



Software Graphics Processing Unit (sGPU) Development and Evaluation Report

*George A. Salazar
Johnson Space Center, Houston, Texas*

*Glen. F. Steele
Johnson Space Center, Houston, Texas*

National Aeronautics and
Space Administration

*Johnson Space Center
Houston, Texas 77058*

The use of trademarks or names of manufacturers in this report is for accurate reporting and does not constitute an official endorsement, either expressed or implied, of such products or manufacturers by the National Aeronautics and Space Administration.

NASA STI Program ... in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA scientific and technical information (STI) program plays a key part in helping NASA maintain this important role.

The NASA STI program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI program provides access to the NASA Aeronautics and Space Database and its public interface, the NASA Technical Report Server, thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- **TECHNICAL PUBLICATION.** Reports of completed research or a major significant phase of research that present the results of NASA Programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.
- **TECHNICAL MEMORANDUM.** Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.
- **CONTRACTOR REPORT.** Scientific and technical findings by NASA-sponsored contractors and grantees.

- **CONFERENCE PUBLICATION.** Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.
- **SPECIAL PUBLICATION.** Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.
- **TECHNICAL TRANSLATION.** English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include creating custom thesauri, building customized databases, and organizing and publishing research results.

For more information about the NASA STI program, see the following:

- Access the NASA STI program home page at <http://www.sti.nasa.gov>
- E-mail your question via the Internet to help@sti.nasa.gov
- Fax your question to the NASA STI Help Desk at 443-757-5803
- Phone the NASA STI Help Desk at 443-757-5802
- Write to:
NASA Center for AeroSpace Information
7115 Standard Drive
Hanover, MD 21076-1320



Software Graphics Processing Unit (sGPU) Development and Evaluation Report

*George A. Salazar
Johnson Space Center, Houston, Texas*

*Glen. F. Steele
Johnson Space Center, Houston, Texas*

National Aeronautics and
Space Administration

*Johnson Space Center
Houston, Texas 77058*

The use of trademarks or names of manufacturers in this report is for accurate reporting and does not constitute an official endorsement, either expressed or implied, of such products or manufacturers by the National Aeronautics and Space Administration.

Available from:

NASA Center for AeroSpace Information
7115 Standard Drive
Hanover, MD 21076-1320

National Technical Information Service
5301 Shawnee Road
Alexandria, VA 22312

Available in electric form at <http://ston.jsc.nasa.gov/collections/TRS>

Abstract

Visualization of spacecraft telemetry is an essential safety critical component to an astronaut crew's ability to interface with their spacecraft. The preferred approach would be to use a modern Graphics Processing Unit (GPU) to make the presentation of such telemetry possible. The situational experience aboard a spacecraft may mirror modern technologies similar to what might be found in the commercial aviation field. Unfortunately, the GPUs available from commercial aviation glass cockpit display systems or even the GPUs found in desktop personal computers are unsuitable for incorporation into a spacecraft that is expected to travel beyond Earth orbit (BEO). The principal factors working against direct application of commercial GPU-based systems are the radiation environment experienced by a spacecraft traveling BEO and the need for the cockpit display systems to be safety certified. Currently, the human space program has had a very limited choice of cockpit display systems from aerospace companies. The cost associated with such systems is not just the cost to procure and install hardware on a spacecraft, but also includes the lifetime display modification and update cost to generate and certify the graphical contents used to convey the spacecraft's telemetry to the crew. This report describes the software Graphics Processing Unit (sGPU) development work and the preliminary test results of a GPU architecture that may be suitable to not only meet the deep space environment and spacecraft situational awareness requirements, but also provide a means of safety certification of displays along with potentially reduced life-cycle cost associated with spacecraft displays.

Table of Contents

Abstract	i
Table of Contents	ii
List of Figures	iii
Acronyms	iv
1.0 Introduction	1
2.0 sGPU Architecture	2
2.1 Overview	2
2.2 C903-Based sGPU Implementation	4
2.2.1 Design Approach	4
2.2.2 Performance Optimization	6
2.3 C925-Based sGPU Implementation	10
2.3.1 Design Approach	10
2.3.2 Performance Optimization	10
3.0 Conclusion	13
4.0 References	13

List of Figures

Figure 1: sGPU Conceptual Architecture	3
Figure 2: C903 sGPU Block Diagram	4
Figure 3: Illustration of Display Screen Banding	9
Figure 4: C925 sGPU Implementation Diagram	11
Figure 5: Illustration of Draw-on-Demand Approach	12

Acronyms

Acronym	Definition
AD	Alpha Data
BEO	beyond Earth orbit
CPU	Central Processing Unit
DB	database
DVI	Digital Visual Interface
DDR	Double Data Rate
DMA	Direct Memory Management
ECC	Error Correcting Code
FPGA	field-programmable gate array
FPS	Frames Per Second
Gbytes	Gigabytes
GPU	Graphics Processing Unit
IGL	Ensco's implementation of the safety critical open graphics libraries
IP	Intellectual Property
LCD	Liquid Crystal Display
OpenGL	Open Graphics Library
PCI	Peripheral Component Interconnect
PCIe	Peripheral Component Interconnect Express
RTOS	real -time operating system
SBC	Single Board Computer
SDRAM	Synchronous Dynamic Random Access Memory
sGPU	software Graphics Processing Unit
VESA	Video Electronics Standards Association

