Assignment 3

Aditya Jain (20111004) | Mahi Agrawal (20111029)
adityaj20@iitk.ac.in | maahi20@iitk.ac.in
Group Id: 23

1. Computer Specification on which trace was generated is as follows

Architecture:	x86_64
CPU op-mode(s):	32-bit, 64-bit
CPU(s):	24
On-line CPU(s) list:	0-23
Thread(s) per core:	2
Core(s) per socket:	6
Socket(s):	2
NUMA node(s):	2
Vendor ID:	GenuineIntel
CPU family:	6
Model:	85
Model name:	Intel(R) Xeon(R) Gold 6128 CPU @ 3.40GHz
Stepping:	4
CPU MHz:	1200.455
CPU max MHz:	3700.0000
CPU min MHz:	1200.0000
BogoMIPS:	6800.00
L1d cache:	32K
L1i cache:	32K
L2 cache:	1024K
L3 cache:	19712K
NUMA node0 CPU(s):	0-5,12-17
NUMA node1 CPU(s):	6-11,18-23

Table 1. CPU Specifications

Note :- Trace for prog 1 was generated on a 8 core machine, and so it does not follow the the system specifications mentioned above. All the traces of other programs were generated on the system whose specifications are mentioned above.

2. Messages

GET: Message send by L1 cache and received by L2 cache banks, if there is a read miss and the block is in either shared or modified state.

GETX: Message send by L1 cache and received by L2 cache bank, if a core needs a block in modified state or if the block is not in any other private L1 caches.

UPGRADE: Message send by L1 cache and received by L2 cache banks, if a core already has a block in shared state, but it needs the block in modified state.

Sharing Write Back (SWB): Message send by L1 cache and received by L2 cache banks, if

a private L1 cache receives XFER or XFER_INV then in reply L1 cache writes the data in L2 cache.

REPL_S: Message send by L1 cache and received by L2 cache banks, if L1 needs to evict a modified block.

XFER: Message send by home bank to the L1 cache of the core who is the owner of the particular block, and it informs the core that it needs to send the block to the core,who wants the block in shared mode.

XFER_INV: Message send by home bank to the L1 cache who is the current of the a particular block, and another core has requested for the block in modified block.

Invalidation(INV): Message send by home bank of a block to the sharers of a block whenever it receives a "GETX" request,and the block is in the shared state.

Invalidation Acknowledgement(INVACK): Whenever L1 cache receives "INV" then it sends this message to the core,who requested the block in modified state.

PUT: This message is either send by home bank to L1 cache on receiving "GET" message or it is send by owner of a particular block to the requester when the owner receives "XFER" message.

PUTX: This message is either send by home bank to L1 cache on receiving "GETX", or "UPGRADE" message or it is send by owner of a particular block to the requester when the owner receives "XFER_INV" message.

NACK: Negative acknowledgment is send by home bank to the requester whenever the block is in the pending state.

3. Tables

	prog1	prog2	prog3	prog4
Total Machine Accesses	140525855	2570794	9794322	1064715
Total Cycles	137704326	2565311	9774537	1064511

Table 2. All programs machine accesses and cycle counts

L2 Cache Details:				
	prog1	prog2	prog3	prog4
L2 accesses:	6856866	245753	652628	131812
L2 hits:	359738	179049	585034	65046
L2 misses:	6497128	66704	67594	66766

Table 3. L2 Cache Access Details for all programs

Blocks Accessed by	Programs			
	Prog1	Prog2	Prog3	Prog4
Private	432	432	441	8621
Two Threads	70	8262	63	57410
Three Threads	1872	16384	0	6
Four Threads	32456	40958	1	0
Five Threads	143251	5	1	0
Six Threads	244970	0	0	0
Seven Threads	173832	1	1	2
Eight Threads	124528	11	65546	12
Total	721411	66053	66053	66051

TABLE 4. Access Profile

Table 4 shows us the number of blocks that are accessed by multiple threads not specifically at same instance of time.

3.1. **Program 1 Results.**

Note: Trace for this program was generated on 8 core machine.

