NC State University

Department of Electrical and Computer Engineering

ECE 506 – Architecture of Parallel Computers

Project 2

By

RAMACHANDRAN SEKANIPURAM SRIKANTHAN

Introduction

The project focusses on developing a simulator for analysing the performance various cache coherence protocols. In the typical multicore processor systems, the coherence among various caches is obtained by using two classes of coherence protocols.

- Invalidation based coherence protocols.
- Updated based coherence protocols.

The Invalidation based coherence protocols sends invalidation messages to other caches whenever a value in the current cache which is in the shared state is being modified by the instruction in that processor. The invalidation protocols which are being analysed here are MSI and MESI.

On the other hand, update-based coherence protocols update the value in all the caches whenever a shared variable is being modified. The Updated based protocols which are being analysed here is Dragon protocol.

The performance is being analysed using three experiments which tests the performance based on various cache configurations. Experiment 1 is done by varying cache size from 256KB, 512KB, 1MB to 2MB by keeping the associativity to 8 and block size to 64. Experiment 2 is performed by varying the associativity from 4-way, 8-way to 16-way by keeping cache size fixed at 1MB and block size fixed at 64B. Experiment 3 is performed by varying the block size of cache from 64B,128B to 256B by keeping the cache size fixed at 1MB and cache associativity fixed at 8-way associative. The performance is analysed for four processor system having L1 level cache. All the cache's uses the same configuration and same policy which is being write back and write allocate policy. The Least Recently used replacement policy is being used to evict the block whenever the cache is full.

Experiment 1: Cache Performance analysis for various Cache Sizes:

MSI Statistics:

Cache Size 256 KB				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5775	5805	5771	5813
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.67%	4.77%	4.57%	4.66%
06. number of writebacks	254	235	278	234
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	6745	6740	6774	6761
09. number of interventions	68	47	81	63
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	113	92	120	88
12. number of BusRdX	716	700	725	714

Cache Size 512 KB				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5757	5792	5756	5796
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.65%	4.76%	4.55%	4.65%
06. number of writebacks	190	170	205	171
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	6663	6662	6686	6681
09. number of interventions	71	47	82	63
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	116	92	121	88
12. number of BusRdX	716	700	725	714

Cache Size 1 MB				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5752	5781	5752	5790
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.65%	4.75%	4.55%	4.64%
06. number of writebacks	170	155	186	157
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	6638	6636	6663	6661
09. number of interventions	71	48	82	64
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	116	93	121	89
12. number of BusRdX	716	700	725	714

Cache Size 2 MB				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5750	5779	5751	5789
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.65%	4.75%	4.55%	4.64%
06. number of writebacks	166	143	176	152
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	6632	6622	6652	6655
09. number of interventions	71	48	82	64
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	116	93	121	89
12. number of BusRdX	716	700	725	714

MESI Statistics:

Cache Size 256 KB				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5775	5805	5771	5813
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.67%	4.77%	4.57%	4.66%
06. number of writebacks	254	235	278	234
07. number of cache-to-cache transfers	4405	4441	4406	4411
08. number of memory transactions	1663	1640	1685	1675
09. number of interventions	1468	1432	1469	1479
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	113	92	120	88
12. number of BusRdX	39	41	42	39

Cache Size 512 KB				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5757	5792	5756	5796
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.65%	4.76%	4.55%	4.65%
06. number of writebacks	190	170	205	171
07. number of cache-to-cache transfers	4392	4431	4396	4400
08. number of memory transactions	1594	1572	1607	1606
09. number of interventions	1466	1429	1465	1474
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	116	92	121	88
12. number of BusRdX	39	41	42	39

Cache Size 1 MB				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5752	5781	5752	5790
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.65%	4.75%	4.55%	4.64%
06. number of writebacks	170	155	186	157
07. number of cache-to-cache transfers	4389	4422	4393	4395
08. number of memory transactions	1572	1555	1587	1591
09. number of interventions	1464	1428	1464	1474
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	116	93	121	89
12. number of BusRdX	39	41	42	39

Cache Size 2 MB				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5750	5779	5751	5789
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.65%	4.75%	4.55%	4.64%
06. number of writebacks	166	143	176	152
07. number of cache-to-cache transfers	4387	4420	4392	4394
08. number of memory transactions	1568	1543	1577	1586
09. number of interventions	1464	1428	1464	1474
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	116	93	121	89
12. number of BusRdX	39	41	42	39

Dragon Protocol Statistics:

Cache Size 256 KB				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5635	5646	5644	5652
03. number of writes	11942	11710	12383	12108
04. number of write misses	3	2	2	0
05. total miss rate	4.52%	4.61%	4.43%	4.50%
06. number of writebacks	226	243	232	234
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	5864	5891	5878	5886
09. number of interventions	1405	1396	1398	1430
10. number of invalidations	0	0	0	0
11. number of flushes	3	9	6	9
12. number of BusRdX	0	0	0	0

Cache Size 512 KB				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5601	5610	5610	5617
03. number of writes	11942	11710	12383	12108
04. number of write misses	3	2	2	0
05. total miss rate	4.50%	4.58%	4.41%	4.47%
06. number of writebacks	128	127	135	141
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	5732	5739	5747	5758
09. number of interventions	1400	1388	1389	1418
10. number of invalidations	0	0	0	0
11. number of flushes	3	9	6	6
12. number of BusRdX	0	0	0	0

Cache Size 1 MB				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5595	5604	5604	5611
03. number of writes	11942	11710	12383	12108
04. number of write misses	3	2	2	0
05. total miss rate	4.49%	4.57%	4.40%	4.47%
06. number of writebacks	107	110	105	123
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	5705	5716	5711	5734
09. number of interventions	1398	1387	1387	1417
10. number of invalidations	0	0	0	0
11. number of flushes	3	9	6	6
12. number of BusRdX	0	0	0	0

Cache Size 2 MB				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5591	5603	5601	5608
03. number of writes	11942	11710	12383	12108
04. number of write misses	3	2	2	0
05. total miss rate	4.49%	4.57%	4.40%	4.47%
06. number of writebacks	102	105	98	121
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	5696	5710	5701	5729
09. number of interventions	1396	1387	1387	1416
10. number of invalidations	0	0	0	0
11. number of flushes	3	9	6	6
12. number of BusRdX	0	0	0	0

Experiment 2: Cache Performance Analysis for Various Associativity:

MSI Statistics:

Associativity = 4				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5768	5797	5762	5803
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.66%	4.76%	4.56%	4.65%
06. number of writebacks	239	211	239	224
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	6723	6708	6726	6741
09. number of interventions	69	47	81	63
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	114	92	120	88
12. number of BusRdX	716	700	725	714

Associativity = 8				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5752	5781	5752	5790
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.65%	4.75%	4.55%	4.64%
06. number of writebacks	170	155	186	157
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	6638	6636	6663	6661
09. number of interventions	71	48	82	64
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	116	93	121	89
12. number of BusRdX	716	700	725	714

Associativity = 16				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5741	5772	5741	5780
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.64%	4.74%	4.54%	4.64%
06. number of writebacks	120	101	130	98
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	6577	6573	6596	6592
09. number of interventions	71	48	83	64
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	116	93	122	89
12. number of BusRdX	716	700	725	714

MESI Statistics:

Associativity = 4				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5768	5797	5762	5803
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.66%	4.76%	4.56%	4.65%
06. number of writebacks	239	211	239	224
07. number of cache-to-cache transfers	4402	4434	4401	4401
08. number of memory transactions	1644	1615	1642	1665
09. number of interventions	1465	1431	1465	1480
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	114	92	120	88
12. number of BusRdX	39	41	42	39

Associativity = 8				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5752	5781	5752	5790
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.65%	4.75%	4.55%	4.64%
06. number of writebacks	170	155	186	157
07. number of cache-to-cache transfers	4389	4422	4393	4395
08. number of memory transactions	1572	1555	1587	1591
09. number of interventions	1464	1428	1464	1474
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	116	93	121	89
12. number of BusRdX	39	41	42	39

Associativity = 16				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5741	5772	5741	5780
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.64%	4.74%	4.54%	4.64%
06. number of writebacks	120	101	130	98
07. number of cache-to-cache transfers	4379	4415	4386	4386
08. number of memory transactions	1521	1499	1527	1531
09. number of interventions	1463	1426	1461	1473
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	116	93	122	89
12. number of BusRdX	39	41	42	39

