PCIe Data Recording Overview

Using StoreEngine™ and StorePak™ in Recording Mode

Abstract

PCI Express is increasingly being used as a sensor and processor interconnect technology in high performance embedded systems. This paper provides a brief overview of the configuration and usage of the Critical I/O StoreEngine and StorePak based data recorders for ultra high performance and easily configurable multi-GB/s PCIe data recording.

PCIe Data Recording Overview

PCIe is seeing increasing use as a sensor and processor interconnect technology in high performance embedded systems. It provides very high bandwidth data transport between sensors, processors, and storage systems. PCIe serves as a flexible, high performance interconnect for these systems, and is well suited for connections between sensor and processing systems and high-speed recorder systems, supporting multi-GB/s recording bandwidths with storage capacities ranging to tens of TB.

This paper provides an overview of the configuration and usage of the Critical I/O StoreEngine and StorePak based data recorders for flexible PCIe data recording. An overview of PCIe technology is provided, followed by an overview of the Critical I/O data recorder architecture. This is followed by a discussion of a variety of example recorder architectures and the related PCIe connectivity and recorder control options.

StoreEngine and StorePak –Recorder Building Blocks

StoreEngine and StorePak are flexible storage building blocks that can be used to implement a wide range of data storage systems. StoreEngine is an ultra-high performance VPX storage controller blade that hosts up to 6 TB of non-removable on-board SSD storage. StoreEngine simultaneously provides high performance recording functionality; serves block data (like a disk drive or RAID system); and provides NAS file sharing (like a NFS/CIFS file server).

StorePak is a VPX storage expansion blade that can host up to 10 TB of easily hot swappable SSD storage per blade. Together, StoreEngine and StorePak provide unmatched storage capability, ultra high performance and high capacity within a small size, weight, and power (SWaP) footprint.

Both StoreEngine and StorePak provide rich PCIe connectivity, with up to eight x4 PCIe backplane ports per VPX board. These ports are used for connections to the user's data sources as well as for interconnections between StoreEngines and StorePaks. Both boards feature PCIe switches which are fully partitionable, providing greatly increased system architecture options.

StoreEngine/StorePak Data Recorder Overview

A simple example of a *Direct Mode* StoreEngine/StorePak data recorder configuration is shown in Figure 1. It is comprised of a set of StoreEngine and StorePak boards and associated software functionality that work together to provide a high speed PCIe based data recording function. The Recorder provides a capability to record multiple streams of data directly from PCIe connected "data sources" at rates of over 2.0 GB/s per StorePak, or 750 MB/s per StoreEngine (if used alone). Multiple StorePak and/or StoreEngine modules can be combined for higher aggregate recording rates. Recorded data may be played back to PCIe connected "data destinations", or alternatively recorded data may be accessed via 1 or 10 Gb Ethernet connections from the StoreEngine using standard NFS, FTP, or CIFS/SMB file access.

Note that this example used *Direct Mode*. The StoreEngine/StorePak recorder architecture also supports a very flexible *Buffered Mode*, where data is first buffered within the StoreEngine before being written to SSD storage.

