The Axion-CL

Hardware Reference Manual

BitFlow, Inc.
400 West Cummings Park, Suite 5050
Woburn, MA 01801
USA
Tel: 781-932-2900
Sales: sales@bitflow.com
Support: support@bitflow.com
Web: www.bitflow.com
Revision A.2

Revision History:

Revision	Date	Comments
A.0	2016-04-05	First printing
A.1	2017-12-16	Updates, synchronized to SDK 6.30
A.2	2021-03-03	Added 1xB, 2xB, 4xB and synchronize to SDK 6.5

Table of Contents

P - Preface

1 - General Description and Architecture

2 - The StreamSync Acquisition Engine

5 - The Aon, Axion, Claxon and Cyton I/O System

6 - The I/O System Registers

7 - Encoder Divider

8 - Quadrature Encoder

9 - Quadrature Encoder and Divider Registers

10 - System Probe

11 - Axion Camera Link Registers

12 - Axion Power and Miscellaneous Registers

13 - Specifications

14 - Mechanical

Preface

Chapter P

P.1 Purpose

This Hardware Reference Manual is intended for anyone using an Axion-CL frame grabber. The purpose of this manual is two-fold. First, this manual completely describes how the board works. Second, it is a reference manual describing in detail the functionality of all of the board's registers.

P.1.1 Support Services

BitFlow, Inc. provides both sales and technical support for the Axion family of products.

P.1.2 Technical Support

Our web site is www.bitflow.com.

Technical support is available at 781-932-2900 from 9:00 AM to 6:00 PM Eastern Standard Time, Monday through Friday.

For technical support by email (support@bitflow.com) or by FAX (781-933-9965), please include the following:

Product name
Camera type and mode being used
Software revision number
Computer CPU type, PCI chipset, bus speed
Operating system
Example code (if applicable)

P.1.3 Sales Support

Contact your local BitFlow Sales Representative, Dealer, or Distributor for information about how BitFlow can help you solve your most demanding camera interfacing problems. Refer to the BitFlow, Inc. web site (www.bitflow.com) for a list of North American representatives and worldwide distributors.

P.1.4 Conventions

Table P-1 shows the conventions that are used for numerical notation in this manual.

Table P-1 Base Abbreviations

Base	Designator	Example
Binary	b	1010b
Decimal	None	4223
Hexidecimal	h	12fah

Table P-2 shows the numerical abbreviations that are used in this manual.

Table P-2 Numeric Abbreviations

Abbreviation	Value	Example
K	1024	256K
M	1048576	1M

P.2 Bitfield definitions

P.2.1 Example Bitfield Definition

is what each bitfield definition looks like:

BITFIELD R/W, CON0[7..0], Axion-CL

Bitfield discussion.

P.2.2 Bitfield Definition Explanation.

The definitions is broken into three sections (see Table P-3).

Table P-3 Bitfield Sections.

Section	Meaning
Bitfield name	This is the name of the bitfield. This name is use to program this bitfield from software or from within and camera configuration file. When programming bitfields from software using a Peek or Poke function, the bitfield is preceded with "REG_". For example the bitfield CFREQ is referred to in software as REG_CFREQ.
Bitfield details	This section describes how the bitfield is accessed. The first part describes the how the bits can be accessed. For example R/W means the register can be both read and writen. See theTable P-4 for details.The second part is the wide register that the bitfield is located in. In the example above this bitfield is in CON0. Following the wide register name is a bitfield location description, in hardware engineering format. For example, [7..0], means the bitfield has 8 bits, location in positions 0 to 7. Finally this section also indicates if the register is specific to only one product family.
Bitfield discussion	This section explains the purposed of the bitfield in detail. Usually meaning of every possible value of the bitfield is listed.

Table P-4 explains the abbreviations used in the bitfield definitions.

Table P-4 Abbreviations

Access	Meaning
R/W	Bitfield can be read and written.
RO	Bitfield can only be read. Writing to this bit has no effect.
WO	Bitfield can only be written. Reading from this bit will return meaningless values.
Karbon-CL	This bitfield is functional only the Karbon-CL.
Karbon-CXP	This bitfield is functional only the Karbon-CXP.
Neon	This bitfield is functional only the Neon
R64	This bitfield is functional only the R64 family.
Alta	This bitfield is functional only the Alta family.
Cyton-CXP	This bitfield is functional only on the Cyton-CXP family
Axion-CL	This bitfield is functional only on the Axion-CL family
Aon-CXP	This bitfield is functional on the Aon-CXP family
Claxon-CXP	This bitfield is functional on the Axion-CXP family

General Description and Architecture

Chapter 1

1.1 The Axion-CL family

The purpose of this chapter is to explain, at a block diagram level, how the Axion-CL works. Currently there are five main models in the Axion-CL family:

AXN-PC2-1xE, support for one Base, Medium, Full or 80-bit camera
AXN-PC2-2xE, support for two Base, Medium, Full or 80-bit cameras
AXN-PC2-1xB, supports for one Base camera
AXN-PC2-2xB, supports for two Base cameras
AXN-PC2-4xB, supports for four Base cameras

1.1.1 Camera Link

In order to understand how the Axion-CL works, it is helpful to understand the basics of Camera Link. It is beyond the scope of this manual to describe how Camera Link works, however, more information on the Camera Link specification is available from http://www.visiononline.org/.

1.1.2 Virtual vs. Hardware

It's important to understand how this manual works. Some chapters of this manual discuss the Axion-CL as a hardware platform (this chapter is a good example). While other chapters discuss the details of the virtual frame grabbers (VFG) that this hardware platform supports. The concept of the virtual frame grabber is described below, but basically the idea is that one hardware platform can support more than one device. In the case of the Axion-CL, these devices are frame grabbers.

Note that we are not using the word virtual in the sense of "a software virtualization of a hardware device", these VFGs are real hardware. The reason we using "virtual" is because the term "frame grabber" has more than one meaning. It can mean the piece of hardware that you put in your computer, or it can mean the device that the your software application is controlling and getting images from. For the purposes of this manual, "virtual frame grabber" means the device that your application is interfaces with. While this might sound complicated, the implementation is simple. You plug our Axion-CL frame grabber into your PC, and your application interacts with one or more VFGs available. Everything else is taken care of by the BitFlow drivers.

1.1.3 The Virtual Frame Grabber (VFG)

The idea behind the VFG is to separate the hardware platform (connectors, laminate, FPGAs, etc.) from the frame grabbing functionality that software applications work with. The primary reason behind this separation is that the turn around time for hardware is much longer than the turn around time for modifying virtual frame grabbers. To create a brand new virtual frame grabber, or to modify an existing one, simply requires writing new firmware or updating existing firmware.

The idea of modifying a frame grabber by making changes to its firmware is not new. BitFlow has been doing this since its very first product. However, unique to BitFlow products is the fact the entire frame grabber is written in firmware. The only fixed hardware components are the interfaces to the outside world (e.g. the interface chips on the front end). Everything else that makes up the board, camera control, data buffering, DMA engine, etc. is written in firmware. This gives the platform incredible levels of flexibility and opens the door to unlimited customization.

1.1.4 Axion Configuration Spaces

The Axion-CL model supports up to four VFGs. Each VFG appears to operating system and your software as a separate device. The block diagrams for the different models are shown in the following section.

1.2 General Description

The Axion-CL is a x4 PCI Express Gen 2 board. It can work in any PCI Express slot that it can fit it. Usually this means an x4, x8 or x16 slot. However, some mother boards have x1 slots with x4 connectors. The Axion-CL will work in these slots, although performance my be somewhat reduced. The Axion-CL is a Gen 2 PCIe device, but it will work in Gen 1 slots, though DMA performance will be degraded. DMA performance will be the same for both Gen 2 and Gen 3 slots.

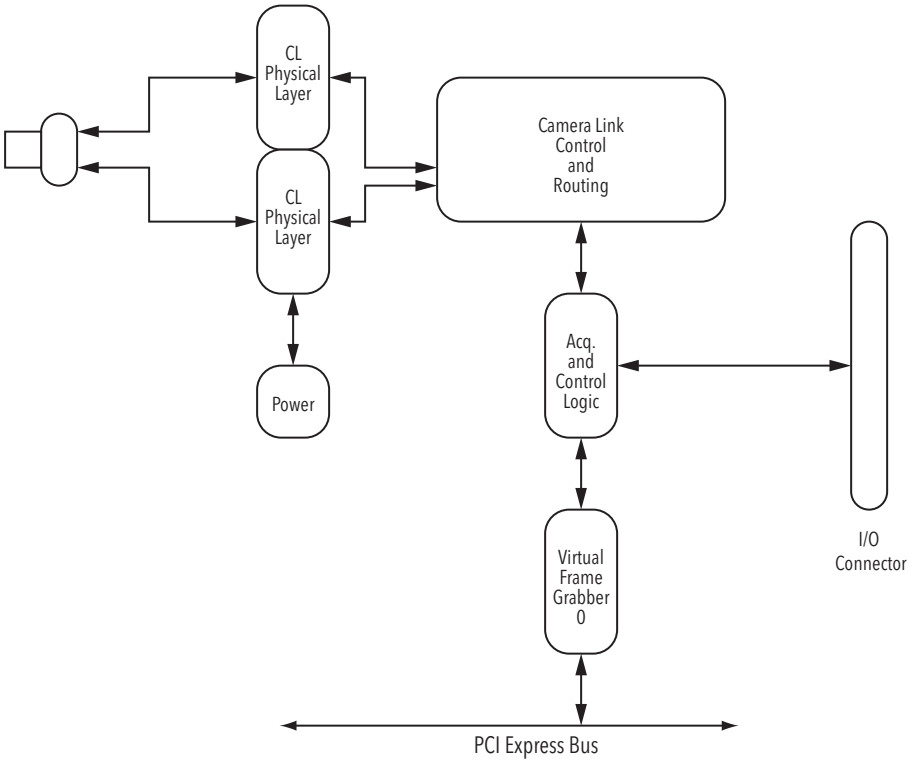

Figure 1-1 The Axion-1xE Block Diagram

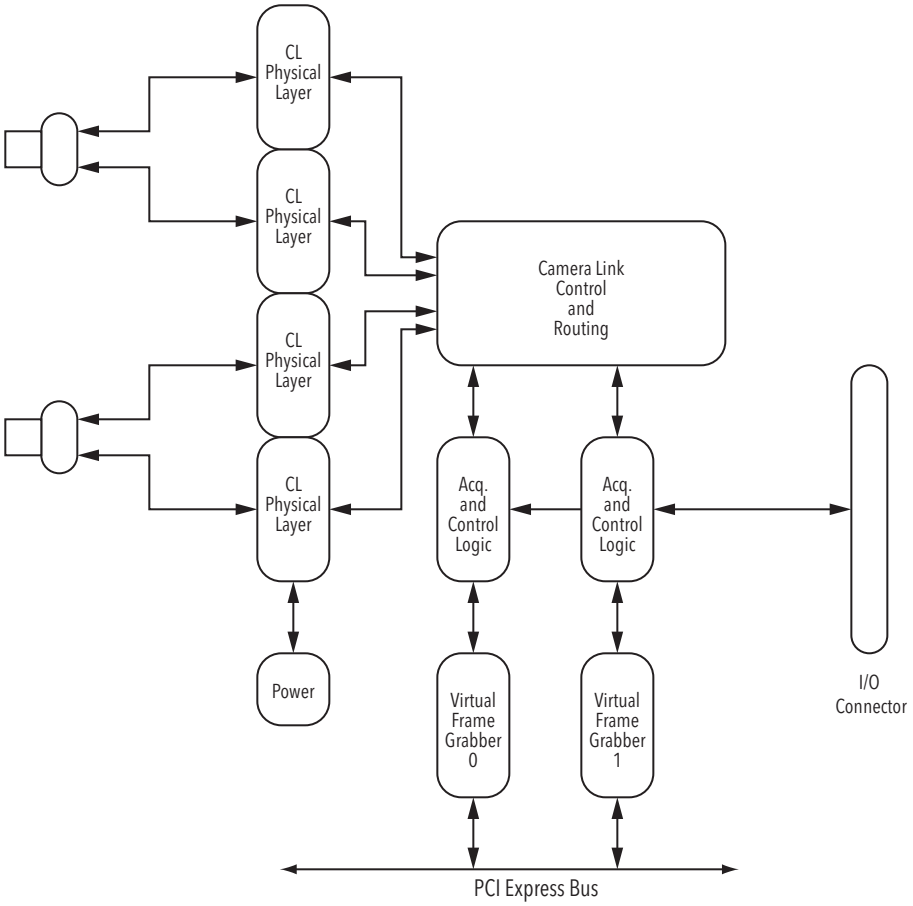

Figure 1-2 The Axion-2xE Block Diagram

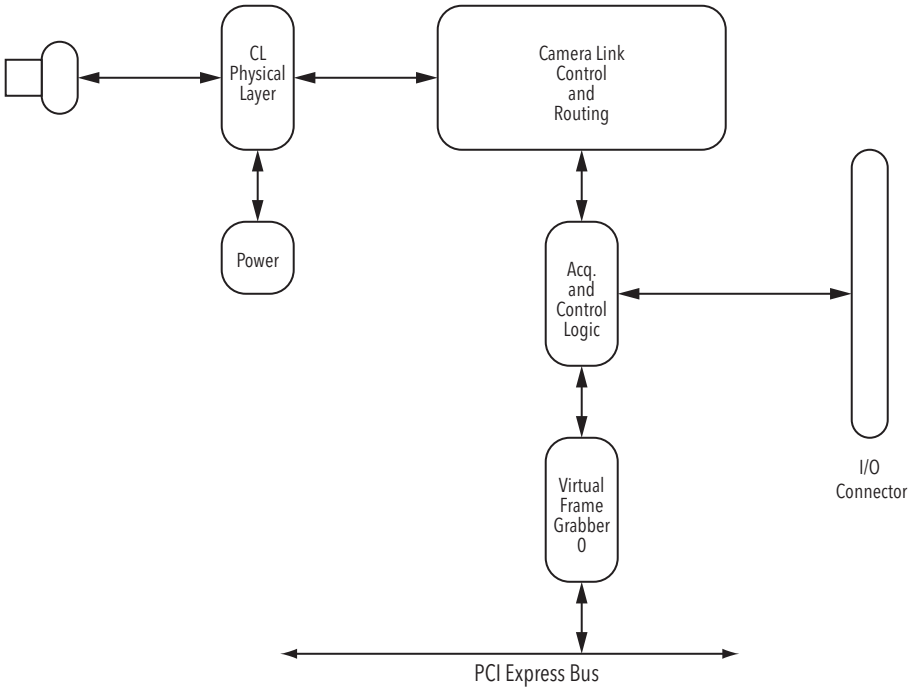

Figure 1-3 The Axion-1xB Block Diagram

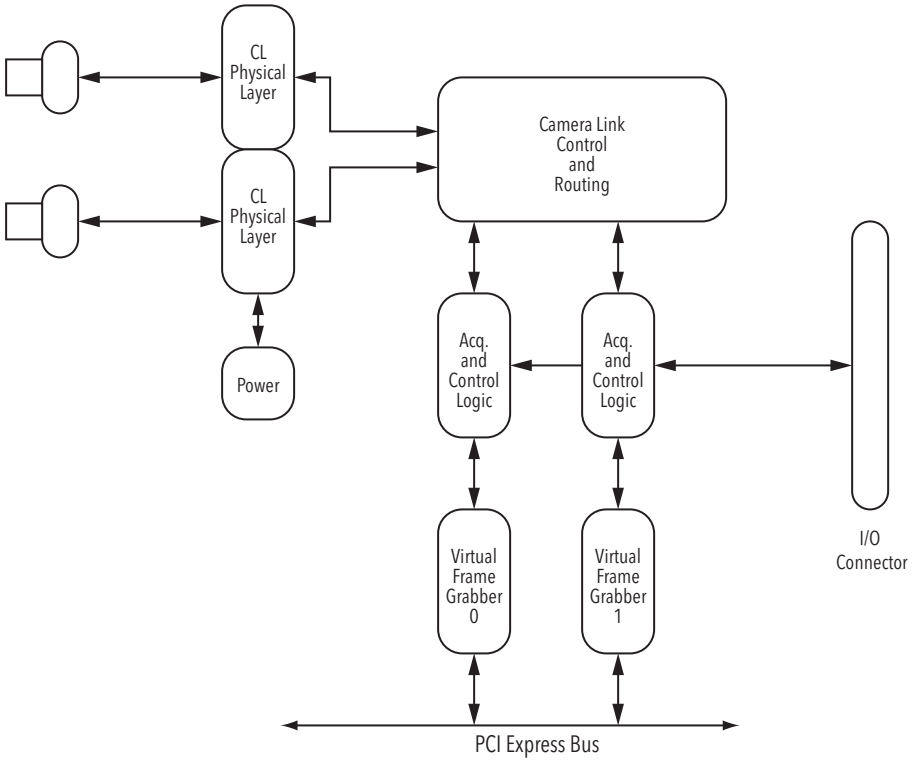

Figure 1-4 The Axion-2xB Block Diagram

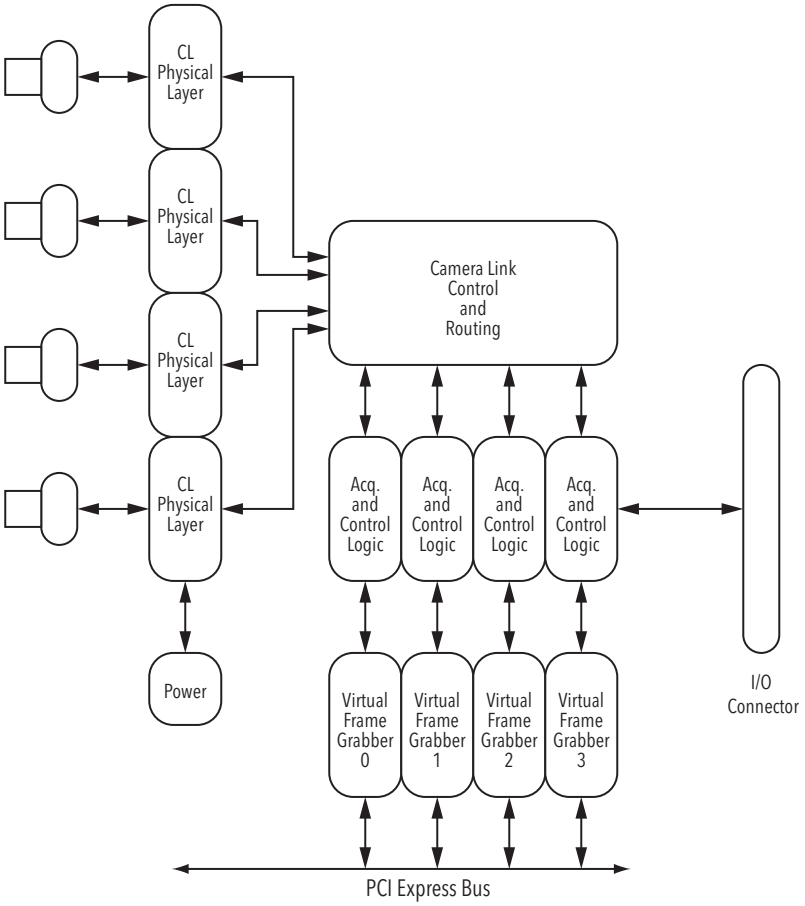

Figure 1-5 The Axion-4xB Block Diagram

1.2.1 Video Data

Camera Link Video data is parallel digital data. The Camera Link physical layer takes a number of parallel signals and "serializes" them on to a small number of high speed signals. One the frame grabber size, the data is "de-serialized". The video data is not sent it packets. Synchronization is achieved via individual clock and synchronization signals. Data width can be from 8 to 80 bits.

1.2.2 Camera Control

Camera Link facilities camera control via a RS232 serial connection. The Camera Link cable contains a bidirectional serial link. This can be used to send and receive control data to/from the camera. The serial link is always synchronous. The camera does not send data without being requested from the host.

The Camera Link specification requires frame grabber manufactures to provide a serial communications DLL which exposes a communications API. This API is open and can be used by customers in their own software. This is installed automatically when the BitFlow SDK installer is run.

Each camera vendor has its on Camera Link control protocol. There is no standard. Most camera vendors provide a Camera Control utility which uses the frame grabber serial DLL to communicate through the frame grabber to the camera. Most camera vendors also document their protocol so end users can directly program their cameras.

1.2.3 Camera Link Trigger Support

Camera Link supports very low latency triggering (from the frame grabber to the camera) via four dedicated signals CC1, CC2, CC3 and CC4. These can be driven by a number of different sources on the Axion-CL.

1.2.4 Axion I/O system

The Axion-CL has a sophisticated I/O system, which is extremely flexible. The system take in many inputs, routes them to a number of internal signals which can be further manipulate, then routes the results to a wide rand of outputs. The I/O system is discuss in more detail in Section 7.1.

1.2.5 The Timing Sequencer Signal Generator

With the introduction of the Axion-CL BitFlow introduced a new signal generator, the Timing Sequencer. The Axion-CL also uses this timing generator. The Timing Sequencer (TS) is more flexible and more power than the timing generators used on early BitFlow frame grabbers. It has the ability to output multiple different size pulses, each of which can free-run or require a trigger. The TS is more accurate than the NTG and has a finer granularity. The TS can also be changed on the fly, with switch overs to the new timing exactly synchronized. See section 4.1 for more information.

1.2.6 The StreamSync System

Starting with the Cyton-CXP, BitFlow introduced a new DMA engine, the Axion-CL has this same engine. It is designed from scratch acquisition and called the StreamSync system. The StreamSync system has been designed to optimize acquisition and DMA throughput over the PCIe bus given a wider variety of internal PC conditions. In addition, the Stream Sync system has been designed to automatically resync and recover should t every be packet lost (either on the input or the output side of the board), resulting in much more usable and fault tolerant image sequences in host memory. For more information see Section 2.1 and Section 3.1.

1.2.7 The Volume Of Interest Acquisition Engine

The Axion-CL introduces the concept of Volume of Interest (VOI) as part of its Stream-Sync Acquisition Engine. This has been designed from the ground up to satisfy the needs of real world machine vision application. The VOI provide robust and flexible programming that can handle of a wide variety of pixel, line, frame and sequence acquisition commands either manually from software control, or externally via hardware triggers. There is full support for X and Y offsets, X and Y Region of interests, sequences and sequences of sequences. See section 2.2 for more information.

1.2.8 Camera Link Camera Power (PoCL)

The Camera Link PoCL specification specifies that the frame grabber provide up to 4W at 12V for powering an attached camera. The Axion-CL conforms to this specification and provides power on all of its CL connectors. This can provide up to 8 W for medium/full/80-bit Camera Link cameras. Some cameras do not require power, so the Axion-CL can optionally turn power on or off via its registers. Normally this information is part of the camera configuration file, thus files for cameras that require power are so indicated.

The Axion-CL automatically powers up all connectors that need power (i.e. correctly respond to the sense circuit). This happens as soon as the system is booted.

The Axion-CL constantly monitors the current on each Camera Link connector, if either over current or under current conditions exist, the power will be turned off. The monitoring system is purely in hardware, so no host computer intervention is required in order to safeguard the power source.

For situation where the camera requires more power than the PCIe bus can supply to the frame grabber, the P4 connector can be use. This connector can connect to the PC's power supply and all camera power will come from this connector.

1.3 Firmware

Unlike many of BitFlow's previous models of frame grabbers, the Axion family does not swap firmware on the fly (this is similar to the Cyton). The Axion is shipped with firmware that supports the latest Camera Link Specification and has been tested with all known cameras at the time of the release. However, new features may be added and anomalies will be corrected from time to time. These updates will take the form of a new firmware file (*.fsh). You may receive an updated firmware file as part of support issue, or a new firmware release may be part of a new SDK. In general, it is best to update the firmware on your board whenever you upgrade to a major new version of the SDK.

To update the board's firmware, type to the following command in a console window:

FWdownload

follow the instruction of the download program.

Note: After the firmware download process is complete, you must power down your computer in order for the new firmware to become active.

1.4 Axion Camera Configuration Files

The Axion is the second member of BitFlow's Gen 2 family. These frame grabber all use an XML based camera configuration file. This differs from previous models of BitFlow's frame grabbers that have all used a binary proprietary file format (which mean they could only be edited using BitFlow's tools). The file format uses the extension "BFML" but is actually an XML file with an XML compliant schema. The schema file is installed automatically and is called "BFML.xsd".

BFML files can be edited in any text editor. User's familiar with XML will understand the format right away. Users not familiar with XML files should not have too much trouble editing the files, but the XML file format is used everyw and t are many resources available for learning the format. A dedicated XML file editor can also be used, this can sometimes simplify editing when used in association with a schema file.

The BFML file format is documented on BitFlow's website. Please see the downloads page for a link to the BFML documentation.

Note: The tools used to edit previous BitFlow camera configuration files (CamEd, CamVert) can not be used to edit BFML files. We are working on a dedicated BFML editor which should be available in a future SDK release.

1.4.1 BFML Camera File Modes

Previous BitFlow camera configuration file only supported a single mode of camera operation. In order to support multiple (e.g. free-running, one-shot, triggered, encoder, etc.) modes for a given camera file, multiple files were required, one for each mode. The BFML file format can contain an unlimited number of camera modes. This makes things much simpler since only one file is needed for each model of camera. Then the different modes of operation are contained within that one BFML file.

Switching between camera modes is easy, this can be done via SysReg, with each of the camera modes are enumerated, and the user can pick which mode they want to use. The modes are also available from the API. There is a function to enumerate the modes and s function to switch modes on the fly.

Customer can create their own modes as they see fit. They can simple copy an existing mode and change to suite t needs. The new mode should have a new name (also the comment should be updated). Once this file is save, the mode will be available in SysReg as well as from the API.

1.5 The Axion Models

There are five models of the Axion-CL. Table 1-1 illustrates the capabilities of each model.

Table 1-1 The Axion Models

Capability	AXN-PC2-1xE	AXN-PC2-2xE	AXN-PC2-1xB	AXN-PC2-2xB	AXN-PC2-4xB
Number of Base CL cameras supported	1	2	1	2	4
Number of Medium CL cameras supported	1	2	0	0	0
Number of Full CL cameras supported	1	2	0	0	0
Number of 80-bit CL cameras supported	1	2	0	0	0
Number of Virtual Frame Grabbers supported	1	2	1	2	4
Number of independent trigger inputs	1	2	1	2	4
Number of independent encoder inputs	1	2	1	2	4
Number of PCI configurations (devices)	1	2	1	2	4
Maximum DMA bandwidth	1.75 GB/S	1.75 GB/S	1.75 GB/S	1.75 GB/S	1.75 GB/S

The StreamSync Acquisition Engine

Chapter 2

2.1 Introduction

The StreamSync system consists of an Acquisition Engine and a Buffer Manager. The StreamSync system was first released on the Cyton-CXP and is a departure from previous BitFlow frame grabbers. Subsequently it was ported to the Axion, and will be used on all new frame grabbers moving forward. The StreamSync system is a start-from-scratch complete redesign of the acquisition and DMA parts of a frame grabber. BitFlow used its years of experience in this area to design a next generation, super efficient capture system.

Currently the StreamSync engine is used on the Aon-CXP, the Axion-CL, Claxon-CXP and the Cyton-CXP.

From a software perspective, the StreamSync system is compatible with the previous BitFlow products. However, digging deeper, these new system have a lot more power and flexibility. These new features will be described in the following sections.

The StreamSync system has many improvements over previous systems. The main improvements are:

- Efficient support for variable sized images with fast context switches between frames
- Per frame control of acquisition properties (AOI specifically)
- Hardware control of image sequencing
- Enhanced debug capabilities
- Efficient support for on-demand buffer allocation (GenICam model)
- Gracefully recovery from dropped packets (either on the input side or the DMA side)

This chapter describes the StreamSync Acquisition Engine while the next chapter describes the StreamSync Buffer Manager.

2.2 The StreamSync Acquisition Engine World

We are used to the concept that images have an X and a Y dimension. The Acquisition Engine expands on this concept by adding two further dimension Z and V. The Z dimension controls a sequence of frames or "Volume" of frames. The V dimension controls a sequence of volumes, or "Hypervolume". Figure 2-1 illustrates these concepts.

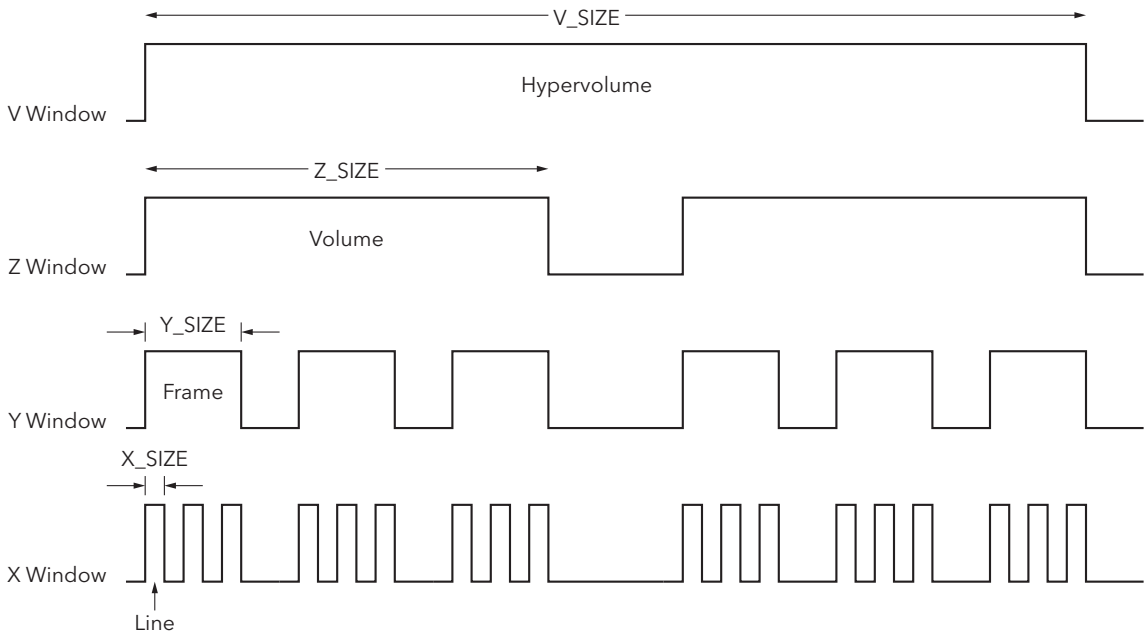

Figure 2-1 StreamSync Acquisition Engine Dimensions

The size of the X window, i.e. the number of pixels per line, is controlled by the X_SIZE register. The size of the Y window, i.e. number of lines per frame, is controlled by the Y_SIZE register. The size of the Z window, i.e. number of frames per volume, is controlled by the Z_SIZE register. Finally, the size of the V window, i.e. number of volumes to acquire, is controlled by the V_SIZE register. Note that the size of the Y window and the Z window can be dynamically controlled by external triggers, see below for more details.

2.2.1 Controlling the StreamSync Acquisition Engine

Acquisition of images is controlled by the AE_RUN_LEVEL register. The run level controls the conditions under which the Acquisition Engine will start or stop acquiring image data. Acquisition can be idle, which means nothing will be acquired, or it can be running, which means data will be acquired when the engine is inside the V, Z,Y and X windows. There are various conditions which control whether the engine is inside or outside of these windows.

Acquisition can be aborted on any X, Y, Z, or V boundary. The choice of boundary depends on whether one wants to abort immediate, which can cause acquisition of incomplete frames, or one wants to stop at the end of the line/frame/volume, which provides a more graceful end to acquisition.

It's easiest to think of the Acquisition Engine level in terms of a state machine. When a window is "opened" the run level moves down to the next state below. When a window "closes", the run level moves to the state above. If the window at the top level closes, the run level goes to idle. Figure 2-2 illustrates this type of state machine:

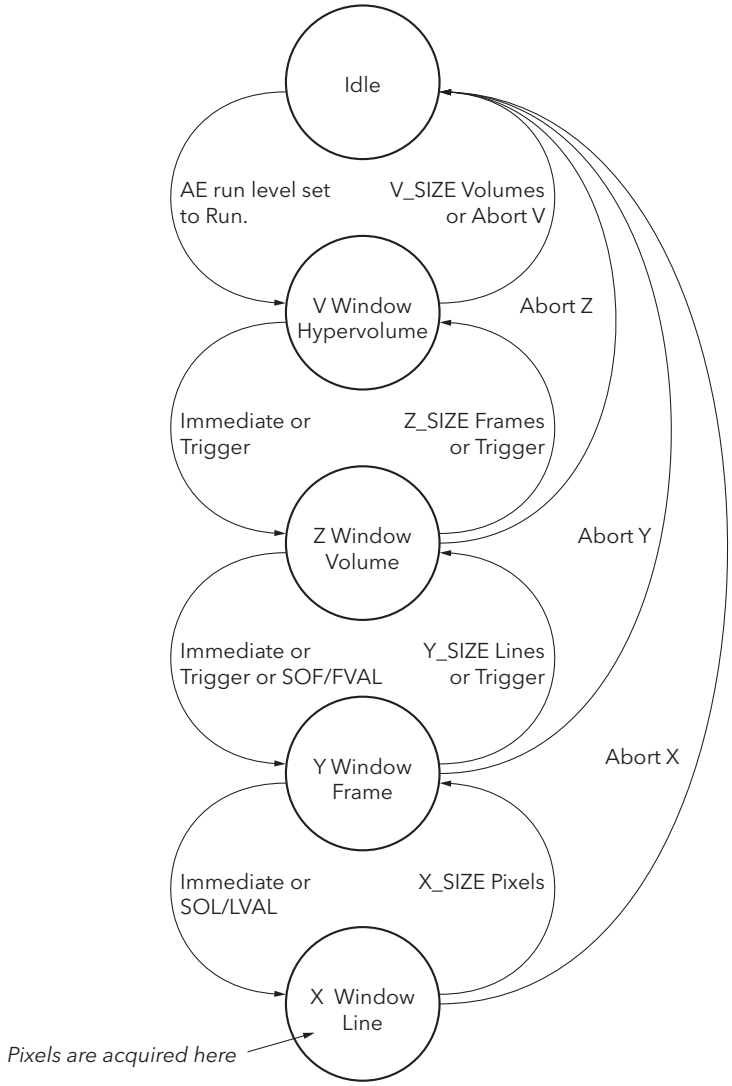

Figure 2-2 Acquisition Engine Run Level

The action that causes a window to be opened or closed depends on the type of window. Some windows can be opened in more than one way. For example the Y window can be opened when a Start Of Frame (SOF) packet (or FVAL from a CL camera) is sent from the camera, or it can be opened by a trigger (all SOF packets are ignored

until the trigger condition is met) or it can just be opened immediately, as soon the Acquisition Engine level is inside the X window (i.e. the stat above). Table 2-1 enumerates all of these conditions..

Table 2-1 Open Close Conditions

Window	Open	Close
V	AE run level set to Run	Abort V, V_SIZE volumes
Z	Trigger, Immediate	Trigger, Abort Z, Z_SIZE frame
Y	Trigger, SOF/FVAL, Immediate	Trigger, Abort Y, Y_SIZE lines
X	SOL/LVAL, Immediate	Abort X, X_SIZE pixels

2.2.2 Observing the StreamSync Acquisition Engine

The state of the Acquisition Engine can be observed at any time. The register AE_LEVEL indicates the current run level of the Acquisition Engine. In other words, this register returns the current state as shown in Figure 2-2. While this is not very useful in a free-running situation, as the value will be changing constantly, it can be very helpful debugging if the system gets stuck (e.g. waiting for a trigger).

2.2.3 Synchronizing the StreamSync Acquisition Engine With a CXP Camera

Normally acquisition is synchronized with camera by special CXP header packets called Start Of Frame (SOF) and Start Of Line (SOL). The Acquisition Engine will synchronize its Y window (frame) with the SOF and X window with the SOL. This means that all packets from the camera will be dropped until the SOF is seen (causing the Acquisition Engine to open the Y window), and then packets are further dropped until SOL must be seen (opening the X window). Each line requires an SOL packet. This process keeps the Acquisition Engine synchronized to the camera even if packets are dropped. This functionality can be enable/disable by the ZSYNC and YSYNC bits.

2.2.4 Synchronizing the StreamSync Acquisition Engine With a Camera Link Camera

Normally acquisition is synchronized with camera by FVAL (Frame VALid) and the LVAL (Line VALid). The Acquisition Engine will synchronize its Y window (frame) with the FVAL and X window (line) with the LVAL. This means that all pixels from the camera will be dropped until the FVAL is seen (causing the Acquisition Engine to open the Y window), and then packets are further dropped until LVAL must be seen (opening the X window). Each line requires an LVAL edge. This process keeps the Acquisition Engine synchronized to the camera even if packets are dropped. This functionality can be enable/disable by the ZSYNC and YSYNC bits.

Note: For line scan camera, FVAL is not used and the Y Window will open either immediately in free-run mode, or based on a trigger eddge.

2.2.5 Regions Of Interest (ROI) with the StreamSync Acquisition Engine.

The Acquisition Engine support capturing a subwindow or ROI of the image that the camera is putting out. The Y_SIZE and X_SIZE registers control how many lines and pixels are acquired per frame, regardless of the actual frames size coming out of the camera. Further there are Y_OFFS and X_OFFS registers which can locate the subwindow anywhere inside of the camera's frame. These concepts are show in Figure 2-3.

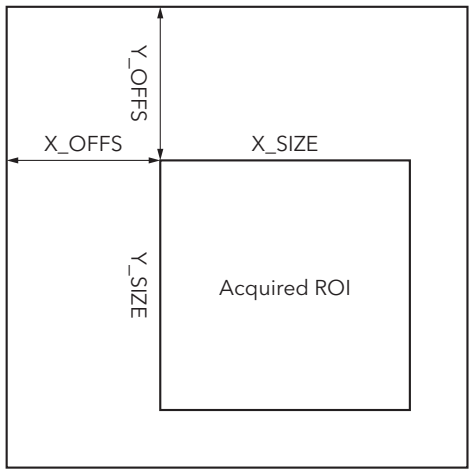

Figure 2-3 Acquisition Engine ROI

Similarly there is a Z_OFFS register which if non-zero can cause the board to discard a certain number of frames before starting an acquisition of a sequence. This concept is illustrated in Figure 2-4.

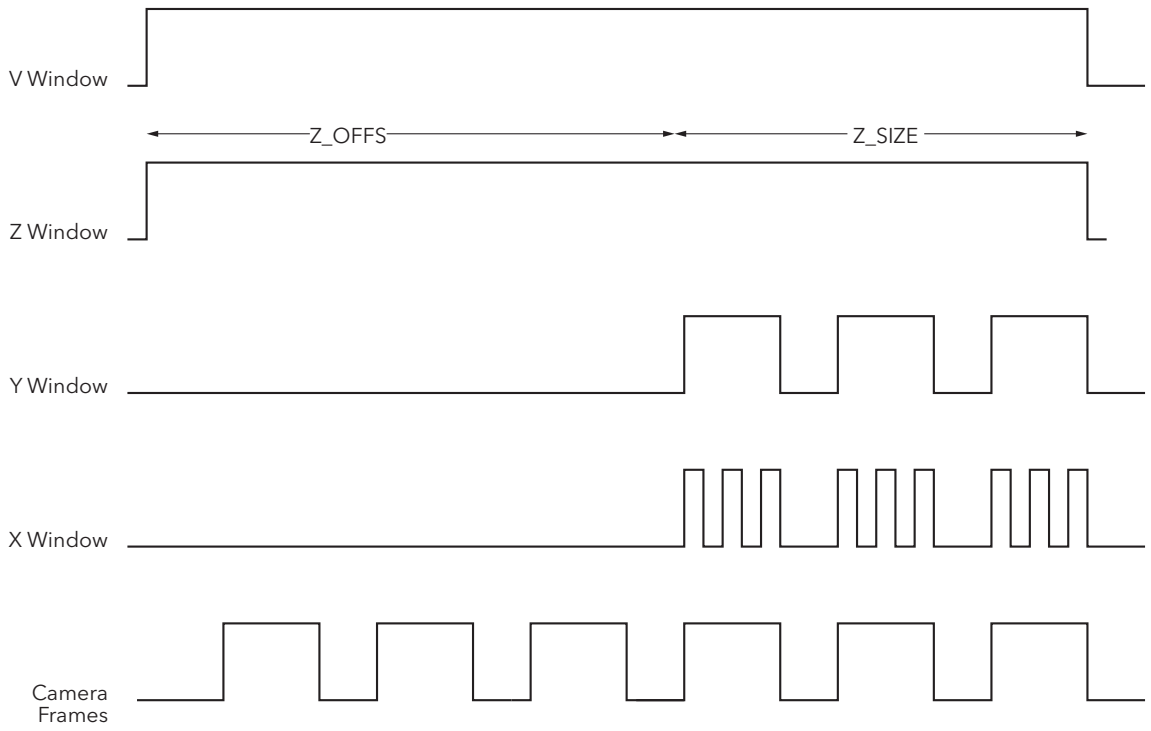

Figure 2-4 Z_OFFS Illustration

2.3 Window Interrupts

Interrupts at the start and end of each major window are available for use by host processes. The X, Y and Z windows (line, frame volume) all can provide interrupts. There are no interrupts for the V Window (hypervolume). Figure 2-5 Show the relationship between the interrupts and the acquisition Windows.

