International Journal of Multimedia and Ubiquitous Engineering
Vol.9, No.8 (2014), pp.157-164
http://dx.doi.org/10.14257/ijmue.2014.9.8.14

# On-Demand Data Pumping for Digital Micromirror Device in Maskless Lithography Systems

Hee-Dong Park

*Joongbu University, Kumsan, Korea*
*hdpark@joongbu.ac.kr*

***Abstract***

*The maskless lithography technology requires the management and transmission of exposure data at very high data rates. To meet these requirements, there were techniques on lossless data compression and on high performance data transmission. Many applications using Digital Micro-mirror Device (DMD) require huge and reliable data transmission and high performance processing capabilities enough to make continuous resource flow to the DMD image processing system. In this paper, we present a high data rate pumping method the pattern data to DMD unit on-demand, and implement on Windows-based system as kernel driver for 64-bit PCI interface. We use an evaluation DMD adapter which supports high performance 32/64-bit PCI interface, high-speed FIFO operation with DMA and critical interrupt priority. High priority interrupt with tightly-coupled DMA access technology is used to get high-performance transmission, and we can see that the performance is about 42% of PCI full bandwidth.*

***Keywords:*** *Digital Micro-mirror Device (DMD), Maskless Photolithography, High-performance transmission, On-demand interrupt, PCI interface, DMA, Device driver*

## 1. Introduction

One of the common problems in maskless lithography is the management and transfer of huge data required to define die structures on a wafer. To achieve effective exposure rates the electronics inside an aperture plate system must be provided with a large amount of data in very short intervals [7]. Data compression is an effective technique for saving storage space and transmission bandwidth, but it has also limitations in a sense. There are many high-speed transmission technologies, recently, to transfer enormous data between devices, such as host storage, processor board, optical devices and exposure substrate. Most industries require a direct write method compatible with the UV-sensitive materials and processes that have been developed over many years. Its simple reflective properties, allowing efficient modulation across a broad spectral band, coupled with its superior data rate, make the DMD (Digital Micro-mirror Device) an ideal device for lithographic systems.

DMD is an optical semiconductor on which DLP (Digital Light Processing) technology is based, and is a semiconductor-based light switch array of thousands of individually addressable, tiltable, mirror-pixels. This technology offers better performance in terms of optical fill factor (85% with DMD vs. 64% with LCD) and light transmission (71% with DMD vs. 21% with LCD). Furthermore, computer projectors, like the ones widely used for Power-Point presentations, are commercially available for utilizing DMD technology in image transferring [10]. DMD is very attractive for many applications, including volumetric display, holographic data storage, maskless lithography, scientific instrumentation, and medical imaging. These

ISSN: 1975-0080 IJMUE

applications requires huge data generation, data transmission and processing. So high speed transmission technology should be needed between host adapter and DMD hardware for efficient data processing [1].

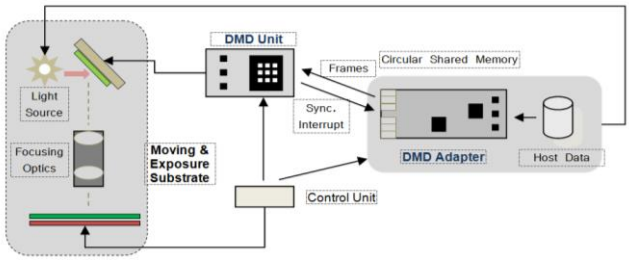



**Figure 1. DMD based maskless Photolithographic System**

In DMD based maskless photolithography system, the micromirror array works as a virtual photomask to write patterns directly onto glass substrates at high speed with low cost [2, 8-9]. However, the task providing huge data for DMD frame of the photolithographic pattern is essential and crucial, even though it is not easy to synchronize between sequence of patterns and irradiation rate off micromirrors. To achieve transfer speeds comparable to optical systems requires a DMD interface capable of transferring high throughputs onto the DMD hardware system. Our goal in this paper is to design a data pumping system on demand, which is capable of meeting the enormous throughput requirement.

This paper is organized as follows; Section 2 describes system overview of DMD architecture and operation, and in Section 3, we present our on-demand data pumping technique. Section 4 presents the implementation and experimentation result of our DMD data pumping system, and finally describes the conclusions.

## 2. System Overview

Lithographic applications such as PCB manufacturing and semiconductor patterning have historically utilized materials and processes based on pattern exposure to ultraviolet light. While the pattern mask has traditionally been provided via film or photomasks, all these industries desire the ability to pattern directly from a digital file. This reflective properties, allowing efficient modulation across a broad spectral band, coupled with its superior data rate, make the DMD an ideal device for lithographic systems.

