

# Scaleable Intelligent Video Server System

<i>Title</i>	Input Acquisition System SW Report
<i>Revision</i>	B
<i>Deliverable #</i>	D12.1 complete input acquisition system
<i>Author</i>	Frances Dose, Max Schlee, Wolfgang Huther (all Grass Valley)
<i>Company</i>	Grass Valley Germany GmbH
<i>Date</i>	16/4/2006
<i>Filename</i>	D12.1 Input Acquisition System SW Report B.doc
<i>Dissemination<sup>†</sup></i>	CO

REVISION	DATE	DESCRIPTION
A	16/2/2006	Created by WPH
B	16/4/2006	24 month report included

<sup>†</sup> **CO** = Confidential (only for members of the consortium + EC); **RE** = Restricted to a stated circulation list (+ EC)  
 [replace this footnote with the list]; **PP** = Restricted to other FP6 participants (+ EC); **PU** = Public

## TABLE OF CONTENTS

<b>1 INTRODUCTION</b> .....	<b>4</b>
<b>2 WORK DONE</b> .....	<b>4</b>
2.1 STATUS AT THE BEGINNING OF 2005 .....	4
2.2 SIVSS INGEST INTERFACE INVESTIGATION .....	4
2.2.1 GSN (Gigabit System Network formerly known as Hippi 6400) .....	5
2.2.2 Fibre Channel.....	6
2.2.3 10 Gigabit Ethernet.....	6
2.2.4 InfiniBand.....	6
2.3 SELECTION OF WORKSTATION/SERVER BUS SYSTEM .....	7
2.4 EVALUATION OF ACQUISITION SYSTEM PERFORMANCE .....	8
2.4.1 Single disks tests.....	9
2.4.2 Multiple disks tests .....	9
2.5 INTEGRATION WORK AND IBC DEMONSTRATION .....	11
2.5.1 Test with GFS File System.....	11
2.5.2 IBC Show floor performance.....	12
<b>3 DESCRIPTION OF INPUT ACQUISITION SYSTEM</b> .....	<b>13</b>
3.1 HARDWARE ARCHITECTURE .....	13
3.2 SOFTWARE ARCHITECTURE.....	14
3.2.1 Schema .....	14
3.2.2 Modules .....	14
3.2.3 Control and Data Flow .....	16
3.2.4 Interfaces to other modules .....	17
3.3 DEVELOPMENT ENVIRONMENT .....	18
3.4 STATISTICS .....	18
3.4.1 Lines of Code.....	18
3.4.2 Developed Libraries .....	18
3.4.3 Used our libraries .....	18
3.4.4 Used 3 <sup>rd</sup> party Libraries .....	18

## **List of Figures**

Figure 1: Layers for IB based Scanner to Ingest Application data transfer .....	7
Figure 2: Single disk transfer rate at different I/O block sizes .....	9
Figure 3: Read performance at different block-sizes of a disk array (single thread) .....	10
Figure 4: Read performance at different threads (2,4,8,16) (block size 8MB) .....	10
Figure 5: IBC Demonstrator Software & Interfaces .....	11
Figure 6: IBC Demonstrator Hardware .....	13
Figure 7: Video I/O Hardware Architecture .....	13
Figure 8: Video I/O Software Organization .....	14
Figure 9: Video Ingest Operation .....	16
Figure 10: Video Playback Operation .....	16

## **List of Tables**

<i>Table 1: Bandwidth Requirements for Production Applications</i> .....	4
<i>Table 2: Interface data rates</i> .....	5
<i>Table 3: Bus Data rates</i> .....	7
<i>Table 4: Achievable Data Rates on PCIe</i> .....	8
<i>Table 5: Workstation platforms used in 2005</i> .....	8
<i>Table 6: GFS file system read performance</i> .....	12

## 1 INTRODUCTION

This document describes

- (a) the work done for deliverable 12.1: complete Input Acquisition system
- (b) the description of the Input Acquisition system
- (c) the development environment
- (d) the statistics

## 2 WORK DONE

### 2.1 Status at the beginning of 2005

The system design for a scalable ingest and playout system has been finished. Several of the shelf workstations were tested and evaluated and one was selected as the basic processing module of the SIVSS acquisition system. The basic software of the ingest station has been developed and the integration into the workstation has started.

Being in month 14 of a 24 month project it became clear that we would have to focus our efforts on a single application in order to show demonstrable results. Moreover, due to the advent of HDTV in Europe and the advances in Digital Cinema specifications the demand for high end, high speed postproduction tools increased. We therefore decided to concentrate our investigation in this domain.

# active samples	# of active lines	# of bits	# of streams	frame rate	Net data rate max.	Format	Application
1920	1080	10	2	24-30	1.244Gb/s	YCrCb 422	274M
1920	1080	10	2	50-60	2.488Gb/s	YCrCb 422	372M
1920	1080	12	3	1-30	2.239Gb/s	RGB 444	Camera
2048	1080	12	3	48	3.822Gb/s	XYZ 444	Display
2048	1080	12	3	1-60	4.778Gb/s	RGB 444	Camera
2048	1756	10	3	24	2.761Gb/s	RGB 444	filmscanner
4096	2160	12	3	24	7.644Gb/s	XYZ 444	Display
4096	2160	12	3	1-30	9.555Gb/s	RGB 444	Camera
...							
4096	2160	12	3	1-60	19.110Gb/s	RGB 444	Camera
4096	3512	16	3	10	6.905Gb/s	RGB 444	filmscanner

