CS429: Computer Organization and Architecture

Pipeline II

Warren Hunt, Jr. and Bill Young
Department of Computer Sciences
University of Texas at Austin

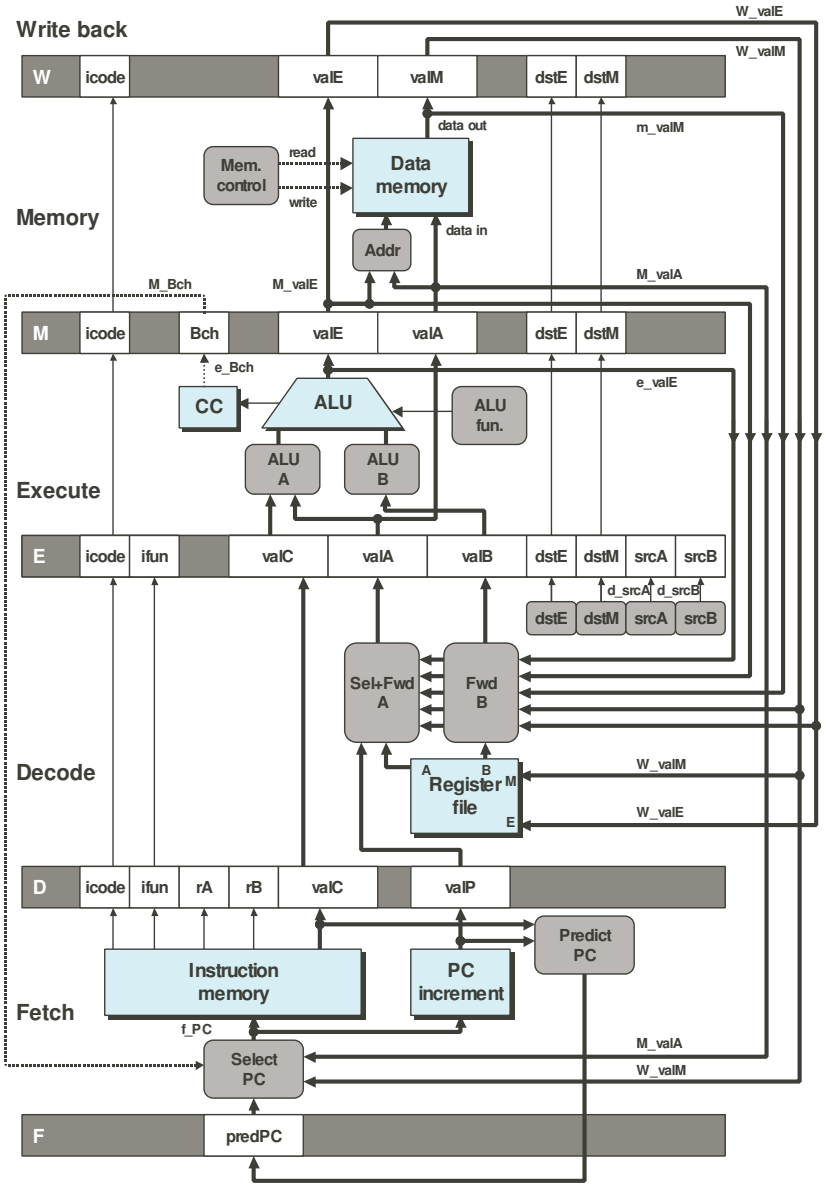
Last updated: October 31, 2014 at 11:03

PIPE- Hardware

Pipeline registers hold intermediate values from instruction execution.