Dragon Protocol Statistics:

Associativity = 4				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5609	5619	5619	5626
03. number of writes	11942	11710	12383	12108
04. number of write misses	3	2	2	0
05. total miss rate	4.50%	4.59%	4.41%	4.48%
06. number of writebacks	148	149	142	158
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	5760	5770	5763	5784
09. number of interventions	1400	1392	1388	1424
10. number of invalidations	0	0	0	0
11. number of flushes	3	9	6	6
12. number of BusRdX	0	0	0	0

Associativity = 8				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5595	5604	5604	5611
03. number of writes	11942	11710	12383	12108
04. number of write misses	3	2	2	0
05. total miss rate	4.49%	4.57%	4.40%	4.47%
06. number of writebacks	107	110	105	123
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	5705	5716	5711	5734
09. number of interventions	1398	1387	1387	1417
10. number of invalidations	0	0	0	0
11. number of flushes	3	9	6	6
12. number of BusRdX	0	0	0	0

Associativity = 16				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5576	5586	5586	5593
03. number of writes	11942	11710	12383	12108
04. number of write misses	3	2	2	0
05. total miss rate	4.48%	4.56%	4.39%	4.46%
06. number of writebacks	47	43	62	57
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	5626	5631	5650	5650
09. number of interventions	1395	1382	1383	1411
10. number of invalidations	0	0	0	0
11. number of flushes	3	9	6	6
12. number of BusRdX	0	0	0	0

Experiment 3: Cache Performance Analysis for various block sizes:

MSI Statistics:

Block Size - 64 B				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5752	5781	5752	5790
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.65%	4.75%	4.55%	4.64%
06. number of writebacks	170	155	186	157
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	6638	6636	6663	6661
09. number of interventions	71	48	82	64
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	116	93	121	89
12. number of BusRdX	716	700	725	714

Block Size - 128 B				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5340	5386	5341	5384
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	40	42	39
05. total miss rate	4.32%	4.43%	4.23%	4.32%
06. number of writebacks	275	250	283	269
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	6344	6347	6369	6386
09. number of interventions	119	84	127	110
10. number of invalidations	2066	2089	2053	2068
11. number of flushes	164	129	166	135
12. number of BusRdX	729	711	745	733

Block Size - 256 B				
Statistics	**Core 0**	**Core 1**	**Core 2**	**Core 3**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5023	5070	5004	5084
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	40	42	39
05. total miss rate	4.06%	4.17%	3.96%	4.08%
06. number of writebacks	363	342	383	332
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	6137	6147	6167	6159
09. number of interventions	166	133	192	145
10. number of invalidations	2132	2157	2107	2148
11. number of flushes	211	178	231	170
12. number of BusRdX	751	735	780	743

MESI Statistics:

Block size 64 B				
Statistics	**Core 1**	**Core 2**	**Core 3**	**Core 4**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5752	5781	5752	5790
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	41	42	39
05. total miss rate	4.65%	4.75%	4.55%	4.64%
06. number of write backs	170	155	186	157
07. number of cache-to-cache transfers	4389	4422	4393	4395
08. number of memory transactions	1572	1555	1587	1591
09. number of interventions	1464	1428	1464	1474
10. number of invalidations	2014	2034	2008	2020
11. number of flushes	116	93	121	89
12. number of BusRdX	39	41	42	39

Block size 128 B				
Statistics	**Core 1**	**Core 2**	**Core 3**	**Core 4**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5340	5386	5341	5384
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	40	42	39
05. total miss rate	4.32%	4.43%	4.23%	4.32%
06. number of write backs	275	250	283	269
07. number of cache-to-cache transfers	4124	4155	4115	4125
08. number of memory transactions	1530	1521	1551	1567
09. number of interventions	1367	1337	1377	1385
10. number of invalidations	2066	2089	2053	2068
11. number of flushes	164	129	166	135
12. number of BusRdX	39	40	42	39

Block size 256 B				
Statistics	**Core 1**	**Core 2**	**Core 3**	**Core 4**
02. number of read misses	5023	5070	5004	5084
03. number of writes	11942	11710	12383	12108
04. number of write misses	39	40	42	39
05. total miss rate	4.06%	4.17%	3.96%	4.08%
06. number of write backs	363	342	383	332
07. number of cache-to-cache transfers	3938	3943	3903	3939
08. number of memory transactions	1487	1509	1526	1516
09. number of interventions	1283	1284	1321	1309
10. number of invalidations	2132	2157	2107	2148
11. number of flushes	211	178	231	170
12. number of BusRdX	39	40	42	39

Dragon Statistics:

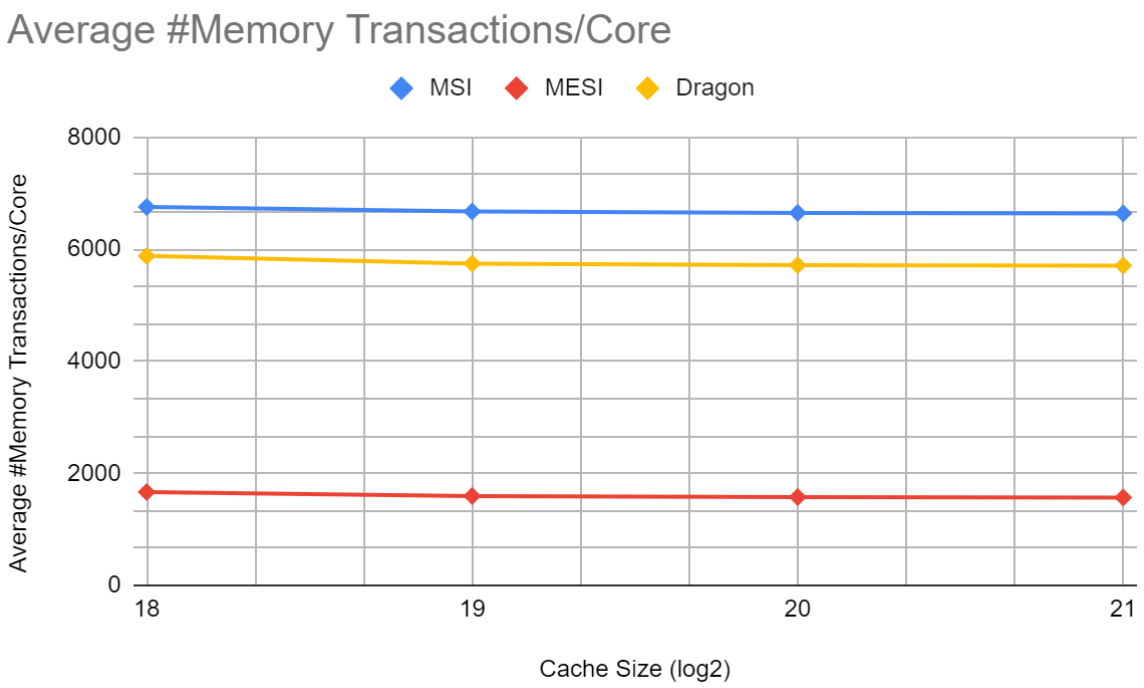
Block size 64 B				
Statistics	**Core 1**	**Core 2**	**Core 3**	**Core 4**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5595	5604	5604	5611
03. number of writes	11942	11710	12383	12108
04. number of write misses	3	2	2	0
05. total miss rate	4.49%	4.57%	4.40%	4.47%
06. number of write backs	107	110	105	123
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	5705	5716	5711	5734
09. number of interventions	1398	1387	1387	1417
10. number of invalidations	0	0	0	0
11. number of flushes	3	9	6	6
12. number of BusRdX	0	0	0	0

Block size 128 B				
Statistics	**Core 1**	**Core 2**	**Core 3**	**Core 4**
01. number of reads	112661	110830	114938	113428
02. number of read misses	5069	5080	5080	5086
03. number of writes	11942	11710	12383	12108
04. number of write misses	3	1	2	0
05. total miss rate	4.07%	4.15%	3.99%	4.05%
06. number of write backs	145	149	145	160
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	5217	5230	5227	5246
09. number of interventions	1256	1266	1257	1287
10. number of invalidations	0	0	0	0
11. number of flushes	3	9	6	9
12. number of BusRdX	0	0	0	0