Table 1: Bandwidth Requirements for Production Applications

### 2.2 SIVSS Ingest Interface Investigation

With the advent of HDTV in Europe, high resolution, high speed digitisation/scanning is a foundation for future HDTV programs. For the SIVSS postproduction ingest part the most important Video/Audio sources are

- ◆ 2K real-time filmscanner (2048 x 1756)
- ◆ 4K real-fast filmscanner (4096 x 3512)
- ◆ Digital HDTV Video camera (1920 x 1080, up to 150 frames/sec)
- ◆ Digital film camera (2048 x 1080)

Therefore the requirement for SIVSS ingest data rate is defined as follows:

Each frame has a resolution of 2K coded in the three components R, G and B where each component is coded with at least 10bits. The Film Frames are transmitted with a rate of 24 frames per second. The required bandwidth is calculated according to

$$BW = \text{pixel\_per\_line} * \text{lines\_per\_image} * (\text{approx. } 3 * 10\text{bit}) * \text{frame\_rate}$$

$$BW = 2048 * 1756 * 4\text{Byte} * 24 \text{ frames/sec}$$

$$BW = 346 \text{ MByte/sec}$$

(Because of computer-systems are byte oriented we have chosen the nearest Byte number to 3x10bit which is 4Bytes).

For the SIVSS ingest system only interfaces with a data rate > 400MByte/Sec were investigated. The interfaces should be commercially available, therefore more or less proprietary interfaces like Quadrics QNet II were not considered here.

Table 1 shows the nominal data rate of interface candidates.

Technology	Nominal Bandwidth
Fibre Channel 4G FC	400 MByte/s
GSN	800 MByte/s
10 Gigabit Ethernet	1 000 MByte/s
InfiniBand 4x	1 000 MByte/s
InfiniBand 8x	2 000 MByte/s
InfiniBand 12x	3 000 MByte/s

Table 2: Interface data rates

### 2.2.1 GSN (Gigabit System Network formerly known as Hippi 6400)

GSN has been used in research labs and government sites for several years. It never has been accepted by a broader community and therefore production quantities are very low with very high costs (e.g. 15k\$ per NIC). Real implementations have proofed that reliable transfers are possible up to 600MByte/sec. Physical transport medium is copper or fibre. This technology is currently used as an interface for the filmscanner and allows reliable data transfer up to 400MB/sec. But recently the hardware support of this technology has been stopped by the manufactures, so for new development this kind of interface is not appropriate anymore.

### 2.2.2 Fibre Channel

Fibre Channel has been used in disc arrays for several years, therefore this interface was evaluated during integration of SIVSS storage devices. As SCSI- devices, the Fibre Channel-drives and RAID-arrays allow to monitor and tune a set of functionalities, like ReadCaching, WriteCaching, CommandTagQueuing etc, this is essential for tuning the throughput of a system for high-performance real-time-playback and recording. With other technologies like ATA or SATA this functionality is not available. Compared to other technologies the protocol overhead of a Fibre Channel transmission reduces the data rates significantly. However the flexibility with regards to number of devices of up to 127 per Fibre Channel and the possibility to extend the optical cable length to hundreds of meters rates the Fibre Channel network superior to any other technology.

For larger installations with many high performance clients working on the same set of media without the need of copying the Data from one machine to the other Fibre Channel is the only technology that supports the required Storage Area Network (SAN) technology. From the viewpoint of a local or SAN file system there is no difference between a single Fibre Channel disc in a “Just a Bunch of Disc” (JBOD) array and a RAID controller in a RAID arrays. Thus the internal array technology is of no relevance for the file system. The physical transport medium is copper or fibre.

### 2.2.3 10 Gigabit Ethernet

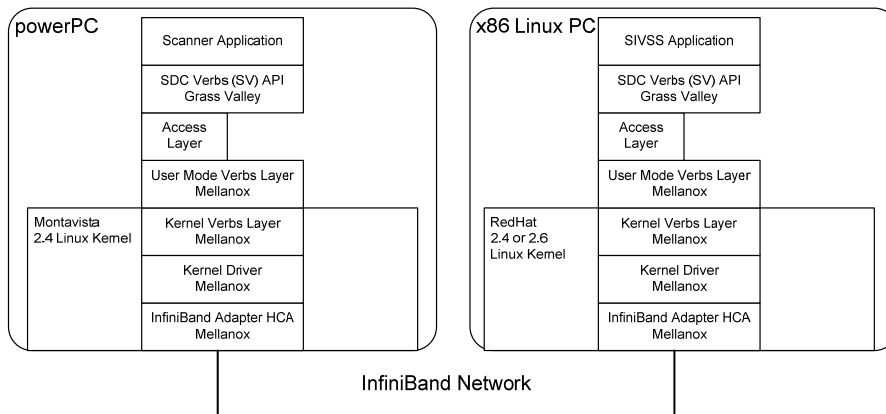
This network is the natural upgrade path of existing computer networks and will be widespread available. This is a logical candidate for every high speed network. The nominal high data rate can only be achieved with unreliable connections. There is a dramatic decrease in data rate if a reliable connection is required. Reliable connections normally require a significant protocol overhead (TCP/IP stack). New interface boards became available in 2005 which implement the protocol stack in hardware as an integral part of the host channel adapter. This technology is known as **TCP Offload Engine (TOE)**. But TOEs require a modification within the controlling operating system. This technology is not widely supported by currently available operating systems. Physical transport medium is fibre but cheaper implementations using copper will become available in the future.