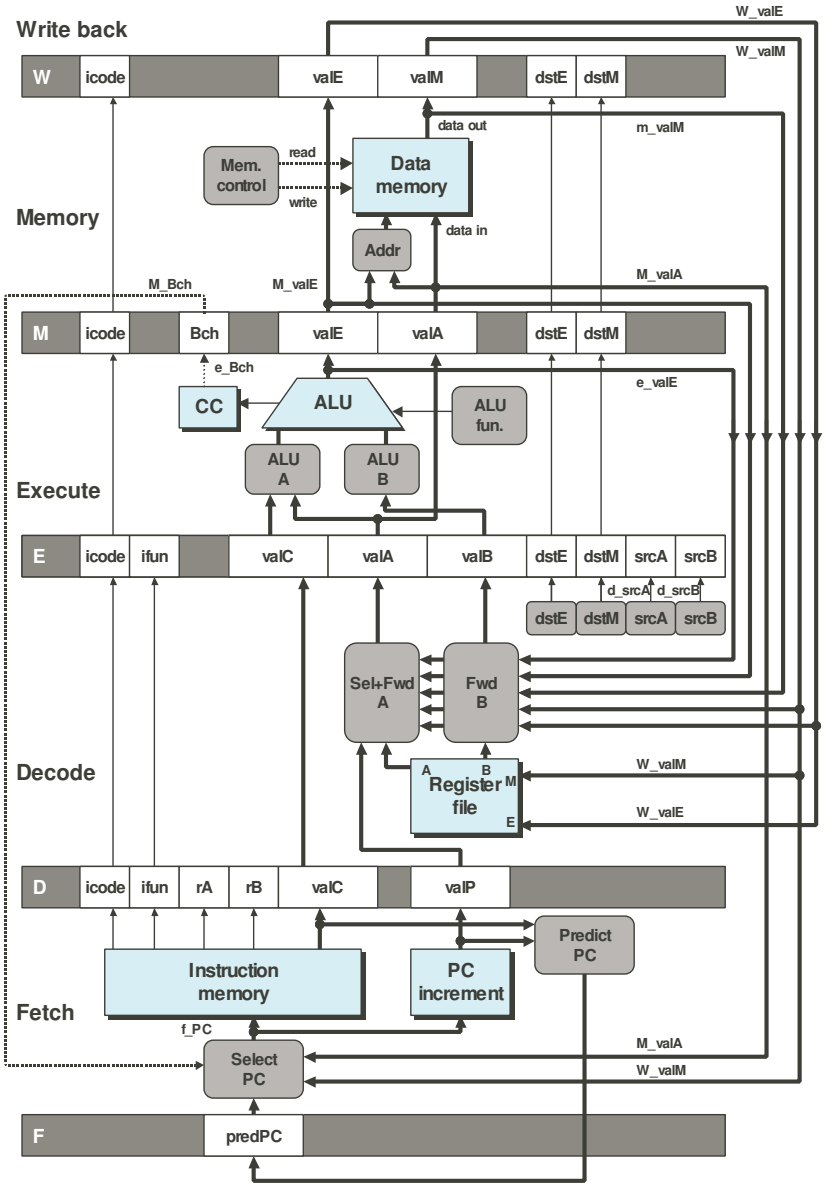
Forward (Upward) Paths

- Values passed from one stage to the next.
- Cannot jump past stages.
- E.g., `valC` must pass through `decode`



