

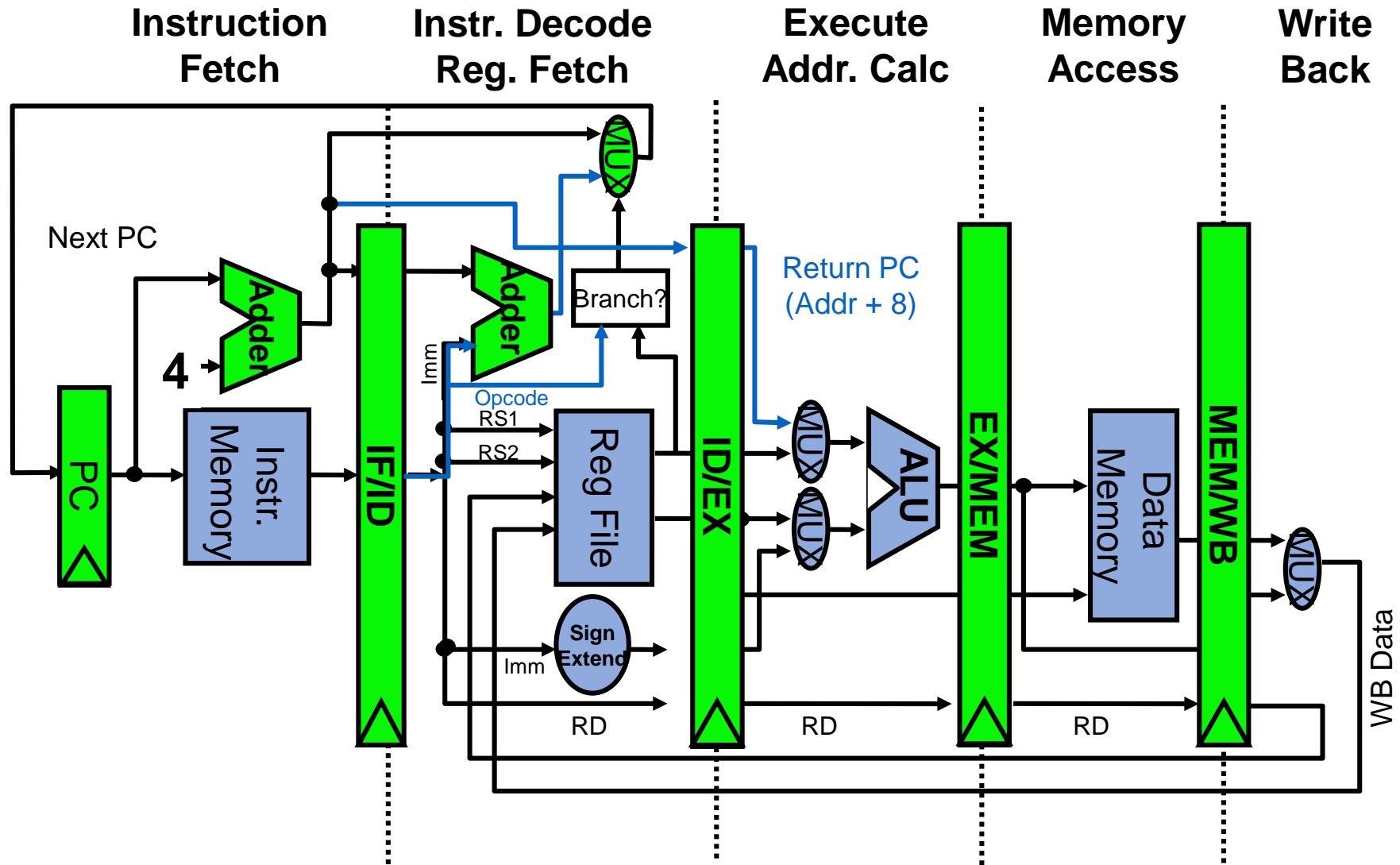
CS425

Computer Systems Architecture

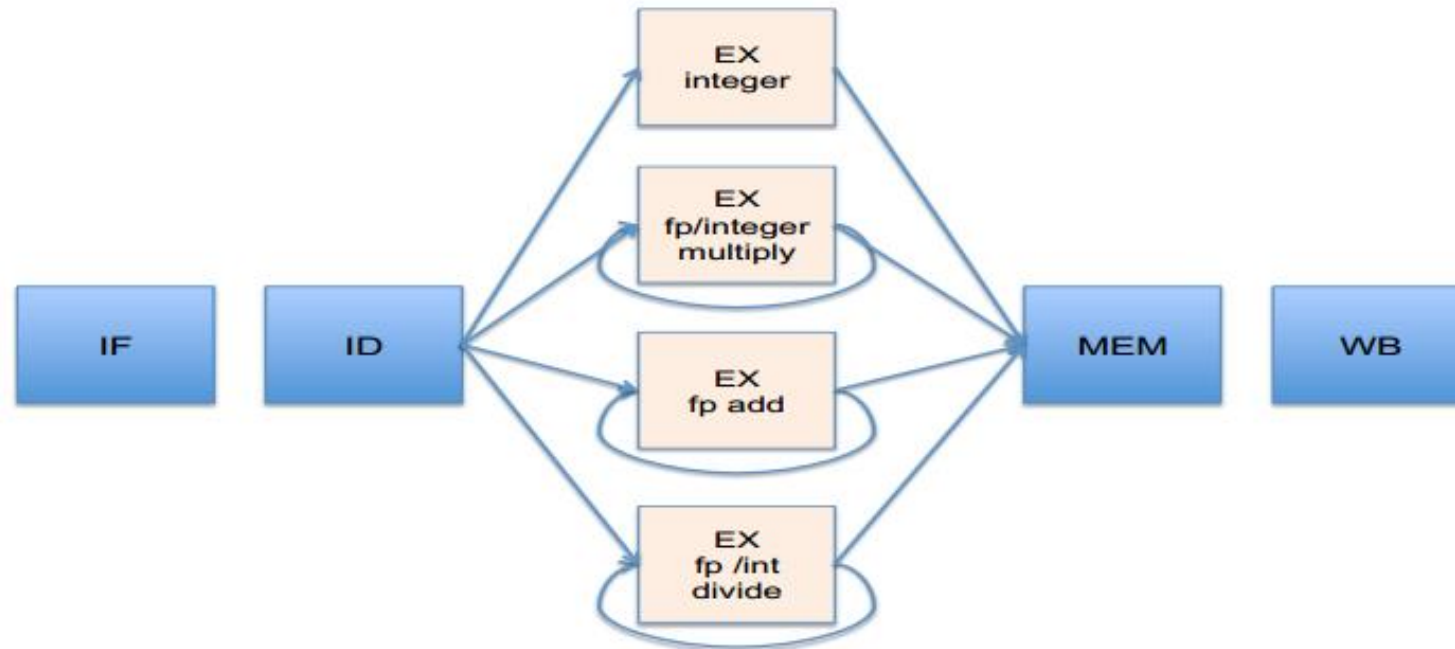
Fall 2019

**Dynamic Instruction Scheduling:
Scoreboard**

DLX Processor

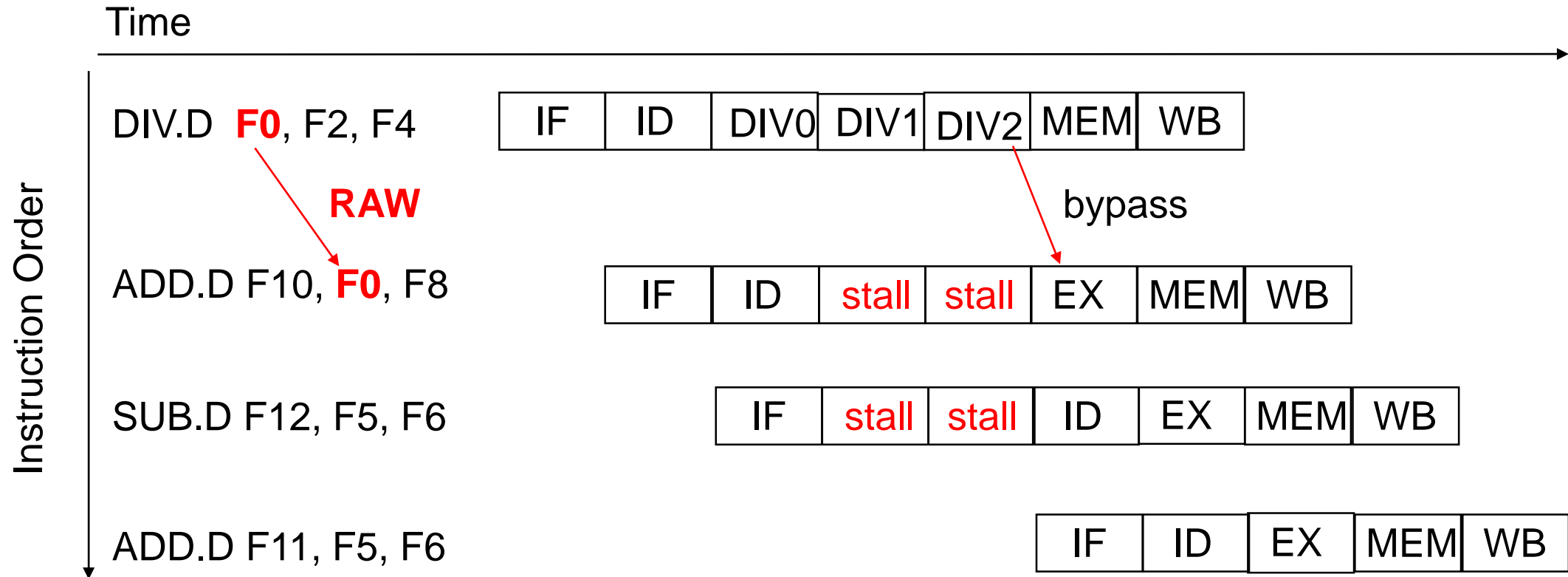


Pipelines with variable instruction latencies



- FP instructions need to have multiple execution cycles.
- DLX pipeline does not work efficiently.

In-order Execution

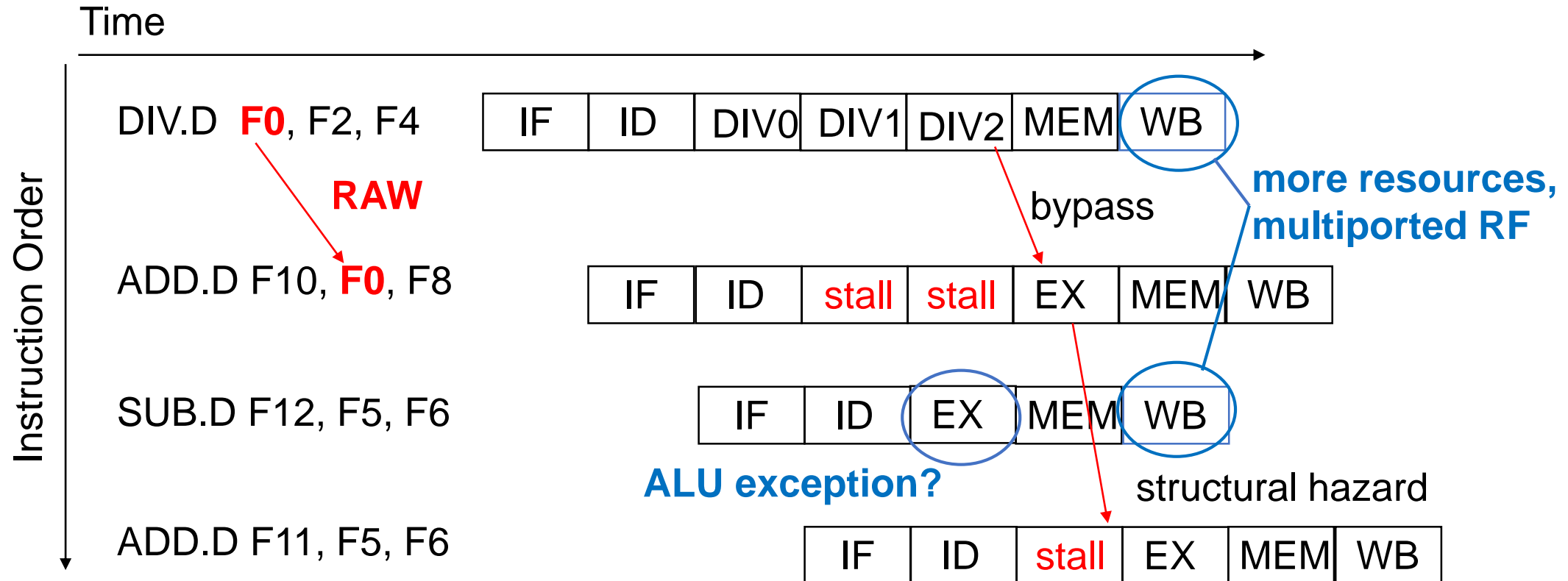


- If DIV takes 10 clock cycles (DIV0, DIV1, ..., DIV9) we need 9 stall cycles

In-order vs Out-of-order (OoO – O3)

- Execution of a stage **in order**
 - The instructions enter the stage in program order
- Execution of a stage **out of order**
 - No constraint regarding the order the instructions enter the stage
- **Example 1:**
 - in order fetch
 - in order decode
 - out of order execution
 - out of order write back (commit)
- **Example 2:**
 - in order fetch
 - in order decode
 - out of order execution
 - in order write back (commit)

