# MODELING IO LATENCY OF SSDS

**Jinsoo Yoo[1], Youjip Won[1]**

[1]Department of Computer and Software, Hanyang University, Seoul, Korea
jedisty@hanyang.ac.kr, yjwon@hanyang.ac.kr

**Abstract:** SSDs support various type of parallel IO mechanism. Thus, simulating IO latency of SSDs is difficult. In this paper, we develop SSD IO latency model for multi-channel, multi-way SSD. With the IO latency model, An SSD simulator can generate IO latency with a single thread. The IO latency models has 4% error rate compared with a real SSD, Intel X25-M.

**Keywords:** modeling, IO latency, SSD simulation

## 1 Introduction

The use of SSD is increasing in various computing environment [1]. For developing SSD, software based simulation is widely used. This is because developing SSD with H/W implement [2-3] is time consuming and expensive.

Many software based SSD simulators have been proposed to examine various aspect of SSD [4-8]. An SSD simulator should be developed to support IO parallelism because SSD provides it in various levels. There are three IO parallel operations in SSD. First, SSD uses channel parallelism by connecting flash memories to flash controller with different channels. Second, way parallelism can be used by managing several flash memories in a channel. Lastly, in a flash memory, plane parallelism processes IOs in parallel. IO parallelism makes simulating a SSD difficult. One of the possible solutions in simulating SSD is multi thread method.

In multi-threaded simulator, a thread is allocated to every flash memory and channel. Each thread is allocated to each plane and processes IO command. Thus, a more number of threads are required as the number of channel and way increases. For example, to simulate Intel X25-M SSD [9] which has 10 channels 2 ways, at least 20 threads is required. Furthermore, to consider channel thread which manages channel IO behavior, 30 threads is needed. As the number of threads increases, Lock contention can occur and multi-core CPU is demanded. Also overhead related to thread scheduling increases. Therefore, multi-threaded simulator is hard to simulate SSD IO behavior accurately.

In this paper, we propose analytical IO latency model of SSDs to calculate IO processing time of SSD. The models can be used for single-threaded IO simulator algorithm that supports IO parallelism of SSD. The simulator manages all IOs which are processed in each flash memory with a single thread and imposes proper delay using the delay models.

## 2 Backgrond

In this section, we describe the internal of an SSD. Fig. 1 shows the internal of an SSD with 4 channels and 4 ways. Through host interface, an SSD receives IO command. The host command is enqueued in command queue managed by firmware and the corresponding data is stored in IO buffer. The command queue and I/O buffer are located in DRAM.

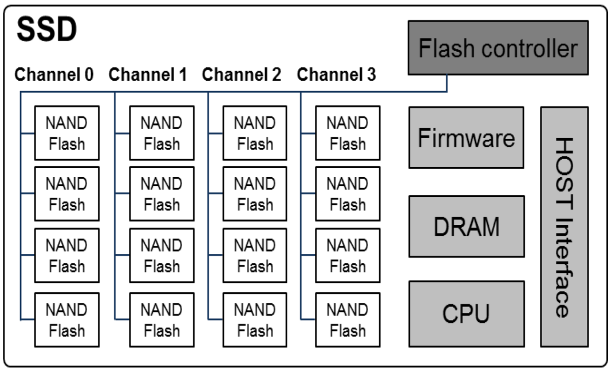

**Figure 1** SSD block diagram (4 channels, 4 ways)

As the processing unit of a flash memory is a page, firmware changes the sector-based IO command to page-based IO command and transfers the command to a flash controller. And then, flash controller processes the page IO with a flash memory. The path used to transfer data between a flash controller and a flash memory is called channel. And each channel consists of multiple flash memories which are described as ways. SSD exploits various levels of IO parallelism: Plain Parallelism, Channel Parallelism, and Way Parallelism. Using these IO methods, SSD fully utilizes flash memories performance and increases IO bandwidth

## 3 Modeling the IO delay of SSDs

In this section, we formulate SSD IO processing time according to various workloads. An IO processing time alter significantly by architecture of SSD and flash controller algorithm. For modeling the SSD IO processing model, we adopt Intel X25-M [9] IO processing behavior [10]. X25-M allocates every new page write operations to different channel in same way. If every flash memory in the way receives IO command, flash controller transfer IO commands to flash memories in a next way. SSD IO delay model formulated in this section can be used in the IO simulator. Terms used in this section are listed in Table I.

