# Controlling the Grating Light Valve™ in Real-Time Applications

**Greg Myatt, Silicon Light Machines, Sunnyvale CA**

## Introduction

The Grating Light Valve (GLV) has significantly faster switching characteristics than most other multi-channel spatial light modulators (SLMs). The GLV technology can support modulation rates up to 1 MHz and GLV products are currently available that operate up to 350 kHz with 10-bit amplitude gray-scale pixel data and 500 kHz when using binary pixel data.

New applications are emerging for spatial light modulators where pixel data is computed in real-time in response to real-time sensor data. For example, a system may have a sensor and a spatial light modulator, where the next output pattern of the spatial light modulator (or array of pixel data) is computed in real-time based on the last image captured from a camera. After the spatial light modulator is updated with a new output pattern then an altered image is produced. A real-time application may implement a continuous cycle of:

- Sensor acquisition
- SLM pattern computation
- SLM pattern update to modify image

To take advantage of the GLV's high modulation rates, the GLV's control electronics (i.e. controller board) must write pixel data to the GLV Module at very high data rates.

Past application spaces for the GLV have typically been applications where pixel data has been computed prior to the imaging operations. Applications such as computer to plate printing, 3D printing, mask-less lithography, display, and laser marking all operate with pixel data that can be computed prior to the run-time imaging operation.

During run time in these applications, the GLV controller board reads the pre-determined pixel data from the on-board pixel memory and then writes the data to the GLV. This pre-determined pixel data can be written to the controller's on-board memory during initialization or during time breaks in the imaging operations (such as between layers in 3D printing, or between scans in mask-less lithography). So in applications using pre-determined pixel data, the minimum achievable modulation period encompasses reading the pixel data from the controller's on-board pixel memory and writing it to the GLV Module. In this case, the pixel memory is typically large and only needs to have a single port to support reading from pixel memory and transferring the data to the GLV module.

## Cosmo Test Board for support Applications using Pre-Determined Pixel Data

Silicon Light Machines currently has a GLV evaluation kit to support applications using pre-determined pixel data. The Cosmo test board performs the role of the controller board in the evaluation kit. The Cosmo can support the full modulation rates of 350 kHz for the 1088 pixel GLV Module when operating with pre-determined data. A simplified block diagram of the Cosmo test board and a typical evaluation test bench is illustrated in figure 1. The simplified block diagram focuses on the path of the pixel data.

For more information on the Cosmo Test Board see the data sheet at:

- http://www.siliconlight.com/wp-content/uploads/2014/12/T1088-HS.pdf

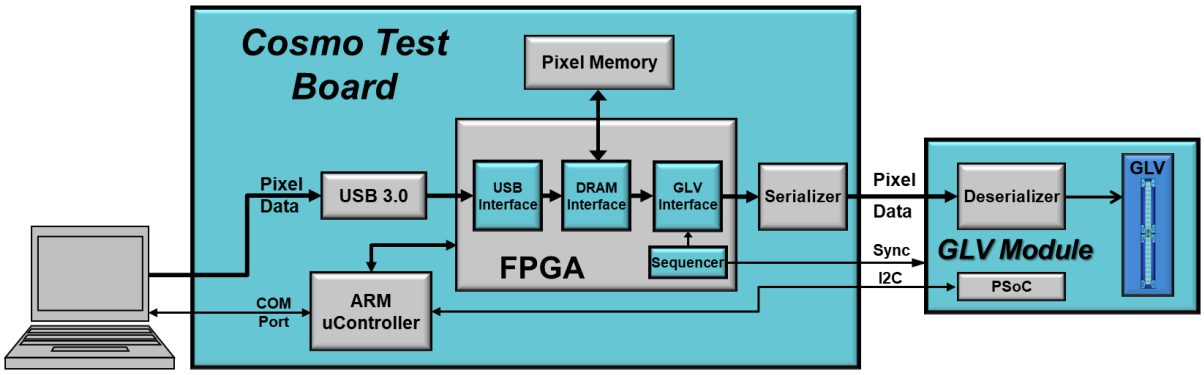


**Figure 1: GLV Controller Board for Pre-Determined Pixel Data (Cosmo Test Board)**

Typically, the Windows computer sends GLV pixel data to the Cosmo board via the USB 3.0 interface prior to run time. The data is stored in pixel memory. The pixel memory is implemented in DRAM memory. The advantage of DRAM is that the pixel memory is large and can store up to 65k lines of pixel data. (Note: the evaluation kit for the 1088-pixel GLV has been recently been upgraded to support 131k lines of pixel data.) After the pixel memory has been pre-loaded, a sequencer command is sent from the host computer to the Cosmo's embedded ARM uController, which in turn sets up the sequencer that is resident in the FPGA. Once the sequencer is running, the sequencer reads the pixel data from pixel memory and writes it to the GLV Module. In this manner the controller can supply pixel data at the maximum specified modulation rates of the GLV Module.

