ECE 4130/6130 FPGA Project
Summer 2020

Professor Vincent J. Mooney III
School of Electrical and Computer Engineering
Georgia Institute of Technology, Atlanta, GA

1. Introduction

The ECE 4130/6130 project for the Summer 2020 semester will focus on the design and implementation of a scalable traditional (fine-grained) reconfigurable logic chip known as a Field Programmable Gate Array (FPGA). The FPGA design will target fabrication at 45nm; potentially, some students will also design in more recent chip technology using rules available under a non-disclosure agreement (NDA). Alternatively, a free 15nm design library is available but requires installation (which is nontrivial…); please ask Prof. Mooney if this interests you.

Reconfigurable Logic Architectures

Reconfigurable logic families including FPGAs are devices with programmable logic blocks and programmable interconnect. Figure 1 shows an example of an island-style FPGA.

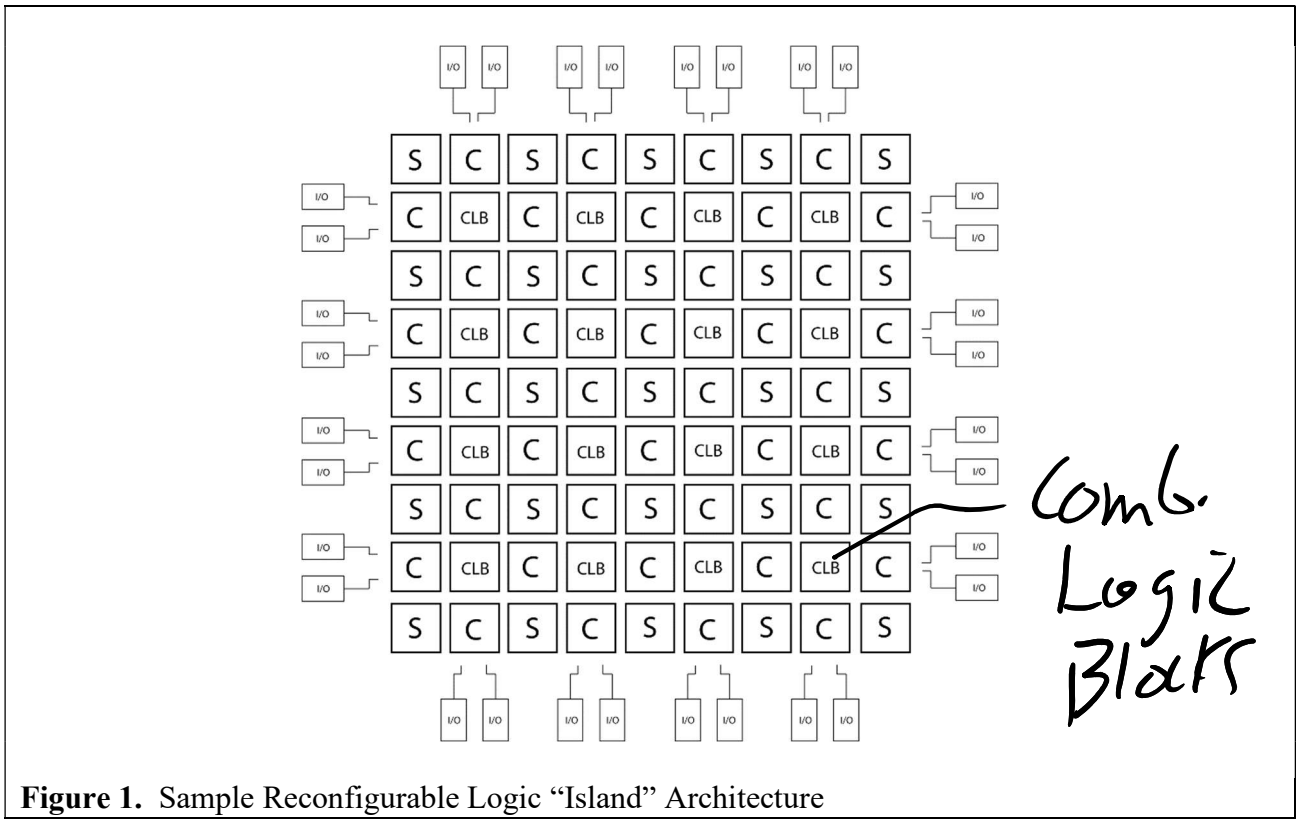

Figure 1. Sample Reconfigurable Logic "Island" Architecture

The FPGA is comprised of a large amount of programmable basic logic elements capable of implementing arbitrary Boolean equations with reconfigurable interconnect to wire them together.

There is a seemingly endless design space for implementing reconfigurable logic, so for the purpose of this project, we will use an SRAM[1]-based island architecture with three block types as shown in Figure 1: Combinational Logic Block (CLB), Switch Block (S-Block) and Connect Block (C-Block). CLBs are configurable logic blocks which contain the programmable logic elements. C-Blocks connect CLBs to the rows and columns of the interconnect, and S-Blocks comprise the space where the rows and columns meet and allow signals to pass or make turns or to not pass at all.

A single CLB is composed of multiple "Basic Logic Elements" (BLEs). A single BLE comprises a k-input Look-Up Table (LUT) and a Flip-Flop (FF). The k-input LUT in essence is a 2^k to one multiplexer with "k" select lines. Any arbitrary logical function of k variables can then be implemented as the output of the multiplexer by statically setting the 2^k inputs to the functionally correct values and then using the "k" select lines as the variable inputs. The output of each BLE is either registered or not registered by a bypassable FF; see Figure 2.