Note: The labels in italics in Figure 2-5 are are the actual interrupt names that can be used with the BitFlow SDK function calls.

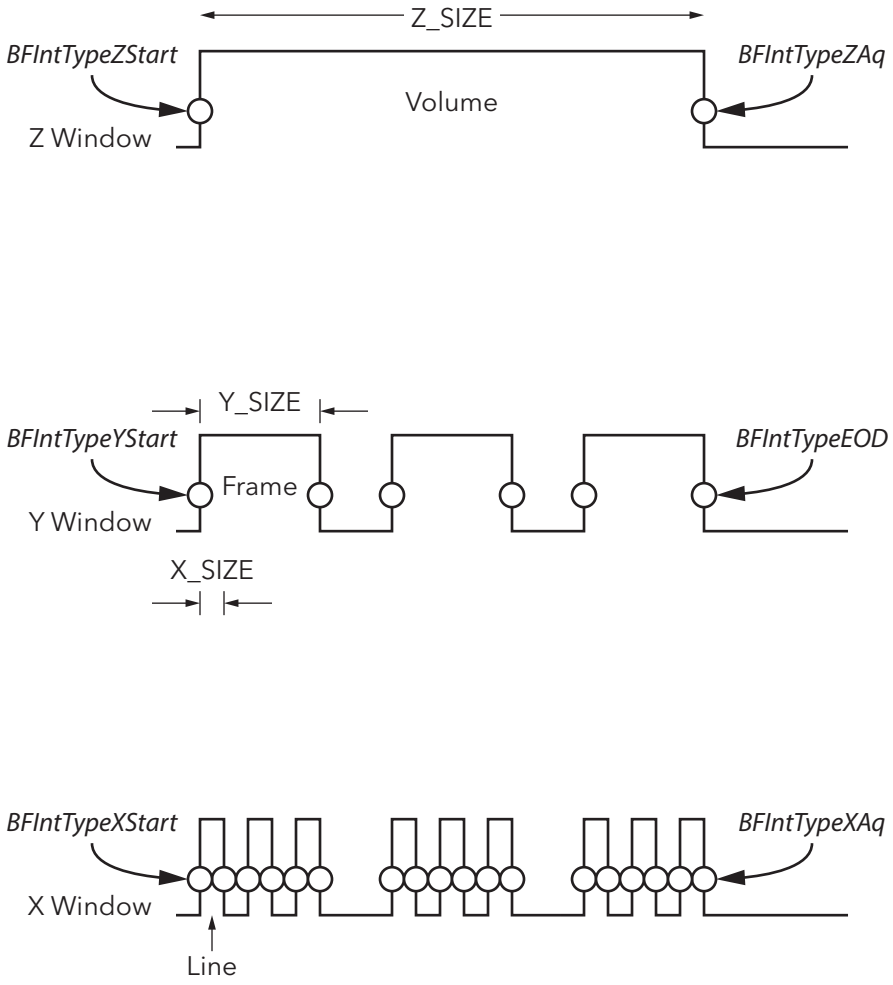

Figure 2-5 X/Y/Z Interrupts

2.4 AE State Machine

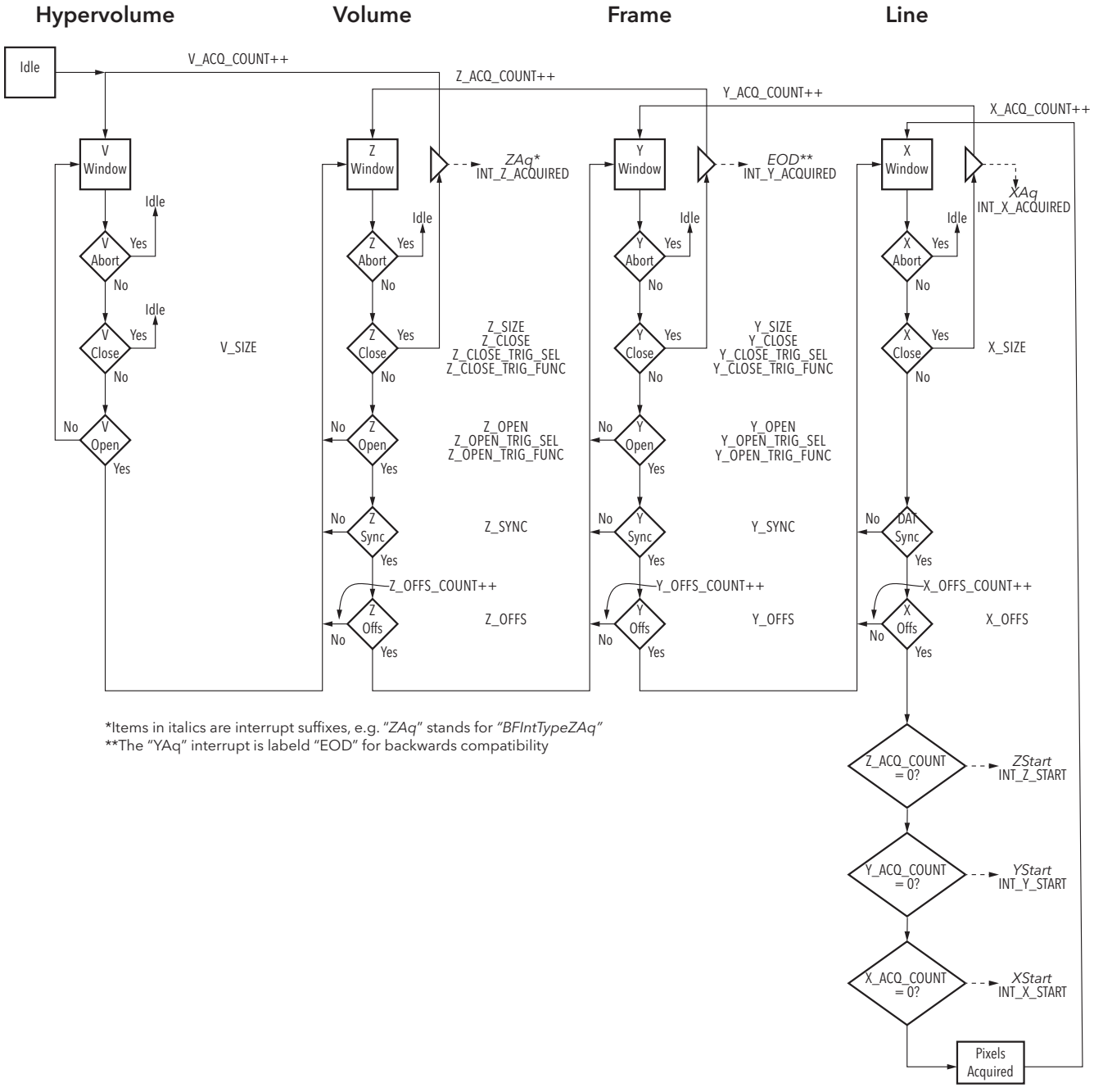

Figure 2-6 AE State Machine

2.5 Triggering the StreamSync Acquisition Engine

One of the areas where the power of the Acquisition Engine is really seen is with regards to triggering. There are many more ways to use triggers in the Acquisition Engine. Primarily triggers can be used to “open” a window and/or to “close” a window. For example, a trigger could be used to start the acquisition of each frame and/or end the acquisition of each frame.

Further, a trigger could be use to start the acquisition of each volume (sequence of frames) and/or end the acquisition of a volume. Further flexibility comes from the fact that the source for each event (i.e. open or close) can be different or the same. This means a frame could be started with one trigger or ended with another, or the frame could start on the rising edge and end on the falling edge of the same trigger. Please refer to Figure 2-2 for more information on how a trigger can be used to change the state of the Acquisition Engine.

2.6 Comparing the StreamSync Acquisition Engine to Other BitFlow products

While the Acquisition Engine might seem very complex, it is actually quite simple to use and has considerably more power than previous acquisition engines used on all previous BitFlow frame grabbers. From a software point of view, the BitFlow API hides the differences between the traditional acquisition systems and the newer Acquisition Engine. However, for users that desire more flexibility and are willing to do some lower level coding, the Acquisition Engine can handle almost any acquisition scenario.

For users who were already doing some lower level programming using other BitFlow products, it's helpful to see how this new system relates to the tradition acquisition engine.Table 2-2 shows some examples of the traditional and the new system.

Table 2-2 Comparing Traditional and New Acquisition Systems

Traditional Command	X_SIZE	Y_SIZE	Z_SIZE	V_SIZE	AE_RUN_ LEVEL
Snap	Camera Width	Camera Height	1	1	Run
Grab	Camera Width	Camera Height	1	0xffff	Run
Grab N frames	Camera Width	Camera Height	N	1	Run
Freeze					Abort Z
Abort					Abort X

2.7 AE_CON

Bit	Name
0	AE_RUN_LEVEL
1	AE_RUN_LEVEL
2	AE_RUN_LEVEL
3	AE_RUN_LEVEL
4	Reserved
5	Reserved
6	Reserved
7	Reserved
8	Reserved
9	CLR_ACQ_COUNT
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

AE_RUN_LEVEL R/W, AE_CON[3..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the main control for starting/aborting acquisition. Writing this register changes the current run level. Reading this register returns the current run level command (not the current status). The abort run levels exit acquisition on a clean boundary. V exits on a volume boundary, Z on a frame boundary, Y on a line boundary, X on a 128-byte data boundary.

AE_RUN_LEVEL	Meaning
0 (0000b)	System is idle
1 (0001b)	Run - start running (i.e. acquiring)
2 (0010b)	Abort V - stop at the end of the next volume
3 (0011b)	Abort Z - stop at the end of the next frame
4 (0100b)	Abort Y - stop at the end of the next line
5 (0101b)	Abort X - stop at the end of the next 128-byte block

CLR_ACQ_COUNT RO, AE_CON[9], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Write a 1 to this bitfield will reset to 0 the following registers V_ACQ_COUNT, Z_ACQ_COUNT, Y_ACQ_COUNT and X_ACQ_COUNT.

2.8 AE_STATUS

Bit	Name
0	AE_STATE
1	AE_STATE
2	AE_STATE
3	Reserved
4	AE_FIFO_OVERFLOW
5	Reserved
6	Reserved
7	Reserved
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

AE_STATE RO, AE_STATUS[2..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register indicates the current run level of the acquisition engine. The following table shows the meanings of each state.

AE_STATE	Meaning
0 (000b)	Idle - System is idle
1 (001b)	System is inside the V window
2 (010b)	System is inside the Z window
3 (011b)	System is inside the Y window
4 (100b)	System is inside the X window

AE_FIFO_ OVERFLOW RO, AE_STATUS[4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

If this bit is 1, the FIFO between acquisition engine and packet generation overflowed. The acquisition engine will abort.

2.9 AE_STREAM_SEL

Bit	Name
0	STREAM_SEL
1	STREAM_SEL
2	STREAM_SEL
3	STREAM_SEL
4	STREAM_SEL
5	STREAM_SEL
6	STREAM_SEL
7	STREAM_SEL
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	USE_SYNTHETIC_FRAME

STREAM_SEL R/W, AE_STREAM_SEL[7..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Program this register to the stream aggregator that this acquisition engine should get its data from. Currently only the values 0 to 3 are supported. Generally this register should be programmed to correspond to the VFG number that is being used to access the acquisition engine. For example, for VFG1 set this register to 1.

USE_SYNTHETIC_FRAME R/W, AE_STREAM_SEL[31], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Use the Synthetic Frame generator instead of the camera.

2.10 V_WIN_DIM

Bit	Name
0	V_SIZE
1	V_SIZE
2	V_SIZE
3	V_SIZE
4	V_SIZE
5	V_SIZE
6	V_SIZE
7	V_SIZE
8	V_SIZE
9	V_SIZE
10	V_SIZE
11	V_SIZE
12	V_SIZE
13	V_SIZE
14	V_SIZE
15	V_SIZE
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

V_SIZE R/W, V_WIN_DIM[15..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register defines size of the V window, that is, the number of volumes to acquire. A value of 0XFFFF means infinite. When set to infinite, the acquisition engine can be stopped by writing AE_RUN_LEVEL.

The most common setting for this field is either 1 or 0xFFFF.

This register is writable only when AE_STATE is 0 (idle). Writes to this field will be ignored if AE_STATE is not 0.

2.11 Z_WIN_CON

Bit	Name
0	Z_CLOSE_TRIG_FUNC
1	Z_CLOSE_TRIG_FUNC
2	Z_CLOSE_TRIG_FUNC
3	Z_CLOSE_TRIG_FUNC
4	Z_CLOSE_TRIG_SEL
5	Z_CLOSE_TRIG_SEL
6	Z_CLOSE_TRIG_SEL
7	Z_CLOSE_TRIG_SEL
8	Z_CLOSE
9	Z_CLOSE
10	Z_CLOSE
11	Z_CLOSE
12	Z_OPEN_TRIG_FUNC
13	Z_OPEN_TRIG_FUNC
14	Z_OPEN_TRIG_FUNC
15	Z_OPEN_TRIG_FUNC
16	Z_OPEN_TRIG_SEL
17	Z_OPEN_TRIG_SEL
18	Z_OPEN_TRIG_SEL
19	Z_OPEN_TRIG_SEL
20	Z_OPEN
21	Z_OPEN
22	Z_OPEN
23	Z_OPEN
24	Z_SYNC
25	Z_SYNC
26	Z_SYNC
27	Z_SYNC
28	Reserved
29	Reserved
30	Reserved
31	Reserved

Z_CLOSE_TRIG_FUNC R/W, Z_WIN_CON[3..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register determines which trigger change (if any) will end the Z window.

Z_CLOSE_TRIG_FUNC	Meaning
0 (0000b)	Rising edge of trigger
1 (0001b)	Falling edge of trigger
2 (0010b)	Trigger is high
3 (0011b)	Trigger is low
4 (0100b)	Either edge of trigger

Z_CLOSE_TRIG_SEL R/W, Z_WIN_CON[7..4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects which trigger will control the end the Z window.

Z_CLOSE_TRIG_SEL	Meaning
0 (0000b)	The selected trigger (VFGx_TRIG_SEL)
1 (0001b)	The selected encoder A (VFGx_ENCA_SEL)
2 (0010b)	The selected encoder B (VFGx_ENCB_SEL)
3 (0011b)	The selected encoder divider (VFGx_ENCDIV_SEL)
4 (0100b)	The selected encoder quad (VFGx_ENCQ_SEL)

Z_CLOSE R/W, Z_WIN_CON[11..8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This field specifies how the Z window closes. Possible values are: 0 - size mode, 1 - trigger mode.

If size mode is specified, the acquisition engine waits for Z_SIZE number of complete frames, it then closes the Z window and then checks to see if t are more volumes to acquire.

If trigger mode is specified, the trigger is selected by Z_CLOSE_TRIG_SEL and the conditioning function is specified by Z_CLOSE_TRIG_FUNC. The acquisition engine waits for the trigger condition to be satisfied, then continues acquiring to the next frame boundary, it then closes the Z window and then checks to see if t are more volumes to acquire.

Z_OPEN_TRIG_FUNC R/W, Z_WIN_CON[15..12], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register determines which trigger change (if any) will start Z window.

Z_OPEN_TRIG_FUNC	Meaning
0 (0000b)	Rising edge of trigger
1 (0001b)	Falling edge of trigger
2 (0010b)	Trigger is high
3 (0011b)	Trigger is low
4 (0100b)	Either edge of trigger

Z_OPEN_TRIG_SEL R/W, Z_WIN_CON[19..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects which trigger will control the start Z window.

Z_OPEN_TRIG_SEL	Meaning
0 (0000b)	The selected trigger (VFGx_TRIG_SEL)
1 (0001b)	The selected encoder A (VFGx_ENCA_SEL)
2 (0010b)	The selected encoder B (VFGx_ENCB_SEL)
3 (0011b)	The selected encoder divider (VFGx_ENCDIV_SEL)
4 (0100b)	The selected encoder quad (VFGx_ENCQ_SEL)

Z_OPEN R/W, Z_WIN_CON[23..20], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This field specifies how the Z window starts. Possible values are: 0 - immediate mode, 1 - trigger mode.

If immediate mode is specified, no trigger synchronization is required. The acquisition engine waits for any frame sync requirements, opens the Z window, then starts the setup of the Y window.

If trigger mode is specified, the trigger is selected by Z_OPEN_TRIG_SEL and the conditioning function is specified by Z_OPEN_TRIG_FUNC. The acquisition engine waits for the trigger condition to be satisfied, opens the Z window, then starts the setup of the Y window.

Z_SYNC R/W, Z_WIN_CON[27..24], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This field enforces the data-synchronization of streaming video to the acquisition engine for each individual frame in the z window. The following table shows explains this field.

Z_SYNC	Meaning
0	No synchronization. All streaming packets received after the Z Window are open will be acquired as part of the current frame.
1	Start Of Frame (SOF) synchronization. All pixels received before the SOF will be ignored. This conditions is enforced for each frame in the Z window. On the Axion, SOF is generated by the FVAL signal coming from the camera.

2.12 Z_WIN_DIM

Bit	Name
0	Z_SIZE
1	Z_SIZE
2	Z_SIZE
3	Z_SIZE
4	Z_SIZE
5	Z_SIZE
6	Z_SIZE
7	Z_SIZE
8	Z_SIZE
9	Z_SIZE
10	Z_SIZE
11	Z_SIZE
12	Z_SIZE
13	Z_SIZE
14	Z_SIZE
15	Z_SIZE
16	Z_OFFS
17	Z_OFFS
18	Z_OFFS
19	Z_OFFS
20	Z_OFFS
21	Z_OFFS
22	Z_OFFS
23	Z_OFFS
24	Z_OFFS
25	Z_OFFS
26	Z_OFFS
27	Z_OFFS
28	Z_OFFS
29	Z_OFFS
30	Z_OFFS
31	Z_OFFS

Z_SIZE R/W, Z_WIN_DIM[15..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Number of frames (Y windows) to acquire per sequence (Z windows). The acquisition of frames will only start after Z_OFFS frames have been skipped after the Z window is opened. Note this register is actually 24 bits, the upper 8 MSB are located in the Z_SIZE_MSB register.

Z_OFFS R/W, Z_WIN_DIM[31..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The number of frames (Y windows) to skip before starting acquisition after the Z window has been opened. Note this register is actually 24 bits, the upper 8 MSB are located in the Z_OFFS_MSB register.

2.13 Z_WIN_DIM_EXT

Bit	Name
0	Z_SIZE_MSB
1	Z_SIZE_MSB
2	Z_SIZE_MSB
3	Z_SIZE_MSB
4	Z_SIZE_MSB
5	Z_SIZE_MSB
6	Z_SIZE_MSB
7	Z_SIZE_MSB
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Z_OFFS_MSB
17	Z_OFFS_MSB
18	Z_OFFS_MSB
19	Z_OFFS_MSB
20	Z_OFFS_MSB
21	Z_OFFS_MSB
22	Z_OFFS_MSB
23	Z_OFFS_MSB
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

Z_SIZE_MSB R/W, Z_WIN_DIM_EXT[7..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Extends the Z_SIZE register by 8 more bits.

Z_OFFS_MSB R/W, Z_WIN_DIM_EXT[23..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Extends the Z_OFFS register by 8 more bits.

2.14 Y_INT_DEC

Bit	Name
0	Y_INT_DEC_COUNT
1	Y_INT_DEC_COUNT
2	Y_INT_DEC_COUNT
3	Y_INT_DEC_COUNT
4	Y_INT_DEC_COUNT
5	Y_INT_DEC_COUNT
6	Y_INT_DEC_COUNT
7	Y_INT_DEC_COUNT
8	Y_INT_DEC_COUNT
9	Y_INT_DEC_COUNT
10	Y_INT_DEC_COUNT
11	Y_INT_DEC_COUNT
12	Y_INT_DEC_COUNT
13	Y_INT_DEC_COUNT
14	Y_INT_DEC_COUNT
15	Y_INT_DEC_COUNT
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Y_INT_DEC_RST
29	Y_INT_DEC_RST
30	Y_INT_DEC_MODE
31	Y_INT_DEC_MODE

Y_INT_DEC_COUNT R/W, Y_INT_DEC[15..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

When Y interrupt decimate mode is enable, the register determines the decimation amount. In other words, Y_INT_DEC_COUNT interrupts must occur before the board emits a real interrupt.

Y_INT_DEC_RST R/W, Y_INT_DEC[29..28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Resets the Y interrupt decimation counter to 0.

Y_INT_DEC_MODE R/W, Y_INT_DEC[29..28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register controls how Y interrupts are decimated.

Y_DEC_MODE	Meaning
0 (0000b)	Y interrupts are not decimated
1 (0001b)	Y interrupts are decimated by Y_INT_DEC_COUNT
2 (0010b)	reserved
3 (0011b)	reserved

2.15 Y_WIN_CON

Bit	Name
0	Y_CLOSE_TRIG_FUNC
1	Y_CLOSE_TRIG_FUNC
2	Y_CLOSE_TRIG_FUNC
3	Y_CLOSE_TRIG_FUNC
4	Y_CLOSE_TRIG_SEL
5	Y_CLOSE_TRIG_SEL
6	Y_CLOSE_TRIG_SEL
7	Y_CLOSE_TRIG_SEL
8	Y_CLOSE
9	Y_CLOSE
10	Y_CLOSE
11	Y_CLOSE
12	Y_OPEN_TRIG_FUNC
13	Y_OPEN_TRIG_FUNC
14	Y_OPEN_TRIG_FUNC
15	Y_OPEN_TRIG_FUNC
16	Y_OPEN_TRIG_SEL
17	Y_OPEN_TRIG_SEL
18	Y_OPEN_TRIG_SEL
19	Y_OPEN_TRIG_SEL
20	Y_OPEN
21	Y_OPEN
22	Y_OPEN
23	Y_OPEN
24	Y_SYNC
25	Y_SYNC
26	Y_SYNC
27	Y_SYNC
28	Reserved
29	Reserved
30	Reserved
31	Reserved

Y_CLOSE_TRIG_FUNC R/W, Y_WIN_CON[3..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register determines which trigger change (if any) will end the Y window.

Y_CLOSE_TRIG_FUNC	Meaning
0 (0000b)	Rising edge of trigger
1 (0001b)	Falling edge of trigger
2 (0010b)	Trigger is high
3 (0011b)	Trigger is low
4 (0100b)	Either edge of trigger

Y_CLOSE_TRIG_SEL R/W, Y_WIN_CON[7..4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects which trigger will control the end the Y window.

Y_CLOSE_TRIG_SEL	Meaning
0 (0000b)	The selected trigger (VFGx_TRIG_SEL)
1 (0001b)	The selected encoder A (VFGx_ENCA_SEL)
2 (0010b)	The selected encoder B (VFGx_ENCB_SEL)
3 (0011b)	The selected encoder divider (VFGx_ENCDIV_SEL)
4 (0100b)	The selected encoder quad (VFGx_ENCQ_SEL)
5 (0101b)	The falling edge of FVAL will close the window (Axion only)

Y_CLOSE R/W, Y_WIN_CON[11..8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This field specifies how the Y window closes. Possible values are: 0 - size mode, 1 - trigger mode, 2 - size or trigger mode.

If Y_CLOSE = 0, the acquisition engine waits for Y_SIZE_SIZE number of complete lines, it then closes the Y window and then checks to see if there are more frames to acquire.

If Y_CLOSE = 1, the trigger is selected by Y_CLOSE_TRIG_SEL and the conditioning function is specified by Y_CLOSE_TRIG_FUNC. The acquisition engine waits for the trigger condition to be satisfied, then continues acquiring to the next line boundary, it then closes the Y window and then checks to see if there are more frames to acquire.

If Y_CLOSE = 2, either condition above will close the window. Whichever conditions comes first will close the window (end the frame). This mode is used to acquire variable sized images where the trigger controls the frame size. Usually the rising edge of

trigger opens the window (starts the frame) lines are acquired until either the trigger goes low, or Y_SIZE lines have been acquired (i.e. the maximum frame size has been reached).

Y_OPEN_TRIG_FUNC R/W, Y_WIN_CON[15..12], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register determines which trigger change (if any) will start Y window.

Y_OPEN_TRIG_FUNC	Meaning
0 (0000b)	Rising edge of trigger
1 (0001b)	Falling edge of trigger
2 (0010b)	Trigger is high
3 (0011b)	Trigger is low
4 (0100b)	Either edge of trigger

Y_OPEN_TRIG_SEL R/W, Y_WIN_CON[19..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects which trigger will control the start Y window.

Y_OPEN_TRIG_SEL	Meaning
0 (0000b)	The selected trigger (VFGx_TRIG_SEL)
1 (0001b)	The selected encoder A (VFGx_ENCA_SEL)
2 (0010b)	The selected encoder B (VFGx_ENCB_SEL)
3 (0011b)	The selected encoder divider (VFGx_ENCDIV_SEL)
4 (0100b)	The selected encoder quad (VFGx_ENCQ_SEL)

Y_OPEN R/W, Y_WIN_CON[23..20], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This field specifies how the Y window starts. Possible values are: 0 - immediate mode, 1 - trigger mode.

If immediate mode is specified, no trigger synchronization is required. The acquisition engine waits for any line sync requirements, opens the Y window, then starts the setup of the X window.

If trigger mode is specified, the trigger is selected by Y_OPEN_TRIG_SEL and the conditioning function is specified by Y_OPEN_TRIG_FUNC. The acquisition engine waits for the trigger condition to be satisfied, opens the Y window, then starts the setup of the X window.

Y_SYNC R/W, Y_WIN_CON[27..24], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This field enforces the data-synchronization of streaming video to the acquisition engine for each individual line in the y window. The following table explains this field.

Y_SYNC	Meaning
0	No synchronization. All streaming packets received after the Y Window are open will be acquired as part of the current line.
1	Start Of Line (SOL) synchronization. All pixels received before the SOL will be ignored. This conditions is enforced for each line in the Y window. On the Axion, SOL is generated from the LVAL signal.

2.16 Y_WIN_DIM

Bit	Name
0	Y_SIZE
1	Y_SIZE
2	Y_SIZE
3	Y_SIZE
4	Y_SIZE
5	Y_SIZE
6	Y_SIZE
7	Y_SIZE
8	Y_SIZE
9	Y_SIZE
10	Y_SIZE
11	Y_SIZE
12	Y_SIZE
13	Y_SIZE
14	Y_SIZE
15	Y_SIZE
16	Y_OFFS
17	Y_OFFS
18	Y_OFFS
19	Y_OFFS
20	Y_OFFS
21	Y_OFFS
22	Y_OFFS
23	Y_OFFS
24	Y_OFFS
25	Y_OFFS
26	Y_OFFS
27	Y_OFFS
28	Y_OFFS
29	Y_OFFS
30	Y_OFFS
31	Y_OFFS

Y_SIZE R/W, Y_WIN_DIM[15..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Number of lines per frame (Y window) to acquire. This number is only acquired after the Y window is opened and after Y_OFFS lines have been skipped.

Y_OFFS R/W, Y_WIN_DIM[31..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Number of lines to skip before starting the acquisition of lines (after the Y windows is opened).

2.17 Y_WIN_DIM_EXT

Bit	Name
0	Y_SIZE_MSB
1	Y_SIZE_MSB
2	Y_SIZE_MSB
3	Y_SIZE_MSB
4	Y_SIZE_MSB
5	Y_SIZE_MSB
6	Y_SIZE_MSB
7	Y_SIZE_MSB
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Y_OFFS_MSB
17	Y_OFFS_MSB
18	Y_OFFS_MSB
19	Y_OFFS_MSB
20	Y_OFFS_MSB
21	Y_OFFS_MSB
22	Y_OFFS_MSB
23	Y_OFFS_MSB
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

Y_SIZE_MSB R/W, Y_WIN_DIM_EXT[7..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Extends the Y_SIZE register by 8 more bits.

Y_OFFS_MSB R/W,Y_WIN_DIM_EXT[23..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Extends the Y_OFFS register by 8 more bits.

2.18 X_WIN_DIM

Bit	Name
0	X_SIZE
1	X_SIZE
2	X_SIZE
3	X_SIZE
4	X_SIZE
5	X_SIZE
6	X_SIZE
7	X_SIZE
8	X_SIZE
9	X_SIZE
10	X_SIZE
11	X_SIZE
12	X_SIZE
13	X_SIZE
14	X_SIZE
15	X_SIZE
16	X_OFFS
17	X_OFFS
18	X_OFFS
19	X_OFFS
20	X_OFFS
21	X_OFFS
22	X_OFFS
23	X_OFFS
24	X_OFFS
25	X_OFFS
26	X_OFFS
27	X_OFFS
28	X_OFFS
29	X_OFFS
30	X_OFFS
31	X_OFFS

X_SIZE R/W, X_WIN_DIM[15..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Number of 16-byte data words to acquired per line (X window). This number is only acquired after the X window is opened and after X_OFFS words have been skipped.

X_OFFS R/W, X_WIN_DIM[31..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Number of 16-byte data words to skip per line (after the X window is opened).

2.19 X_WIN_DIM_EXT

Bit	Name
0	X_SIZE_MSB
1	X_SIZE_MSB
2	X_SIZE_MSB
3	X_SIZE_MSB
4	X_SIZE_MSB
5	X_SIZE_MSB
6	X_SIZE_MSB
7	X_SIZE_MSB
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	X_OFFS_MSB
17	X_OFFS_MSB
18	X_OFFS_MSB
19	X_OFFS_MSB
20	X_OFFS_MSB
21	X_OFFS_MSB
22	X_OFFS_MSB
23	X_OFFS_MSB
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

X_SIZE_MSB R/W, X_WIN_DIM_EXT[7..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Extends the X_SIZE register by 8 more bits.

X_OFFS_MSB R/W,X_WIN_DIM_EXT[23..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Extends the X_OFFS register by 8 more bits.

2.20 V_ACQUIRED

Bit	Name
0	V_ACQ_COUNT
1	V_ACQ_COUNT
2	V_ACQ_COUNT
3	V_ACQ_COUNT
4	V_ACQ_COUNT
5	V_ACQ_COUNT
6	V_ACQ_COUNT
7	V_ACQ_COUNT
8	V_ACQ_COUNT
9	V_ACQ_COUNT
10	V_ACQ_COUNT
11	V_ACQ_COUNT
12	V_ACQ_COUNT
13	V_ACQ_COUNT
14	V_ACQ_COUNT
15	V_ACQ_COUNT
16	V_ACQ_COUNT
17	V_ACQ_COUNT
18	V_ACQ_COUNT
19	V_ACQ_COUNT
20	V_ACQ_COUNT
21	V_ACQ_COUNT
22	V_ACQ_COUNT
23	V_ACQ_COUNT
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	V_ACQ_COUNT_CLR_MODE
29	V_ACQ_COUNT_CLR_MODE
30	V_ACQ_COUNT_UPD_MODE
31	V_ACQ_COUNT_UPD_MODE

V_ACQ_COUNT R/W, V_ACQUIRED[23..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Returns the total number of volumes (frame sequence) acquired since the last reset of this register. The behavior of this register when it reaches it maximum value depends on the register V_ACQ_COUNT_CLEAR_MODE. This register can be written to 0 by software at any time.

V_ACQ_COUNT_CLR_MODE R/W, V_ACQUIRED[29..28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Controls how the V_ACQ_COUNT register is cleared.

V_ACQ_COUNT_CLR_MODE	Meaning
0 (00b)	Clear count on the start of acquisition
1 (01b)	Clear count on the start of V Window
2 (10b)	Clear count on the start of Z Window
3 (11b)	Clear count on the start of Y Window

V_ACQ_COUNT_UPD_MODE R/W, V_ACQUIRED[31..30], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Controls when the V_ACQ_COUNT register is update.

V_ACQ_COUNT_UPD_MODE	Meaning
0 (00b)	Update continuously
1 (01b)	Update end of V Window
2 (10b)	Update end of Z Window
3 (11b)	Update end of Y Window

2.21 Z_ACQUIRED

Bit	Name
0	Z_ACQ_COUNT
1	Z_ACQ_COUNT
2	Z_ACQ_COUNT
3	Z_ACQ_COUNT
4	Z_ACQ_COUNT
5	Z_ACQ_COUNT
6	Z_ACQ_COUNT
7	Z_ACQ_COUNT
8	Z_ACQ_COUNT
9	Z_ACQ_COUNT
10	Z_ACQ_COUNT
11	Z_ACQ_COUNT
12	Z_ACQ_COUNT
13	Z_ACQ_COUNT
14	Z_ACQ_COUNT
15	Z_ACQ_COUNT
16	Z_ACQ_COUNT
17	Z_ACQ_COUNT
18	Z_ACQ_COUNT
19	Z_ACQ_COUNT
20	Z_ACQ_COUNT
21	Z_ACQ_COUNT
22	Z_ACQ_COUNT
23	Z_ACQ_COUNT
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Z_ACQ_COUNT_CLR_MODE
29	Z_ACQ_COUNT_CLR_MODE
30	Z_ACQ_COUNT_UPD_MODE
31	Z_ACQ_COUNT_UPD_MODE

Z_ACQ_COUNT R/W, Z_ACQUIRED[23..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Returns the total number of frames acquired since the last reset of this register. The behavior of this register when it reaches it maximum value depends on the register Z_ACQ_COUNT_CLEAR_MODE. This register can be written to 0 by software at any time.

Z_ACQ_COUNT_CLR_MODE R/W, Z_ACQUIRED[29..28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Controls how the Z_ACQ_COUNT register is cleared.

Z_ACQ_COUNT_CLR_MODE	Meaning
0 (00b)	Clear count on the start of acquisition
1 (01b)	Clear count on the start of V Window
2 (10b)	Clear count on the start of Z Window
3 (11b)	Clear count on the start of Y Window

Z_ACQ_COUNT_UPD_MODE R/W, Z_ACQUIRED[31..30], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Controls when the Z_ACQ_COUNT register is update.

Z_ACQ_COUNT_UPD_MODE	Meaning
0 (00b)	Update continuously
1 (01b)	Update end of V Window
2 (10b)	Update end of Z Window
3 (11b)	Update end of Y Window

2.22 Y_ACQUIRED

Bit	Name
0	Y_ACQ_COUNT
1	Y_ACQ_COUNT
2	Y_ACQ_COUNT
3	Y_ACQ_COUNT
4	Y_ACQ_COUNT
5	Y_ACQ_COUNT
6	Y_ACQ_COUNT
7	Y_ACQ_COUNT
8	Y_ACQ_COUNT
9	Y_ACQ_COUNT
10	Y_ACQ_COUNT
11	Y_ACQ_COUNT
12	Y_ACQ_COUNT
13	Y_ACQ_COUNT
14	Y_ACQ_COUNT
15	Y_ACQ_COUNT
16	Y_ACQ_COUNT
17	Y_ACQ_COUNT
18	Y_ACQ_COUNT
19	Y_ACQ_COUNT
20	Y_ACQ_COUNT
21	Y_ACQ_COUNT
22	Y_ACQ_COUNT
23	Y_ACQ_COUNT
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Y_ACQ_COUNT_CLR_MODE
29	Y_ACQ_COUNT_CLR_MODE
30	Y_ACQ_COUNT_UPD_MODE
31	Y_ACQ_COUNT_UPD_MODE

Y_ACQ_COUNT R/W, Y_ACQUIRED[23..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Returns the total number of lines acquired since the last reset of this register. The behavior of this register when it reaches it maximum value depends on the register Y_ACQ_COUNT_CLEAR_MODE. This register can be written to 0 by software at any time.

Y_ACQ_COUNT_CLR_MODE R/W, Y_ACQUIRED[29..28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Controls how the Y_ACQ_COUNT register is cleared.

Y_ACQ_COUNT_CLR_MODE	Meaning
0 (00b)	Clear count on the start of acquisition
1 (01b)	Clear count on the start of V Window
2 (10b)	Clear count on the start of Z Window
3 (11b)	Clear count on the start of Y Window

Y_ACQ_COUNT_UPD_MODE R/W, Y_ACQUIRED[31..30], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Controls when the Y_ACQ_COUNT register is update.

Y_ACQ_COUNT_UPD_MODE	Meaning
0 (00b)	Update continuously
1 (01b)	Update end of V Window
2 (10b)	Update end of Z Window
3 (11b)	Update end of Y Window

2.23 X_ACQUIRED

Bit	Name
0	X_ACQ_COUNT
1	X_ACQ_COUNT
2	X_ACQ_COUNT
3	X_ACQ_COUNT
4	X_ACQ_COUNT
5	X_ACQ_COUNT
6	X_ACQ_COUNT
7	X_ACQ_COUNT
8	X_ACQ_COUNT
9	X_ACQ_COUNT
10	X_ACQ_COUNT
11	X_ACQ_COUNT
12	X_ACQ_COUNT
13	X_ACQ_COUNT
14	X_ACQ_COUNT
15	X_ACQ_COUNT
16	X_ACQ_COUNT
17	X_ACQ_COUNT
18	X_ACQ_COUNT
19	X_ACQ_COUNT
20	X_ACQ_COUNT
21	X_ACQ_COUNT
22	X_ACQ_COUNT
23	X_ACQ_COUNT
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	X_ACQ_COUNT_CLR_MODE
29	X_ACQ_COUNT_CLR_MODE
30	X_ACQ_COUNT_UPD_MODE
31	X_ACQ_COUNT_UPD_MODE

X_ACQ_COUNT R/W, X_ACQUIRED[23..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Returns the total number of 16-byte words acquired since the last reset of this register. The behavior of this register when it reaches it maximum value depends on the register X_ACQ_COUNT_CLEAR_MODE. This register can be written to 0 by software at any time.

X_ACQ_COUNT_CLR_MODE R/W, X_ACQUIRED[29..28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Controls how the X_ACQ_COUNT register is cleared.

X_ACQ_COUNT_CLEAR_MODE	Meaning
0 (00b)	Clear count on the start of acquisition
1 (01b)	Clear count on the start of Z Window
2 (10b)	Clear count on the start of Y Window
3 (11b)	Clear count on the start of X Window

X_ACQ_COUNT_UPD_MODE R/W, X_ACQUIRED[31..30], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Controls when the X_ACQ_COUNT register is update.

