FlashPower: A Detailed Power Model for NAND Flash Memory

Vidyabhushan Mohan Sudhanva Gurumurthi
Department of Computer Science
University of Virginia, Charlottesville, VA 22904
{mohan,gurumurthi}@cs.virginia.edu

Mircea R. Stan
Department of Electrical and Computer Engg
University of Virginia, Charlottesville, VA 22904
mircea@virginia.edu

***Abstract*— Flash memory is widely used in consumer electronics products, such as cell-phones and music players, and is increasingly displacing hard disk drives as the primary storage device in laptops, desktops, and even servers. There is a rich microarchitectural design space for flash memory and there are several architectural options for incorporating flash into the memory hierarchy. Exploring this design space requires detailed insights into the power characteristics of flash memory. In this paper, we present *FlashPower*, a detailed analytical power model for Single-Level Cell (SLC) based NAND flash memory, which is used in high-performance flash products. We have integrated *FlashPower* with CACTI 5.3, which is widely used in the architecture community for studying memory organizations. *FlashPower* takes as input device technology and microarchitectural parameters to estimate the power consumed by a flash chip during its various operating modes. We have validated *FlashPower* against published chip power measurements and show that they are comparable.**

I. Introduction

Flash is the most popular solid-state memory technology used today. Flash memory is widely used in consumer electronics products, such as cell-phones and portable music players, and flash-based solid-state disks (SSDs) are increasingly displacing hard disk drives as the storage of choice in laptops, desktops, and even servers. While most research on flash has focused on the device technology and the circuit-level design of flash chips [2], or on high-level system issues such as file-system and Flash-Translation Layer design [4], there has been growing interest in the computer architecture community on flash memory. Computer architects have begun exploring a variety of topics related to flash, including the design of SSDs [3], disk-caches [8], and even new flash-based server architectures [1]. In order to study this architecture design space, architects require simulation tools that can provide detailed insights into the behavior of different flash memory organizations. In particular, a tool that provides an accurate estimate of the power consumed by various flash memory organizations is necessary. To the best of our knowledge, there is no such publicly available tool.

In this paper, we present *FlashPower*, a detailed analytical power model for Single-Level Cell (SLC) NAND flash memory chips, which are used in high-performance flash-based architectures. *FlashPower* models the key components of a flash chip during the read, program, and erase operations and when idle and is parameterized to facilitate the exploration of a wide spectrum of flash memory organizations. We have integrated *FlashPower* with CACTI 5.3 [17], which is a widely used tool in the architecture community for studying memory organizations, and is suitable for use in conjunction with an architecture simulator. We validate *FlashPower* against published chip power measurements [5] and show that the power estimations provided by our tool are comparable to the measured values.

The organization of the rest of this paper is as follows. The next section provides an overview of the microarchitecture and operation of NAND flash memory and Section III presents the details of the power model. The validation of *FlashPower* is given in Section IV and Section V concludes this paper.

II. The Microarchitecture of NAND Flash Memory

A. Components of NAND Flash Memory

Flash is a type of EEPROM (Electrically Erasable Programmable Read-Only Memory) that supports read, program, and erase as its basic operations. The main component of a NAND flash memory chip is the flash memory array. Flash memory array is organized into banks (referred to as *planes*). The structure of a plane is shown in Figure 1. Each plane has a page buffer (composed of sense-amplifiers and latches) which senses and stores the data to be read from or programmed into a plane. Each plane is physically organized as *blocks* and the blocks are composed of *pages*. Thus a plane is a two dimensional grid composed of rows (bit-lines) and columns (word-lines). At the intersection of each row and column is a Floating Gate Transistor (FGT) which stores a logical bit of data. In this paper, the terms “cell” and “FGT” refer to the same physical entity and are used interchangeably.

A page is the smallest granularity of data addressable by the controller and corresponds to one row of a plane. Each NAND flash block consists of a *string* of FGTs connected in series with access transistors to the String Select Line (SSL), the Source line (SL), and the Ground Select Line (GSL), as shown in Figure 1. The number of pages in a block is equal to the number of FGTs connected in series while the size of each page is equal to the number of bit-lines running through the block. A pass transistor is connected to each word-line to select/unselect the page.

B. Basic Flash Operations

NAND flash uses Fowler-Nordheim (FN) tunneling to move charges to/from the floating gate. A program operation involves tunneling charges to the floating gate while an erase involves tunneling charges off the floating gate. A read operation involves sensing whether a floating gate is charged or not. The program and read operations are performed at a page granularity, while an erase is performed at the granularity of a block. More details on these operations and on the design of flash memory arrays are given in [2], [14], [15].

III. The Power Model

This section presents the details of the analytical power model that we have developed for NAND flash memory chips.

