

Utilizing RF-I and Intelligent Scheduling for Better Throughput/Watt in a Mobile GPU Memory System

KANIT THERDSTEERASUKDI, University of California, Los Angeles

GYUNGSU BYUN, West Virginia University

JASON CONG, University of California, Los Angeles

M. FRANK CHANG, University of California, Los Angeles

GLENN REINMAN, University of California, Los Angeles

Smartphones and tablets are becoming more and more powerful, replacing desktops and laptops as the users' main computing system. As these systems support higher and higher resolutions with more complex 3D graphics, a high throughput and low power memory system is essential for the mobile GPU. In this article, we propose to improve throughput/watt in a mobile GPU memory system by using intelligent scheduling to reduce power, and multi-band radio frequency interconnect (MRF-I) to offset any throughput degradation caused by our intelligent scheduling. Overall, we are able to improve throughput 17% up to 66% while increasing throughput per watt by an average of 18% up to 26%.

Categories and Subject Descriptors: **B.3.1 [Memory Structures]**: Semiconductor Memories---Dynamic memory (DRAM); **B.3.2 [Memory Structures]**: Design Styles---Primary Memory; **C.3 [Special-Purpose and Application-Based Systems]**: Real-time and embedded systems

General Terms: Design, Performance

Additional Key Words and Phrases: Mobile, GPU, Memory, DRAM, Power, RF-I, Scheduling

ACM Reference Format:

Therdsteerasukdi, K., Byun, G., Reinman, G., Cong, J., and Chang, M.F. 2011. Utilizing RF-I and intelligent scheduling for better throughput/watt in a mobile GPU memory system. *ACM Trans. Architec. Code Optim.*, 15 pages.

1. INTRODUCTION

We have entered a mobile revolution where smartphones are becoming so powerful that they can replace our desktops and laptops as our main computing system. As smartphones, tablets, and other mobile platforms become more advanced they will require higher performing but still low power GPUs to render images more smoothly, faster, and at higher resolutions. A higher performing GPU also means a greater demand for DRAM bandwidth, which could lead to higher power consumption. Multiband radio frequency interconnect (MRF-I) [Byun et al. 2011; Chang et al. 2005; Ko et al. 2005] is a promising technology that offers high bandwidth, concurrency, and low power. Even though MRF-I has been shown to operate with very good power efficiency at high rates [Byun et al. 2011], we can also utilize MRF-I for much lower bandwidth mobile GPU memory systems in order to increase throughput but still maintain low power.

In this article, we propose two intelligent scheduling policies for GPU memory transactions in order to reduce power. The first intelligent scheduling policy is to delay servicing DRAM transactions until the transaction queue is full in order to increase page hits and reduce power hungry DRAM page activates. The second intelligent scheduling policy is to increase the opportunity for the DRAM chips to enter low power mode by scheduling bursts [Shao and Davis 2007], which are transactions to the same page and bank, in groups, which we call *burst groups*. We also propose using MRF-I to provide concurrent logical channels to each bank in order to offset any throughput loss caused by our intelligent scheduling. Combined, these techniques will provide a mobile GPU memory system with higher throughput and at better throughput per watt.

The remainder of this article is as follows. Section 2 gives a background on DRAM, mobile GPU memory systems, and multi-band radio frequency interconnect (MRF-I). Section 3 discusses our proposed intelligent scheduling to reduce activates. Section 4 discusses our proposed intelligent scheduling to increase opportunity for DRAM to entering low power mode. Section 5 discusses creating concurrent logical channels for each bank using MRF-I. Section 6 discusses our experimental framework. Section 7 discusses our results. Section 8 discusses related work. Section 9 concludes this article.

2. BACKGROUND

In this section we will give an overview of DRAM, mobile GPU memory systems, and MRF-I. Here we refer to mobile systems as those found in smartphones and tablets, not as those found in laptops.

2.1 DRAM Background

Figure 1 shows the internal contents of a single DRAM chip. A single DRAM chip contains multiple banks. Two banks are shown in this DRAM chip. Banks are independent two dimensional arrays of DRAM cells within a DRAM chip allowing for concurrent execution of DRAM commands. However, in order to limit the size of the DRAM chip, all banks share a common command, address, and data bus. Bank selection for the command and address lines is determined by the bank control logic, which takes the bank address in as input.

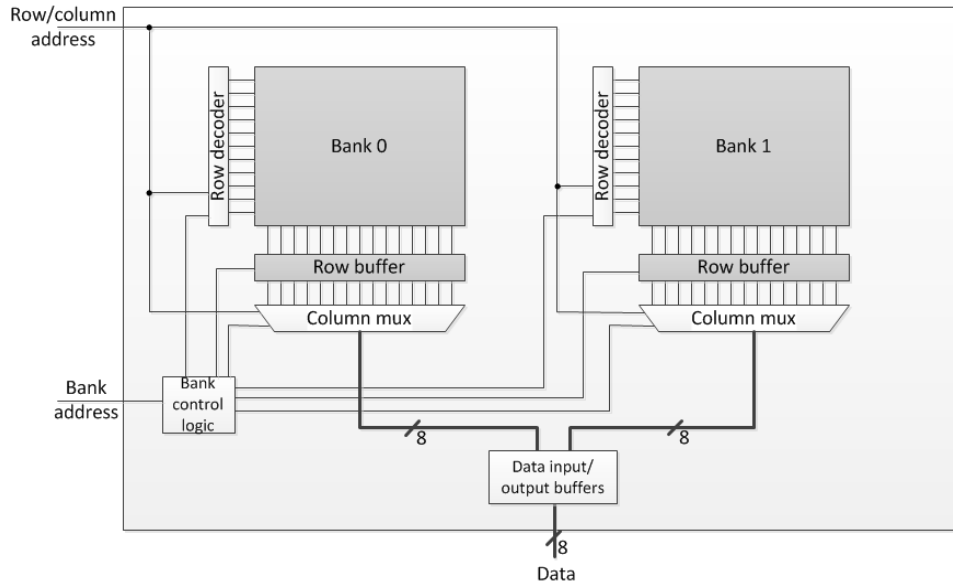


Fig. 1. A single DRAM chip

Retrieval of a single DRAM cell within a bank for reading or writing is accomplished by specifying the row and column of the cell in the bank. The requested row, also known as a page, is first read into the row buffer. This process is called activating or opening a page. A row buffer is also sometimes referred to as a page buffer. From there, the column can be selected from the row buffer using the column mux. Each bank has its own row buffer and column mux.

At a circuit level point of view, before activating a page, the bit lines must all first be precharged to a value halfway between logical 0 and logical 1 by issuing a precharge command. In order to create very dense arrays of DRAM cells, each DRAM cell is constructed using a small capacitor to store the charge, and a transistor to connect the capacitor to the bit line. The charge in the capacitor is very small and not enough to represent a logical 0 or 1. Therefore, the logical value of the cell is determined by the change in the precharged bit line voltage when the row is activated and amplified through sense amps. The sense amps are what make up the row buffer. Once a page is in the row buffer, a read or write can be performed on any of the values in the row buffer.

In this article, we will be using low power DDR (LPDDR) low power DRAM (LPDRAM). The advantages of LPDDR over DDRx technology is that it consumes much lower power in normal operating mode, consumes even lower power in its special power-down mode, and is able to transition to power-down mode faster than DDRx. However, with intelligent scheduling, we can reduce the power consumed by LPDDR even more. Table I shows the breakdown of power for a LPDDR-400 DRAM chip. The numbers are calculated from [Micron 2009] and the power name values are the same as those used in DRAMsim [Wang et al. 2005].

The precharge power-down mode (P_PRE_PDN) consumes the least power with 1mW. This occurs when all banks have finished precharging. If all banks are either finished precharging or in the middle of precharging, but not all done precharging, the condition is known as precharge standby or precharge nonpower-down mode (P_PRE_STBY). In P_PRE_STBY, a lot more power is consumed at 30mW than P_PRE_PDN. Similarly, activate also has a power-down (P_ACT_PDN) and nonpower-down (P_ACT_STBY) mode. The main difference between the active and precharge modes is that the activate modes have at least one bank active. Therefore, we can change between active and precharge mode just by

precharging all the banks. Using this knowledge, our intelligent scheduler will try to achieve the lowest of these 4 modes (P_PRE_PDN) whenever possible.