Although the Cosmo can support full modulation rates when the pixel memory is pre-loaded, it has the following limitations when attempting to support real-time applications:

- Windows operating system is not real-time
- Speed and latency of the USB 3.0 interface
- Pixel memory is implemented with single port memory

*Windows:* The Windows operating system is a multi-tasking operating system and has limitations when attempting to support real-time applications. The Window operating system can have large delays in sending the next line of pixel data if other threads are running.

*USB 3.0:* The required data rate to support the full modulation rate (350 kHz) of the 1088 pixel GLV is 11.5 Gbps (4kbyte packet per line of pixel data) which is faster than USB 3.0 maximum theoretical data rate of 5 Gbps. In addition, USB can have significant latencies especially if multiple devices are sharing the same USB hub.

*Pixel Memory*: The Cosmo's pixel memory was optimized for a large capacity and for transferring pre-stored data from the pixel memory to the GLV Module at the maximum modulation rate. The pixel memory was implemented with DRAM chips and has a single data port. So, write and read operations must share the bandwidth over the DRAM data port. Sharing the port can potentially limit the modulation rates in real-time applications, especially when latency from Windows or USB may prevent tightly synchronized write and read operations.

## Cosmo Test Board Modification for Support of a Specific Real-Time Application

The GLV along with the Cosmo test board has been used in scientific experiments in universities and corporations in various locations around the world. In particular, the University of Colorado used the GLV and Cosmo for their real-time application of wavefront shaping in biological tissue. The 1088 pixel GLV was configured as a phase modulator and both the Cosmo firmware and FPGA design were modified to support their application. The GLV system allowed for unprecedented high-speed waveform measurements. The results of the experiments were published Nature Photonics:

> Wavefront shaping in complex media with a 350 kHz modulator via a 1D-to-2D transform
> O Tzang, E Niv, S Singh, S Labouesse, G Myatt, R Piestun
> Nature Photonics 13 (11), 788-793

The electronics system configuration was similar to the illustration in Figure 1. In addition, the system had a sensor with a data acquisition card. The Cosmo software and FPGA was modified for this application to allow the host computer to write a new line of pixel data via the USB interface into the pixel memory without stopping the sequencer. During system initialization, a group of fixed patterns were downloaded to the Cosmo pixel memory. During operation the system runs in cycles that consisted of the following steps: (1) Cosmo sequencer writes the group of fixed patterns to GLV, (2) DAQ acquires the sensor data, (3) a new one-line pattern is computed by the PC and sent to the Cosmo pixel memory via the USB3 interface and (4) upon receiving the new pattern the Cosmo sequencer writes the new data to the GLV. The cycle time was greatly accelerated by customizing the Cosmo's firmware and FPGA. The cycle times were on the order of 2 to 10 milliseconds.

This application could take advantage of the fast modulation rates of the GLV since the group of fixed pattern operated at the maximum modulation rate of the 1088 pixel GLV (i.e. 350 kHz).

## Real-Time GLV Applications – Processing and Data Transmission

Typical processing and data transmission steps for a GLV real-time application are illustrated in the figure 2. Types of sensors and the algorithms for pattern computation will vary from application to application. The transmission of the pixel data to pixel memory and the subsequent transmission from pixel memory to the GLV are similar among many applications. The requirements for a high-speed application is high-bandwidth and low latency.

