Lab 5: Microprocessor Memory Caching 3: The Direct-Mapped & Set-Associative Cache

CME433-01

Addi Amaya

Caa746

11255790

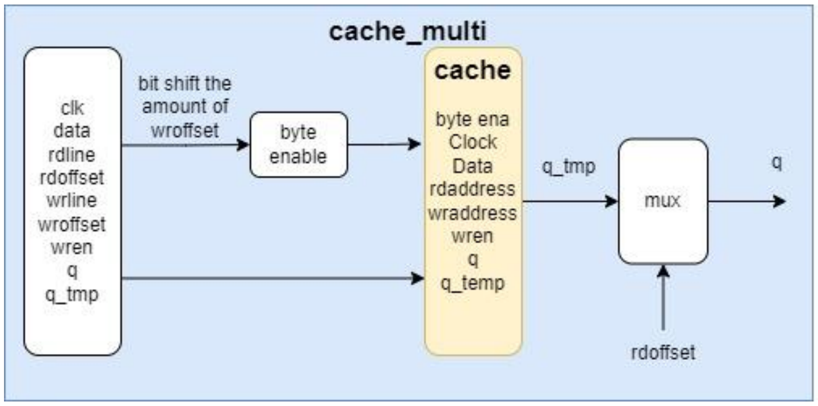
Nov 26th, 2021

Contents

Multi-Line Direct-Mapped Cache

Question 1: Block Diagram

Here you can see that the cache_multi is wrapper for the cache that takes a series of inputs and encodes them to prepare for the cache. The cache output is a 64-bit output which is truncated depending on the input of rdoffset



Question 2: Cache Operation

Direct-mapped cache refers to a piece of memory that has sections and each section is tied to a part of the main memory. The cache takes in address and takes the 3LSB and creates an index and with the 2MSB creates the tag. The index is checked to the address in the cache, if the tag is the same it is hit otherwise it's a miss. Hit referring to that an address already exists in the cache and miss saying the opposite.

Example:

The photos below demonstrate a simple example of how direct-mapping works. Address 10011 is broken into an index and tag. The index would be 011 and the tag is 10. So it checks the index of the initial (row 4) then it checks the tag. Because the initial tag is 11 and the given address tag 10, then it's a miss because they aren't the same. The opposite happens for address 00001. it has an index of 001 and a tag of 00. So it checks index 001 (row 2) then it checks the tag but because the tags are the same it's a hit and thus ignored

Addr	10011	00001
h/m	miss	hit

index	valid	tag	Data
000	n		
001	y	00	M[00001]
010	n		
011	y	11	M[11011]
100	y	10	M[10100]
101	y	01	M[01101]
110	y	00	M[00110]
111	n		

index	valid	tag	Data
000	n		
001	y	00	M[00001]
010	n		
011	y	10	M[10011]
100	y	10	M[10100]
101	y	01	M[01101]
110	y	00	M[00110]
111	n		