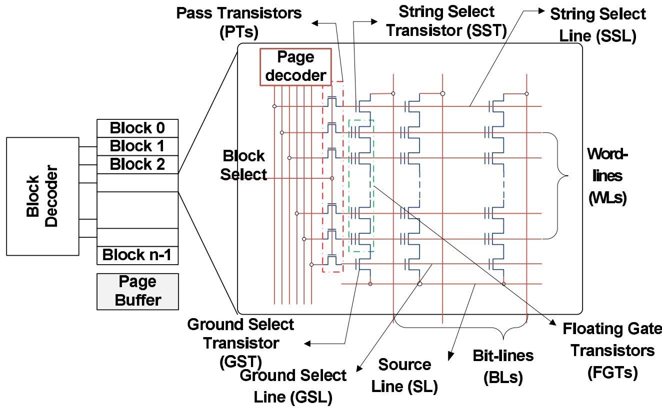


Figure 1: A NAND Flash Memory Plane. Adapted from [2]

FlashPower models the energy dissipated during the basic flash operations and when the chip is idle. Before delving into the details of the model, we list the components that are modeled and the power state machine. We then explain how *FlashPower* was integrated with CACTI [17], followed by the details of the model itself. We then conclude this section with a discussion of the current limitations of *FlashPower*.

A. Circuit Components

With respect to Figure 1, the components that dissipate energy are,

- The bit-line (BL) and word-line (WL) wires.
- The SSL, GSL and SL.
- The drain, source and the gate of the SST, GST and PTs.
- The drain, source and control gate of the FGTs.
- The floating gate of the FGTs - Energy dissipated during program and erase operation.

In addition to the above components, the energy dissipated by the block and page decoders, the sense amplifiers present in the page buffer (for the read operation), the charge pumps (that provide high voltages for program and erase operation) and the I/O pins are modeled. The energy per read, program and erase operation is determined by aggregating the energy dissipated by all the aforementioned components.

B. Power State Machine

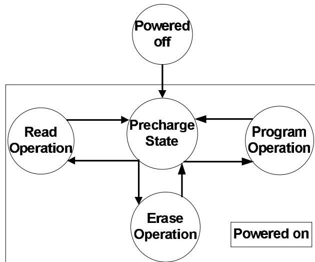


Figure 2: Power State Machine for a SLC NAND flash chip

Figure 2 describes the power state machine for a SLC NAND flash chip. The circles represent the individual states, while the solid lines denote state transitions. We model the energy dissipated when the chip is powered. When the chip is on but is not performing any operations, it is said to be in a "precharge state". In this state, the bit-lines are precharged while the word-lines, and the select lines (SSL, GSL and SL) are grounded. The array is isolated by the select lines but is ready to respond to the commands from the controller. Upon receiving a read, program, or erase command from the controller, the state machine switches to the corresponding state. This state transition along with the actual operation dissipates energy, which we model. Upon completion of the command, the state machine switches back to the precharge state, dissipating energy in the process.

C. Integration with CACTI

FlashPower is designed as a plug-in to CACTI 5.3 [17], which is a widely used memory modeling tool. The power consumed by array peripherals such as decoders and sense amplifiers are estimated assuming that they are high performance devices and the bit-lines and word-lines are estimated assuming aggressive interconnect projections. We also model a FGT as a CMOS transistor and then use the CMOS transistor models of CACTI to calculate the parasitic capacitances of a FGT. However unlike CACTI, which models individual components of a memory system, we have chosen to model the basic operations on the flash memory. This is because, in the memories that CACTI models, individual operations operate at the same supply voltage to drive the bit-lines and the word-lines. For example in a SRAM cache, the voltage at which the word-line is charged is same for both read and write. The granularity of a read/write is also the same. However for NAND flash, read, program and erase operate at different bias conditions and the granularity of an erase differs from read/program. Since the circuitry behaves differently for these different operations, we believe that it is more appropriate to model energy for the basic operations on the flash memory. The inputs to *FlashPower* are summarized in Table I.

D. Power Modeling Methodology

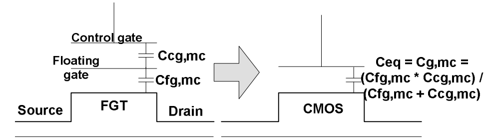


Figure 3: Modeling a FGT

Modeling a FGT: To calculate the energy dissipation, it is necessary to estimate a FGT's parasitic capacitances - i.e. a FGT's source, drain and gate capacitances. While the source and the drain of a FGT and a CMOS transistor are similar, FGT has a two gate structure (floating and control) while CMOS has a single gate. A dual gate FGT structure is modeled as a single gate CMOS structure and the parasitic capacitances of this CMOS structure is modeled using CACTI. Modeling a FGT as a CMOS structure is done by calculating the equivalent capacitance of the two capacitors (one across the interpoly dielectric and other across the tunnel oxide). This is illustrated in Figure 3. In the figure, the control gate capacitance $C_{cg,mc}$ is calculated using the information on gate coupling ratio(GCR) available in [7], while the floating gate capacitance $C_{fg,mc}$ is calculated using the overlap, fringe and area capacitance of a CMOS transistor of the same feature size. The source and drain capacitance of the transistor depends on whether the transistor is folded or not. *FlashPower* assumes that the transistor is not folded as the feature size is in the order of tens of nanometers. The drain capacitance is modeled using CACTI. Other CMOS transistors like the GST, PT and SST are modeled using CACTI.

