Embedded Parallel Computing – Halmstad University

EPC-Ambric Lab2

Name:…………………………………………………………………………………….

Email:…………………………………………………………………………………….

Reg ID:…………………………………………………………………………………...

Group No.:……………………………………………………………………………….

Parallel Color Space Conversion

Operation to perform

In this lab the functionality implemented will be identical to the functionality implemented in the first lab. This will be confirmed by using the same test files to verify the operation of the design.

The objective of this lab is to perform the task done in the first lab in parallel across multiple processors for higher performance. This design has several clear opportunities for parallelization. There are two main types of parallelism: parallel execution and pipelined execution. This design has aspects that fit into each of those types. The conversion from RGB to YUV and then the conversion from 4:4:4 to 4:2:2 are two tasks that are separable, but must be done sequentially. That makes these tasks good candidates for pipeling. The operations that are done for each of the three components Y, U and V are independent from each other. That makes these tasks good candidates for parallel execution. There are additional possibilities for parallelism such as processing different lines in parallel, but for this lab we won’t look for any additional parallelism.

The topology that will be implemented in this lab is shown in Figure 1:

From the diagram you can observe that there are 7 objects which are implemented with 6 unique pieces of code. The same Subsample422 object implementation can be used for subsampling both the U and V color components.

Implementation

The same project and package can be used for this lab as was used in the first lab. A unique design file will be needed for this design (Lab2.design). The Java code for each object will need to be placed in its own file (this is a Java requirement). The aStruct code can be placed in separate aStruct files for each object, combined together in a single aStruct file or anything in between.

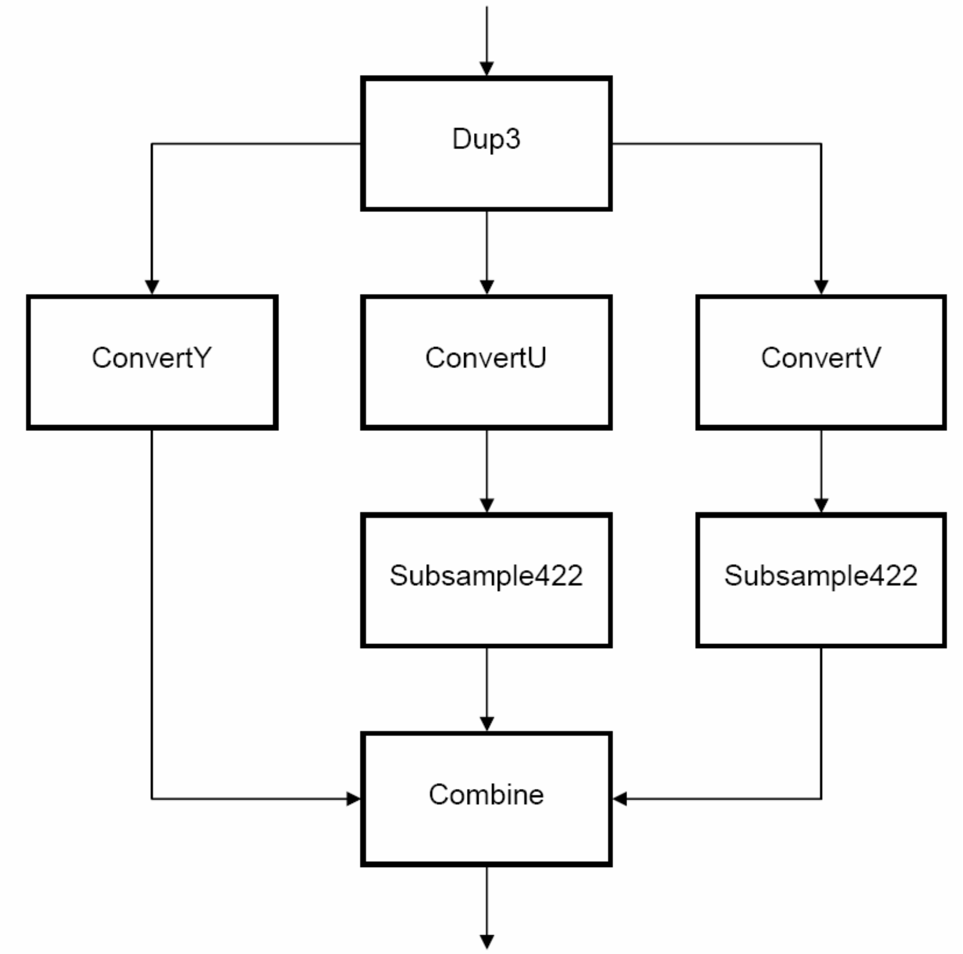

Figure 1. Block diagram of Parallel Color Space Conversion.

Write the implementation for all of the objects and the top level design.

Simulating the design

Create a new launch configuration in a similar way that you created it for the first lab, but instead of clicking the button to create a new launch configuration click on the button to create a duplicate launch configuration using the configuration from Lab1. Then give the launch configuration a new name (Lab2) and select the design file for this lab (Lab2.design). Then you can run the simulation. Since this launch configuration was copied from the first lab, the test files specified will be the same. Now in order to do the performance analysis, you will first need to make a spec file with a specimen as following:

```
# Simple measurement file that shows both interval and window
measurements

#Lab1
interval Stage_Interval_0 20 {} on Root.Convert

#Lab2
interval Stage_Interval_1 20 {} on Root.Convert.Dup3
interval Stage_Interval_2_1 20 {} on Root.Convert.ConvertY
interval Stage_Interval_2_2 20 {} on Root.Convert.ConvertU
interval Stage_Interval_2_3 20 {} on Root.Convert.ConvertV
interval Stage_Interval_3_1 20 {} on Root.Convert.SubsampleU
interval Stage_Interval_3_2 20 {} on Root.Convert.SubsampleV
interval Stage_Interval_4 20 {} on Root.Convert.Combine
```

Now perform the following tasks:

- Go to the Performance Analysis tab and select “Enable non-intrusive processor profiling”
- Specify the location of the “measurement.spec” file.
- Now by selecting the deign file and looking at the “PA Asim ISS” tab, you will find a number of trace files corresponding to each entry in the spec file.
- Open each file one by one and examine the input and output registers to get the cycle count results for each sample of output.
- For each object, record the cycle count taken between applying of first input and when the first output was available in a table.
- Now repeat the same procedure for Lab1 design and record its cycle count and compare the performance of the two designs.
- Each group is supposed to hand in a report including their source code, their results and a brief explanation of their implementation and results.

Running the design on hardware (Optional)

Running the design on hardware operates very much like running the design in simulation. Bring up the launch configuration that was just used to simulate the design and select the “Hardware Chip” button instead of “Functional Simulation”. Then select “Run” and the design will be built for the chip and run on the Software Development Board.