Report Lab 2 Parallel System Architectures

Corneliu-Dumitru Soficu
corneliu.soficu@student.uva.nl

January 2020

1 Cache Design

1.1 Structure

The cache is represented as an 8-way set associative structure as follows:

- Entire cache size of **32kB**
- Line size of **32 Bytes**
- Total number of lines: 32.768 bytes / 32 bytes = **1024 lines**
- Total number of sets: 1024 lines / 8 lines per set = **128 sets**

An illustration of the cache structure can be viewed in Fig 1.

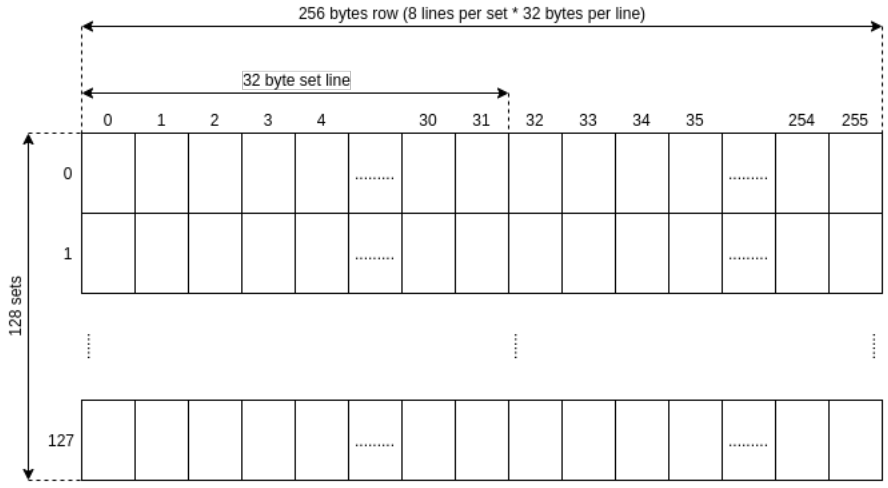

Figure 1: The structure of the cache

1.2 Calculations for address

Given an address of **32** bytes, the following parts of the address can be extracted based on the bit positions, with the least significant bit having index 0:

Address part	**Bit positions**	**Operation for extraction**
Byte in line	0-4	address & 31
Set address	5-11	address & 40640 >> 5
Tag	12-31	address & 4294963200 >> 12

These address components are then used to identify the **Set**, **Line**, and **Position in line** where the data can be stored and retrieved.

1.3 Read operation

The cache controller first identifies the set in which the data is stored, based on the set address component of the address. It then uses the tag to identify the line that is part of the set, where the data is stored. Afterwards, the byte in line part is used to identify which of the 32 possible bytes in the line is the one that stores the data and then, the data is retrieved from the cache using the addressing scheme. Before retrieving the data, the set address is checked to see if it is valid or not based on status.

1.4 Write operation

The cache controller uses the same technique as the one described for the read operation to write in the cache. When writing in the cache, the first invalid cache line is used and if there is none, the least-recently used cache line is used to store a new cache line.

1.5 Addressing scheme

In order to retrieve the data stored in cache once the tag component matches the tag of the line stored in cache, the following addressing scheme is being used:
$data = cache[set\_address][line\_index \cdot 32 + byte\_in\_line]$. The $line\_index$ is obtained based on the matched tag of the line.

1.6 Valid-Invalid Protocol

The Valid-Invalid protocol is being used in order to maintain the cache coherency between multiple caches for a multiprocessor architecture. Each cache line has a state that can be either valid or invalid and the state of the line is modified based on the outcomes of a cache read or write. The transitions between the VALID and INVALID states follow the Transition Diagram in Fig 2.

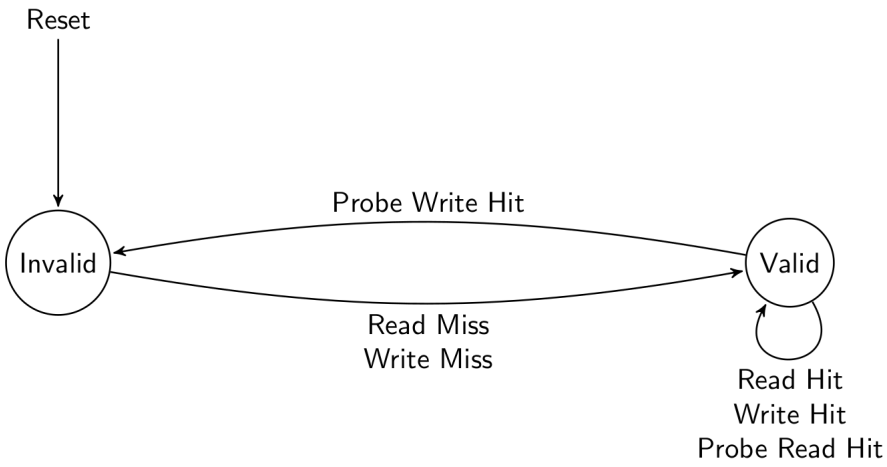

Figure 2: VALID INVALID Cache Line Transition Diagram

2 Simulation Results

After implementing the multiprocessor cache design using the valid-invalid protocol and using SystemC as a simulation environment, the following results were obtained:

2.1 Results for DBG traces

2.1.1 1 Processor Results

CPU Index	Reads	RHits	RMisses	Writes	WHits	WMisses	Hitrate
0	40	19	21	60	26	34	45,00%

Following results regarding bus access were recorded:

Number of bus reads	21
Number of bus write	26
Number of bus readxs	34
Total bus accesses	81
Number of waits for bus	0
Average waiting time for bus access	0,00 ns
Number of waits to maintain bus consistency	0
Number of invalidated addresses while snooping the bus	0
Average per memory access time	145,46 ns
Total execution time	11.782 ns

2.1.2 2 Processors Results

CPU Index	Reads	RHits	RMisses	Writes	WHits	WMisses	Hitrate
0	30	0	30	25	1	24	1,8181%
1	32	0	32	35	1	34	1,4925%

Following results regarding bus access were recorded:

Number of bus reads	62
Number of bus write	58
Number of bus readxs	2
Total bus accesses	122
Number of waits for bus	0
Average waiting time for bus access	0,00 ns
Number of waits to maintain bus consistency	0
Number of invalidated addresses while snooping the bus	0
Average per memory access time	84,71 ns
Total execution time	10.335 ns

2.1.3 4 Processors Results

CPU Index	Reads	RHits	RMisses	Writes	WHits	WMisses	Hitrate
0	8	0	8	8	1	7	6,2500%
1	27	0	27	32	0	32	0,0000%
2	43	1	42	38	2	36	3,7037%
3	45	0	45	42	0	42	0,0000%

Following results regarding bus access were recorded:

Number of bus reads	122
Number of bus write	3
Number of bus readxs	117
Total bus accesses	242
Number of waits for bus	6
Average waiting time for bus access	0,02 ns
Number of waits to maintain bus consistency	0
Number of invalidated addresses while snooping the bus	2
Average per memory access time	54,46 ns
Total execution time	13.180 ns

2.1.4 8 Processors Results

CPU Index	Reads	RHits	RMisses	Writes	WHits	WMisses	Hitrate
0	6	0	6	4	0	4	0,0000%
1	34	0	34	22	0	22	0,0000%
2	35	0	35	43	0	43	0,0000%
3	39	2	37	46	2	44	4,7058%
4	36	0	36	55	0	55	0,0000%
5	52	0	52	47	0	47	0,0000%
6	48	3	45	51	2	49	5,0505%
7	42	1	41	55	5	50	6,1855%

Following results regarding bus access were recorded:

Number of bus reads	286
Number of bus writes	9
Number of bus readxs	314
Total bus accesses	609
Number of waits for bus	75
Average waiting time for bus access	0,12 ns
Number of waits to maintain bus consistency	0
Number of invalidated addresses while snooping the bus	7
Average per memory access time	24,49 ns
Total execution time	14.913 ns

2.2 Results for FFT traces

2.2.1 1 Processor Results

CPU Index	Reads	RHits	RMisses	Writes	WHits	WMisses	Hitrate
0	86.298	7.652	78.646	43.195	10.882	32.313	14,3127%

Following results regarding bus access were recorded:

Number of bus reads	78.646
Number of bus writes	10.882
Number of bus readxs	32.313
Total bus accesses	121.841
Number of waits for bus	0
Average waiting time for bus access	0,00 ns
Number of waits to maintain bus consistency	0
Number of invalidated addresses while snooping the bus	0
Average per memory access time	129,65 ns
Total execution time	15.796.228 ns

2.2.2 2 Processors Results

CPU Index	Reads	RHits	RMisses	Writes	WHits	WMisses	Hitrate
0	29.313	3.402	25.911	15.417	2.013	13.404	12,1059%
1	29.262	3.151	26.111	14.801	1.898	12.903	11,4585%

Following results regarding bus access were recorded:

Number of bus reads	52.022
Number of bus writes	3.911
Number of bus readxs	26.307
Total bus accesses	82.240
Number of waits for bus	564
Average waiting time for bus access	0,01 ns
Number of waits to maintain bus consistency	0
Number of invalidated addresses while snooping the bus	7
Average per memory access time	68,53 ns
Total execution time	5,636,250 ns

2.2.3 4 Processors Results

CPU Index	Reads	RHits	RMisses	Writes	WHits	WMisses	Hitrate
0	11.274	1.744	9.530	6.553	938	5.615	15,0445%
1	9.417	1.462	7.955	6.083	759	5.324	14,3290%
2	9.834	1.526	8.308	5.523	613	4.910	13.9285%
3	9.881	1.499	8.382	5.972	706	5.266	13.9090%

Following results regarding bus access were recorded:

Number of bus reads	34.175
Number of bus writes	3.016
Number of bus readxs	21.115
Total bus accesses	58.306
Number of waits for bus	1.091
Average waiting time for bus access	0,02 ns
Number of waits to maintain bus consistency	0
Number of invalidated addresses while snooping the bus	119
Average per memory access time	38,46 ns
Total execution time	2.242.439 ns

2.2.4 8 Processors Results

CPU Index	Reads	RHits	RMisses	Writes	WHits	WMisses	Hitrate
0	4.956	1.117	3.839	3.154	754	2.400	23,0702%
1	3.983	820	3.163	2.890	543	2.347	19,8312%
2	3.997	873	3.124	2.703	508	2.195	20,6119%
3	4.043	867	3.176	2.695	497	2.198	20,2433%
4	4.055	853	3.202	2.662	476	2.186	19,7856%
5	4.055	854	3.201	2.733	512	2.221	20,1237%
6	4.074	821	3.253	2.710	489	2.221	19,3101%
7	4.124	831	3.293	2.705	470	2.235	19,0511%

Following results regarding bus access were recorded:

Number of bus reads	26.251
Number of bus writes	4.249
Number of bus readxs	18.003
Total bus accesses	48.503
Number of waits for bus	2.149
Average waiting time for bus access	0,04 ns
Number of waits to maintain bus consistency	0
Number of invalidated addresses while snooping the bus	629
Average per memory access time	20,08 ns
Total execution time	973.839 ns

2.3 Results for RND traces

2.3.1 1 Processor Results

CPU Index	Reads	RHits	RMisses	Writes	WHits	WMisses	Hitrate
0	33.031	18.659	14.372	32.505	18.306	14.199	56.4041%

Following results regarding bus access were recorded:

Number of bus reads	14.372
Number of bus writes	18.306
Number of bus readxs	14.199
Total bus accesses	46.877
Number of waits for bus	0
Average waiting time for bus access	0,00 ns
Number of waits to maintain bus consistency	0
Number of invalidated addresses while snooping the bus	0
Average per memory access time	134,09 ns
Total execution time	6.285.550 ns

2.3.2 2 Processors Results

CPU Index	Reads	RHits	RMisses	Writes	WHits	WMisses	Hitrate
0	16.496	385	16.111	16.402	393	16.009	2,3648%
1	24.556	865	23.691	24.804	877	23.927	3,5291%

Following results regarding bus access were recorded:

Number of bus reads	39.802
Number of bus writes	1.270
Number of bus readxs	39.936
Total bus accesses	81.008
Number of waits for bus	327
Average waiting time for bus access	0,00 ns
Number of waits to maintain bus consistency	0
Number of invalidated addresses while snooping the bus	0
Average per memory access time	91,42 ns
Total execution time	7.405.768 ns

2.3.3 4 Processors Results

CPU Index	Reads	RHits	RMisses	Writes	WHits	WMisses	Hitrate
0	8.039	85	7.954	8.251	89	8.162	1,0681%
1	20.450	0	20.450	20.221	0	20.221	0,0000%
2	26.553	982	25.571	26.498	1.056	25.442	3,8415%
3	29.600	0	29.600	29.695	3	29.692	0,0050%

Following results regarding bus access were recorded:

Number of bus reads	83.575
Number of bus writes	1.148
Number of bus readxs	83.517
Total bus accesses	168.240
Number of waits for bus	1.764
Average waiting time for bus access	0,01 ns
Number of waits to maintain bus consistency	0
Number of invalidated addresses while snooping the bus	78
Average per memory access time	53,99 ns
Total execution time	9.083.398 ns

2.3.4 8 Processors Results

CPU Index	Reads	RHits	RMisses	Writes	WHits	WMisses	Hitrate
0	4.113	22	4.091	4.030	35	3.995	0,6999%
1	18.318	0	18.318	18.440	0	18.440	0,0000%
2	25.834	21	25.813	25.470	30	25.440	0,0994%
3	29.549	1.275	28.274	28.801	1.180	27.621	4,2073%
4	31.366	2	31.364	30.548	1	30.547	0,0048%
5	32.015	12	32.003	31.782	26	31.756	0,0595%
6	32.327	1.626	30.701	32.285	1.653	30.632	5,0749%
7	32.536	1.687	30.849	32.584	1.771	30.813	5,3101%

Following results regarding bus access were recorded:

Number of bus reads	201.413
Number of bus writes	4.696
Number of bus readxs	199.244
Total bus accesses	405.353
Number of waits for bus	8.804
Average waiting time for bus access	0,02 ns
Number of waits to maintain bus consistency	0
Number of invalidated addresses while snooping the bus	2.324
Average per memory access time	24,05 ns
Total execution time	9.748.384 ns

2.4 Results when BUS snooping is deactivated

When deactivating the snooping feature of a cache, we are expecting to see an increase in the Hitrate. This is because, by deactivating snooping, the lines of a cache will stop being marked as invalid when another cache is executing a READX or WRITE instruction, thus having more valid instructions that can be fetched by the processor. A consequence of this is that the cache will become incoherent with the memory thus allowing the processor to consume the old invalid data, without knowing that is no longer invalid.