Derived Parameters: The length of each bit-line $L_{bitline}$ and each word-line $L_{wordline}$ are derived as:

$$L_{wordline} = N_{bitlines-perblock} * N_{bcols} * pitch_{bit-line} \quad (1)$$

$$L_{bitline} = (N_{pages} + 3) * N_{brows} * pitch_{word-line} \quad (2)$$

Table I: Inputs to *FlashPower*.

Microarchitectural Parameters		Bias Parameters	
$N_{pagesize}$ (R)	Size (in bytes) for the data area of each page.	V_{dd} (R)	Maximum operating voltage of the chip.
$N_{sparebytes}$ (R)	Size (in bytes) for the spare area for each page.	V_{read} (O)	Read voltage.
N_{pages} (R)	Number of pages per block.	V_{pgm} (O)	Program voltage for selected page.
N_{brows} (R)	Number of rows of blocks in a plane.	V_{pass} (O)	Pass voltage during program for un-selected page.
N_{bcols} (O)	Number of columns of blocks in a plane.	V_{era} (O)	Erase voltage to bias the substrate.
N_{planes} (O)	Number of planes per die.	$V_{bl,pre}$ (O)	Bit-line precharge voltage.
N_{dies} (O)	Number of dies per chip.	Timing Parameters	
$tech$ (R)	Feature size of FGTs.	$t_{program}$ (O)	Latency to program a page from the page buffer to the plane.
Device-level Parameters		N_{loop} (R)	Maximum partial program cycles.
N_A, N_D (O)	Doping level of P-well and N-well.	t_{read} (R)	Latency to read a page from a plane to the page buffer.
β (O)	Capacitive coupling between control gate and the P-well.	Workload Parameters	
GCR (O)	Ratio of control gate to total floating gate capacitance	N_{bits_1} (O)	Number of 1's to be read, programmed or stored.
Chip-level Parameters		(R) - Indicates required argument.	
C_{pin} (R)	Capacitance of I/O pins in the chip.	(O) - Indicates optional argument.	

where $N_{bitlines-perblock} = (N_{pagesize} + N_{sparebytes}) * 8$. "3" is added to N_{pages} because for each block, the bit-line crosses 3 select lines (GSL, SSL and SL). The terms $pitch_{bit-line}$ and $pitch_{word-line}$ refer to the bit-line and word-line pitch respectively and are equal to $2 * featuresize$.

The total capacitance to be driven along a word-line is given by

$$C_{wl} = C_{d,pt} + C_{g,mc} * N_{bitlines-perblock} + C_{wl,wire} * L_{wordline} \quad (3)$$

where, $C_{d,pt}$ is the drain capacitance of the pass transistor, $C_{g,mc}$ the equivalent gate capacitance of FGT and and $C_{wl,wire}$ is the word-line wire capacitance. Similarly, the total capacitance to be driven along the bit-line is equal to

$$C_{bl} = 2 * C_{d,st} + C_{d,mc} * N_{pages} + C_{bl,wire} * L_{bitline} \quad (4)$$

where $C_{d,st}$ is the drain capacitance of the select transistors (SST), $C_{d,mc}$ is the drain and source capacitance of the FGT and $C_{bl,wire}$ is the bit-line wire capacitance. It is assumed that adjacent FGTs connected in series share the source and the drain. Because of this assumption, the source capacitance of one FGT is considered equal to the drain capacitance of the neighbouring FGT. The source of SST is shared with the drain of the first FGT in the string while the source of the last FGT in the string is shared with the drain of GST. The total capacitance of the SSL is calculated as,

$$C_{ssl} = C_{gsl} = C_{d,pt} + C_{g,st} * N_{bitlines-perblock} + C_{wl,wire} * L_{wordline} \quad (5)$$

It should also be noted that the GSL needs to drive the same components as the SSL and hence $C_{ssl} = C_{gsl}$. The capacitive component of the SL is equal to:

$$C_{srcline} = C_{wl,wire} * L_{wordline} + C_{d,st} \quad (6)$$

It is assumed that the source capacitance of GST is equal to it's drain capacitance ($C_{d,st}$).

Transition from the Idle to the Precharged State: When this transition occurs, the bit-lines and word-lines are precharged to $V_{bl,pre}$ and $V_{wl,pre}$ respectively. The SST, GST and the PTs are biased so that the current flowing through the FGTs is disabled. The source line is also grounded. The total energy dissipated by the array while in the precharge state (E_{pre}) is given by:

$$E_{pre} = E_{bl,pre} + E_{wl,pre}$$

$$E_{bl,pre} = 0.5 * C_{bl,wire} * (0 - V_{bl,pre})^2 * N_{bitlines-perblock} * L_{bitline}$$

$$E_{wl,pre} = 0.5 * C_{wl,wire} * (0 - V_{wl,pre})^2 * N_{pages} * L_{wordline}$$

While bit-lines are generally precharged to reduce the latency of read accesses [9], the word-lines are not. During our validation $V_{wl,pre}$ was set to zero, but the model includes the word-line precharge $V_{wl,pre}$ as a parameter so that devices that perform word-line precharging can use this parameter.