X_ACQ_COUNT_UPD_MODE	Meaning
0 (00b)	Update continuously
1 (01b)	Update end of Z Window
2 (10b)	Update end of Y Window
3 (11b)	Update end of X Window

2.24 CON489

Bit	Name
0	INT_Y_ACQUIRED
1	INT_X_ACQUIRED
2	INT_Z_ACQUIRED
3	Reserved
4	INT_ENC_B
5	INT_ENC_A
6	INT_TRIG
7	INT_Y_START
8	INT_X_START
9	INT_Z_START
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	INT_BM_ERROR
27	INT_AE_LOSS_OF_SYNC
28	INT_PCIE_PKT_DROPPED
29	INT_Y_ACQUIRED_LEGACY
30	Reserved
31	Reserved

INT_Y_ACQUIRED R/W, CON489[0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Z window closed interrupt.

INT_X_ACQUIRED R/W, CON489[1], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Y window closed interrupt.

INT_Z_ACQUIRED R/W, CON489[2], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

V window closed interrupt..

INT_ENC_B R/W, CON489[4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Encoder B interrupt.

INT_ENC_A R/W, CON489[5], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Encoder A interrupt.

INT_TRIG R/W, CON489[6], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Trigger interrupt.

INT_Y_START R/W, CON489[7], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Start of Z Window interrupt.

INT_X_START R/W, CON489[8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Start of Y Window interrupt.

INT_Z_START R/W, CON489[9], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Start of V Window interrupt.

INT_BM_ERROR R/W, CON489[26], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Buffer manager interrupt.

INT_AE_LOSS_OF_SYNC R/W, CON489[27], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Loss of sync in the Acquisition Engine interrupt.

INT_PCIE_PKT_DROPPED R/W, CON489[28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

PCIe packet dropped interrupt.

INT_Y_ACQUIRED_LEGACY R/W, CON489[29], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Copy of INT_Y_ACQUIRED.

2.25 CON490

Bit	Name
0	Reserved
1	Reserved
2	Reserved
3	Reserved
4	Reserved
5	Reserved
6	Reserved
7	INT_ANY
8	ENINT_ALL
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

INT_ANY RO, CON490[7], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

T is at least on active interrupt on the board.

ENINT_ALL R/W, CON490[8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Set to 1 to enable board interrupts.

2.26 CON548

Bit	Name
0	INT_Y_ACQUIRED_M
1	INT_X_ACQUIRED_M
2	INT_Z_ACQUIRED_M
3	Reserved
4	INT_ENC_B_M
5	INT_ENC_A_M
6	INT_TRIG_M
7	INT_Y_START_M
8	INT_X_START_M
9	INT_Z_START_M
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	INT_BM_ERROR_M
27	INT_AE_LOSS_OF_SYNC_M
28	INT_PCIE_PKT_DROPPED_M
29	INT_Y_ACQUIRED_LEGACY_M
30	Reserved
31	Reserved

INT_Y_ACQUIRED_M R/W, CON548[0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_Y_ACQUIRED mask.

INT_X_ACQUIRED_M R/W, CON548[1], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_X_ACQUIRED mask.

INT_Z_ACQUIRED_M R/W, CON548[2], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_Z_ACQUIRED mask.

INT_ENC_B_M R/W, CON548[4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_ENC_B mask.

INT_ENC_A_M R/W, CON548[5], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_ENC_A mask.

INT_TRIG_M R/W, CON548[6], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_TRIG mask.

INT_Y_START_M R/W, CON548[7], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_Y_START mask.

INT_X_START_M R/W, CON548[8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_X_START mask.

INT_Z_START_M R/W, CON548[9], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_Z_START mask.

INT_BM_ERROR_M R/W, CON548[26], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_BM_ERROR mask.

INT_AE_LOSS_OF_SYNC_M R/W, CON548[27], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_AE_LOSS_OF_SYNC mask.

INT_PCIE_PKT_DROPPED_M R/W, CON548[28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_PCIE_PKT_DROPPED mask.

INT_Y_ACQUIRED_LEGACY_M R/W, CON548[29], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_Y_ACQUIRED_LEGACY mask.

2.27 CON549

Bit	Name
0	INT_Y_ACQUIRED_WP
1	INT_X_ACQUIRED_WP
2	INT_Z_ACQUIRED_WP
3	Reserved
4	INT_ENC_B_WP
5	INT_ENC_A_WP
6	INT_TRIG_WP
7	INT_Y_START_WP
8	INT_X_START_WP
9	INT_Z_START_WP
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	INT_BM_ERROR_WP
27	INT_AE_LOSS_OF_SYNC_WP
28	INT_PCIE_PKT_DROPPED_WP
29	INT_Y_ACQUIRED_LEGACY_WP
30	Reserved
31	Reserved

INT_Y_ACQUIRED_WP R/W, CON549[0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_Y_ACQUIRED write protect.

INT_X_ACQUIRED_WP R/W, CON549[1], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_X_ACQUIRED write protect.

INT_Z_ACQUIRED_WP R/W, CON549[2], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_Z_ACQUIRED write protect.

INT_ENC_B_WP R/W, CON549[4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_ENC_B write protect.

INT_ENC_A_WP R/W, CON549[5], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_ENC_A write protect.

INT_Y_START_WP R/W, CON548[7], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_Y_START write protect.

INT_X_START_WP R/W, CON548[8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_X_START write protect.

INT_Z_START_WP R/W, CON548[9], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_Z_START write protect.

INT_TRIG_WP R/W, CON549[6], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_TRIG write protect.

INT_BM_ERROR_WP R/W, CON549[26], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_BM_ERROR write protect.

INT_AE_LOSS_OF_SYNC_WP R/W, CON549[27], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_AE_LOSS_OF_SYNC write protect.

INT_PCIE_PKT_DROPPED_WP R/W, CON549[28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_PCIE_PKT_DROPPED write protect.

INT_Y_ACQUIRED_LEGACY_WP R/W, CON549[29], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

INT_Y_ACQUIRED_LEGACY write protect.

2.28 SF_DIM

Bit	Name
0	SF_HEIGHT
1	SF_HEIGHT
2	SF_HEIGHT
3	SF_HEIGHT
4	SF_HEIGHT
5	SF_HEIGHT
6	SF_HEIGHT
7	SF_HEIGHT
8	SF_HEIGHT
9	SF_HEIGHT
10	SF_HEIGHT
11	SF_HEIGHT
12	SF_HEIGHT
13	SF_HEIGHT
14	SF_HEIGHT
15	SF_HEIGHT
16	SF_WIDTH
17	SF_WIDTH
18	SF_WIDTH
19	SF_WIDTH
20	SF_WIDTH
21	SF_WIDTH
22	SF_WIDTH
23	SF_WIDTH
24	SF_WIDTH
25	SF_WIDTH
26	SF_WIDTH
27	SF_WIDTH
28	SF_WIDTH
29	SF_WIDTH
30	SF_WIDTH
31	SF_WIDTH

SF_HEIGHT R/W, SF_DIM[15..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The height (in lines) of the Synthetic Frame (internally generated synthetic image).

SF_WIDTH R/W, SF_DIM[31..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The width of the Synthetic frame. Units are 16 byte chunks.

2.29 SF_CON

Bit	Name
0	SF_RUN_LEVEL
1	SF_RUN_LEVEL
2	SF_STATE
3	SF_STATE
4	SF_MODE
5	SF_MODE
6	Reserved
7	SF_LINE_SCAN
8	SF_INIT_BYTE
9	SF_INIT_BYTE
10	SF_INIT_BYTE
11	SF_INIT_BYTE
12	SF_INIT_BYTE
13	SF_INIT_BYTE
14	SF_INIT_BYTE
15	SF_INIT_BYTE
16	SF_X_GAP
17	SF_X_GAP
18	SF_X_GAP
19	SF_X_GAP
20	SF_Y_GAP
21	SF_Y_GAP
22	SF_Y_GAP
23	SF_Y_GAP
24	SF_Z_GAP
25	SF_Z_GAP
26	SF_Z_GAP
27	SF_Z_GAP
28	SF_INC_X
29	SF_INC_Y
30	SF_INC_Z
31	Reserved

SF_RUN_LEVEL R/W, SF_CON[1..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The register controls the Synthetic Frame generator.

SF_RUN_LEVEL	Meaning/Command
0	Idle
1	Run
2	Abort
3	Reserved

SF_STATE RO, SF_CON[3..2], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register can be used to check the current state of the Synthetic Frame generator.

SF_MODE R/W, SF_CON[5..4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register controls if the Synthetic Frame generator is in free-running or triggered mode.

SF_MODE	Meaning
0	Free run
1	Triggered
2	Reserved
3	Reserved

SF_LINE_SCAN R/W, SF_CON[7], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Setting SF_LINE_SCAN to one will put the Synthetic Frame generator in line scan mode .

SF_INIT_BYTE R/W, SF_CON[15..8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The value of the first 8-bit pixel in the synthetic frame.

SF_X_GAP R/W, SF_CON[19..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The number of pixels between lines. Units are 16 byte chunks.

SF_Y_GAP R/W, SF_CON[23..20], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The number of lines between frames.

SF_Z_GAP R/W, SF_CON[27..24], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The number of frames between volumes.

SF_INC_X R/W, SF_CON[28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The amount to increment the grey scale output value every pixel.

SF_INC_Y R/W, SF_CON[29], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The amount to increment the grey scale output value every line.

SF_INC_Z R/W, SF_CON[30], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The amount to increment the grey scale output value every frame.

The StreamSync Buffer Manager

Chapter 3

3.1 Introduction

The StreamSync system consists of an Acquisition Engine and a Buffer Manager. The StreamSync system was first released on the Cyton-CXP and is a departure from previous BitFlow frame grabbers. The StreamSync system is a start-from-scratch complete redesign of the acquisition and DMA parts of a frame grabber. BitFlow used its years of experience in this area to design a next generation, super efficient capture system.

The StreamSync system is supported on the Aon-CXP, the Axion-CL, the Claxon-CXP and the Cyton-CXP.

From a software perspective, the StreamSync system is compatible with the previous BitFlow products. However, digging deeper, these new system have a lot more power and flexibility. These new features will be described in the following sections.

The StreamSync system has many improvements over previous systems. The main improvements are:

- Efficient support for variable sized images with fast context switches between frames
- Per frame control of acquisition properties (AOI specifically)
- Hardware control of image sequencing
- Enhanced debug capabilities
- Efficient support for on-demand buffer allocation (GenIcam model)
- Gracefully recovery from dropped packets (either on the input side or the DMA side)

This chapter describes the StreamSync Buffer Manager while the previous chapter describes the StreamSync Acquisition Engine.

3.2 The Buffer Manager Details

The Buffer Manager interacts with a remote, software managed, set of Scatter Gather DMA lists. A single Scatter Gather DMA list is called a QTab. A QTab is made of individual DMA instructions (descriptors) called Quads. One Quad contains the information to DMA one contiguous chunk of data from the board to host memory. The Buffer Manager reads in and precaches QTabs and the associated Quads. It makes the cached Quads and QTabs available to the Acquisition Engine in queued order. The Buffer Manager works independently of the Acquisition Engine and can be throttled by software.

The Buffer Manager and Acquisition Engine are designed to work asynchronously from each other. The Buffer Manager is capable of reading in Quads from the remote QTab while the Acquisition Engine is Running/Stopping/Aborting/or Stopped. If the local Buffer Cache fills and the Acquisition Engine is not currently consuming Quads, the Buffer Manager simply waits until room becomes available and pauses loading Quads from the remote QTab. Likewise, the Acquisition Engine is capable of acquiring frames as long as it is running and has Quad available to work on. If no Quads are available it will simply wait for more to become available from the Buffer Manager. The Acquisition Engine can accept commands of Stop/Abort/Start, all while the Buffer Manager is running independently.

The starting, stopping, and restarting of the Acquisition Engine and Buffer Manager, however, does require some synchronization. The Buffer Manager pre-fetches Quad and Quads for efficiency. This built up pipeline and caching structure requires the Acquisition Engine to be in the Stopped state before the Buffer Manager can be safely flushed. Flushing of the Buffer Manager happens when the user wants to completely shut down the StreamSync Acquisition Engine or simply start acquiring to a new QTab.

3.3 CON485 Register

Bit	Name
0	FIRST_QUAD_PTR_LO
1	FIRST_QUAD_PTR_LO
2	FIRST_QUAD_PTR_LO
3	FIRST_QUAD_PTR_LO
4	FIRST_QUAD_PTR_LO
5	FIRST_QUAD_PTR_LO
6	FIRST_QUAD_PTR_LO
7	FIRST_QUAD_PTR_LO
8	FIRST_QUAD_PTR_LO
9	FIRST_QUAD_PTR_LO
10	FIRST_QUAD_PTR_LO
11	FIRST_QUAD_PTR_LO
12	FIRST_QUAD_PTR_LO
13	FIRST_QUAD_PTR_LO
14	FIRST_QUAD_PTR_LO
15	FIRST_QUAD_PTR_LO
16	FIRST_QUAD_PTR_LO
17	FIRST_QUAD_PTR_LO
18	FIRST_QUAD_PTR_LO
19	FIRST_QUAD_PTR_LO
20	FIRST_QUAD_PTR_LO
21	FIRST_QUAD_PTR_LO
22	FIRST_QUAD_PTR_LO
23	FIRST_QUAD_PTR_LO
24	FIRST_QUAD_PTR_LO
25	FIRST_QUAD_PTR_LO
26	FIRST_QUAD_PTR_LO
27	FIRST_QUAD_PTR_LO
28	FIRST_QUAD_PTR_LO
29	FIRST_QUAD_PTR_LO
30	FIRST_QUAD_PTR_LO
31	FIRST_QUAD_PTR_LO

FIRST_QUAD_PTR_LO R/W, CON28[31..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the low word of the 64-bit address of the first DMA scatter-gather instruction in a chain of instructions.

3.4 CON486 Register

Bit	Name
0	FIRST_QUAD_PTR_HI
1	FIRST_QUAD_PTR_HI
2	FIRST_QUAD_PTR_HI
3	FIRST_QUAD_PTR_HI
4	FIRST_QUAD_PTR_HI
5	FIRST_QUAD_PTR_HI
6	FIRST_QUAD_PTR_HI
7	FIRST_QUAD_PTR_HI
8	FIRST_QUAD_PTR_HI
9	FIRST_QUAD_PTR_HI
10	FIRST_QUAD_PTR_HI
11	FIRST_QUAD_PTR_HI
12	FIRST_QUAD_PTR_HI
13	FIRST_QUAD_PTR_HI
14	FIRST_QUAD_PTR_HI
15	FIRST_QUAD_PTR_HI
16	FIRST_QUAD_PTR_HI
17	FIRST_QUAD_PTR_HI
18	FIRST_QUAD_PTR_HI
19	FIRST_QUAD_PTR_HI
20	FIRST_QUAD_PTR_HI
21	FIRST_QUAD_PTR_HI
22	FIRST_QUAD_PTR_HI
23	FIRST_QUAD_PTR_HI
24	FIRST_QUAD_PTR_HI
25	FIRST_QUAD_PTR_HI
26	FIRST_QUAD_PTR_HI
27	FIRST_QUAD_PTR_HI
28	FIRST_QUAD_PTR_HI
29	FIRST_QUAD_PTR_HI
30	FIRST_QUAD_PTR_HI
31	FIRST_QUAD_PTR_HI

FIRST_QUAD_PTR_HI R/W, CON29[31..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the high word of the 64-bit address of the first DMA scatter-gather instruction in a chain of instructions.

3.5 BUF_MGR_CON

Bit	Name
0	BM_RUN_LEVEL
1	BM_RUN_LEVEL
2	BM_RUN_LEVEL
3	BM_RUN_LEVEL
4	Reserved
5	Reserved
6	Reserved
7	Reserved
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	CURR_FETCH_SIZE
25	CURR_FETCH_SIZE
26	CURR_FETCH_SIZE
27	CURR_FETCH_SIZE
28	MAX_FETCH_SIZE
29	MAX_FETCH_SIZE
30	MAX_FETCH_SIZE
31	MAX_FETCH_SIZE

BM_RUN_LEVEL R/W, BUF_MGR_CON[3..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the main control for starting/stopping the Buffer Manager.

BM_RUN_LEVEL	Meaning
0 (0000b)	Idle - The Buffer Manager is not moving data
1 (0001b)	Run - The Buffer Manger will start to move data
2 (0010b)	Abort - Abort DMA and go to Idle
3 (0011b)	

CURR_FETCH_SIZE RO, BUF_MGR_CON[27..24], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the number of Quads that will be fetched at a time by the Buffer Manager. A large read is issued over the PCIe bus to read all these Quads at one time.

CURR_FETCH_SIZE	Meaning
0 (0000b)	1 Quad
1 (0001b)	2 Quads
2 (0010b)	4 Quads
3 (0011b)	8 Quads
15 (1111b)	32K Quads

MAX_FETCH_SIZE RO, BUF_MGR_CON[31..28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the maximum number of Quads that can be fetched as a group by the Buffer Manager. The value in this register is derived as a function of the maximum PCIe read request size set by PCI enumeration.

3.6 BUF_MGR_TIMEOUT

Bit	Name
0	QUAD_COMPLETE_TIMEOUT
1	QUAD_COMPLETE_TIMEOUT
2	QUAD_COMPLETE_TIMEOUT
3	QUAD_COMPLETE_TIMEOUT
4	QUAD_COMPLETE_TIMEOUT
5	QUAD_COMPLETE_TIMEOUT
6	QUAD_COMPLETE_TIMEOUT
7	QUAD_COMPLETE_TIMEOUT
8	QUAD_COMPLETE_TIMEOUT
9	QUAD_COMPLETE_TIMEOUT
10	QUAD_COMPLETE_TIMEOUT
11	QUAD_COMPLETE_TIMEOUT
12	QUAD_COMPLETE_TIMEOUT
13	QUAD_COMPLETE_TIMEOUT
14	QUAD_COMPLETE_TIMEOUT
15	QUAD_COMPLETE_TIMEOUT
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	DISABLE_TIMEOUT

QUAD_COMPLETE_TIMEOUT

R/W, BUF_MGR_TIMEOUT[15..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The maximum amount of time to wait for a Quad completion. Units are 4 nanoseconds. Writable only when BM_STATE is Idle.

DISABLE_TIMEOUT

R/W, BUF_MGR_TIMEOUT[31], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Setting this bit to 1 will disable the Quad completion timeout mechanism. The Buffer Manager will wait an infinite amount of time for a Quad completion to return. For debug only. Writable only when BM_STATE is Idle.

3.7 BOARD_CONFIG

Bit	Name
0	SW
1	SW
2	Reserved
3	Reserved
4	CPLD_MODE
5	CPLD_MODE
6	CPLD_MODE
7	CPLD_MODE
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	CPLD_STRAP
13	CPLD_STRAP
14	CPLD_STRAP
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

SW RO, BOARD_CONFIG[1..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The current value of the on board switch SW1.

CPLD_MODE RO, BOARD_CONFIG[7..4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The current value of switch S3. This switch controls the firmware bank that the FPGA boots from.

CPLD_STRAP RO, BOARD_CONFIG[14..12], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The current value of the three on board straps.

3.8 PACKETS_SENT_STATUS

Bit	Name
0	NUM_PACKETS_SENT
1	NUM_PACKETS_SENT
2	NUM_PACKETS_SENT
3	NUM_PACKETS_SENT
4	NUM_PACKETS_SENT
5	NUM_PACKETS_SENT
6	NUM_PACKETS_SENT
7	NUM_PACKETS_SENT
8	NUM_PACKETS_SENT
9	NUM_PACKETS_SENT
10	NUM_PACKETS_SENT
11	NUM_PACKETS_SENT
12	NUM_PACKETS_SENT
13	NUM_PACKETS_SENT
14	NUM_PACKETS_SENT
15	NUM_PACKETS_SENT
16	NUM_PACKETS_DROP
17	NUM_PACKETS_DROP
18	NUM_PACKETS_DROP
19	NUM_PACKETS_DROP
20	NUM_PACKETS_DROP
21	NUM_PACKETS_DROP
22	NUM_PACKETS_DROP
23	NUM_PACKETS_DROP
24	NUM_PACKETS_DROP
25	NUM_PACKETS_DROP
26	NUM_PACKETS_DROP
27	NUM_PACKETS_DROP
28	NUM_PACKETS_DROP
29	NUM_PACKETS_DROP
30	NUM_PACKETS_DROP
31	NUM_PACKETS_DROP

NUM_PACKETS_SENT RO, PACKETS_SENT_STATUS[15..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The register indicates the number of PCIe packets that the Buffer Manager has sent across the PCIe bus. This register rolls over to 0 at 0xffff.

NUM_PACKETS_DROP RO, PACKETS_SENT_STATUS[31..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register indicates the number of PCIe packets that the buffer Manager was not able to send across the PCIe bus because the PCIe bus was busy. These packets are dropped, but the corresponding Quads are also "consumed". This means that the Buffer Manager still stays synchronized and any subsequent packets will be DMA to their correct locations.

3.9 QUADS_USED_STATUS

Bit	Name
0	NUM_QUADS_USED
1	NUM_QUADS_USED
2	NUM_QUADS_USED
3	NUM_QUADS_USED
4	NUM_QUADS_USED
5	NUM_QUADS_USED
6	NUM_QUADS_USED
7	NUM_QUADS_USED
8	NUM_QUADS_USED
9	NUM_QUADS_USED
10	NUM_QUADS_USED
11	NUM_QUADS_USED
12	NUM_QUADS_USED
13	NUM_QUADS_USED
14	NUM_QUADS_USED
15	NUM_QUADS_USED
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

NUM_QUADS_USED RO, QUADS_USED_STATUS[15..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register indicates the number of Quads that have been “consumed” by the Buffer Manager. This register rolls over to 0 at 0xffff.

3.10 QTABS_USED_STATUS

Bit	Name
0	NUM_QTABS_USED
1	NUM_QTABS_USED
2	NUM_QTABS_USED
3	NUM_QTABS_USED
4	NUM_QTABS_USED
5	NUM_QTABS_USED
6	NUM_QTABS_USED
7	NUM_QTABS_USED
8	NUM_QTABS_USED
9	NUM_QTABS_USED
10	NUM_QTABS_USED
11	NUM_QTABS_USED
12	NUM_QTABS_USED
13	NUM_QTABS_USED
14	NUM_QTABS_USED
15	NUM_QTABS_USED
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

NUM_QTABS_USED RO, QTABS_USED_STATUS[15..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register indicates the number of QTabs that have been “consumed” by the Buffer Manager. This register rolls over to 0 at 0xffff.

3.11 PKT_STAT

Bit	Name
0	PKT_STATE
1	PKT_STATE
2	Reserved
3	Reserved
4	Reserved
5	Reserved
6	Reserved
7	Reserved
8	NO_QUAD_AVAIL
9	VIDEO_DROPPED
10	QUAD_DROPPED
11	Reserved
12	NEW_FRAME_RESYNC
13	RD_ON_EMPTY
14	WR_ON_FULL
15	Reserved
16	PKT_FLUSH_ENABLE
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

PKT_STATE RO, PKT_STAT[1..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Current state of the DMA engine.

PKT_STATE	Meaning
0 (00b)	PKT_SYNC - Synchronizing DMA descriptors with video
1 (01b)	PKT_HDR - Generating PCIe header
2 (10b)~	PKT_DAT - Placing data in PCIe packet
3 (11b)	Reserved

NO_QUAD_AVAIL RO, PKT_STAT[8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

StreamSync DMA has data to transmit but no descriptor (effectively no valid place to send data). This indicates a problem with fetching descriptors.

VIDEO_DROPPED RO, PKT_STAT[9], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Can occur during PKT_SYNC as video from acquisition engine is dropped in order to resynchronize.

QUAD_DROPPED RO, PKT_STAT[10], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Similar to VIDEO_DROPPED but indicates quad was dropped during re-sync process when PKT_STATE equals PKT_SYNC.

NEW_FRAME_RESYNC RO, PKT_STAT[12], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Reserved.

RD_ON_EMPTY RO, PKT_STAT[13], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

FIFO underflow in Packet Engine.

WR_ON_FULL R/W, PKT_STAT[14], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

FIFO overflow in Packet Engine.

PKT_FLUSH_ENABLE R/W, PKT_STAT[16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

DMA tries to send as large as packets as possible for efficiency. Data is collected in a FIFO until certain size rules are met. However, sometimes no more data will be coming (end of frame). In this case, a timeout forces the Packet Engine to transmit the remaining data. PKT_FLUSH_ENABLE = 1 indicates that this has taken place.

3.12 QUADS_LOADED_STATUS

Bit	Name
0	NUM_QUADS_LOADED
1	NUM_QUADS_LOADED
2	NUM_QUADS_LOADED
3	NUM_QUADS_LOADED
4	NUM_QUADS_LOADED
5	NUM_QUADS_LOADED
6	NUM_QUADS_LOADED
7	NUM_QUADS_LOADED
8	NUM_QUADS_LOADED
9	NUM_QUADS_LOADED
10	NUM_QUADS_LOADED
11	NUM_QUADS_LOADED
12	NUM_QUADS_LOADED
13	NUM_QUADS_LOADED
14	NUM_QUADS_LOADED
15	NUM_QUADS_LOADED
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

NUM_QUADS_LOADED RO, QUADS_LOADED_STATUS[15..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register indicates the number of Quads that have been loaded by the Buffer Manager. This register will roll over to 0 at 0xffff.

3.13 QTABS_LOADED_STATUS

Bit	Name
0	NUM_QTABS_LOADED
1	NUM_QTABS_LOADED
2	NUM_QTABS_LOADED
3	NUM_QTABS_LOADED
4	NUM_QTABS_LOADED
5	NUM_QTABS_LOADED
6	NUM_QTABS_LOADED
7	NUM_QTABS_LOADED
8	NUM_QTABS_LOADED
9	NUM_QTABS_LOADED
10	NUM_QTABS_LOADED
11	NUM_QTABS_LOADED
12	NUM_QTABS_LOADED
13	NUM_QTABS_LOADED
14	NUM_QTABS_LOADED
15	NUM_QTABS_LOADED
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

NUM_QTABS_LOADED RO, QTABS_LOADED_STATUS[15..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register indicates the number of QTabs that have been loaded by the Buffer Manager. This register will roll over to 0 at 0xffff.

3.14 BUF_MGR_STATUS

Bit	Name
0	BM_STATE
1	BM_STATE
2	BM_STATE
3	Reserved
4	CPL_STATUS
5	CPL_STATUS
6	CPL_STATUS
7	Reserved
8	BM_QUADS_CACHED
9	BM_QUADS_CACHED
10	BM_QUADS_CACHED
11	BM_QUADS_CACHED
12	BM_QUADS_CACHED
13	BM_QUADS_CACHED
14	BM_QUADS_CACHED
15	BM_QUADS_CACHED
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	DST_ADDR_ERROR_LSB
21	NEXT_ADDR_ERROR_LSB
22	SIZE_ERROR_LSB
23	SIZE_ERROR_MSB
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	CPL_ERROR
29	QUAD_NUM_MISMATCH
30	QUAD_FIFO_OVERFLOW
31	QUAD_TIMEOUT_DETECTED

BM_STATE RO, BUF_MGR_STATUS[2..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Returns the current state of the Buffer Manager.

BM_STATE	Meaning
0 (0000b)	Idle - The buffer manager is not current active
1 (0001b)	Active - The buffer manager is currently DMAing
2 (0010b)	Req64
3 (0011b)	Req32
4 (0100b)	Wait CPL
4 (0101b)	Parse CPL 0
6 (0110b)	Parse CPL 0
7 (0111b)	Flush

CPL_STATUS RO, BUF_MGR_STATUS[6..4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

PCIe completion status from last received Quad.

BM_QUADS_CACHED RO, BUF_MGR_STATUS[15..8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Number of QUADS currently in the cache.

DST_ADDR_ERROR_LSB RO, BUF_MGR_STATUS[20], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Quad destination address is not 16 byte aligned.

NEXT_ADDR_ERROR_LSB RO, BUF_MGR_STATUS[21], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Quad points to a next quad that is not 16-byte aligned..

SIZE_ERROR_LSB RO, BUF_MGR_STATUS[22], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Quad size is not a multiple of 16 bytes.

SIZE_ERROR_MSB RO, BUF_MGR_STATUS[23], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Quad size is > 4K.

CPL_ERROR RO, BUF_MGR_STATUS[28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Error code received as a result of fetching a Quad. Check CPL_STATUS.

QUAD_NUM_MISMATCH RO, BUF_MGR_STATUS[29], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Actual quad number does not match expected.

QUAD_FIFO_OVERFLOW RO, BUF_MGR_STATUS[30], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Quad cache overflowed.

QUAD_TIMEOUT_DETECTED RO, BUF_MGR_STATUS[31], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Timeout waiting for a Quad completion. A different timeout value can be set in the QUAD_COMPLETE_TIMEOUT register.

3.15 PKT_CON

Bit	Name
0	MAX_PAYLOAD_USER
1	MAX_PAYLOAD_USER
2	MAX_PAYLOAD_USER
3	MAX_PAYLOAD_USER
4	MAX_PAYLOAD_PCIE
5	MAX_PAYLOAD_PCIE
6	MAX_PAYLOAD_PCIE
7	MAX_PAYLOAD_PCIE
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	DISABLE_PKT_FLUSH_TIMER
31	DISABLE_PKT_GEN

MAX_ PAYLOAD_USER RO, PKT_CON[3..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the maximum sized PCIe packet that will be generated by the Buffer Manager. Writes to this register of values higher than MAX_PAYLOAD_PCIE will be ignored. The coding is shown in the following table.

MAX_PAYLOAD_USER	Meaning
0 (0000b)	16 bytes
1 (0001b)	32 bytes
2 (0010b)	64 bytes
3 (0011b)	128 bytes
4 (0100b)	256 bytes
4 (0101b)	512 bytes
6 (0110b)	1024 bytes
7 (0111b)	2048 bytes
8 (1000b)	4096 bytes

MAX_ PAYLOAD_PCIE RO, PKT_CON[7..4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the maximum sized PCIe write packet that can be generated by the Buffer Manager for video data. This value is set by the PCIe enumeration. The coding for this field is the same as listed under MAX_PAYLOAD_USER. Only values of 128 bytes to 4096 bytes are possible in PCIe, however, MAX_PAYLOAD_USER provides a few smaller values (16 bytes to 64 bytes) for testing purposes only. MAX_PAYLOAD_PCIE is status only and does not control internal logic. MAX_PAYLOAD_USER does control internal logic.

DISABLE_PKT_ FLUSH_TIMER R/W, PKT_CON[30], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Deactivate the timer that flushes video data to PCIe when the FIFO is inactive for an extended period.

Note: This bit is for degging purposes only.

DISABLE_PKT_ GEN R/W, PKT_CON[31], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Disable the generation of outbound PCIe video packets. This is a final stage disable. The packet is actually generated by the logic as if it were going to be transmitted. This means that all address's and other counters increment as if the packet were generated. However it is simply dropped afterwards.

Note: This bit is for debugging purposes only.

Timing Sequencer

Chapter 4

4.1 Introduction

This section covers the Timing Sequencer (TS) which is available on the Aon, Axion, Claxon and Cyton. The TS is a sophisticated programmable pulse generator. The TS takes the place of the NTG on previous models of BitFlow frame grabbers.

The TS improves on the NTG in the following ways:

- Driven by a "nice" clock frequency clock so that "normal" pulse sizes and periods can easily be reproduced (for example 1 micro second pulse every 10 milliseconds)
- Higher accuracy signals, granularity down to 100 nanoseconds
- Supports complex pulse trains of different lengths
- Provides synchronized method to switch from one pulse sequence to another
- Can be reprogrammed while being used (with some restrictions)
- Supports triggering at arbitrary points in a pulse train

4.1.1 The Auxiliary Timing Sequencer

The first Virtual Frame Grabber (VFG0) has two independent Timing Sequencers: TS and ATS. They work exactly the same, and have the same control registers. The Auxiliary Timing Sequencer registers are listed here, even though they work in the exact same way as the main timing sequencer. All of the description that follows applies to both the TS and the ATS.

Note: The Auxiliary Timing Sequencer (ATS) is only available on VFG0.

4.1.2 Description

TS Table

The TS is programmed through the TS registers. The sequence of pulses that the TS will output is programmed by building up "instructions" in the TS table. The table can hold up to 256 instructions, which can create extremely complex signals. The TS Table is programmed indirectly via address/data type registers. Once the table is programmed it can be run at any time via the TS control registers.

Building Pulses

The TS was designed to support a wide range of pulse lengths. At the same time, the TS was design to be able to create pulses of very accurate duration. The solution to these two opposing problems is to build up a pulse of a desired length via multiple sub-pulses, each sub-pulse programmed with a different granularity. The following granularities are available:

100 seconds
100 milliseconds
100 microseconds
100 nanoseconds

Each sub-pulse can have a length of 1 to 1023 units, where the units are selected from the list above.

For example, let's say you want a pulse that is 1.2345678 seconds long. This is done by programing the TS table with three sub pulse as shown below:

Entry 1: 12 * 100 milliseconds = 1.2 seconds
Entry 2: 345 * 100 microseconds = 0.0345 seconds
Entry 3: 678 * 100 nanoseconds = 0.0000678 seconds

When these three entries are "run", they are output sequentially and seamlessly, creating a single pulse of the desired length:

1.2 + 0.0345 + 0.0000678 = 1.2345678

As you can see, this system provides both a wide range of durations as well as very accurate durations.

Pulses can be either high or low (0 or 1). By programming both high pulses and low pulses any pulse train can be created.

Chaining Pulses

Pulses are chained together by a linked list. Each pulse entry has a "next" field which tells the system where in the table the next pulse should come from. This facility is used to build up complex sequences as well as looping sequences.

Triggering

Each entry in the TS table can produce one pulse. Each entry has a "condition" under which it will get executed. The conditions can be immediate. In other words, as soon as the TS gets to this entry, it immediately produces the programmed pulses. Or it can be programmed to wait for a trigger. Various trigger conditions are supported.

By adding this condition, the pulse train produced can be run in "one-shot" mode, where one trigger produces one or more pulses.

4.2 TS_CONTROL

Bit	Name
0	TS_RUN_LEVEL
1	TS_RUN_LEVEL
2	TS_RUN_LEVEL
3	Reserved
4	TS_CT0_DEFAULT_STATE
5	TS_CT1_DEFAULT_STATE
6	TS_CT2_DEFAULT_STATE
7	TS_CT3_DEFAULT_STATE
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	TS_IDX_JUMP
17	TS_IDX_JUMP
18	TS_IDX_JUMP
19	TS_IDX_JUMP
20	TS_IDX_JUMP
21	TS_IDX_JUMP
22	TS_IDX_JUMP
23	TS_IDX_JUMP
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	TS_TRIG_SEL
29	TS_TRIG_SEL
30	TS_TRIG_SEL
31	TS_TRIG_SEL

TS_RUN_LEVEL R/W, TS_CONTROL[2..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

These bits control the operation of the TS. These bits are used to start and stop the sequencer. They can also be used to program the table to jump to a new section. Jumps are always synchronous (i.e. not immediate). Jumps will only occur from an index in the sequence that has the TS_END_OF_SEQUENCE bit set.

This bit can be read at any time in order get the current status.

The following table shows the command available for this register.

TS_RUN_LEVEL	Meaning
0 (000b)	Idle - TS is not running
1 (001b)	Run - Start running immediately from index in the TS_IDX_ JUMP register
2 (010b)	Jump - Jump to index set in the TS_IDX_JUMP register next time the current index has the TS_END_OF_SEQUENCE bit set to 1
3 (011b)	Stop - Stop running the next time the current index has the TS_END_OF_SEQUENCE bit set to 1
4 (100b)	Abort - Stop running immediately

TS_CT0_ DEFAULT_STATE R/W, TS_CONTROL[4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the output state of CT0 when the TS is Idle.

TS_CT1_ DEFAULT_STATE R/W, TS_CONTROL[5], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the output state of CT1 when the TS is Idle.

TS_CT2_ DEFAULT_STATE R/W, TS_CONTROL[6], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the output state of CT2 when the TS is Idle.

TS_CT3_ DEFAULT_STATE R/W, TS_CONTROL[7], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the output state of CT3 when the TS is Idle.

TS_IDX_JUMP R/W, TS_CONTROL[23..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the entry that the table will start from when the TS_RUN_LEVEL register is set to Run.

This is the entry that the table will jump to (synchronously) when the TS_RUN_LEVEL register is set to Jump.

TS_TRIG_SEL R/W, TS_CONTROL[31..28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

These bits select the source of the TS trigger.

TS_TRIG_SEL	Meaning
0 (0000b)	Selected trigger (VGFx_TRIG_SEL)
1 (0001b)	Selected encoder A (VFGx_ENCA_SEL)
2 (0010b)	Selected encoder B (VFGx_ENCB_SEL)
3 (0011b)	Selected quad encoder output (VFGx_ENCQ_SEL)
4 (0100b)	Gated trigger (VGFx_TRIG_SEL gated by VFGx_ENCB_SEL)
5 (0101b)	Selected encoder divider output (VFGx_ENCDIV_SEL)
6 (0110b)	AE start of frame (Y Window Opens)
7 (0111b)	AE end of frame (Y Window Closes)
8 (1000b)	AE start of line (X Window Opens)
9 (1001b)	AE end of line (X Window Closes)
10 (1010b)	FVAL rising edge*
11 (1011b)	FVAL falling edge*
12 (1100b)	LVAL rising edge*
13 (1101b)	LVAL falling edge*
14 (1110b)	Reserved
15 (1111b)	Reserved

*Note: * FVAL and LVAL triggers are only support on the Axion models*

4.3 TS_TABLE_CONTROL

Bit	Name
0	TS_IDX_ACCESS
1	TS_IDX_ACCESS
2	TS_IDX_ACCESS
3	TS_IDX_ACCESS
4	TS_IDX_ACCESS
5	TS_IDX_ACCESS
6	TS_IDX_ACCESS
7	TS_IDX_ACCESS
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

TS_IDX_ACCESS R/W, TS_TABLE_CONTROL[7..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Indirect access to the TS table. Set this bitfield to the index value that you wish to modify. Access is done through via TS_TABLE_ENTRY register.

4.4 TS_TABLE_ENTRY

Bit	Name
0	TS_NEXT
1	TS_NEXT
2	TS_NEXT
3	TS_NEXT
4	TS_NEXT
5	TS_NEXT
6	TS_NEXT
7	TS_NEXT
8	Reserved
9	Reserved
10	TS_RESOLUTION
11	TS_RESOLUTION
12	TS_STATE_CT0
13	TS_STATE_CT1
14	TS_STATE_CT2
15	TS_STATE_CT3
16	Reserved
17	TS_COUNT
18	TS_COUNT
19	TS_COUNT
20	TS_COUNT
21	TS_COUNT
22	TS_COUNT
23	TS_COUNT
24	TS_COUNT
25	TS_COUNT
26	TS_COUNT
27	TS_CONDITION
28	TS_CONDITION
29	TS_CONDITION
30	TS_TERMINATE
31	TS_END_OF_SEQUENCE

TS_NEXT R/W, TS_TABLE_ENTRY[7..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Index of next pulse. Only relevant if TS_TERMINATE = 0.

TS_RESOLUTION R/W, TS_TABLE_ENTRY[11..10], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The time units of this pulse. The length of this pulse is set in the register TS_COUNT. The following table shows the available resolutions.

TS_RESOLUTION	Meaning
0 (000b)	100 nanoseconds
1 (001b)	100 microseconds
2 (010b)	100 milliseconds
3 (011b)	100 seconds

TS_STATE_CT0 R/W, TS_TABLE_ENTRY[12], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The level of the CT0 signal for this pulse.

TS_STATE_CT1 R/W, TS_TABLE_ENTRY[13], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The level of the CT1 signal for this pulse.

TS_STATE_CT2 R/W, TS_TABLE_ENTRY[14], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The level of the CT2 signal for this pulse.

TS_STATE_CT3 R/W, TS_TABLE_ENTRY[15], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The level of the CT3 signal for this pulse.

TS_COUNT R/W, TS_TABLE_ENTRY[26..17], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The length of this pulse. The units for the length are set in the TS_RESOLUTION register.

TS_CONDITION R/W, TS_TABLE_ENTRY[29..27], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register is used to control the conditions under which this pulse will be output. The following table shows the options for this bitfield.

