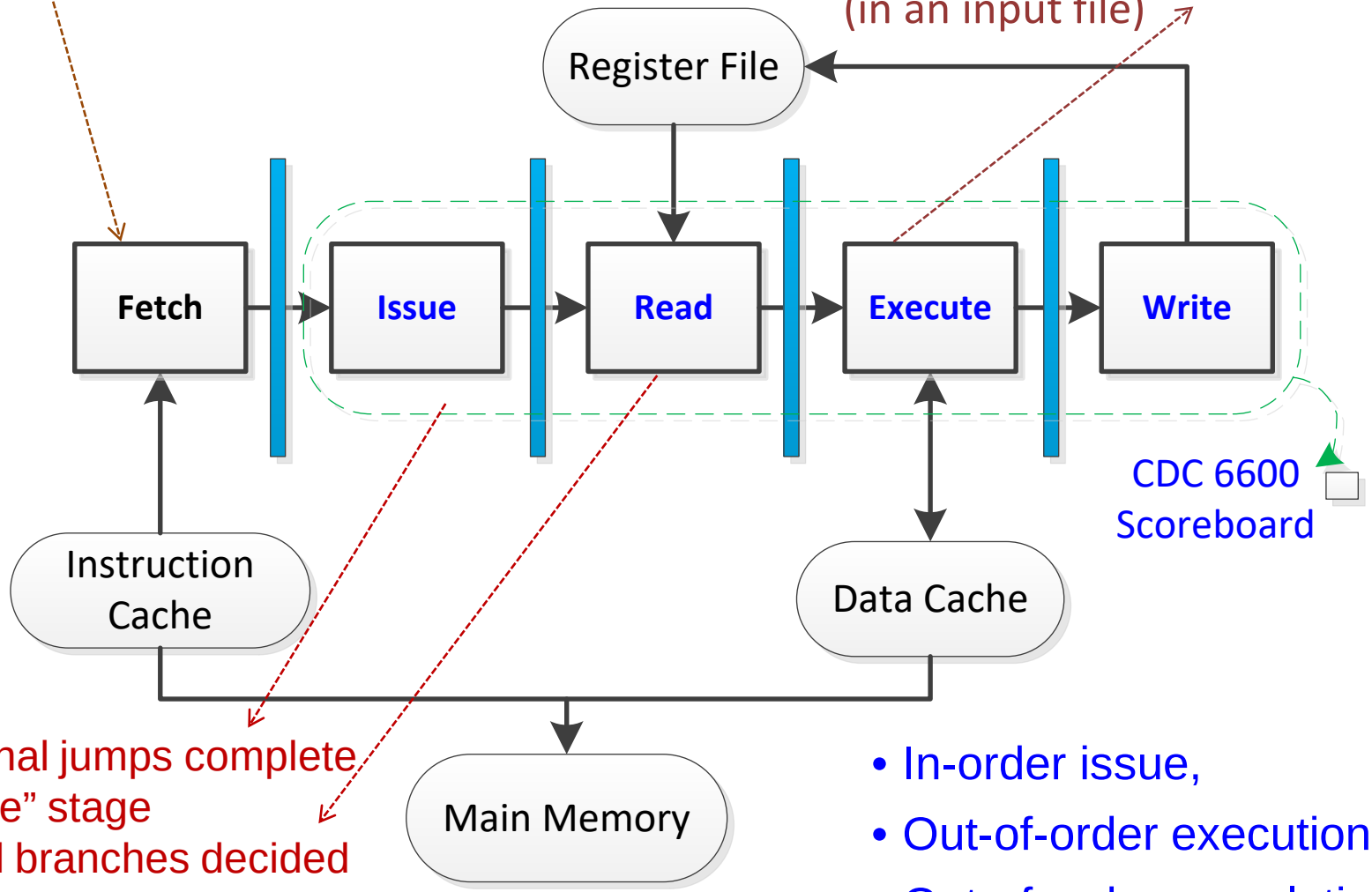
When a fetched instruction cannot be issued, the next instruction cannot be fetched.

Multiple functional units whose numbers are provided as configuration parameters (in an input file)



- Unconditional jumps complete in the "Issue" stage
- Conditional branches decided in the "Operand Read" stage.
- "not-taken prediction" will be used in "Fetch" stage.