The Read Operation: To perform a read operation, the block decoder selects one of the blocks to be read while the page decoder selects one of the N_{pages} inside a block. In the selected block, the word-line of the *selected* page is grounded while the *un-selected* word-lines are biased to V_{read} from $V_{wl,pre}$. This causes the un-selected pages to serve as transfer gates. Hence the energy dissipated in biasing the word-line of the selected page to ground and the un-selected pages to $V_{read}$is equal to:

$$E_{selected-page,r} = 0.5 * C_{wl} * (0 - V_{wl,pre})^2 \quad (7)$$

$$E_{unselected-pages,r} = 0.5 * C_{wl} * (V_{read} - V_{wl,pre})^2 * (N_{pages} - 1) \quad (8)$$

For the read operation, the bit-lines remain at $V_{bl,pre}$. If the FGT is on (i.e., there is no charge on the floating gate, which corresponds to a logical "1"), then the bit-line is pulled down. Otherwise, the FGT is off and the bit-line is not pulled down. Assuming N_{bits_1} to be the number of bits corresponding to logical "1", the resulting energy dissipation is given by:

$$E_{bl-1,r} = 0.5 * C_{bl} * (0 - V_{bl,pre})^2 * N_{bits_1} \quad (9)$$

$$E_{sl,r} = E_{ssl,r} + E_{gsl,r} + E_{srcline,r} \quad (10)$$

where

$$E_{ssl,r} = E_{gsl,r} = 0.5 * C_{ssl} * (V_{read} - 0)^2$$

$$E_{srcline,r} = 0.5 * C_{srcline} * (V_{read} - 0)^2$$

The state of the cell is detected using the sense amplifier connected to each bit-line. The sense amplifier is a part of the page buffer and contains a latch unit [9]. It detects the voltage changes in the bit-line and compares it with a reference voltage. Since this is very similar to DRAM sense amplifier [16], we use CACTI's DRAM sense amplifier model to determine the dissipated energy $E_{senseamp,r}$.

Once the read operation is complete, the system transitions from the read state to the precharged state. This means that bit-lines corresponding to logical "1"are biased back to the precharge voltage $V_{bl,pre}$, while the select lines are biased back to ground from V_{read} and the word-lines are precharged back to $V_{wl,pre}$. (Note that bit-lines corresponding to logical "0" would not have discharged and hence it is not necessary to precharge them again). But the biasing for the transition from read operation to precharge state is equal but opposite to the biasing for the transition from the precharge state to read operation. Hence the energy to transition from the read to the precharge state (E_{r-pre}) is given by,

$$E_{r-pre} = E_{bl-1,r} + E_{selected-page,r} + E_{unselected-pages,r} + E_{sl,r} \quad (11)$$

We can calculate the energy dissipated by the decoders for the read operation, $E_{dec,r}$,using CACTI. We modify the CACTI decoder model so that for each read and program operation, two levels of decoding (block and page decode) are performed. Hence the total energy dissipated per read operation is

$$E_r = E_{selected-page,r} + E_{unselected-pages,r} + E_{bl-1,r} + E_{sl,r} + E_{r-pre} + E_{senseamp,r} + E_{dec,r} \quad (12)$$

The Program Operation: The program operation involves decoding the page to be programmed and transferring the data from the controller to the page buffers. Since the decoding for the program operation is the same as that of the read operation, the energy dissipated for decoding during the program operation is equal to $E_{dec,p} = E_{dec,r}$, where $E_{dec,r}$ was estimated using CACTI for read operation. After the data to be programmed is latched in the page buffers, the selected word-line is biased to V_{pgm} while the un-selected word-lines are set to V_{pass} to inhibit them from programming. Thus the energy dissipation for the selected page and the un-selected page is given by:

$$E_{selected-page,p} = 0.5 * C_{wl} * (V_{pgm} - V_{wl,pre})^2 \quad (13)$$

$$E_{unselected-page,p} = 0.5 * C_{wl} * (V_{pass} - V_{wl,pre})^2 * (N_{pages} - 1) \quad (14)$$

FlashPower adopts the self-boosted program inhibit model [14] to prevent cells corresponding to logical "1" from being programmed. This is achieved by boosting the channel voltage by biasing the bit-lines corresponding to logical "1" at $V_{bl,ip}$ and setting the SSL to V_{dd}. Assuming N_{bits_1} to be the number of bits corresponding to logical "1", the energy is given by

$$E_{bl,ip} = 0.5 * (C_{bl} - C_{d,mc} * N_{pages}) * (V_{bl,p} - V_{bl,pre})^2 * N_{bits_1} \quad (15)$$

