ECE 6130 Reconfigurable Logic Project
Professors: Dr. V. Mooney, Dr. P. Hasler
Contributors: Richard Wunderlich, Brian Degnan
Stuart Duerson, Upendra Godse

Introduction

Previous graduate classes taught by Professor Mooney (ECE 6130 and ECE 6132) have looked closely at multiple processors on a single silicon chip as a way of implementing reprogrammable systems, e.g., for wireless. This year, ECE 6130 will focus on design of a traditional reconfigurable logic chip; such a chip is also known as a Field Programmable Gate Array (FPGA) or a Complex Programmable Logic Device (CPLD). The goal of this semester's project is to design and implement a scalable traditional (i.e., fine grained) reconfigurable logic chip for fabrication targeted at a 0.13u process. Please note that a legal Non-Disclosure Agreement (NDA) must be signed to receive access to the 0.13u process parameters and design kit. Students who prefer not to sign the NDA will use the TSMC 0.18u process but may have less choice as to specific sub-project (details later).

Reconfigurable Logic Architectures

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

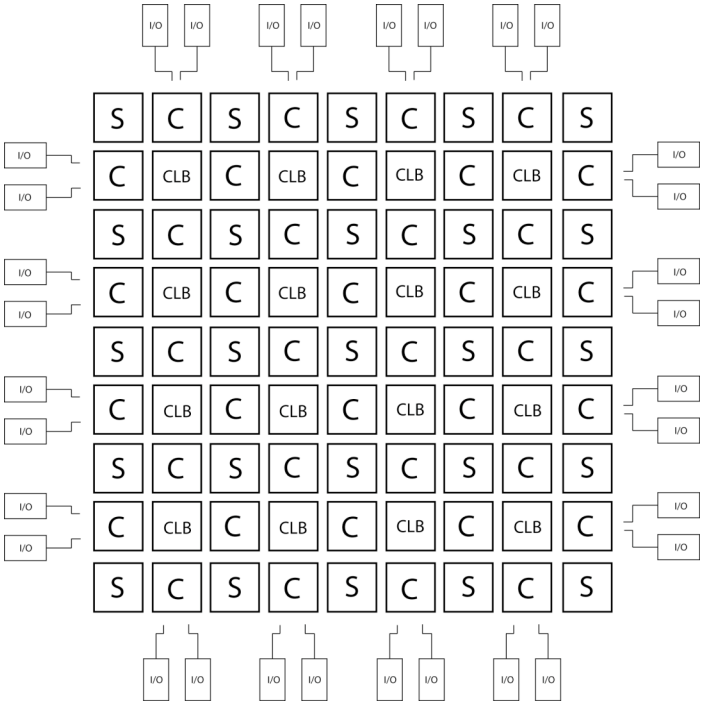


Figure 1. Sample Reconfigurable Logic "Island" Architecture

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 the BLE is either registered or not by a bypassable FF; see Figure 2.

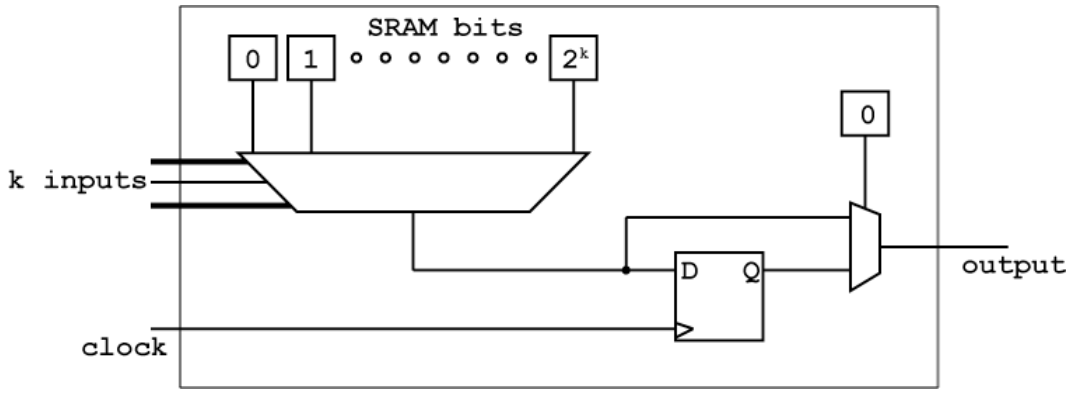


Figure 2. The K-input BLE.

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 is misleading, as the memory is not random access. It is sequentially set via a Flip-Flop chain. However, SRAM is the industry standard name for this type of device.

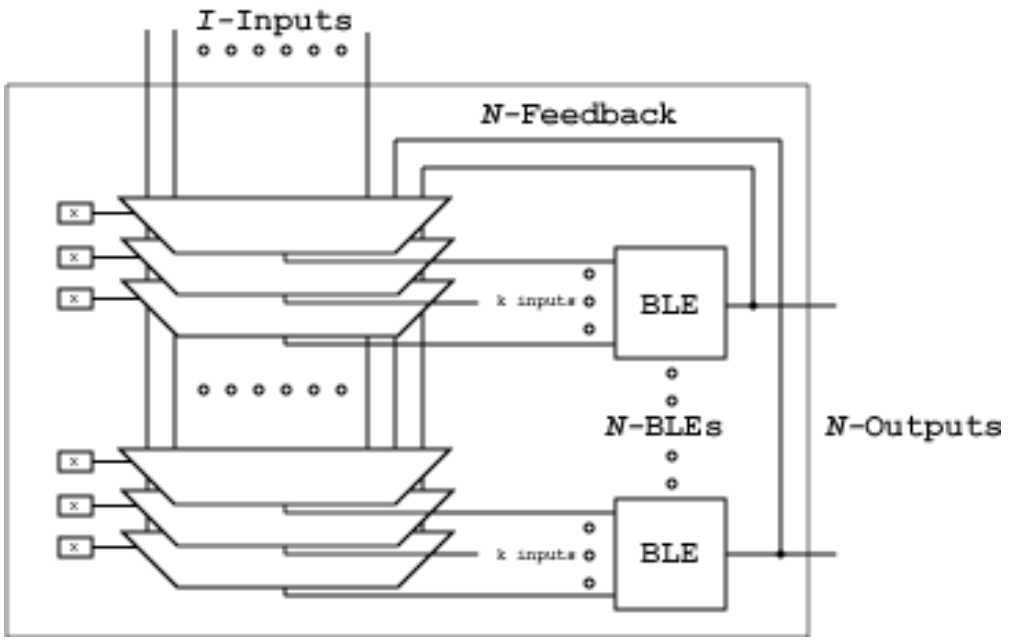


Figure 3. A Single CLB

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 it keeps this local routing out of the C-block matrix.

The CLBs, when arranged in an array with row and column based interconnect, forms the basis of an island or mesh style traditional reconfigurable logic architecture as shown in Figure 1, repeated below 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 a 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.

The I inputs and N outputs from each CLB are connected to a fraction (Fc[2]) of the total 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 have connections to the CLB MUX input.