- In-order issue,
- Out-of-order execution
- Out-of-order completion
- No data bypassing

Instruction Class	Instruction Mnemonic
Data Transfers	LW, SW, L.D, S.D
Arithmetic/ logical	DADD, DADDI, DSUB, DSUBI, AND, ANDI, OR, ORI, LI, LUI ADD.D, MUL.D, DIV.D, SUB.D
Control	J, BEQ, BNE
Special purpose	HLT (to stop fetching new instructions)

The table below shows the number of cycles each instruction takes in the EX stage.

Instructions	Number of Cycles in "Execute" Stage
HLT, J	0 Cycles (finish in issue stage)
BEQ, BNE	0 Cycle (finish in Read stage)
DADD, DADDI, DSUB, DSUBI, AND, ANDI, OR, ORI, LI, LUI	1 Cycle
LW, SW,	1 Cycle + D-Cache miss penalty if applicable
L.D, S.D	2 Cycle + D-Cache miss penalty if applicable
ADD.D, SUB.D	Specified in the "*config.txt*" file
MUL.D	Specified in the "*config.txt*" file
DIV.D	Specified in the "*config.txt*" file

Input/Output Files

```
linux2[1]% ./simulator
Usage: simulator inst.txt data.txt config.txt result.txt
```

Program:

- set of MIPS instruction
- Two HLT instruction marks end of program
- Labels are used for branching

Initial values for data memory.
Registers are to be initialized using LI and LUI

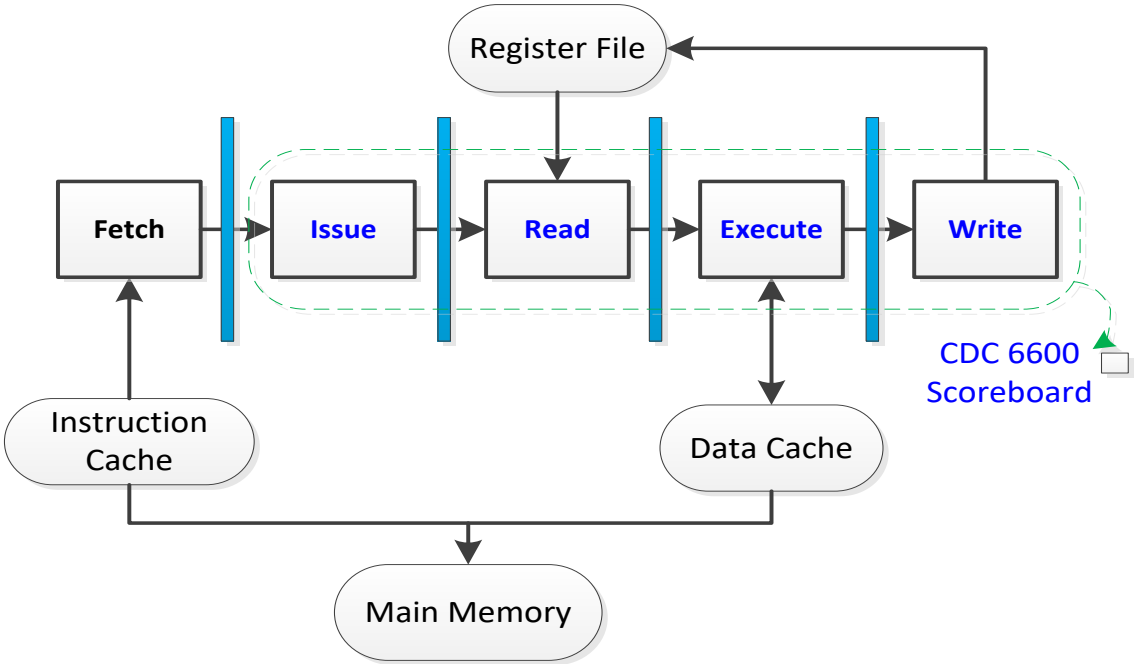
The "config.txt" file should include:

FP adder: *< number of units >, < cycle count >*

FP Multiplier: *< number of units >, < cycle count >*

FP divider: *< number of units >, < cycle count >*

I-Cache: *< number of blocks >, < block size in words >*



`result.txt`

Trace the execution by listing every instr. and when it passed through the various stages and what hazards it suffered, as well as the instr. and data cache performance

Important Notes

- In addition to the 32 word-size registers (for integers), there are 32 FP registers; each has 64 bits.
- Floating point calculations will have no impact on the required output of your simulator. In fact, only the contents of the integer registers need to be read from the input file, and you do not even need to allocate storage in your simulator for floating point registers.
- # cycles required by the ALU depends on the latency of the involved functional unit.
- Data transfer instructions (L.D, S.D, LW and SW) do not use the integer unit.
  ➔ there is no structural hazard will be experienced if an integer instruction is being executed and a load/store instruction is to be issued.
- Instructions and data are stored in memory starting at address 0x0 and 0x100 respectively. Load and store instructions use word addresses when accessing data.
- Both conditional and unconditional jump instructions can be forward and backward. You can assume that a program will not create a closed loop.
- HLT instruction marks the end of the program, i.e., fetching will seize as soon as the HLT instruction is decoded. Assume that the program will have two HLT instructions at the end in order to stop accessing the cache once the first HLT reaches the issue stage. You can ignore the second HLT instruction.

Important Notes

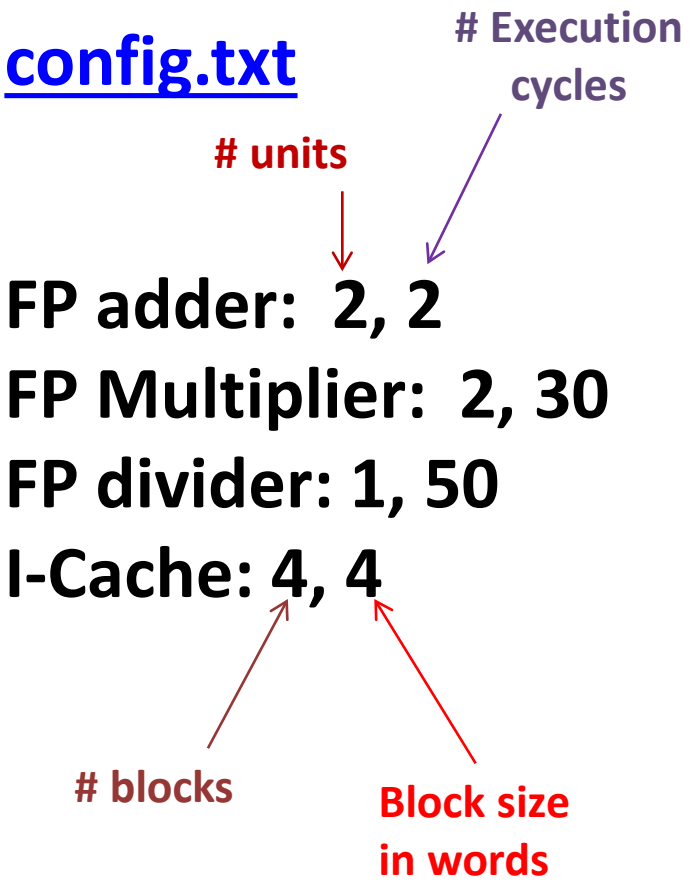
- Unconditional jumps complete in the “Issue” stage. The fetched instruction (the next instruction in the program) will be flushed from the “Fetch” stage in that case. In other words a “J” instruction will waste one cycle.
- Conditional branches are resolved in the “Read” stage.
  - CPU will go ahead and fetch the next instruction, i.e., “not-taken prediction”
- When a “BEQ” or “BNE” instruction is issued, the pipeline will be stalled until the condition is resolved
  - if the branch is “non-taken” one cycle will be wasted.
  - if we find out in the “Read” stage that the branch is taken, the control unit will flush the “Fetch” stage and update the program counter so the CPU in next cycle will fetch from the branch target address, i.e., , if the branch is “Taken”, 2 cycles will be wasted.
- Please note that if a “Fetch” causes an I-Cache miss, the “Fetch” is to be completed before it is flushed when the branch is “Taken”. The branching instructions do not stall because of structural hazard related to the integer unit.