Block size 256 B				
Statistics	**Core 1**	**Core 2**	**Core 3**	**Core 4**
01. number of reads	112661	110830	114938	113428
02. number of read misses	4628	4633	4630	4640
03. number of writes	11942	11710	12383	12108
04. number of write misses	3	1	2	0
05. total miss rate	3.72%	3.78%	3.64%	3.70%
06. number of write backs	198	200	185	196
07. number of cache-to-cache transfers	0	0	0	0
08. number of memory transactions	4829	4834	4817	4836
09. number of interventions	1125	1175	1143	1176
10. number of invalidations	0	0	0	0
11. number of flushes	3	11	9	9
12. number of BusRdX	0	0	0	0

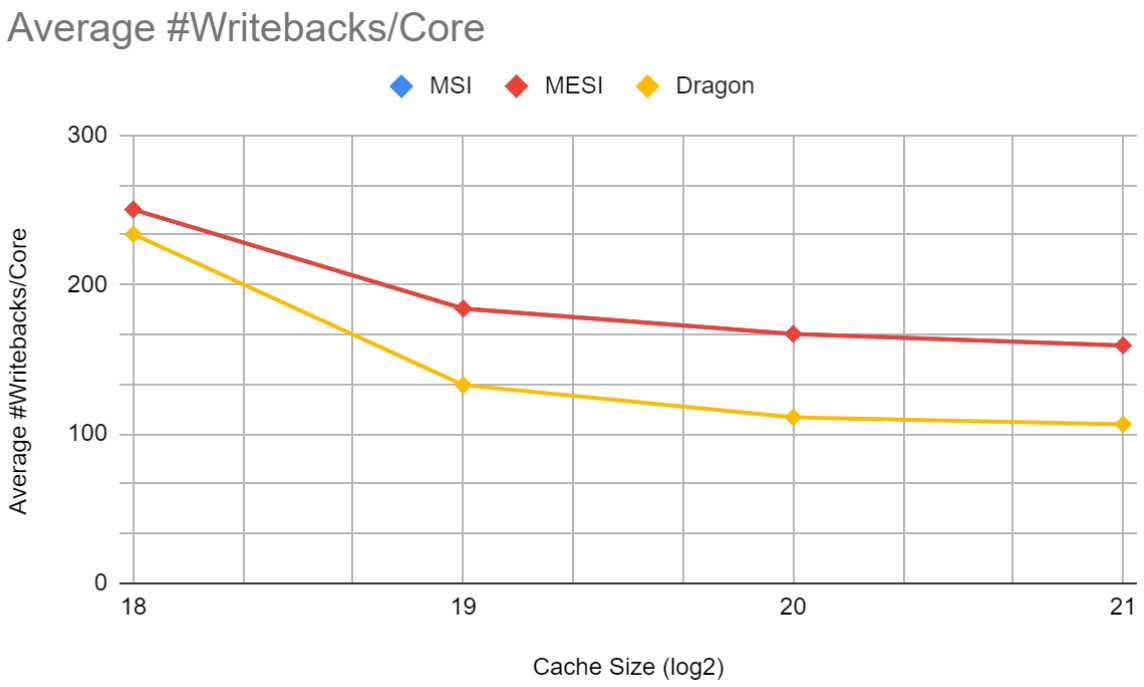
Plots for Various Experiments

Experiment 1:

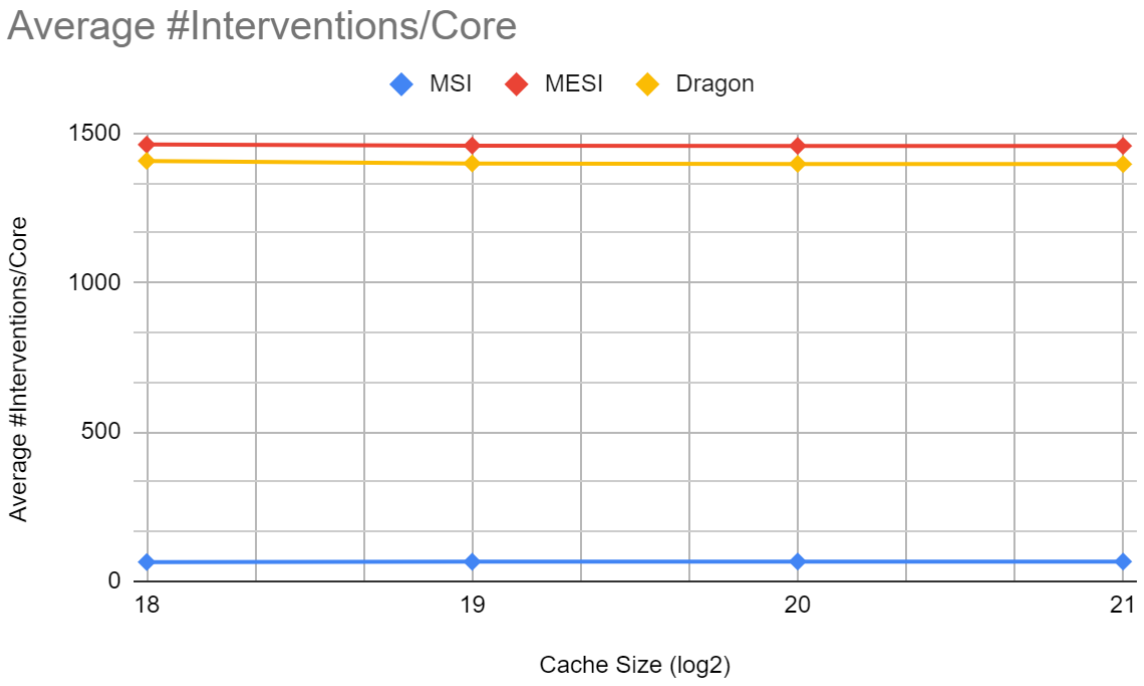


From the graph, we can observe that the average number of memory transactions per core in the case of MSI is higher than that of Dragon which in turn is much higher than that of MESI. The reason being that MSI does not support cache to cache transfer and so each time when there is a miss, even though the value is being cached in other core, the value is written to memory and fetched from there. Even if Dragon protocol doesn't have any explicit cache to cache transfer, each time when any value is modified it is being updated in all the caches and even in the memory. Thus, there are fewer memory accesses in Dragon than in MSI. The MESI protocol, supports cache to cache transfer and so the number memory accesses drastically get reduced here.

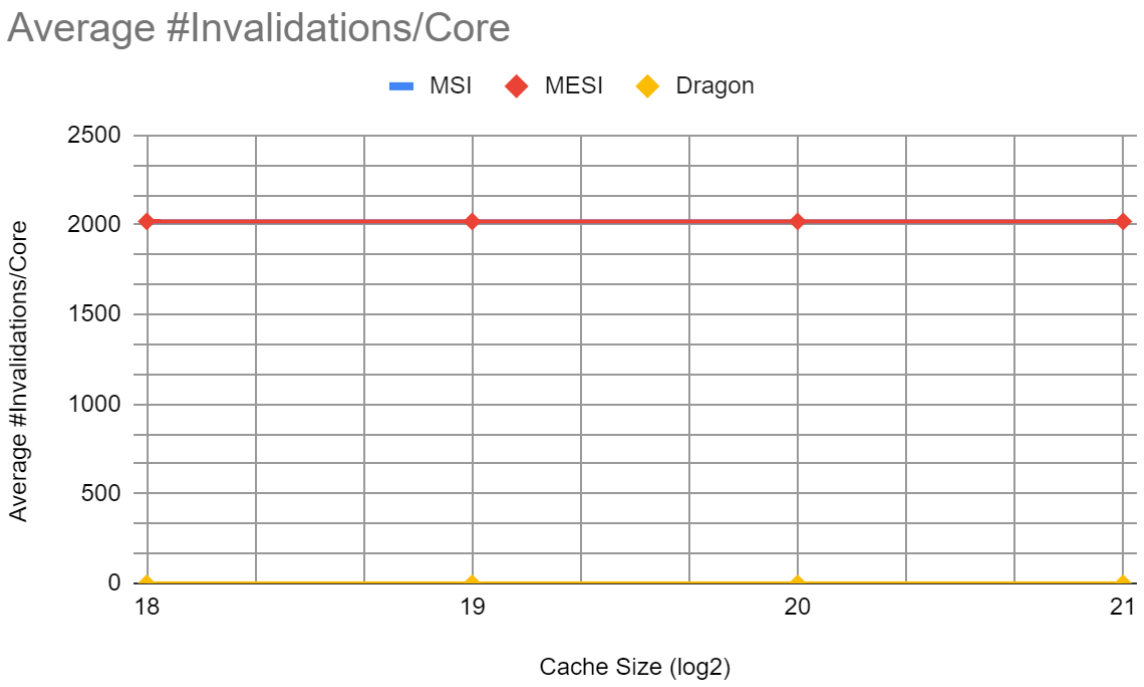
We can also observe that the size of cache has little influence in the average number of memory accesses. The reason being that the number of memory accesses depends mostly on the protocol being used.