### 2.2.4 InfiniBand

InfiniBand is a technology which became popular in building up computer clusters and will probably replace proprietary technologies like Quadrics QSNNet and Myrinet. InfiniBand is also scalable to some amount. Reliable connections can be handled without large overhead. Physical transport medium is copper but optical enabled host channel adapters and optical media converters are available as well.

Several low level and high level protocols are already established. **Remote Direct Memory Access RDMA** is an integral part of the InfiniBand technology which gives some advantages over 10Gig Ethernet. InfiniBand was tested with a low level protocol (VERBS layer) and transfer rates of 500 MBytes/sec (server to server) could be achieved.

It was decided to select InfiniBand 4x as future platform for any scanner ingest activities. It combines low cost, reliable connection together with a minimized CPU overhead.



**Figure 1: Layers for IB based Scanner to Ingest Application data transfer**

### 2.3 Selection of Workstation/Server Bus System

Host channel adapters are offered with different PCI Bus technologies. Table 2 shows the nominal data rates of the different PCI implementations.

Technology	Nominal Bandwidth
PCI – 64bit, 66 MHz	512Mbyte/Sec
PCI –X 66	512Mbyte/Sec
PCI –X 100	768Mbyte/Sec
PCI –X 133	1 GByte/s
PCIe 4 x	1 GByte/s
PCIe 8 x	2 GByte/s
PCIe 16x	4 GByte/s

Table 3: Bus Data rates

The tests during year 2004 were made with PCI-X based systems. In 2005 PCIexpress (PCIe) emerged as a new technology which was tested as well. The tests showed that the required data rates could be achieved with PCI-X and PCIe 4x systems. The nominal data rate of the PCIe bus is decayed by currently available chipsets for PC motherboards. The PCIe standard allows transactions with a burst-length up to 4KByte. The tests of the SIVSS ingest system were made with a HP8200 workstation which allows only transactions of 64 Byte. With the 4x implementation two ingest data streams can be transferred in parallel. It is assumed that the transaction size of future motherboards will go up to 256 in the next 2 years. Table 3 shows the data rates which can be achieved with different PCIexpress implementations.

Transaction Payload Size	PCIe 4 x	PCIe 8 x	PCIe 16 x
64 Bytes	760 MByte/s	1446 MByte/s	2895 MByte/s
128 Bytes	862 MByte/s	1643 MByte/s	3286 MByte/s
256 Bytes	926 MByte/s	1762 MByte/s	3525 MByte/s

Table 4: Achievable Data Rates on PCIe

## 2.4 Evaluation of Acquisition System performance

In 2004 we proofed that a system based on the xw8200 was able to deliver the required internal bandwidth for real-time 2k streaming. The goal of the new evaluation tests was to prove that the performance of the components of a real system, including clustered file system and disk array, meets the requirements. For this test the new hp xw9300 workstation was used, which will be the successor of the hp xw8200. The xw9300 was the first 64 bit CPU we used in the SIVSS project.

	Xw8000	Xw8200	Xw9300 (64Bit)
CPU	2 x XEON 3.0 GHz	2 x XEON 3.2 GHz	2 x Opteron 280 DP 2.6 GHz
RAM	2 GByte	2 GByte	4 GByte
GRAPHICS	nVidia FX3000	nVidia FX 3400	nVidia FX 3400
I/O	2 x PCI-X 100 MHz 1 x PCI-X 133 MHz 2 x PCI	1 x PCI-Express 4 x 2 x PCI-X 100 MHz 1 x PCI-X 133 MHz 2 x PCI	2 x PCI-Express 16x 1 x PCI-X 100 MHz 2x PCI-X 133 MHz 1 x PCI
Networking	1 x Gig-Ethernet	1 x Gig-Ethernet	1 x Gig-Ethernet
OS		RedHat EL 3.0 WSS Update 6 x86_64, kernel 2.4.21-37.ELsmp	
File system		ADIC Cvfs 2.6.3 Build 30 (Beta version)	
Storage		Xyratex JBOD RS-1600, 6 Disks Seagate 36GByte, two loops with 8 disks each.	
Controller:		QLogic qla2342: double 2gbit, connected with two optical cables.	

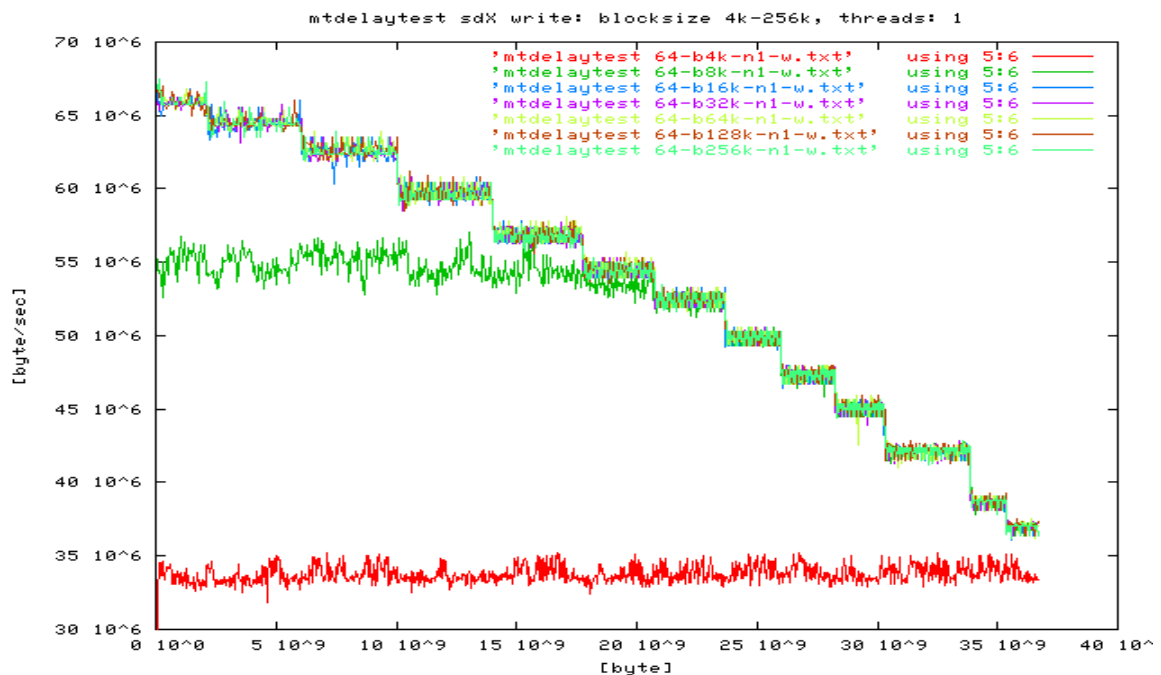
Table 5: Workstation platforms used in 2005

