ICAP: Designing Inrush Current Aware Power Gating Switch for GPGPU

Hadi Zamani[‡], Devashree Tripathy[‡], Ali Jahanshahi[‡], Daniel Wong[‡] [‡]University of California, Riverside, CA, USA
{hzama001, dtrip003, ajaha004, danwong}@ucr.edu

***Abstract*—The leakage energy of GPGPU can be reduced by power gating the idle logic or undervolting the storage structures; however, the performance and reliability of the system degrades due to large wake up time and inrush current at time of activation. In this paper, we thoroughly analyze the realistic Break-Even Time (BET) and inrush current for various components in GPGPU architecture considering the recent design of multi-modal Power Gating Switch (PGS). Then, we introduce a new PGS which covers the current PGS drawbacks. Our redesigned PGS is carefully tailored to minimize the inrush current and BET. GPGPU-Sim simulation results for various applications, show that, with incorporating the proposed PGS into GPGPU-Sim, we can save leakage energy up to 82%, 38%, and 60% for register files, integer units, and floating units respectively.**

***Index Terms*—Break-Even Time (BET), Power Gating Switch (PGS), Inrush Current**

I. Introduction

A large body of research has been conducted on reducing static power for different types of hardware accelerators including FPGAs, GPGPUs, and AISCs designed for various computation-intensive, latency-sensitive applications such as artificial intelligence algorithms [1], machine learning [2], bioinformatics [3], and automata processing [4]. General Purpose Graphics Processing Units (GPGPUs) with Single-Instruction-Multiple-Thread (SIMT) execution model, in particular, are widely used for massively parallel applications [5]. Recent years have witnessed an increasing trend in the computational capabilities and core count in the GPGPUs which comes at the cost of the increased power consumption [6]. Different components like execution units, register files and caches consume significant portion of the GPU static power [7]. However, due to inter-kernel data dependency [5], inefficient [8] or imbalanced workload distribution [9], branch divergence, irregular memory access patterns and cache contention [10], average utilization of the GPU pipeline components such as register file, Special Function Units (SFU) and Streaming processors (SP) for various GPGPU applications from Rodinia, ISPASS, Cuda SDK and Polybench are 44%, 47%, and 15% respectively [11]. When the circuit is in idle state, there is a dramatic rise in the leakage current due to the exponential nature of the leakage current in the sub-threshold region of the transistor, leading to increased static power [12]. These components can be power gated during idle periods [13], [14].

Previous GPU power gating works assume constant BET of 14 cycles for different GPU components [15], [16]. However, they are based on simulation of a simple circuit and do not reflect the complex components in the GPU pipeline. As a major contribution, using HSpice, we determine the realistic BET of GPU components and measure the energy saving opportunity with the realistic BET. Then using GPGPU-Sim, we compare the energy saving results with other works considering simplistic BET of 14 cycle for every component.

Besides the large wake up time of power gating technique, inrush current is the main drawback of power gating. Any inter-kernel or intra-kernel activity can cause voltage noise in the power delivery network. If the pipeline suddenly becomes active after a stall, it results in an inrush current [17]. **Inrush current is more serious in GPGPUs, compared to CPUs, due to large number of threads which are waiting at the barriers to continue their execution.** As soon as all threads within the thread block finish their execution, all threads start to continue their execution which incur large amount of inrush current. Inrush current also causes voltage fluctuations in the power-delivery network(PDN) and must be dealt with carefully so as to avoid huge voltage droop in the power network. Otherwise, the functionality and the state of the other active units in the PDN (other active cores in the GPU) could be corrupted when the power-gated unit goes through a sequence of deep sleep/active states. Also, the other key challenge in PGS design is managing the inrush current at time of wake-up. Unless the PGS can withstand this high inrush current, the circuit will burn and the target component will be disconnected

[18]. Various circuit level techniques have employed PG to turnoff the idle circuits by creating a high impedance path to ground [19] [20]–[22]. However, not much research has been done to design a reliable PGS for GPGPUs considering the unique inrush current and voltage noise challenges. PGS can be employed for memory units, such as register files and caches, but the contents will be lost due to power gating the storage structure. They must be stored safely in an active memory and transferred back to the registers and caches, which involves significant overheads. To avoid this, the voltage level of these memory elements are simply lowered to drowsy states to save power and retain states which is explained in section IV.

Due to large performance overhead of memory/storage, where the contents of the state-retentive memory structures is lost when power-gated [23], it is under-volted to a low leakage state-retentive drowsy voltage [20]–[22]. In drowsy mode, the information in the SRAM cells are preserved but it is non-functional for read and write accesses.

The prior works assume that the transition time between supply voltage and drowsy voltage is negligible (1-2 cc) [20], [22]. According to our results from simulation of the shared memory and register file, we observe that these voltage transition times are non-negligible and they indeed affect the BET adversely similar to the power gating [24].

