

ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Visit our website - *Click HERE*

The Karbon-CL

Hardware Reference Manual

Preliminary

BitFlow, Inc.
300 Wildwood Ave
Woburn, MA 01801
USA
Tel: 781-932-2900
Fax: 781-933-9965
Email: support@bitflow.com
Web: www.bitflow.com
Revision F.0

© 2006 BitFlow, Inc. All Rights Reserved.

This document, in whole or in part, may not be copied, photocopied, reproduced, translated or reduced to any other electronic medium or machine readable form without the prior written consent of BitFlow, Inc.

BitFlow, Inc. makes no implicit warranty for the use of its products and assumes no responsibility for any errors that may appear in this document, nor does it make a commitment to update the information contained herein.

BitFlow, Inc. retains the right to make changes to these specifications at any time without notice.

All trademarks are properties of their respective holders.

Revision History:

Revision	Date	Comments
F.0	2006-09-01	First Printing

Preliminary

Table of Contents

P - Preface

Purpose KBN-P-1
- Support Services KBN-P-1
- Technical Support KBN-P-1
- Sales Support KBN-P-1
- Conventions KBN-P-2

1 - General Description and Architecture

The Karbon-CL family KBN-1-1
- Virtual vs. Hardware KBN-1-1
- The Virtual Frame Grabber (VFG) KBN-1-1
- Karbon Configuration Spaces KBN-1-2

Firmware, Camera Files and Downloads KBN-1-5
General Description KBN-1-6
- The Full CL Virtual Frame Grabber (Full CL VGF) KBN-1-7
- The Base CL Virtual Frame Grabber (Base CL VGF) KBN-1-8

The Karbon-CL Models KBN-1-11

2 - Acquisition and Camera Control

Introduction KBN-2-1
BitFlow's Flow-Thru Architecture KBN-2-2
Camera Specific Firmware KBN-2-5
Generation of Acquisition Windows KBN-2-7
- The Horizontal Active Window, HAW KBN-2-7
- The Vertical Active Window, VAW KBN-2-8

The Control Tables (CTABs) KBN-2-10
- Vertical Control Table KBN-2-10
- The VCTAB Functions KBN-2-11
- Horizontal Control Table KBN-2-13
- The HCTAB Functions KBN-2-15

Synchronizing Acquisition, Camera, CTABs and External Signals KBN-2-18
- Vertical Operations and Events KBN-2-18
- Horizontal Operations and Events KBN-2-22

Acquisition Command and Status KBN-2-26
- The Acquisition Bitfields KBN-2-26

Trigger Processing KBN-2-31
Encoder Processing KBN-2-32
The On-Board Signal Generator KBN-2-33

Preliminary

3 - System Status

Introduction KBN-3-1
FACTIVE, FCOUNT KBN-3-2
PCOUNT, LCOUNT, FENCOUNT KBN-3-3
RD_TRIG_DIFF/TTL/OPTO, RD_ENC_DIFF/TTL/OPTO KBN-3-4
VCOUNT, HCOUNT, LINES_TOGO KBN-3-5
FIFO_EQ KBN-3-6
DEST_ADD KBN-3-7

4 - Camera Control Registers

Introduction KBN-4-1
Bitfield definitions KBN-4-2
Example Bitfield Definition KBN-4-2
Bitfield Definition Explanation. KBN-4-2
CON0 Register KBN-4-4
CON1 Register KBN-4-8
CON2 Register KBN-4-15
CON3 Register KBN-4-21
CON4 Register KBN-4-24
CON5 Register KBN-4-31
CON6 Register KBN-4-37
CON7 Register KBN-4-39
CON8 Register KBN-4-41
CON9 Register KBN-4-46
CON10 Register KBN-4-50
CON11 Register KBN-4-54
CON12 Register KBN-4-56
CON13 Register KBN-4-58
CON14 Register KBN-4-60
CON17 Register KBN-4-64
CON18 Register KBN-4-66
CON19 Register KBN-4-68
CON20 Register KBN-4-70
CON21 Register (Bayer Version) KBN-4-73
CON23 Register KBN-4-76
CON24 Register KBN-4-78
CON25 Register KBN-4-82
CON26 Register KBN-4-84
CON27 Register KBN-4-85
CON36 Register KBN-4-86
CON37 Register KBN-4-88
CON38 Register KBN-4-90

5 - Karbon/Neon DMA Registers

Introduction KBN-5-1

Preliminary

CON28 Register KBN-5-2
CON29 Register KBN-5-4
CON30 Register KBN-5-6
CON31 Register KBN-5-8
CON32 Register KBN-5-10
CON33 Register KBN-5-13

6 - Register and Memory Mapping

Introduction KBN-6-1
Memory Types KBN-6-2
    Registers KBN-6-2
    UART KBN-6-2
    DPM KBN-6-2
    TAG KBN-6-2
    CTABs KBN-6-2
Memory Map KBN-6-3
Downloading Firmware KBN-6-5
PCI Configuration Space and Model/Revision Information KBN-6-6
Scatter Gather DMA Instructions KBN-6-7
Destination Address KBN-6-8
Size of Transfer KBN-6-9
Next Quad Address KBN-6-10

7 - Electrical Interfacing

Introduction KBN-7-1
Trigger KBN-7-2
    Trigger Input Types KBN-7-2
    The Optocoupled Trigger KBN-7-2
Encoder KBN-7-4
    Encoder Input Types KBN-7-4
    The Optocoupled Encoder KBN-7-4
General Purpose Inputs (GPIN) KBN-7-6
General Purpose Outputs (GPOUT) KBN-7-7
    The Open Collector Drivers for GPOUT5 and GPOUT6 KBN-7-7
Camera Link Controls (CCs) KBN-7-11

8 - Specifications

Introduction KBN-8-1
Maximum Pixels Per Line KBN-8-2
Maximum Lines Per Frame KBN-8-3
Power Consumption KBN-8-4

Preliminary

9 - Mechanical

Introduction KBN-9-1
The Karbon-CL Connectors KBN-9-3
The CL Connectors KBN-9-3
The I/O Connector KBN-9-4
The Auxiliary Connectors KBN-9-4
The Jumpers and Switches KBN-9-5
Jumpers KBN-9-5
Switches KBN-9-7
The Camera Link Connector Pinouts (CL1 to CL4) KBN-9-8
The I/O Connector Pinout (P11) KBN-9-9

Preliminary

Preface

Chapter P

P.1 Purpose

This Hardware Reference Manual is intended for anyone using the Karbon-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 Karbon 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. website (www.bitflow.com) for a list of North American representatives and worldwide distributors.

Preliminary

P.1.4 Conventions

Table P-1 shos 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

Preliminary

General Description and Architecture

Chapter 1

1.1 The Karbon-CL family

The purpose of this chapter is to explain, at a block diagram level, how the Karbon-CL works, and what different versions are available. There are four main models in the Karbon-CL family:

KBN-PCE-CL2-D, supports two base CL cameras
KBN-PCE-CL2-F, supports one medium, full or 10-tap CL camera
KBN-PCE-CL4-D, supports four base CL camers
KBN-PCE-CL4-F, supports two medium, full or 10-tap CL cameras

The differences between the models are explained in more detail in Section 1.4.

1.1.1 Virtual vs. Hardware

It's important to understand how this manual works. Some chapters of this manual discuss the Karbon-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 Karbon-CL, these devices are frame grabbers.

Note that we are not using the word virtual here 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 interfacing to. While this might sound complicated, the implementation is simple. You plug our Karbon-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.

1.1.2 The Virtual Frame Grabber (VFG)

The Karbon family is the first board from BitFlow that supports the concept of 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 vitual frame grabber, or to modify an exising one, simply requires writing new firmware or updating existing firmware.

Preliminary

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, what is new about the Karbon family, 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 CL 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 Karbon plaform incredible levels of flexiblity and opens the door to unlimited customization.

To understand which models support which modes see Section 1.4. Also to see examples of the different modes, see Section 1.3. However, keep in mind, the Karbon is only limited by your imagination, if you need a mode you don't see, just let us know.

1.1.3 Karbon Configuration Spaces

The four basic Karbon-CL boards support from one to four VFGs. Each VFG has its own configuration space (PCI interface) and will look like a separate device to the operating system. Each VFG will be connected to one or more CL interface chips. Figure 1-1shows the block diagram of the "D" models. The D models support base two or four base CL cameras. Figure 1-1 shows the diagram of the F models, which support one or two medium, full or 10-tap CL cameras. Of cousre, F models support base cameras as well.

Preliminary

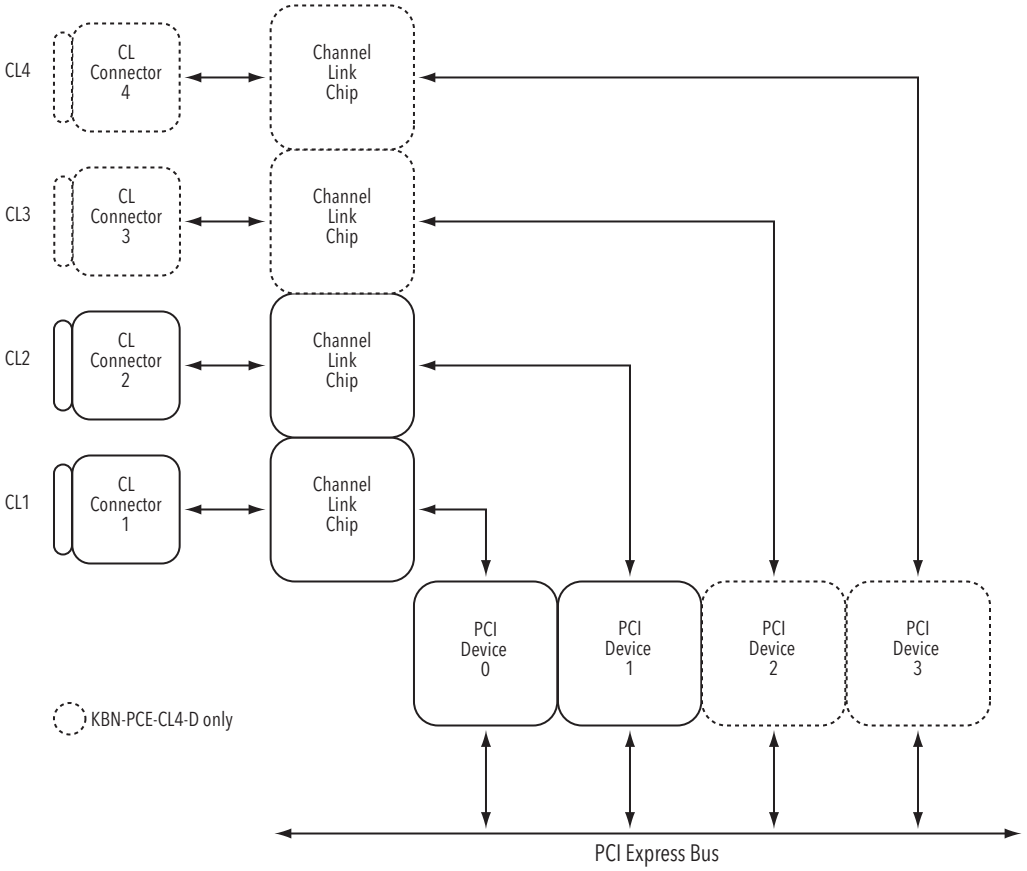


Figure 1-1 KBN-PCE-CL2-D and KBN-PCE-CL4-D

Pre

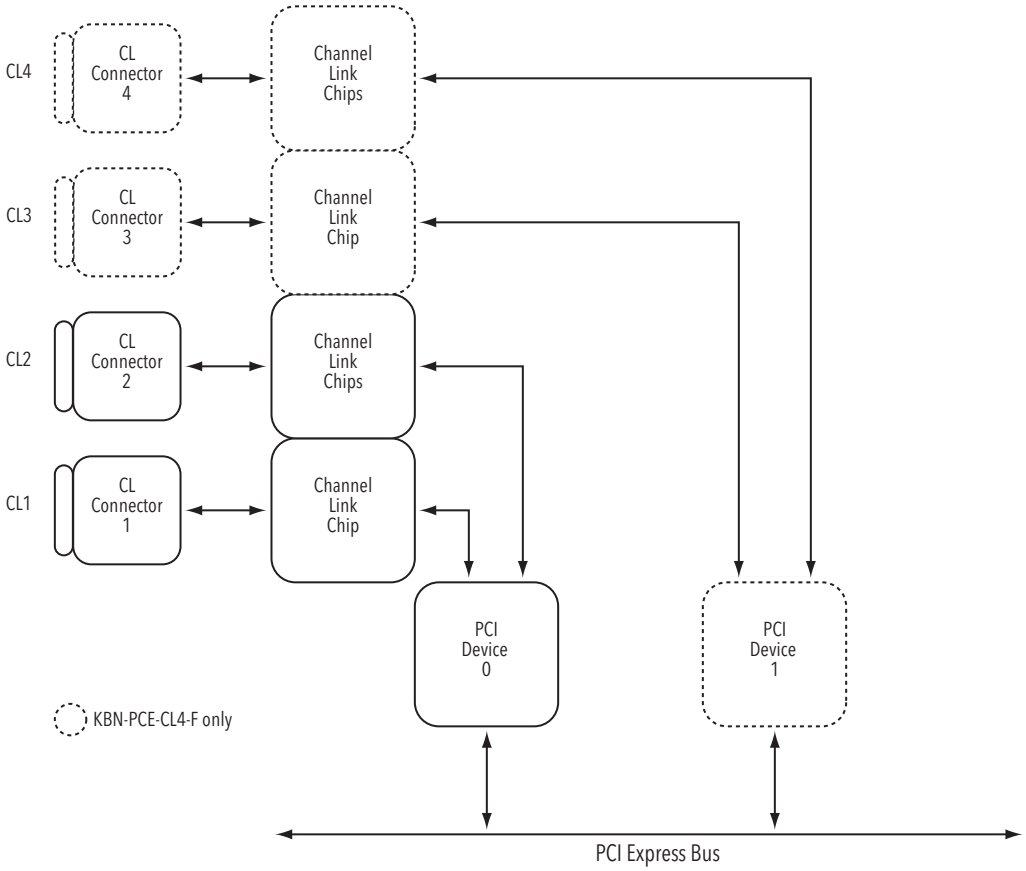


Figure 1-2 KBN-PCE-CL2-F and KBN-PCE-CL4-F

Pre

1.2 Firmware, Camera Files and Downloads

Note that all the devices present on a Karbon will appear in SysReg as separate BitFlow Boards Found. The order that the VFGs appear in SysReg is determined by the operating system and is somewhat arbitrary. However, SysReg list the connector(s) associated with each VFG, so that the connection can be made between VFG and physical connected camera.

Recall that, even though Karbon may appear like four frame grabbers, there is only one actual hardware platform. For this reason the firmware of the of the VFGs on one board is linked. The selection of the master VFG, determines the configurations of all of slave VFGs. For example, if you have a KBN-PCE-CL2-D and you configure the master VFG with a two-tap odd/even pixel camera, then the slave VFG will also have to be configure for a two-tap odd/even pixel camera. In all other ways, however, the two configurations do not have to match. If you have a requirement where this rule must be broken, please contact BitFlow's support department. Custom combinations of firmware are available.

If there is a mismatch between the firmware required by one VFG's camera file and the firmware required by another VFG's camera file, the master VFG's firmware will get priority. In practice this means that if you change the camera file for the master VFG, and it requires different firmware than is already on the board, new firmware will be downloaded next time you start an application. However, in the case of a slave VFG, if different firmware is required than is on the board, an error message will pop up indicating the problem. Thus all the camera files for a all the VFGs on one Karbon should all require the same firmwarre. That said, if you have a custom need for a particular arrangement of cameras, please let us know. We can create custom firmware to solve almost any problem.

Preliminary

1.3 General Description

The Karbon-CL is a x8 PCI Express board. It can work in any PCI Express slot that it can fit it. Usually this means an x8 or x16 slot. However, some mother boards have x4 slots with x8 connectors. The Karbon-CL will work in these slots, although performance my be somewhat reduced.

Figure 1-3 shows the block diagram of the Karbon-CL hardware platform. Recall that this platform is available in a number of configurations. The block diagrams of the current supported configurations is shown in Figure 1-3. Subequent sections discussed the details of the various models.

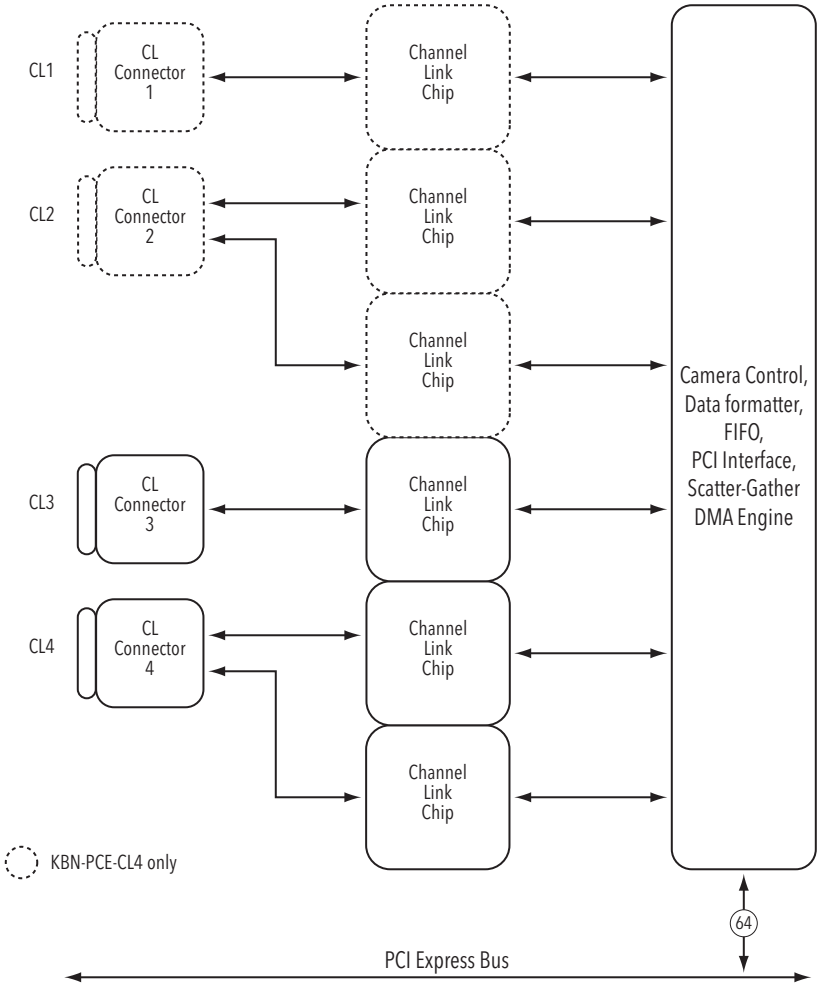


Figure 1-3 Karbon-CL Hardware Platform

As discussed previously, the Karbon platform can support a number of different virtual frame grabber arrangements. Thus the hardware resources seen in Figure 1-3 can be utilized in a number of different ways.

The Karbon-CL is offered in two major configuration: the KBN-PCE-CL4, which support up to four base CL cameras or two full CL cameras, and the KBN-PCE-CL2, which supports up to two base CL cameras or one full CL camera. The final D or F in the model number determines whether the board supports base CL cameras only (i.e. the D models) or medium and full models (i.e. the F models). The following sections discuss these different virtual frame grabbers in more detail.

1.3.1 The Full CL Virtual Frame Grabber (Full CL VGF)

Figure 1-4 illustrates the Full CL virtual frame grabber (VFG) that is support on the Karbon-CL hardware. Since the Full CL interface requires two CL connectors, the KBN-PCE-CL2-F can only support one of these virtual frame grabbers, and the KBN-PCE-CL4-F can support two of these virtual frame grabbers.

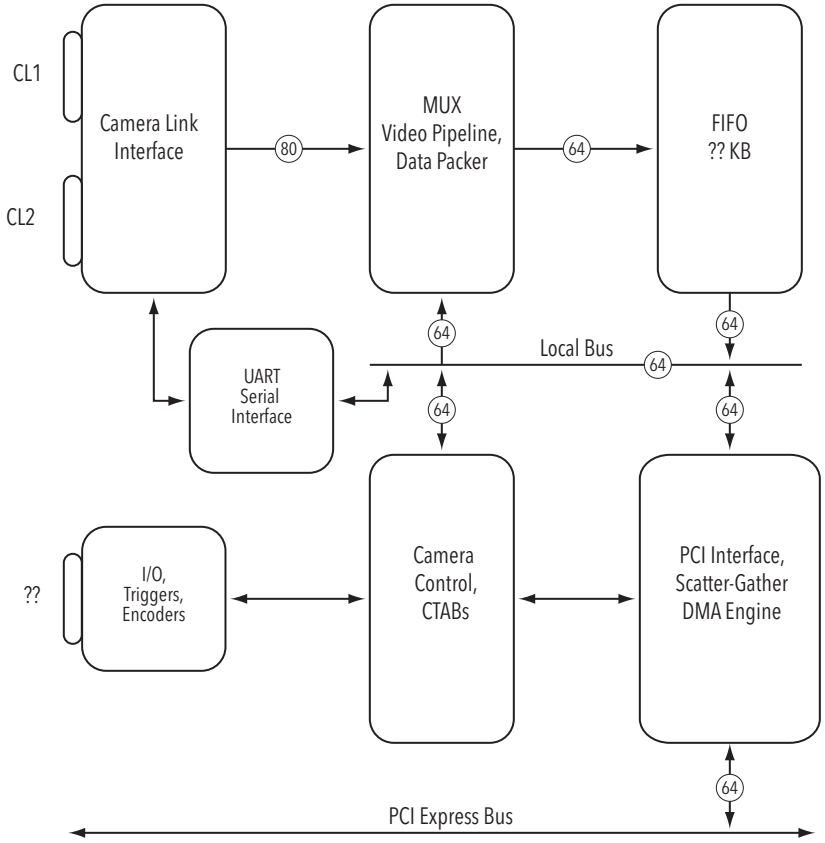


Figure 1-4 Full CL Virtual Frame Grabber

The Full CL VFG implements the Camera Link full configuration, i.e. it can accept a single camera putting out up to 64 bits of data. In addition, it also supports the 10 tap extension to the full CL spefication. This extension allows for the acquisition of up to 80 bits on two CL connectors. Not the the full VFG also supports base and medium CL cameras.

The Full CL VGF can accept input data at up to 85 Mhz.

The following paragraphs are a short description of each block.

The Camera Link Interface block implements the CL full configuration This block has the Channel Link ICs, the Camera Control drivers and the serial communication tranceivers.

The MUX block packs and assembles the data from the Camera Link block before it is pushed into the FIFO. This block re-arranges on-the-fly the data from the camera's taps so that the data is written in raster scan format in the host memory.

The FIFO block, ?? Kilobytes deep, decouples the camera from the DMA engine. It is implemented with dual ported memories.

The Camera Control block handles both camera synchronization as well as external I/O. The block contains the CTABs which are uses to synchronize acquisition with the camera, determine which pixels/lines get acquired and which do not, generate control signals to the camera and to extenal devices. This block also handls start/stoping acquisition based on triggers and encoders.

The PCI interface block handles host reads/writes to/from the board. These reads/writes are used to program the board, and to control its modes. This block is also responsible for DMAing image data to the host memory (or other devices). The DMA engine uses chaining scatter-gather DMA, which can DMA a virtually unlimited amount of data to memory without using any CPU cycles.

There is an on-board UART, as required by the CL specification. There is only one UART on the Karbon platform. It can be switch between the various CL connectors for programming each camera in turn.

The IO connector block has transmitters/receivers to communicate with external industrial equipment (triggers, encoders, light strobes etc.).

CL1 and CL2 are the two Camera Link connectors. CL1 is the main Camera Link connector. It carries data bits 0-23, Camera Controls and serial data port. CL2 is the auxiliary connector. It carries data bits 24-63, for the full mode and bits 24-79 in ten-tap mode.

Preliminary

1.3.2 The Base CL Virtual Frame Grabber (Base CL VGF)

Figure 1-5 illustrates the Full CL virtual frame grabber (VFG) that is support on the Karbon-CL hardware. Since the Base CL interface requires one CL connectors, the KBN-PCE-CL2-D can support two of these virtual frame grabbers, and the KBN-PCE-CL4-D can support four of these virtual frame grabbers.

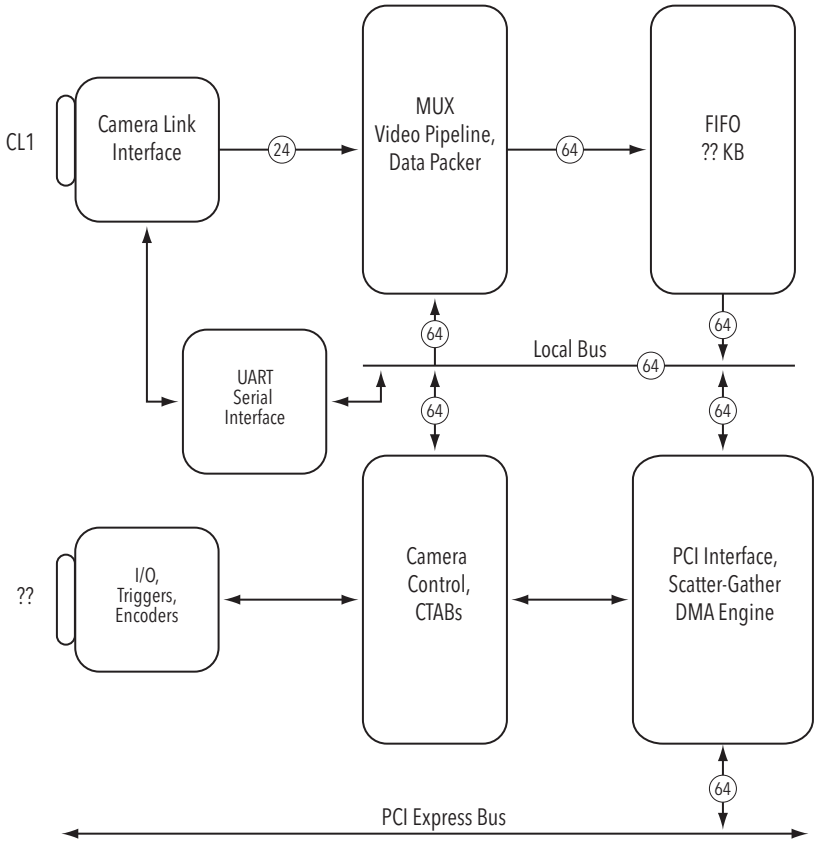


Figure 1-5 Base CL Virtual Frame Grabber

The Base CL VFG implements the Camera Link base configuration, i.e. it can accept a single camera putting out up to 24 bits of data.

The Base CL VGF can accept input data at up to 85 Mhz.

The following paragraphs are a short description of each block.

The Camera Link Interface block implements the CL base configuration This block has the Channel Link IC, the Camera Control drivers and the serial communication tranceivers.

The MUX block packs and assembles the data from the Camera Link block before it is pushed into the FIFO. This block re-arranges on-the-fly the data from the camera's taps so that the data is written in raster scan format in the host memory.

The FIFO block, ?? Kilobytes deep, decouples the camera from the DMA engine. It is implemented with dual ported memories.

The Camera Control block handles both camera synchronization as well as external I/O. The block contains the CTABs which are uses to synchronize acquisition with the camera, determine which pixels/lines get acquired and which do not, generate control signals to the camera and to extenal devices. This block also handls start/stoping acquisition based on triggers and encoders.

The PCI interface block handles host reads/writes to/from the board. These reads/writes are used to program the board, and to control its modes. This block is also responsible for DMAing image data to the host memory (or other devices). The DMA engine uses chaining scatter-gather DMA, which can DMA a virtually unlimited amount of data to memory without using any CPU cycles.

There is an on-board UART, as required by the CL specification. There is only one UART on the Karbon platform. It can be switch between the various CL connectors for programming each camera in turn.

The IO connector block has transmitters/receivers to communicate with external industrial equipment (triggers, encoders, light strobes etc.).

Preliminary

1.4 The Karbon-CL Models

There are two models in the Karbon-CL family, the KBN-PCE-CL2 and the KBN-PCE-CL4. Table 1-1 illustrates the capablitlites of each model.

