Skintone Detector Project: Datapath Design Specification

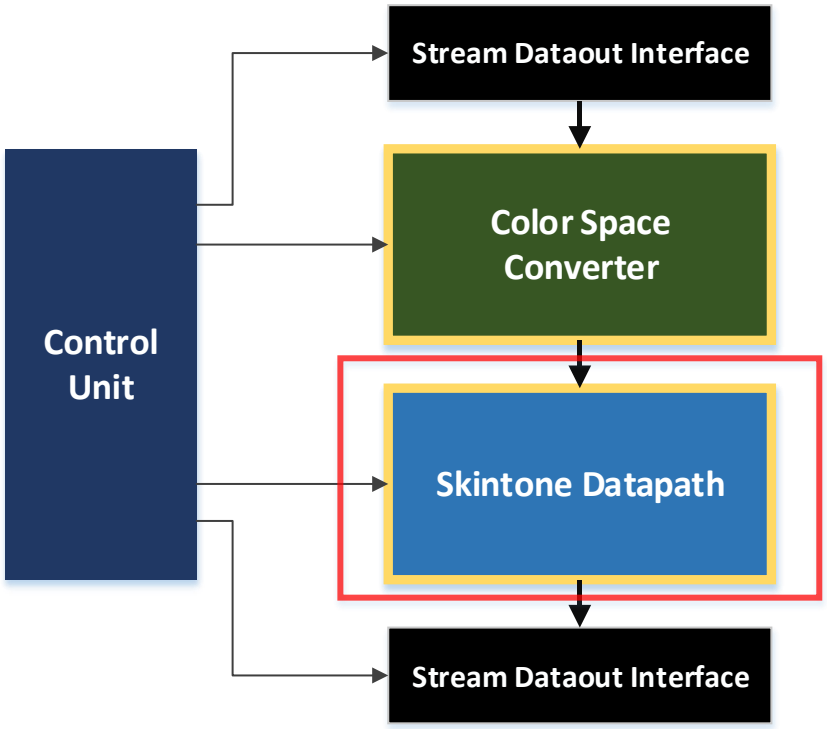

Figure 1 Skintone Detection Accelerator

Overview

In this project you will design a skintone pixel detector that has the functionality of evaluating the color components of individual pixels that comprise an image and classifying the pixel as either skintone or non-skintone. A C++ implementation of a skintone detection algorithm will be the basis of the design. Your task is to convert the C++ implementation to a fully pipelined hardware accelerator. The accelerator must be capable of processing a 1920 x 1080 video stream with a throughput of at least 30 Frames Per Second (FPS). Your design must have a minimum operating frequency of 100 MHz and must fit entirely in a Xilinx Virtex-7 690T device. The latency from input to output should be no more than the period of a single frame. You will need to decide the specific numerical representations that you will employ to achieve the most efficient design. The efficiency of your design will be scored as a ratio of performance-to-cost, where performance is the throughput (FPS) and cost is the sum of the individual cost of the component types that your design uses. You can maximize your efficiency score by maximizing throughput, minimizing cost, or both. See Table 1 for the schedule of component cost.

Table 1 Cost schedule of FPGA resources

Component Type	Cost
LUT	2 units
Registers	1 unit
BRAM	8 units
DSP	8 units

The skintone project is split into four phases: datapath design phase, controller design, system verification, and system test. Datapath design involves implementing and integrating the computational components of the skintone module. Controller design involves implementation of the datapath control state machine as well as the host driver software. System verification involves performing simulation of both the hardware and software in tandem (i.e. the software will control the simulated hardware design). Finally, system testing involves generating a design bitstream and executing the design along with the application software.

This specification describes the Datapath Design phase.

Step 1. Review the provided C++ code to understand the skintone operation. A good technique would be to bring the code into your favorite C/C++ IDE and single-step through the code.

Step 2. Create a block diagram of the architecture of your accelerator. Create individual block diagrams of the submodules that comprise your architecture. Diagrams should be detailed enough such that an individual versed in HDL design could translate your architectural diagrams to Verilog/VHDL designs. See Figure 2 for an example of detail that is expected. Determine an estimated performance and cost of your design using the cost schedule shown in Table 1.

