Archipelago - An Open Source FPGA with Toolflow Support

by

Hao Jun Liu

BASc. (University of Toronto) 2011

A Technical Report Submitted in Partial Satisfaction
of the Requirements for the Degree of

Master of Science

in

Electrical Engineering and Computer Sciences

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, BERKELEY

Committee in Charge:

Prof. John Wawrzynek, Chair
Prof. Elad Alon

Spring 2013

Archipelago - An Open Source FPGA with Toolflow Support

by

Hao Jun Liu

Abstract

Archipelago - An Open Source FPGA with Toolflow Support

by

Hao Jun Liu

Master of Science in Electrical Engineering and Computer Sciences

University of California, Berkeley

Prof. John Wawrzynek, Chair

Field Programmable Gate Arrays (FPGAs) are effective in satisfying different computing requirements such as high performance computing, digital signal processing and embedded computing. Academic research efforts cover architecture optimization, design space exploration, circuit level design and new CAD algorithms. However, actual FPGA devices in common use are all commercial. In this work, we designed and implemented an open source FPGA with toolflow support. We named the project Archipelago and designed the architecture with two goals. First, it explores the quality of the physical implementation result that is produced by a standard ASIC design flow in a modern ASIC process. Second, it enables other people to use and extend the project at will. The outcome of this project is a parameterizable and user expandable FPGA with toolflow support. We verified its functionality. The performance of the work is good enough for real world work loads. A 64 Bit counter can run up to 364MHz on average on three different FPGA instances. The result is comparable to a commercial FPGA implemented in a similar process technology.

Prof. John Wawrzynek
Committee Chair

This Report is Dedicated to Everyone Who Contributes to the Success of My Graduate Education at University of California Berkeley and the Creation of This Report, either Directly through Guidance, Discussion and Financial Support or Indirectly through Encouragement, Sharing of my Suffers and Prayers or Both.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my research advisor Prof. John Wawrzynek for his time and effort in guiding and supporting of my graduate study at University of California Berkeley. Discussions we had, academicÂărelated or non-academicÂărelated, are highlights of my graduate study and will continue to guide me afterward.

I would also like to thank Prof. Elad Alon for his time and feedback for this work.

Special thanks to The Natural Sciences and Engineering Research Council of Canada (NSERC) for two graduate fellowships in 2011 and 2012 respectively.

I would like to thank Synopsys for their ASIC Tools and TSMC for their Standard Cell Library.

Many thanks to Shaoyi for this assistance through many discussions with him on the project. Special thanks to Mr. Brain Richard for his support in using the ASIC Tools. I would also like to acknowledge the reconfigurable computing group's members for their critiques on the work.

To my parents, Yuan An Liu and Gui Ju Li, I do not know words that describe your love to me. Thanks for all the spiritual and financial support throughout my life.

Finally, I would thank Mr. Chu Tong for proofreading and editing my report.

Special thanks to CEF's brothers and sisters support in their fellowship, encouragement, prayers.

Curriculum Vitæ

Hao Jun Liu

Education

2011	Univ. of Toronto, BASc, Computer Engineering
2013	Univ. of California Berkeley, Certificate, MOT

Personal

Born	Feb 12, 1988, Rong Cheng, People's Republic of China

Chapter 1

Introduction

It is the first step that is troublesome.

– English Proverb

1.1 FPGA's Applications and Research

Field Programmable Gate Arrays (FPGAs) are effective in satisfying different computing requirements such as high performance computing [1], digital signal processing [2] and embedded computing [3]. Academic research efforts cover architecture optimization, design space exploration, circuit level design and new CAD algorithms.

The Versatile Packing, Placement and Routing tool (VPR) developed at the University of Toronto [4] has enabled a wide variety of researches from novel FPGA architectures to new CAD algorithms. Following the 2011 FPGA Conference Evening Panel on "Should the academic community launch an open-source FPGA device and tools effort?" [5] and the release of the Verilog to Routing (VTR) Project developed and integrated by the University of Toronto [6], we believe it is worth the effort to design and implement an open source FPGA hardware with toolflow support. We name the project "Archipelago". An archipelago is composed of many islands; similarly, an FPGA is composed by many tiles.

1.2 Goals and Design Decisions

We designed the Archipelago architecture with two goals. First, it explores the quality of the physical implementation result that is produced by a standard ASIC design flow in a modern ASIC process. Second, it enables other people to use and extend the project at will. The goal, however, is not to build the best performing FPGA in the world. With those goals in mind, we made several design decisions.

1. To minimize human intervention, the tool flow is streamlined and scripted.

2. The design is customizable with parameters. Parameters are centrally located in the architectural description file for VPR. Only a subset of parameters inside the architectural description file for VPR are designed into the Archipelago architecture.

3. FPGA RTLs are fully synthesizable with a standard ASIC flow with no hand layout required. Users can perform performance evaluations of the physical implementation rather than deriving performance numbers with technology scaling.

4. Users can treat this work as an FPGA generator and use it in their own circuits.

5. Not all designed features are fully implemented and supported at this point, namely the Memory Tile and Multiplier Tile.

1.3 Report Structure

This report is organized as follows. This chapter introduces the motivations, goals and design decisions for the Archipelago project. Chapter 2 provides a review on past work and related development. Chapter 3 gives a top-down review of the Archipelago open source FPGA architecture. Chapter 4 documents a bottom-up description of modules at the RTL level. Chapter 5 describes the reconfiguration architecture and techniques used. Chapter 6 explains the supporting FPGA software with flow diagrams. Chapter 7 discusses functional test cases and methodologies. Chapter 8 presents critical path delay, area and static power consumption data and plots for different design parameters. Chapter 9 concludes the achievements in the work and suggests future directions of the project.

Chapter 2

Background

If I have seen further it is by standing on the shoulders of giants.

– Isaac Newton

2.1 Research on Open Source FPGA Tools

An FPGA has limited use without a toolflow which transforms a user RTL to a bitstream just as an instruction set architecture has limited use without a compiler which transforms a user program described in high level language to a piece of machine code. The VTR Project [6] provides a toolflow from RTL written in Verilog HDL to a placed and routed design. The work integrates ODIN II [7], ABC [8] and VPR [9] into a flow. They provide Verilog HDL elaboration, logic synthesis and technology mapping, packing, instances placing and inter-instances routing respectively. The VTR Project provides us a foundation of the toolflow required. All tools are open source and are publicly available.

2.2 Research on Open Source FPGA Hardware

FPGA design and implementation are a time consuming process. It has been estimated that the layout of an FPGA tile can take from 50 to 200 man years to complete. [10] Using a standard cell flow in this project saves user effort. Some works have been devoted in studying the results of a standard cell flow based approach in generating FPGA layouts and have demonstrated promising results. [11], [12] One recent work even demonstrates that with a 45nm process technology, reduction in area and improvement in yield can be achieved at the same time using a standard cell flow. [13]

One project that attempted to generate full FPGA layout with bitstream generation presented by Kuon et al from the University of Toronto [14] is over seven years old

and only minor extensions are publicly available to this point. However the results from the work are promising.

With results from previous work, we believe it is totally feasible and cost effective to implement an FPGA in RTL and to use standard ASIC tools doing the layout generation.

2.3 The Chisel Language

Verilog HDL has been one of the de Facto languages in describing an RTL. However, its support in parametrization and speed in the simulation are two big drawbacks of the language.

In the Archipelago project, we use Chisel, an RTL generation language developed at University of California Berkeley, to implement all the components in an FPGA. [15] Chisel stands for Constructing Hardware in Scala Embedded Language. The language offers concise description of an RTL with versatile support on parametrization and offers fast C++ cycle by cycle simulation. The top level connections are generated in Verilog HDL because Chisel cannot generate a circuit with combinational paths. An FPGA must have combinational paths to have functionality and routability. It is also beyond Chisel capability to perform large scale simulation as the C++ simulator flattens a design. The flattened simulator can also be too large to fit in most host computers' caches. This results very slow simulation speed.

Chapter 3

The Archipelago Architecture

> Architecture starts when you carefully put two bricks together. There it begins.
>
> – Ludwig Mies van der Rohe

3.1 Column Based FPGA Architecture

Column based FPGA architecture has been adapted in commercial FPGA devices. [16] It is characterized by each column of the FPGA has identical functional tiles by ignoring IO Tiles. (Fig. 3.1) We choose the column based architecture as it offers simplicity in standard ASIC flow, is well supported in VTR and is relatively simple in generating bitstreams. Fig. 3.1 gives a high level view of the architecture. It illustrates an FPGA that is 8 by 8 in tiles with IOs on the perimeter and logic, storage and computing resources in the middle.

There are four different tiles in the architecture. They are IO Tiles, LUT Tiles, MEM Tiles, MUL Tiles. Section 3.2 provides detail descriptions on their functions. Commercial FPGAs have more advanced tiles to provide additional functionality such as A/D converters, differential IOs and memory controllers. Those circuits are mixed signal circuits and require special support in design and tools, so they are beyond the scope of this work.

3.2 Tile Functions

The four tiles have different functions with one common feature: they all have connections on their sides to make neighbor tiles connected. Non-neighbor tiles are indirectly connected via tiles among them. How they are connected is part of the routing architecture which is described in Section 3.3. We depict the difference of their functions as follows.

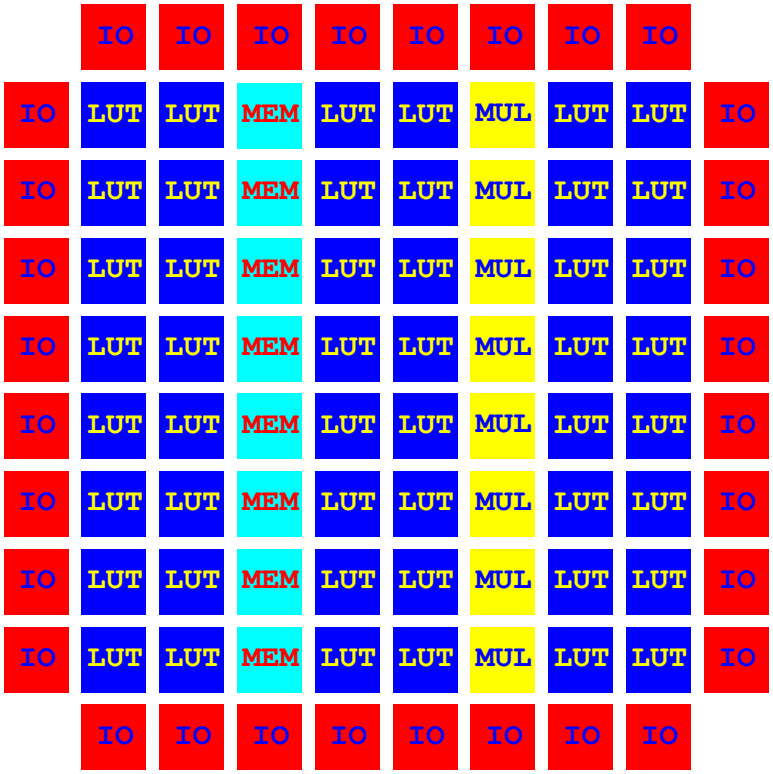


Figure 3.1: Full FPGA Arch

IO Tile

The IO Tile (Fig. 3.2.(a)) terminates input routing tracks and initiate output routing tracks. It provides a gateway for the FPGA, so input and output to the FPGA are routed into the FPGA routing fabric.

LUT Tile

LUT Tile (Fig. 3.2.(b)) stands for Look Up Table Tile. It is the foundation of FPGA as it contains Look Up Tables for implementing combinational functions and hard flip-flops for storage.

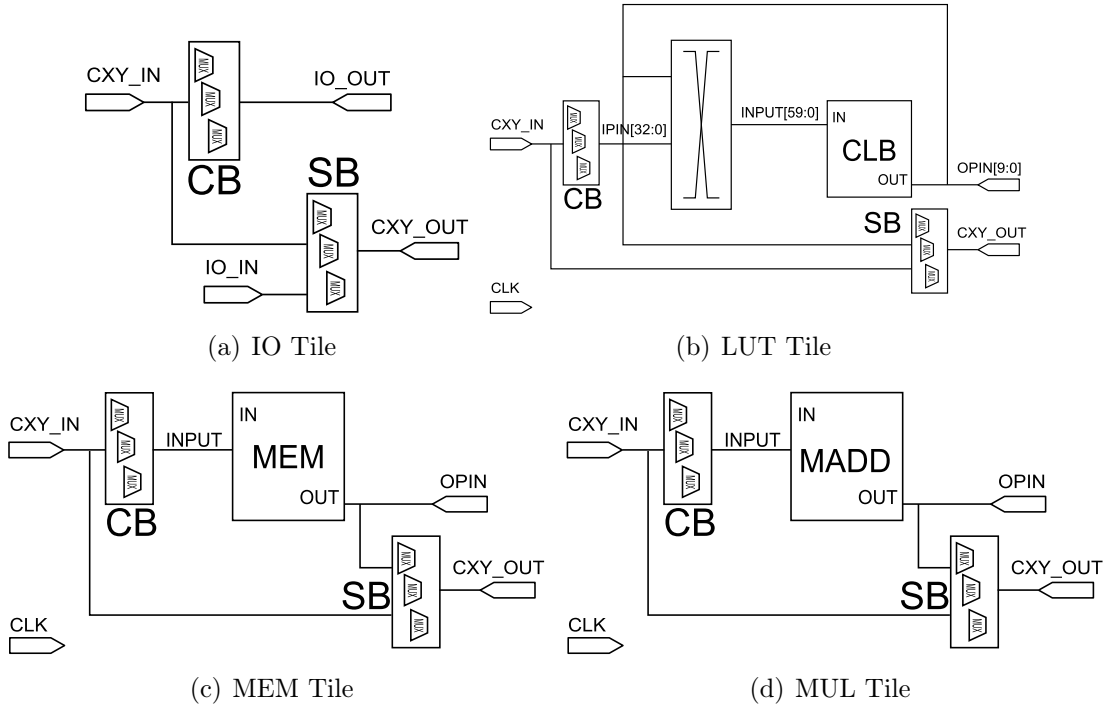
MEM Tile

MEM Tile (Fig. 3.2.(c)) contains a dual ports configurable width memory accessible by the user. It is a designed feature in the project but is not fully implemented and not yet supported. However, we have an evaluation work to test the performance of the tile. It demonstrates operating speed up to 600MHz for a 32KBits configurable widths memory. It is implemented with a 65nm process technology and all the implementation is in Verilog HDL.

MUL Tile

Implementing a generic multiplier is costly and has low performance using Look Up Tables (LUT). It also consumes precious routing resources as many LUTs are needed and interconnections have to be made. Hardware Multipliers are

cost effective in implementing multiple functions. The MUL Tile (Fig. 3.2.(d)) contains configurable and fracturable width hard multipliers. For example, a MUL Tile with a 36 by 36 multiplier can be configured into two 18 by 18 multipliers or four 9 by 9 multipliers. Implementing a similar function uses more than 100 LUT Tiles. It is a designed feature in the project but is not fully implemented and not yet supported. Our quick evaluation work just to test the performance of the tile reveals operating speed up to 700MHz. It is implemented with a 65nm process technology and all the implementation is in Verilog HDL.



(a) IO Tile (b) LUT Tile

(c) MEM Tile (d) MUL Tile

Figure 3.2: Structure of tiles

3.3 Routing Architecture

3.3.1 High Level Overview

On a high level view, there are horizontal and vertical wires in an FPGA. They are named *X Track* and *Y Track* respectively. A *switch box (SW)* connects several tracks together. The architecture can be abstracted as a graph with no islands. That implies routability for any signal with one source and one destination. If the signal flows in only one way on a track, the track is called *uni-directional*. If the signal can flow both ways on a track, the track is *bi-directional*. A track driven by a single signal is a *single-driver* wire. The single signal is the output of a *multiplexer (MUX)*. A

track driven by several signals is a *multi-driver* wire. The sources are output of several MUXs and there are pass transistors to select one of the MUXs to be the source.