Table 1-1 Karbon-CL Models

Capability	KBN-PCE-CL2-D	KBN-PCE-CL2-F	KBN-PCE-CL4-D	KBN-PCE-CL4-F
Number of base CL cameras supported	2	1	4	2
Number of medium CL cameras supported	0	1	0	2
Number of full CL cameras supported	0	1	0	2
Number of 10 tap full CL cameras supported	0	1	0	2
Number of Virtual Frame Grabbers supported	2	1	4	2
Maximum number of input bits	80	80	160	160
Number of independent trigger inputs	2	2??	4??	4??
Number of independent encoder inputs	2	4??	??	??
Number of PCI configurations (devices)	2	1	4	1

Note that the R64 and R64e frame grabbers from BitFlow support acquisition from two cameras in what is known as dual base mode. Dual base mode requires that the two cameras be identical, and that they be synchronized. The Karbon-CL platform, supports multiple cameras in a completely independent fashion. This means there there are no restrictions on the type or resolutions of the cameras that you connect to the same Karbon. There are also no restrictions on the synchronization between the cameras. Not only do they no need to be syncrhonized, they don't even need to be acquiring at the same time. As far as your application is concerned each camera connected to the Karbon-CL has it's own independent frame grabber.

Preliminary

Preliminary

Acquisition and Camera Control

Chapter 2

2.1 Introduction

This section covers acquisition and camera control for the R64-CL, Karbon-CL and the Neon-CL.

Preliminary

2.2 BitFlow's Flow-Thru Architecture

The MUX block in Figure 2-1 is composed of a chain of sub-blocks that make up the Flow-Thru Architecture (FTA). Figure 2-1 shows the structure of the FTA. All the data paths are 64-bit. The implementation of the individual blocks depends on the camera format, i.e. it is specific to the firmware downloaded for each sensor architecture. There is a bitfield, FORMAT, which indicates the currently downloaded firmware.

Below is a description of the individual blocks. For each block are shown the signals that are defined by the user.

Data from the Camera Link is synchronized and assigned to data lanes according to the camera format. The user has no control over these operations. From this block the data goes to a Barrel Shifter.

The Barrel Shifter is composed of four 16-bit barrel shifters. All shifters receive the same command, Left/Right and the amount of shift, up to 15 bits. The main purpose of the Barrel Shifter is for cameras that have more than 8 bits per pixel. The Barrel Shifter can down-shift the data to 8-bit suitable for display. Any camera with up to four taps can be accommodated.

There is a Video Delay Line (not shown) in the data path which can delay the video by up to 8 clocks. This is useful for accurate alignment of the video on the display.

The Video Selector selects the data source: the video from the camera or the on-board generated synthetic video. The various patterns of synthetic video are useful mainly for the on-board Built In Self Test (BIST).

The Mask is a 32-bit mask replicated over the upper and lower 32 bits of the 64-bit data path. The purpose of this mask is to be able to set to zero any bit in the data path (a one will pass the data as is, a zero will set that bit to zero).

The Clip is a clipping mechanism replicated on each one of the eight 8-bit data lanes. If enabled, it will clip the 256 gray levels in each lane according to the formula:

If video > 245 then video = 245
If video <10 then video = 10

This mechanism is useful for displaying gray level data on a VGA that is set in 256-color mode. In this mode the upper and lower 10 gray levels are dedicated to the Windows graphics.

The Assembler will assemble and pack the video data before it is written in the FIFO. This block does the raster scan re-arranging of the data. The packing is dependent on the pixel depth, which is defined in the PIX_DEPTH bitfield in CON10. The DISPLAY bit will force this block to assemble the data as 8-bit pixels, suitable for display. When using this mode, the barrel shifters must be set to down-shift each pixel by the correct amount. A 10-bit camera, for example, would need a 2-bit right shift.

The SWAP bit will swap between odd-even data streams for cameras that supply odd-even pixels.

Preliminary

The amount of data written in the FIFOs is controlled by the Acquisition Window. The vertical and horizontal size of this window is programmed in the ALPF and the ACLP registers respectively (see Section 2.4). The timing of this window is determined by the camera and the acquisition state machine.

Preliminary

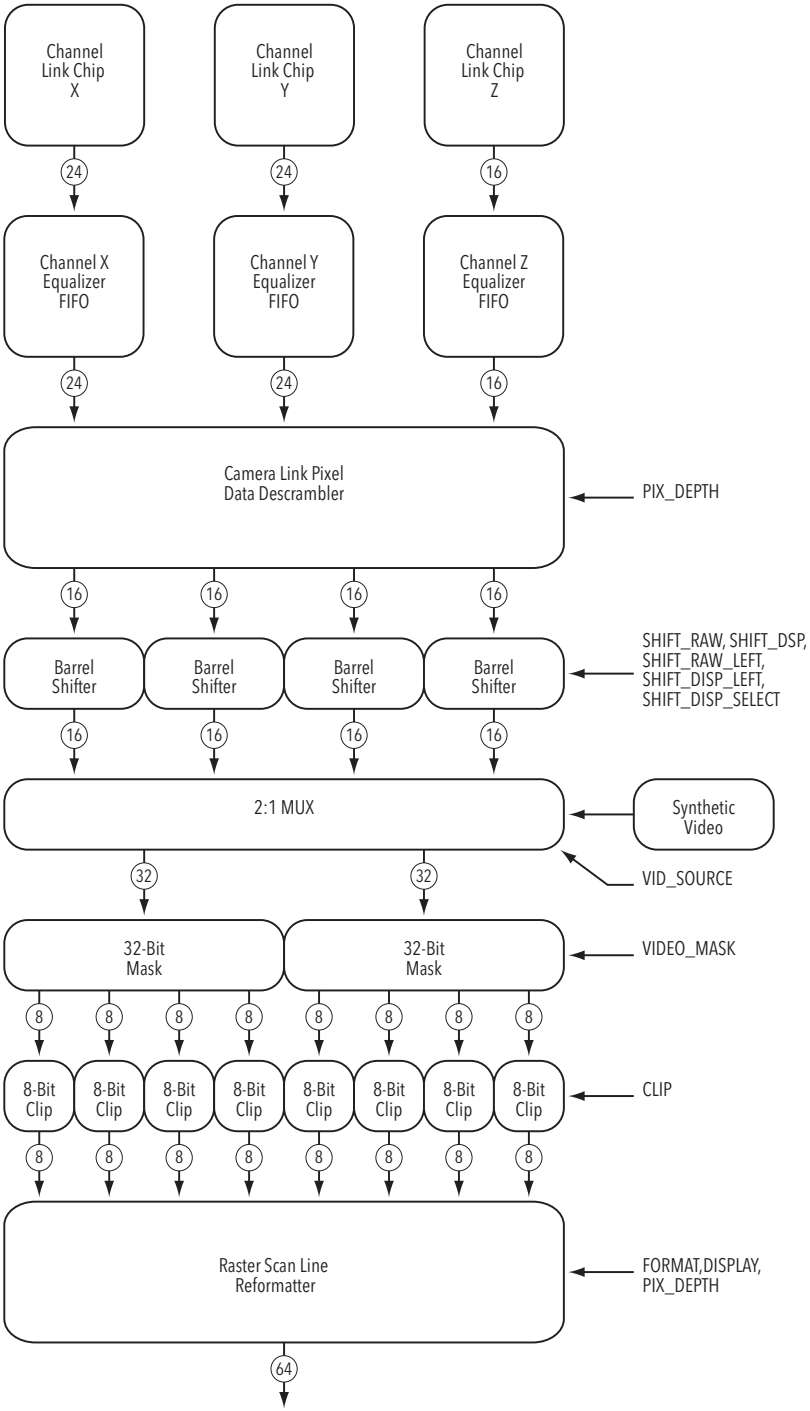


Figure 2-1 Flow Thru Architecture

2.3 Camera Specific Firmware

The Flow-Thru architecture is flexible and can be adapted to different cameras architectures. The main intelligence is in the firmware that gets downloaded into the on-board Field Programmable Gate Arrays (FPGAs). This firmware is different for every camera architecture. The firmware is called out in the camera file. On initialization, the driver will download into the FPGAs the firmware called-out in the camera file.

The type of the firmware is hard-coded in the FORMAT field in register CON10. The list of the formats is shown in Table 2-1

Table 2-1 Firmware Options

FORMAT	Firmware Name	Format Description
0	MUX	1 tap cameras
1	MUX_2TOEP	2 taps, odd-even pixels
2	MUX_2TOEL	2 taps, odd-even lines
3	MUX_2TS	2 taps, segmented
4	MUX_2TS1RI	2 taps, segmented, right inverted
5	MUX_4TS	4 taps, segmented
6	MUX_4T2S2RIOEP	4 taps, odd-even pixels, right taps inverted
7	MUX_4TQ2RI2BU	4 quads, right quads inverted, bottom quads upside down
8	MUX_2CAM	2 cameras: 1 tap each
9	MUX_2CAM_2TOEP	2 cameras: 2 taps, odd-even pixels
10	MUX_2CAM_2TS1RI	2 cameras: 2 taps, segmented, right-inverted
11	MUX_2CAM_2TS	2 cameras: 2 taps, segmented
12	MUX_2CAM_2TOEL	2 cameras: 2 taps, odd-even lines
13	MUX_8TS	8 taps, segmented
14	MUX_BAY	Bayer decoder, 1 tap 8 bit
15	MUX_BAY_OE	Bayer decoder, 2 taps, odd-even pixels
16	MUX_BAY_2TS	Bayer decoder, 2 taps, segmented
17	MUX_4WI	4 taps, 4-way interleaved

Preliminary

Table 2-1 Firmware Options

FORMAT	Firmware Name	Format Description
18	MUX_2TOEPI	2 taps, odd-even pixels, both inverted
19	MUX_1TI	1 tap, inverted
20	MUX_8WI	8 taps, 8-way interleaved
21	MUX_BAY_2TS_RI	Bayer decoder, 2 taps, segmented, right inverted

Preliminary

2.4 Generation of Acquisition Windows

2.4.1 The Horizontal Active Window, HAW

The Horizontal Active Window (HAW) is a square wave that defines the portion of the line that will be acquired horizontally. HAW can span less than the whole camera line, if we want to acquire only a portion of that line. The size of the HAW is determined by a single number, the Active Clocks Per Line (ACPL). The ACPL is defined as the number of clocks during which the HAW is active. The ACPL field is programmed in CON10. The 17 bits define the maximum HAW as minimum 128K pixels.

The total number of pixels per line that will be acquired can be different than the ACPL. For a dual tap camera that supplies odd-even pixels for example, the total number of pixels acquired will be twice the ACPL, as for every clock the camera supplies two pixels. Note that the ACPL is not a function of the bits per pixel. The relationship between the number of pixels per line and the number of clocks per line is controlled by the firmware currently downloaded to the board. The FORMAT register will indicate which firmware is currently downloaded. Each tap configuration requires a different firmware file be downloaded. The correct firmware is automatically downloaded based on the information contained in the camera configuration file.

The size of the HAW is on an arbitrary boundary. The start in time of the HAW can come from two sources, depending on the setting of the HAW_START bit:

The HSTART bit in the HCTAB, if HAW_START = 1.
The start of LEN, if HAW_START = 0.

In both modes, the start of the HAW can be delayed 0-7 clocks relative to the start function (HSTART or LEN). This is done by the TRIM bits in CON9.

In both modes, the start of the HAW can be advanced 0-7 clocks relative to the start function (HSTART or LEN). This is done by the DELAY bits in CON14.

The HCTAB is the Horizontal Control Table that generates the HSTART, see section on CTABs.

Figure 2-2 shows the controls that generate the HAW.

Preliminary

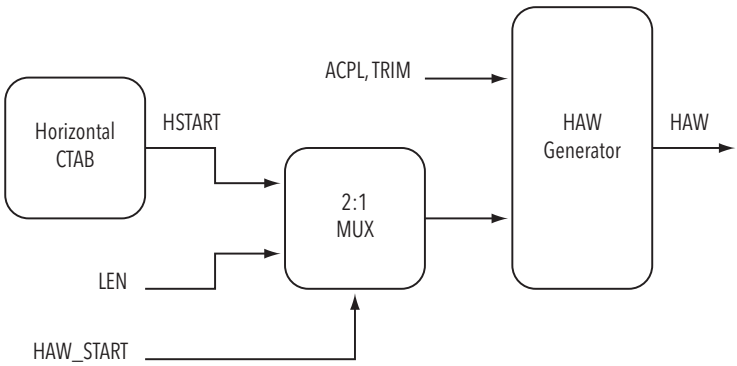


Figure 2-2 Generation of the Horizontal Active Window, HAW

2.4.2 The Vertical Active Window, VAW

The Vertical Active Window (VAW) is a square wave that defines the portion of the frame that will be acquired vertically. VAW can span less than the whole camera frame, if we want to acquire only a portion of that frame. The size of the VAW is determined by a single number, the Active Lines Per Frame (ALPF). The ALPF is defined as the number of HAW periods during which the VAW is active. The ALPF field is programmed in CON17. The 17 bits define the maximum VAW as minimum 128K lines.

The total number of lines per frame that will be acquired can be different than the ALPF. For a dual tap camera that supplies odd-even lines for example, the total number of lines acquired will be twice the ALPF as in the period of one HAW the camera supplies two lines.

The size of the VAW is on an arbitrary boundary. The start in time of the VAW can come from two sources, depending on the setting of the VAW_START bit:

The VSTART bit in the VCTAB, if VAW_START = 1.
The start of FEN, if VAW_START = 0.

Data will be acquired in the window defined by the HAW and the VAW

Figure 2-3 shows the controls that generate the VAW.

Preliminary

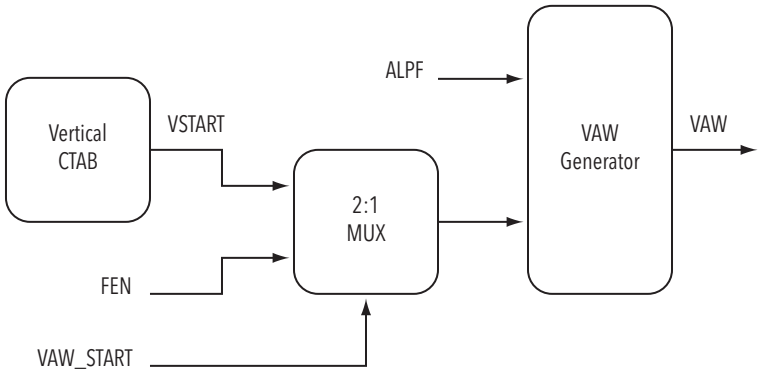


Figure 2-3 Generation of the Vertical Active Window, VAW

Preliminary

2.5 The Control Tables (CTABs)

The CTABs are two memories that are programmed by the host computer and read by the board's acquisition circuitry. The read-out is done in a sequential fashion, i.e. the memories' addresses are scanned sequentially. There is a vertical' memory (VCTAB) and a horizontal' memory (HCTAB). The vertical and horizontal memory each has an associated counter which scans its addresses, VCOUNT and HCOUNT respectively. The concept here is similar to how a CPU runs a program. There is a PC which works it's way through memory, processing each instruction in turn. Each bit in the CTAB corresponds to a different operation. For example, one bit might control the level of a signal going to the camera. In this case the CTABs can be thought of as programmable waveform generators. Another bit might cause an interrupt to occur on the PCI bus, yet another bit might force the HCOUNT to go to zero. The CTABs are fully programmable by software. The details of the CTABs are describe in this section.

2.5.1 Vertical Control Table

Figure 2-4 depicts the structure of the Vertical Control Table (VCTAB). For clarity, the address and data path that allow the host to program the VCTAB are not shown.

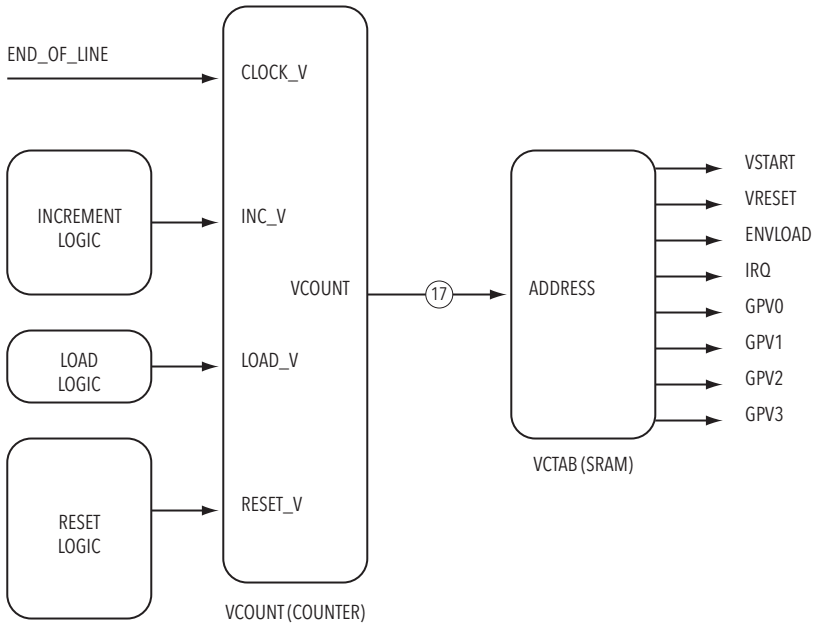


Figure 2-4 Vertical Control Table

The Vertical Control Table is made up of the following blocks:

VCOUNT - a synchronous counter that can be incremented, loaded and reset. The clock that drives VCOUNT is derived from the HCTAB, see below. VCOUNT is 17-bit wide and is connected to the address input of the VCTAB.

Logic for generating INC_V - the increment control signal to the VCOUNT.

Logic for generating LOAD_V - the load control signal to the VCOUNT. When LOAD_V is asserted, VCOUNT is loaded with the value of 8000h (32,768 in decimal).

Logic for generating RESET_V - the reset control signal to the VCOUNT. When RESET_V is asserted, VCOUNT is reset to 0.

Logic for generating CLOCK_V - the clock to the VCOUNT.

VCTAB - a static memory (SRAM) that outputs eight VCTAB control signals. The address of this SRAM is driven by VCOUNT.

If RESET_V and LOAD_V are asserted simultaneous, RESET_V overrides.

As the VCOUNT increments, it scans the addresses of the VCTAB in ascending order. The output of the VCTAB depends on the data that has been written in the VCTAB by the host. If the VCOUNT is free running, it will cyclically scan all the VCTAB's addresses. Any arbitrary cyclic waveform can be implemented by programming the VCTAB with the adequate data.

The LOAD_V and RESET_V will enable the synchronization between external events and the waveforms generated by the VCTAB. LOAD_V and RESET_V will force the VCOUNT to known values, 8000h and 0 respectively.

The INC_V signal will allow for stopping the counter from incrementing. In that case, the output of the VCTAB will be constant. While the VCOUNT is not incrementing, it can still be loaded or reset, see the logic below.

2.5.2 The VCTAB Functions

The functions assigned to the columns in the VCTAB are shown in Table 2-2.

Table 2-2 The VCTAB Functions

Bit	Name	Function
0	VSTART	Start of VAW
1	VRESET	Vertical Reset
2	ENVLOAD	FEN Mask, enable load
3	IRQ	CTAB Interrupt
4	GPV0	General purpose vertical function 0
5	GPV1	General purpose vertical function 1
6	GPV2	General purpose vertical function 2
7	GPV3	General purpose vertical function 3

VSTART defines the start of the Vertical Acquisition Window (VAW), if the start is provided by the VCTAB, see previous section.

Preliminary

VRESET defines the reset of the VCOUNT, in case this function is programmed in the VCTAB.

ENVLOAD enables the FEN to load the VCOUNT. The rationale behind the ENVLOAD column from the VCTAB is that some cameras might not give a FEN, but only two pulses: the start and end of FEN. With the ENVLOAD, we can mask out the unwanted one.

IRQ provides an interrupt to the host, allowing an interrupt to occur at any point on the vertical axis.

GPV are general purpose vertical functions, see usage below.

Note: If a VRESET pulse happens coincident with a LOAD, the LOAD is overriding.

The CLOCK_V Control

CLOCK_V is the clocking of the VCOUNT is generated by the end of the line (horizontal reset).

The INC_V Control

INC_V is the logic for incrementing VCOUNT.

There are only two instances when we want to inhibit the incrementing of VCTAB. The first instance is when VCOUNT reaches 0000h, the “Stop at Zero” case. The other instance is when VCOUNT reaches 7FF0h, the “Vertical Stick” case.

Stop at Zero

Usually, VCOUNT will reach zero because of a RESET_V signal. After VCOUNT is reset, there are programmable options defined by VCNT_RLS_ZERO. Depending on this bitfield, VCOUNT can continue to count or wait at zero till some event occurs, usually the assertion of the TRIGGER.

This operating mode is especially useful for synchronizing cameras to external events. TRIGGER is usually the output of a part-in-place signal. Until this signal is asserted, the VCOUNT waits at address 0000h. After the TRIGGER is asserted, VCOUNT starts counting, i.e., scanning the VCTAB in ascending order. At some address we will program a sync signal to be sent to the camera, usually through GPV0. In response to this sync signal, the camera will give back a frame, and it will assert FEN. The FEN will load address 8000h into VCOUNT. In the VCTAB, we will program the vertical acquisition window to start after address 8000h. At the end of the vertical acquisition window the RESET_V will be asserted, which in turn will reset the VCOUNT. VCOUNT will wait at address 0000h until TRIGGER is asserted.

Vertical Stick

Using the previous example, assume that after we asserted the sync signal to the camera, we expect the camera to give us a frame, i.e., assert FEN. While we expect the camera to assert FEN, VCOUNT is still being incremented. If it takes too long for the camera to respond, VCOUNT will eventually reach and pass beyond 8000h. A vertical

Preliminary

acquisition window will be asserted, even though the camera did not assert FEN. To avoid such a situation, just before address 8000h, when VCOUNT reaches 7FF0h, it will stop. It will stay at 7FF0h until FEN is asserted. Then, VCOUNT will be loaded with 8000h.

The Vertical Stick will occur according to the setting of VCNT_RLS_STK.

The LOAD_V Control

LOAD_V is the logic of loading VCOUNT. In other words, loading VCOUNT means it jump to a new address.

VCOUNT will be loaded with the value 8000h by the rising/falling edge of FEN, if ENVLOAD is asserted. FEN usually marks the start of a valid frame. The start of the vertical acquisition window can be placed starting at address 8000h.

ENVLOAD is a column in the VCTAB. There are cameras that do not assert FEN. Some other type of cameras assert only the start and stop of a frame. In this case, ENVLOAD can mask out the unwanted signals.

Operation on the rising/falling edge of FEN is selected by FENPOL, see CON14

The RESET_V Control

RESET_V is the logic of resetting VCOUNT.

VCOUNT can be reset from different sources, under different conditions. The resetting of the VCOUNT is controlled by the VCNT_RST bitfield.

2.5.3 Horizontal Control Table

The Horizontal Control Table (HCTAB) is 8 bits wide. The function of each bit is show in the following table.

Figure 2-5 depicts the structure of the HCTAB. For clarity, the address and data path that allow the host to program the HCTAB are not shown

Preliminary

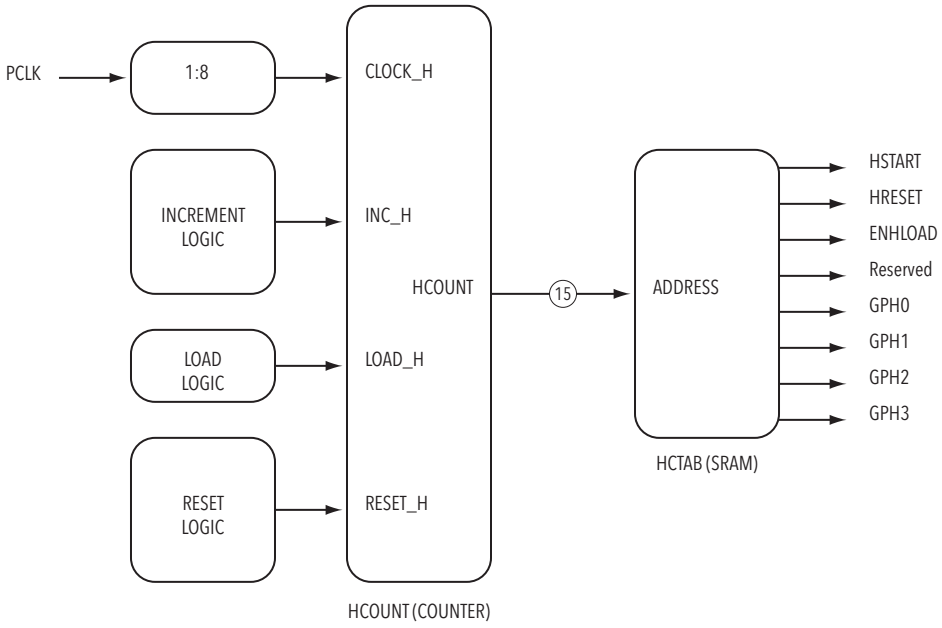


Figure 2-5 Horizontal Control

The Horizontal Control Table is made up of the following blocks:

HCOUNT - a synchronous counter that can be incremented, loaded and reset. The clock that drives HCOUNT is a free running clock, PCLK/8, i.e., the pixel clock divided by eight. HCOUNT is 15 bits wide and is connected to the address input of the HCTAB.

The 15 bit HCOUNT with the PCLK/8 can generate functions up to 256K PCLKs long, on boundaries of 8 PCLKs.

The ACPL is 17 bit, and that can generate an HAW of up to 128K PCLKs.

Logic for generating INC_H - the increment control signal to the HCOUNT.

Logic for generating LOAD_H - the load control signal to the HCOUNT. When LOAD_H is asserted, HCOUNT is loaded with the value of 2000h.

Logic for generating RESET_H - the reset control signal to the HCOUNT. When RESET_H is asserted, HCOUNT is reset to 0.

HCTAB - a static memory (SRAM) that outputs eight HCTAB control signals. The address of this SRAM is driven by HCOUNT.

Logic for generating CLOCK_H - the clock to the HCOUNT. This is a frequency divider. CLOCK_H is PCLK, the pixel clock divided by eight.

Note: If RESET_H and LOAD_H are asserted simultaneously, RESET_H overrides.

As the HCOUNT increments, it scans the address of the HCTAB in ascending order. The output of the HCTAB depends on the data that has been written in the HCTAB by the host. If the HCOUNT is free running, it will cyclically scan all the HCTAB's addresses. Any arbitrary cyclic waveform can be implemented by programming the HCTAB with the adequate data.

The LOAD_H and RESET_H will enable the synchronization between external events and the waveforms generated by the HCTAB. LOAD_H and RESET_H will force the HCOUNT to fixed values, 2000h and 0 respectively.

The INC_H signal will allow for stopping the counter from incrementing. In that case, the output of the HCTAB will be constant. While the HCOUNT is not incrementing, it can still be loaded or reset, see the logic below.

2.5.4 The HCTAB Functions

The functions assigned to the columns in the HCTAB are shown in Table 2-3.

Table 2-3 The HCTAB Functions

HCTAB	Name	Function
D0	HSTART	Start of HAW
D1	HRESET	Reset HCOUNT and increment the VCOUNT
D2	ENHLOAD	LEN Mask, enable horizontal load
D3	reserved	
D4	GPH0	General purpose horizontal function 0
D5	GPH1	General purpose horizontal function 1
D6	GPH2	General purpose horizontal function 2
D7	GPH3	General purpose horizontal function 3

HSTART marks the start of the Horizontal Acquisition Window, HAW. Video will be acquired only while the HAW is active.

HRESET will reset the HCOUNT.

ENHLOAD will allow the loading of the HCOUNT. A location that has 1 will allow the loading of the HCOUNT. A 0 will inhibit the loading of the HCOUNT.

GPH are general purpose horizontal functions. See usage below.

The INC_H Control

INC_H is the logic for incrementing HCOUNT.

There are only two instances when we want to inhibit the incrementing of HCTAB. The first instance is when HCOUNT reaches 0000h, “Stop at Zero” case. The other instance is when HCOUNT reaches 1FF1h, the “Horizontal Stick” case.

Preliminary

The "Stop at Zero" Case

Usually, HCOUNT will reach zero because of a RESET_H signal. After HCOUNT is reset, there are two programmable options:

HCOUNT keeps on counting.
HCOUNT stays at zero until ENCODER is asserted.

The selection between the two options is done by the HCNT_RLS_ZERO bitfield, see next section on camera synchronization.

This operating mode is especially useful for synchronizing line scan cameras to external events. ENCODER is usually the output of an encoder or a tachometer signal. Until this signal is asserted, the HCOUNT waits at address 000oh. After the ENCODER is asserted, HCOUNT starts counting, i.e., scanning the HCTAB in ascending order. At some address we will program a sync signal to be sent to the scan camera, usually through GPH0. In response to this sync signal, the camera will give back a line, and it will assert LEN. The LEN will load address 2000h into HCOUNT. In the HCTAB, we will program the horizontal acquisition window after address 2000h. At the end of the horizontal acquisition window the RESET_H will be asserted, which in turn will reset the HCOUNT. HCOUNT will wait at address 00o0h until ENCODER is asserted.

Horizontal Stick

Using the previous example, assume that after we asserted the sync signal to the camera, we expect the camera to give us a line, i.e., assert LEN. While we expect the camera to assert LEN, HCOUNT is still being incremented. If it takes too long for the camera to respond, HCOUNT will eventually reach and pass beyond 2000h. A horizontal acquisition window will be asserted even though the camera did not assert LEN. To avoid such a situation, just before address 2000h, when HCOUNT reaches 1FF0h, it will stop. It will stay at 1FF0h until LEN is asserted. Then, HCOUNT will be loaded with 2000h.