TS_CONDITION	Condition when pulse is output
0 (000b)	Immediate
1 (001b)	Rising edge of trigger
2 (010b)	Falling edge of trigger
3 (011b)	Trigger high
4 (100b)	Trigger low
5 (101b)	Both rising and falling edge of trigger

TS_TERMINATE R/W, TS_TABLE_ENTRY[30], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

When this bit is set to 1, the table will stop running after the current pulse is output. The TS_RUN_LEVEL bitfield will then read back idle.

When this bit is set to 0, the table will jump to the index set in the TS_NEXT bitfield after the current pulse is finished.

TS_END_OF_SEQUENCE R/W, TS_TABLE_ENTRY[31], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

If this bit is set to 1 and TS_RUN_LEVEL is set to Jump, the TS will jump to the index set in the TS_INDX_JUMP bitfield after the current pulse is output. This bit allows for synchronous switching between one section of the table and another section.

If the TS_RUN_LEVEL bitfield is not set to Jump, then this bitfield will have no effect.

If this bit is set to 0, the TS will not jump from this index.

4.5 ATS_CONTROL

Bit	Name
0	ATS_RUN_LEVEL
1	ATS_RUN_LEVEL
2	ATS_RUN_LEVEL
3	Reserved
4	ATS_CT0_DEFAULT_STATE
5	ATS_CT1_DEFAULT_STATE
6	ATS_CT2_DEFAULT_STATE
7	ATS_CT3_DEFAULT_STATE
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	ATS_IDX_JUMP
17	ATS_IDX_JUMP
18	ATS_IDX_JUMP
19	ATS_IDX_JUMP
20	ATS_IDX_JUMP
21	ATS_IDX_JUMP
22	ATS_IDX_JUMP
23	ATS_IDX_JUMP
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	ATS_TRIG_SEL
29	ATS_TRIG_SEL
30	ATS_TRIG_SEL
31	ATS_TRIG_SEL

ATS_RUN_LEVEL R/W, ATS_CONTROL[2..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

These bits control the operation of the ATS. These bits are used to start and stop the sequencer. They can also be used to program the table to jump to a new section. Jumps are always synchronous (i.e. not immediate). Jumps will only occur from an index in the sequence that has the ATS_END_OF_SEQUENCE bit set.

This bit can be read at any time in order get the current status.

The following table shows the command available for this register.

ATS_RUN_LEVEL	Meaning
0 (000b)	Idle - ATS is not running
1 (001b)	Run - Start running immediately from index in the ATS_IDX_JUMP register
2 (010b)	Jump - Jump to index set in the ATS_IDX_JUMP register next time the current index has the ATS_END_OF_SEQUENCE bit set to 1
3 (011b)	Stop - Stop running the next time the current index has the ATS_END_OF_SEQUENCE bit set to 1
4 (100b)	Abort - Stop running immediately

ATS_CT0_DEFAULT_STATE R/W, ATS_CONTROL[4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the output state of CT0 when the ATS is Idle.

ATS_CT1_DEFAULT_STATE R/W, ATS_CONTROL[5], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the output state of CT1 when the ATS is Idle.

ATS_CT2_DEFAULT_STATE R/W, ATS_CONTROL[6], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the output state of CT2 when the ATS is Idle.

ATS_CT3_DEFAULT_STATE R/W, ATS_CONTROL[7], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the output state of CT3 when the ATS is Idle.

ATS_IDX_JUMP R/W, ATS_CONTROL[23..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This is the entry that the table will start from when the ATS_RUN_LEVEL register is set to Run.

This is also the entry that the table will jump to (synchronously) when the ATS_RUN_LEVEL register is set to Jump.

ATS_TRIG_SEL R/W, ATS_CONTROL[28..31], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

These bits select the source of the ATS trigger.

TS_TRIG_SEL	Meaning
0 (0000b)	Selected trigger (VGFx_TRIG_SEL)
1 (0001b)	Selected encoder A (VFGx_ENCA_SEL)
2 (0010b)	Selected encoder B (VFGx_ENCB_SEL)
3 (0011b)	Selected quad encoder output (VFGx_ENCQ_SEL)
4 (0100b)	Gated trigger (VGFx_TRIG_SEL gated by VFGx_ENCB_SEL)
5 (0101b)	Selected encoder divider output (VFGx_ENCDIV_SEL)
6 (0110b)	Acquisition start of frame (Y Window Open)
7 (0111b)	Acquisition end of frame (Y Window Close)
8 (1000b)	Acquisition start of line (X Window Open)
9 (1001b)	Acquisition end of line (X Window Open)
10 (1010b)	FVAL rising edge*
11 (1011b)	FVAL falling edge*
12 (1100b)	LVAL rising edge*
13 (1101b)	LVAL falling edge*
14 (1110b)	Reserved
15 (1111b)	Reserved

*Note: * FVAL and LVAL triggers are only support on the Axion models*

4.6 ATS_TABLE_CONTROL

Bit	Name
0	ATS_IDX_ACCESS
1	ATS_IDX_ACCESS
2	ATS_IDX_ACCESS
3	ATS_IDX_ACCESS
4	ATS_IDX_ACCESS
5	ATS_IDX_ACCESS
6	ATS_IDX_ACCESS
7	ATS_IDX_ACCESS
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

ATS_IDX_ACCESS R/W, ATS_TABLE_CONTROL[7..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Indirect access to the TS table. Set this bitfield to the index value that you wish to modify. Access is done through via TS_TABLE_ENTRY register.

4.7 ATS_TABLE_ENTRY

Bit	Name
0	ATS_NEXT
1	ATS_NEXT
2	ATS_NEXT
3	ATS_NEXT
4	ATS_NEXT
5	ATS_NEXT
6	ATS_NEXT
7	ATS_NEXT
8	Reserved
9	Reserved
10	ATS_RESOLUTION
11	ATS_RESOLUTION
12	ATS_STATE_CT0
13	ATS_STATE_CT1
14	ATS_STATE_CT2
15	ATS_STATE_CT3
16	Reserved
17	ATS_COUNT
18	ATS_COUNT
19	ATS_COUNT
20	ATS_COUNT
21	ATS_COUNT
22	ATS_COUNT
23	ATS_COUNT
24	ATS_COUNT
25	ATS_COUNT
26	ATS_COUNT
27	ATS_CONDITION
28	ATS_CONDITION
29	ATS_CONDITION
30	ATS_TERMINATE
31	ATS_END_OF_SEQUENCE

ATS_NEXT R/W, ATS_TABLE_ENTRY[7..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Index of next pulse. Only follow if ATS_TERMINATE = 0.

ATS_ RESOLUTION R/W, ATS_TABLE_ENTRY[11..10], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The time units of this pulse. The length of this pulse is set in the register ATS_COUNT. The following table shows the available resolutions.

ATS_RESOLUTION	Meaning
0 (000b)	100 nanoseconds
1 (001b)	100 microseconds
2 (010b)	100 milliseconds
3 (011b)	100 seconds

ATS_STATE_CT0 R/W, ATS_TABLE_ENTRY[12], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The level of the CT0 signal for this pulse.

ATS_STATE_CT1 R/W, ATS_TABLE_ENTRY[13], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The level of the CT1 signal for this pulse.

ATS_STATE_CT2 R/W, ATS_TABLE_ENTRY[14], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The level of the CT2 signal for this pulse.

ATS_STATE_CT3 R/W, ATS_TABLE_ENTRY[15], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The level of the CT3 signal for this pulse.

ATS_COUNT R/W, ATS_TABLE_ENTRY[26..17], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The length of this pulse. The units for the length are set in the ATS_RESOLUTION register.

ATS_ CONDITION

R/W, ATS_TABLE_ENTRY[29..27], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register is used to control the conditions under which this pulse will be output. The following table shows the options for this bitfield.

ATS_CONDITION	Condition when pulse is output
0 (000b)	Immediate
1 (001b)	Rising edge of trigger
2 (010b)	Falling edge of trigger
3 (011b)	Trigger high
4 (100b)	Trigger low
5 (101b)	Both rising and falling edge of trigger

ATS_ TERMINATE

R/W, ATS_TABLE_ENTRY[30], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

When this bit is set to 0, the table will stop running after the current pulse is output. The ATS_RUN_LEVEL bitfield will then read back idle.

When this bit is set to 1, the table will jump to the index set in the ATS_NEXT bitfield after the current pulse is finished.

ATS_END_OF_ SEQUENCE

R/W, ATS_TABLE_ENTRY[31], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

If this bit is set to 1 and ATS_RUN_LEVEL is set to Jump, the ATS will jump to the index set in the ATS_INDX_JUMP bitfield after the current pulse is output. This bit allows for synchronous switching between one section of the table and another section.

If the ATS_RUN_LEVEL bitfield is not set to Jump, then this bitfield will have no effect.

If this bit is set to 0, the ATS will not jump from this index.

The Aon, Axion, Claxon and Cyton I/O System

Chapter 5

5.1 Introduction

The I/O system on the Aon, Axion, Claxon and Cyton families of frame grabbers are based on the Karbon family with some minor changes. This system provides unprecedented flexibility. The goal of this system is to handle all the I/O needs of any machine vision application connected to the real world in a wide variety of ways. The goal is flexibility and observability.

5.1.1 Concepts

The basic concept is that the outside world of a machine vision system can have a wide variety of signals, possibly using different electrical standards. The Cyton and/or Axion user can choose whichever ones best suit their needs. These inputs can then be routed to a wide range of internal destinations. Also, the board can generate its own signals of use in this system.

Once the input sources are chosen, they are routed to one of a number of internal signals. These internal signals can then be used to control a wide variety of functions. For example, a function might be to cause the board to acquire a frame.

Finally the internal signals can be routed off the board to a wide number of destinations. These can be used to control something in the outside world. For example, a signal might be routed such that it fires a strobe light or initiates the start of exposure in a camera.

The state of all of the possible inputs can be observed by software at any time by peeking the associated RD_XXX bit.

5.1.2 I/O Between Virtual Frame Grabbers

Because BitFlow's frame grabber can acquire from more than one camera, it has always been a contention between the desire to let each camera be independent, for example, each with its own trigger, and for them to be synchronized, i.e. all cameras using one trigger. The Cyton and Axion I/O system provides the best of both worlds by fully supporting independent and synchronized triggers using the same flexibility as each individual VFG gets.

What this means is that one of the sources, the master VFG's trigger can be the trigger for all of the VFGs. Thus whatever signal is triggering VFG0, can also trigger all the other VFGs on the board. Of course, each VFG can choose to use the master VFG's trigger, or choose amongst its own sources. Further, the triggers can be synchronized while the encoders are independent.

5.2 Overview of the I/O System Routing

Figure 5-1 below shows a generic version of the I/O system routing. For each internal signal (A and B in this case) a source must be chosen. Each internal signal can be used to control one or more internal circuits or can be routed straight to an output signal. For each output signal (X & Y in this case) a source must be chosen. In addition, there are internal signal generators that can be chosen as a source for an internal signal or an external signal. Even the internal signal generators can be triggered by an internal signal.

Note: Figure 5-1 is a schematic version of the actual circuit, it is greatly simplified to make it easier to understand.

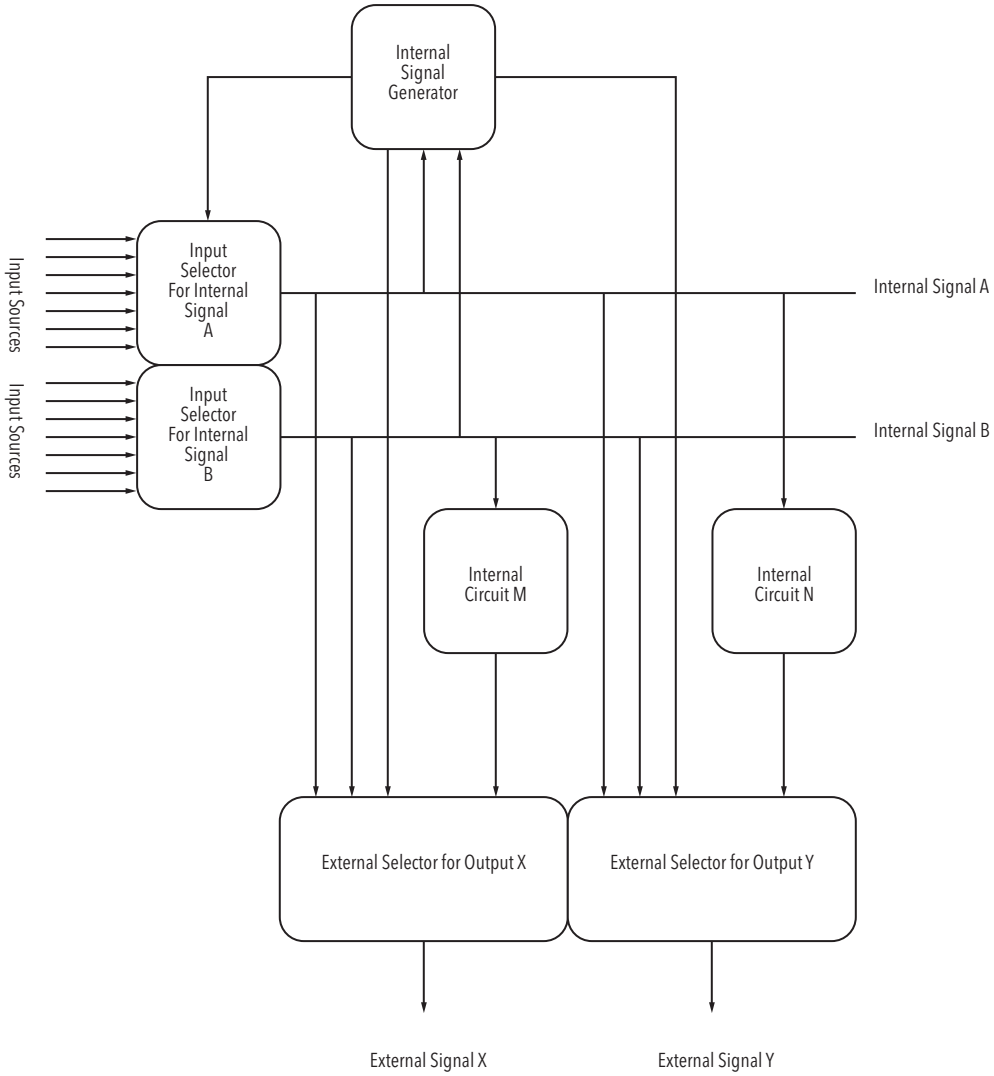

Figure 5-1 Conceptual I/O System Routing

5.3 Input Selection

Figure 5-2 illustrates the I/O System input selection circuitry. As discussed above, each internal signal has its own selector. For each internal signal there are 64 possible sources.

Note: The signals BOX_IN_XXX are available via an external I/O Box, the BitFlow BitBox, which can be mounted on an external rail system. Contact BitFlow for more information on the BitBox.

Note: Each VFG has a copy of the circuit shown in Figure 5-2. This is why the outputs do not specify the VFG number (e.g. "VFGx_TRIG_SEL). However, some inputs do specify the VFG number (e.g. VFG0_TRIG_SEL), which means this input comes explicitly from VFG0.

Note: The Aon-CXP, Claxon-CXP1, Axion-1xE and Axion-1xB have only one VFG.

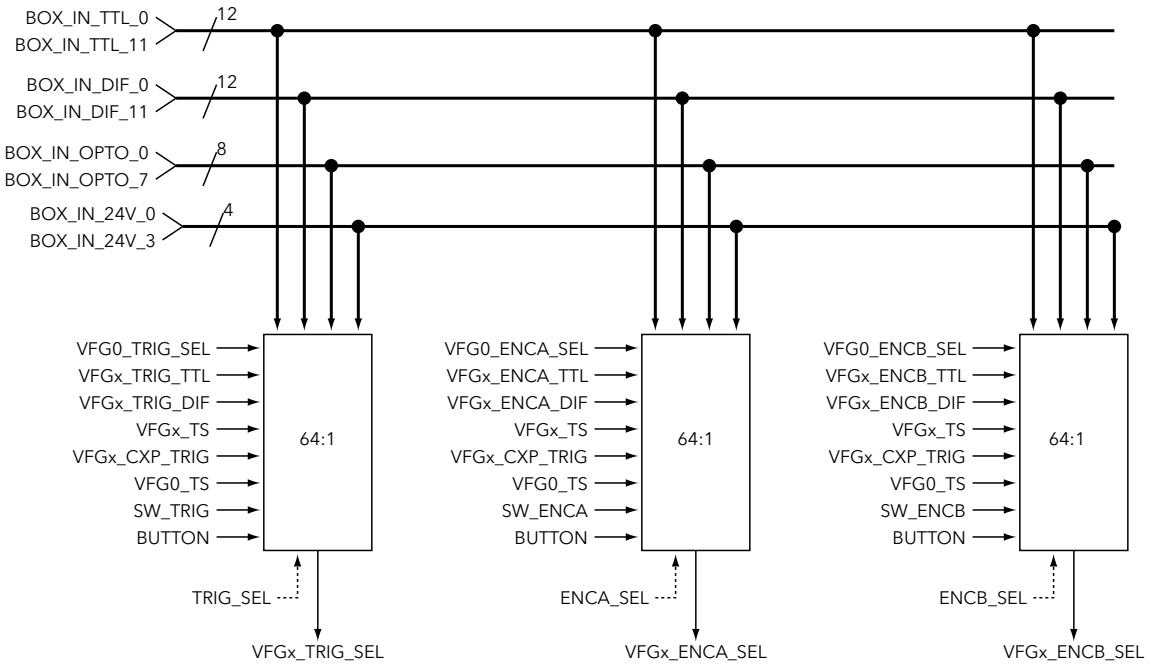

Figure 5-2 I/O System Input Selection

5.4 Internal Signals

There are five internal I/O signals. The "x" in the signal name refers to the VFG number of the current VFG being used. In general, the "x" is always used as all VFGs are symmetrical. The one exceptions is VFG0, which can route many of its signals to other VFGs on the same physical board. For example VFG2 can use the trigger selected on VFG 0 (i.e. VFG0_TRIG_SEL) as its trigger source.

VFGx_TRIG_SEL - normally used as a frame trigger, can also be used to initiate acquisition of N frames, or can be used to control the start and end of a frame when used with a line scan camera.

VFGx_ENCA_SEL - Normally used as a line trigger, to initiate the capture of one line. When using a single phase encoder, this is the signal to use.

VFGx_ENCB_SEL - Normally used as a line trigger when using a quadrature encoder. The encoder B should be the second phase of the quadrature encoder with Encoder A being the first phase.

VFGx_ENCDIV_SEL - This signal is the output of the encoder divider (multiplier) circuit. This circuit can be driven by more than one source. The output signal is related to the input signal, but the frequency is either divided down or multiplied up.

VFGx_ENCQ_SEL - This signal is the output of the quadrature encoder circuit. This circuit takes two inputs, the selected encoder A and the selected encoder B. The output signal follows the rules as programmed in to the quadrature encoder circuit, see Section 8.1 for more information.

Note: The internal signals have names such as "Trigger" and "Encoder" because they are hardwired to certain internal circuits. However, if you are not using them for this functionality, they can be used for any purpose that the routing supports.

Each of the internal signals is hardwired to a number of destinations. Even though a signal might be connected to a circuit, the circuit may not be "listening" to the signal. Each circuit has its own control registers which tells it to use a given signal or not. Figure 5-3 shows all the internal circuits that each internal signal is connected to.

Figure 5-4 shows the details of the encoder handler block that is part of Figure 5-3.

The Filter is used to remove unwanted noise on the incoming signal. The filter is programmable and it will "swallow" a pulse shorter than the programmed size.

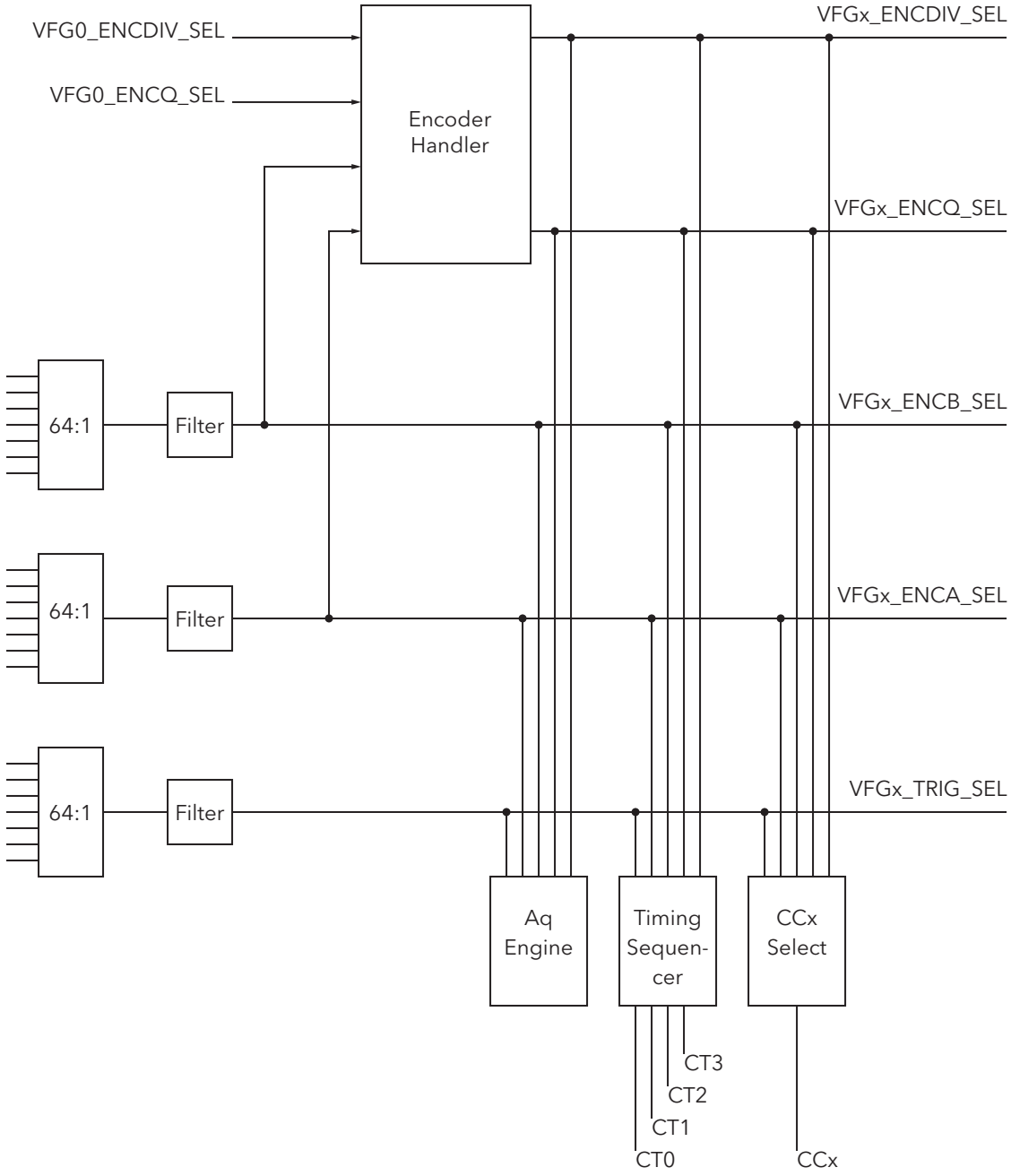

Figure 5-3 Internal Signal Routing

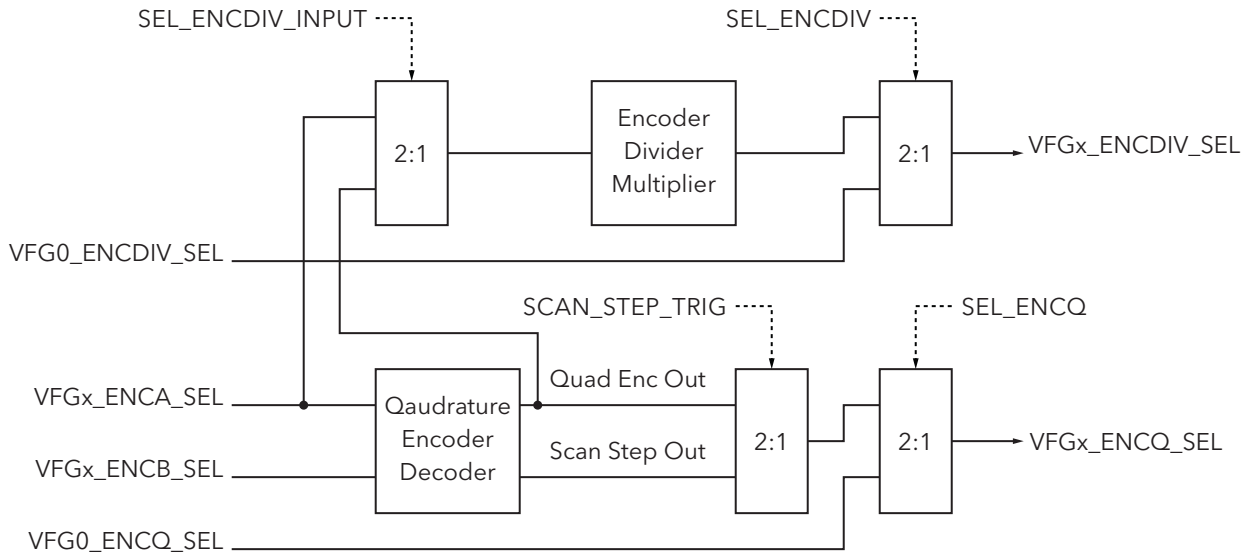

Figure 5-4 Encoder Handler

5.5 Output Signal Selection

There are four dynamic output signals for each VFG: CC1, CC2, CC3 and CC4. These dynamic signals can be driven by a variety of sources as shown in Figure 5-5.

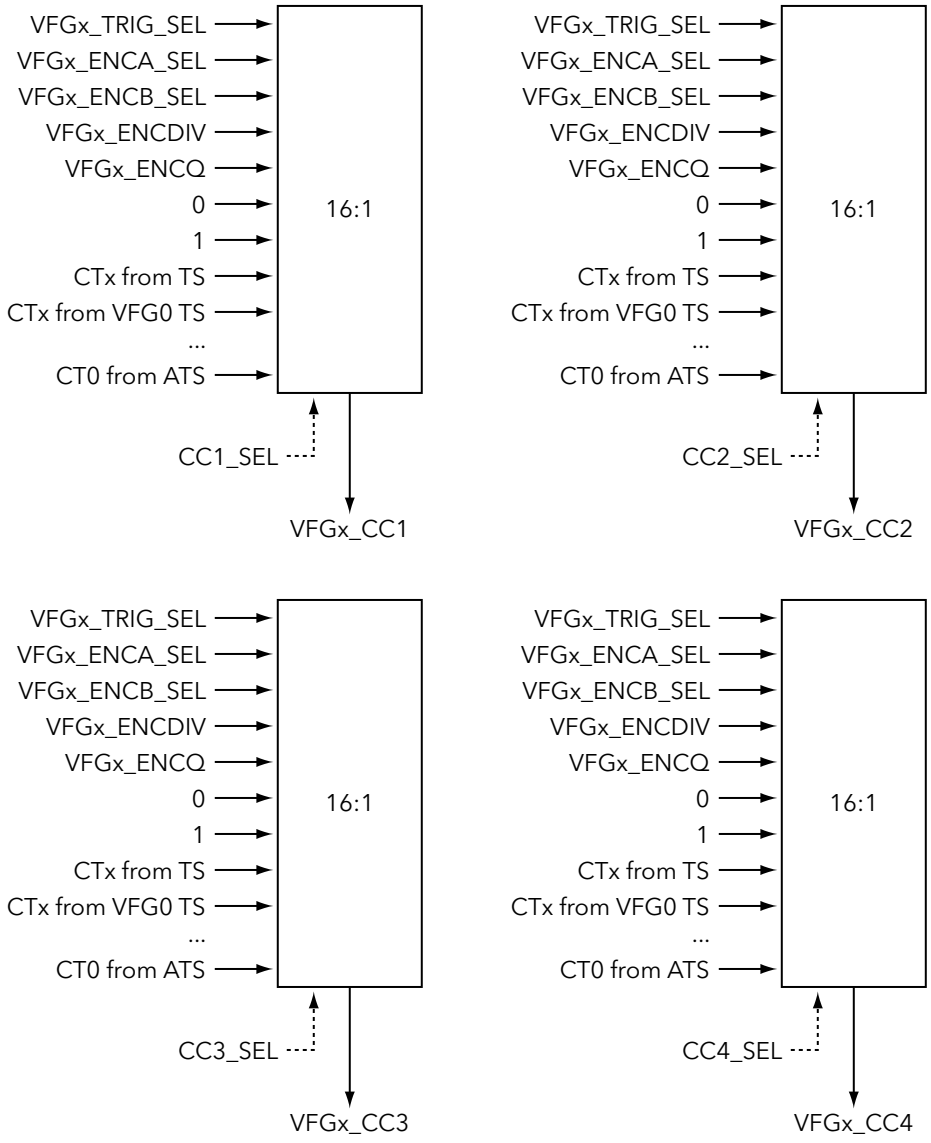

Figure 5-5 Output Signal Source Selection

5.6 I/O Connector Output Signal Routing

The CCx output signals are routed to the CXP link or the board's main I/O connector Figure 5-6 illustrates how they are routed.

Note: Each VFG has an instance of the circuit shown below. This means each VFG has its own CC2, CC3 and CC4 signals present on the board's main I/O connector.

Note: The signals VFGx_CC1 to VFGx_CC3 can be sourced from many different signals. Please see Section 5.5 for more information on selecting the source.

Note: The signals VFGx_CC1 to VFGx_CC3 can also be routed to the BitBox, which is an externally mountable I/O Box.

Figure 5-6 Output Signal Routing

5.7 BitBox Output Signal Routing

The BitFlow BitBox has 3 banks of 12 outputs. One bank is TTL, one bank is differential and one bank is a mix of Optocoupled and Open Collector outputs. Each of the 36 outputs can be set to a static output level, or controlled by a dynamic source (waveform). Figure 5-7 shows how the outputs are controlled.

Note: Figure 5-7 shows the choices and routing for just 3 outputs. A total of 12 of these circuits (each with their own control registers) are provided on the board in order to control 36 outputs.

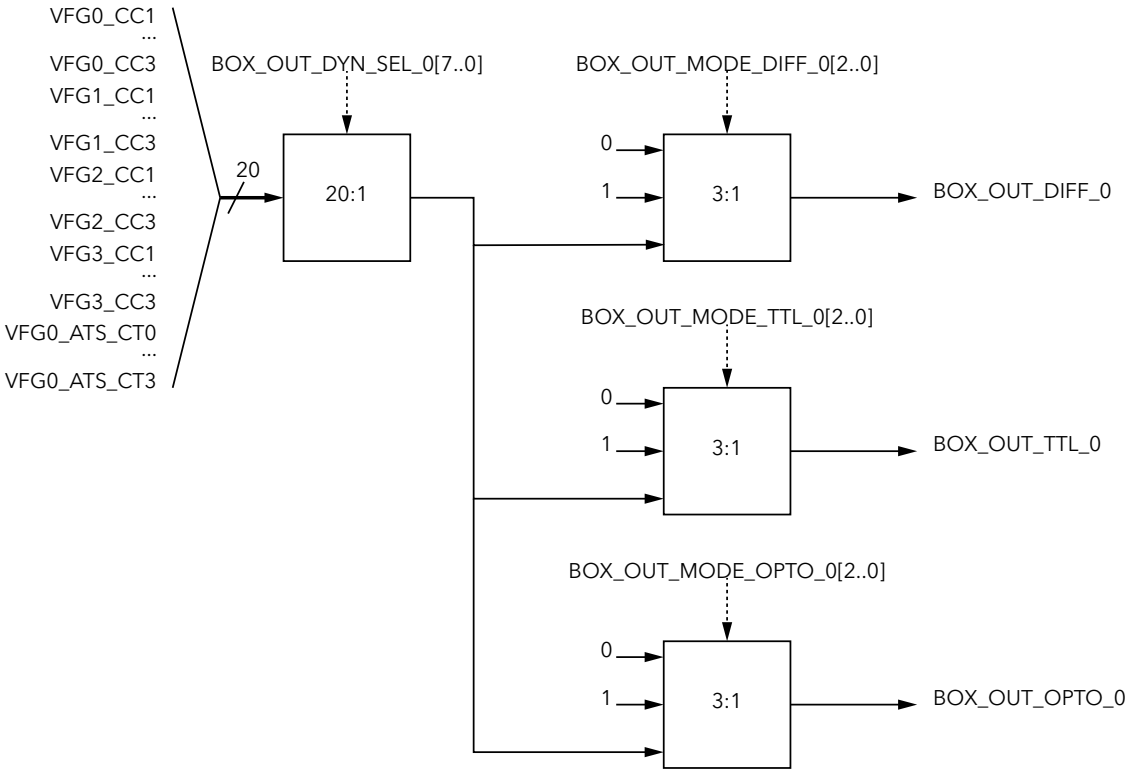

Figure 5-7 BitBox Output Signal Routing

Note: The BitFlow BitBox is an externally rail mounted box which can take a wide variety of inputs and outputs. It is connected to the Cyton/Axion though a small cable. Contact BitFlow for more information on the BitBox.

The I/O System Registers

Chapter 6

6.1 Introduction

The registers documented in this section are used to control the I/O system on the Aon, Axion, Claxon and Cyton.

6.2 CON60

Bit	Name
0	RD_BOX_IN_TTL_0
1	RD_BOX_IN_TTL_1
2	RD_BOX_IN_TTL_2
3	RD_BOX_IN_TTL_3
4	RD_BOX_IN_TTL_4
5	RD_BOX_IN_TTL_5
6	RD_BOX_IN_TTL_6
7	RD_BOX_IN_TTL_7
8	RD_BOX_IN_TTL_8
9	RD_BOX_IN_TTL_9
10	RD_BOX_IN_TTL_10
11	RD_BOX_IN_TTL_11
12	RD_BOX_IN_DIF_0
13	RD_BOX_IN_DIF_1
14	RD_BOX_IN_DIF_2
15	RD_BOX_IN_DIF_3
16	RD_BOX_IN_DIF_4
17	RD_BOX_IN_DIF_5
18	RD_BOX_IN_DIF_6
19	RD_BOX_IN_DIF_7
20	RD_BOX_IN_DIF_8
21	RD_BOX_IN_DIF_9
22	RD_BOX_IN_DIF_10
23	RD_BOX_IN_DIF_11
24	ENINT_CXP
25	INT_CXP
26	Reserved
27	Reserved
28	Reserved
29	SW_TRIG
30	SW_ENCA
31	SW_ENCB

RD_BOX_IN_TTL_X RO, CON60[11..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

These bits reflect the real-time state of the 12 TTL inputs on the BitBox.

RD_BOX_IN_DIF_X RO, CON60[23..12], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

These bits reflect the real-time state of the 12 differential inputs on the BitBox.

ENINT_CXP R/W, CON60[24], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit enables interrupts from the CXP subsystem.

INT_CXP RO, CON60[25], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit indicates the existence of an interrupt from the CXP subsystem. The individual interrupt must be cleared in the CXP subsystem in order for this bit to reset.

SW_TRIG R/W, CON60[29], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Writing the bit to 1 causes the internal software trigger signal to be asserted. Writing it to a 0 will de-assert the internal software trigger signal.

SW_ENCA R/W, CON60[30], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Writing the bit to 1 causes the internal encoder A signal to be asserted. Writing it to a 0 will de-assert the internal encoder A signal.

SW_ENCB R/W, CON60[31], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Writing the bit to 1 causes the internal encoder B signal to be asserted. Writing it to a 0 will de-assert the internal encoder B signal.

6.3 CON61

Bit	Name
0	RD_BOX_IN_OPTO_0
1	RD_BOX_IN_OPTO_1
2	RD_BOX_IN_OPTO_2
3	RD_BOX_IN_OPTO_3
4	RD_BOX_IN_OPTO_4
5	RD_BOX_IN_OPTO_5
6	RD_BOX_IN_OPTO_6
7	RD_BOX_IN_OPTO_7
8	RD_BOX_IN_24V_0
9	RD_BOX_IN_24V_1
10	RD_BOX_IN_24V_2
11	RD_BOX_IN_24V_3
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	RD_CXP_TRIG_OUT
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

RD_BOX_IN_OPTO_X RO, CON61[7..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

These bits reflect the real-time state of the eight Opto-Isolated inputs on the BitBox.

RD_BOX_IN_24V_X RO, CON61[11..8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

These bits reflect the real-time state of the four 24V inputs on the BitBox.

RD_CXP_TRIG_OUT RO, CON61[23], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of the CXP trigger signal going to the camera.

6.4 CON62

Bit	Name
0	RD_TRIG_TTL
1	RD_TRIG_DIF
2	RD_TRIG_VFG0
3	RD_SCAN_STEP
4	RD_SW_TRIG
5	RD_ENCA_TTL
6	RD_ENCA_DIF
7	RD_ENCA_VFG0
8	RD_ENCA_SW
9	RD_ENCB_TTL
10	RD_ENCB_DIF
11	RD_ENCB_VFG0
12	RD_ENCB_SW
13	RD_BUTTON
14	Reserved
15	DIV_RESET_DISABLE
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	RD_CXP_TRIG_IN
25	EN_TRIG
26	EN_ENCA
27	EN_ENCB
28	Reserved
29	RD_ENCB_SELECTED
30	RD_ENCA_SELECTED
31	RD_TRIG_SELECTED

RD_TRIG_TTL RO, CON62[0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of the VFG's TTL trigger input.

RD_TRIG_DIF RO, CON62[1], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of the VFG's differential trigger input.

RD_TRIG_VFG0 RO, CON62[2], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of VFG0's selected trigger signal.

RD_SCAN_STEP RO, CON62[3], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of the VFG's scan step circuitry output (from the quadrature encoder circuit).

RD_SW_TRIG RO, CON62[4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of the VFG's software trigger.

RD_ENCA_TTL RO, CON62[5], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of the VFG's TTL encoder A input.

RD_ENCA_DIF RO, CON62[6], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of the VFG's differential encoder A input.

RD_ENCA_VFG0 RO, CON62[7], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of VFG0's selected encoder A signal.

RD_ENCA_SW RO, CON62[8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of the VFG's software encoder A.

RD_ENCB_TTL RO, CON62[9], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of the VFG's TTL encoder B input.

RD_ENCB_DIF RO, CON62[10], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of the VFG's differential encoder B input.

RD_ENCB_VFG0 RO, CON62[11], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of VFG0's selected encoder B signal.

RD_ENCB_SW RO, CON62[12], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of the VFG's software encoder B.

RD_BUTTON RO, CON62[13], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of the VFG's button input.

DIV_RESET_DISABLE R/W, CON62[15], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit disables the Divider/Multiplier auto-reset function.

RD_CXP_TRIG_IN RO, CON62[24], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit reflects the real-time state of the CXP trigger signal coming from the camera.

EN_TRIG R/W, CON62[25], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit is used to enable the selected trigger.

EN_ENCA R/W, CON62[26], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit is used to enable the selected encoder A.

EN_ENCB R/W, CON62[27], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit is used to enable the selected encoder B.

RD_ENCB_SELECTED RO, CON62[29], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The bit reflects the real-time status of the VFG's select encoder B input.

RD_ENCA_SELECTED RO, CON62[30], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The bit reflects the real-time status of the VFG's selected encoder A input.

RD_TRIG_SELECTED RO, CON62[31], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The bit reflects the real-time status of the VFG's selected trigger input.