With a high voltage, V_{pgm}, on the word-line, the electric field across the channel and the floating gate gets high enough for FN tunneling to take effect. As electrons get tunneled from the channel to the floating gate, a tunneling current, I_{FN}, flows across the channel and the threshold voltage of the FGT increases by $\Delta V_{th}$to V_{pref}. To avoid wearing out of the tunnel oxide, the increase in threshold voltage is generally achieved by using multiple short pulses (referred to as sub-program operation) of V_{pgm} applied to the control gate. The voltage pulse is applied for a short interval and after each interval, a verify-program operation is performed to check if ΔV_{th} increase is achieved. If the desired increase is not achieved, this process is repeated (for a maximum of N_{loop} times) for successful programming. Thus, the total program energy along the bit-line is calculated as,

$$E_{bl,p} = (0.5 * (C_{bl} - C_{d,mc} * N_{pages}) * (0 - V_{bl,pre})^2 + E_{tunnel,mc}) * (N_{bitlines-perblock} - N_{bits_1}) \quad (16)$$

where the term $E_{tunnel,mc}$ is the tunneling energy per cell. $E_{tunnel,mc}$ is calculated as

$$E_{tunnel,mc} = \Delta V_{th} * I_{FN} * t_{sub-program} \quad (17)$$

where I_{FN} is the tunnel current and $t_{sub-program}$ is the duration of sub-program operation. I_{FN} is calculated as $I_{FN} = J_{FN} * A_{fgt}$, where J_{FN} is the tunnel current density calculated using [10] and A_{fgt} is the area of the floating gate.

Then the energy dissipated in charging the select lines is given by:

$$E_{sl,p} = E_{ssl,p} + E_{gsl,p} + E_{srcline,p} \quad (18)$$

where

$$E_{ssl,p} = E_{gsl,p} = 0.5 * C_{ssl} * (V_{dd} - 0)^2$$

$$E_{srcline,p} = 0.5 * C_{srcline} * (V_{dd} - 0)^2$$

Hence, the total energy per sub-program operation is :

$$E_{subp} = E_{selected-page,p} + E_{unselected-page,p} + E_{bl,ip} + E_{bl,p} + E_{sl,p} \quad (19)$$

Since the entire process of sub-program and verify-program is repeated a maximum of N_{loop} number of times, the maximum energy for programming is given by :

$$E_{pgm} = N_{loop} * \left(E_{subp} + E_{vp}\right) \quad (20)$$

where E_{vp}, the energy spent in verify-program. *FlashPower* currently assumes the verify-program to be the same as a read operation. N_{loop} is obtained from datasheets like [12] or can be fed as a input to the model.

Since a program operation concludes with a read operation, the transition from the Program to Precharge is same as the transition from a read to precharge.

$$E_{p-pre} = E_{r-pre} \quad (21)$$

where E_{r-pre} is given by equation (11).

Hence the total energy dissipated in the program operation is equal to:

$$E_p = E_{pgm} + E_{p-pre} + E_{dec,p} \quad (22)$$

The Erase Operation: Since erasure happens at a block granularity, the controller sends the address of the block to be erased. Only the block decoder is used and the energy dissipated for block decoding ($E_{dec,e}$) is calculated using CACTI. To aid block-level erasing, the blocks are physically laid out such that all pages in a single block share a common P-well. Moreover, multiple blocks share the same P-well [14] and therefore it is necessary to prevent other blocks sharing the same P-well from being erased. *FlashPower* assumes the self-boosted erase inhibit model [14] to inhibit other blocks sharing the same P-well from getting erased.

For the erase operation, the control gates of all the word-lines in the selected block are grounded. The P-well for the selected block is biased to erase voltage V_{era}. The SSL and the GSL are biased to $V_{era} * \beta$ where β is the capacitive coupling ratio of cells between control gate and the P-well. Typical value of β is 0.8 [2]. The SL is biased to $V_{bl,e}$. Here $V_{bl,e}$ is equal to $V_{era} - V_{bi}$ where V_{bi} is the built-in potential between the bitline and the P-well of the cell array [11]. Adding up the charging of the SSL, GSL and SL, we get the energy dissipated in the select lines to be:

$$E_{sl,e} = E_{ssl,e} + E_{gsl,e} + E_{srcline,e} \quad (23)$$

where $E_{ssl,e}$, $E_{gsl,e}$ and $E_{srcline,e}$ are calculated using the SSL, GSL and SL capacitance and the bias condition explained above.

The bit-lines are biased to $V_{bl,e}$ which dissipates energy that is modeled as,

$$E_{bl,e} = 0.5 * C_{bl} * (V_{bl,e} - V_{bl,pre})^2 * N_{bitlines-perblock} \quad (24)$$

With the P-well biased to V_{era}, cells that were 0-programmed earlier undergo FN tunneling. Electrons tunnel off the floating gate onto the substrate and the threshold voltage of the cell is reduced to the level of 1-programmed cells. It is assumed that 1-programmed cells in the same block do not undergo tunneling because their threshold voltage is already low and tunneling would further reduce this.