The LOAD_H Control

LOAD_H is the logic of loading HCOUNT.

HCOUNT will be loaded with the value 2000h by the rising/falling edge of LEN, if ENHLOAD is asserted. LEN usually marks the start of valid data in a line. The Horizontal Acquisition Window can be placed starting at address 2000h.

ENHLOAD is a column in the HCTAB that enables the LEN. There are cameras that do not assert LEN. In that case, the LEN input must be disabled, otherwise its behavior is unpredictable.

Operation on the rising/falling edge of LEN is selected by LENPOL, see CON14.

The RESET_H Control

RESET_H is the logic of reset HCOUNT.

HCOUNT can be reset from several sources, according to HCNT_RST bitfield, see next section on camera synchronization.

Preliminary

Example

Lets look at a simple example to clarify the concept of the HCTAB. Assume we want to program a free running horizontal window of 32 pixels active area. Just before the active area we want to fire a strobe using GPH0. D0 (HSTART) defines the start of the HAW. Bit D1 defines the reset of the HCOUNT. D4, GPH0, is the strobe pulse. The size of the HAW is programmed in ACLP register.

Taking into account that the address counter is clocked by 1/8 the pixel clock, the HCTAB memory map will be as shown in Table 2-4.

Table 2-4 HCTAB Example

HCTAB Address	HRESET	HSTART	GPH0	Comments
0	0	0	0	You got here from address 9
1	0	0	0	
2	0	0	0	
3	0	0	1	Fire the strobe
4	0	1	0	Start Horizontal Acquisition Window
5	0	0	0	Acquire
6	0	0	0	Acquire
7	0	0	0	Acquire
8	0	0	0	Acquire
9	1	0	0	Assert RESET_H
10	0	0	0	

The CT Functions

The CT's are four functions derived from the HCTAB and the VCTAB. Those functions can define an arbitrary horizontal and/or vertical waveform. The definition of the CT's is given below:

CT[0] = GPV[0] AND GPH[0]
CT[1] = GPV[1] AND GPH[1]
CT[2] = GPV[2] AND GPH[2]
CT[3] = GPV[3] AND GPH[3]

Each CT has a vertical and a horizontal component. Both components are programmed in the CTABs. The minimum horizontal pulse is 8 PCLKs. The minimum vertical pulse is one line.

The CT's can be steered to the Camera Controls (on the CL connectors) and to the GPOUTs, on the IO connector.

2.6 Synchronizing Acquisition, Camera, CTABs and External Signals

These boards have extremely flexible camera interfaces. They have been designed to acquire from almost any Camera Link camera and to synchronize with almost all industrial signals. There are two layers of synchronization. The first layer is to synchronize to signals coming from the industrial environment. For example triggers and encoders. The second layer is to synchronize to the camera. This requires both sending signals to the camera (e.g. exposure control) and receiving signals from the camera (e.g. Pixel Clock, Line Enable and Frame Enable). All of these synchronization problems are solved by the Control Tables (CTABs) and Vertical/Horizontal Operations. See previous section for detailed operation on the CTABs.

The Vertical and Horizontal Operations describe different state changes that the board goes through. For example one operation might be to begin acquiring pixels, another might be to reset the VCOUNT back to zero. Generally these state changes are caused by one or more events. There are a number of events, both horizontal and vertical, that the board can react to. These events are tied to operations by a set of programmable bitfields. The details of these events are outlined in this section.

2.6.1 Vertical Operations and Events

The vertical operations are related to the vertical axis of an image (in memory or on the display) or frame timing (of a camera). The operations are mainly commands to VCOUNT and acquisition commands. Each operation can be initiated by some event. The selection of the event that will initiate the specific operation is done by a set of three control bits related to each operation.

Table 2-5 is a list of the vertical operations and their related control bits.

Table 2-5 Vertical Operations

Vertical operation	Control bits
VCOUNT frozen/released from 0000h	VCNT_RLS_ZERO
VCOUNT reset to 0000h	VCNT_RST
VCOUNT load with 8000h	VCNT_LD
VCOUNT frozen/released from 7FF0h	VCNT_RLS_STK
VCOUNT increment	VCNT_INC
Acquire (SNAP, GRAB, CONTINUOUS)	ACQ_CON
FREEZE acquisition	FREEZE_CON
ABORT acquisition	ABORT_CON
START vertical acquisition window	VAW_START

Preliminary

Table 2-6 is a list of the events that initiate the vertical operations:

Table 2-6 Vertical Events

Event description	Event Name
TRIGGER asserted	TRIG_ASRT
TRIGGER de-asserted	TRIG_DASRT
FEN asserted	FEN_ASRT
FEN de-asserted	FEN_DASRT
TRIGGER is HI	TRIG_HI
TRIGGER is LO	TRIG_LO
RESET from VCTAB	RST_VCTAB
RESET from SW	RST_SW
Host writes acquisition command	HOST_WCMD_GRAB/SNAP
Acquisition frame counter reaches programmed value	AQ_COUNT

What follows is a list of each operation and the corresponding events that can be used to cause the operation. Included is the bitfield settings for each operations. It is important to understand that each operation is independent and can be programmed without regard for how the other events are programmed. However, some combinations might not make sense.

VCOUNT Release From Zero

This operation controls the behavior of VCOUNT when it reached zero. See Table 2-7.

Table 2-7 VCNT_RLS_ZERO

Initiator	VCNT_RLS_ZERO	Comments
None	0	Normal operation mode, no stop at 0000h
TRIG_ASRT	1	Edge Mode (aka Letter Mode), always stay at 0000h, release on TRIG_ASRT
TRIG_HI	2	Level Mode (aka Luggage Mode), stay at 0000h if TRIG_LO, release on TRIG_ASRT

Preliminary

VCOUNT Reset To Zero

This operation controls how VCOUNT is reset to zero. See Table 2-8.

Table 2-8 VCNT_RST

Initiator	VCNT_RST	Comments
End_of_VAW	0	Default operation, reset at end of VAW
TRIG_DASRT or End_ of_VAW	1	Triggered termination
RST_VCTAB	2	Reset from VCTAB
FEN asserted or	3	Reset from start of FEN
TRIG_DASRT or RST_ VCTAB	4	Triggered termination
TRIG_DASRT	5	Triggered termination

Note: The VCOUNT is always reset by the RST_SW (sofware reset) and by the HOST_ WCMD_ABORT (host writes ABORT command).

VCOUNT Release From Stick Point (7FF0h)

This operation controls the behavior of VCOUNT when it hits the stick point. The purpose of the stick point is to allow for very long periods of time between frames. The stick point is located at 7ff0h. See Table 2-9.

Table 2-9 VCNT_RLS_STK

Initiator	VCNT_RLS_STK	Comments
None	0	Normal operation mode, no stop at 7FF0h
VLOAD or VRESET	1	Stick at 7FF0h till load (usually FEN) or reset asserted

Preliminary

VCOUNT Load To 8000h

This operation controls how and when VCOUNT loads (jumps to) 8000h. See Table 2-10.

Table 2-10 VCNT_LD

Initiator	VCNT_LD	Comments
None	0	No load
FEN_ASRT and ENV-LOAD	1	Assertion of FEN qualified with ENV-LOAD
FEN_ASRT	2	Assertion of FEN only
TRIG_ASRT	3	Assertion of TRIGGER

Acquisition Command Control

This operations controls how the acquisition commands get initiated. There are two major acquisition commands. The SNAP command, which only acquires one frame. The GRAB command, which continuously acquires frames until a freeze or abort command is issued. In addition, the board has a continuous data mode which is not frame oriented. In continuous data mode, the board will acquire data based only on the clock and data qualifying signals. There are no acquisition commands in this mode. See Table 2-11.

Table 2-11 ACQ_CON

Initiator	ACQ_CON	Comments
HOST_WCMD_GRAB/ SNAP	0	normal, host initiated GRAB/SNAP/ FREEZE
TRIG_ASRT	1	Triggered initiated GRAB/SNAP/ FREEZE
TRIG_ASRT and HOST_ WCMD_GRAB	2	Triggered SNAP
TRIG_HI	3	Continuous data, wo. CTABs

Note: See also Section 2.7 for more details on the how the acquisition commands work.

Preliminary

Freeze Command Control

This operation is used to stop acquisition when the board is in GRAB mode. Acquisition will stop immediately if the board is between frames, or at the end of the current frame, if the board is in the middle of a frame. See Table 2-12.

Table 2-12 FREEZE_CON

Initiator	FREEZE_CON	Comments
HOST_WCMD_ FREEZE	0	Normal, host initiated
AQ_COUNT or HOST_WCMD_ FREEZE	1	Acquisition counter reaches number of frames programmed in the AQ_COUNT register
TRIG_DASRT	2	Trigger de-asserted

Abort Command Control

This operations terminates the current acquisition immediately. This operation will terminate both a SNAP and a GRAB command. If the board is in the middle of a frame, only part of the frame will be acquired. See Table 2-13.

Table 2-13 ABORT_CON

Initiator	ABORT_CON	Comments
HOST_WCMD_ ABORT	0	Normal, host initiated
TRIG_DASRT or HOST_WCMD_ ABORT	1	Abort on falling edge TRIG or host command.

Preliminary

2.6.2 Horizontal Operations and Events

The horizontal operations and events are related to the horizontal axis (of an image in memory or on the display) or line timing (of a camera). The operations are mainly commands to HCOUNT. Each operation can be initiated by some event. The selection of the event that will initiate the specific operation is done by a set of three control bits related to each operation.

Table 2-14 lists the horizontal events.

Table 2-14 Horizontal Operations

Horizontal operation	Control bits
HCOUNT released from zero	HCNT_RLS_ZERO
HCOUNT reset to zero	HCNT_RST
HCOUNT load with 2000h	HCNT_LD
HCOUNT release from 1FF0h	HCNT_RLS7F0
HCOUNT increment	HCNT_INC
Start horizontal active window	HAW_START

The events that can initiate the horizontal operations are listed in Table 2-15.

Table 2-15 Horizontal Events

Event description	Event Name
ENCODER asserted	ENC_ASRT
ENCODER de-asserted	ENC_DASRT
LEN asserted	LEN_ASRT
LEN de-asserted	LEN_DASRT
ENCODER is HI	ENC_HI
ENCODER is LO	ENC_LO
RESET from HCTAB	RST_HCTAB
RESET from SW	RST_SW
FEN asserted	FEN_ASRT

The sections below enumerate all of the horizontal operations and how the various events can initiate them. The control of each operation is independent from all of the others.

Preliminary

HCOUNT Release From Zero

This operation controls the behavior of HCOUNT when it reached zero (see Table 2-16).

Table 2-16 HCNT_RLS_ZERO

Initiator	HCNT_RLS_ ZERO	Comments
None	0	Normal operation mode, no stop at zero
ENC_ASRT	1	One-shot mode, wait for encoder for release

HCOUNT Reset To Zero

This operation controls how HCOUNT is reset to zero (see Table 2-17).

Table 2-17 HCNT_RST

Initiator	HCNT_RST	Comments
END_OF_HAW	0	Default operation, end of HAW
FEN_ASRT or RST_ HCTAB	1	Reset on FEN_ASRT, Random FEN mode
RST_HCTAB	2	Reset from HCTAB

Note: The HCOUNT can always be reset by RST_SW or HOST_WCMD_ABORT

HCOUNT Release From Stick Point (1FF0h)

This operation controls the behavior of HCOUNT when it hits the stick point. The purpose of the stick point is to allow for very long periods of time between lines. The stick point is located at 1ff0h. See Table 2-18.

Table 2-18 HCNT_RLS_STK

Initiator	HCNT_RLS_STK	Comments
None	0	Normal operation mode, no stop at 1FF0h
HLOAD or HRESET	1	Stay at x1FF0 till load (usually LEN) or reset asserted

Preliminary

HCOUNT Load To 2000h

This operation controls how and when HCOUNT loads (jumps to) 2000h (see Table 2-19).

Table 2-19 HCNT_LD

Initiator	HCNT_LD	Comments
None	0	No load
LEN_ASRT	1	Load on LEN assert, qualified with ENH-LOAD column
ENC_ASRT	2	Load on ENCODER assert, qualified with ENHLOAD column

Preliminary

2.7 Acquisition Command and Status

This section describes how the acquisition state machine works. This state machine controls which frames from the camera are acquired and which are ignored. As the commands can be issued asynchronous to the camera's timing, the acquisition state machine will remember the command and execute it starting at the beginning of the frame. That will guarantee that whole frames will be acquired. Note that the acquisition state machine only marks the frames to be acquired. The amount of pixels/line and lines/frame to be acquired in the marked frame is determined by the HAW and VAW, see section Section 2.4.

The acquisition state machine is controlled by the following signals:

AQCMD, the acquisition command bitfield

VACTIVE, the camera's vertical active timing (usually FEN for area scan cameras)..

TRIGGER, the selected trigger.

ACQ_CON, a bitfield that defines special acquisition modes for the state machine.

The current state of the machine can be observed by the AQCMD and AQSTAT bitfields described below.

2.7.1 The Acquisition Bitfields

The acquisition command bits, AQCMD describe the command to be performed in the next frame. The acquisition status bits, AQSTAT, describe the current command that is performed. The four acquisition commands are described in the Table 2-20.

Table 2-20 ACQCMD

AQCMD	Command	Comment
0 (00b)	FREEZE	Stop acquiring at end of current frame
1 (01b)	ABORT	Stop acquiring immediately, unconditionally
2 (10b)	SNAP	Acquire one frame
3 (11b)	GRAB	Acquire continuously

The following list details the behaviors of these bitfields.

- The AQSTAT bits are set at the beginning of the VACTIVE. The last instance a command can be issued is about 4 LCLKs before the start of the VACTIVE.
- For a SNAP command, when the SNAP starts, the AQCMD bits are cleared. Note that for SNAP, the AQCMD bits are written by the host and cleared by the state machine.
- If during a SNAP/GRAB operation another SNAP/GRAB command is issued, it is ignored.

Preliminary

Table 2-11 below describes the acquisition modes for ACQ_CON

Table 2-21 ACQ_CON

ACQ_CON	Mode Description
0 (000b)	Host command performed on next frame
1 (001b)	Host command issued when TRIGGER asserted
2 (010b)	As long as GRAB command is on, a single frame will be SNAPped at every assertion of the TRIGGER.
3 (011b)	Continuous acquisition mode. Host commands are ignored. Data will be acquired continuously as long as the TRIGGER is asserted.

Figure 2-6 shows a timing diagram of the SNAP command, with ACQ_CON = 0. The command is written by the host during the active frame. The acquisition will start at the beginning of the next frame.

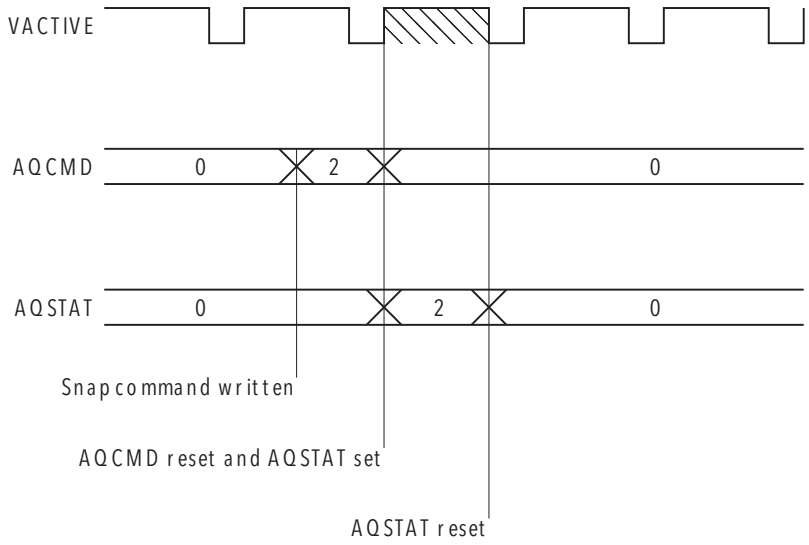


Figure 2-6 Snap Command Timing

Figure 2-7 shows the timing of the GRAB operation with ACQ_CON=0. Figure 2-8 shows the timing of the ABORT operation with ACQ_CON=0. Note that the ABORT command will cut off part of the frame. This command is useful for resetting the board without having to wait for the end of the frame. Figure 2-9 shows a SNAP operation with ACQ_CON=1. In this mode, after the TRIGGER has been asserted and the command executed, the host must write a new command in the AQCMD field. Figure 2-10 shows acquisition in ACQ_CON=2 mode. Here, as long as the GRAB command is on, a frame will be acquired for every assertion of the TRIGGER. In this mode, there is no need for the host to write a new command.

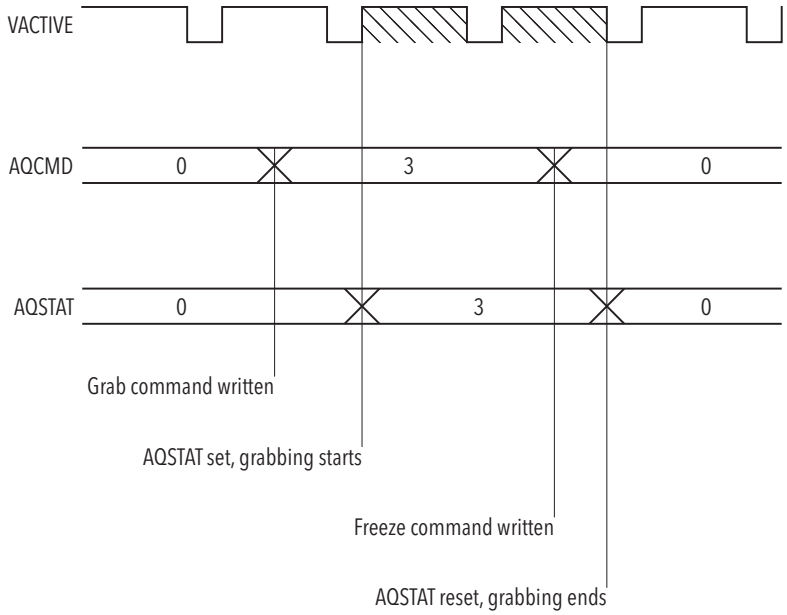


Figure 2-7 Grab Command Timing

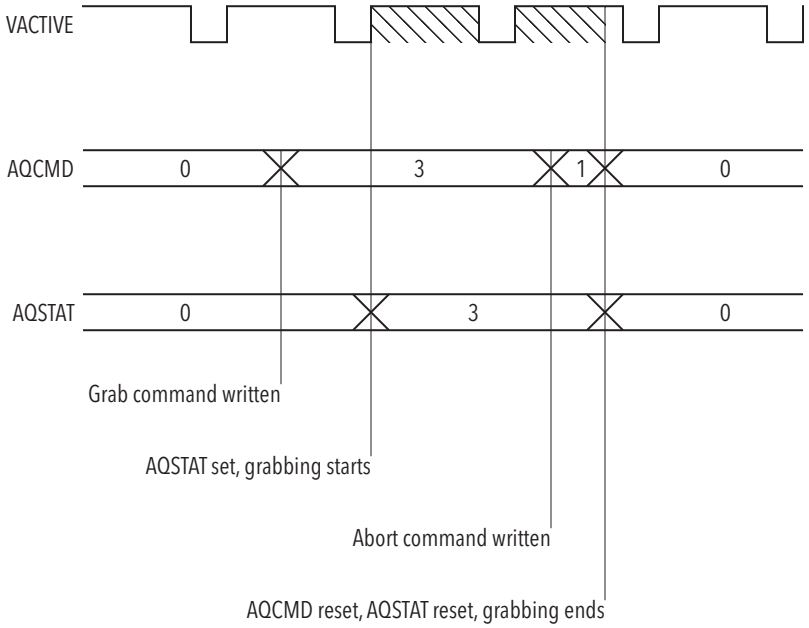


Figure 2-8 Abort Command Timing

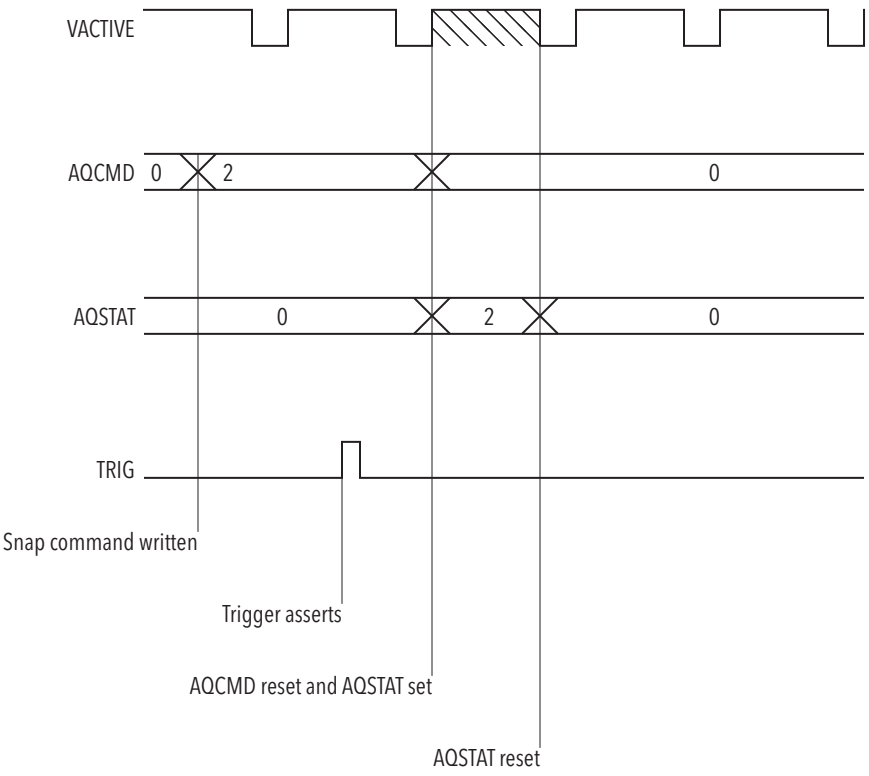


Figure 2-9 Snap Command Timing with ACQ_CON = 2

Prelim

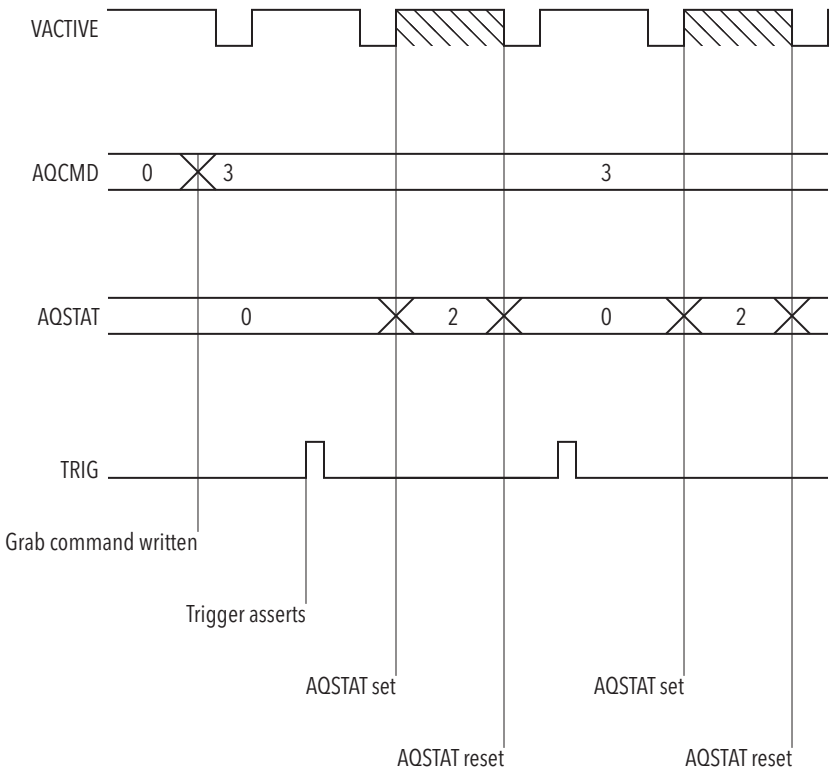


Figure 2-10 Grab Command Timing with ACQ_CON = 2

Prelim

2.8 Trigger Processing

This section describes how the trigger circuit works. The trigger is used to initiate a vertical operation (for example, capturing one frame). There are three possible external hardware inputs to the trigger circuit and a software input. Assertion of the trigger can be delayed by up to 8192 lines (granularity is 8 lines). This delay works only with the external hardware trigger. Figure 2-11 illustrates the trigger circuit.

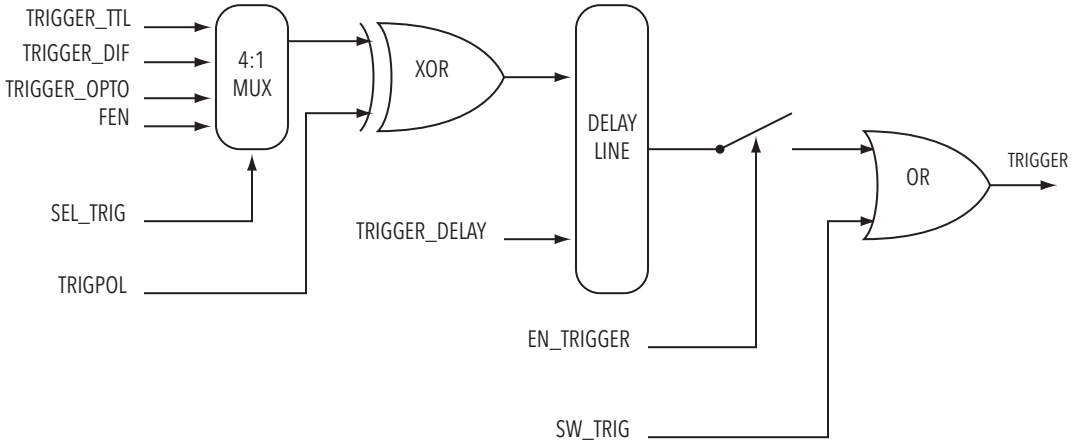


Figure 2-11 Trigger Circuit

Preli

2.9 Encoder Processing

This section describes how the encoder circuit works. The encoder is used to initiate a horizontal operation (for example, capturing one line). There are three possible external hardware inputs to the encoder circuit and a software input. The selected external encoder can be divided by the value in the ENC_DIV register. Figure 2-12 illustrates the encoder circuit.

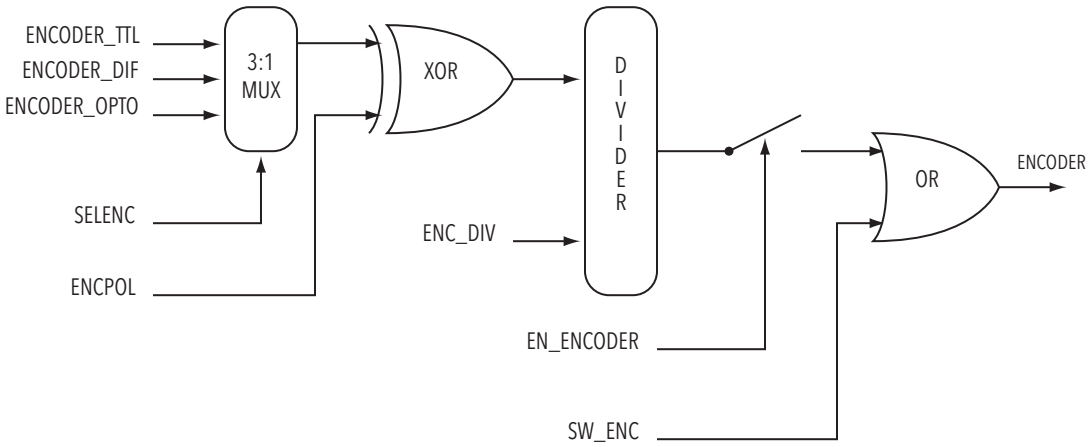


Figure 2-12 Encoder Circuit

Preli

2.10 The On-Board Signal Generator

Part of the acquisition cirtuitry includes an on-board free running signal generator. This signal can be steered to the GPOUTs and/or to the camera CCs. This provides an additional tool for synchronization, triggering and camera control. The output of this generator is programmable. Figure 2-13 shows a block diagram of the generator.

The clock driving the generator is derived from the on-board clock. The frequency is controlled by CFREQ in CON0. The selected clock frequency can be optionally scaled down by 8192, depending on the SCALE_BY8K bit in CON18. The rate of the signal generator is set by FREE_RUN_RATE in CON17. The CLOCK_IN will be divided by the number programmed in FREE_RUN_RATE .