The activate power (P_ACT) is consumed whenever we activate or open a new page. The only way to reduce this is to perform less activates, which means achieving a higher page hit rate. Our intelligent scheduler will also attempt to improve the hit rate in order to reduce P_ACT. The read (P_RD) and write (P_WR) power cannot be reduced, since it is a function of the amount of data transferred. Ideally, if we were to generate a chart of the power breakup, a perfectly efficient DRAM system where all power is used only for the transference of data would consist of 100% P_RD and P_WR power. Therefore, we can use this knowledge to gauge how power efficient our DRAM system is. The higher percentage the P_RD and P_WR power are, the more power efficient our DRAM system is.

Table I. Breakdown of LPDDR-400 power

Power name	Description	mW	How to lower power
P_PRE_PDN	Precharge power-down power: bank idle, done precharging	1	
P_PRE_STBY	Precharge nonpower-down power: bank idle, still precharging	30	Increase periods when all banks precharged. Change to P_PRE_PDN.
P_ACT_PDN	Active power-down power: at least one bank active, not transferring data	6	Increase periods when all banks idle. Change to P_PRE_PDN.
P_ACT_STBY	Active nonpower-down power: at least one bank active, transferring data	33	Increase periods when all banks idle. Change to P_PRE_PDN.
P_ACT	Activate power	101	Decrease # activates
P_WR	Write power	183	
P_RD	Read power	183	

2.2 Mobile GPU Memory Systems Background

Due to the space and power constraints in a mobile system, mobile GPU memory systems are very different from conventional desktop memory systems. Mobile GPU memory can either be integrated on the same die as the GPU or on a separate die. DRAM capacity requirements have been increasing as smartphones become more powerful. Therefore, the trend has been shifting towards DRAM on a separate die, since it would be inefficient to integrate such a large amount of DRAM on the same die. In systems with DRAM on a separate die, the DRAM is soldered directly on board and connected to the GPU with a point-to-point link. Therefore, there is only one DRAM chip per channel in a GPU memory system. However, there may be multiple channels at the expense of adding more pins for each channel. Most current mobile GPU memory systems though only have one channel.

The type of DRAM used in mobile GPU systems is different from desktop GPU systems. Desktop GPU systems tend to use GDDR. The latest, GDDR5, can operate at 7Gbps/pin. While GDDR5 can support very high data rates, it is not suited for a mobile environment, since it is too power hungry. Instead, mobile systems tend to use low power optimized DRAM technologies such as LPDDR. LPDDR has a much lower data rate than GDDR, but has much better power efficiency. In this article, we use LPDDR-400 as our mobile DRAM chip.

Another important characteristic of mobile GPU memory systems are the graphics applications. Graphics applications, for both mobile and desktop environments, tend to have very high page hit rates. This is because there is a lot of spatial locality in processing graphics. For example, when updating an image on the screen, the spatial locality comes from processing neighboring pixels. Non-graphics applications, however, may not have such high page hit rates. Graphics applications also exhibit a very good tolerance to latency due to the GPUs ability to swap in and out threads with little overhead to hide latency, and due to the massively parallel nature of graphics. For example, a transformation can occur in parallel across many pixels at a time. These application characteristics of high page hit rates and strong latency tolerance will help us develop our intelligent scheduler to improve throughput/watt for mobile GPU memory systems.

2.3 MRF-I Background

Multiband RF-I [Byun et al. 2011; Chang et al. 2005; Ko et al. 2005] is a high aggregate bandwidth and power saving alternative to a traditional interconnect. MRF-I is realized via transmission of

electromagnetic waves through multiple carrier channels over a shared transmission line, rather than the transmission of a voltage signal through a single baseband over a wire. In MRF-I, carrier waves are continuously propagated along the transmission line, and data is generated through either the amplitude or phase modulation of the carrier wave. By transmitting independent data streams each over different RF bands, MRF-I can provide simultaneous transmissions of multiple data streams over a shared physical transmission line to improve the aggregate bandwidth and data rates.

There has been much advancement in off-chip MRF-I in recent years [Byun et al. 2011; Chang et al. 2005; Ko et al. 2005]. The most recent advancement [Byun et al. 2011] uses ASK modulation with differential signaling, i.e. two lines to propagate a signal, which we refer to as ASK MRF-I. [Byun et al. 2011] was successful in demonstrating the high data rate and low power of MRF-I, the low BER, and the feasibility of process integration by implementation in a general-purpose logical CMOS process of 65nm. [Byun et al. 2011] demonstrated a dual band MRF-I transceiver operating over 10cm on a FR4 board and Roger 4003C board at 8.4Gbps aggregate data rate and 10Gbps aggregate data rate respectively. The power consumption of the dual band MRF-I transceivers on the FR4 and Roger boards were 21mW and 25mW respectively. Both boards operated with less than a 10^{-15} BER.

MRF-I can also be used to create multiple logical channels over a single physical channel by supporting more than two bands per pair of differential lines i.e. greater than one band per pin. For example, with 2 RF bands per pin (4 RF bands per pair of differential lines), we could support two concurrent logical channels with the same number of pins as a single physical channel. [Byun et al. 2011] can currently support up to 4 RF bands per pin. However, as MRF-I technology advances, we expect the number of RF bands per pin to increase even more.

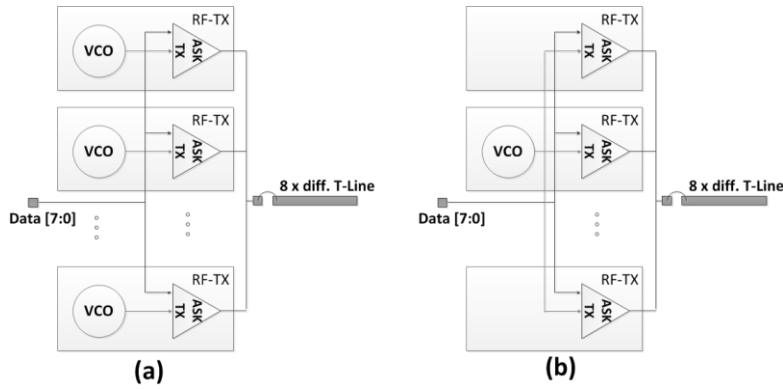


Fig. 2. (a) MRF-I from [Byun et al. 2011] (b) reduced VCO

There is also an area savings improvement that can be made for multi-bit transceivers. [Byun et al. 2011] demonstrates a dual band transceiver over a single pair of differential lines. When creating a multiple bit transceiver, the simplest approach would be just to replicate the design. However, this is very area and energy inefficient. The capacitive loading on each ASK transmitter is very low, so each ASK transmitter does not require its own dedicated voltage-controlled oscillator (VCO) in order to produce the RF carrier (as shown in Figure 2a). Instead, a single VCO can be shared among up to 8 MRF-I transmitters as long as they are using the same RF band (as shown in Figure 2b). This optimization results in both an area and energy savings. We are able to validate these area and energy optimizations by layout and simulation using the Spectre circuit simulator [Cadence Virtuoso Spectre Circuit Simulator 2011] as was done in [Kundert 1999]. The values are shown in Table II and Table III.

Table II. Area of 8-bit transceivers

	BB	2ASK MRF-I	4ASK MRF-I	8ASK MRF-I
Area (mm ²)	0.528	0.372	0.341	0.31
# pins	8	8	4	2
# transceivers	8	4	2	1

The area of 8-bit transceivers, including pads, for baseband (BB) and RF-I transceivers is shown in Table II for 65nm technology. We label transceivers for 2, 4, and 8 RF bands per differential lines as 2ASK, 4ASK, and 8ASK respectively. The individual transceiver size can be obtained by taking the “Area”

and dividing by “#transceivers.” For example, a single 2 ASK transceiver is $0.372\text{mm}^2/4$. Table II shows that as the number of RF bands per pin increases, the area and number of pins required to transmit 8 bits of data shrinks significantly. While the area of MRF-I transceivers are competitive with baseband transceivers, we expect MRF-I transceivers to become much smaller as the technology progresses. Currently, our group is working on a new design to make the MRF-I transceiver several orders of magnitude smaller while still maintaining the same throughput and power characteristics.