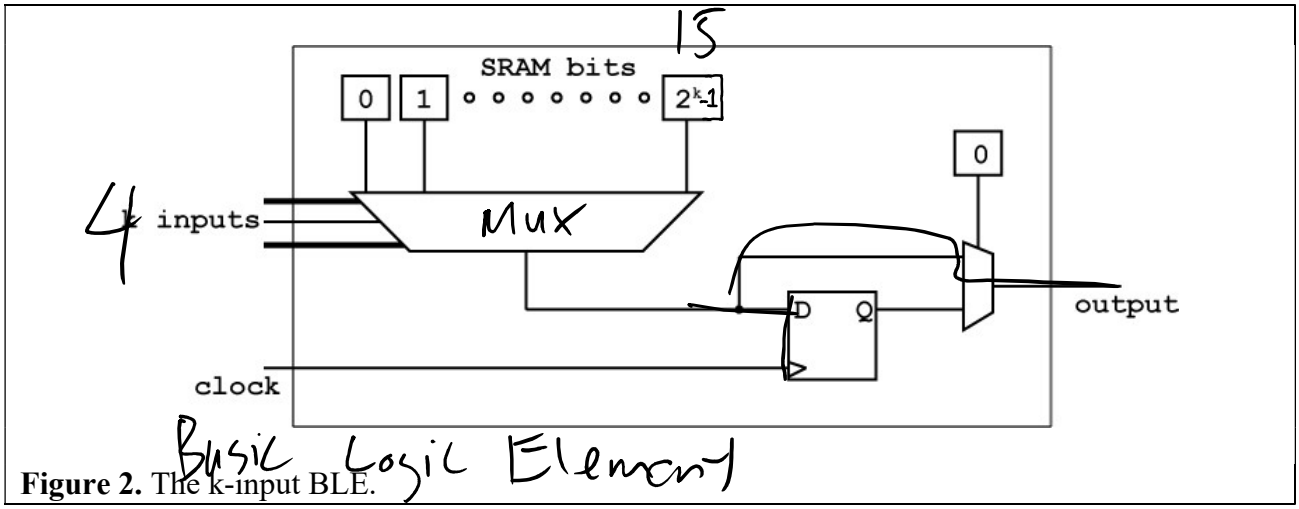

Figure 2. The k-input BLE.

The target Combinational Logic Block (CLB) design contains N BLEs producing "N" outputs from "I" inputs. These I inputs along with the N outputs are shared and multiplexed (using Log_2 (I + N) SRAM bits per multiplexor) amongst the k*N BLE inputs as shown in Figure 3.

[1] SRAM may be misleading, as the memory is not necessarily random access at all times. We will explore exactly how to implement a 6T-based reconfigurable array in this class. In any case, "SRAM-based FPGA" is the industry standard name for this type of device.

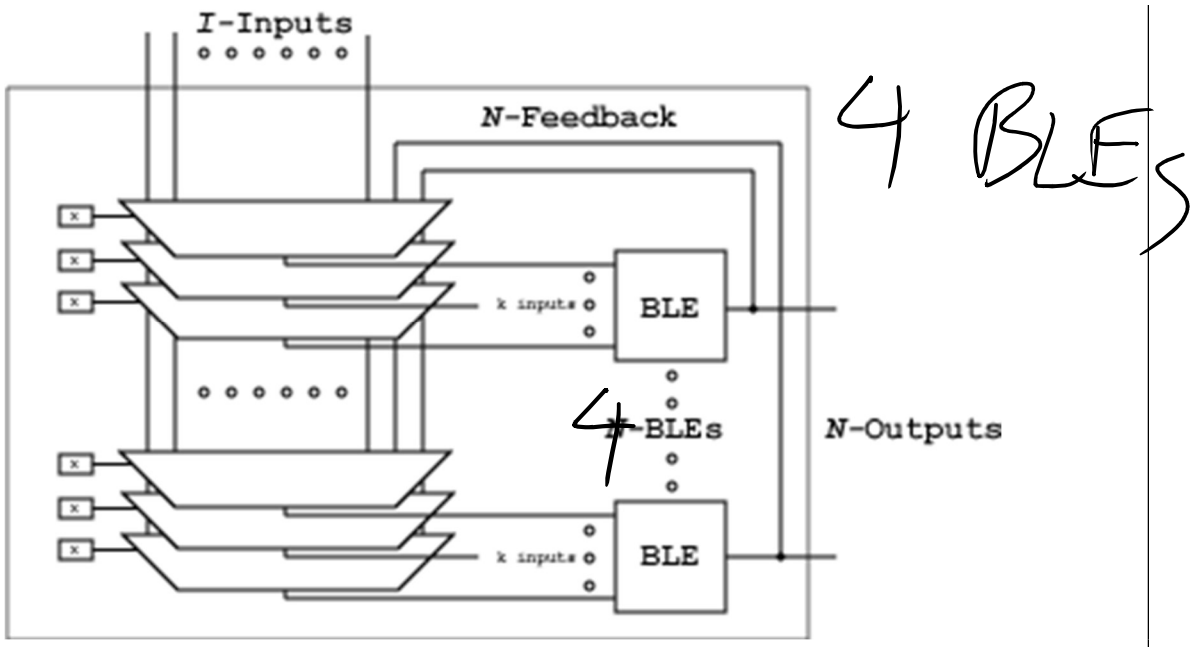

Figure 3. A Single CLB

The result is that k*N multiplexers are needed where each multiplexer takes in I+N inputs and produces one output bit. Clearly, not all values possible of I, k and N make sense. The feedback lines allow feedback between the local BLEs; also, the feedback lines keep this local routing out of the C-block matrix.

The CLBs, when arranged in an array with row and column based interconnect, form the basis of an island or mesh style traditional reconfigurable logic architecture as shown earlier in Figure 1 and repeated on the next page as Figure 4 for convenience. The inputs and outputs from each CLB are connected to the rows and columns of interconnect by C blocks. Signals can be passed along the rows and columns or turned from a column to a row and vice versa by S blocks. One S block, two C blocks, and a CLB form the unit that tiles and becomes the majority of the fabric of the FPGA as shown in Figure 5. Due to lack of time, we will not design the "corner" and "NSEW" (north-south-east-west) side tiles. Thus, for example, we will not consider how to connect to I/O pads.