Question 3: Comparison

Multi_cache

Total Logic elements: 315

Total Registers: 76

Total Pins: 145

Memory bits: 2368

Single_cache

Total Logic elements: 428

Total Registers: 66

Total Pins: 138

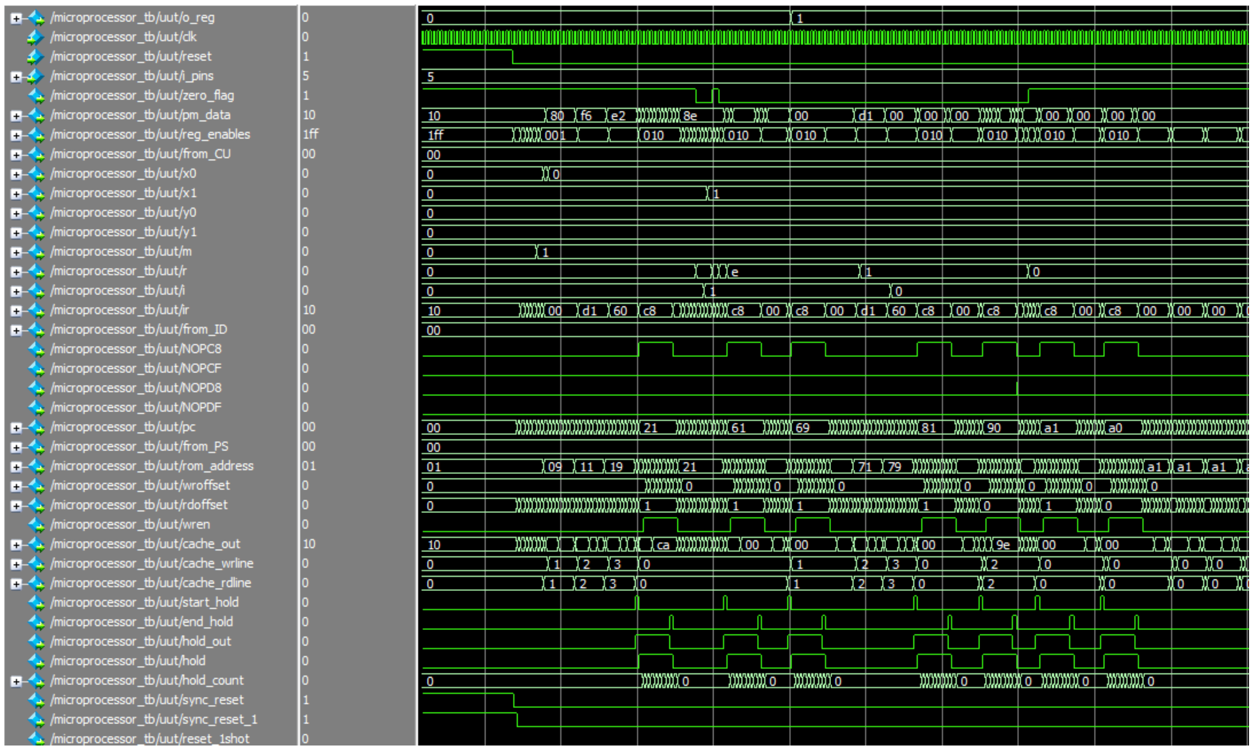
Memory bits: 2624

Observations

You can see above that the single cache has more logic elements, registers and memory bits but it has fewer total pins used. The memory bits for the multi-cache are less meaning it is requiring less memory to function.