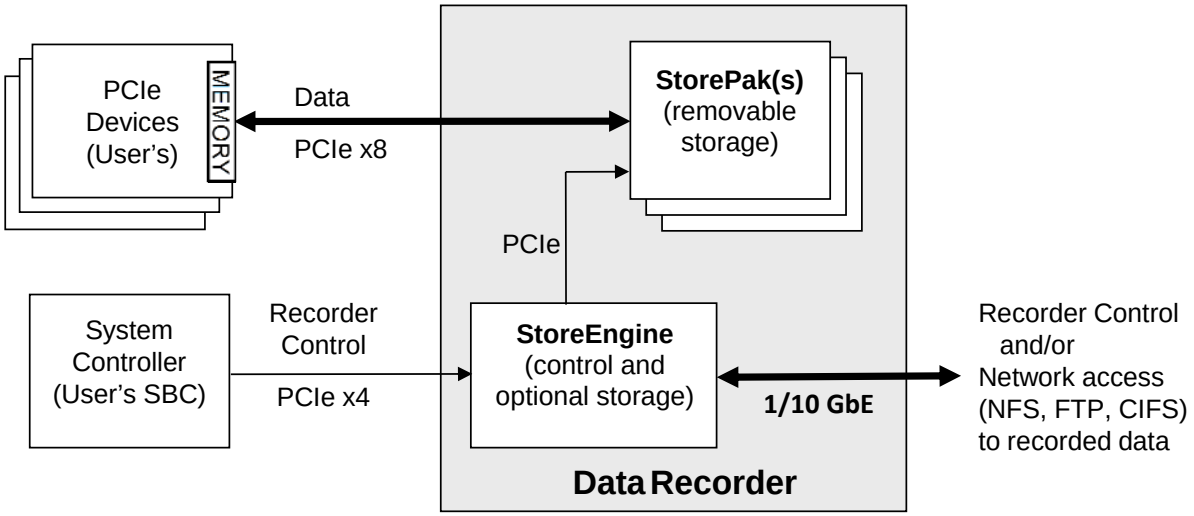

Figure 1. One example of a *Direct Mode* StoreEngine/StorePak based PCIe recorder

In Recording mode the StoreEngines (and StorePaks) accept a stream of raw data blocks from either "simple" PCIe sources (ADC, FPGA board with built-in DMA), "intelligent" PCIe sources (e.g., DSP or FPGA board with a memory mapped PCIe interface), or Processor Board sources. The StoreEngine completely manages the SSD storage, and writes the data blocks to SSDs, using a recording file system. This file system also allows striping data from a single data stream across multiple StoreEngines (or StorePaks) for increased capacity and performance. The playback of recorded data may be via PCIe, or via 1/10 Gb Ethernet (NFS, FTP, or CIFS/SMB).

Regardless of the number of boards used in the recorder system, all data recorder functionality is managed through a single StoreEngine board. In some applications, this access and control is provided through a Recording Driver that can be hosted on a user supplied "system controller" SBC, which is PCIe connected to the StoreEngine. For initial Data Recorder system configuration, a built-in web based management function that resides on the StoreEngine is used to configure the data recorder, and can subsequently be used to monitor the StoreEngine and StorePak boards, and associated data recorder functionality. Recording control can also be provided through Recording Drivers available for VxWorks and Linux, or through a Recorder Network Control Protocol.

PCIe Board to Board Connectivity

Several different methods can be used to provide PCIe connections within a chassis, and between multiple chasses. Backplane connections are most often used within a chassis to provide PCIe connectivity between boards. VPX backplanes in particular pre-define a "data plane" and an "expansion plane", both of which are used to provide multiple high speed serial data paths between boards which can be leveraged for PCIe connectivity. Most often, these intra-board data paths implement "fat pipes", which consist of a group of four high speed serial lanes in each direction. Each VPX fat-pipe is thus equivalent to a PCIe x4 link.

VPX backplane *expansion plane* connections are almost always point to point, and are generally "nearest neighbor" connections. That is, they are used to connect two adjacent boards. VPX expansion plane connections can be used in conjunction with point-to-point mesh connections to expand the set of boards that can directly communicate in point-to-point architectures.

VPX backplane *data plane* architectures are divided into two general categories: *point-to-point*, and *switched*. A point-to-point architecture implements multiple point-to-point fat-pipe connections between boards. Often a mesh topology is used; where groups of five boards are interconnected using a set of point-to-point connections that allow each board in the group to connect directly to every other board in the group. The chief disadvantage of a point-to-point architecture is a limitation on the number of boards that can directly communicate.

