7. International Immersive Projection Technologies Workshop
9. Eurographics Workshop on Virtual Environments (2003)
J. Deisinger, A. Kunz (Editors)

Softgenlock: Active Stereo and GenLock for PC Cluster

Jérémie Allard[1]

Valérie Gouranton[2]

Guy Lamarque[3]

Emmanuel Melin[2]

Bruno Raffin[1]

[1] Laboratoire Informatique et Distribution (ID)
CNRS - INPG - INRIA - UJF
38330 Montbonnot, France
[2] Laboratoire d'Informatique Fondamentale d'Orléans (LIFO)
Université d'Orléans
F-45067 Orleans Cedex 2, France
[3] Laboratoire d'Electronique, Signaux, Images (LESI)
Université d'Orléans
F-45067 Orleans Cedex 2, France

Abstract

In this paper, we present SoftGenLock, an open source software that enables genlock and active stereo on commodity graphics cards. SoftGenLock is implemented on top of Linux. It does not require any hardware modification of the graphics card. Rather than to gain total control on signal generation, which would make the software deeply dependent on the graphics card specification, SoftGenLock applies continuous small modifications to converge and maintain genlocked video signals. To be properly synchronized with each video retrace, SoftGenLock is executed as a real-time task. The genlock signal is propagated along the different machines using the parallel port, a low latency device present on all PCs. It results in a software that only requires access to few specific registers on a graphics card: it can be ported with minimal effort on potentially any graphics card.

Categories and Subject Descriptors (according to ACM CCS): I.3.2 [Computer Graphics]: Distributed/network graphics H.5.1 [Information Interfaces and Presentation]: Artificial, augmented, and virtual realities

Keywords: Active Stereo; Genlock; Real-time; Immersive Projection Environment; PC Cluster.

1. Introduction

Immersive environments [12] are classically powered by dedicated graphics supercomputers like SGI Onyx machines [15]. However, today the anatomy of super-computing is quickly and deeply changing. Clusters of commodity components are becoming the leading choice architecture. They are scalable and modular with a high performance-price ratio. Clusters range from a few low-end machines connected through an Ethernet network to thousands of 64-bit processors using Myrinet or Quadrix interconnects. These architectures proved efficient for classical (non interactive) intensive computations. Recently the availability of low cost high performance graphics cards have foster researches to use clusters to run interactive virtual reality applications in multi-projector immersive environments [6, 8, 14, 16].

One difficulty is to harness the distributed resources (CPU, memory, GPU, projector) such that the images displayed with the different projectors appear as a single high

resolution image. Three levels of synchronization can be identified:

- *DataLock* ensures the images on each node are computed on coherent data sets. For example, the user position should be the same for all nodes.
- *SwapLock* ensures the images computed on each node are released at the same time, i.e. the buffer swaps are synchronized.
- *GenLock* ensures video signals are synchronized. Synchronization can occur at a pixel level, line or frame level. In this paper we consider a frame level genlock.

Several approaches have been developed to ensure proper datalock and swaplock. A master server can ensure datalock by broadcasting graphics primitives to the render nodes [14]. An other approach consists in duplicating data on each node at starting time and broadcasting any subsequent data modification event [6, 11, 17]. A synchronization barrier, executed on a classical Fast Ethernet network just before the buffer swap, is sufficient to ensure a proper swaplock [6, 11, 17]. Genlock, as it concerns video signal generation, is close to the hardware and then much more difficult to ensure with a software only approach. Some vendors, like 3DLabs or nVIDIA (the recently announced quadro FX), develop high-end graphics cards with built-in support for genlock. Other companies like Artabel, Orad or MetaVR, modify existing graphics cards. However, main commodity graphics cards from nVIDIA, ATI or Matrox, do not support genlock. There is no strong technical limitation to provide this feature at a hardware level. It is rather an economical reason as the very small size of the market does not justify the extra cost incurred : extra hardware, extra connector for genlock signal, extra space to locate this connector on the graphics card, and drivers that support it.

A frame level genlock is mandatory for active stereo [8]. Left and right eye images are displayed alternatively. Swap occurs during the vertical blanking, i.e. when no pixel is generated before the next video retrace starts. The user wears glasses with shutters opening alternatively to let him see the right eye images with the right eye only and vice-versa. If genlock is not properly ensured, the user may see from the same eye both, a right and a left eye image displayed by two different projectors. The quality of stereo is then affected.