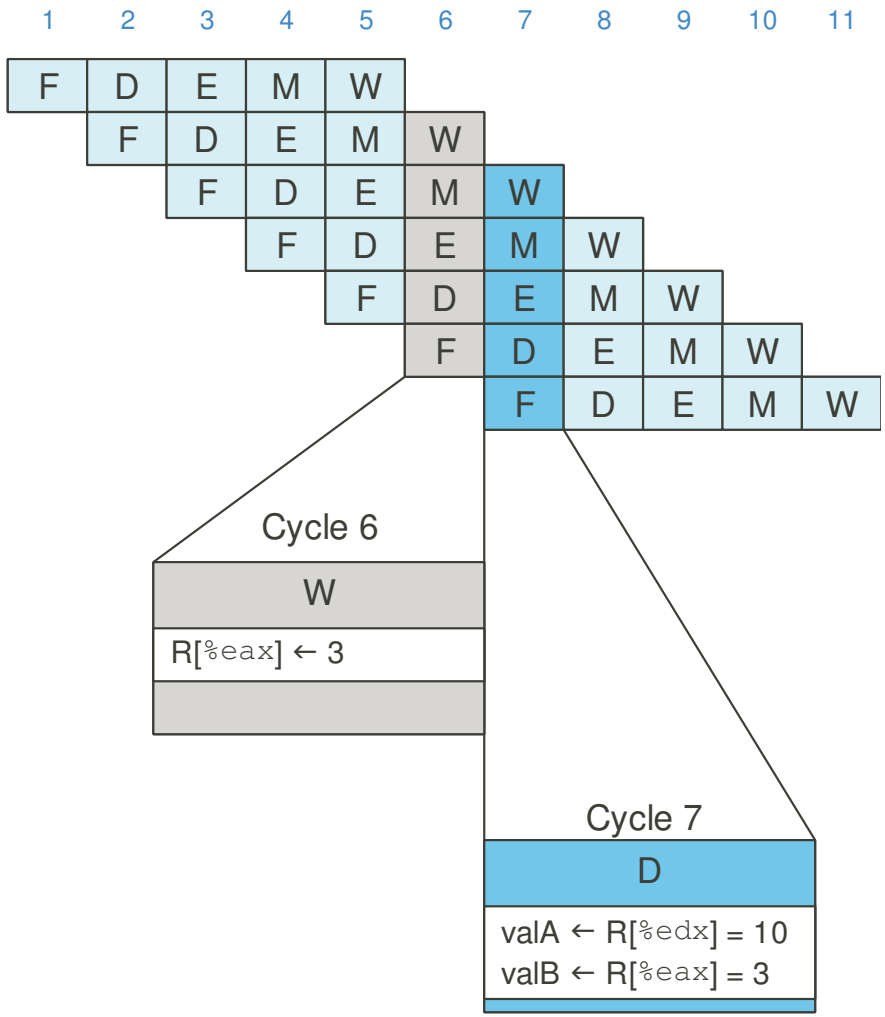
Feedback Paths

- Predicted PC: guess value of next PC
- Branch information:
  - Jump taken/not taken
  - Fall-through or target address
- Return point: read from memory (stack)
- Register updates: To register file write ports



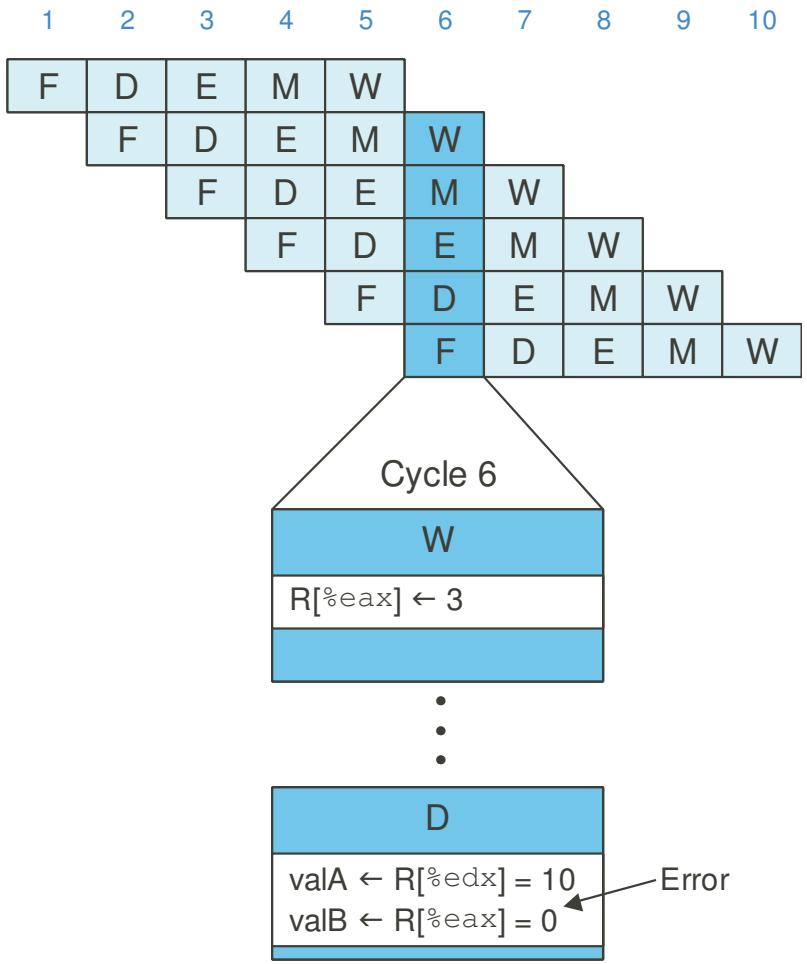
Data Dependencies: 3 Nop's

```
# prog1
0x000: irmovl $10,%edx
0x006: irmovl  $3,%eax
0x00c: nop
0x00d: nop
0x00e: nop
0x00f: addl %edx,%eax
0x011: halt
```