6.5 CON63

Bit	Name
0	SEL_TRIG
1	SEL_TRIG
2	SEL_TRIG
3	SEL_TRIG
4	SEL_TRIG
5	SEL_TRIG
6	SEL_ENCA
7	SEL_ENCA
8	SEL_ENCA
9	SEL_ENCA
10	SEL_ENCA
11	SEL_ENCA
12	SEL_ENCB
13	SEL_ENCB
14	SEL_ENCB
15	SEL_ENCB
16	SEL_ENCB
17	SEL_ENCB
18	SEL_CC1
19	SEL_CC1
20	SEL_CC1
21	SEL_CC1
22	SEL_CC2
23	SEL_CC2
24	SEL_CC2
25	SEL_CC2
26	Reserved
27	Reserved
28	SEL_LED
29	SEL_LED
30	SEL_LED
31	SEL_LED

SEL_TRIG R/W, CON63[5..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects the VFG's trigger source.

SEL_TRIG	Source
0 (000000b)	Forced low
1 (000001b)	Forced high
2 (000010b)	This VFG's differential trigger VFGx_TRIGGER+/-
3 (000011b)	This VFG's TTL trigger VFGx_TRIGGER_TTL
4 (000100b)	Selected trigger from VFG0
5 (000101b)	This VFG's Timing Sequencer (TS)
6 (000110b)	Button
7 (000111b)	The camera's CXP trigger
8 (001000b)	This VFG's software trigger, SW_TRIG
9 (001001b)	This VFG's scan step circuit
10 (001010b)	VFG0's Timing Sequencer (TS)
11-27	Reserved
28 to 39	BOX_IN_TTL_0 to BOX_IN_TTL_11
40 to 51	BOX_IN_DIF_0 to BOX_IN_DIF_11
52 to 59	BOX_IN_OPTO_0 to BOX_IN_OPTO_7
61 to 63	BOX_IN_24V_0 to BOX_IN_24V_3

SEL_ENCA R/W, CON63[11..6], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects this VFG's encoder A source.

SEL_ENCA	Source
0 (000000b)	Forced low
1 (000001b)	Forced high
2 (000010b)	This VFG's differential encoder A, VFGx_ENCA+/-
3 (000011b)	This VFG's TTL encoder A, VFGx_ENCA_TTL
4 (000100b)	Selected encoder A from VFG0
5 (000101b)	This VFG's Timing Sequencer (TS)
6 (000110b)	Button
7 (000111b)	The camera's CXP trigger

SEL_ENCA	Source
8 (001000b)	This VFG's software encoder A, SW_ENCA
9 (001001b)	VFG0's Timing Sequencer (TSS
10-27	Reserved
28 to 39	BOX_IN_TTL_0 to BOX_IN_TTL_11
40 to 51	BOX_IN_DIF_0 to BOX_IN_DIF_11
52 to 59	BOX_IN_OPTO_0 to BOX_IN_OPTO_7
61 to 63	BOX_IN_24V_0 to BOX_IN_24V_3

SEL_ENCB R/W, CON63[17..12], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects the source of encoder B.

SEL_ENCB	Source
0 (000000b)	Forced low
1 (000001b)	Forced high
2 (000010b)	This VFG's differential encoder B VFGx_ENCB+/-
3 (000011b)	This VFG's TTL encoder B VFGx_ENCB_TTL
4 (000100b)	Selected encoder B from VFG0
5 (000101b)	This VFG's Timing Sequencer (TS)
6 (000110b)	Button
7 (000111b)	The camera's CXP trigger
8 (001000b)	This VFG's software encoder B, SW_ENCB
9 (001001b)	VFG0's Timing Sequencer (TS)
10-27	Reserved
28 to 39	BOX_IN_TTL_0 to BOX_IN_TTL_11
40 to 51	BOX_IN_DIF_0 to BOX_IN_DIF_11
52 to 59	BOX_IN_OPTO_0 to BOX_IN_OPTO_7
61 to 63	BOX_IN_24V_0 to BOX_IN_24V_3

SEL_CC1 R/W, CON63[21..18], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects the source of CC1.

SEL_CC1	Source
0 (0000b)	Forced low
1 (0001b)	Forced high
2 (0010b)	CT0 (from TS)
3 (0011b)	CT1 (from TS)
4 (0100b)	CT2 (from TS)
5 (0101b)	CT3 (from TS)
6 (0110b)	VFGx_TRIG_SEL
7 (0111b)	VFGx_ENCA_SEL
8 (1000b)	VFGx_ENCB_SEL
9 (1001b)	VFG0_CT0
10 (1010b)	VFG0_CT1
11 (1011b)	VFG0_CT2
12 (1100b)	VFG0_CT3
13 (1101b)	VFGx_ENCDIV_SEL
14 (1110b)	VFGx_ENCQ_SEL
15 (111b)	CT0 (from ATS)

SEL_CC2 R/W, CON63[25..22], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects the source of CC2.

SEL_CC2	Source
0 (0000b)	Forced low
1 (0001b)	Forced high
2 (0010b)	CT0 (from TS)
3 (0011b)	CT1 (from TS)
4 (0100b)	CT2 (from S)
5 (0101b)	CT3 (from TS)
6 (0110b)	VFGx_TRIG_SEL
7 (0111b)	VFGx_ENCA_SEL

SEL_CC2	Source
8 (1000b)	VFGx_ENCB_SEL
9 (1001b)	VFG0_CT0
10 (1010b)	VFG0_CT1
11 (1011b)	VFG0_CT2
12 (1100b)	VFG0_CT3
13 (1101b)	VFGx_ENCDIV_SEL
14 (1110b)	VFGx_ENCQ_SEL
15 (111b)	CT0 (from ATS)

SEL_LED R/W, CON63[31..28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects the source of the VFG's green LED. The LED receives a 1/2 second pulse every time the selected event occurs.

SEL_LED	Source
0 (0000b)	Board emits an interrupt to the host
1 (0001b)	VFGx_TRIG_SEL
2 (0010b)	VFG0_TRIG_SEL
3 (0011b)	Button
4 (0100b)	FVAL from camera
5 (0101b)	VAW
6 (0110b)	VWIN
7 (0111b)	CC1
8 (1000b)	CC2
9 (1001b)	CC3
10 (1010b)	CC4
11 (1011b)	VFGx_NTG or VFGx_TS
12 (1100b)	VFG0_NTG or VFG0_TS
13 (1101b)	AQSTAT[1]
14 (1110b)	Overstep, OVS
15 (1111b)	Reserved

6.6 CON64

Bit	Name
0	SEL_CC3
1	SEL_CC3
2	SEL_CC3
3	SEL_CC3
4	SEL_CC4
5	SEL_CC4
6	SEL_CC4
7	SEL_CC4
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	TRIGPOL
14	ENCA_POL
15	ENCB_POL
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	LED_RED
29	LED_ORANGE
30	LED_GREEN
31	LED_BLUE

SEL_CC3 R/W, CON64[3..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects the source of CC3.

SEL_CC3	Source
0 (0000b)	Forced low
1 (0001b)	Forced high
2 (0010b)	CT0 (from TS)
3 (0011b)	CT1 (from TS)
4 (0100b)	CT2 (from TS)
5 (0101b)	CT3 (from TS)
6 (0110b)	VFGx_TRIG_SEL
7 (0111b)	VFGx_ENCA_SEL
8 (1000b)	VFGx_ENCB_SEL
9 (1001b)	VFG0_CT0
10 (1010b)	VFG0_CT1
11 (1011b)	VFG0_CT2
12 (1100b)	VFG0_CT3
13 (1101b)	VFGx_ENCDIV_SEL
14 (1110b)	VFGx_ENCQ_SEL
15 (111b)	CT0 (from ATS)

SEL_CC4 R/W, CON64[7..4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects the source of CC4.

SEL_CC4	Source
0 (0000b)	Forced low
1 (0001b)	Forced high
2 (0010b)	CT0 (from TS)
3 (0011b)	CT1 (from TS)
4 (0100b)	CT2 (from TS)
5 (0101b)	CT3 (from TS)
6 (0110b)	VFGx_TRIG_SEL
7 (0111b)	VFGx_ENCA_SEL

SEL_CC4	Source
8 (1000b)	VFGx_ENCB_SEL
9 (1001b)	VFG0_CT0
10 (1010b)	VFG0_CT1
11 (1011b)	VFG0_CT2
12 (1100b)	VFG0_CT3
13 (1101b)	VFGx_ENCDIV_SEL
14 (1110b)	VFGx_ENCQ_SEL
15 (111b)	CT0 (from ATS)

TRIGPOL R/W, CON64[13], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects the edge of the trigger signal that corresponds to its assertion.

TRIGPOL	Meaning
0	Trigger asserted on rising edge
1	Trigger asserted on falling edge

ENCA_POL R/W, CON64[14], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects the edge of encoder A signal that corresponds to its assertion.

ENCA_POL	Meaning
0	Encoder A asserted on rising edge
1	Encoder A asserted on falling edge

ENCB_POL R/W, CON64[15], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Selects the edge of encoder B signal that corresponds to its assertion.

ENCB_POL	Meaning
0	Encoder B asserted on rising edge
1	Encoder B asserted on falling edge

LED_RED R/W, CON64[28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Setting this bit to 1 turns the red LED on. Setting this bit to 1 on any VFG turns the LED on. This bit must be set to 0 on all VFGs in order to turn this LED off.

LED_ORANGE R/W, CON64[29], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Setting this bit to 1 turns the orange LED on. Setting this bit to 1 on any VFG turns the LED on. This bit must be set to 0 on all VFGs in order to turn this LED off.

LED_GREEN R/W, CON64[30], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Setting this bit to 1 turns the green LED on. Setting this bit to 1 on any VFG turns the LED on. This bit must be set to 0 on all VFGs in order to turn this LED off.

6.7 TRIG_OPTS

Bit	Name
0	TRIG_FILTER
1	TRIG_FILTER
2	TRIG_FILTER
3	TRIG_FILTER
4	TRIG_FILTER
5	TRIG_FILTER
6	TRIG_FILTER
7	TRIG_FILTER
8	TRIG_FILTER
9	TRIG_FILTER
10	TRIG_FILTER
11	TRIG_FILTER
12	TRIG_FILTER
13	TRIG_FILTER
14	TRIG_FILTER
15	TRIG_FILTER
16	TRIG_FILTER
17	TRIG_FILTER
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

TRIG_FILTER RO, TRIG_OPTS[17..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The trigger circuit includes a programmable noise filter. The value of this register controls the size of the noise pulse that will be considered noise and will be filtered out. Any pulses over this size will be consider signal. The units of this register are 4 nanoseconds. If this register is programmed to 0, nothing will be filtered out. If this register is programmed to 0x3ffff, pulses of up to 1 millisecond will be removed.

6.8 ENCA_OPTS

Bit	Name
0	ENCA_FILTER
1	ENCA_FILTER
2	ENCA_FILTER
3	ENCA_FILTER
4	ENCA_FILTER
5	ENCA_FILTER
6	ENCA_FILTER
7	ENCA_FILTER
8	ENCA_FILTER
9	ENCA_FILTER
10	ENCA_FILTER
11	ENCA_FILTER
12	ENCA_FILTER
13	ENCA_FILTER
14	ENCA_FILTER
15	ENCA_FILTER
16	ENCA_FILTER
17	ENCA_FILTER
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

ENCA_FILTER RO, ENCA_OPTS[17..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The encoder A circuit includes a programmable noise filter. The value of this register controls the size of the noise pulse that will be considered noise and will be filtered out. Any pulses over this size will be consider signal. The units of this register are 4 nanoseconds. If this register is programmed to 0, nothing will be filtered out. If this register is programmed to 0x3ffff, pulses of up to 1 millisecond will be removed.

6.9 ENCB_OPTS

Bit	Name
0	ENCB_FILTER
1	ENCB_FILTER
2	ENCB_FILTER
3	ENCB_FILTER
4	ENCB_FILTER
5	ENCB_FILTER
6	ENCB_FILTER
7	ENCB_FILTER
8	ENCB_FILTER
9	ENCB_FILTER
10	ENCB_FILTER
11	ENCB_FILTER
12	ENCB_FILTER
13	ENCB_FILTER
14	ENCB_FILTER
15	ENCB_FILTER
16	ENCB_FILTER
17	ENCB_FILTER
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

ENCB_FILTER RO, ENCB_OPTS[17..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The encoder B circuit includes a programmable noise filter. The value of this register controls the size of the noise pulse that will be considered noise and will be filtered out. Any pulses over this size will be consider signal. The units of this register are 4 nanoseconds. If this register is programmed to 0, nothing will be filtered out. If this register is programmed to 0x3ffff, pulses of up to 1 millisecond will be removed.

6.10 BOX_OUT_DYN_SEL_SET_A

Bit	Name
0	BOX_OUT_DYN_SEL_0
1	BOX_OUT_DYN_SEL_0
2	BOX_OUT_DYN_SEL_0
3	BOX_OUT_DYN_SEL_0
4	BOX_OUT_DYN_SEL_0
5	BOX_OUT_DYN_SEL_0
6	BOX_OUT_DYN_SEL_0
7	BOX_OUT_DYN_SEL_0
8	BOX_OUT_DYN_SEL_1
9	BOX_OUT_DYN_SEL_1
10	BOX_OUT_DYN_SEL_1
11	BOX_OUT_DYN_SEL_1
12	BOX_OUT_DYN_SEL_1
13	BOX_OUT_DYN_SEL_1
14	BOX_OUT_DYN_SEL_1
15	BOX_OUT_DYN_SEL_1
16	BOX_OUT_DYN_SEL_2
17	BOX_OUT_DYN_SEL_2
18	BOX_OUT_DYN_SEL_2
19	BOX_OUT_DYN_SEL_2
20	BOX_OUT_DYN_SEL_2
21	BOX_OUT_DYN_SEL_2
22	BOX_OUT_DYN_SEL_2
23	BOX_OUT_DYN_SEL_2
24	BOX_OUT_DYN_SEL_3
25	BOX_OUT_DYN_SEL_3
26	BOX_OUT_DYN_SEL_3
27	BOX_OUT_DYN_SEL_3
28	BOX_OUT_DYN_SEL_3
29	BOX_OUT_DYN_SEL_3
30	BOX_OUT_DYN_SEL_3
31	BOX_OUT_DYN_SEL_3

BOX_OUT_DYN_SEL_0 R/W, BOX_OUT_DYN_SEL_SET_A[7..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register controls the dynamic source for the three BitBox output signals BOX_OUT_DIFF_0, BOX_OUT_TTL_0 and BOX_OUT_OPTO_0. These three outputs always have the same dynamic source (they can individually be set to static high or low value). The following table lists all the possible sources.

Table 6-1 BitBox Dynamic Output Source Selections

BOX_OUT_DYN_SEL_0	Source
0 (00000b)	VFG0_CC1
1 (00001b)	VFG0_CC2
2 (00010b)	VFG0_CC3
3 (00011b)	VFG0_CC4
4 (00100b)	VFG1_CC1
5 (00101b)	VFG1_CC2
6 (00110b)	VFG1_CC3
7 (00111b)	VFG1_CC4
8 (01000b)	VFG2_CC1
9 (01001b)	VFG2_CC2
10 (01010b)	VFG2_CC3
11 (01011b)	VFG2_CC4
12 (01100b)	VFG3_CC1
13 (01101b)	VFG3_CC2
14 (01110b)	VFG3_CC3
15 (01111b)	VFG3_CC4
16 (10000b)	VFG0_ATS_CT0
17 (10001b)	VFG0_ATS_CT1
18 (10010b)	VFG0_ATS_CT2
19 (10011b)	VFG0_ATS_CT3
20 to 256	Reserved

Note: Not all of the sources above are available on all models. For example, the Axion 1xE only has one VFG, so sources with the prefix "VFG1", "VFG2" and "VFG3" will not be available. Similarly, a two VFG model (e.g. Axion-2xB) will not have the sources that start with "VF2" or "VFG3".

BOX_OUT_DYN_SEL_1 R/W, BOX_OUT_DYN_SEL_SET_A[15..8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register controls the dynamic source for the three BitBox output signals BOX_OUT_DIFF_1, BOX_OUT_TTL_1 and BOX_OUT_OPTO_1. These three outputs always have the same dynamic source (they can individually be set to static high or low value). Table 6-1 lists all the possible sources.

BOX_OUT_DYN_SEL_2 R/W, BOX_OUT_DYN_SEL_SET_A[23..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register controls the dynamic source for the three BitBox output signals BOX_OUT_DIFF_2, BOX_OUT_TTL_2 and BOX_OUT_OPTO_2. These three outputs always have the same dynamic source (they can individually be set to static high or low value). Table 6-1 lists all the possible sources.

BOX_OUT_DYN_SEL_3 R/W, BOX_OUT_DYN_SEL_SET_A[31..24], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register controls the dynamic source for the three BitBox output signals BOX_OUT_DIFF_3, BOX_OUT_TTL_3 and BOX_OUT_OPTO_3. These three outputs always have the same dynamic source (they can individually be set to static high or low value). Table 6-1 lists all the possible sources.

6.11 BOX_OUT_DYN_SEL_SET_B

Bit	Name
0	BOX_OUT_DYN_SEL_4
1	BOX_OUT_DYN_SEL_4
2	BOX_OUT_DYN_SEL_4
3	BOX_OUT_DYN_SEL_4
4	BOX_OUT_DYN_SEL_4
5	BOX_OUT_DYN_SEL_4
6	BOX_OUT_DYN_SEL_4
7	BOX_OUT_DYN_SEL_4
8	BOX_OUT_DYN_SEL_5
9	BOX_OUT_DYN_SEL_5
10	BOX_OUT_DYN_SEL_5
11	BOX_OUT_DYN_SEL_5
12	BOX_OUT_DYN_SEL_5
13	BOX_OUT_DYN_SEL_5
14	BOX_OUT_DYN_SEL_5
15	BOX_OUT_DYN_SEL_5
16	BOX_OUT_DYN_SEL_6
17	BOX_OUT_DYN_SEL_6
18	BOX_OUT_DYN_SEL_6
19	BOX_OUT_DYN_SEL_6
20	BOX_OUT_DYN_SEL_6
21	BOX_OUT_DYN_SEL_6
22	BOX_OUT_DYN_SEL_6
23	BOX_OUT_DYN_SEL_6
24	BOX_OUT_DYN_SEL_7
25	BOX_OUT_DYN_SEL_7
26	BOX_OUT_DYN_SEL_7
27	BOX_OUT_DYN_SEL_7
28	BOX_OUT_DYN_SEL_7
29	BOX_OUT_DYN_SEL_7
30	BOX_OUT_DYN_SEL_7
31	BOX_OUT_DYN_SEL_7

BOX_OUT_DYN_SEL_4 R/W, BOX_OUT_DYN_SEL_SET_B[7..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register controls the dynamic source for the three BitBox output signals BOX_OUT_DIFF_4, BOX_OUT_TTL_4 and BOX_OUT_OPTO_4. These three outputs always have the same dynamic source (they can individually be set to static high or low value). Table 6-1 lists all the possible sources.

BOX_OUT_DYN_SEL_5 R/W, BOX_OUT_DYN_SEL_SET_B[15..8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register controls the dynamic source for the three BitBox output signals BOX_OUT_DIFF_5, BOX_OUT_TTL_5 and BOX_OUT_OPTO_5. These three outputs always have the same dynamic source (they can individually be set to static high or low value). Table 6-1 lists all the possible sources.

BOX_OUT_DYN_SEL_6 R/W, BOX_OUT_DYN_SEL_SET_B[23..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register controls the dynamic source for the three BitBox output signals BOX_OUT_DIFF_6, BOX_OUT_TTL_6 and BOX_OUT_OPTO_6. These three outputs always have the same dynamic source (they can individually be set to static high or low value). Table 6-1 lists all the possible sources.

BOX_OUT_DYN_SEL_7 R/W, BOX_OUT_DYN_SEL_SET_B[31..24], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register controls the dynamic source for the three BitBox output signals BOX_OUT_DIFF_7, BOX_OUT_TTL_7 and BOX_OUT_OPTO_7. These three outputs always have the same dynamic source (they can individually be set to static high or low value). Table 6-1 lists all the possible sources.

6.12 BOX_OUT_DYN_SEL_SET_C

Bit	Name
0	BOX_OUT_DYN_SEL_8
1	BOX_OUT_DYN_SEL_8
2	BOX_OUT_DYN_SEL_8
3	BOX_OUT_DYN_SEL_8
4	BOX_OUT_DYN_SEL_8
5	BOX_OUT_DYN_SEL_8
6	BOX_OUT_DYN_SEL_8
7	BOX_OUT_DYN_SEL_8
8	BOX_OUT_DYN_SEL_9
9	BOX_OUT_DYN_SEL_9
10	BOX_OUT_DYN_SEL_9
11	BOX_OUT_DYN_SEL_9
12	BOX_OUT_DYN_SEL_9
13	BOX_OUT_DYN_SEL_9
14	BOX_OUT_DYN_SEL_9
15	BOX_OUT_DYN_SEL_9
16	BOX_OUT_DYN_SEL_10
17	BOX_OUT_DYN_SEL_10
18	BOX_OUT_DYN_SEL_10
19	BOX_OUT_DYN_SEL_10
20	BOX_OUT_DYN_SEL_10
21	BOX_OUT_DYN_SEL_10
22	BOX_OUT_DYN_SEL_10
23	BOX_OUT_DYN_SEL_10
24	BOX_OUT_DYN_SEL_11
25	BOX_OUT_DYN_SEL_11
26	BOX_OUT_DYN_SEL_11
27	BOX_OUT_DYN_SEL_11
28	BOX_OUT_DYN_SEL_11
29	BOX_OUT_DYN_SEL_11
30	BOX_OUT_DYN_SEL_11
31	BOX_OUT_DYN_SEL_11

BOX_OUT_DYN_SEL_8 R/W, BOX_OUT_DYN_SEL_SET_C[7..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register controls the dynamic source for the three BitBox output signals BOX_OUT_DIFF_8, BOX_OUT_TTL_8 and BOX_OUT_OC_0. These three outputs always have the same dynamic source (they can individually be set to static high or low value). Table 6-1 lists all the possible sources.

BOX_OUT_DYN_SEL_9 R/W, BOX_OUT_DYN_SEL_SET_C[15..8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register controls the dynamic source for the three BitBox output signals BOX_OUT_DIFF_9, BOX_OUT_TTL_9 and BOX_OUT_OC_1. These three outputs always have the same dynamic source (they can individually be set to static high or low value). Table 6-1 lists all the possible sources.

BOX_OUT_DYN_SEL_10 R/W, BOX_OUT_DYN_SEL_SET_C[23..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register controls the dynamic source for the three BitBox output signals BOX_OUT_DIFF_10, BOX_OUT_TTL_10 and BOX_OC_2. These three outputs always have the same dynamic source (they can individually be set to static high or low value). Table 6-1 lists all the possible sources.

BOX_OUT_DYN_SEL_11 R/W, BOX_OUT_DYN_SEL_SET_C[31..24], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register controls the dynamic source for the three BitBox output signals BOX_OUT_DIFF_11, BOX_OUT_TTL_11 and BOX_OUT_OC_3. These three outputs always have the same dynamic source (they can individually be set to static high or low value). Table 6-1 lists all the possible sources.

6.13 BOX_OUT_MODE_SET_A

Bit	Name
0	BOX_OUT_MODE_TTL_0
1	BOX_OUT_MODE_TTL_0
2	BOX_OUT_MODE_TTL_1
3	BOX_OUT_MODE_TTL_1
4	BOX_OUT_MODE_TTL_2
5	BOX_OUT_MODE_TTL_2
6	BOX_OUT_MODE_TTL_3
7	BOX_OUT_MODE_TTL_3
8	BOX_OUT_MODE_TTL_4
9	BOX_OUT_MODE_TTL_4
10	BOX_OUT_MODE_TTL_5
11	BOX_OUT_MODE_TTL_5
12	BOX_OUT_MODE_TTL_6
13	BOX_OUT_MODE_TTL_6
14	BOX_OUT_MODE_TTL_7
15	BOX_OUT_MODE_TTL_7
16	BOX_OUT_MODE_TTL_8
17	BOX_OUT_MODE_TTL_8
18	BOX_OUT_MODE_TTL_9
19	BOX_OUT_MODE_TTL_9
20	BOX_OUT_MODE_TTL_10
21	BOX_OUT_MODE_TTL_10
22	BOX_OUT_MODE_TTL_11
23	BOX_OUT_MODE_TTL_11
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

BOX_OUT_MODE_TTL_0 R/W, BOX_OUT_MODE_SET_A[1..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_TTL_0. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_0 bitfield.

BOX_OUT_MODE_TTL_0	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_TTL_1 R/W, BOX_OUT_MODE_SET_A[3..2], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_TTL_1. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_1 bitfield.

BOX_OUT_MODE_TTL_1	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_TTL_2 R/W, BOX_OUT_MODE_SET_A[5..4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_TTL_2. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_2 bitfield.

BOX_OUT_MODE_TTL_2	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_TTL_3 R/W, BOX_OUT_MODE_SET_A[7..6], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_TTL_3. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_3 bitfield.

BOX_OUT_MODE_TTL_3	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_TTL_4 R/W, BOX_OUT_MODE_SET_A[9..8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_TTL_4. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_4 bitfield.

BOX_OUT_MODE_TTL_$	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_TTL_5 R/W, BOX_OUT_MODE_SET_A[11..10], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_TTL_5. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_5 bitfield.

BOX_OUT_MODE_TTL_5	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_TTL_6 R/W, BOX_OUT_MODE_SET_A[13..12], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_TTL_6. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_6 bitfield.

BOX_OUT_MODE_TTL_6	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_TTL_7 R/W, BOX_OUT_MODE_SET_A[15..14], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_TTL_7. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_7 bitfield.

BOX_OUT_MODE_TTL_7	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_TTL_8 R/W, BOX_OUT_MODE_SET_A[17..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_TTL_8. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_8 bitfield.

BOX_OUT_MODE_TTL_8	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_TTL_9 R/W, BOX_OUT_MODE_SET_A[19..18], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_TTL_9. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_9 bitfield.

BOX_OUT_MODE_TTL_9	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_TTL_10 R/W, BOX_OUT_MODE_SET_A[21..20], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_TTL_10. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_10 bitfield.

BOX_OUT_MODE_TTL_10	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_TTL_11 R/W, BOX_OUT_MODE_SET_A[23..22], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_TTL_11. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_11 bitfield.

BOX_OUT_MODE_TTL_11	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

6.14 BOX_OUT_MODE_SET_B

Bit	Name
0	BOX_OUT_MODE_DIFF_0
1	BOX_OUT_MODE_DIFF_0
2	BOX_OUT_MODE_DIFF_1
3	BOX_OUT_MODE_DIFF_1
4	BOX_OUT_MODE_DIFF_2
5	BOX_OUT_MODE_DIFF_2
6	BOX_OUT_MODE_DIFF_3
7	BOX_OUT_MODE_DIFF_3
8	BOX_OUT_MODE_DIFF_4
9	BOX_OUT_MODE_DIFF_4
10	BOX_OUT_MODE_DIFF_5
11	BOX_OUT_MODE_DIFF_5
12	BOX_OUT_MODE_DIFF_6
13	BOX_OUT_MODE_DIFF_6
14	BOX_OUT_MODE_DIFF_7
15	BOX_OUT_MODE_DIFF_7
16	BOX_OUT_MODE_DIFF_8
17	BOX_OUT_MODE_DIFF_8
18	BOX_OUT_MODE_DIFF_9
19	BOX_OUT_MODE_DIFF_9
20	BOX_OUT_MODE_DIFF_10
21	BOX_OUT_MODE_DIFF_10
22	BOX_OUT_MODE_DIFF_11
23	BOX_OUT_MODE_DIFF_11
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

BOX_OUT_MODE_DIFF_0 R/W, BOX_OUT_MODE_SET_B[1..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_DIFF_0. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_0 bitfield.

BOX_OUT_MODE_DIFF_0	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_DIFF_1 R/W, BOX_OUT_MODE_SET_B[3..2], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_DIFF_1. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_1 bitfield.

BOX_OUT_MODE_DIFF_1	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_DIFF_2 R/W, BOX_OUT_MODE_SET_B[5..4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_DIFF_2. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_2 bitfield.

BOX_OUT_MODE_DIFF_2	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_DIFF_3 R/W, BOX_OUT_MODE_SET_B[7..6], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_DIFF_3 . This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_d bitfield.

BOX_OUT_MODE_DIFF_3	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_DIFF_4 R/W, BOX_OUT_MODE_SET_B[9..8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_DIFF_4. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_4 bitfield.

BOX_OUT_MODE_DIFF_4	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_DIFF_5 R/W, BOX_OUT_MODE_SET_B[11..10], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_DIFF_5. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_5 bitfield.

BOX_OUT_MODE_DIFF_5	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_DIFF_6 R/W, BOX_OUT_MODE_SET_B[13..12], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_DIFF_6. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_6 bitfield.

BOX_OUT_MODE_DIFF_6	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_DIFF_7 R/W, BOX_OUT_MODE_SET_B[15..14], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_DIFF_7. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_7 bitfield.

BOX_OUT_MODE_DIFF_7	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_DIFF_8 R/W, BOX_OUT_MODE_SET_B[17..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_DIFF_8. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_8 bitfield.

BOX_OUT_MODE_DIFF_8	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_DIFF_9 R/W, BOX_OUT_MODE_SET_B[19..18], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_DIFF_9. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_9bitfield.

BOX_OUT_MODE_DIFF_9	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_DIFF_10 R/W, BOX_OUT_MODE_SET_B[21..20], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_DIFF_10. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_10 bitfield.

BOX_OUT_MODE_DIFF_10	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_DIFF_11 R/W, BOX_OUT_MODE_SET_B[23..22], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_DIFF_11. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_11 bitfield.

BOX_OUT_MODE_DIFF_11	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

6.15 BOX_OUT_MODE_SET_C

Bit	Name
0	BOX_OUT_MODE_OPTO_0
1	BOX_OUT_MODE_OPTO_0
2	BOX_OUT_MODE_OPTO_1
3	BOX_OUT_MODE_OPTO_1
4	BOX_OUT_MODE_OPTO_2
5	BOX_OUT_MODE_OPTO_2
6	BOX_OUT_MODE_OPTO_3
7	BOX_OUT_MODE_OPTO_3
8	BOX_OUT_MODE_OPTO_4
9	BOX_OUT_MODE_OPTO_4
10	BOX_OUT_MODE_OPTO_5
11	BOX_OUT_MODE_OPTO_5
12	BOX_OUT_MODE_OPTO_6
13	BOX_OUT_MODE_OPTO_6
14	BOX_OUT_MODE_OPTO_7
15	BOX_OUT_MODE_OPTO_7
16	BOX_OUT_MODE_OC_0
17	BOX_OUT_MODE_OC_0
18	BOX_OUT_MODE_OC_1
19	BOX_OUT_MODE_OC_1
20	BOX_OUT_MODE_OC_2
21	BOX_OUT_MODE_OC_2
22	BOX_OUT_MODE_OC_3
23	BOX_OUT_MODE_OC_3
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

BOX_OUT_MODE_OPTO_0 R/W, BOX_OUT_MODE_SET_C[1..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_OPTO_0. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_0 bitfield.

BOX_OUT_MODE_OPTO_0	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_OPTO_1 R/W, BOX_OUT_MODE_SET_C[3..2], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_OPTO_1 This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_1 bitfield.

BOX_OUT_MODE_OPTO_1	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_OPTO_2 R/W, BOX_OUT_MODE_SET_C[5..4], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_OPTO_2 This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_2 bitfield.

BOX_OUT_MODE_OPTO_2	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_OPTO_3 R/W, BOX_OUT_MODE_SET_C[7..6], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_OPTO_3. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_3 bitfield.

BOX_OUT_MODE_XXX_3	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_OPTO_4 R/W, BOX_OUT_MODE_SET_C[9..8], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_OPTO_4. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_4 bitfield.

BOX_OUT_MODE_OPTO_4	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_OPTO_5 R/W, BOX_OUT_MODE_SET_C[11..10], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_OPTO_5. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_5 bitfield.

BOX_OUT_MODE_OPTO_5	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_OPTO_6 R/W, BOX_OUT_MODE_SET_C[13..12], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_OPTO_6. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_6 bitfield.

BOX_OUT_MODE_OPTO_6	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_OPTO_7 R/W, BOX_OUT_MODE_SET_C[15..14], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_OPTO_7. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_7 bitfield.

BOX_OUT_MODE_OPTO_7	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_OC_0 R/W, BOX_OUT_MODE_SET_C[17..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_OC_0 This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_8 bitfield.

BOX_OUT_MODE_OC_0	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_OC_1 R/W, BOX_OUT_MODE_SET_C[19..18], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_OC_1. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_9 bitfield.

BOX_OUT_MODE_OC_1	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_OC_2 R/W, BOX_OUT_MODE_SET_C[21..20], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_OC_2. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_10 bitfield.

BOX_OUT_MODE_OC_2	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

BOX_OUT_MODE_OC_3 R/W, BOX_OUT_MODE_SET_C[23..22], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls the BitBox output BOX_OUT_MODE_OC_3. This bit controls whether this output is static (high or low) or dynamic. If the bit is dynamic, the source is controlled by the associated BOX_OUT_DYN_SEL_11 bitfield.

BOX_OUT_MODE_OC_3	Meaning
0 (00b)	Static output low
1 (01b)	Static output high
2 (10b)	Dynamic output
3 (11b)	Reserved

Encoder Divider

Chapter 7

7.1 Introduction

This section covers the encoder divider which supported on the all of BitFlow's modern frame grabber families. The purpose of Encoder Divider is to provide the ability to use an encoder running at one rate to drive a line scan camera at a different rate. This circuit is only useful for line scan cameras. The Encoder Divider can scale up or down the incoming encoder frequency. The encoder divider is fully programmable and is easily controlled from software and/or from camera configuration files.

The factor used to scaled the incoming encoder frequency does not have to be a whole number. For example, the encoder could be scaled by 0.03448 or 4.2666). Of course, not all rational numbers in the available scaling range can be selected (there are an infinite number of them). However, a useful selection of values is available which should support most applications.

The Encoder Divider circuit takes as its input the selected encoder input. The output of the encoder divider drives the same parts of the board the normal encoder usually does. The actual circuit(s) being driven depends on how the board is programmed.

7.2 Encoder Divider Details

7.2.1 Formula

The following formula shows the equation used to scale the encoming encoder rate into the camera's line rate:

$$F_{out} = F_{in} \frac{2^N}{M}$$

W:

Fout = The frequency used to driver the camera or the NTG or the CTabs
Fin = The encoder (input) frequency
N = An integer between 0 and 6 (set by the register ENC_DIV_N)
M = An integer between 1 and 1023 (set by the register ENC_DIV_M)

The above formula provides an effective scaling factor from 0.001 (N = 0, M =1023) to 64 (N = 6, M = 1). Not every scaling factor can be achieved between these two extremes, and the scaling factors are not evenly distributed. However, a scaling factor can be generally found that meets the requirements of most applications.

7.2.2 Example

Let's assume that the encoder frequency (Fin) is 10 KHz and that we need an output (Fout) of ~30 KHz. This means that we need to multiply by 3. Set N = 6 and M = 21. This will a scaling factor of 3.048. The result is an effective line rate of 30.48 KHz.

7.2.3 Restrictions

Because the encoder divider uses a digital PLL run by a high frequency clock, not all encoder input frequencies can be accurately scaled. The PLL has been designed to work in most machine visions applications. Support, therefore, is provided for the following input frequency range:

Minimum input encoder frequency: 1.6 KHz
Maximum input encoder frequency: 300 KHz

7.2.4 PLL Locking

The encoder divider achieves its scaling using a PLL. By default the output waveform is locked to the input waveform. However, this locking can result in a small amount of jitter. To reduce the jitter, the output waveform can be run open loop. This mode is accessed by setting the register ENC_DIV_OPEN_LOOP to 1.

7.2.5 Handling Encoder Slow Down or Stopping

On some machine vision systems, the encoder is attached to a mechanism that may slow and/or stop. Any PLL has a limited range that it can track (based on the PLL master clock), outside of this range, the output signal can become unpredictable. The Encoder Divider circuit's master clock is 50 MHz, which makes the minimum frequency that it can accurately track around 1.6 KHz. In order to avoid this situation and handle encoder slow down/stop gracefully, the encoder divider has has limiting circuit that can be run in one of two different mode described in the following two sections.

Slow Tracking Mode (ENC_DIC_FORCE_DC = 0)

In this mode, when the input frequency goes below the minimum of 1.6 KHz, the Encoder Divider circuit's output will continue to track the input, but the output frequency will become simple divider on the input freqency. In this mode the output will track the input using the following formula (the variables are the same as in Section 7.2.1)

$$F_{out} = \frac{F_{in}}{4M}$$

DC Mode (ENC_DIV_FORCE_DC = 1)

In this mode, the Fin goes below 1.6 KHz, Fout will goes to DC. This means that when the input frequency goes below the minimum the camera will be frozen, acquisition will stop. The board will stay in this state until Fin goes above 1.6 KHz. This is useful when the encoder is being driven by a stage that is traveling back and forth. At both ends of travel when the stage changes directions, the board will not acquire.

7.3 Encoder Divider Control Registers

The following table summarizes the registers:

Table 7-1 Encoder Divider Registers

Name	Purpose
ENC_DIV_M	This controls the M factor in the Encoder Divider equation (see Section 7.2.1)
ENC_DIV_N	The controls the N factor the Encoder Divider equation
ENC_DIV_FORCE_DC	Controls the behavior when Fin falls below the minimum. 0 = Output runs in simple divider mode. 1 = Output goes to DC.
ENC_DIV_OPEN_LOOP	Controls whether the output signal phase of the Encoder Divider is lock to the intput or is allowed to free run. 0 = Output phased locked to input. 1 = Ouput runs open loop.
ENC_DIV_FCLK_SEL	Reserved for future support for alternate Encoder Divider PLL Master clock frequencies. Currently must be set to 0, which selects 50 MHz clock.

See Chapter 9 for details on the registers needed to control the encoder divider system.

Quadrature Encoder

Chapter 8

8.1 Introduction

This section discusses support for quadrature encoders. A quadrature encoder is an encoder that outputs two signals A and B. Both signals are used as a line trigger. However, the signals are 90 degrees out of phase. By comparing the A and B signals, the direction of the encoder motion can be determined. There are a number of ways that quadrature encoders can be used to control acquisition. The following sections cover all of the support methods.

Most of the quadrature encoder system is based around a 24-bit counter. This normally starts at zero and then counts up or down every time the encoder moves. The counter can be observed at any time via the QENC_COUNT register. This registers is the heart of the encoder system. For example, triggers values can be programmed to start and end acquisition of lines. Also, as the counter tracks the motion of the stage attached to the encoder exactly, the system can be programmed to acquire forward only or backward only stage movements. The system can be programmed to only acquire one line for each encoder count that corresponds to a physical location on the stage. The encoder counter can be used in many different ways, described in more details below.

8.1.1 Simple Encoder Mode

The most basic method of using a quadrature encoder is to use it like a standard signal phase encoder. In this mode, the quadrature encoder provides a higher resolution signal, as both the A and B signals can be used to trigger lines. Also, by setting QENC_DECODE = 1, both the rising and the falling edges of both the A and B signals are used to trigger lines, providing a 4x increase in resolution over a signal phase encoder.

In this mode, every encoder edge triggers a line, the direction information from the encoder is ignored.

8.1.2 Positive or Negative Only Acquisition

The board can be programmed to only acquired lines when the encoder moves forward (increase the encoder count in a positive direction) or moves backwards (decrease the encoder count in a negative direction). This mode is useful in situations where a stage is moving back and forth, and lines need only be acquired while the stage is moving in one direction, and not the other direction. The direction of acquisition is controlled by the QENC_AQ_DIR register.

8.1.3 Interval Mode

Often in situations when a stage is moving back and forth, acquisition is only required over a subsection of the total stage range. Interval mode has been designed for these situations. When the board is in interval mode, it only acquires lines when the encoder counter is between a lower limit and an upper limit. If the counter is outside these limits, lines are not acquired.