PCIe Data Recorder Configuration

The PCIe capabilities discussed above can be leveraged to build highly flexible and capable data recording systems. Switch partitioning and non-transparent bridging in particular are key capabilities needed for certain multi-channel recording architectures. Critical I/O's StoreEngine and StorePak provide very rich PCIe connectivity, with up to seven x4 PCIe backplane ports per VPX board. These ports are used for connections to the user's data and control sources as well as for interconnections between StoreEngines and StorePaks. Both boards feature PCIe switches which are fully partitionable, providing increased recorder architecture options.

StoreEngine boards can be configured to act as PCIe root devices, or PCIe endpoint devices, or to do both simultaneously on separate PCIe domains (using StoreEngine's switch partitioning capability). A StoreEngine will typically be configured as a PCIe endpoint when it is connected to a processor board's which has a PCIe root capability. A StoreEngine will (additionally) be configured as a PCIe root when it is controlling StorePak board(s), or if it is controlling "simple" PCIe data sources such as ADCs. StorePak boards are always PCIe endpoint devices, and are generally connected to a StoreEngine board root port for recorder applications.

Direct vs. Buffered Mode Recorder Operation

Direct mode is a way of using StoreEngine or StorePak that results in data being transferred directly from PCIe data sources to/from the storage resources hosted on StorePak modules, without first being buffered in a StoreEngine. This contrasts with Buffered mode, where data is buffered in a StoreEngine prior to being moved to the storage resource. The two modes are illustrated in figures 2 and 3 below.

Direct mode has the advantage of higher performance, as data is not first buffered in a StoreEngine. Direct mode requires that the storage resource (e.g. StorePaks) be able to read data from PCIe accessible memory on the PCIe data sources. This is sometimes referred to as a "pull" data transfer model. Direct mode has a further advantage of increased storage density, as multiple StorePak modules may be controlled using a single StoreEngine module. Only RAID 0 is supported in Direct mode.

In contrast, buffered mode operation requires that data first be buffered in a StoreEngine. Buffered mode has the advantage of flexibility. PCIe data sources may "push" data into data buffers on the StoreEngine(s), rather than relying on StorePaks/StoreEngines to "pull" data. The timing of these data transfers is more flexible, and can be controlled by the data source. StoreEngine transfers data to the StorePak(s) on a decoupled timeline. Both RAID 0 and RAID 5 are supported in buffered mode.

