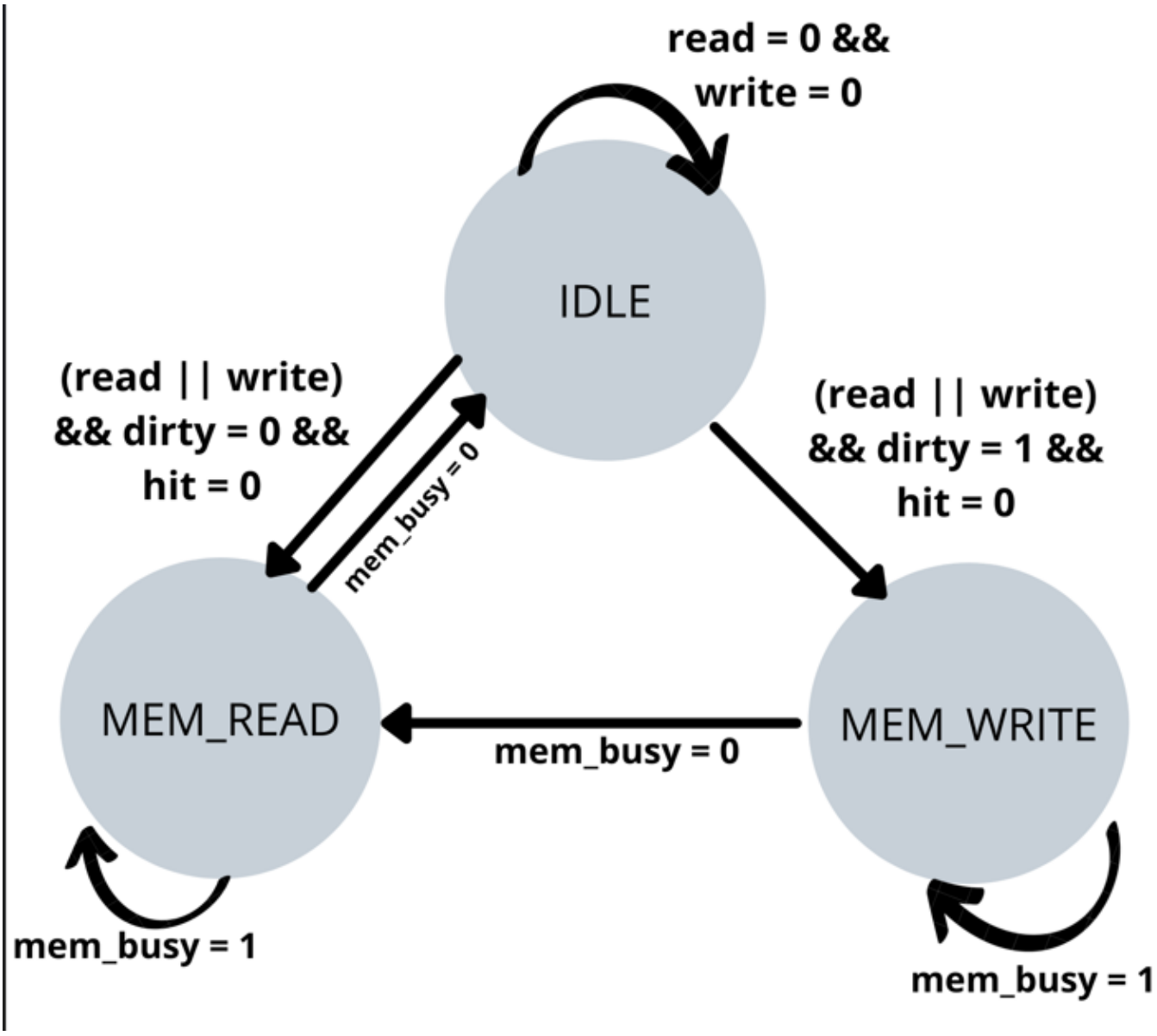
CO224

LAB 6 – Part 2

Group 06

Finite State Machine



Addressing

bits in the address coming to the cache = 8 bits

blocks in the cache = 8

= 2^3

bits for indexing = 3

bits used for the offset = 2 (As one block in cache stores 1 word = 2^2 bytes)

bits for the tag = 8 – 3 – 2

= 3

Ex: (Tag, Index, Offset)

Incoming address: 00110011

The format of a block in cache

Index	Valid	Dirty	Tag	Data

The sample code used to test the program

sample_program.s

```
1    loadi 0 0x09
2    loadi 1 0x01
3    swd 0 1
4    swi 1 0x00
5    lwd 2 1
6    lwd 3 1
7    sub 4 0 1
8    swi 4 0x07
9    lwi 5 0x07
10   lwi 6 0x20
11   swi 4 0x20
```

Determining hits/ misses:

Index	Valid	Dirty	Tag	Data	Status
000	~~0~~ ~~1~~ ~~1~~ 1	~~0~~ ~~1~~ ~~1~~ 1	~~X~~ ~~000~~ 001	~~X~~ ~~9_{10}~~ ~~1_{10}~~ 8_{10}	~~Cold miss (3)~~ ~~Hit (4)~~ Conflict miss (11)
001	~~0~~ 1	~~0~~ 1	~~X~~ 000	~~X~~ 8_{10}	Cold miss (8)
010	0	0			
011	0	0			
100	0	0			
101					
110	0	0			
111	0	0			

Hit/ miss status for loads

Line 5 & 6 are **hits.**

Line 9 is a **hit.**

Line 10 is a **miss** -> Read data from main memory (value is 0_{10})

Compare Cache vs Cache-less CPU

Cache

CHANGE OF REGISTER CONTENT STARTING FROM TIME #5

time	reg0	reg1	reg2	reg3	reg4	reg5	reg6	reg7
5	0	0	0	0	0	0	0	0
13	9	0	0	0	0	0	0	0
21	9	1	0	0	0	0	0	0
221	9	1	9	0	0	0	0	0
229	9	1	9	9	0	0	0	0
237	9	1	9	9	8	0	0	0
429	9	1	9	9	8	8	0	0
781	9	1	9	9	8	8	0	0

Cache-less

time	reg0	reg1	reg2	reg3	reg4	reg5	reg6	reg7
5	0	0	0	0	0	0	0	0
13	9	0	0	0	0	0	0	0
21	9	1	0	0	0	0	0	0
165	9	1	9	0	0	0	0	0
213	9	1	9	9	0	0	0	0
221	9	1	9	9	8	0	0	0
317	9	1	9	9	8	8	0	0
365	9	1	9	9	8	8	0	0

1. Let’s compare the first ldr instruction.

   The cache serves one miss before the ldr instruction. This write miss penalty (21 cycles) is large compared to reading directly from the memory in the cache-less CPU (5 cycles)

2. The second load word

   For the CPU without the cache this ldr instruction can be done without stalling the CPU as it is a hit and hence completes in 1 cycle.

   However, the cache-less CPU must load this word directly from the memory and hence takes 5 clock cycles.

   The CPU with a cache works x5 times faster in this case.

3. The load word in lines 9 and 10

   Here, the cache seems to have a higher instruction latency than the cache-less CPU. This is because the cache comes a cross a miss before serving the ldr instructions.

- As we can see, when there is a hit in the cache, the cache works roughly 5 times faster than the cache-less CPU.
- This sample program shows that the overall time taken by the cache-less CPU is less but a larger program (with loops, conditional statements, etc) will have more hits, and thus will be able to depict the true overall performance of the cache.
- A real program adheres to spatial and temporal locality much more and thus will make good use of the cache.