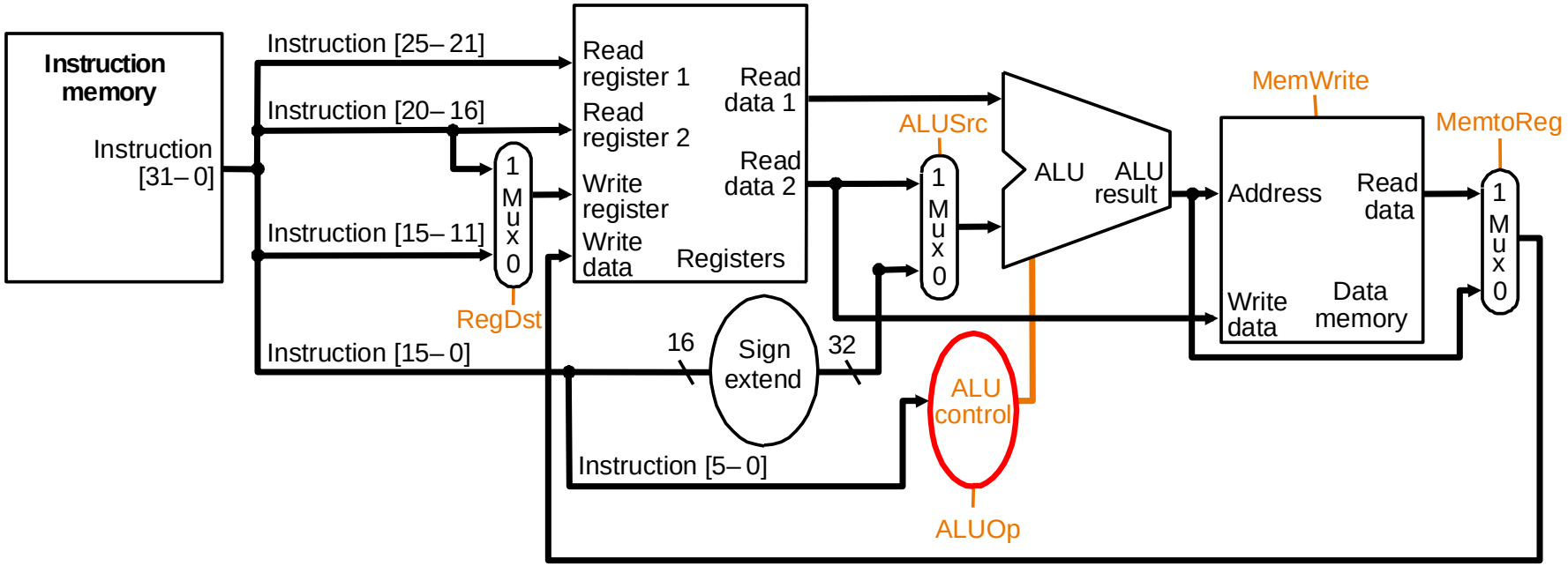
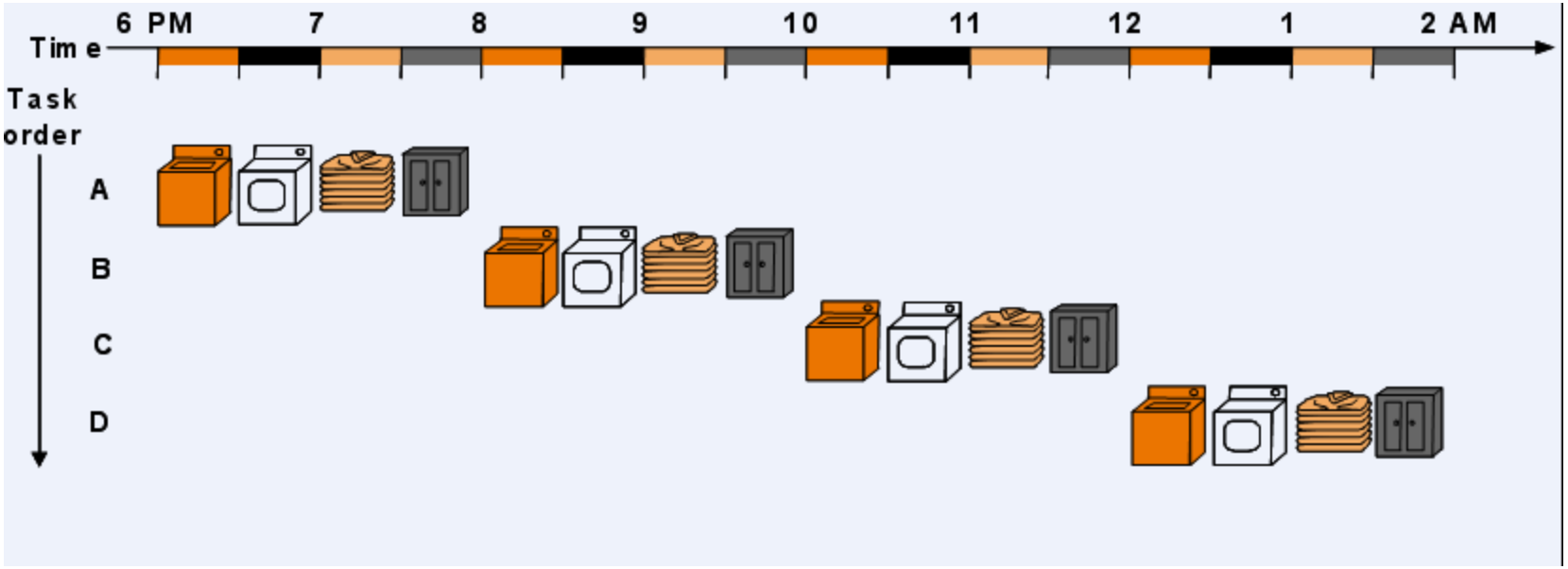
CS 220

