FPGA Compiler Project

EE 599
Fall 2010
M. A. Breuer

Ming Hsieh Department of Electrical Engineering
Viterbi School of Engineering
USC
Compilation for the Emulator

- This project deals with the design of a compiler to support the emulation of a digital circuit.
- Four person teams will do the project.
- Each person will be responsible for algorithm development, coding, debugging and integration with respect to their assigned module.
- The TA will support and grade this project. You will be provided with test cases, a detailed router visualizer, and other items to be discussed.
The Emulation System

- Workstation (Compiler)
- I/O Unit
- Circuit description
- FPGA emulation system
Motivation (a.k.a. simulation is too slow)

- Consider the FPGA based emulation system architecture shown previously. A net list of a circuit C to be emulated is read into a compiler that runs on a workstation. The format for the net list is shown elsewhere. After compilation, the compiler outputs a configuration file to the FPGA system that configures the FPGAs so that they will emulate the circuit C. Then the workstation supplies simulation data to the FPGA system, the system is clocked, and the output sampled. By repeating this process for all the input test data, the circuit is emulated. The focus of this project is to build the compiler.

- *(Note: In the real world such emulation systems exist, though we do not have one at USC. Quickturn, a Division of Cadence, marketed one such system; another was marketed by Meta, a Division of Mentor Graphics. For single FPGAs, many such compilers exist, such as the one marketed by Xilinx.)*
The FPGA System

L1 → S1
L1 → S2
L1 → S3
L1 → S4
L2 → S1
L2 → S2
L2 → S3
L2 → S4
L3 → S1
L3 → S2
L3 → S3
L3 → S4
L4 → S1
L4 → S2
L4 → S3
L4 → S4
L5 → S1
L5 → S2
L5 → S3
L5 → S4
L6 → S1
L6 → S2
L6 → S3
L6 → S4
S1 → SA
S2 → SA
S3 → SB
S4 → SB
System Description

- The previous slide shows the FPGA emulation hardware. Each block is an FPGA. Blocks L1 to L6 are used to emulate the logic and flip-flops in a circuit. Blocks S1 to S4 are used only for interconnection purposes, as are blocks SA and SB.

- (Again in the real world, many thousands of FPGAs are used in such a system. In some cases these FPGAs are custom designs.) The interconnection lines shown are busses and are hardwired. Those between Li and Sj are $N_{LS}$ bits wide; those between Sj and SA or SB are $N_{SS}$ bits wide. The value of $N_{LS}$ and $N_{SS}$ will be specified at run time. They each will be an integer in the range from 1 to 32.
An FPGA

- This figure shows a 4 X 4 array of Configurable Logic Blocks (CLBs). In general, there will be an array of M X M CLBs, where for this project M will be specified at compile time.
- The array of CLBs is surrounded by rows and columns (channels) of interconnect.
Details of a Switch Box

1: via
0: no via

1: no cut
0: a cut
The previous slide shows more details on the interconnect. Here we have shown that each channel has three wires. In general, the horizontal channels have H tracks of wires, and the vertical channels have V tracks. The value of H and V will be specified at run time.

Whenever a horizontal track on metal layer 1 crosses a vertical track on metal layer 2, a complex Xbar switch exists, indicated by the red square. This switch, in turn, is composed of four programmable cut switches, indicated by the blue diamonds and one programmable via, represented by the block dot. Each diamond is associated with a configuration flip flop, which, if set to 1 leaves the wire intact; if set to 0 cuts the wire. Each programmable via is associated with a flip-flop, which, if set to 1 indicates that metal 1 and metal 2 are connected; if set to 0 they are not connected. There are thus 32 different configurations for a red switch. Referring to the ends of the four wires associated with a red switch, as N, S, E, W, several useful connections can be made, such as: N-S and W-E; W-N but not S-E; W-N-E-S; W-N-E, etc. The five configuration flip-flops for a switch are in the order N, S, E, W, V. The switches in an Xbar are ordered from left to right; then from top to bottom. The I/O pads are ordered clockwise, starting at the top left corner. There are a total of 4K I/O pads per chip, K on each side.
CLBs and I/O Pads

R-S or D or J-K F/F or a gate

I/O pad

(c) M.A. Breuer
The previous slide indicates some details regarding the 4-input 1 output CLBs and the I/O pads. Each CLB is associated with four configuration flip-flops that determine the function of the CLB. The configuration mapping is shown in Table 2. If a CLB is used as a RS (JK) flip flop, then the R (J) input is the topmost one, and the S (K) input is the second one. The other inputs should not be connected to any wire in a channel. If the CLB is used as a gate, then any input can be used or not used. In any case, any input not used must not be connected to any wire in a channel. So if you assign a CLB to be a 3-input NAND, then the 4th input is ignored.
Circuit and Input Description