The goal of the open source SoftGenLock [7] project was to develop a software approach to enable genlock and active stereo on commodity graphics cards. SoftGenLock, developed for Linux, was first released in June 2001. It has been designed to minimize dependencies on graphics cards. It does not require any hardware modification of the graphics card. The genlock signal is propagated along the different machines using the parallel port. Thus no direct wiring of the graphics card is required. Genlocking and active stereo requires a strong synchronization with video retrace. This is usually ensured at a hardware level. SoftGenLock avoids this dependency by taking advantage of real-time systems like RTAI [4] or RTLinux [9]: SoftGenLock is executed as a real-time task. For stereo, SoftGenLock counts on X to setup a virtual screen twice as large as the actual screen, that can contain side by side the left and right eye images.

During video blanking, it just has to swap the address of the half screen to display (the left or right eye image). The switching signal for the shutter glasses is sent through the parallel port. Genlocking is achieved by applying continuous small modifications to the video signal, adding or removing hidden pixels for example, an approach that requires a limited control on the signal generation. As a result, SoftGenLock only requires access to few specific registers on a graphics card: to detect a specific position of the video retrace, to apply small modifications to the video signal and to change the address of the buffer to display. Thus, supporting a new graphics card does not require a full code rewrite but only to support the new set of registers. Given a family of cards, vendors usually ensures an upward compatibility. In this case, SoftGenLock code is identical for all the family. This is for example the case of all cards based on nVIDIA Geforce/Quadro GPUs.

SoftGenLock design and algorithm are discussed first in section 2. Specific features are detailed in sections 3, 4, 5, 6 and 7, before to end with results in section 8.

2. SoftGenLock Design and Algorithm

Rather than to gain total control on signal generation, SoftGenLock applies continuous small modifications to converge and maintain genlocked video signals. This approach requires minimal control over the graphics card. We basically need two features: a way to inflect the signal generation speed, and a way to periodically detect the signal position (once per frame for a frame level genlock). The former can be obtained by modifying the speed of the pixel clock or the number of pixels to generate. The latter, we call it the *sync event*, is implemented using the vertical blanking interrupt or by pooling a CRTC state register [1]. Information must be exchanged between machines to measure the time gap between the different sync events. The video signals will be considered genlocked if the time gaps are small enough (usually 5-40 μs for good stereo quality). Based on this measure each machine can locally decide to slow down or accelerate the video signal to reduce this time gap. Amongst the machines one is considered the reference: the master. It never modifies its video-signal. Each other machine, the slaves, measures the time gap between the master sync event and its own sync event and corrects the video signal speed if required.

To obtain a genlock and stereo of quality, SoftGenLock must be executed in real-time. If glasses switch a few hundreds of microseconds after the expected time, because an interruption suspended SoftGenLock execution for example, the stereo quality is affected (switch occurs during video retrace). As we target a genlock with a discrepancy of about

5-40 μs, the time gap must be measured with a precision of a few microseconds. The Linux kernel [10] does not guarantee a process will not be interrupted or will be granted CPU access with a short response time (few μs). The response time of the Linux Kernel 2.4 can be as large as 10 ms. Thus, Linux does not allow SoftGenLock to measure the time gap with the required precision. To satisfy these real-time constraints we run SoftGenLock as a RTLinux [9] or RTAI [4] task.

Before SoftGenLock starts to genlock the video signals and activate stereo, it goes through a calibration step and a local synchronization step. During the calibration step, SoftGenLock evaluates different times, like the time gap between two consecutive sync events. During the local synchronization step, SoftGenLock calibrates its timers to wake up when needed during the video blanking. The goal is to minimize busy waiting to avoid monopolizing a CPU. Next, once per video retrace, SoftGenLock executes the following tasks (the order may change depending on the implementation):

- The master sends a signal to the slaves a soon as it detects its sync event.
- Each slave detects the time t_m it receives the signal from the master. It subtracts from t_m the expected signal transmission (user given, generally about 5μs).
- Each slave measures the time t_l the local sync event occurred.
- Each slave modifies the video signal generation speed if $|t_m - t_l|$ is considered too large (usually about 5-40 μs).
- The machine driving the shutter glasses sends the switching signal.
- All machines switch the address of the image to display (left or right eye image).

3. Real-time System

SoftGenLock is implemented as a real-time task using RTLinux [9] or RTAI [4]. These add-ons to the Linux kernel ensures real-time tasks are triggered with micro-second response times and are not interrupted without their consent.