Messages							
L1 Input Que Message Counts Processor wise							
Core Number	XFER	INV	PUT	INVACK	NACK	XFER_INV	PUTX
Core 0:	23	30	16	25	911	400458	1544503
Core 1:	5	458	4	457	949	389130	1106437
Core 2:	4	469	4	469	963	394437	1145751
Core 3:	5	377	4	376	943	400856	1315435
Core 4:	5	522	6	523	985	406436	1224085
Core 5:	7	483	13	489	811	395201	1244012
Core 6:	2	508	4	510	1022	389519	1272454
Core 7:	5	512	5	510	997	393771	1173997

TABLE 5. prog1 L1 Cache Messages

L2 Input Que Message Count.					
Cache Name	REPL_S	GET	GETX	UPGRADE	SWB
L2	6853517	56	10034105	47	3169864

TABLE 6. prog1 L2 Cache Messages

L1 Cache Details				
Core Number	L1 accesses	L1 cache misses	L1 cache hits	L1 upgrade misses
Core 0:	37764269	1544508	36219750	11
Core 1:	14680248	1106438	13573807	3
Core 2:	14680208	1145752	13534453	3
Core 3:	14680228	1315436	13364789	3
Core 4:	14680208	1224086	13456117	5
Core 5:	14680228	1244013	13436203	12
Core 6:	14680208	1272455	13407750	3
Core 7:	14680258	1173999	13506256	3

TABLE 7. prog1 L1 Cache Details

L1 cache Details in terms of read miss , write miss and upgrade miss							
Core	Read Acc.	Wr Acc.	Rd Hits	Wr Hits	Rd Miss	Wr Miss	Upgd. Miss
Core 0:	18354678	19409591	17633466	18586284	721212	823296	11
Core 1:	5242958	9437290	5242949	8330858	9	1106429	3
Core 2:	5242942	9437266	5242933	8291520	9	1145743	3
Core 3:	5242950	9437278	5242942	8121847	8	1315428	3
Core 4:	5242942	9437266	5242931	8213186	11	1224075	5
Core 5:	5242950	9437278	5242932	8193271	18	1243995	12
Core 6:	5242942	9437266	5242933	8164817	9	1272446	3
Core 7:	5242962	9437296	5242952	8263304	10	1173989	3

TABLE 8. prog1 L1 Cache Details in terms of read miss , write miss and upgrade miss

3.2. **Program 2 Results.**

Results for program 2 are as follows:

Messages							
L1 Input Que Message Counts Processor wise							
Core Number	XFER	INV	PUT	INVACK	NACK	XFER_INV	PUTX
Core 0:	24	17	7	14	0	549	82559
Core 1:	1	3	4	4	0	19	24614
Core 2:	1	3	4	4	0	26	24619
Core 3:	1	3	4	4	0	41	24641
Core 4:	1	2	3	2	0	17	24612
Core 5:	1	2	3	2	0	18	24613
Core 6:	1	2	3	2	0	53	24648
Core 7:	1	2	3	2	0	14	16420

TABLE 9. prog2 L1 Cache Messages

L2 Input Que Message Count.					
Cache Name	REPL_S	GET	GETX	UPGRADE	SWB
L2	242358	31	246708	18	768

TABLE 10. prog2 L2 Cache Messages

L1 Cache Details				
Core Number	L1 accesses	L1 cache misses	L1 cache hits	L1 upgrade misses
Core 0	735969	82558	653403	8
Core 1	283178	24616	258560	2
Core 2	265202	24621	240579	2
Core 3	282462	24643	257817	2
Core 4	262299	24614	237684	1
Core 5	262467	24615	237851	1
Core 6	282456	24650	257805	1
Core 7	196761	16422	180338	1

TABLE 11. prog2 L1 Cache Details

L1 cache Details in terms of read miss , write miss and upgrade miss							
Core	Read Acc.	Wr Acc.	Rd Hits	Wr Hits	Rd Miss	Wr Miss	Upgd. Miss
Core 0:	594298	141671	528460	124946	65838	16717	8
Core 1:	65600	220418	65591	195797	9	24619	2
Core 2:	65600	196699	65591	172093	9	24605	1
Core 3:	65600	225608	65591	200990	9	24616	2
Core 4:	65600	196699	65591	172092	9	24605	2
Core 5:	65600	196699	65591	172092	9	24606	1
Core 6:	65600	211761	65591	187148	9	24612	1
Core 7:	65600	131161	65591	114746	9	16414	1