1.0 Introduction

NASA is investigating deep space mission concepts and the systems and technology necessary to support those missions. Deep space missions imply large round-trip communication delays between human space vehicles and ground controllers. Thus, the burden upon crew will shift from a shared responsibility with ground controllers to more fully upon the crew because communication delays will require the crew to respond to urgent operational scenarios without immediate ground controller support. For mission-critical on-board applications, visualization of spacecraft telemetry is an essential safety critical component to an astronaut crew's ability to interface with their spacecraft. Architecture concepts under development require the use of graphics processing hardware to enable increased situational awareness of the vehicle through high-resolution graphics. Additionally, applications targeting vehicle maintenance as well as the maintenance of astronaut health, both psychological and physiological, will be necessary (e.g., Telepresence/Telemedicine and augmented reality for just-in-time training or real-time maintenance).

The preferred approach would be to use a modern Graphics Processing Unit (GPU) to make the presentation of such visualization applications possible. Unfortunately, the GPU available from commercial aviation glass cockpit display systems or even the GPU found in desktop personal computers are unsuitable for incorporation into a spacecraft that is expected to travel beyond Earth orbit (BEO). The principal factors working against direct application of commercial GPU-based systems are the radiation environment experienced by spacecraft while traveling BEO and the need for the cockpit display systems to be safety certified. Currently, the human spaceflight has had a very limited choice of cockpit display systems from aerospace companies. Those that are available are not suitable for deep space missions.

The cost associated with such systems is not just the cost to procure and install hardware on a spacecraft but also includes the lifetime display modification and update cost to generate and certify the graphical content used to convey the spacecraft's telemetry to the crew. This report describes the software Graphics Processing Unit (sGPU) development work and the preliminary test results of a GPU architecture that may be suitable to not only meet the deep space environment and spacecraft situational awareness requirements, but also provide a means of safety certification of displays along with potentially reduced life-cycle cost associated with spacecraft displays.

2.0 sGPU Architecture

2.1 Overview

The sGPU project explores the spacecraft GPU issue in a way that differs from the traditional approach of using a commercially available, dedicated silicon-based integrated circuit device to provide graphic processing functionality. Instead, the sGPU approach leverages the combination of several separate commercial efforts to produce the functionality of a GPU that has a certifiable path for use in safety critical systems while at the same time exhibiting sufficient radiation tolerance suitable for use BEO. Specifically, the sGPU project capitalizes on the availability of vendor-provided:

- Radiation-tolerant Single Board Computers (SBCs) and field-programmable gate array (FPGA) devices
- Safety critical subset of the open source open graphics language software libraries
- Commercially available real-time operating systems (RTOS) suitable for use in safety critical systems.

Figure 1 shows the sGPU's conceptual architecture. This solution provides a GPU function capable of operating in the BEO radiation environment while at the same time decoupling the proprietary relationship between the providers of the GPU hardware and the providers of displayable graphical content. Conceptually, the process of creating signals capable of driving external display devices begins with the identification of the spacecraft telemetry and the format for how the telemetry is to be display to crew members upon the output display device. The Displays and Controls Supervisory Process running on the radiation tolerant SBC combines the spacecraft telemetry source with the display format layout. The safety critical Open Graphics Library (OpenGL) renders a low-level rasterized version of the combined telemetry as described by the display format provided by the display database (DB). Once the rasterized data are available, the SBC utilizes a hardware bus interface (e.g. Figure 1 HW Bus Interface) to transfer the rendered image data to a radiation tolerant FPGA and supporting electronics. At this point, the FPGA firmware captures the rasterized format packets from the hardware bus interface and stores them into memory. The rate the SBC processes the telemetry into an atomic frame of rasterized data is variable and dependent upon the complexity of the displayable content. However, the timing characteristics necessary to drive an external display device are static. Therefore, during the final step in the process, the FPGA firmware reconciles differences between the rate the SBC renders display contents into a low-level rasterized format and the required isochronous transfer of data to the output display device via a display electronics device interface (e.g. Digital Visual Interface [DVI]).