Table III shows the power of MRF-I at data rates for LPDDR-400 versus a traditional interconnect, which is labeled as baseband (BB). We compare BB transceiver and receiver power against 2ASK, 4ASK, and 8ASK MRF-I. The MRF-I transceivers and receivers consume slightly more power than baseband. However, as we shall see later on, using MRF-I we are able to decrease the power of the DRAM system overall. Please note that since [Byun et al. 2011] was a proof of concept paper to demonstrate the feasibility of off-chip MRF-I, the circuits were not optimized for area or power. One area reducing improvement that can be made without affecting the operation of the design is to place the digital logic circuits directly underneath the passive structures.

Table III. Power of baseband and MRF-I transceivers

	BB	2ASK MRF-I	4ASK MRF-I	8ASK MRF-I
TX power @ 200Mbps per RF band (mW)	0.3	0.36	0.36	0.36
RX power @ 200Mbps per RF band (mW)	0.35	0.4	0.4	0.4
Line power @ 200Mbps per RF band (mW)	0.1	0.1	0.1	0.1
TX power @ 400Mbps per RF band (mW)	0.5	0.42	0.42	0.42
RX power @ 400Mbps per RF band (mW)	0.55	0.61	0.61	0.61
Line power @ 400Mbps per RF band (mW)	0.2	0.2	0.2	0.2

3. INTELLIGENT SCHEDULING TO REDUCE ACTIVATES

Reducing the number of activates involves increasing the page hit rate. One way to increase the page hit rate is increase the window of available transactions the memory scheduler can choose from. The larger the window, the more likely the memory scheduler can find a transaction to an open page. One simple way to increase the window of transactions is to increase the transaction queue size. However, increasing the transaction queue size would lead to more complex lookup logic and more power. Therefore, we will not increase the transaction queue size. Instead, we will increase the utilization of the transaction queue by waiting until the transaction queue is full before selecting a transaction to schedule.

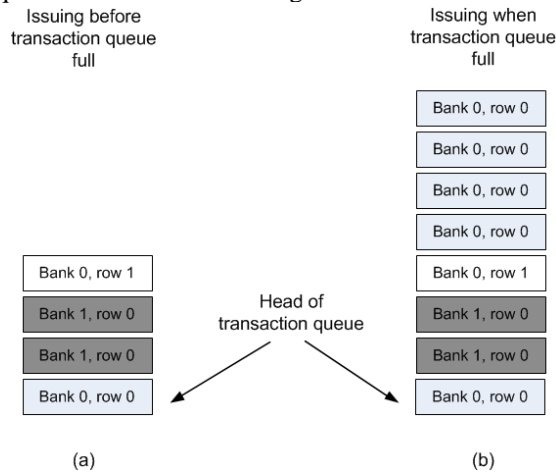


Fig. 3. Example improving page hit rate from (a) issuing before transaction queue full to (b) issuing when transaction queue full

An example of the increase in page hit rate by waiting for the transaction queue to fill up is shown in Figure 3. In Figure 3a, we schedule the transactions without waiting for the transaction queue to fill up. First the transaction to bank 0, row 0 is issued, then the two transactions to bank 1 are issued, then the transaction to bank 0, row 1 is issued. Later on, if 4 transactions to bank 0, row 0 arrive as in Figure 3b, then we would have to reopen bank 0, row 0. Assuming the banks are all initially closed, we would have done 4

activates to service group of 8 transactions. In Figure 3b, we wait until the transaction queue is full before issuing. First all the transactions to bank 0, row 0 are issued before the transaction to bank 0, row 1 is issued. In this case, we perform only 3 activates to service the group of 8 transactions.

There is one problem, however, with waiting for a full transaction queue before issuing. Precious memory bandwidth is wasted as the memory controller waits until the transaction queue fills up. This could lead to a significant degradation in throughput. Therefore, we offset the bandwidth wasted by using MRF-I to provide more bandwidth. We use MRF-I to create a logical channel for each bank over a single physical channel by transmitting each logical channel over a different RF band. This allows us to increase the rate at which memory transactions are serviced. Graphics applications can tolerate the latency as mentioned in Section 2.2, and are more affected by overall throughput than latency. An example of using MRF-I to create per bank logical channels is shown in Figure 4. Figure 4a shows a memory system with a single physical channel. Figure 4b shows a memory system where MRF-I is used to create a logical channel per bank. With multiple logical channels, we are able to improve the time it takes to service these 8 transactions from 8 time steps to just 6 time steps.

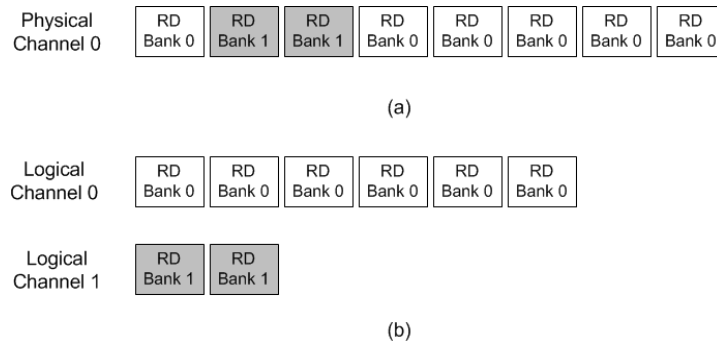


Fig. 4. Improving bandwidth from (a) a single physical channel to (b) multiple logical channels using MRF-I

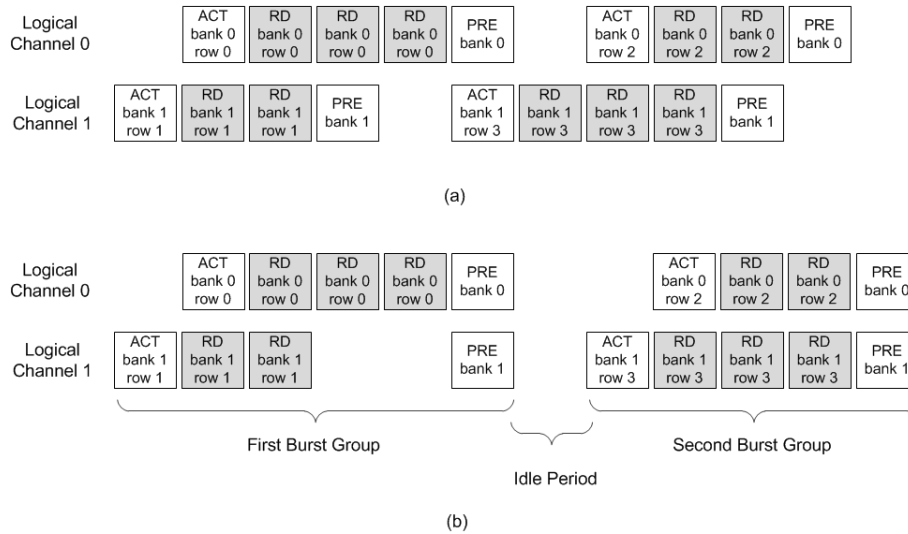


Fig. 5. Increasing idle periods from scheduling (a) without burst groups to (b) with burst groups

4. INTELLIGENT SCHEDULING TO INCREASE OPPORTUNITY FOR DRAM TO ENTER LOW POWER MODE

As mentioned in Section 2.1, the lowest power mode is P_PRE_PDN when all the banks are precharged. In a regular scheduler optimized for throughput, the periods when all banks are precharged are not very common. This is because the scheduler tries to open the next page to a bank immediately in order to overlap the long time it takes to precharge and activate a page with transactions to other banks. However, when we use MRF-I to provide per bank logical channels, we can delay activates in order to increase the periods of when all banks are precharged and enter precharge power-down mode. Therefore, we will

increase periods when all banks are idle by scheduling bursts, which are transactions to the same page, in groups called burst groups.