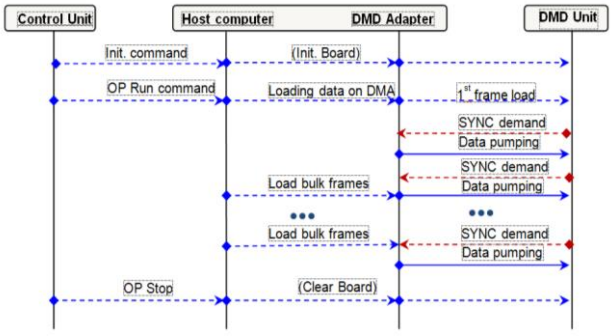


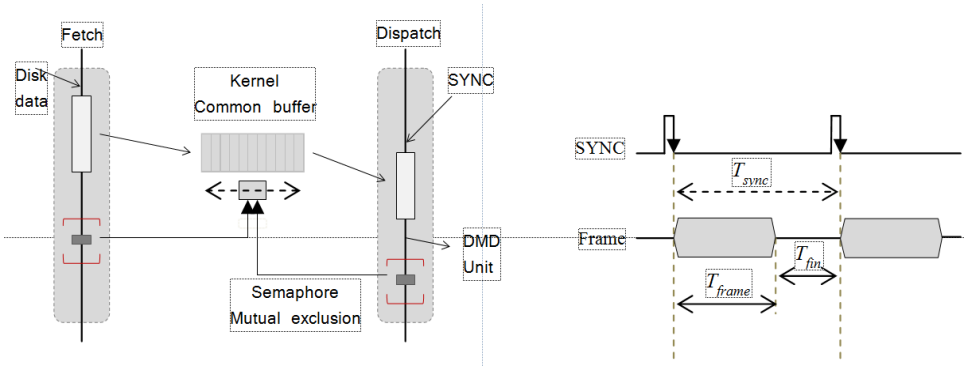

**Figure 2. DMD System Flow for Data Pumping on Demand**

The DMD based maskless photolithography equipment consists of radiation device, irradiation device including DMD system, the host system and the stage control unit. Figure 1 shows the schematic diagram of DMD based maskless photolithography equipment, where the data from the host system to DMD adapter will be transferred in bulk frame format to DMD unit on consecutive frame requests so that it control and mask the moving and exposure substrate by the light reflected off the micromirrors. Throughout DMD based maskless photolithography in concern, all DMD unit does is only control the light reflection for micromirrors, *i.e.*, it gives the approval of reflection as on or off. Therefore, the operation of the maskless photolithography equipment might be thought so simple. However, sending the approval of on/off reflection for each of millions of micromirrors to DMD unit is much complicated. Both reflection by the rotated DMD frame and projection into the scrolling object are imposed [6]. Therefore, besides a customized pattern generating system entirely, we focused our attention on the development of loading photolithographic pattern data from host computer and delivering it to DMD unit.

As shown in Figure 2, our target DMD-based maskless lithography system has the specific data and control flow operation, where control unit makes the operation of system initialization and stop sequences, and also data loading on demand. The DMD adapter fetches pattern data from disk and transfers each frame data on demand with periodic consecutive SYNC signal from DMD unit.

The DMD adapter on the host computer has high-speed bus interface, which has a switched PCI architecture and it maximizes transfer concurrency and improves the efficiency for burst performance on the PCI and processor buses. Using bus mastering and arbitration facilities through the system chipset, the system allows full bus throughput with priority if no other device needs to transfer anything [3].

Our method to transfer huge data from DMD adapter to DMD unit uses two level of processing architecture, which includes getting data from application in kernel mode and dispatching the data to DMD unit on demand with synchronization constructs. The technique uses PCI bus mastering method with critical race free interrupt and DMA interface under kernel level to provide efficient usage of system resources.




a. Data Pumping Algorithm Interface b. Timing Requirement for Frame Sync

**Figure 3. On Demand Pumping Algorithm and Timing Requirement**

## 3. On-demand Data Pumping Architecture

Almost every application depends on the APIs (Application Program Interfaces) of the underlying operating system to perform basic functions as accessing the device or

the file system. Although APIs provide a quick and easy way to tap into an application, they can be constraining for certain power users such as independent software vendors, and for the critical performance operations.

When sending data from disk to transmission hardware, we first copy the pattern data to the application buffer and send the data to the device driver, which also copy the data in application buffer to the driver buffer. This kind of normal algorithm sequence requires twice copy of memory which degrades the performance of transmission system. The technique for pumping data on demand can run in kernel-mode, which runs as part of the operating system's executive, the underlying operating system component that supports one or more protected subsystems. Kernel-mode drivers can perform certain protected operations and can access system structures that user-mode drivers cannot access.