There has been some prior works in the PGS design [19], [25], [26], however, our work is the first one to consider the effect of inrush current on the PG switch design in GPUs to the best of our knowledge. An always powered-on buffer is used to generate the sleep signal, thereby incurring large leakage current in drowsy state [25]. Multiple sleep modes are enabled using multiple reference voltages at the source of the sleep transistor in [26], which is power consuming. To reduce the leakage power, multiple sleep modes with low-leakage switch are used in [19].

This work makes **three major contributions:** (1) we analyze the realistic BET of the GPU components through Hspice simulation. We show that these times vary considerably depending on the size, complexity and technology of the components. (2) We explore the design of PG switch considering the BET and inrush current for GPGPUs. (3) We select optimum undervolting level for storage structures, taking into account the voltage transition time, switching energy and static power at each level of voltage, as well as idleness length of storage structure during application execution. We redesign a multi-modal PGS considering the tolerable inrush current as well as reasonable BET.

The rest of the paper is organized a follows. In section II, we model BET and inrush current from other circuit parameters. In section III, we analyze the effect of BET and inrush current on PGS switch design. In section IV, we estimate the optimum under-volting level considering idleness length for storage structures. We evaluate our design in GPGPU-Sim for leakage power-savings in Register file, SP-Int and SP-Float units in Section V. And finally, conclusion and remarks are provided in section VI.

II. Power Gating Analysis

First, we explore opportunities of reducing the leakage/static power in each and every stage of the pipeline. Static power consumption of the GPU components are extracted using GPUWattch for GPU GTX 480 [27]. The Static power breakdown is shown in Table I. Execution units (including FPU and SFU) and the storage structures (like Register Files and Shared Memory) consume 21% and 46% of the total Streaming Multiprocessor (SM) static power respectively.

As shown in the Fig. 1, PG switch is placed between power supply and the target circuit. Note that transition times encountered during the switch off and on operations, determine whether or not, energy is saved during the PG operation. So, BET, which is defined as the minimum time that the PG switch must be disabled to save energy, is the key point which is discussed in the following.

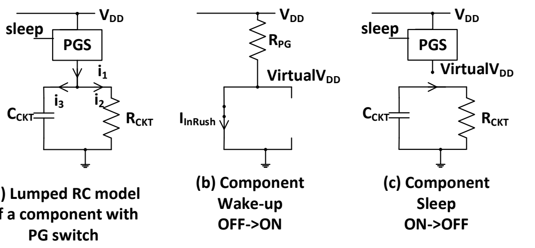
A. *Break-Even Time Estimation Method*

BET is defined as the minimum time period which the target circuit should sleep in-order to save energy. The BET of the GPU components is estimated using Hspice simulation of the lumped RC model of the target component as shown in Fig. 1. In Fig. 1(a), C_{CKT} and R_{CKT} represent equivalent capacitance and resistance of the GPU components respectively. These values are extracted by reverse engineering the GPUWattch.

As shown in Fig. 1, when the PG switch is ON, the capacitor C_{CKT} charges up through the PG switch that is represented by R_{PG} (Fig. 1b). The current through PG switch, R_{CKT} and C_{CKT} are i_1, i_2 and i_3 respectively and can be represented by the following equations:

TABLE I: Static power breakdown of different components [27]

GPU Pipeline Units	**Static Power (%)**		
Floating Point Unit	17	**Cache**	3
Special Function Unit	4	**Shared Memory**	14
Register File	32	**Instruction Decoder**	9
Instruction Fetch Unit	5	**Other**	16



sleep
PGS
V_{DD}
i_1
i_3
i_2
C_{CKT}
R_{CKT}
R_{PG}
VirtualV_{DD}
I_{InRush}
(a) Lumped RC model of a component with PG switch
(b) Component Wake-up OFF->ON
(c) Component Sleep ON->OFF

Fig. 1: Lumped RC model of a GPGPU component

$$i_2(t) = \frac{V_{DD}}{R_{PG}+R_{CKT}}(1-e^{\frac{-\beta t}{C_{CKT}}}), \beta = \frac{R_{PG}+R_{CKT}}{R_{PG}*R_{CKT}} \quad (1)$$

$$i_3(t) = \frac{V_{DD}}{R_{PG}}(e^{\frac{-\beta t}{C_{CKT}}}) \quad (2)$$

The time constant ($\tau_{charging}$) for charging the capacitor is $\frac{C_{CKT}}{\beta}$. Where β is measured according to Eq. 1

As shown in Fig. 1(c), when the circuit block is idle, a "sleep" signal is applied to the PGS input signal so that the virtual V_{DD} is set to 'Zero'; and subsequently the target circuit is turned off (sleep mode). When the PG switch is OFF, the target component is put to sleep mode by discharging the C_{CKT} through R_{CKT}. The time constant for discharging the capacitor is $\tau_{discharging}$ = $C_{CKT} * R_{CKT}$. Alternatively, while waking up the target circuit, virtual V_{DD} is set to V_{DD}. Waking up the power gated component, incurs the energy overhead during the transition time from sleep to active mode.