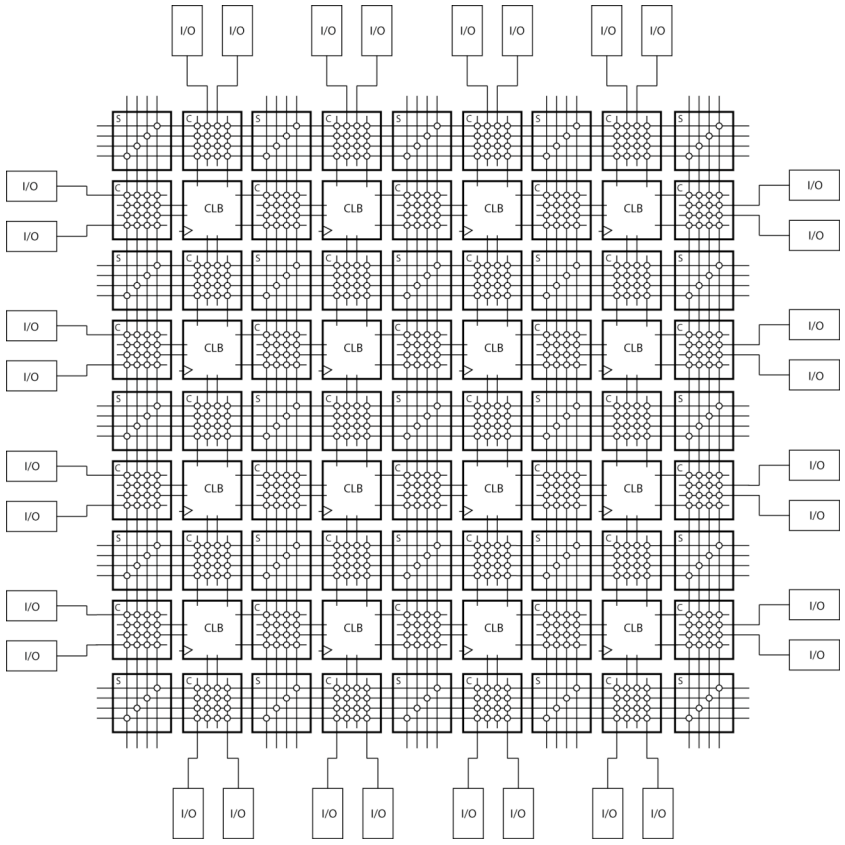


Figure 4. Reconfigurable Logic "Island" Architecture

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, Figure 5. This allows the signal to switch direction, but not switch line.

The state of every LUT, multiplexer, and every switch is held in SRAM that is organized as a long, Flip-Flop based, shift register. The contents of this shift register controls the operation of the FPGA and therefore its implemented design, and is called the bitstream.

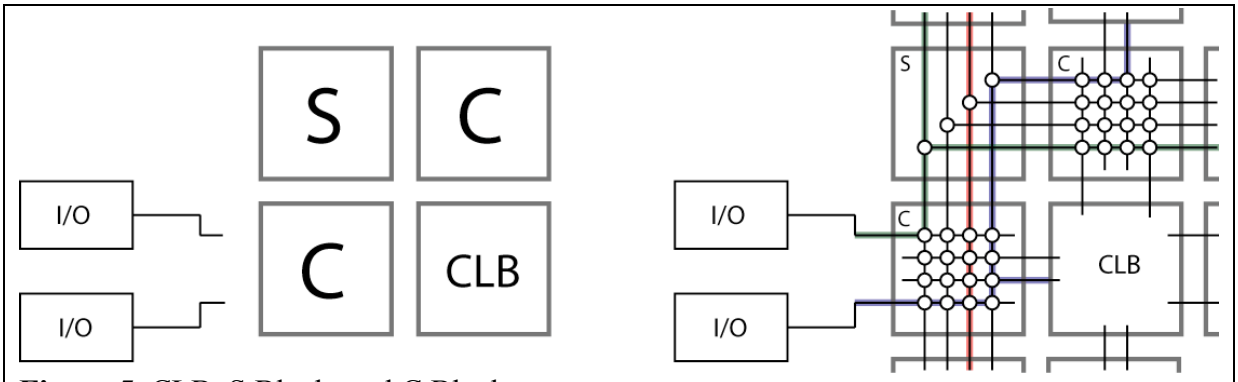


Figure 5. CLB, S Block, and C Block

The CAD Flow for Logic Synthesis

In order of facilitate programming, an existing software called VPR will be uses to route and program the FPGA. Figure 6 outlines the VPR CAD flow [1]. First, the SIS [2] synthesis package is used to perform technology-independent logic optimization of each circuit. Next, each circuit is technology-mapped into 4-LUTs and flip flops by FlowMap [3] (SIS and FlowMap may be replaced with Icarus and Odin, depending on which software gets up and running first). The output of FlowMap is a .blif format netlist of LUTs and flip-flops. The T-VPack [3] program then packs this netlist of 4-LUTs and flip flops into more coarse-grained logic blocks (CLBs), and outputs a netlist in the .net format VPR uses. VPR then takes in this netlist of CLB packed logic as well as a text file describing the FPGA architecture. VPR can place the circuit, or a pre-existing placement can be read in. VPR can then perform either a global route or a combined global/detailed route of the placement. VPR's output consists of the placement and routing, as well as statistics useful in assessing the utility of an FPGA architecture, such as routed wire length, track count, and maximum net length.

VPR was designed for simulation of high level FPGA architectural decisions that takes into account low level area, timing, and power models, and does not actually produce a bitstream for FPGA programming.

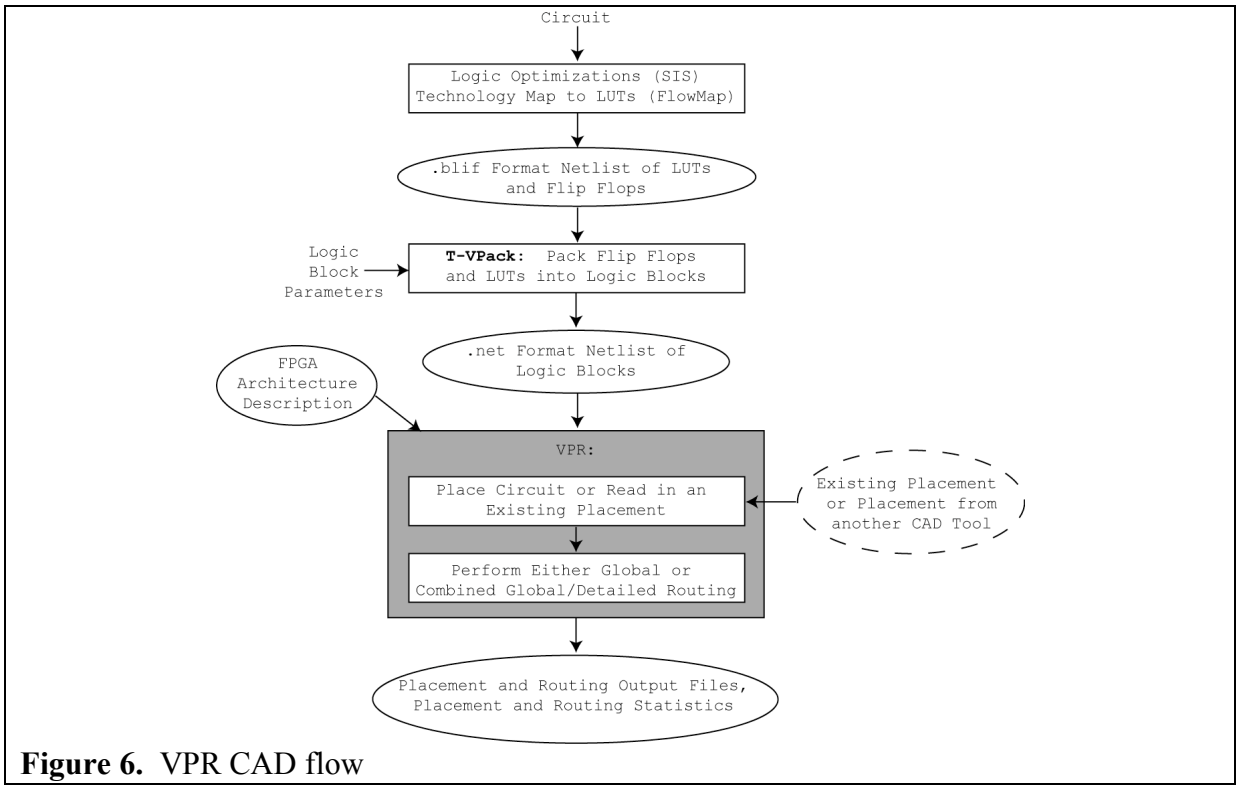


Figure 6. VPR CAD flow

Bitstream Specification[3]

This section specifies the constraints placed on the FPGA architecture by the tool flow the Bitstream Team is implementing/extending. The generation of netlist, placement, and routing information by the TV-Pack/VPR frontend has certain expectations placed on the layout of the CLBs, C-blocks, and S-blocks. This document should serve as a guide to those teams designing these components for the effective integration of these components into a *programmable* FPGA architecture.