To effect FN tunneling, the depletion layer capacitance between the P-well and the N-well should be charged. This capacitance of the junction between the P-well and the N-well, which form a P-N junction, is a function of the applied voltage V_{era}, the area of the p-well A_{well}. The dynamic power dissipated to charge this capacitance as the voltage across the junction raises from $0V$ to V_{era} is determined as:

$$E_{junction,e} = \left(\frac{C_{j0}}{(1 - V_{era}/\phi_0)^m} * A_{fgt}\right) * V_{era}^2 \quad (25)$$

where C_{j0} is the capacitance at zero-bias condition and ϕ_0 is the built-in potential of the P-N junction. m, the grading coefficient is assumed to be 0.5.

Once this P-N junction capacitance is fully charged, tunneling happens in all 0-programmed cells. The tunneling energy is calculated assuming that the duration of the V_{era} pulse is same as $t_{sub-program}$. Hence the total energy due to tunneling of all 0-programmed cells in the array is calculated as,

$$E_{tunnelerase,mc} = E_{tunnel,mc} * (N_{bitlines-perblock} - N_{bits_1}) \quad (26)$$

where $E_{tunnel,mc}$ is given by equation (17).

After a block is erased, a verify erase operation is performed to ensure that all FGTs in the block are erased [14]. The verify erase operation is a single read operation which consumes energy equivalent to a read operation. This is modeled as $E_{block,ve} = E_r - E_{dec,r}$, where E_r is given by equation (12). Since an erase operation ends with a read operation, the transition from the erase to precharge is same as the transition from read to precharge.

$$E_{e-pre} = E_{r-pre} \quad (27)$$

where E_{r-pre} is given by equation (11).

Hence the total energy dissipated in erase operation E_{erase}

is equal to:

$$E_{erase} = E_{dec,e} + E_{bl,e} + E_{tunnelerase,mc} + E_{junction,e} + E_{sl,e} + E_{e-pre} + E_{block,ve} \quad (28)$$

I/O pins: Since a read/program operation involves transfer of data to/from the controller to the chip, the I/O pins in the chip switch continuously during the data transfer which dissipates energy. We estimate I/O pin energy as,

$$E_{iopin} = C_{pin} * V_{dd}^2 * N_{bitlines-perblock} \quad (29)$$

where C_{pin} is the capacitance of a pin that needs to be driven, V_{dd} is the applied voltage to the pin.

Multi-plane operation: The power model described thus far corresponds to single-plane operations. However modern NAND flash chips allow multiple planes to operate in parallel. These are referred to as multi-plane operations. Since *FlashPower* models single-plane operations, energy consumption for multi-plane operations can be determined by multiplying the results of single-plane operations with the number of planes operating in parallel.

Charge pumps: We also model the energy dissipated by charge-pumps during the program and erase operations. The energy consumed by charge-pumps is provided in [6], where the authors specify that conventional charge-pumps operating at 1.8V consume 0.25μJ while charge-pumps operating at 3.3V consumed 0.15μJ. We use these constant values in *FlashPower*, but since *FlashPower* is parameterized, a detailed charge-pump model can be easily incorporated in lieu of the current one.

Limitations of FlashPower: has two limitations. *FlashPower* currently does not model the energy consumption of MLC NAND flash memories and NOR flash memories. *FlashPower* can be extended to model MLC-NAND flash by increasing the number of verify-program cycles for the program operation and modeling sense-amplifiers that can detect multiple voltage levels for read operation. We plan to extend *FlashPower* to include power models for MLC chips and NOR flash in future work.

IV. Validation

Validating *FlashPower* is challenging since there is very little publicly available information on the power consumption of flash chips and manufacturer datasheets are not sufficient for validation purposes. Interestingly, researchers at the University of California, San Diego recently published a paper on an empirical study they undertook to understand the characteristics of flash chips [5]. This study includes power measurements. We validate *FlashPower* by comparing the power estimates provided by our model to the measured values reported in the UCSD paper. This paper reports the energy dissipation of a single plane and includes the data transfer between the page buffer and the I/O pins of the flash chip. The data provided in that paper regarding the microarchitecture of flash chips is given in Table II. The chips are from two different manufacturers which we denote as: “A” and “B”. In addition to the data given in the table, we require additional details about the microarchitecture of those chips and the flash device technology used in them to calculate the energy dissipation in *FlashPower*. These unknown parameters, which tend to be manufacturer specific, include:

- Device feature sizes
- Applied bias voltages for the read, program, and erase operations
- Capacitance of the chip I/O pins, which can vary from 5 pF-40 pF across chips and manufacturers
- The number of partial program/erase cycles, which can be as high as 4-8 cycles across chips and manufacturers
- Manufacturer-specific device technology and circuit optimizations