Fig. 2 shows the energy consumption of the circuit block during the PG interval. At $t_{breakeven}$, E_{saved} = $E_{overhead}$; and the values of E_{saved} and $E_{overhead}$ are obtained from equations 3 and 4 respectively, where P_{max} is the static power of target circuit at the supply voltage. Moreover, $t_{detect} = t_1$ is the time taken by the control circuit to make a decision to power gate the target circuit; Finally, $t_{fall} = t_3 - t_2$ is the transition time to turn-off and $t_{rise} = t_5 - t_4$ is the transition time to wake-up the target circuit [28]. $E_{switching}$ is a function of P_{max}, t_{detect}, t_{fall} and t_{rise}.

$$E_{saved} = P_{max} * t_{breakeven} \quad (3)$$

$$E_{overhead} = E_{switching} = f(P_{max}, t_{detect}, t_{fall}, t_{rise}) \quad (4)$$

$$t_{breakeven} = \frac{E_{switching}}{P_{max}} \quad (5)$$

Using Hspice, switching energy, wake-up time, fall time, and static power of the GPU components at a given voltage is measured using 45nm technology which is discussed in Section III. We use 45nm technology because GPGPU-Sim simulator which is based on 45nm technology, will be used to estimate the impact of redesigned PGS for different application.

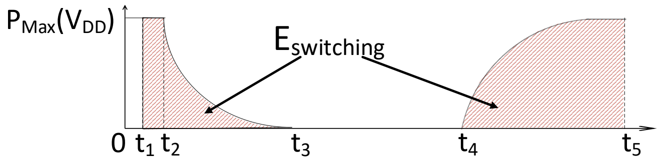


Fig. 2: Circuit block state transition during power gating interval

B. Inrush Current

Inrush current is the high instantaneous current drawn by the electrical circuit when powered on. When the circuit is switched on from the power gated state, the capacitor acts as a short circuit, i_1 = i_3 = $\frac{V_{DD}}{R_{PG}}$ and i_2 = 0 as shown in Fig. 1(b).

As of now, we define the power gated and active components as downstream and upstream components. The PGS switch ensures to wake up the downstream component without disrupting the components functioning on the upstream domain. When the power gated component is abruptly waked-up, current draws from the upstream side to the downstream side. If a low-resistance power gate switch is used between the upstream and downstream components, charge sharing occurs in less than 1 nsec and there is not sufficient time to charge the downstream side from the external PDN [18]. This is because, the package inductance is high impedance and it blocks the current from the outside for a short period of time. And the charge in the components which share the same power is immediately shared with downstream components. Assuming half of the SP units within the SM are active and half of them are power gated. when the power gated components wake up, the equivalent capacitance of upstream and downstream is same; $C_{active} = C_{pg}$. So, the voltage drops in the active components in upstream domain. According to equations 7, the upstream domain voltage abruptly cut is half.

$$\text{Q}= \text{C}_{up} * V_{dd} \quad (6)$$

$$\text{V}_{dd_a} = \frac{Q}{C_{up}+C_{down}} = \frac{C_{up}*V_{dd_b}}{C_{up}+C_{down}} = V_{dd_b} * \frac{C_{up}}{C_{up}+C_{down}} = \frac{V_{dd_b}}{2} \quad (7)$$

As illustrated in Equation 8, I_{inrush} is the amount of inrush current due to the capacitance C_{load} and dV is the change in voltage during ramp up and dt is the rise time of the input voltage signal during ramp up. If the inrush current is not addressed, then it might lead to damages to circuit components. As the current exceeds current handling capacity of the components, causes voltage drops and the circuit will fall out of regulation resulting in the system entering a faulty state. Hence, PG switch must be designed to withstand the high inrush current and control the voltage noise on the adjacent circuit blocks which share the same power lane.

$$I_{inrush} = C_{load} * dV/dT \tag{8}$$

Inrush current is inversely proportional to the wake-up time of the component. So, in order to reduce the inrush current, the wake-up time of the power gated component/s need to be increased which degrades the performance. So, there is a trade-off between amount of inrush current and BET. We address both BET and inrush current in redesigning the PG switch in Section III.

III. GPGPU Power Gating Switch Design

As discussed earlier, we power gate the logics and undervolt the storage structures. Several researches have designed PGS or voltage regulator without considering their impact on the BET and inrush current [29], [30]. Inrush current plays an important role in GPUs because of higher number of core which can be power gated or activated at the same time due to barrier synchronization. We simulate the impact of widely used PGS on switching energy, wake-up time, and inrush current considering the GPGPU components [19]. Then we relax its drawbacks by addressing these issues with redesigning the PGS with considering the GPGPU limitations.

The designed PGS can power gate the logics or make the storage subsystems drowsy based on the input control signals. It can operates in active, sleep, and drowsy modes. Different levels of voltages can be generated and applied at the target circuit. If we use PGS, as footer cell, we should use NMOS transistor between circuit block and ground. NMOS has higher leakage current in comparison to PMOS. So, the sleep mode can not be maintained while we are using the PGS as the footer cell. So, we use PMOS transistor in PGS as the header cell.