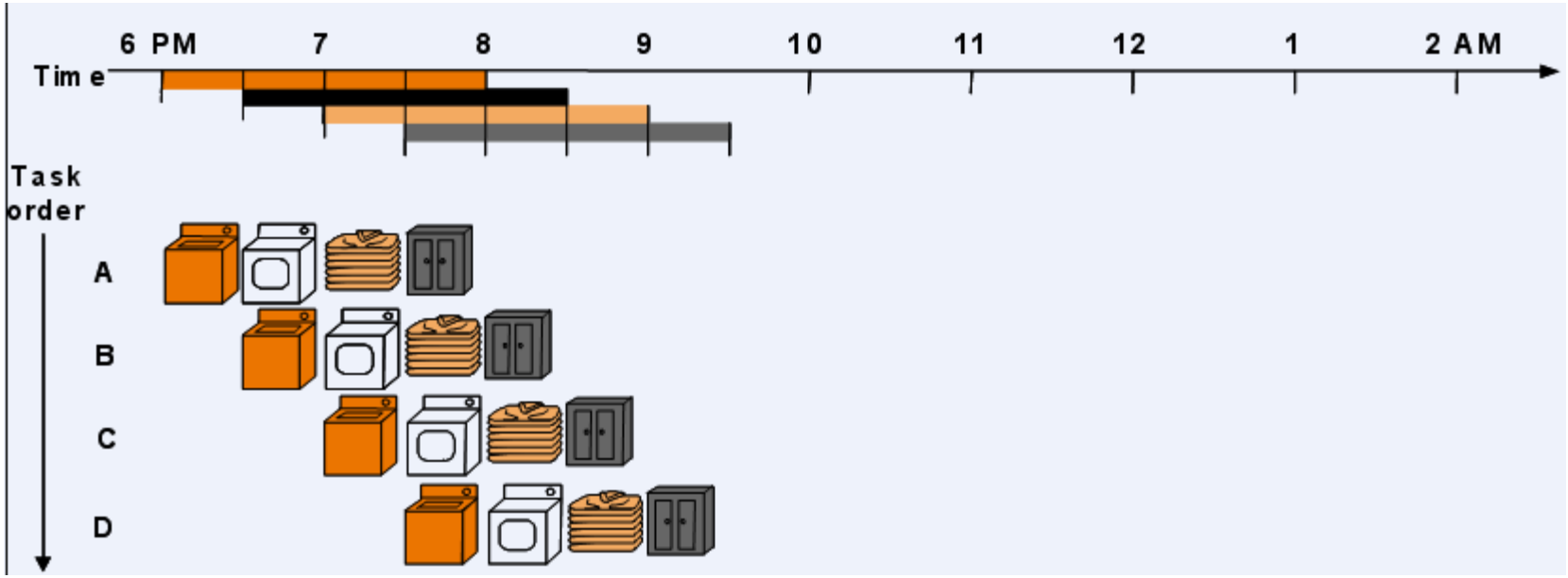
Instruction Pipelines

Single Cycle MIPS Implementation



CPU Time = _____


6 PM
7
8
9
10
11
12
1
2 AM
Time
Task order
A
B
C
D

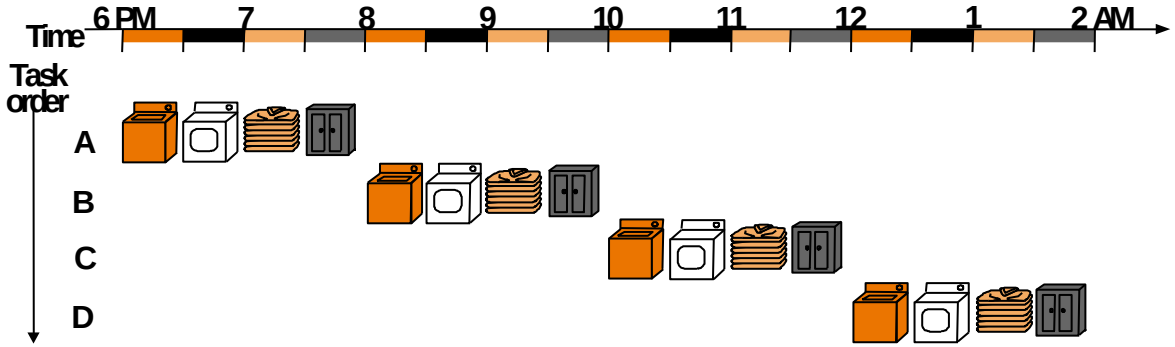

6 PM
7
8
9
10
11
12
1
2 AM
Time
Task order
A
B
C
D

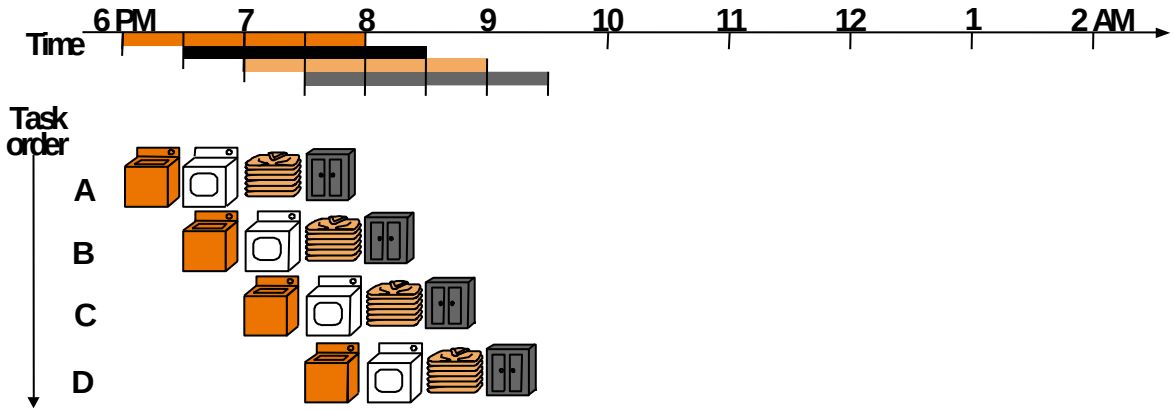
Pipelining speedup

- After the pipeline is full
  - ➢ washer, dryer, folder, put-away are all doing something
  - ➢ A load of clothes is being finished every time unit.
- assume there are k stages in the pipeline
  - ➢ How long does first load take?
  - ➢ How about next n – 1 loads?
  - ➢ total time is _____

Pipelining Speedup

- Sequential washing takes nk time units
  - Speedup = (time before improve/time after improve)
  - Speedup = _____
- In our example k is 4 and n is 4
  - Speedup is _____
- What if the number of loads gets large
  - n = 100
  - Speedup is _____
  - As n gets large speedup goes towards _____