An example is shown in Figure 5. Figure 5a shows that as soon as bank 1 is precharged closing row 1, the activate to bank 1, row 3 is issued as soon as the timing constraints allow. In this case, bank 1, row 3 is opened before bank 0, row 0 is closed. Therefore, there are no periods when all banks are precharged. In Figure 5b, our intelligent scheduler schedules one burst group at a time. The first burst groups consist of the burst to bank 0, row 0 and the burst to bank 1, row 1. Once the longest burst in the burst group is done, all banks are precharged. The second burst group can then proceed whenever DRAM timing constraints allow it. Since all banks are precharged when all the bursts in the burst group are done, this creates an idle period when all banks are precharged. Furthermore, when we combine the scheduling of burst groups with waiting for the transaction queue to fill up from Section 3, the idle period becomes even longer.

5. USING MRF-I TO CREATE PER BANK LOGICAL CHANNELS

There are not that many modifications that need to be made to a DRAM chip in order to support per bank logical channels with MRF-I. In order to support a logical channel per bank, each bank must be able receive signals for the row buffer, row decoder, column mux, and data input/output buffers independently. Therefore, a MRF-I transceiver that supports 2 RF bands per pin will be required at the row/column address input, bank address input, and data lines in Figure 1 to convert a MRF-I signal to multiple conventional baseband signals. Furthermore, the mux which is located inside the data input/output buffers block is no longer needed, since MRF-I provides independent logical channels for the data. The bank control logic would also have to be modified to be able to generate control signals for multiple banks concurrently. Since MRF-I is fully compatible with CMOS logic, as demonstrated in [Byun et al. 2011], there are no challenges in integrating MRF-I with DRAM chips.

Table IV. GPU traces

Trace name	3D game	Frames
gpu_doom3-640x480_F100_F139	Doom 3	100 to 139
gpu_doom3-640x480_F200_F239	Doom 3	200 to 239
gpu_Quake4-640x480_F100_F139	Quake 4	100 to 139
gpu_Quake4-640x480_F200_F239	Quake 4	200 to 239
gpu_Riddick-640x480_F100_F139	The Chronicles of Riddick	100 to 139
gpu_Riddick-640x480_F200_F239	The Chronicles of Riddick	200 to 239
gpu_UT2004-640x480_F100_F139	Unreal Tournament 2004	100 to 139
gpu_UT2004-640x480_F200_F239	Unreal Tournament 2004	200 to 239

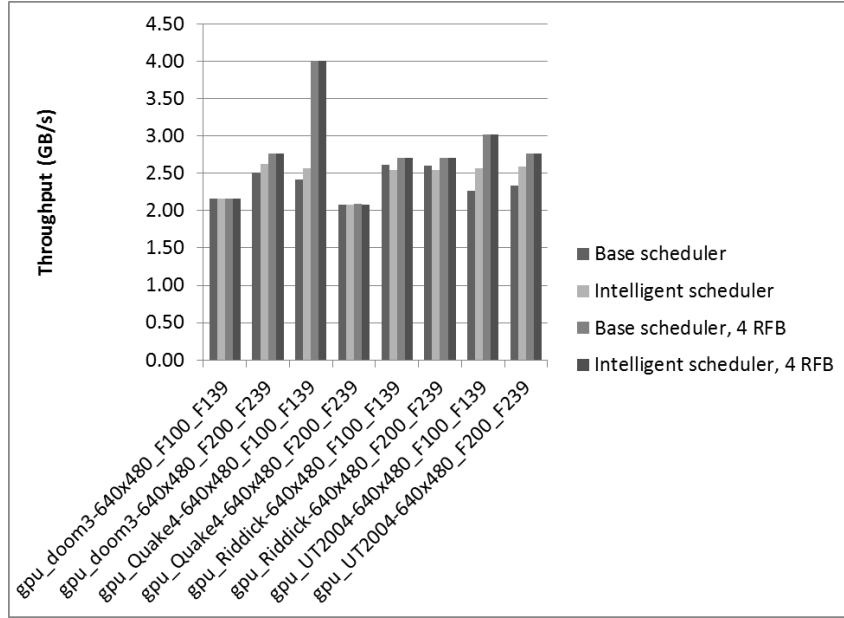
6. EXPERIMENTAL FRAMEWORK

The traces were gathered by modifying the ATTILA GPU simulator [del Barrio 2006] to dump out memory requests. ATTILA was configured as an ATI RV515 GPU operating at 400MHz with the DRAM configured as 4-channel LPDDR SDRAM from a Micron datasheet [Micron 2009]. We use 4 channels, since we do not want the trace to limit our maximum throughput possible when simulating 4 logical channels. ATTILA generated the traces from popular 3D games operating at a resolution of 640 by 480. The 3D games were taken from the ATTILA project webpage [ATTILA traces 2011]. The traces were gathered for frames 100 to 139 and 200 to 239 for each benchmark, similar to [Moya et al. 2005]. The traces are listed in Table IV.

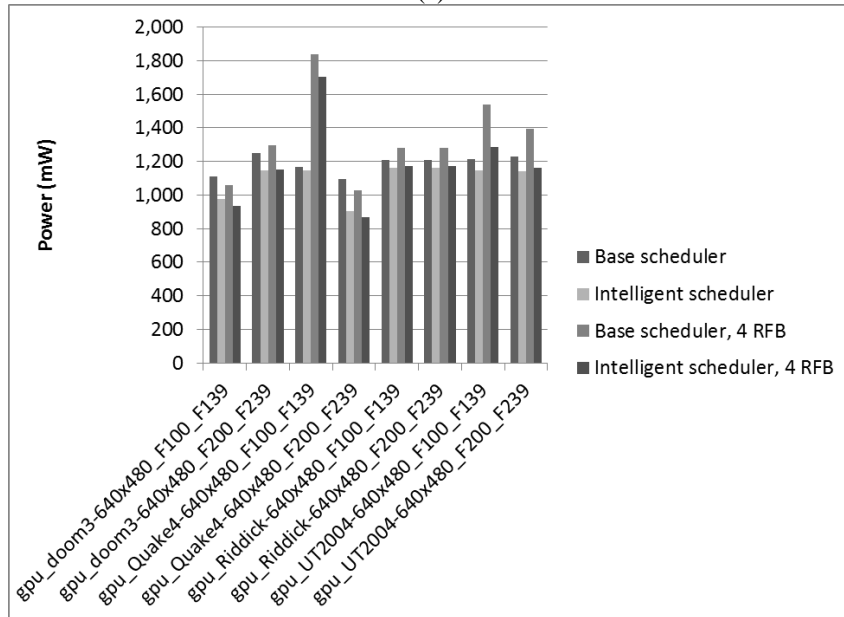
Table V. DRAMsim Configuration

	GPU configuration
DRAM type	LPDDR-400
CPU frequency	400MHz
Channel width	8 bytes
Address mapping policy	sdram_hiperf_map
Row buffer policy	open_page
Ranks per channel	1
Banks per rank	4
Row count	16384
Column count	1024
Rank-to-rank switch time	1 DRAM cycle

All simulations in this work use the parameters in Table V. The traces were taken as input into DRAMsim [Wang et al. 2005], a detailed memory system simulator. With the “open_page” row buffer policy in DRAMsim, the scheduler is configured as a first ready first come first serve (FR-FCFS) scheduler similar to the one in [Rixner et al. 2000]. While DRAMsim has a power model for the DRAM chips, it does not model interconnect power. Therefore, we add our own interconnect power model for baseband and MRF-I which also includes transceiver power. The interconnect power numbers and structures not modeled in DRAMsim were obtained from a highly accurate circuit simulator, Spectre [Cadence Virtuoso Spectre Circuit Simulator 2011], as was done in [Kundert 1999].