INPUT(G1)
INPUT(G2)
INPUT(G3)
INPUT(G4)
INPUT(G5)

OUTPUT(G16)
OUTPUT(G17)

G8 = NAND(G1,G3)
G9 = NAND(G3,G4)
G12 = NAND(G2,G9)
G15 = NAND(G9,G5)
G16 = NAND(G8,G12)
G17 = NAND(G12,G15)
## Example Coding of CLBs

<table>
<thead>
<tr>
<th>PROGRAMMABLE CLB</th>
<th>CONFIGURATION DATA</th>
<th>FUNCTION OF CELL</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0 0 0 0</td>
<td>RS flip flop</td>
</tr>
<tr>
<td>0</td>
<td>0 0 0 1</td>
<td>JK flip flop</td>
</tr>
<tr>
<td>0</td>
<td>0 0 1 0</td>
<td>D flip flop</td>
</tr>
<tr>
<td>0</td>
<td>0 0 1 1</td>
<td>inverter</td>
</tr>
<tr>
<td>0</td>
<td>1 0 0 0</td>
<td>buffer</td>
</tr>
<tr>
<td>0</td>
<td>1 0 0 1</td>
<td>NAND2</td>
</tr>
<tr>
<td>0</td>
<td>1 1 0 0</td>
<td>NAND3</td>
</tr>
<tr>
<td>0</td>
<td>1 1 1 0</td>
<td>NAND4</td>
</tr>
<tr>
<td>1</td>
<td>0 0 0 0</td>
<td>NOR2</td>
</tr>
<tr>
<td>1</td>
<td>0 0 0 1</td>
<td>NOR3</td>
</tr>
<tr>
<td>1</td>
<td>0 1 0 0</td>
<td>NOR4</td>
</tr>
<tr>
<td>1</td>
<td>0 1 1 1</td>
<td>EOR</td>
</tr>
<tr>
<td>1</td>
<td>1 0 0 0</td>
<td>NOT-EOR</td>
</tr>
<tr>
<td>1</td>
<td>1 1 0 0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 1 1 0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 1 1 1</td>
<td></td>
</tr>
</tbody>
</table>
What is to be done?

- **Read** in the parameters that determine the physical values for the emulation system, such as the number of tracks per channel and the number of I/O pads. (Format specified elsewhere.)

- **Parse** and process the input description of circuit C and build all necessary related tables. (Format specified elsewhere.) An error message should occur if the circuit contains element types not realizable for our CLBs.

- **Partition** the circuit, and **assign** gates and flip-flop to one of the 6 Li’s FPGAs.

- **Assign (place)** gates and F/F already assigned (partitioning) to an FPGA to specific CLBs within each FPGA.

- Optional - assign logic signals to the input terminals of each CLB used.

- **Pseudo-interconnect** the pins from one FPGA to another FPGA. (Since they are already hardwired, this is really a pin assignment problem.)

- Finish all **CLB and I/O pin assignments** and **interconnect** the CLBs on each chip.
Example of Intra-routing
An Example of Inter-routing
More to be done

- Note: The process does not have to be carried out exactly in this order. For example, one can first assign I/O signals to pins and then route between FPGAs. If no solution exists, one can try another assignment. Once the signals between FPGAs are assigned to wires, then each Li FPGA can be routed. The problem here is to determine which is harder to do, the inter (between) FPGA routing, or the intra (within) FPGA routing.
Some Constraints

- A fixed FPGA and system architecture
- A fixed (input parameter) number of I/O pins - they are each bidirectional - namely #IO;
- A fixed number of bits per bus
- A fixed number of horizontal and vertical routing channels
- A fixed number of CLBs per chip
- Gate and I/O pad delays
- Gate and F/F types
- Etc.
Some Objectives

- The problem is to map a circuit C onto this emulation system subject to the given constraints, i.e. input parameters. If you cannot find a solution then we can increase some of the parameters, such as allowing for more CLBs per FPGA, or more tracks for routing within an FPGA, or for larger busses between the interconnect FPGAs.