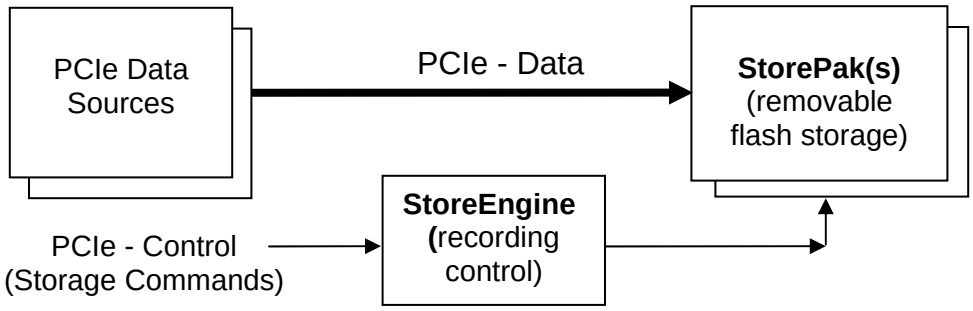

Figure 2. Direct Mode Operation

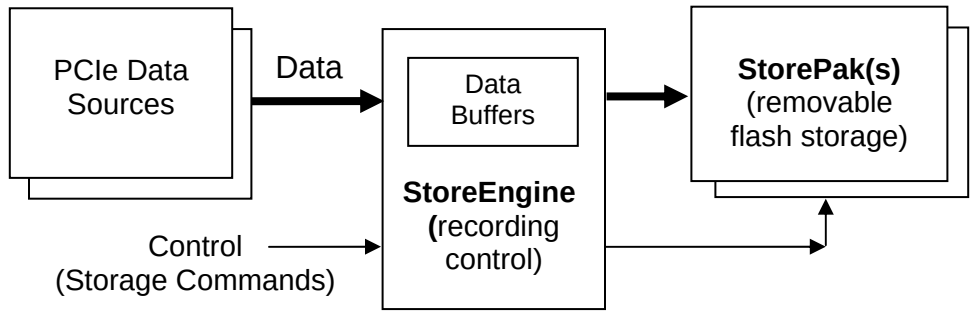

Figure 3. Buffered Mode Operation

The recorder also supports concurrent use of direct mode and buffered mode operation. For example, direct mode may be used to record streams of data sourced directly from PCIe data source devices, while buffered mode may be used concurrently to record data streams sourced from the System Controller SBC.

Recorder Data Source Options

Options for the recorded data source range from a data stream from a simple PCIe connected ADC, to a more intelligent PCIe connected device such as an FPGA processor, or a PCIe connected standard CPU board (SBC). The source may also consist of 10GbE network connection to record from an Ethernet or UDP or TCP data stream. Recorder operation is completely symmetrical; data recorded from any source can be "played back" to any destination.

Simple PCIe Source (example: ADC)

For simple PCIe sources like ADCs, the StoreEngine feeds the PCI source with a sequence of PCIe addresses and block sizes. The PCIe source (ADC) then DMAs blocks of data to the supplied addresses, which actually point to memory buffers hosted on the StoreEngine(s). Simple recording sources like ADCs can be completely controlled by the StoreEngine recorder.

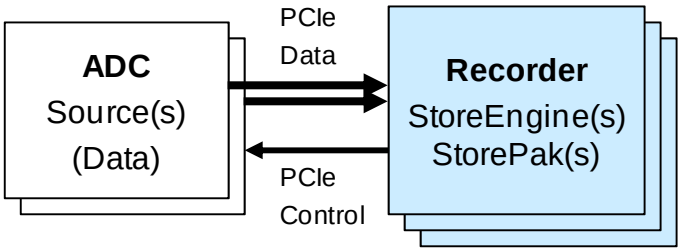

Figure 4. Example: A simple PCIe data source

Intelligent PCIe Source (example: FPGA signal processor board)

Intelligent PCIe sources differ from simple PCIe sources in that an external controller element (typically some sort of processor board, labeled System Controller in the diagram below) is assumed. The system controller function controls the writing/reading of data to/from the recorder via commands that are sent from the system controller to the recorder through the Recorder Mode driver hosted on the system controller board.

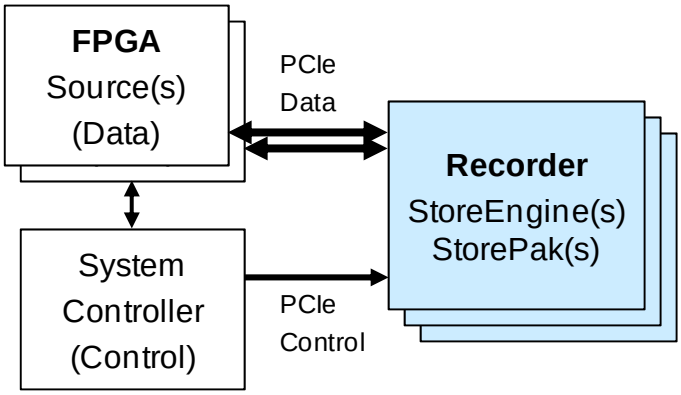

Figure 5. Example: An intelligent PCIe Data Source

Processor Data Source (example: PPC or x86 CPU board)

In this model, a user's processor board runs a lightweight Recorder Driver, which simply coordinates the DMA transfer of large blocks of data directly from host memory to the recorder. This is a highly efficient method of recording large volumes of data from a processor board that relieves the processor from the CPU intensive task of managing low level storage.