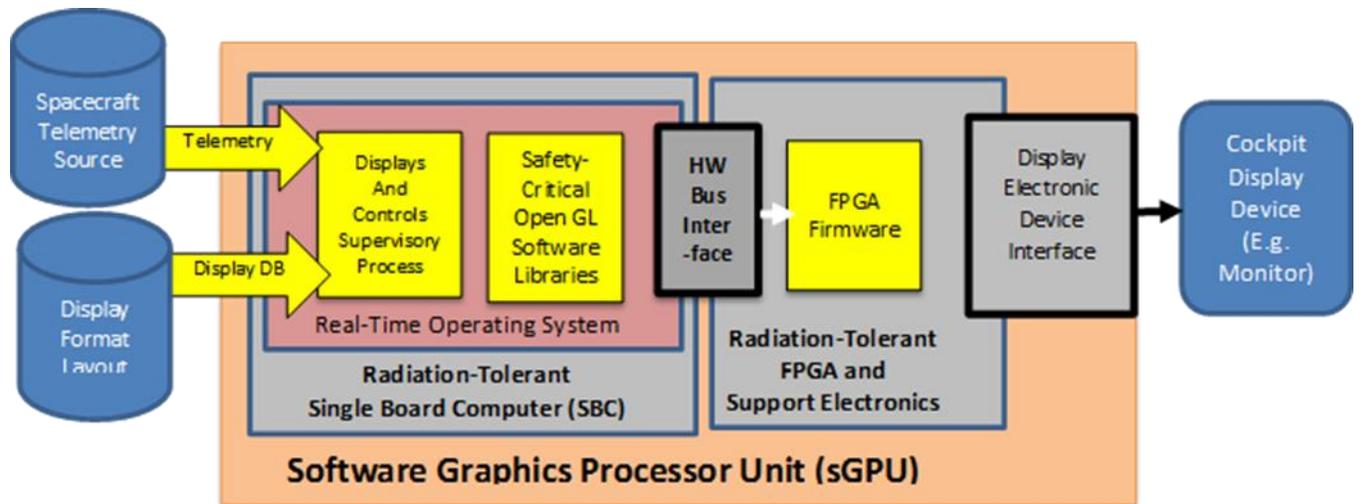


Figure 1: sGPU Conceptual Architecture

Advantages to this approach are principally twofold. First, radiation tolerant SBCs and FPGAs are commercially available. The SBC could even be common to other SBC applications used within a spacecraft. This could have implications associated with long-term missions where it might be possible to swap components to achieve mission-sparing goals. Second, methods used to describe displayable contents derive from a subset of OpenGL routines. Use of OpenGL removes the proprietary relationship between the methods used to describe a display and the GPU’s hardware. The subset of routines available for this task has been specifically reengineered to meet software safety critical criteria; namely, using an RTOS and a display software certified to the DO-178b “Software Considerations in Airborne Systems and Equipment Certification” standard. Using a familiar set of graphical library routines that are well understood to those who design graphical display contents breaks the proprietary relationship previously held by the providers of the GPU hardware (one of the large contributing factors to lifetime costs), thus increasing competitive options.

To date, two efforts at realizing the sGPU architecture concept have been made. The first implementation is based upon a C903 SBC, which is manufactured by AiTech Corporation. It was selected for initial investigation because it was available from previous unrelated projects. The second implementation is based upon the C925 SBC, also manufactured by AiTech Corporation. This implementation was selected to address issues realized during the first iteration. Both implementations interfaced to the same FPGA card, an ADM-XRC-5TZ by Alpha Data (AD) Corporation. Each of the implementations utilized different interfaces and resources that are part of the ADM-XRC-5TZ. Both of these commercial products identified were selected because they represent low-cost analogs suitable for use in a development environment instead of attempting to prove sGPU concepts using expensive spaceflight-qualified components. This approach permits the sGPU project the freedom to explore alternative hardware configurations while remaining

committed to components that have a path to flight (i.e. components suitable for use in a human spacecraft functioning in the environment BEO).

Even though the project feels confident about the approach of using lab-grade components during this development phase for the SBC and FPGA, there are other components that still need to be evaluated for the purpose of positioning the sGPU project to support BEO missions. For example, the DVI driver components that are a part of the Display Electronic Device Interface and the frame buffer memories that are a part of the supporting FPGA electronics must be examined in more detail for their radiation tolerance. At this point in the project development cycle, the goal is to prove feasibility with the major components having an anticipated path to flight and not to focus on examination of the aforementioned less-crucial frame buffer memory and DVI driver card components.

2.2 C903-Based sGPU Implementation

2.2.1 Design Approach

The first sGPU implementation was based on the C903 SBC. Figure 2 shows the block diagram of the C903-based sGPU implementation. Conceptually, the supervisory process and safety critical OpenGL libraries as shown in Figure 1 are allocated to functionality of the C903 block as shown in Figure 2. Data flow between the SBC and the radiation tolerant FPGA and supporting electronics portion of the design is represented by the 66 MHz Peripheral Component Interconnect (PCI) bus, as shown. Next, the functionality of the radiation tolerant FPGA and supporting electronics portion of the conceptual sGPU design is allocated to the AD XMC FPGA card. Finally, the functionality of the Display Electronic Device Interface is allocated to the DVI driver board.