The I inputs and N outputs from each CLB are connected to a fraction (Fc[2]) of the total number of lines, W, of the C block through switches (denoted as circles in the figures). The terminations of the rows and columns are the FPGA IO blocks connected through C blocks.

[2] "Fc" is fractional connect. This is a ratio between 0 and 1 which represents what percentage of C-Block routing has connections to the CLB MUX input.

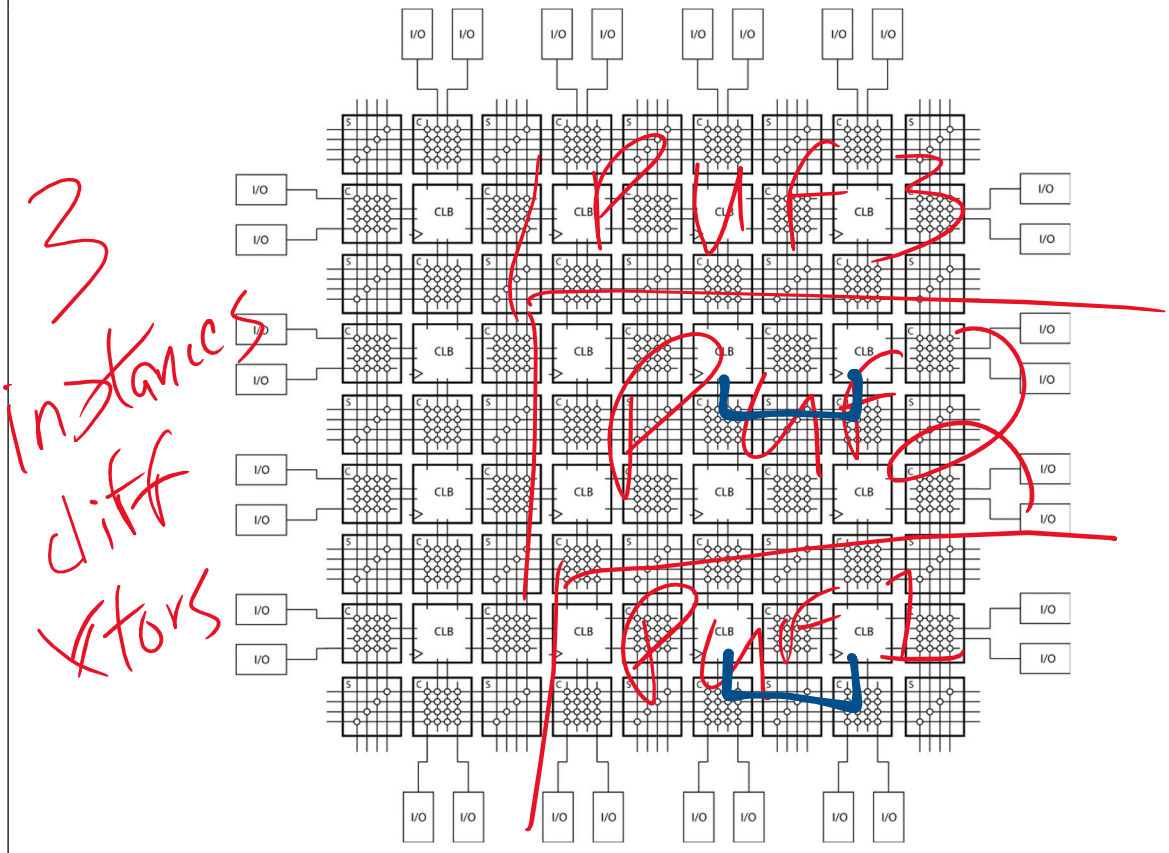

Figure 4. Reconfigurable Logic "Island" Architecture

Whenever a vertical and a horizontal channel intersect, there is an S block. In this architecture, when a wire enters an S block, there are three programmable switches that allow it to connect to three other wires in adjacent channel segments. This type of S block architecture is called the planar or domain-based switch box topology. In this switch box topology, a wire in track number one connects only to wires in track number one in adjacent channel segments, wires in track number 2 connect only to other wires in track number 2, and so on, please see Figure 5. This arrangement allows each signal to switch directions (but not to switch lines).

The state of every look-up table (LUT), every multiplexer, and every switch is held in SRAM. The contents of SRAM, called the bitstream, controls the operation of the FPGA and its implemented design.

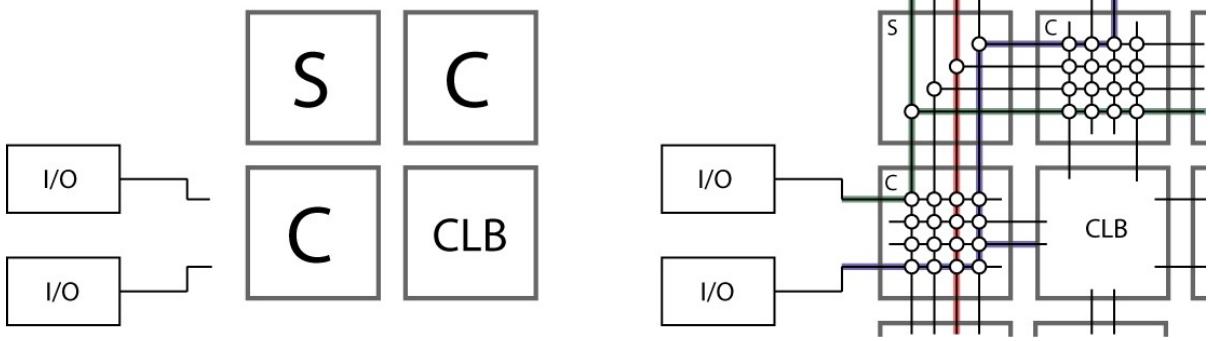

Figure 5. CLB, S Block, and C Block

Pin/Wire List

The following is a pin/wire list of the currently specified pins/wires with the active state in parenthesis if specified. There may be several Ground pins/wires.