Register FREE_RUN_HI in CON17 defines the amount of CLOCK_INs that the signal will be high in a period. The purpose of this register is to be able to control the duty cycle (pulse width) of the signal generator independent of the frequency.

The output of the generator can be optionally inverted by the SIG_GEN_POL bit in CON18.

Let's look at an example. Assume we want to generate a signal 35 milliseconds wide at a rate of 2Hz. You would then program the signal generator as follows.

Set CFREQ = 5. This will program the clock generator to 30Mhz.

Set SCALE_BY8K = 1. This will divide the 30Mhz by 8K, yielding a CLOCK_IN of 3662.1Hz.

Set FREE_RUN_RATE = 1831 (3662/2). This will produce the 2Hz rate.

Set FREE_RUN_HI = 128 (3662.1x0.035 = 128.1). This will produce a pulse 0.035 seconds wide.

Set SIG_GEN_POL = 0. The output needed is active high.

Preliminary

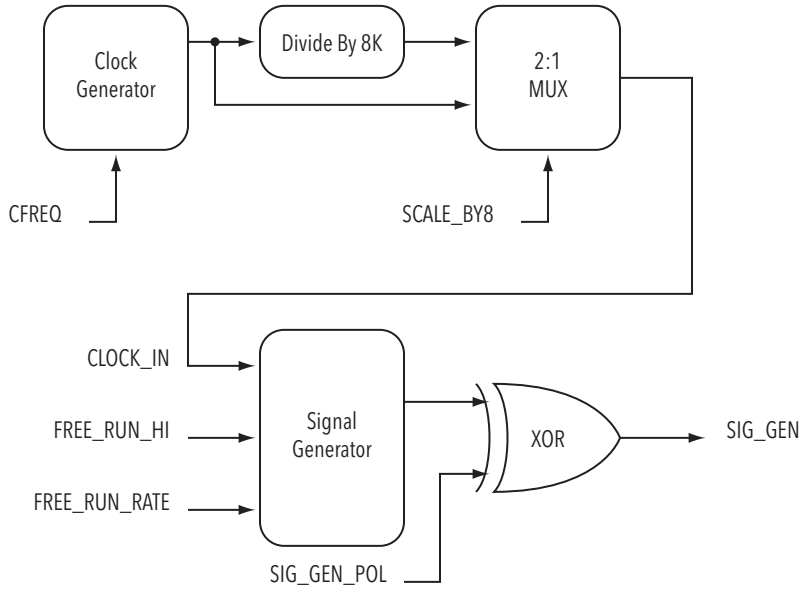


Figure 2-13 Signal Generator

Preliminary

System Status

Chapter 3

3.1 Introduction

This chapter describes the system status report that the board supplies through its registers. The system status will help the users in setting up their system: the camera, the frame grabber, the cabling, the I/O and the software. The list of the status bits is given in the Table 3-1. If more information is available for a given specification there will be an entry in the column marked "Details". In addition, all of these registers are also described in Register Map chapter of this manual.

Table 3-1 Status Bits

Status Bits	Function and Relationship	Register	Details
AQSTAT	Acquistion status	CON3	Section 4.6
FACTIVE	Acquistion status, vertical active	CON3	Section 3.2
FCOUNT	Acquistion status, 3-bit frames counter	CON3	Section 3.2
LCOUNT	Camera status, LEN is toggling	CON4	Section 3.3
PCOUNT	Camera status, PCLK is toggling	CON4	Section 3.3
FENCOUNT	Camera status, FEN is toggling	CON4	Section 3.3
RD_TRIG_DIFF/TTL/OPTO	Trigger status	CON5	Section 3.4
RD_ENC_DIFF/TTL/OPTO	Encoder status	CON5	Section 3.4
VCOUNT	Acquistion status, VCTAB cycling	CON6	Section 3.5
HCOUNT	Acquistion status, HCTAB cycling	CON6	Section 3.5
LINES_TOGO	Acquistion status, current line in frame	CON19	Section 3.5
FIFO_EQ	Camera status, video value	CON20	Section 3.6
DEST_ADD	DMA running	CON22	Section 3.7

Preliminary

3.2 FACTIVE, FCOUNT

FACTIVE is 1 during the active vertical. It works for both area scan and line scan cameras. For both line scan and area scan cameras there is always a vertical size defined by ALPF.

FCOUNT is a 3-bit frame counter that is incremented by the rising edge of FACTIVE. It can be used to track acquisition, especially in triggered modes. FCOUNT works for both area scan and line scan cameras.

Preliminary

3.3 PCOUNT, LCOUNT, FENCOUNT

Thase three registers give an indication of the status the camera connected to the main connector:

PCOUNT is a 2-bit counter clocked by the camera's PCLK. Reading a constant value from this register indicates that the camera's clock does not reach the acquisition circuitry.

LCOUNT is a 2-bit counter clocked by the camera's LEN. Reading a constant value from this register indicates that the camera's LEN does not reach the acquisition circuitry.

FENCOUNT is a 2-bit counter clocked by the camera's FEN. Reading a constant value from this register indicates that the camera's FEN does not reach the acquisition circuitry.

Preliminary

3.4 RD_TRIG_DIFF/TTL/OPTO, RD_ENC_DIFF/TTL/OPTO

The level of all three trigger and all three encoder inputs can be read. This helps establish connection with external industrial equipment.

Preliminary

3.5 VCOUNT, HCOUNT, LINES_TOGO

These three registers give feedback about the operation of the horizontal and vertical CTABs. VCOUNT is the address counter of the VCTAB. This register indicates the current VCTAB address.

HCOUNT is the 2 LSB of the HCTAB address counter. This register indicates only if the HCTAB is cycling. Reading a constant value on HCOUNT indicates that the HCTAB address is stuck.

LINES_TOGO specifies how more many lines there are till the end of the frame.

Preliminary

3.6 FIFO_EQ

This register gives the 8-bit value of the video from the first eight bits of the main connector. It is helpful to determine if the camera is reacting to light. Convering the camera's lense will yield a low value in this register. Pointing the camera to a light source will yield a high value in this register.

Preliminary

3.7 DEST_ADD

This register gives the DMA destination address. During acquisition, this register should change. Reading a constant value from this register suggests that the DMA operation is not progressing.

Preliminary

Preliminary

Camera Control Registers

Chapter 4

4.1 Introduction

This section enumerates all of the bitfields in all of the registers used to control the acquisition and external I/O. If you compare the Karbon, Neon and R64 manuals you will see that almost all of these registers are the same. There are only a few bits that are different between these two models, these will be indicated in the bitfield defnitions. Registers that are related to DMA operations, which are different between the Neon, R64 and the Karbon families, have their own chapters.

All of the registers are 32 bits wide. These wide registers are named CON0, CON1, etc. Each registers is broken into one or more bitfields. Bitfields can be from one to 32 bits wide. Each bitfield controls a specific function on the board.

Preliminary

4.2 Bitfield definitions

4.2.1 Example Bitfield Definition

Here is what each bitfield definition looks like:

BITFIELD R/W, CON0[7..0], Karbon, Neon, R64

BitField discussion.

4.2.2 Bitfield Definition Explanation.

The definitions is broken into three sections (see Table 4-1).

Table 4-1 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 4-2 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.

Preliminary

Table 4-2 explains the abbreviations used in the bitfield definitions.

Table 4-2 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	This bitfield is functional on the Karbon.
Neon	This bitfield is functional on the Neon
R64	This bitfield is functional on the R64 family.

Preliminary

4.3 CON0 Register

Bit	Name
0	CFGDATA
1	CFGSTATUS
2	CFGEN
3	CFGDONE
4	CFGCLOCK
5	FW_7MHZ
6	Reserved
7	POCL_EN
8	CFREQ
9	CFREQ
10	CFREQ
11	Reserved
12	L_CLKCON
13	L_CLKCON
14	SEL_UCLKC_7MHZ
15	RELOAD_FPGA
16	FW_SEL
17	FW_SEL
18	FW_SEL
19	CPLD_MODE
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

Preliminary

CFGDATA R/W, CON0[0], R64

Used for downloading firmware.

CFGSTATUS R/W, CON0[1], R64

Used for downloading firmware.

CFGEN R/W, CON0[2], R64

Used for downloading firmware.

CFGDONE R/W, CON0[3], R64

Used for downloading firmware.

CFGCLOCK R/W, CON0[4], R64

Used for downloading firmware.

FW_7MHZ RO, CON0[5], Karbon, Neon, R64

If this bit is set, then the board has the update Firmware which can program the frequency of the UART clock to 7.3 MHz. If this bit is zero, then the board has the origonal firmware and the UART can only be driver by an 8 MHz clock. See also the bit: SEL_UCLK_7MHz.

POCL_EN R/W, CON0[7], Neon - [New]

This bit turns the the PoCL Safe Power system. This bit must be set to one in order to enable power to PoCL cameras. However, the system uses the Safe Power system, so a number of conditions must be met before power is actually applied to the camera.

Preliminary

CFREQ R/W, CON0[10..8], Karbon, Neon, R64

These bits control the frequency of the CLOCK generated on-board..

CFREQ	Frequency
0 (000b)	DC
1 (001b)	3.75 MHz
2 (010b)	7.5 MHz
3 (011b)	15 MHz
4 (100b)	24 MHz
5 (101b)	30 MHz
6 (110b)	48 MHz
7 (111b)	60 MHz

L_CLKCON0 R/W, CON0[13..12], Karbon, Neon, R64

These bits control the local bus clock frequency. For normal operation, this register should always be set for 00. The other codes are for test/diagnostics.

L_CLKCON	Frequency
0 (000b)	60 MHz
1 (001b)	48 MHz
2 (010b)	24 MHz
3 (011b)	Reserved

SEL_UCLK_ 7MHZ R/W, CON0[14], Karbon, Neon, R64

This bit selects the frequency that is used to driver the UART for serial communications. This functionality is only available on boards with update firmware. The bit FW_ 7MHZ can be used to check the version of the firmware.

SEL_UCLK_ 7MHZ	Frequency
0	8 MHz
1	7.3 MHz

Preliminary

RELOAD_FPGA WO, CON0[15], Karbon, Neon - [New]

Writing to this bit causes the FGPA to be reloaded from Flash memory. This bit shold only be accessed from the driver as the PCI Configuration space is overwritten by this operation. This is not a user programmable bit.

FW_SEL R/W, CON0[18..16], Karbon, Neon, R64 - [New]

These bits are used to select different modes for a given type of firmware. For each major type of CCD tap configuratoin, there is a separate firmware file that is down-loaded to the board. However, in some cases different manufacturers choses slightly different ways to implement the same tap configuration. In these cases this bitfield is used to select between the different modes. As the meaning for this bitfield differ for each firwmare file, and these bits are rarely used, the specific definitions of this bit-field are not enumerated here.

CPLD_MODE R/W, CON0[19], Neon - [New]

On the Neon, the CPLD used to load the FPGAs has two modes. This bit is used to set the mode. This is not a user programmable bit.

Preliminary

4.4 CON1 Register

Bit	Name
0	VCNT_RLS_ZERO
1	VCNT_RLS_ZERO
2	VCNT_RLS_ZERO
3	VCNT_RST
4	VCNT_RST
5	VCNT_RST
6	VCNT_LD
7	VCNT_LD
8	VCNT_LD
9	VCNT_RLS_STK
10	VCNT_RLS_STK
11	VCNT_RLS_STK
12	ABORT_CON
13	ABORT_CON
14	ABORT_CON
15	NO_VB_WAIT
16	ACQ_CON
17	ACQ_CON
18	ACQ_CON
19	FREEZE_CON
20	FREEZE_CON
21	FREEZE_CON
22	ACQ_SAFETY
23	NO_RULE
24	INT_CTAB
25	INT_OVSTEP
26	INT_HW
27	INT_TRIG
28	INT_SER
29	INT_QUAD
30	INT_TRIGCON
31	INT_TRIGCON

Preliminary

VCNT_RLS_ZERO R/W, CON1[2..0], Karbon, Neon, R64

This register controls how the Vertical CTAB counter (VCOUNT) is released from zero.

VCNT_RLS_ZERO	Meaning
0 (000b)	Normal operation. VCOUNT does not stick at zero.
1 (001b)	Edge Mode - VCOUNT sticks at zero. VCOUNT is released from zero by the leading edge of the trigger.
2 (010b)	Level Mode - VCOUNT sticks at zero only if the trigger is de-asserted. If trigger is asserted, then VCOUNT does not stick at zero. VCOUNT is released from zero by the leading edge of trigger.
3 (011b)	Reserved.

VCNT_RST R/W, CON1[5..3], Karbon, Neon, R64

This register controls how the Vertical CTAB counter (VCOUNT) is reset to zero. In all modes the VCOUNT will also be reset by any of the following four signals/events:

The SW_RESET
The ABORT command
The RST_HVCOUNT bit in CON4
VCOUNT reaching a 1 in the VRESET CTAB

VCNT_RST	Meaning
0 (000b)	VCOUNT is reset by the End of Vertical Acquisition Window or by the VRESET column in the VCTAB.
1 (001b)	VCOUNT is reset by the de-assertion of the trigger (triggered termination) or the end of VAW or by the VRESET column in the VCTAB.
2 (010b)	VCOUNT is reset by the VRESET column in the VCTAB.
3 (011b)	VCOUNT is reset by the assertion of FEN or by the VRESET column in the VCTAB.
4 (100b)	VCOUNT is reset by the de-assertion of the trigger, or by the VRESET column in the VCTAB.

Preliminary

VCNT_LD R/W, CON1[8..6], Karbon, Neon, R64

This registers controls how the Vertical CTAB counter (VCOUNT) is loaded with the 8000h value.

VCNT_LD	Meaning
0 (000b)	No load operation performed.
1 (001b)	VCOUNT is loaded at the assertion of FEN qualified with the ENVLOAD column in the VCTAB.
2 (010b)	VCOUNT is loaded at the assertion of FEN.
3 (011b)	VCOUNT is loaded at the assertion of trigger.

VCNT_RLS_STK R/W, CON1[11..9], Karbon, Neon, R64

This register controls the stick/release of the VCOUNT to/from address 7ff0h.

VCNT_RLS_STK	Meaning
0	VCOUNT does not stick at 7FF0h.
1	VCOUNT sticks at 7FF0h. It will be released from that address by a LOAD or RESET operation.

ABORT_CON R/W, CON1[14..12], Karbon, Neon, R64

This register controls the ABORT operation of the Acquisition State Machine.

ABORT_CON	Meaning
0	Acquisition will be aborted by a host command.
1	Acquisition will be aborted by de-assertion of trigger of by a host ABORT command.

NO_VB_WAIT R/W, CON1[15], Karbon, Neon, R64

This bit has the following properties.

NO_VB_WAIT	Meaning
0	Wait for the Vertical Active Window before executing the Head Tag Quad.
1	Do not wait for the Vertical Active Window for executing the Head Tag Quad.

Preliminary

ACQ_CON R/W, CON1[18..16], Karbon, Neon, R64

This register controls the execution of the acquisition command.

ACQ_CON	Meaning
0 (000b)	Acquisition is initiated by host writing the command.
1 (001b)	The acquisition command written by host will start executing at assertion of trigger (triggered acquisition).
2 (010b)	While the GRAB command is on, a frame will be acquired at the assertion of trigger.
3 (011b)	Continuous acquisition mode. Host commands are ignored. Data will be acquired continuously as long as the TRIGGER is asserted.

FREEZE_CON R/W, CON1[21..19], Karbon, Neon, R64

This register controls the FREEZE operation.

FREEZE_CON	Meaning
0 (000b)	FREEZE initiated by host command.
1 (001b)	FREEZE initiated by the Acquisition Counter or by the host command.
2 (010b)	FREEZE initiated by the de-assertion of trigger or by the host command.

ACQ_SAFETY R/W, CON1[22], Karbon, Neon, R64

Future use

NO_RULE R/W, CON1[23], R64

Test/diagnostic bit. For normal operation this bit should always be set to 0. When set to 1, the DMA engine will DMA data at maximum speed, regardless of wether the data is valid.

Preliminary

INT_CTAB R/W, CON1[24], Karbon, Neon, R64

This interrupt will be set by the interrupt bit in the VCTAB or by the host writing to this bit. The interrupt will be enabled if its corresponding mask, ENINT_CTAB, has been set to 1. This interrupt can be cleared by the host writing a 0 to this location. For the host to be able to write to this location, the CMDWRITE code must be set to 1.

INT_CTAB	Meaning
0	No interrupt from CTAB
1	Interrupt from CTAB asserted.

INT_OVSTEP R/W, CON1[25], Karbon, Neon, R64

This interrupt will be set if an overflow occurred in the FIFO or by the host writing to this bit. The interrupt will be enabled if its corresponding mask, ENINT_OVSTP, has been set to 1. This interrupt can be cleared by the host writing a 0 to this location. For the host to be able to write to this location, the CMDWRITE code must be set to 2.

INT_OVSTP	Meaning
0	No interrupt from overflow
1	Interrupt from overflow asserted.

INT_HW R/W, CON1[26], Karbon, Neon, R64

This interrupt will be set by a hardware exception, a loss of sync or by the host writing to this bit. The interrupt will be enabled if its corresponding mask, ENINT_HW, has been set to 1. This interrupt can be cleared by the host writing a 0 to this location. For the host to be able to write to this location, the CMDWRITE code must be set to 3.

INT_HW	Meaning
0	No interrupt from HW
1	Interrupt from HW asserted.

Preliminary

INT_TRIG R/W, CON1[27], Karbon, Neon, R64

This interrupt will be set by a trigger edge or by the host writing to this bit (see register INT_TRIGCON below). The interrupt will be enabled if its corresponding mask, ENINT_TRIG, has been set to 1. This interrupt can be cleared by the host writing a 0 to this location. For the host to be able to write to this location, the CMDWRITE code must be set to 4.

INT_TRIG	Meaning
0	No interrupt from trigger
1	Interrupt from trigger asserted.

INT_SER RO, CON1[18], Karbon, Neon, R64

This interrupt will be set by the on board UART that implements the serial communication protocol. The interrupt will be enabled if its corresponding mask, ENINT_SER, has been set to 1. This interrupt can be cleared by the host writing to the UART.

INT_SER	Meaning
0	No interrupt from UART
1	Interrupt from UART asserted.

INT_QUAD R/W, CON1[29], Karbon, Neon, R64

This interrupt will be set by a DMA QUAD or by the host writing to this bit. The interrupt will be enabled if its corresponding mask, ENINT_QUAD, has been set to 1. This interrupt can be cleared by the host writing a 0 to this location. For the host to be able to write to this location, the CMDWRITE code must be set to 5.

INT_QUAD	Meaning
0	No interrupt from QUAD
1	Interrupt from QUAD asserted.

Preliminary

INT_TRIGCON R/W, CON1[31..20], Karbon, Neon, R64

This register controls the trigger edge that will cause an interrupt:

INT_TRIGCON	Meaning
0 (00b)	reserved
1 (01b)	Assert interrupt on rising edge of trigger.
2 (10b)	Assert interrupt on falling edge of trigger.
3 (11b)	Assert interrupt on both the rising and the falling edge of the trigger.

Preliminary

4.5 CON2 Register

Bit	Name
0	HCNT_RLS_ZERO
1	HCNT_RLS_ZERO
2	HCNT_RLS_ZERO
3	HCNT_RST
4	HCNT_RST
5	HCNT_RST
6	HCNT_LD
7	HCNT_LD
8	HCNT_LD
9	HCNT_RLS_STK
10	HCNT_RLS_STK
11	HCNT_RLS_STK
12	RST_HVCOUNT
13	RST_DPM_ADDR
14	CTABHOLD
15	Reserved
16	CC1_CON
17	CC1_CON
18	CC1_CON
19	CC2_CON
20	CC2_CON
21	CC2_CON
22	CC3_CON
23	CC3_CON
24	CC3_CON
25	CC4_CON
26	CC4_CON
27	CC4_CON
28	CMDWRITE
29	CMDWRITE
30	CMDWRITE
31	QTBSRC

Preliminary

HCNT_RLS_ZERO R/W, CON2[2..0], Karbon, Neon, R64

This register controls the release of the HCOUNT from zero.

HCNT_RLS_ZERO	Meaning
0	HCOUNT does not stop at 000h.
1	HCOUNT stops at 000h. It will be released by the assertion of the encoder.

HCNT_RST R/W, CON2[5..3], Karbon, Neon, R64

This register controls the reset of the HCOUNT. In all cases, the HCOUNT will also be reset by any of the following functions:

SW_RESET
RST_HVCOUNT
ABORT command.

HCNT_RST	Meaning
0 (000b)	HCOUNT will be reset by the end of the Horizontal Active Window.
1 (001b)	HCOUNT will be reset by the assertion of FEN or the HRESET from the HCTAB.
2 (010b)	HCOUNT will be reset by the HRESET in the HCTAB.

HCNT_LD R/W, CON2[8..6], Karbon, Neon, R64

This register controls the loading of the HCOUNT with 2000h.

HCNT_LD	Meaning
0 (000b)	HCOUNT will not be loaded.
1 (001b)	HCOUNT will be loaded by assertion of LEN if the ENHLOAD function in the HCTAB is set to 1.
2 (010b)	HCOUNT will be loaded by assertion of encoder if the ENHLOAD function in the HCTAB is set to 1.

Preliminary

HCNT_RLS_STK R/W, CON2[11..9], Karbon, Neon, R64

This register controls the HCOUNT sticking at 1FF0h.

HCNT_RLS_STK	Meaning
0	HCOUNT will not stick at 1FF0h.
1	HCOUNT will stick at 1FF0h. It will be released by a load or reset command.

RST_HVCOUNT WO, CON2[12], Karbon, Neon, R64

This bit has the following properties.

RST_HVCOUNT	Meaning
0	Normal operation for HCOUNT, VCOUNT
1	Reset HCOUNT, VCOUNT..

RST_DPM_ADDR WO, CON2[13], Karbon, Neon, R64

This bit has the following properties.

RST_DPM_ADDR	Meaning
0	Normal operation for DPM_ADDR
1	Reset DPM_ADDR..

CTABHOLD R/W, CON2[14], Karbon, Neon, R64

This bit has the following properties.

CTABHOLD	Meaing
0	Normal operation for CTABs
1	Freeze outputs and operation of CTABs..

Preliminary

CCI_CON R/W, CON2[18..16], Karbon, Neon, R64

This register selects the signal steered to the CC1.

CC1_CON	Signal steered to CC1
0 (000b)	CT0 from CTAB
1 (001b)	CT1 from CTAB
2 (010b)	CT2 from CTAB
3 (011b)	Free running signal generated on-board
4 (100b)	On-board generated CLOCK
5 (101b)	GPIN0
6 (110b)	0
7 (111b)	1

CC2_CON R/W, CON2[21..19], Karbon, Neon, R64

This register selects the signal steered to the CC2.

CC2_CON	Signal steered to CC2
0 (000b)	CT0 from CTAB
1 (001b)	CT1 from CTAB
2 (010b)	CT2 from CTAB
3 (011b)	Free running signal generated on-board
4 (100b)	On-board generated CLOCK
5 (101b)	GPIN0
6 (110b)	0
7 (111b)	1

Preliminary

CC3_CON R/W, CON2[24..22], Karbon, Neon, R64

This register selects the signal steered to the CC3.

CC3_CON	Signal steered to CC3
0 (000b)	CT0 from CTAB
1 (001b)	CT1 from CTAB
2 (010b)	CT2 from CTAB
3 (011b)	Free running signal generated on-board
4 (100b)	On-board generated CLOCK
5 (101b)	GPIN0
6 (110b)	0
7 (111b)	1

CC4_CON R/W, CON2[27..25], Karbon, Neon, R64

This register selects the signal steered to the CC4. Note that this CC control is slightly different than the previous three. CC4 can be controlled by CT3. This changes allows all four CTs to be tied to a CC.

CC4_CON	Signal steered to CC4
0 (000b)	CT0 from CTAB
1 (001b)	CT1 from CTAB
2 (010b)	CT2 from CTAB
3 (011b)	Free running signal generated on-board
4 (100b)	CT3 from CTAB
5 (101b)	GPIN0
6 (110b)	0
7 (111b)	1

Preliminary

CMDWRITE R/W, CON2[30..28], Karbon, Neon, R64

This registers selects the interrupt bit to be modified by the host. While an interrupt's code is set, that interrupt can not be asserted by its source. It can be modified only by the host. This mechanism allows to perform reliable read-modify-write cycles.

CMDWRITE	Interrupt allowed for host access
0 (000b)	No interrupt can be accessed by host
1 (001b)	INT_CTAB
2 (010b)	INT_OVSTP
3 (011b)	INT_HW
4 (100b)	INT_TRIG
5 (101b)	INT_QTAB
6 (110b)	INT_EOF
7 (111b)	reserved

QTBSRC RO, CON2[31], Karbon, Neon, R64

Always read back 1.

Preliminary

4.6 CON3 Register

Bit	Name
0	AQCMD
1	AQCMD
2	AQSTAT
3	AQSTAT
4	FACTIVE
5	FCOUNT
6	FCOUNT
7	FCOUNT
8	REV_DCC
9	REV_DCC
10	REV_DCC
11	REV_DCC
12	REV_DCC
13	REV_DCC
14	REV_DCC
15	REV_DCC
16	REV_DCC
17	REV_DCC
18	REV_DCC
19	REV_DCC
20	REV_DCC
21	REV_DCC
22	REV_DCC
23	REV_DCC
24	AUX_DETECT
25	GPIN0
26	GPIN1
27	GPIN2
28	GPIN3
29	GPIN4
30	SW
31	SW

Preliminary

AQCMD R/W, CON3[1..0], Karbon, Neon, R64

This register is the acquisition command to be executed in the next frame.

AQCMD	Meaning
0 (00b)	FREEZE
1 (01b)	ABORT
2 (10b)	SNAP
3 (11b)	GRAB

AQSTAT RO, CON3[3..2], Karbon, Neon, R64

The AQSTAT register describes the acquisition command currently being executed.

AQSTAT	Meaning
0 (00b)	FREEZE
1 (01b)	ABORT
2 (10b)	SNAP
3 (11b)	GRAB

FACTIVE RO, CON3[4], Karbon, Neon, R64

FACTIVE	Meaning
0	Camera outside the Vertical Acquisition Window
1	Camera within the Vertical Acquisition Window

FCOUNT RO, CON3[7..5], Karbon, Neon, R64

This is a 3-bit modulo-8 counter. The counter is incremented by the start of the Vertical Acquisition Window. It is used as a debug/diagnostic tool.

REV_DCC WO, CON3[23..8], Karbon, Neon, R64

FW revision.

Preliminary

AUX_DETECT RO, CON3[24], Karbon - [New]

This bit is set to a one if the Karbon auxilliary board is attached.

GPIN0 RO, CON3[25], Karbon, Neon, R64

TTL general purpose input, from pin 13 of the IO connector. The logical value applied to this pin will be reflected in this register.

GPIN1 RO, CON3[26], Karbon, Neon, R64

TTL general purpose input, from pin 14 of the IO connector. The logical value applied to this pin will be reflected in this register

GPIN2 RO, CON3[27], Karbon, Neon, R64

LVDS general purpose input, from pins 15,16 of the IO connector. The logical value applied to those pins will be reflected in this register

GPIN3 RO, CON3[28], Karbon, Neon, R64

LVDS general purpose input, from pins 17, 18 of the IO connector. The logical value applied to those pins will be reflected in this register

GPIN4 RO, CON3[29], Karbon, Neon, R64

LVDS general purpose input, from pins 19, 20 of the IO connector. The logical value applied to those pins will be reflected in this register

SW RO, CON3[31..30], Karbon, Neon, R64

This register reads back the setting of the two mechanical switches installed on the edge of the board. Helps physical identification of up to four boards installed in one PC.

Preliminary

4.7 CON4 Register

Bit	Name
0	ENINT_CTAB
1	ENINT_OVSTEP
2	ENINT_HW
3	ENINT_TRIG
4	ENINT_SER
5	ENINT_QUAD
6	EOF_IN_AQ
7	INT_ANY
8	ENINT_ALL
9	AUX_CAM
10	GPOUT0
11	GPOUT1
12	GPOUT2
13	GPOUT3
14	GPOUT4
15	GPOUT5
16	GPOUT6
17	RST_SER
18	OVS
19	RST_OVS
20	CL_DISABLE
21	LCOUNT
22	LCOUNT
23	PCOUNT
24	PCOUNT
25	FENCOUNT
26	FENCOUNT
27	POP_TOSS
28	PUMP_OFF
29	DMA_BUSY
30	HAW_START
31	VAW_START

Preliminary

ENINT_CTAB R/W, CON4[0], Karbon, Neon, R64

This bit has the following properties.

ENINT_CTAB	Meaning
0	CTAB interrupt disabled
1	CATB interrupt enabled

ENINT_OVSTEP R/W, CON4[1], Karbon, Neon, R64

This bit has the following properties.