(a)



(b)

Fig. 6. Base vs. intelligent scheduler in (a) throughput and (b) power

7. RESULTS

In this section, we discuss our results. Figure 6 shows the throughput and power results of our intelligent scheduler and the addition of MRF-I to support per bank logical channels compared to the base scheduler. The base scheduler without per bank logical channels is labeled as “Base scheduler.” The intelligent scheduler without per bank logical channels is labeled as “Intelligent scheduler.” The base scheduler using MRF-I with 4 RF bands per pin to create per bank logical channels is labeled as “Base scheduler, 4 RFB.” The intelligent scheduler using MRF-I with 4 RF bands per pin to create per bank logical channels is labeled as “Intelligent scheduler, 4 RFB.”

The throughput results, shown in Figure 6a, show that even without MRF-I, the intelligent scheduler for the most part does better than the base scheduler with an average increase of 4% up to 11%. This is because the higher page hit rate of the intelligent scheduler increases throughput more than the degradation caused by waiting until the transaction queue is full to issue requests. The exception is Riddick, which does slightly worse, at 2%, with the intelligent scheduler and no MRF-I. The base and intelligent scheduler with MRF-I both do better than their counterparts without MRF-I with an average 17% up to 66% improvement in throughput compared to just the base scheduler with no MRF-I. While the throughput of the base scheduler and intelligent scheduler both with MRF-I is comparable, we can see in Figure 6b that the power is not. The power of the intelligent scheduler with MRF-I is 12% up to 17% lower than the base scheduler with MRF-I.

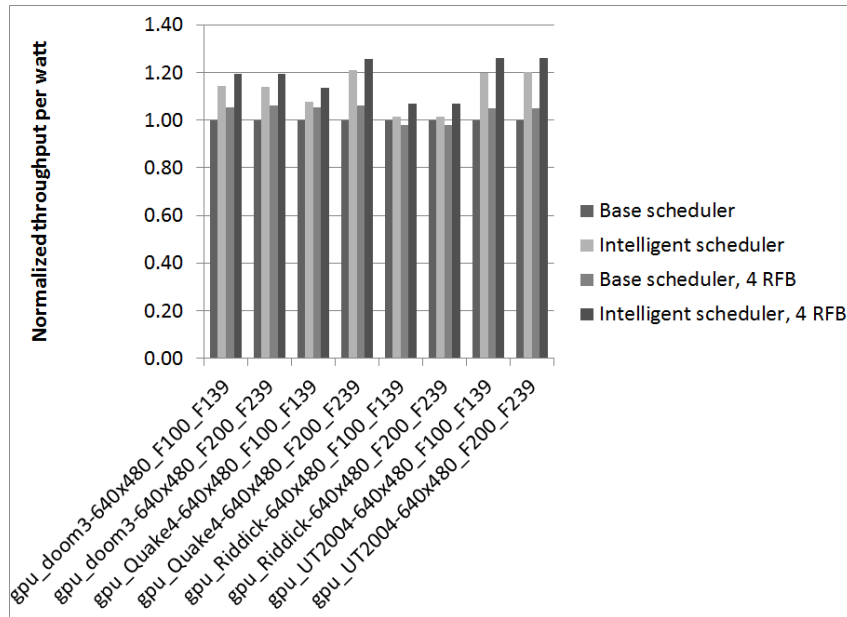


Fig. 7. Normalized throughput per watt of base vs. intelligent scheduler

In order to see how power efficient our design is, we look at throughput per watt. Figure 7 shows the normalized throughput per watt for our simulations. We can see that in all cases, our intelligent scheduler has better throughput per watt. Our intelligent scheduler without MRF-I has an average of 12% up to 21% better throughput per watt than the base scheduler without MRF-I. Even compared to the base scheduler with MRF-I, our intelligent scheduler without MRF-I does better by an average of 9% up to 14%. When MRF-I is added to our intelligent scheduler, the throughput per watt is on average 18% up to 26% better than the base scheduler without MRF-I.

Figure 8 shows the fraction of the time DRAM is in the P_PRE_PDN state. We can see that for the base scheduler both with and without MRF-I, the DRAM is rarely ever on the P_PRE_PDN state, and therefore cannot take advantage of the low power consumed by that state. The intelligent scheduler without MRF-I does see some increases in the P_PRE_PDN state with doom3 frames 100 to 139 and Quake 4 frames 200 to 239. However, it is not until the intelligent scheduler is paired with MRF-I that we are able to see substantial gains in the P_PRE_PDN state across all the benchmarks, where we are able to see an increase of 43% on average up to 60%.

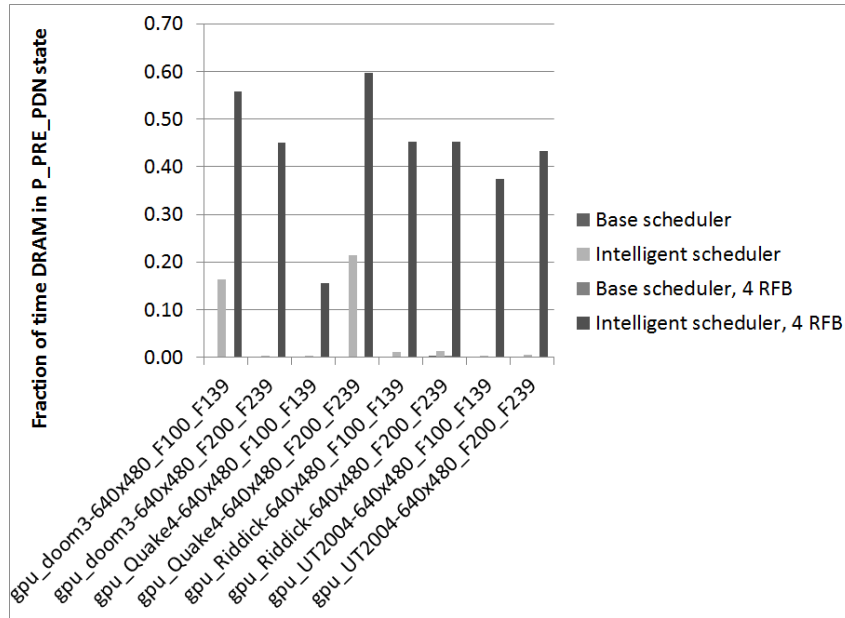


Fig. 8. Fraction of time DRAM in P_PRE_PDN state

Figure 9 shows the page hit rate. Our intelligent scheduler improves the page hit rate both with and without MRF-I. Our intelligent scheduler without MRF-I is able to improve hit rate by 17% on average up to 31% over the base scheduler with MRF-I. When adding MRF-I to our intelligent scheduler, the page hit rate actually gets about 1% worse on average than our intelligent scheduler with MRF-I. This is due to the high bandwidth which causes the service rate to increase leading to lower transaction queue occupancy. Even though the page hit rate is worse with adding MRF-I to our intelligent scheduler, the throughput is much better as we demonstrated in Figure 6a. Therefore, the per bank logical channels created by MRF-I is doing its job of allowing us to increase the page hit rate by delaying issuing until the transaction queue is full, but without degrading throughput.

