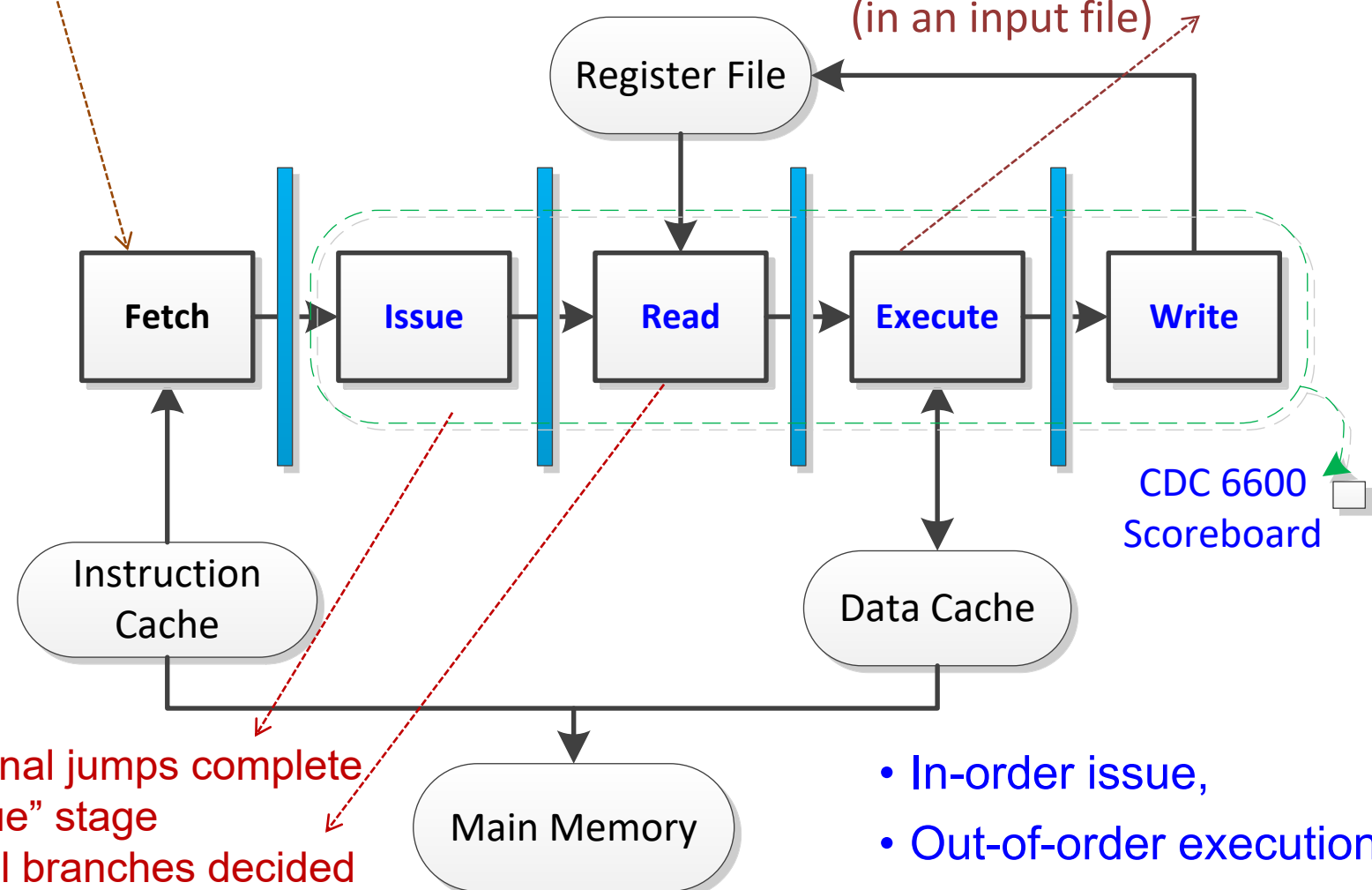


When a fetched instruction cannot be issued, the next instruction cannot be fetched.

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 “ <i>config.txt</i> ” file
MUL.D	Specified in the “ <i>config.txt</i> ” file
DIV.D	Specified in the “ <i>config.txt</i> ” 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