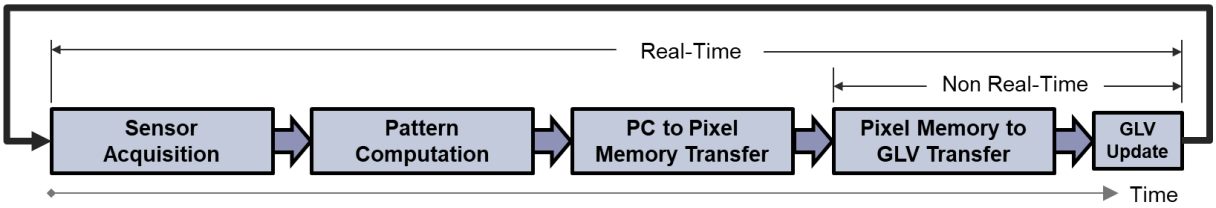


**Figure 2: Sequential Real-Time Processing and Pixel Data Transmission**

The current high-speed 1088 GLV Module can transfer the pixel data from pixel memory to the GLV internal registers in under 2.86 microseconds. Afterwards the newly registered data can be simultaneously updated onto the GLV ribbons with a 10 nanosecond strobe. This timing can be achieved with the current Cosmo board when using pre-stored pixel data. Since real-time applications have extra steps that take time within the application's cycle, the full modulation rate of the GLV cannot be achieved. However, the goal is to minimize the time consumed by the additional steps required for real-time applications by optimizing the real-time controller. The modulation period is computed as:

- Modulation Period = Sensor_Acquisition_Time + Pattern_Computation_Time + PC_to_Memory_Transfer_Time + Memory_to_GLV_Transfer_Time

## Optimizing Real-Time Performance in Prototypes with Host Computer

The objectives of high modulation rates in real-time applications can best be achieved with dedicated hardware. However, many first prototypes of systems are developed with a host computer in the system architecture. The host computer allows utilization of off-the-shelf components and a flexible platform for developing algorithms. The downside is the latency caused by sharing the host PC's hardware resources. Latency is especially prevalent in the Windows multi-tasking operating system. Latency can be somewhat unpredictable depending on what other tasks are running on the operating system, making the goal of repeatable cycle times difficult to achieve. Steps that can help to reduce latency are turning off unneeded tasks, such as the Ethernet connection to the network.

Since many prototypes use a host computer in their real-time systems, Silicon Light Machines is developing a real-time controller which includes a Windows computer in the system architecture. To enhance the real-time performance, two specific enhancements over the Cosmo board are being made to reduce the time to write pixel data into the pixel memory:

- Replace USB 3.0 interface with 4 lanes of PCIe 3.0 interface
- Replace single port DRAM with embedded dual port RAM

PCIe 3.0 interfaces are currently available in Windows PCs today. Each PCIe 3.0 lanes offers 8.0 GTps data rates so a 4x lane implementation would offer 32GTps. Therefore, the theoretical PCIe write time for one line of pixel data should decrease by a factor of ten as compared to the Cosmos's USB 3.0 interface. In addition, PCIe offers lower latency than the USB interface.

The pixel memory will be implemented in dual port RAM and will be embedded in an FPGA. The advantage of dual port memory allows the user to write to pixel memory while the GLV downlink controller reads data from the pixel memory. With dual port memory these write and read events do not need to be tightly synchronized. In addition, the GLV has uninterrupted access to the pixel memory so that it can operate at the full data rate supported by the module.

The initial controller board will use a FPGA Evaluation Board as the development platform. Silicon Light Machines will develop the customizations listed below to enable the FPGA evaluation board to support GLV control in real-time applications. A block diagram is illustrated in Figure 3.

- FPGA Mezzanine Card (FMC) to provide connectivity to the GLV Module
- FPGA design to support real-time GLV applications
- Dynamic link library (.DLL) with API to initialize and operate the GLV