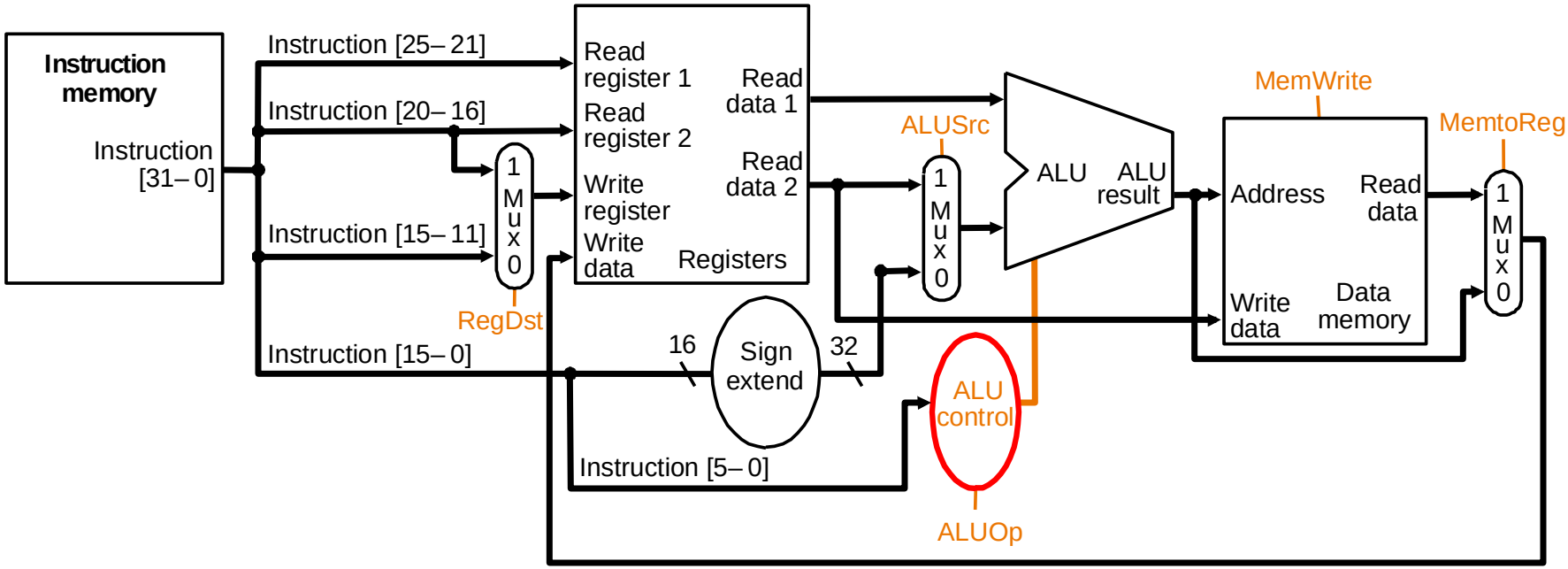
Is this realistic?

Does folding take as long as drying?

Lets assume drying takes ½ as long as washing.

How can we keep the dryer busy?

Pipelining MIPS instructions



What steps are needed to execute a complete instruction?

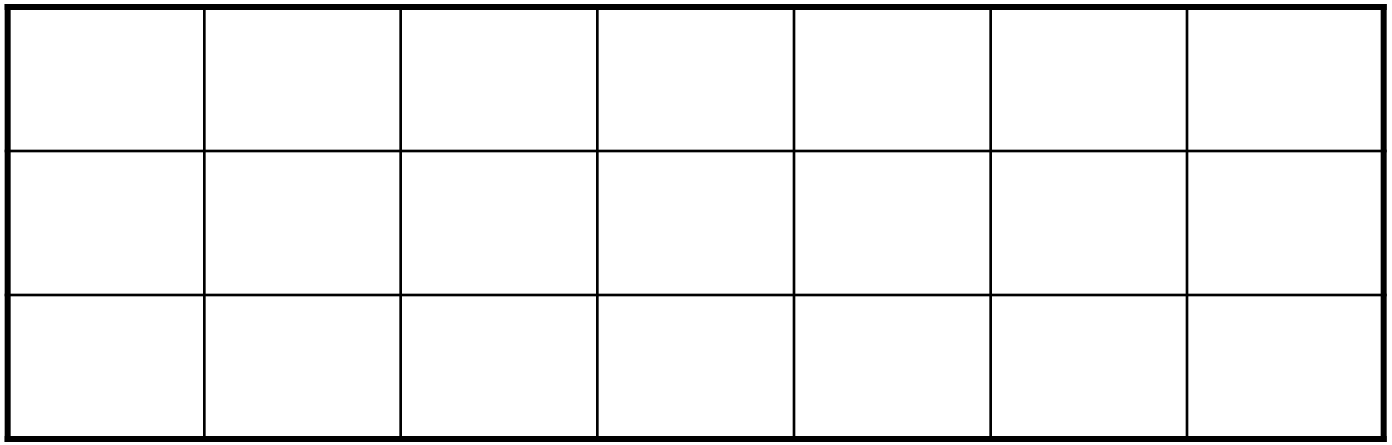
Stages of Instruction Execution

- How many cycles does an integer ALU instruction take to execute?
- Visualizing pipelined instruction execution
- Space/time diagrams

```
add t0,t1,t2

slti s0,s1,9

mul  t4,t5
```