- You can specify your own objective function, such as to minimize the number of FPGAs used, or to minimize the longest delay in a path between a flip-flop output and a flip-flop input. You can consider attributes such as power dissipation, routing congestion, etc.
Group Personnel and Duties

- Person 1: Deals with the routing (intra) of an FPGA.
- Person 2: Deals with the routing (inter) between FPGAs—this reduces to assigning signals to FPGA pins.
- Person 3: Deals with partitioning the logic between FPGAs.
- Person 4: Deals with assigning logic to CLB cells.

There is one lead person on each major phase of this project, as described above. All students are assumed to contribute ideas to all phases, but only the lead person is responsible for documenting the final algorithm, programming and debugging.

Each group must have a group leader who is responsible for calling group meetings, integrating individual reports into a group report, and communication with an assigned TA regarding group meetings.
Your grade will be in two parts. 20% of your grade will be a group grade, based on the success and quality of your total solution. 70% of the grade will be an individual grade, based on the section of the project that you were the lead, and the quality of this work. As we will discuss in class, all phases of this project are inter-related. Thus, in selecting a procedure to solve one specific task, we are looking to see how the overall specifications of the project influenced your decisions. You may in fact, consider an iterative approach where you do re-partitioning, re-placement, and re-routing. For example, if a FPGA cannot be routed because of lack of enough channels, maybe the placement procedure can push items further apart so there is more room for routing.

Grading will be based on several documents you turn in, as well as the code and the results. If you successfully produce a configuration file, then we will develop a program to read your configuration file and reconstruct a net list. If the resulting circuit is the same as the original one, then you have achieved one major goal - a correct configuration. Interim task will be required that deal with the quality of your partitioning, placement, and your routing techniques.
What we are Looking for?

- This is not a programming class, but rather a CAD class dealing with constraints, objectives, algorithms and heuristics
- Your original ideas are the most important thing
- Your ability to communicate and illustrate your ideas to the TAs is very important
- After that is implementation and documentation and executing test cases.
- We are not that interested in CPU time
- We are somewhat interested in quality of results. But if you come up with a really innovative approach that later we find does not work so well, that is OK too.
Delays, power dissipation and area

- Each gate will be associated with a delay value of 1, 2 or 3.
- Each signal that goes through a switch box picks up a unit of delay (per switchbox).
- Each signal that goes through a I/O pad of an FPGA picks up 3 units of delay (per pad).
- Each gate dissipates $D^2$ nano-watts, where $D$ is its delay.
- Some gates must be placed adjacent to other gates as illustrated below.

<table>
<thead>
<tr>
<th>Gi</th>
<th>Gj</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Denoted by H: Gi, Gj

<table>
<thead>
<tr>
<th>Gi</th>
<th>Gj</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gk</td>
<td>Gl</td>
</tr>
</tbody>
</table>

Denoted by S: Gi, Gj, Gk, Gl

<table>
<thead>
<tr>
<th>Gi</th>
<th>Gj</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Denoted by V: Gi, Gj
General Overview

- Which design flow are you going to use, and why? How are you going to address the situation if your embedding is not routable?

Design Flow A
- Partitioning
- Placement
- I/O Pin Assignment
- Intra-FPGA Routing
- Inter-FPGA Routing

Design Flow B
- Partitioning
- Inter-FPGA Routing
- I/O Pin Assignment
- Placement
- Intra-FPGA Routing
More on flows

- We plan to support the two flows discussed next, in terms of test cases and analysis of results. If you plan to use a different flow that cannot use the inputs we provide, or be evaluated as we will describe, then please get permission from your TA before proceeding with your work.
Input (and Output) Netlist, Data and Format for Design Flow “A”
Partitioning operation

G1 → G8 → G16
G2 → G8 → G12 → G16
G3 → G8 → G12 → G17
G4 → G9 → G15 → G17
G5 → G9 → G15 → G17

G1 → G9
G2 → G9
G3 → G9
G4 → G9
G5 → G9
Objectives and constraints

- **Constraints**
  - Fixed number of I/O pads per FPGA
  - Fixed number of CLBs per FPGA
  - What about signal propagation from F/F to F/F along a long logic path?
  - What else?