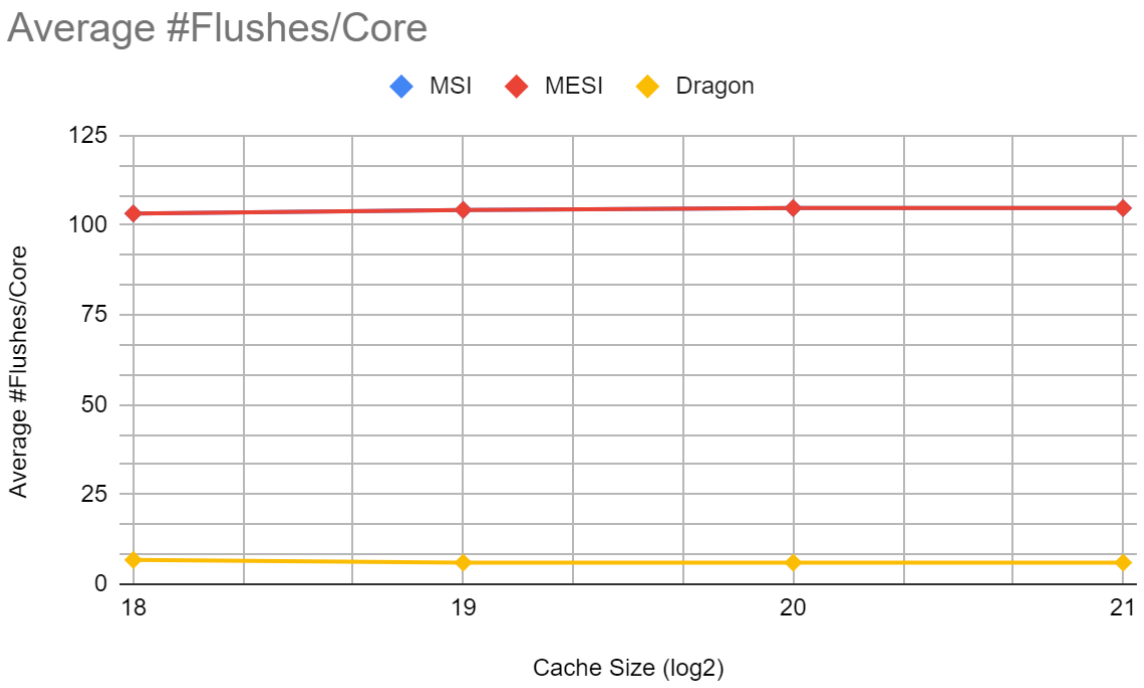
Here we can see from the graph that the average number of writebacks for MSI and MESI are being similar and less compared to MESI. As we increase the cache size, the number of write backs to memory is reduced. Since Dragon protocol allows dirty sharing, the number of writebacks gets reduced as compared to MSI and MESI because these 2 protocols don't allow dirty sharing.



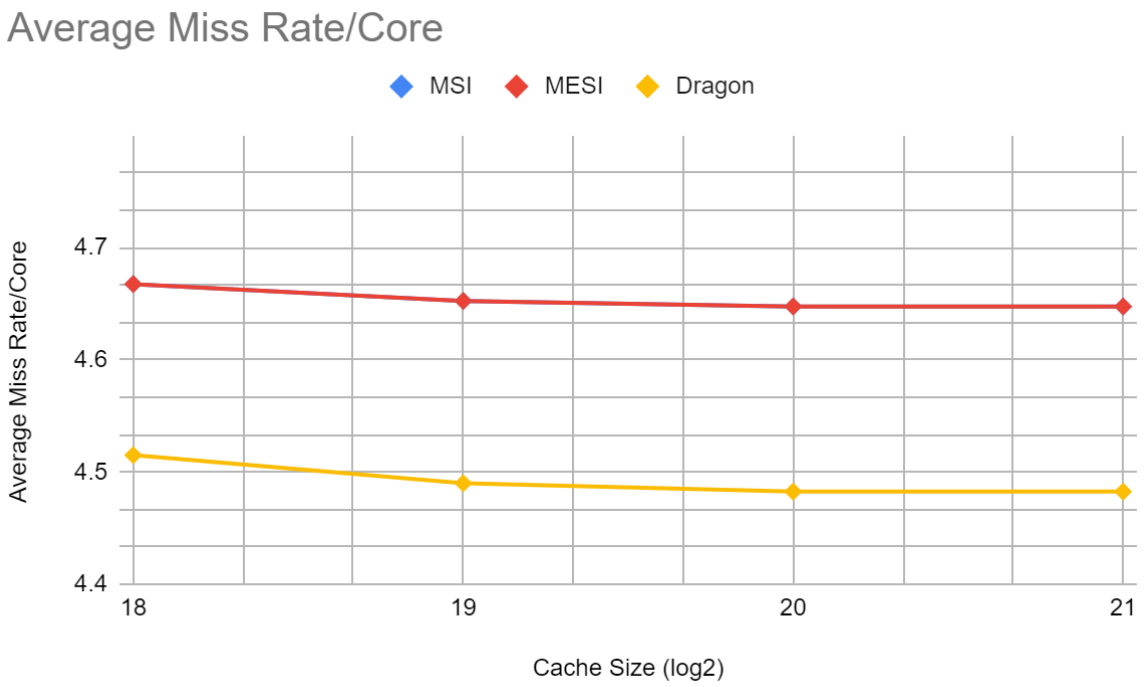
Since MESI and Dragon has a separate Exclusive state which maintains the value as exclusive to all the caches, the number of interventions (Moving from state E/M to Shared) is higher compared to Dragon.



Since Dragon is an update-based protocol, it doesn't have any invalidation messages. Thus, MSI and MESI have large invalidations request.

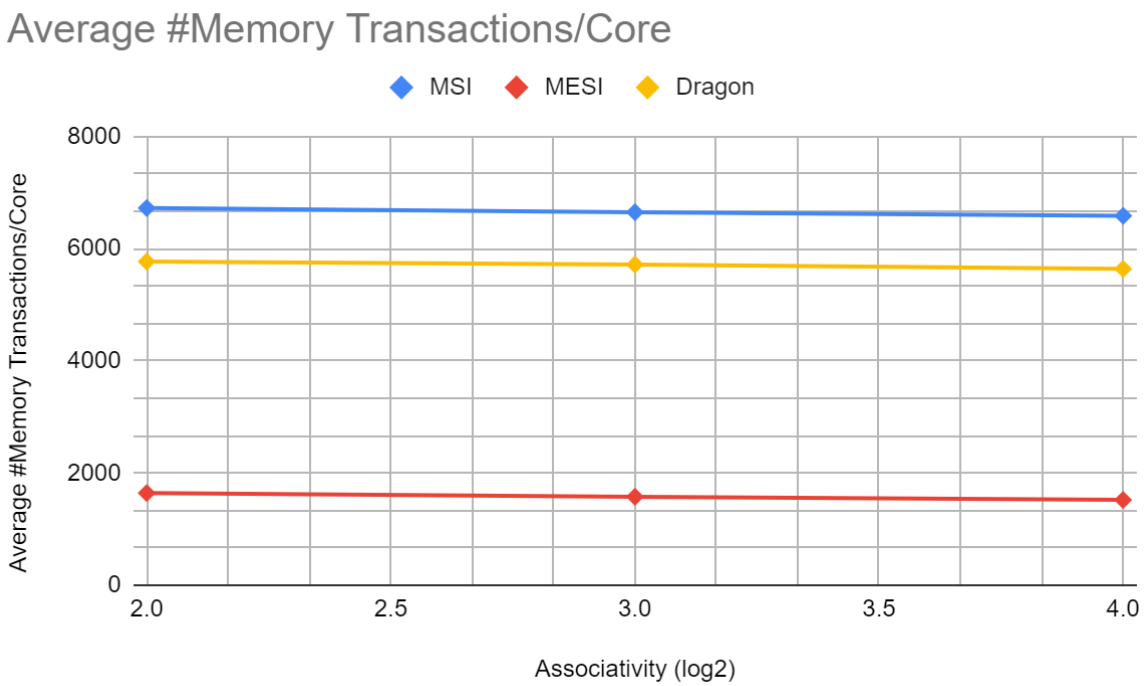


Whenever the Bus Read request is snooped for the state in Modified or in Shared state, there is a flush signal which sends the data. Since Update based protocols update the modified variable each time, a separate flush signal is not being used.