The first algorithm for the high-performance transmission we propose is that we copy application file data to the driver buffer directly without application buffer transfer. This technique increases data transmission speed while the mechanism of file read within the device driver requires more complexity and refinement. Our second algorithm is that we use the occurrence of interrupt at the end of each DMA transfer or at SYNC pulse for normal data dispatch, so makes a processing mechanism of tightly coupled data transmission architecture. Interrupt service routine under interrupt request level must be processed in high priority with software interrupt level, so data transmission with DMA and interrupt can also be processed with high priority than other processes or threads.

As shown in Figure 3(a), our algorithm prepares DMD pattern data from application to the continuous common buffer in kernel, and it handles the status of buffer with some critical variables. Common buffer should be a continuous physical memory enough for holding sufficient number of data frames to avoid data exhaustion. For periodic request signal from DMD interface(SYNC) on demand, high priority interrupt routine dispatches one frame of pattern data in buffer to the DMD interface which is connected to fiber optic media. Period of SYNC signal from DMD interface will be short enough to transfer massive pattern data, so the time to transfer one frame should be fitted for marginal interval which can leave post processing.

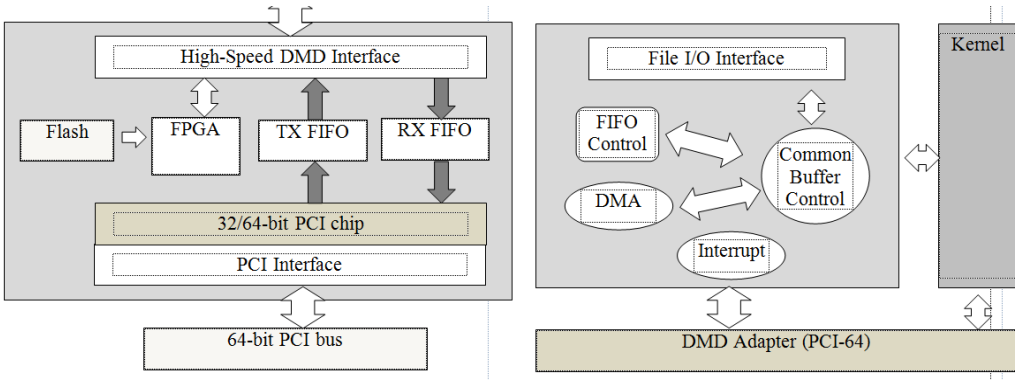
There is a trade-off between the size of the DMA transfer and the number of transfers. Smaller transfer size requires more transfer cycles, which can degrade the efficiency of transmission and performance of DMA interrupt requesting and process scheduling. Conversely, larger amounts of DMA consume more transfer time, which blocks the other interrupts from being processed normally. The traditional trade-off solution is to choose proper DMA size. With the help of PCI arbiter and optimized design for devices, the problems of interrupt and bus competition can be resolved properly. So DMA burst size must be increased to decrease the total number of interrupts in order to increase the transmitting efficiency. Using DMA mechanism can greatly increase the throughput to or from a device, because a great deal of computational overhead is eliminated. However, the main issue that arises with DMA buffers is that, when they are bigger than one block, they must occupy contiguous blocks in physical memory. The architecture of the tightly coupled DMA and interrupt processing can increase the transmission performance than that of traditional technology. Let $T_{frame}$ as the time for one frame transmission rate using DMA as shown in Figure 3(b), then minimum time interval between consecutive SYNC signal, $T_{sync}$, is as following;

$$T_{sync} \geq \alpha\, T_{frame} \quad \text{where} \quad \alpha > 1$$

From the above equation, we can see that the closer to 1 the value of $\alpha$, the shorter $T_{fin}(=T_{sync} - T_{frame})$, so this makes higher risk of DMA system processing error. Therefore, it will be necessary to keep the value of $\alpha$ at least above 1 in case of maintaining transmission rate to DMD unit with constant value of $T_{sync}$. We can also get the performance and stability for processing DMA transfer by adjusting $T_{frame}$ interval.

