CO224 Computer Architecture
Thilakarathna W M D U
E/16/366

Lab06 - Part 02

Performance comparison report

In part 01 of Lab06, we have to implement a Memory module for the Single-cycle CPU which we implement in Lab05. In that situation every Main memory read or write access requires 5 clock cycles. In that period we have to stall the CPU. that is a huge wastage of resources. So we have to implement a cache memory in Lab06-part 02. After we attached Cache Memory in between CPU and the Main memory. CPU doesn't know there is a Cache in-between.

Although it requires less time to access the cache memory. If we want to access the main memory we have to sacrifice about 20 clock cycles because the main memory and the cache memory deals with data blocks with for data words. Although the cache memory have higher hit rate it also comes with a missed penalty. Not only that but also, if we have a read miss and a dirty value in the current index of the cache. We have to spend about 40 clock cycles because we have to put the dirty value to the CPU in order to fetch data. The efficiency depends on the kind of program we are executing currently in the CPU.

Most of the programs used the same set of memory locations. So using the cache memory is very efficient on that but the below program does not use much memory.

The program I submitted with Lab06 - part 01

```
loadi 0 0xFA
sll 1 0 0x02
srl 1 0 0x02
sra 1 0 0x02
ror 1 0 0x02
loadi 2 0x05
loadi 3 0x0A
mul 4 2 3
loadi 0 0x0A
swi 0 0X03
lwi 1 0X03
loadi 2 0X03
loadi 3 0xAA
swd 3 2
lwd 4 2
```

This program took around 288ns which means about 36 clock cycles to execute. After I execute this program after attaching the cache memory it took around 308ns. When compared with the cache less memory this takes much more time. The reason for this is this program takes more time because it has much less memory-related instructions. So the time consumed for taking a block of memory takes more time than the cache less memory.

So I have written the following program to loop and save data to the main memory about 10 times. Then the cache memory is better than without cache memory.

```
loadi 0 0x01
loadi 1 0x00
loadi 2 0x0A
add 1 1 0
swi 1 0x00
bne 0xFD 1 2
```

This program took about 617ns to fully execute without the cache memory, which is about 78 clock cycles. But after attaching the cache memory to the CPU the program only takes around 481ns, that's about 60 clock cycles. Like that when we consider this with the normal program. Caches can achieve up to a 99% hit rate that means the can serve most of the data without taking them from the main memory.

If a program jumps around more memory addresses then it reduces the efficiency of using cache memory. But that does not happen most of the cases because the OS allocates memory to programs as they get the memory at the nearest locations of the main memory.