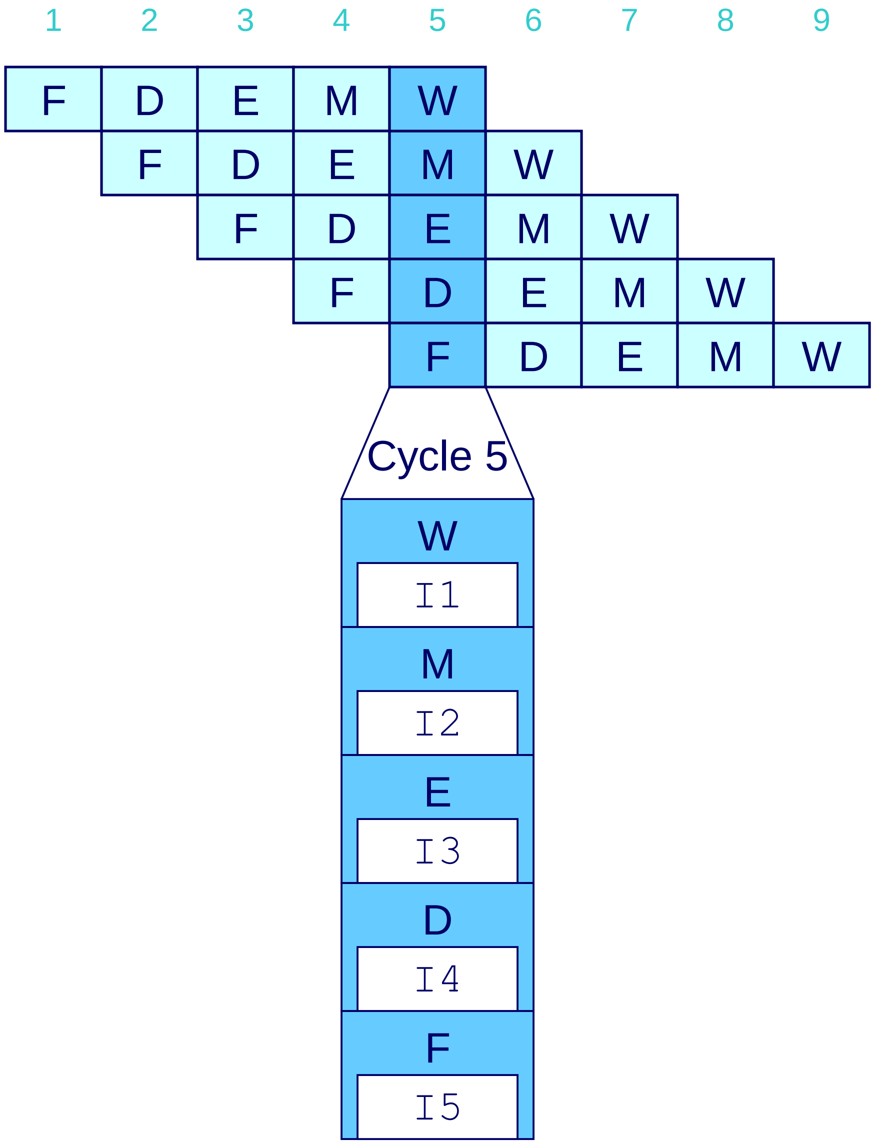
# Pipeline Demonstration

```
irmovq   $1,%rax   #I1
irmovq   $2,%rcx   #I2
irmovq   $3,%rdx   #I3
irmovq   $4,%rbx   #I4
halt               #I5
```



| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|
| F | D | E | M | W | | | | |
| | F | D | E | M | W | | | |
| | | F | D | E | M | W | | |
| | | | F | D | E | M | W | |
| | | | | F | D | E | M | W |