There can exist some race conditions between fetch and dispatch routine, therefore we use system semaphore synchronization function for mutual exclusion to protect nondeterministic behavior of buffer control. Processing to overflow and underflow on buffer management is also considered to make sure flow control between DMD adapter and DMD unit. To reduce few occasional suspend of transfer interruption due to the difference between the rate of disk read and PCI transfer, appropriate size of common buffer in kernel level should be allocated. Managing buffer control in concurrent processing for DMA transfer and disk reading is an important and critical problem, so we use the semaphore mutual exclusion technique to adjust shared memory access with some atomic variables.




a. DMD Adapter Block Diagram b. Interfaces between Kernel and Hardware I/O

**Figure 4. Hardware and Software Diagram of DMD Adapter**

## 4. Implementation and Analysis

a. DMD Adapter Board b. DMD Unit Board

**Figure 5. Implementation of DMD Adapter and DMD Unit**

The hardware architecture for DMD data transmission includes PCI 32/64-bit interface with 33/66MHz bus clock, high-speed FIFO with DMA operation [5], fiber optic transmission connection and TI(Texas Instruments) 0.7 XGA DMD hardware board [4]. The DMD interface board we developed is composed of high-speed DMD interface with optical fiber media, FIFO and PCI interface chip as shown in Figure 4(a). This PCI interface chip has a switched PCI architecture and maximizes transfer concurrency, so improves the efficiency and burst performance on the PCI and processor busses. Along with a revolutionary switched PCI architecture, this features integrated PCI-to-PCI bridging, concurrent prefetch read capability, multi-port DMA operation and hot swap support.

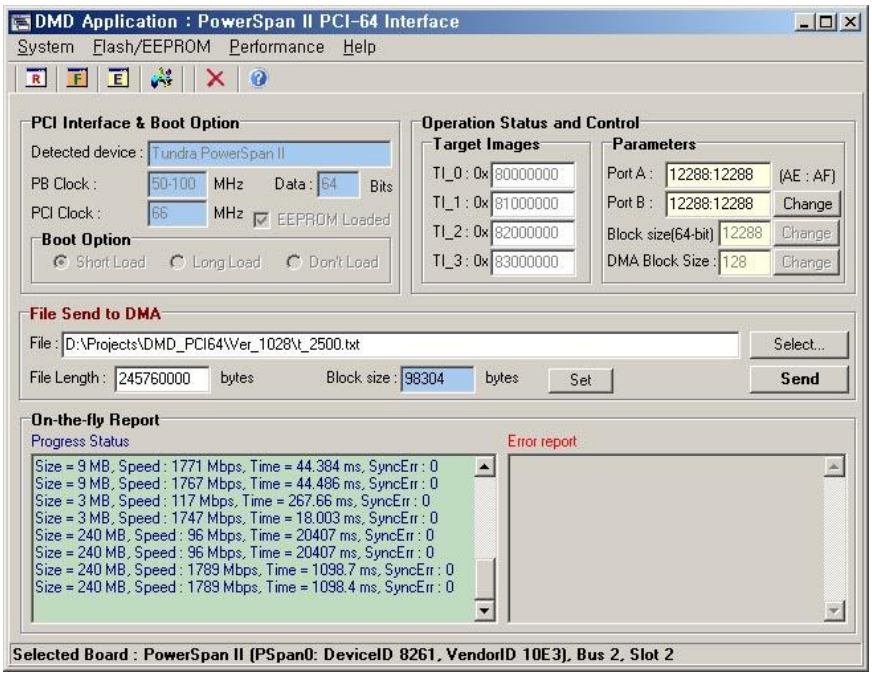



**Figure 6. Experiment Result for PCI-64 66MHz on Windows Environment**

The software interfaces between kernel and hardware I/O are shown as Figure 4(b). The Windows executive represents files by file objects, which are executive objects that are managed by the object manager. Every time we open a handle to a file, the Windows executive creates a file object that represents the file, and it returns an open handle to that object. We can fetch pattern data from the file handle directly without application buffer copy (File I/O Interface). The copy rate from file is lower than memory read due to the low speed disk interface. Recently, however, it becomes common to use solid-state drive (SSD) as storage devices, so the transfer rate will be much faster than that of traditional hard disk. During initialization, the driver checks available resources and allocates continuous memory for DMA transfer. The size of allocated physical memory is depend on total system memory, and our implementation involves 1,024 frames in size for kernel buffer.

After some block read from a DMD-specific file which includes huge pattern data, DMA transfer to FIFO on DMD adapter will be started when on-demand SYNC signal occurred. Our device has bus-mastering capability, it has the necessary electronics to access main memory if we have some basic facts whether we're performing an input or

an output operation. The common buffer within kernel level can be allocated having physical continuous space when kernel mode device driver is started.

The DMD unit we used has a characteristic which can generate the first SYNC signal while at least one frame was already received from DMD adapter before. After the initialization state, the implemented application requests device driver to fetch pattern data as much as possible from disk to store in kernel buffer and the driver is waiting SYNC signal from DMD unit. As shown in Figure 2, control unit signals running(OP Run) command to all prepared devices to make the operation of frame transfer.