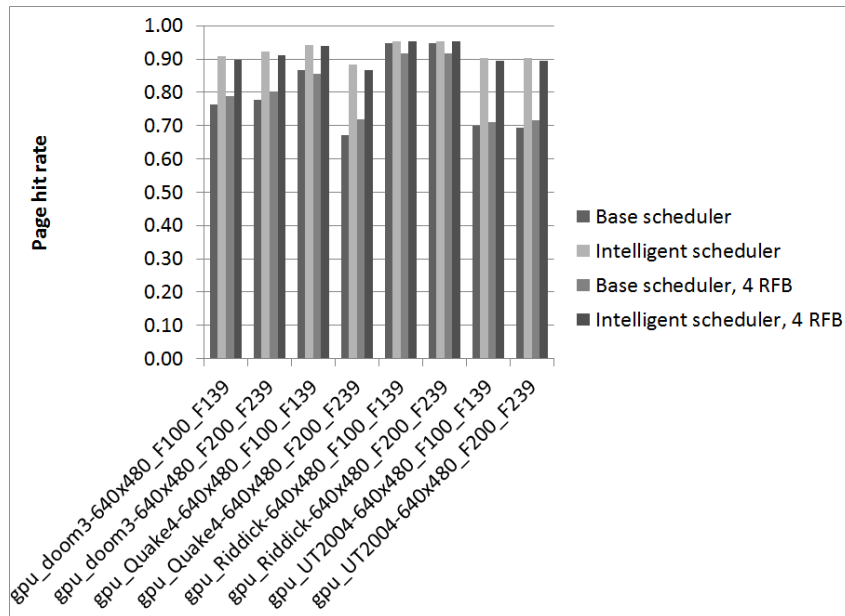
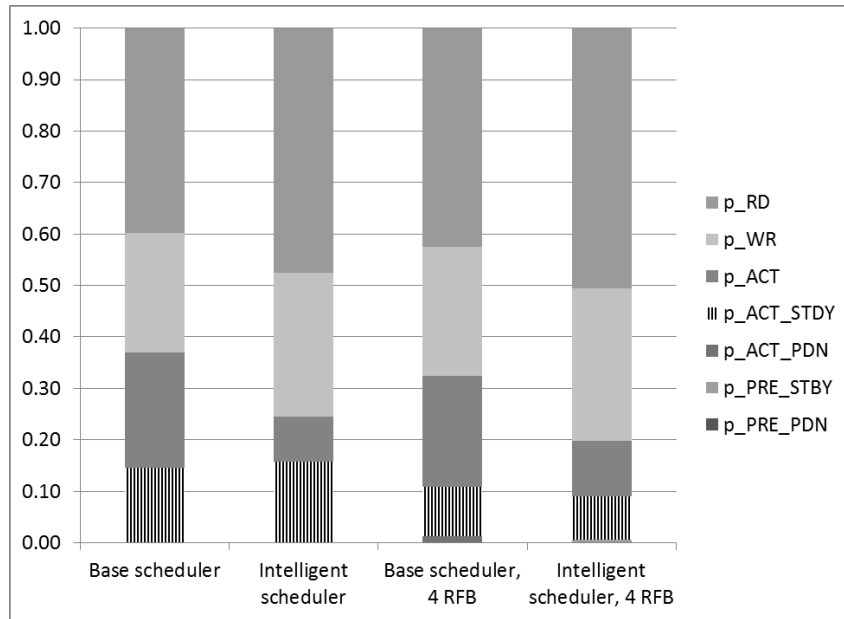


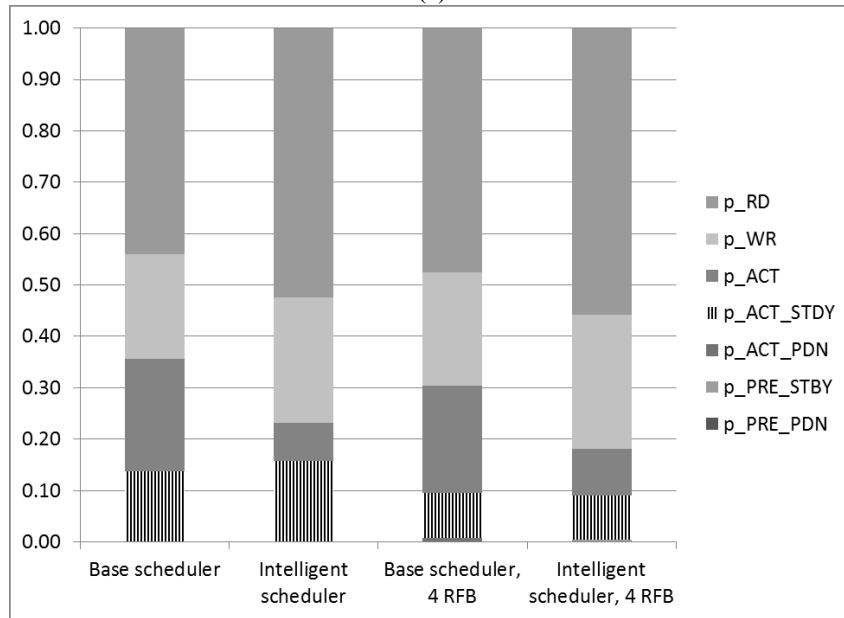
Fig. 9. Page hit rate of base vs. intelligent scheduler

Figure 10 shows the breakdown of DRAM chip power into each of the possible states for doom3. The power is made up mostly of P_RD, P_WR, P_ACT, and P_ACT_STDY. P_ACT_PDN, P_PRE_STBY,

and P_{PRE_PDN} are negligible. In Figure 10a, we see the breakdown for doom3 for frames 100 to 139. In a perfect system where power is only spent transferring data, P_{RD} and P_{WR} would make up 100% of the power. Therefore, we can see how well our scheduler is doing by how close we get P_{RD} and P_{WR} to 100% of the power. We can see that the intelligent scheduler always does better than the base scheduler. The P_{RD} and P_{WR} values change from taking up 63% of the power in the base scheduler without MRF-I to 80% of the power in the intelligent scheduler with MRF-I. Since the intelligent scheduler without MRF-I mainly increases the activates, we see the P_{ACT} value go down from 22% to 9%. When we add MRF-I to the intelligent scheduler, we are converting P_{ACT_STDY} to P_{PRE_PDN} , so we see P_{ACT_STBY} decrease from 15% to 8%. The rest of the benchmarks show similar behavior and are also shown in Figure 11, Figure 12, and Figure 13 for completion.

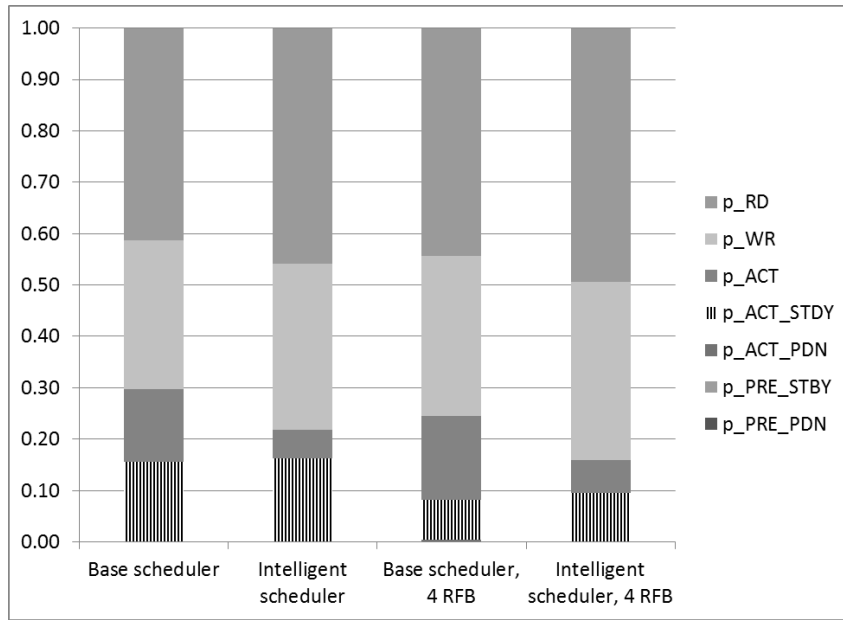


(a)