The rightmost column in the table above shows the system configuration we used. After some minor problems with Red Hat Enterprise 3 (due to insufficient support of the new AMD-ET64-Architecture) have been fixed and the latest "Engineering-build" of ADIC's Cvfs file system has been installed we were able to run first performance tests, but never were able to achieve more than

14 fps in ingest and 15 fps (< 200Mbyte/sec) in play mode. Using Linux kernel 2.6 improved performance by approx. 10% but was still not sufficient.

### 2.4.1 Single disks tests

First the performance of each single disk with different I/O sizes was examined by using our own test tool (mtdelaytest). So all disk were read from beginning to the end. The write performance is shown in fig.1. Here the different zones on the disks can be seen. It is important to notice that the disk's transfer rate depends on the location of the disc, so at the end the achievable data rate is about half of the maximum at the beginning! As a result of this evaluation it can be stated that I/O-sizes of 16kByte or bigger must be used for these disks. The values of the read process are similar.



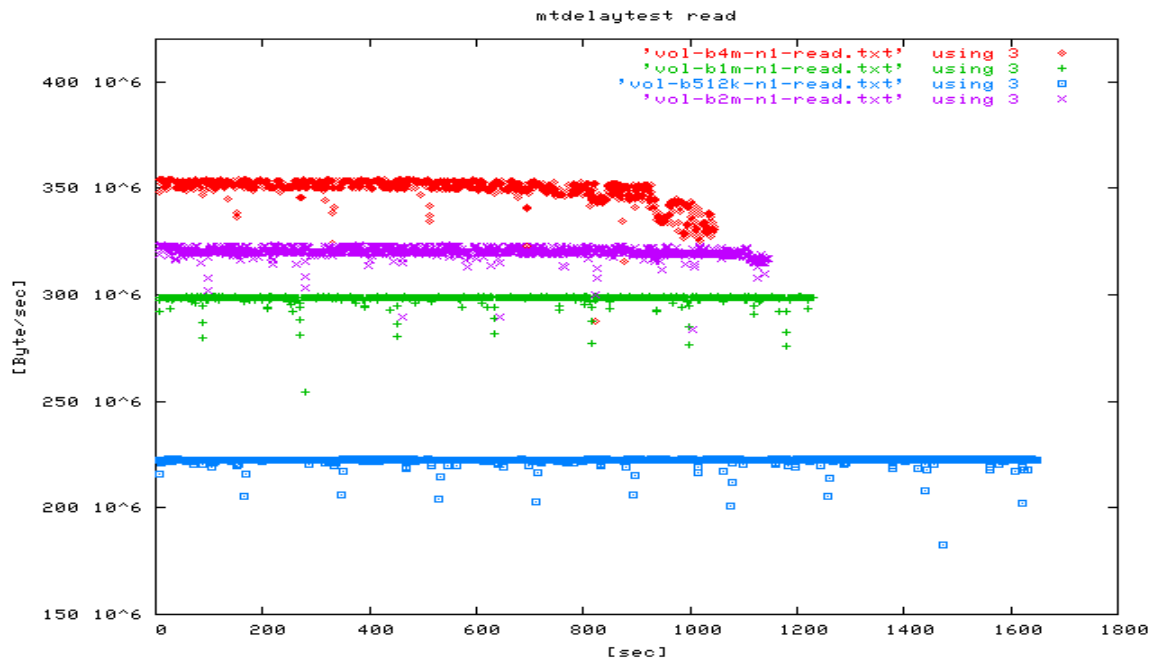
**Figure 2: Single disk transfer rate at different I/O block sizes**

