



Intellectual Property

Component Histogram

Version 1.1

February, 2015

Corporate HQ & Design Center
380 Stevens Ave. Suite 206
Solana Beach, CA 92075
<http://www.macnica-na.com>

License and Terms of Use

This IP Core with its associated source code and support files, are being provided on an "as-is" basis and as an accommodation. Therefore all warranties, representations or guarantees of any kind (whether express, implied or statutory) including, without limitation, warranties of merchantability, non-infringement, or fitness for a particular purpose, are specifically disclaimed.

This source code may only be used in an Altera programmable logic device and may not be distributed without permission from Macnica Americas, Inc. It is provided free of royalties or fees of any kind.

Document Revision History

Revision	Date	Comments
1.0	August, 2014	Initial Draft
1.1	February, 2015	Minor clean-up and corrections

1 Contents

- 1 Contents..... 3
- 2 Introduction 4
- 3 References 4
 - 3.1 Industry Standards and Specifications..... 4
 - 3.2 Related Documents..... 4
- 4 Functional Description 5
 - 4.1 Interface Description 6
 - 4.1.1 Control 6
 - 4.1.2 Video 6
 - 4.2 Registers..... 6
 - 4.3 Functional Description 7
 - 4.4 Programming 8
 - 4.5 Specifications 9
 - 4.6 Image Size 9
 - 4.7 Performance 9
 - 4.8 Resource Utilization..... 9

2 Introduction

The Component Histogram module generates histogram data for three color components in parallel. If the input pixel data is of the format RGB, then the R component is assumed to be in the upper 8 bits of the Avalon-ST data bus, followed by Green, and then a Blue value in the lower 8 bits.

Three independent histogram tables are implemented – with one for each color component. The tables have a width of 20 bits.

The image data is sub-sampled both horizontally and vertically. Additionally, the horizontal sub-sampling averages four pixel samples prior to updating the table. The first line of the frame is sampled, followed by the 5th line etc.

The histogram data is useful in video systems to analyze the content of the image in order to implement software-controlled Auto White Balance or other algorithms, typically in systems using an image sensor as the video source.

3 References

3.1 Industry Standards and Specifications

Citations

3.2 Related Documents

Video and Image Processing User Guide, Altera Corp.

www.altera.com/literature/ug/ug_vip.pdf

Avalon Interface Specifications, Altera Corp.

www.altera.com/literature/manual/mnl_avalon_spec.pdf

4 Functional Description

The symbol below depicts the QSYS component and all of the signals. There are only two interfaces:

- An Avalon Slave with separate clock and reset, plus interrupt
- An Avalon Streaming video sink