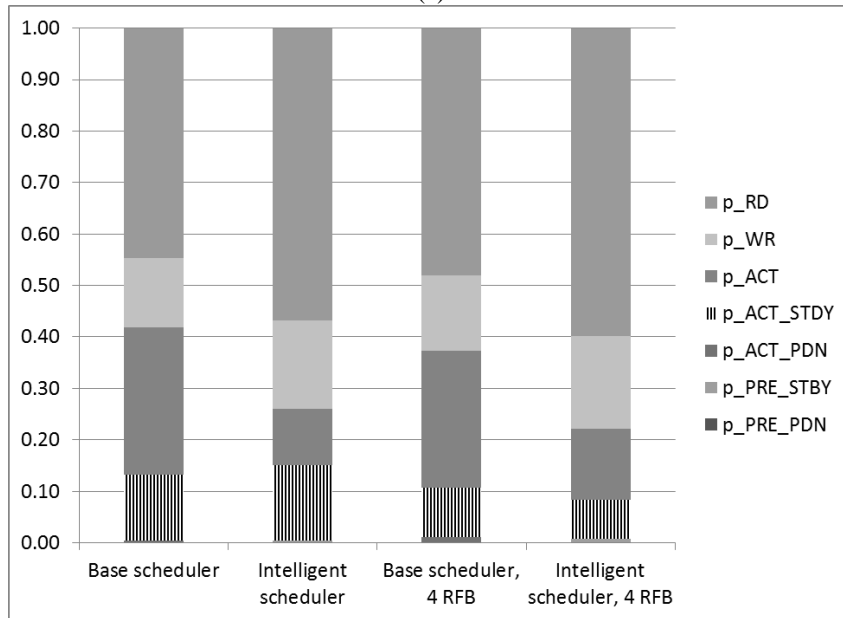


(b)

Fig. 10. Power breakup for doom3 (a) frames 100 to 139 (b) frames 200 to 239

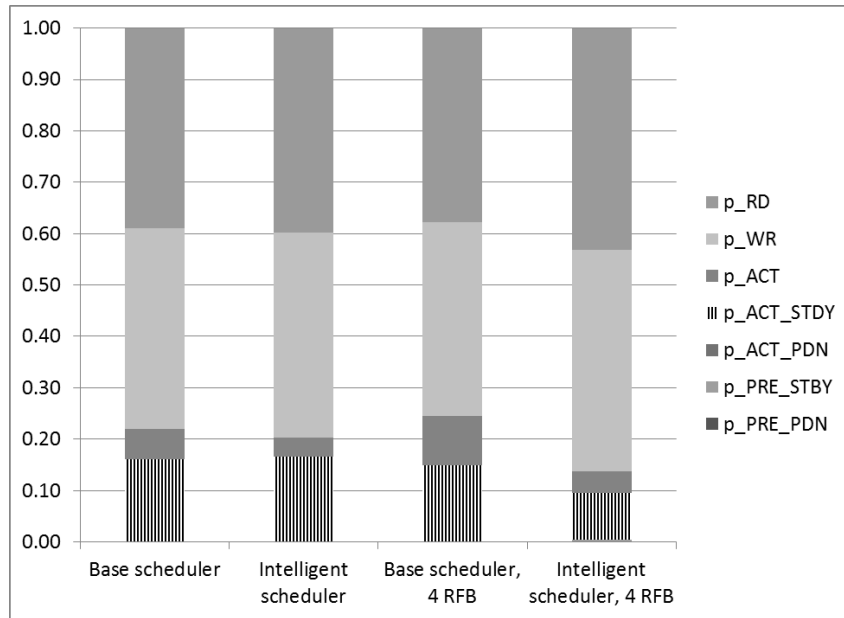


(a)

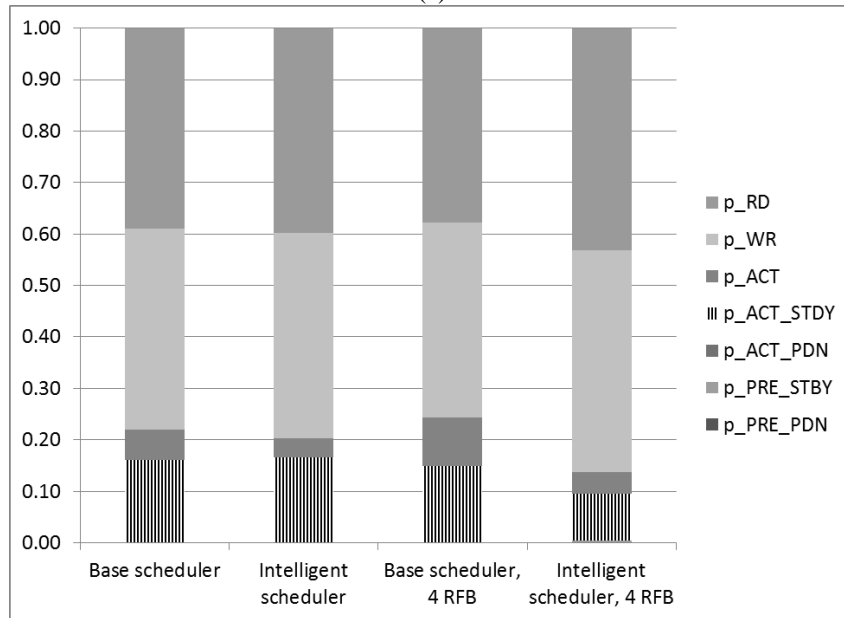


(b)

Fig. 11. Power breakup for Quake (a) frames 100 to 139 (b) frames 200 to 239

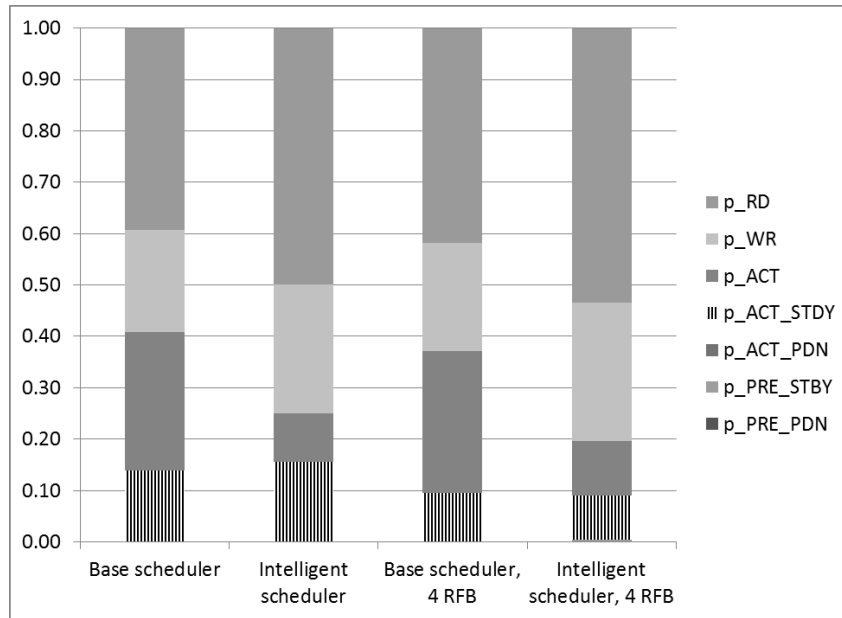


(a)

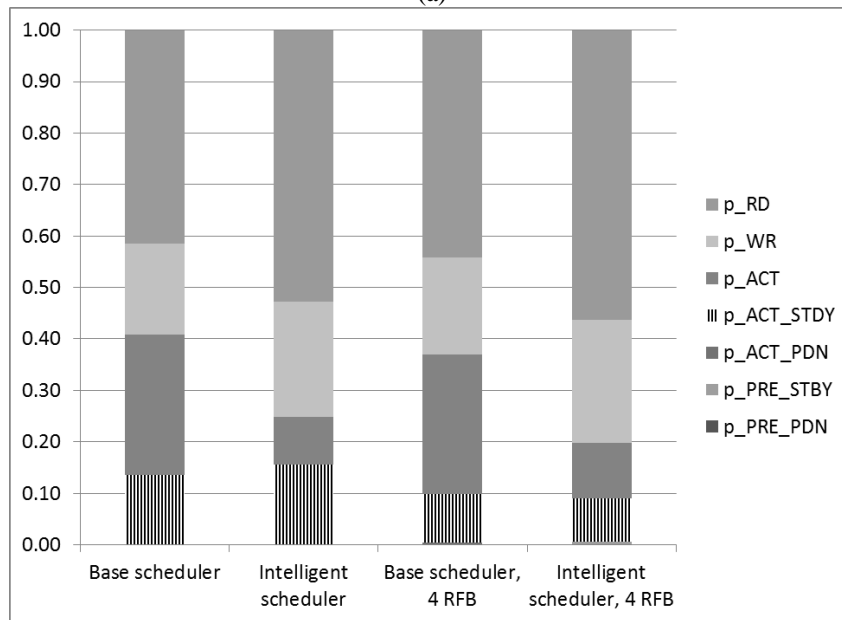


(b)

Fig. 12. Power breakup for Riddick (a) frames 100 to 139 (b) frames 200 to 239



(a)



(b)

Fig. 13. Power breakup for UT2004 (a) frames 100 to 139 (b) frames 200 to 239

8. RELATED WORK

Rixner et al. [2000] proposed reordering and scheduling individual DRAM commands (e.g. activate, precharge, column reads, and column writes) in order to schedule transactions to the same page consecutively and improve throughput. Their work forms the basis of the FR-FCFS scheduler we use in our baseline configuration.