A routing track that spans more than one tile is called *segmented track*. A segment 1 track connects two neighbor tiles. A segment 4 track connects five tiles either horizontally (a Segment 4 X Track) or vertically (a Segment 4 Y Track). Segmentation is especially useful for high performance global routing as the number of MUXs a signal passes through is reduced. Fig. 3.5 illustrate a segment of 4 wire in both X and Y direction.

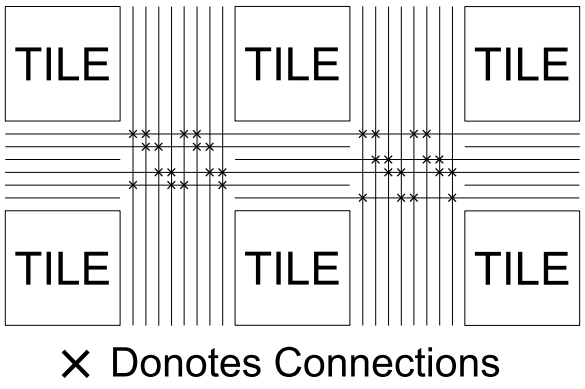
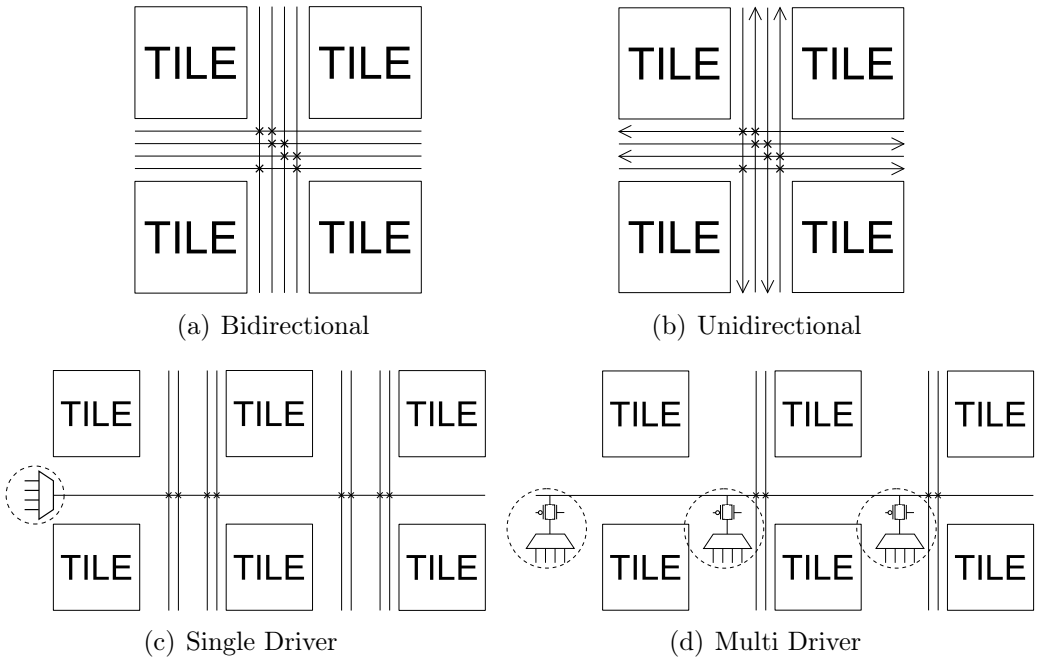


Figure 3.3: Routing Tracks



(a) Bidirectional (b) Unidirectional

(c) Single Driver (d) Multi Driver

Figure 3.4: Routing Illustrated

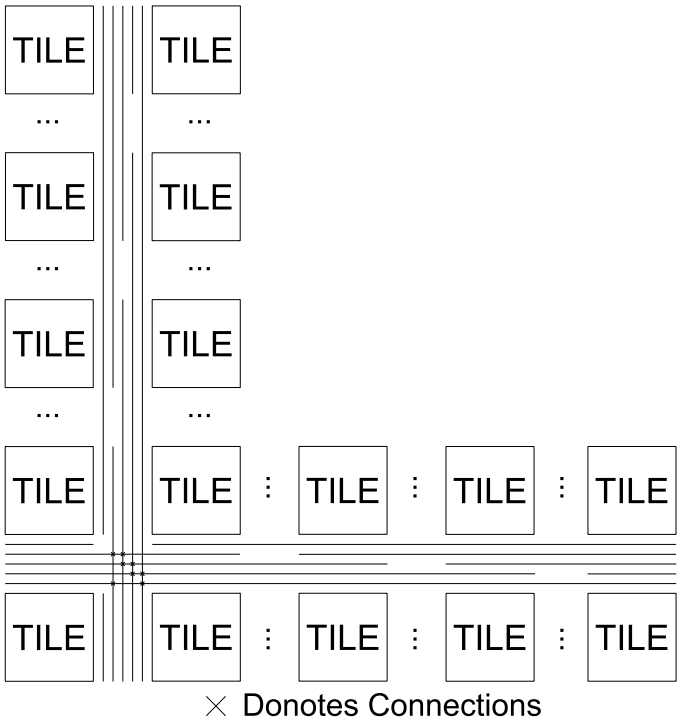


Figure 3.5: Segmentation Illustrated.

3.3.2 Routing Track

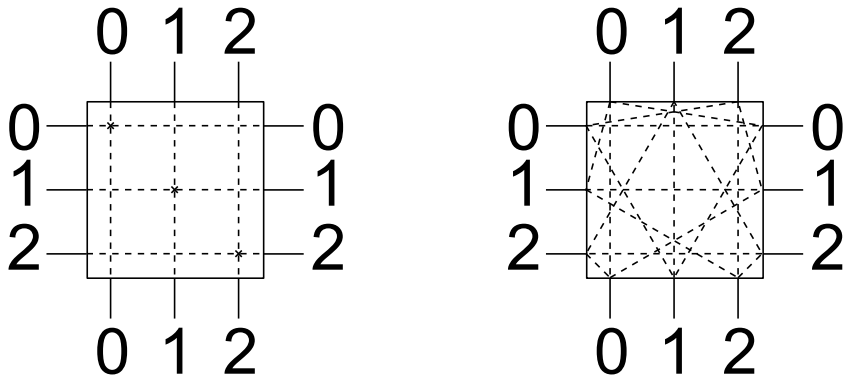
In this work we choose a single driver unidirectional routing architecture because of the following constraints and benefits:

1. It is area efficient, energy efficient and enables higher clock speed. [17], [18]
2. VPR supports only single-driver routing. [9]
3. Both bi-directional and multi-driver routing require the use of pass transistors which are not available in a standard cell ASIC flow. This is in contradiction with the objective to use only standard cell ASIC flow in the work. Also, pass transistors require special handling in static timing analysis.

3.3.3 Switch Box (SB)

A switch box has switches that can connect two routing tracks. It is part of the inter-tile routing architecture. It has two variants: one is a simple design and is called *planar topology*; the other is more sophisticated but enables better connectivity and it called *Wilton topology*. (Fig. 3.6) Assuming routing tracks between two neighbor tiles are numbered from 0 and up with track 0 being the left most track and bottom most track, the planar topology only connects tracks that have the same number;

the Wilton topology connects tracks with the same or different track numbers so the topology is more flexible. In this work, we only use Wilton switch box topology. All the switch boxes are incorporated into tiles so the architecture diagram does not reflect their existence.

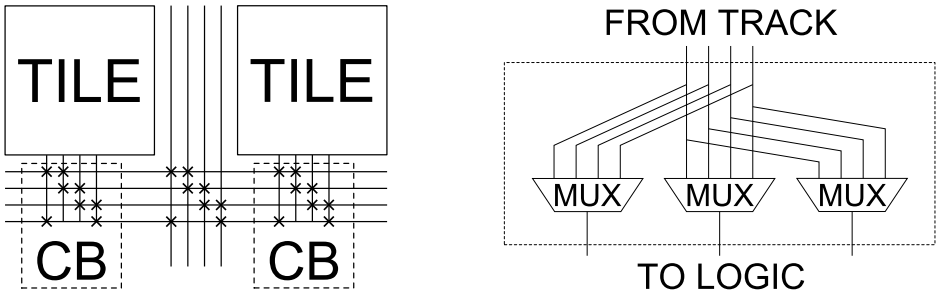


(a) Planar Topology (b) Wilton Topology

Figure 3.6: Switch Box Illustrated

3.3.4 Connection Box (CB)

A connection box has switches that can connect a routing track to an *input pin (IPIN)*. It is part the intra-tile routing architecture. On the circuit level, it has a group of MUXs with routing tracks as input and IPINs as output. (Fig. 3.7)



(a) Connection Box (b) Connection Circuit

Figure 3.7: Connection Box Illustrated

3.4 Clocking Structure

All the edge-triggered elements in the FPGA are on the same clock domain. This is because 1) the open source FPGA tools do not support multi-clock domain circuit

placement and timing analysis, 2) it adds complexity in physical implementation of the FPGA circuit 3) it significantly adds routing complexity and 4) the routing architecture needs to be modified for feeding clock signals into routing fabric. However, it is feasible to implement it as a future work.

In the physical implementation phase, the clock tree is synthesized, placed and routed with ASIC flow tools.

3.5 Regularities in Tiling

Excluding IO Tiles, all tiles not on the boundary share the same configuration switch box and connection box. Therefore, we have one tile generator for such type of tiles. Using a single generator enables us to take advantage of a faster ASIC flow as the placement and routing result inside many tiles is shared. Below is the actual code that implements a switch box and a connection box inside a non-boundary LUT Tile.

```
1  class sbcb extends Component {
2    val io = new Bundle {
3      val ipin_in = Bits(INPUT, VAR_NUM_IPIN_PER_TILE*VAR_IPIN_INPUT_WIDTH)
4      val ipin_config = Bits(INPUT, VAR_NUM_IPIN_PER_TILE*VAR_IPIN_CONFIG_WIDTH)
5
6      val chanxy_in = Bits(INPUT, TOTAL_CHANXY_IN)
7      val chanxy_config = Bits(INPUT, VAR_TOTAL_CHANXY_CONFIGS)
8
9      val ipin_out = Bits(OUTPUT, VAR_NUM_IPIN_PER_TILE)
10     val chanxy_out = Bits(OUTPUT, VAR_NUM_CHANXY_PER_TILE)
11   }
12
13   val ipin_in_break = new Array [Bits] (VAR_NUM_IPIN_PER_TILE)
14   val ipin_config_break = new Array [Bits] (VAR_NUM_IPIN_PER_TILE)
15   val to_ipin_out = new Array [Bits] (VAR_NUM_IPIN_PER_TILE)
16   val chanxy_in_break = new Array [Bits] (VAR_NUM_CHANXY_PER_TILE)
17   val chanxy_config_break = new Array [Bits] (VAR_NUM_CHANXY_PER_TILE)
18   val to_chanxy_out = new Array [Bits] (VAR_NUM_CHANXY_PER_TILE)
19
20   var i = 0;
21
22   for ( i <- 0 until VAR_NUM_IPIN_PER_TILE) {
23     ipin_in_break(i) = Bits (width = VAR_IPIN_INPUT_WIDTH)
24     ipin_in_break(i) := io.ipin_in((i+1)*VAR_IPIN_INPUT_WIDTH - 1,
           i*VAR_IPIN_INPUT_WIDTH)
25
26     ipin_config_break(i) = Bits (width = VAR_IPIN_CONFIG_WIDTH)
27     ipin_config_break(i) := io.ipin_config((i+1)*VAR_IPIN_CONFIG_WIDTH - 1,
           i*VAR_IPIN_CONFIG_WIDTH)
28
29     to_ipin_out(i) = Bits (width = 1)
30     to_ipin_out(i) := ipin_in_break(i)(ipin_config_break(i))
31   }
32
```

```
  for ( i <- 0 until VAR_NUM_CHANXY_PER_TILE )
  {
    var chanxy_in_start_bit = VAR_CHANXY_INPUT_START_ARR(i)
    var chanxy_in_end_bit = VAR_CHANXY_INPUT_START_ARR(i) +
        VAR_CHANXY_INPUT_WIDTH_ARR(i) - 1

    var chanxy_config_start_bit = VAR_CHANXY_CONFIG_START_ARR(i)
    var chanxy_config_end_bit = VAR_CHANXY_CONFIG_START_ARR(i) +
        VAR_CHANXY_CONFIG_WIDTH_ARR(i) - 1

    chanxy_in_break(i) = Bits (width =  VAR_CHANXY_INPUT_WIDTH_ARR(i))
    chanxy_in_break(i) := io.chanxy_in(chanxy_in_end_bit, chanxy_in_start_bit)

    chanxy_config_break(i) = Bits (width = VAR_CHANXY_CONFIG_WIDTH_ARR(i))
    chanxy_config_break(i) := io.chanxy_config(chanxy_config_end_bit,
        chanxy_config_start_bit)

    to_chanxy_out(i) = Bits (width = 1)
    to_chanxy_out(i) := (chanxy_in_break(i))(chanxy_config_break(i))
  }

  io.ipin_out := convert_Array_to_Bits (to_ipin_out, VAR_NUM_IPIN_PER_TILE)
  io.chanxy_out := convert_Array_to_Bits (to_chanxy_out, VAR_NUM_CHANXY_PER_TILE)
}
```

Some boundary tiles are unique so there is a more evolved generator for such tiles. Below is the actual code that implements a switch box and a connection box inside a boundary LUT Tile.

```
class sbcb_sp (param: lut_Param) extends Component {
  val io = new Bundle {
    val ipin_in = Bits(INPUT, VAR_NUM_IPIN_PER_TILE*VAR_IPIN_INPUT_WIDTH)
    val ipin_config = Bits(INPUT, VAR_NUM_IPIN_PER_TILE*VAR_IPIN_CONFIG_WIDTH)

    val chanxy_in = Bits(INPUT, param.num_chanxy_in)
    val chanxy_config = Bits(INPUT, param.num_chanxy_configs)

    val ipin_out = Bits(OUTPUT, VAR_NUM_IPIN_PER_TILE)
    val chanxy_out = Bits(OUTPUT, param.num_chanxy_out)
  }

  val ipin_in_break = new Array [Bits] (VAR_NUM_IPIN_PER_TILE)
  val ipin_config_break = new Array [Bits] (VAR_NUM_IPIN_PER_TILE)
  val to_ipin_out = new Array [Bits] (VAR_NUM_IPIN_PER_TILE)
  val chanxy_in_break = new Array [Bits] (param.num_chanxy_out)
  val chanxy_config_break = new Array [Bits] (param.num_chanxy_configs)
  val to_chanxy_out = new Array[Bits] (param.num_chanxy_out)

  var i = 0;

  for ( i <- 0 until VAR_NUM_IPIN_PER_TILE) {
    ipin_in_break(i) = Bits (width = VAR_IPIN_INPUT_WIDTH)
```

```
24      ipin_in_break(i) := io.ipin_in((i+1)*VAR_IPIN_INPUT_WIDTH - 1,
            i*VAR_IPIN_INPUT_WIDTH)
25
26      ipin_config_break(i) = Bits (width = VAR_IPIN_CONFIG_WIDTH)
27      ipin_config_break(i) := io.ipin_config((i+1)*VAR_IPIN_CONFIG_WIDTH - 1,
            i*VAR_IPIN_CONFIG_WIDTH)
28
29      to_ipin_out(i) = Bits (width = 1)
30      to_ipin_out(i) := ipin_in_break(i)(ipin_config_break(i))
31    }
32
33    for ( i <- 0 until param.num_chanxy_out )
34    {
35        if ( param.chanxy_config_width_list(i) == 0 )
36        {
37            to_chanxy_out(i) = Bits (width = 1)
38            to_chanxy_out(i) := io.chanxy_in ( param.chanxy_out_index_start_list(i) )
39        }
40        else
41        {
42            var chanxy_in_start_bit = param.chanxy_out_index_start_list(i)
43            var chanxy_in_end_bit = param.chanxy_out_index_start_list(i) +
                param.chanxy_out_input_width_list(i) - 1
44
45            var chanxy_config_start_bit = param.chanxy_config_start_list(i)
46            var chanxy_config_end_bit = param.chanxy_config_start_list(i) +
                param.chanxy_config_width_list(i) - 1
47
48            chanxy_in_break(i) = Bits (width = param.chanxy_out_input_width_list(i))
49            chanxy_in_break(i) := io.chanxy_in(chanxy_in_end_bit, chanxy_in_start_bit)
50
51            chanxy_config_break(i) = Bits (width = param.chanxy_config_width_list(i))
52            chanxy_config_break(i) := io.chanxy_config(chanxy_config_end_bit,
                chanxy_config_start_bit)
53
54            to_chanxy_out(i) = Bits (width = 1)
55            to_chanxy_out(i) := (chanxy_in_break(i))(chanxy_config_break(i))
56        }
57    }
58
59    io.ipin_out := convert_Array_to_Bits (to_ipin_out, VAR_NUM_IPIN_PER_TILE)
60    io.chanxy_out := convert_Array_to_Bits (to_chanxy_out, param.num_chanxy_out)
61  }
```

3.6 Architecture of Current Work

The MEM Tile and MUL Tile are implemented in hardware using Verilog HDL. However, because of the effort takes in setting up an additional tool flow, we choose to leave it as part of the future work. Therefore, the fully implemented architecture is like the one illustrated in Fig. 3.8. The are many parameters user can change and

the tool will handle all the rest of the work. Appendix A lists all user accessible parameters and their legal values.