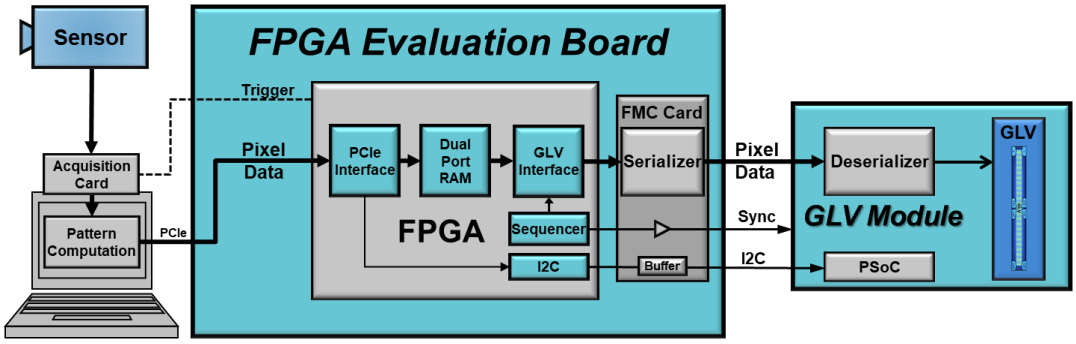


**Figure 3: Real-Time GLV Controller Board using FPGA Evaluation Board**

The size of the FPGA Evaluation Board plus the FMC card will require that the cover of the host PC be removed for installation and operation. So, the FPGA Evaluation Board solution is intended for prototyping and scientific experiments and not for productization.

To support products, future architectures may include a Thunderbolt interface. The Thunderbolt 3 interface can transmit 4x lanes of PCIe 3.0 data at 32 GTps. The Thunderbolt interface would allow the GLV controller to be mechanically independent of host computer and would provide a high-bandwidth with a convenient cable connection. An ARM microcontroller could be added to take care of GLV Module initialization. Figure 4 illustrates one possible implementation of a real-time controller product.

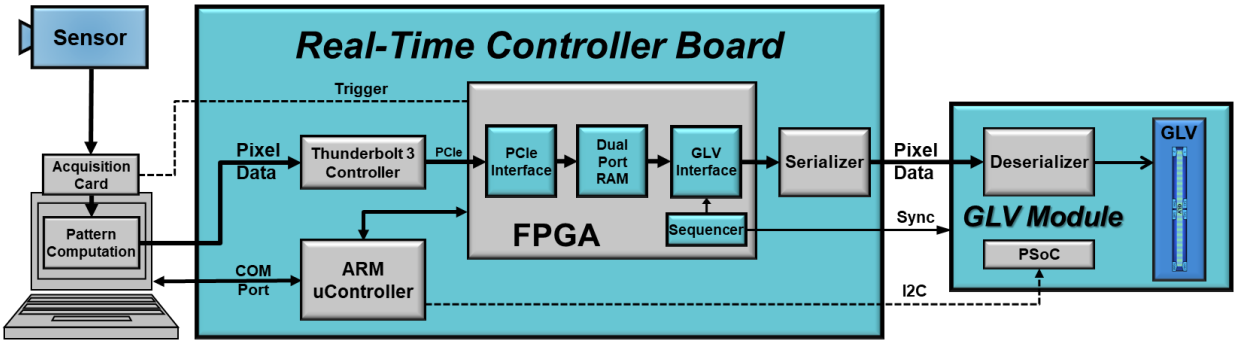


**Figure 4: Real-Time GLV Controller Board with Host Computer**

## Architecture for a Real-Time Controller Board with High Modulation Rates

The pixel memory to GLV transfer in the 350 kHz High-Speed Buddy GLV Module takes most of the 2.86 microseconds of the modulation period. Therefore, if the additional real-time processes are performed serially then the full modulation rate of the GLV could not be achieved. In a host computer system the processing and data transfers typically get serialized due to general purpose processors (see figure 2). That is, acquiring all pixel data from sensor prior to starting the pattern computation process. And then, completing the pattern computation for all pixels prior to starting the transfer of pixel data to the GLV.

Achieving maximum modulation rates in a real-time GLV application can best be achieved with dedicated custom hardware. Further optimization could be achieved by parallel processing with digital logic in an FPGA or ASIC. In a GLV real-time application this means executing the sensor acquisition, pattern computation and data transmission processes in parallel (see figure 5).

This would mean that the new GLV line pattern computation starts shortly after the sensor acquisition starts and the pixel data transfer starts shortly after the pattern computation starts.

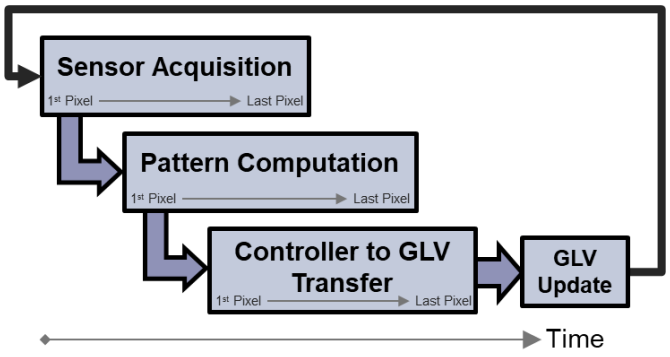


**Figure 5: Parallel Real-Time Processing and Pixel Data Transmission**

For example, if the pixel's amplitude value of GLV's pixel number 1 can be computed based on sensor pixel number 1 thru 5. Then the pixel data transfer to the GLV could potentially start after the sensor data of pixels 1 thru 5 was acquired and the pixel value for GLV pixel number 1 was computed.