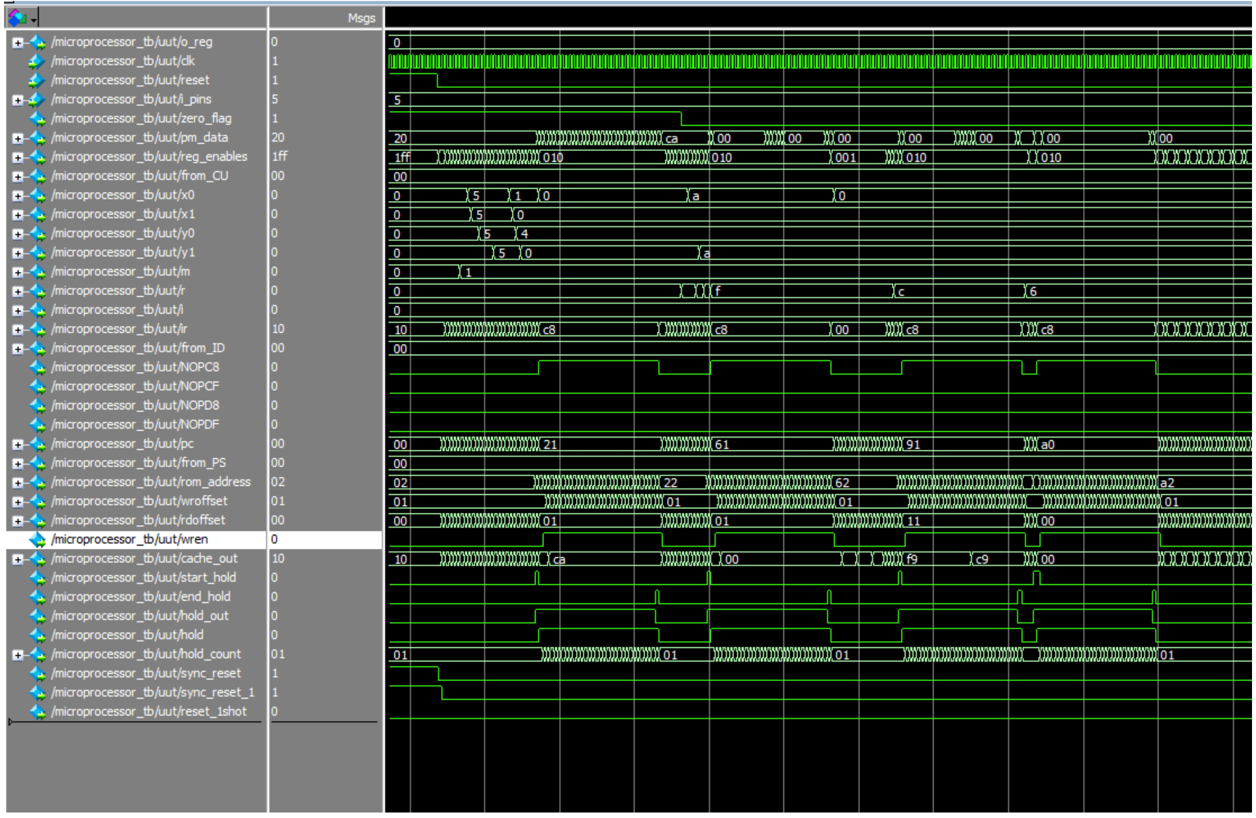
Question 4: End of Execution

Multi-cache



```
/microprocessor_tb/uut/o_reg          0
/microprocessor_tb/uut/clk            0
/microprocessor_tb/uut/reset          1
/microprocessor_tb/uut/i_pins         5
/microprocessor_tb/uut/zero_flag      1
/microprocessor_tb/uut/pm_data        10
/microprocessor_tb/uut/reg_enables    1ff
/microprocessor_tb/uut/from_CU        00
/microprocessor_tb/uut/x0             0
/microprocessor_tb/uut/x1             0
/microprocessor_tb/uut/y0             0
/microprocessor_tb/uut/y1             0
/microprocessor_tb/uut/m              0
/microprocessor_tb/uut/r              0
/microprocessor_tb/uut/i              0
/microprocessor_tb/uut/ir             10
/microprocessor_tb/uut/from_ID        00
/microprocessor_tb/uut/NOPC8          0
/microprocessor_tb/uut/NOPCF          0
/microprocessor_tb/uut/NOPD8          0
/microprocessor_tb/uut/NOPDF          0
/microprocessor_tb/uut/pc             00
/microprocessor_tb/uut/from_PS        00
/microprocessor_tb/uut/rom_address    01
/microprocessor_tb/uut/wroffset       0
/microprocessor_tb/uut/rdoffset       0
/microprocessor_tb/uut/wren           0
/microprocessor_tb/uut/cache_out      10
/microprocessor_tb/uut/cache_wrline   0
/microprocessor_tb/uut/cache_rdline   0
/microprocessor_tb/uut/start_hold     0
/microprocessor_tb/uut/end_hold       0
/microprocessor_tb/uut/hold_out       0
/microprocessor_tb/uut/hold           0
/microprocessor_tb/uut/hold_count     0
/microprocessor_tb/uut/sync_reset     1
/microprocessor_tb/uut/sync_reset_1   1
/microprocessor_tb/uut/reset_1shot    0
```