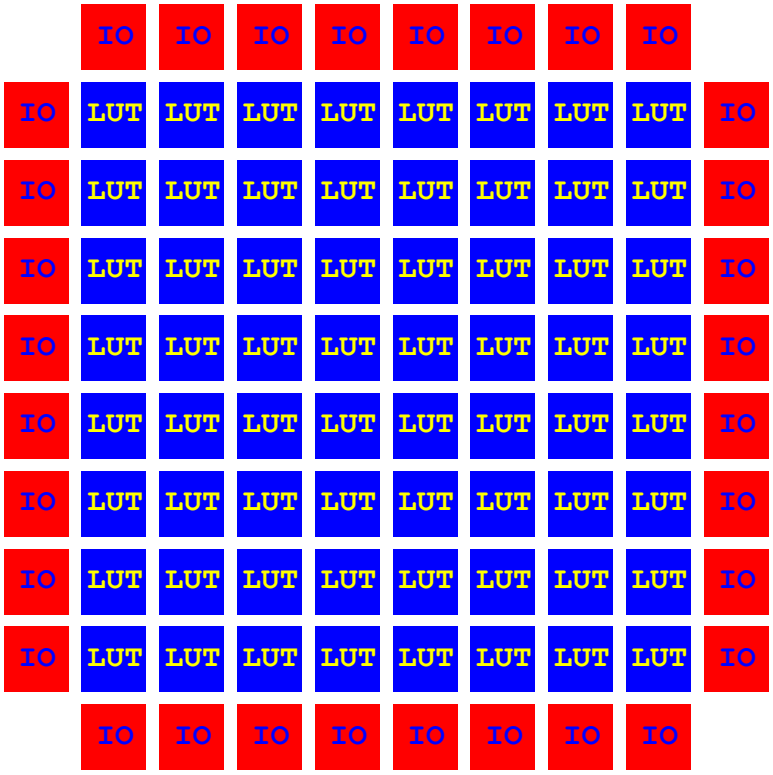


Figure 3.8: LUT FPGA Architecture

Chapter 4

Modules

Big things have small beginnings.

– Lawrence of Arabia

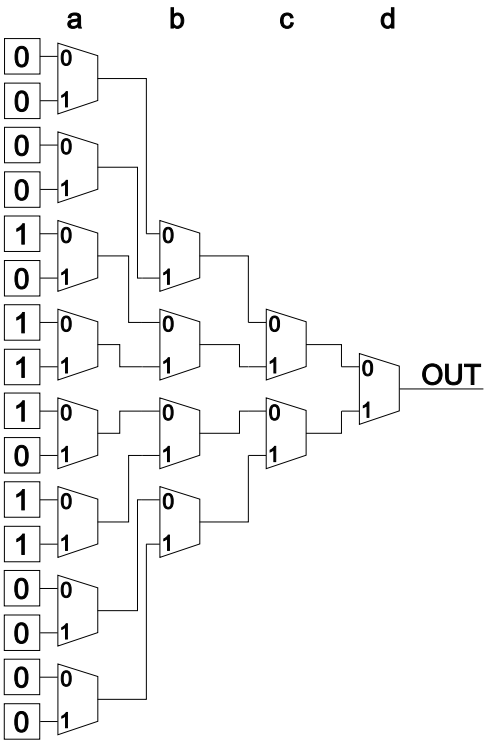
4.1 Look Up Table (LUT)

A Look Up Table (LUT) implements combinational functions. The function is programmed with configuration bits. It has two input signals, config_in and config_sel. It has one output signal lut_out. The width of the config_sel has to be equal or more than two. Config_in's width is two to the power of the width of config_sel. For example, a 4-input 1-output LUT can implement any combinational circuit with four variables and one output and has 16 Bits wide config_in. Fig. 4.1.(a) illustrates a 4-input 1-output LUT that implements $out = (a \oplus b) \wedge (\neg c \vee d)$.

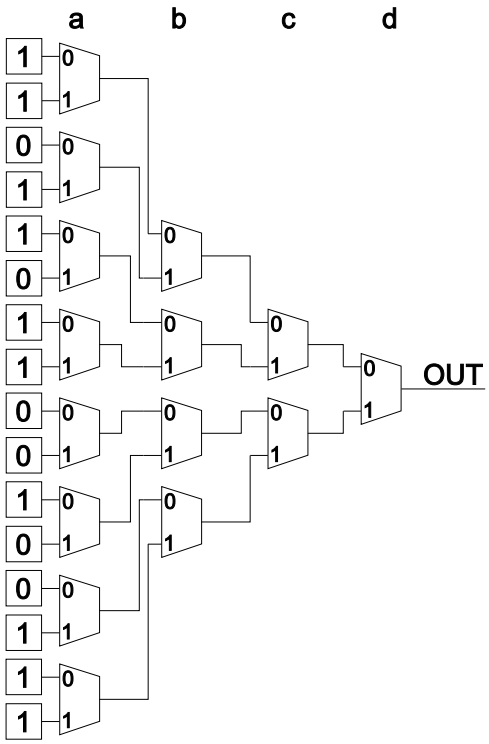
A LUT with 4-input, 5-input and 6-input provides the best area-delay product. [19] A 6-input LUT can be splitted into two 5-input LUTs to implement two different 5-input functions provided the two functions' inputs are same. This only adds an overhead of one MUX. This scheme is illustrated in Fig. 4.1(c).

4.2 Basic Logic Element (BLE)

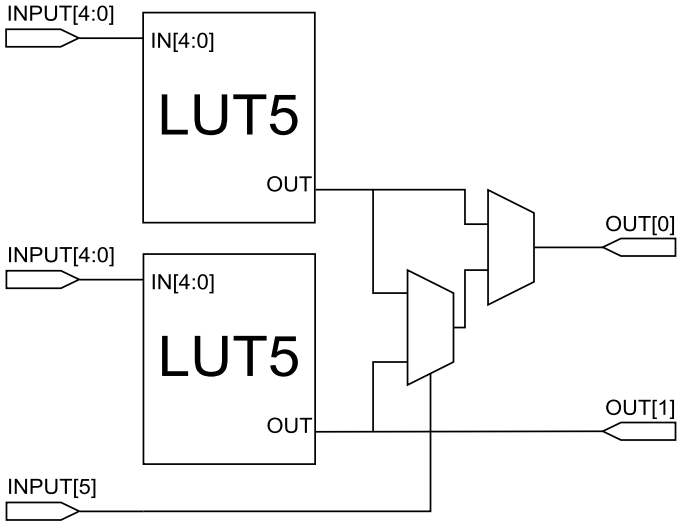
A pure combinational circuit is not as interesting as a computing machine with storage. A Flip Flop (FF) is a clock edge triggered storage element. In our design, each BLE has a LUT and a FF. The FF can be bypassed, so the output from the LUT is connected directly to the output of the BLE. (Fig. 4.2) Below is the code that implements a BLE with a 6-input LUT and one FF.



(a) 4-input 1-output LUT Function 1



(b) 4-input 1-output LUT Function 2



(c) 6-input fractured LUT

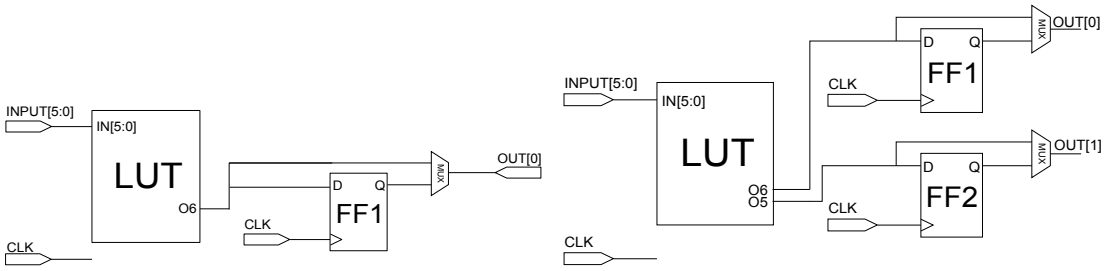
Figure 4.1: LUT Illustrated

```
class lut6 extends Component {
  val io = new Bundle {
    val lut_in = Bits(INPUT, 6)
    val lut_out = Bits(OUTPUT, 1)
    val lut_configs = Bits(INPUT, 64)
    val mux_configs = Bits(INPUT, 1)
    val ff_en = Bool(INPUT)
  }

  val lut6_o = io.lut_configs(io.lut_in)

  // declare one posedge flip-flops
  val ff1 = Reg(resetVal = Bits("b0", 1))

  // ff_en is disabled while programming
  when (io.ff_en) {
    ff1 := lut6_o
  }

  io.lut_out := Cat(ff1, lut6_o)(io.mux_configs(0))
}
```

The following code implements a BLE with a fracturable 6-input LUT and two FFs. It is not yet supported in the tool flow (Bitstream Generation Phase) but is shown to demonstrate our work is designed to be expandable at the hardware level.

```
// This is a 6-input fracturable lut with two outputs and two FFs
class lut6s extends Component {
  val io = new Bundle {
    val lut_in  = Bits(INPUT, 6)
    val lut_out = Bits(OUTPUT, 2)
    val lut_configs = Bits(INPUT, 64)
    val mux_configs = Bits(INPUT, 3)
    val ff_en = Bool(INPUT)
  }

  // assign lut5 outputs from lut_config and lut input
  val lut5_o0 = io.lut_configs(Cat(Bits(0), io.lut_in(4, 0)))
  val lut5_o1 = io.lut_configs(Cat(Bits(1), io.lut_in(4, 0)))

  // assign lut6 outputs
  val lut6_o = Cat(lut5_o1, lut5_o0)(io.lut_in(5))

  // declare two posedge flip-flops
  val ff1 = Reg(resetVal = Bits("b0", 1))
  val ff2 = Reg(resetVal = Bits("b0", 1))

  // mux to select between lut5o0 and lut6
  val ff1_in = Cat(lut6_o, lut5_o0)(io.mux_configs(0))

  // ff_en is disabled while programming
  when (io.ff_en) {
    ff1 := ff1_in
```

```
28      ff2 := lut5_o1
29    }
30
31    io.lut_out := Cat(Cat(ff2, lut5_o1)(io.mux_configs(2)), Cat(ff1,
          ff1_in)(io.mux_configs(1)))
32  }
```



(a) 6-input fractured LUT (b) 6-input LUT

Figure 4.2: BLE Illustrated

4.3 Configurable Logic Block (CLB)

Several BLEs clustered into a block is a CLB. (Fig. 4.3) Previous studies has shown a CLB with three to ten BLEs has the best area-delay product. [19] Input to the CLBs is the output of a crossbar switch to be described in Section 4.4.

4.4 Crossbar Switch (XBAR)

The crossbar switch provides routing support inside a CLB. The input to the crossbar switch is output of the connection box and output of the CLB. It is a fully connected routing paradigm that is implemented using a MUXs Tree for each output signal. The implementation is clean as it is a very regular structure.

4.5 Inside A LUT Tile

A LUT Tile contains a CLB, an XBAR, a CB and a SB. It also contains a configuration memory for controlling all the components. Details on the configuration architecture are documented in Chapter 5. Fig.3.2.(b) illustrates all the components.

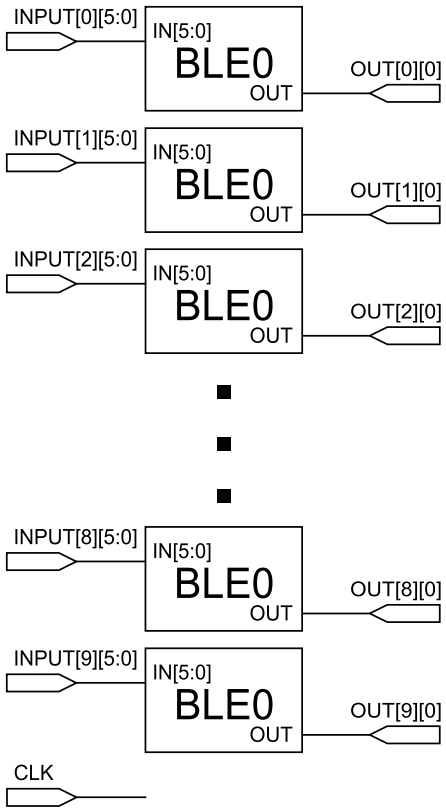


Figure 4.3: CLB Illustrated

4.6 Inside an IO Tile

The IO Tile is simple. In this work, we assume input and output of an IO Tile only originates inside a chip, so no special drivers need to be in place. It has no CLB and no XBAR, but only a CB and a SB. Configuration memory are also necessary. Detail on the configuration architecture is documented in Chapter 5. Fig.3.2.(a) illustrates all the components.

Chapter 5

Reconfiguration

> The computer would do anything you programmed it to do.
>
> – Vinton Cerf

5.1 High Level Reconfiguration Architecture

LUTs, crossbar switches and channel switches are all controlled by configuration stored in the configuration memory. On a very high level overview, the configuration memory is similar to an SRAM. However, there is no row access in reading the memory because all configurations are used at the same time. When programming the configuration memory, the writing is done on a row by row basis.

To construct the configuration memory in a logical way and to facilitate bit generation program, the configuration memory is organized into frames. Each frame contains the configuration for an entire tile. For example, an IO TILE requires a frame of configuration while a LUT TILE also requires a frame. Therefore, frames can have different sizes, but the width is always 32. This design decision simplifies both the FPGA architecture and the bitstream generator.

On the top level, frames within a column share the same Config_In signal. Frames within a row share same Config_En signal bus. Some of the signals in Config_En may not be used in all frames as their size can be different. The unused signals are synthesized away through optimizations in ASIC flow. We decide not to have any state in the configuration circuit so users can choose an implementation that fits their need. Fig. 5.1 provides a top level view of the configuration frames.

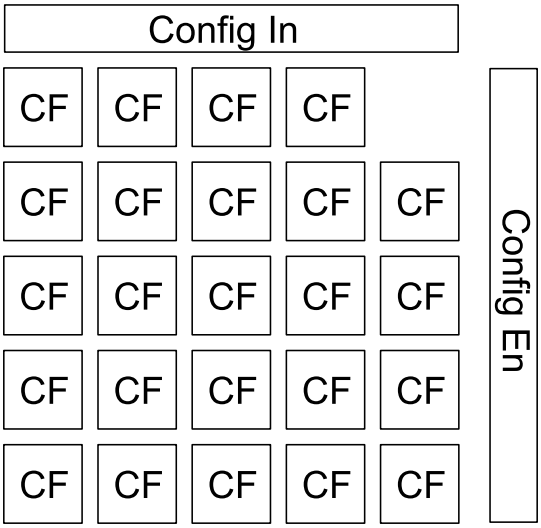


Figure 5.1: FPGA Configuration

5.2 Configuration Organization Inside Tiles

The configuration memory is a sequence of latches. We used latch in the design to minimize area overhead as a latch is about 35% smaller than a flip-flop in a 65nm process technology we use. In addition, smaller area implies shorter wires which is instrumental in obtaining high operating frequency.