CLBs

There are two primary things to consider when constructing the CLB: the interconnection of its constituent BLEs and its connections with the surrounding C-blocks. These and additional issues are addressed below:

1. **BLE Connections** – The Bitstream team goes on the assumption that an arbitrary CLB input can be muxed to an arbitrary BLE. Per the original design document, this makes sense. This allows TV-Pack its requisite flexibility in assigning BLE connections within a CLB.
2. **CLB Input Pin Placement** – The routing performed by VPR has a certain expectation placed on the location of CLB input pins in the architecture. For

[3] Original by Stuart Duerson and Upendra Godse, edited by Brian Degnan 13 April 2006

instance, if a net (signal path) has its sink at "Pin 2" on the south side of a given CLB, having that pin on the north or east side of the CLB defeats the point of place and route in the first place. This specification is needed to ensure that C-block and S-block tracks are properly utilized. For the purposes of this project, this constraint is absolute. A diagram of pin placement is shown in Figure 1.

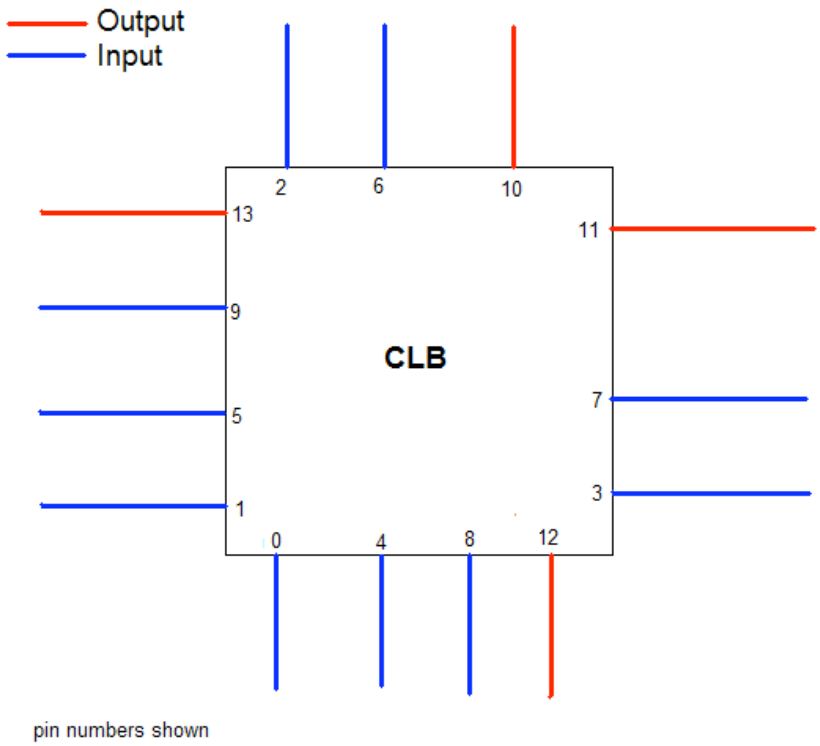


Figure 1. Input/output pin configuration for a CLB.

The general details of where the input and output pins are located and how many exist on each side of the CLB are the most relevant for the CLB and C-block teams

3. **Programming Specification** – The Bitstream team places no constraint on how the programming path is placed between BLEs or the sequential programming logic for the interconnecting muxes in a CLB. The Bitstream Team does however place the constraint that each BLE/mux have its programming logic in identical ordering. That is, if flip-flop 0 on BLE0 programs the 0^{th} bit of BLE0, then flip-flip 0 on BLE 1 should program the 0^{th} bit of BLE1. In summary, the bitstream path can visit BLE/muxes in any order, but the order must be consistent *within* a given BLE/mux. Furthermore, once this order is specified for a given CLB, it would be convenient to see this order used for all other CLB implementations. If this effort could be coordinated it would make things easier.

4. **Disable Bit** - The Bitstream will make available the use of a Disable bit. One Disable bit will be granted per BLE. That is, if BLE0 is not being used for *any* computation, then it shall be disabled via a Disable bit. A suggested location for this bit will be in a flip-flop directly before the flip flop that programs the 0^{th} bit (in relation to the bitstream path).

C-Blocks

For proper track utilization, the input/output pins of each CLB needs to be connected to the tracks in each C-block in a certain fashion. The following paragraphs illustrate the C-Block/CLB pin connections that the tool flow will expect.

In the architecture specification, the parameters Fc_in and Fc_out are used to specify the fraction of tracks each input and output are connected to, respectively. In using these numbers, most teams have made arbitrary assumptions as to which specific tracks they should connect each input to (e.g. connect every other track, etc.). In fact, the flow will expect specific tracks to be accessible to specific input/outputs. Figure 2 will act as a map so we can efficiently specify which tracks need be connected in Tables 1-4.