- bypass clock signal / program clock
- bypass clock enable
- program enable
- bitstream in
- bitstream out
- sleep (high)
- global reset (low)
- VDD1, e.g., 0.8 volts
- VDD2, e.g., 1.1 volts
- VDD Pad, e.g., 2.5 volts
- Ground

2. Project Description

Each student in this class will design an entire CLB, C-block, S-block tile with the choice of one of the following two approaches: (1) **ultra low power (ULP) and high speed** (2) **super high speed (SHS) and low power**. (Note that some students may have arranged a specialized project and thus may have customized goals.) However, each student will participate in a competition where intermediate designs will be shared with the entire class, but the "competition" will be greater for students adopting the same approach. Those choosing **super high speed** will optimize their design for high speed first while maintaining **low power**; for **ultra low power**, minimizing energy consumption will be the metric of choice while also maintaining **high speed**. Students are free to use any logic families or digital tricks of their choice; furthermore, students are encouraged to explore the design space to come up with their best solution. All designs will be verified using HSPICE for speed and power.

This project is worth 30% of each student's final grade and will also include approximately two thirds of the lab grade. As such, it needs to be carried out with efforts proportional to approximately half of a course grade. Given the importance of properly interconnected intermediate design results, together with a need for efficiency, we will use the following guidelines.

Every class will be held every Monday/Wednesday with very few exceptions. In each class, some lecture material may be presented by Professor Mooney; furthermore, a few students will briefly report their status in each class by using a schedule organized by Prof. Mooney.

Each week for the first few weeks, each student will turn in an architecture and schematic of the basic elements (refer to project schedule and deliverables section) along with a writeup of what are the next steps planned. A written report (handwritten is not acceptable) of no less than three and no more than 10 pages (including figures) must be turned in. The lowest level allowed in the report (and also required to be in the report) is transistor-level. Gate-level and higher-level schematics/diagrams should also be presented in the report. Any assumptions should be clearly stated up front. Please do not expect questions to be answered on a continuous, ongoing, daily basis. Email questions may or may not be answered (email is typically a very inefficient method of communication for open-ended technical discussions about VLSI). Additional due dates will be posted/explained as the project progresses. Roughly, every four days or so some item will be due.

Anyone can make any suggestion at any level (circuit, architecture, etc.) at any time. Prof. Mooney will give feedback on which suggestion to take/follow. Arguments are welcome but need to be carried out in a discussion. There will be some evolution – perhaps significant! – of the specification and expectations; this will be taken into account in grading the final project.

The design requirements of CLB, C-Block and S-Block are described in the following sections in detail.

CLB Design

The FPGA architecture of choice is the island style or mesh based architecture shown in Figures 1 and 4. It consists of "islands" of combinational logic blocks surrounded by a "sea" of interconnect. Each student will be responsible for designing and laying out a functional CLB of functional design shown in Figure 3 with the following parameters: K = 4, N = 4, I = 10. The logic Flip Flop in the BLE should have an active low global asynchronous reset.

S Block Design

While CLB designs are about designing the islands of the FPGA, S Block and C Block Designs are responsible for the "sea" or the interconnect. A planar style S Block is detailed functionally in Figure 6, with a possible but not mandatory switch implementation. The number of vertical and horizontal tracks or wiring paths should be W = 24; the percentage of tracks that each input is connected to should be Fc_in = 0.5 (so 12 of the 24 lines are connected), while each output has Fc_out = 0.25 (so every 4^{th} line is connected). The S-Block line junction proposed in Figure 6 produces the desired function at a junction; however, other appropriate solutions may exist and should be explored. As can be seen in the figure, the S-Block can only switch the direction of signals, so the remaining ways to fan-out an output would be by routing in either the CLB or the C-Block.