Inside a LUT Tile, five segments of configurations exist logically. The first one is for LUT configuration. The second is for MUXs inside a BLE. The third is for XBAR configuration. The fourth is for CB and the fifth is for SB. Fig. 5.2 illustrate the configuration inside a LUT Tile. An IO Tile has configuration memory for CB and SB only.

The work is designed to be expandable, so we also defined configuration format for MEM Tile and MUL Tile. The first segment controls internal parameters. The rests (XBAR, CB, SB) are all common all tiles. An XBAR is not mandatory in MEM Tile and MUL Tile.

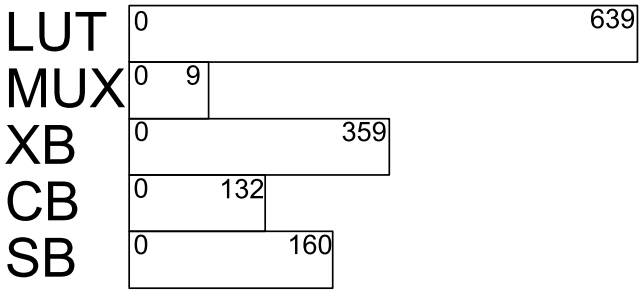


Figure 5.2: LUT Tile Configuration Organization

Chapter 6

Open Source FPGA Design Flow

> When it is obvious that the goals cannot be reached, don't adjust the goals, adjust the action steps.
>
> – Confucius

6.1 Design Flow Overview

What is more significant is the tool flow that generates hardware and software. In this section, we describe all the tools we have used. Fig. 6.1 gives an overview of the tool flow. All the tools are open source with the exception of ASIC tool flow from Open Source FPGA RTL to ASIC. We describe the ASIC Flow in Appendix B.

There are three inputs in the flow diagram, **fpga.scala**, **architecture.xml** and **user_design.v**. **fpga.scala** is a set of files that comprise as a generator for the FPGA. It is implemented in Chisel. **architecture.xml** is the description of the FPGA. It is also input to the VTR flow. [6] **user_design.v** is the user's circuit to implement.

6.2 The Open Source FPGA Flow

The core of the open source FPGA flow is the VTR Project. [6] We extend the work to add FPGA RTL configuration generation capability and bit stream generation capability. Fig. 6.2 illustrates the flow. All process inside the dashed box is from VTR Project. In the next section, we describe the process flow inside the "FPGA Gen and Bit Gen" box.

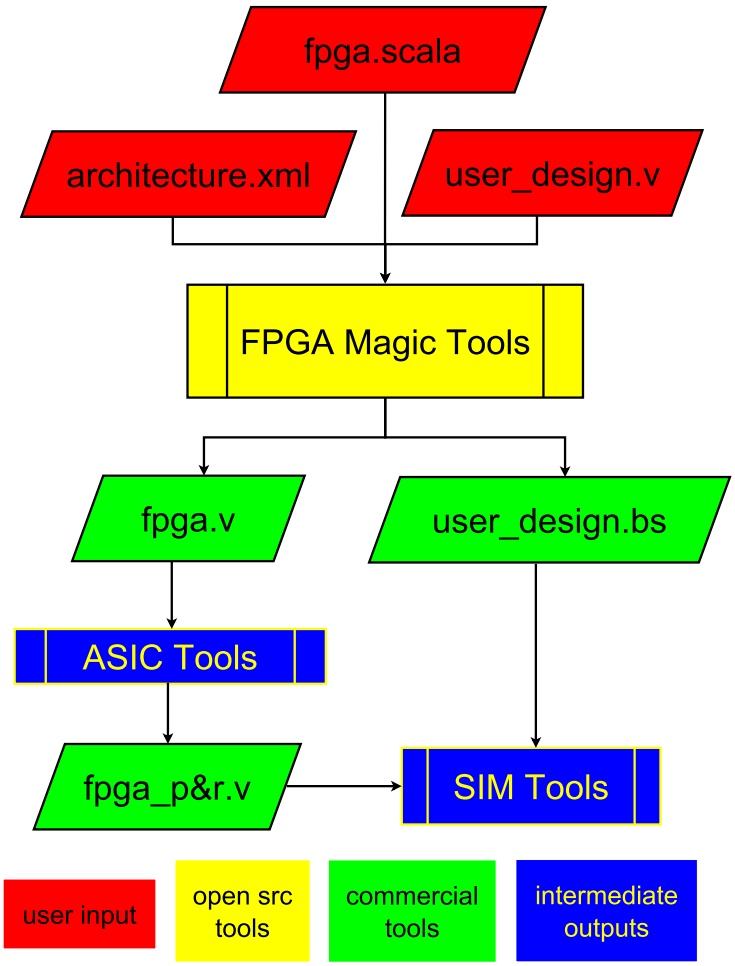


Figure 6.1: Design Flow Overview

6.3 FPGA Generation and Bitstream Generation Flow

This part is a major work in the software flow. It is implemented in python. Fig. 6.3 illustrates major steps inside the flow.

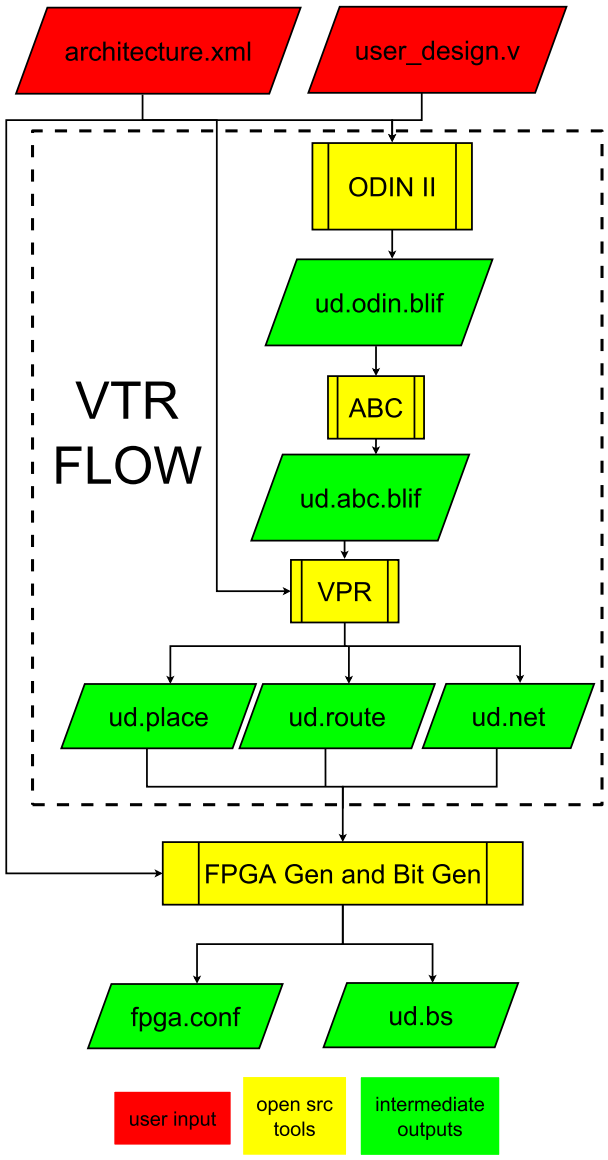


Figure 6.2: FPGA Flow

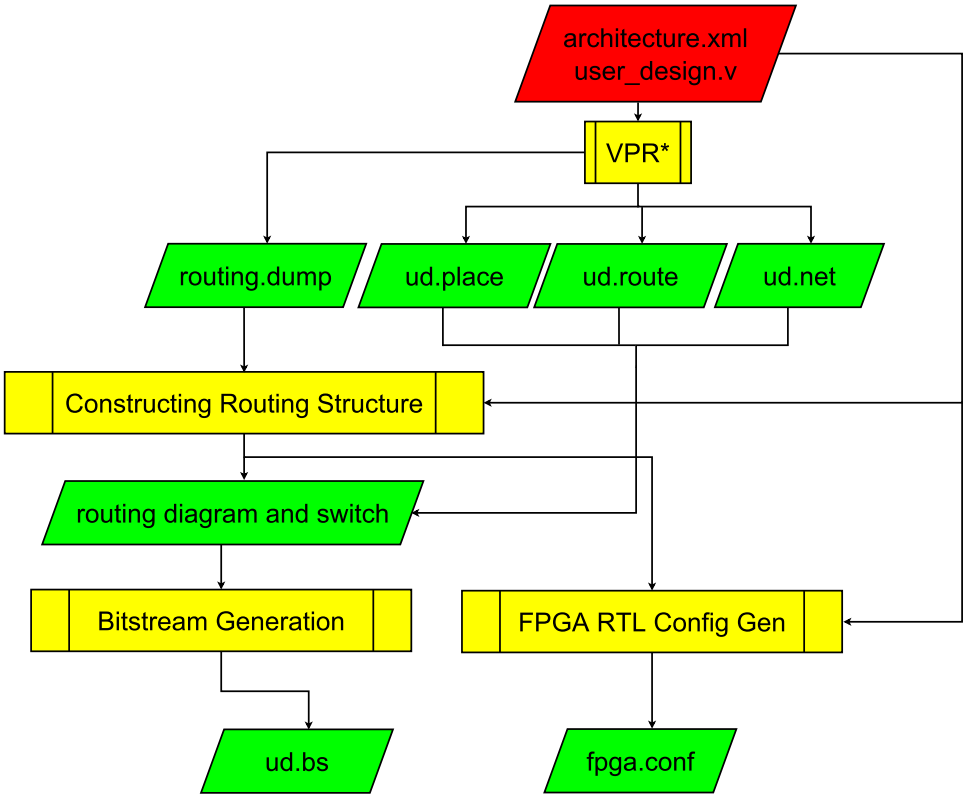


Figure 6.3: FPGA Generation and Bitstream Generation Flow

Chapter 7

Functional Testing

> Never stop testing, and your advertising will never stop improving.
>
> – David Ogilvy

7.1 Module Testing

To take advantage of Chisel's emulation capability, we simulate modules of the LUT Tile and itself as whole. The module level testing is intended to eliminate most of the bugs. It is relatively easy to provide good coverage as the number of states for the LUT Tile is small and many signals are independent of each other. This is because the only elements with states are FFs inside a BLE. To have good routability, outputs have to be independent so each of them can be configured for different signals. Although configuration latches are storage elements, the signals are treated as regular input vectors to improve module simulation speed.

7.2 System Testing

The system level testing includes two components at the same time: the hardware being generated, and the tool flow used to generate bitstream. We designed a total of eight test cases with different focuses. Table 7.1 lists the test cases and their testing objective.

To leverage generation capability of the work, we varies different user accessible parameters of the FPGA and present the parameters in Table 7.2 lists the parameters we varied. Note some combination of parameters generates a valid FPGA, but the FPGA may be too small to hold a user design.

It is possible to leverage a seed number feature in VPR for improving testing coverage. The placement and routing stage is deterministic based on a seed number

Test Case	Description
single_inv	A single bit inverter tests essential functions of LUTs and routing resources of the FPGA
single_inv_reg	A single bit inverter with registered output. It adds storage element (FF) testing to single_inv
wide_inv	A wide inverter (32 Bits) tests fundamental functions of LUTs and routing resources of the FPGA
wide_inv_reg	A wide inverter (32 Bits) with registered output. It adds storage element (FF) testing to wide_inv
ff_en	A register (32 Bits) test case for testing if inputs can be correctly stored
simple_comp	Simple computations test both computing and storage as the same time
multi_consumer	A more evolved test case of simple_comp, tests multiple sinks routing
counter	A counter (12 Bits) for testing both intra tile routing and inter tile routing

Table 7.1: System Test Cases

FPGA Size	LUT Size	CLB Size	IPIN W	CHAN W	CHAN NUM
5 by 5	6	10	16	16	160
10 by 10	5	8	12	12	120
25 by 25	4	6	8	8	80

Table 7.2: Test Parameters

provided to VPR. By varying the seed number, different bitstreams are generated. They should be equivalent circuits. More coverage on the hardware can be achieved using the same testing vectors by placing equivalent circuit bitstreams on the FPGA.

In summary, all test cases have passed if the test case fits in the FPGA. If the test case is too large for the FPGA, VPR reports an error.

Chapter 8

Performance

An ounce of performance is worth pounds of promises.

– Mae West

8.1 Tile ASIC Performance

We sweep a combination of design parameters listed in Table 8.1. In this part of the work, we only push through a tile but not the entire FPGA to reduce ASIC flow speed. It is implemented in a 65nm process technology and standard cell flow. We report and analyze critical path delay, configuration memory area, total tile area and static power. The maximum absolute operating speed is the highest operating speed of the FPGA. The real operating speed depends on circuits implemented on FPGA.

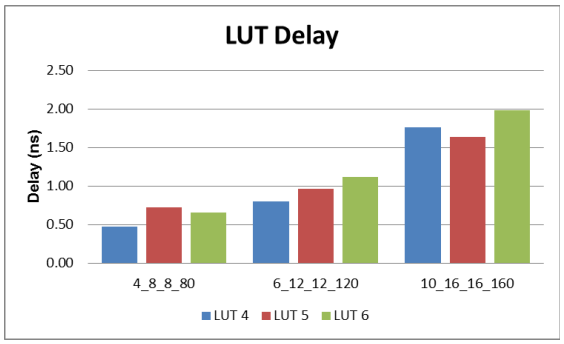
Parameters	Valid Value and Description
LUT Size	4, 5, 6
CLB Size	4, 5, 6, 7, 8, 9, 10
IPIN Width	8, 12, 16
CHANXY Width	8, 12, 16
CHANXY Num	80, 120, 160

Table 8.1: Tile Performance Parameters

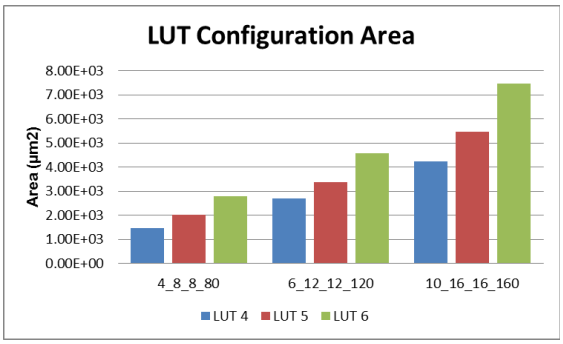
We present all of the data in tables in Appendix C for user reference, so they can select configurations based on their use cases. Following subsections only present result on varying one variable and analysis on the data trend.

8.1.1 Effect of LUT Size

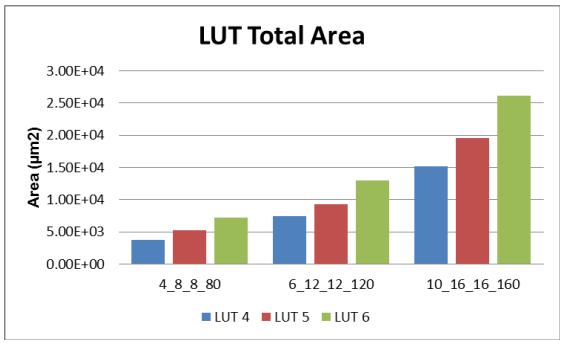
As LUT Size increases, configuration area, total area and static power all increases. The delay in Fig. 8.1.(a) varies but still displays a general trend of higher delay as LUT Size increases. We believe the variation is because variation in ASIC Tool performance. Sometimes the tool gives better QoRs.



