CS2610

Assignment 6

Title : DRAM Row Buffer management

Objective :To understand the internals of DRAM row-buffer management

The memory controller is the key component of computer architecture that is in charge of handling communication between the processor and the main memory.Since,in the multicore era, the main performance bottleneck has moved toward the memory system the performance of DRAM controllers has become critical.After performing the read or write operation a DRAM device can either keep data in the row-buffer or return the data to the original row. DRAM controllers employ techniques, collectively called the Row-Buffer Management Policy, to decide either to leave a row-buffer open or close it immediately after it has been accessed.Memory controllers have used static page closure policies to decide whether a row should be left open, open-page policy, or closed immediately, close-page policy, after the row has been accessed. The appropriate choice for a particular access can reduce the average memory latency.The performance of DRAM is sensitive to the memory access pattern of the running applications.

Memory controller that employs an Open-Page policy leaves a page open after it has been accessed. This reduces the access latency of the next memory request to the same bank and the same row since the target row is open and there is no extra cost to opening it again (Page-Hit). This policy is desired for workloads with high-locality behaviour.Let us know about page empty,pagr hit and page miss for better understanding.To perform a read or write operation in a DRAM device the target row must be activated i.e moved to the Row-Buffer first. In this scenario if the Row-Buffer is empty the accessed row will be activated and ready for a read/write operation after T_{RCD}. This is called “Page-Empty”.Considering two memory requests to the same row within the same bank, after the first memory request the target row is open in the Row-Buffer. Thus, the

second memory request does not need to wait for T_{RCD} and the read/write operation can be done immediately. This is called a “Page-Hit” which delivers the lowest access latency across the Row-Buffer access classifications.Considering two consecutive memory requests to different rows within the same bank,since after the first access the Row-Buffer holds the previously target row then to access a new row the row buffer must first be precharged before the second row can be activated. This imposes an extra T_{RP} to the overall memory access latency. Moreover,the Row-Buffer cannot be precharged until the data is fully restored to the DRAM cells (T_{RAS}). Therefore, $T_{RC} = T_{RAS} + T_{RP}$ is the memory latency of accessing different rows within the same bank. This is called a “Page-Miss” which imposes the highest access latency in a DRAM device.Now let us get back to the closed page policy. A memory controller that employs the Close-Page policy closes a row immediately after it has been accessed. This reduces the access latency for the next memory request to a different row within the same bank since the row-buffer has been prepared (closed) in advanced for the new row to be activated. This page policy is desired for workloads with random memory access behaviour.

Besides,memory controllers also take advantage of Hybrid (Dynamic) page policies which are a combination of Open-Page and Close-Page policies. This type of policy, instead of using a fixed page closure policy, monitors the memory access pattern at run time and switches between static policies considering the locality and randomness of the memory accesses. Although,the idea of Hybrid page closure polices looks promising the main challenge is the compromising of the prediction accuracy and cost of implementation.

The implementation of the hybrid policy is as follows.We took a saturation counter of value 9 and threshold values low_threshold of value 7 and high_threshold of value 12.The thresholds are of difference 5 and saturation counter in the middle of thresholds.Initially started with open policy.

Code is written as such if at present in open page policy and a page hit is occurred then no action is taken.Page hit is seen when addresses are same and checked in actual address,channel,rank,bank,row.And if currently in open policy and page miss is occured,incremented counter.If currently in close page and if a page hit occurs with last closed page,then counter is decremented.Else if a page miss occurs,no action was taken.

Input File(using 4Gb_X8.vi)	**Open Page (No of cycles)**	**Closed Page (No of cycles)**	**Adaptive hybrid (No of cycles)**
black	218317756	208520848	208805800
comm1	288462200	266310456	264910848
comm2	371416971	334369454	34389321
face	264310764	242561956	24132704
ferret	278521230	260200556	260201409
fluid	306021140	290268761	290368669
freq	194471667	183767470	185762042
MT0-canneal	465313549	429177185	429277289
MT1-canneal	-	-	-
MT2-canneal	-	-	-
MT3-canneal	-	-	-
stream	217414664	206548276	206648156
swapt	322854177	306640181	306116335

Input File(using 4Gb_X8.vi)	Open Page (Total system power in W)	Closed Page (Total system power in W)	Adaptive hybrid (Total system power in W)
black	87.827289	87.582190	87.780585
comm1	90.211572	90.557900	90.641370
comm2	91.521699	91.998304	92.079967
face	91.361812	88.942524	91.897654
ferret	91.157678	90.349765	91.604782
fluid	88.281506	88.352678	88.358368
freq	88.198738	86.106415	88.175537
MT0-canneal	93.173143	90.456967	90.459765
MT1-canneal	-	-	-
MT2-canneal	-	-	-
MT3-canneal	-	-	-
stream	88.723456	88.808543	88.808647
swapt	88.378464	88.245140	88.318924

Commands that are used : We start with executing with **make clean and clean** commands

To model quad core system and execute a benchmark on a quad-core system,we specify the same benchmark four times on the command line.

For example,to execute ‘libq’,we run the following command

“bin/usimm input/4channel.cfg input/libq input/libq input/libq input/libq”

Used 4 channel configuration and 4Gb x8 devices for all the simulations.

The USIMM simulator can run arbitrary multi-application workloads using multiple traces.The workloads include a wide range of memory intensive applications from different benchmark suites and representative regions of interest for each application.

Summarizing the performances of different policies for different workloads.The PARSEC,BIOBENCH and Commercial workloads are taken and performances are compared.The black,comm1,comm2,face,ferret,fluid,freq,stream,swapt show low no of cycles statistics for closed page policy than open page policy.The lower the cycles the better the performance.For workloads black,face,ferret,fluid,freq,stream,swapt closed page policy show lower statistics of total system power.Mostly the workloads prefer adaptive hybrid page policy on average.

Let us go through sensitivity to address mapping schemes.The address mapping unit is one of the most important components of a DRAM controller and has a significant effect on the overall performance of the memory system.The first step to service a memory request is to decompose the physical address to the internal structure of the memory system. This process maps given physical address bits (e.g. 32 address bits) to their corresponding Channel, Rank, Bank, Row and Column indices of a given memory organization. Typically, DRAMs use a fixed address-mapping schemeTo investigate the sensitivity to different address mapping we select the best page policy.The results show that the adaptive hybrid page policy delivers identical or slightly different cycles for the two address mappings. The number of page conflicts in DRAMs and as a result the memory performance is susceptible to the memory address mapping scheme.