ENINT_OVSTEP	Meaning
0	OVERSTEP interrupt disabled
1	OVERSTEP interrupt enabled

ENINT_HW R/W, CON4[2], Karbon, Neon, R64

This bit has the following properties.

ENINT_HW	Meaning
0	HW interrupt disabled
1	HW interrupt enabled

ENINT_TRIG R/W, CON4[3], Karbon, Neon, R64

This bit has the following properties.

ENINT_TRIG	Meaning
0	Trigger interrupt disabled
1	Trigger interrupt enabled

Preliminary

ENINT_SER R/W, CON4[4], Karbon, Neon, R64

This bit has the following properties.

ENINT_SER	Meaning
0	UART interrupt disabled
1	UART interrupt enabled

ENINT_QUAD R/W, CON4[5], Karbon, Neon, R64

This bit has the following properties.

ENINT_QUAD	Meaning
0	QUAD interrupt disabled
1	QUAD interrupt enabled

EOF_IN_AQ R/W, CON4[6], Karbon, Neon, R64

This bit enables gating the INT_EOF interrupt with acquisition. This functionality makes it easier to write software that relies on the INT_EOF interrupt. Without this functionality, the INT_EOF interrupt would occur continuously based on the camera or trigger's timing, even if the board is not currently acquiring. In this case, the software will have to flush these interrupts out of the queue before starting acquisition. With this functionality enable, interrupts only occur during acquisition..

EOF_IN_AQ	Meaning
0	INT_EOF interrupt will be asserted unconditionally at the end of the frame.
1	INT_EOF interrupt will be asserted at the end of the frame only if the board is acquiring, i.e. only during SNAP/GRAB states.

INT_ANY RO, CON4[7], Neon, R64 - [New]

On the products that use the PLDA engine, this bit indicates that an interrupt was emitted by the board. This bit can be checked first to see if some event caused the interrupt, before inquiring other bits to see the actual cause of the interrupt.

ENINT_ALL R/W, CON4[8], Neon, R64 - [New]

This bit enables or disables all interrupts on boards that use the PLDA engine.

Preliminary

AUX_CAM R/W, CON4[8], Neon, R64 - [New]

???

GPOUT0 R/W, CON4[10], Karbon, Neon, R64

The value written in this register will be reflected on the differential outputs 22, 23 on the IO connector. See also CON8 for signals steered to the GPOUTs.

GPOUT1 R/W, CON4[11], Karbon, Neon, R64

The value written in this register will be reflected on the differential outputs 24, 25 on the IO connector. See also CON8 for signals steered to the GPOUTs.

GPOUT2 R/W, CON4[12], Karbon, Neon, R64

The value written in this register will be reflected on the differential outputs 26, 27 on the IO connector. See also CON8 for signals steered to the GPOUTs.

GPOUT3 R/W, CON4[13], Karbon, Neon, R64

The value written in this register will be reflected on the TTL output 28 on the IO connector. See also CON8 for signals steered to the GPOUTs.

GPOUT4 R/W, CON4[14], Karbon, Neon, R64

The value written in this register will be reflected on the TTL output 29 on the IO connector. See also CON8 for signals steered to the GPOUTs

GPOUT5 R/W, CON4[15], Karbon, Neon, R64

The value written in this register will be reflected on the open collector output 31 on the IO connector. See also CON8 for signals steered to the GPOUTs

GPOUT6 R/W, CON4[16], Karbon, Neon, R64

The value written in this register will be reflected on the open collector output 33 on the IO connector. See also CON8 for signals steered to the GPOUTs

Preliminary

RST_SER R/W, CON4[17], Karbon, Neon, R64

This bit has the following properties.

RST_SER	Meaning
0	UART normal operation
1	UART's reset line asserted

OVS RO, CON4[18], Karbon, Neon, R64

This is a latched overstep bit.

OVS	Meaning
0	No overstep occurred since this bit was cleared
1	At least one overstep occurred since this bit was cleared

RST_OVS R/W, CON4[19], Karbon, Neon, R64

This bit has the following properties.

RST_OVS	Meaning
0	OVS bit in normal operation
1	OVS bit is reset

CL_DISABLE R/W, CON4[20], Karbon, Neon, R64

This bit enables/disables the CL chips; used for diagnostics. For normal operation this bit should always be set to 0.

CL_DISABLE	Meaning
0	All CL receivers are enabled.
1	All CL receivers are disabled.

LCOUNT RO, CON4[22..21], Karbon, Neon, R64

This is a 2-bit counter clocked by the LEN supplied by the Camera Link main connector. Reading this counter and observing changes between reads indicates an active LEN.

Preliminary

PCOUNT RO, CON4[24..23], Karbon, Neon, R64

This is a 2-bit counter clocked by the PCLK supplied by the Camera Link main connector. Reading this counter and observing changes between reads indicates an active PCLK.

FENCOUNT RO, CON4[26..25], Karbon, Neon, R64

This is a 2-bit counter clocked by the FEN supplied by the Camera Link main connector. Reading this counter and observing changes between reads indicates an active FEN.

POP_TOSS R/W, CON4[27], R64, Karbon, Neon, R64

For normal operation this bit should be set to 0. It is used for high-level clean-up.

POP_TOSS	Meaning
0	DMA engine's normal operation.
1	DMA engine flushes the receive FIFO without executing any QUADs.

PUMP_OFF R/W, CON4[28], R64, Karbon, Neon, R64

For normal operation this bit should be set to 0. It is used for high-level clean-up.

PUMP_OFF	Meaning
0	DMA engine's normal operation.
1	Inhibit DMA operation.

DMA_BUSY RO, CON4[29], R64, Karbon, Neon, R64

This bit indicates the state of the DMA engine.

DMA_BUYS	Meaning
0	DMA engine is idle.
1	DMA engine is currently DMAing data.

Preliminary

HAW_START R/W, CON4[30], Karbon, Neon, R64

This bit has the following properties.

HAW_START	Meaning
0	The start of the Horizontal Active Window (HAW) is controlled by the start of the LEN.
1	The start of the Horizontal Active Window is controlled by the HSTART column in the HCTAB.

VAW_START R/W, CON4[31], Karbon, Neon, R64

This bit has the following properties.

VAW_START	Meaning
0	The start of the Vertical Active Window (VAW) is controlled by the start of the FEN.
1	The start of the Vertical Active Window is controlled by the VSTART column in the VCTAB.

Preliminary

4.8 CON5 Register

Bit	Name
0	SEL_TRIG
1	SEL_TRIG
2	TRIGPOL
3	SW_TRIG
4	SELENC
5	SELENC
6	ENCPOL
7	SW_ENC
8	RD_TRIG_DIFF
9	RD_TRIG_TTL
10	RD_TRIG_OPTO
11	RD_ENC_DIFF
12	RD_ENC_TTL
13	RD_ENC_OPTO
14	TRIGGER_DELAY
15	TRIGGER_DELAY
16	TRIGGER_DELAY
17	TRIGGER_DELAY
18	TRIGGER_DELAY
19	TRIGGER_DELAY
20	TRIGGER_DELAY
21	TRIGGER_DELAY
22	TRIGGER_DELAY
23	TRIGGER_DELAY
24	ENINT_EOF
25	INT_EOF
26	RD_FEN
27	CCSYNC
28	CCSYNC
29	CCSYNC
30	EN_TRIGGER
31	EN_ENCODER

Preliminary

SEL_TRIG R/W, CON5[1..0], Karbon, Neon, R64

Controls the source of the internal trigger signal.

SEL_TRIG	Meaning
0 (00b)	The trigger used by the board is the differential trigger on the IO connector.
1 (01b)	The trigger used by the board is the TTL trigger on the IO connector.
2 (10b)	The trigger used by the board is the opto-coupled trigger on the IO connector.
3 (11b)	The FEN signal on the CL1 connector will be used as trigger. When this mode is used, the register FENPOL is used to control the polarity of the trigger signal.

TRIGPOL R/W, CON5[2], Karbon, Neon, R64

This bit has the following properties.

TRIGPOL	Meaning
0	Trigger is asserted on the rising edge.
1	Trigger is asserted on the falling edge.

SW_TRIG R/W, CON5[3], Karbon, Neon, R64

The SW trigger is OR-ed with the external trigger. The polarity of the SW trigger is always active-HI. TRIGPOL has no effect on the SW trigger.

SW_TRIG	Meaning
0	SW trigger de-asserted.
1	SW trigger asserted.

Preliminary

SELENC R/W, CON5[5..4], Karbon, Neon, R64

This bitfield has the following properties.

SELENC	Meaning
0 (00b)	The encoder used by the board is the differential encoder on the IO connector.
1 (01b)	The encoder used by the board is the TTL encoder on the IO connector.
2 (10b)	The encoder used by the board is the opto-coupled encoder on the IO connector.
3 (11b)	Reserved

ENCPOL R/W, CON5[6], Karbon, Neon, R64

This bitfield has the following properties.

ENCPOL	Meaning
0	Encoder is asserted on rising edge.
1	Encoder is asserted on falling edge.

SW_ENC R/W, CON5[7], Karbon, Neon, R64

The SW encoder is OR-ed with the external encoder. The polarity of the SW encoder is always active-HI. ENCPOL has no effect on the SW encoder

SW_ENC	Meaning
0	SW encoder de-asserted.
1	SW encoder asserted.

RD_TRIG_DIFF RO, CON5[8], Karbon, Neon, R64

This register reflects the status of the differential trigger input on the IO connector, pins 1,2.

RD_TRIG_TTL RO, CON5[9], Karbon, Neon, R64

This register reflects the status of the TTL trigger input on the IO connector, pin 3.

Preliminary

RD_TRIG_OPTO RO, CON5[10], Karbon, Neon, R64

This register reflects the status of the opto-coupled trigger input on the IO connector, pins 4,5.

RD_ENC_DIFF RO, CON5[11], Karbon, Neon, R64

This register reflects the status of the differential encoder input on the IO connector, pins 7,8.

RD_ENC_TTL RO, CON5[12], Karbon, Neon, R64

This register reflects the status of the TTL encoder input on the IO connector, pin 9

RD_ENC_OPTO RO, CON5[13], Karbon, Neon, R64

This register reflects the status of the opto-coupled encoder input on the IO connector, pins 10, 11.

TRIGGER_DELAY R/W, CON5[23..14], Karbon, Neon, R64

The number N written in this register will delay the trigger by 8N lines.

ENINT_EOF R/W, CON4[24], Karbon, Neon, R64

This bitfield has the following properties.

ENINT_EOF	Meaning
0	End of frame interrupt disabled.
1	End of frame interrupt enabled.

INT_EOF R/W, CON4[25], Karbon, Neon, R64

This interrupt will be set by the acquisition state machine at the end of the frame (end of VAW). This ordinarily corresponds to the camera's end of frame. However, if the board is in start-stop triggered mode, this interrupt with also occur when the trigger de-asserts. The host writing a 1 to this bit will also cause and interrupt. The interrupt

Preliminary

will be enabled if its corresponding mask, ENINT_EOF, has been set to 1. This interrupt can be cleared by the host writing a 0 to this location. For the host to be able to write to this location, the CMDWRITE code must be set to 6.

INT_TRIG	Meaning
0	No interrupt from end of frame.
1	Interrupt from end of frame.

RD_FEN RO, CON5[26], Karbon, Neon, R64

This register reflects the status of the FEN signal on the CL1 connector.

CC_SYNC R/W, CON5[29..26], Karbon, Neon, R64

This register controls how the CC outputs are synchronized.

CCSYNC	Meaning
0 (000b)	CCs are not synchronized
1 (001b)	CCs are synchronized to the pixel clock of the primary camera
2 (010b)	CCs are synchronized to the pixel clock of the secondary camera
3 (011b)	Each set of CCs are synchronized to the pixel clock of their corresponding camera. In other words, the CCs on the primary connector are synchronized to the primary camera's pixel clock, and the CCs on the secondary camera are synchronized to secondary camera's pixel clock.
4 (100b)	Reserved
5 (101b)	Reserved
6 (110b)	Reserved
7 (111b)	Reserved

EN_TRIGGER R/W, CON5[30], Karbon, Neon, R64

This bitfield has the following properties.

EN_TRIGGER	Meaning
0	External (HW) selected trigger is disabled.
1	External (HW) selected trigger is enabled.

Preliminary

EN_ENCODER R/W, CON5[31], Karbon, Neon, R64

This bitfield has the following properties.

EN_ENCODER	Meaning
0	External (HW) selected encoder is disabled.
1	External (HW) selected encoder is enabled.

Preliminary

4.9 CON6 Register

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

Preliminary

VCOUNT RO, CON6[16..0], Karbon, Neon, R64

This is the value of the VCOUNT, the VCTAB's address counter.

LAL R/W, CON6[17], Karbon, Neon, R64

LAL stands for Last Active Line. It controls the mode the VCOUNT is read.

LAL	Meaning
0	VCOUNT register reflects the current value of the VCOUNT.
1	VCOUNT register holds the last active line of the previous frame.

ENC_DIV R/W, CON6[27..18], Karbon, Neon, R64

The encoder pulses will be divided down by the number written in this register. If, for example, ENC_DIV[] = 5, for every five pulses on the selected encoder, the divider will supply one pulse to the board. Programming this register to 0 or 1 will both divide by 1.

HCOUNT R/W, CON6[29..28], Karbon, Neon, R64

This register reflects the current value of the two LSBs of the HCOUNT. Reading this register and observing changes in its value means that the HCOUNT is cycling.

Preliminary

4.10 CON7 Register

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

Preliminary

AQ_COUNT R/W, CON7[19..0], Karbon, Neon, R64

The number N written in this register is used to tell the acquisition logic to FREEZE acquisition after N frames have been acquired. The register FREEZE_CON code must be set to 1.

SEL_REG_GEN R/W, CON7[21..20], Karbon, Neon, R64 - [New]

???

GEN_ONESHOT R/W, CON7[22], R64 - [New]

This bit controls the mode of the special signal generator available in the TVI camera specific firmware.

GEN_ONESHOT	Meaning
0	Signal generator is free running.
1	Signal generator syncrhonized to the external encoder signal.

TAG_BANK RO, CON7[29..24], R64

This is the calculated bank from the address generator latched by the TAG QUAD; diagnostics/test register.

Preliminary

4.11 CON8 Register

Bit	Name
0	GPOUT0_CON
1	GPOUT0_CON
2	GPOUT0_CON
3	GPOUT1_CON
4	GPOUT1_CON
5	GPOUT1_CON
6	GPOUT2_CON
7	GPOUT2_CON
8	GPOUT2_CON
9	GPOUT3_CON
10	GPOUT3_CON
11	GPOUT3_CON
12	GPOUT4_CON
13	GPOUT4_CON
14	GPOUT4_CON
15	GPOUT5_CON
16	GPOUT5_CON
17	GPOUT5_CON
18	GPOUT6_CON
19	GPOUT6_CON
20	GPOUT6_CON
21	Reserved
22	Reserved
23	Reserved
24	RLE_LOAD_H
25	RLE_LOAD_H
26	RLE_LOAD_H
27	RLE_LOAD_H
28	RLE_LOAD_V
29	RLE_LOAD_V
30	RLE_LOAD_V
31	RLE_LOAD_V

Preliminary

GPOUT0_CON R/W, CON8[2..0], Karbon, Neon, R64

This register selects the signal steered to GPOUT0 on the IO connector

GPOUT0_CON	Selected signal steered to GPOUT0
0 (000b)	GPOUT0 bit written by host in CON4
1 (001b)	CT0 from CTAB.
2 (010b)	CT1 from CTAB.
3 (011b)	CT2 from CTAB.
4 (100b)	CT3 from CTAB.
5 (101b)	Internally generated CLOCK (frequency controlled by CFREQ in CON1).
6 (110b)	Internally generated signal (frequency and duty-cycle controlled by CON17).
7 (111b)	The encoder input signal is routed to the GPOUT0 output signal.

GPOUT1_CON R/W, CON8[5..3], Karbon, Neon, R64

This register selects the signal steered to GPOUT1 on the IO connector

GPOUT1_CON	Selected signal steered to GPOUT1
0 (000b)	GPOUT1 bit written by host in CON4
1 (001b)	CT0 from CTAB.
2 (010b)	CT1 from CTAB.
3 (011b)	CT2 from CTAB.
4 (100b)	CT3 from CTAB.
5 (101b)	Internally generated CLOCK (frequency controlled by CFREQ in CON1).
6 (110b)	Internally generated signal (frequency and duty-cycle controlled by CON17).
7 (111b)	The encoder input signal is routed to the GPOUT0 output signal.

Preliminary

GPOUT2_CON R/W, CON8[8..6], Karbon, Neon, R64

This register selects the signal steered to GPOUT2 on the IO connector

GPOUT2_CON	Selected signal steered to GPOUT2
0 (000b)	GPOUT2 bit written by host in CON4
1 (001b)	CT0 from CTAB.
2 (010b)	CT1 from CTAB.
3 (011b)	CT2 from CTAB.
4 (100b)	CT3 from CTAB.
5 (101b)	Internally generated CLOCK (frequency controlled by CFREQ in CON1).
6 (110b)	Internally generated signal (frequency and duty-cycle controlled by CON17).
7 (111b)	reserved.

GPOUT3_CON R/W, CON8[11..9], Karbon, Neon, R64

This register selects the signal steered to GPOUT3 on the IO connector

GPOUT3_CON	Selected signal steered to GPOUT3
0 (000b)	GPOUT3 bit written by host in CON4
1 (001b)	CT0 from CTAB.
2 (010b)	CT1 from CTAB.
3 (011b)	CT2 from CTAB.
4 (100b)	CT3 from CTAB.
5 (101b)	Internally generated CLOCK (frequency controlled by CFREQ in CON1).
6 (110b)	Internally generated signal (frequency and duty-cycle controlled by CON17).
7 (111b)	reserved.

Preliminary

GPOUT4_CON R/W, CON8[14..12], Karbon, Neon, R64

This register selects the signal steered to GPOUT4 on the IO connector

GPOUT4_CON	Selected signal steered to GPOUT4
0 (000b)	GPOUT4 bit written by host in CON4
1 (001b)	CT0 from CTAB.
2 (010b)	CT1 from CTAB.
3 (011b)	CT2 from CTAB.
4 (100b)	CT3 from CTAB.
5 (101b)	Internally generated CLOCK (frequency controlled by CFREQ in CON1).
6 (110b)	Internally generated signal (frequency and duty-cycle controlled by CON17).
7 (111b)	reserved.

GPOUT5_CON R/W, CON8[17..15], Karbon, Neon, R64

This register selects the signal steered to GPOUT5 on the IO connector

GPOUT5_CON	Selected signal steered to GPOUT5
0 (000b)	GPOUT5 bit written by host in CON4
1 (001b)	CT0 from CTAB.
2 (010b)	CT1 from CTAB.
3 (011b)	CT2 from CTAB.
4 (100b)	CT3 from CTAB.
5 (101b)	Internally generated CLOCK (frequency controlled by CFREQ in CON1).
6 (110b)	Internally generated signal (frequency and duty-cycle controlled by CON17).
7 (111b)	reserved.

Preliminary

GPOUT6_CON R/W, CON8[20..18], Karbon, Neon, R64

This register selects the signal steered to GPOUT6 on the IO connector

GPOUT6_CON	Selected signal steered to GPOUT6
0 (000b)	GPOUT6 bit written by host in CON4
1 (001b)	CT0 from CTAB.
2 (010b)	CT1 from CTAB.
3 (011b)	CT2 from CTAB.
4 (100b)	CT3 from CTAB.
5 (101b)	Internally generated CLOCK (frequency controlled by CFREQ in CON1).
6 (110b)	Internally generated signal (frequency and duty-cycle controlled by CON17).
7 (111b)	reserved.

RLE_LOAD_H R/W, CON8[27..24], Karbon, Neon - [New]

For boards that use RLE CTabs, this register controls the location that the horizontal RLE counter jumps to when the LEN signal is asserted. The units of value in this bitfield is RLE entry, not the CTAB location. In other words, if the jump point is 0x8000 CTAB location, but the RLE entry for this location is 3, then this register should be programmed to 3.

RLE_LOAD_V R/W, CON8[31..28], Karbon, Neon - [New]

For boards that use RLE CTabs, this register controls the location that the vertical RLE counter jumps to when the FEN signal is asserted. The units of value in this bitfield is RLE entry, not the CTAB location. In other words, if the jump point is 0x20000 CTAB location, but the RLE entry for this location is 3, then this register should be programmed to 3.

Preliminary

4.12 CON9 Register

Bit	Name
0	MUX_REV
1	MUX_REV
2	MUX_REV
3	MUX_REV
4	MUX_REV
5	MUX_REV
6	MUX_REV
7	MUX_REV
8	MUX_REV
9	MUX_REV
10	MUX_REV
11	MUX_REV
12	TRIM
13	TRIM
14	TRIM
15	TRIM
16	FW_TYPE
17	FW_TYPE
18	FW_TYPE
19	FW_TYPE
20	DISPLAY
21	CLIP
22	SHORT_FRAME
23	RST_CALC_BANK
24	CALC_BANK
25	CALC_BANK
26	CALC_BANK
27	CALC_BANK
28	CALC_BANK
29	CALC_BANK
30	ACPL_MUL
31	ACPL_MUL

Preliminary

MUX_REV RO, CON9[11..0], Karbon, Neon, R64

Firmware revision.

TRIM R/W, CON9[15..12], Karbon, Neon, R64

This bit field will delay the LEN relative to the video. The delay, in units of pixels, equals the number programmed in the TRIM[] field.

The net effect for a simple, one tap camera will be a shifting to the left of the displayed image. For multi-tap cameras the visual effect is more complex and depends on the CCD architecture.

The purpose of this bit field is to align the image presented by the different taps. This bit field has the opposite effect of the DELAY field in CON14.

TRIM	Meaning
0 (000b)	LEN is not delayed
1 (001b)	LEN is delayed by 1 clocks
2 (010b)	LEN is delayed by 2 clocks
3 (011b)	LEN is delayed by 3 clocks
4 (100b)	LEN is delayed by 4 clocks
5 (101b)	LEN is delayed by 5 clocks
6 (11ob)	LEN is delayed by 6 clocks
7 (111b)	LEN is delayed by 7 clocks

FW_TYPE RO, CON9[19..16], Karbon, Neon, R64

Firmware type.

DISPLAY R/W, CON9[20], Karbon, Neon, R64

This bit controls the acquisition of data that is more than 8 bits/pixel. When this bit is set, only the 8 LSB of the data will be acquired in each lane. To be able to display the 8 MSB (or any other consecutive group of 8 bits), the data must be shifted accordingly with the barrel shifter. For 9 to 16-bit cameras, setting this bit will result in an 8-bit dis-

Preliminary

playable pixel. For high bit depth color cameras, setting this bit will result in a 24-bit color displayable pixel. In both cases. the barrel shifter must be set correctly to get the displayable bits out of the incoming pixels.

DISPLAY	Meaning
0	Acquire full bit depth.
1	Acquire 8 LSBs of each data lane.

CLIP R/W, CON9[21], Karbon, Neon, R64

This bit will clip the upper and lower 15 gray levels. This is useful for displaying gray level images on VGA monitors set in 256 colors mode. The upper and lower 15 gray levels are dedicated to Windows graphics.

CLIP	Meaning
0	Acquire data as is.
1	Clip upper and lower 15 gray levels to 240 and 15 respectively.

SHORT_FRAME R/W, CON9[22], Karbon, Neon, R64

Future use.

RST_CALC_BANK R/W, CON9[23], Karbon, Neon, R64

For normal operation this bit should be 0.

RST_CALC_BANK	Meaning
0	Normal operation
1	Reset the calculated starting bank.

CALC_BANK RO, CON9[29..24], Karbon, Neon, R64

Value of the current calculated starting bank.

Preliminary

ACPL_MUL R/W, CON9[31..30], Karbon, Neon, R64

This register is used to increase the maximum line size the board can acquire. The settings act as muliplier for the ACPL (Active Clocks Per Line) register.

ACPL_MUL	Meaning
0 (00b)	Normal operation. ACPL is used as is
1 (01b)	ACPL is multiplied by 2
2 (10b)	Reserved
3 (11b)	Reserved

Preliminary

4.13 CON10 Register

Bit	Name
0	ACPL
1	ACPL
2	ACPL
3	ACPL
4	ACPL
5	ACPL
6	ACPL
7	ACPL
8	ACPL
9	ACPL
10	ACPL
11	ACPL
12	ACPL
13	ACPL
14	ACPL
15	ACPL
16	ACPL
17	FORMAT
18	FORMAT
19	FORMAT
20	FORMAT
21	FORMAT
22	VID_SOURCE
23	VID_SOURCE
24	VID_SOURCE
25	VID_SOURCE
26	PIX_DEPTH
27	PIX_DEPTH
28	PIX_DEPTH
29	PIX_DEPTH
30	PIX_DEPTH
31	FORCE_8BIT

Preliminary

ACPL R/W, CON10[16..0], Karbon, Neon, R64

This register defines the Active Clocks Per Line of the horizontal acquisition window. Let's assume for example a single tap camera with 2K pixels per line. If we want to acquire 400 pixels per line, the ACLP will be programmed to 400. For a 2-tap odd-even pixels camera, with 2K pixels per line, to acquire 400 pixels the ACLP will be programmed with the value 200, as for every clock the camera supplies two pixels.

FORMAT RO, CON10[21..17], Karbon, Neon, R64

This register defines the camera(s) format in terms of taps and scanning architecture. For every FORMAT there is an associated firmware that is downloaded in the FPGAs. The firmware is identified in the camera file by the FORMAT.

FORMAT	Firmware Name	Format Description
0 (00000b)	MUX	1 tap cameras
1 (00001b)	MUX_2TOEP	2 taps, odd-even pixels
2 (00010b)	MUX_2TOEL	2 taps, odd-even lines
3 (00011b)	MUX_2TS	2 taps, segmented
4 (00100b)	MUX_2TS1RI	2 taps, segmented, right inverted
5 (00101b)	MUX_4TS	4 taps, segmented
6 (00110b)	MUX_4T2S2RIOEP	4 taps, odd-even pixels, right taps inverted
7 (00111b)	MUX_4TQ2RI2BU	4 quads, right quads inverted, bottom quads upside down
8 (01000b)	MUX_2CAM	2 cameras: 1 tap each
9 (01001b)	MUX_2CAM_2TOEP	2 cameras: 2 taps, odd-even pixels
10 (01010b)	MUX_2CAM_2TS1RI	2 cameras: 2 taps, segmented, right-inverted
11 (01011b)	MUX_2CAM_2TS	2 cameras: 2 taps, segmented
12 (01100b)	MUX_2CAM_2TOEL	2 cameras: 2 taps, odd-even lines
13 (01101b)	MUX_8TS	8 taps, segmented
14 (01110b)	MUX_BAY	Bayer decoder, 1 tap 8 bit
15 (01111b)	MUX_BAY_OE	Bayer decoder, 2 taps, odd-even pixels
16 (10000b)	MUX_BAY_2TS	Bayer decoder, 2 taps, segmented
17 (10001b)	MUX_4WI	4 taps, 4-way interleaved

Preliminary

FORMAT	Firmware Name	Format Description
18 (10010b)	MUX_2TOEPI	2 taps, odd-even pixels, both inverted
19 (10011b)	MUX_1TI	1 tap, inverted
20 (10100b)	MUX_8WI	8 taps, 8-way interleaved
21 (10101b)	MUX_BAY_2TS_RI	Bayer decoder, 2 taps, segmented, right inverted

VID_SOURCE R/W, CON10[25..22], Karbon, Neon, R64

This register defines the video source and the pattern for the synthetic video. The synthetic patterns appear on all 8 taps, except for code 9.

VID_SOURCE	Video source
0 (0000b)	Camera
1 (0001b)	Camera, special mode for cameras that do not assert the VALID signal.
2 (0010b)	reserved
3 (0011b)	Synthetic horizontal static wedge
4 (0100b)	Synthetic dynamic wedge
5 (0101b)	Synthetic 00h
6 (0110b)	Synthetic FFh
7 (0111b)	Synthetic AAh
8 (1000b)	Synthetic 55h
9 (1001b)	Synthetic ABCDEF0123456789h
10 (1010b)	Synthetic vertical static wedge

Preliminary

PIX_DEPTH R/W, CON10[30..26], Karbon, Neon, R64

This register defines the pixel depth as well as the color order and packing mode for RGB cameras.

PIX_DEPTH	Bit/pixel, color order and packing
0 (0000b)	8 bits
1 (0001b)	10 bits
2 (0010b)	12 bits
3 (0011b)	14 bits

PIX_DEPTH	Bit/pixel, color order and packing
4 (0100b)	16 bits
5 (0101b)	3x8 bits BGR, DMAed as 32 bits, upper MSB set to 00h
6 (0110b)	3x8 bits BGR, DMAed as 24 bits (packed)
7 (0111b)	3x10 bits RGB, DMAed as 32 bits, display mode is 24 bits
8 (1000b)	3x12 bits RGB, DMAed as 48 bits (packed), display mode is 24 bits
9 (1001b)	32 bits
10 (1010b)	64 bits
11 (1011b)	3x8 bits RGB, DMAed as 32 bits, upper MSB set to 00h
12 (1100b)	3x8 bits RGB, DMAed as 24 bits (packed)
13 (1101b)	3x10 BGR, DMAed as 32 bits, display mode is 24 bits
14 (1110b)	3x12 BGR, DMAed as 48 bits (packed), display mode is 24 bits

FORCE_8BIT R/W, CON10[31], Karbon, Neon, R64

This bitfield has the following properties.