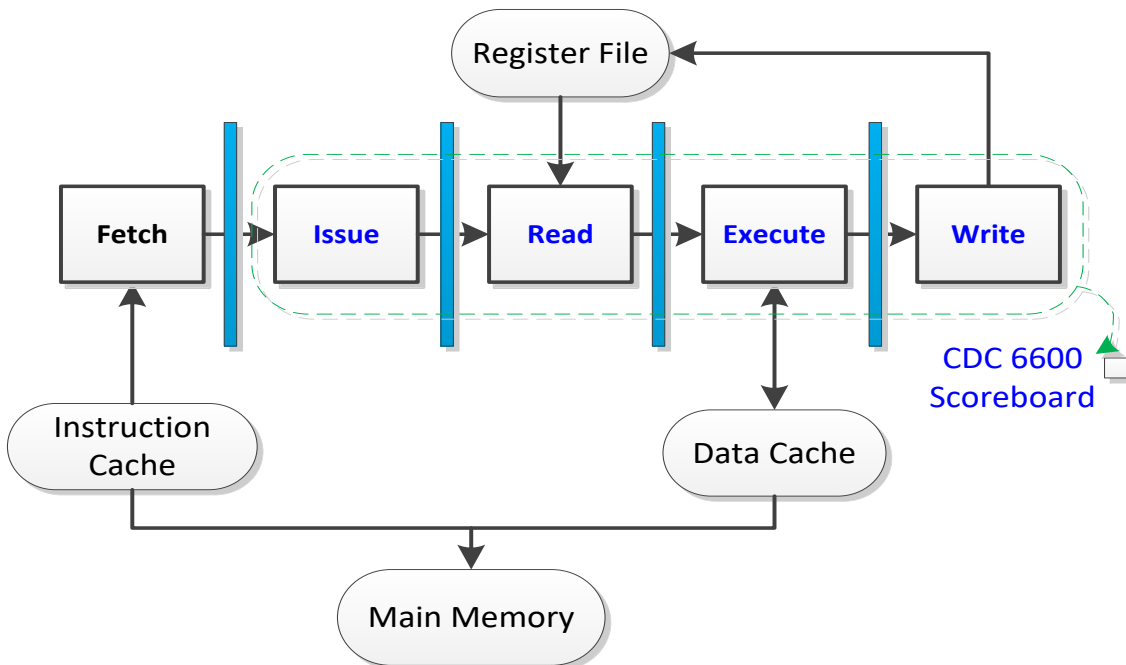
The “**config.txt**” file should include:

*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 cease 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, 266
      LI R5, 274
      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, 2
      DADDI R5, R5, 2
      DSUB  R1, R1, R2
      BNE   R1, R3, GG
      HLT
      HLT
```

## config.txt

# Execution cycles

# units

FP adder: 2, 2

FP Multiplier: 2, 30

FP divider: 1, 50

I-Cache: 4, 4

# blocks

Block size in words

# Example: Without Memory Hierarchy

## (1<sup>st</sup> iteration)

<u>Instruction</u>	<u>Fetch</u>	<u>Issue</u>	<u>Read</u>	<u>Exec</u>	<u>Write</u>	<u>RAW</u>	<u>WAW</u>	<u>Struct</u>
LI R4, 266	1	2	3	4	5	N	N	N
LI R5, 274	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, 2	63	64	65	66	67	N	N	N

# Example: Without Memory Hierarchy

## (1<sup>st</sup> iteration)

<u>Instruction</u>	<u>Fetch</u>	<u>Issue</u>	<u>Read</u>	<u>Exec</u>	<u>Write</u>	<u>RAW</u>	<u>WAW</u>	<u>Struct</u>
GG: L.D F1, 4(R4)	19	20	21	23	24	N	N	N
L.D F2, 8(R5)	20	25	26	28	29	N	N	Y
ADD.D F4, F6, F2	25	26	30	32	33	Y	N	N
SUB.D F5, F7, F1	26	27	28	30	31	N	N	N
MUL.D F6, F1, F5	27	28	32	62	63	N	N	N
ADD.D F7, F2, F6	28	32	64	66	67	Y	N	Y
ADD.D F6, F1, F7	32	64	68	70	71	Y	Y	Y
DADDI R4, R4, 2	64	65	66	67	68	N	N	N
DADDI R5, R5, 2	65	69	70	71	72	N	N	Y
DSUB R1, R1, R2	69	73	74	75	76	N	N	Y
BNE R1, R3, GG	73	74	77			Y	N	N
HLT	74	“BNE” is issued → stall pipeline until the condition is resolved.				N	N	N



# Example: With Memory Hierarchy

## (2<sup>nd</sup> iteration)

<u>Instruction</u>	<u>Fetch</u>	<u>Issue</u>	<u>Read</u>	<u>Exec</u>	<u>Write</u>	<u>RAW</u>	<u>WAW</u>	<u>Struct</u>
GG: LD F1, 4(R4)	78	79	80	82	83	N	N	N
LD F2, 8(R5)	79	84	85	87	88	N	N	Y
ADD.D F4, F6, F2	84	85	89	91	92	Y	N	N
SUB.D F5, F7, F1	85	86	87	89	90	N	N	N
MUL.D F6, F1, F5	86	87	91	121	122	Y	N	N
ADD.D F7, F2, F6	87	91	123	125	126	Y	N	Y
ADD.D F6, F1, F7	91	123	127	129	130	Y	Y	Y
DADDI R4, R4, 2	123	124	125	126	127	N	N	N
DADDI R5, R5, 2	124	128	129	130	131	N	N	Y
DSUB R1, R1, R2	128	132	133	134	135	N	N	Y
BNE R1, R3, GG	132	133	136			Y	N	N
HLT	133	137				N	N	N
HLT	137					N	N	N