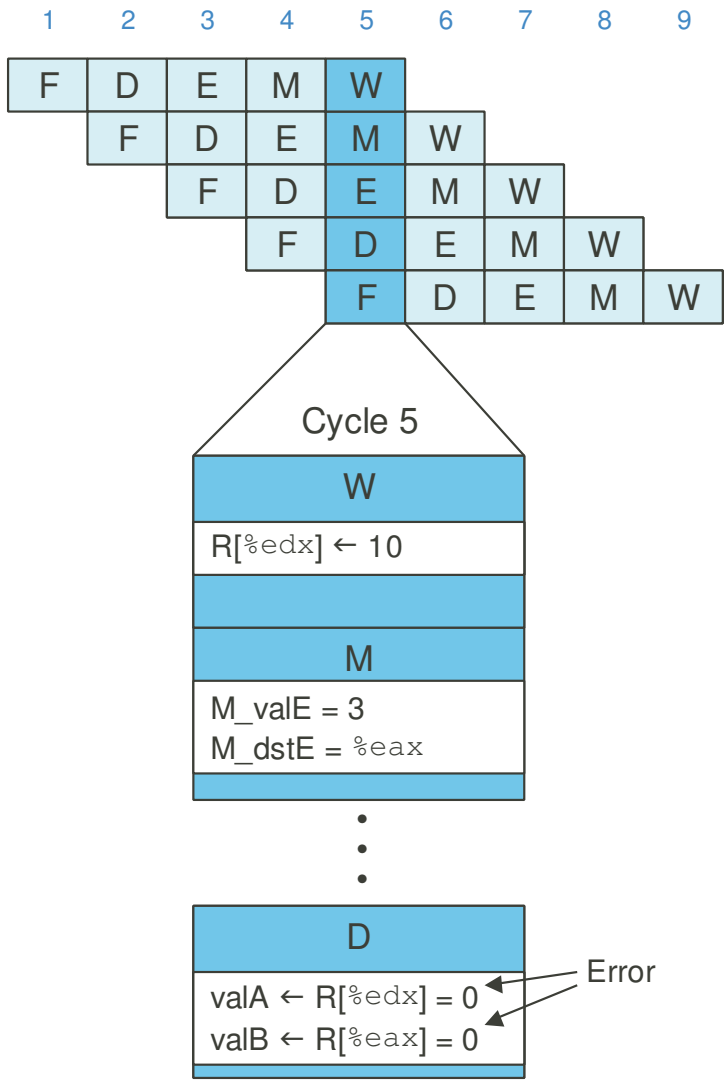
Data Dependencies: 2 Nop's

```
# prog2
0x000: irmovl $10,%edx
0x006: irmovl  $3,%eax
0x00c: nop
0x00d: nop
0x00e: addl %edx,%eax
0x010: halt
```



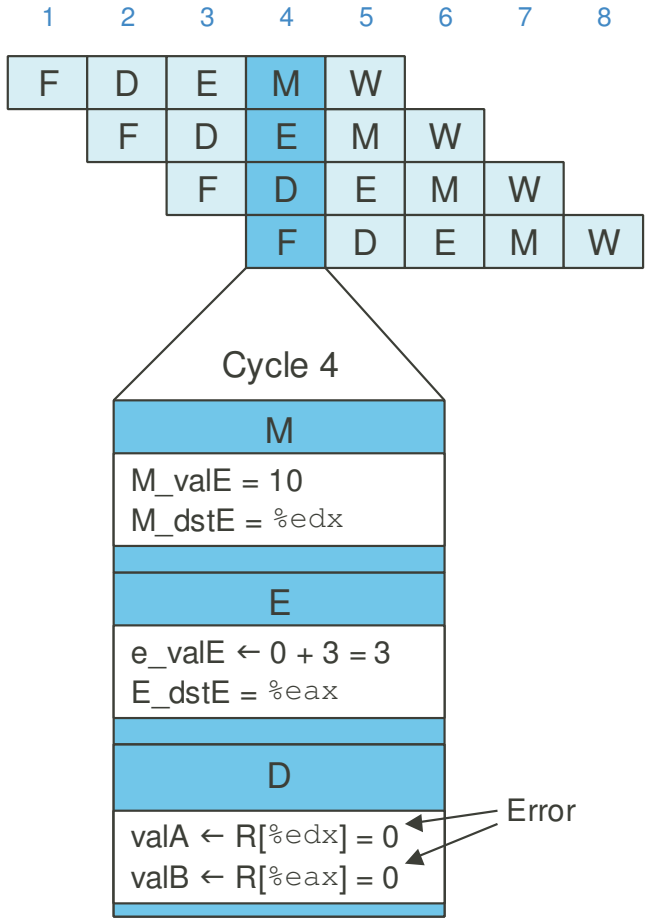
Data Dependencies: 1 Nop

```
# prog3
0x000: irmovl $10,%edx
0x006: irmovl  $3,%eax
0x00c: nop
0x00d: addl %edx,%eax
0x00f: halt
```