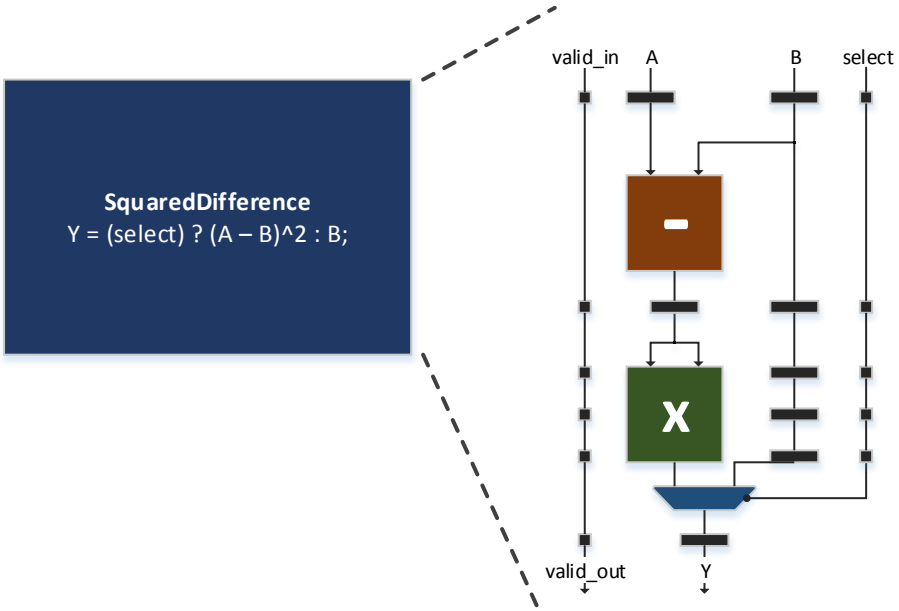

Figure 2 Example block diagram and the degree of detail expected for an architectural diagram.

Step 3. Translate your architecture into a Verilog implementation and perform functional verification. You may use the testbench provided for the color-space converter as a basis for the verification of your skintone datapath.

Step 4. Perform design synthesis to determine the actual cost of your datapath.

Interfaces

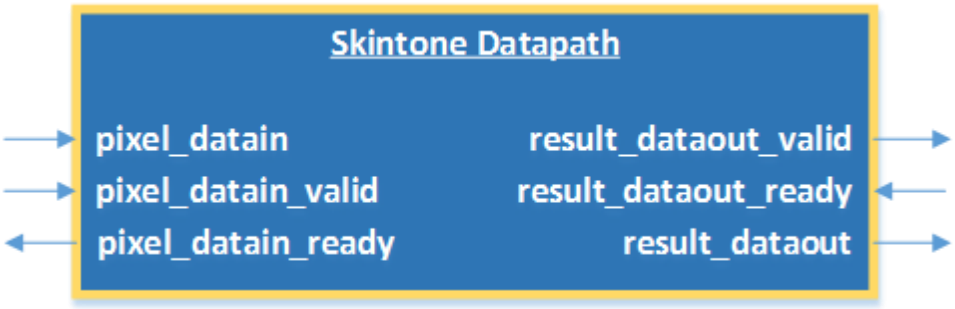

Figure 3 Skintone datapath block diagram

The following tables and diagrams specify the skintone datapath's external signals and their corresponding handshake protocols.

Clocking

Table 2 Clocking signal definitions

Signal Name	Direction	Description
clk	Input	Positive edge active clock. All signals are synchronous to this clock
rst	Input	Synchronous active high reset

Data Stream Interface

Table 3 Data Stream Interface signal definitions

Signal Name	Direction	Description
pixel_datain[23:0]	Input	Pixel bus carrying a single YCbCr pixel. The ordering is as follows Y,Cb,Cr
pixel_datain_valid	Input	Qualifier signal signifying that the data on the pixel_datain bus is valid
pixel_datain_ready	Output	Qualifier signal signifying that the accelerator is capable of accepting data on the pixel_datain bus.
result_dataout[7:0]	Output	Result bus carrying skin score.
result_dataout_valid	Output	Qualifier signal signifying that the data on the result_dataout bus is valid
result_dataout_ready	Input	Qualifier signal signifying that the downstream component is capable of accepting data on the result_dataout bus.

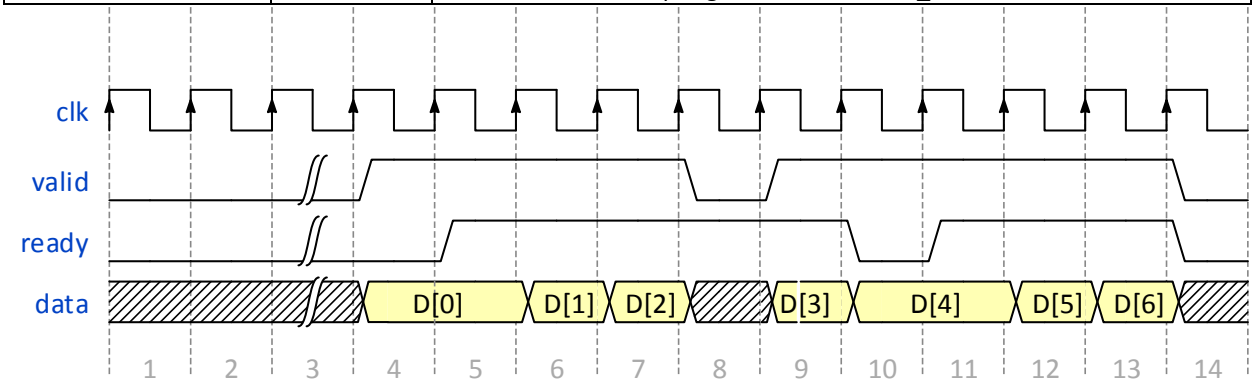

Figure 4 Data Stream Interface timing diagram

Deliverables

The Datapath phase of the project is due by midnight Tuesday April 12th. Several milestone deliverables will be due according to the following schedule:

March 27th, 2016 – Detailed datapath architecture along with estimated performance and cost.

April 8th, 2016 – Implemented and verified datapath design.

April 12th, 2016 – Synthesized design with updated performance and cost estimates.