Delaying the issue of DRAM requests in order to improve the page hit rate has been previously proposed by Eckert [2008]. A hold-off counter was used to count the number of cycles to hold off the DRAM requests until a certain threshold was met. Once the threshold was met, the DRAM requests were free to issue. However, since the motivation behind the hold-off counter is performance, and not power, the hold-off counter may not necessarily wait until the transaction queue is full before allowing requests to issue. The reason why is because waiting for the transaction queue to timeout may cause overall throughput

to decrease even though the page hit rate increases. The hold-off is a gamble that the DRAM bandwidth saved by increasing the page hit rate will be greater than the DRAM bandwidth wasted by delaying transactions. In our intelligent scheduler, our motivation is both performance and power. Therefore, we are willing to wait until the transaction queue is full before issuing requests in order to minimize power.

Mutlu et al.[2008] proposed scheduling transactions in groups called batches. However, their heuristic for grouping together their batches is very different from our burst groups. The batches were chosen based on fairness, and so each batch contained a limited number of requests from each thread, in order to allocate a fair amount to each thread. In contrast, our burst groups do not take thread ID into account at all. Burst groups are chosen based on requests to the same page.

There has been much prior work to reduce power consumption in DRAM. Fan et al. [2001] showed that transitioning immediately to the DRAM low power state when the DRAM was idle was better than using a complex predictor to detect DRAM idle periods. Huang et al.[2005] created longer DRAM idle periods, so they could enter DRAM low power states in a DRAM system with multiple ranks by diving ranks into hot ranks and cold ranks. They migrated the most frequently accessed pages to the hot ranks and infrequently accessed pages to the cold ranks. Diniz et al. [2007] limited the maximum power consumption of a system by scheduling DRAM transactions based on how they would affect the power state of a DRAM chip. Zhang et al. [2008] proposed Mini-ranks, which split up ranks into smaller ranks. The mini-ranks reduce the number of DRAM chips involved in each memory access, allowing power to be reduced.

9. CONCLUSION

In this article we proposed an intelligent scheduler to improve throughput per watt in a mobile GPU memory system by delaying the issuing of transactions until the transaction queue was full, and by scheduling bursts in groups to increase the periods when all banks are precharged. By adding MRF-I to our intelligent scheduler to create per bank logical channels, we are able to also increase throughput by an average of 17% up to 66% while increasing throughput per watt by an average of 18% up to 26%. We have shown that using just intelligent scheduling or just using MRF-I to create per bank logical channels is not as good as using both techniques combined to create a low power high throughput mobile GPU memory system.

ACKNOWLEDGEMENTS

This work has been supported in part by the Center for Domain-Specific Computing (CDSC).

REFERENCES

- ATTILA, 2011. ATTLA traces. <http://attila.ac.upc.edu/traceList/>
- BYUN, G., KIM, Y., KIM, J., TAM, S., HSIEH, H., WU, P., JOU, C., CONG, J., REINMAN, G., AND CHANG, M.F. 2011. An 8.4Gb/s 2.5pJ/b mobile memory I/O interface using bidirectional and simultaneous dual (baseband and RF-band) signaling. In *2011 IEEE International Solid-State Circuits Conference (ISSCC) Digest of Technical Papers*, 488-490.
- CADENCE, 2011. Cadence Virtuoso Spectre Circuit Simulator. http://www.cadence.com/products/rf/spectre_circuit/pages/default.aspx
- CHANG, M.F., VERBAUWHEDE, I., CHIEN, C., XU, Z., KIM, J., KO, J., GU, Q., AND LAI, B. 2005. Advanced RF/baseband interconnect schemes for inter- and intra-ULSI communications. In *IEEE Transactions on Electron Devices* 52, 7, 1271-1285.
- DEL BARRIO, V., GONZALEZ, C., ROCA, J., AND FERNANDEZ, A. 2006. ATTLA: a cycle-level execution-driven simulator for modern GPU architectures. In *Int'l Symp. on Performance Analysis of Systems and Software*, 231-241.
- DINIZ, B., GUEDES, D., MEIRA, JR., W., AND BIANCHINI, R. 2007. Limiting the power consumption of main memory. In *Proceedings of the 34th International Symposium on Computer Architecture*, 290-301.
- ECKERT, R.E. 2008. Page Streams Sorter for DRAM Systems. Assignee NVIDIA Corporation, United States Patent 7,376,803.
- FAN, X., ELLIS, C., AND LEBECK, A. 2001. Memory controller policies for DRAM power management. In *Proceedings of the 2001 International Symposium on Low Power Electronics and Design*, 129-134.
- HUANG, H., SHIN, K. G., LEFURGY, C., AND KELLER, T. 2005. Improving energy efficiency by making DRAM less randomly accessed. In *Proceedings of the 2005 International Symposium on Low Power Electronics and Design*, 393-398.
- KO, J., KIM, J., XU, Z., GU, Q., CHIEN, C., AND CHANG, M.F. 2005. An RF/baseband FDMA-interconnect transceiver for reconfigurable multiple access chip-to-chip communication. In *IEEE International Solid-State Circuits Conference (ISSCC) Digest of Technical Papers*, 338-602.
- KUNDERT, K. 1999. Introduction to RF simulation and its application. In *IEEE Journal of Solid-State Circuits* 34, 9, 1298-1319.
- MICRON, 2009. Micron 1Gb: x16,x32 Mobile LPDDR SDRAM Features. http://www.micron.com/products/dram/mobile_lpddram.html
- MUTLU, O. AND MOSCIBRODA, T. 2008. Parallelism-aware batch scheduling: Enhancing both performance and fairness of shared DRAM Systems. In *Proceedings of the 36th International Symposium on Computer Architecture*.
- MOYA, V., GONZALEZ, C., ROCA, J., FERNANDEZ, A., AND ESPASA, R. 2005. Shader performance analysis on a modern GPU architecture. In *Proc. 38th Ann. ACM/IEEE Int'l Symp. Microarchitecture (MICRO '05)*, 355-364.

- RIXNER, S., DALLY, W.J., KAPASI, U.J., MATTSON, P., AND OWENS, J.D. 2000. Memory access scheduling. In *Proceedings of the 27th International Symposium on Computer Architecture*.
- SHAO, J. AND DAVIS, B. T. 2007. A burst scheduling access reordering mechanism," In *Proceedings of the 2007 IEEE 13th International Symposium on High Performance Computer Architecture*, 285-294.
- WANG, D., GANESH, B., TUAYCHAROEN, N., BAYNES, K., JALEEL, A., AND JACOB, B. 2005. Dramsim: a memory-system simulator. In *ACM SIGARCH Computer Architecture News* 33, 4, 100-107.
- ZHENG, H., LIN, J., ZHANG, Z., GORBATOV, E., DAVID, H., ZHU, Z. 2008. Mini-rank: adaptive DRAM architecture for improving memory power efficiency. In *Proceedings of the 41st annual IEEE/ACM International Symposium on Microarchitecture*, 210-221.