**Table I** Parameters for SSD IO modeling

| Represent | Description |
|---|---|
| $S_{\text{page}}$ | Page Size |
| $W_{\text{ch}}$ | Channel Switch Delay for Write |
| $R_{\text{ch}}$ | Channel Switch Delay for Read |
| $W_{\text{reg}}$ | Register Write Delay |
| $R_{\text{reg}}$ | Register Read Delay |
| $W_{\text{cell}}$ | NAND Write Delay |
| $R_{\text{cell}}$ | NAND Read Delay |
| $R_{\text{cell}}$ | NAND Read Delay |
| $W_{\text{page}}$ | 1 Page Write Delay |
| $N_{\text{ch}}$ | The number of Channels |
| $N_{\text{way}}$ | The number of Ways |
| $N_{\text{plane}}$ | The number of Planes per a Flash |
| ρ | The maximum number of IOs per a cycle |
| $S_{\text{file}}$ | File size |
| $S_{\text{record}}$ | Record size |
| $N_{\text{page}}$ | The number of pages per a record |
| $N_{\text{cycle}}$ | The number of cycle per a record |
| $N_{\text{remain}}$ | The number of IOs in a last IO cycle |

### 3.1 Modeling the Write Delay

The total write time of a page ($W_{\text{page}}$) is a summation of three processing time of SSD: channel switching delay ($W_{\text{ch}}$), data transfer delay between a flash controller and a flash memory register ($W_{\text{reg}}$), and data programming delay in a free page in a flash memory ( $W_{\text{cell}}$ ). When a SSD uses channel parallelism, way parallelism, and plane parallelism, we denote the maximum number of pages which can be processed in parallel at the same time as ρ. An SSD can process ρ page writes in 1 cycle. In each cycle, a flash memory which deals with 1 ~ ρ page IO is denoted as $FM_1$ ~ $FM_\rho$. First, the flash controller sends a page write request to $FM_1$ and channel switching delay $W_{\text{ch}}$ is imposed. After that, a register write delay ($W_{\text{reg}}$) occurs followed by NAND page write delay ($W_{\text{cell}}$). The summation of these operation times is one page write time of SSD and denoted as $W_{\text{page}}$. After channel switching delay for $FM_1$ is finished, the flash controller writes data to the register in $FM_1$ while sending another page write request to $FM_2$. Before a data is written to $FM_2$, channel switching delay also occurs. These are because $FM_1$ and $FM_2$ are use different channel. In the same way, after the channel switching delay for $FM_2$ is done, the flash controller send another page write request to the next flash memory. Until Sending page write request to $FM_\rho$, the flash controller repeats this behavior. To start first write operation of second cycle at the $FM_1$, the write operation of $FM_1$ should be done. The waiting time of second cycle for starting the first page write operation is denoted as $t_{\text{wait}}$ which is formulated as Eq. (1).

$$t_{\text{wait}} = W_{\text{page}} - W_{\text{ch}} \times \rho \tag{1}$$

The processing time for a cycle is a time from the start of first page write of the cycle to the start of first page write of the next cycle. It is denoted as $t_{cycle}$ and can be calculated as Eq. (2).

$$t_{\text{cycle}} = W_{\text{ch}} \times \rho + t_{\text{wait}} \tag{2}$$

The number of page IOs which is conducted in the last cycle can be less than ρ and we denote it as $N_{\text{remain}}$. We can easily get the processing time of the last cycle($t_{\text{cycle\_last}}$) as follow.

$$t_{\text{cycle\_last}} = W_{ch} \times (N_{\text{remain}} - 1) + W_{\text{page}} \tag{3}$$