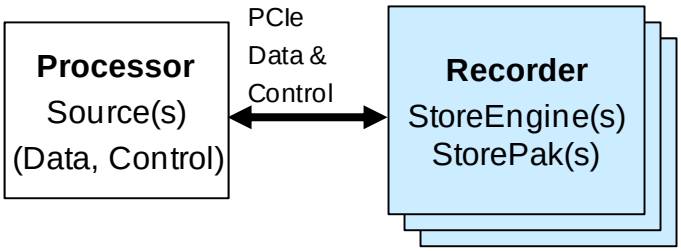

Figure 6. Example: A Processor data source

Multi-Channel and Multi-Stream Configurations

A multi-channel data recorder configuration is based on the concept of data streams. A data stream is a bidirectional "logical connection" (flow of data) between an *external PCIe Device* and a *StoreEngine LUN*. For a non-switched PCIe backplane, each stream can be fully defined by the StoreEngine or StorePak PCIe port to which the PCIe source/destination device is connected and the StoreEngine LUN that is used to store the data. Note that for direct mode the term LUN and RAID are equivalent (RAID md0 is LUN 0 and vice-versa). Note that while the storage for the LUNs may be physically hosted on multiple StorePak modules, it is always configured and managed through a single StoreEngine module.

A single LUN can service multiple streams. For example, two or more different PCIe devices may read or write data for the same StorePak LUN. In addition, each StorePak can host multiple LUNs. A single stream can provide multiple user channels of data. Each channel of user data is written to a sequence of files that are associated with that stream and channel. Each time a user does an open/close (i.e., start/stop) on the channel data between the open and close is written to a unique file. Note that these channels are shared for all streams that access a common LUN.

Terminology

- LUN (RAID) – A LUN (Logical Unit Number) is a storage container that is hosted on one or more StorePaks. There is a one-to-one mapping between a LUN and an underlying RAID (i.e. LUN0 is RAID0 (md0) and vice-versa). A StorePak may host multiple LUNs.

- PCIe Device – A PCIe device is a data source/destination PCIe endpoint that is connected to a StorePak via a direct PCIe link. A StorePak may be connected to multiple PCIe devices.

- Stream – A bidirectional "connection" defined by 1) a PCIe data source (or destination) and 2) a LUN hosted on a StorePak. A StorePak LUN may service multiple streams.

- Channel – A channel is a logical flow of data over a stream. Each channel of data is associated with a set of files within a StorePak LUN. A stream may be used to transport multiple channels. *Note that if several streams are used to access the same LUN, each stream will access the same set of files, i.e., Stream 1, Channel 1 will access the exact same files as Stream 2, Channel 1 if both streams are connected to the same LUN.*

It is important to understand that Channels are defined with respect to their associated LUNs. That is, an access to Channel Y using any stream that connected to LUN X will result in an access to the exact same file, regardless of which stream is used for the access.

Recorder Control

There are three methods available to control a StoreEngine based data recorder:

- The Recording Mode driver (on a user's SBC), communicating with StoreEngine over a PCIe interface (recorder Processor Controlled mode)

- StoreEngine's built-in recorder management web pages. (for UDP, TCP, ADC and FPGA modes)

- The Recorder Network Control protocol. (for UDP, TCP, ADC and FPGA recording modes)

Recorder Control Using the Recorder Network Control Protocol

The Recorder Network Control Protocol is an Ethernet network protocol designed to provide control a StoreEngine based data recorder. Through this interface, a user can perform functions such as set the direction of data flow, start and stop data flow, get the state of the recorder, etc.

There are two components to the control of the Recorder Network Control interface:

- The Recorder Network Contorl Server, which resides on a StoreEngine.
- The Recorder Network Control Client, which resides on the users SBC or PC.

The Recorder Server resides on the StoreEngine. The Recorder Client can reside on any computer and uses a standard Ethernet network interface to communicate with the Server. The Recorder Client issues commands to the Recorder Server. The Server then executes the commands on the StoreEngine and returns a response to the Client.