Its not as easy as it sounds

- Hazards – instruction conflicts
  - Structural Hazard
  - Control hazard
  - Data Hazard

Structural Hazards

```
add $s0, $s1, $s2
beq $s3, $zero, offset
addi $t0, $t1, -1
```

Imagine that we didn't have separate adders for updating the PC but used the ALU instead.

add							
beq							
addi							

How can we fix a structural hazard?

Control Hazard

```
loop:
     addi $t0, $t0, -1
     add $s0, $s1, $s2
     bne $t4, $zero, loop
     ori $v0, $zero, 9
```


Data Hazard

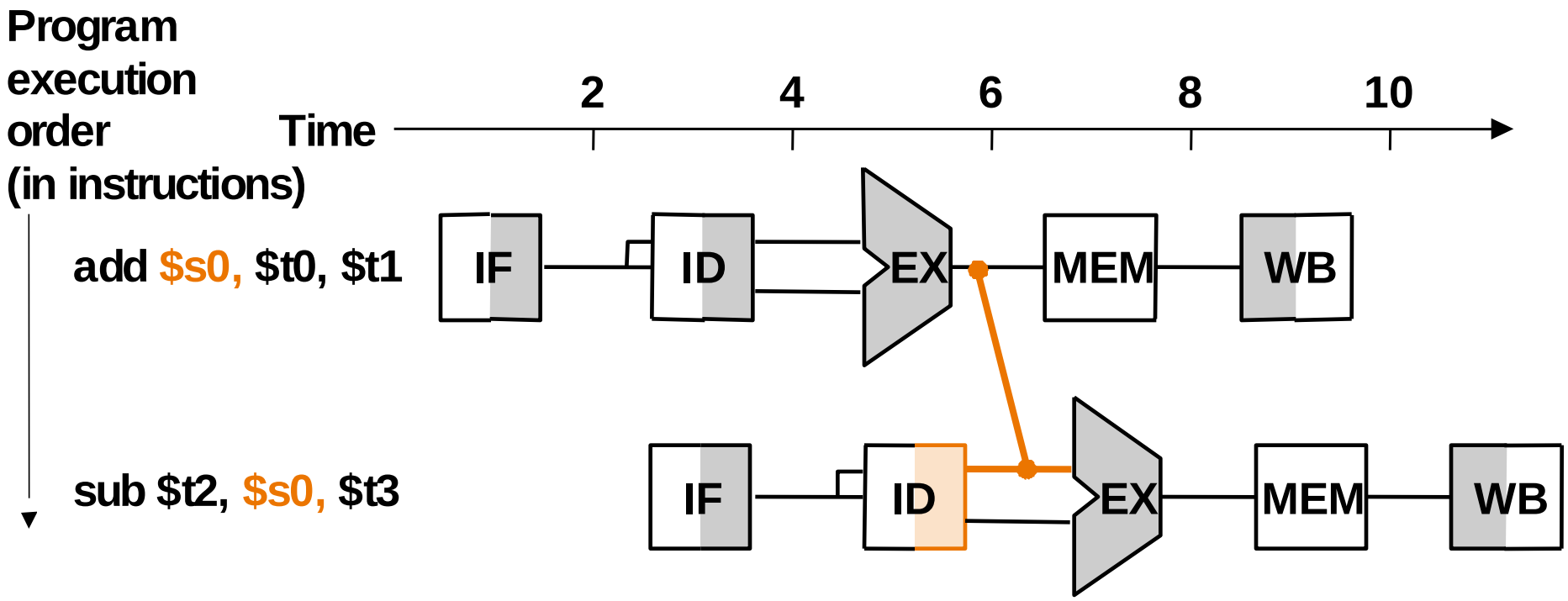
```
add $s0, $s1, $s2

addi $s3, $s0, 13
```


Solution?

Data Hazard

- With a little extra hardware we can ***forward*** the output of the ALU to the ID stage.



Pipelined Datapath