### 2.4.2 Multiple disks tests

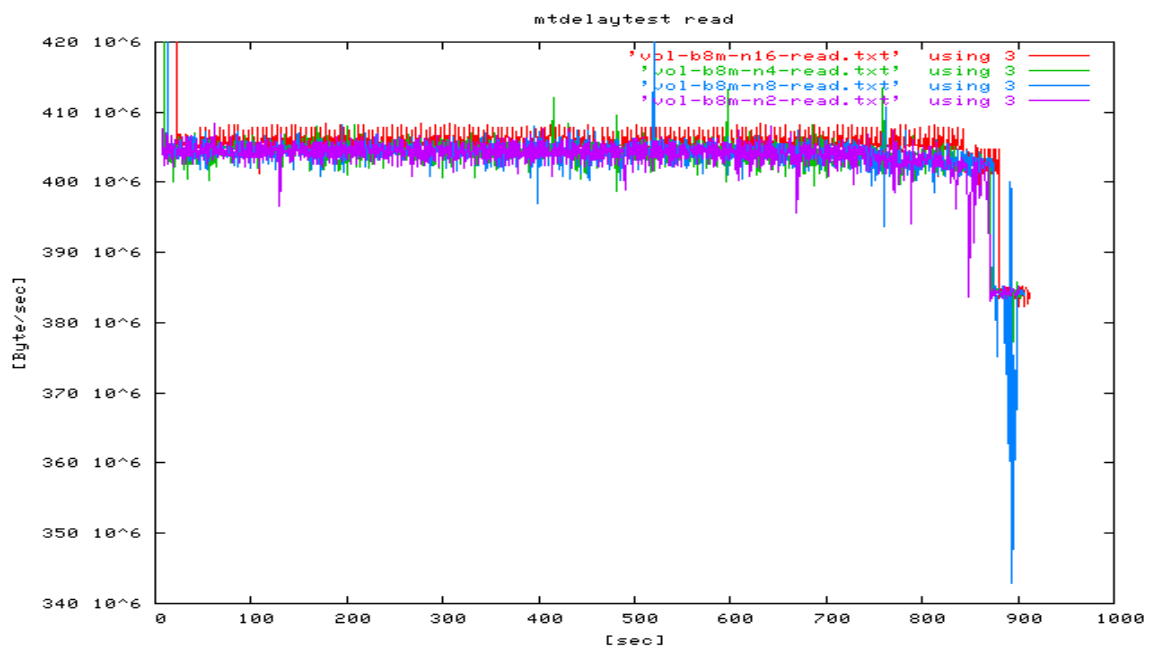
The next tests were done with a disk array of 10 disks with single thread. Fig.2 shows the read performance at different block-sizes. To achieve the best results block-sizes of at least 4MByte should be used. In fig.3 the number of threads is increased. It can be seen that by using two threads the data rate is increased by about 15%. More than two threads have influence on the performance of the disk system.

Also write performance tests were done. Here the achievable data rates were lower and not consistent. Only 14 frames/sec could be achieved, with a frame rate of 15 frames/sec or more it was not possible to write the data onto the disks.

Further tests showed that the poor performance of some disks spoiled the overall system performance. After we exchanged the defective disks we were able to achieve 2k real-time ingest and playout i.e. bandwidth larger than 310 MByte/sec.



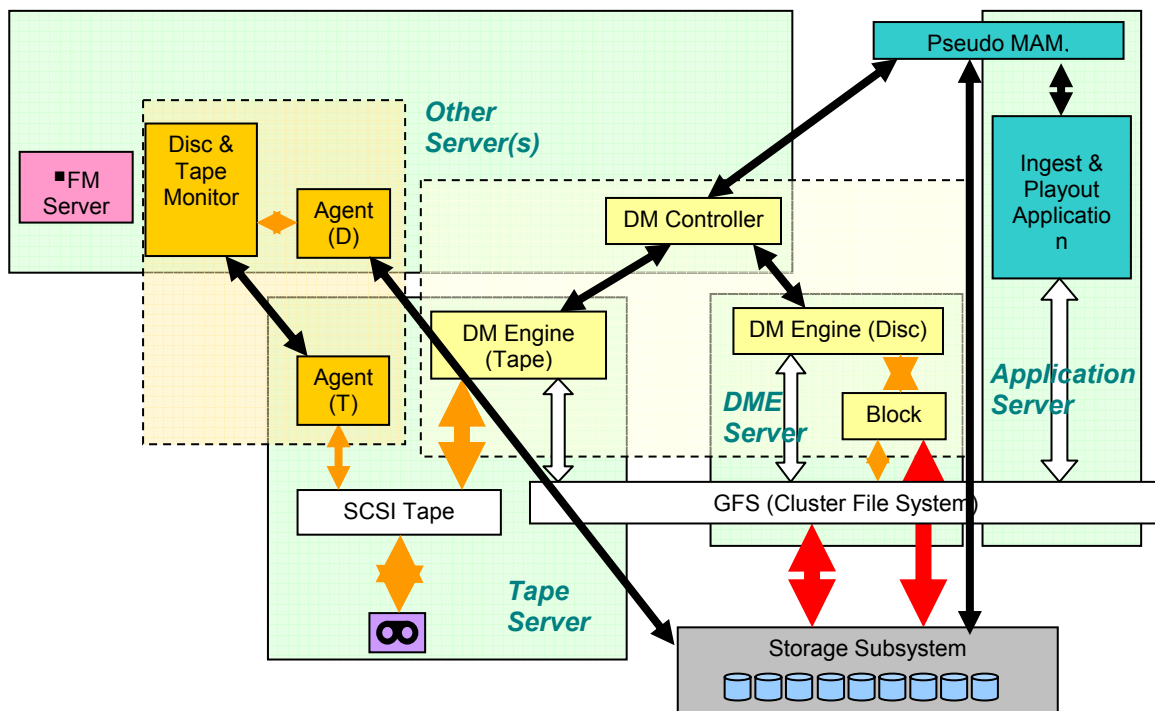
**Figure 3: Read performance at different block-sizes of a disk array (single thread)**



**Figure 4: Read performance at different threads (2,4,8,16) (block size 8MB)**

## 2.5 Integration work and IBC demonstration

During the integration phase in April 2005 it became obvious that the support of the Thomson system software team would not be available in time and we would not be able to develop the MAM interface software for the IBC demonstration. Therefore we decided to implement some to specify a simple Pseudo MAM to Data Mover interface which than was developed and integrated into the SIVSS IBC demonstrator by HiStor and Grass Valley. This leads to a structure as s shown below.