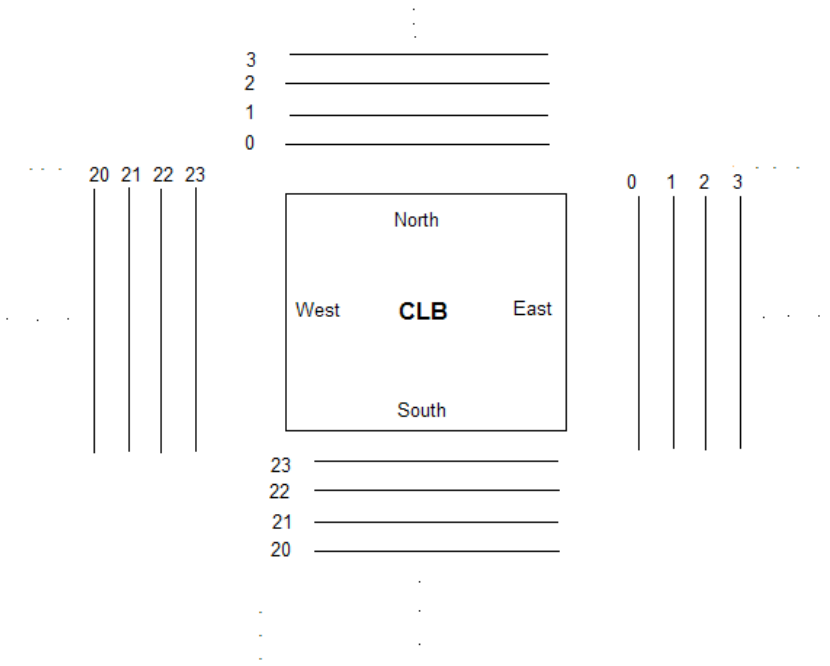
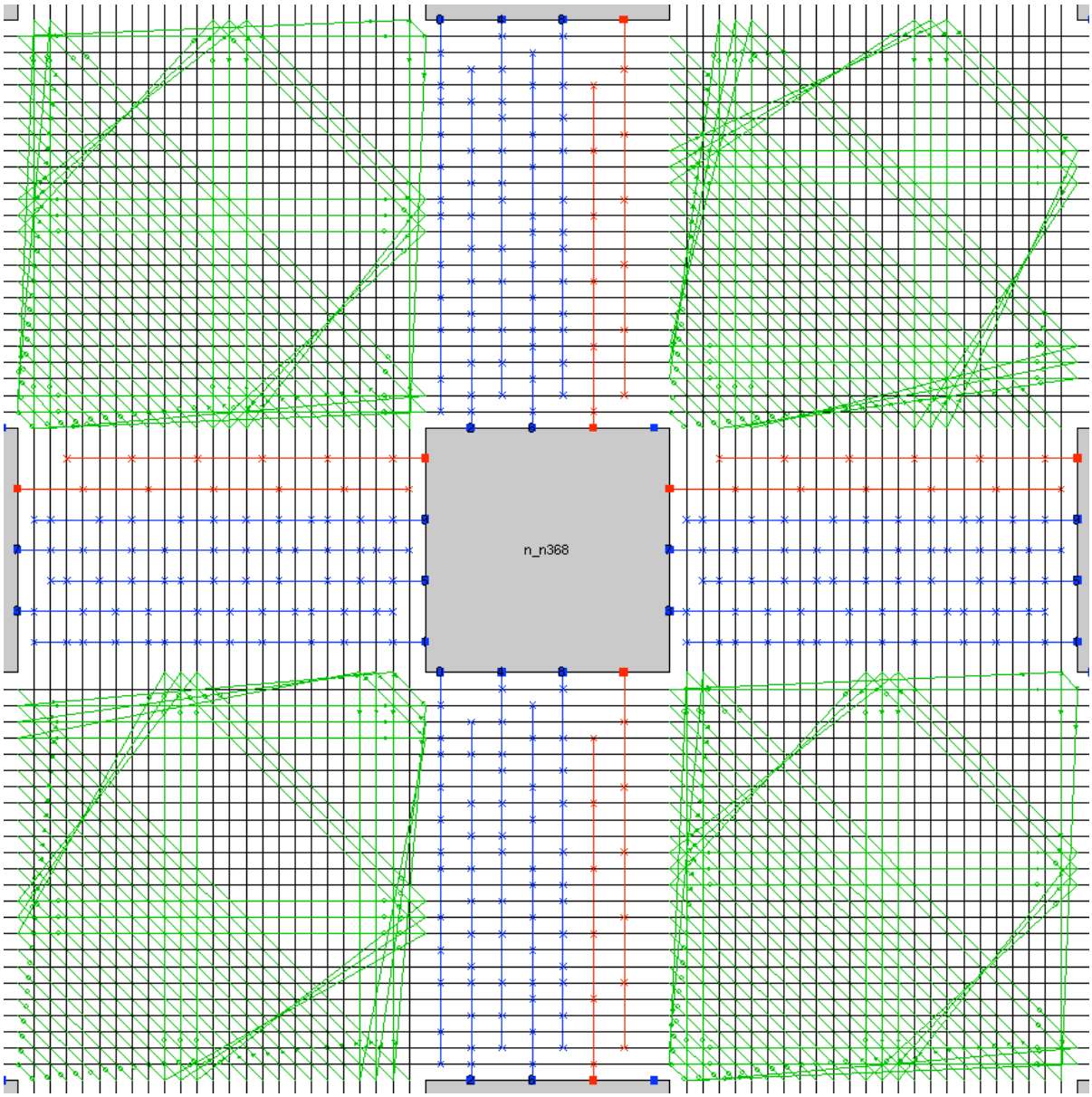


Figure 2. Key for referring to C-block tracks relative to CLB.

There are 24 tracks between each CLB. The following tables will spell out the connections from each CLB pin to each track relative to which side (North, East, South, etc.) the table refers to.



The above picture is from VPR. Please make the CLB connections from the C-Block in the manner that is provided above. This will facilitate programming with the current tools.

Table 1. Pin-Track Connections for North Side of CLB	
Pin #	Track Connections
2	0, 1, 3, 5, 6 , 8, 10, 12, 14, 16, 19, 21
6	0, 2, 4, 5, 7, 9, 11, 12, 15, 17, 20, 22
10	0, 4, 8, 12, 16, 20

Table 2. Pin-Track Connections for East Side of CLB	
Pin #	Track Connections
11	3, 7, 11, 15, 19, 23
7	1, 3, 6, 8, 11, 13, 15, 16, 18, 20, 21, 23
3	0, 3, 5, 7, 10, 12, 14, 16, 17, 19, 21, 22

Table 3. Pin-Track Connections for South Side of CLB	
Pin #	Track Connections
12	1, 5, 9, 13, 17, 21
8	1, 3, 5, 6, 8, 10, 11, 13, 16, 18, 20, 23
4	1, 3, 5, 8, 10, 13, 14, 16, 18, 19, 21, 23
0	0, 2, 5, 7, 9, 12, 13, 15, 17, 19, 20, 22

Table 4. Pin-Track Connections for West Side of CLB	
Pin #	Track Connections
1	0,2, 3, 5, 7, 8, 10, 12, 14, 17, 19, 22
5	1, 2, 4 , 6, 8, 9, 11, 13, 15, 18, 20, 22
9	0,1, 4, 6, 9, 11, 13, 15, 17, 18, 20, 22
13	2, 6, 10, 14, 18, 22

Disable Bit – To save power, the Bitstream team will make available a Disable bit for each C-block. In the event that an *entire* C-block is not being used, s single FF within the C-block will programmed with this disable bit.

Bitstream Path – The Bitstream Teams sees that it is important that a unified scheme be implemented for connecting the programming flip-flops for the C-block implementations. The granularity associated with programming BLEs allows their order in the bitstream to be somewhat arbitrary. However, a standard flip-flop path

for each C-block is necessary to contain the complexity of programming the architecture. We suggest that the various C-block teams meet with the Bitstream team to discuss this.

S-Blocks

The Bitstream Team assumes that an S-Block has the ability to connect any vertical track to any horizontal track.

Disable Bit- The Bitstream team offers the same type of block-applicable disable bit mentioned in the C-block section.

Bitstream Path – Since the S-block is a "connection block", the Bitstream team will make the same argument for a standard programming path for all S-block implementations as mentioned in the C-block section.

I/O and Clocking

The bitstream will be a series of bits fed into a chain of flip-flops that spans the entire FPGA architecture. The FFs that store these bits must be clocked for a precise number of cycles until programming has finished. The Bitstream team will request a chip-level enable/disable signal which will be used to shut off this clocking once programming has finished.

Group Projects

Each team will consist of two persons and will choose a project number. Some project numbers will have competing teams generating designs. The teams choosing projects 1-5, those projects requiring layout, will have two fabrication technology choices. The preferred choice is 0.13u available through MOSIS, and it will be required that at least one team per project number target this technology; however, in doing so it will require that those team members sign a non-disclosure agreement. The project options are listed below.

1	**High Speed CLB**	**3 Teams**
2	**Low Power CLB**	**3 Teams**
3	**High Speed S Block C Block**	**3 Teams**
4	**Low Power S Block C Block**	**3 Teams**
5	**I/O / Clock**	**2 Teams**
6	**Benchmark, CAD to Bitstream**	**1 Team**

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 teams.

- Metal 4 will be reserved for clock distribution, other metal layers, intents and metal related decisions should be specified by each interface team and documented. The I/O pads are a special case, and will have metal usages completely specified by the teams.
- The state FlipFlops, which hold MUX state 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 FlipFlops in the BLE.
- The functioning of the FPGA blocks should be considered from the point of view runtime. Optimize for the run condition and not the programming condition.\

- A global “sleep” signal, which is active-high, should be present in all cells which will trigger the design into a low-power mode.
- 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.

Pin List

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

- bypass clock signal / program clock
- bypass clock enable
- program enable
- bitstream in
- bitstream out
- sleep (high)
- global reset (low)
- VDD 1.2 volts
- VDD 2.4 volts
- VDD Pad, 3.3 volts
- Ground

Project 1 / 2: High Speed / Low Power CLB

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. Teams choosing this project will be responsible for designing and laying out a functional CLB of functional design shown in Figures 2 and 3 with the following parameters: K = 4, N = 4, I = 10. Not explicitly shown in the figures are the SRAM required to hold the configuration of the CLB. The state of every LUT, as well as every multiplexer in the CLB will have to be held in SRAM. The SRAM should be organized as 1-bit flip-flops or registers chained together as a shift register. So, in addition to the logical and routed inputs and outputs of the CLB, there will have to be a bitstream in and a bitstream out as well as some means to selectively clock the SRAM that holds the bitstream. The logic FlipFlop in the BLE should an active low global reset. The blocks should also have a sleep mechanism which is set by the global, active high, sleep pin as well as any other low-power techniques that the low-power teams may implement.