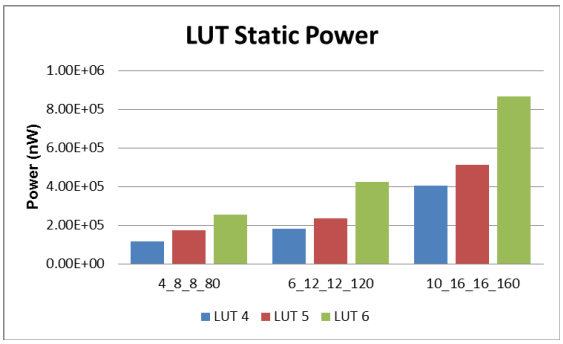
(a) Delay



(b) Configuration Area



(c) Total Area

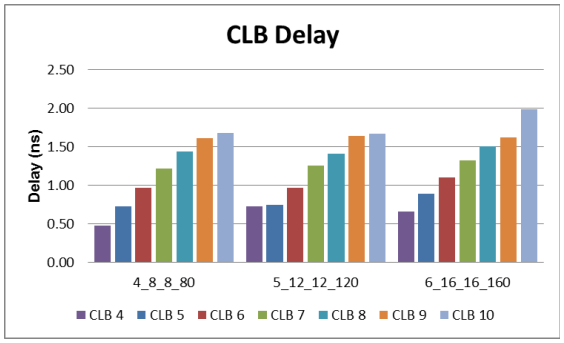


(d) Static Power

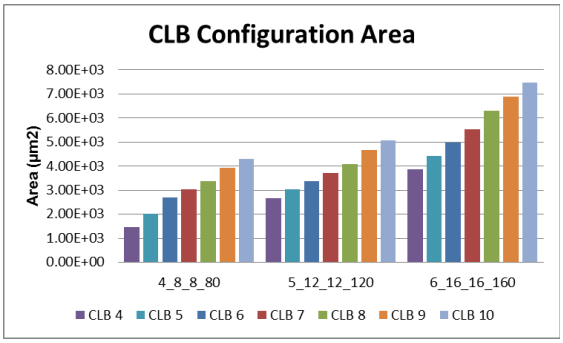
Figure 8.1: LUT SIZE Effect on Performance

8.1.2 Effect of CLB Size

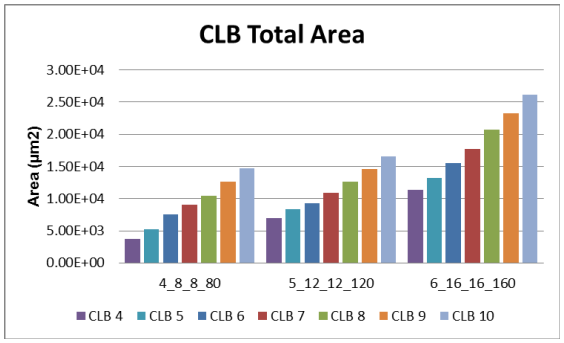
As CLB Size increases, CLB delay, configuration area, total area and static power all increases. That is within our expectation.



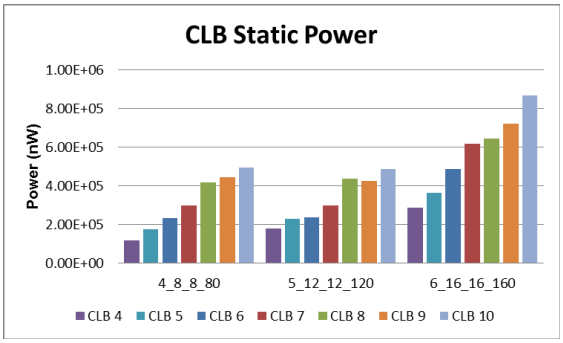
(a) Delay



(b) Configuration Area



(c) Total Area

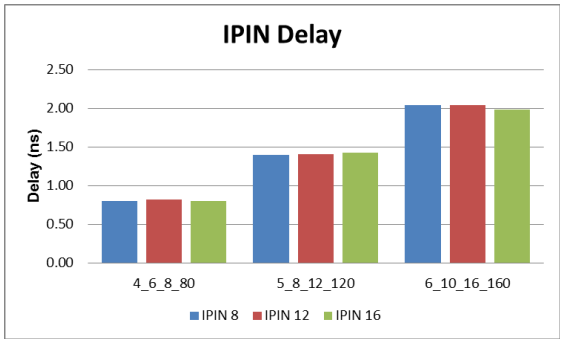


(d) Static Power

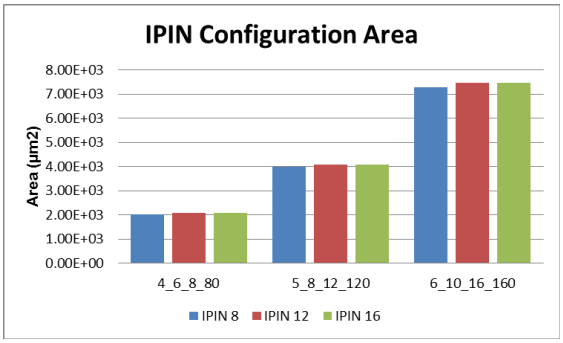
Figure 8.2: CLB SIZE Effect on Performance

8.1.3 Effect of IPIN Width

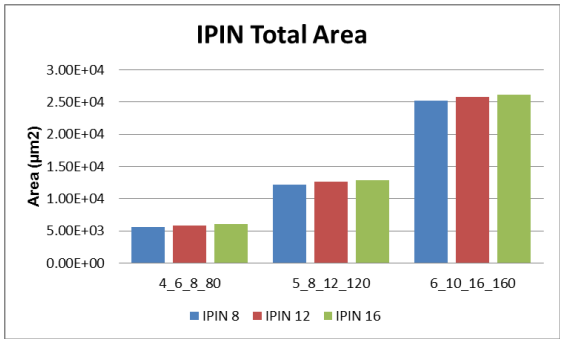
General performance trend matches our expectation: increased size leads to increased delay, area and power. We believe the slight variation is due to ASIC tool variation.



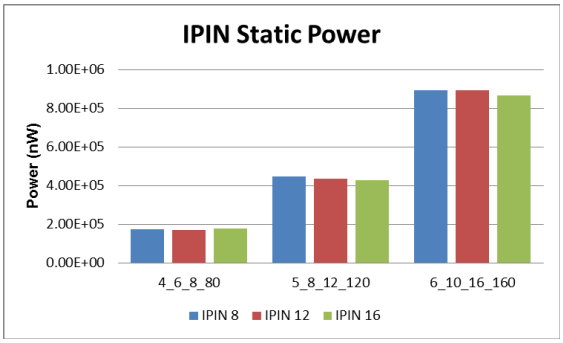
(a) Delay



(b) Configuration Area



(c) Total Area

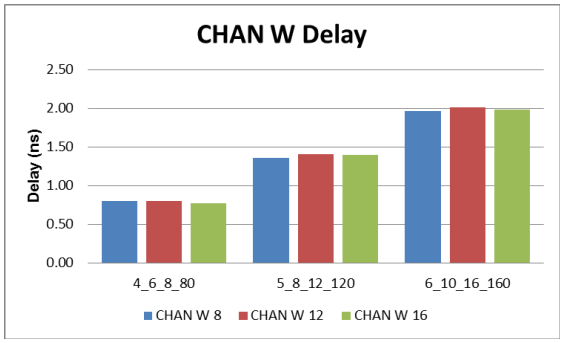


(d) Static Power

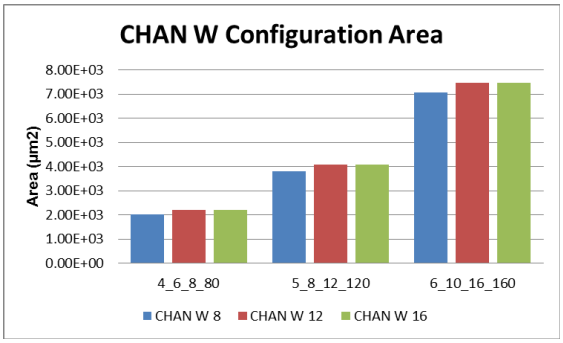
Figure 8.3: IPIN Width Effect on Performance

8.1.4 Effect of CHANXY Width

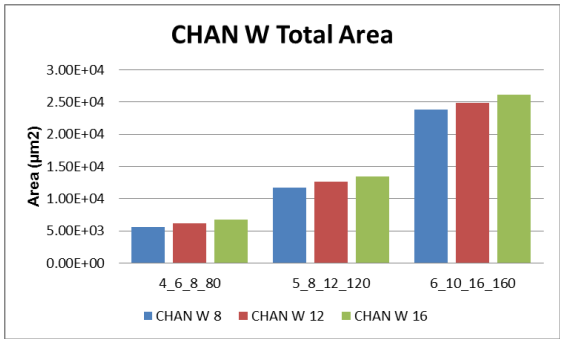
General performance trend matches our expectation: increased size leads to increased delay, area and power. We believe the slight variation is due to ASIC tool variation.



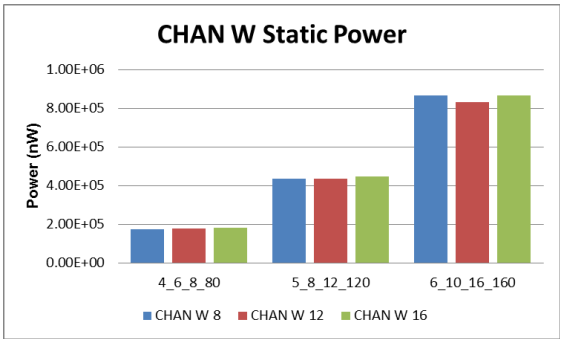
(a) Delay



(b) Configuration Area



(c) Total Area

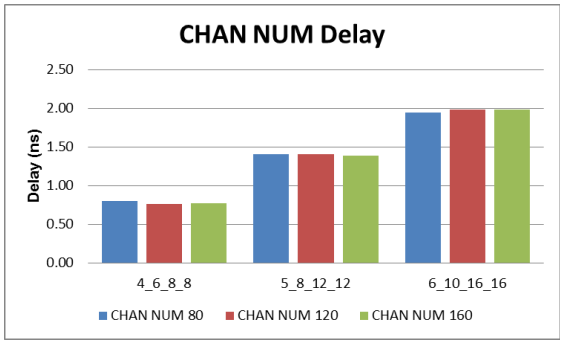


(d) Static Power

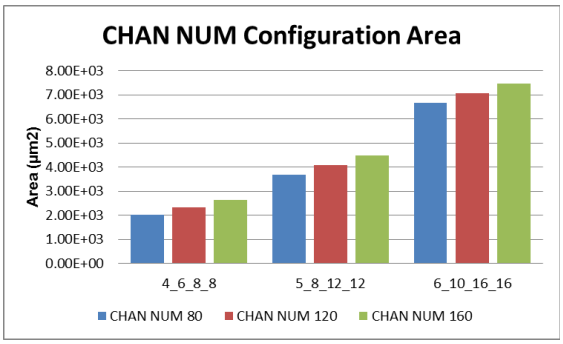
Figure 8.4: CHAN Width Effect on Performance

8.1.5 Effect of Track Width

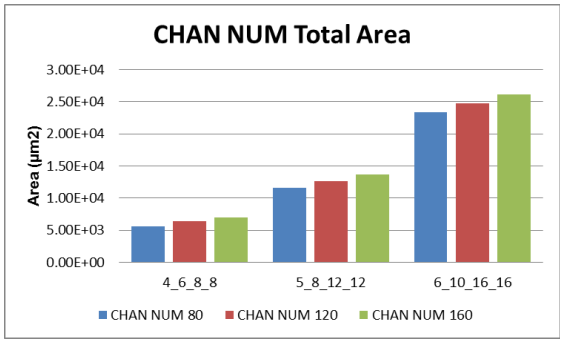
As track number increases, delay, area and power all increases. Variation in delay is caused by ASIC tool variation.



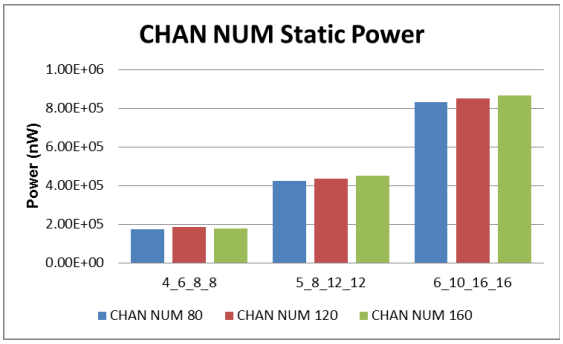
(a) Delay



(b) Configuration Area



(c) Total Area



(d) Static Power

Figure 8.5: CHAN NUM Effect on Performance

8.2 User Circuit Performance on FPGA

FPGA Size	LUT Size	CLB Size	IPIN W	CHAN W	CHAN NUM	Max Freq
5 by 5	6	10	16	16	160	416.138 MHz
10 by 10	5	8	12	12	120	377.845 MHz
25 by 25	4	6	8	8	80	339.451 MHz

Table 8.2: 64 Bit Counter Performance

We feed VPR with three sets of parameters and a 64 Bit counter to test the performance of the FPGA. Table 8.2 lists the maximum operating speed on three FPGAs we tested. As the goal of the project is not to aim the best performance, we

report the performance number only for demonstrating the operating speed is good enough for real applications and presenting user some references so they can make their own decisions in selecting architectural parameters.

Chapter 9

Conclusion and Future Works

> Prediction is very difficult, especially about the future.
>
> – Niels Bohr

9.1 Conclusion

In conclusion, we designed and implemented an Open FPGA with tool support. The design is extensible. We verified its' functionality. The performance of the work is good enough for real world work load.

9.2 Future Works

The current implementation has only LUT Tiles. To have a more powerful Open FPGA, the MEM Tile and MUL Tile with tool support should be implemented. In addition to more hardware features, an enhanced tool flow that supports intermediate step verification can help user debug the tool flow. We would also like to know how technology scaling going to affect the QoR of this work.

References

[1] M. Saldaña, A. Patel, C. Madill, D. Nunes, D. Wang, P. Chow, R. Wittig, H. Styles, and A. Putnam, "Mpi as a programming model for high-performance reconfigurable computers," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 3, no. 4, pp. 22:1–22:29, Nov. 2010. [Online]. Available: http://doi.acm.org/10.1145/1862648.1862652

[2] C. Chang, "Design and applications of a reconfigurable computing system for high performance digital signal processing," Ph.D. dissertation, University of California at Berkeley, Berkeley, CA, USA, 2005, aAI3210531.