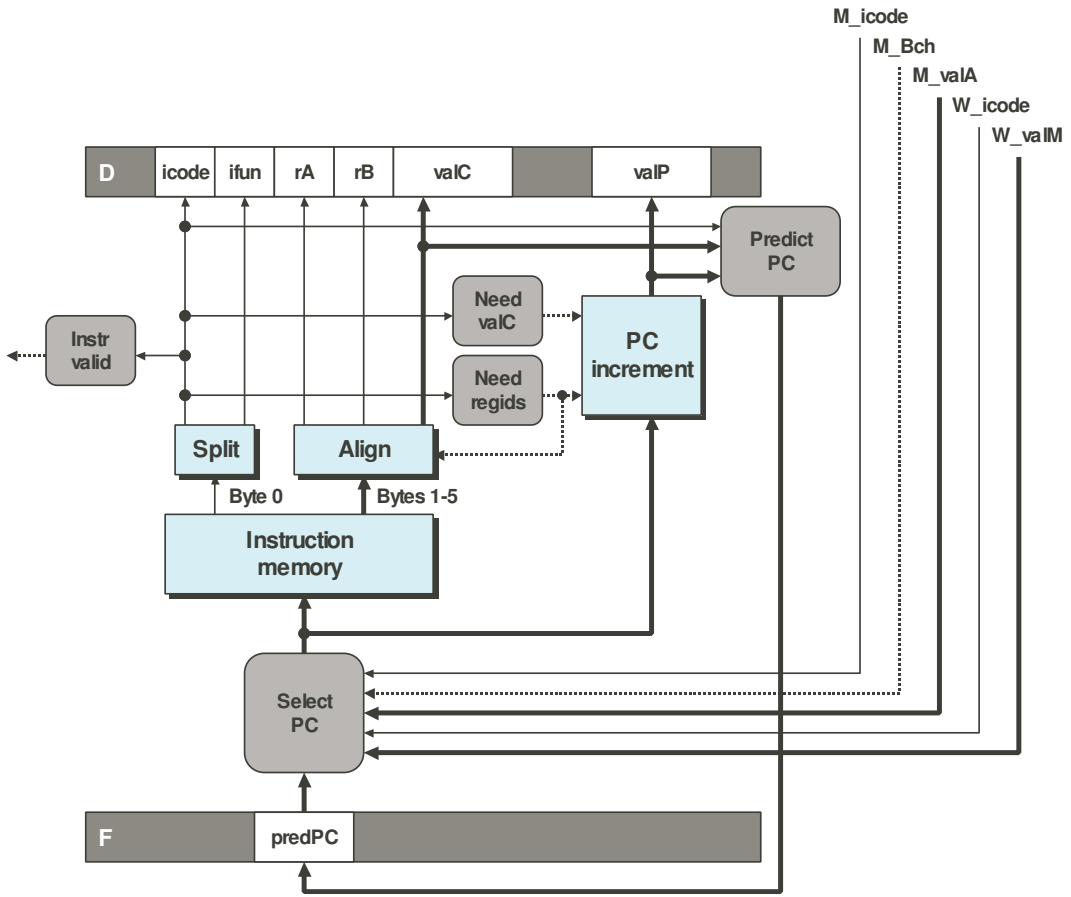
Data Dependencies: No Nop's

```
# prog4
0x000: irmovl $10,%edx
0x006: irmovl  $3,%eax
0x00c: addl %edx,%eax
0x00e: halt
```



Predicting the PC

- Start fetch of a new instruction after the current one has completed the fetch stage.
- There's not enough time to reliably determine the next instruction.
- Guess which instruction will follow.
- Then, recover if the prediction was incorrect.



Our Prediction Strategy

- **Instructions that don't transfer control:**
  - Predict next PC to be valPC.
  - This is always reliable.
- **Call and Unconditional Jumps:**
  - Predict next PC to be valPC (destination).
  - This is always reliable.
- **Conditional Jumps:**
  - Predict next PC to be valPC (destination).
  - Only correct if the branch is taken; right about 60% of the time.
- **Return Instruction:**
  - Don't try to predict.

Recovering from PC Misprediction

Mispredicted Jump:

- Will see branch flag once instruction reaches memory stage.
- Can get fall-through PC from `valA`.

Return Instruction:

- Will get return PC when `ret` reaches write-back stage.

In both cases:

- Need to throw away instructions fetched between prediction and resolution.