- **Objectives**
  - That the inter and intra-routing is successful
  - That you use a minimal number of FPGAs, or use all of the FPGAs and minimize the variance in power dissipation among the Li’s, or use more than the minimal number so each FPGA can be routed, etc.
  - Minimize the number of “hops” between FPGAs in going from some F/F output, through gates, to some F/F input
  - What else?
Approach

- You cannot just program KL or FM and be done
- You **must** use both pre-clustering and partitioning
- You should consider objectives that help the other parts of this project
- We want to know how, if at all, gate delay, power and area impacts your approach
The input format to the partitioner is similar to that used for the ISCAS’89 benchmarks, which has the following form:

- **INPUT(X)**
  - where X is primary input signal (net) name

- **OUTPUT(Y)**
  - where Y is primary output signal (net)

- **O= FUNC(I1, I2, … , IN, D)**
  - where O is the output signal;
  - Ii is an input signal;
  - FUNC is the gate operation or function;
  - where D is the delay associated with the gate
Input Netlist: Example with no delays

- INPUT(G1)
- INPUT(G2)
- INPUT(G3)
- INPUT(G4)
- INPUT(G5)

- OUTPUT(G16)
- OUTPUT(G17)

- G8 = NAND(G1, G3)
- G9 = NAND(G3, G4)
- G12 = NAND(G2, G9)
- G15 = NAND(G9, G5)
- G16 = NAND(G8, G12)
- G17 = NAND(G12, G15)

ISCAS C17 netlist
Input Netlist: Sequential Circuit

- # 7 inputs
- # 7 outputs
- # 6 D-type flip-flops
- # 41 inverters
- # 118 gates (83 ANDs + 0 NANDs + 35 ORs + 0 NORs)

- INPUT(v6)
- ...
- INPUT(v0)

- OUTPUT(v13_D_12)
- ...
- OUTPUT(v13_D_6)
Input Netlist: Sequential Circuit-Cont’d

- $v_{12} = \text{DFF}(v_{13\_D\_5})$
- $v_7 = \text{DFF}(v_{13\_D\_0})$

- $\overline{v_3} = \text{NOT}(v_3)$
- $v_{13\_D\_0} = \text{NOT}(\overline{II234})$

- $\overline{II64} = \text{AND}(\overline{v0\text{bar}}, v_5, \overline{v7\text{bar}}, \overline{v8\text{bar}})$
- $L v_{13\_D\_1} = \text{AND}(\overline{v9\text{bar}}, \overline{v10\text{bar}}, II195)$

- $B41B = \text{OR}(\overline{IIII113}, \overline{IIII114})$
- $L v_{13\_D\_11} = \text{OR}(\overline{IIII17}, \overline{IIII18})$

A more formal description will be provided to you by the TA.
Example: Initial Placement (from partitioning)

An I/O signal

A function like $T = \text{NAND}(g2, ft)$
Final Placement

Note: Signals have been assigned to physical I/O pins

Signal G2 at pin 45

Gate G8-type NAND2-at location (2,4)
Placement with Mixed Sized Gates
Approaches

- Mixed sizes of gates – consider floorplanning technique
- How is the initial placement achieved?
- Since the FPGA area is fixed, how do you trade off minimal wire length vs cut-lines, vs wire congestion vs uniform power distribution, etc.
- How do you handle delay and switch-boxes
Constraints and Objectives

- **Constraints**
  - Unique assignment of each function (gate or F/F) to a CLB
  - Maximum number of tracks per channel -- how can this be used? Think about signal-cuts
  - Maximum power/heat generated -- how can this be addressed?
  - Area and delay constraints

- **Objectives (suggested)**
  - That the inter- and intra-routing is successful
  - That we minimize total interconnect length on FPGAs
  - That we minimize signal delay. Can you estimate switch boxes use even though the chip is not routed?
  - Uniform power distribution
Input Netlist Format: Placement

- The output format of the partitioning system is the same as the input format for the placement system.

- For the placement tool in design flow “A”, we assume that the circuit has been partitioned onto the “Li” FPGAs. You are to “assign” gates/cells to CLBs and also “assign” sub-circuit I/O signals to the FPGA’s I/O pins:

- Each Li is given the following configuration information:
  - INPUT_SIGNAL(X) where X is an input signal to the FPGA
  - OUTPUT_SIGNAL(Y) where Y is an output signal of the FPGA
  - O= FUNC(I1, I2, … , IN) where O is an output signal (net), and the li are input signals. FUNC is the gate operation or function.
Input Netlist: Placement Example

Here we assume that the entire combinational logic circuit C17 is assigned to one FPGA.