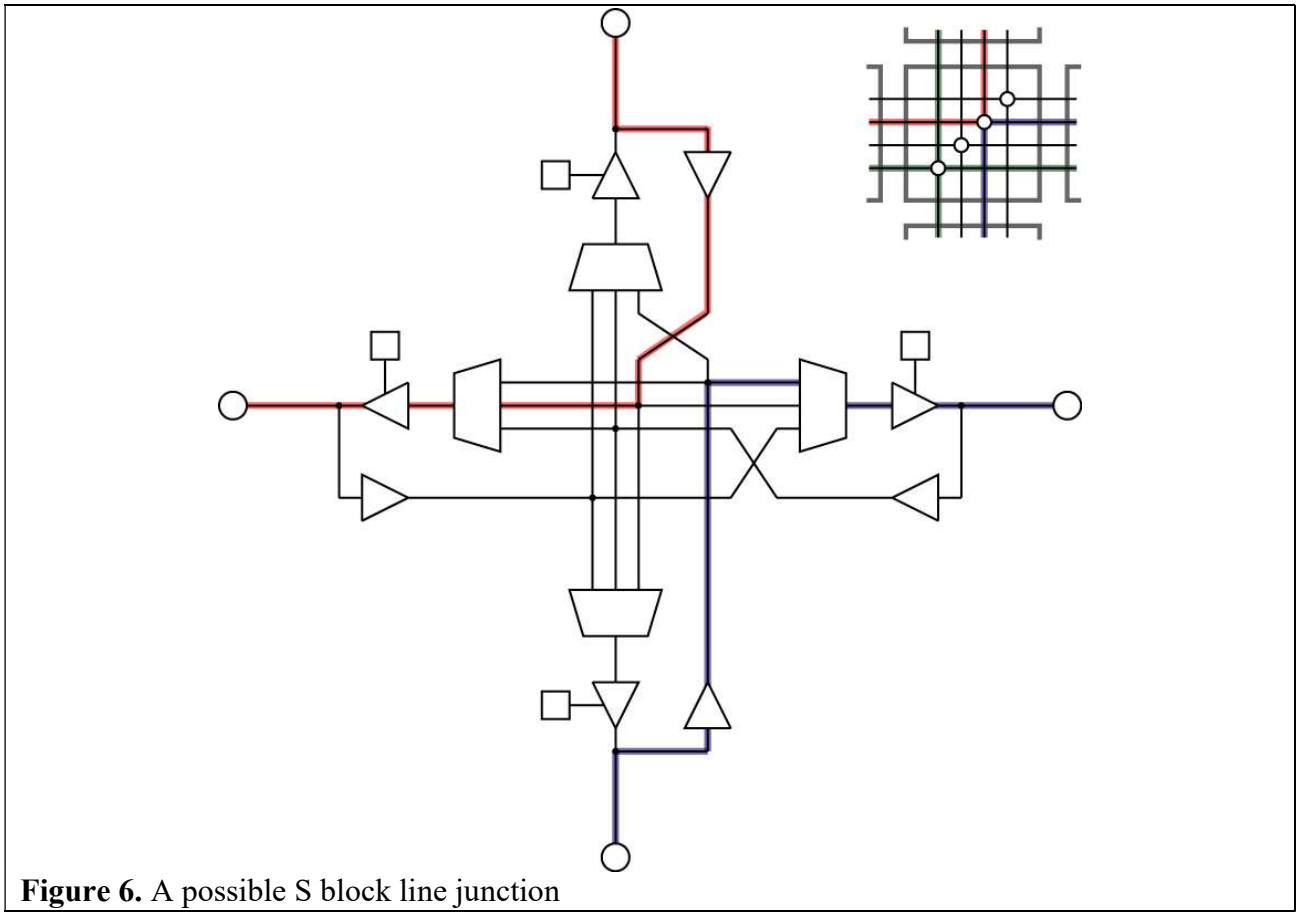

Figure 6. A possible S block line junction

C Block Design

While the S-Block can only switch the direction of signals, C Blocks connect the signals between CLBs. Functionally, the C-Block will supply 10 input lines and 4 output lines from the CLB as shown in Figures 5 and 7.

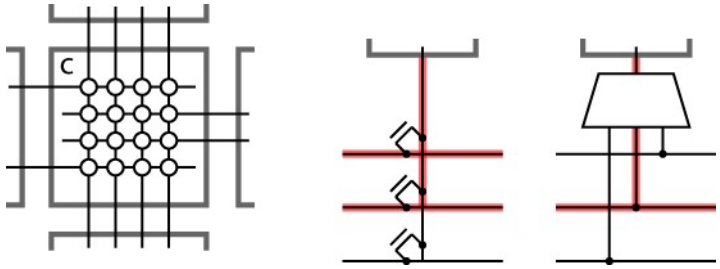

Figure 7. C Block, doglegging, no doglegging

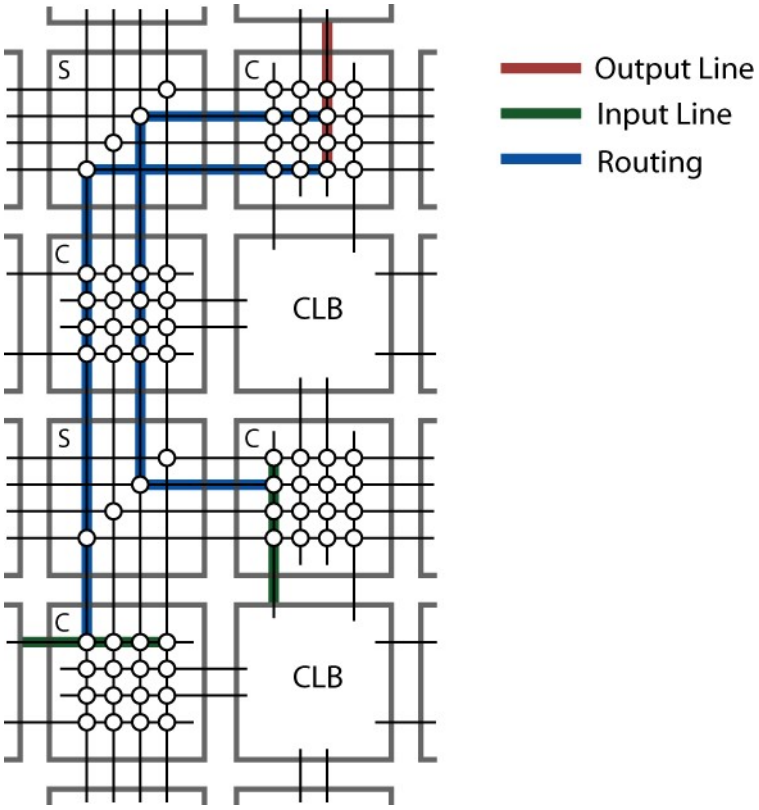

Figure 8. Doglegging on outputs allows multiple lines to be routed to the next input

Doglegging is a way to switch lines, or fan-out a connection. It is required that no doglegging will be allowed on input pins in the C block; however, doglegging is allowed on the output pins. One

output from the CLB may be tied to multiple lines in the doglegged scheme as shown in Figures 7 and 8. Furthermore, the input/output lines from each CLB must be exclusive to that CLB. Adjacent CLBs should not share input or output lines across a C-Block.

Much of the FPGA literature alludes/implies that designs do not allow doglegs between tracks off of the switches to the input pins, a feature that would allow signals to switch tracks in a C block and therefore would provide more routing options; instead, it seems that most FPGAs opt for a faster design where the inputs are buffered off of the tracks and multiplexed into the input pins.