[3] Xilinx. (2011, August) Xilinx zynq-7000 epp. [Online]. Available: http://www.xilinx.com/innovation/research-labs/conferences/hotchips23-wittig.pdf

[4] V. Betz and J. Rose, "Vpr: A new packing, placement and routing tool for fpga research," in *Proceedings of the 7th International Workshop on Field-Programmable Logic and Applications*, ser. FPL '97. London, UK, UK: Springer-Verlag, 1997, pp. 213–222. [Online]. Available: http://dl.acm.org/citation.cfm?id=647924.738755

[5] J. Wawrzynek, "Should the academic community launch an open-source fpga device and tools effort?: evening panel," in *Proceedings of the 19th ACM/SIGDA international symposium on Field programmable gate arrays*, ser. FPGA '11. New York, NY, USA: ACM, 2011, pp. 3–4. [Online]. Available: http://doi.acm.org/10.1145/1950413.1950417

[6] J. Rose, J. Luu, C. W. Yu, O. Densmore, J. Goeders, A. Somerville, K. B. Kent, P. Jamieson, and J. Anderson, "The vtr project: architecture and cad for fpgas from verilog to routing," in *Proceedings of the ACM/SIGDA international symposium on Field Programmable Gate Arrays*, ser. FPGA '12. New York, NY, USA: ACM, 2012, pp. 77–86. [Online]. Available: http://doi.acm.org/10.1145/2145694.2145708

[7] P. Jamieson, K. B. Kent, F. Gharibian, and L. Shannon, "Odin ii - an open-source verilog hdl synthesis tool for cad research," in *Proceedings of the 2010 18th IEEE Annual International Symposium on Field-Programmable Custom Computing Machines*, ser. FCCM '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 149–156. [Online]. Available: http://dx.doi.org/10.1109/FCCM.2010.31

[8] R. Brayton and A. Mishchenko, "Abc: an academic industrial-strength verification tool," in *Proceedings of the 22nd international conference on Computer Aided Verification*, ser. CAV'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 24–40. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-14295-6_5

[9] J. Luu, I. Kuon, P. Jamieson, T. Campbell, A. Ye, W. M. Fang, K. Kent, and J. Rose, "Vpr 5.0: Fpga cad and architecture exploration tools with single-driver routing, heterogeneity and process scaling," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 4, no. 4, pp. 32:1–32:23, Dec. 2011. [Online]. Available: http://doi.acm.org/10.1145/2068716.2068718

[10] K. Padalia, R. Fung, M. Bourgeault, A. Egier, and J. Rose, "Automatic transistor and physical design of fpga tiles from an architectural specification," in *Proceedings of the 2003 ACM/SIGDA eleventh international symposium on Field programmable gate arrays*, ser. FPGA '03. New York, NY, USA: ACM, 2003, pp. 164–172. [Online]. Available: http://doi.acm.org/10.1145/611817.611842

[11] B. Neumann, T. von Sydow, H. Blume, and T. G. Noll, "Design flow for embedded fpgas based on a flexible architecture template," in *Proceedings of the conference on Design, automation and test in Europe*, ser. DATE '08. New York, NY, USA: ACM, 2008, pp. 56–61. [Online]. Available: http://doi.acm.org/10.1145/1403375.1403391

[12] S. Chaudhuri, S. Guilley, F. Flament, P. Hoogvorst, and J.-L. Danger, "An 8x8 run-time reconfigurable fpga embedded in a soc," in *Proceedings of the 45th annual Design Automation Conference*, ser. DAC '08. New York, NY, USA: ACM, 2008, pp. 120–125. [Online]. Available: http://doi.acm.org/10.1145/1391469.1391500

[13] X. Chen and J. Zhu, "Regular fabric for regular fpga (abstract only)," in *Proceedings of the 19th ACM/SIGDA international symposium on Field programmable gate arrays*, ser. FPGA '11. New York, NY, USA: ACM, 2011, pp. 284–284. [Online]. Available: http://doi.acm.org/10.1145/1950413.1950484

[14] I. Kuon, A. Egier, and J. Rose, "Design, layout and verification of an fpga using automated tools," in *Proceedings of the 2005 ACM/SIGDA 13th international symposium on Field-programmable gate arrays*, ser. FPGA '05. New York, NY, USA: ACM, 2005, pp. 215–226. [Online]. Available: http://doi.acm.org/10.1145/1046192.1046220

[15] J. Bachrach, H. Vo, B. Richards, Y. Lee, A. Waterman, R. Avižienis, J. Wawrzynek, and K. Asanović, "Chisel: constructing hardware in a scala embedded language," in *Proceedings of the 49th Annual Design Automation Conference*, ser. DAC '12. New York, NY, USA: ACM, 2012, pp. 1216–1225. [Online]. Available: http://doi.acm.org/10.1145/2228488.2228584

[16] Xilinx. (2004, June) Xilinx unveils virtex-4âĎć fpga family. [Online]. Available: http://www.xilinx.com/company/press/kits/v4_arch/v4arch_editorpres_final3.pdf

[17] P. Jamieson, W. Luk, S. J. E. Wilton, and G. Constantinides, "An energy and power consumption analysis of fpga routing architectures," in *Field-Programmable Technology, 2009. FPT 2009. International Conference on*, 2009, pp. 324–327.

[18] G. Lemieux, E. Lee, M. Tom, and A. Yu, "Directional and single-driver wires in fpga interconnect," in *Field-Programmable Technology, 2004. Proceedings. 2004 IEEE International Conference on*, 2004, pp. 41–48.

[19] E. Ahmed and J. Rose, "The effect of lut and cluster size on deep-submicron fpga performance and density," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 12, no. 3, pp. 288–298, 2004.

[20] V. Betz and J. Rose, "How much logic should go in an fpga logic block," *Design Test of Computers, IEEE*, vol. 15, no. 1, pp. 10–15, 1998.

Appendix A

User Controlled Parameters

A.1 LUT and CLB Parameters

Parameters	Valid Value	Description
LUT Size	4, 5, 6	Number of Inputs to a LUT
CLB Size	4 to 10	Also called cluster size, Number of LUT inside a LUT Tile

Table A.1: LUT and CLB Parameters

A.2 Routing Parameters

Parameters	Valid Value	Description
IPIN Num	$\geq$ LUT Size $\leq$ (LUT Size)$*$(CLB Size)	Number of inputs from routing tracks to XBAR, in other word, the width of CB. Research has shown a value of 0.5$\times$(LUT Size)$\times$(CLB Size)+3 provides best design trade-off. [20]
IPIN Width	8 to 16	Number of routing tracks driving an IPIN, The circuit is a MUX tree. Too small causes routability problem. Too large increases delay.
CHANXY Num	40, 60, 80, 100, 120, 140, 160	Number of routing tracks driven inside a LUT Tile. Must be divisible by four so four directions have an equal number of tracks.
CHANXY Width	8 to 16	Number of routing tracks driving a routing track, The circuit is a MUX tree. Too small causes routability problem. Too large increases delay.
Segmentation	1 to 8	Please reference Sec. 3.3.1 for a description.

Table A.2: Routing Parameters

A.3 IO Parameters

IO Parameters share the same configuration for *CHANXY Num* and *CHANXY Input Width* in a LUT Tile.

Parameters	Valid Value	Description
Input Num	1 to 16	Number of input connections driven by signals outside of the FPGA. Not meaningful if greater than one fourth of the number of tracks.
Output Num	1 to 16	Number of output connections driving signals outside of the FPGA. Not meaningful if greater than one fourth of the number of tracks.

Table A.3: IO Parameters

A.4 Whole FPGA Parameters

The X Size and Y Size of the FPGA should be similar to create a square shape in layout for best corner to corner routing performance. Absolute minimum is *Segmentation*+1. There is no absolute maximum. 25 by 25 is the largest we have ever tested.

Appendix B

Tile Generation Flow

B.1 ASIC Flow Overview

Input to a ASIC flow is RTL, in this case, in Verilog HDL. Constraints set Quality of Result (QoR) targets. Output of an ASIC flow is layout. (Fig. B.1)

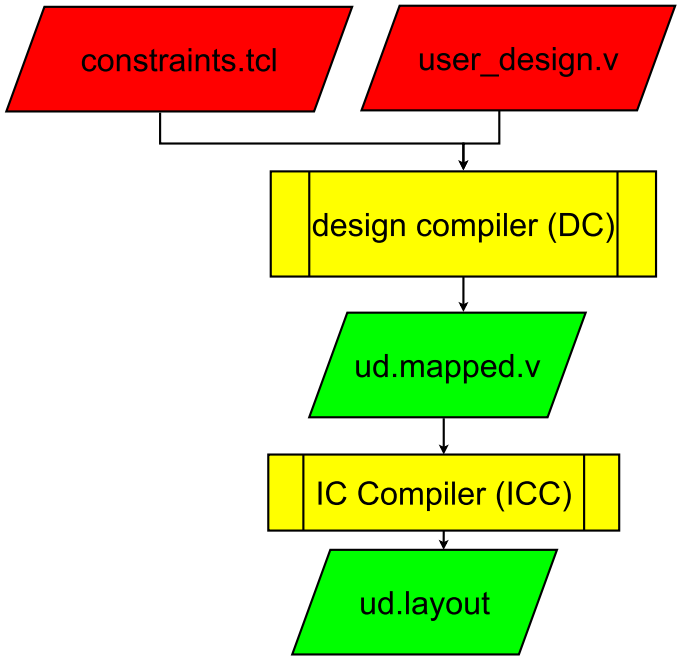


Figure B.1: ASIC Flow Overview

B.2 MIM Flow Using Synopsys IC Compiler

The Multiple Instantiated Module (MIM) is a flow that takes advantage of modules with same RTL and provides faster placement and routing and preserves QoR. Placing a large design can take several days. The MIM flow requires a large design that contains many identical modules. In the MIM flow, the tool first performs global planning and places all the identical modules. It then performs local placement and routing on only one of the module within the identical module group so the placement and routing step is only executed once and available for all. This reduces physical implementation time.

Appendix C

Tile Performance

C.1 Tile Performance Description

We present all the data collected for LUT Tiles with different parameters. We present them in different sections according to LUT Size and CLB Size. The unit of critical path delay is ns; The unit of area is μm^2; The unit of power is nW.

C.2 ASIC Performance for LUT SIZE 4

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	0.48	1479.95	3814.20	1.18e+05
8	8	120	0.49	1781.99	4514.04	1.21e+05
8	8	160	0.48	2085.11	5244.84	1.30e+05
8	12	80	0.47	1681.19	4426.56	1.17e+05
8	12	120	0.48	2082.95	5433.84	1.22e+05
8	12	160	0.49	2487.59	6449.76	1.31e+05
8	16	80	0.48	1682.63	4993.20	1.25e+05
8	16	120	0.48	2084.75	6236.28	1.32e+05
8	16	160	0.48	2488.67	7506.00	1.44e+05
12	8	80	0.46	1532.51	3997.80	1.10e+05
12	8	120	0.47	1837.07	4746.24	1.25e+05
12	8	160	0.47	2139.83	5412.96	1.24e+05
12	12	80	0.48	1735.91	4634.28	1.19e+05
12	12	120	0.47	2140.19	5627.52	1.23e+05
12	12	160	0.48	2543.03	6580.44	1.30e+05
12	16	80	0.48	1736.63	5161.32	1.21e+05
12	16	120	0.48	2140.19	6463.80	1.37e+05
12	16	160	0.48	2544.47	7692.48	1.38e+05
16	8	80	0.47	1532.15	4107.24	1.12e+05
16	8	120	0.47	1838.51	4806.00	1.11e+05
16	8	160	0.47	2139.47	5545.80	1.22e+05
16	12	80	0.47	1736.63	4753.08	1.13e+05
16	12	120	0.47	2139.11	5756.76	1.19e+05
16	12	160	0.47	2542.31	6800.40	1.37e+05
16	16	80	0.47	1736.63	5275.08	1.16e+05
16	16	120	0.47	2140.91	6595.56	1.36e+05
16	16	160	0.48	2543.39	7824.24	1.41e+05

Table C.1: Tile Performance, LUT Size 4, CLB Size 4

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	0.73	1786.31	4702.68	1.53e+05
8	8	120	0.72	2087.99	5468.40	1.71e+05
8	8	160	0.71	2392.55	6189.12	1.81e+05
8	12	80	0.75	1988.27	5409.00	1.74e+05
8	12	120	0.74	2389.67	6387.84	1.69e+05
8	12	160	0.72	2792.15	7326.00	1.67e+05
8	16	80	0.72	1989.71	5879.52	1.68e+05
8	16	120	0.71	2391.47	7133.04	1.72e+05
8	16	160	0.71	2794.67	8416.80	1.85e+05
12	8	80	0.73	1851.11	4981.68	1.67e+05
12	8	120	0.74	2152.43	5689.08	1.70e+05
12	8	160	0.72	2455.91	6455.52	1.81e+05
12	12	80	0.75	2050.91	5546.16	1.56e+05
12	12	120	0.73	2455.19	6561.36	1.65e+05
12	12	160	0.74	2858.03	7620.48	1.76e+05
12	16	80	0.74	2053.43	6212.88	1.81e+05
12	16	120	0.73	2455.55	7364.52	1.76e+05
12	16	160	0.73	2856.95	8614.80	1.82e+05
16	8	80	0.74	1852.19	5128.56	1.62e+05
16	8	120	0.72	2152.07	5814.72	1.70e+05
16	8	160	0.71	2455.91	6538.32	1.74e+05
16	12	80	0.73	2051.27	5705.64	1.59e+05
16	12	120	0.72	2455.19	6703.20	1.67e+05
16	12	160	0.74	2855.87	7733.52	1.72e+05
16	16	80	0.74	2052.71	6261.12	1.67e+05
16	16	120	0.74	2454.11	7542.72	1.73e+05
16	16	160	0.72	2856.23	8843.76	1.91e+05

Table C.2: Tile Performance, LUT Size 4, CLB Size 5

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	0.80	2017.07	5604.48	1.77e+05
8	8	120	0.77	2320.19	6366.60	1.89e+05
8	8	160	0.78	2620.43	7020.36	1.80e+05
8	12	80	0.80	2216.15	6207.12	1.81e+05
8	12	120	0.79	2619.71	7174.44	1.76e+05
8	12	160	0.79	3022.55	8260.20	1.89e+05
8	16	80	0.78	2217.59	6754.68	1.82e+05
8	16	120	0.78	2621.15	8058.96	1.96e+05
8	16	160	0.79	3025.43	9324.72	2.06e+05
12	8	80	0.82	2090.51	5811.48	1.71e+05
12	8	120	0.82	2392.91	6513.12	1.81e+05
12	8	160	0.79	2696.39	7236.36	1.86e+05
12	12	80	0.80	2291.39	6450.84	1.73e+05
12	12	120	0.80	2696.03	7455.96	1.83e+05
12	12	160	0.81	3098.15	8442.00	1.89e+05
12	16	80	0.81	2295.35	7011.00	1.88e+05
12	16	120	0.80	2696.03	8247.60	1.95e+05
12	16	160	0.80	3099.59	9531.36	2.00e+05
16	8	80	0.80	2093.75	6035.76	1.80e+05
16	8	120	0.78	2397.23	6864.84	1.97e+05
16	8	160	0.78	2696.39	7512.12	1.99e+05
16	12	80	0.80	2293.91	6693.84	1.85e+05
16	12	120	0.81	2695.31	7700.04	1.82e+05
16	12	160	0.79	3096.35	8666.28	1.88e+05
16	16	80	0.78	2294.99	7296.12	1.95e+05
16	16	120	0.79	2697.11	8533.80	2.02e+05
16	16	160	0.80	3099.95	9788.76	2.15e+05

Table C.3: Tile Performance, LUT Size 4, CLB Size 6

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	0.98	2234.87	6557.40	2.02e+05
8	8	120	0.98	2536.91	7328.16	2.07e+05
8	8	160	0.98	2844.35	8058.24	2.23e+05
8	12	80	0.98	2433.59	7183.08	2.10e+05
8	12	120	1.01	2842.91	8191.44	2.11e+05
8	12	160	0.99	3241.07	9237.24	2.19e+05
8	16	80	0.99	2438.63	7655.04	2.05e+05
8	16	120	0.98	2845.07	9077.04	2.28e+05
8	16	160	1.00	3242.87	10262.88	2.36e+05
12	8	80	1.01	2322.35	6897.24	2.04e+05
12	8	120	1.00	2620.07	7474.32	1.96e+05
12	8	160	0.98	2925.71	8357.40	2.23e+05
12	12	80	1.01	2524.31	7432.92	2.07e+05
12	12	120	1.01	2929.31	8363.16	2.00e+05
12	12	160	1.00	3326.75	9436.32	2.11e+05
12	16	80	0.99	2521.79	7928.64	2.07e+05
12	16	120	0.99	2926.43	9298.80	2.22e+05
12	16	160	0.99	3329.63	10464.12	2.30e+05
16	8	80	0.97	2321.99	7057.08	2.00e+05
16	8	120	0.97	2621.87	7817.04	2.15e+05
16	8	160	0.98	2927.15	8481.24	2.16e+05
16	12	80	0.99	2519.99	7607.88	2.03e+05
16	12	120	0.98	2926.07	8616.24	2.05e+05
16	12	160	0.99	3328.91	9621.00	2.11e+05
16	16	80	0.95	2521.43	8253.36	2.16e+05
16	16	120	0.96	2923.55	9532.80	2.25e+05
16	16	160	0.98	3329.99	10841.40	2.45e+05