Example

inst.txt

```
        LI  R4,   260
        LI  R5,   272
        LI  R1,   8
        LI  R2,   4
        LI  R3,   0
GG:     L.D  F1,  4(R4)
        L.D  F2,  8(R5)
        ADD.D   F4,   F6,  F2
        SUB.D   F5,   F7,  F1
        MUL.D   F6,   F1, F5
        ADD.D   F7,  F2,  F6
        ADD.D   F6,  F1,  F7
        DADDI   R4,   R4,  20
        DADDI   R5,   R5,  8
        DSUB    R1,   R1,  R2
        BNE     R1,   R3,  GG
        HLT
        HLT
```

config.txt



Example: Without Memory Hierarchy

(1st iteration)

Instruction	Fetch	Issue	Read	Exec	Write	RAW	WAW	Struct
LI R4, 260	1	2	3	4	5	N	N	N
LI R5, 272	2	6	7	8	9	N	N	Y
LI R1, 8	6	10	11	12	13	N	N	Y
LI R2, 4	10	14	15	16	17	N	N	Y
LI R3, 0	14	18	19	20	21	N	N	Y
GG: L.D F1, 4(R4)	18	19	20	22	23	N	N	N
L.D F2, 8(R5)	19	24	25	27	28	N	N	Y
ADD.D F4, F6, F2	24	25	29	31	32	Y	N	N
SUB.D F5, F7, F1	25	26	27	29	30	N	N	N
MUL.D F6, F1, F5	26	27	31	61	62	Y	N	N
ADD.D F7, F2, F6	27	31	63	65	66	Y	N	Y
ADD.D F6, F1, F7	31	63	67	69	70	Y	Y	Y
DADDI R4, R4, 20	63	64	65	66	67	N	N	N

Example: Without Memory Hierarchy

(1st iteration)