FORCE_8BIT	Meaning
0 (000b)	Normal operation
1 (001b)	Only 8 LSB of pixel will be acquired

Preliminary

4.14 CON11 Register

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

Preliminary

ALAST_ADD RO, CON11[14..0], Karbon, Neon, R64

Last address for lane A (used for diagnostics).

BLAST_ADD RO, CON11[30..16], Karbon, Neon, R64

Last address for lane B (used for diagnostics).

UART_MASTER R/W, CON11[31], Karbon - [New]

This bit controls which Karbon VFG is in control of the UART. Poke this bit to one in order to take control of the UART.

Preliminary

4.15 CON12 Register

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

Preliminary

CLAST_ADD RO, CON11[14..0], Karbon, Neon, R64

Last address for lane C (used for diagnostics).

DLAST_ADD R/W, CON11[30..16], Karbon, Neon, R64

Last address for lane D (used for diagnostics).

Preliminary

4.16 CON13 Register

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

Preliminary

VIDEO_MASK R/W, CON13[31..0], Karbon, Neon, R64

With the aid of this mask, individual bits in the video data stream can be set to 0. The 32 bit mask is duplicated for the 32 MSB of a 64 bit word.

Bit N in VIDEO_MASK	Meaning
0	Set bit N to 0
1	Pass bit N as is

Preliminary

4.17 CON14 Register

Bit	Name
0	SWRESET
1	FENPOL
2	LENPOL
3	BUTTONS
4	BUTTONS
5	BUTTONS
6	BUTTONS
7	BUTTONS
8	BUTTONS
9	BUTTONS
10	BUTTONS
11	BUTTONS
12	BUTTONS
13	BUTTONS
14	BUTTONS
15	BUTTONS
16	SHIFT_RAW
17	SHIFT_RAW
18	SHIFT_RAW
19	SHIFT_RAW
20	SHIFT_RAW_LEFT
21	DELAY
22	DELAY
23	DELAY
24	SWAP
25	UART_CON
26	UART_CON
27	Reserved
28	DPM_SPLIT
29	DPM_SPLIT
30	DPM_SPLIT
31	DPM_SPLIT

Preliminary

SW_RESET WO, CON14[0], Karbon, Neon, R64

This bit will always read back 0.

SW_RESET	Meaning
0	Reset de-asserted
1	General reset to acquisition circuitry.

FENPOL R/W, CON14[1], Karbon, Neon, R64

This bitfield has the following properties.

FENPOL	Meaning
0	FEN is asserted on rising edge.
1	FEN is asserted on falling edge.

LENPOL R/W, CON14[2], Karbon, Neon, R64

This bitfield has the following properties.

LENPOL	Meaning
0	LEN is asserted on rising edge.
1	LEN is asserted on falling edge.

BUTTONS R/W, CON14[15..3], Karbon, Neon, R64

R/W register for test/diagnostics.

SHIFT_RAW R/W, CON14[19..16], Karbon, Neon, R64

This register defines for the barrel shifter the amount of shift for the data to be acquired

Preliminary

SHIFT_RAW_LEFT R/W, CON14[20], Karbon, Neon, R64

This register defines for the barrel shifter the shift direction for the data to be acquired.

SHIFT_RAW_LEFT	Meaning
0	Shift right
1	Shift left

DELAY R/W, CON14[23..21], Karbon, Neon, R64

This register is used for aligning the Horizontal Active Window (HAW) relative the Line ENable signal (LEN). If the first active pixel occurs simultaneously with the assertion of LEN, then no delay is needed. If the first active pixel occurs between 1 and 7 clocks later than LEN, the DELAY register must be programmed accordingly. The CTABs can be used to compensate for delays over 7 clocks. However, the granularity of the CTABs is 8 clocks, so both DELAY and the CTABS may have to be used in together to accommodate for some delays.

The net effect for a simple, one tap camera will be a shifting to the right of the displayed image. For multi-tap cameras the visual effect is more complex and depends on the taps architecture. All taps are delayed by the same amount.

The purpose of this bit field is to align the image presented by the different taps. This bit field has the opposite effect of the TRIM field in CON9.

DELAY	Meaning
0 (000b)	HAW is not delayed
1 (001b)	HAW is delayed by 1 clocks
2 (010b)	HAW is delayed by 2 clocks
3 (011b)	HAW is delayed by 3 clocks
4 (100b)	HAW is delayed by 4 clocks
5 (101b)	HAW is delayed by 5 clocks
6 (11ob)	HAW is delayed by 6 clocks
7 (111b)	HAW is delayed by 7 clocks

SWAP R/W, CON14[24], Karbon, Neon, R64

Future use.

Preliminary

UART_CON R/W, CON14[26..25], Karbon, Neon, R64

This register will control the connection of the serial ports of the two camera Link connectors.

UART_CON	Connection
0 (00b)	On-board UART connected to CL serial port of main connector
1 (01b)	On-board UART connected to CL serial port of auxiliary connector
2 (10b)	Serial port of main connector CL connected to external serial port, through the IO connector.
3 (11b)	Serial port of auxiliary connector CL connected to external serial port, through the IO connector

DPM_SPLIT R/W, CON14[31..28], Karbon, Neon, R64

This register controls how incoming data is written to the DPM.

DPM_SPLIT	Mode
0 (0000b)	Normal mode
1 (0001b)	Each tap's output is split in half.
2 (0010b) to 14 (1110b)	Reserved
15 (1111b)	Each tap's output is written in 4K chunks

Preliminary

4.18 CON17 Register

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

Preliminary

FREE_RUN_RATE R/W, CON17[15..0], Karbon, Neon, R64

This register defines the period of the on-board free running generator. The clock driving the generator is the on-board generated clock, whose frequency is programmed in CON0. Programming the FREE_RUN_RATE to 200 for example, will yield a signals whose period is 200 clocks.

FREE_RUN_HI R/W, CON17[31..16], Karbon, Neon, R64

This register defines the amount of time the free running generator is 1 during a period. Setting this register to 100, for example, will yield a signal that is 1 for 100 clocks, regardless of the period. Obviously, the number programmed in FREE_RUN_RATE must be larger than the number programmed in FREE_RUN_HI.

Preliminary

4.19 CON18 Register

Bit	Name
0	ALPF
1	ALPF
2	ALPF
3	ALPF
4	ALPF
5	ALPF
6	ALPF
7	ALPF
8	ALPF
9	ALPF
10	ALPF
11	ALPF
12	ALPF
13	ALPF
14	ALPF
15	ALPF
16	ALPF
17	TOP_REV
18	TOP_REV
19	TOP_REV
20	TOP_REV
21	TOP_REV
22	TOP_REV
23	TOP_REV
24	TOP_REV
25	TOP_REV
26	TOP_REV
27	TOP_REV
28	TOP_REV
29	TOP_REV
30	SIG_GEN_POL
31	SCALE_BY8K

Preliminary

ALPF R/W, CON18[16..0], Karbon, Neon, R64

This register defines the Active Lines Per Frame of the vertical acquisition window. Let's assume for example a single tap camera with 2K lines per frame. If we want to acquire 400 lines per frame, the ALPF will be programmed to 400. For a 2-tap odd-even lines camera, with 2K lines per frame, to acquire 400 pixels the ALPF will be programmed with the value 200, as for every LEN the camera supplies two lines.

TOP_REV RO, CON9[11..0], Karbon, Neon, R64

Firmware revision.

SIG_GEN_POL R/W, CON18[30], Karbon, Neon, R64

This bit allows for the inversion of the free running generator's output.

SIG_GEN_POL	Meaning
0	Free running generator supplies signal as is
1	Invert output of the free running signal generator.

SCALE_BY8K R/W, CON18[31], Karbon, Neon, R64

Scaling down the clock to the free running generator can yield waveforms on the order of single Hz.

SCALE_BY8K	Meaning
0	Use on board CLOCK as is for the free running signal generator
1	Divide by 8K the on-board CLOCK to be used by the free running generator..

Preliminary

4.20 CON19 Register

Bit	Name
0	LINES_TOGO
1	LINES_TOGO
2	LINES_TOGO
3	LINES_TOGO
4	LINES_TOGO
5	LINES_TOGO
6	LINES_TOGO
7	LINES_TOGO
8	LINES_TOGO
9	LINES_TOGO
10	LINES_TOGO
11	LINES_TOGO
12	LINES_TOGO
13	LINES_TOGO
14	LINES_TOGO
15	LINES_TOGO
16	LINES_TOGO
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

Preliminary

LINES_TOGO R/W, CON19[16..0], Karbon, Neon, R64

This register will reflect the number of remaining lines left to be acquired till the end of the frame.

Preliminary

4.21 CON20 Register

Bit	Name
0	FIFO_EQ
1	FIFO_EQ
2	FIFO_EQ
3	FIFO_EQ
4	FIFO_EQ
5	FIFO_EQ
6	FIFO_EQ
7	FIFO_EQ
8	VID_BRL
9	VID_BRL
10	VID_BRL
11	VID_BRL
12	VID_BRL
13	VID_BRL
14	VID_BRL
15	VID_BRL
16	VIDEO_2DPM
17	VIDEO_2DPM
18	VIDEO_2DPM
19	VIDEO_2DPM
20	VIDEO_2DPM
21	VIDEO_2DPM
22	VIDEO_2DPM
23	VIDEO_2DPM
24	COLOR_MASK
25	COLOR_MASK
26	SHIFT_DSP_SELECT
27	SHIFT_DSP
28	SHIFT_DSP
29	SHIFT_DSP
30	SHIFT_DSP
31	SHIFT_DSP_LEFT

Preliminary

FIFO_EQS RO, CON20[7..0], Karbon, Neon, R64

This register reflects the instantaneous value of the 8 LSB of the first tap of the main CL connector. Used for diagnostics/test.

VID_BRL RO, CON20[15..8], Karbon, Neon, R64

This register reflects the instantaneous value of the 8 LSB of the barrel shifter of lane A. Used for diagnostics/test.

VIDEO_2DPM RO, CON20[23..16], Karbon, Neon, R64

This register reflects the instantaneous value of the 8 LSB of the video written in the DPM's lane A. Used for diagnostics/test.

COLOR_MASK R/W, CON20[25..24], Karbon, Neon, R64

This bitfield can be used to mask out color channels when acquiring color pixels formats (e.g 24-bit color, 36-bit color, etc.).

COLOR_MASK	Meaning
0 (00b)	Pass all colors
1 (01b)	Pass only the red channel, set the blue and green channels to 0
2 (10b)	Pass only the green channel, set the blue and red channels to 0
3 (11b)	Pass only the blue channel, set the red and green channels to 0

SHIFT_DSP_SELECT R/W, CON20[26], Karbon, Neon, R64

This bitfield has the following properties.

SHIFT_DSP_SELECT	Meaning
0	Supply barrel shifter with the acquisition shift code
1	Supply barrel shifter with the display shift code.

SHIFT_DISP R/W, CON20[30..27], Karbon, Neon, R64

This register holds the shift amount for data to be displayed.

Preliminary

SHIFT_DSP_LEFT R/W, CON20[31], Karbon, Neon, R64

This bitfield has the following properties.

SHIFT_DSP_LEFT	Meaning
0	Shift display data right
1	Shift display data left.

Preliminary

4.22 CON21 Register (Bayer Version)

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

CON21 is a “soft” register. Soft registers change definitions depending on the version board and the firmware that is downloaded to the board.

Preliminary

REG_GAIN R/W, CON21[7..0], Karbon, R64

This register controls the gain of the red channel. The video value is multiplied by the value in RED_GAIN and after that scaled down by 64. Numbers above 255 are clipped to 255 (saturation effect).

GREEN_GAIN R/W, CON21[15..8], Karbon, R64

This register controls the gain of the green channel. The video value is multiplied by the value in GREEN_GAIN and after that scaled down by 64. Numbers above 255 are clipped to 255 (saturation effect).

BLUE_GAIN R/W, CON21[23..16], Karbon, R64

This register controls the gain of the blue channel. The video value is multiplied by the value in BLUE_GAIN and after that scaled down by 64. Numbers above 255 are clipped to 255 (saturation effect).

DECODER_OUT R/W, CON21[26..24], Karbon, R64

These bits controls the output of the Bayer decoder.

DECODER_OUT	Meaning
0 (000b)	Decode RGB on all three channels
1 (001b)	Raw data on all three channels
2 (010b)	Decode intensity on all three channels
3 (011b)	Decode red on all three channels
4 (100b)	Decode green on all three channels
5 (101b)	Decode blue on all three channels
6 (110b)	Reserved
7 (111b)	Reserved

Preliminary

BAYER_BIT_DEPTH

R/W, CON21[29..28], Karbon, R64

This bit is set if the pixel depth from the camera is 10 bits.

BAYER_10_BIT	Meaning
0 (00b)	8-bit pixels
1 (01b)	12-bit pixels
2 (10b)	10-bit pixels
3 (11b)	Reserved

DECODER_PHASE

R/W, CON21[31..30], Karbon, R64

These bits control the starting phase of the Bayer decoder. This register is set based on the arragement of the color matrix in the camera's CCD.

DECODER_PHASE	Meaning
0 (00b)	First two pixels: Blue, Green
1 (01b)	First two pixels: Green, Blue
2 (10b)	First two pixels: Red, Green
3 (11b)	First two pixels: Green, Red

Preliminary

4.23 CON23 Register

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

Preliminary

DPM_SIZE RO, CON23[7..0], Karbon, Neon, R64 - [New]

This register specifies the size of the DPM in units of 4096 bytes.

CTAB_INT_CON RW, CON23[15], Karbon, Neon - [New]

This bit controls the source of the CTAB interrupt.

CTAB_INT_CON	Meaning
0	CTAB interrupt sourced from CTAB
1	CTAB interrupt sourced from lines per int circuit.

LINES_PER_INT RW, CON23[23..16], Karbon, Neon - [New]

This bit controls the lines per interrup circuit. This circuit can be used to create a periodic interrupt on the CTAB interrupt. The interrupt rate will be every N lines, where N is the value programmed in this register. Note that CTAB_INT_CON must be set to one in order for the interrupts to be seen by the host.

Preliminary

4.24 CON24 Register

Bit	Name
0	LUT_HOST_DATA
1	LUT_HOST_DATA
2	LUT_HOST_DATA
3	LUT_HOST_DATA
4	LUT_HOST_DATA
5	LUT_HOST_DATA
6	LUT_HOST_DATA
7	LUT_HOST_DATA
8	Reserved
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	LUT_ON
16	LUT_HOST_ADDR
17	LUT_HOST_ADDR
18	LUT_HOST_ADDR
19	LUT_HOST_ADDR
20	LUT_HOST_ADDR
21	LUT_HOST_ADDR
22	LUT_HOST_ADDR
23	LUT_HOST_ADDR
24	LUT_BANK
25	LUT_BANK
26	Reserved
27	LUT_DATA_WRITE_SEL
28	LUT_HOST_LANE
29	LUT_HOST_LANE
30	LUT_WEN
31	LUT_HOST_ACCESS

CON24 is a "soft" register. Soft registers change definitions depending on the version board and the firmware that is downloaded to the board.

Preliminary

LUT_HOST_DATA R/W, CON24[7..0], Karbon, R64

This register is used to read and write data from the LUT. The LUT is programmed indrectly using this and the other registers in CON24.

The procedure to write data to the LUT is as follows:

Set LUT_HOST_ACCESS to 1.
Set LUT_DATA_WRITE_SEL to 1.
Set LUT_BANK to the desired bank to program.
Set LUT_HOST_LANE to the desired LUT lane to program.
Set LUT_HOST_ADDR to the desired LUT location to program (LUT input).
Set LUT_HOST_DATA to the value desired (LUT output).
Write LUT_WEN to 1, this copies the value from the LUT_HOST_DATA register to the LUT's memory location as specified by LUT_HOST_ADDR.

The procedure to read data from the LUT is as follows:

Set LUT_HOST_ACCESS to 1.
Set LUT_DATA_WRITE_SEL to 0.
Set LUT_BANK to the desired bank to read.
Set LUT_HOST_LANE to the desired LUT lane to read.
Set LUT_HOST_ADDR to the desired LUT location to read.
Read LUT_HOST_DATA, the value returned in this register is the value in the LUT's memory.

LUT_ON R/W, CON24[15], Karbon, R64

This register is used to insert or bypass the LUT from the path of incoming camera data.

LUT_ON	Meaning
0	LUT is not in the data path (bypassed)
1	LUT is in the data path

LUT_HOST_ADDR R/W CON24[23..16], Karbon, R64

This register is used to set the address for a read operation from the LUT memory or a write operation to the LUT memory. See the description of LUT_HOST_DATA for more details.

Preliminary

LUT_BANK R/W, CON24[25..24], Karbon, R64

These bits control which bank of the LUT is being programmed, and which bank is being used to pass image data.

LUT_BANK	Meaning
0 (000b)	Host access to bank 0, data passes through bank 0
1 (001b)	Host access to bank 1, data passes through bank 1
2 (010b)	Host access to bank 2, data passes through bank 2
3 (011b)	Host access to bank 3, data passes through bank 3

LUT_DATA_WRITE_SEL R/W, CON24[27], Karbon, R64

This bit is used to control how the LUT data is being accessed.

LUT_DATA_WRITE_SEL	Meaning
0	When reading LUT_HOST_DATA, the register returns the value currently in the LUT. In this mode LUT_HOST_DATA is read only.
1	LUT_HOST_DATA acts like a normal register. It can be written and read normally. The value in the LUT_HOST_DATA register is not transfered to the LUT memory until a 1 is written to LUT_WEN.

LUT_HOST_LANE R/W, CON24[29..28], Karbon, R64

These bits control which LUT lane can be access by the host.

LUT_HOST_LANE	Meaning
0 (000b)	Host access to lane 0
1 (001b)	Host access to lane 1
2 (010b)	Host access to lane 2
3 (011b)	Host access to lane 3

LUT_WEN WO, CON24[30], Karbon, R64

When LUT_DATA_WRITE_SEL is set to 1, Writing a 1 to this bit casues the value in LUT_HOST_DATA to be transfered to the LUT memory. When LUT_DATA_WRITE_SEL is set to 0, writing to this bit has no effect. See LUT_HOST_DATA for more information.

Preliminary

LUT_HOST_ ACCESS

R/W, CON24[31], Karbon, R64

These bits turns on and off host access to the LUT..

DECODER_OUT	Meaning
0	The LUT cannot be accessed by the host
1	The LUT can be accessed by the host

Preliminary

4.25 CON25 Register

Bit	Name
0	DELAY_TAP1
1	DELAY_TAP1
2	DELAY_TAP1
3	DELAY_TAP1_SEL
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

Preliminary

DELAY_TAP1 R/W, CON25[2..0], Karbon, Neon, R64

These bits control the delay for tap 1 only when DELAY_TAP1_SEL = 1. In this mode, tap 0 and tap 1 can be delayed independently. These bits have no effect if DELAY_ TAP1_SEL = 0. This register works in a similar manner to the DELAY register.

DELAY_TAP1	Meaning
0 (000b)	HAW is not delayed
1 (001b)	HAW is delayed by 1 clocks
2 (010b)	HAW is delayed by 2 clocks
3 (011b)	HAW is delayed by 3 clocks
4 (100b)	HAW is delayed by 4 clocks
5 (101b)	HAW is delayed by 5 clocks
6 (11ob)	HAW is delayed by 6 clocks
7 (111b)	HAW is delayed by 7 clocks

DELAY_TAP1_ SEL R/W, CON25[3], Karbon, Neon, R64

This bit selects the register that controls the delay for tap 1. Tap 0 is always controlled by the register DELAY.

DELAY_TAP1_SEL	Meaning
0	Tap 1 is controlled by DELAY
1	Tap 1 is controlled by DELAY_TAP1

Preliminary

4.26 CON26 Register

Bit	Name
0	Reserved
1	Reserved
2	Reserved
3	Reserved
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

Preliminary

4.27 CON27 Register

Bit	Name
0	Reserved
1	Reserved
2	Reserved
3	Reserved
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

Preliminary

4.28 CON36 Register

Bit	Name
0	MEM_ADDR_LO
1	MEM_ADDR_LO
2	MEM_ADDR_LO
3	MEM_ADDR_LO
4	MEM_ADDR_LO
5	MEM_ADDR_LO
6	MEM_ADDR_LO
7	MEM_ADDR_LO
8	MEM_ADDR_LO
9	MEM_ADDR_LO
10	MEM_ADDR_LO
11	MEM_ADDR_LO
12	MEM_ADDR_LO
13	MEM_ADDR_LO
14	MEM_ADDR_LO
15	MEM_ADDR_LO
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

Preliminary

MEM_ADDR_LO R/W, CON25[15..0], Neon - [New]

This register is the lower 16 bits used to access the flash or ROM memory on boards that have it. This is not a user programmable register.

Preliminary

4.29 CON37 Register

Bit	Name
0	MEM_ADDR_HI
1	MEM_ADDR_HI
2	MEM_ADDR_HI
3	MEM_ADDR_HI
4	MEM_CS
5	MEM_WRITE
6	DWNLD_MODE
7	DWNLD_MODE
8	MEM_DATA
9	MEM_DATA
10	MEM_DATA
11	MEM_DATA
12	MEM_DATA
13	MEM_DATA
14	MEM_DATA
15	MEM_DATA
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

Preliminary

MEM_ADDR_HI R/W, CON37[3..0], Neon - [New]

This register is the upper 4 bits used to access the flash or ROM memory on boards that have it. This is not a user programmable register.

MEM_CS R/W, CON37[4], Neon - [New]

This bit is the chip select which controls both reading and writing to either the flash or the ROM. This bit controls both host access and FPGA download source.This is not a user programmable register.

MEM_CS	Meaning
0	Host and FPGA access is to/from the the ROM
1	Host and FPGA access is to/from the flash

MEM_WRITE R/W, CON37[5], Neon - [New]

???

DWNLD_MODE R/W, CON37[7..6], Neon - [New]

???

MEM_DATA R/W, CON37[15..8], Neon - [New]

This bitfield provides data access used when reading or writting the flash or ROM on boards that support these features. This is not a user programmable register.

Preliminary

4.30 CON38 Register

Bit	Name
0	POCL_POWER_ON
1	POCL_EN_GND
2	POCL_CLOCK_WAIT
3	Reserved
4	POCL_SENSE
5	POCL_CLK_ DETECTED
6	POCL_DETECTED
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

Preliminary

POCL_POWER_ON RO, CON38[0], Neon - [New]

This register indicates the state of the power on the Camera Link connector.

POCL_EN_POWER	Meaning
0	Power is not applied to the power wires on the CL cable.
1	Power is applied to the power wires on the CL cable.

POCL_GND_ON RO, CON38[1], Neon - [New]

This register indicates the state of the ground on the Camera Link connector.

POCL_GND_ON	Meaning
0	The power wires on the CL cable are not grounded.
1	The power wires on the CL cable are grounded.

POCL_CLOCK_WAIT RO, CON38[2], Neon - [New]

This register indicates that the PoCL state machine is the “waiting for clock” state. In this state, the power has been applied to the camera, but the camera may be powered up yet and may not be output a pixel clock. The PoCL state machine stays in the state for a few seconds, Once it leaves this state, the PoCL state machine will immediately remove the power if it sense that the pixel clock has stopped.

POCL_CLOCK_WAIT	Meaning
0	The PoCL state machine is not in the waiting for clock state.
1	The PoCL state machine is waiting for the pixel from the camera.

POCL_SENSE RO, CON38[3], Neon - [New]

This register indicates that the PoCL state machine is the “sense” state. In this state, the powers has not been applied, and the PoCL state machine is watching the impendence on the CL cable. If the impendence of a PoCL camera is detected, the power will be applied. It a short is detected, indicating a legacy camera/cable has been con-

Preliminary

nected, the PoCL power lines will be grounded. The PoCL state machine can stay in this state indefinitely if neither of the above two conditions are detected (i.e. nothing is connected).

POCL_SENSE	Meaning
0	The PoCL state machine is not in the sense state.
1	The PoCL state machine is in the sense state.

POCL_CLK_ DETECTED

RO, CON38[5], Neon - [New]

This register indicates that the PoCL state machine is the "PoCL camera clock has been detected" state. This state is the normal powered up steady state for the PoCL state machine.

POCL_CLK_DETECTED	Meaning
0	The PoCL state machine has not detected a camera pixel clock.
1	The PoCL state machine has detected a camera pixel clock.

POCL_ DETECTED

RO, CON38[6], Neon - [New]

This register indicates that the PoCL state machine has detected a PoCL camera..

POCL_DETECTED	Meaning
0	The PoCL state machine has not detected a PoCL-camera.
1	The PoCL state machine has detected a PoCL camera.

Preliminary

Karbon/Neon DMA Registers

Chapter 5

5.1 Introduction

This section enumerates all of the registers that control DMA on boards using the PLDA DMA engine (e.g. Karbon-CL). The formatting is explained in Section 4.2.

Preliminary

5.2 CON28 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

Preliminary

FIRST_QUAD_PTR_LO R/W, CON28[31..0], Karbon, Neon

This is the low word of the 64-bit address of the first DMA scatter-gather instruction in a chain of instructions. This register can be written at any time, but the DMA engine only loads this value when byte count (as set by CHAIN_DATA_SIZE) goes to zero.

Preliminary

5.3 CON29 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

Preliminary

FIRST_QUAD_PTR_HI R/W, CON29[31..0], Karbon, Neon

This is the high word of the 64-bit address of the first DMA scatter-gather instruction in a chain of instructions. This register can be written at any time, but the DMA engine only loads this value when byte count (as set by CHAIN_DATA_SIZE) goes to zero.

Preliminary

5.4 CON30 Register

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

Preliminary

CHAIN_DATA_SIZE R/W, CON30[31..0], Karbon, Neon

This is the total amount of data in the chain in bytes. The value in this register is loaded into the DMA engine when DMA is initiated. This value is then decremented every DMA transfer. When the count reached zero, this value in this register is reloaded into the DMA engine, and the first scatter gather instruction pointed to by FIRST_QUAD_PTR_HI and FIRST_QUAD_PTR_LO is loaded.

Preliminary

5.5 CON31 Register

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

Preliminary

CHAIN_DATA_TOGO RO, CON31[31..0], Karbon, Neon

This register indicates the number of bytes remaining the DMA chain.

Preliminary

5.6 CON32 Register

Bit	Name
0	DMA_AUTO_START
1	DMA_ABORT
2	DMA_DIRECTION
3	DMA_DONE
4	DMA_STATUS
5	DMA_STATUS
6	DMA_STATUS
7	DMA_STATUS
8	DMA_NO_RULE
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved
16	DMA_INIT_FUNC
17	DMA_PRIORITY
18	DMA_64_BIT
19	DMA_CHAINING
20	DMA_COMMAND
21	DMA_COMMAND
22	DMA_COMMAND
23	DMA_COMMAND
24	DMA_BEN
25	DMA_BEN
26	DMA_BEN
27	DMA_BEN
28	LATCH_CONTROL
29	LATCH_CONTROL
30	Reserved
31	Reserved

Preliminary

DMA_AUTO_START R/W, CON32[0], Karbon, Neon

This bit controls how the DMA engine starts.

DMA_AUTO_START	Meaning
0	Do nothing
1	Reload and re-start when CHAIN_DATA_TOGO = 0

DMA_ABORT RO, CON32[1], Karbon, Neon

This bit immediately aborts DMA. Always reads back 0.

DMA_DIRECTION R/W, CON32[3], Karbon, Neon

This bit indicates the direction that DMA engine will move data.

DMA_DIRECTION	Meaning
0	DMA write (to host memory)
1	DMA read (from host memory)

DMA_DONE RO, CON32[3], Karbon, Neon

This indicates the status of the DMA engine.

DMA_DONE	Meaning
0	???
1	???

DMA_STATUS RO, CON32[7..4], Karbon, Neon

Current status of the DMA engine. [Do we want to add a table???]

DMA_NO_RULE R/W, CON32[8], Karbon, Neon

Setting this bit to a 1 will cause the DMA engine to DMA data as fast as it can. It will not wait for data to be available from the acquisition engine. The actual data that is DMAed will be unpredictable. This bit, therefore, is only useful for diagnostics.

Preliminary

DMA_INIT_FUNC R/W, CON32[16], Karbon, Neon

???

DMA_PRIORITY R/W, CON32[17], Karbon, Neon

???

DMA_64_BIT R/W, CON3[18], Karbon, Neon

Controls where the DMA operations are 64-bit or 32-bit.