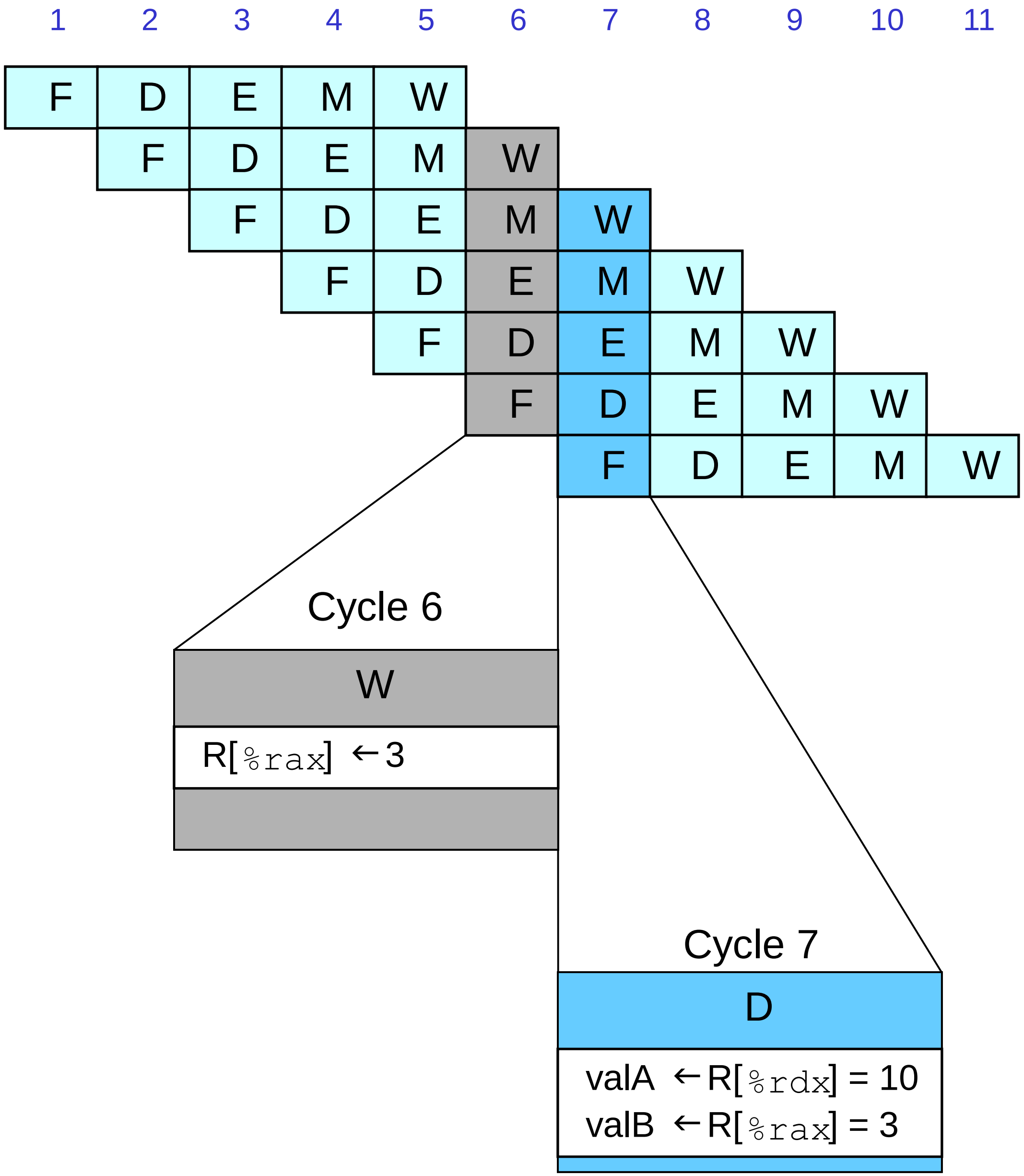
Cycle 5

| W | M | E | D | F |
|---|---|---|---|---|
| I1 | I2 | I3 | I4 | I5 |

**File:** **`demo-basic.ys`**

# Data Dependencies: 3 Nop's

```
# demo-h3.ys
0x000: irmovq $10,%rdx
0x00a: irmovq  $3,%rax
0x014: nop
0x015: nop
0x016: nop
0x017: addq %rdx,%rax
0x019: halt
```



| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|
| F | D | E | M | W | | | | | | |
| | F | D | E | M | W | | | | | |
| | | F | D | E | M | W | | | | |
| | | | F | D | E | M | W | | | |
| | | | | F | D | E | M | W | | |
| | | | | | F | D | E | M | W | |
| | | | | | | F | D | E | M | W |