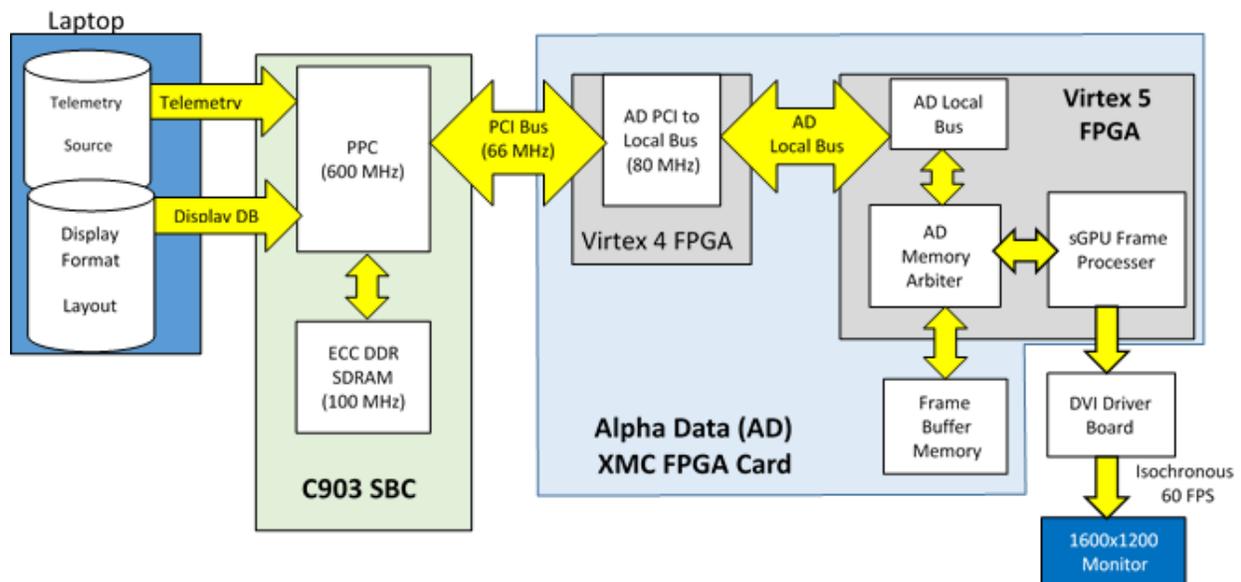


Figure 2: C903 sGPU Block Diagram

In the C903 implementation, two FPGAs are utilized. The Virtex 4 FPGA serves as a bridge between the SBC PCI bus and the Virtex 5 FPGA via an AD local bus. AD manufactured the ADM-XRC-5TZ to be used in this fashion when dealing with a PCI bus (i.e. the electrical signaling for the PCI bus is fixed and cannot be altered to route PCI bus signals directly to the Virtex 5 FPGA). In an alternative embodiment, the functionality provided by the PCI Bridge within the Virtex 4 FPGA could be assimilated by newly developed Intellectual Property (IP) components developed for the Virtex 5 FPGA, but would require the development of a custom FPGA card for this purpose. In the exploratory phases of the sGPU development, it seemed prudent to stick with commercially available components as much as possible before incurring any overhead associated with the manufacture of a custom PCB used to host the FPGA portion of the project. Because of the need to interface to the Virtex 4 Bridge function's local bus, additional IP blocks provided by AD were included in the Virtex 5 FPGA's design to the Virtex 4 provided local bus. These AD IP blocks significantly constrained the data throughput. The AD IP components together with the PCI bus itself contribute to factors limiting the C903-based design overall system performance (performance to be discussed later in the paper).

The software for the C903 implementation has been developed to operate in the VxWorks 6.7 RTOS environment and consists primarily of ENSCO, Inc. software that combined the supervisory process and associated safety critical OpenGL libraries (otherwise known as IGL) with software for initializing interfaces and launching the supervisory process. The supervisory control process is the software tasked with the gathering of spacecraft telemetry (simulated in this case) and combining it with the display format information retrieved from the display DB.

The display DB describes the way the telemetry is intended to be viewed conceptually to the end user. The supervisory process then makes calls to IGL to render the display as described in the display DB with the combined telemetry to a rasterized format. Calls to the IGL libraries rasterize the displayed telemetry as described by the Display DB and stores the resultant rasterized image into local processor memory or directly into FPGA frame buffer memory via the PCI bus.

A modern display device such as a Liquid Crystal Display (LCD) monitor requires specific timing requirements at both the pixel and frame level to properly display the images on the monitor. The specific rates the sGPU must transfer the pixel data to the output device depend upon the resolution of the display and desired frame rates anticipated for the output display device. It also must take into account vertical and horizontal blanking areas in addition to the visible regions established by the display's selected resolution. The set of supported frame rates and display resolution combinations are constrained by the manufacture of the display device and associated industry standards (Video Electronics Standards Association [VESA] Monitor Timing Standards).

Currently, the sGPU design is configured to drive a display device supporting a visible 1600x1200 resolution at a frame rate of 60 Hz. This requires the output stage of the sGPU design (i.e. the sGPU Frame Processor IP) to push 1.92e6 pixels (i.e. 1600x1200 pixels) 60 times per second in

addition to any required blanked vertical or horizontal pixels data to the DVI Driver board via an isochronously timed interface. Use of the term isochronous in this context is intended to convey the associated data transfer clock rates are strictly confined and do not wander. This is in contrast to systems where the clocked data rates wander considerably as long as the overall average aggregate rate remains constant. This strict isochronous timing for the output is necessary for the sGPU to conform to accepted standards associated with commercial output display devices (e.g. DVI input to a commercially available LCD monitor).