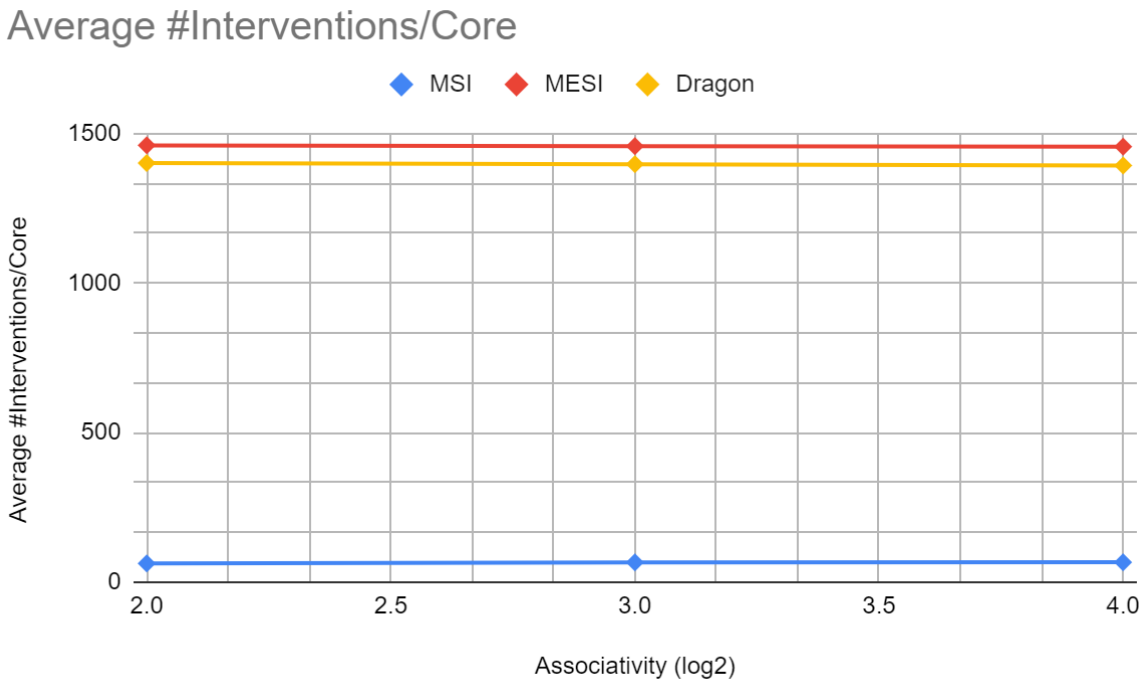


As we increase the cache size, the miss rate decreases since we now have a lot of space to store the data in cache block. We can observe that in case of MSI and MESI protocols, the Miss rate is comparatively higher when compared to Dragon protocol. The reason being that MSI and MESI protocol has a separate invalidation signal unlike Dragon where values are updated in each modification. Thus, due to many invalidation signals MSI and MESI suffers a higher miss rate.

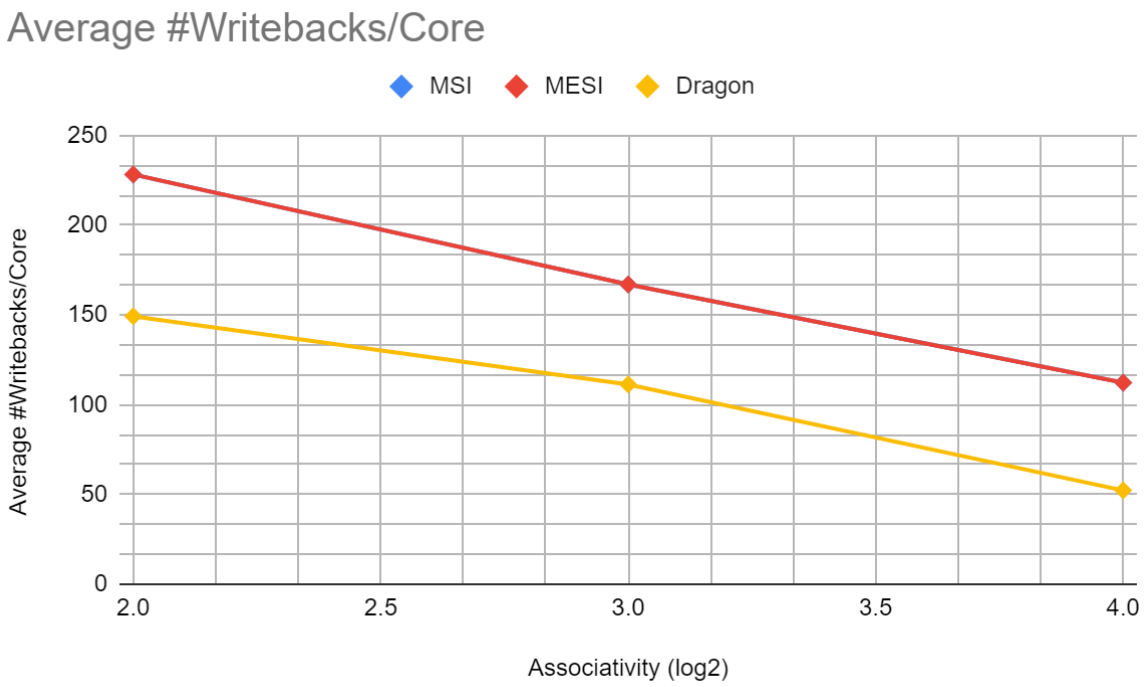
Experiment 2



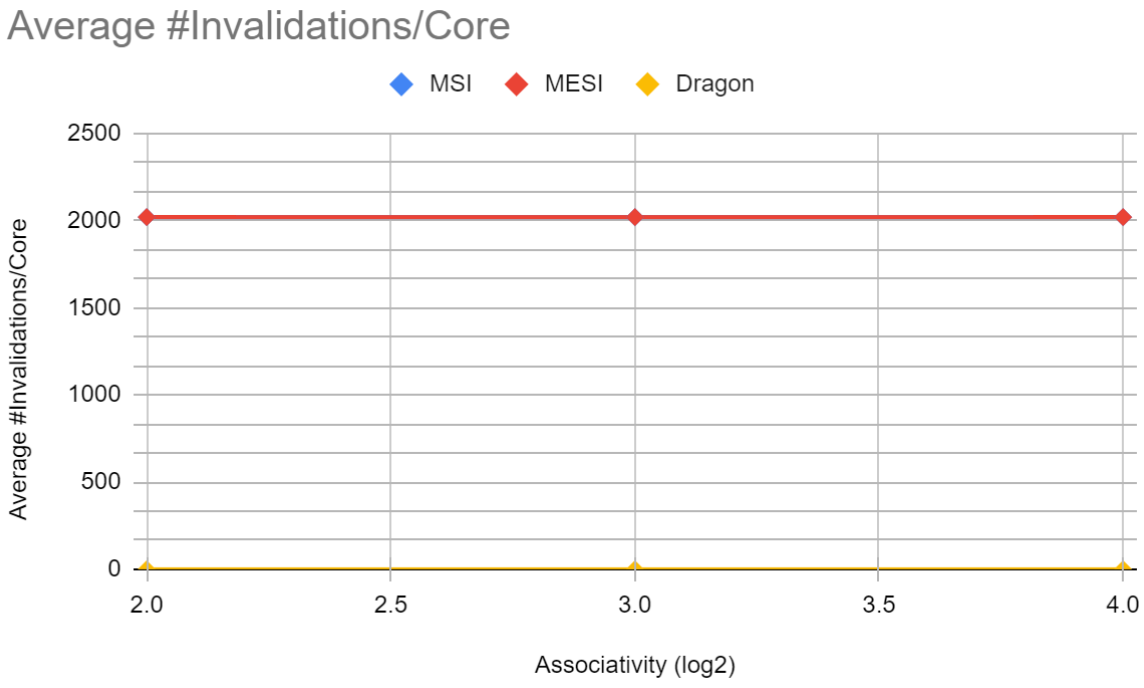
We can observe from the graph that, the associativity has a lesser influence in the average number of memory transactions. Since MSI and Dragon protocol doesn't have any exclusive cache to cache transfer mechanics, it incurs a higher memory transaction compared to MESI where cache to cache transfer is being present.