As shown in Fig. 3, if SLEEP = 0, regardless of the value of the DROWSY signal, the MS1 and MS transistors are ON which generates V_{dd} at virtual V_{DD} and the logic circuit will be in active mode. When SLEEP = 1, and DROWSY = 0, the circuit block starts to discharge through ground since it's not connected to VDD. Finally, when SLEEP = 1, and DROWSY = 1, the output of the sleep inverter will be drowsy voltage. There will be a negative feedback loop which generates virtual V_{DD} at VVDD. The amount virtual V_{DD} is depending on the transistor width of MD1 and MS2.

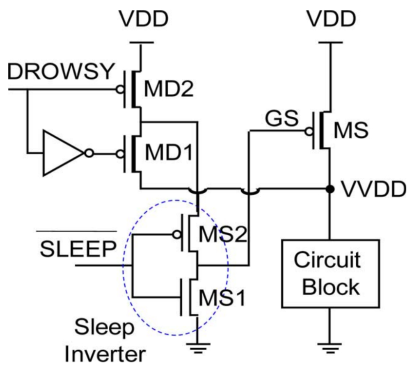


Fig. 3: Power Gating Switch Design [29]

Correct sizing of different transistors in the PGS has direct impacts including logic gate switching speeds in the active mode, leakage currents in sleep, and drowsy modes, wake up latency, and area overhead. For instance, as shown in Fig. 3, larger MS size results in higher active mode switching speeds but also increases sleep and drowsy leakage currents in active mode. In previous researches, they have not investigated the relation of PGS and BET.

We redesign the PGS to consider the inrush current as well as BET of the GPGPU components. By modeling the target circuit with lumped RC model of the circuit, we find the rise/fall times, switching energy and as a result, the BET of the target circuit.

To extract these parameters for the GPU main components, the equivalent resistance and capacitance of target components such as register file bank, integer unit, floating point units, and etc, are measured according to Section III-A. Then, as shown in Fig. 1, we construct the PDN model of each component and estimate the rise/fall times, switching energy and inrush current of the GPU components.

A. Equivalent Capacitance and Resistance

The PDN model of each component is created using capacitance (C) and resistance (R) of the component obtained using McPAT simulator, as in Eq. 9 and 11 respectively, where $P_{switching}$ is the dynamic switching power. V_{dd} is the supply voltage which is 1V, and f is the frequency considered 750 MHz in GTX 480. $I_{short-circuit}$ is the short-circuit current and occurs when PMOS and NMOS devices become on simultaneously ON and I_{peak} is the maximum value of the short circuit current. In Eq. 11, I_{peak} is derived as $P_{switching}$/V_{dd} when the activity factor (α) is 1. Short-circuit power ($P_{short-circuit}$ = V_{dd} * $I_{short-circuit}$) is modelled in Eq. 10 and is the power consumed when both pull-up and pull-down devices are partially on for a small finite amount of time [31].

By solving the mentioned equations, we extract the R and C values for the GPU pipeline components including register file bank, shared memory, floating point unit, integer unit, and instruction fetch unit. C and R values corresponded for storage structures are extracted at each bank level. This is because each bank can be power gated separately since they do not share the power lines and we can have fine granularity power gating for register file.

$$P_{switching} = \alpha * C_L * V_{dd} * V_{dd} * f \tag{9}$$

$$P_{short-circuit} = V_{dd} * I_{peak} * \frac{rise + fall}{2} * f \tag{10}$$

$$I_{short-circuit} = 5 * I_{peak} * R * C_{sc} * f \tag{11}$$

B. Break-Even Time Analysis

Prior researches assumed a BET of 14 cycles irrespective of the underlying circuit [28] [16]. The rise/fall times of different components are extracted using 45 nm technology. In our simulation, channel width and length of the transistor are assumed 3 uM and 1.5 uM respectively according to [19]. The rise/fall times for floating point, integer, and instruction fetch units are extracted according to lumped RC model while the rise and fall time of Register File, and Shared Memory are extracted using Hspice simulations. According to the simulation results which are discussed in the following section, rise time of the circuit blocks does not follow the analytical model anymore and shows an aggressive increment. There is a huge difference between analytical and measurement results with and without using the PGS. Previous researches have not considered the effect of PGS on the BET of the component. For simple components with lower capacitance and resistance, the performance overhead of switch is negligible. But with increasing the capacitance and resistance of the component, the switch transistor plays an important role in BET of the component.

Using Hspice, we also find the peak power (P_{peak}) and switching energy(E_{switch}) which is shown in Table II. The switching energy divided by the peak power adds tens of cycles to the BET of the GPU components as per equation 5. We aim to optimize the impact of PGS on the switching energy as well as the wake-up time that are key factors in the BET as illustrated in equation 5.