Using Eq. (2) and Eq. (3), we can calculate $t_{record}$ which represents a processing time of a write record. For a write request, the number of cycle SSD has to repeat is denoted as $N_{\text{cycle}}$ . Then, $t_{record}$ is calculated as Eq. (4).

$$t_{\text{record}} = W_{ch} \times (N_{\text{page}} - 1) + (W_{\text{page}} - W_{\text{ch}} \times \rho) \times (N_{\text{cycle}} - 1) + W_{\text{page}} \tag{4}$$

### 3.2 Modeling the Read Delay

In the same way, we can get the read delay model. Using Eq. (4), read delay model can be formulated like Eq. (5).

$$t_{\text{record}} = R_{\text{ch}} \times (N_{\text{page}} - 1) + (R_{\text{page}} - R_{\text{ch}} \times \rho) \times (N_{\text{cycle}} - 1) + R_{\text{page}} \tag{5}$$

## 4 Validating the IO Delay Models

We validated the write/read delay model with X25-M. Table II show the SSD configuration of X25-M and the performance of NAND flash memory used in X25-M [11]. We measure the channel switching delay time of X25-M. Channel switching delay for read operation is 16 usec and channel switching delay for write operation is 33 usec. When writing or reading data with X25-M, we use O_DIRECT option and open the X25-M as raw device.

To validate the write delay model, we use 4 sequential write workloads. The workloads write 512MB file sequentially with record size 512KB, 256KB, 128KB, and 64KB respectively.

**Table II** Intel X25-M SSD specifications

| Parameter | Value |
|---|---|
| PAGE_SIZE | 4 KB |
| SECTOR_SIZE | 512 B |
| FLASH_NB | 20 |
| BLOCK_NB | 4 096 blocks |
| PAGE_NB | 256 pages |
| CHANNEL_NB | 10 |
| WAY_NB | 2 |
| PLANES_PER_FLASH | 2 |
| Sequential Read | 250 MB/sec |
| Sequential Write | 70 MB/sec |
| 4KB Random Read | 35 000 IOPS |
| 4KB Random Write | 6 600 IOPS |
| REG_WRITE_DELAY | 82 usec |
| REG_READ_DELAY | 82 usec |
| CELL_PROGRAM_DELAY | 900 usec |
| CELL_READ_DELAY | 50 usec |
| BLOCK_ERASE_DELAY | 2 000 usec |

To calculate the theoretical performance of the write latency, we use Eq. (4). In the equation, we adjust NAND programming delay, $W_{cell}$, 900 usec to 940 usec for considering a performance degrade of the X25-M which has been worn out by intensive IO test [12]. Fig. 2 show performances based on the write delay model and that of X25-M. The difference between the write delay model and x25-m is only 4%.

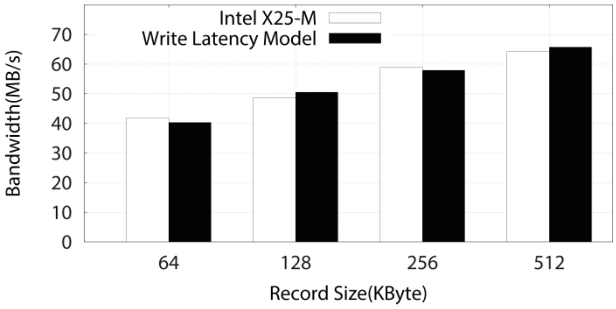

**Figure 2** The write delay model validation with X25-M (Filesize 512MB, raw device, O_DIRECT)

The read delay model shows more accurate results than the write delay model. To validate the read delay model, 4 sequential read workloads which read 512 MB file sequentially with 512 KB ~ 64 KB record size are used. Calculating the theoretical performance with Eq. (5), we also adjust NAND read delay, $R_{\text{cell}}$, 50 usec to 140 usec for considering a performance degrade of the X25-M for the same reason explained above. Fig. 3 shows the results. For sequential read workloads, the error rate of the read delay model based on that of X25-M is within 1.3%.