To use interval mode, set QENC_INTRVL_MODE = 1, and program QENC_INTRVL_LL and QENC_INTRVL_UL to the encoder ranges that bracket the section of your stage range that you wish to acquire. Interval mode can be used in conjunction with QENC_AQ_DIR to acquire lines passing over the interval in the positive direction, the negative direction or both directions.

8.1.4 Re-Acquisition Prevention

Encoders are usually connected to mechanical systems which do not always move smoothly. Because of these imperfections, t can be “jitter” in the quadrature encoder signal. This jitter is not an electrical imperfection, but represent the reality of the mechanical system vibrating, jumping, bouncing, etc. If these imperfections occur during the period of time w lines are being acquired, the image will be distorted. Lines on the object can be acquired more than once as the stage jitters. To prevent re-acquisition of lines, a circuit has been added to the quadrature encoder system that can prevent any line from being acquired more than once. To enable this mode, set QENC_NO_REAQ = 1.

8.1.5 Scan Step Mode

The encoder can also be used to trigger acquisition of full frames from an area scan camera. The idea is that every N lines, a trigger is issued to the board, which causes acquisition of a frame. This can be used, for example, with a linear stage, w an image is needed in steps across the range of the stage. This mode is enable by setting SCAN_STEP_TRIG = 1, and programming SCAN_STEP to the number of encoder counts per trigger.

8.1.6 Combining Modes

All of the modes above can be combined to support complicated encoder requirements. For example, the board can be programmed to acquire an interval in the positive direction only, with no lines being reacquired. Many combinations are possible.

8.1.7 Control Registers

See Chapter 9 for the registers needed to control the quadrature encoder system.

8.1.8 Observability

The status of the quadrature encoder system can be observed at any time. Shown in Table 8-1are all the registers that can be used.:

Table 8-1 Observability Registers.

Register	Meaning
QENC_COUNT	Encoder counter
QENC_PHASEA	Phase of input A
QENC_PHASEB	Phase of input B
QENC_DIR	Direction of encoder
QENC_INTRVL_IN	Interval status
QENC_NEW_LINES	Indicates new lines are being acquired

8.1.9 Electrical Connections

Both TTL and LVDS (differential) quadrature encoders are supported. TTL connections are shown in Table 8-2 and LVDS connections are shown in Table 8-3.

Table 8-2 TTL Quadrature Encoder Connections

Encoder	Frame Grabber
A	VFGx_ENCODERA_TTL
B	VFGx_ENCODERB_TTL
Ground	GND

Table 8-3 LVDS Quadrature Encoder Connections

Encoder	Frame Grabber
A+	VFGx_ENCODERA+
A-	VFGx_ENCODERA-
B+	VFGx_ENCODERB+
B-	VFGx_ENCODERB-

Note: VFGx - refers to the VFG number that you wish to connect to. For example, if you want to connect a TLL A output to VFG 0, then you would use VFG0_ENCODERA_TTL.

8.2 Understanding Stage Movement vs. Quadrature Encoder Modes

The quadrature encoder system has many modes that can be used in various combinations. These combinations are easier to understand through a few simple illustrations. Figure 8-1 shows the basic Encoder Count vs. Time graph and how it corresponds to stage movement. Keep in mind that the encoder could be attached to any mechanical system, however, a back and forth stage is a simple way to illustrate these modes.

In Figure 8-1 you can see as the stage moves back and forth, the encoder counts up and down. Further, in this example we assume QENC_AQ_DIR = 1, which tells the system to only acquire when the encoder counter is moving in the positive direction. This is illustrated by solid lines in the positive direction and dashed lines in the negative direction.

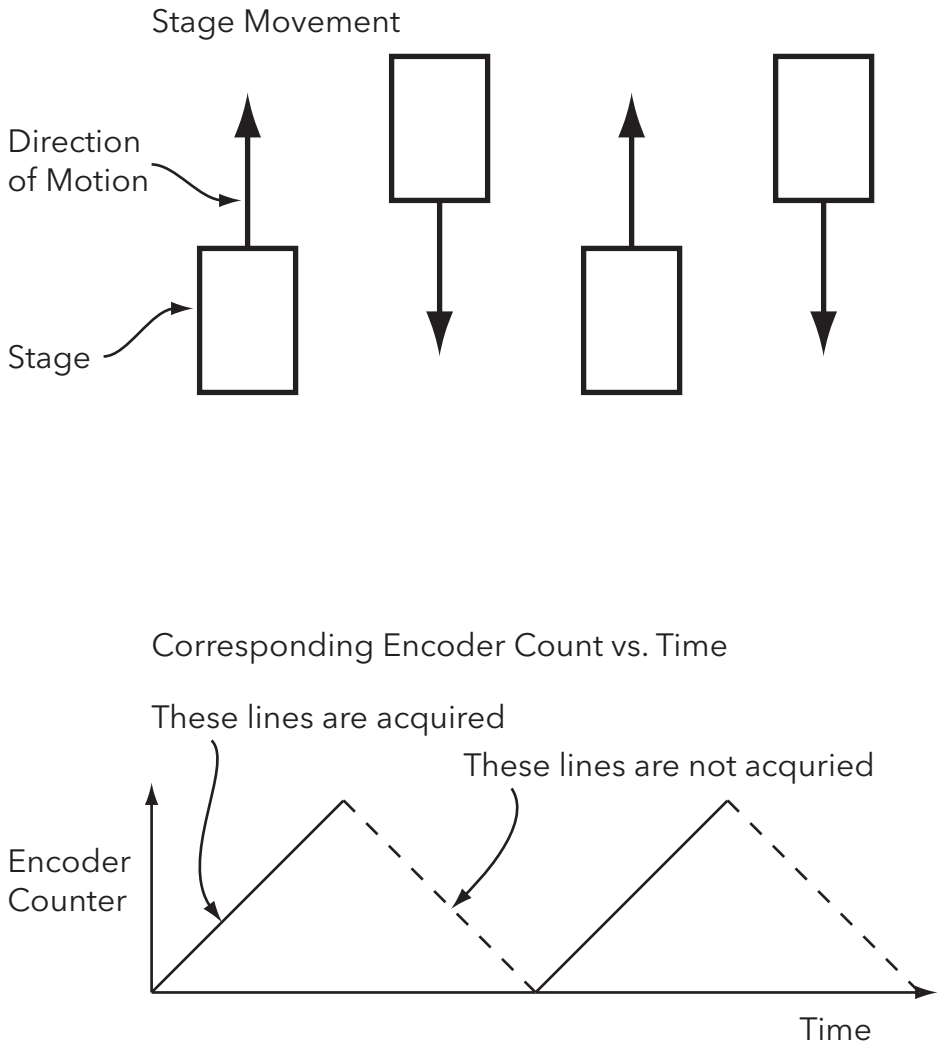

Figure 8-1 Encoder Count vs Time

Figure 8-2 shows all of the major quadrature encoder modes.

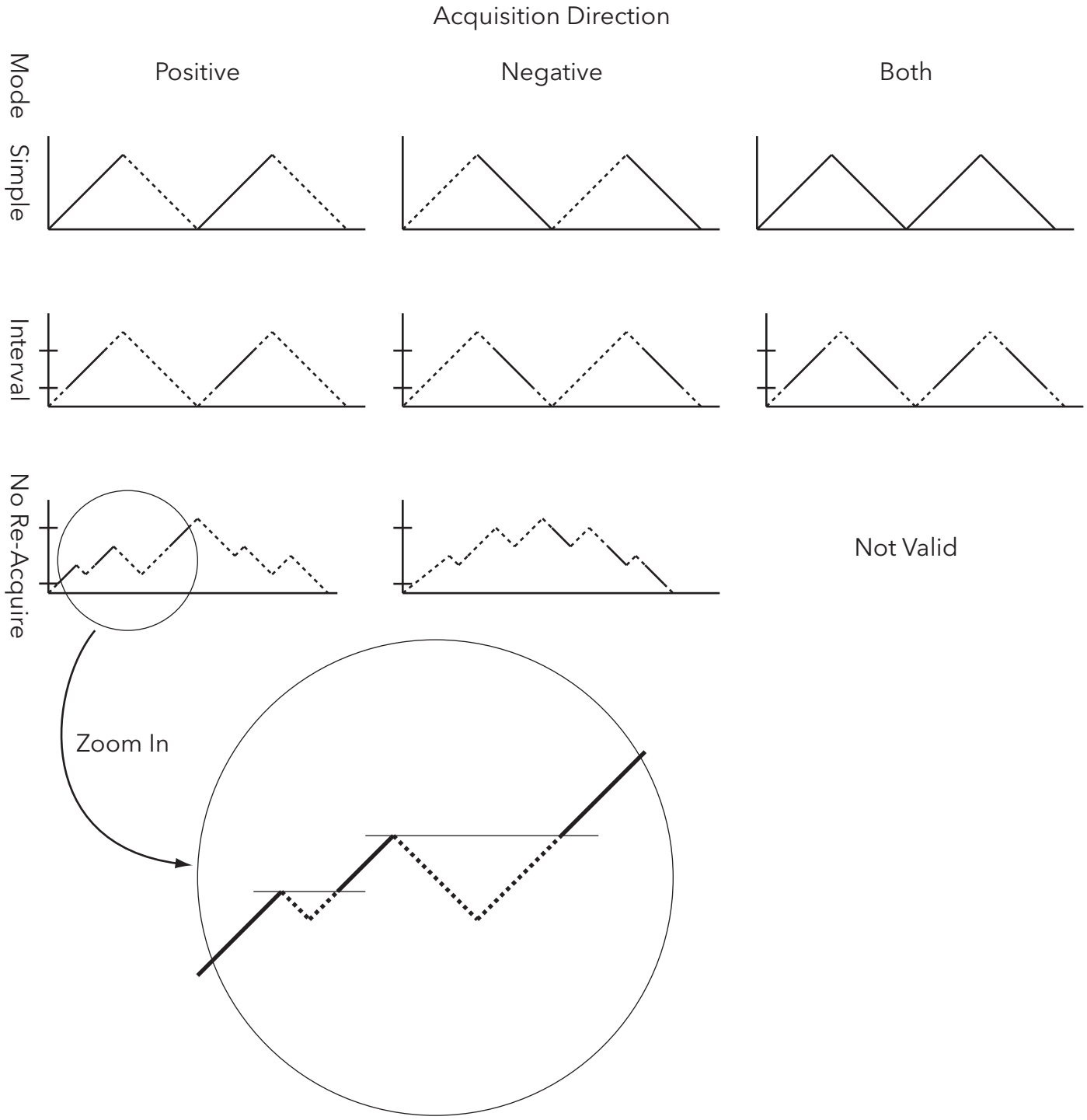

Figure 8-2 Quadrature Encoder Modes vs. Acquisition

Quadrature Encoder and Divider Registers

Chapter 9

9.1 Introduction

This section enumerates the registers used to control the boards quadrature encoder circuit and encoder divider circuit.

9.2 CON65 Register

Bit	Name
0	SEL_ENCQ
1	SEL_ENCDIV_INPUT
2	SEL_ENCDIV
3	ENC_DIV_N
4	ENC_DIV_N
5	ENC_DIV_N
6	ENC_DIV_M
7	ENC_DIV_M
8	ENC_DIV_M
9	ENC_DIV_M
10	ENC_DIV_M
11	ENC_DIV_M
12	ENC_DIV_M
13	ENC_DIV_M
14	ENC_DIV_M
15	ENC_DIV_M
16	SCAN_STEP
17	SCAN_STEP
18	SCAN_STEP
19	SCAN_STEP
20	SCAN_STEP
21	SCAN_STEP
22	SCAN_STEP
23	SCAN_STEP
24	SCAN_STEP
25	SCAN_STEP
26	SCAN_STEP
27	SCAN_STEP
28	SCAN_STEP
29	SCAN_STEP
30	SCAN_STEP
31	SCAN_STEP

SEL_ENCQ R/W, CON65[0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit selects which quadrature encoder circuit output will be used on this VFG.

SEL_ENCQ	Meaning
0	Select the output of this VFG's quadrature circuit output
1	Select the output of VFG0's quadrature circuit output

SEL_ENCDIV_INPUT R/W, CON65[1], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit selects which input will drive the encoder divider circuit.

SEL_ENC_DIV_INPUT	Meaning
0	VFGx_ENCA_SEL
1	The output of this VFG's quadrature circuit output

SEL_ENCDIV R/W, CON65[2], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit selects which encoder divider circuit output will be used on this VFG.

SEL_ENCDIV	Meaning
0	Select the output of this VFG's encoder divider
1	Select the output of VFG0's encoder divider

ENC_DIV_N R/W, CON65[5..3], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

These bits set the N part of the encoder divider equation. See Section 7.1 for more information.

ENC_DIV_M R/W, CON65[15..6], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

These bits set the M part of the encoder divider equation. See Section 7.1 for more information.

SCAN_STEP R/W, CON65[31..16], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bitfield controls the number of encoder pulses that must occur before a trigger is issued to the system. See SCAN_STEP_TRIG for more information. The Scan Step circuit takes into account the interval and re-acquisition functions.

9.3 CON66 Register

Bit	Name
0	QENC_INTRVL_LL
1	QENC_INTRVL_LL
2	QENC_INTRVL_LL
3	QENC_INTRVL_LL
4	QENC_INTRVL_LL
5	QENC_INTRVL_LL
6	QENC_INTRVL_LL
7	QENC_INTRVL_LL
8	QENC_INTRVL_LL
9	QENC_INTRVL_LL
10	QENC_INTRVL_LL
11	QENC_INTRVL_LL
12	QENC_INTRVL_LL
13	QENC_INTRVL_LL
14	QENC_INTRVL_LL
15	QENC_INTRVL_LL
16	QENC_INTRVL_LL
17	QENC_INTRVL_LL
18	QENC_INTRVL_LL
19	QENC_INTRVL_LL
20	QENC_INTRVL_LL
21	QENC_INTRVL_LL
22	QENC_INTRVL_LL
23	QENC_INTRVL_LL
24	QENC_DECODE
25	QENC_AQ_DIR
26	QENC_AQ_DIR
27	QENC_INTRVL_MODE
28	QENC_NO_REAQ
29	Reserved
30	SCAN_STEP_TRIG
31	QENC_RESET

QENC_INTRVL_LL R/WR/W, CON66[23..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register contains the lower limit value that is used to start acquisition when the system is in interval mode (see QENC_INTRVL_MODE).

QENC_DECODE R/W, CON66[24], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit determines how often the quadrature counter is incremented.

QENC_DECODE	Meaning
0	Counter increments on the rising edge of input A and the rising edge of input B. This is also called "2x" mode.
1	Counter increments on both the rising and falling edge of A and both the rising and falling edge of B. This is also called "4x" mode.

QENC_AQ_DIR R/W, CON66[26..25], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls which quadrature encoder direction is used for acquisition.

QENC_AQ_DIR	Meaning
0 (00b)	Lines are acquired in both directions
1 (01b)	Lines are acquired only in the positive direction.
2 (10b)	Lines are acquired only in the negative direction.
3 (11b)	Reserved

QENC_INTRVL_MODE R/W, CON66[27], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

When this bit is 1, interval mode is turned on. When interval mode is on, lines are only captured when the encoder counter is between the lower limit (set by QENC_INTRVL_LL) and the upper limit (set by QENT_INTRVL_UL). If the counter is outside of this range, lines are not acquired. Whether lines are acquired as the counter increments through the interval, or decrements through the interval, or in both directions are controlled by QENC_AQ_DIR.

QENC_NO_REAQ R/W, CON66[28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls how the quadrature encoder system handles the situation w the encoder does not smoothly increase (or decrease if QENC_AQ_DIR = 1). If there is "jitter" in the encoder signal, often caused by problems with the mechanical systems, it is possible for the board to acquire the same line or lines more than once as the

mechanical system backs up and moves forward (jitter). This re-acquisition can cause problems as the resulting images will have distortions and will not accurately represent the object in front of the camera.

Programming this bit to a 1 turns on the no-reacquisition circuit. This circuit eliminates this problem as each line in the image will only be acquired once, regardless of how much jitter occurs in the quadrature encoder input. The circuit does this by making sure that only one line is acquired for each encoder counter value. If the quadrature encoder backs up, and then moves forward, the board will not acquire lines until a new encoder counter value is reached.

This system handles any amount of jitter, regardless of how many times the counter passes through a value, or to what extremes the counter goes. New lines will only be acquired when new values are reached.

Once the entire frame has been acquired, the system must be reset. The system can always be reset by poking QENC_RESET to 1. There are also ways that the system can automatically be reset, see QENC_RESET_MODE.

QENC_NO_REAQ	Meaning
0	Lines are acquired every change in the encoder counter (as controlled by QENC_AQ_DIR)
1	Lines are only acquired when the encoder counter reaches new values (also controlled by QENC_AQ_DIR)

SCAN_STEP_TRIG R/W, CON66[30], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

The scan step circuit uses the encoder to generate a trigger to the system. The scan step trigger generates a trigger every N lines (N is set in the SCAN_STEP register).

SCAN_STEP_TRIG	Meaning
0	The internal signal VFGx_ENCQ_SEL is the output of the quadrature encoder circuit
1	The internal signal VFGx_ENCQ_SEL is the output of the scan step circuit

QENC_RESET WO, CON66[31], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Poking this bit to a 1 resets the entire quadrature encoder system.

9.4 CON67 Register

Bit	Name
0	QENC_INTRVL_UL
1	QENC_INTRVL_UL
2	QENC_INTRVL_UL
3	QENC_INTRVL_UL
4	QENC_INTRVL_UL
5	QENC_INTRVL_UL
6	QENC_INTRVL_UL
7	QENC_INTRVL_UL
8	QENC_INTRVL_UL
9	QENC_INTRVL_UL
10	QENC_INTRVL_UL
11	QENC_INTRVL_UL
12	QENC_INTRVL_UL
13	QENC_INTRVL_UL
14	QENC_INTRVL_UL
15	QENC_INTRVL_UL
16	QENC_INTRVL_UL
17	QENC_INTRVL_UL
18	QENC_INTRVL_UL
19	QENC_INTRVL_UL
20	QENC_INTRVL_UL
21	QENC_INTRVL_UL
22	QENC_INTRVL_UL
23	QENC_INTRVL_UL
24	QENC_REAQ_MODE
25	QENC_REAQ_MODE
26	QENC_RESET_REAQ
27	ENC_DIV_FOURCE_DC
28	ENC_DIV_OPEN_LOOP
29	ENC_DIV_FCLK_SEL
30	ENC_DIV_FCLK_SEL
31	ENC_DIV_FCLK_SEL

QENC_INTRVL_UL R/W, CON67[23..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register contains the upper limit value that is used to start acquisition when the system is in interval mode (see QENC_INTRVL_MODE).

QENC_RESET_MODE R/W, CON67[25..24], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit controls how the circuit that prevents re-acquisition from encoder jitter is reset. Re-acquisition is prevented by keeping a list of lines that have been acquired, and making sure that only lines that are not on the list are acquired. Once the entire frame is acquired, t must be some way to reset the list, otherwise no new lines will ever be acquired See QENC_NO_REAQ for more information.

The reset can be either automatic or manual. Manual modes require that the host application software poke the QENC_RESET_REAQ bit when the reset is desired. Automatic modes do not require host interaction, the reset will occur automatically when the specified conditions are met.

QENC_REAQ_MODE	Mode	Meaning
0 (00b)	Manual	Reset the list of acquired lines when QENC_RESET_REAQ is poked to 1.
1 (01b)	Automatic	Reset the list of lines when the encoder counter is outside of the interval set by the upper limit and lower limit. Whether the reset occurs above the upper limit or below the lower limit depends on the QENC_AQ_DIR register.
2 (10b)		Reserved
3 (11b)		Reserved

QENC_RESET_REAQ WO, CON67[26], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This register is used to reset the circuit that prevents the re-acquisition of lines when QENC_NO_REAQ is set to 1. Writing a 1 to this register deletes the list of acquired lines, thus next time the lines are passed over, they will be acquired again. Writing to this bit always resets the no re-acquisition circuit, regardless of the mode set by the QENC_REAQ_MODE. However, the register QENC_REAQ_MODE can be used to set the board in a mode w the no re-acquisition circuit is reset automatically every pass over the image.

ENC_DIV_FORCE_DC R/W, CON67[27], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Setting this bit to 1 forces the encoder divider to output a DC signal when the input signal falls below a certain frequency.

ENC_DIV_OPEN_LOOP R/W, CON67[28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Setting this bit to 1 forces the encoder divider to run open loop.

ENC_DIV_FCLK_SEL R/W, CON67[31..29], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Reserved for future support for alternate Encoder Divider PLL Master clock frequencies. Currently must be set to 0, which selects 125 MHz clock.

9.5 CON68 Register

Bit	Name
0	RD_ENCQ_SELECTED
1	RD_ENCDIV_SELECTED
2	ENC_DIV_RESET
3	ENC_DIV_AUTO_RESET_DISABLE
4	Reserved
5	Reserved
6	Reserved
7	Reserved
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

RD_ENCQ_SELECTED RO, CON68[0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit indicates the current state of selected quad encoder circuit output.

RD_ENCDIV_SELECTED RO, CON68[1], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit displays the current state of the selected encoder divider output.

ENC_DIV_RESET WO, CON68[2], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Writing a1 to this register resets the encoder divider/multiplier.

ENC_DIV_AUTO_RESET_DISABLE R/W, CON68[3], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Set this bit to 1 to disable the encoder divider/multiplier auto-reset function.

9.6 CON69 Register

Bit	Name
0	QENC_COUNT
1	QENC_COUNT
2	QENC_COUNT
3	QENC_COUNT
4	QENC_COUNT
5	QENC_COUNT
6	QENC_COUNT
7	QENC_COUNT
8	QENC_COUNT
9	QENC_COUNT
10	QENC_COUNT
11	QENC_COUNT
12	QENC_COUNT
13	QENC_COUNT
14	QENC_COUNT
15	QENC_COUNT
16	QENC_COUNT
17	QENC_COUNT
18	QENC_COUNT
19	QENC_COUNT
20	QENC_COUNT
21	QENC_COUNT
22	QENC_COUNT
23	QENC_COUNT
24	QENC_PHASEA
25	QENC_PHASEB
26	QENC_DIR
27	QENC_INTRVL_IN
28	QENC_NEW_LINES
29	Reserved
30	QENC_FREQ
31	QENC_FREQ

QENC_COUNT RO, CON69[23..0], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bitfield displays the current quadrature encoder count.

QENC_PHASEA RO, CON69[24], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit displays the current logic level of the A quadrature encoder phase.

QENC_PHASEB RO, CON69[25], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit displays the current logic level of the B quadrature encoder phase.

QENC_DIR RO, CON69[26], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit displays the current quadrature encoder direction.

QENC_DIR	Meaning
0	Direction is negative
1	Direction is positive

QENC_INTRVL_IN RO, CON69[27], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit indicates the current status of the quadrature encoder if the system is in interval mode (see QENC_INTRVL_MODE).

QENC_INTRVL_IN	Meaning
0	System is not inside the interval. Encoder counter is not between QENC_INTRVL_LL and QENC_INTRVL_UL. Lines are not being acquired.
1	System is inside the interval. Encoder counter is between QENC_INTRVL_LL and QENC_INTRVL_UL. Lines are being acquired.

QENC_NEW_LINES RO, CON69[28], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

This bit indicates if the system is at an encoder count that corresponds to a new line. When QENC_NO_REAQ = 1, only lines that have not yet been scanned are acquired. This bit can be used to determine of new lines are being traversed, or if the system has backed up, and is revisiting old lines.

QENC_NEW_LINES	Meaning
0	The system is traversing lines that have already been visited. If QENC_NO_REAQ = 1, lines are not being acquired.
1	The system is traversing new lines. Lines are being acquired.

QENC_FREQ RO, CON69[31..30], Aon-CXP, Axion-CL, Claxon-CXP, Cyton-CXP

Reserved for future support for alternate decoder timing.

System Probe

Chapter 10

10.1 Introduction

System Probe is a flexible event monitor for analyzing system metrics in real time, performance tuning, and debug. At the core of the monitor is a programmable recording window defined by SP_START_EVENT, SP_STOP_EVENT, and SP_STOP_LIMIT. While a recording window is active SP_TARGET_EVENT(s) are monitored and recorded to SP_COUNT according to one of the SP_COUNT_MODEs.

The set of supported events include Camera Link framing signals, Axion IO, PCIe, and several timing sources. Any of these events can be individually selected as the bounds of a recording window (SP_START_EVENT, SP_STOP_EVENT), the target event to monitor (SP_TARGET_EVENT), and the optional synchronization event SP_SYNC_EVENT. Since all parameters of the recording window can be controlled by unique events, a recording window can be:

Timed - Record a metric for a specified time interval. Example: frame rate (number of frames in a given time period).

Untimed - Record a metric between bounding events. Example: lines per frame.

SP_TARGET_EVENTs are recorded between SP_START_EVENT and SP_STOP_EVENT. SP_STOP_LIMIT allows this START/STOP record sequence to be run over multiple iterations. Among other things, this is useful for supporting SP_COUNT_MODEs min and max. Example: Determine the maximum number of Camera Link clocks occurring between lines over the course of a complete frame.

10.2 System Probe Constants

This subsection lists the constants that be used to program the System Probe.

10.2.1 List of System Probe Events

Table 10-1 lists all the events the System Probe can count and can use as the start, stop or sync event. These are used when programming the SP_XXX_EVENT registers.

Note: The Event Constants in Table 10-1 are defined in the SDK in the file "Gen2Def.h"

Table 10-1 System Probe Events

Event Constant	Numeric Value	Meaning
SPE_NEVER	0x00	No Event, never fires
SPE_IMMEDIATE	0x01	Immediate - Fires every SYS_CLK, same as SYS_CLK. Redundant but included anyways because the label IMMEDIATE is more intuitive
SPE_SYS_CLK	0x02	125MHz system clock
SPE_100_NS	0x03	100 nanosecond pulse
SPE_TSAUX_0	0x04	Auxiliary timing sequencer CT0
SPE_TSAUX_1	0x05	Auxiliary timing sequencer CT1
SPE_TSAUX_2	0x06	Auxiliary timing sequencer CT2
SPE_TSAUX_3	0x07	Auxiliary timing sequencer CT3
SPE_TRIG	0x08	VFGx_TRIG_SEL, the currently selected trigger
SPE_ENCA	0x09	VFGx_ENCA_SEL, the currently selected encoder A signal
SPE_ENCB	0x0a	VFGx_ENCB_SEL, the currently selected encoder B signal
SPE_ENCQ	0x0b	VFGx_ENCQ_SEL, the output of the quadrature encoder circuit
SPE_ENCDIV	0x0c	VFGx_ENCDIV_SEL, the output of the encoder divider
SPE_INTERRUPT	0x0f	System interrupt for this VFG
SPE_PCI_SENT	0x10	PCIe bytes sent (16-byte chunks)
SPE_PCI_DROP	0x11	PCIe bytes dropped (16-byte chunks)
SPE_PCI_BUSY	0x12	PCIe bus towards the host is busy ??
SPE_CL_CLK_B	0x20	CL Clock Base Channel
SPE_CL_CLK_M	0x21	CL Clock Medium Channel
SPE_CL_CLK_F	0x22	CL Clock Full Channel

Table 10-1 System Probe Events

Event Constant	Numeric Value	Meaning
SPE_PLL_LOCK_B	0x23	PLL Lock Base Channel
SPE_PLL_LOCK_M	0x24	PLL Lock Medium Channel
SPE_PLL_LOCK_F	0x25	PLL Lock Full Channel
SPE_FVAL_B	0x26	FVAL Base Channel
SPE_FVAL_M	0x27	FVAL Medium Channel
SPE_FVAL_F	0x28	FVAL Full Channel
SPE_LVAL_B	0x29	LVAL Base Channel
SPE_LVAL_M	0x2A	LVAL Medium Channel
SPE_LVAL_F	0x2B	LVAL Full Channel
SPE_DVAL_B	0x2C	DVAL Base Channel
SPE_DVAL_M	0x2D	DVAL Medium Channel
SPE_DVAL_F	0x2E	DVAL Full Channel
SPE_CC1	0x2F	CC1
SPE_CC2	0x30	CC2
SPE_CC3	0x31	CC3
SPE_CC4	0x32	CC4

10.2.2 List of System Probe Functions

Table 10-2 lists all the functions that are available for each System Probe event. These are used when programming the SP_XXX_FUNC registers.

Note: The Event Constants in Table 10-2 are defined in the SDK in the file "Gen2Def.h"

Table 10-2 System Probe Events Functions

Event Constant	Numeric Value	Meaning
SPF_RISE	0x00	The rising edge of the signal
SPF_FALL	0x01	The falling edge of the signal
SPF_HIGH	0x02	The signal is high
SPF_LOW	0x03	The signal is low

10.2.3 List of System Probe Counting Modes

Table 10-3 lists different modes that the System Probe can count events. These are used when programing the SP_COUNT_MODE registers.

Note: The Event Constants in Table 10-3 are defined in the SDK in the file "Gen2Def.h"

Table 10-3 System Probe Counting Modes

Event Constant	Numeric Value	Meaning
SPCM_CURR	0x00	Current: on each update SP_COUNT will reflect the current event capture.
SPCM_ACC	0x01	Accumulate: on each update SP_COUNT will reflect the Accumulation of all event captures since the System Probe was armed. Averaging can be handled by s/w in conjunction with the SP_STOP_EVENT_LIMIT register.
SPCM_MIN	0x02	Minimum: on each update SP_COUNT will reflect the minimum of all event captures since System Probe was armed.
SPCM_MAX	0x03	Maximum: on each update SP_COUNT will reflect the maximum of all event captures since System Probe was armed.

10.3 System Probe Examples

The System probe is best understood via an example. The subsections below illustrated a variety of ways to use the System Probe to measure different events and/or signals.

Note: The examples below are not actual code. These examples just list how each register should be programmed in order to measure the example quantity.

Note: The contanstants used in the examples below are from Table 10-1.

10.3.1 Example - Clocks per Line

In our first example we will illustrate how to set the system probe up to measure the number of Camera Link clocks per line (LVAL) attached cameras. Below is a list of how to set each register (this uses the constants from Table 10-1).

```
SP_SYNC_EVENT = SPE_IMMEDIATE
SP_SYNC_FUNC = SPF_RISE
SP_START_EVENT = SPE_LVAL_B
SP_START_FUNC = SPF_RISE
SP_STOP_EVENT = SPE_LVAL_B
SP_STOP_FUNC = SPF_FALL
SP_TARGET_EVENT = SPE_CL_CLK_B
SP_TARGET_FUNC = SPF_RISE
SP_STOP_EVENT_LIMIT = 1
SP_COUNT_MODE = SPCM_ACC
```

In this example, there is no sync event. The rising edge of LVAL starts the event recording and the falling edge of LVAL stops the event recording. The event that is being recorded is the CL clock on the base channel. The event recorder runs for exactly one occurrence of the stop event (i.e. one line) because SP_STOP_EVENT_LIMIT = 1. The number of clocks will be accumulated in the SP_COUNT register and can be read from the host when SP_BUSY = 0.

10.3.2 Example - Camera Frame Rate

For a slightly more complicated example, we will set the system probe up to measure the attached cameras frame rate. Below is a list of how to set each register (this uses the constants from Table 10-1).

```
SP_SYNC_EVENT = SPE_IMMEDIATE
SP_SYNC_FUNC = SPF_RISE
SP_START_EVENT = SPE_IMMEDIATE
SP_START_FUNC = SPF_RISE
SP_STOP_EVENT = SPE_100_NS
SP_STOP_FUNC = SPF_RISE
SP_TARGET_EVENT = SPE_FVAL_B
SP_TARGET_FUNC = SPF_RISE
```

```
SP_STOP_EVENT_LIMIT = 1000000
SP_COUNT_MODE = SPCM_ACC
```

For this example, there is no sync or start event. The recorder starts immediately. The events being recorded are the rising edge of FVAL. The SP_STOP_EVENT_LIMIT controls how long the counter accumulates (in this case it is counting the number of 100 nanosecond clocks). The 100 nanosecond internal clock is useful for measuring things in terms of time, rather than Camera Link clocks or another signal whose period is unknown. When the recorder reaches this limit, SP_COUNT will contain the number FVALs (rising edges) that occurred during 1,000,000 count of the 100 nanosecond clock ticks (rising edges). The value that is found in SP_COUNT will have to be scaled to get to the units of Frames per Second.

10.4 SP_EVENTS

Bit	Name
0	SP_TARGET_EVENT
1	SP_TARGET_EVENT
2	SP_TARGET_EVENT
3	SP_TARGET_EVENT
4	SP_TARGET_EVENT
5	SP_TARGET_EVENT
6	SP_TARGET_FUNC
7	SP_TARGET_FUNC
8	SP_STOP_EVENT
9	SP_STOP_EVENT
10	SP_STOP_EVENT
11	SP_STOP_EVENT
12	SP_STOP_EVENT
13	SP_STOP_EVENT
14	SP_STOP_FUNC
15	SP_STOP_FUNC
16	SP_START_EVENT
17	SP_START_EVENT
18	SP_START_EVENT
19	SP_START_EVENT
20	SP_START_EVENT
21	SP_START_EVENT
22	SP_START_FUNC
23	SP_START_FUNC
24	SP_SYNC_EVENT
25	SP_SYNC_EVENT
26	SP_SYNC_EVENT
27	SP_SYNC_EVENT
28	SP_SYNC_EVENT
29	SP_SYNC_EVENT
30	SP_SYNC_FUNC
31	SP_SYNC_FUNC

SP_TARGET_EVENT R/W, SP_EVENTS[5..0], Axion-CL

Specifies the event that will be monitored and counted. Event counting starts at the event written to SP_START_EVENT and ends at the event written to the SP_STOP_EVENT register. See Table 10-1 for a list of events.

Note: This register can only be writeen when SP_BUSY reads back 0.

SP_TARGET_FUNC R/W, SP_EVENTS[7..6], Axion-CL

Specifies the event function that will be counted.

SP_TARGET _FUNC	Meaning
0 (00b)	Rising edge of event
1 (01b)	Falling edge of event
2 (10b)	Event is high
3 (11b)	Event is low

SP_STOP_EVENT R/W, SP_EVENTS[13..8], Axion-CL

Specifies the event that will stop recording. See Table 10-1 for a list of events.

Note: This register can only be writeen when SP_BUSY reads back 0.

SP_STOP_FUNC R/W, SP_EVENTS[15..14], Axion-CL

Specifies the event function that will stop recording.

SP_STOP _FUNC	Meaning
0 (00b)	Rising edge of event
1 (01b)	Falling edge of event
2 (10b)	Event is high
3 (11b)	Event is low

SP_START_EVENT R/W, SP_EVENTS[21..16], Axion-CL

Specifies the event that will start recording. See Table 10-1 for a list of events.

Note: This register can only be writeen when SP_BUSY reads back 0.

SP_START_FUNC R/W, SP_EVENTS[23..22], Axion-CL

Specifies the event function that will start recording. See Table 10-1 for a list of events.

SP_START_FUNC	Meaning
0 (00b)	Rising edge of event
1 (01b)	Falling edge of event
2 (10b)	Event is high
3 (11b)	Event is low

SP_SYNC_EVENT R/W, SP_EVENTS[29..24], Axion-CL

Specifies the event that is used the sync the recording. The sync event must occur first, before the System Problem waits for the start event. See Table 10-1 for a list of events.

Note: This register can only be writeen when SP_BUSY reads back 0.

SP_SYNC_FUNC R/W, SP_EVENTS[31..30], Axion-CL

Specifies the event function syncing the recording

SP_SYNC_FUNC	Meaning
0 (00b)	Rising edge of event
1 (01b)	Falling edge of event
2 (10b)	Event is high
3 (11b)	Event is low

10.5 SP_CON

Bit	Name
0	SP_COUNT_MODE
1	SP_COUNT_MODE
2	Reserved
3	Reserved
4	Reserved
5	Reserved
6	Reserved
7	Reserved
8	Reserved
9	SP_WRAP_COUNT
10	SP_ARM
11	SP_RST
12	Reserved
13	Reserved
14	SP_COUNT_UPDATED
15	SP_BUSY
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

SP_COUNT_MODE R/W, SP_CON[1..0], Axion-CL

This register sets the mode of how the System Probe count (in register SP_COUNT) is updated.

SP_COUNT_MODE	Meaning
0 (00b)	Current: on each update SP_COUNT will reflect the current event capture.
1 (01b)	Accumulate: on each update SP_COUNT will reflect the Accumulation of all event captures since the System Probe was armed. Averaging can be handled by s/w in conjunction with the SP_STOP_EVENT_LIMIT register.
2 (10b)	Minimum: on each update SP_COUNT will reflect the minimum of all event captures since System Probe was armed.
3 (11b)	Maximum: on each update SP_COUNT will reflect the maximum of all event captures since System Probe was armed.

SP_WRAP_COUNT R/W, SP_CON[9], Axion-CL

Controls what happens when the System Probe count (SP_COUNT) reaches its maximum value.

Note: This bit can only be written when SP_BUSY reads back 0.

SP_WRAP_COUNT	Meaning
0	When SP_COUNT reaches its maximum value, it sticks at this value until cleared
1	When SP_COUNT reaches its maximum value, it rolls over to 0

SP_ARM WO, SP_CON[10], Axion-CL

Writing this register to 1 arms the System Probe. Event counting will not start until the event specified in SP_SYNC_EVENT has occurred and the event specified in SP_START_EVENT occurs. This bit always reads back 0.

Note: This bit can only be written when SP_BUSY reads back 0.

SP_RST WO, SP_CON[11], Axion-CL

Reset the System Probe. Always writable. Always reads back 0.

SP_COUNT_UPDATED RO, SP_CON[14], Axion-CL

SP_COUNT updates after each STOP event. SP_COUNT_UPDATED is set to 1 by hardware when the SP_COUNT is updated. In cases where SP_STOP_EVENT_LIMIT is > 0, there will be multiple updates to SP_COUNT. SP_COUNT_UPDATED is provided as a mechanism for s/w to know if it is reading new data. SP_COUNT_UPDATED clears automatically when SP_COUNT is read.

SP_BUSY RO, SP_CON[15], Axion-CL

Indicates the current state of the System Probe.

SP_BUSY	Meaning
0	System Probe is idle
1	System Probe is busy

10.6 SP_STAT

Bit	Name
0	SP_COUNT
1	SP_COUNT
2	SP_COUNT
3	SP_COUNT
4	SP_COUNT
5	SP_COUNT
6	SP_COUNT
7	SP_COUNT
8	SP_COUNT
9	SP_COUNT
10	SP_COUNT
11	SP_COUNT
12	SP_COUNT
13	SP_COUNT
14	SP_COUNT
15	SP_COUNT
16	SP_COUNT
17	SP_COUNT
18	SP_COUNT
19	SP_COUNT
20	SP_COUNT
21	SP_COUNT
22	SP_COUNT
23	SP_COUNT
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

SP_COUNT RO, SP_STAT[23..0], Axion-CL

This register indicates the number of events that have occurred. This can be read at any time. When SP_BUSY is running, there may be several updates to SP_COUNT (it is updated after each STOP event). SP_STOP_EVENT_LIMIT controls how many stop events we require to end the capture. Busy goes low when SP_STOP_EVENT_LIMIT is reached. However, during this time (when busy SP_BUSY=1) SP_COUNT can be updated. To assist with this problem, please see the register SP_COUNT_UPDATED.

