ECE 497: Special Project Weekly Report

Week 03

Alexander Lukens Karl Hallsby

Illinois Institute of Technology

February 11, 2021

What We Did

- Attempted to flash default chip to Alex's FPGA.
- Successfully passed USB connection to FPGA through to VirtualBox VM, interfaced with FPGA in Vivado
- Conducted "Hello World" project on FPGA to ensure that bitstream was being sent to FPGA correctly.
- Used FPGA Prototyping Flow in Chipyard to generate a bitstream for the Arty FPGA board. Strangely, the default "example" project for the Arty did not pass all timing constraints.
    - Will require additional investigation
- When creating bitstream in Chipyard, Vivado runs several tests on the design and produces detailed reports in a Chipyard folder.

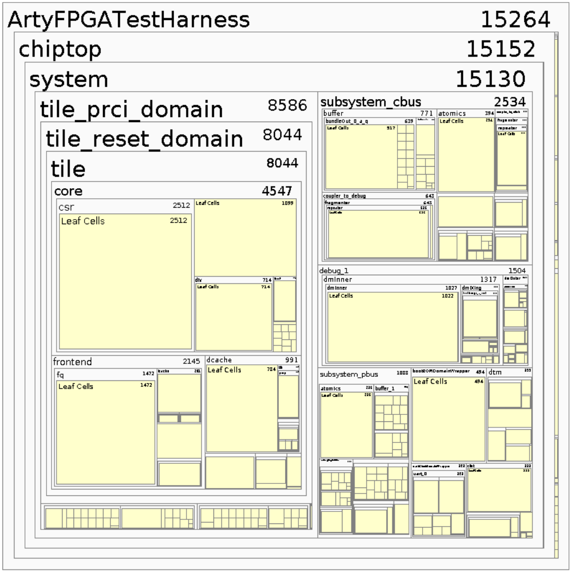
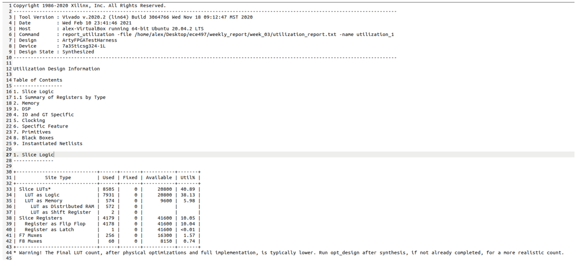


Figure: Block Diagram for Example FPGA Design



```
Copyright 1986-2020 Xilinx, Inc. All Rights Reserved.
----------------------------------------------------------------------------------------------------------------------------------
| Tool Version : Vivado v.2020.2 (lin64) Build 3064766 Wed Nov 18 09:12:47 MST 2020
| Date         : Wed Feb 10 23:41:46 2021
| Host         : alex-VirtualBox running 64-bit Ubuntu 20.04.2 LTS
| Command      : report_utilization -file /home/alex/Desktop/ece497/weekly_report/week_03/utilization_report.txt -name utilization_1
| Design       : ArtyFPGATestHarness
| Device       : 7a35ticsg324-1L
| Design State : Synthesized
----------------------------------------------------------------------------------------------------------------------------------

Utilization Design Information

Table of Contents
-----------------
1. Slice Logic
1.1 Summary of Registers by Type
2. Memory
3. DSP
4. IO and GT Specific
5. Clocking
6. Specific Feature
7. Primitives
8. Black Boxes
9. Instantiated Netlists

1. Slice Logic
--------------

+----------------------------+------+-------+-----------+-------+
|          Site Type         | Used | Fixed | Available | Util% |
+----------------------------+------+-------+-----------+-------+
| Slice LUTs*                | 8505 |     0 |     20800 | 40.89 |
|   LUT as Logic             | 7931 |     0 |     20800 | 38.13 |
|   LUT as Memory            |  574 |     0 |      9600 |  5.98 |
|     LUT as Distributed RAM |  572 |     0 |           |       |
|     LUT as Shift Register  |    2 |     0 |           |       |
| Slice Registers            | 4179 |     0 |     41600 | 10.05 |
|   Register as Flip Flop    | 4178 |     0 |     41600 | 10.04 |
|   Register as Latch        |    1 |     0 |     41600 | <0.01 |
| F7 Muxes                   |  256 |     0 |     16300 |  1.57 |
| F8 Muxes                   |   60 |     0 |      8150 |  0.74 |
+----------------------------+------+-------+-----------+-------+
* Warning! The Final LUT count, after physical optimizations and full implementation, is typically lower. Run opt_design after synthesis, if not already completed, for a more realistic count.
```

Figure: Utilization Report Overview

What We Did

- Generate a non-default chip
- Run the `asm` tests on the non-default chip
- Went spelunking through the repository to find options, their definitions, and their overrides.

What We Learned

- The repository is incredibly complicated
- **VERY** deep directory nesting (Partly due to Scala/Java project directory conventions).
- Putting the generated chip on an FPGA seems to be much more difficult than originally thought.
- Generating a non-default chip can be very easy or very hard.
  - Some of the options that must be overridden to ensure a different chip is built and simulated/benchmarked are not easy to understand or find.

Next Steps

- Continue trying to write the default chip out to Alex's FPGA and test.
- Using Alex's discovery in Vivado, collect gate counts of components within the default chip.
- Practice generating other non-default chips to understand all the options used when generating a new chip.
- Hopefully, start defining a new, custom, chip using what we know, and building a *very* small proof-of-concept.

References

Alon Amid et al. “Chipyard: Integrated Design, Simulation, and Implementation Framework for Custom SoCs.” In: *IEEE Micro* 40.4 (2020), pp. 10–21. ISSN: 1937-4143. DOI: 10.1109/MM.2020.2996616.