Cycle 6

W

R[%rax] ← 3

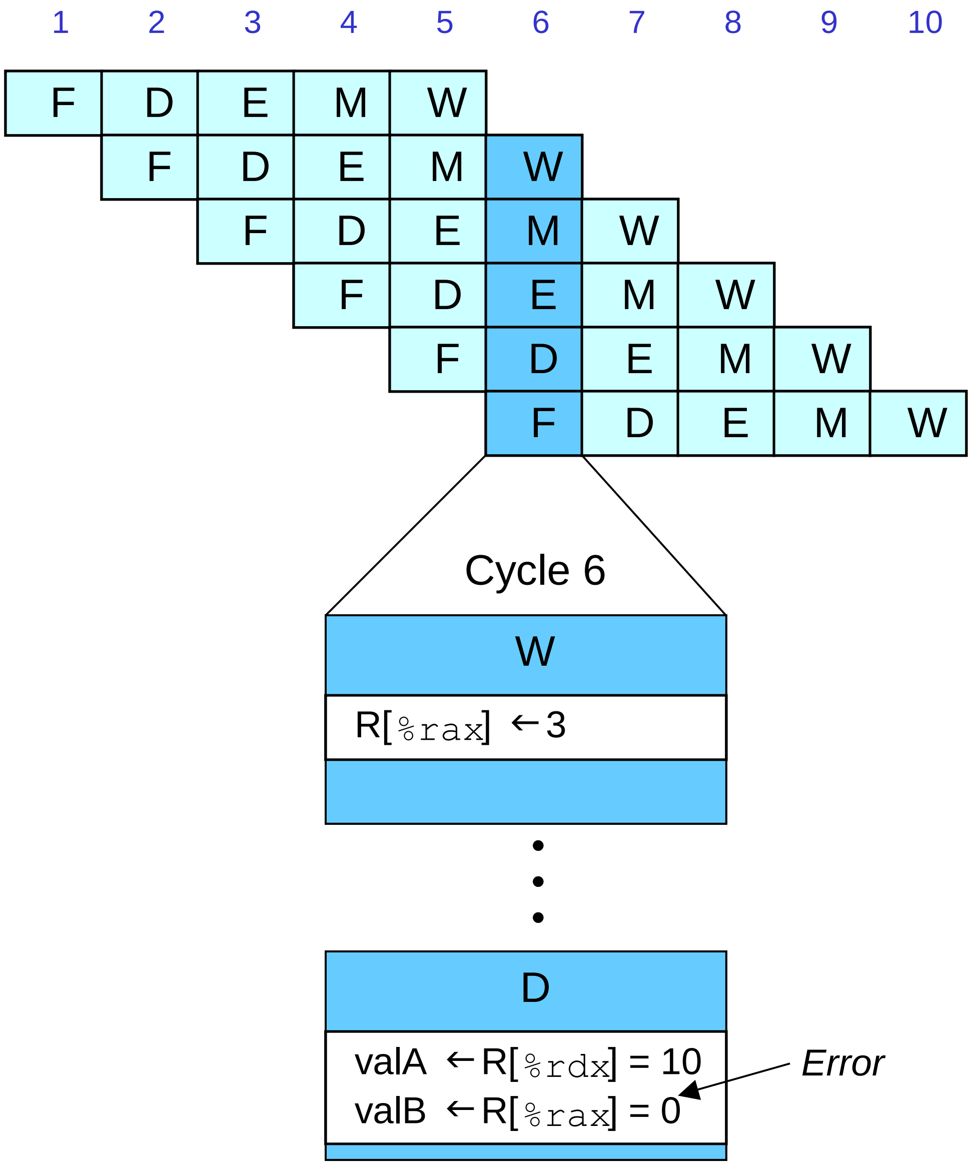
Cycle 7

D

valA ← R[%rdx] = 10
valB ← R[%rax] = 3

# Data Dependencies: 2 Nop's

```
# demo-h2.ys
0x000: irmovq $10,%rdx
0x00a: irmovq  $3,%rax
0x014: nop
0x015: nop
0x016: addq %rdx,%rax
0x018: halt
```



| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| F | D | E | M | W | | | | | |
| | F | D | E | M | W | | | | |
| | | F | D | E | M | W | | | |
| | | | F | D | E | M | W | | |
| | | | | F | D | E | M | W | |
| | | | | | F | D | E | M | W |