DMA_64_BIT	Meaning
0	???
1	???

DMA_CHAINING RW, CON32[19], Karbon, Neon

This bit determines whether the DMA engine will execute chaining DMA or not.

DMA_CHAINING	Meaning
0	???
1	???

DMA_COMMAND R/W, CON32[23..20], Karbon, Neon

???

DMA_BEN R/W, CON32[27..24], Karbon, Neon

???

LATCH_CONTROL R/W, CON32[29..28], Karbon, Neon

???

Preliminary

5.7 CON33 Register

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

Preliminary

XFR_PER_INT R/W, CON33[31..0], Karbon, Neon

This register controls how often the board issues an EOF interupt. Every time XFR_PER_INT bytes have been DMAed, the board will emit an interrupt.

Preliminary

Register and Memory Mapping

Chapter 6

6.1 Introduction

This section explains how the registers and the various chunks of memory are mapped and accessed on the Karbon-CL and its virtual frame grabbers.

Preliminary

6.2 Memory Types

6.2.1 Registers

?? Update. All registers are 64 bits wide and on 64-bit boundary. With the exception of the DPM, only data bits 31 to 0 are used, bits 63 to 32 are always “don't care”. The DPM uses all 64 bits. Out of the lower 32 LSBs, some registers use only a portion of the bits. Registers can also be accessed as 32 bit wide. Little endian addressing is used, i.e. MSB is bits 63-56.

For structure of the QL5064 register see QL documentation.

Note: There is no access to the QL5064 from the local bus

6.2.2 UART

The UART is 8 bit wide and its registers are on 64-bit boundary.

6.2.3 DPM

The DPM is 64 bit wide. The DPM can be accessed as 64-bit (on 64-bit boundary) or as 32-bit wide (on 32-bit boundary) memory. During acquisition (GRAB/SNAP), the slave read from DPM is inhibited. In this case, data read will be always zero.

6.2.4 TAG

At the start and the end of every frame there is an 8-byte long TAG quad. The source address of the TAG is irrelevant as there is no data read. The destination is a special trash can section of memory. The lower 4 bytes transferred are 13131313h. The upper four bytes are undefined.

6.2.5 CTABs

For host access, the CTABs are on 64-bit boundaries. Access is simultaneous to both the HCTAB and the VCTAB. The HCTAB is mapped on data bits 7 to 0; the VCTAB is mapped on data bits 15 to 8. The VCTAB address is 17 bits, AD, i.e. 128K locations. The HCTAB address is 15 bits, AD. The CTAB can be accessed only as 64 bit wide, i.e. there is no provision for accessing less than 64 bits.

Preliminary

6.3 Memory Map

?? Update The following table illustrates the physical location of the various sections of memory on the Karbon.

Memory	Bytes used	Address (hex)	Comment
CON0	nnnnnnyy	00 80 00 00	download, clock control
CON1	nnnnyyyy	00 00 00 00	
CON2	nnnnyyyy	00 02 00 00	
CON3	nnnnyyyy	00 04 00 00	
CON4	nnnnyyyy	00 06 00 00	
CON5	nnnnyyyy	00 08 00 00	
CON6	nnnnyyyy	00 0A 00 00	
CON7	nnnnyyyy	00 0C 00 00	
CON8	nnnnyyyy	00 0E 00 00	
CON9	nnnnyyyy	00 10 00 00	
CON10	nnnnyyyy	00 10 00 08	
CON11	nnnnyyyy	00 10 00 10	
CON12	nnnnyyyy	00 10 00 18	
CON13	nnnnyyyy	00 10 00 20	
CON14	nnnnyyyy	00 10 00 28	
CON15	nnnnyyyy	00 00 00 08	
CON16	nnnnyyyy	00 00 00 10	
CON17	nnnnyyyy	00 00 00 18	
CON18	nnnnyyyy	00 00 00 20	
CON19	nnnnyyyy	00 00 00 28	
CON20	nnnnyyyy	00 10 00 30	
CON21	nnnnyyyy	00 10 00 38	Board version dependent
CON22	nnnnyyyy	00 00 00 30	
CON23	nnnnyyyy	00 00 00 38	Board version dependent
CON24	nnnnyyyy	00 10 00 40	Board version dependent
CON25	nnnnyyyy	00 10 00 48	
CON26	nnnnyyyy	00 00 00 40	
CON27	nnnnyyyy	00 00 00 48	
CON28	?	?	
CON29	?	?	
CON30	?	?	
CON31	?	?	
CON32	?	?	
CTABS	?	?	ADD select 128K addresses
DPM	?	?	ADD select 16K addresses for a 128K DPM
UART	?	?	8 internal 8-bit registers on 64 bit boundary
RO_INFOHI	?	?	R/O info, model/rev, etc.
RO_INFOLO	?	?	R/O info

Preliminary

The following pertains to the table above.

Bytes used in the 64-bit word: n - not used, y - used

All registers are treated as 64 bit wide. The Karbon uses two BARs for PCI access:

??? BAR0, memory mapped, 1M size, is used for access to QL5064 internal registers.

BAR1, memory mapped, 16M size, is used for access to R64 registers, CTABs, DPM.

The Karbon uses address lines 23 to 3. Pre-fetch and posting WRITEs are disabled for both BARs

Preliminary

6.4 Downloading Firmware

Update???

Firmware is downloaded using the tiny interface. The selection between the 'Tiny' interface (for download) and the full interface will be done on-the-fly according to the destination address.

The mechanics of the download (assertion of CLOCK, DATA, CONFIG etc..) is beyond the scope of this manual.

There are only two FPGAs to download: MUX and DCC. The MUX is downloaded first. Different types of firmware types can be downloaded for different applications. The board has been designed to handle different size gate arrays if needed.

Preliminary

6.5 PCI Configuration Space and Model/Revision Information

All Karbon boards will have the same device ID. Information about different models and board capabilities will be stored in the INFO_HI and INFO_LO registers.

Preliminary

6.6 Scatter Gather DMA Instructions

The DMA engine is run by Scatter Gather DMA instructions. These called “quads” because they generally consist of four words, although the Karbon’s DMA engine only uses three words.. A list of instructions are called a Quad Table or QTAB. Each quad consists of the following entries.

1. Destination address
2. Size of transfer
3. Next quad address.

The following sections document the structure of these quads.

Preliminary

6.7 Destination Address

Bit	Name	Bit	Name
0	Destination Address	32	Destination Address
1	Destination Address	33	Destination Address
2	Destination Address	34	Destination Address
3	Destination Address	35	Destination Address
4	Destination Address	36	Destination Address
5	Destination Address	37	Destination Address
6	Destination Address	38	Destination Address
7	Destination Address	39	Destination Address
8	Destination Address	40	Destination Address
9	Destination Address	41	Destination Address
10	Destination Address	42	Destination Address
11	Destination Address	43	Destination Address
12	Destination Address	44	Destination Address
13	Destination Address	45	Destination Address
14	Destination Address	46	Destination Address
15	Destination Address	47	Destination Address
16	Destination Address	48	Destination Address
17	Destination Address	49	Destination Address
18	Destination Address	50	Destination Address
19	Destination Address	51	Destination Address
20	Destination Address	52	Destination Address
21	Destination Address	53	Destination Address
22	Destination Address	54	Destination Address
23	Destination Address	55	Destination Address
24	Destination Address	56	Destination Address
25	Destination Address	57	Destination Address
26	Destination Address	58	Destination Address
27	Destination Address	59	Destination Address
28	Destination Address	60	Destination Address
29	Destination Address	61	Destination Address
30	Destination Address	62	Destination Address
31	Destination Address	63	Destination Address

Preliminary

6.8 Size of Transfer

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

Preliminary

6.9 Next Quad Address

Bit	Name	Bit	Name
0	Next Quad Address	32	Next Quad Address
1	Next Quad Address	33	Next Quad Address
2	Next Quad Address	34	Next Quad Address
3	Next Quad Address	35	Next Quad Address
4	Next Quad Address	36	Next Quad Address
5	Next Quad Address	37	Next Quad Address
6	Next Quad Address	38	Next Quad Address
7	Next Quad Address	39	Next Quad Address
8	Next Quad Address	40	Next Quad Address
9	Next Quad Address	41	Next Quad Address
10	Next Quad Address	42	Next Quad Address
11	Next Quad Address	43	Next Quad Address
12	Next Quad Address	44	Next Quad Address
13	Next Quad Address	45	Next Quad Address
14	Next Quad Address	46	Next Quad Address
15	Next Quad Address	47	Next Quad Address
16	Next Quad Address	48	Next Quad Address
17	Next Quad Address	49	Next Quad Address
18	Next Quad Address	50	Next Quad Address
19	Next Quad Address	51	Next Quad Address
20	Next Quad Address	52	Next Quad Address
21	Next Quad Address	53	Next Quad Address
22	Next Quad Address	54	Next Quad Address
23	Next Quad Address	55	Next Quad Address
24	Next Quad Address	56	Next Quad Address
25	Next Quad Address	57	Next Quad Address
26	Next Quad Address	58	Next Quad Address
27	Next Quad Address	59	Next Quad Address
28	Next Quad Address	60	Next Quad Address
29	Next Quad Address	61	Next Quad Address
30	Next Quad Address	62	Next Quad Address
31	Next Quad Address	63	Next Quad Address

Preliminary

Electrical Interfacing

Chapter 7

7.1 Introduction

This chapter describes the electrical interface of the Karbon/Neon/R64. This includes detailed information on the all if the input and output signals. In addition, information is provided on recommend circuits to use when connecting to these signals.

Preliminary

7.2 Trigger

7.2.1 Trigger Input Types

There are four trigger inputs.

TRIGGER_TTL - Single ended TTL level trigger
TRIGGER_DIFF - Differential (LVDS) trigger
TRIGGER_OPTO - Optocoupled trigger
FEN - The FEN signal on the CL1 connector.

The hardware trigger is enabled/disabled by the bit EN_TRIGGER. Only one input at a time is active (the software trigger bit, SW_TRIG, is always active). The active trigger is selected by the bitfield SEL_TRIG. The current level of these inputs can be read from software by reading the bits RD_TRIG_DIFF, RD_TRIG_TTL, RD_TRIG_OPTO and RD_FEN. The unselected triggers will have no effect on the board. However, they can be used as general purpose inputs.

7.2.2 The Optocoupled Trigger

The opto-coupled trigger allows the acquisition circuitry to accept a trigger signal without having a galvanic connection to the trigger source. This is mandatory in some medical and industrial application. The trigger information is passed as a light pulse from an on-board LED that is coupled to a receiving phototransistor. The LED and the phototransistor are in the same package, a Sharp PC3H711. Figure 7-1 shows the electrical diagram of the optocoupled trigger circuit and a suggested circuit for the driver. The LED is driven by an open collector driver. The user must supply his +5V power to the LED. Note that there is no galvanic connection between the user's circuit and the acquisition circuitry.

Preliminary

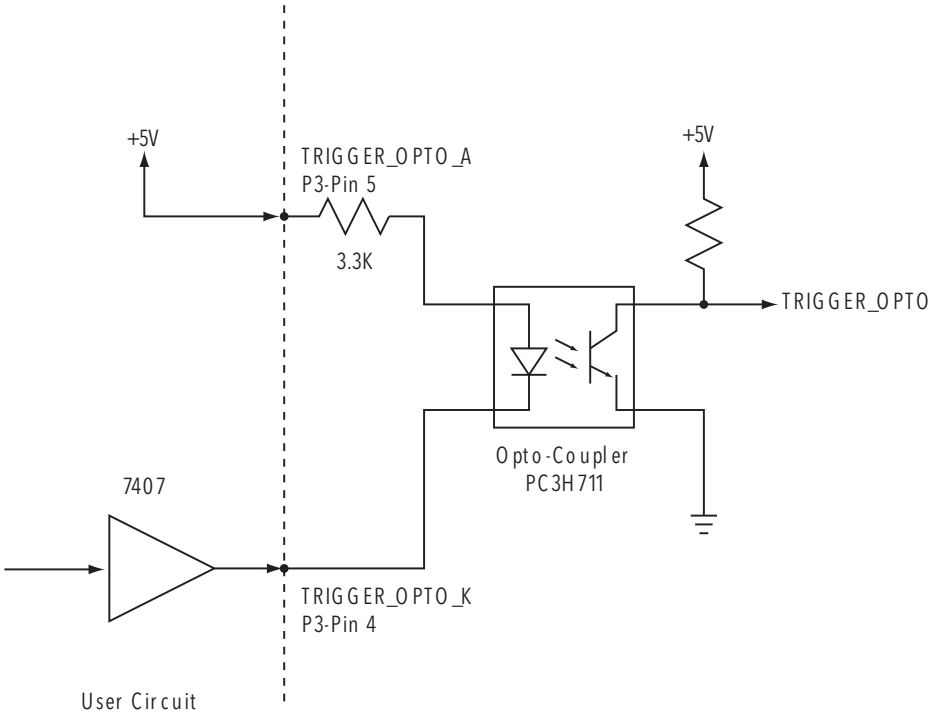


Figure 7-1 Driver Circuit for Opto-Coupled Trigger

Prelim

7.3 Encoder

7.3.1 Encoder Input Types

There are three encoder inputs.

ENCODER_TTL - Single ended TTL level encoder
ENCODER_DIFF - Differential (LVDS) encoder
ENCODER_OPTO - Optocoupled encoder

The hardware encoder is enabled/disabled by the bit EN_ENCODER. Only one input at a time is active (the software encoder bit, SW_ENC, is always active). The active encoder is selected by the bitfield SEL_ENC. The current level of these inputs can be read from software by reading the bits RD_ENC_DIFF, RD_ENC_TTL and RD_ENCOPTO. The unselected inputs will have no effect on the board. However, they can be used as general purpose inputs.

7.3.2 The Optocoupled Encoder

The opto-coupled encoder allows the acquisition circuitry to accept an encoder signal without having a galvanic connection to the encoder source. This is mandatory in some medical and industrial application. The encoder information is passed as a light pulse from an on-board LED that is coupled to a receiving phototransistor. The LED and the phototransistor are in the same package, a Sharp PC3H711. Figure 7-2 shows the electrical diagram of the optocoupled encoder circuit and a suggested circuit for the driver. The LED is driven by an open collector driver. The user must supply his +5V power to the LED. Note that there is no galvanic connection between the user's circuit and the acquisition circuitry.

Preliminary

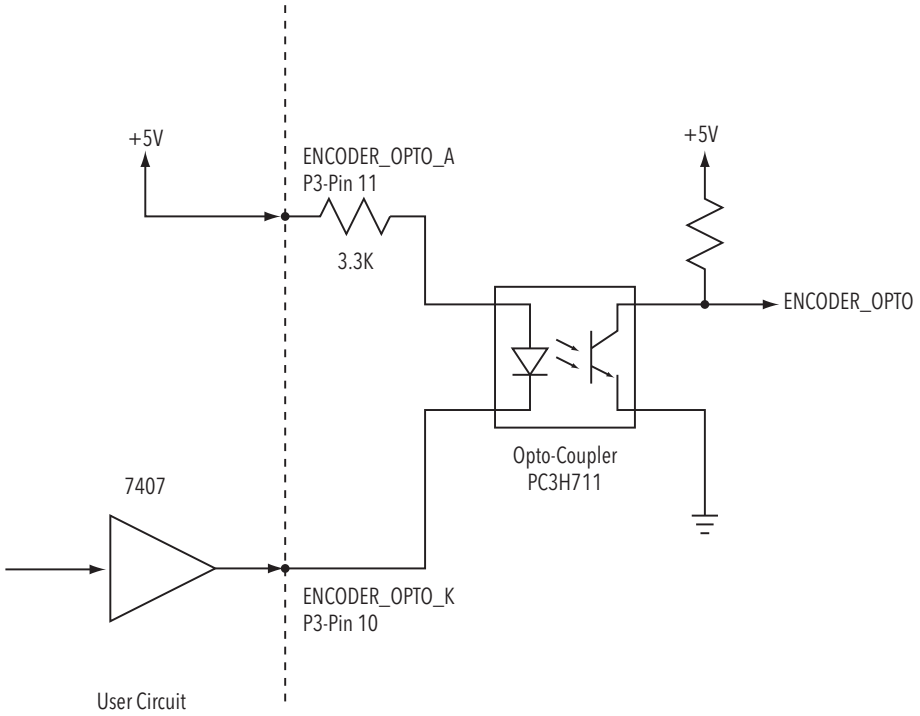


Figure 7-2 Driver Circuit for Opto-Coupled Encoder

Prelim

7.4 General Purpose Inputs (GPIN)

There are five general purpose inputs. The signal level on each input can be read on the corresponding GPIN bit. The electrical characteristic of these inputs is shown in the following list.

GPIN0, GPIN1 - Single ended TTL level inputs
GPIN2, GPIN3, GPIN4 - Differential (LVDS) inputs

Preliminary

7.5 General Purpose Outputs (GPOUT)

There are seven general purpose outputs. The signal level on each output is controlled by one of several sources (see below). The electrical characteristic of these outputs is shown in the following list.

GPOUT0, GPOUT1, GPOUT2 - Differential (LVDS) outputs
GPOUT3, GPOUT4 - Single ended TTL level outputs
GPOUT5 - Open collector
GPOUT6 - Open collector

The source for each GPOUT bit is controlled by the corresponding GPOUTx_CON bitfield. These are located in CON8. The source for each GPOUT can be programmed independently of the others. Table 7-1 shows the sources for each GPOUT.

Table 7-1 GPOUTx_CON

GPOUTx_CON	GPOUTx Source
0 (000b)	GPOUTx bit
1 (001b)	CT3 from CTAB
2 (010b)	CT2 from CTAB
3 (011b)	CT1 from CTAB
4 (100b)	CT0 from CTAB
5 (101b)	Internally generated clock. Frequency set by CFREQ.
6 (110b)	Free running on board signal generator. Controlled by FREE_RUN_RATE and FREE_RUN_HIGH
7 (111b)	Reserved

Preliminary

7.5.1 The Open Collector Drivers for GPOUT5 and GPOUT6

GPOUT5 and GPOUT6 are open collector drivers. Figure 7-3 shows the GPOUT5 and GPOUT6 driver circuits and their associated jumpers for various configurations. ???[how to handle difference between boards?] The board is shipped in with JP2 in position 1-2, JP4 in position 1-2, no jumpers in JP1 and JP3.

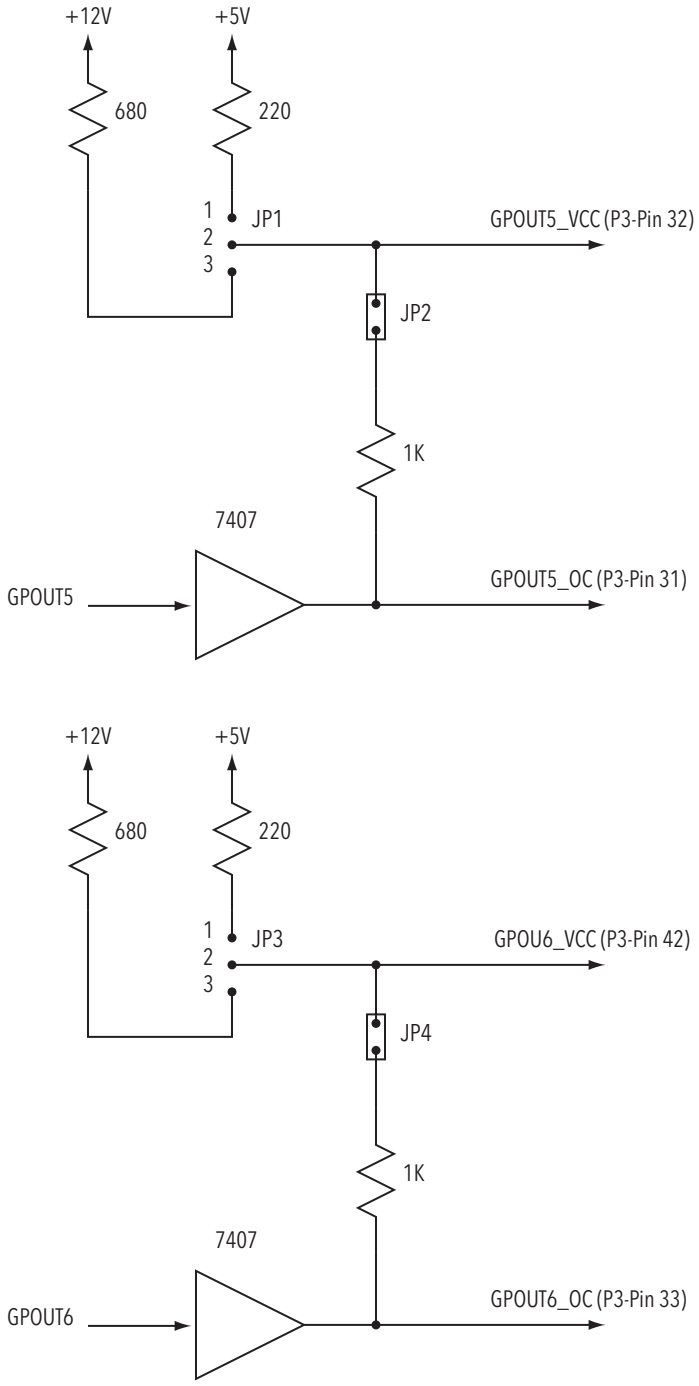


Figure 7-3 Driver Circuits for GPOUT5 and GPOUT6 (in Default Configuration)

Figure 7-4 shows how the open collector GPOUT5 in the factory configuration can drive an opto-coupling device. The user must supply the +5V to his LED and the two systems must have their grounds connected. In this configuration the board and the user's system must have a common electrical ground.

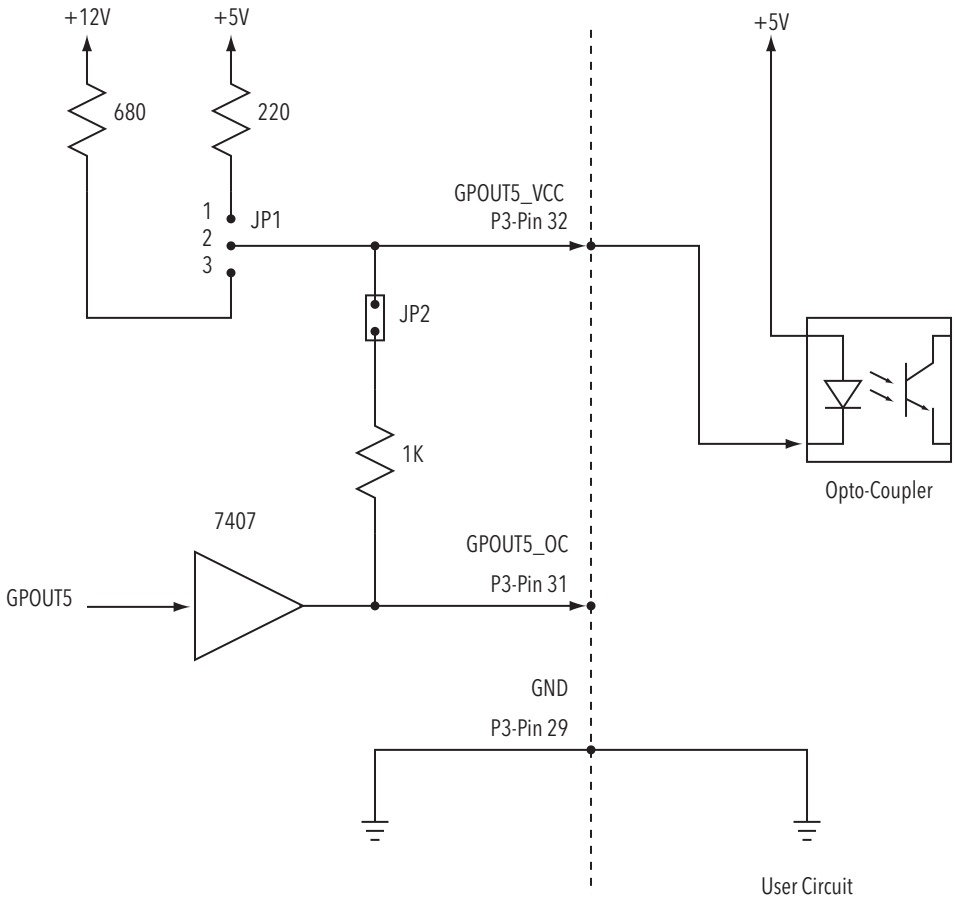


Figure 7-4 GPOUT5 Driving Opto-Coupled Circuit (in Default Configuration)

Figure 7-5 shows how the board's open collector GPOUT5 can drive an user's opto-coupled device configured for galvanic isolation between the board and the user. The power to the user's LED is supplied by the board's 5V through a 220 Ohm resistor. This is achieved by inserting the short in position 1-2 at JP1. The jumper at JP2 is removed. The open collector driver will sink the current from the LED. There is no galvanic connection between the board and the user's circuit. Information is passed from the board to the user as light, transmitted by the LED and received by the photo-transistor.

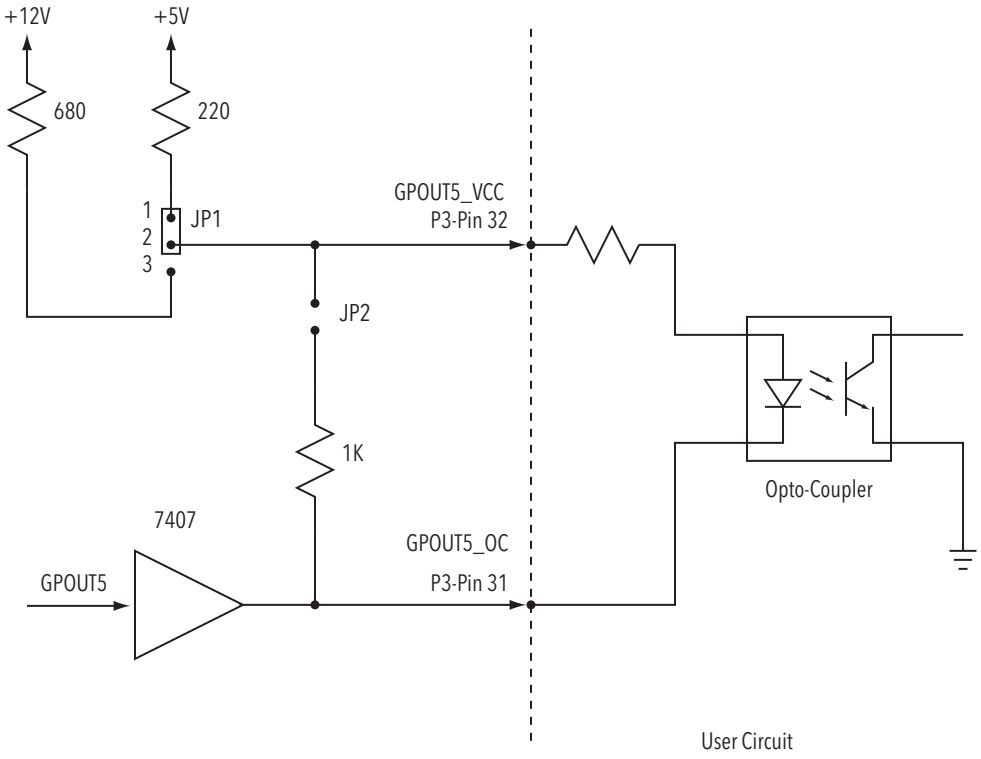


Figure 7-5 GPOUT5 Driving Opto-Coupled Circuit using Galvanic Isolation

Preli

7.6 Camera Link Controls (CCs)

The Camera Link cable caries four general purpose Camera Control (CC) signals that can be use to control the camera. These are labeled CC1, CC2, CC3, CC4. The source for each CC is controlled by the corresponding CCx_CON bitfield. Table 7-2 illustrates the source for each CCx as a function of its associated CCx_CON bitfield.

Table 7-2 CCx_CON

CCx_CON	CCx Source
0 (000b)	CT0 from CTAB
1 (001b)	CT1 from CTAB
2 (010b)	CT2 from CTAB
3 (011b)	Free running on board signal generator. Controlled by FREE_RUN_RATE and FREE_RUN_HIGH
4 (100b)	Internally generated clock. Frequency set by CFREQ.
5 (101b)	GPIN0's signal level
6 (110b)	Forced low
7 (111b)	Forced high

Preliminary

Preliminary

Specifications

Chapter 8

8.1 Introduction

This chapter describes the general specifications of the Neon family. The numerical values for he specifications are listed in Table 8-1. If more information is available for a given specification there will be an entry in the column marked “Details”.