Those choosing project 1 will optimize their design for high speed first, and low power will be the metric of choice for those pursuing project 2. The teams are free to use any logic families or digital tricks of their choice, and 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.

The clock distribution scheme will be routed on metal 4, so if your design does not use metal 4 you will not have to worry about clock routing (Figure 10), just assume it will make it to the clock input pin. The CLB block will be provided with 10 input lines and 4 output lines from the C-Block as well as two non-overlapping clock lines from the Clock Team.

Project 3 / 4: High Speed / Low Power S Block C Block

While projects 1 and 2 are designing the islands of the FPGA, projects 3 and 4 are responsible for the sea, or the interconnect. The interconnect blocks will be of the architectures seen in Figures 5, 7, and 8 with the following parameters: a planar style S Block detailed functionally in Figure 8, with a possible but not mandatory switch implementation, the number of vertical and horizontal tracks or wiring paths, W = 24, the percentage of tracks that each input is connected to, Fc_in = 0.5 (so 12 of the 24 lines are connected), while each output, Fc_out = 0.25 (so every 4^{th} line is connected). As is the case with projects 1 / 2 the state of all switches will be held in SRAM (not shown in the figures) organized into one giant shift register. The interconnect will have to interface its bitstream with that of the CLB design as well as provide a bistream in and out that tiles, this will require some ingenuity as the SRAM of the entire chip will be one long shift register which will probably zig-zag through the chip, thus having a bistream that flows in one direction on one row of tiles and flows in another direction on the next row.

Furthermore, it will be required (mostly because VPR does not support it) that no doglegging will be allowed on input pins in the C block; however, doglegging is allowed on the output

pins. Much of the FPGA literature eludes 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 more routing options, and instead opt for a faster design where the inputs are buffered off of the tracks and multiplexed into the input pins as shown in Figure 7.

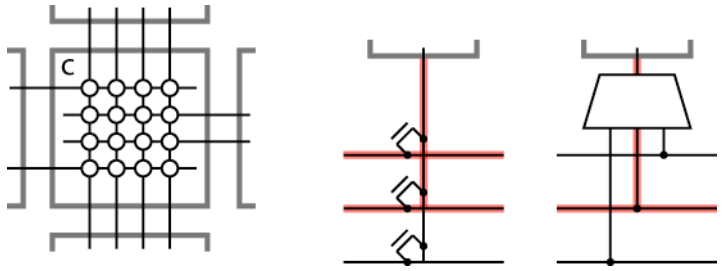


Figure 7. C Block, doglegging, no doglegging

The teams should keep in mind that their design should be easily stretched to conform to the CLB design as the final CLB area will not be determined for some time. It is also in the interconnect fabric that the grid size for the entire chip, and the clock distribution scheme (Figure 10) will be set. Spaces need to be left in a regular grid fashion for the clock distribution buffers that appear at every branch in the clock tree. As with projects 1 and 2, if metal 4 is not used in the interconnect layout then the team does not need to worry about leaving channels for clock routing, but rather just leaving spaces for the clock buffers. Doglegging may be used on the output lines from the CLB. One output from the CLB may be tied to multiple lines in the doglegged scheme. Furthermore, the input/output lines from each CLB must be exclusive to that CLB. Adjacent CLBs may not share input or output lines across a C-Block. Any active blocks should also have a sleep mechanism which is set by the global, active high, sleep pin as well as any other low-power techniques that the low-power teams may implement.

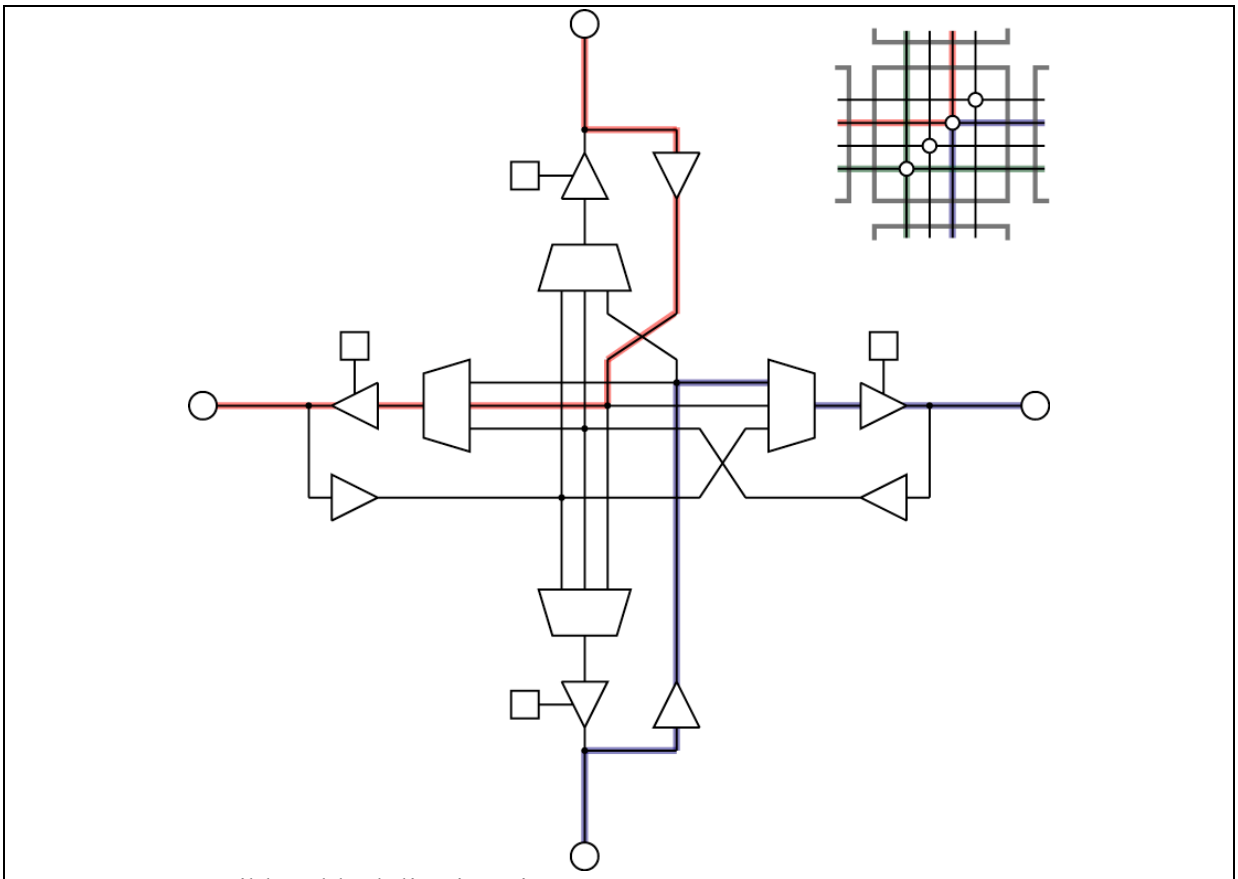


Figure 8. A possible S block line junction

The S-Block line junction proposed in Figure 8 produces the desired function at a junction; however other, more appropriate, may exists and should be explored. The C-Block will supply 10 input lines and 4 output lines from the CLB.

Project 5: I/O / Clock

These groups will be responsible for creating the I/O pads to be used in the padframe for the FPGA, as well as the buffers that connect the I/O pads to the C blocks as shown in Figures 4 and 9. The I/O pads should have some considerations for voltage level shifting, and ESD protection. For the 130nm process, 3.3 volts is the output voltage of the I/O[4]. The number of I/O pads per C block is denoted by io_rat[5] and is desired to be 4, but will probably change pending the packaging pin count and the CLB grid size allowed by our die.