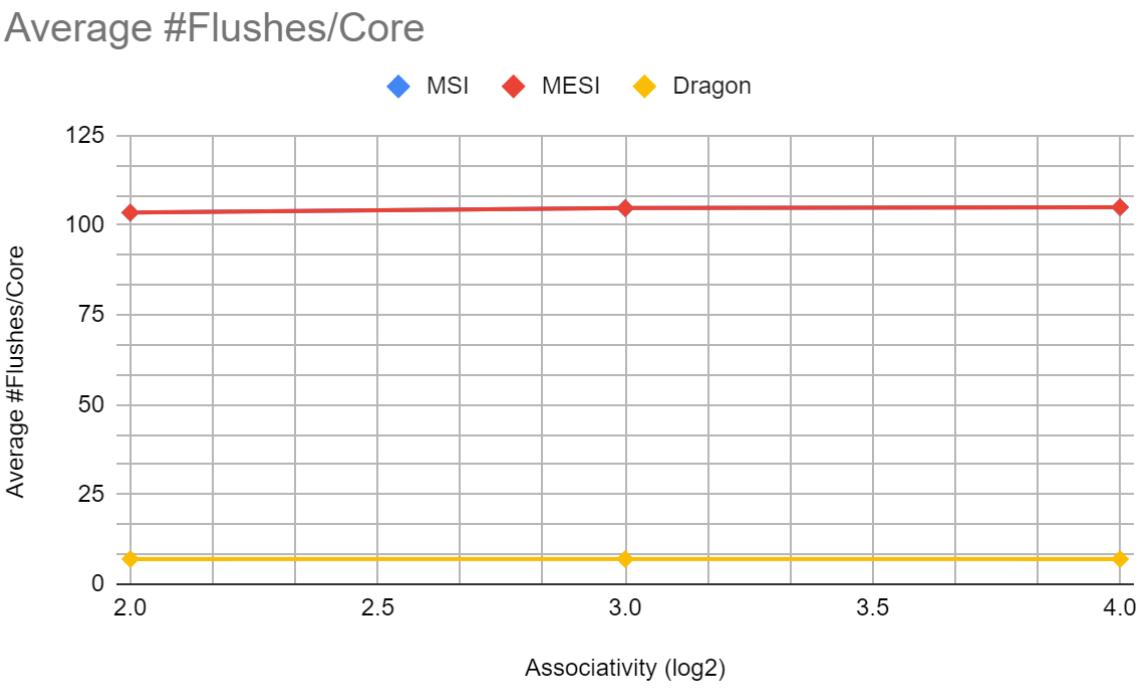
We can observe that associativity has a little impact on the average number of interventions. Since MESI and Dragon protocol has a separate Exclusive state, the number of interventions increase in this case compared to MSI which does not support exclusive values.



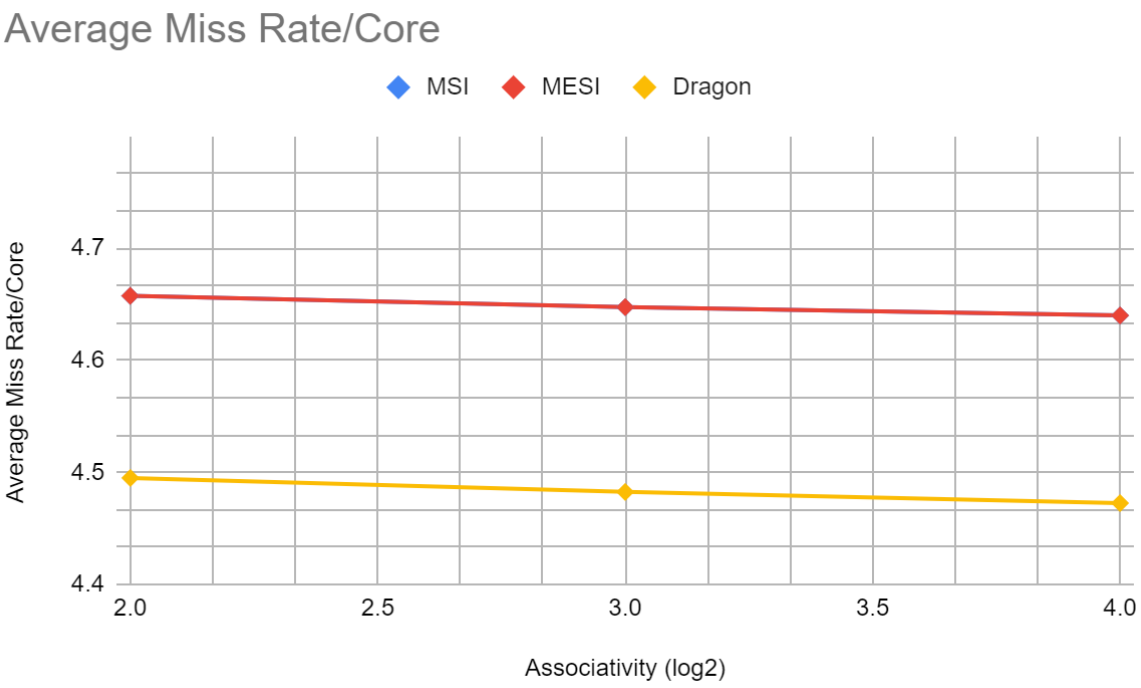
We can see that as we increase the associativity there is a decrease in number of write backs. The reason being that increasing the cache configuration decreases conflict misses thereby providing more space to store values in the cache. Since Dragon protocol allows dirty sharing, the number of writebacks is lesser compared to MSI and MESI wherein dirty sharing is not allowed.



Since MSI and MESI are invalidations-based protocols, we can see many invalidations messages which is being snooped. On the other hand, the Dragon protocol doesn't have any invalidations messages as it updates all the other processors whenever there is a modification.

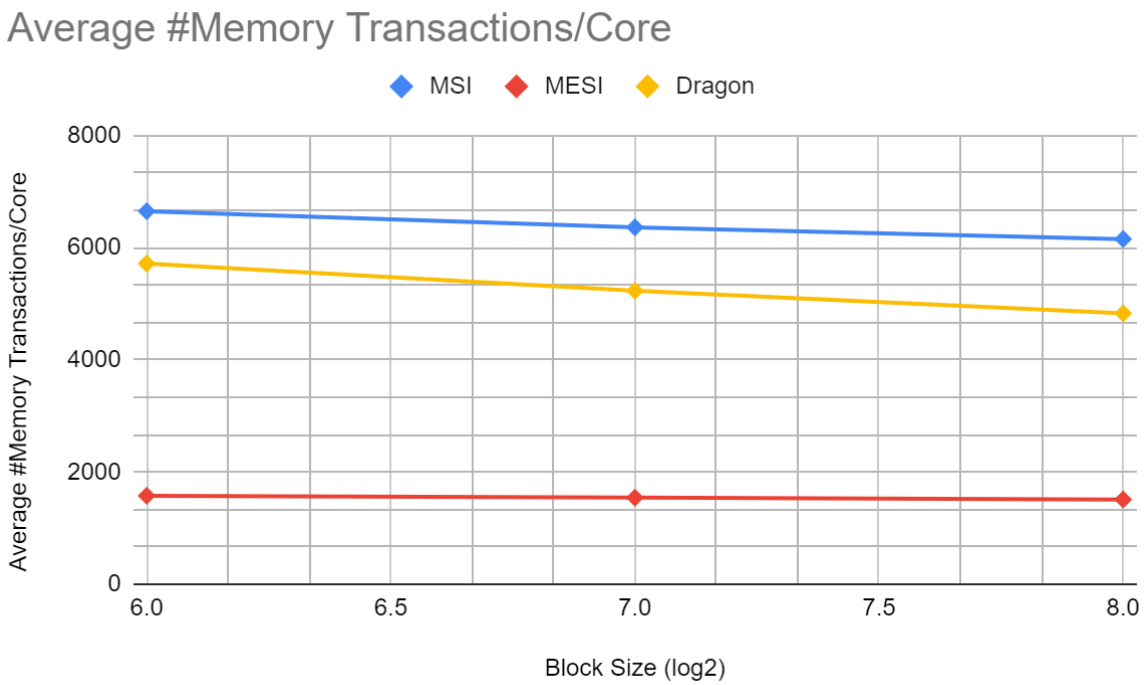


Since Dragon protocol updates each processor's cache during every modification, it doesn't have any separate flush signal unlike MSI and MESI.