Rasterization of displayable content by the C903 SBC's Central Processing Unit (CPU) is asynchronous to the display frame rate. The use of a frame buffer is necessary as the render rate (i.e. the rate a frame of completed display contents is converted to a rasterized format) of the SBC is variable dependent upon complexity of the display's visual content and task loading of the SBC. Essentially, the SBC by itself cannot conform to the strict isochronous timing requirements necessary to directly drive a display output device. Reconciliation of the C903's asynchronous rasterization frame rate to strict isochronous display rate timing required to interface to a VESA compliant output display device is accomplished by the sGPU frame processor IP block (see the Virtex 5 component in Figure 2). The memory arbitration block and sGPU frame processor interact to convert the asynchronously rendered image data stored in the frame buffer memory into a VESA compliant isochronous data stream suitable for driving an LCD monitor. The last step in the process is to pass this VESA compliant stream of data over a parallel interface to a DVI driver board where it is converted into multiple high-speed serial data paths conforming to industry DVI interface specifications. Thus, the sGPU is capable of driving a display output device via an industry standard display interface.

2.2.2 Performance Optimization

For the first performance optimization approach, the C903's CPU was directed to store each of the 1600x1200 pixel frames directly into the FPGA's frame buffer, one pixel at a time via individual transactions over the PCI bus. Utilizing this direct approach allows for checkout of basic system connectivity and health of the configuration (including software). Performance testing of the C903-based sGPU implementation for this first approach resulted in a disappointing rendered frame rate at less than 1 frames/second (FPS). (Note: the rendering frame rate is the rate at which the sGPU can process the displayable content into a complete rasterized representation and is not the same thing as the isochronous display update rate, which remains at a constant 60 Hz.) Even though the rendered frame rate was very low, the 60 Hz isochronous display output frame rate provided by the sGPU frame processor block functioned as intended and the displayable contents was viewable on an attached LCD monitor.

Use of the PCI bus to transfer individual pixels in a PCI bus transaction is highly inefficient where a bus transaction is a PCI frame. As a general rule, the efficiency of the PCI bus transaction is proportional to the payload size (i.e. the number of pixels) contained within the bus transaction. Essentially, this initial exploration used the least-efficient way to transfer data over the PCI bus

and contributed to the low render rate. Wherein the initial approach did not yield an acceptable frame rate, it did prove the system is capable of displaying telemetry combined into a correctly comprised display as dictated by the display DB (Figure 2) and stressed the importance of using the PCI bus in more effective transmission modes. Lessons learned were applied to a second approach for performance optimization where the CPU and PCI bus were used more efficiently to increase the effective rendered frame rate.

The second approach to improve performance focused on leveraging the SBC processing strengths. In this approach, the C903 rendered to the C903 memory local to the CPU (i.e. the 100 MHz Error Correcting Code Double Data Rate Synchronous Dynamic Random Access Memory) and then later utilized Direct Memory Management (DMA) transfers to move the rasterized image data from the memory local to the CPU to the FPGA memory (i.e. the frame buffer memory). The C903 SBC CPU has the best possible rendering performance when manipulating its local memory as opposed to forcing the SBC CPU to directly render to the FPGA frame buffer memory locations over the PCI bus. The burden or overhead of PCI transactions across the bus makes this direct manipulation of memory by the CPU on the FPGA side of the design inefficient.

The C903 SBC's DMA engine is superior at moving data over the PCI bus when compared to using the CPU to move data in memory for two reasons. First, the DMA engine is specially constructed to make efficient access to the memory local to the CPU without the need to fetch instructions (even from CPU cache) to carry the move operations. Second, the DMA engine makes it possible to transfer multiple pixels in a block of pixel data for each PCI bus transaction instead of just one pixel per transaction as in the case when the CPU is used to transfer data across the PCI bus. The overhead in a PCI transaction is at its worst when the payload (i.e. the number of pixels in the transaction) is one.

Performance results regarding the second approach yielded some improvement, but revealed two new limiting factors. Even when using the DMA-based pixel block transfer approach, PCI bus performance still dominated the C903-based system's behavior. This "Bus Bound" condition effectively caps the system performance rendering rate of the sGPU in the C903 implementation to approximately 2 frames per second. It is important to note, the PCI bus throughput issue may not only lay with the inherent PCI bus capabilities of the C903, but also with the speed and overall data architecture of the AD-provided IP components previously mention that were used in the Virtex 4 and 5. These IP components were used as provided by AD and not constructed specifically for the sGPU project. The inefficiency is exaggerated by the fact that the target resolution of the sGPU is for a 1600x1200 pixel display (e.g. 1 frame is 1.92E6 pixels).

This resolution may seem excessive; however it is intended that in the case of the sGPU design, the display device be rotated 90 degrees and divided into an upper and lower display area (i.e. two 800x1200 pixel areas). The two display areas function as two separate displayable content areas displayed on one physical display device (i.e. the 1200x1600 monitor device as shown in Figure

2). Lowering the resolution of the display would help the situation by lowering the amount of data that would be manipulated, but would compromise the resolution in each display area.