TABLE II: Peak Power and Switching Energy With and Without optimizing the PG switch

	No Optimization		Optimized PGS	
GPU Component	$\mathbf{P_{peak}}$ **(uW)**	$\mathbf{E_{switch}}$ **(pJ)**	$\mathbf{P_{peak}}$ **(uW)**	$\mathbf{E_{switch}}$ **(pJ)**
Reg File Bank	750.86	87.01	15	0.01
Shared Mem	752.97	55.34	12	0.01
FPU	825.72	242.99	18	0.06
Integer Unit	435.14	63.84	12	0.02
Instruction Fetch	22.14	14.55	14	0.003

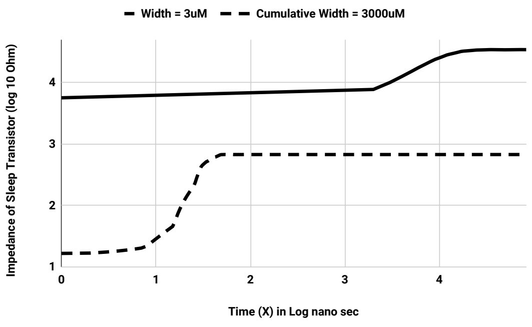


Fig. 4: The impedance of sleep transistor during the wake-up time using two different widths

C. Optimizing the BET

As illustrated in Table III, with current PGS design with the original transistor widths, the BET is a very large number. This is because, PG switch uses a transistor with small channel width which is considered as high resistive transistor and it adds to the equivalent resistance of charging path that results in less current or higher wake-up time.

The equivalent impedance of the PG switch varies during the different operational modes of the transistor [32]. The operation of a MOSFET can be separated into three different modes, depending on the voltages at the terminals (V_{gate}, V_{source} and V_{drain}). There are three operational modes: cutoff, triode, and saturation mode. [32] simplifies the transistor as an impedance which varies at different operational modes. As shown in Fig. 3, during the wake-up time or charging period, MS1 and MS transistors experience different impedance at saturation and triode mode.

1) Impedance Analysis of the Sleep Transistor Switch: Using Hspice, we measure impedance of the sleep transistor during the wake-up time. To measure this value, we model the GPU component with a lumped RC model as illustrated in Fig. 1. Fig. 4 shows the impedance of MS transistor during the wake-up time. Fig. 4 shows the results in logarithmic scale. It's obvious that, the impedance of the transistor decreases aggressively with increasing the channel width of MS transistor. To simplify the PG sleep transistor and extract the main factors that affect the impedance, PG switch is modeled as a resistor with an impedance that is given in Eq. 12 [32]. Where k_n and $\frac{W}{L}$ are the process-trans-conductance and width to length ratio of the sleep transistor, respectively. According to Eq. 12, changing the transistor width changes the impedance of the switch transistor.

$$R = \frac{1}{k_n(\frac{W}{L})(V_{GS} - V_{thn})} \tag{12}$$

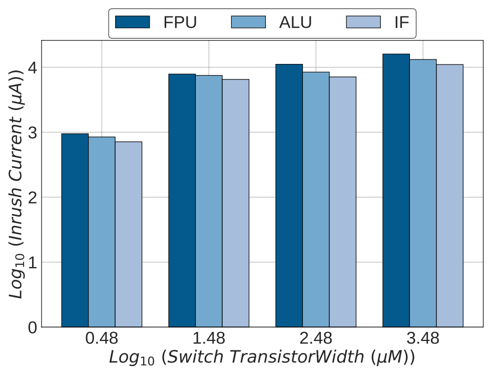


Fig. 5: Inrush current in presence of different switch transistor widths

To reduce the wake-up time and as a result BET, sleep transistor width is increased according to Eq. 12. Using Hspice, first, we measure the effect of width on the BET. By increasing width of the switch transistor channel, the impedance of the the switch transistor is reduced in different operational modes which leads to less impedance during the charging period and as a result lower BET. In this case, we need to use a switch transistor with a large width that is not practical.

Instead of using one transistor with large width, we are using several switch transistors which are activated sequentially. When these transistors activate, they have the lowest equivalent resistance and draw maximum amount of current. So, we use few number of transistors with small channel widths which give us the same BET equal to the estimated larger transistor. With optimizing the channel width of transistors, we also decreased the switching energy of each component. Table II shows the switching energy and peak power of the GPU components. Previously, with the default transistor width, the switching energy was playing an important factor in the BET of the components [28]. But, as shown in Table II, the switching energy is decreased sensibly and as a result it's contribution in the BET is not sensible anymore.

Table III shows the rise and fall time of the GPU pipeline main components with none optimized PGS that uses the default transistor width and optimized PGS which employs enough number of small transistors to drive the sufficient current into the circuit blocks in a reasonable time. As shown in the table, the rise and fall time of the circuit blocks are sensibly decreased so that power gating and undervolting can be applied now.