For example, amount of pattern data for one semiconductor wafer requires about 250,000 frames, and single frame for full XGA (1,024x768) bit pattern captures 98,304 bytes for our experimental DMD system. To complete total frames in 1 minute it requires the rate of 4,000 frames/sec, and 1,000 frames/sec transfer rate for 4 minutes.

We first designed a DMD adapter board for PCI 64-bit interface operating at 66MHz, and implemented as shown in Figure 5. This adapter has peak bandwidth capability of 533 MBytes/sec, which is enough for transmission of 1,000 or 4,000 frames/sec. However, there exists a bottleneck in the system where fetching the pattern data from hard disk file, so it limits the overall pumping speed. On the average, for hard disk driver, once the head is positioned, when reading or writing a continuous track, an enterprise disk driver can transfer data at about 140 MBytes/s. In practice transfer speeds are many times lower due to constant seeking, as files are read from various locations or they are fragmented. Data transfer rate depends also upon rotational speed, which can range from 4,200 to 15,000 rpm. The fetch rate of our disk file was measured about 100 MBytes/sec. This means we should improve the file read interface with another higher performance device, such as solid state drive (SSD).

We made an application program to test our algorithm on Windows platform. Our experimental results for 2,500 frames pattern data are shown in Figure 6, where we can see the transfer rate is about up to 1.8Gbps speed of transmission with 64-bit, 66 MHz PCI operation, which means about 42% of hardware bandwidth, while other normal programs reached at most 20% of full bandwidth.

## 5. Conclusion

In this paper, we present high performance data pumping method for transferring huge data to DMD unit on-demand, and implement on Windows-based system as kernel driver for 64-bit PCI interface. To get a high-performance, we use high priority interrupt access with DMA access, and direct file access in device driver level. Our technique increases data transfer and makes a processing mechanism of tightly coupled data transmission architecture. We can see that the performance is about 42% of PCI full bandwidth which is much higher than that of other application access of system.

To increase the performance of huge data transfer with PCI interface, it is desirable to use PCI-Express and SSD interface.

## References

[1] D. Dudley, W. Duncan, J. Slaughter, Emerging Digital Micromirror Device (DMD) Applications Proc. SPIE 4985, MOEMS Display and Imaging Systems, **(2003)** January.
[2] S.-H. Voss and M. Talmi, “Lossless high-speed data compression for optical interconnects as used in maskless lithography systems”, Microelectronic Engineering, **(2006)**, pp. 972-975.
[3] PCI Local Bus Specification, Rev. 2.2, **(1998)**.
[4] http://www.ti.com
[5] Tundra, PowerSpan-II PowerPC-to-PCI Bus Switch User Manual, **(2003)**.

[6] Y. Jin, K. Park, J. Choi, S. Kim, C. An and M. Seo, "Region-based Pattern Generating System for Maskless Photolithography", ICCAS 2005, **(2005)** June, pp. 389-392.
[7] S. Vossa, M. Talmia, J. Sanitera, J. Eindorfa, A. Reisiga, J. Heinitzb and E. Haugeneder, "High-speed data storage and processing for projection mask-less lithography systems", Microelectronic Engineering, Proc. of the 31st International Conference on Micro- and Nano-Engineering, **(2006)**, vol. 83, pp. 976-979.
[8] D. V. Bernardo and D. B. Hoang, "A Pragmatic Approach: Achieving Acceptable Security Mechanisms for High Speed Data Transfer Protocol- UDT", International Journal of Security and Its Applications, vol. 4, no.4, **(2010),** pp. 1-16.
[9] M. Choi, F. A. Alisherov, S.-H. Jeon and F. Y. Sattarova, "Review on Transfer and Processing of the Continuous Information", International Journal of Security and Its Applications, International Journal of Smart Home, vol. 3, no.4, **(2009),** pp. 23-42.
[10] Y. Lu, G. Mapili, G. Suhali, S. Chen and K. Roy, "A digital micro-mirror device-based system for the microfabrication of complex", spatially patterned tissue engineering scaffolds, Wiley Periodicals, pp. 396-405, **(2006)**.

## Author

**Hee-Dong Park,** He received M.S. degree in Computer Engineering from Pohang Institute of Science and Technology (POSTECH), and Ph.D. degree in Computer Science from Gyeongsang National University. He was a research staff in Electronics and Telecommunications Research Institute (ETRI). He is now an associate professor in Department of Information & Communications, JoongBu University. His research interests include computer network, parallel and distributed computing, embedded systems, and system software.