[4] Please see the 130nm documentation regarding construction of "high voltage" FETs.
[5] "io_rat" is the ratio of I/O pads per border C-Block as required by VPR. This is a variable set in the architecture description.

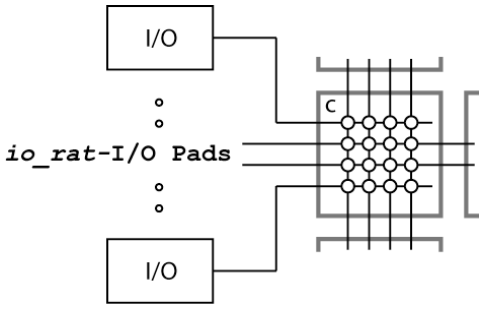


Figure 9. I/O Pads connected to C Block

These groups will also be responsible for designing and implementing a variable frequency clock generator as well as the buffers used in the H Bar clock distribution scheme for the CLBs, shown in Figure 10. While the teams working on projects 1-4 will be concerned with

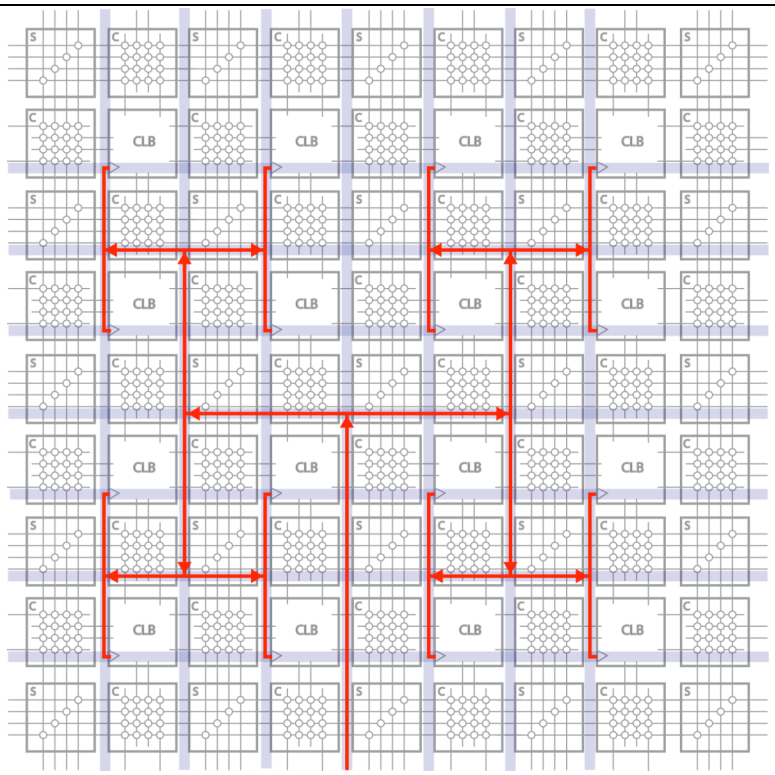


Figure 10. H Bar Clock Tree

making sure that there is space to route the clock tree, the clock tree design and characterization lies with project 5. The project 5 teams will also be responsible for supplying non-overlapping clocks which are generated at the leafs of the H-Tree for the

purpose of FlipFlop clocking. A non-overlapping clock generator will be present in each and every CLB block. The operational frequency of these devices should be able to handle 100Mhz.

Project 6: Computer-Aided Design to Bitstream and Benchmarking

The current CAD flow (Figure 6) facilitates the simulation of high level FPGA architectures, but does not actually produce a bitstream for programming an FPGA. The team that chooses project 6 will be responsible for adding an extra step to the CAD flow to generate a bitstream targeting our specific FPGA implementation.

VPR outputs a placement file (.p) and a routing file (.r). The routing file details how every switch in every C and S block are configured, while the placement file simply specifies what CLB ended up where. The placement file does not describe how the individual CLBs are configured. The output of T-Vpack, the technology mapped and packed netlist file (.net) specifies how the internals of the CLB are configured, i.e, which inputs to the CLB are tied to which BLEs within the CLB, but the netlist does not contain the information about how each individual BLE is configured, i.e, the stored state of the LUT. This information resides in the .blif file output from FlowMap.

Therefore, the addition to the flow will have to take information from the .blif, .net, .r, and .p files output from the current flow to figure out the exact configuration of the FPGA to generate a bitstream for configuring out actual FPGA implementation.

In addition to the backend of VPR, the frontend will be targeted as well. In particular, both the VPR frontend will be tested and a state-of-the-art FPGA board from Xilinx may be used. The Xilinx board in question is in Professor Mooney's lab. Example benchmarks include AES and the MCNC logic synthesis benchmark suite. The focus will be on real-world applications and comparisons.

Project Organization

This project is worth 50% of each and every student's grade. As such, it needs to be carried out with effort proportional to approximately half of a course grade. Given the importance of properly interconnected project teams, together with a need for efficiency, we will use the following guidelines.

Every class will be held every Tuesday/Thursday with very few exceptions. In each class, some lecture material may be presented by Professor Mooney; furthermore, a few teams will present their status each class using a schedule organized by Prof. Mooney.

After the first two weeks, each team will turn in their architecture and layout of the basic elements (defined on a per project basis). On March 9, the first item is due: a writeup of

what the team plans to do. A written report 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 which were made or seem to be needed should be clearly stated up front. Please do not expect questions to be answered in 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 week some item will be due.

Anyone can make any suggestion at any level (circuit, architecture, etc.) at any time. Prof. Mooney will decide what suggestion to take/follow. Arguments are welcome but need to be carried out in person.

In general, each team picks from one of the two following approaches: (i) ultra-low power and fast, or (ii) super fast, low power.

There will be some evolution – perhaps significant! – of the specification and expectations; this will be taken into account in grading the final project. All layouts will have to characterize power with HSPICE. We will provide a lab assignment/tutorial on this later. Note that all clocking is two-phase. It must be delivered as two-phase, but the internal clocking methods can be proposed.

Project Teams

The following is based on the result of emails sent.

1 High Speed CLB
- **Team 1.1: Michael Heard and Eric Mullen — Process: .13u**
- **Team 1.2: Dean Lewis and Demijan Klinc — Process: .18u**
- **Team 1.3: Nob Kladjarern and Sean McAllister — Process: .13u**

2 Low Power CLB
- **Team 2.1: Matthew Livianu and Omar Mustafa — Process: .13u**
- **Team 2.2: Abdemanaf Tambawala and Kemal S. Demirci — Process: .13u**
- **Team 2.3: Sundeep Roy and Shane Mahon — Process: .18u**

3 High Speed S Block C Block
- **Team 3.1: Mrinmoy Ghosh and Tapobrata Bandyopadhyay, Process: .13u**
- **Team 3.2: Jason Uher and Andrew Kerr — Process .18u**

4 Low Power S Block C Block
- **Team 4.1: Christopher Day and Adrian Sequeira — Process: .13u**
- **Team 4.2: Anirudh Prasad and Mrunal Shah — Process: .18u**
- **Team 4.3: Chris Scott and Dae Hyun Kim — Process: .13u**