TABLE 12. prog2 L1 Cache Details in terms of read miss , write miss and upgrade miss

3.3. **Program 3 Results.**

Results for program 3 are as follows:

Messages							
L1 Input Que Message Counts Processor wise							
Core Number	XFER	INV	PUT	INVACK	NACK	XFER_INV	PUTX
Core 0:	21	8	1	2	0	84	197289
Core 1:	0	7	3	8	0	531	65589
Core 2:	0	1	3	2	0	568	65625
Core 3:	0	7	3	8	0	540	65598
Core 4:	0	7	3	8	0	569	65626
Core 5:	1	13	3	13	0	563	65624
Core 6:	0	7	3	8	0	561	65620
Core 7:	0	7	3	8	0	587	65646

TABLE 13. prog3 L1 Cache Messages

L2 Input Que Message Count.					
Cache Name	REPL_S	GET	GETX	UPGRADE	SWB
L2	652063	22	656606	11	4025

TABLE 14. prog3 L2 Cache Messages

L1 Cache Details				
Core Number	L1 accesses	L1 cache misses	L1 cache hits	L1 upgrade misses
Core 0:	2132378	197286	1935088	4
Core 1:	1068084	65591	1002492	1
Core 2:	1088792	65627	1023164	1
Core 3:	1096486	65600	1030885	1
Core 4:	1108926	65628	1043297	1
Core 5:	1089702	65626	1024075	1
Core 6:	1079751	65622	1014128	1
Core 7:	1130203	65648	1064554	1

TABLE 15. prog3 L1 Cache Details

L1 cache Details in terms of read miss , write miss and upgrade miss							
Core	Read Acc.	Wr Acc.	Rd Hits	Wr Hits	Rd Miss	Wr Miss	Upgd. Miss
Core 0:	1053041	1140412	987202	1008949	65839	131460	3
Core 1:	524351	618770	524343	553151	8	65618	1
Core 2:	524351	600752	524343	535144	8	65607	1
Core 3:	524351	673014	524343	607400	8	65613	1
Core 4:	524351	606666	524343	541088	8	65576	2
Core 5:	524351	630335	524343	564716	8	65618	1
Core 6:	524351	655138	524343	589513	8	65624	1
Core 7:	524351	584336	524343	518737	8	65598	1

TABLE 16. prog3

3.4. **Program 4 Results.**

Results for program 4 are as follows:

Messages							
L1 Input Que Message Counts Processor wise							
Core Number	XFER	INV	PUT	INVACK	NACK	XFER_INV	PUTX
Core 0:	20	17	9	16	0	45	74357
Core 1:	1	3	3	4	0	14	8228
Core 2:	3	5	4	6	0	17	8228
Core 3:	2	4	5	4	0	16	8227
Core 4:	2	4	4	4	0	16	8228
Core 5:	2	4	4	4	0	13	8228
Core 6:	2	4	5	4	0	13	8227
Core 7:	3	6	4	5	0	9	8228

TABLE 17. prog4 L1 Cache Messages

L2 Input Que Message Count.					
Cache Name	REPL_S	GET	GETX	UPGRADE	SWB
L2	127805	38	131928	23	178

TABLE 18. prog4 L2 Cache Messages

L1 Cache Details				
Core Number	L1 accesses	L1 cache misses	L1 cache hits	L1 upgrade misses
Core 0:	604885	74356	530519	10
Core 1:	65690	8230	57459	1
Core 2:	65690	8230	57458	2
Core 3:	65690	8230	57458	2
Core 4:	65690	8230	57458	2
Core 5:	65690	8230	57458	2
Core 6:	65690	8230	57458	2
Core 7:	65690	8230	57458	2