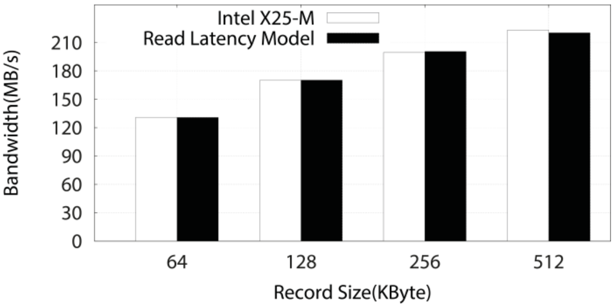

**Figure 3** The read delay model validation with X25-M (Filesize 512MB, raw device, O_DIRECT)

As shown above, we validate the IO performance model and random IO delay model with X25-M. From the experiments, we confirm that the SSD IO delay model can be used in an SSD IO simulator which has multi-channel, multi-way to generate IO latency or can be used to determine whether a SSD IO simulator which processes IO as X25-M simulates an SSD accurately or not.

## 5 Conclusions

In this paper, we propose an analytical delay models that calculates IO latency of SSD. We develop the read/write delay model in the light of the parallel architecture of SSDs. The accuracy of the models are verified with a commercial SSD, Intel X25-M. The IO performance calculated with our IO delay model are closed to the IO performance of the Intel X25-M within 4% offset. From these results, we confirm that the IO delay equations developed in this paper accurately model the IO parallelism of SSD.

## Acknowledgements

MLC SSD: This work was sponsored by IT R&D program MKE/KEIT. [No.10035202, Large Scale hyper-MLC SSD Technology Development].

## References

[1] Wong, G. SSD market overview. In Proc. of Inside Solid State Drives(SSDs), Springer, 2013, volume 37, pages 1-17

[2] OpenSSD Project. http://www.openssd-project.org

[3] Lee, S., Fleming, K., Park, J., Ha, K., Caufield, A. M., Swanson, S., Arvind, and Kim, J. Bluessd: An open platform for cross-layer experiments for nand flash-based ssds. In Proc. of The Workshop on Architectural Research Prototyping(WARP), 2010, Saint-Malo, France

[4] Desnoyers, P. Analytic modeling of ssd write performance. In Proc. of the Annual International Systems and Storage Conference(SYSTOR), ACM, 2012, Haifa, Israel.

[5] Agrawal, N., Prabhakaran, V., Wobber, T., Davis, J. D., Manasse, M., and Panigrahy, R. Design tradeoffs for SSD performance. In Proc. of Annual Technical Conference(ATC), USENIX, 2008, pages 57–70, Boston, Massachusetts, USA

[6] Kim, Y., Tauras, B., Gupta, A., and Urgaonkar, B.

Flashsim: A simulator for nand flash-based solid-state drives. In Proc. of the International Conference on Advances in System Simulation(SIMUL), IEEE, 2009, pages 125–131, Washington, DC, USA.

[7] Cho, S., Won, Y., Kang, S., Cha, J., Choi, J., and Yoon, S. Influence of number of channels and ftl on ssd power consumption. In Proc. of International Conference of Electrical, Computer, Electronics and Biomedical Engineering (ICECEBE), 2012, Bangkok, Thailand.

[8] Yoo, J., Won, Y., Hwang, J., Kang, S., Choi, J., Yoon, S., and Cha, J. VSSIM: Virtual machine based ssd simulator. In Proc. of Mass Storage Systems and Technologies(MSST), IEEE, 2013, Long Beach, CA, USA.

[9] Intel, Corporate. Intel x25-m sata solid-state drive. Specification. http://download.intel.com/design/flash/nand/mainstream/mainstream-sata-ssddatasheet.pdf, 2009

[10] Yoo, B., Won, Y., Cho, S., Kang, S., Choi, J., and Yoon, S. SSD characterization: From energy consumption's perspective. In Proc. of HotStorage 2011, Portland, OR, USA.

[11] Samsung, E. 2g x 8 bit / 4g x 8 bit nand flash memory (k9xxg08uxm). Specification, 2006

[12] Dijkstra, E. W. Stepwise program construction, Springer, 1982