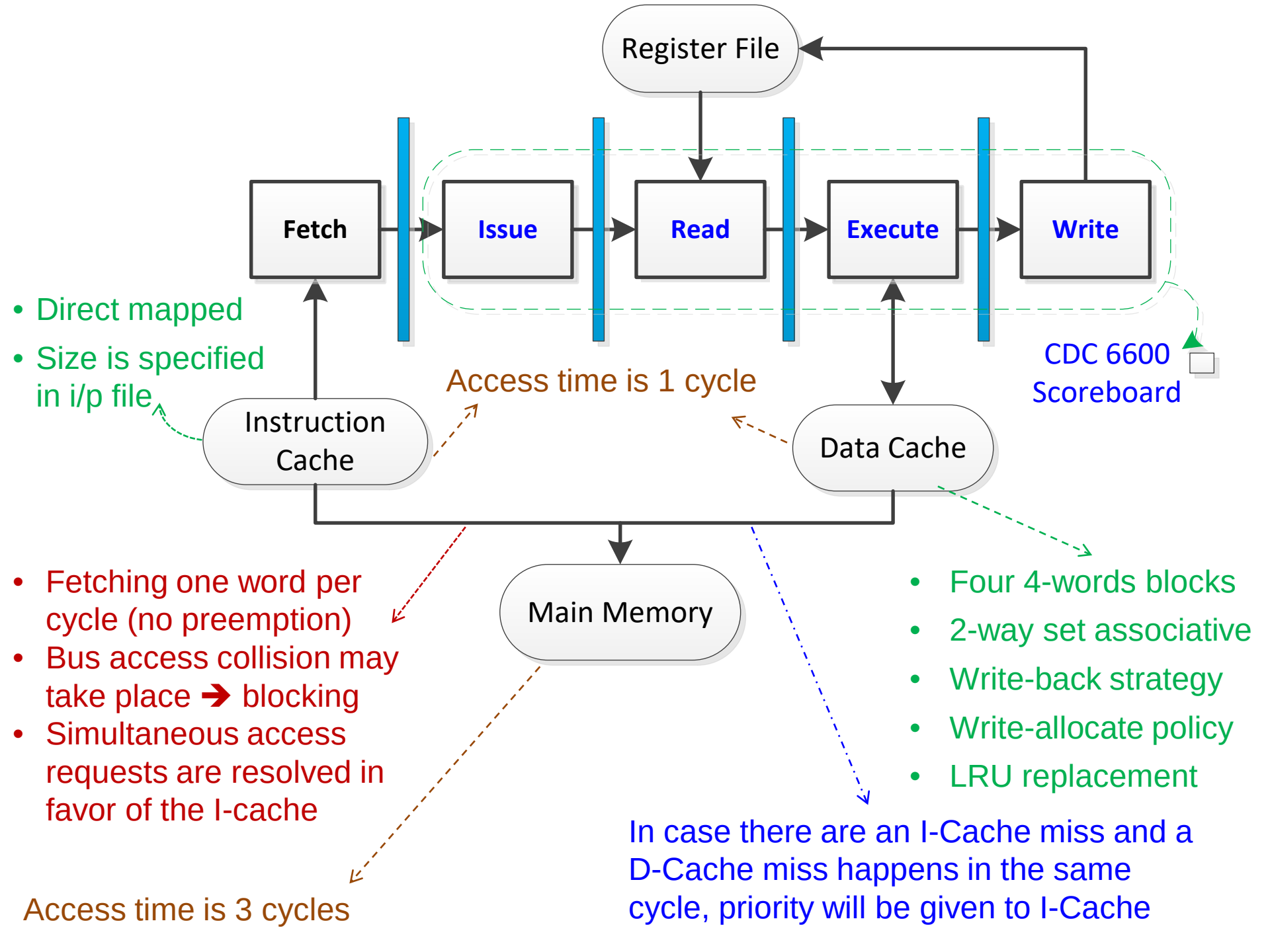
Instruction	Fetch	Issue	Read	Exec	Write	RAW	WAW	Struct
GG: L.D F1, 4(R4)	18	19	20	22	23	N	N	N
L.D F2, 8(R5)	19	24	25	27	28	N	N	Y
ADD.D F4, F6, F2	24	25	29	31	32	Y	N	N
SUB.D F5, F7, F1	25	26	27	29	30	N	N	N
MUL.D F6, F1, F5	26	27	31	61	62	N	N	N
ADD.D F7, F2, F6	27	31	63	65	66	Y	N	Y
ADD.D F6, F1, F7	31	63	67	69	70	Y	Y	Y
DADDI R4, R4, 20	63	64	65	66	67	N	N	N
DADDI R5, R5, 8	64	68	69	70	71	N	N	Y
DSUB R1, R1, R2	68	72	73	74	75	N	N	Y
BNE R1, R3, GG	72	73	76			Y	N	N
HLT	73					N	N	N

"BNE" is issued ➔ stall pipeline until the condition is resolved.

Example: With Memory Hierarchy

(2nd iteration)

Instruction	Fetch	Issue	Read	Exec	Write	RAW	WAW	Struct
GG: LD F1, 4(R4)	77	78	79	81	82	N	N	N
LD F2, 8(R5)	78	83	84	86	87	N	N	Y
ADD.D F4, F6, F2	83	84	88	90	91	Y	N	N
SUB.D F5, F7, F1	84	85	86	88	89	N	N	N
MUL.D F6, F1, F5	85	86	90	120	121	Y	N	N
ADD.D F7, F2, F6	86	90	122	124	125	Y	N	Y
ADD.D F6, F1, F7	90	122	126	128	129	Y	Y	Y
DADDI R4, R4, 20	122	123	124	125	126	N	N	N
DADDI R5, R5, 8	123	127	128	129	130	N	N	Y
DSUB R1, R1, R2	127	131	132	133	134	N	N	Y
BNE R1, R3, GG	131	132	135			Y	N	N
HLT	132	136				N	N	N
HLT	136					N	N	N


Register File
Fetch
Issue
Read
Execute
Write
CDC 6600 Scoreboard
Direct mapped
Size is specified in i/p file
Instruction Cache
Access time is 1 cycle
Data Cache
Main Memory
Fetching one word per cycle (no preemption)
Bus access collision may take place ➔ blocking
Simultaneous access requests are resolved in favor of the I-cache
Access time is 3 cycles
Four 4-words blocks
2-way set associative
Write-back strategy
Write-allocate policy
LRU replacement
In case there are an I-Cache miss and a D-Cache miss happens in the same cycle, priority will be given to I-Cache