During the evaluation of the second approach, another bottleneck related to the CPU's ability to render the image became apparent. How the CPU uses the buffers during in the OpenGL methodologies to render the display contents into a rasterized format has a significant effect on performance. When rendering displayable content into a rasterized format it first must clear a number of buffers equivalent in resolution to that of the final output resolution of the output display device (1600x1200) and then change a fraction of the pixels in those buffers to values representing the displayable content's visible features. The amount of time required by the CPU to represent the displayable content's visible features varies proportionally to the complexity of those displayable features. The burden for clearing the display buffers remains constant when following the typical OpenGL processing paradigm. The CPU must typically clear all 1.92E6 pixel frame locations once each time the displayable content is converted to a rasterized format. Conventionally, this process of clearing the entire frame buffer and redrawing the displayable content in a rasterized format is repeated for each frame even when there is no change to the underlying displayable content's features. The nature of displayable content for cockpit displays is such that typically only small areas of the displayable contents change (such as numbers in a field) and a sizable portion of the display remains static. Thus, for displayable contents reflective of what might be used for cockpit monitoring of spacecraft telemetry using a conventional approach, the CPU performing clearing and redrawing of static portions of the display impacts overall frame rate and is wasteful.

The third attempt at performance optimization focused on ways to mitigate the identified PCI bus throughput limitations (bus bound) while simultaneously attempting to mitigate the impact of the buffer clear and redraw cycles previously mentioned. During this attempt, it was decided to take advantage of some previous unutilized features of the IGL product to vary the rate of clear and redraw cycles for different portions of the frame buffer local to the CPU. The ENSCO IGL product permits the construction regions of displayable content that vary from other regions that are part of the same displayable screen content. For example, if a control such as a textbox changes the value it is to display once every 30 frames, then the associated buffer locations associated with that textbox only need to be cleared and redrawn in a rasterized format once every 30 frames. This is in contrast to the typical method of clearing and redrawing the textbox for each of the 30 frames. This potentially represents a significant savings to the number of CPU cycles required in the display generation/rasterization process. For the remainder of this paper, this technique will be known as "multi-rate" or "draw on demand." The downside to this draw on demand approach is that it requires greater coordination with the creation of the displayable content to implement and only works really well when the nature of the displayable content varies in small ways from frame to frame (this is typical to the stylized nature of telemetry-based displays associated with human-rated spacecraft). Also with this approach, it is possible to track which regions of the displayable

content have changed and use this knowledge to reduce the amount of data pushed through the dominant PCI bottle neck, thus improving overall system performance.

To reduce the amount of data to be pushed through the PCI bottle neck (the most dominate of the two performance bottle necks), it was decided to exploit one of the idiosyncrasies identified in the addressing of the Virtex 5's frame buffer memory that came about because of the inclusion of the AD provided IP for arbitrating access to the FPGA frame buffer memory. That is, the FPGA frame buffer memory was paged and could not be linearly addressed, effectively breaking the frame buffer memory and associated displayable screen area into a screen band comprised of 12 vertical stripes or bars (see Figure 3 for an illustration). Using the draw on demand approach made it possible to identify which equivalent bars in the CPU's local memory contained changes and then to only move the identified frame buffer bars (those with changes) over the PCI bus to the FGPA frame buffer memory. Since this greatly reduced the amount of information pushed over the PCI bus, the system realized a performance bump to approximately 7 frames per second – a seven-fold increase from the initial experience with the C903-based implementation.

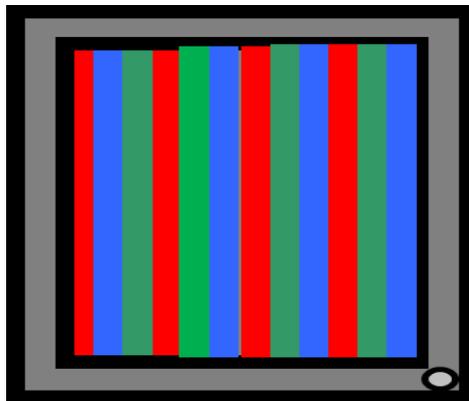


Figure 3: Illustration of Display Screen Banding

The next step to improve performance was to only push the area of the displayed telemetry that were identified to have changed with the draw on demand method instead of the bars. This would further create improvements to the C903-based system's performance. However, the complexity associated with the segmentation of telemetry regions crossed two or more bands made this solution unattractive from a software development perspective. The added complexity in software design really would not address the root cause issue created by the dominant bus throughput bottleneck. Thus, the project's next steps in sGPU development was to eliminate the bus bound bottleneck created by use of the PCI bus. Therefore, work on the C903 stopped in pursuit of an SBC that utilized a higher-speed PCI bus.