Table C.4: Tile Performance, LUT Size 4, CLB Size 7

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	1.37	2467.43	7913.88	2.83e+05
8	8	120	1.36	2770.19	8678.16	3.12e+05
8	8	160	1.33	3069.35	9333.00	2.96e+05
8	12	80	1.47	2663.99	8337.96	2.62e+05
8	12	120	1.37	3070.07	9527.40	3.08e+05
8	12	160	1.44	3471.47	10470.24	2.83e+05
8	16	80	1.38	2671.91	9050.76	2.94e+05
8	16	120	1.37	3072.59	10382.40	3.07e+05
8	16	160	1.36	3481.91	11656.08	3.26e+05
12	8	80	1.39	2559.23	8197.56	2.83e+05
12	8	120	1.43	2867.39	8839.80	2.76e+05
12	8	160	1.39	3166.91	9630.72	2.94e+05
12	12	80	1.47	2761.55	8798.76	2.92e+05
12	12	120	1.43	3164.75	9778.68	2.80e+05
12	12	160	1.46	3564.35	10706.76	2.79e+05
12	16	80	1.39	2763.71	9295.92	2.84e+05
12	16	120	1.38	3167.99	10663.20	3.18e+05
12	16	160	1.37	3569.39	11888.28	3.07e+05
16	8	80	1.34	2563.19	8445.60	2.93e+05
16	8	120	1.35	2864.15	9133.20	2.87e+05
16	8	160	1.36	3167.99	9913.68	3.03e+05
16	12	80	1.41	2764.07	9010.80	2.93e+05
16	12	120	1.42	3166.19	10024.56	2.89e+05
16	12	160	1.45	3567.95	10989.00	2.80e+05
16	16	80	1.42	2763.35	9569.88	2.93e+05
16	16	120	1.37	3171.95	10804.32	3.01e+05
16	16	160	1.35	3574.07	12119.76	3.14e+05

Table C.5: Tile Performance, LUT Size 4, CLB Size 8

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	1.44	2692.79	9214.92	3.61e+05
8	8	120	1.49	2994.11	9910.80	3.54e+05
8	8	160	1.47	3296.87	10622.16	3.57e+05
8	12	80	1.40	2891.15	9747.72	3.34e+05
8	12	120	1.43	3298.67	10869.48	3.66e+05
8	12	160	1.46	3700.43	11899.80	3.84e+05
8	16	80	1.41	2896.19	10320.84	3.59e+05
8	16	120	1.45	3303.71	11672.64	3.73e+05
8	16	160	1.44	3713.03	12890.52	3.90e+05
12	8	80	1.41	2800.07	9545.40	3.51e+05
12	8	120	1.44	3103.55	10291.68	3.73e+05
12	8	160	1.44	3403.07	10904.76	3.54e+05
12	12	80	1.46	2996.63	10158.12	3.54e+05
12	12	120	1.41	3399.47	11181.24	3.58e+05
12	12	160	1.41	3806.99	12212.64	3.69e+05
12	16	80	1.43	3005.63	10663.20	3.57e+05
12	16	120	1.44	3407.39	11970.36	3.71e+05
12	16	160	1.39	3806.99	13190.04	3.73e+05
16	8	80	1.45	2799.35	9860.04	3.75e+05
16	8	120	1.40	3093.47	10594.44	3.73e+05
16	8	160	1.42	3407.75	11272.68	3.76e+05
16	12	80	1.45	3002.75	10523.52	3.79e+05
16	12	120	1.46	3398.39	11475.72	3.68e+05
16	12	160	1.44	3804.83	12601.80	3.80e+05
16	16	80	1.44	3000.95	11000.88	3.80e+05
16	16	120	1.42	3407.75	12288.60	3.84e+05
16	16	160	1.49	3806.63	13540.32	3.89e+05

Table C.6: Tile Performance, LUT Size 4, CLB Size 9

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	1.77	3113.63	10722.60	3.64e+05
8	8	120	1.77	3418.55	11575.80	3.92e+05
8	8	160	1.72	3720.95	12065.04	3.49e+05
8	12	80	1.80	3318.83	11414.88	3.81e+05
8	12	120	1.80	3722.75	12270.24	3.62e+05
8	12	160	1.77	4125.23	13357.44	3.70e+05
8	16	80	1.80	3318.83	11869.56	3.81e+05
8	16	120	1.78	3723.47	13133.88	3.73e+05
8	16	160	1.74	4120.91	14473.08	3.89e+05
12	8	80	1.78	3234.23	11011.68	3.55e+05
12	8	120	1.76	3538.07	11643.84	3.48e+05
12	8	160	1.76	3842.27	12528.72	3.68e+05
12	12	80	1.74	3438.35	11613.96	3.72e+05
12	12	120	1.76	3836.15	12653.64	3.70e+05
12	12	160	1.73	4240.79	13697.64	3.77e+05
12	16	80	1.75	3437.99	12178.44	3.64e+05
12	16	120	1.73	3837.59	13434.84	3.84e+05
12	16	160	1.73	4240.07	14844.24	4.05e+05
16	8	80	1.77	3237.11	11371.32	3.77e+05
16	8	120	1.79	3532.67	12098.52	3.79e+05
16	8	160	1.77	3837.95	12923.64	4.05e+05
16	12	80	1.78	3431.87	11989.44	3.68e+05
16	12	120	1.82	3833.99	13171.68	3.98e+05
16	12	160	1.79	4237.19	14097.24	3.75e+05
16	16	80	1.77	3432.23	12562.92	3.79e+05
16	16	120	1.75	3836.51	13925.16	4.05e+05
16	16	160	1.77	4239.71	15148.80	4.06e+05

Table C.7: Tile Performance, LUT Size 4, CLB Size 10

C.3 ASIC Performance for LUT SIZE 5

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	0.73	2012.75	5208.84	1.77e+05
8	8	120	0.72	2315.15	5899.32	1.87e+05
8	8	160	0.72	2615.75	6579.36	1.82e+05
8	12	80	0.75	2211.83	5771.52	1.71e+05
8	12	120	0.75	2615.03	6822.72	1.85e+05
8	12	160	0.74	3018.59	7761.24	1.87e+05
8	16	80	0.72	2212.19	6304.32	1.81e+05
8	16	120	0.73	2615.75	7619.04	1.96e+05
8	16	160	0.72	3019.31	8887.32	2.10e+05
12	8	80	0.76	2077.19	5361.84	1.71e+05
12	8	120	0.72	2379.95	6047.64	1.76e+05
12	8	160	0.74	2680.55	6765.84	1.84e+05
12	12	80	0.73	2277.71	6003.00	1.80e+05
12	12	120	0.73	2680.91	6964.92	1.80e+05
12	12	160	0.75	3083.75	8028.72	1.86e+05
12	16	80	0.72	2278.43	6591.96	1.92e+05
12	16	120	0.73	2681.63	7799.04	1.92e+05
12	16	160	0.72	3084.11	9107.64	2.02e+05
16	8	80	0.73	2077.19	5515.20	1.68e+05
16	8	120	0.74	2379.95	6219.36	1.75e+05
16	8	160	0.74	2682.35	7010.64	1.87e+05
16	12	80	0.73	2277.71	6104.52	1.61e+05
16	12	120	0.75	2680.19	7140.60	1.72e+05
16	12	160	0.75	3084.11	8185.68	1.89e+05
16	16	80	0.73	2277.71	6757.92	1.80e+05
16	16	120	0.72	2681.99	8028.00	2.02e+05
16	16	160	0.73	3085.19	9262.80	1.95e+05

Table C.8: Tile Performance, LUT Size 5, CLB Size 4

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	0.75	2347.55	6428.52	2.11e+05
8	8	120	0.73	2650.31	7233.48	2.47e+05
8	8	160	0.75	2953.43	7895.52	2.35e+05
8	12	80	0.75	2548.43	7144.92	2.38e+05
8	12	120	0.75	2951.63	8142.48	2.34e+05
8	12	160	0.75	3354.83	9138.96	2.40e+05
8	16	80	0.74	2549.87	7643.52	2.34e+05
8	16	120	0.75	2953.79	8892.72	2.40e+05
8	16	160	0.73	3355.91	10051.20	2.35e+05
12	8	80	0.73	2423.51	6845.04	2.41e+05
12	8	120	0.75	2725.91	7469.64	2.32e+05
12	8	160	0.75	3027.95	8083.08	2.24e+05
12	12	80	0.77	2623.67	7355.88	2.30e+05
12	12	120	0.75	3027.23	8357.04	2.30e+05
12	12	160	0.75	3430.07	9418.68	2.56e+05
12	16	80	0.73	2624.75	7843.32	2.31e+05
12	16	120	0.74	3029.39	9253.44	2.65e+05
12	16	160	0.75	3431.15	10455.84	2.52e+05
16	8	80	0.75	2422.79	6914.52	2.32e+05
16	8	120	0.76	2724.11	7577.64	2.27e+05
16	8	160	0.74	3027.23	8326.08	2.38e+05
16	12	80	0.78	2623.31	7416.00	2.16e+05
16	12	120	0.75	3025.79	8484.12	2.29e+05
16	12	160	0.77	3429.35	9517.32	2.33e+05
16	16	80	0.75	2622.59	8076.60	2.35e+05
16	16	120	0.74	3027.59	9363.96	2.47e+05
16	16	160	0.76	3430.79	10531.08	2.47e+05

Table C.9: Tile Performance, LUT Size 5, CLB Size 5

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	0.97	2695.67	7558.20	2.35e+05
8	8	120	0.96	2997.35	8365.32	2.54e+05
8	8	160	0.96	3301.19	9093.60	2.63e+05
8	12	80	0.96	2896.91	8154.72	2.43e+05
8	12	120	0.97	3301.19	9159.12	2.47e+05
8	12	160	0.99	3700.43	10117.08	2.53e+05
8	16	80	0.94	2897.27	8667.00	2.43e+05
8	16	120	0.95	3300.11	9901.80	2.42e+05
8	16	160	0.94	3704.03	11239.92	2.68e+05
12	8	80	0.95	2786.03	7770.60	2.32e+05
12	8	120	0.97	3088.43	8597.88	2.60e+05
12	8	160	0.96	3390.83	9310.32	2.65e+05
12	12	80	1.00	2986.19	8380.44	2.47e+05
12	12	120	0.97	3388.67	9345.60	2.39e+05
12	12	160	0.95	3794.03	10507.32	2.59e+05
12	16	80	0.95	2988.35	8983.08	2.70e+05
12	16	120	0.97	3388.67	10245.24	2.58e+05
12	16	160	0.95	3794.03	11547.00	2.67e+05
16	8	80	0.94	2787.11	8052.84	2.47e+05
16	8	120	0.97	3090.59	8804.52	2.56e+05
16	8	160	0.99	3391.19	9402.12	2.47e+05
16	12	80	0.97	2987.63	8517.96	2.28e+05
16	12	120	0.97	3391.55	9565.92	2.46e+05
16	12	160	0.98	3793.31	10672.92	2.63e+05
16	16	80	0.94	2989.79	9245.88	2.62e+05
16	16	120	0.94	3391.19	10477.44	2.56e+05
16	16	160	0.93	3795.47	11765.16	2.72e+05

Table C.10: Tile Performance, LUT Size 5, CLB Size 6

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	1.22	3023.99	9038.16	2.98e+05
8	8	120	1.19	3325.67	9675.00	2.99e+05
8	8	160	1.16	3629.51	10662.84	3.43e+05
8	12	80	1.21	3224.87	9643.32	3.17e+05
8	12	120	1.23	3626.99	10632.24	3.04e+05
8	12	160	1.20	4031.99	11725.92	3.37e+05
8	16	80	1.17	3226.67	10309.68	3.31e+05
8	16	120	1.21	3630.59	11560.68	3.42e+05
8	16	160	1.20	4037.75	12780.36	3.50e+05
12	8	80	1.23	3122.63	9403.92	3.29e+05
12	8	120	1.19	3424.67	10086.84	3.14e+05
12	8	160	1.24	3727.79	10872.36	3.22e+05
12	12	80	1.26	3324.23	9918.72	2.98e+05
12	12	120	1.26	3725.63	10899.00	2.97e+05
12	12	160	1.20	4129.91	11940.48	3.14e+05
12	16	80	1.20	3324.59	10438.56	3.11e+05
12	16	120	1.22	3728.87	11842.56	3.36e+05
12	16	160	1.21	4130.27	13023.36	3.36e+05
16	8	80	1.24	3129.83	9694.08	3.32e+05
16	8	120	1.20	3426.83	10335.60	3.24e+05
16	8	160	1.23	3731.03	11107.44	3.27e+05
16	12	80	1.25	3325.67	10117.44	3.02e+05
16	12	120	1.25	3729.23	11129.04	3.05e+05
16	12	160	1.24	4132.07	12135.24	3.13e+05
16	16	80	1.22	3327.47	10703.52	3.16e+05
16	16	120	1.24	3730.31	11952.00	3.15e+05
16	16	160	1.23	4132.79	13272.12	3.37e+05

Table C.11: Tile Performance, LUT Size 5, CLB Size 7

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	1.44	3378.59	10503.36	4.16e+05
8	8	120	1.42	3677.39	11217.60	4.25e+05
8	8	160	1.38	3983.03	12134.52	4.48e+05
8	12	80	1.39	3576.23	11214.36	4.24e+05
8	12	120	1.40	3978.35	12182.04	4.47e+05
8	12	160	1.38	4381.91	13181.40	4.31e+05
8	16	80	1.37	3579.47	11657.88	4.21e+05
8	16	120	1.39	3985.55	13133.88	4.63e+05
8	16	160	1.43	4387.31	14215.68	4.49e+05
12	8	80	1.40	3495.23	11126.88	4.32e+05
12	8	120	1.36	3794.03	11730.60	4.39e+05
12	8	160	1.42	4097.15	12517.20	4.52e+05
12	12	80	1.41	3691.43	11660.76	4.26e+05
12	12	120	1.41	4095.35	12673.44	4.39e+05
12	12	160	1.39	4493.87	13708.44	4.52e+05
12	16	80	1.40	3696.83	12090.60	4.37e+05
12	16	120	1.40	4096.43	13495.32	4.47e+05
12	16	160	1.43	4499.27	14720.40	4.60e+05
16	8	80	1.39	3496.67	11052.00	4.22e+05
16	8	120	1.40	3797.99	12025.44	4.44e+05
16	8	160	1.41	4101.47	12790.80	4.42e+05
16	12	80	1.44	3693.23	11668.32	4.13e+05
16	12	120	1.43	4095.71	12847.32	4.29e+05
16	12	160	1.43	4497.11	13882.32	4.55e+05
16	16	80	1.41	3696.83	12420.36	4.42e+05
16	16	120	1.44	4100.75	13698.00	4.59e+05
16	16	160	1.42	4501.43	15060.24	4.69e+05

Table C.12: Tile Performance, LUT Size 5, CLB Size 8

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	1.61	3938.39	12685.68	4.46e+05
8	8	120	1.59	4237.19	13263.84	4.26e+05
8	8	160	1.60	4541.39	14186.16	4.50e+05
8	12	80	1.59	4137.83	13235.76	4.33e+05
8	12	120	1.60	4538.87	14220.36	4.24e+05
8	12	160	1.60	4944.59	15346.44	4.51e+05
8	16	80	1.56	4138.55	13707.00	4.23e+05
8	16	120	1.58	4541.75	15167.52	4.58e+05
8	16	160	1.61	4946.03	16404.48	4.61e+05
12	8	80	1.64	4063.67	12951.72	4.29e+05
12	8	120	1.61	4365.35	13730.04	4.25e+05
12	8	160	1.62	4667.75	14585.76	4.40e+05
12	12	80	1.65	4262.39	13587.48	4.41e+05
12	12	120	1.64	4665.95	14581.44	4.26e+05
12	12	160	1.60	5069.15	15709.32	4.43e+05
12	16	80	1.62	4265.27	14212.08	4.59e+05
12	16	120	1.62	4666.67	15563.88	4.59e+05
12	16	160	1.56	5071.31	16824.96	4.85e+05
16	8	80	1.55	4059.71	13520.88	4.57e+05
16	8	120	1.59	4363.91	14177.52	4.48e+05
16	8	160	1.57	4668.47	14986.44	4.57e+05
16	12	80	1.56	4261.31	14012.64	4.29e+05
16	12	120	1.57	4666.67	14957.64	4.26e+05
16	12	160	1.62	5069.51	16154.64	4.77e+05
16	16	80	1.63	4263.47	14664.96	4.62e+05
16	16	120	1.56	4668.11	15898.68	4.55e+05
16	16	160	1.57	5071.67	17300.52	5.09e+05