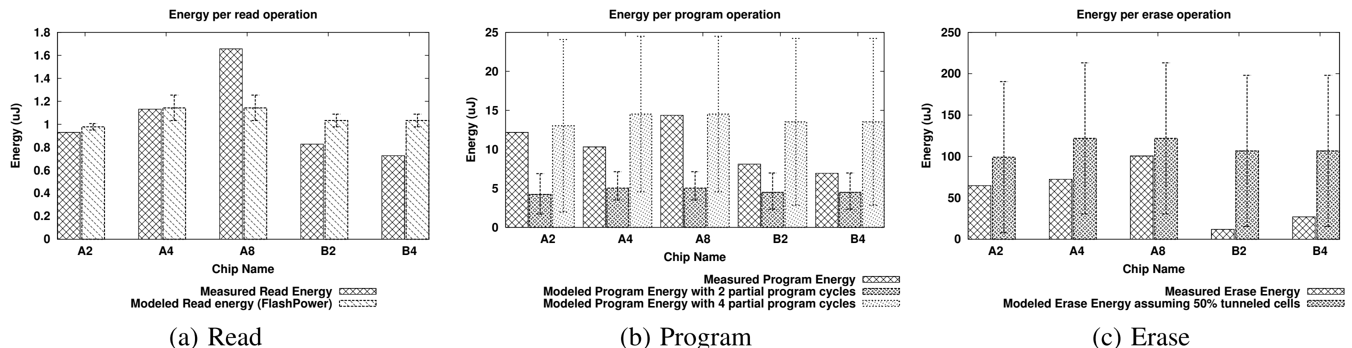
Chip Name	Capacity (Gb)	Pagesize + Spare area (B)	Pages/ Block	Blocks/ Plane	Planes/ Die	Dies
A2	2	2048+64	64	1024	2	1
A4	4	2048+64	64	4096	1	1
A8	8	2048+64	64	4096	2	1
B2	2	2048+64	64	2048	1	1
B4	4	2048+64	64	2048	2	1

Table II: Microarchitectural parameters of the validation chips. Chips “A” and “B” are from different manufacturers. [5].

In our validation study, we assume the feature size to be 80nm. In addition to the feature size, *FlashPower* also requires other device technology information such as the doping-levels of the P- and N-wells, estimates for which we obtained from industry [11]. The values for the various voltage bias parameters were obtained from [7], [2]. The value for the capacitance of the I/O pins and the latencies for the various flash operations were obtained from [12].

Another key factor that can influence the energy dissipation is the mix of 0’s and 1’s that are read from, written to, or stored in the flash cells (in the case of erase). The read energies are only slightly affected by the mix of 0’s and 1’s because the read energy is determined by the number of bit-lines that are discharged, which depends on the number of cells that hold a “1”, but is more strongly affected by the energy required to transfer the data from the page buffer to the I/O pins, which is relatively unaffected by the value of the bits. On the other hand, the program energies are heavily impacted by the bit values because writing a “0” to a flash cell requires driving its corresponding bit-line and therefore the bit-line energy scales up with the number of 0’s that need to be written to a page. Since we do not know the mix of 0’s and 1’s in the workloads used for the power measurements, we present the energy dissipation data for the case where half the bits are 0’s and the other half are 1’s as well as cases where the bit distribution causes the least or the most amount of energy to be dissipated for the read, program, and erase operations.

The validation results are given in Figure 4. For each of the three flash operations, we present the power measurement data from [5] as well as estimates provided by *FlashPower*. Since the distribution of 0’s and 1’s and the number of partial program/erase cycles both impact the energy dissipation and their values for the actual power measurements are not reported in their paper [5], we vary both these parameters in our validation experiments to ascertain whether the measured data is close to the estimated energy within this range of uncertainty. For the read operation shown in Figure 4(a), the first bar in each pair corresponds to the measured value whereas the second bar corresponds to the energy estimate provided by *FlashPower* assuming that an equal number of 0’s and 1’s are read from the page. The error-bar is to account for the range of possibilities with regard to the bit distribution; the lower edge of the bar corresponds to a page with all 0’s and the upper edge to all 1’s. For the write operation, shown in Figure 4(b), the energy dissipation depends on the number of partial program cycles and also on the mix of 0’s and 1’s written to the page. We consider two values for the partial program cycles (2 and 4 cycles), which correspond to the second and third bars in the triplet of bars for each chip. The error-bars for each of the estimates show the range of energy dissipation values from a pattern of all 1’s (least energy dissipation) to all 0’s (highest energy dissipation). For the erase operation (Figure 4(c)), the energy dissipation depends on the number of cells that need to undergo tunneling, which in turn depends on the bits they contain when the erase operation is initiated. The second bar in each pair corresponds to the energy dissipation for the case where the cells contain an equal number of 0’s and 1’s and the error-bar shows the variation in energy for cells that contain



(a) Read (b) Program (c) Erase

Figure 4: Validation results for *FlashPower*. The error-bars depict the energy variation based on the mix of 0's and 1's read from/written to the cells of a page during reads and programs, or stored within the cells of a block during erase. Source [5].

all 1's to those that contain all 0's. *Note that the error-bars do not indicate a limitation of the model.* Instead, these bars represent the range of values that a workload may generate or access, which will be input to *FlashPower* by an architecture simulator.