SRAM Design

Not explicitly shown in the figures are the SRAM bits required to hold the configuration of every CLB, S-Block and C-Block. The state of every switch, connection, LUT, as well as every multiplexer in the CLB will have to be held in SRAM. The SRAM should be organized as 1-bit modified "flip-flops" or "registers" as shown in Figure 9. So, in addition to the logical and routed inputs and outputs of the CLB, there are word lines and bit lines from an address decoder (not part of your design) for read/write operations of the SRAM that holds the bitstream. Therefore, the S-Block or C-Block will have to provide word lines and bit lines in and out; of course, the word and bit line must properly tile.

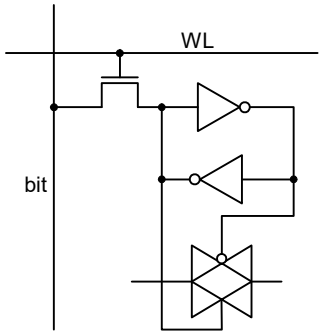

Figure 9: an SRAM cell

Critical Path Selection

The critical paths in the tile are from the output of the flip flop in one BLE to the input of the flip flop in another BLE inside a different CLB. The paths may go through several BLEs in other CLBs through the connections of C Blocks before arriving at the destination flip flop as shown in Figure 10.

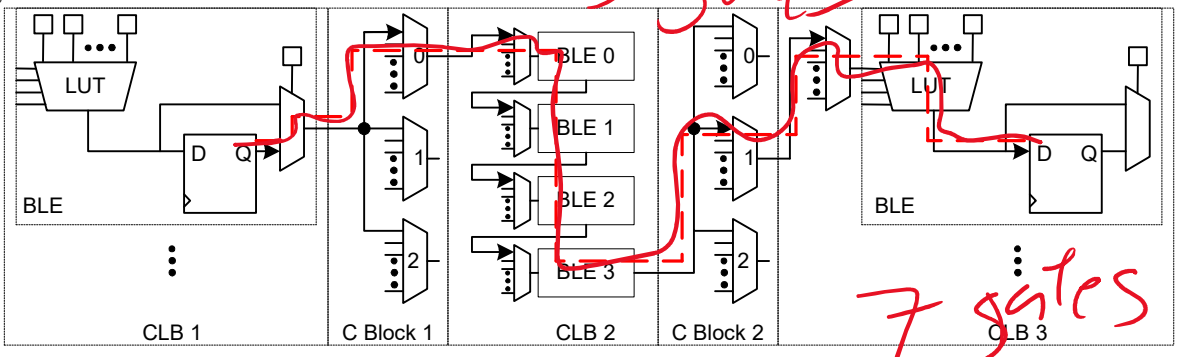

Figure 10: Critical Path

Clock Routing and Buffers

Even though global clock routing and clock generation are not major concerns of your design, each student is responsible for clock routing in the tile. Since there is no restriction on the metal layers to be used in your design, in order to employ the clock distribution scheme shown in Figure 11, one vertical and two horizontal routing spaces should be reserved for clock routing. Hence, the CLB block should have 10 input lines and 4 output lines from the C-Block as well as two non-overlapping clock lines from the clock routing.

In addition, due to the grid size for the entire chip and the clock distribution scheme as shown in Figure 11, spaces need to be left in a regular grid fashion for the clock distribution buffers that appear at every branch in the clock tree. It is required to leave channels for clock routing and spaces for inserting the clock buffers.

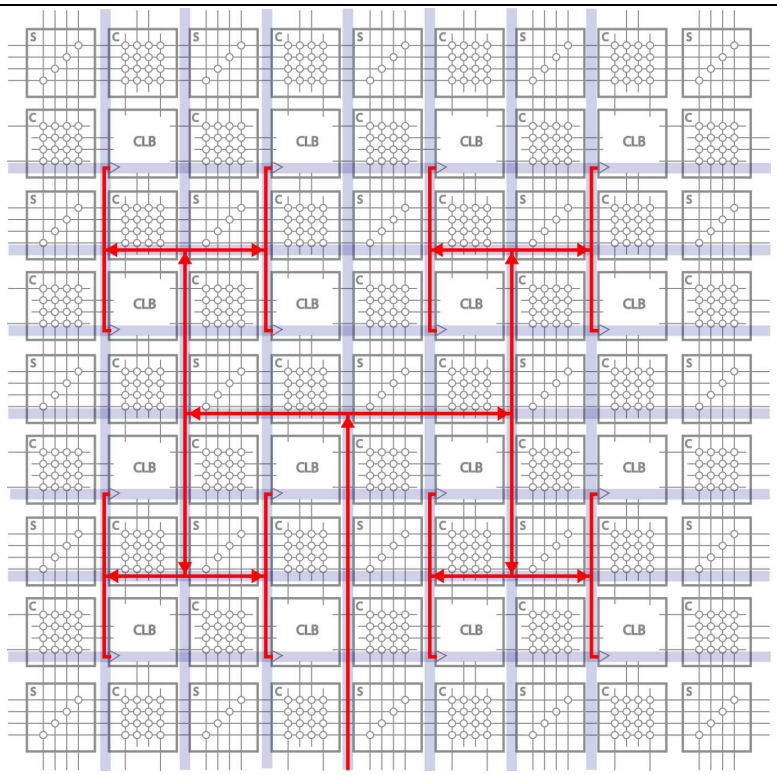

Figure 11. H Bar Clock Tree

General Design Considerations

As all of the individual components will be integrated to form a functional FPGA, the following general design rules apply to all projects.

- Metal layers, intents and metal related decisions should be specified and documented.