TABLE 19. prog4 L1 Cache Details

L1 cache Details in terms of read miss , write miss and upgrade miss							
Core	Read Acc.	Wr Acc.	Rd Hits	Wr Hits	Rd Miss	Wr Miss	Upgd. Miss
Core 0:	528737	76144	462903	67612	65834	8522	10
Core 1:	65	65625	54	57403	11	8219	3
Core 2:	65	65625	54	57404	11	8220	1
Core 3:	65	65625	54	57404	11	8219	2
Core 4:	65	65625	54	57404	11	8219	2
Core 5:	65	65625	54	57404	11	8219	2
Core 6:	65	65625	54	57404	11	8219	2
Core 7:	65	65625	54	57404	11	8219	2

TABLE 20. prog4

4. OBSERVATIONS

4.1. **Observation 1.**

We can observe from Table 3 that there are very less L2 accesses, that essentially means either all four programs hold very good locality of reference, or the reason for less L2 accesses can be that the request of "GET" and "GETX" are forwarded to the core which is the owner of the block,but the latter reason does not holds for prog 2,3 and 4 as the number of "XFER" and "XFER_INV" messages for the respective programs are very small, where as for prog 1 approximately 31% of requests of one core are forwarded to another core (for write requests). Therefore rate of l2 access with respect to total machine access is lesser than or equal to 10%.

4.2. **Observation 2.**

As for the same programs the sharing profile is given in the table 4, the sharing profile tells us that a large number of blocks are accessed by multiple threads, but as we can see that XFER_INV are very less as compared to GETX messages received by L2 cache(lesser than 1% for prog 2,3 and 4 and approximate 4% for prog 1) which essentially means that though there are large number of blocks which are accessed by multiple threads but they are not accessed at same time. Which essentially means that the block is evicted from the core and then only it is used by another core. That is, at a particular instance of time very few blocks are shared among the cores.

4.3. **Observation 3.**

For prog 2, prog 3 and prog 4 , almost none of core requests for the block which is in pending state and so the "NACK" is zero. Where as for prog 1, nack is not zero which essentially means that this program requests for the blocks which are in the pending state.

4.4. **Observation 4.**
We can observe that **upgrade misses** are very less as compared to total machine accesses, this essentially means that there are very few blocks which are present in the cache in shared state. on which processor needs to perform any write operation.

4.5. **Observation 5.**
We can observe that the total "Read Misses" are very high as compared to "GET" message and the sum of "Write Misses" is less than the number of "GETX" message this essentially means that a lot of blocks are not in the shared or modified mode and hence block is provided in "E state".

4.6. **Observation 6.**
We observed that total cycles are less than the total machine accesses. As multiple cores can run in parallel and so multiple requests (each from a different core) are handled in a single cycle.

4.7. **Observation 7.**
We observed that the work is divided equally between all the cores as L1 cache accesses of all the cores except the core 0 are same. Since core 0 also executes the code of main() function also, more cache accesses from it are justified.

4.8. **Observation 8.**
Note :- This observation is according to 8 core machine
We can see that the sharing of blocks or cache to cache transferring for the (for write requests) is high approximately 31% (as total XFER_INV messages are 31% of PUTX). Where as for other programs it is quite low, even when we experimented on a 8 core machine, the percentage was still below 1%(approximately 0.6% for 4-core machine and approximately 0.8 for 8-core machine (for program 3), for other programs as well the percentage remained below 1, on both 8-core and 4-core machine) and hence we can say that the cache to cache sharing for other programs is quite low as when compared to program 1.

5. Verification

- Sum of number of messages of "PUT" and "PUTX" corresponding to a particular core should be equal to the sum of number of L1 cache misses and L1 upgrade messages corresponding to the same core.

- Sum of all XFER and XFER_INV should be equal to the number of sharing writeback "SWB" received by the L2 bank.

- Total number of invalidation should be equal to total number of invalidation acknowledgments "INVACK".

- Sum of XFER corresponding to one particular core should be less than or equal to sum of all the PUT messages of other cores.

- Sum of XFER_INV corresponding to one particular core should be less than or equal to sum of all the PUTX messages of other cores.

- If there are a lot of NACK messages then GET or GETX should be greater than PUT and PUTX messages.

- If upgrade messages are less and if there are zero NACK messages then sum of all PUT messages will be equals to GET messages, and PUTX will be greater than or equal to GETX message. PUTX will be greater than GETX when the L2 bank receives Upgrade messages, (replies to upgrade messages are in form of PUTX messages).