RTLinux and RTAI share the same main concepts. The real time system implements a basic kernel that handles two kinds of tasks: the Linux kernel and real-time tasks. The real-time kernel does not implement preemptive multi-tasking. Task scheduling must be programmed at the task level. This is the responsibility of the programmer to ensure two tasks do not try to grab the same CPU at the same time. Classically, real-time tasks are activated through a timer interruption: the real-time kernel intercepts the interruption and activate the real-time task that is waiting for that interruption. The task releases the CPU when it is done. No other task will be scheduled during that period of time. The real-time kernel has been designed to ensure the delay between the hardware interruption and the task activation is as small as possible, generally about a microsecond. The Linux kernel is the lowest priority task. It has access to the CPU only when no real-time task asks for it. Hardware interruptions for the Linux kernel are intercepted and stored by the real-time kernel as long as the Linux kernel does not have access to a CPU.

SoftGenLock is programmed as a real-time task. There is no time sharing difficulty as it is the only real-time task. SoftGenLock reprograms the timer interruption to be activated only when required. As a result, its only uses about 50μs of CPU time per video retrace. The guaranteed microsecond reactivity of real-time tasks and the non-preemptive time sharing ensures SoftGenLock actions are all performed on time.

Real-time execution allows a fine-tuning of event triggering, making it possible to optimize genlock and stereo quality for each configuration. For example shutter latency may vary from one model to the other. SoftGenLock can precisely control when the shutter switching signal is sent to minimize ghosting: glasses should switch as late as possible to avoid the opening shutter let the user see the image just drawn, but not too late to ensure the other shutter is closed when the next video retrace starts.

4. Sync Event Detection

SoftGenLock supports two different approaches. The first one detects the vertical blanking pooling the VGA Input Status #1 register [1] (3DAh). To avoid monopolizing a CPU, SoftGenLock uses a real-time timer to wake up just before the sync event is expected. The second approach uses the vertical blanking interrupt. The interrupt is intercepted by RTAI that wakes-up SoftGenLock within about a microsecond. Next, RTAI forwards the interrupt to the Linux kernel as it may be required by the graphics card driver. This second approach only supports nVIDIA cards for the moment.

5. Communication Network

SoftGenLock requirements for the communication network are the following :

- The master has to send a signal (one bit) to the slaves every video retrace.
- The communication time of the signal should not suffer unpredictable variations. A slave must be able to know the time the signal was sent to deduce the time the sync event was detected on the master slave.
- The network should be accessible from a real-time task.

Using a parallel port based network appeared as the best solution. Parallel port is still present at no extra cost on all PCs. The parallel port can be accessed from a real-time task. The signal (a bit) is directly written on a parallel port register and transformed into an electrical signal, without having to go through a protocol layer [2]. It ensures fast transmission times (about 5μs to send a bit) and low variations. Notice that classical high performance networks (Giga Ethernet, Myrinet or SCI for example) are not accessible from

RTAI or RTLinux tasks. This would require specific real-time drivers that are not available.

The parallel port has 8 pins (pins 2-9) that the master can use to send data [2]. On one pin, it is possible to connect several slaves. We did not make intensive tests, but it seems reasonable to connect 4 slaves per pin without requiring any signal amplifier. Thus, we can easily build a cheap and electronics free network for 33 nodes (one master, 8×4 slaves). By default the master writes the signal on all 8 output pins. This is useful as it allows to add or remove slaves online.

We also developed an advanced network based on the TTL_PAPERS design [13]. Using a reprogramable chip, it can implements more complex tasks than just to propagate a signal. The idea behind this advanced network is to have a reconfigurable networks to use SoftGenLock like algorithms for other tasks, like swaplocking or clock synchronization for example.

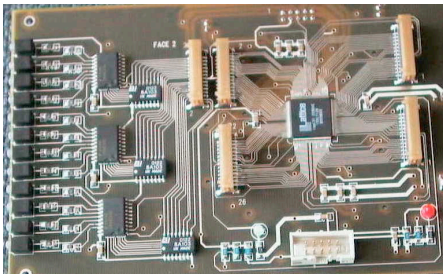

Figure 1: *The basic board with a 2096VE device (front and back side)*

The basic circuit board (Fig. 1) is four layered, the two internal layers are reserved for ground and power planes. The other layers are fully available for signal routing. The architecture is based around two CPLD (Complex Programmable Logic Device) ispLSI 2096VE, one on each side of the board. These devices have 96 general purpose Input/Output pins. These non volatile CPLD are electrically erasable and system programmable. The basic unit of logic on the ispLSI 2096VE devices is the Generic Logic Block (GLB). There are a total of 24 GLBs in the 2096VE device. Each GLB has a programmable AND/OR/Exclusive OR array. Each outputs can be configured to be either combinatorial or registered (96 registers). Even though these devices have a 3.3V low voltage architecture, the signal levels are TTL compatible with standard 5V TTL devices: they are compliant with the IEEE 1284 standard for parallel port. It is possible to interconnect several boards for large clusters. Each additional board only increases communication times by a few nanoseconds.