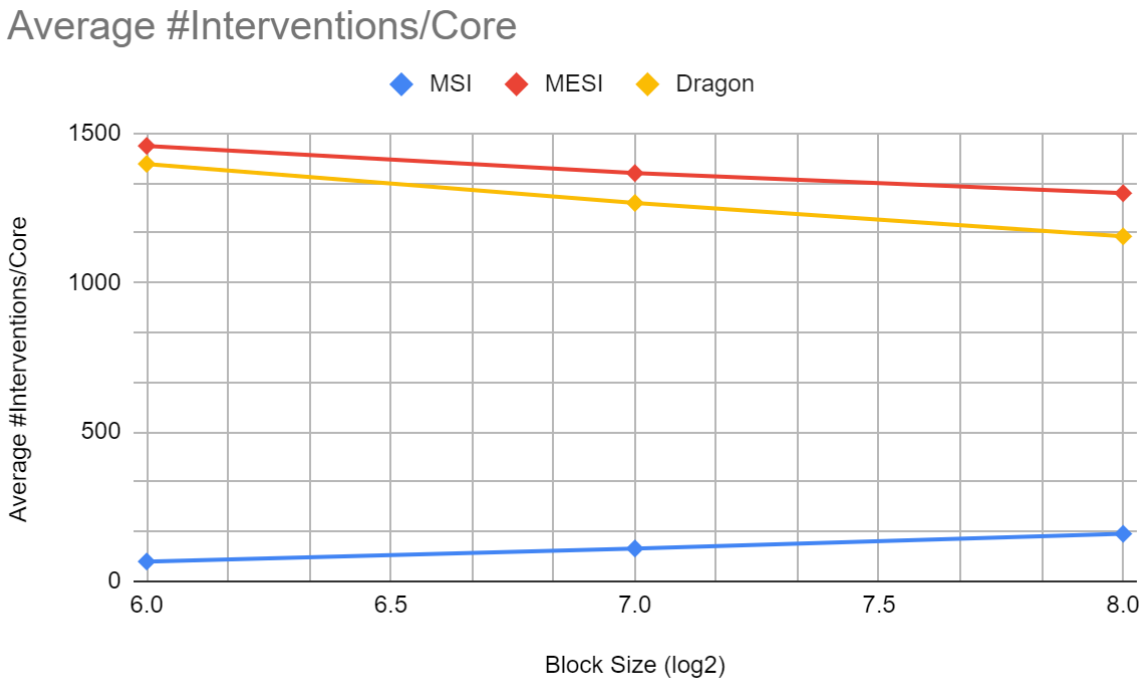


Since associativity has a higher influence on conflict misses (misses due to cache structure), increasing associativity provides more cache blocks thereby reducing the collisions to indexes by various addresses. Also, the MSI and MESI protocols sends additional invalidation request, the average miss rate increases when compared to Dragon which is an update-based protocol.

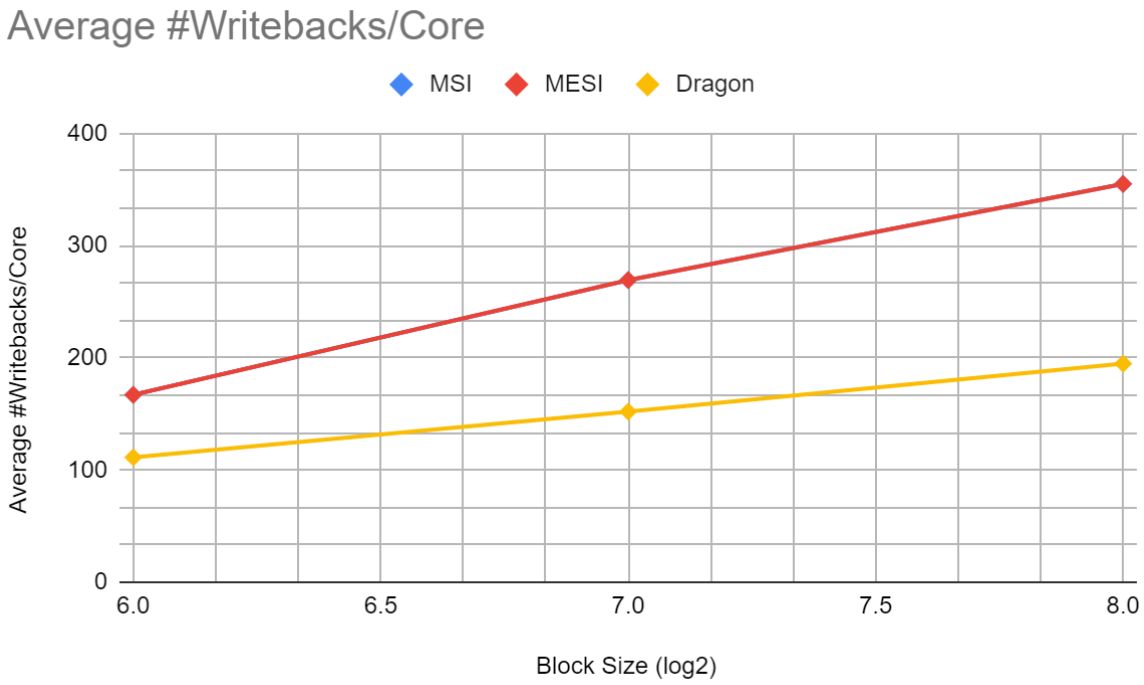
Experiment 3



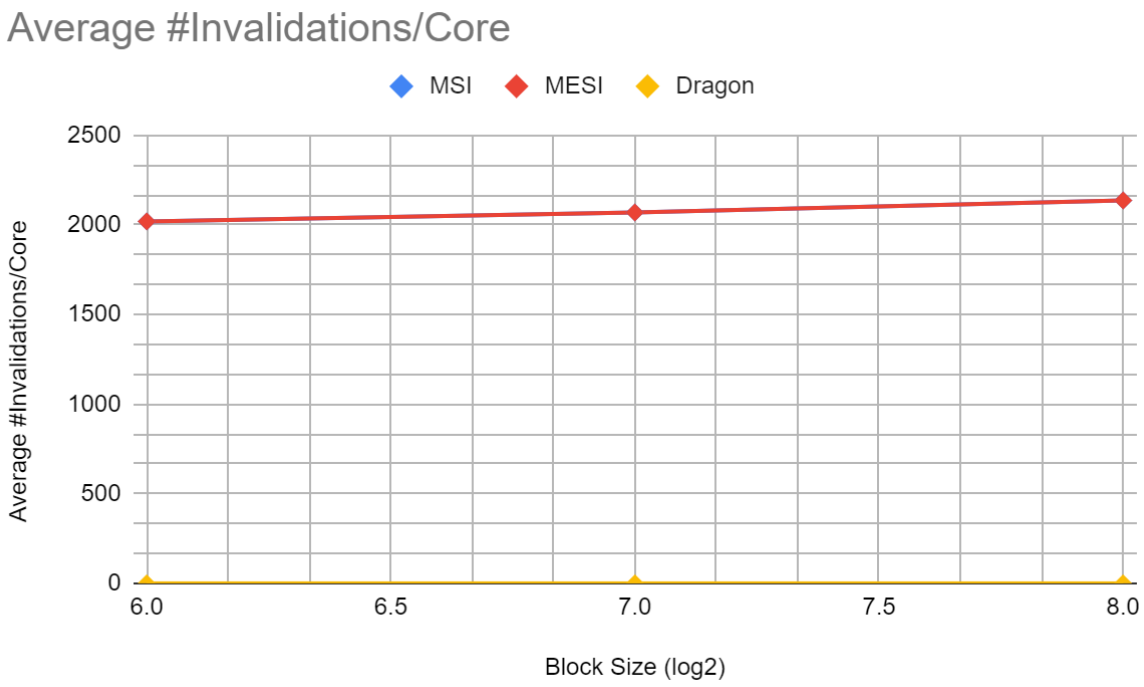
We can see from the graph that, increasing the block size decreases the average number of memory transactions. The reason being that, increasing the block size causes storage of large number of nearby blocks thereby exploiting increases spatial locality. But the graph becomes parabolic after a threshold point where a large amount of unused data is being stored thereby causing increase in memory transactions. The MSI and Dragon protocol does not support cache to cache transfer thereby having a higher number of memory transactions compared to MESI which supports cache to cache transfer.