Example: With Memory Hierarchy

(1st iteration)

Instruction	Fetch	Issue	Read	Exec	Write	RAW	WAW	Struct
LI R4, 260	13	14	15	16	17	N	N	N
LI R5, 272	14	18	19	20	21	N	N	Y
LI R1, 8	18	22	23	24	25	N	N	Y
LI R2, 4	22	26	27	28	29	N	N	Y
LI R3, 0	35	36	37	38	39	N	N	N
GG: L.D F1, 4(R4)	36	37	38	52	53	N	N	N
L.D F2, 8(R5)	37	54	55	80	81	N	N	Y
ADD.D F4, F6, F2	54	55	82	84	85	Y	N	N
SUB.D F5, F7, F1	67	68	69	71	72	N	N	N
MUL.D F6, F1, F5	68	69	73	103	104	Y	N	N
ADD.D F7, F2, F6	69	73	105	107	108	Y	N	Y
ADD.D F6, F1, F7	73	105	109	111	112	Y	Y	Y
DADDI R4, R4, 20	105	106	107	108	109	N	N	N

Example: With Memory Hierarchy

(1st iteration)

Instruction	Fetch	Issue	Read	Exec	Write	RAW	WAW	Struct
GG: L.D F1, 4(R4)	36	37	38	52	53	N	N	N
L.D F2, 8(R5)	37	54	55	80	81	N	N	Y
ADD.D F4, F6, F2	54	55	82	84	85	Y	N	N
SUB.D F5, F7, F1	67	68	69	71	72	N	N	N
MUL.D F6, F1, F5	68	69	73	103	104	Y	N	N
ADD.D F7, F2, F6	69	73	105	107	108	Y	N	Y
ADD.D F6, F1, F7	73	105	109	111	112	Y	Y	Y
DADDI R4, R4, 20	105	106	107	108	109	N	N	N
DADDI R5, R5, 8	106	110	111	112	113	N	N	Y
DSUB R1, R1, R2	110	114	115	116	117	N	N	Y
BNE R1, R3, GG	114	115	118			Y	N	N
HLT	127					N	N	N

Example: With Memory Hierarchy

(2nd iteration)

Instruction	Fetch	Issue	Read	Exec	Write	RAW	WAW	Struct
GG: LD F1, 4(R4)	128	129	130	144	145	N	N	N
LD F2, 8(R5)	129	146	147	149	150	N	N	Y
ADD.D F4, F6, F2	146	147	151	153	154	Y	N	N
SUB.D F5, F7, F1	147	148	149	151	152	N	N	N
MUL.D F6, F1, F5	148	149	153	183	184	Y	N	N
ADD.D F7, F2, F6	149	153	185	187	188	Y	N	Y
ADD.D F6, F1, F7	153	185	189	191	192	Y	Y	Y
DADDI R4, R4, 20	185	186	187	188	189	N	N	N
DADDI R5, R5, 8	186	190	191	192	193	N	N	Y
DSUB R1, R1, R2	190	194	195	196	197	N	N	Y
BNE R1, R3, GG	194	195	198			Y	N	N
HLT	195	199	"BNE" issued ➔ stall pipeline until the condition is resolved			N	N	N
HLT	199					N	N	N

Output

clock cycle that instruction leaves each stage

Hazards that caused stalls

Instruction	Fetch	Issue	Read	Exec	Write	RAW	WAW	Struct
:								
GG: LD F1, 4(R4)	36	37	38	52	53	N	N	N
LD F2, 8(R5)	37	54	55	80	81	N	N	Y
:								
DSUB R1, R1, R2	190	194	195	196	197	N	N	Y
BNE R1, R3, GG	194	195	198			Y	N	N
HLT	195	199				N	N	N
HLT	199					N	N	N

Cycle number of last stage (memory for load/store)

A branching instruction terminates in "Read" stage and does not have entries in the Exec and Write stages.

Total number of access requests for instruction cache: 30

Number of instruction cache hits: 25

Total number of access requests for data cache: 8

Number of data cache hits: 5