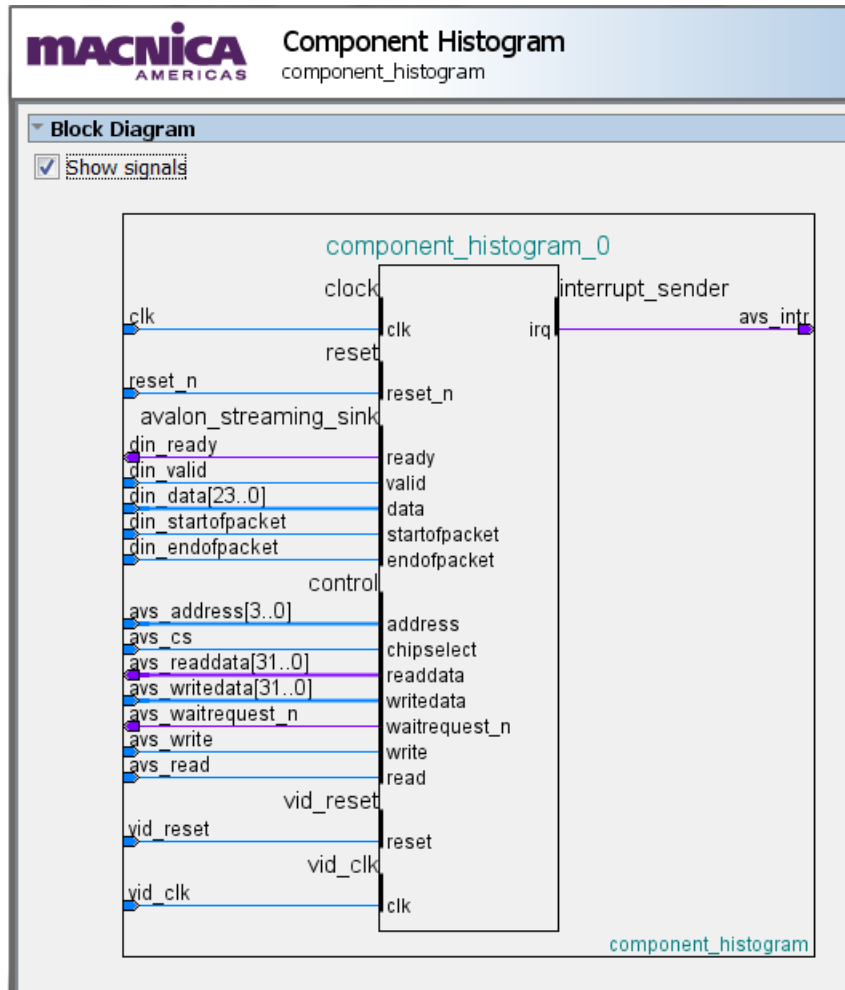


Figure 1 – Component Histogram top-level interfaces

4.1 Interface Description

4.1.1 Control

Interface	Description	Notes
Avalon Slave	A standard Avalon Slave interface for controlling the various settings and reading the histogram table.	

4.1.2 Video

Interface	Description	Notes
Avalon Streaming (sink)	A standard Avalon-ST Video interface, 8 bits per color, two colors in parallel	8bpp, parallel

4.2 Registers

Register	Description	Notes
Control Register (address 0)	This is the standard VIP control register. As usual, bit '0' is the "Go" bit that enables the module. Additional bits are: [0] – The "go" enable bit [1] – interrupt enable	
Status (address 1)	[0] = '1' = frame completed. Writing a '0' to this bit will enable the capture of another frame if the "go" bit is still '1'. [1] = interrupt pending. Writing a '0' to this bit will clear the interrupt.	
Histogram Data (address 2)	[0] = clear the histogram table contents by writing a '1' to this bit [2:1] = Ram Select. There are three independent tables that hold the histogram results. The bits written to this register select the table to be read back as follows: "00" = red component histogram "01" = green component histogram "10" = blue component histogram After writing the Ram_Select bits to select the desired table, reads from this address will return the data in the histogram ram, starting from address zero. The address is auto-incrementing, and cleared internally when a frame capture is completed, OR when a write to this register occurs (e.g. when the ram_select bits are written). The table data is 20-bits and is stored in the LSB of the returned 32-bit value.	

4.3 Functional Description

The Component Histogram is not a pass-through implementation and has no Avalon Streaming video Source output. Instead, it relies on a Splitter to provide a separate Avalon Streaming Source to the component histogram module. In the test system shown below it is fed directly by a Test Pattern Generator.

A sample verification testbench implementation is shown below.

Use	Connections	Name	Description	Export	Clock	Base
<input checked="" type="checkbox"/>		⊟ vid_clk clk_in clk_in_reset clk clk_reset dout	Clock Source Clock Input Reset Input Clock Output Reset Output	vid_clk vid_reset <i>Double-click to export</i> <i>Double-click to export</i>	<i>exported</i> vid_clk	
<input checked="" type="checkbox"/>		⊟ alt_vip_tpg_0 clock reset dout	Test Pattern Generator Clock Input Reset Input Avalon Streaming Source	<i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i>	vid_clk [clock] [clock]	
<input checked="" type="checkbox"/>		⊟ component_histo... clock reset control vid_reset vid_clk video	Component Histogram Clock Input Reset Input Avalon Memory Mapped Slave Reset Input Clock Input Avalon Streaming Sink	<i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i>	clk_0 [clock] [clock] [vid_clk] vid_clk [vid_clk]	0x50
<input checked="" type="checkbox"/>		⊟ onchip_memory2_0 clk1 s1 reset1	On-Chip Memory (RAM or ROM) Clock Input Avalon Memory Mapped Slave Reset Input	<i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i>	clk_0 [clk1] [clk1]	0x00
<input checked="" type="checkbox"/>		⊟ nios2_qsys_0 clk reset_n data_master instruction_master jtag_debug_modul... jtag_debug_module custom_instructio...	Nios II Processor Clock Input Reset Input Avalon Memory Mapped Master Avalon Memory Mapped Master Reset Output Avalon Memory Mapped Slave Custom Instruction Master	<i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i>	clk_0 [clk] [clk] [clk] [clk]	0x48
<input checked="" type="checkbox"/>		⊟ clk_0 clk_in clk_in_reset clk clk_reset	Clock Source Clock Input Reset Input Clock Output Reset Output	clk reset <i>Double-click to export</i> <i>Double-click to export</i>	<i>exported</i> clk_0	
<input checked="" type="checkbox"/>		⊟ jtag_uart_0 clk reset avalon_jtag_slave	JTAG UART Clock Input Reset Input Avalon Memory Mapped Slave	<i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i>	clk_0 [clk]	0x50