Out-of-order Execution



- Instructions execute when source operands (registers) values are available and not structural hazards exist
- Even if DIV takes 10 clock cycles the pipeline stalls only for 1 clock cycle

Instruction Level Parallelism

- **Instruction Level Parallelism (ILP)**

- Potential overlap among instructions
- Parallel or out-of-order execution
- Requires extensions on the simple pipeline

- **Loop Level Parallelism**

- Exploit ILP between instructions from different basic blocks (e.g iterations of a loop)

```
for (i=1; i<=100; i++) x[i] = x[i] + y[i];
```

- **Must maintain:**

- Data flow: the real flow of values between instructions that write (produce) values and the instructions that read (consume) these values (RAW, WAW, WAR)
- Exception behavior: changes in execution order should not change the order of exceptions (and should not generate new exceptions)

Data Flow and Exception Behavior

- **Program Order:** The result of instruction execution should be the same as the sequential execution of the instructions
 1. Preserve control dependencies
 2. Preserve data flow: preserve data dependencies
 3. Exception behavior: changing the order of instruction execution should not create new exceptions

DADDU R2 , R3 , R4

BEQZ R2 , L1 ↑

LW R1 , 0 (R2)

execute before branch?

L1 :

Techniques to reduce stalls

- $CPI = \text{Ideal CPI} + \text{Structural stalls per instruction} + \text{RAW stalls per instruction} + \text{WAR stalls per instruction} + \text{WAW stalls per instruction}$
- 1st example (in-order) had 2 RAW stalls
- 2nd example (out-of-order) has only 1 structural stall
- We will study two types of techniques:

| Dynamic instruction scheduling | Static instruction scheduling (SW/compiler) |
|---|---|
| Scoreboard (reduce RAW stalls) | Loop Unrolling |
| Register Renaming (reduce WAR & WAW stalls) <ul style="list-style-type: none">• Tomasulo• Reorder buffer | SW pipelining |
| Branch Prediction (reduce control stalls) | Trace Scheduling |

Can we use hardware techniques to achieve CPI close to 1?

- Why in hardware during program executions ?
 - It works even if we don't know the real dependencies at compile-time (e.g. memory references).
 - The compiler's job is simpler, otherwise it needs to know all HW details
 - Control dependences
 - The assembly code for a machine can execute well on another machine too.
- **Key idea:** Allow instructions that appear after a stalled instruction to execute (stall only the data dependent instruction)

DIV.D F0,F2,F4

ADD.D F10,F0,F8

SUB.D F12,F8,F14

- **Out-of-order** execution => **out-of-order** completion (unless special HW is used)

Dependencies between Instructions

- What are the sources of stalls/bubbles;
 - instructions that use the same registers
- **Parallel** instructions can execute without imposing any stalls (if we ignore structural hazards)
 - DIV.D F0, F2, F4
 - ADD.D F10, F1, F3
- **Dependencies** between instructions may lead to stalls
 - DIV.D F0, F2, F4
 - ADD.D F10, F0, F3

RAW must enter the execution stage in order
- The dependencies between instructions limit the order of execution of these instructions (impose in order execution). In the 2nd example ADD.D **must** execute after DIV.D has completed. On the other hand, parallel instructions **may** execute in the any order (out-of-order execution). In the 1st example ADD.D can execute before DIV.D.

Dependencies between Instructions

- **(True) Data Dependences** : instructions are data dependent when there is a chain of RAW hazards between them.

Loop:

L.D **F0**, 0(R1)

ADD.D **F4**, **F0**, F2

S.D **F4**, 0(R1)

DADDI **R1**, R1, -8

BNE **R1**, R2, Loop

S.D F0, 100(R4)

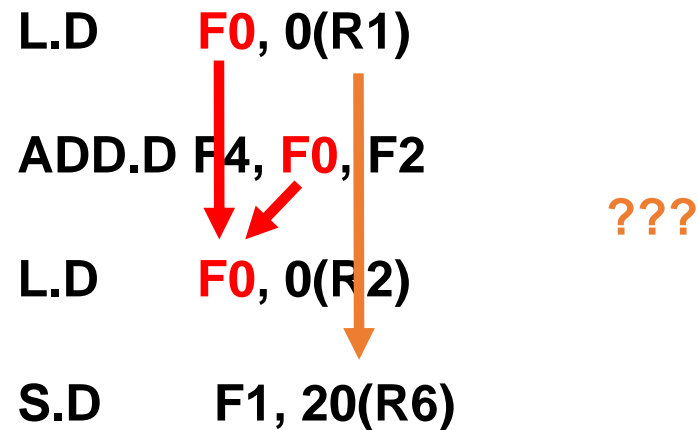
L.D F1, 20(R6)

???

What does this code do?

Dependencies between Instructions

- **Name Dependencies** : instructions are name dependent when there is a WAR (anti-dependence) or WAW (output-dependence) hazard between them.



Spot the anti-dependence!

- **Control Dependencies** : Instructions dependent via branches.
if p1 { S1; }

Hazard Issues

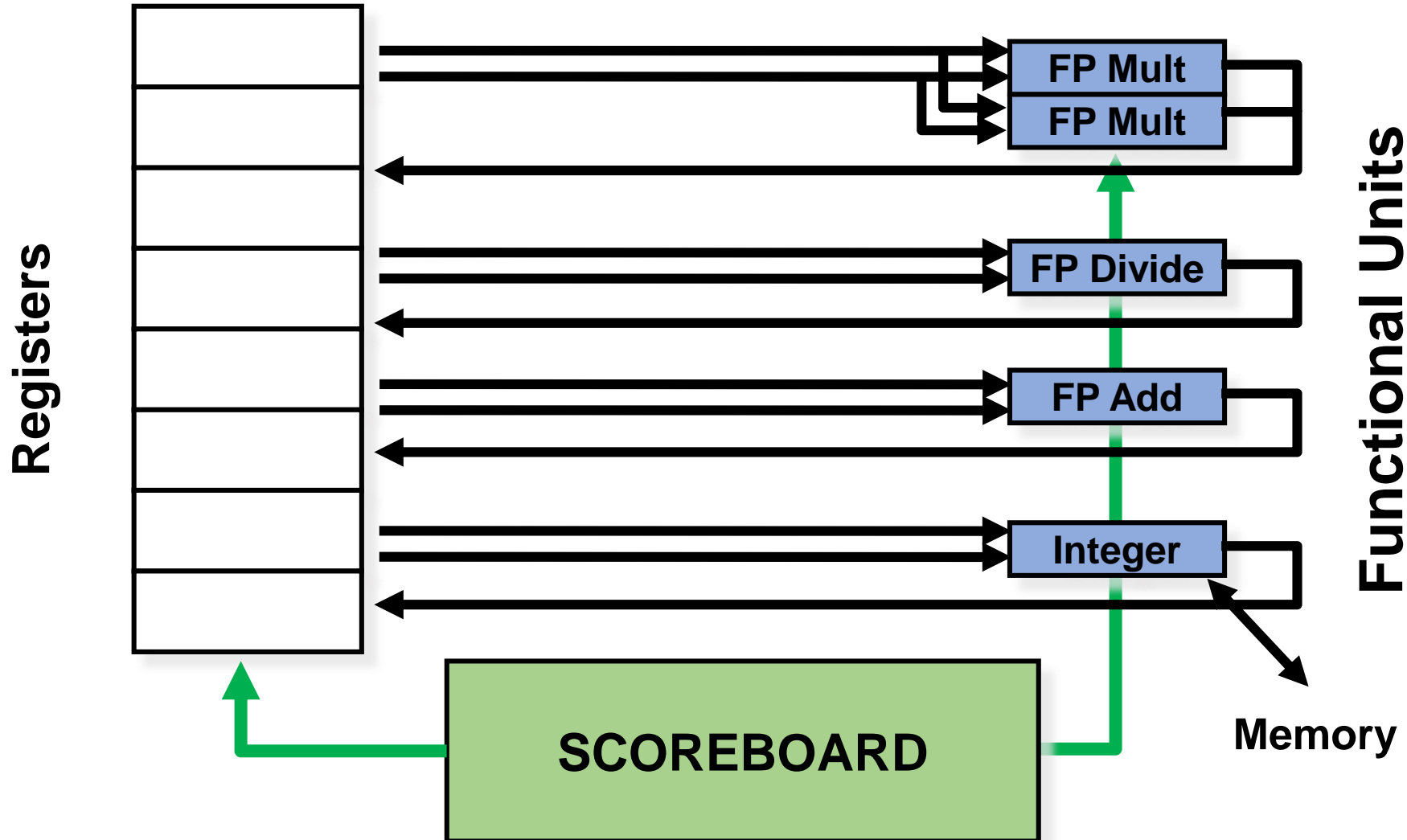
- How do we prevent WAR and WAW hazards?
- How do we handle variable latency execution units?
 - Forwarding for handling RAW hazards is a lot harder!

| Instruction | Clock Cycle Number | | | | | | | | | | | | | | | | |
|----------------|--------------------|----|----|-----|-------|----|-------|-------|-------|-------|-------|-------|-------|-------|-------|-----|----|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| LD F6,34(R2) | IF | ID | EX | MEM | WB | | | | | | | | | | | | |
| LD F2,45(R3) | | IF | ID | EX | MEM | WB | | | | | | | | | | | |
| MULTD F0,F2,F4 | | | IF | ID | stall | M1 | M2 | M3 | M4 | M5 | M6 | M7 | M8 | M9 | M10 | MEM | WB |
| SUBD F8,F6,F2 | | | | IF | ID | A1 | A2 | MEM | WB | | | | | | | | |
| DIVD F10,F0,F6 | | | | | IF | ID | stall | stall | stall | stall | stall | stall | stall | stall | stall | D1 | D2 |
| ADDD F6,F8,F2 | | | | | | IF | ID | A1 | A2 | MEM | WB | | | | | | |

Scoreboard: Out-of-Order Execution

- Split the ID stage into two different stages:
 1. **Issue:** Decode instruction, check for structural hazards. In order instruction issue.
 2. **Read Operands:** Wait until all data hazards are resolved, then read source registers. Out-of-order execution.
- Scoreboard appeared in CDC6600 (1963)
- The instructions execute when they do not depend on previous instructions (which have not been executed yet) and at the same time there are no hazards.
- CDC6600: In order issue, out-of-order execution, out-of-order commit (or completion)
 - 16 functional units: 4 floating-point units, 5 units for memory references, and 7 units for integer operations
 - No forwarding!

Scoreboard Architecture (CDC 6600)



Instruction to scoreboard \Rightarrow
data dependences \Rightarrow
hazard detection and
resolution centralized

Scoreboard: decides when
the instruction can **execute**
and when it can **write its
result**

Scoreboard Implications

- Out-of-order completion => **WAR, WAW hazards?**
- Solutions for WAR:
 - Stall register *write-backs* until registers have been read.
 - Read registers only on the **Read Operands** stage
- Solutions for WAW:
 - Detect this hazard and stall the new instruction (no issue) until the previous instruction has completed/executed.
- There is no register renaming!
- Multiple instruction in execution stage => multiple execution units or pipelined execution units
- The Scoreboard keeps information about the dependencies of the instructions that have been issued.
- The Scoreboard-based pipeline replaces the ID stage with 2 stages: **Issue** and **Read Operands**

Four Stages of Scoreboard Control

- **Issue:** decode instruction & record data dependences & check for structural hazards (ID1)
 - Instructions issued in program order (for hazard checking)
 - Don't issue if **structural hazard**
 - Don't issue if instruction is **output dependent** on any previously issued but uncompleted instruction (no WAW hazards)
- **Read operands:** wait for data hazards to be resolved, then read registers (ID2)
 - All real dependencies (RAW hazards) resolved in this stage, since we wait for instructions to write back data.
 - **No forwarding of data** in this model!

Four Stages of Scoreboard Control

- **Execution:** execute the instruction on a functional unit (EX)
 - The functional unit begins execution upon receiving operands. When the result is ready, it notifies the scoreboard that it has completed execution.
- **Write result:** End of execution (WB)
 - Stall until there are no WAR hazards with previous instructions:

Example:

| | |
|-------|-----------|
| DIV.D | F0,F2,F4 |
| ADD.D | F10,F0,F8 |
| SUB.D | F8,F8,F14 |

- CDC 6600 scoreboard would stall SUBD until ADDD reads operands
- Instructions write their results into the register file as soon as they complete execution (assuming no WAR hazard)

Data structure: 3 parts of Scoreboard

- **Instruction status:** In which of the four stages is the instruction
- **Functional unit status:** Indicates the current state of the functional unit (FU). 9 fields per functional unit.

Busy: Indicates whether the unit is busy or not
Op: Operation to perform in the unit (e.g., + or -)
Fi: Destination register
Fj, Fk: Source-register numbers
Qj, Qk: Functional units producing source registers Fj, Fk
Rj, Rk: Flags indicating when Fj, Fk are ready and not yet read

- **Register result status:** Indicates which functional unit will write each register, if any. Empty when no pending instructions will write that register

Scoreboard Example

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Oper</i> | <i>Comp</i> | <i>Result</i> |
|-------------|----------|----------|--------------|-------------|-------------|---------------|
| L.D | F6 | 34+ | R2 | | | |
| L.D | F2 | 45+ | R3 | | | |
| MULT.D | F0 | F2 | F4 | | | |
| SUB.D | F8 | F6 | F2 | | | |
| DIV.D | F10 | F0 | F6 | | | |
| ADD.D | F6 | F8 | F2 | | | |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest</i> <i>Fi</i> | <i>S1</i> <i>Fj</i> | <i>S2</i> <i>Fk</i> | <i>FU</i> <i>Qj</i> | <i>FU</i> <i>Qk</i> | <i>Fj?</i> <i>Rj</i> | <i>Fk?</i> <i>Rk</i> |
|-------------|-------------|-------------|-----------|--------------------------|------------------------|------------------------|------------------------|------------------------|-------------------------|-------------------------|
| | Integer | No | | | | | | | | |
| | Mult1 | No | | | | | | | | |
| | Mult2 | No | | | | | | | | |
| | Add | No | | | | | | | | |
| | Divide | No | | | | | | | | |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-----------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| <i>FU</i> | | | | | | | | | |

Detailed Scoreboard Pipeline Control

| Instruction status | Wait until | Bookkeeping |
|--------------------|---|--|
| Issue | Busy (FU)=No and result(D)=0 (struct hazard, WAW) | Busy(FU) \leftarrow yes; Op(FU) \leftarrow op; Fi(FU) \leftarrow `D'; Fj(FU) \leftarrow `S1'; Fk(FU) \leftarrow `S2'; Qj \leftarrow Result('S1'); Qk \leftarrow Result(`S2'); Rj \leftarrow not Qj; Rk \leftarrow not Qk; Result('D') \leftarrow FU; |
| Read operands | Rj and Rk (RAW) | Clear flags: Rj \leftarrow No; Rk \leftarrow No |
| Execution complete | Functional unit done | |
| Write result | $\forall f((Fj(f) \neq Fi(FU)$ or $Rj(f) = \text{No}) \&$ $(Fk(f) \neq Fi(FU)$ or $Rk(f) = \text{No}))$ (WAR) | $\forall f(\text{if } Qj(f) = \text{FU then } Rj(f) \leftarrow \text{Yes});$ $\forall f(\text{if } Qk(f) = \text{FU then } Rj(f) \leftarrow \text{Yes});$ Result(Fi(FU)) \leftarrow 0; Busy(FU) \leftarrow No |

Scoreboard Example: Cycle 1

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | | |
| LD | F2 | 45+ | R3 | | | |
| MULTD | F0 | F2 | F4 | | | |
| SUBD | F8 | F6 | F2 | | | |
| DIVD | F10 | F0 | F6 | | | |
| ADDD | F6 | F8 | F2 | | | |

Functional unit status:

| Time | Name | Busy | Op | dest | | FU | | Fj? Fk? | |
|------|---------|------|------|------|----|----|----|---------|-----|
| | | | | Fi | Fj | Fk | Oj | Ok | Rj |
| | Integer | Yes | Load | F6 | | R2 | | | Yes |
| | Mult1 | No | | | | | | | |
| | Mult2 | No | | | | | | | |
| | Add | No | | | | | | | |
| | Divide | No | | | | | | | |

Register result status:

| | F0 | F2 | F4 | F6 | F8 | F10 | F12 | ... | F30 |
|----|----|----|----|---------|----|-----|-----|-----|-----|
| FU | | | | Integer | | | | | |

Clock
1

Scoreboard Example: Cycle 2

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | 2 | |
| LD | F2 | 45+ | R3 | | | |
| MULTD | F0 | F2 | F4 | | | |
| SUBD | F8 | F6 | F2 | | | |
| DIVD | F10 | F0 | F6 | | | |
| ADDD | F6 | F8 | F2 | | | |

Issue second LD?

No. Structural Hazard!

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | Yes | Load | F6 | | R2 | | | | No |
| | Mult1 | No | | | | | | | | |
| | Mult2 | No | | | | | | | | |
| | Add | No | | | | | | | | |
| | Divide | No | | | | | | | | |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-------------------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 2 | <i>FU</i> Integer | | | | | | | | |

Scoreboard Example: Cycle 3

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | Issue | Read Oper | Exec Comp | Write Result |
|-------------|----------|----------|-------|-----------|-----------|--------------|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 |
| LD | F2 | 45+ | R3 | | | |
| MULTD | F0 | F2 | F4 | | | |
| SUBD | F8 | F6 | F2 | | | |
| DIVD | F10 | F0 | F6 | | | |
| ADDD | F6 | F8 | F2 | | | |

Issue MULTD?

No. In order issue!

Functional unit status:

| Time | Name | Busy | Op | dest <i>Fi</i> | <i>S1</i> <i>Fj</i> | <i>S2</i> <i>Fk</i> | <i>FU</i> <i>Qj</i> | <i>FU</i> <i>Qk</i> | <i>Fj?</i> <i>Rj</i> | <i>Fk?</i> <i>Rk</i> |
|------|---------|------|------|-------------------|------------------------|------------------------|------------------------|------------------------|-------------------------|-------------------------|
| | Integer | Yes | Load | F6 | | R2 | | | | No |
| | Mult1 | No | | | | | | | | |
| | Mult2 | No | | | | | | | | |
| | Add | No | | | | | | | | |
| | Divide | No | | | | | | | | |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-------------------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 3 | <i>FU</i> Integer | | | | | | | | |

Scoreboard Example: Cycle 4

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ R2 | 1 | 2 | 3 | 4 |
| LD | F2 | 45+ R3 | | | | |
| MULTD | F0 | F2 F4 | | | | |
| SUBD | F8 | F6 F2 | | | | |
| DIVD | F10 | F0 F6 | | | | |
| ADDD | F6 | F8 F2 | | | | |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest</i> <i>Fi</i> | <i>S1</i> <i>Fj</i> | <i>S2</i> <i>Fk</i> | <i>FU</i> <i>Qj</i> | <i>FU</i> <i>Qk</i> | <i>Fj?</i> <i>Rj</i> | <i>Fk?</i> <i>Rk</i> |
|-------------|-------------|-------------|-----------|--------------------------|------------------------|------------------------|------------------------|------------------------|-------------------------|-------------------------|
| | Integer | No | | | | | | | | |
| | Mult1 | No | | | | | | | | |
| | Mult2 | No | | | | | | | | |
| | Add | No | | | | | | | | |
| | Divide | No | | | | | | | | |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 4 | | | | | | | | | |

Scoreboard Example: Cycle 5

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 4 |
| LD | F2 | 45+ | R3 | 5 | | |
| MULTD | F0 | F2 | F4 | | | |
| SUBD | F8 | F6 | F2 | | | |
| DIVD | F10 | F0 | F6 | | | |
| ADDD | F6 | F8 | F2 | | | |

Functional unit status:

| Time | Name | Busy | Op | dest | | FU | | Fj? Fk? | |
|------|---------|------|------|------|----|----|----|---------|-----|
| | | | | Fi | Fj | Qj | Qk | Rj | Rk |
| | Integer | Yes | Load | F2 | R3 | | | | Yes |
| | Mult1 | No | | | | | | | |
| | Mult2 | No | | | | | | | |
| | Add | No | | | | | | | |
| | Divide | No | | | | | | | |

Register result status:

| Clock | F0 | F2 | F4 | F6 | F8 | F10 | F12 | ... | F30 |
|-------|------------|----|----|----|----|-----|-----|-----|-----|
| 5 | FU Integer | | | | | | | | |

Scoreboard Example: Cycle 6

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ R2 | 1 | 2 | 3 | 4 |
| LD | F2 | 45+ R3 | 5 | 6 | | |
| MULTD | F0 | F2 F4 | 6 | | | |
| SUBD | F8 | F6 F2 | | | | |
| DIVD | F10 | F0 F6 | | | | |
| ADDD | F6 | F8 F2 | | | | |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | Yes | Load | F2 | | R3 | | | | No |
| | Mult1 | Yes | Mult | F0 | F2 | F4 | Integer | | No | Yes |
| | Mult2 | No | | | | | | | | |
| | Add | No | | | | | | | | |
| | Divide | No | | | | | | | | |

Register result status:

| Clock | F0 | F2 | F4 | F6 | F8 | F10 | F12 | ... | F30 |
|-------|-------|---------|----|----|----|-----|-----|-----|-----|
| 6 | Mult1 | Integer | | | | | | | |

Scoreboard Example: Cycle 7

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ R2 | 1 | 2 | 3 | 4 |
| LD | F2 | 45+ R3 | 5 | 6 | 7 | |
| MULTD | F0 | F2 F4 | 6 | | | |
| SUBD | F8 | F6 F2 | 7 | | | |
| DIVD | F10 | F0 F6 | | | | |
| ADDD | F6 | F8 F2 | | | | |

Read MULTD operands?

No. RAW Hazard!

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| Integer | | Yes | Load | F2 | | R3 | | | | No |
| Mult1 | | Yes | Mult | F0 | F2 | F4 | Integer | | No | Yes |
| Mult2 | | No | | | | | | | | |
| Add | | Yes | Sub | F8 | F6 | F2 | | Integer | Yes | No |
| Divide | | No | | | | | | | | |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-----------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 7 | | | | | | | | | |
| <i>FU</i> | Mult1 | Integer | | | Add | | | | |

Scoreboard Example: Cycle 8a

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> | |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|---|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 | 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 | |
| MULTD | F0 | F2 | F4 | 6 | | | |
| SUBD | F8 | F6 | F2 | 7 | | | |
| DIVD | F10 | F0 | F6 | 8 | | | |
| ADDD | F6 | F8 | F2 | | | | |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | Yes | Load | F2 | | R3 | | | | No |
| | Mult1 | Yes | Mult | F0 | F2 | F4 | Integer | | No | Yes |
| | Mult2 | No | | | | | | | | |
| | Add | Yes | Sub | F8 | F6 | F2 | | Integer | Yes | No |
| | Divide | Yes | Div | F10 | F0 | F6 | Mult1 | | No | Yes |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 8 | Mult1 | Integer | | | Add | Divide | | | |

Scoreboard Example: Cycle 8b

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 8 |
| MULTD | F0 | F2 | F4 | 6 | | |
| SUBD | F8 | F6 | F2 | 7 | | |
| DIVD | F10 | F0 | F6 | 8 | | |
| ADDD | F6 | F8 | F2 | | | |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| | Mult1 | Yes | Mult | F0 | F2 | F4 | | | Yes | Yes |
| | Mult2 | No | | | | | | | | |
| | Add | Yes | Sub | F8 | F6 | F2 | | | Yes | Yes |
| | Divide | Yes | Div | F10 | F0 | F6 | Mult1 | | No | Yes |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> | | | | | | | | | | |
|-----------|---|-----------|-----------|-----------|-----------|------------|------------|-----|------------|-----------|-------|--|--|--|-----|--------|--|--|--|
| 8 | <table border="0"> <tr> <td><i>FU</i></td> <td>Mult1</td> <td></td> <td></td> <td></td> <td>Add</td> <td>Divide</td> <td></td> <td></td> <td></td> </tr> </table> | | | | | | | | | <i>FU</i> | Mult1 | | | | Add | Divide | | | |
| <i>FU</i> | Mult1 | | | | Add | Divide | | | | | | | | | | | | | |

Scoreboard Example: Cycle 9

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> | |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|---|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 | 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 | 8 |
| MULTD | F0 | F2 | F4 | 6 | | | |
| SUBD | F8 | F6 | F2 | 7 | | | |
| DIVD | F10 | F0 | F6 | 8 | | | |
| ADDD | F6 | F8 | F2 | | | | |

Read MULTD & SUBD operands? **Yes**

Issue ADDD?

No. Structural hazard!

Functional unit status:

| Time | Name | Busy | Op | dest <i>Fi</i> | <i>S1</i> <i>Fj</i> | <i>S2</i> <i>Fk</i> | <i>FU</i> <i>Qj</i> | <i>FU</i> <i>Qk</i> | <i>Fj?</i> <i>Rj</i> | <i>Fk?</i> <i>Rk</i> |
|------|---------|------|------|-------------------|------------------------|------------------------|------------------------|------------------------|-------------------------|-------------------------|
| | | | | | | | | | | |
| | Integer | No | | | | | | | | |
| 10 | Mult1 | Yes | Mult | F0 | F2 | F4 | | | No | No |
| | Mult2 | No | | | | | | | | |
| 2 | Add | Yes | Sub | F8 | F6 | F2 | | | No | No |
| | Divide | Yes | Div | F10 | F0 | F6 | Mult1 | | No | Yes |

Execution Latency →

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 9 | FU Mult1 | | | | Add | Divide | | | |

Scoreboard Example: Cycle 10

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 8 |
| MULTD | F0 | F2 | F4 | 6 | 9 | |
| SUBD | F8 | F6 | F2 | 7 | 9 | |
| DIVD | F10 | F0 | F6 | 8 | | |
| ADDD | F6 | F8 | F2 | | | |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| 9 | Mult1 | Yes | Mult | F0 | F2 | F4 | | | No | No |
| | Mult2 | No | | | | | | | | |
| 1 | Add | Yes | Sub | F8 | F6 | F2 | | | No | No |
| | Divide | Yes | Div | F10 | F0 | F6 | Mult1 | | No | Yes |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 10 | FU Mult1 | | | | Add | Divide | | | |

Scoreboard Example: Cycle 11

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 8 |
| MULTD | F0 | F2 | F4 | 6 | 9 | |
| SUBD | F8 | F6 | F2 | 7 | 9 | 11 |
| DIVD | F10 | F0 | F6 | 8 | | |
| ADDD | F6 | F8 | F2 | | | |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| 8 | Mult1 | Yes | Mult | F0 | F2 | F4 | | | No | No |
| | Mult2 | No | | | | | | | | |
| 0 | Add | Yes | Sub | F8 | F6 | F2 | | | No | No |
| | Divide | Yes | Div | F10 | F0 | F6 | Mult1 | | No | Yes |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|----------------------------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 11 | <i>FU</i> Mult1 Add Divide | | | | | | | | |

Scoreboard Example: Cycle 12

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ R2 | 1 | 2 | 3 | 4 |
| LD | F2 | 45+ R3 | 5 | 6 | 7 | 8 |
| MULTD | F0 | F2 F4 | 6 | 9 | | |
| SUBD | F8 | F6 F2 | 7 | 9 | 11 | 12 |
| DIVD | F10 | F0 F6 | 8 | | | |
| ADDD | F6 | F8 F2 | | | | |

Read DIVD Operands?

No. RAW Hazard!

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| 7 | Mult1 | Yes | Mult | F0 | F2 | F4 | | | No | No |
| | Mult2 | No | | | | | | | | |
| | Add | No | | | | | | | | |
| | Divide | Yes | Div | F10 | F0 | F6 | Mult1 | | No | Yes |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> | |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|--|
| 12 | FU Mult1 | | Divide | | | | | | | |

Scoreboard Example: Cycle 13

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ R2 | 1 | 2 | 3 | 4 |
| LD | F2 | 45+ R3 | 5 | 6 | 7 | 8 |
| MULTD | F0 | F2 F4 | 6 | 9 | | |
| SUBD | F8 | F6 F2 | 7 | 9 | 11 | 12 |
| DIVD | F10 | F0 F6 | 8 | | | |
| ADDD | F6 | F8 F2 | 13 | | | |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| 6 | Mult1 | Yes | Mult | F0 | F2 | F4 | | | No | No |
| | Mult2 | No | | | | | | | | |
| | Add | Yes | Add | F6 | F8 | F2 | | | Yes | Yes |
| | Divide | Yes | Div | F10 | F0 | F6 | Mult1 | | No | Yes |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 13 | FU Mult1 | | | Add | | Divide | | | |

Scoreboard Example: Cycle 14

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 8 |
| MULTD | F0 | F2 | F4 | 6 | 9 | |
| SUBD | F8 | F6 | F2 | 7 | 9 | 11 12 |
| DIVD | F10 | F0 | F6 | 8 | | |
| ADDD | F6 | F8 | F2 | 13 | 14 | |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| 5 | Mult1 | Yes | Mult | F0 | F2 | F4 | | | No | No |
| | Mult2 | No | | | | | | | | |
| 2 | Add | Yes | Add | F6 | F8 | F2 | | | No | No |
| | Divide | Yes | Div | F10 | F0 | F6 | Mult1 | | No | Yes |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 14 | FU Mult1 | | | Add | | Divide | | | |

Scoreboard Example: Cycle 15

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 8 |
| MULTD | F0 | F2 | F4 | 6 | 9 | |
| SUBD | F8 | F6 | F2 | 7 | 9 | 11 12 |
| DIVD | F10 | F0 | F6 | 8 | | |
| ADDD | F6 | F8 | F2 | 13 | 14 | |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| 4 | Mult1 | Yes | Mult | F0 | F2 | F4 | | | No | No |
| | Mult2 | No | | | | | | | | |
| 1 | Add | Yes | Add | F6 | F8 | F2 | | | No | No |
| | Divide | Yes | Div | F10 | F0 | F6 | Mult1 | | No | Yes |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 15 | FU Mult1 | | | Add | | Divide | | | |

Scoreboard Example: Cycle 16

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 8 |
| MULTD | F0 | F2 | F4 | 6 | 9 | |
| SUBD | F8 | F6 | F2 | 7 | 9 | 11 12 |
| DIVD | F10 | F0 | F6 | 8 | | |
| ADDD | F6 | F8 | F2 | 13 | 14 | 16 |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| 3 | Mult1 | Yes | Mult | F0 | F2 | F4 | | | No | No |
| | Mult2 | No | | | | | | | | |
| 0 | Add | Yes | Add | F6 | F8 | F2 | | | No | No |
| | Divide | Yes | Div | F10 | F0 | F6 | Mult1 | | No | Yes |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 16 | FU Mult1 | | Add | | | Divide | | | |

Scoreboard Example: Cycle 17

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Oper</i> | <i>Comp</i> | <i>Write</i> |
|-------------|----------|----------|--------------|-------------|-------------|--------------|
| LD | F6 | 34+ R2 | 1 | 2 | 3 | 4 |
| LD | F2 | 45+ R3 | 5 | 6 | 7 | 8 |
| MULTD | F0 | F2 F4 | 6 | 9 | | |
| SUBD | F8 | F6 F2 | 7 | 9 | 11 | 12 |
| DIVD | F10 | F0 F6 | 8 | | | |
| ADDD | F6 | F8 F2 | 13 | 14 | 16 | |

Write ADDD result?

No. WAR hazard!

Functional unit status:

| Time | Name | Busy | Op | dest <i>Fi</i> | <i>S1</i> <i>Fj</i> | <i>S2</i> <i>Fk</i> | <i>FU</i> <i>Qj</i> | <i>FU</i> <i>Qk</i> | <i>Fj?</i> <i>Rj</i> | <i>Fk?</i> <i>Rk</i> |
|------|---------|------|------|-------------------|------------------------|------------------------|------------------------|------------------------|-------------------------|-------------------------|
| | Integer | No | | | | | | | | |
| 2 | Mult1 | Yes | Mult | F0 | F2 | F4 | | | No | No |
| | Mult2 | No | | | | | | | | |
| | Add | Yes | Add | F6 | F8 | F2 | | | No | No |
| | Divide | Yes | Div | F10 | F0 | F6 | Mult1 | | No | Yes |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-----------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 17 | | | | | | | | | |
| <i>FU</i> | Mult1 | | | Add | | Divide | | | |

Scoreboard Example: Cycle 18

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 8 |
| MULTD | F0 | F2 | F4 | 6 | 9 | |
| SUBD | F8 | F6 | F2 | 7 | 9 | 11 12 |
| DIVD | F10 | F0 | F6 | 8 | | |
| ADDD | F6 | F8 | F2 | 13 | 14 | 16 |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| 1 | Mult1 | Yes | Mult | F0 | F2 | F4 | | | No | No |
| | Mult2 | No | | | | | | | | |
| | Add | Yes | Add | F6 | F8 | F2 | | | No | No |
| | Divide | Yes | Div | F10 | F0 | F6 | Mult1 | | No | Yes |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 18 | FU Mult1 | | Add | | Divide | | | | |

Scoreboard Example: Cycle 19

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 8 |
| MULTD | F0 | F2 | F4 | 6 | 9 | 19 |
| SUBD | F8 | F6 | F2 | 7 | 9 | 11 12 |
| DIVD | F10 | F0 | F6 | 8 | | |
| ADDD | F6 | F8 | F2 | 13 | 14 | 16 |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| 0 | Mult1 | Yes | Mult | F0 | F2 | F4 | | | No | No |
| | Mult2 | No | | | | | | | | |
| | Add | Yes | Add | F6 | F8 | F2 | | | No | No |
| | Divide | Yes | Div | F10 | F0 | F6 | Mult1 | | No | Yes |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 19 | FU Mult1 | | Add | | Divide | | | | |

Scoreboard Example: Cycle 20

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 8 |
| MULTD | F0 | F2 | F4 | 6 | 9 | 19 20 |
| SUBD | F8 | F6 | F2 | 7 | 9 | 11 12 |
| DIVD | F10 | F0 | F6 | 8 | | |
| ADDD | F6 | F8 | F2 | 13 | 14 | 16 |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| | Mult1 | No | | | | | | | | |
| | Mult2 | No | | | | | | | | |
| | Add | Yes | Add | F6 | F8 | F2 | | | No | No |
| | Divide | Yes | Div | F10 | F0 | F6 | | | Yes | Yes |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 20 | | | | Add | | Divide | | | |

Scoreboard Example: Cycle 21

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ R2 | 1 | 2 | 3 | 4 |
| LD | F2 | 45+ R3 | 5 | 6 | 7 | 8 |
| MULTD | F0 | F2 F4 | 6 | 9 | 19 | 20 |
| SUBD | F8 | F6 F2 | 7 | 9 | 11 | 12 |
| DIVD | F10 | F0 F6 | 8 | 21 | | |
| ADDD | F6 | F8 F2 | 13 | 14 | 16 | |

No WAR hazard anymore.

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| | Mult1 | No | | | | | | | | |
| | Mult2 | No | | | | | | | | |
| | Add | Yes | Add | F6 | F8 | F2 | | | No | No |
| | Divide | Yes | Div | F10 | F0 | F6 | | | No | No |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 21 | | | | Add | | Divide | | | |

Scoreboard Example: Cycle 22

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 8 |
| MULTD | F0 | F2 | F4 | 6 | 9 | 19 20 |
| SUBD | F8 | F6 | F2 | 7 | 9 | 11 12 |
| DIVD | F10 | F0 | F6 | 8 | 21 | |
| ADDD | F6 | F8 | F2 | 13 | 14 | 16 22 |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| | Mult1 | No | | | | | | | | |
| | Mult2 | No | | | | | | | | |
| | Add | No | | | | | | | | |
| 39 | Divide | Yes | Div | F10 | F0 | F6 | | | No | No |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 22 | FU Divide | | | | | | | | |

After a few clock cycles...

Scoreboard Example: Cycle 61

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 8 |
| MULTD | F0 | F2 | F4 | 6 | 9 | 19 20 |
| SUBD | F8 | F6 | F2 | 7 | 9 | 11 12 |
| DIVD | F10 | F0 | F6 | 8 | 21 | 61 |
| ADDD | F6 | F8 | F2 | 13 | 14 | 16 22 |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| | Mult1 | No | | | | | | | | |
| | Mult2 | No | | | | | | | | |
| | Add | No | | | | | | | | |
| | 0 Divide | Yes | Div | F10 | F0 | F6 | | | No | No |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 61 | FU Divide | | | | | | | | |

Scoreboard Example: Cycle 62

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 8 |
| MULTD | F0 | F2 | F4 | 6 | 9 | 19 20 |
| SUBD | F8 | F6 | F2 | 7 | 9 | 11 12 |
| DIVD | F10 | F0 | F6 | 8 | 21 | 61 62 |
| ADDD | F6 | F8 | F2 | 13 | 14 | 16 22 |

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest Fi</i> | <i>S1 Fj</i> | <i>S2 Fk</i> | <i>FU Qj</i> | <i>FU Qk</i> | <i>Fj? Rj</i> | <i>Fk? Rk</i> |
|-------------|-------------|-------------|-----------|----------------|--------------|--------------|--------------|--------------|---------------|---------------|
| | Integer | No | | | | | | | | |
| | Mult1 | No | | | | | | | | |
| | Mult2 | No | | | | | | | | |
| | Add | No | | | | | | | | |
| | Divide | No | | | | | | | | |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 62 | <i>FU</i> | | | | | | | | |

Scoreboard Example: Cycle 62

Instruction status:

| Instruction | <i>j</i> | <i>k</i> | <i>Issue</i> | <i>Read Oper</i> | <i>Exec Comp</i> | <i>Write Result</i> | |
|-------------|----------|----------|--------------|------------------|------------------|---------------------|----|
| LD | F6 | 34+ | R2 | 1 | 2 | 3 | 4 |
| LD | F2 | 45+ | R3 | 5 | 6 | 7 | 8 |
| MULTD | F0 | F2 | F4 | 6 | 9 | 19 | 20 |
| SUBD | F8 | F6 | F2 | 7 | 9 | 11 | 12 |
| DIVD | F10 | F0 | F6 | 8 | 21 | 61 | 62 |
| ADDD | F6 | F8 | F2 | 13 | 14 | 16 | 22 |

In-order issue?

Out-of-order execute?

Out-of-order commit?

Functional unit status:

| <i>Time</i> | <i>Name</i> | <i>Busy</i> | <i>Op</i> | <i>dest</i> <i>Fi</i> | <i>S1</i> <i>Fj</i> | <i>S2</i> <i>Fk</i> | <i>FU</i> <i>Qj</i> | <i>FU</i> <i>Qk</i> | <i>Fj?</i> <i>Rj</i> | <i>Fk?</i> <i>Rk</i> |
|-------------|-------------|-------------|-----------|--------------------------|------------------------|------------------------|------------------------|------------------------|-------------------------|-------------------------|
| | Integer | No | | | | | | | | |
| | Mult1 | No | | | | | | | | |
| | Mult2 | No | | | | | | | | |
| | Add | No | | | | | | | | |
| | Divide | No | | | | | | | | |

Register result status:

| Clock | <i>F0</i> | <i>F2</i> | <i>F4</i> | <i>F6</i> | <i>F8</i> | <i>F10</i> | <i>F12</i> | ... | <i>F30</i> |
|-------|-----------|-----------|-----------|-----------|-----------|------------|------------|-----|------------|
| 62 | <i>FU</i> | | | | | | | | |

CDC 6600 Scoreboard

- Speedup 1.7 for FORTRAN programs; 2.5 by hand (outdated measurements)
- Limitations of 6600 scoreboard:
 - No forwarding hardware
 - Limited to instructions in basic block (small window)
 - Small number of functional units (structural hazards), especially integer/load store units
 - Do not issue on structural hazards
 - Wait for WAR hazards
 - Prevent WAW hazards