TABLE III: Rise and fall time of the GPU components in presence of none optimized PGS and optimized PGS

GPU Component	None Optimized	Optimized
	Rise + Fall time (ns)	Rise+Fall time (ns)
Register File bank	69	15
Shared Memory	48	12
Floating Point Unit	457.4	18
Integer Unit	98	12
Instruction Fetch Unit	25	4

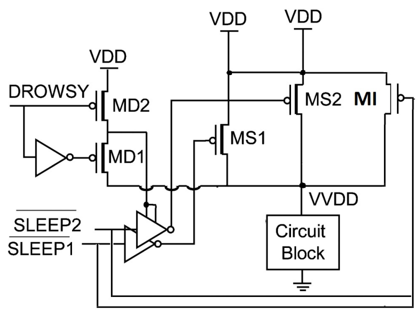


Fig. 6: Modifying the PGS to relax the inrush current without changing the BET

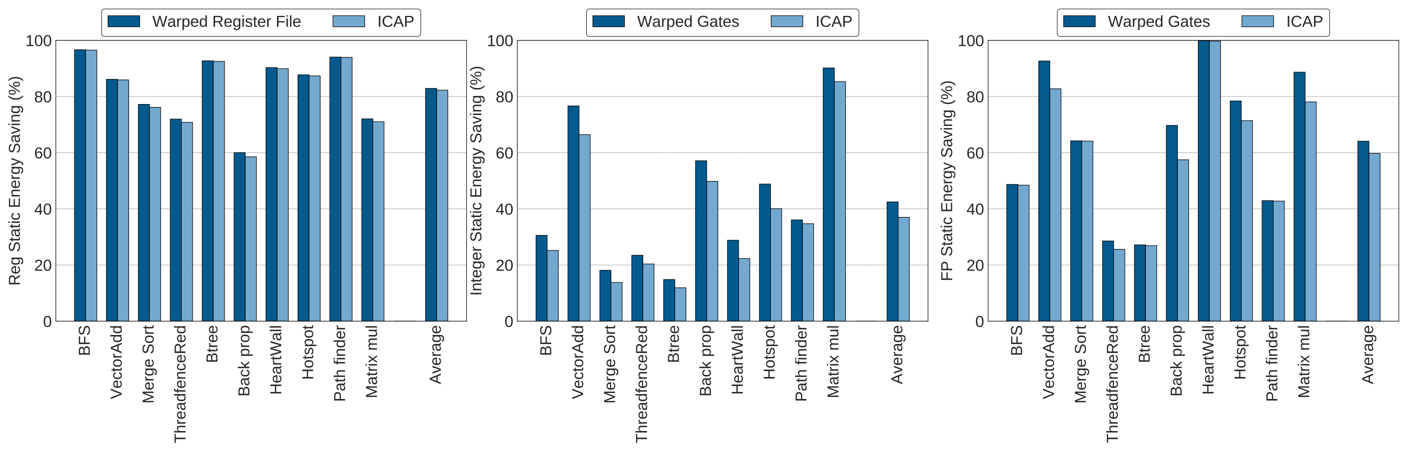
D. *Relaxing The Inrush Current*

Fig. 5 shows the inrush current for GPU components such as Floating Point, Instruction Fetch (IF), and Integer Unit. The amount of inrush current increases sensibly with employing the cumulative larger transistor widths. As shown in Fig. 6, we address this problem by adding a transistor "MI" with small width in parallel with sleep transistor. At time of wake-up, MI transistor wakes up earlier than MS1/MS2 transistor since the sleep signal is directly connected to the MI transistor but takes more time to change the GS signal to 0 volt, because, first, transistors inside the sleep inverter should be activated which incurs a delay to change of input signal of MS1/MS2. So, there would be a delay between activation of MS1/MS2 transistor and MI transistor. MI transistor has small width and as a result larger impedance according to Eq. 12. MI transistor leads to less inrush current, because, by the time MS transistor is activated, the circuit block charge is not 0V anymore and hence the equivalent capacitor of the component does not perform as short circuit.

The amount of inrush current depends on the transistor width of MI transistor. Smaller width leads to lesser inrush current. With default transistor width, the inrush current does not change due to resizing the original design.

IV. Undervolting the Storage Structures

The optimum level of undervolting is decided by considering the static power at each level, transition latency and switching energy between different power modes which is extracted through Hspice simulation. Table IV, shows static power (P_{static}), switching energy ($E_{switching}$), transition latency ($T_{wake-up}$) from each voltage level till the nominal voltage (1 V) and BET. BET time corresponding to each voltage



(a) Register Energy Saving (b) Integer Energy Saving (c) Floating Point Energy Saving

Fig. 7: Leakage energy savings for registers, integer and floating point unit with oracle knowledge

is calculated using Equation 5. Based on the idleness length of the register file bank, the optimum level of voltage is selected. According to our observations and for simplicity of the PGS design, it is sufficient to use only 3 voltages called ”shallow sleep = 0.7 V”, ”deep sleep = 0.3 V”, and ”active state = 1 V”. Hence, storage structures are switched to a state-retentive low power mode to have maximum energy savings without compromising the performance. In our approach, GPGPU pipeline storage structures are connected to the supply voltage (V_{dd}) rail via the multi-modal power gating switch in Figure 6.