The hardware design can be programmed with tools like Very high speed Integrated Circuit (VHSIC) or Hardware Description Language (VHDL) for example. Once the program written, a file is generated and loaded to the non volatile Electrically Erasable Programmable Read Only Memory (EEPROM) architecture of the ispLSI.

6. Video Signal Control

Once we determined the time gap between the master sync event and the local sync event, corrections must be applied to accelerate or slow-down the video signal generation. The alteration of the video signal must be done carefully, otherwise the projector detects it : during a few milliseconds it does not display anything to reinitialize its state (the phenomena is classically experienced when changing the resolution of a CRT screen).

Depending on the characteristics of the graphics card, two approaches are supported.

6.1. VGA compatible cards

The VGA standard [1] specifies the registers that control the video signal. Some registers control the number of lines and columns of pixels (CRTC registers 00/06). SoftGenLock accesses these registers to add or remove pixels. When small modifications are required, hidden pixels are added/removed by adding/removing a few columns to some hidden lines. For larger modifications an extra blank line is added (hidden pixels too).

This approach can be used for any graphics card VGA-compatible on register level. It has been successfully used on nVIDIA and S3 cards. Unfortunately, several graphics cards are not compliant (Voodoo, ATI).

As VGA is an old standard, it has several limitations. It only supports one video head on multi-head cards, and some high-resolution modes are not supported. Another important draw-back is the possible interference with the video driver. As SoftGenLock is designed no to require access to the video driver's source code, mutual exclusion between the driver and SoftGenLock cannot be ensured. Accessing a VGA register is not atomic. It is a 2-step operation: first setting the index of the register and then reading/writing the value. Conflicts can occur when SoftGenLock and the driver are concurrently accessing the registers. This may lead to occasional corruptions of the VGA registers, which can lead to a corrupted display, X server crashes, or even hard lock of the entire system. Depending on the hardware environment, these problems can be quite rare (only after several hours of operation), or they can occur after only a few minutes. Resolving this issue require either more knowledge of the graphics card hardware (see next section) or to modify the driver to implement a lock that would guarantee mutual exclusion.

6.2. Pixel Clock Access

The pixel clock is the hardware component used to activate the output of each pixel of the video signal. Depending on

the graphics card model, it is possible to alter its speed. This requires knowledge on the hardware, but provides a very efficient way to control the video signal. For the moment only nVIDIA cards are supported, but this approach can easily be adapted to other models.

The pixel clock is based on a static base clock divided by a programmable factor. On nVIDIA cards, a register (offset 0x680508 in the MMIO area) contains 3 parameters M, N and P, which are used to specify the frequency using the following formula:

$$\text{Pixel Frequency} = \frac{\text{BaseFreq} * N}{M * 2^P}$$

To sync the master sync event and the local sync event, SoftGenLock modifies this register for a small period of time during the vertical blanking.

This approach supports any resolution. Both video heads can be controlled. Writing in this register is atomic so there is no concurrency issues.

7. Active Stereo

Figure 2: *A parallel cable to genlock 2 machines. Beside the two DB25 connectors, we can distinguish a female mini-din 3 connector to plug the IR emitter of the shutter glasses.*

Active stereo requires displaying alternatively the left eye and right eye images with shutter glasses masking the user's opposite eye. Image and shutter switching must occur before each video retrace:

- *Glasses Control:* The signal to control the shutter glasses is written on a pin of the parallel port and forwarded to the glasses with an adapted connector (Fig 2). It can be written at a different address if, for example, the graphics card has a dedicated output for the glasses. The time the signal is sent is critical to ensure the wrong image is not seen while shutters are switching. A microsecond precision is reached using RTAI or RTLinux. Experience shows that stereo signal should be sent about 700 μs before the next video retrace starts (this may vary depending on the glasses).
- *Image Switching:* SoftGenLock provides left/right image switching without requiring any specific hardware or driver support. XFree86 is set to have a virtual screen twice as large as the actual screen (a simple modification in the XF86Config file). We thus have a double size frame buffer. The 3D software must be configured to write the left eye image on the left half screen and the right eye image on the right half screen. This can be done through a simple modification of configuration files with Net Juggler [6]. During the vertical retrace, SoftGenLock switches the starting address of the half screen to display. This is done writing into a VGA register (CRTC registers 0C/0D) or a nVIDIA-specific register.