- The state SRAM/Flip Flops, which hold MUX state (i.e., the select input bits of all muxes are inputs from SRAM) and the logic of the LUTs, should have a program enable line which is active high. This line will be tied to ground during normal operation to ensure that erroneous programming will not occur during operation in the programmed, running state.
- Floating lines should be avoided whenever possible, and extra bits may be used to tie lines to known states.
- A global reset, which is active low, should be connected to all clockable Flip Flops in the BLE.
- The functioning of the FPGA blocks should be considered from the point of view of runtime. Optimize for the run condition and not the programming condition.
- MUX designs where a floating input is possible should have the ability to tie line to a known state. This is primarily a concern in the CLB, but this should be considered in other blocks as well.
- Area guideline:
  You can roughly trade off area for power/speed percent to percent (e.g., 50% power decrease at a cost of 50% area increase). Hence, it is not allowed to trade off large area increase (50%) for small percent (10%) of power/speed improvement (e.g., a ratio of 5 to 1). However, what exactly is "in-between" equal percentage to equal percentage and a ratio of 5 to 1 is debatable; do not focus on exactly what is the precise tradeoff ratio you have achieved; instead, justify/explain/rationalize your choices with qualitative and quantitative reasons (both are needed).

Specific Design Considerations: Super-High Speed (SHS) vs. Ultra Low Power (ULP)

Among many super high speed design options, one is to use pass transistors such as pass nFETs with the gate voltage increased to $V_{DD} + Vt_N$ instead of using T-gates to save area while achieving high speed. Student choosing SHS should investigate the pros and cons of this option.

For students choosing ultra low power, one of the low power design options is to design a scheme to turn partial or entire design into a low power mode. For example, a disable bit for each BLE and C-Block is used to turn the entire block off when not used. In such a case, an SRAM bit within the block is programmed to set the disable bit. Another example is that a global "sleep" mechanism or signal, which is active-high and should be present in all cells, will trigger the design to enter low-power mode. Student choosing ULP should look into the advantages and disadvantages of these options.

Brainstorm any other super high speed or ultra low power architecture and design implementation ideas for SHS or ULP, respectively. Each student should include the following into the final report:

- At least "top two" super high speed ideas for SHS or at least "top two" ultra low power ideas for ULP.
- The analysis of the pass nFET with increased voltage option for SHS, or the analysis of utilizing a disable bit and a sleep signal options for ULP. Discuss the pros and cons.
- Compare your SHS or ULP solutions to the corresponding option described above.

Schedule and Deliverables

Part 1: Critical Path Analysis (Due June 22)

We start our project with the paper analysis of a sample FPGA critical path using logical effort. The sample critical path, which is only a small portion of the complete critical path in Figure 10, is from the input of a BLE to the input of the LUT of the next BLE through the connection of a C Block. The entire path consists of multiplexers, flip-flops, LUTs and inverters/buffers as shown in Figure 12 below. However, Figure 12 is a functional diagram only. You are free to optimize the entire critical path by using any logic structures and/or high-speed circuits of your choice. In other words, you are free to choose any implementation of gates and topology. We will pay particular attention to whether or not these decisions are thought through and explained in the report you turn in.

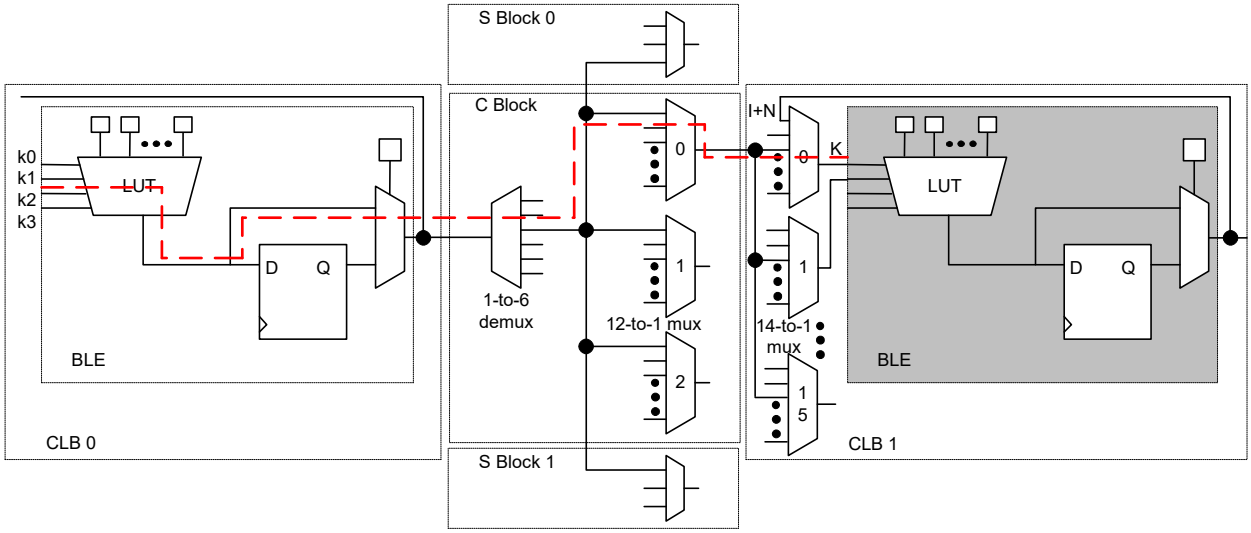

Figure 12: Sample Critical Path

Turn in a written report including the following:

1) Logical effort analysis of the critical path
2) Functional block diagrams
3) Hierarchical transistor schematics

Note:

No layouts at all
No floorplan
No stick diagrams

Part 2: SPICE to Logical Effort and Power Analysis (Due June 26)

Use SPICE simulation for gate characterization and power analysis on the sample critical path. Compare your results of simulation with those of hand calculations from the previous week.