10.7 SP_LIMIT

Bit	Name
0	SP_STOP_EVENT_LIMIT
1	SP_STOP_EVENT_LIMIT
2	SP_STOP_EVENT_LIMIT
3	SP_STOP_EVENT_LIMIT
4	SP_STOP_EVENT_LIMIT
5	SP_STOP_EVENT_LIMIT
6	SP_STOP_EVENT_LIMIT
7	SP_STOP_EVENT_LIMIT
8	SP_STOP_EVENT_LIMIT
9	SP_STOP_EVENT_LIMIT
10	SP_STOP_EVENT_LIMIT
11	SP_STOP_EVENT_LIMIT
12	SP_STOP_EVENT_LIMIT
13	SP_STOP_EVENT_LIMIT
14	SP_STOP_EVENT_LIMIT
15	SP_STOP_EVENT_LIMIT
16	SP_STOP_EVENT_LIMIT
17	SP_STOP_EVENT_LIMIT
18	SP_STOP_EVENT_LIMIT
19	SP_STOP_EVENT_LIMIT
20	SP_STOP_EVENT_LIMIT
21	SP_STOP_EVENT_LIMIT
22	SP_STOP_EVENT_LIMIT
23	SP_STOP_EVENT_LIMIT
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

SP_STOP_ EVENT_LIMIT

R/W, SP_LIMIT[23..0], Axion-CL

When collecting statistical data (i.e. SP_COUNT_MODE is not equal to 0), this register controls how many start/stop events should occur before the System Probe stops recording event data. Set this register to 1 for "single shot" event recording. Set it to the maximum number of event windows to count when using the System Probe for statistical calculations.

Axion Camera Link Registers

Chapter 11

11.1 Introduction

This section contains definitions for registers that are only on the Axion-CL platform. The Cyton-CXP and the Axion-CL have many of the same registers, but some are only relevant to CXP and some are only used for Camera Link. This chapter contains the latter.

11.2 CL_IOBUF_CTL

Bit	Name
0	IOBUF_SETTING
1	IOBUF_SETTING
2	IOBUF_SETTING
3	IOBUF_SETTING
4	IOBUF_SETTING
5	Reserved
6	Reserved
7	Reserved
8	IOBUF_LANE
9	IOBUF_LANE
10	IOBUF_LANE
11	Reserved
12	IOBUF_CHAN
13	IOBUF_CHAN
14	IOBUF_CHAN
15	IOBUF_CHAN
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	IOBUF_WRITE
31	IOBUF_BUSY

IOBUF_SETTING R/W, CL_IOBUF_CTL[4..0], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not be programmed directly by customers.

IOBUF_LANE R/W, CL_IOBUF_CTL[10..8], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not be programmed directly by customers.

IOBUF_CHAN R/W, CL_IOBUF_CTL[15..12], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not be programmed directly by customers.

IOBUF_WRITE R/W, CL_IOBUF_CTL[30], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not be programmed directly by customers.

IOBUF_BUSY R/W, CL_IOBUF_CTL[31], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not be programmed directly by customers.

11.3 CL_CHAN_CONFIG

Bit	Name
0	PLL_PLL_PHASE_DIR
1	PLL_ADJUST_PLL_PHASE
2	PLL_CONFIG_ERROR
3	PLL_RST
4	ALIGN_MANUAL_RST
5	ALIGN_MANUAL_DELAY
6	ALIGN_MANUAL_LOCK
7	ALIGN_MANUAL_EN
8	ALIGN_AUTO_EN
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	PLL_CHAN
29	PLL_CHAN
30	PLL_CHAN
31	PLL_CONFIG_BUSY

PLL_PLL_PHASE_DIR R/W, CL_CHAN_CONFIG[0], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not but programmed directly by customers.

PLL_ADJUST_PLL_PHASE R/W, CL_CHAN_CONFIG[1], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not but programmed directly by customers.

PLL_CONFIG_ERROR R/W, CL_CHAN_CONFIG[2], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not but programmed directly by customers.

PLL_RST R/W, CL_CHAN_CONFIG[3], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not but programmed directly by customers.

ALIGN_MANUAL_RST R/W, CL_CHAN_CONFIG[4], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not but programmed directly by customers.

ALIGN_MANUAL_DELAY R/W, CL_CHAN_CONFIG[5], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not but programmed directly by customers.

ALIGN_MANUAL_LOCK R/W, CL_CHAN_CONFIG[6], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not but programmed directly by customers.

ALIGN_MANUAL_EN R/W, CL_CHAN_CONFIG[7], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not but programmed directly by customers.

ALIGN_AUTO_EN R/W, CL_CHAN_CONFIG[8], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not but programmed directly by customers.

PLL_CHAN R/W, CL_CHAN_CONFIG[30..28], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not but programmed directly by customers.

PLL_CONFIG_BUSY R/W, CL_CHAN_CONFIG[31], Axion-CL

This Bitfield is used to program the Camera Link receiver settings. It should not but programmed directly by customers.

11.4 UART_CON_BASE

Bit	Name
0	RS232_TX_DATA
1	RS232_TX_DATA
2	RS232_TX_DATA
3	RS232_TX_DATA
4	RS232_TX_DATA
5	RS232_TX_DATA
6	RS232_TX_DATA
7	RS232_TX_DATA
8	RS232_TX_GO
9	RS232_RX_INT_ENABLE
10	RS232_RX_FIFO_CLEAR
11	RS232_RX_INVERT
12	RS232_TX_INVERT
13	Reserved
14	Reserved
15	Reserved
16	RS232_BAUD_RATE
17	RS232_BAUD_RATE
18	RS232_BAUD_RATE
19	RS232_BAUD_RATE
20	RS232_BAUD_RATE
21	RS232_BAUD_RATE
22	RS232_BAUD_RATE
23	RS232_BAUD_RATE
24	RS232_RX_LEVEL
25	RS232_RX_LEVEL
26	RS232_RX_LEVEL
27	RS232_RX_LEVEL
28	Reserved
29	RS232_RX_OVERFLOW
30	RS232_RX_REQ
31	RS232_TX_READY

RS232_TX_DATA R/W, UART_CON_BASE[7..0], Axion-CL

Data byte to be sent out the CL RS-232 link.

RS232_TX_GO R/W, UART_CON_BASE[8], Axion-CL

Cause the data byte to be sent out the CL RS-232 link.

RS232_RX_INT_ENABLE R/W, UART_CON_BASE[9], Axion-CL

Enable the interrupt that is asserted whenever a byte is received on the CL RS-232 link.

RS232_RX_FIFO_CLEAR R/W, UART_CON_BASE[10], Axion-CL

Clear the CL RS-232 receive FIFO.

RS232_RX_INVERT R/W, UART_CON_BASE[11], Axion-CL

Describe RS232_RX_INVERT .

RS232_TX_INVERT R/W, UART_CON_BASE[12], Axion-CL

Describe RS232_TX_INVERT .

RS232_BAUD_RATE R/W, UART_CON_BASE[23..16], Axion-CL

Sets the CL RS-232 baud rate.

RS232_RX_LEVEL R/W, UART_CON_BASE[27..24], Axion-CL

Describe RS232_RX_LEVEL .

RS232_RX_OVERFLOW R/W, UART_CON_BASE[29], Axion-CL

Indicates that the CL RS-232 receive FIFO has overflowed.

RS232_RX_REQ R/W, UART_CON_BASE[30], Axion-CL

Register is 1 when t are bytes in the CL RS-232 receive FIFO..

RS232_TX_READY R/W, UART_CON_BASE[31], Axion-CL

When this bitfield is 1 the UART is ready to send another byte.

11.5 UART_RDAT_BASE

Bit	Name
0	RS232_RX_DATA
1	RS232_RX_DATA
2	RS232_RX_DATA
3	RS232_RX_DATA
4	RS232_RX_DATA
5	RS232_RX_DATA
6	RS232_RX_DATA
7	RS232_RX_DATA
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

RS232_RX_DATA RO, UART_RDAT_BASE[7..0], Axion-CL

The top of the CL RS-232 receive FIFO. Ready this register removes this byte from the FIFO.

11.6 CL_CON_BASE

Bit	Name
0	CL_USE_FVAL
1	CL_USE_DVAL
2	Reserved
3	Reserved
4	CL_CHAN_EN
5	CL_CHAN_EN
6	CL_CHAN_EN
7	Reserved
8	CL_LVAL_POS
9	CL_LVAL_POS
10	CL_LVAL_POS
11	Reserved
12	CL_MODE
13	CL_MODE
14	CL_MODE
15	Reserved
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

CL_USE_FVAL R/W, CL_CON_BASE[0], Axion-CL

Setting this bit to 1 will cause the CL front end to honor the FVAL (Frame Valid) signal. This should be set to 1 for area scan cameras and 0 for line scan cameras.

CL_USE_DVAL R/W, CL_CON_BASE[1], Axion-CL

Setting the bit to 1 will cause the CL front end to honor the DVAL signal. DVAL is used to tell the frame grabber which PCLOCK strings should be used to acquire data and which should be ignored. FVAL is primarily meant to support cameras with very low data rates (below the minimum supported by the channel link chips).

CL_CHAN_EN R/W, CL_CON_BASE[6..4], Axion-CL

Each bit in this bitfield enables the corresponding CL channel. The LSB is the base CL channel.

CL_LVAL_POS R/W, CL_CON_BASE[10..8], Axion-CL

Describe CL_LVAL_POS .

CL_MODE R/W, CL_CON_BASE[14..12], Axion-CL

Controls the CL tap decoder mode.

CL_MODE	Meaning
0 (000b)	Base/Medium/Full, 8-Bit
1 (001b)	Base/Medium/Full, 10 to 12-Bit
2 (010b)	Base/Medium/Full, 14 to 16Bit
3 (011b)	Ten-Tap, 8-bit
4 (100b)	Eight-Tap, 10-Bit
5 (101b)	Reserved
6 (110b)	Reserved
7 (111b)	Reserved

11.7 TAP_CON_BASE

Bit	Name
0	TAP_MODE
1	Reserved
2	Reserved
3	Reserved
4	TAP_FIXED_VAL
5	TAP_FIXED_VAL
6	TAP_FIXED_VAL
7	TAP_FIXED_VAL
8	TAP_FIXED_VAL
9	TAP_FIXED_VAL
10	TAP_FIXED_VAL
11	TAP_FIXED_VAL
12	TAP_FIXED_VAL
13	TAP_FIXED_VAL
14	TAP_FIXED_VAL
15	TAP_FIXED_VAL
16	TAP_FIXED_VAL
17	TAP_FIXED_VAL
18	TAP_FIXED_VAL
19	TAP_FIXED_VAL
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	TAP_OUTPUT_16

TAP_MODE R/W, TAP_CON_BASE[0], Axion-CL

Reserved.

TAP_FIXED_VAL R/W, TAP_CON_BASE[19..4], Axion-CL

Reserved.

TAP_OUTPUT_16 R/W, TAP_CON_BASE[31], Axion-CL

This bit should be set to 1 to output 10 to 16 bit pixels as 16-bit words.

11.8 TAP_TABLE_ADDR_BASE

Bit	Name
0	TAP_TABLE_OFFS
1	TAP_TABLE_OFFS
2	TAP_TABLE_OFFS
3	TAP_TABLE_OFFS
4	Reserved
5	Reserved
6	Reserved
7	Reserved
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	TAP_TABLE_INDEX
17	TAP_TABLE_INDEX
18	TAP_TABLE_INDEX
19	TAP_TABLE_INDEX
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	TAP_TABLE_TYPE
25	TAP_TABLE_TYPE
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

TAP_TABLE_OFFS R/W, TAP_TABLE_ADDR_BASE[3..0], Axion-CL

This register is used to program that Axion's tap re-formatter. It should not be programmed directly by users.

TAP_TABLE_INDEX R/W, TAP_TABLE_ADDR_BASE[19..16], Axion-CL

This register is used to program that Axion's tap re-formatter. It should not be programmed directly by users.

TAP_TABLE_TYPE R/W, TAP_TABLE_ADDR_BASE[25..24], Axion-CL

This register is used to program that Axion's tap re-formatter. It should not be programmed directly by users.

11.9 TAP_TABLE_DAT_BASE

Bit	Name
0	TAP_DATA
1	TAP_DATA
2	TAP_DATA
3	TAP_DATA
4	TAP_DATA
5	TAP_DATA
6	TAP_DATA
7	TAP_DATA
8	TAP_DATA
9	TAP_DATA
10	TAP_DATA
11	TAP_DATA
12	TAP_DATA
13	TAP_DATA
14	TAP_DATA
15	TAP_DATA
16	TAP_DATA
17	TAP_DATA
18	TAP_DATA
19	TAP_DATA
20	TAP_DATA
21	TAP_DATA
22	TAP_DATA
23	TAP_DATA
24	TAP_DATA
25	TAP_DATA
26	TAP_DATA
27	TAP_DATA
28	TAP_DATA
29	TAP_DATA
30	TAP_DATA
31	TAP_DATA

TAP_DATA R/W, TAP_TABLE_DAT_BASE[31..0], Axion-CL

This register is used to program that Axion's tap re-formatter. It should not be programmed directly by users.

11.10 FLASH_CON_BASE

Bit	Name
0	FLASH_CODE
1	FLASH_CODE
2	FLASH_CODE
3	FLASH_CODE
4	FLASH_CODE
5	FLASH_CODE
6	FLASH_CODE
7	FLASH_CODE
8	FLASH_WRITE
9	FLASH_SHIFTBYTE
10	FLASH_READ
11	FLASH_RESET
12	FLASH_BULK_ERASE
13	FLASH_SECTOR_ERASE
14	FLASH_SECTOR_PROTECT
15	FLASH_EN4B_ADDR
16	FLASH_EX4B_ADDR
17	FLASH_READ_RDID
18	FLASH_READ_STATUS
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	FLASH_ILLEGAL_WRITE
29	FLASH_ILLEGAL_ERASE
30	FLASH_DATA_VALID
31	FLASH_BUSY

FLASH_CODE R/W, FLASH_CON_BASE[7..0], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_WRITE R/W, FLASH_CON_BASE[8], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_SHIFTBYTE R/W, FLASH_CON_BASE[9], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_READ R/W, FLASH_CON_BASE[10], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_RESET R/W, FLASH_CON_BASE[11], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_BULK_ERASE R/W, FLASH_CON_BASE[12], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_SECTOR_ERASE R/W, FLASH_CON_BASE[13], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_SECTOR_PROTECT R/W, FLASH_CON_BASE[14], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_EN4B_ADDR R/W, FLASH_CON_BASE[15], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_EX4B_ADDR R/W, FLASH_CON_BASE[16], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_READ_RDID R/W, FLASH_CON_BASE[17], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_READ_STATUS R/W, FLASH_CON_BASE[18], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_ILLEGAL_WRITE RO, FLASH_CON_BASE[28], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_ILLEGAL_ERASE RO, FLASH_CON_BASE[29], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_DATA_VALID RO, FLASH_CON_BASE[30], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_BUSY RO, FLASH_CON_BASE[31], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

11.11 FLASH_ADDR_BASE

Bit	Name
0	FLASH_ADDR
1	FLASH_ADDR
2	FLASH_ADDR
3	FLASH_ADDR
4	FLASH_ADDR
5	FLASH_ADDR
6	FLASH_ADDR
7	FLASH_ADDR
8	FLASH_ADDR
9	FLASH_ADDR
10	FLASH_ADDR
11	FLASH_ADDR
12	FLASH_ADDR
13	FLASH_ADDR
14	FLASH_ADDR
15	FLASH_ADDR
16	FLASH_ADDR
17	FLASH_ADDR
18	FLASH_ADDR
19	FLASH_ADDR
20	FLASH_ADDR
21	FLASH_ADDR
22	FLASH_ADDR
23	FLASH_ADDR
24	FLASH_ADDR
25	FLASH_ADDR
26	FLASH_ADDR
27	FLASH_ADDR
28	FLASH_ADDR
29	FLASH_ADDR
30	FLASH_ADDR
31	FLASH_ADDR

FLASH_ADDR R/W, FLASH_ADDR_BASE[31..0], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

11.12 FLASH_DAT_BASE

Bit	Name
0	FLASH_DATA_OUT
1	FLASH_DATA_OUT
2	FLASH_DATA_OUT
3	FLASH_DATA_OUT
4	FLASH_DATA_OUT
5	FLASH_DATA_OUT
6	FLASH_DATA_OUT
7	FLASH_DATA_OUT
8	FLASH_DATA_IN
9	FLASH_DATA_IN
10	FLASH_DATA_IN
11	FLASH_DATA_IN
12	FLASH_DATA_IN
13	FLASH_DATA_IN
14	FLASH_DATA_IN
15	FLASH_DATA_IN
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

FLASH_DATA_OUT R/W, FLASH_DAT_BASE[7..0], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

FLASH_DATA_IN R/W, FLASH_DAT_BASE[15..8], Axion-CL

This bitfield is used to program the boards flash memory. It should not be programmed directly by customers.

11.13 TAP_DIPR_CONTROL

Bit	Name
0	DIPR_NUM_TAPS
1	DIPR_NUM_TAPS
2	DIPR_NUM_TAPS
3	DIPR_NUM_TAPS
4	Reserved
5	Reserved
6	Reserved
7	Reserved
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	DIPR_PIX_SIZE
13	Reserved
14	Reserved
15	DIPR_EN
16	DIPR_MASK
17	DIPR_MASK
18	DIPR_MASK
19	DIPR_MASK
20	DIPR_MASK
21	DIPR_MASK
22	DIPR_MASK
23	DIPR_MASK
24	DIPR_MASK
25	DIPR_MASK
26	DIPR_MASK
27	DIPR_MASK
28	DIPR_MASK
29	DIPR_MASK
30	DIPR_MASK
31	DIPR_MASK

DIPR_NUM_TAPS R/W, TAP_DIPR_CONTROL[3..0], Axion-CL

When the board is using the DIPR (Dedicated Interleaved Pixel Router), this register should be set to the number of taps the camera is putting out.

DIPR_PIX_SIZE R/W, TAP_DIPR_CONTROL[11], Axion-CL

When the board is using the DIPR, set this bit to indicate the pixels size. Set to 0 for 8-bit pixels, set to 1 for 10-bit to 16-bit pixels.

DIPR_EN R/W, TAP_DIPR_CONTROL[15], Axion-CL

Set this bit to 1 to enable the DIPR (Dedicated Interleaved Pixel Router). The DIPR bypasses the normal Axion tap reformatter, and just interleaves the incoming pixels. This only works on cameras that output pixel interleaved tap format (e.g. 1X8-1Y). The advantage of the DIPR is that is can handle unlimited line sizes, whereas the normal tap reformatter has a maximum line size of 32K Bytes.

DIPR_MASK R/W, TAP_DIPR_CONTROL[31..16], Axion-CL

When the board is using the DIPR, use this mask bitfield to “mask off” (i.e. set to zero) bits that are not provided by the camera. For example, when capturing 10 bit pixels, the bitfield should be set to 0xFC00.

Axion Power and Miscellaneous Registers

Chapter 12

12.1 Introduction

This chapter contains details on the Axion PoCL registers as well as some other miscellaneous registers.

12.2 CON104

Bit	Name
0	0_POCL_EN_POWER
1	Reserved
2	0_POCL_EN_CAM_SENSE
3	0_POCL_HW_DIS
4	Reserved
5	0_POCL_OPEN_DETECTED
6	0_POCL_OVER_DETECTED
7	0_POCL_OVER_LATCH
8	Reserved
9	0_CL_CLOCK_LOST_LATCH
10	0_CL_CLOCK_DETECTED
11	0_POCL_STATE
12	0_POCL_STATE
13	0_POCL_OVR_AUTO_RESTART
14	0_POCL_SENSE_BYPASS
15	0_ENABLE_POCL_SYSTEM
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

0_POCL_EN_POWER RO, CON104[0], Axion-CL

PoCL power has been applied to the camera.

0_POCL_EN_CAM_SENSE RO, CON104[2], Axion-CL

PoCL sense is enabled.

0_POCL_HW_DIS RO, CON104[3], Axion-CL

Describe 0_POCL_HW_DIS.

0_POCL_OPEN_DETECTED RO, CON104[5], Axion-CL

Open circuit detected.

0_POCL_OVER_DETECTED RO, CON104[6], Axion-CL

Over current detected.

0_POCL_OVER_LATCH RO, CON104[7], Axion-CL

Latched to 1 if over current has been detected.

0_CL_CLOCK_LOST_LATCH RO, CON104[9], Axion-CL

Latched to one if CL clock is lost.

0_CL_CLOCK_DETECTED RO, CON104[10], Axion-CL

Reads back 1 if CL clock is detected.

0_POCL_STATE RO, CON104[12..11], Axion-CL

Current state of the PoCL state machine.

0_POCL_OVR_AUTO_RESTART R/W, CON104[13], Axion-CL

Automatically restart if over current detected.

0_POCL_SENSE_BYPASS R/W, CON104[14], Axion-CL

Bypass the PoCL sense circuit and apply power. This register is for testing only, it should not be set by the user.

0_ENABLE_POCL_SYSTEM R/W, CON104[15], Axion-CL

Poking this bit to 1 enables the PoCL circuit for this connector.

12.3 CON105

Bit	Name
0	0_POCL_TIMER_OFF
1	0_POCL_TIMER_OFF
2	0_POCL_TIMER_OFF
3	0_POCL_TIMER_OFF
4	0_POCL_TIMER_OFF
5	0_POCL_TIMER_OFF
6	0_POCL_TIMER_OFF
7	0_POCL_TIMER_OFF
8	0_POCL_TIMER_OFF
9	0_POCL_TIMER_OFF
10	0_POCL_TIMER_OFF
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	0_POCL_TIMER_STABLE
17	0_POCL_TIMER_STABLE
18	0_POCL_TIMER_STABLE
19	0_POCL_TIMER_STABLE
20	0_POCL_TIMER_STABLE
21	0_POCL_TIMER_STABLE
22	0_POCL_TIMER_STABLE
23	0_POCL_TIMER_STABLE
24	0_POCL_TIMER_STABLE
25	0_POCL_TIMER_STABLE
26	0_POCL_TIMER_STABLE
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

0_POCL_TIMER_OFF R/W, CON105[10..0], Axion-CL

This bitfield controls the timing of the PoCL state machine. It should not be changed by the user unless instructed by BitFlow Customer Support.

0_POCL_TIMER_STABLE R/W, CON105[26..16], Axion-CL

This bitfield controls the timing of the PoCL state machine. It should not be changed by the user unless instructed by BitFlow Customer Support.

12.4 CON106

Bit	Name
0	0_POCL_TIMER_ON
1	0_POCL_TIMER_ON
2	0_POCL_TIMER_ON
3	0_POCL_TIMER_ON
4	0_POCL_TIMER_ON
5	0_POCL_TIMER_ON
6	0_POCL_TIMER_ON
7	0_POCL_TIMER_ON
8	0_POCL_TIMER_ON
9	0_POCL_TIMER_ON
10	0_POCL_TIMER_ON
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	0_POCL_TIMER_DISCONNECT
17	0_POCL_TIMER_DISCONNECT
18	0_POCL_TIMER_DISCONNECT
19	0_POCL_TIMER_DISCONNECT
20	0_POCL_TIMER_DISCONNECT
21	0_POCL_TIMER_DISCONNECT
22	0_POCL_TIMER_DISCONNECT
23	0_POCL_TIMER_DISCONNECT
24	0_POCL_TIMER_DISCONNECT
25	0_POCL_TIMER_DISCONNECT
26	0_POCL_TIMER_DISCONNECT
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

0_POCL_TIMER_ON R/W, CON106[10..0], Axion-CL

This bitfield controls the timing of the PoCL state machine. It should not be changed by the user unless instructed by BitFlow Customer Support.

0_POCL_TIMER_DISCONNECT R/W, CON106[26..16], Axion-CL

This bitfield controls the timing of the PoCL state machine. It should not be changed by the user unless instructed by BitFlow Customer Support.

12.5 CON136

Bit	Name
0	1_POCL_EN_POWER
1	Reserved
2	1_POCL_EN_CAM_SENSE
3	1_POCL_HW_DIS
4	Reserved
5	1_POCL_OPEN_DETECTED
6	1_POCL_OVER_DETECTED
7	1_POCL_OVER_LATCH
8	Reserved
9	1_CL_CLOCK_LOST_LATCH
10	1_CL_CLOCK_DETECTED
11	1_POCL_STATE
12	1_POCL_STATE
13	1_POCL_OVR_AUTO_RESTART
14	1_POCL_SENSE_BYPASS
15	1_ENABLE_POCL_SYSTEM
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

1_POCL_EN_POWER RO, CON136[0], Axion-CL

See 0_POCL_EN_POWER.

1_POCL_EN_CAM_SENSE RO, CON136[2], Axion-CL

See 0_POCL_EN_CAM_SENSE.

1_POCL_HW_DIS RO, CON136[3], Axion-CL

See 0_POCL_HW_DIS.

1_POCL_OPEN_DETECTED RO, CON136[5], Axion-CL

See 0_POCL_OPEN_DETECTED.

1_POCL_OVER_DETECTED RO, CON136[6], Axion-CL

See 0_POCL_OVER_DETECTED.

1_POCL_OVER_LATCH RO, CON136[7], Axion-CL

See 0_POCL_OVER_LATCH.

1_CL_CLOCK_LOST_LATCH RO, CON136[9], Axion-CL

See 0_CL_CLOCK_LOST_LATCH.

1_CL_CLOCK_DETECTED RO, CON136[10], Axion-CL

See 0_CL_CLOCK_DETECTED.

1_POCL_STATE RO, CON136[12..11], Axion-CL

See 0_POCL_STATE.

1_POCL_OVR_AUTO_RESTART R/W, CON136[13], Axion-CL

See 0_POCL_OVR_AUTO_RESTART.

1_POCL_SENSE_BYPASS R/W, CON136[14], Axion-CL

See 0_POCL_SENSE_BYPASS.

1_ENABLE_POCL_SYSTEM R/W, CON136[15], Axion-CL

See 0_ENABLE_POCL_SYSTEM.

12.6 CON137

Bit	Name
0	1_POCL_TIMER_OFF
1	1_POCL_TIMER_OFF
2	1_POCL_TIMER_OFF
3	1_POCL_TIMER_OFF
4	1_POCL_TIMER_OFF
5	1_POCL_TIMER_OFF
6	1_POCL_TIMER_OFF
7	1_POCL_TIMER_OFF
8	1_POCL_TIMER_OFF
9	1_POCL_TIMER_OFF
10	1_POCL_TIMER_OFF
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	1_POCL_TIMER_STABLE
17	1_POCL_TIMER_STABLE
18	1_POCL_TIMER_STABLE
19	1_POCL_TIMER_STABLE
20	1_POCL_TIMER_STABLE
21	1_POCL_TIMER_STABLE
22	1_POCL_TIMER_STABLE
23	1_POCL_TIMER_STABLE
24	1_POCL_TIMER_STABLE
25	1_POCL_TIMER_STABLE
26	1_POCL_TIMER_STABLE
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

1_POCL_TIMER_OFF R/W, CON137[10..0], Axion-CL

See 0_POCL_TIMER_OFF.

1_POCL_TIMER_STABLE R/W, CON137[26..16], Axion-CL

See 0_POCL_TIMER_STABLE.

12.7 CON138

Bit	Name
0	1_POCL_TIMER_ON
1	1_POCL_TIMER_ON
2	1_POCL_TIMER_ON
3	1_POCL_TIMER_ON
4	1_POCL_TIMER_ON
5	1_POCL_TIMER_ON
6	1_POCL_TIMER_ON
7	1_POCL_TIMER_ON
8	1_POCL_TIMER_ON
9	1_POCL_TIMER_ON
10	1_POCL_TIMER_ON
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	1_POCL_TIMER_DISCONNECT
17	1_POCL_TIMER_DISCONNECT
18	1_POCL_TIMER_DISCONNECT
19	1_POCL_TIMER_DISCONNECT
20	1_POCL_TIMER_DISCONNECT
21	1_POCL_TIMER_DISCONNECT
22	1_POCL_TIMER_DISCONNECT
23	1_POCL_TIMER_DISCONNECT
24	1_POCL_TIMER_DISCONNECT
25	1_POCL_TIMER_DISCONNECT
26	1_POCL_TIMER_DISCONNECT
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

1_POCL_TIMER_ON R/W, CON138[10..0], Axion-CL

See 0_POCL_TIMER_ON.

1_POCL_TIMER_DISCONNECT R/W, CON138[26..16], Axion-CL

See 0_POCL_TIMER_DISCONNECT.

12.8 CON168

Bit	Name
0	2_POCL_EN_POWER
1	Reserved
2	2_POCL_EN_CAM_SENSE
3	2_POCL_HW_DIS
4	Reserved
5	2_POCL_OPEN_DETECTED
6	2_POCL_OVER_DETECTED
7	2_POCL_OVER_LATCH
8	Reserved
9	2_CL_CLOCK_LOST_LATCH
10	2_CL_CLOCK_DETECTED
11	2_POCL_STATE
12	2_POCL_STATE
13	2_POCL_OVR_AUTO_RESTART
14	2_POCL_SENSE_BYPASS
15	2_ENABLE_POCL_SYSTEM
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

2_POCL_EN_POWER RO, CON168[0], Axion-CL

See 0_POCL_EN_POWER.

2_POCL_EN_CAM_SENSE RO, CON168[2], Axion-CL

See 0_POCL_EN_CAM_SENSE.

2_POCL_HW_DIS RO, CON168[3], Axion-CL

See 0_POCL_HW_DIS.

2_POCL_OPEN_DETECTED RO, CON168[5], Axion-CL

See 0_POCL_OPEN_DETECTED.

2_POCL_OVER_DETECTED RO, CON168[6], Axion-CL

See 0_POCL_OVER_DETECTED.

2_POCL_OVER_LATCH RO, CON168[7], Axion-CL

See 0_POCL_OVER_LATCH.

2_CL_CLOCK_LOST_LATCH RO, CON168[9], Axion-CL

See 0_CL_CLOCK_LOST_LATCH.

2_CL_CLOCK_DETECTED RO, CON168[10], Axion-CL

See 0_CL_CLOCK_DETECTED.

2_POCL_STATE RO, CON168[12..11], Axion-CL

See 0_POCL_STATE.

2_POCL_OVR_AUTO_RESTART R/W, CON168[13], Axion-CL

See 0_POCL_OVR_AUTO_RESTART.

2_POCL_SENSE_BYPASS R/W, CON168[14], Axion-CL

See 0_POCL_SENSE_BYPASS.

2_ENABLE_POCL_SYSTEM R/W, CON168[15], Axion-CL

See 0_ENABLE_POCL_SYSTEM.

12.9 CON169

Bit	Name
0	2_POCL_TIMER_OFF
1	2_POCL_TIMER_OFF
2	2_POCL_TIMER_OFF
3	2_POCL_TIMER_OFF
4	2_POCL_TIMER_OFF
5	2_POCL_TIMER_OFF
6	2_POCL_TIMER_OFF
7	2_POCL_TIMER_OFF
8	2_POCL_TIMER_OFF
9	2_POCL_TIMER_OFF
10	2_POCL_TIMER_OFF
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	2_POCL_TIMER_STABLE
17	2_POCL_TIMER_STABLE
18	2_POCL_TIMER_STABLE
19	2_POCL_TIMER_STABLE
20	2_POCL_TIMER_STABLE
21	2_POCL_TIMER_STABLE
22	2_POCL_TIMER_STABLE
23	2_POCL_TIMER_STABLE
24	2_POCL_TIMER_STABLE
25	2_POCL_TIMER_STABLE
26	2_POCL_TIMER_STABLE
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

2_POCL_TIMER_OFF R/W, CON169[10..0], Axion-CL

See 0_POCL_TIMER_OFF.

2_POCL_TIMER_STABLE R/W, CON169[26..16], Axion-CL

See 0_POCL_TIMER_STABLE.

12.10 CON170

Bit	Name
0	2_POCL_TIMER_ON
1	2_POCL_TIMER_ON
2	2_POCL_TIMER_ON
3	2_POCL_TIMER_ON
4	2_POCL_TIMER_ON
5	2_POCL_TIMER_ON
6	2_POCL_TIMER_ON
7	2_POCL_TIMER_ON
8	2_POCL_TIMER_ON
9	2_POCL_TIMER_ON
10	2_POCL_TIMER_ON
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	2_POCL_TIMER_DISCONNECT
17	2_POCL_TIMER_DISCONNECT
18	2_POCL_TIMER_DISCONNECT
19	2_POCL_TIMER_DISCONNECT
20	2_POCL_TIMER_DISCONNECT
21	2_POCL_TIMER_DISCONNECT
22	2_POCL_TIMER_DISCONNECT
23	2_POCL_TIMER_DISCONNECT
24	2_POCL_TIMER_DISCONNECT
25	2_POCL_TIMER_DISCONNECT
26	2_POCL_TIMER_DISCONNECT
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

2_POCL_TIMER_ON R/W, CON170[10..0], Axion-CL

See 0_POCL_TIMER_ON.

2_POCL_TIMER_DISCONNECT R/W, CON170[26..16], Axion-CL

See 0_POCL_TIMER_DISCONNECT.

12.11 CON200

Bit	Name
0	3_POCL_EN_POWER
1	Reserved
2	3_POCL_EN_CAM_SENSE
3	3_POCL_HW_DIS
4	Reserved
5	3_POCL_OPEN_DETECTED
6	3_POCL_OVER_DETECTED
7	3_POCL_OVER_LATCH
8	Reserved
9	3_CL_CLOCK_LOST_LATCH
10	3_CL_CLOCK_DETECTED
11	3_POCL_STATE
12	3_POCL_STATE
13	3_POCL_OVR_AUTO_RESTART
14	3_POCL_SENSE_BYPASS
15	3_ENABLE_POCL_SYSTEM
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

3_POCL_EN_POWER RO, CON200[0], Axion-CL

See 0_POCL_EN_POWER.

3_POCL_EN_CAM_SENSE RO, CON200[2], Axion-CL

See 0_POCL_EN_CAM_SENSE.

3_POCL_HW_DIS RO, CON200[3], Axion-CL

See 0_POCL_HW_DIS.

3_POCL_OPEN_DETECTED RO, CON200[5], Axion-CL

See 0_POCL_OPEN_DETECTED.

3_POCL_OVER_DETECTED RO, CON200[6], Axion-CL

See 0_POCL_OVER_DETECTED.

3_POCL_OVER_LATCH RO, CON200[7], Axion-CL

See 0_POCL_OVER_LATCH.

3_CL_CLOCK_LOST_LATCH RO, CON200[9], Axion-CL

See 0_CL_CLOCK_LOST_LATCH.

3_CL_CLOCK_DETECTED RO, CON200[10], Axion-CL

See 0_CL_CLOCK_DETECTED.

3_POCL_STATE RO, CON200[12..11], Axion-CL

See 0_POCL_STATE.

3_POCL_OVR_AUTO_RESTART R/W, CON200[13], Axion-CL

See 0_POCL_OVR_AUTO_RESTART.

3_POCL_SENSE_BYPASS R/W, CON200[14], Axion-CL

See 0_POCL_SENSE_BYPASS.

3_ENABLE_POCL_SYSTEM R/W, CON200[15], Axion-CL

See 0_ENABLE_POCL_SYSTEM.

12.12 CON201

Bit	Name
0	3_POCL_TIMER_OFF
1	3_POCL_TIMER_OFF
2	3_POCL_TIMER_OFF
3	3_POCL_TIMER_OFF
4	3_POCL_TIMER_OFF
5	3_POCL_TIMER_OFF
6	3_POCL_TIMER_OFF
7	3_POCL_TIMER_OFF
8	3_POCL_TIMER_OFF
9	3_POCL_TIMER_OFF
10	3_POCL_TIMER_OFF
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	3_POCL_TIMER_STABLE
17	3_POCL_TIMER_STABLE
18	3_POCL_TIMER_STABLE
19	3_POCL_TIMER_STABLE
20	3_POCL_TIMER_STABLE
21	3_POCL_TIMER_STABLE
22	3_POCL_TIMER_STABLE
23	3_POCL_TIMER_STABLE
24	3_POCL_TIMER_STABLE
25	3_POCL_TIMER_STABLE
26	3_POCL_TIMER_STABLE
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

3_POCL_TIMER_OFF R/W, CON201[10..0], Axion-CL

See 0_POCL_TIMER_OFF.

3_POCL_TIMER_STABLE R/W, CON201[26..16], Axion-CL

See 0_POCL_TIMER_STABLE.

12.13 CON202

Bit	Name
0	3_POCL_TIMER_ON
1	3_POCL_TIMER_ON
2	3_POCL_TIMER_ON
3	3_POCL_TIMER_ON
4	3_POCL_TIMER_ON
5	3_POCL_TIMER_ON
6	3_POCL_TIMER_ON
7	3_POCL_TIMER_ON
8	3_POCL_TIMER_ON
9	3_POCL_TIMER_ON
10	3_POCL_TIMER_ON
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	3_POCL_TIMER_DISCONNECT
17	3_POCL_TIMER_DISCONNECT
18	3_POCL_TIMER_DISCONNECT
19	3_POCL_TIMER_DISCONNECT
20	3_POCL_TIMER_DISCONNECT
21	3_POCL_TIMER_DISCONNECT
22	3_POCL_TIMER_DISCONNECT
23	3_POCL_TIMER_DISCONNECT
24	3_POCL_TIMER_DISCONNECT
25	3_POCL_TIMER_DISCONNECT
26	3_POCL_TIMER_DISCONNECT
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

3_POCL_TIMER_ON R/W, CON202[10..0], Axion-CL

See 0_POCL_TIMER_ON.

3_POCL_TIMER_DISCONNECT R/W, CON202[26..16], Axion-CL

See 0_POCL_TIMER_DISCONNECT.

12.14 CON356

Bit	Name
0	FW_BUILD_YEAR
1	FW_BUILD_YEAR
2	FW_BUILD_YEAR
3	FW_BUILD_YEAR
4	FW_BUILD_YEAR
5	FW_BUILD_YEAR
6	FW_BUILD_YEAR
7	FW_BUILD_YEAR
8	FW_BUILD_YEAR
9	FW_BUILD_YEAR
10	FW_BUILD_YEAR
11	FW_BUILD_YEAR
12	FW_BUILD_YEAR
13	FW_BUILD_YEAR
14	FW_BUILD_YEAR
15	Reserved
16	FW_BUILD_DAY
17	FW_BUILD_DAY
18	FW_BUILD_DAY
19	FW_BUILD_DAY
20	FW_BUILD_DAY
21	FW_BUILD_DAY
22	FW_BUILD_DAY
23	FW_BUILD_DAY
24	FW_BUILD_MONTH
25	FW_BUILD_MONTH
26	FW_BUILD_MONTH
27	FW_BUILD_MONTH
28	FW_BUILD_MONTH
29	FW_BUILD_MONTH
30	FW_BUILD_MONTH
31	FW_BUILD_MONTH

FW_BUILD_YEAR RO, CON356[15..0], Karbon-CXP, Cyton-CXP

Year that this firmware was compiled in BCD format. Example: 0x2012 is year 2012

FW_BUILD_DAY RO, CON356[23..16], Karbon-CXP, Cyton-CXP

Day that this firmware was compiled in BCD format. Example: 0x18 the 18th of the month.

FW_BUILD_MONTH RO, CON356[31..24], Karbon-CXP, Cyton-CXP

Month that this firmware was compiled in BCD format. Example: 0x12 is december.

12.15 CON357

Bit	Name
0	FW_BUILD_MIN
1	FW_BUILD_MIN
2	FW_BUILD_MIN
3	FW_BUILD_MIN
4	FW_BUILD_MIN
5	FW_BUILD_MIN
6	FW_BUILD_MIN
7	FW_BUILD_MIN
8	FW_BUILD_HOUR
9	FW_BUILD_HOUR
10	FW_BUILD_HOUR
11	FW_BUILD_HOUR
12	FW_BUILD_HOUR
13	FW_BUILD_HOUR
14	FW_BUILD_HOUR
15	FW_BUILD_HOUR
16	FPGA_ID
17	FPGA_ID
18	FPGA_ID
19	FPGA_ID
20	FPGA_ID
21	FPGA_ID
22	FPGA_ID
23	FPGA_ID
24	FW_CMPTBL
25	FW_CMPTBL
26	FW_CMPTBL
27	FW_CMPTBL
28	FW_CMPTBL
29	FW_CMPTBL
30	FW_CMPTBL
31	FW_CMPTBL

FW_BUILD_MIN RO, CON357[7..0], Karbon-CXP, Cyton-CXP

Minute that this firmware was compiled. Example: 0x35 is 35 minutes past the hour.