This concept may not apply to all real-time applications where all (or most) of the sensor data is required for computation of the first GLV pixel.

The 1088-pixel High-Speed Buddy Module has 4 drivers (each with 272 high-voltage drive channels), where 2 drivers drive pixels 1 to 544 and 2 drivers drive pixels 545 to 1088. To achieve high speed all 4 drivers are written at a time (i.e. pixels 1, 2, 545 & 545 are written on one write cycle). So, the parallel processing described above may be more easily achieved by just using pixels 1 to 544.

Figure 6 shows an example block diagram where a parallel processing engine is implemented in a FPGA. The host computer in this case is not part of the pixel data path but only used for initializing the system.

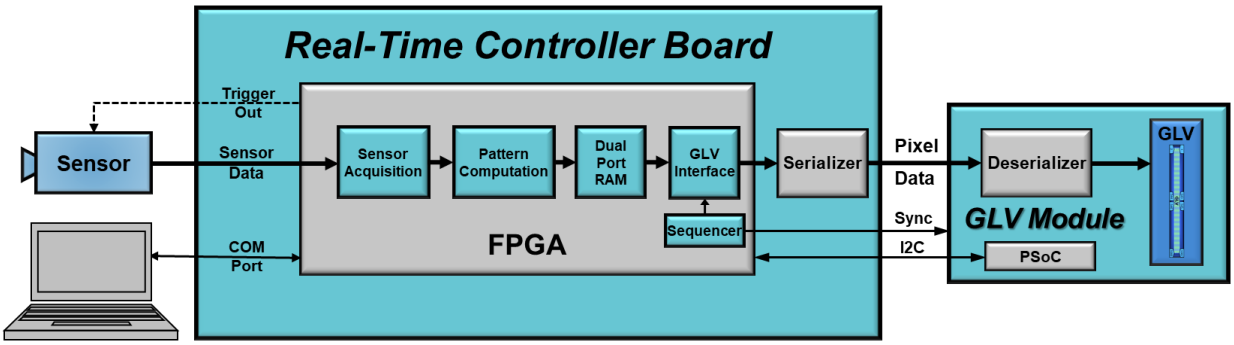


**Figure 6: Real-Time GLV Controller Board for High Modulation Rates**

## Calibration

The GLV is an analog device where more voltage results in more deflection of the ribbon. Like most analog devices, calibration is required to achieve optimal performance. The goal of calibrating the GLV is to provide uniformity across the array of pixels. However, the calibration process can go beyond just calibrating the GLV. The process can compensate for non-idealities in most of the optical path, so that a uniform line image can be projected onto the focal plane.

Uncalibrated GLV pixel data would typically be converted to calibrated pixel data at the end of the pixel computation step. The conversion to a calibrated value can be done with a look up table (LUT). When a host computer is executing the pixel computation step, then the host computer can also convert uncalibrated pixels to calibrated pixel values. In custom parallel processing engine, a embedded LUT could do the conversion. A calibration LUT can be large when compared to embedded FPGA memories. The size of the calibration LUT is a product of:

- Number of Pixels * Number of Amplitude Levels * Amplitude Resolution

So, in a 1088 pixel system with full 10-bit resolution the number of memory bits would be ~11.1 M-bits. But since the numbers 1088 & 10 do not fit efficiently into typical memory architectures (power of 2), then actual implementation size of the LUT could be much larger.

## Conclusions

In real-time applications, added processes in the modulation cycle take time and can limit the modulation rate of the GLV. The architecture of a real-time controller must be optimized to achieve high GLV modulation rates. In prototype systems where a PC is used for sensor acquisition and pattern generation, the PC's general purpose hardware and shared resources can limit system modulation rates. However, custom electronics with dedicated resources and parallel processing can help to maximize the GLV's modulation rates. An optimized real-time controller enables real-time applications which can approach the 350kHz modulation rate of today's GLV's.