**Figure 5: IBC Demonstrator Software & Interfaces**

### 2.5.1 Test with GFS File System

The SIVSS project aims to couple emerging commodity hardware technologies with open source system software. One of these software components is a cluster file systems. Even though one goal of SIVSS project is to be independent of any specific file system, the performance of a complete video server system will be dependant on it. Here GFS, an open source implementation that can be tuned to an adequate performance and functionality level was chosen.

The read performance was measured with different play speeds and two resolutions (2048x1556 and 1920x1080) with different values for `DPX_READ_IOSIZE` and `DPX_READ_THREADS` variables.

DPX read threads	Resolution	DPX ReadOISize				
		128K	256K	512K	1024K	2048K
8	2048x1556	10-11 fps	16-17 fps	20-21 fps	22-23 fps	21-23 fps
	1920x1080	12-14 fps	19-20 fps	26-31 fps	32-35 fps	32-35 fps
4	2048x1556	6-7 fps	11-12 fps	13-15 fps	20-21 fps	21-23 fps
	1920x1080	7-9 fps	12-15 fps	16-21 fps	26-30 fps	32-35 fps
2	2048x1556	4-5 fps	6-7 fps	9-11 fps	14-15 fps	16-19 fps
	1920x1080	5-6 fps	7-9 fps	10-14 fps	16-19 fps	24-26 fps

Table 6: GFS file system read performance

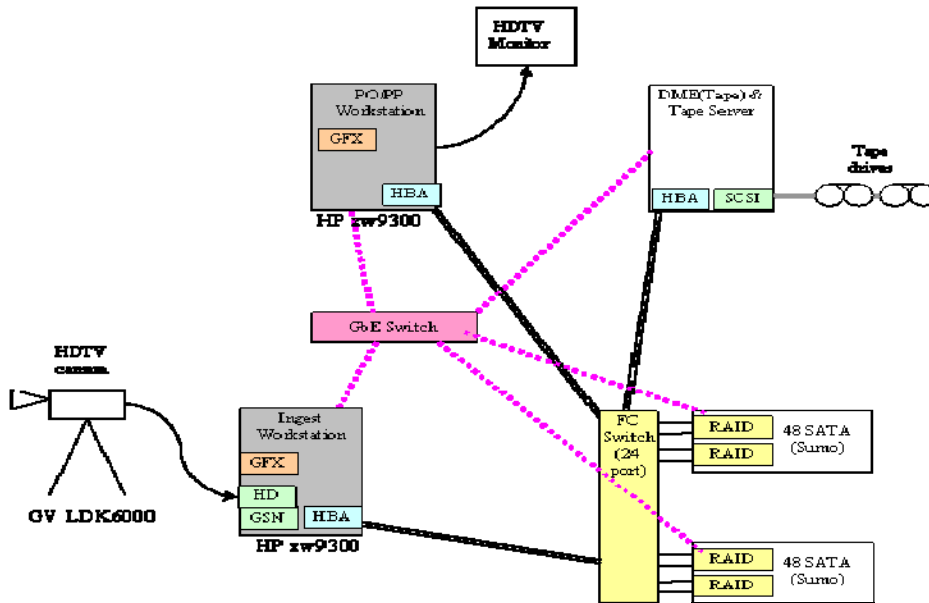
As one result it could be stated that 24 frames/sec could not yet achieved for 2k (2048x1556) resolution. This didn't matter for the SIVSS IBC demonstrator as the camera we planned to use at the stand was limited to 1920x1080 resolution.

### 2.5.2 IBC Show floor performance

Achieved sustained simultaneous ingest and playback of 1920x1080 live video from 2 independent hosts to Xyratex storage through GFS filesystem. This equals a sustained read/write performance of approx. 220 MByte, i.e. 25 file i/o per second to GFS. The basic MAM functionality allowed browsing, managing and viewing of recorded material together with archiving and retrieval to/from tape by using the HISTOR data mover.

Current limitations:

The code of the GFS Packages proofed to be immature, implementation of GFS caused Kernel "Oops Messages". Archiving speed was limited to 20-45MByte/sec.

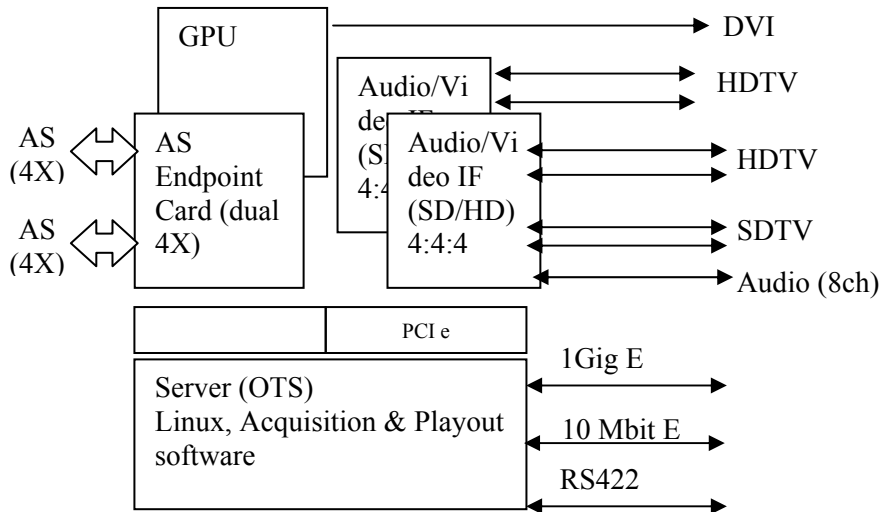


**Figure 6: IBC Demonstrator Hardware**

### 3 DESCRIPTION OF INPUT ACQUISITION SYSTEM

#### 3.1 Hardware Architecture

The Video I/O system is composed of the following:

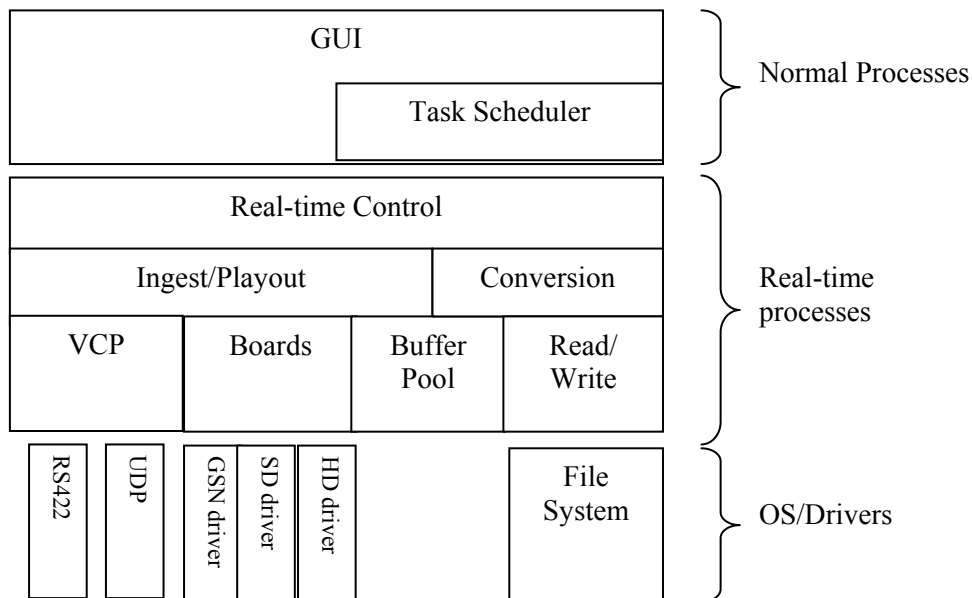


**Figure 7: Video I/O Hardware Architecture**

## 3.2 Software Architecture

### 3.2.1 Schema

The high-level Video I/O software modules are organized as follows:



**Figure 8: Video I/O Software Organization**

### 3.2.2 Modules

#### GUI

The GUI supports the following functions:

- Device configuration and setup.
- Selection of source material (for playout) or target location (for ingest).
- Manual control of ingest/playout.
- Defining a list of scheduled ingest/playout tasks: load/save task list; manual task entry & editing, import and conversion of EDL or pull list; task start/stop; progress monitoring.
- Monitoring of image data as ingest/playout proceeds.

#### Task Scheduler

The Task Scheduler processes the list of ingest/playout tasks defined by the GUI, sending the setup information to the real-time control module and monitoring the progress of each task.

#### Real-time Control

This software controls the overall processes of ingest or playout. Separate processes, with real-time scheduling priority, push/pull data through the buffer pool.

#### Ingest/Playout

This module is responsible for interface to the I/O devices at a high level, moving data between buffers and the devices.

### **Boards**

These modules, one per supported video/data I/O device type, provide an abstract interface layer to the different devices.

### **VCP (Virtual Control Panel)**

This module provides an interface to control functions of the external devices connected to the video/data I/O port (e.g. RS422, UDP, ...) and is used to control the device during an automated ingest/playout process.

### **Conversion**

This module applies whatever conversions are to be performed at ingest time. These are limited to those operations that can be performed in real time, such as conversions from one file format to another, although other operations can be supported with hardware acceleration.

### **Read/Write**

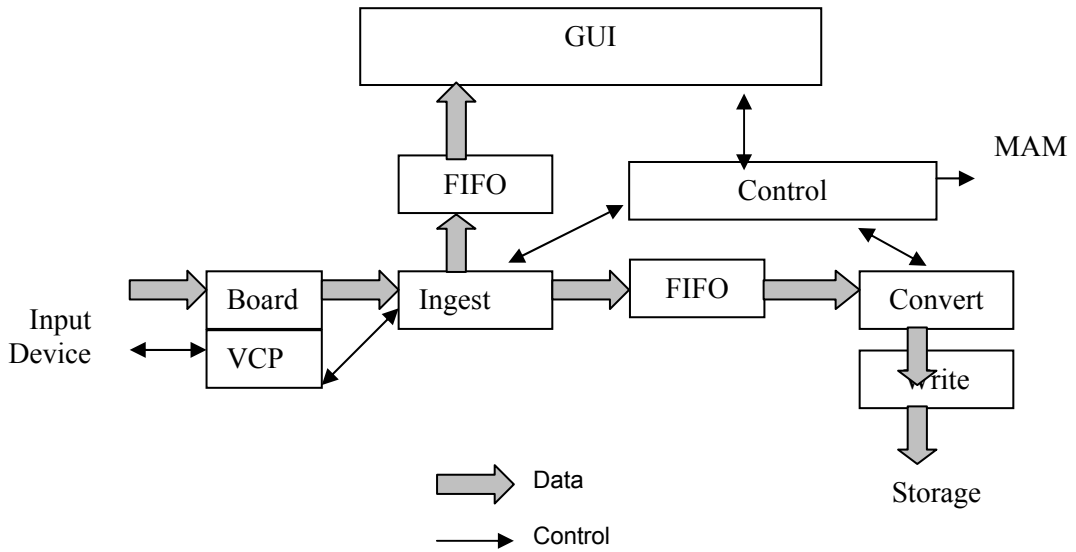
This module interfaces to the filesystem and manages reading/writing of image data. It runs with real-time scheduling priority. One or more threads operate in parallel as necessary, especially when writing, to hide latency associated with operations such as file creation.