The multi-modal PGS shown in Figure 6 is used to generate different levels of voltage across the memory subsystem. As shown in Fig. 6, PGS employs two inverters, which, only one of them is active at a time. They have different channel widths, so that they can generate two levels of voltage at virtual V_{DD}. The channel widths are extracted through Hspice simulations. Finding the optimum level of voltage depends on the performance overhead of transition between different low-power states. In low power mode, storage structure should be state-retentive, such that during the low power mode, memory contents are preserved. The conservative data-retention voltage used is 0.3V for the register file file bank [20]. PGS illustrated in Figure 6 is able to generate V_{dd}, ’0’, ’shallow sleep= 0.7 ’, and ’deep sleep = 0.3’. We can generate more number of

TABLE IV: Determining optimum undervolting level for register file

Voltage	$P_{static}(uW)$	$E_{switch}(nJ)$	$T_{wake-up}(ns)$	$BET(nS)$
0.3	1.77	27.8	12.83	15.78
0.4	2.02	28.33	11.61	14.05
0.5	2.4	29.87	10.39	12.45
0.6	2.75	31.01	9.17	11.3
0.7	3.2	31.34	7.92	9.72
0.8	3.6	28.61	6.54	7.92
0.9	3.97	25.93	3.78	6.54

voltage levels with adding MS transistor with different channel width.

V. Evaluation

We have implemented Inrush Current Aware Power Gating Switch (ICAP) in GPGPU-Sim v3.2.1 [11], based on an Nvidia Fermi-like GPU configuration with 15 SMs. Each SM has a 128 KB register file organized into four banks, and 2 SPs. We enabled PTXPlus for all of our evaluations. Each SM also has two warp schedulers configured to use a two-level warp scheduler. In all experiments, we use 10 benchmarks selected from Rodinia, Nvidia Cuda-SDK, ISPASS, and Polybench benchmark suites. The benchmarks cover a range of behaviors and instruction mixes (load store/integer/floating point). The *Streaming Processor(SP)* comprises of the integer and floating point pipeline which do not support concurrent execution. So, one unit is likely to remain idle when the other unit is active. Each Integer and Floating point unit is connected to a multi-modal switch to switch between different voltage levels depending on the predicted idle period length.

Fig. 7, shows the comparison of maximum Leakage energy savings of the Inrush Current Aware Power Gating Switch (ICAP) with the state-of-art leakage energy saving techniques for GPU register file, integer, and FP unit [20].

Since the average register reuse distance is in order of 100 clock cycles, the increased wake-up latency from drowsy to ON state results in marginal differences in the leakage energy savings. However, in case of the SP unit, frequency of the shorted idle period (less than 20 cc) comprises of over 50% of the total idle cycles. Hence changing the BET from 14 cc to 20 cc in case Integer and 14 cc to 47cc in the FP unit results in reduced leakage energy saving of 6% and 5% in the Integer and FP unit respectively.

VI. Conclusion

In this paper, realistic BET of the GPU pipeline components which consume significant portion of the total static power was analyzed in presence of the PGS. We observed that, BET differs from the values stated by the previous researches. We optimised the BET by increasing the channel width of transistor. Still this change, results in higher inrush current at time of wake-up which can degrades the reliability of system. We proposed a new design to reduce the inrush current without changing the BET. Also, analyzing the realistic BET for storage structures, we estimated the optimum level of voltage for a given idleness length.

References

[1] A. Jahanshahi, M. K. Taram, and N. Eskandari, "Blokus duo game on fpga," in *International Symposium on Computer Architecture & Digital Systems (CADS)*, 2013.

[2] A. Jahanshahi, "Tinycnn: A tiny modular cnn accelerator for embedded fpga," *arXiv preprint arXiv:1911.06777*, 2019.

[3] L. Wu, R. Sharifi, M. Lenjani, K. Skadron, and A. Venkat, "Sieve: Scalable in-situ dram-based accelerator designs for massively parallel k-mer matching," in *International Symposium on Computer Architecture (ISCA)*, 2021.

[4] J. Wadden, T. Tracy, E. Sadredini, L. Wu, C. Bo, J. Du, Y. Wei, J. Udall, M. Wallace, M. Stan *et al.*, "Automatazoo: A modern automata processing benchmark suite," in *International Symposium on Workload Characterization (IISWC)*, 2018.

[5] A. Abdolrashidi, H. A. Esfeden, A. Jahanshahi, K. Singh, N. Abu-Ghazaleh, and D. Wong, "Blockmaestro: Enabling programmer-transparent task-based execution in gpu systems," in *International Symposium on Computer Architecture (ISCA). IEEE*, 2021.

[6] H. Zamani, Y. Liu, D. Tripathy, L. Bhuyan, and others, "GreenMM: energy efficient GPU matrix multiplication through undervolting," *Proceedings of the ACM*, 2019.

[7] H. Zamani, D. Tripathy, L. Bhuyan, and Z. Chen, "SAOU: safe adaptive overclocking and undervolting for energy-efficient GPU computing," in *International Symposium on Low Power Electronics and Design*, ser. ISLPED '20. New York, NY, USA: Association for Computing Machinery, Aug. 2020, pp. 205–210.

[8] A. Jahanshahi, H. Zamani, C. Lau, and D. Wong, "Gpu-nest: Characterizing energy efficiency of multi-gpu inference servers," *IEEE Computer Architecture Letters*, vol. 19, no. 2, pp. 139–142, 2020.