Table C.13: Tile Performance, LUT Size 5, CLB Size 9

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	1.68	4305.23	14779.08	4.94e+05
8	8	120	1.64	4609.79	15668.64	5.00e+05
8	8	160	1.68	4913.27	16530.84	5.15e+05
8	12	80	1.64	4509.35	15549.48	4.98e+05
8	12	120	1.64	4913.63	16647.84	5.09e+05
8	12	160	1.62	5315.03	17776.80	5.21e+05
8	16	80	1.63	4512.23	16213.68	5.36e+05
8	16	120	1.68	4911.83	17379.00	4.97e+05
8	16	160	1.67	5317.91	18661.32	5.13e+05
12	8	80	1.65	4447.07	15263.28	4.95e+05
12	8	120	1.67	4748.75	15881.04	5.10e+05
12	8	160	1.67	5052.59	16642.80	4.94e+05
12	12	80	1.67	4646.51	15657.12	4.59e+05
12	12	120	1.67	5051.15	16603.20	4.86e+05
12	12	160	1.65	5455.43	17822.88	5.01e+05
12	16	80	1.67	4648.31	16186.32	4.74e+05
12	16	120	1.68	5048.99	17835.48	5.40e+05
12	16	160	1.68	5456.51	19105.92	5.46e+05
16	8	80	1.62	4451.39	16007.04	5.21e+05
16	8	120	1.62	4753.43	16453.08	5.05e+05
16	8	160	1.66	5051.51	17021.88	5.07e+05
16	12	80	1.67	4651.19	16175.88	4.80e+05
16	12	120	1.65	5051.87	17228.88	4.99e+05
16	12	160	1.68	5454.35	18210.24	5.10e+05
16	16	80	1.66	4652.27	16658.64	4.89e+05
16	16	120	1.65	5052.59	18262.44	5.34e+05
16	16	160	1.64	5454.35	19539.72	5.15e+05

Table C.14: Tile Performance, LUT Size 5, CLB Size 10

C.4 ASIC Performance for LUT SIZE 6

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	0.66	2793.95	7273.44	2.56e+05
8	8	120	0.65	3094.91	7979.40	2.79e+05
8	8	160	0.67	3398.39	8558.28	2.57e+05
8	12	80	0.67	2993.39	7824.24	2.46e+05
8	12	120	0.67	3394.79	8834.04	2.68e+05
8	12	160	0.66	3797.27	9785.88	2.75e+05
8	16	80	0.66	2991.95	8367.84	2.58e+05
8	16	120	0.65	3398.75	9618.12	2.72e+05
8	16	160	0.66	3799.43	10893.24	2.73e+05
12	8	80	0.66	2869.19	7469.28	2.62e+05
12	8	120	0.66	3170.87	8187.12	2.67e+05
12	8	160	0.67	3472.55	8882.64	2.76e+05
12	12	80	0.69	3067.19	8028.00	2.42e+05
12	12	120	0.69	3470.39	9066.60	2.71e+05
12	12	160	0.68	3874.67	9983.16	2.66e+05
12	16	80	0.66	3072.23	8564.40	2.74e+05
12	16	120	0.68	3473.27	9911.16	2.91e+05
12	16	160	0.66	3875.39	11114.28	2.78e+05
16	8	80	0.66	2867.03	7529.40	2.44e+05
16	8	120	0.64	3170.15	8467.92	2.86e+05
16	8	160	0.65	3473.63	9126.72	2.80e+05
16	12	80	0.68	3068.27	8189.28	2.50e+05
16	12	120	0.66	3471.11	9245.16	2.64e+05
16	12	160	0.68	3872.87	10299.60	2.82e+05
16	16	80	0.67	3068.99	8668.80	2.61e+05
16	16	120	0.65	3475.43	10057.32	2.82e+05
16	16	160	0.66	3875.39	11340.36	2.87e+05

Table C.15: Tile Performance, LUT Size 6, CLB Size 4

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	0.88	3315.95	8899.20	3.20e+05
8	8	120	0.85	3622.31	9701.64	3.32e+05
8	8	160	0.87	3923.63	10449.36	3.55e+05
8	12	80	0.90	3519.35	9508.32	3.28e+05
8	12	120	0.89	3927.59	10615.68	3.53e+05
8	12	160	0.87	4327.55	11648.52	3.55e+05
8	16	80	0.90	3521.15	10138.32	3.37e+05
8	16	120	0.86	3923.27	11401.92	3.53e+05
8	16	160	0.90	4326.83	12593.52	3.45e+05
12	8	80	0.88	3412.43	9377.64	3.51e+05
12	8	120	0.86	3716.99	10005.48	3.51e+05
12	8	160	0.89	4016.87	10741.68	3.61e+05
12	12	80	0.86	3612.59	9820.08	3.33e+05
12	12	120	0.87	4016.15	10931.76	3.53e+05
12	12	160	0.87	4420.43	11856.24	3.45e+05
12	16	80	0.90	3612.95	10461.24	3.53e+05
12	16	120	0.88	4018.67	11722.32	3.63e+05
12	16	160	0.89	4420.43	13089.96	3.97e+05
16	8	80	0.91	3404.87	9469.08	3.44e+05
16	8	120	0.88	3709.79	10378.80	3.67e+05
16	8	160	0.85	4010.75	11014.56	3.55e+05
16	12	80	0.88	3610.07	10107.72	3.46e+05
16	12	120	0.91	4009.67	11104.56	3.50e+05
16	12	160	0.89	4412.15	12098.16	3.47e+05
16	16	80	0.89	3610.07	10641.24	3.51e+05
16	16	120	0.88	4011.11	11913.48	3.52e+05
16	16	160	0.89	4414.67	13182.12	3.65e+05

Table C.16: Tile Performance, LUT Size 6, CLB Size 5

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	1.10	3856.31	11117.52	4.30e+05
8	8	120	1.10	4157.63	11959.56	4.55e+05
8	8	160	1.09	4462.19	12814.56	4.80e+05
8	12	80	1.11	4055.39	11850.12	4.46e+05
8	12	120	1.11	4458.59	12629.16	4.26e+05
8	12	160	1.12	4862.87	13811.40	4.44e+05
8	16	80	1.11	4055.75	12172.32	4.33e+05
8	16	120	1.15	4459.67	13559.04	4.51e+05
8	16	160	1.08	4863.95	14967.72	4.91e+05
12	8	80	1.12	3959.99	11583.72	4.60e+05
12	8	120	1.13	4265.27	12193.20	4.57e+05
12	8	160	1.10	4564.79	13120.20	4.73e+05
12	12	80	1.13	4169.51	12000.60	4.26e+05
12	12	120	1.12	4569.11	12987.72	4.24e+05
12	12	160	1.13	4971.95	14054.40	4.51e+05
12	16	80	1.13	4163.03	12686.76	4.66e+05
12	16	120	1.10	4567.67	14173.92	5.17e+05
12	16	160	1.12	4967.99	15173.64	4.71e+05
16	8	80	1.11	3958.91	11686.68	4.41e+05
16	8	120	1.13	4263.47	12555.00	4.64e+05
16	8	160	1.10	4567.31	13295.52	4.77e+05
16	12	80	1.10	4162.67	12389.04	4.50e+05
16	12	120	1.14	4565.51	13281.12	4.41e+05
16	12	160	1.07	4967.99	14456.52	4.67e+05
16	16	80	1.08	4162.31	12981.60	4.62e+05
16	16	120	1.11	4569.83	14262.48	4.93e+05
16	16	160	1.10	4968.35	15527.52	4.87e+05

Table C.17: Tile Performance, LUT Size 6, CLB Size 6

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	1.34	4390.55	13345.92	5.75e+05
8	8	120	1.33	4693.67	14142.24	5.97e+05
8	8	160	1.34	4996.79	14818.68	5.80e+05
8	12	80	1.34	4590.71	13800.96	5.50e+05
8	12	120	1.33	4994.27	14955.12	5.73e+05
8	12	160	1.34	5396.39	15965.28	5.79e+05
8	16	80	1.34	4592.51	14370.84	5.68e+05
8	16	120	1.34	4995.71	15907.32	6.03e+05
8	16	160	1.33	5397.47	16922.88	5.71e+05
12	8	80	1.34	4514.03	13624.56	5.75e+05
12	8	120	1.37	4815.71	14395.32	5.81e+05
12	8	160	1.37	5117.03	15093.00	5.64e+05
12	12	80	1.37	4714.55	14107.32	5.40e+05
12	12	120	1.36	5117.75	15162.48	5.58e+05
12	12	160	1.35	5521.31	16216.20	5.64e+05
12	16	80	1.35	4715.27	14822.28	5.80e+05
12	16	120	1.38	5119.19	15987.24	5.75e+05
12	16	160	1.33	5518.79	17215.56	5.74e+05
16	8	80	1.33	4511.87	13934.16	5.72e+05
16	8	120	1.34	4813.19	14720.04	5.88e+05
16	8	160	1.32	5117.39	15468.84	5.92e+05
16	12	80	1.34	4712.03	14529.60	5.71e+05
16	12	120	1.33	5113.07	15394.68	5.45e+05
16	12	160	1.34	5516.27	16565.76	5.58e+05
16	16	80	1.32	4712.39	15068.88	5.83e+05
16	16	120	1.32	5116.67	16344.72	5.94e+05
16	16	160	1.32	5517.71	17688.96	6.18e+05

Table C.18: Tile Performance, LUT Size 6, CLB Size 7

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	1.49	5165.27	16133.04	5.86e+05
8	8	120	1.47	5468.39	16873.56	6.15e+05
8	8	160	1.49	5771.15	17564.76	6.27e+05
8	12	80	1.47	5363.63	16671.60	6.04e+05
8	12	120	1.46	5770.43	17725.68	6.16e+05
8	12	160	1.47	6173.63	19058.76	6.28e+05
8	16	80	1.46	5367.95	17256.24	6.31e+05
8	16	120	1.46	5771.87	18632.52	6.24e+05
8	16	160	1.48	6174.35	19784.16	6.19e+05
12	8	80	1.49	5297.03	16422.48	6.00e+05
12	8	120	1.47	5599.43	17280.00	6.13e+05
12	8	160	1.50	5901.47	17959.68	5.89e+05
12	12	80	1.48	5498.63	17190.72	6.00e+05
12	12	120	1.47	5901.11	18393.48	6.06e+05
12	12	160	1.48	6306.83	19286.28	6.10e+05
12	16	80	1.48	5497.91	17811.36	6.23e+05
12	16	120	1.47	5905.07	18812.52	6.06e+05
12	16	160	1.47	6305.03	20228.76	6.23e+05
16	8	80	1.48	5297.75	16840.08	5.83e+05
16	8	120	1.48	5600.51	17732.16	6.16e+05
16	8	160	1.44	5903.63	18536.04	6.55e+05
16	12	80	1.48	5497.19	17458.92	6.18e+05
16	12	120	1.45	5903.63	18515.16	6.16e+05
16	12	160	1.50	6305.75	19796.76	6.28e+05
16	16	80	1.51	5498.99	17959.32	6.38e+05
16	16	120	1.47	5902.19	19413.36	6.38e+05
16	16	160	1.51	6303.95	20694.96	6.43e+05

Table C.19: Tile Performance, LUT Size 6, CLB Size 8

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	1.67	5730.47	18813.24	7.12e+05
8	8	120	1.64	6029.27	19361.16	6.79e+05
8	8	160	1.65	6334.55	20297.16	7.68e+05
8	12	80	1.72	5929.55	19260.36	6.98e+05
8	12	120	1.72	6333.83	20048.40	6.78e+05
8	12	160	1.71	6738.83	21247.20	7.04e+05
8	16	80	1.66	5931.71	19634.40	7.07e+05
8	16	120	1.65	6336.35	21302.64	7.42e+05
8	16	160	1.65	6737.39	22612.32	7.70e+05
12	8	80	1.68	5880.23	19420.20	7.23e+05
12	8	120	1.71	6186.59	20046.24	7.26e+05
12	8	160	1.69	6484.31	20735.28	7.30e+05
12	12	80	1.67	6084.71	19746.00	7.15e+05
12	12	120	1.65	6487.55	20566.08	7.02e+05
12	12	160	1.70	6887.87	21817.44	7.15e+05
12	16	80	1.67	6083.27	20369.16	7.03e+05
12	16	120	1.66	6484.67	21962.88	7.53e+05
12	16	160	1.65	6891.47	22975.92	7.39e+05
16	8	80	1.68	5880.95	19610.28	7.14e+05
16	8	120	1.66	6185.51	20198.88	7.12e+05
16	8	160	1.63	6484.67	21004.20	7.45e+05
16	12	80	1.65	6085.43	20151.00	7.18e+05
16	12	120	1.68	6482.87	21230.64	7.21e+05
16	12	160	1.66	6887.51	22264.20	7.17e+05
16	16	80	1.65	6081.11	20937.60	7.63e+05
16	16	120	1.63	6486.47	22241.16	7.80e+05
16	16	160	1.62	6891.83	23234.04	7.21e+05

Table C.20: Tile Performance, LUT Size 6, CLB Size 9

IPIN W	CHAN W	CHAN N	Crit. Path	Config Area	Total Area	Static Pwr
8	8	80	2.12	6291.35	21166.56	8.08e+05
8	8	120	2.01	6586.91	22088.16	8.52e+05
8	8	160	2.00	6896.15	22695.12	8.18e+05
8	12	80	2.04	6485.39	21988.80	8.17e+05
8	12	120	2.07	6899.75	22977.72	8.17e+05
8	12	160	2.08	7301.15	24079.68	8.46e+05
8	16	80	2.04	6492.59	22593.96	8.31e+05
8	16	120	1.98	6894.71	23931.36	8.53e+05
8	16	160	2.04	7296.47	25173.36	8.95e+05
12	8	80	2.06	6459.47	21870.36	8.41e+05
12	8	120	2.01	6762.95	22583.16	8.32e+05
12	8	160	2.01	7059.95	23607.00	8.65e+05
12	12	80	2.01	6661.79	22701.60	8.73e+05
12	12	120	1.99	7061.75	23541.12	8.66e+05
12	12	160	2.04	7465.67	24703.56	8.54e+05
12	16	80	2.01	6663.95	23105.16	8.39e+05
12	16	120	2.06	7058.87	24448.32	8.62e+05
12	16	160	2.04	7467.11	25773.48	8.93e+05
16	8	80	2.03	6450.11	22462.20	8.65e+05
16	8	120	2.09	6756.47	23137.20	8.70e+05
16	8	160	1.97	7058.51	23889.60	8.68e+05
16	12	80	1.99	6655.31	23182.20	8.76e+05
16	12	120	2.00	7053.83	23974.92	8.69e+05
16	12	160	2.02	7459.19	24930.36	8.34e+05
16	16	80	1.95	6652.43	23338.80	8.35e+05
16	16	120	1.99	7057.07	24771.24	8.53e+05
16	16	160	1.99	7460.99	26187.48	8.67e+05

Table C.21: Tile Performance, LUT Size 6, CLB Size 10

Appendix D

List of Acronyms and Abbreviations

ASIC Application Specific Integrated Circuit

BLE Basic Logic Element

CAD Computer Aided Design

CHANXY Routing Channel X and Channel Y

CB Connection Box

CLB Configuration Logic Block

DC Design Compiler

DUT Device Under Test

FF Flip-Flop

FPGA Field Programmable Gate Array

HDL Hardware Description Language

IC Integrated Circuit

ICC IC Compiler

IO Input and Output

IPIN Input Pin

LUT Look Up Table

MEM Memory

MIM Multiple Instantiated Module

MUL Multiply

MUX Multiplexer

QoR Quality of Result

RTL Register Transfer Level

SB Switch Box

SRAM Static Random Access Memory

Verilog HDL Verilog Hardware Description Language

VPR The Versatile Packing, Placement and Routing tool

VTR Verilog to Routing

XBAR Cross Bar