Single-cache



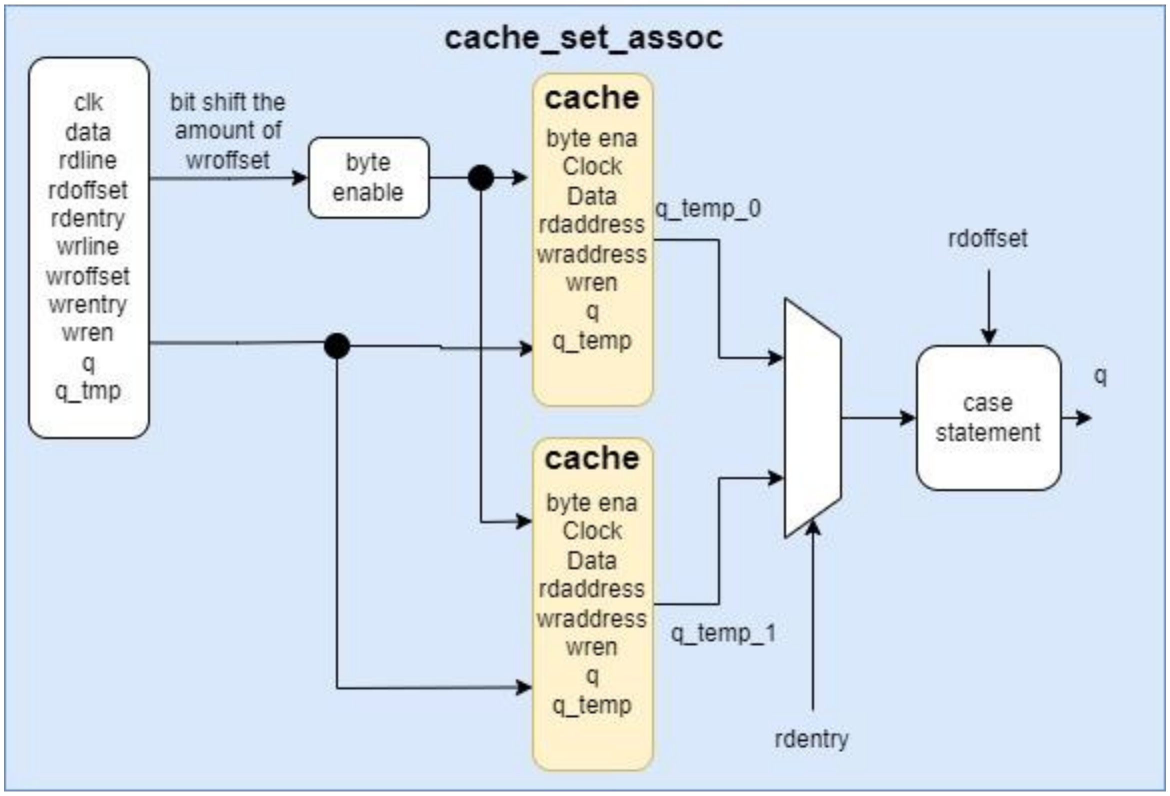
	Msgs
/microprocessor_tb/uut/o_reg	0
/microprocessor_tb/uut/clk	1
/microprocessor_tb/uut/reset	1
/microprocessor_tb/uut/i_pins	5
/microprocessor_tb/uut/zero_flag	1
/microprocessor_tb/uut/pm_data	20
/microprocessor_tb/uut/reg_enables	1ff
/microprocessor_tb/uut/from_CU	00
/microprocessor_tb/uut/x0	0
/microprocessor_tb/uut/x1	0
/microprocessor_tb/uut/y0	0
/microprocessor_tb/uut/y1	0
/microprocessor_tb/uut/m	0
/microprocessor_tb/uut/r	0
/microprocessor_tb/uut/i	0
/microprocessor_tb/uut/ir	10
/microprocessor_tb/uut/from_ID	00
/microprocessor_tb/uut/NOPC8	0
/microprocessor_tb/uut/NOPCF	0
/microprocessor_tb/uut/NOPD8	0
/microprocessor_tb/uut/NOPDF	0
/microprocessor_tb/uut/pc	00
/microprocessor_tb/uut/from_PS	00
/microprocessor_tb/uut/rom_address	02
/microprocessor_tb/uut/wroffset	01
/microprocessor_tb/uut/rdoffset	00
/microprocessor_tb/uut/wren	0
/microprocessor_tb/uut/cache_out	10
/microprocessor_tb/uut/start_hold	0
/microprocessor_tb/uut/end_hold	0
/microprocessor_tb/uut/hold_out	0
/microprocessor_tb/uut/hold	0
/microprocessor_tb/uut/hold_count	01
/microprocessor_tb/uut/sync_reset	1
/microprocessor_tb/uut/sync_reset_1	1
/microprocessor_tb/uut/reset_1shot	0

Observations

There are a few things to note. One is being how long the delays are with respects to eachother. The single cache has a longer delay. Another thing to note is that the pm_data and ir are the same values the only difference, again, is the delay they have on them. One difference between the two is the in the multi-cache there is an oscillation around 0xa1 for the rom address whereas for the single there is a constant 0xa2. Another observation is that even though the delay in the multi-cache is short is it keeping up with the writing and reading

2-way Set Associative Cache

Question 1: Block Diagram



Question 2: Operation

The set associative cache is like direct-mapped but instead it has two caches to represent the same number of spaces in the main memory as direct-mapped. The index selects a set from the cache and there are two tags that are set and compared in parallel, and the data is selected based on the tag result.

Example:

From the example below starting at the first address we can see that the tag of address 10011 is 1001. The set is represented as a 1. So, the tag and set together represent the address. And because the initial state has an memory location open at that address it just fits into the cache. For the next address, 00001, the tag is 0000 and the set is 1, now it can be seen that the address is already taken so it so it's a hit.

Addr	10011	00001
h/m	Miss	hit

a. initial state

Set	Tag	Data	Tag	Data	Tag	Data	Tag	Data
0			0011	M[00110]			1010	M[10100]
1	0000	M[00001]	1101	M[11011]	0110	M[01101]		

b. 10011

Set	Tag	Data	Tag	Data	Tag	Data	Tag	Data
0			0011	M[00110]			1010	M[10100]
1	0000	M[00001]	1101	M[11011]	0110	M[01101]	1001	M[10011]

Question 3: Comparison

Set_assoc_cache

Total Logic elements: 365

Total Registers: 93

Total Pins: 147

Memory bits: 2368

Multi_cache

Total Logic elements: 315

Total Registers: 76

Total Pins: 145

Memory bits: 2368

Single_cache

Total Logic elements: 428

Total Registers: 66

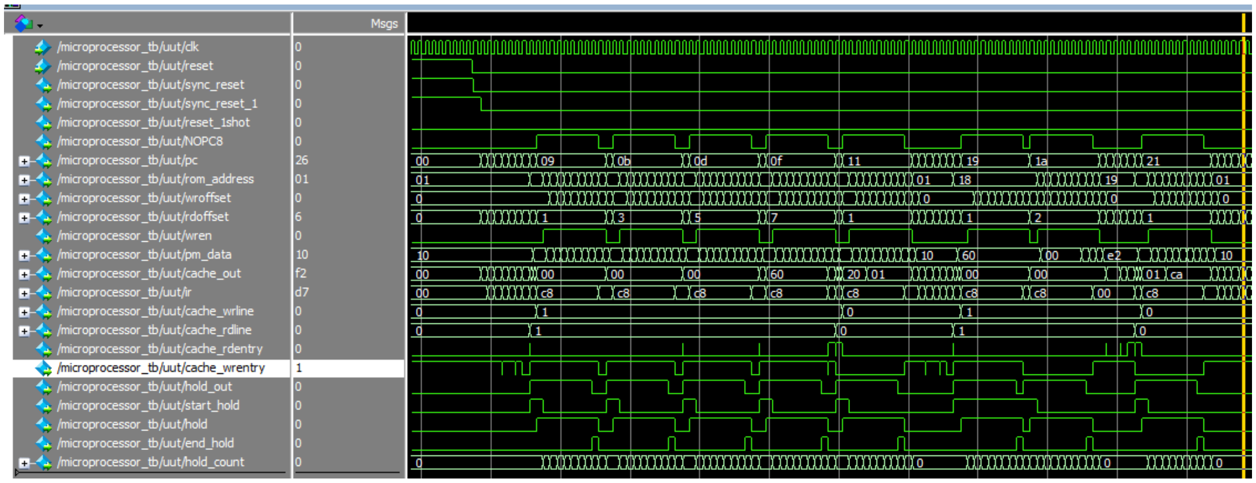
Total Pins: 138

Memory bits: 2624

Observations

The set_assoc_cache has more logic elements than the multi_cache but less than the single. It has more registers than both the mulit and single cache. Final it has the most pins out of all the caches. The memory bits is the same as the multi-cache implementation.

Question 4: End of Execution


Msgs
/microprocessor_tb/uut/clk 0
/microprocessor_tb/uut/reset 0
/microprocessor_tb/uut/sync_reset 0
/microprocessor_tb/uut/sync_reset_1 0
/microprocessor_tb/uut/reset_1shot 0
/microprocessor_tb/uut/NOPC8 0
/microprocessor_tb/uut/pc 26
/microprocessor_tb/uut/rom_address 01
/microprocessor_tb/uut/wroffset 0
/microprocessor_tb/uut/rdoffset 6
/microprocessor_tb/uut/wren 0
/microprocessor_tb/uut/pm_data 10
/microprocessor_tb/uut/cache_out f2
/microprocessor_tb/uut/ir d7
/microprocessor_tb/uut/cache_wrline 0
/microprocessor_tb/uut/cache_rdline 0
/microprocessor_tb/uut/cache_rdentry 0
/microprocessor_tb/uut/cache_wrentry 1
/microprocessor_tb/uut/hold_out 0
/microprocessor_tb/uut/start_hold 0
/microprocessor_tb/uut/hold 0
/microprocessor_tb/uut/end_hold 0
/microprocessor_tb/uut/hold_count 0

One of my observations is that the cache_out is loading halfway through the hold_count counter.

Question 5: Definitions

TagID represents the address of the tag. Valid in represents the valid bit which tells the memory that it was a valid memory address. Currdentry represents the current read entry which shares the address that the current address is trying to read from. The lastused represents the last used address in each set to ensure that the most recent address isn't being overwritten but the oldest is

Conclusion

The recorded length of the cache_set_assoc (beginning to last instruction) was 472500000ps

The recorded length of the multi_cache (beginning to last instruction) was 532500000ps

The recorded length of the single_cache (beginning to last instruction) was 559500000ps