[9] A. Abdolrashidi, D. Tripathy, M. E. Belviranli, L. N. Bhuyan, and D. Wong, "Wireframe: Supporting data-dependent parallelism through dependency graph execution in gpus," in *International Symposium on Microarchitecture*, 2017, pp. 600–611.

[10] D. Tripathy, A. Abdolrashidi, L. N. Bhuyan, L. Zhou, and D. Wong, "Paver: Locality graph-based thread block scheduling for gpus," *ACM Transactions on Architecture and Code Optimization (TACO)*, 2021.

[11] A. Bakhoda, G. L. Yuan, W. W. Fung, H. Wong, and T. M. Aamodt, "Analyzing cuda workloads using a detailed gpu simulator," in *Performance Analysis of Systems and Software, 2009. ISPASS 2009.*

[12] R. G. Dreslinski, M. Wieckowski, D. Blaauw, D. Sylvester, and T. Mudge, "Near-threshold computing: Reclaiming moore's law through energy efficient integrated circuits," *Proceedings of the IEEE*, 2010.

[13] O. Kayiran, A. Jog, A. Pattnaik, R. Ausavarungnirun, X. Tang, M. T. Kandemir, G. H. Loh, O. Mutlu, and C. R. Das, "c-states: Fine-grained gpu datapath power management," in *2016 International Conference on Parallel Architecture and Compilation Techniques (PACT)*, 2016, pp. 17–30.

[14] D. Tripathy, H. Zamani, D. Sahoo, L. N. Bhuyan, and M. Satpathy, "Slumber: static-power management for GPGPU register files," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, ser. ISLPED. Association for Computing Machinery, 2020.

[15] M. Abdel-Majeed, D. Wong, and M. Annavaram, "Warped gates: gating aware scheduling and power gating for gpgpus, micro 2013."

[16] M. Sadrosadati, , and O. Mutlu, "Itap: Idle-time-aware power management for gpu execution units," *(TACO)*, 2019.

[17] J. Leng, Y. Zu, M. Rhu, M. S. Gupta, and V. J. Reddi, "Gpuvolt: Modeling and characterizing voltage noise in gpu architectures," in *International Symposium on Low Power Electronics and Design (ISLPED)*, 2014.

[18] K. Jeong, A. B. Kahng, S. Kang, T. S. Rosing, and R. Strong, "Mapg: Memory access power gating, date 2012," in *2012 Design, Automation Test in Europe Conference Exhibition (DATE)*.

[19] E. Pakbaznia and M. Pedram, "Design and application of multimodal power gating structures," in *2009 10th International Symposium on Quality Electronic Design*, March 2009, pp. 120–126.

[20] M. Abdel-Majeed and M. Annavaram, "Warped register file: A power efficient register file for gpgpus," in *International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2013.

[21] Y. Wang, S. Roy, and N. Ranganathan, "Run-time power-gating in caches of gpus for leakage energy savings," in *2012 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2012, pp. 300–303.

[22] K. Flautner, N. S. Kim, S. Martin, D. Blaauw, and T. Mudge, "Drowsy caches: simple techniques for reducing leakage power," in *ACM SIGARCH Computer Architecture News 2002*.

[23] R. Sharifi and Z. Navabi, "Online profiling for cluster-specific variable rate refreshing in high-density dram systems," in *2017 22nd IEEE European Test Symposium (ETS)*. IEEE, 2017, pp. 1–6.

[24] S. T. Publications. [Online]. Available: www.synopsys.com

[25] S. Kim, S. Kim, and S. V. Kosonocky, "Experimental measurement of a novel power gating structure with intermediate power saving mode," in *International symposium on Low power electronics and design, 2004.*

[26] K. Agarwal, H. Deogun, D. Sylvester, and K. Nowka, "Power gating with multiple sleep modes," in *ISQED'06*, March 2006.

[27] J. Leng, T. Hetherington, A. ElTantawy, S. Gilani, N. S. Kim, T. M. Aamodt, and V. J. Reddi, "Gpuwattch: Enabling energy optimizations in gpgpus," *SIGARCH Comput. Archit. News 2013.*

[28] Z. Hu, A. Buyuktosunoglu, V. Srinivasan, V. Zyuban, H. Jacobson, and P. Bose, "Microarchitectural techniques for power gating of execution units," ser. ISLPED '04.

[29] E. Pakbaznia and M. Pedram, "Design and application of multimodal power gating structures, isqed 2009."

[30] B. Gopireddy, C. Song, J. Torrellas, N. S. Kim, A. Agrawal, and A. Mishra, "Scalcore: Designing a core for voltage scalability," in *HPCA*, 2016.

[31] H. J. Veendrick, "Short-circuit dissipation of static cmos circuitry and its impact on the design of buffer circuits," *IEEE Journal of Solid-State Circuits*, 1984.

[32] S. M. Sharroush, "Representing the transistor by an equivalent resistor," in *International Conference on Electronic Devices, Systems and Applications (ICEDSA)*. IEEE, 2016.