Communications between these two components is accomplished by a socket interface. A TCP socket is opened up to accommodate request/response pairs between the Client and Server. The Client will only write requests to the Server and the Server will only send responses to the Client/

The Recorder Network Control Protocol is described detail in the Recorder Network Control User's Manual.

Recorder Control Using the Recording Mode Driver

The Recording Mode Driver allows a user provided System Controller SBC to interface with a StoreEngine Recorder over a PCIe interface. The driver provides functionality to initialize and control the StoreEngine Recorder, and open, read, and write (record) data for multiple recorder streams. A standard character device interface is provided as the user API. The recording driver is used to initiate data transfers to/from the SBC that hosts the driver, as well as to/from the other PCIe Devices that are directly connected to StoreEngines and StorePaks.

The User Recorder Control Application is the code that runs on the user's System Controller SBC that communicates with and coordinates data flow to (writes) and from (reads) the recorder. The User Recorder Control Application makes calls to standard Linux file I/O functions to do this, as illustrated in the example pseudo code below. (files are name Sxx_Cxx_Fxx, where S=stream, C=Channel, F=file number whithin that stream/channel).

```
fd_a = open(/dev/S01_C01)   // Channel 1, Stream 1
fd_b = open(/dev/S01_C02)   // Channel 2, also within Stream 1
fd_c = open(/dev/S02_C01)   // Channel 1, Stream 2
write(fd_a, block_w)        // write data block w to Stream1 file S01_C01_F01
write(fd_b, block_x)        // write data block x to Stream1 file S01_C02_F01
```

```
write(fd_c, block_y)          // write data block y to Stream2 file S02_C01_F01
close(fd_a)                   // close Stream1 file S01_C01_F01
fd_a = open(/dev/S01_C01)     // create a new file Stream1 file S01_C00_F02
write(fd_a, block_z)          // write data block z to Stream1 file S01_C00_F02
```

Use of the Recorder Driver is described detail in the Recorder Driver User's Manual

Ethernet (NFS, FTP, CIFS/SMB) Network Access to Recorded Data

Recorded data may be network accessed via a 1/10 Gb Ethernet connection using NFS (or FTP). Each Recorder LUN has an associated instance of a pseudo file system. These pseudo file system instances can be individually NFS exported (or CIFS/SMB or FTP) through StoreEngine. Each LUN (NFS export) will then appear to the user (NFS client) as a separate NFS mount. Files within these NFS mounts will appear as:

Data Files: Sxx_Cyy_Fzz (Stream_Channel_File)
BTS Files: Sxx_Cyy_Tz (BTS is Block Time Stamp; has timestamps for each block)

Summary

PCI Express provides a very high bandwidth data transport between sensors, processors, and storage systems. Core PCIe capabilities are leveraged by Critical I/O's StoreEngine and StorePak storage blades to build highly flexible and capable data recording system. StoreEngine is an ultra-high performance VPX storage controller blade that hosts up to 6 TB of non-removable on-board SSD storage. StorePak is a VPX storage expansion blade that can host up to 10 TB of easily hot swappable SSD storage per blade. Both boards have rich PCIe connectivity and integrated PCIe switches which are fully partitionable, supporting flexible data recorder architecture options.

StoreEngine/StorePak based recorders can record multiple streams of data directly from PCIe connected data sources. Sustained recording performance of over 2 GB/s per StorePak, or 750 MB/s per StoreEngine (if used alone) can be achieved, and multiple StorePak and/or StoreEngine modules can be combined for higher aggregate recording rates and capacities. Recorded data may be played back to PCIe connected "data destinations", or alternatively recorded data may be played back via a Gigabit Ethernet connection from StoreEngine, using standard NFS, FTP, or CIFS/SMB file access.

Critical I/O's StoreEngine and StorePak based PCIe recording architectures and recording management software offer a cost effective, readily scalable, highly flexible, and easily tailorable high performance data recording solution.