FW_BUILD_HOUR RO, CON357[15..8], Karbon-CXP, Cyton-CXP

Hour that this firmware was compiled. Example: 0x23 is 11pm (23rd hour).

FPGA_ID RO, CON357[23..16], Karbon-CXP, Cyton-CXP

FPGA Identifier

FW_CMPTBL RO, CON357[31..24], Karbon-CXP, Cyton-CXP

Firmware compatibility version (must match SDK driver internal firmware version).

Specifications

Chapter 13

13.1 Introduction

This chapter describes the general specifications of the Axion-CL family. The numerical values for he specifications are listed in Table 13-1. If more information is available for a given specification t will be an entry in the column marked “Details”.

Table 13-1 Axion-CL Specifications

Specifications	Value	Units	Details
PCIe Compatibility, slot type	x1, x4, x8 and x16	Electrical slot size	Section 13.2
PCIe Compatibility, generation	Gen1, 2 and 3		Section 13.2
Maximum Input CL clock	85	MHz	
Minimum Input CL Rate	45	MHz	
Maximum Pixels Per Line	268,435,456	Pixels (8-bit)	Section 13.3
Maximum Lines Per Frame	16,777,216	Lines	Section 13.4
Image width granularity	16	bytes	
Image height granularity	1	line	
Minimum trigger pulse	10	Nanoseconds	
Minimum encoder pulse	10	Nanoseconds	
Axion-1xE Current (3.3 Volt)	0.7	Amps	Section 13.5
Axion-1xE Current (12 Volt)	2.8	Amps	Section 13.5
Axion-2xE Current (3.3 Volt)	0.75	Amps	Section 13.5
Axion-2xE Current (12 Volt)	0.30	Amps	Section 13.5
Axion-1xB Current (3.3 Volt)	0.42	Amps	Section 13.5
Axion-1xB Current (12 Volt)	0.22	Amps	Section 13.5
Axion-2xB Current (3.3 Volt)	0.53	Amps	Section 13.5
Axion-2xB Current (12 Volt)	0.26	Amps	Section 13.5
Axion-4xB Current (3.3 Volt)	0.63	Amps	Section 13.5
Axion-4xB Current (12 Volt)	0.34	Amps	Section 13.5
Temperature range	0 to 50	Degrees Celsius	
Humidity	25% to 80%		

Table 13-1 Axion-CL Specifications

Specifications	Value	Units	Details
Mechanical dimensions	6.8 x 4.2	Inches	
Mechanical dimensions	17.4 x 10.6	Centimeters	
Maximum PoCL Power @12 Volts	4	Watts	Per CL Connector
LVDS Drivers	SN65LVDS31D		
LVDS Receivers	SNLVDS3486		
TTL Drivers	SN74LVTH241		
TTL Receivers	SN74LVTH241		

13.2 PCI Express Compatibility

The Axion-CL is a PCIe x4 Gen 2 board. However, it will work in any PCIe slot that it fits into. This means it will work in x4, x8 and x16 slots, however, it will also work in x1 slots if these slots are mechanically compatibility with an x4 board, though performance will be greatly degraded. Similarly, the Axion-CL will work in Gen 1, Gen 2 and Gen 3 slot, however, performance will be degraded in a Gen 1 slot. Further, t is no performance gained by putting the Axion-CL in a Gen 3 slot, performance will be the same as a Gen 2 slot.

Note: For best DMA performance, put the Axion-CL in a PCIe x4, x8 or x16 Gen 2 or Gen 3 slot on a high quality motherboard.

13.3 Maximum Pixels Per Line

The value 268,435,456 applies to interleaved tap formats with 8-bit pixels. Interleaved tap format lines over 32,768 require the use of DIPR mode. With segmented tap formats, the maximum line size is 32,768 with 8-bit pixels.

13.4 Maximum Lines Per Frame

This limitation is for area scan cameras. For line scan cameras, the number of lines per frame is essentially unlimited. Please contact BitFlow customer support for more information.

13.5 Axion Power Requirements

The Axion-CL power requirements listed inTable 13-1 are the requirements of the board's circuitry only. In addition, the Axion-CL can provide up to 4 watts of power to each Camera Link connector. The 12 Volt rail of the PCIe bus cannot provide enough power if all CL connector links are drawing maximum power. The board has an auxiliary connector which can be use to provide additional PoCL power for these situations. There is a jumper which is used to switch the power source from the PCIe bus to the auxiliary connector. See Section 14.4 for more information on this jumper.

Note: If the total amount of all cameras connected to the Axion exceeds 15 Watts, then the auxiliary power connector must be used. For example, if two full Camera Link cameras each taking 4 watts perCL connector are connected, then auxiliary power should be used.

Mechanical

Chapter 14

14.1 Introduction

This chapter describes the mechanical characteristics of the Axion-CL This includes description of all of the connectors on the board and pin-outs for these connectors.

The mechanical layouts of the Axions are shown on the following pages. Please see the table for cross reference.

Table 14-1 Axion Block Diagrams

Model	Diagram
Axion-1xE	Figure 14-1
Axion-2xE	Figure 14-2
Axion-2xE 4th Connector	Figure 14-3
Axion-1xB	Figure 14-4
Axion-2xB	Figure 14-5
Axion-4xB	Figure 14-6
Axion-4xB 4th Connector	Figure 14-7

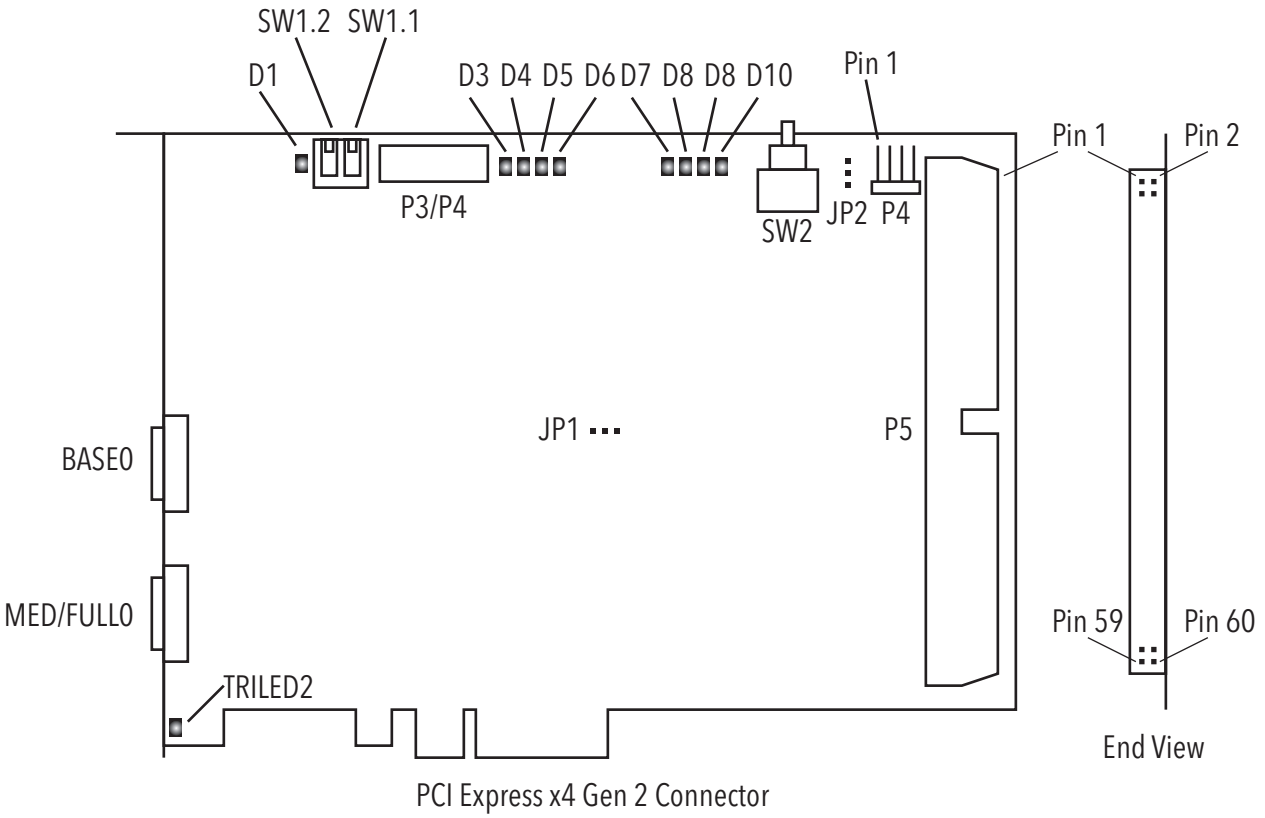

Figure 14-1 Axion-1xE Board Layout

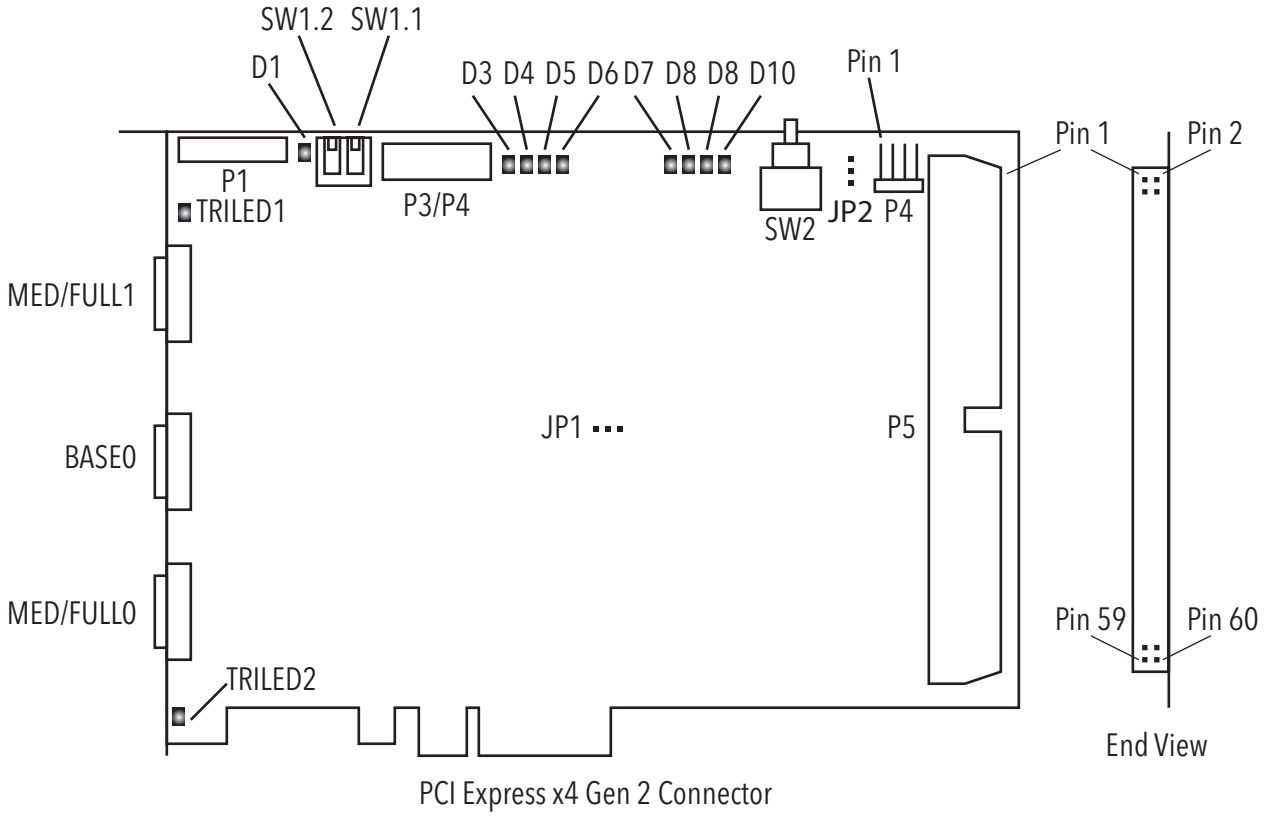

Figure 14-2 Axion-2xE Board Layout

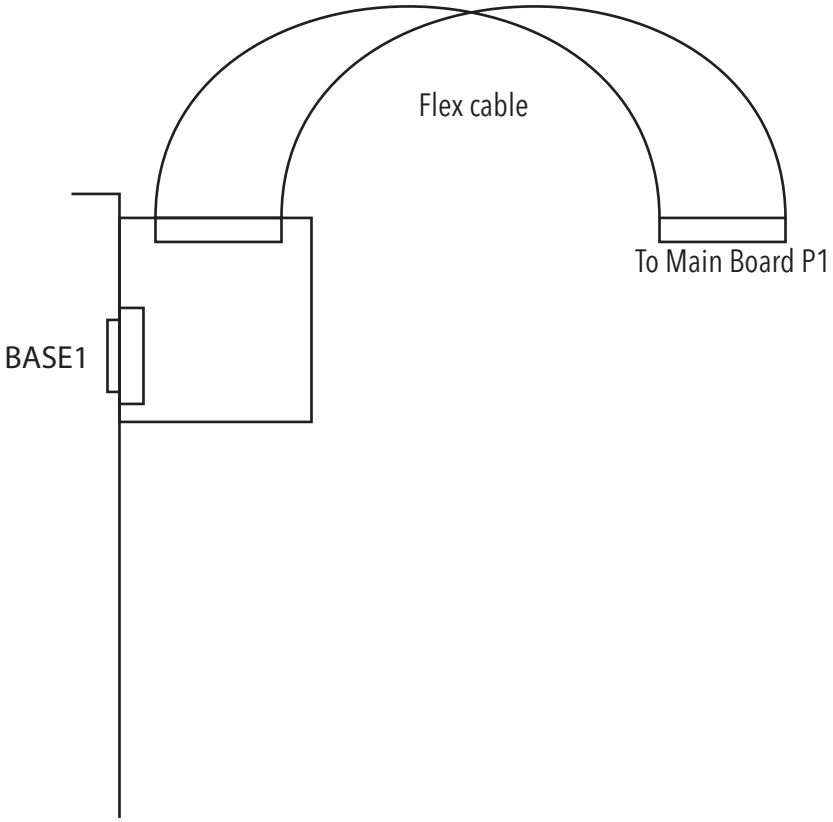

Figure 14-3 Axion-2xE Fourth CL Connector

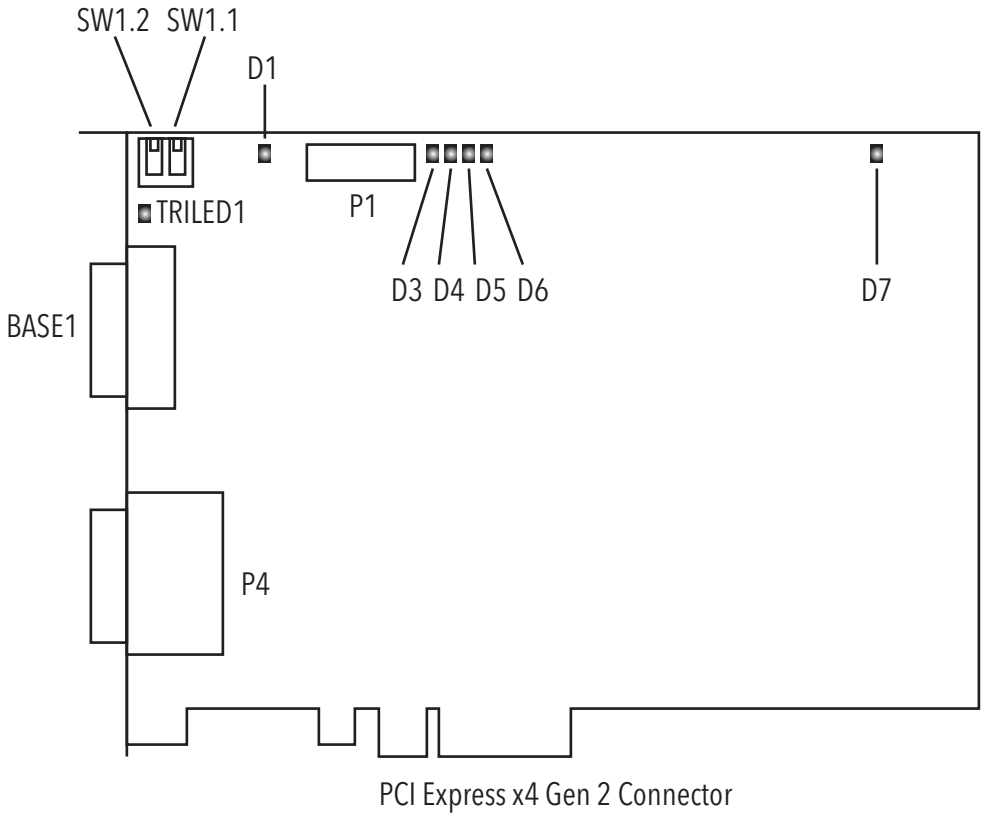

Figure 14-4 Axion-1xB Board Layout

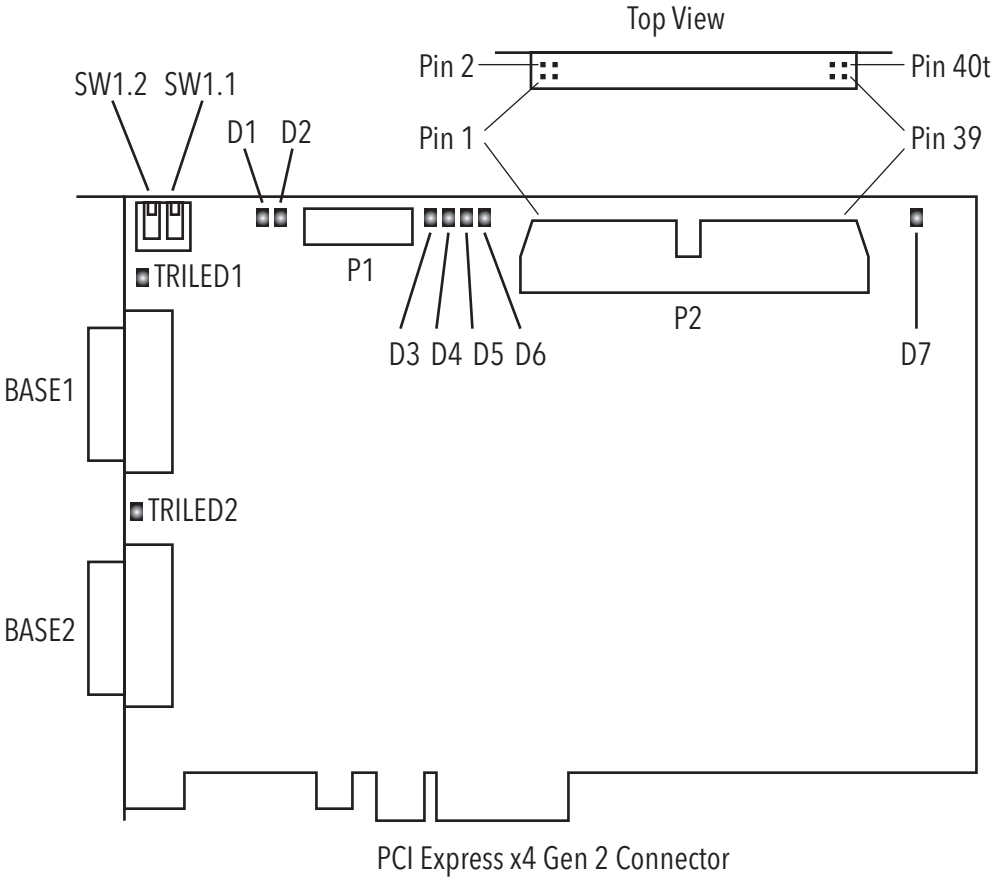

Figure 14-5 Axion-2xB Board Layout

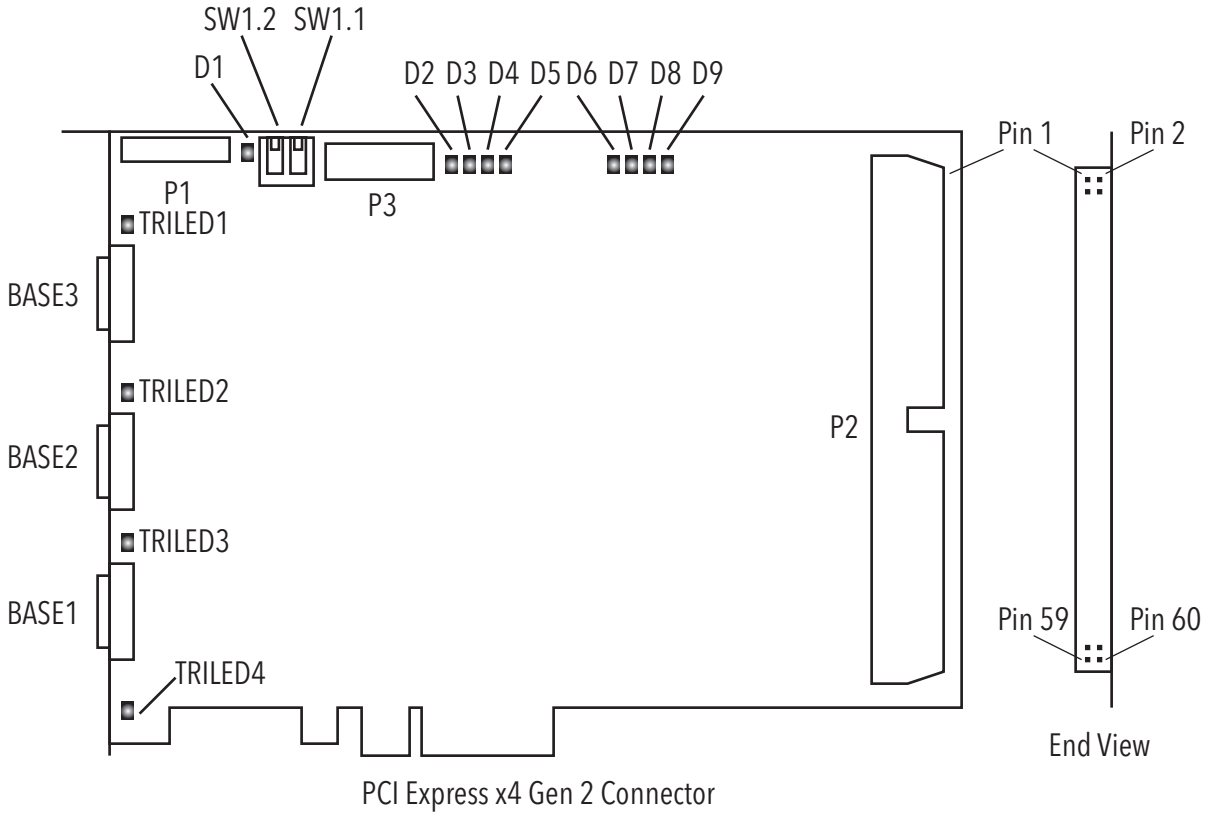

Figure 14-6 Axion-4xB Board Layout

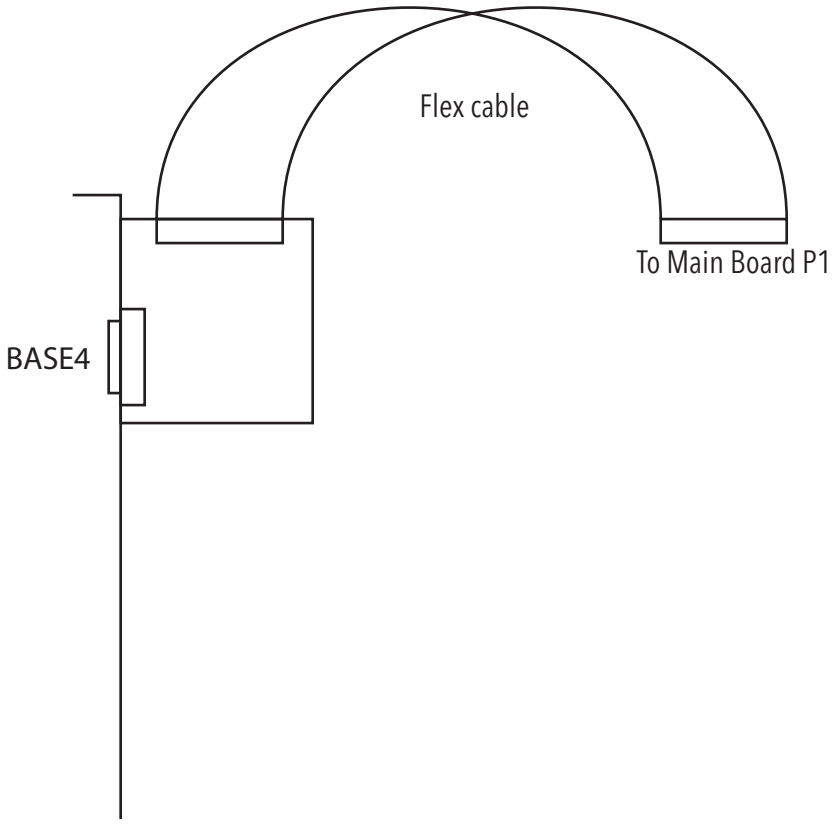

Figure 14-7 Axion-4xB Fourth CL Connector

14.2 The Axion-CL Connectors

Each model Axion has a slightly different arrangement of connectors. Table shows the connectors for each model.

Table 14-2 Axion Connectors

Model	Camera 1	Camera 2	Camera 3	Camera 4	4th Camera	I/O	BitBox
Axion-1xE	Base0, Med/Full0					P5	P3/P4*
Axion-2xE	Base0, Med/Full0	Base1, Med/Full1			P1	P5	P3/P4*
Axion-1xB	Base1					P4	P1
Axion-2xB	Base1	Base2				P2	P1
Axion-4xB	Base1	Base2	Base3	Base4	P1	P2	P3

* This connector is labeled P4 on the Rev 1.x boards and P3 on the Rev 2.x and later boards.

Figure 14-1 through Figure 14-7 show the locations of these connectors. The following sections show the details of each of these connectors.

14.2.1 The CL Connectors

The CL connectors are for connecting Camera Link cameras. The Axion-CL uses both MDR and SDR connectors. These connectors are fully compliant with the Camera Link version 1.1 and later specification. See Table 14-3 to see which model has whic kind of connectors

Table 14-3 CL Connector Types

Model	Connector
Axion-1xE	SDR
Axion-2xE	SDR
Axion-1xB	MDR
Axion-2xB	MDR
Axion-4xB	SDR

Table 14-4 illustrates how to connect the Axion-1xE and 2xE models to various types and numbers of Camera Link Cameras.

Table 14-4 Camera Link Connectors - Axion-1xE and Axion 2xE

Camera(s)	BASE0	MED/FULL0	BASE1	MED/FULL1
One Base CL Camera	Camera 0 - CL1			
One Medium CL Camera	Camera 0 - CL1	Camera 0 - CL2		
One Full CL Camera	Camera 0 - CL1	Camera 0 - CL2		
One 80-bit CL Camera	Camera 0 - CL1	Camera 0 - CL2		
Two Base CL Cameras	Camera 0 - CL1		Camera 1 - CL1	
Two Medium CL Cameras	Camera 0 - CL1	Camera 0 - CL2	Camera 1 - CL1	Camera 1 - CL2
Two Full CL Cameras	Camera 0 - CL1	Camera 0 - CL2	Camera 1 - CL1	Camera 1 - CL2
Two 80-bit CL Cameras	Camera 0 - CL1	Camera 0 - CL2	Camera 1 - CL1	Camera 1 - CL2

14.3 Switches

There is one piano-type switch block, SW1, on the Axion-CL with two switches. These are used to identify individual boards when there is more than one board in a system. The idea is to set the switches differently on each board in the system. The switch settings can be read for each board from software (by reading the SW bitfield). SysReg also shows the switch setting for each board. See Table 14-5 below shows the switch settings and the corresponding value in the SW bitfield.

Table 14-5 Switch S1 Setting

SW1.2	SW1.1	SW register
up	up	0
up	down	1
down	up	2
down	down	3

14.4 Jumpers

The Axion-1xE and Axion 2xE Rev 1.x have jumpers. They are described below. The Axion Rev 2.x and later do not have any jumpers.

14.4.1 Jumper JP1

JP1 is used for diagnostic purposed. Please to no make changes to this jumper unless instructed to do so by BitFlow's Customer Support team.

14.4.2 Jumper JP2

T is one user configurable jumper on the Axion-CL, it controls the source of the power that is provided to the Camera Link camera(s) connected to the Axion (PoCL power). Table 14-6 shows the two settings.

Table 14-6 Jumper JP2

Position	Meaning
Top	PoCL power comes from the PCIe connector. Use when total PoCL power on all connectors is less than 15 watts.
Bottom	PoCL power comes from P4. Use when total PoCL power on all connectors exceeds 15 watts.

If jumper JP2 is in the bottom position, the connector P4 is use to provide PoCL power to the cameras. Contact BitFlow for adapter cables to connect internal PC power to P4.

Note: If the total amount of all cameras connected to the Axion exceeds 15 Watts, then the auxiliary power connector must be used.

14.5 LEDs

The Axions have a number of LEDS depending on the mode. Table 14-7 Describes the function of these LEDS for each model.

Table 14-7 LEDs

1xE	2xE	1xB	2xB	4xB	Color	Function
D1	D1	D7	D7	D1	Green	FPGA Configured
D3	D3	D3	D3	D2	Blue	General purpose, see register LED_BLUE
D4	D4	D4	D4	D3	Red	General purpose, see register LED_RED
D5	D5	D5	D5	D4	Orange	General purpose, see register LED_ORANGE
D6	D6	D6	D6	D5	Green	General purpose, see register LED_GREEN
D7	D7	D1	D1	D6	Green	VFG0 Status, see register SEL_LED
	D8		D2	D7	Green	VFG1 Status, see register SEL_LED
				D8	Green	VFG2 Status, see register SEL_LED
				D9	Green	VFG3 Status, see register SEL_LED
TRILED2	TRILED2	TRILED1	TRILED1	TRILED3	Various	Camera 1 status
	TRILED1		TRILED2	TRILED2	Various	Camera 2 status
				TRILED1	Various	Camera 3 status
				TRILED4	Various	Camera 4 status

Note: The general purpose LEDs are shared between VFGs on the Axions with more than one VFG. These LEDS are controlled by dedicated registers (e.g. LED_BLUE). The registers from each VFG on the mulit-VFG Axions are ORed together. The following table show ena example for one LED on aboard with two VFGs.

Table 14-8 LED control from two VFGs

VFG0 LED_BLUE	VFG1 LED_BLUE	LED
0	0	Off
0	1	On
1	0	On
1	1	On

14.5.1 Camera Status LEDS

The Camera Status LEDS change color and are blinking or steady depending on the status of the connected cameras and the camera. There is one status LED for each cameras. Cameras that are using two connectors still only get one status LED. See Table 14-9 for detailed information on the meaning of these LEDs.

Table 14-9 Camera Status LEDS Explanation

Color	State	Meaning
Blue	Blinking	Power sense mode, searching for PoCL camera, no power is applied
Blue	Steady	PoCL camera is seen, PoCL power is applied
Green	Blinking	CL clock is detected
Green	Steady	LVAL from camera is detected

Note: Green takes precendence over blue, this means once the CL clock is detected, regardless of the PoCL status, the LED will switch to green.

Note: The CL clock detection system takes into account what type of camera the board has been configured for. For example, if the board is set up for a Medium format CL camera, then LED will only go green if it sees a clock on both the X and Y CL chips.

14.6 Button

The Axion-1xE and 2xE Rev 1.x have a general purpose button, SW2, that can be routed to many different destinations. The purpose of the button is primarily to help debug I/O problems. It can be used as a trigger, encoder, or I/O that is routed off the board. Please see Section 2.1 for more information on how the button can be routed.

Note: The Axion Rev 2.x models do not have this button.

14.7 The Auxiliary Power Connector

The Axion-1xE and Axion-2xE Rev 1.x have an auxiliary power connector to provide power to PoCL cameras directly from the PC's main power supply. In general this is not necessary as the PCIe connector can provide plenty of power for any connector cameras.

Note: The Axion Rev 2.x and later models do not have this auxiliary power connector.

For cameras that require more power than can be provide by the PCIe bus, the Axion has a connector, P4, which can take auxiliary power from the PC's power supply. The pin out for this connector is shown in Table 14-10.

Note: Connector P4 is compatible with Berg 4-pin peripheral connectors available in many PCs. This connector is also known as the "floppy connector". For PCs that do have this type of connector, BitFlow offers an adapter cable that goes between P4 and a standard Molex 4-pin peripheral connector available in almost all PCs.

Note: Jumper JP2 must be in set to the correct position to route power from this connector to the Camera Link connectors.

Table 14-10 Auxiliary Power Connector

Pin	Voltage
1	NC
2	GND
3	GND
4	12 Volts

14.8 The BitBox Box Connector

This connector is for the BitBox, and I/O break out box that is available from BitFlow. Please contact BitFlow for more information.

14.9 I/O Connector Pinout for the Axion-1xE, Axion-2xE and Axion-4xB

The pin-out for the I/O Connector for the Axion-1xE, Axion2xE and Axion-4xB are illustrated in the Table 14-11.

Note: Signal names start with the Virtual Frame Grabber (VFG) that they are routed to. For example, the signal VFG0_TRIGGER_TTL is wired to VFG0., while VFG2_TRIGGER_TTL is wired to VGF2.

Note: Not all signals are available on all models. For example, the Axion-1xE, with only one VFG, will not have signals that start with "VFG1", "VFG2" or "VFG3"

Table 14-11 I/O Connector for the Axion-1xE, 2xE and 4xB

Pin	I/O	Signal	Comment
1	In	VFG0_TRIGGER+	LVDS
2	In	VFG0_TRIGGER-	LVDS
3	In	VFG0_ENCODERA+	LVDS
4	In	VFG0_ENCODERA-	LVDS
5	In	VFG0_ENCODERB+	LVDS
6	In	VFG0_ENCODERB-	LVDS
7	In	VFG1_TRIGGER+	LVDS
8	In	VFG1_TRIGGER-	LVDS
9	In	VFG1_ENCODERA+	LVDS
10	In	VFG1_ENCODERA-	LVDS
11	In	VFG1_ENCODERB+	LVDS
12	In	VFG1_ENCODERB-	LVDS
13	In	VFG2_TRIGGER+	LVDS
14	In	VFG2_TRIGGER-	LVDS
15	In	VFG2_ENCODERA+	LVDS
16	In	VFG2_ENCODERA-	LVDS
17	In	VFG2_ENCODERB+	LVDS
18	In	VFG2_ENCODERB-	LVDS
19	In	VFG3_TRIGGER+	LVDS
20	In	VFG3_TRIGGER-	LVDS
21	In	VFG3_ENCODERA+	LVDS
22	In	VFG3_ENCODERA-	LVDS
23	In	VFG3_ENCODERB+	LVDS
24	In	VFG3_ENCODERB-	LVDS
25		GND	
26	Out	VFG0_CC3+	LVDS
27	Out	VFG0_CC3-	LVDS
28	Out	VFG1_CC3+	LVDS
29	Out	VFG1_CC3-	LVDS

Table 14-11 I/O Connector for the Axion-1xE, 2xE and 4xB

Pin	I/O	Signal	Comment
30	Out	VFG2_CC3+	LVDS
31	Out	VFG2_CC3-	LVDS
32	Out	VFG3_CC3+	LVDS
33	Out	VFG3_CC3-	LVDS
34		GND	
35	In	VFG0_TRIGGER_TTL	TTL
36	In	VFG0_ENCODERA_TTL	TTL
37	In	VFG0_ENCODERB_TTL	TTL
38	In	VFG1_TRIGGER_TTL	TTL
39	In	VFG1_ENCODERA_TTL	TTL
40	In	VFG1_ENCODERB_TTL	TTL
41	In	VFG2_TRIGGER_TTL	TTL
42	In	VFG2_ENCODERA_TTL	TTL
43	In	VFG2_ENCODERB_TTL	TTL
44	In	VFG3_TRIGGER_TTL	TTL
45	In	VFG3_ENCODERA_TTL	TTL
46	In	VFG3_ENCODERB_TTL	TTL
47		Reserved	
48	Out	VFG0_CC3_TTL	TTL
49	Out	VFG0_CC4_TTL	TTL
50	Out	VFG0_CC2_TTL	TTL
51	Out	VFG1_CC3_TTL	TTL
52	Out	VFG1_CC4_TTL	TTL
53	Out	VFG1_CC2_TTL	TTL
54	Out	VFG2_CC3_TTL	TTL
55	Out	VFG2_CC4_TTL	TTL
56	Out	VFG2_CC2_TTL	TTL
57	Out	VFG3_CC3_TTL	TTL
58	Out	VFG3_CC4_TTL	TTL
59	Out	VFG3_CC2_TTL	TTL
60		GND	

14.10 I/O Connector Pinout for the Axion-1xB

The pin-out for the I/O Connector for the Axion-1xB is illustrated in the Table 14-12.

Table 14-12 I/O Connector for the Axion-1xB

Pin	I/O	Signal	Comment
1	In	VFG0_TRIGGER+	LVDS
2	In	VFG0_ENCODERA+	LVDS
3	In	VFG0_TRIGGER_TTL	TTL
4	Out	VFG0_CC3_TTL	
5		GND	
6	In	VFG0_ENCODERA_TTL	TTL
7	Out	VFG0_CC3+	LVDS
8	In	VFG0_ENCODERB+	LVDS
9	In	VFG0_TRIGGER-	LVDS
10	In	VFG0_ENCODERA-	LVDS
11	Out	VFG0_CC2_TTL	TTL
12	Out	VFG0_CC4_TTL	TTL
13	In	VFG0_ENCODERB_TTL	TTL
14	Out	VFG0_CC3-	LVDS
15	In	VFG0_ENCODERB-	LVDS

14.11 I/O Connector Pinout for the Axion-2xB

The pin-out for the I/O Connector for the Axion-2xB is illustrated in the Table 14-13.

Table 14-13 I/O Connector for the Axion-2xB

Pin	I/O	Signal	Comment
1	In	VFG0_TRIGGER+	LVDS
2	In	VFG0_TRIGGER-	LVDS
3	In	VFG0_ENCODERA+	LVDS
4	In	VFG0_ENCODERA-	LVDS
5	In	VFG0_ENCODERB+	LVDS
6	In	VFG0_ENCODERB-	LVDS
7	Out	VFG0_CC3+	LVDS
8	Out	VFG0_CC3-	LVDS
9		GND	
10	In	VFG0_ENCODERA_TTL	TTL
11	In	VFG0_TRIGGER_TTL	TTL
13	Out	VFG0_CC3_TTL	TTL
14	Out	VFG0_CC4_TTL	TTL
15	Out	VFG0_CC2_TTL	TTL
16		GND	
17	In	VFG1_TRIGGER+	LVDS
18	In	VFG1_TRIGGER-	LVDS
19	In	VFG1_ENCODERA+	LVDS
20	In	VFG1_ENCODERA-	LVDS
21	In	VFG1_ENCODERB+	LVDS
22	In	VFG1_ENCODERB-	LVDS
23	Out	VFG1_CC3+	LVDS
24	Out	VFG1_CC3-	LVDS
25	In	VFG1_TRIGGER_TTL	TTL
26	In	VFG1_ENCODERA_TTL	TTL
27	Out	VFG1_CC3_TTL	TTL
28	In	VFG1_ENCODERB_TTL	TTL

Table 14-13 I/O Connector for the Axion-2xB

Pin	I/O	Signal	Comment
29	Out	VFG1_CC2_TTL	TTL
30	Out	VFG1_CC4_TTL	TTL
31		GND	
32		GND	

Index

Numerics

A

B

C

D

E

F

I

J

L

M

N

P

Q

R

S

T

U

V

W

X

Y

Z