Cycle 6

W

R[%rax] ← 3

•
•
•

D

valA ← R[%rdx] = 10

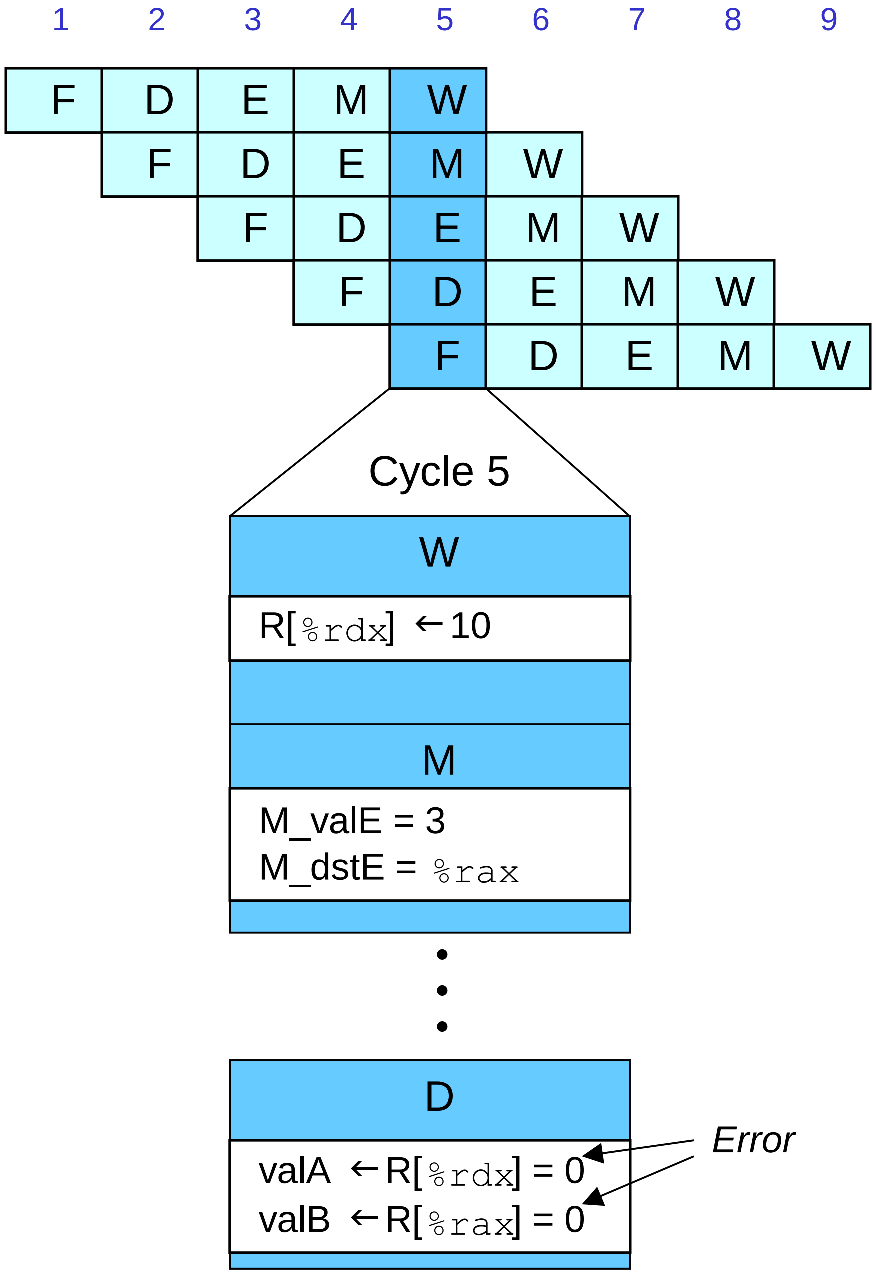
valB ← R[%rax] = 0

*Error*

# Data Dependencies: 1 Nop

```
# demo-h1.ys

0x000: irmovq $10,%rdx
0x00a: irmovq  $3,%rax
0x014: nop
0x015: addq %rdx,%rax
0x017: halt
```



| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|
| F | D | E | M | W | | | | |
| | F | D | E | M | W | | | |
| | | F | D | E | M | W | | |
| | | | F | D | E | M | W | |
| | | | | F | D | E | M | W |