8. Results

The current implementation supports all cards having VGA registers. SoftGenLock is mainly used with nVIDIA cards (Geforce 2, Quadro 2, Geforce 4, Quadro 4), but was also tested with other VGA cards from S3 for example. It has been used for genlocking and active stereo with up to 6 machines. It requires about 50μs of CPU time per video retrace. SoftGenLock can be adapted to use non VGA registers, assuming the required hardware specification is available. We led developments to support specific nVIDIA registers. Only one output is VGA compliant on nVIDIA cards (the DVI one on GeForce 4 models). Using nVIDIA registers, SoftGenLock proved more reliable and can use the second output (one at a time for the moment, both at the same time in the future). Other non VGA cards, in particular from ATI, Matrox or 3Dlabs, are not supported for the moment.

Hardware specifications required to port SoftGenLock on a new graphics card are basics features that can be found in the 2D open source drivers that are usually available on Linux. We ported SoftGenLock on nVIDIA graphics cards using the informations from the nv driver of The XFree86 Project [5] and the RivaTV [3] developer resources.

9. Conclusion

Today's commodity graphics cards have reached a level of quality that allow them to be used in immersive environments. However, multi-projector displays and active stereo require genlocked video outputs, a feature only supported by some high-end graphics cards. In opposite to these dedicated hardware approaches, SoftGenLock minimizes hardware dependencies by implementing a genlock algorithm on top of real-time Linux systems. It results in a software that can enable genlock and active stereo for potentially any graphics card, assuming the required hardware specification is available.

Future works will investigate possibilities to support directly OpenGL stereo mode. Two directions will be investigated: to associate SoftGenLock with the stereo mode some drivers support, or to emulate quad-buffer stereo using an OpenGL wrapper library.

References

1. Hardware Level VGA and SVGA Video Programming Information Page. http://web.inter.nl.net/hcc/S.Weijgers/FreeVGA/home.htm.
2. Interfacing the Standard Parallel Port. http://www.beyondlogic.org/spp/parallel.htm.
3. RivaTV. http://rivatv.sourceforge.net/.
4. The RTAI Manual. http://www.aero.polimi.it/ rtai/.
5. The XFree86 Project. http://www.xfree86.org/.
6. J. Allard, V. Gouranton, L. Lecointre, E. Melin, and B. Raffin. Net Juggler: Running VR Juggler with Multiple Displays on a Commodity Component Cluster. In *IEEE VR*, pages 275–276, Orlando, USA, March 2002.
7. J. Allard, V. Gouranton, E. Melin, and B. Raffin. Softgenlock manual: Software active stereo and genlock for linux, 2002. http://netjuggler.sourceforge.net.
8. P. Augerat, C. Goudeseune, H. Kaczmarski, B. Raffin, B. Schaeffer, L. Soares, and M. K. Zuffo. Commodity clusters for immersive projection environments. Siggraph 2002 Course, July 2002.
9. M. Barabanov and V. Yodaiken. Real-Time Linux, 1996. http://www.fsmlabs.com.
10. D. P. Bovet and M. Cesati. *Understanding the Linux Kernel*. O'Reilly, 2001.
11. M. Bues, R. Blach, S. Stegmaier, U. Häfner, H. Hoffmann, and F. Haselberger. Towards a Scalable High Performance Application Platform for Immersive Virtual Environements. In *Immersive Projection Technology and Virtual Environements 2001*, pages 165–174, Stuttgart, Germany, May 2001. Springer.
12. C. Cruz-Neira, D. J. Sandin, T. A. DeFanti, R. V. Kenyon, and J. C. Hart. The Cave Audio VIsual Experience Automatic Virtual Environement. *Communication of the ACM*, 35(6):64–72, 1992.
13. H. G. Dietz, R. Hoare, and T. Mattox. A Fine-Grain Parallel Architecture Based On Barrier Synchronization. In *Proceedings of the International Conference on Parallel Processing*, pages 247–250, 1996.
14. G. Humphreys, M. Eldridge, I. Buck, G. Stoll, M. Everett, and P. Hanrahan. WireGL: A Scalable Graphics System for Clusters. In *Proceedings of SIGGRAPH 2001*, 2001.
15. J. Montrym, D. Baum, D. Dignam, and C. Migdal. InfiniteReality: A Real-Time Graphics System. In *Computer Graphics (SIGGRAPH 97)*, pages 293–303. ACM Press, August 1997.
16. R. Samanta, T. Funkhouser, K. Li, and J. P. Singh. Hybrid Sort-First and Sort-Last Parallel Rendering with a Cluster of PCs. In *SIGGRAPH/Eurographics Workshop on Graphics Hardware*, August 2000.
17. B. Schaeffer. Networking Management Framewoks for Cluster-Based Graphics. http://www.isl.uiuc.edu/ClusteredVR/ClusteredVR.htm, 2002.