Exceptions & Pipelining

Exceptions affect the pipeline like a function call
- control is diverted to the exception handler
- interrupts are a particular type of exception

Exceptions are triggered by unexpected events
- detecting invalid instructions during decode
- arithmetic overflow
- memory errors
- syscall
- interrupts from external I/O controllers

Exceptions & Pipelining

System Coprocessor (CP0) manages MIPS exceptions
- EPC is the exception program counter (CP0 $14)
- Holds address of offending (or interrupted) instruction

Cause register (CP0 $13) indicates problem

Control is transferred to handler at 0x800000018
- handler takes the appropriate action
- resumes program using address in EPC
- terminates program if problem cannot be resolved

Exceptions & Pipelining

Transfer to exception handler causes a control hazard
  program instructions are flushed from pipeline
  this creates pipeline bubbles

The effect on pipeline is similar to a mispredicted branch

Exception Example

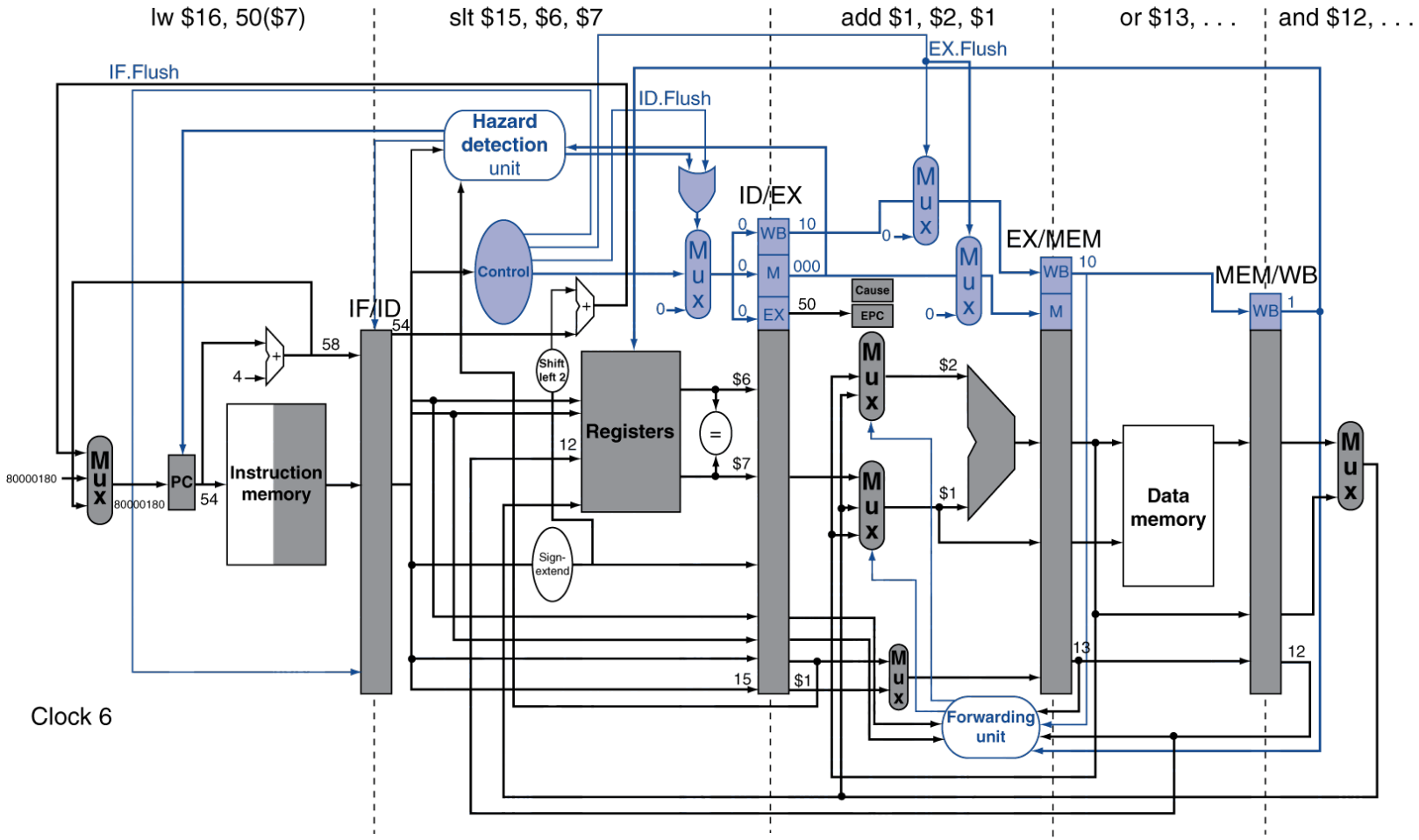
- Exception on add in

```
40      sub   $11, $2, $4
44      and   $12, $2, $5
48      or    $13, $2, $6
4C      add   $1,  $2, $1
50      slt   $15, $6, $7
54      lw    $16, 50($7)
…
```

- Handler

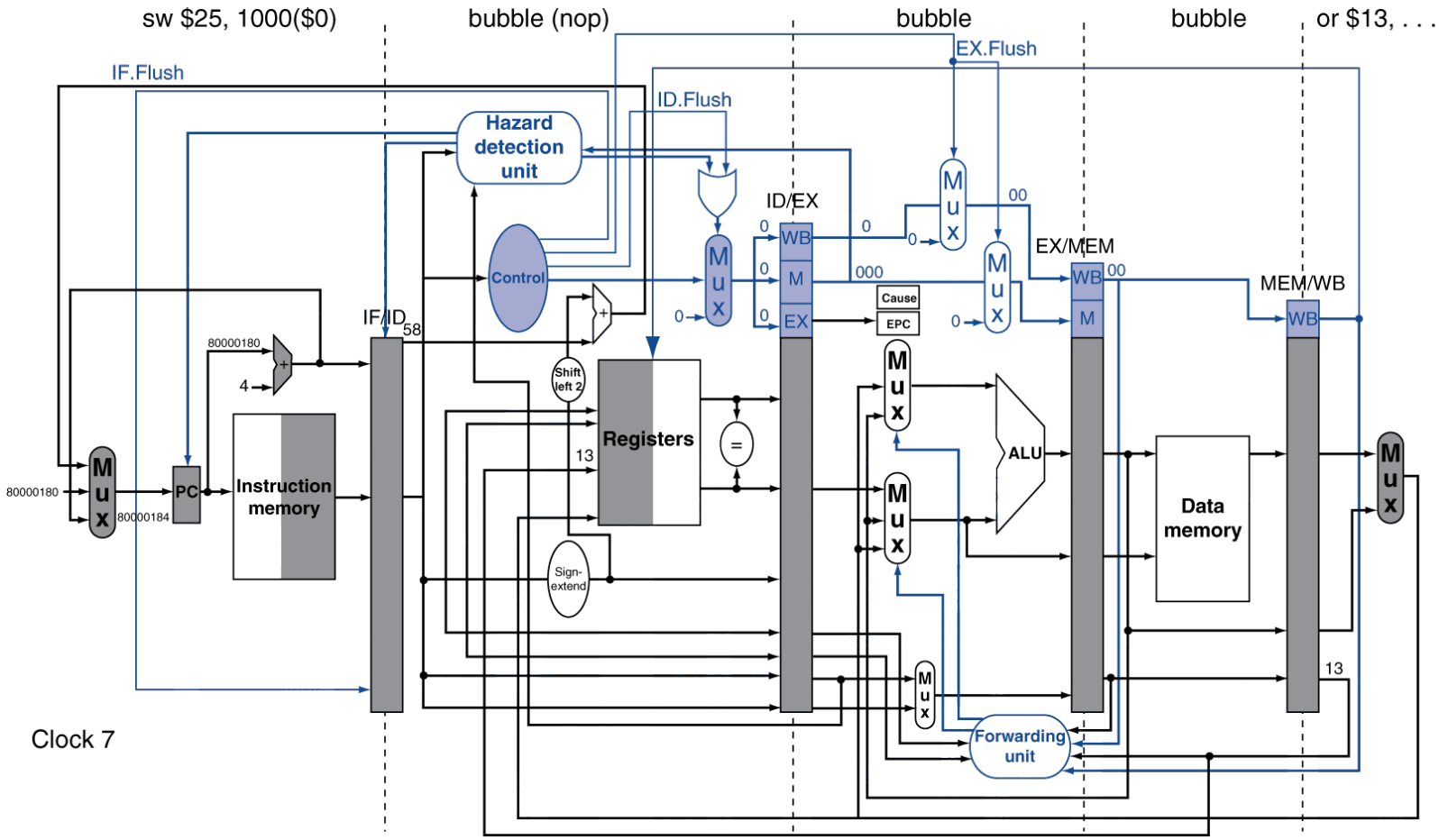
```
80000180    sw    $25, 1000($0)
80000184    sw    $26, 1004($0)
…
```

Exception Example



Flush control signal inserts bubbles

Exception Example



Flush control signal inserts bubbles

Precise vs Imprecise Exceptions

Some systems support precise exceptions
- no register writes after the offending instruction
- leaves the system in a consistent state

Imprecise exceptions
- instruction causing exception is identified
- but succeeding instructions may complete
- allows the system to be in an inconsistent state