NOTE: Note that the Component Histogram has been renamed from the default component_histogram_0 convention. Due to an anomaly with QSYS, the source files are not properly copied to the simulation or synthesis sub-directories unless the _0 suffix is removed. Rename your instance as shown in the example above by right-clicking on the name and selected “rename” from the menu.

4.4 Programming

The programming flow is fairly simple:

- 1) Enable the Histogram by setting the “go” bit, and optionally the Interrupt Enable bit, in the Control register. The next single valid frame that is received will be analyzed.
- 2) The host then either waits for the completion interrupt, or checks the “frame complete” bit in the status register.
- 3) The host clears the interrupt bit (if used)
- 4) The host reads the contents of the table.
- 5) The host initiates the clearing of the table by writing to register (2) with a ‘1’ in bit 0.
- 6) The host writes a ‘0’ to bit zero of the status register to enable another capture, or alternatively turns off the “go” bit to disable the function
- 7) The clearing of the histogram ram requires two clocks per location, so with these 256-deep table approximately 512 clocks are required. All three tables are cleared simultaneously. The frame complete and “go” bits are not evaluated until after the histogram table is cleared, so there is obviously ample time for the host CPU to set these bits as desired.

A very simple polled implementation is shown below:

```
#include "sys/alt_stdio.h"
#include <io.h>
#include "../component_histogram_test_bsp/system.h"
int main()
{
    alt_putstr("Hello from Nios II!\n");
    int frame_done,i,temp,table;
    /* Event loop never exits. */
    while (1){
        // enable histogram
        IOWR(COMPONENT_HISTOGRAM_BASE,0,1);
        frame_done = 0;
        while(!frame_done){
            frame_done=IORD(COMPONENT_HISTOGRAM_BASE,1);
        }
        //read a bit from each of the three tables
        for(table=0;table<3;table++){
            IOWR(COMPONENT_HISTOGRAM_BASE,2,table << 1);
            for(i=0;i<32;i++)
                temp = IORD(COMPONENT_HISTOGRAM_BASE,2); // We don't set bit 0, so table is
not cleared
        }

        //alt_printf("data %x\n",temp);
        // clear it
        IOWR(COMPONENT_HISTOGRAM_BASE,2,1); // NOW we set bit 1, initiates table clear

        //NOW clear the frame completed bit. The init_busy will prevent a new capture
        IOWR(COMPONENT_HISTOGRAM_BASE,1,0);
    }

    return 0;
}
```


4.5 Specifications

4.6 Image Size

The implementation supports an image width of 2K pixels. There is no height restriction. Given that the table implements a 20-bit width for a maximum count of 2^{20} counts, or a “mega”-count, and the 4:1 sub-sampling both vertically and horizontally, a maximum of 16 Mpixels per frame could be supported without overflow.

4.7 Performance

The design supports a pixel rate clock of > 150 MHz and a control port clock of > 100 MHz in the test system shown, with a NIOS/e class selected as the control master.

Fmax Summary				
	Fmax	Restricted Fmax	Clock Name	Note
1	66.02 MHz	66.02 MHz	altera_reserved_tck	
2	147.91 MHz	147.91 MHz	avs_clk	
3	232.61 MHz	232.61 MHz	vid_clk	

4.8 Resource Utilization

Target Device Family	Variation	Memory	ALM
Cyclone V	standard RGB input	15360 bits / 3 M10K	~160