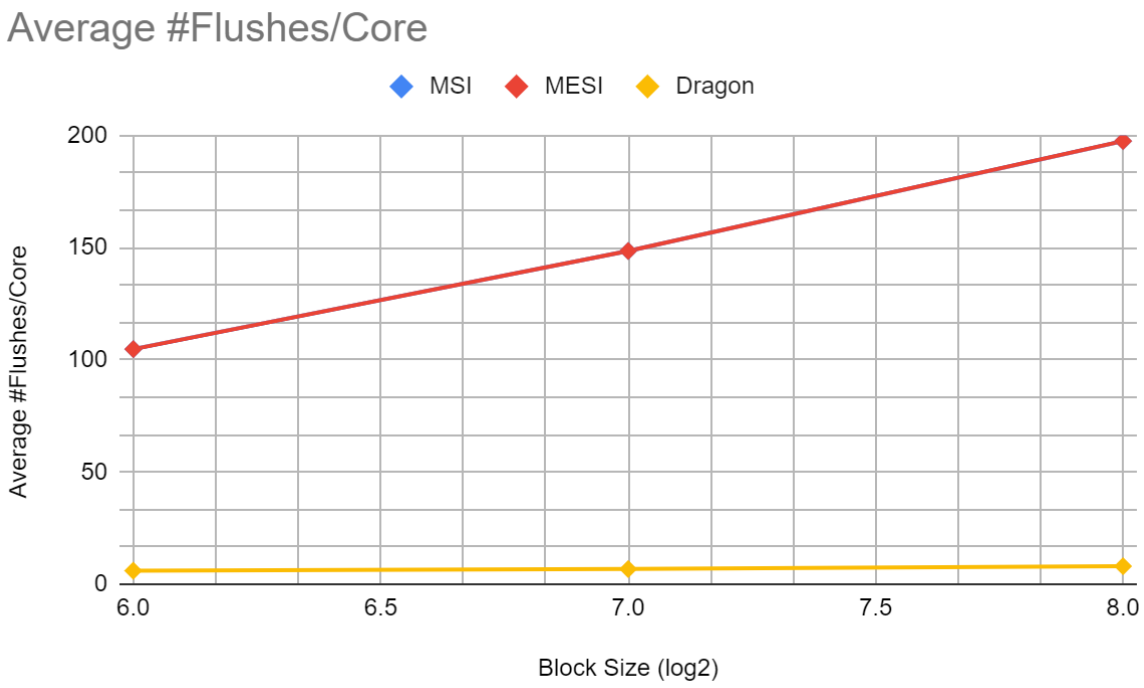
Since MSI does not support a separate exclusive state, the interventions depend only on the transition from modified to shared and so it is lower than MESI and Dragon.



As the block size is increased, more chances are there for the block becoming dirty. Also we can see that, as we increase the size of block there is an higher exploitation of spatial locality at the cost of reduced cache size for storing newer entries. Thus, as we increase the size of block, the number of write backs increases. Also, the number of write backs in MSI and MESI are higher compared to Dragon protocol as Dragon protocol is update based and it does not support invalidations.

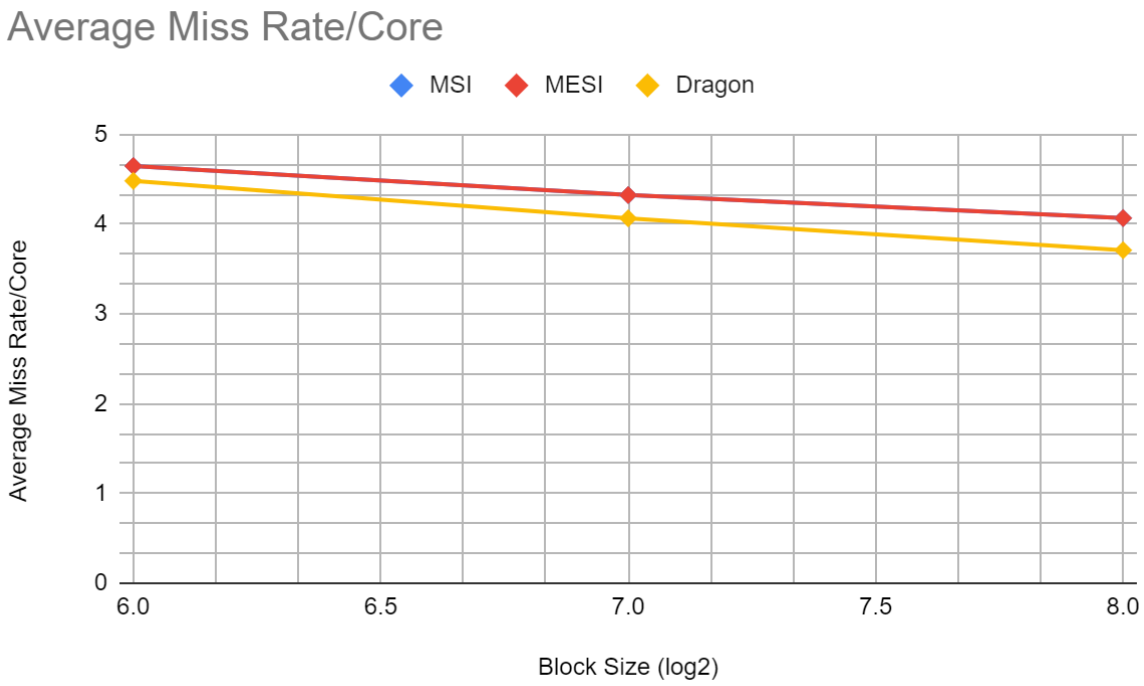


We can see from the graph that as we increase the size of block the number of invalidation snoops increases. The reason being that larger block exploits spatial locality, thereby reducing the size of cache for storing new blocks. So, each time a conflict miss occurs, the values are being written back to memory thereby increasing invalidations. Since Dragon protocol is an update-based protocol, it does not have invalidation messages.



As we increase the size of the block, the is an increase exploitation of spatial locality, which thereby reduces the unique blocks being stored in the cache. Thus, we can see an increase in the flush counts

which indicates a higher memory traffic as well. Since Update based protocols update the modified variable each time, a separate flush signal is not being used.



As we increase the block size, there is an increase in spatial locality usage which thereby causes lesser number of unique blocks being stored in the cache. Thus, it increases the miss rate. As the MSI and MESI protocol has a separate invalidation signal, there is increase in miss rate as compared to Dragon where there are no invalidations.

Inference:

From Experiment 1 we can observe that as we increase the cache size, we can see a significant decrease in the miss rate. The larger the cache size facilitates more blocks to be stored thereby reducing the memory accesses. The writebacks also reduce because there is a smaller number of evictions to the block. This increase in cache size has no impact on number of invalidations, interventions or flushes as it depends on the protocol.

From Experiment 2 we can observe that, as we increase the associativity there is a decrease in miss rate. The reason being that, increasing associativity decreases the conflict miss rate which occurs due to the cache geometry. This also reduces the memory transactions as we have a lot of space in the cache for a particular index thereby reducing the number of collisions. The number write backs also reduces as there is large number of blocks and so lesser evictions.

From Experiment 3 we can observe that, as we increase the block size the miss rate decreases. The reason being that we can exploit the spatial locality to a higher extent. But the miss rate increases after a threshold point where the spatial locality reduces the storage of newer blocks in the cache. The number of writebacks and memory transactions also increases due to increase eviction rate.