Overall, we observe that *FlashPower* is able to estimate the energy characteristics of the three chips from manufacturer "A" with reasonable accuracy for all three operations within the range of uncertainty imposed by the bit distributions and the number of partial program and erase cycles. The only exception is chip A8, where the estimated read energy is approximately 0.35 nJ lower than the measured value. We believe that this difference is due the I/O pin capacitance of this chip. As mentioned previously, the pin capacitance tends to vary from 5-40 pF between different chips, even for the same manufacturer. We find that even a small change in the capacitance that we assume can have a significant impact on the read energy. For example, find that changing the pin capacitance from 5 pF (our default assumption for all the chips) to just 7.5 pF shifts the bar upwards and brings the measured energy dissipation value for A8 within the range of possible values estimated by *FlashPower*. However, our model is unable to accurately estimate the energy characteristics of chips from manufacturer "B", especially for reads and erase. As mentioned earlier, there are several manufacturer-specific implementation details that can have a significant impact on the energy estimation and it is possible that the values that we chose as inputs to the model, a few of which were drawn from a specific datasheet [12], do not match the characteristics of these chips.

The larger range of read energy dissipation values for A4 and A8 compared to A2 is due to the fact that they are higher-capacity devices and therefore the cells that store 1's have to discharge via longer bit-lines. For the program operation, the energy dissipation increases when the programming latency is higher due to an increase in the tunneling energy. The larger range of energy dissipation values for the longer latency program operation is due to the fact that a single page in these devices consists of 2^{14} cells and the energy dissipation depends on how many of those cells tunnel based on the bits they store. Similarly, we can observe that the erase operation dissipates the most energy and also shows a large variation in the dissipation since a block consists of 2^{20} cells. This large variation in the energy dissipation based on the actual values of the bits suggests opportunities for applying architecture level power optimization techniques such as invert coding [13]. *FlashPower* can be used to evaluate such techniques.

V. Conclusions

In this paper, we have presented *FlashPower*, which is a detailed analytical power model for NAND flash memory chips suitable for use in architecture level studies. *FlashPower* models the energy dissipation of flash chips from first principles and is highly parameterized to study a large design space of memory organizations. We have validated *FlashPower* against power measurements from real NAND flash chips and find that our model can accurately estimate the energy dissipation of several real chips.

VI. Acknowledgments

We thank Laura Grupp, Steven Swanson and Koji Sakui for their valuable inputs. This research has been supported in part by NSF CAREER Award CCF-0643925, NSF grant CNS-0551630, and gifts from Google and HP.

References

[1] A.M. Caulfield and et al. Gordon: Using flash memory to build fast, power-efficient clusters for data-intensive applications. In *ASPLOS*, 2009.

[2] J.E. Brewer and M. Gill, editors. *Nonvolatile Memory Technologies with Emphasis on Flash*. IEEE Press, 2008.

[3] C. Dirik and B. Jacob. The Performance of PC Solid-State Disks (SSDs) as a Function of Bandwidth, Concurrency, Device Architecture, and System Organization. In *ISCA*, 2009.

[4] E. Gal and S. Toledo. Algorithms and Data Structures for Flash Memories. *ACM Computing Surveys*, 2005.

[5] L. Grupp and et al. Characterizing flash memory: Anomalies, observations, and applications. In *MICRO*, 2009.

[6] K. Ishida and et al. A 1.8V 30nJ adaptive program-voltage (20V) generator for 3D-integrated NAND flash SSD. In *ISSCC*, 2009.

[7] Process Integration and Device Structures, ITRS 2007 Edition.

[8] T. Kgil, D. Roberts, and T. Mudge. Improving NAND Flash Based Disk Caches. In *ISCA*, 2008.

[9] J. Lee and et al. A 90-nm CMOS 1.8V 2-Gb NAND flash memory for mass storage applications. *ISSCC*, 2003.

[10] M. Lenzlinger and E.H. Snow. Fowler-Nordheim tunneling into thermally grown SiO_2. *IEEE Transactions on Electron Devices*, 1968.

[11] Koji Sakui. Intel Corporation/Tohoku University, May 2009. Private Correspondence.

[12] Samsung corporation. K9XXG08XXM flash memory specification.

[13] M.R. Stan and W.P. Burleson. Bus-Invert Coding for Low-Power I/O. *IEEE Transactions on Very Large Scale Integration Systems*, 1995.

[14] K.D. Suh and et al. A 3.3V 32Mb NAND flash memory with incremental step pulse programming scheme. *IEEE Journal of Solid-State Circuits*, 1995.

[15] T. Tanaka and et al. A quick intelligent page-programming architecture and a shielded bitline sensing method for 3V-only nand flash memory. *IEEE Journal of Solid-State Circuits*, 1994.

[16] David Tawei Wang. *Modern DRAM Memory Systems: Performance analysis and a high performance, power-constrained DRAM scheduling algorithm*. PhD thesis, University of Maryland, College Park, 2005.

[17] S.J.E. Wilton and N.P. Jouppi. Cacti: an enhanced cache access and cycle time model. *Solid-State Circuits, IEEE Journal of*, 1996.