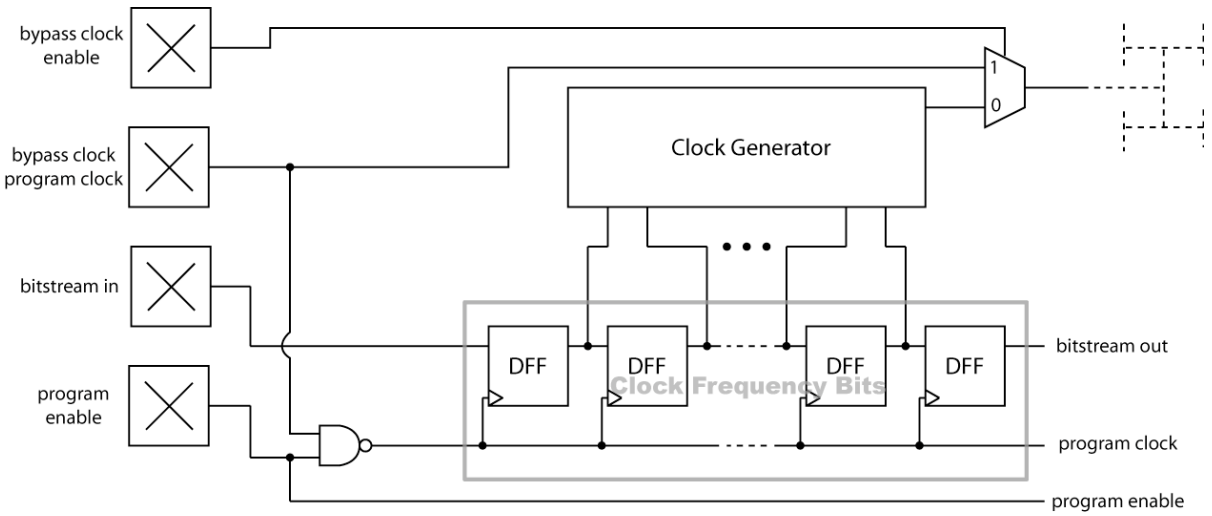
5 I/O / Clock
- **Team 5.1: Rachel Harris and Alvin Yates — Process: .13u**
- **Team 5.2: Harry "Bo" Marr and James Holland — Process: .18u**

6 Benchmark + CAD to Bitstream
- **Team: Stuarts Duerson and Upendra Godse — Process: .13u**

ERRATA

Clock Structure

The frequency of the clock will be set by a number of bits from the bitstream. Each clock team will decide this specific number. Due to the pin-limited nature of the packaging, we will be sharing some pins.

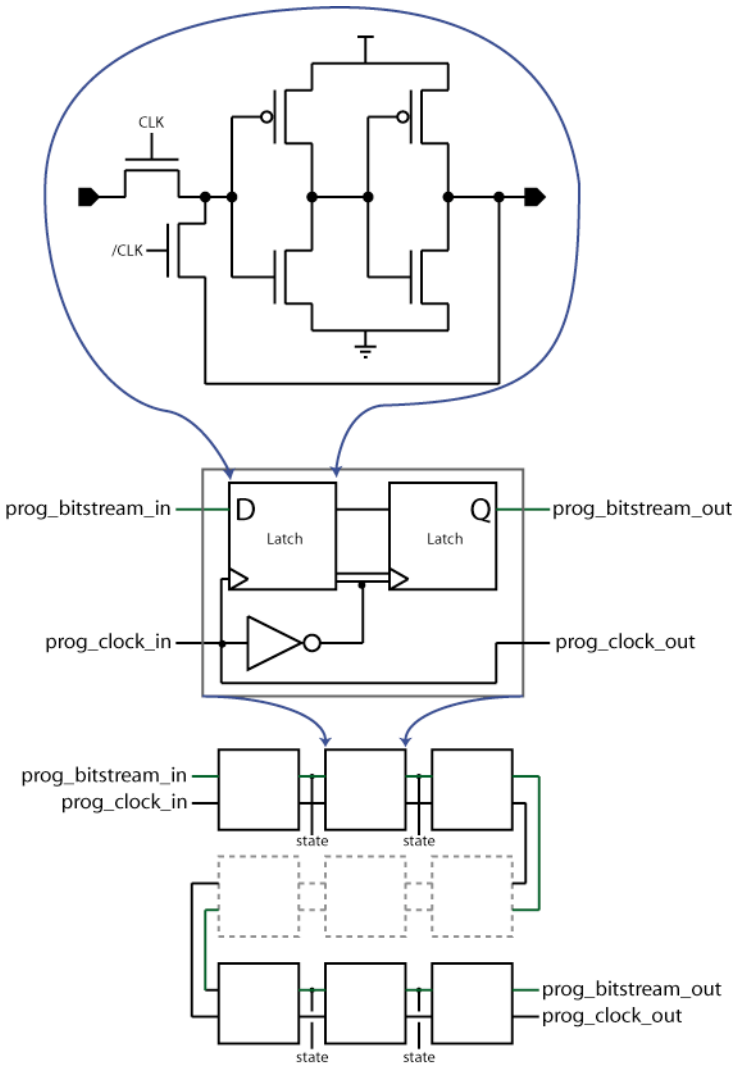


Errata Figure 1. Shared pins of the clock and programming circuit.

A clock bypass will exist for the case of clock malfunction or single-stepping of the device. The clock bypass pin will be used as the programming clock during program mode. The "program enable" line will enable programming, and the line may be used across the IC to determine mode if desired; for example, the program enable line may be used to power on/off the first latch of the bitsteam flipflops.

State Flip Flops

The state flip flops will use a global programming clock with is isolated from the H-Tree clock. This clock will snake through the IC, along with bitstream line and any other necessary lines. There are effectively two types of flip-flops, those which hold state, and those which drive a current.

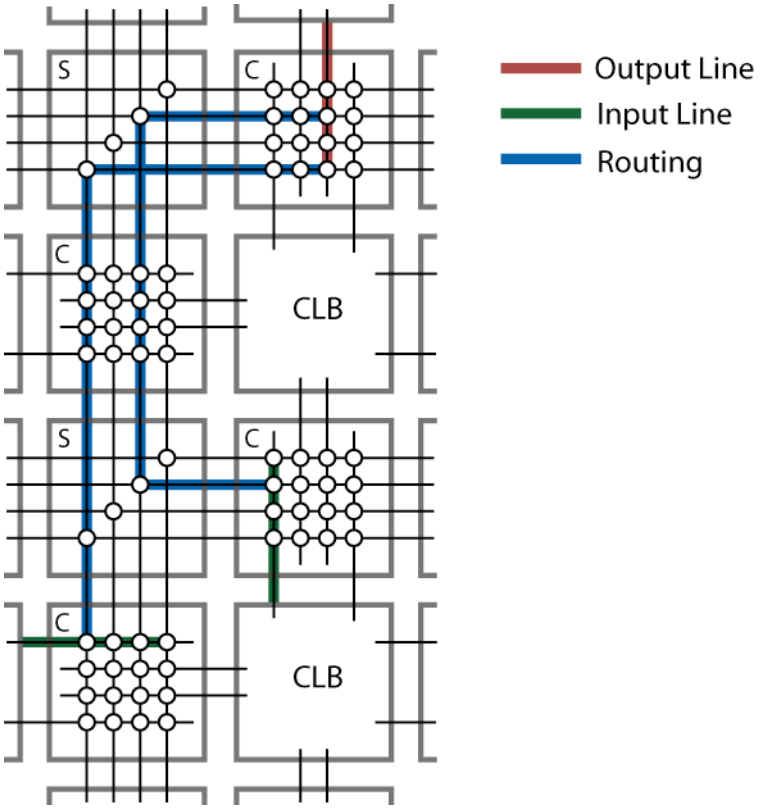


Errata Figure 2. Relationship of the state flip-flops.

The last stage transparent latch of the flip-flop may be long and weak for holding selection states, but he driving latch must be sized appropriately in the case that it will be sourcing a current. The first latch in the flip-flop may me designed for other low-power conditions as it is of moot functionality during the run state.