Table 8-1 Neon Specifications

Specifications	Value	Units	Details
PCIe Compatibility (PCI Express Version)	x4, x8 and x16	Slot size	
Maximum Input Pixel Clock Frequency	85	MHz	
Minimum Input Pixel Clock Frequency	20	MHz	
Maximum Pixels Per Line (1 tap)	262,144 (256K)	Pixels	Section 8.2
Maximum Lines Per Frame	131,072 (128K)	Lines	Section 8.3
Minimum clocks between lines	4	Clocks	
Minimum lines between frames	0	Lines	
Minimum pixel clocks between frames	4	Clocks	
Minimum trigger pulse	600	Nanoseconds	
Minimum encoder pulse	600	Nanoseconds	
Current consumption (+5V)	???	Amps	Section 8.4
Temperature range	0 to 50	Degrees Celsius	
Humidity	25% to 80%		
Mechanical dimensions	6.8 x 4.2	Inches	
Mechanical dimensions	17.27 x 10.67	Centimeters	
Minimum UART baud rate	110	Bits/Second	
Maximum UART baud rate	256K	Bits/Second	

Preliminary

8.2 Maximum Pixels Per Line

The maximum number of pixels per line is given by the following formula:

Max_pix_per_line = 256K x Taps

Taps is the number of taps. A tap supplies a whole pixel.

Examples:

A two tap camera that supplies odd/even pixels, Max_pix_per_line = 512K.
An RGB camera that supplies RGB over 24 bits, Max_pix_per_line = 256K, as every clock the camera supplies one single pixel.
A four tap, two segments, each left right, Max_pix_per_line = 1M

Preliminary

8.3 Maximum Lines Per Frame

For area scan cameras, the maximum number of lines per frame is given by the following formula:

Max_lines_per_frame = 128K x Line_taps

Line_taps is the number of taps that supply a whole line.

Examples:

A one tap camera, Max_lines_per_frame = 128K.
A two tap camera that supplies odd/even lines, Max_lines_per_frame = 256K.
A two tap camera that supplies odd/even pixels, Max_lines_per_frame = 128K.

Preliminary

8.4 Power Consumption

????

Preliminary

Mechanical

Chapter 9

9.1 Introduction

This chapter describes the mechanical characteristics of the Karbon-CL. This includes description of all of the connectors on the board and pin-outs for these connectors.

The Karbon-CL is available in two version. The KBN-PCE-CL2, which consists just of the main board, and the KBN-PCE-CL4 which consists of the main board plus the auxiliary board. The primary purpose of the auxiliary board is to hold the third and fourth Camera Link connectors. This board does not need to be in a PCI Express slot. In fact, it does not require any power from the PC and can be located in any slot that is not used.

The mechanical layout of the Karbon-CL main board is shown in Figure 9-1. The mechanical layout of the Karbon-CL auxiliary board is shown in Figure 9-2.

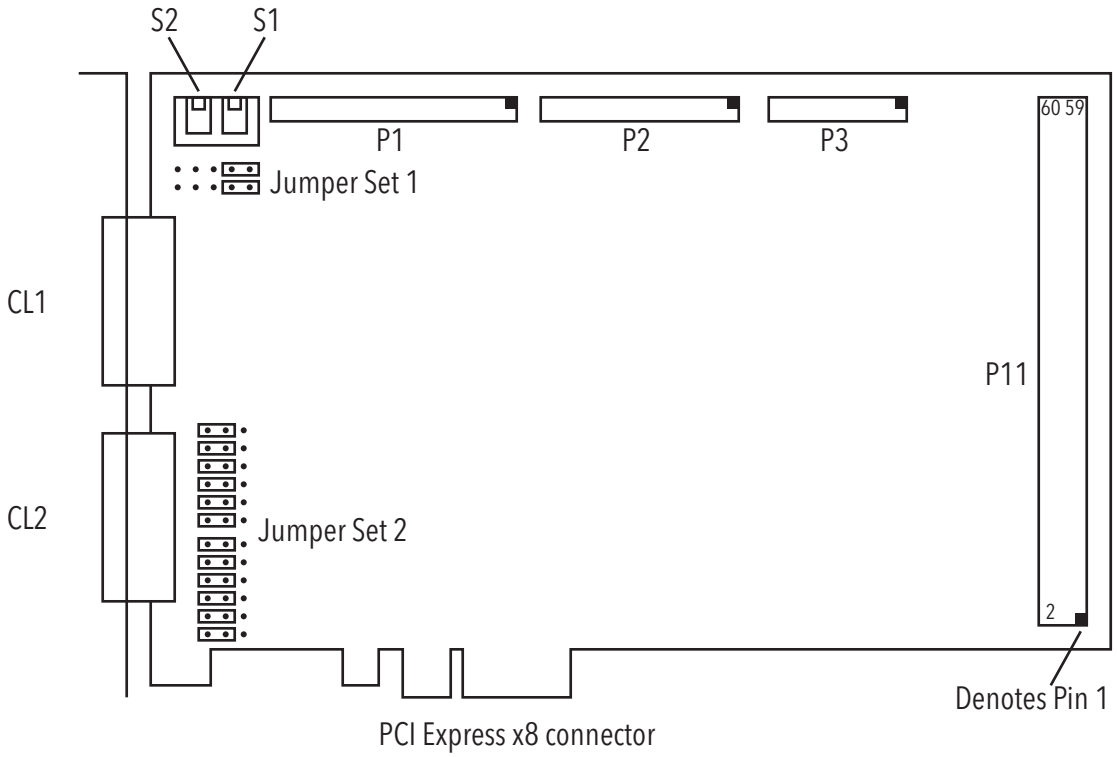


Figure 9-1 Karbon-CL Main Board Layout

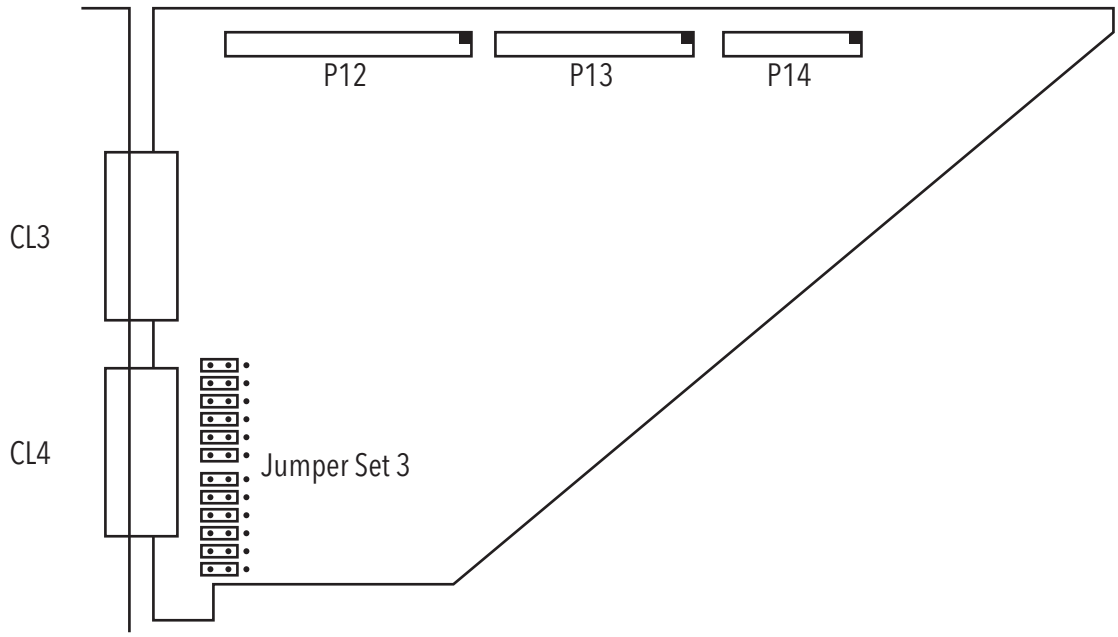


Figure 9-2 Karbon Auxiliary Board Layout

Prelim

9.2 The Karbon-CL Connectors

There are six connectors on the Karbon-CL main board:

CL1 - Camera Link Connector 1
CL2 - Camera Link Connector 2
P1 - Auxiliary Connector 1
P2 - Auxiliary Connector 2
P3 - Auxiliary Connector 3
P11 - The I/O connector

Figure 9-1 shows the locations of these connectors.

There are five connectors on the Karbon-CL auxiliary board:

CL3 - Camera Link Connector 3
CL4 - Camera Link Connector 4
P12 - Auxiliary Connector 1
P13 - Auxiliary Connector 2
P14 - Auxiliary Connector 3

Figure 9-2 shows the locations of these connectors.

9.2.1 The CL Connectors

The CL connectors are for connecting Camera Link cameras. Table 9-1 illustrates how to connect the Karbon to various types and numbers of Camera Link Cameras. Also for each camera, the Virtual Frame Grabber (VFG) that the camera will be connected with is shown..

Table 9-1 CL Connector Configuration

Camera(s)	CL1	CL2	CL3	CL4
One Base Camera	Camera 1 VFG0			
Two Base Cameras	Camera 1 VFG0	Camera 2 VFG1		
Three Base Cameras	Camaera 1 VFG0	Camera 2 VFG1	Camera 3 VFG2	
Four Base Cameras	Camera 1 VFG0	Camera 2 VFG1	Camera 3 VFG2	Camera 4 VFG3

Preliminary

Table 9-1 CL Connector Configuration

Camera(s)	CL1	CL2	CL3	CL4
One Medium Camera	Camera 1, Connector 1 VFG0	Camera 1, Connector 2 VFG0		
Two Medium Cameras	Camera 1, Connector 1 VFG0	Camera 1, Connector 2 VFG0	Camera 2, Connector 1 VFG1	Camera 2, Connector 2 VFG1
One Full Camera	Camera 1, Connector 1 VFG0	Camera 1, Connector 2 VFG0		
Two Full Cameras	Camera 1, Connector 1 VFG0	Camera 1, Connector 2 VFG0	Camera 2, Connector 1 VFG1	Camera 2, Connector 2 VFG1

Another way to look at the connectors is shown in Table 9-2. This describes the relation ship between the physical connectors on the board and the VFG's seen in SysReg.

Table 9-2 Connectors vs. VFGs

Connetor	CL2-D models	CL2-F models	CL4-D models	CL4-F models
CL1	VFG0	VFG0	VFG0	VFG0
CL2	VFG1	VFG1	VFG1	VFG0
CL3			VFG2	VFG1
CL4			VFG3	VFG1

Preliminary

9.2.2 The I/O Connector

The I/O connector, P11, contains a number of general purpose inputs and outputs. The outputs can be used, for example, to control a strobe light or other devices. Some of the inputs have specific functions, for example the Trigger and Encoder, and some are general purpose, for example GPIN0, whose state can be read by software. These signals are described in detail in the following sections.

9.2.3 The Auxiliary Connectors

The auxiliary connectors, P1, P2 and P3, are used to connect the main board to the auxiliary board connectors P12, P13 and P14 respectively. The connectors are are different sizes to avoid incorrect pairing.

9.3 The Jumpers and Switches

On the Karbon-CL main board there are four jumper fields that are user configurable, JP2, JP3, JP4 and JP5 and two that are not, JP10 and JP11. Figure 9-3 shows the locations of these jumpers and switches. On the Karbon-CL auxiliary board, ther are two jumper fields, JP1 and JP2, which are also not user configurable and that have the same functionality JP10 and JP11 on the main board. See Figure 9-2 for the location of these jumpers.

9.3.1 Jumpers

Figure 9-3 shows the detailed structure of the jumpers.

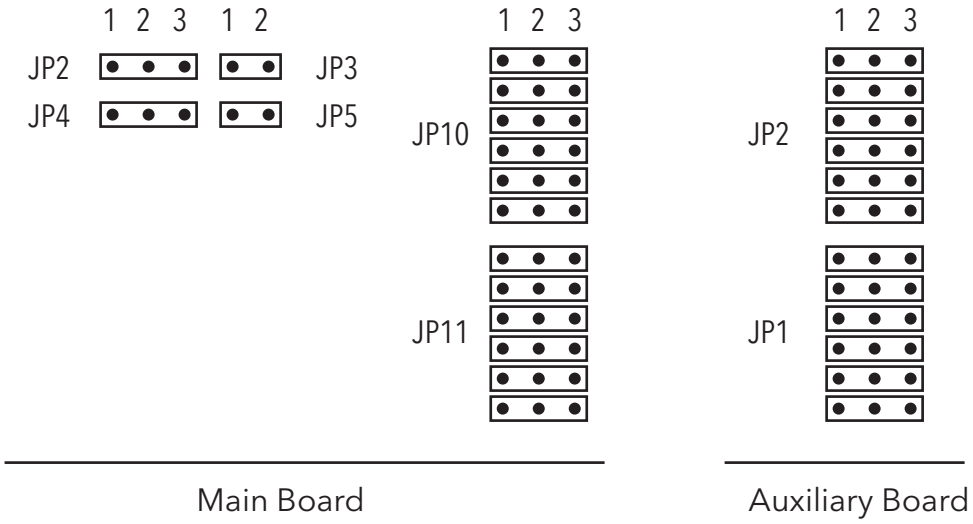


Figure 9-3 Karbon-CL Jumpers

Jumper fields JP2, JP3, JP4 and JP5 control the configuration of the open collector GPOUT5 and GPOUT6. See Section 7.5.1 on GPOUT5 and GPOUT6 configuration.

Jumper fields JP10 and JP11 on the main board and JP1 and JP2 on the auxiliar board are not user configurable. They are set at the factory in conjunction with FLASH programmable logic, when the board is configured as a FULL or DUAL model. The user should not change the factory configuration of these jumpers as it will result in malfunction of the board. The jumpers setting will disagree with the on-board FLASH logic.

Table 9-3 illustrates the configurations of JP2, JP3, JP4 and JP5. See also Section 7.5.1 on configuration of GPOUT5 and GPOUT6.

Table 9-3 Jumper Set 1

Jumper	Position 1-2	Position 2-3	No Jumper
JP2	5 Volt connected to GPOUT5_VCC through a 220 Ohm resistor	12 Volt connected to GPOUT5_VCC through a 680 Ohm resistor	Power disconnected from GPOUT5_VCC (factory default)
JP3	1K pull-up resistor installed between GPOUT5_OC and GPOUT5_VCC (factory default)	N. A.	No pull-up on GPOUT5
JP4	5 Volt connected to GPOUT6_VCC through a 220 Ohm resistor	12 Volt connected to GPOUT6_VCC through a 680 Ohm resistor	Power disconnected from GPOUT6_VCC (factory default)
JP5	1K pull-up resistor installed between GPOUT6_OC and GPOUT6_VCC (factory default)	N.A.	No pull-up on GPOUT6

Table 9-4 illustrates the factory setting for JP10 and JP11 on the the main board and JP1 and JP2 on the auxiliary board..

Table 9-4 Jumper Set 2 and 3

Jumper Field	Position 1-2	Position 2-3
JP10 main, JP1 aux	Full Configuration Cameras	Dual Base and Medium Configuration Cameras
JP11 main, JP2 aux	Full Configuration Cameras	Dual Base and Medium Configuration Cameras

Note: Users should not change the factory setting of JP1,. JP2, JP10 and JP11.

Preliminary

9.4 Switches

There is one piano-type switch block on the Karbon-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 9-5 below shows the switch settings and the corresponding value in the SW bitfield.

Table 9-5 Switch Setting

S1	S2	SW register
down	down	0
down	up	1
up	down	2
up	up	3

Preliminary

9.5 The Camera Link Connector Pinouts (CL1 to CL4)

The pinouts for connectors CL1, CL2, CL3 and CL4 conform to the Camera Link specification for frame grabbers. This specification is maintained by the Automated Imaging Association. Please contact this organization for a copy of the specification. At the time of this printing, it is available on the web at www.machinevisiononline.org. It is important to understand that some of these signals are the output of a high speed serial converter chip, and require special instrumentation to be observed.

Preliminary

9.6 The I/O Connector Pinout (P11)

The pin-out for the I/O Connector (P11) is illustrated in the Table 9-6.

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: The two camera Karbon, KBN-PCE-CL2, does not use the signals that start with "VFG2_" and "VFG3_".

Table 9-6 I/O Connector

Pin	I/O	Signal	Comment
1	In	VFG0TRIGGER+	LVDS
2	In	VFG0_TRIGGER-	LVDS
3	In	VFG0_TRIGGER_TTL	TTL
4	In	VFG0_TRIGGER_OPTO_K	Cathode of optocoupling sensor
5	In	VFG0_TRIGGER_OPTO_A	Anode of optocoupling sensor
6		GND	
7	In	VFG0_ENCODER+	LVDS
8	In	VFG0_ENCODER-	LVDS
9	In	VFG0_ENCODER_TTL	TTL
10	In	VFG0_ENCODER_OPTO_K	Cathode of optocoupling sensor
11	In	VFG0_ENCODER_OPTO_A	Anode of optocoupling sensor
12		GND	
13	In	VFG0_GPIN0_TTL	TTL
14	In	VFG0_GPIN1_TTL	TTL
15	In	VFG0_GPIN2+	LVDS
16	In	VFG0_GPIN2-	LVDS
17	In	VFG0_GPIN3+	LVDS
18	In	VFG0_GPIN3-	LVDS
19	In	VFG0_GPIN4+	LVDS
20	In	VFG0_GPIN4-	LVDS
21		GND	
22	Out	VFG0_GPOUT0+	LVDS
23	Out	VFG0_GPOUT0-	LVDS
24	Out	VFG0_GPOUT1+	LVDS
25	Out	VFG0_GPOUT1-	LVDS
26	Out	VFG0_GPOUT2+	LVDS
27	Out	VFG0_GPOUT2-	LVDS
28	Out	VFG0_GPOUT3_TTL	TTL
29		GND	

Preliminary

Table 9-6 I/O Connector

Pin	I/O	Signal	Comment
30	Out	VFG0_GPOUT4_TTL	TTL
31	Out	VFG0_GPOUT5_OC	Open collector driver
32	Out	VFG0_GPOUT5_VCC	Pull-up or power for the open collector driver
33	Out	VFG0_GPOUT6_OC	Open collector driver
34	Out	VFG0_GPOUT6_VCC	Pull-up or power for the open collector driver
35		GND	
36		GND	
37	Out	SOUT_RS232	RS232 transmit line
38		GND	
39	In	SIN_RS232	RS232 receive line
40		GND	
41	In	VFG1_TRIGGER+	LVDS
42	In	VFG1_TRIGGER-	LVDS
43	In	VFG1_TRIGGER_TTL	TTL
44	In	VFG1_ENCODER+	LVDS
45	In	VFG1_ENCODER-	LVDS
46	In	VFG1_ENCODER_TTL	TTL
47	In	VFG2_TRIGGER+	LVDS
48	In	VFG2_TRIGGER-	LVDS
49	In	VFG2_TRIGGER_TTL	TTL
50	In	VFG2_ENCODER+	LVDS
51	In	VFG2_ENCODER-	LVDS
52	In	VFG2_ENCODER_TTL	TTL
53	In	VFG3_TRIGGER+	LVDS
54	In	VFG3_TRIGGER-	LVDS
55	In	VFG3_TRIGGER_TTL	TTL
56	In	VFG3_ENCODER+	LVDS
57	In	VFG3_ENCODER-	LVDS
58	In	VFG3_ENCODER_TTL	TTL
59	In	Reserverd	TTL
60	Out	Reserved	TTL

Preliminary

Index

A

ABORT_CON KBN-4-10
ACPL KBN-4-51
ACPL_MUL KBN-4-49
ACQ_CON KBN-4-11
ACQ_SAFETY KBN-4-11
ALAST_ADD KBN-4-55
ALPF KBN-4-67
AQ_COUNT KBN-4-40
AQCMD KBN-4-22
AQSTAT KBN-4-22, KBN-5-11

B

BAYER_BIT_DEPTH KBN-4-75
BITFIELDNAME KBN-4-2
BLAST_ADD KBN-4-55
BLUE_GAIN KBN-4-74
BUTTONS KBN-4-61

C

CALC_BANK KBN-4-48
Camera Link Controls (CCs) KBN-7-11
CC_SYNC KBN-4-35
CC1_CON KBN-4-18
CC2_CON KBN-4-18
CC3_CON KBN-4-19
CC4_CON KBN-4-19
CFGCLOCK KBN-4-5
CFGDATA KBN-4-5
CFGDONE KBN-4-5
CFGEN KBN-4-5
CFGSTATUS KBN-4-5
CFREQ KBN-4-6
CHAIN_DATA_SIZE KBN-5-7
CHAIN_DATA_TOGO KBN-5-9
CL_DISABLE KBN-4-28
CLAST_ADD KBN-4-57
CLIP KBN-4-48
CMDWRITE KBN-4-20
CON0 KBN-4-4
CON1 KBN-4-8
CON10 KBN-4-50
CON11 KBN-4-54
CON12 KBN-4-56
CON13 KBN-4-58
CON14 KBN-4-60
CON17 KBN-4-64
CON18 KBN-4-66
CON19 KBN-4-68
CON2 KBN-4-15
CON20 KBN-4-70
CON21 KBN-4-73
CON23 KBN-4-76
CON24 KBN-4-78
CON25 KBN-4-82
CON26 KBN-4-84
CON27 KBN-4-85
CON28 KBN-5-2
CON29 KBN-5-4
CON3 KBN-4-21
CON30 KBN-5-6
CON31 KBN-5-8
CON32 KBN-5-10
CON33 KBN-5-13
CON36 KBN-4-86
CON37 KBN-4-88
CON38 KBN-4-90
CON4 KBN-4-24
CON5 KBN-4-31
CON6 KBN-4-37
CON7 KBN-4-39
CON8 KBN-4-41
CON9 KBN-4-46
CPLD_MODE KBN-4-7
CTAB_INT_CON KBN-4-77
CTABHOLD KBN-4-17

D

DECODER_OUTPUT KBN-4-74
DECODER_PHASE KBN-4-75
DELAY KBN-4-62
DELAY_TAP1 KBN-4-83
DELAY_TAP1_SEL KBN-4-83
DISPLAY KBN-4-47
DLAST_ADD KBN-4-57
DMA_64_BIT KBN-5-12
DMA_AUTO_START KBN-5-11
DMA_BEN KBN-5-12
DMA_BUSY KBN-4-29

Preliminary

DMA_CHAINING KBN-5-12
DMA_COMMAND KBN-5-12
DMA_DIRECTION KBN-5-11
DMA_DONE KBN-5-11
DMA_INIT_FUNC KBN-5-12
DMA_NO_RULE KBN-5-11
DMA_PRIORITY KBN-5-12
DMA_STATUS KBN-5-11
DPM_SIZE KBN-4-77
DPM_SPLIT KBN-4-63
DWNLD_MODE KBN-4-89

E

EN_ENCODER KBN-4-36
EN_TRIGGER KBN-4-35
ENC_DIV KBN-4-38
Encoder KBN-7-4
ENCPOL KBN-4-33
ENINT_CTAB KBN-4-25
ENINT_EOF KBN-4-34
ENINT_HW KBN-4-25
ENINT_OVSTEP KBN-4-25
ENINT_QUAD KBN-4-26
ENINT_SER KBN-4-26
ENINT_TRIG KBN-4-25
EOF_IN_AQ KBN-4-26

F

FACTIVE KBN-4-22
FCOUNT KBN-4-22
FENCOUNT KBN-4-29
FENPOL KBN-4-61
FIFO_EQS KBN-4-71
FIRST_QUAD_PTR_HI KBN-5-5
FIRST_QUAD_PTR_LO KBN-5-3
FORCE_8BIT KBN-4-53
FORMAT KBN-4-51
FREE_RUN_HI KBN-4-65
FREE_RUN_RATE KBN-4-65
FREEZE_CON KBN-4-11
FW_7MHZ KBN-4-5
FW_SEL KBN-4-7
FW_TYPE KBN-4-47

G

GEN_ONESHOT KBN-4-40
General Purpose Inputs (GPIN) KBN-7-6
General Purpose Outputs (GPOUT) KBN-7-7
GPIN0 KBN-4-23
GPIN1 KBN-4-23
GPIN2 KBN-4-23
GPIN3 KBN-4-23
GPIN4 KBN-4-23
GPOUT0 KBN-4-27
GPOUT0_CON KBN-4-42
GPOUT1 KBN-4-27
GPOUT1_CON KBN-4-42
GPOUT2 KBN-4-27
GPOUT2_CON KBN-4-43
GPOUT3 KBN-4-27
GPOUT3_CON KBN-4-43
GPOUT4 KBN-4-27
GPOUT4_CON KBN-4-44
GPOUT5 KBN-4-27
GPOUT5_CON KBN-4-44
GPOUT6 KBN-4-27
GPOUT6_CON KBN-4-45
GREEN_GAIN KBN-4-74

H

HAW_START KBN-4-30
HCNT_LD KBN-4-16
HCNT_RLS_STK KBN-4-17
HCNT_RLS_ZERO KBN-4-16
HCNT_RST KBN-4-16
HCOUNT KBN-4-38

I

INT_ANY KBN-4-26, KBN-4-27
INT_CTAB KBN-4-12
INT_EOF KBN-4-34
INT_HW KBN-4-12
INT_OVSTEP KBN-4-12
INT_QUAD KBN-4-13
INT_SER KBN-4-13
INT_TRIG KBN-4-13
INT_TRIGCON KBN-4-14

J

Jumpers KBN-9-5

L

L_CLKCON0 KBN-4-6
LAL KBN-4-38

Preliminary

LATCH_CONTROL KBN-5-12
LCOUNT KBN-4-28
LENPOL KBN-4-61
LINES_TOGO KBN-4-69
LUT_BANK KBN-4-80
LUT_DATA_WRITE_SEL KBN-4-80
LUT_HOST_ACCESS KBN-4-81
LUT_HOST_ADDR KBN-4-79
LUT_HOST_DATA KBN-4-79
LUT_HOST_LANE KBN-4-80
LUT_ON KBN-4-79
LUT_WEN KBN-4-80

M

MEM_ADDR_LO KBN-4-87, KBN-4-89
MEM_CS KBN-4-89
MEM_DATA KBN-4-89
MEM_WRITE KBN-4-89
MUX_REV KBN-4-47

N

NO_RULE KBN-4-11
NO_VB_WAIT KBN-4-10

O

OVS KBN-4-28

P

P11 KBN-9-9
PCOUNT KBN-4-29
Pinouts (CL1 & CL2) KBN-9-8
PIX_DEPTH KBN-4-52
POCL_CLK_DETECTED KBN-4-92
POCL_CLOCK_WAIT KBN-4-91
POCL_DETECTED KBN-4-92
POCL_EN KBN-4-5
POCL_EN_POWER KBN-4-91
POCL_GND_ON KBN-4-91
POCL_SENSE KBN-4-91
POP_TOSS KBN-4-29
PUMP_OFF KBN-4-29

Q

QTBSRC KBN-4-20

R

R/W KBN-4-3
RD_ENC_DIFF KBN-4-34, KBN-4-35
RD_ENC_OPTO KBN-4-34
RD_ENC_TTL KBN-4-34
RD_TRIG_DIFF KBN-4-33
RD_TRIG_OPTO KBN-4-34
RD_TRIG_TTL KBN-4-33
REG_GAIN KBN-4-74
RELOAD_FPGA KBN-4-7
REV_DCC KBN-4-22
RLE_LOAD_H KBN-4-45
RLE_LOAD_V KBN-4-45
RO KBN-4-3
RST_CALC_BANK KBN-4-48
RST_DPM_ADDR KBN-4-17
RST_HVCOUNT KBN-4-17
RST_OVS KBN-4-28
RST_SER KBN-4-28

S

SCALE_BY8K KBN-4-67
SEL_REG_GEN KBN-4-40
SEL_TRIG KBN-4-32
SEL_UCLK_7MHZ KBN-4-6
SELENC KBN-4-33
SHIFT_DISP KBN-4-71
SHIFT_DSP_LEFT KBN-4-72
SHIFT_DSP_SELECT KBN-4-71
SHIFT_RAW KBN-4-61
SHIFT_RAW_LEFT KBN-4-62
SHORT_FRAME KBN-4-48
SIG_GEN_POL KBN-4-67
Specifications KBN-8-1
SW KBN-4-23
SW_ENC KBN-4-33
SW_RESET KBN-4-61
SW_TRIG KBN-4-32
SWAP KBN-4-62
Switches KBN-9-7

T

TAG_BANK KBN-4-40
TOP_REV KBN-4-67
Trigger KBN-7-2
TRIGGER_DELAY KBN-4-34
TRIGPOL KBN-4-32
TRIM KBN-4-47

Preliminary

U

UART_CON KBN-4-63
UART_MASTER KBN-4-55

V

VAW_START KBN-4-30
VCNT_LD KBN-4-10
VCNT_RLS_STK KBN-4-10
VCNT_RLS_ZERO KBN-4-9
VCNT_RST KBN-4-9
VCOUNT KBN-4-38
VFG KBN-1-1
VID_BRL KBN-4-71
VID_SOURCE KBN-4-52
VIDEO_2DPM KBN-4-71
VIDEO_MASK KBN-4-59

W

WO KBN-4-3

X

XFR_PER_INT KBN-5-14

Preliminary

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP