05 CPU

05.03 Pipeline hazards

- Structural hazards
- Data hazards
- Control-flow hazards
- Multi-cycle instructions
- CPU-time estimate

Structural hazards (problem)

- Von-Neumann architecture: single memory

```
Load    Mem Reg ALU Mem Reg
Instr1      Mem Reg ALU Mem Reg
Instr2          Mem Reg ALU Mem Reg
Instr3              Mem Reg ALU Mem Reg
Instr4                  Mem Reg ALU Mem Reg

Load    Mem Reg ALU Mem Reg
Instr1      Mem Reg ALU Mem Reg
Instr2          Mem Reg ALU Mem Reg
Stall               --- --- --- --- ---
Instr3                  Mem Reg ALU Mem Reg
Instr4                      Mem Reg ALU Mem Reg
```

2 clock cycles per memory access
If 40% of instructions are memory accesses, avgCPI = 1.4

Structural hazards (solution)

- Harvard architecture:
  separate memories for data (DM) and instructions (IM)

```
Load    IM  Reg ALU DM  Reg
Instr1      IM  Reg ALU DM  Reg
Instr2          IM  Reg ALU DM  Reg
Instr3              IM  Reg ALU DM  Reg
Instr4                  IM  Reg ALU DM  Reg
```

Memory accesses cause no structural hazards

Data hazards (problem)

- Define and use / Read after write

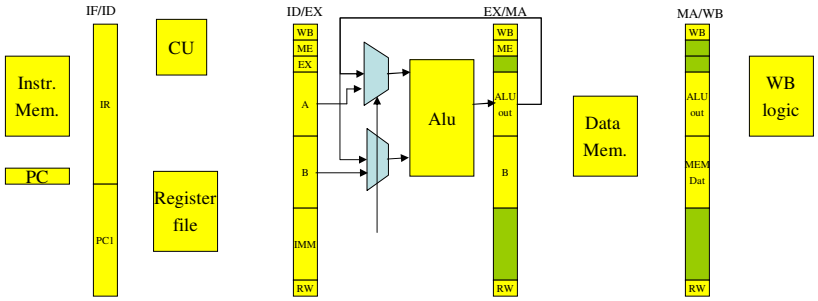
```
ADD r1 r2 r3   IM reg ALU DM  reg
SUB r4 r5 r1       IM  reg ALU DM  reg
AND r6 r1 r7           IM  reg ALU DM  reg
OR  r8 r1 r9               IM  reg ALU DM  reg
ZOR r10 r1 r11                 IM  reg ALU DM  reg
```

```
ADD r1 r2 r3   IM reg ALU DM  reg
SUB r4 r5 r1       IM  --- --- --- reg ALU DM  reg
AND r6 r1 r7                       IM  reg ALU DM  reg
OR  r8 r1 r9                           IM  reg ALU DM  reg
ZOR r10 r1 r11                             IM  reg ALU DM  reg
```

Data hazards (solution 1)

- Data forwarding from MA/WB to ALU

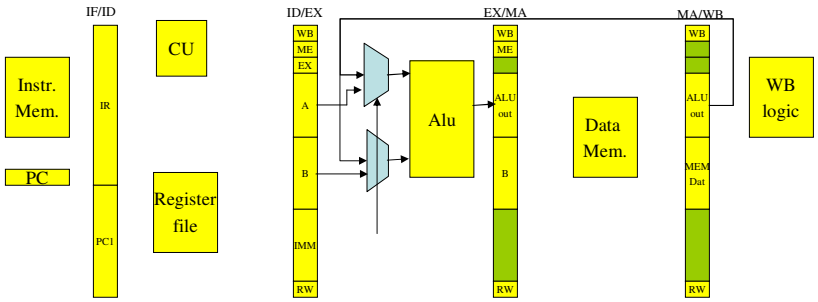
```
ADD r1 r2 r3   IM reg ALU DM  reg
SUB r4 r5 r1      IM  reg ALU DM  reg
```



Data hazards (solution 1)

- Data forwarding from EX/MA to ALU

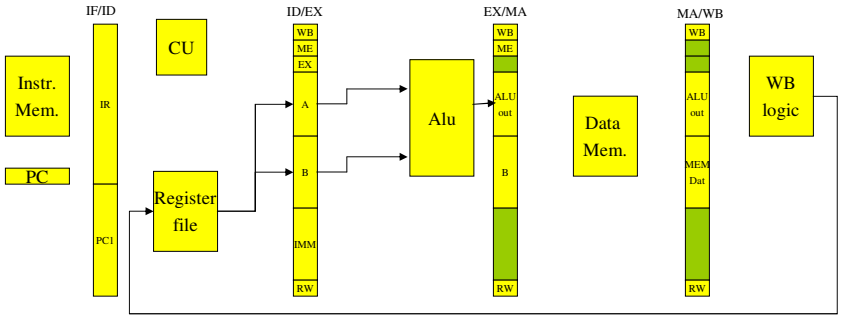
```
ADD r1 r2 r3   IM reg ALU DM  reg
SUB r4 r5 r1      IM  reg ALU DM  reg
AND r6 r1 r7         IM  reg ALU DM  reg
```



Data hazards (solution 3)

- Double-rate registers

```
ADD r1 r2 r3   IM reg ALU DM  reg
SUB r4 r5 r1      IM  reg ALU DM  reg
AND r6 r1 r7          IM  reg ALU DM  reg
OR  r8 r1 r9              IM  reg ALU DM  reg
```



Control-flow hazards (problem)

- Assumptions:
  - Branch recognized during ID
  - Condition tested during EX
  - PC updated during MA

```
CBranch   IF  ID  EX  MEM WB
+1            IF  (abort)
Target            --- --- IF  ID  EX  MEM WB
target+1                      IF  ID  EX  MEM  WB

UCBranch  IF  ID  EX  MEM WB
+1            IF  (abort)
Target            --- IF  ID  EX  MEM WB
target+1              IF  ID  EX  MEM  WB
```

Control-flow hazards (solution 1)

- Assumptions:
  - Branch recognized during ID
  - Target computed during ID (PC updated soon)
  - Condition tested during EX (PC updated soon)

```
CBranch   IF  ID  EX  MEM WB
+1            IF  (abort)
Target            --- IF  ID  EX  MEM WB
target+1                  IF  ID  EX  MEM  WB

UCBranch  IF  ID  EX  MEM WB
+1            IF  (abort)
Target            IF  ID  EX  MEM WB
target+1              IF  ID  EX  MEM  WB
```


Control-flow hazards (solution 2)

- Assumptions:
  - Branch recognized during ID
  - Target computed during ID (PC updated soon)
  - Condition tested during ID (PC updated soon)

```
CBranch   IF  ID  EX  MEM WB
+1            IF  (abort)
Target            IF  ID  EX  MEM WB
target+1              IF  ID  EX  MEM  WB

UCBranch  IF  ID  EX  MEM WB
+1            IF  (abort)
Target            IF  ID  EX  MEM WB
target+1              IF  ID  EX  MEM  WB
```

Control-flow hazards (solution 3)

- Predict untaken:
  - If the branch is not taken, the fetched instruction is not aborted

```
UTBranch  IF  ID  EX  MEM WB
+1            IF  ID  EX  MEM WB
+2                IF  ID  EX  MEM  WB


TBranch   IF  ID  EX  MEM WB
+1            IF  (abort)
Target            IF  ID  EX  MEM WB
target+1              IF  ID  EX  MEM  WB
```


Control-flow hazards (solution 4)

- Delayed branch:
  - The instruction that follows a branch is always executed, regardless of the condition of the branch

```
UTBranch  IF  ID  EX  MEM WB
+1(delay)     IF  ID  EX  MEM WB
+2                IF  ID  EX  MEM  WB


TBranch   IF  ID  EX  MEM WB
+1(delay)     IF  ID  EX  MEM WB
Target            IF  ID  EX  MEM WB
target+1              IF  ID  EX  MEM  WB
```

Multiple-execution cycles (1)

- Different instructions may use different execution units
- The execution phase of an instruction may require more than 1 clock cycle
- Instructions with multiple-execution cycles give rise to data and resource conflicts

```
ADDD F4, F0, F2      IF  ID  EX  EX  MA  WB
ADDD F10, F6, F8         IF  ID  --  EX  EX  MA  WB
ADDD F14, F10, F12           IF  --  ID  --  EX  EX  MA  WB
```


Multiple-execution cycles (2)

- Pipelined execution units
- Replicated execution units

```
ADDD F4, F0, F2      IF  ID  EX  EX  MA  WB
ADDD F10, F6, F8         IF  ID  EX  EX  MA  WB
ADDD F14, F10, F12           IF  ID  --  EX  EX  MA  WB
```

CPU specification

- Functional specification
  - Instruction set
- Interface
  - Pin-out
  - Electrical signals
  - Timing diagrams
- Performance
  - Tclk
  - CPI
  - Latency
  - Repetition time

Performance estimation

$$CPUC = CPI_{avg} IC + N_{stalls} \quad (1)$$

$$CPI_{avg} = \frac{\sum_{i \in IS} CPI_i IC_i}{IC} \quad (2)$$

$$CPUC = \sum_{i \in IS} CPI_i IC_i + N_{stalls} \quad (3)$$

Latency
Repetition time

$$CPI_{avg} > 1$$
$$CPUC > CPI_{avg} IC$$