Cycle 5

W

R[%rdx] ← 10

M

M_valE = 3
M_dstE = %rax

•
•
•

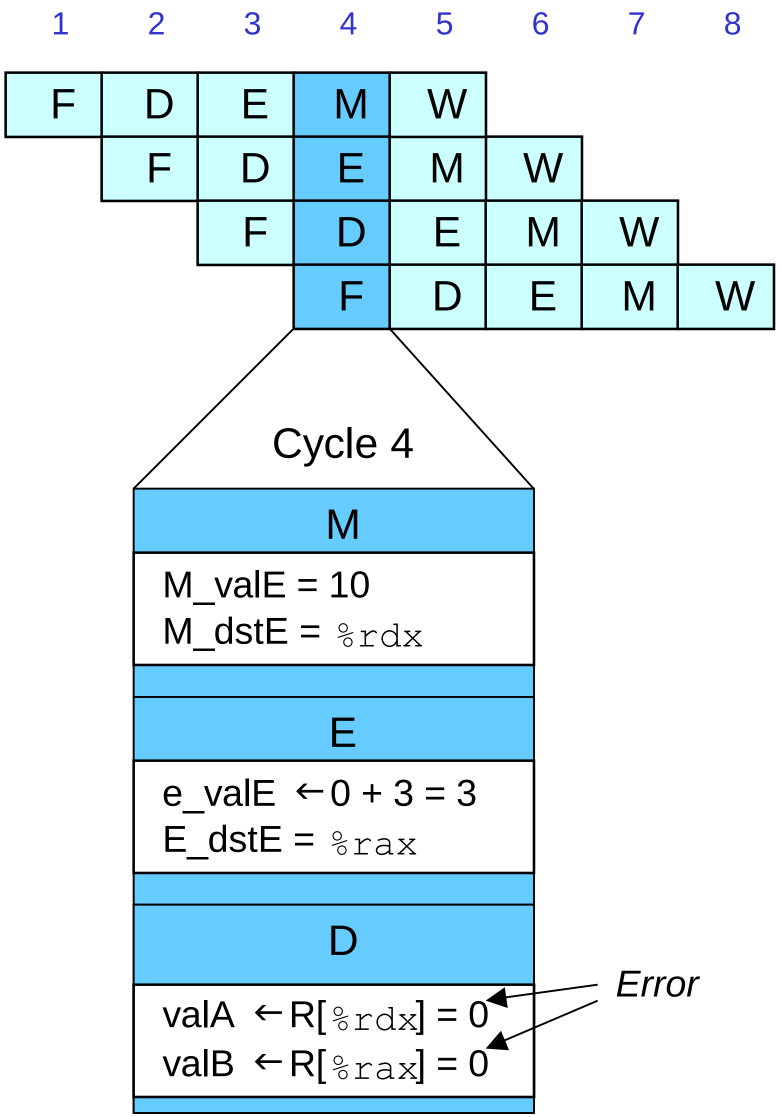
D

valA ← R[%rdx] = 0
valB ← R[%rax] = 0

*Error*

# Data Dependencies: No Nop

```
# demo-h0.ys
0x000: irmovq $10,%rdx
0x00a: irmovq  $3,%rax
0x014: addq %rdx,%rax
0x016: halt
```



| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| F | D | E | M | W | | | |
| | F | D | E | M | W | | |
| | | F | D | E | M | W | |
| | | | F | D | E | M | W |

Cycle 4

M

M_valE = 10
M_dstE = %rdx

E

e_valE ← 0 + 3 = 3
E_dstE = %rax

D

valA ← R[%rdx] = 0
valB ← R[%rax] = 0

*Error*

# Branch Misprediction Example

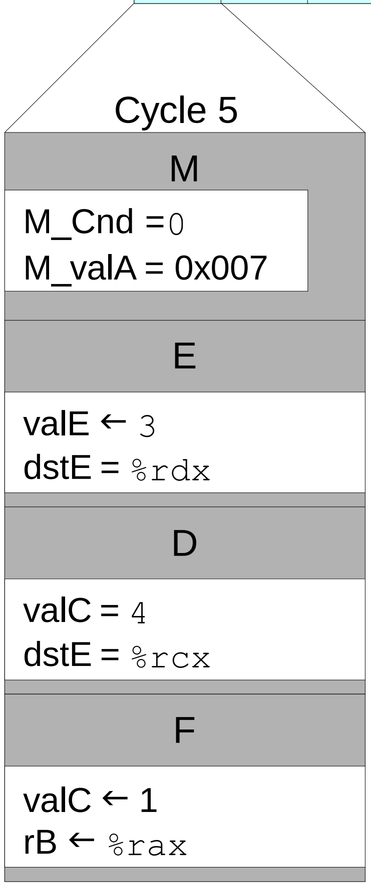
```
demo-j.ys
```

```
0x000:    xorq %rax,%rax
0x002:    jne  t              # Not taken
0x00b:    irmovq $1, %rax     # Fall through
0x015:    nop
0x016:    nop
0x017:    nop
0x018:    halt
0x019: t: irmovq $3, %rdx     # Target (Should not execute)
0x023:    irmovq $4, %rcx     # Should not execute
0x02d:    irmovq $5, %rdx     # Should not execute
```

- Should only execute first 7 instructions

# Branch Misprediction Trace

| # demo-j | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| `0x000:    xorq %rax,%rax` | F | D | E | M | W | | | | |
| `0x002:    jne t # Not taken` | | F | D | E | M | W | | | |
| `0x019: t: irmovq $3, %rdx # Target` | | | F | D | E | M | W | | |
| `0x023:    irmovq $4, %rcx # Target+1` | | | | F | D | E | M | W | |
| `0x00b:    irmovq $1, %rax # Fall Through` | | | | | F | D | E | M | W |


Cycle 5

M

M_Cnd = 0<br>
M_valA = 0x007

E

valE ← 3<br>
dstE = %rdx

D

valC = 4<br>
dstE = %rcx

F

valC ← 1<br>
rB ← %rax

- **Incorrectly execute 3 instructions at branch target**

# Return Example

demo-ret.ys

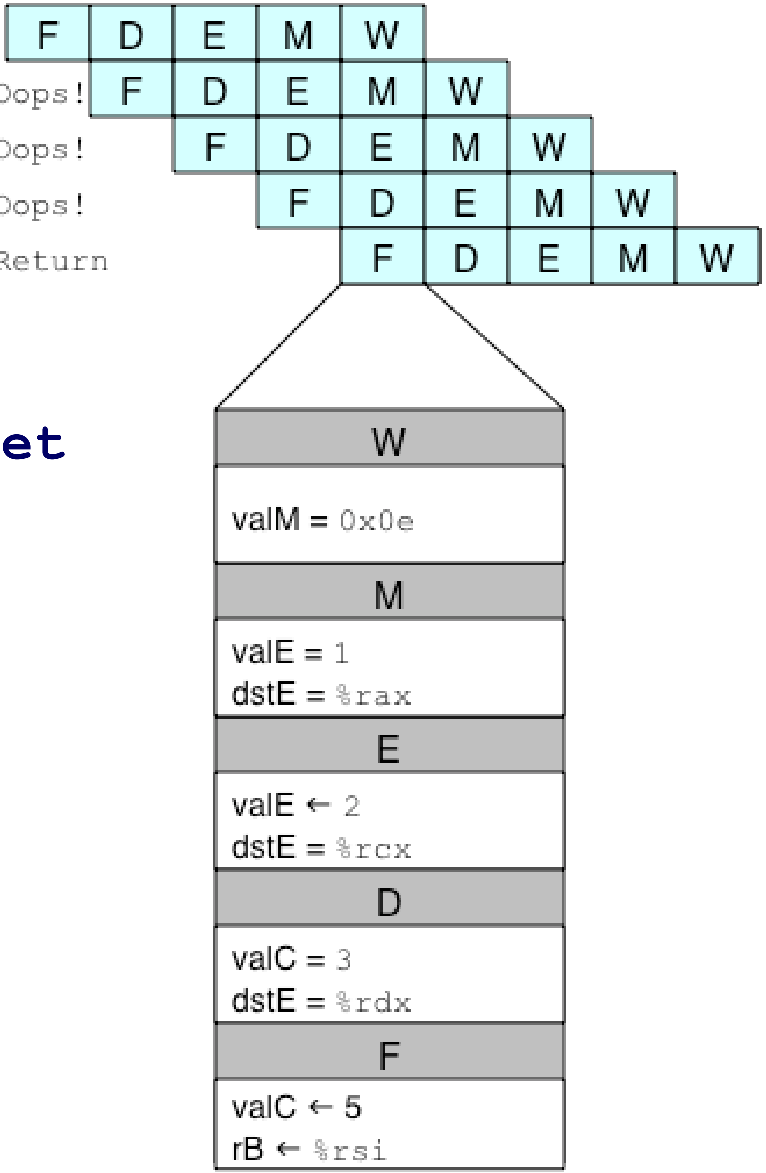
```
 0x000:    irmovq Stack,%rsp  # Intialize stack pointer
0x00a:    nop                 # Avoid hazard on %rsp
0x00b:    nop
0x00c:    nop
0x00d:    call p              # Procedure call
0x016:    irmovq $5,%rsi      # Return point
0x020:    halt
0x020: .pos 0x20
0x020: p: nop                  # procedure
0x021:    nop
0x022:    nop
0x023:    ret
0x024:    irmovq $1,%rax       # Should not be executed
0x02e:    irmovq $2,%rcx       # Should not be executed
0x038:    irmovq $3,%rdx       # Should not be executed
0x042:    irmovq $4,%rbx       # Should not be executed
0x100: .pos 0x100
0x100: Stack:                  # Initial stack pointer
```

- Require lots of nops to avoid data hazards

# Incorrect Return Example

```
# demo-ret

0x033:     ret
0x034:     irmovq $1,%rax # Oops!
0x03e:     irmovq $2,%rcx # Oops!
0x048:     irmovq $3,%rdx # Oops!
0x052:     irmovq $5,%rsi # Return
```



W

valM = 0x0e

M

valE = 1
dstE = %rax

E

valE ← 2
dstE = %rcx

D

valC = 3
dstE = %rdx

F

valC ← 5
rB ← %rsi

- Incorrectly execute 3 instructions following `ret`