2.3 C925-Based sGPU Implementation

2.3.1 Design Approach

Through investigation of other bus options, it was found that a newer version (Version 1.1) of the PCI bus standard called the PCI Express (PCIe) would provide eight lanes of PCIe with a theoretical bidirectional transfer rate of 2 Gbytes/sec that could potentially provide around 40 FPS rates to at least four displays. The PCIe bus is a modern gigabit rated bus and exhibits orders of magnitude increase in performance over what was experienced with the C903's 25-year-old PCI bus technology. The C925 from AiTech was selected that contained a PCIe bus as well as having a path to a flight processor. The C925 theoretically can accommodate eight lanes of PCIe; however, in practice, it was found that due to design defects the C925 could only practically realize four lanes of PCIe. However, four lanes of PCIe were more than enough to move ahead with significant performance improvements in the sGPU project.

The C925 contained the same I/O connectors as the C903, which minimized hardware changes to the chassis, DVI driver board, AD card, and the knowledge gained using the AD FPGA card. Figure 4 shows the block diagram of the C925-based sGPU. Like the C903, the C925 has a path to spaceflight since the CPU is compatible with the radiation tolerant/hardened 750 Performance Optimization with Enhanced Reduced Instruction Set Computing – Performance Computing processor.

2.3.2 Performance Optimization

The C925 contained the same RTOS and IGL software as the C903 and leveraged the development effort based on the C903 with several significant exceptions. Namely, the implementation of the C925-based sGPU eliminated the PCI bus and replaced it with the four-lane PCIe bus. Replacing the PCI bus with a PCIe bus also made it possible to eliminate the AD-provided FPGA IP previously used in the Virtex 4 and Virtex 5 FPGAs that was a part of C903's PCI-based design implementation. This permitted a direct connection between the C925 SBC's CPU via multiple lanes of the PCIe bus to the Virtex 5 FPGA and completely eliminates the use of the Virtex 4 FPGA required in the C903-based implementation. The newly created Virtex 5 FPGA IP associated with the wide band memory arbiter significantly increased the memory bandwidth to the FPGA frame buffer memory by increasing the data path width to the frame buffer memories from 32 bits (from the C903-based design) to 128 bits in width.

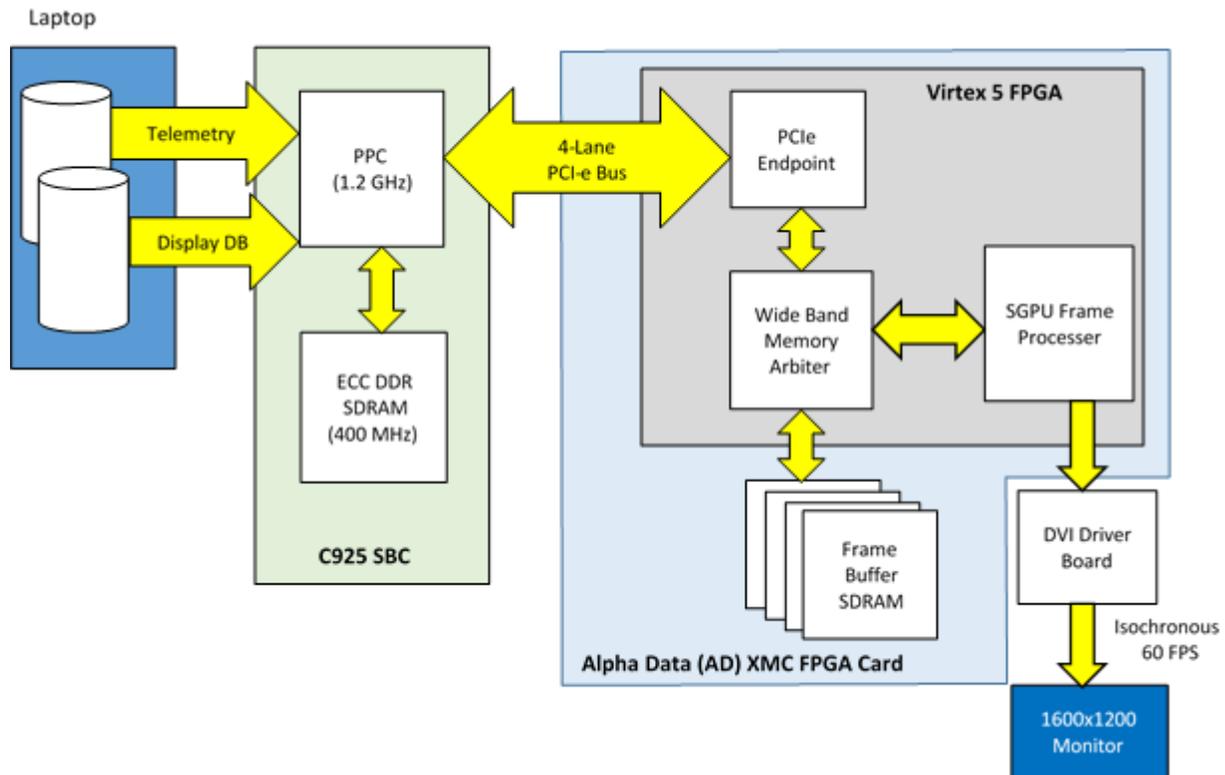


Figure 4: C925 sGPU Implementation Diagram

Another deviation from the C903 method was using a draw on demand approach that provided a simple way of updating the displays without the complexity of the C903 method of updating the entire display when only portions of the display data had changed. Figure 5 notionally shows the draw on demand approach where one-color band represents a full frame/screen of data. Utilizing the draw on demand concept and the ability to push complete frames (frame resolution of 1600x1200 pixels) of rendered memory at much higher rates (compared to the C903) to the FPGA frame buffer memory, the C925 sGPU implementation was able to achieve an overall system rendering performance of approximately 27 to 29 frames per second (depending on the complexity of the displayable content). In comparison, the C903 implementation was at best able to achieve a rendered frame rate of 7 frames per second.