### **Buffer Pool**

A pool of shared memory buffers provides protection from jitter. A FIFO containing references to memory buffers is filled by the ingest or read process and emptied by the playout or write process. A separate FIFO, limited to a maximum depth of one, is also filled by the ingest/read process, allowing the GUI to monitor the ingest/playout process.

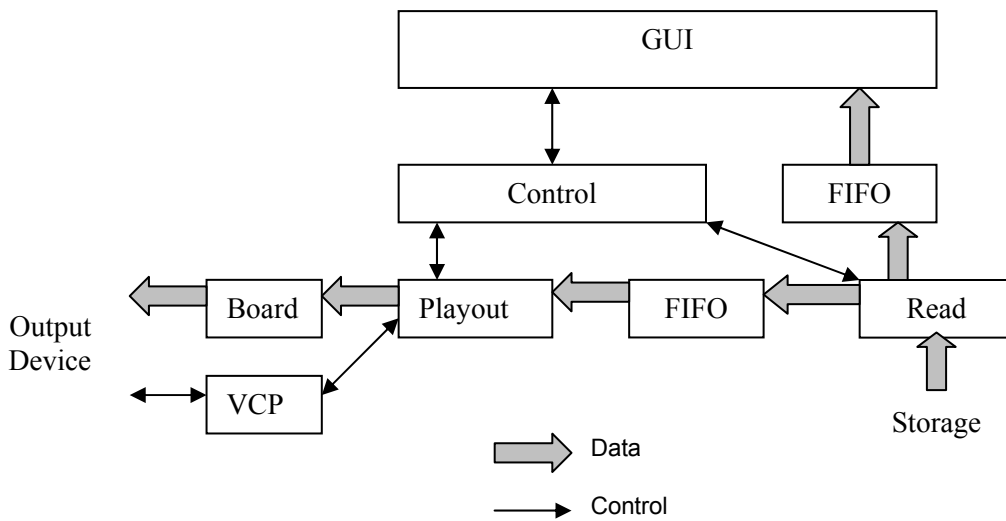
### 3.2.3 Control and Data Flow

#### 3.2.3.1 Ingest



**Figure 9: Video Ingest Operation**

#### 3.2.3.2 Playback



**Figure 10: Video Playback Operation**

### 3.2.4 Interfaces to other modules

#### 3.2.4.1 Interface to the asset management system

The acquisition system sends metadata to the asset management through a separate 1 Gbit Ethernet link. Specific functions are TBD, but anticipated to be

GUI

- Request location for new ingest of material.

Read/Write

- Notify MAM of new asset on completion of clip ingest.

The playout system receives playout commands from the asset management through the same Ethernet link

#### 3.2.4.2 Interface to storage

All interfaces from the Video I/O subsystem to storage are through the filesystem layer, with the following functions. The filesystem interface is a standard POSIX for the mandatory functions. For optional functions, filesystem-specific extensions can be supported.

##### **Get memory I/O requirements (optional)**

This optional function returns requirements, if any, for optimal I/O to/from the memory buffers that will be allocated, e.g. alignment, physical page size, minimum and maximum I/O sizes.

##### **Reserve/unreserve bandwidth (optional)**

This optional function invokes whatever software QoS provisions are supported by the filesystem, if any, to guarantee the requested I/O rate to the video I/O subsystem. If not supported, no bandwidth reservation will be made and the system must be provisioned so that storage is capable of meeting all bandwidth requirements of all clients simultaneously.

##### **Create file (mandatory)**

##### **Open file (mandatory)**

Standard POSIX open and create functions.

##### **Pre-allocate contiguous space (optional)**

Called prior to writing new data to allow the file system to allocate contiguous space, potentially aligned with volume geometry for optimal streaming playback.

##### **Write data (mandatory)**

##### **Read data (mandatory)**

Standard POSIX

### 3.3 Development Environment

- Developed on Linux Red Hat 9.0.
- Gcc compiler 3.2.2
- QT, a C++ application framework for GUI development
- Clearcase source code control system for Linux (development) and Windows (documentation)
- Valgrind

### 3.4 Statistics

#### 3.4.1 Lines of Code

Lines of code approx. 110 000

#### 3.4.2 Developed Libraries

- QGui: Controll Elements (Widgets) for SIVSS GUI
- QPlayer: Clip Preview
- QSBrowser: Device independent browsers, Open/Save/Import/Export dialogs
- QDevBrowser: DFS-Browser, VTR-Browser
- Audio: Audio I/O
- HiStor: Client/Server part for HiStor-DataMover
- QMover: Transfer panel GUI and business logic

#### 3.4.3 Used our libraries

- Img: needs for Clips which will be shown in the preview
- Cmgmt: Elements management (Clip,Directory)
- DevNonRT: Unix device
- DevServer: Device Manager
- Fip: Container classes, Error handling, Smart pointers
- Mover: mover client and server, needed for the realtime transfer
- Wrkl: handling of the workitems of the transfer list

#### 3.4.4 Used 3<sup>rd</sup> party Libraries

- Qt
- Corba
- STL
- Xerces
- DVS
- QLogic