Turn in a written report including the following:

1) SPICE simulation results

2) Comparison between results from SPICE and Logical Effort
3) Power analysis

Note:

No layouts
No floorplan
No stick diagrams

Part 3: CLB/S-Block/C-Block Design (Due July 1)

Brainstorm CLB, S-Block and C-Block architecture ideas and optimize the entire CLB, S-Block and C-Block tile by using any logic structures and/or high-speed/low power circuits of your choice. Please refer to the CLB/S Block/C Block Design Section for design requirements.

Turn in a written report including the following:

1) At least "top two" CLB, S-Block and C-Block architecture ideas
2) Functional block diagrams
3) Schematics
4) Stick diagrams

Note:

1) No layouts
2) No floorplan

Part 4: Path Characterization (Due July 6)

Use SPICE to calculate the logical effort and analyze power of the entire CLB, S-Block and C-Block tile. Compare your results of SPICE simulation with the results of hand calculations from previous weeks. Finalize the architecture, functional block diagram, logic family and the circuit types of the entire CLB, S-Block, and C-Block tile.

Turn in a written report including:

1) SPICE simulation results
2) Functional block diagram
3) Schematics
4) Stick diagrams

Note:

1) No layouts
2) No floorplan

Part 5: Individual Layouts and Floorplan (Due July 10)

By now each student should have a complete list of individual "cells" – NAND gates, INV gates, Flip-Flops, wide MUXes, "SRAM" or "FF" LUT bits, etc. – needed to implement the entire CLB. We have been holding back on layout to reduce wasted effort on layout. Now that each student has a more-or-less complete view (and justification) for the target architecture, functional block diagram, logic family and the circuit types of the entire CLB, we now proceed to lay out individual cells and DRC/LVS *at the individual cell level only* for this week's turn-in. Note that for each

"cell" a stick diagram must be done first; furthermore, prior to implementing layout, the stick diagrams must be used to derive "rectangles" which form a floorplan for your CLB, S-Block and C-Block tile. Please assume Phi1-Phi2 clocking signals are available from an external, unspecified source; include these CLK signals in your floorplan in a way that tiles correctly.

Due to the small class size and the remote usage of Cadence tools this summer, you are only required to provide a layout for the CLB (including, of course, all cells). However, you are also required to have a floorplan for the entire design (CLB, S-Block and C-Block).

Turn in a written report including the following:

1) A complete list of "cells" comprising your entire design (CLB, S-Block and C-Block)
2) For each cell, stick diagram(s) (Note: multiple stick diagrams may be needed if various aspect ratios – ratios of height to width – are needed for the floorplan)
3) A floorplan for your tile (i.e., CLB, S-Block and C-Block tile)
4) For each cell in the CLB, a layout which passes DRC and LVS

Note:

1) No DRC or LVS is required for "multi-cell" combinations; instead, however, layout + DRC + LVS *is required* for each individual cell in the CLB
2) For incomplete work from prior weeks, additional work may need to be provided in the report (e.g., additional SPICE simulations may have to be carried out)

Part 6: Full Layout and Verification of CLB Tile (Due July 15)

For this week all "cells" must be combined into a full layout of your CLB tile. This week's work may be what most engineers and the general public think about when they hear the term "VLSI." This will result in a multi-color layout which dwarfs the complexity and effort put into lab2! ☺

Please note that in going from individual cells to multi-cell combinations it is commonly advised to add at most 10 additional wire connections prior to re-running DRC and/or LVS. In other words, aim for incrementally putting your tile together and checking for correctness.

Turn in a written report including:

1) All information from prior weeks
2) Updated stick diagrams
3) Updated floorplan for your tile
4) Portions of your layout (i.e., parts of the CLB) which pass DRC and LVS
5) A complete layout which passes DRC and LVS

Note:

1) Please DO NOT attempt any major changes to what you decided at the end of Part 4; instead, for new approaches potentially resulting in improved design, add a "lessons learned" and/or "future work" section to your report in which you NOTE (but do not DO) new approaches which you believe will improve your design in some metric or metrics (speed, power, area, etc.)
2) For incomplete work from prior weeks, additional work may need to be provided in the report (e.g., additional SPICE simulations may have to be carried out)

Part 7: Post-Layout Critical Path Timing Analysis (Due July 20)

Use Cadence tools to re-verify the timing of the specified critical path. Do NOT focus on other paths; instead, focus on the identified critical path. Extracting accurate parasitics should result in non-trivial insights into your design. Add comments on how you could improve your CLB tile to the "lessons learned" and/or "future work" sections of your report. At this point, do not make any major changes to your design.

Turn in a written report including the following:

1) All information from prior weeks
2) A "high-level" view of what you learned about your layout due to insights from extracting proper SPICE models; e.g., what parasitics or other issues caught you off guard?
3) SPICE simulation results based on extraction

Note:

1) Please emphasize having a complete, working (DRC + LVS) tile for your final report; it is NOT ACCEPTABLE to say "we tried to add X and Y optimizations to our tile; as a result, our final CLB tile does not fully DRC, LVS or extract"
2) Place extra effort on EXPLAINING your design and your design decisions; Prof. Mooney and any graduate students who may help to grade your project are not omniscient! Please write down even points which you think may be obvious. Place extra effort on writing and rewriting your report. It is best if you write a mostly complete report by the day before the due date; then, go to sleep; then, wake up and spend an hour or an hour and a half slowly re-reading your report and adding clarity/grammar/etc.

FINAL REPORT AND PRESENTATION

Between the last turn-in (Project Part 7 above) and the final exam time (Wednesday July 29 from 8am to 10:50am) you will be expected to iterate on your written report to improve it as well as produce a powerpoint slide deck to present your design on July 29. Final reports and powerpoint slide decks are due prior to 7:30am on Wednesday July 29, 2020. Please note that the format for the presentation must be powerpoint and not some other format such as a Google version (the problem in the past has been that the translation from free on-line document programs to powerpoint results in strange fonts and messed up pictures). The final report may be in pdf or MS-DOC.