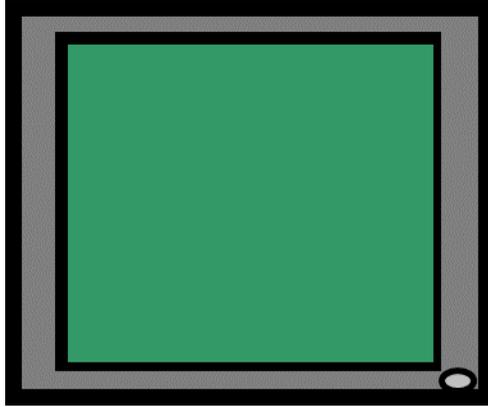


Figure 5: Illustration of Draw-on-Demand Approach

Preliminary timing analysis shows that with four PCIe bus lanes, it takes about 34 milliseconds to render and push/transmit rendered frame data to the frame buffer synchronous data random access memory via the Wide-band Memory Arbiter. Scope measurements show that the PCIe bus is only 40% utilized during direct memory access transmissions. At this point, it is unclear why the bus is underutilized. Once this issue is resolved, increased FPS performance well over 30 FPS is expected.

3.0 Conclusion

Preliminary results indicate the sGPU is capable of rendering the stylized graphical displays typically used for safety critical displays. The latest benchmarks for the C925 (~30 FPS) show a better than 3X increase in FPS performance than the C903 (6-7 FPS) sGPU version. The team plans to continue to explore means to effectively increase the transfer rate of rendered frames between the C925 SBC and the FPGA. Planned improvements to the sGPU suggest the feasibility of reaching display render rates beyond 40 FPS. It is anticipated through use of software optimization techniques the sGPU will reach rates acceptable for the display of safety critical displays while at the same time exhibiting radiation tolerant behavior suitable for use BEO. Supporting components that make up the sGPU, such as the DVI board, have yet to be vetted for their radiation tolerance and represent an area of future work. The team believes it is possible to achieve radiation tolerance for this portion of the system based on prior examples of FPGA development utilized in projects that have already been used BEO. However, exploration of this aspect of the system definitely bears more discussion and practical testing to show the feasibility of using a radiation tolerant FPGA along with other supporting display electronics.

4.0 References

Salazar, G. and Glen Steele. "Commercial Off-The-Shelf Graphics Processing Board Radiation Test Evaluation Report." NASA Technical Memorandum (NASA/TM-2014-217395)

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE August 2015	3. REPORT TYPE AND DATES COVERED Technical Memorandum		
4. TITLE AND SUBTITLE Software Graphics Processing Unit (sGPU) Development and Evaluation Report			5. FUNDING NUMBERS	
6. AUTHOR(S) George A. Salazar; Glen. F. Steele				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Lyndon B. Johnson Space Center Houston, Texas 77058			8. PERFORMING ORGANIZATION REPORT NUMBERS S-1200	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) National Aeronautics and Space Administration Washington, DC 20546-0001			10. SPONSORING/MONITORING AGENCY REPORT NUMBER TM-2015-218585	
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Unclassified/Unlimited Available from the NASA Center for AeroSpace Information (CASI) 7115 Standard Hanover, MD 21076-1320 Category: 32			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) Visualization of spacecraft telemetry is an essential safety critical component to an astronaut crew's ability to interface with their spacecraft. The preferred approach would be to use a modern Graphics Processing Unit (GPU) to make the presentation of such telemetry possible. The situational experience aboard a spacecraft may mirror modern technologies similar to what might be found in the commercial aviation field. Unfortunately, the GPUs available from commercial aviation glass cockpit display systems or even the GPUs found in desktop personal computers are unsuitable for incorporation into a spacecraft that is expected to travel beyond Earth orbit (BEO). The principal factors working against direct application of commercial GPU-based systems are the radiation environment experienced by a spacecraft traveling BEO and the need for the cockpit display systems to be safety certified.. This report describes the software Graphics Processing Unit (sGPU) development work and the preliminary test results of a GPU architecture that may be suitable to not only meet the deep space environment and spacecraft situational awareness requirements, but also provide a means of safety certification of displays along with potentially reduced life-cycle cost associated with spacecraft displays.				
14. SUBJECT TERMS display devices; radiation; long duration space flight; spacecraft environments; deep space;			15. NUMBER OF PAGES 24	16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT Unlimited	