Doglegging

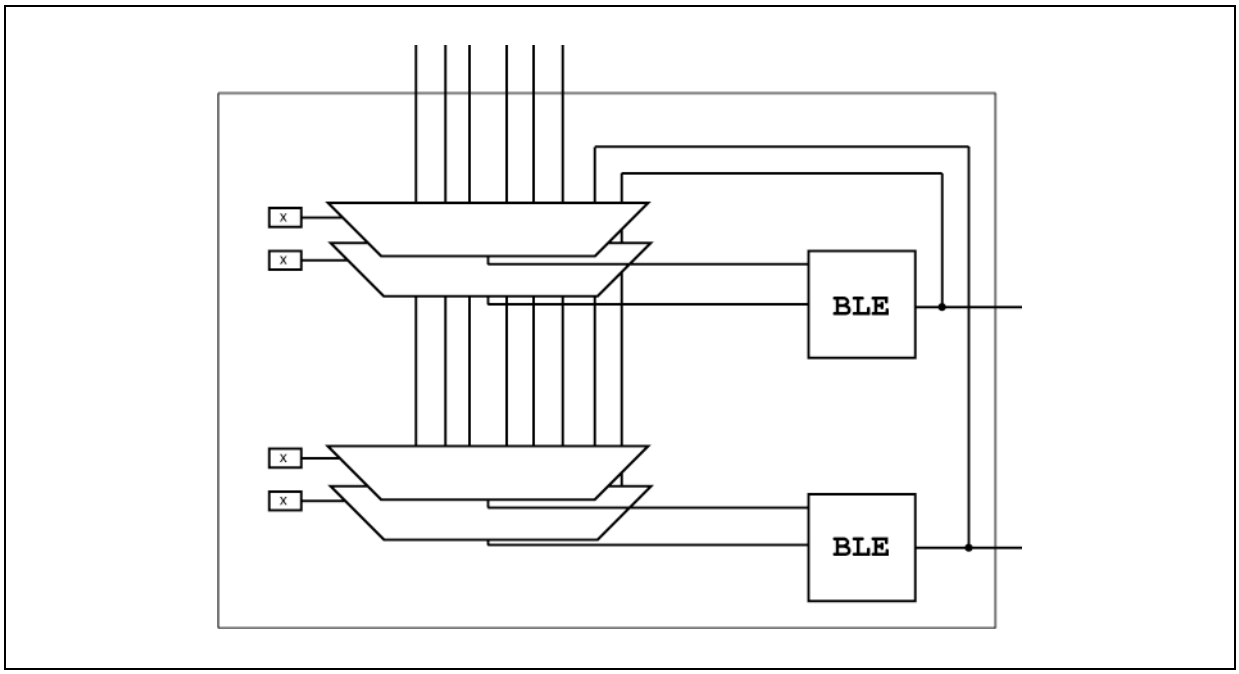
It has been specified that there was no doglegging to occur on the inputs to the CLB. This example hopes to clarify the scope of this statement. Doglegging is a way to switch lines, or fan-out a connection. In fact, it could possibly be the only way to switch lines in the fabric depending on the CLB. A CLB with only one BLE could not support a fan-out of any type.



Errata Figure 3. The output line has doglegging which allows multiple lines to be routed to the next input.

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. We specified that doglegging would not be allowed on the input lines as our tools do not support it; however, there's no reason for the hardware not to support it. The same structures could be used for the output line connections and the input line connections, and the constraint could put on the software to not dogleg the input.

CLB Architecture



An example of a 6 input, 2 output CLB with 2 BLE; however, there should also be additional lines for power and ground so that the BLE inputs will be tied to a known state. There should be no "floating" inputs if it possible that the input will drive a digital input into a metastable state, and thereby have high current draw.

Update Summary 04 May 2006

Clock Tree Errata.
The circuit diagram has been slightly updated to reflect the global program enable.

Update Summary 25 April 2006

Clock Tree Errata.
The circuit diagram has been slightly updated.

General
Clarifications were made throughout the document.

Update Summary 12 April 2006

CLB Design.

The clock generator for the BLEs should be reasonably equidistant from the all BLEs. Please consult the with the clock teams to get area estimates. Metal-4 is reserved for the clock tree as it must run many directions for the H-Tree. The BLE clock generator will be near the middle of the BLE cluster.

General Programming
The state Flip-Flops for the programming chain will also need require a Φ1 and Φ2 and all teams should thing about facilitating the following. The boundary of each team's cells should have a minimum of 4-accessible pins for programming: Φ1, Φ2, state datain, and state dataout. Φ1 and Φ2 for the state FFs will be generated locally or globally depending on how the teams decide. The final IC will have 4 programming pins which are accessible to the user: program clock, program enable, program datain, and program dataout. The program dataout pin is for bit stream sanity checks.
Pins
A pin list has been added to the general design section.

Clocking
Some of the clocking circuitry will be share pins with the programming circuitry have been added to the Errata section.

Bitstream Specification
The bitstream specification by Stuart and Upendra have been added as a section to the document. The C/S-blocks should have the connection in the same way as the pictures in the bitstream document. Incorrect information was given in class, and it has been corrected in this document.

State Flip Flops
The errata section contains a discussion about state flip flops.

Update Summary 14 March 2006
Fixed the spelling of errata, as well as other spelling mistakes.
Merged Project 6 and Project 7.
Updated 09 March 2006 errata to be more clear, and updated the document to reflect such.
Added footnotes for VPR terminology.
Added doglegging section in errata.
Expanded CLB example in errata.

Process voltage
The digital voltage for the 130nm process is 1.2volts; however, 2.4 volts can be used with precautions, and 3.3 volts is the interface voltage. The I/O teams are the only teams who may use 3.3 volts.

Φ Clocking
Specified that non-overlap clock generation will occur in each CLB, and that the target speed will be 100MHz.

C-Block
The input and output lines from a CLB should not be shared with an adjacent CLB through the C-Block.

Sleep signals
A global sleep signal should be present in all blocks. This signal is active high, so the pin may be tied low if sleep is disabled. This signal is to be present in all blocks for all teams; however, this is exclusive from other low-power ideas which may be included by the low-power teams.

Clock sleep
A global sleep signal should be present in all blocks.

MUX states
All MUX designs where is it possible that an input may float should have the ability to tie a floating input to a known state i.e, ground or a supply voltage. This is primarily regarding the CLB select lines for unused BLEs.

Update Summary 09 March 2006
CLB/C-Block interconnect.
The C-Block group is responsible delivering 10 lines to the CLB block as inputs and accepting 4 lines as outputs.

Φ Clocking
A single phase clock will be delivered using an H-Tree and then the non-overlapping 2-Φ should be generated near the CLB. The CLB teams should assume that you have falling edge FlipFlops with a non-overlapping clock.

S-Block Interconnect
The design for the S-Block line junction was just a sample design and is not explicitly specified by this document; however, the functionality of the sample circuit is required Teams should consider other alternatives if they arise while preserving functionality.

State FFs
The state FlipFlops will require an enable line to prevent erroneous programming during regular execution after programming is completed. The program enable line should be active high.

MUX bits
Floating lines should be avoided. MUXes may have extra bits to tie lines to know states.

BLE FFs
The FlipFlops in the BLE structures should all be connected to a global reset pin.

Doglegging
The output lines of the CLB may enable doglegging, which effectively allows a form of fanout on the output lines. One output from the CLB may be tied to multiple C-Block lines in the doglegging scheme.

Programming Considerations
The programming of the FPGA should be considered from the point of view of normal, post-programming operation. Optimize for the run condition and not the programming condition.

Citations

(1) E. M. Sentovich et al, "SIS: A System for Sequential Circuit Analysis," *Tech. Report No. UCB/ERL M92/41*, University of California, Berkeley, 1992.

(2) J. Cong and Y. Ding, "FlowMap: An Optimal Technology Mapping Algorithm for Delay Optimization in Lookup-Table Based FPGA Designs," *IEEE Trans. CAD*, Jan. 1994, pp. 1 - 12.

(3) V. Betz and J. Rose, "VPR: A New Packing, Placement and Routing Tool for FPGA Research," *Seventh International Workshop on Field-Programmable Logic and Applications*, 1997, pp. 213 - 222.

Useful Links

http://www.eecg.toronto.edu/~vaughn/vpr/