- INPUT_SIGNAL(G1)
- INPUT_SIGNAL(G2)
- INPUT_SIGNAL(G3)
- INPUT_SIGNAL(G4)
- INPUT_SIGNAL(G5)
- OUTPUT_SIGNAL(G16)
- OUTPUT_SIGNAL(G17)

- G8 = NAND(G1, G3)
- G9 = NAND(G3, G4)
- G12 = NAND(G2, G9)
- G15 = NAND(G9, G5)
- G16 = NAND(G8, G12)
- G17 = NAND(G12, G15)
Approach

- You must use a 2-phase approach, i.e., global routing followed by detained routing
Constraints and Objectives

- **Constraints**
  - Routing channels, tracks, and switches
  - Placement (assignment) of I/O and functions
  - Satisfy upper bound on maximal interconnect length or signal propagation time
  - What else?

- **Objectives**
  - For each net associated with a FPGA, make all nodes in the net electrically common
  - Maybe, minimize total interconnect length
  - Re-assign I/O assignment to enhance the above objectives
  - Modify placement to enhance the above objectives
  - What else?
Input Netlist Format: Intra-FPGA Routing

- The output format from the placement module is the same as the input format to the intra-FPGA router.
- For the routing tool, for each FPGA we assume that the gates are places and the I/O pins are assigned. We want to route each FPGA according to the gate connectivity profile.
- We are given the following configuration:
  - INPUT_SIGNAL(X) n where X is an input signal and is assigned to the n'th pin of the FPGA
  - OUTPUT_SIGNAL(Y) n where Y is an output signal and is assigned to the n'th pin of the FPGA
  - O= FUNC(I1, I2,…, IN) R C where O is an output signal; Ik is an input signal; and FUNC is the gate operation or function. The integer R indicates the row of the CLB that gate is assigned to, and C indicates the column of the CLB that the gate is assigned.
Intra-FPGA Routing Format: Example

- INPUT_SIGNAL(G1) 10
- INPUT_SIGNAL(G2) 7
- INPUT_SIGNAL(G3) 6
- INPUT_SIGNAL(G4) 4
- INPUT_SIGNAL(G5) 21

- OUTPUT_SIGNAL(G16) 12
- OUTPUT_SIGNAL(G17) 22

- G8 = NAND(G1, G3) 1 4
- G9 = NAND(G3, G4) 1 6
- G12 = NAND(G2, G9) 2 6
- G15 = NAND(G9, G5) 3 5
- G16 = NAND(G8, G12) 1 1
- G17 = NAND(G12, G15) 3 2
Output Format for Intra-FPGA router

- This format will be described later. It is very specific and allows us to take the output from your router and actually plot the wires as shown in the previous slide. (see Templates)
Input Netlist Format: Inter-FPGA Routing

- In this routing tool, we give you the physical connectivity profile between I/O pins of different Li FPGAs, and the primary I/O signals, and your tools is supposed to “route” the connections between different I/O pins on different FPGAs.

- We give the following configuration:
  - CONNECT(FPGA#I, Ni; FPGA#J, Nj; FPGA#K, Nk; …)
  - where pin Ni of FPGA#I, pin Nj of FPGA#J, pin Nk of FPGA#K and … are to be interconnected so that they are electrically common.
  - These pins are all part of the same signal (net)
  - These pins are all on the Li’s or SA or SB
  - To connect them you must use the Si’s
Example of Inter-FPGA Routing

Note: The Si’s need to interconnect fixed pin locations among the Li’s. You need to assign the primary I/O to pins.

- A signal and pin associated with a primary input I/O
- A pin associated with an internal signal
Input Netlist for Design Flow “B”
Input Netlist: Partitioning

- The same as the one for design flow “A”
Because the I/O pin assignment has not yet been done, in this routing tool we just give you the “signal” connectivity between different FPGAs, not the I/O pins.

Your tool determines the I/O pin assignments associated with the signals, and “routes” the connections between different I/O pins on different FPGAs. We give the following configuration:

- \text{CONNECT}(\text{FPGA}#I, \text{FPGA}#J, \text{FPGA}#K, \ldots)

where some signal pin of \text{FPGA}#I is to be connected to some pin of the \text{FPGA}#J and some pin of \text{FPGA}#K, \ldots.
Recall-the interconnections between all FPGAs is pre-defined. This is really a problem of finding paths along a fixed highway system.
The solution results in a pin assignment.
Another Job for the Intra-router

Who connects the squares within an FPGA?
Inter-router Issues

- **Observation:** There are always enough pins to interconnect a legally partitioned circuit
- So what are the issues?
  - Minimize the number of pins you need to use?
  - Let $T_s$ be the number of FPGAs associated with a particular signal $s$. Then minimize $\sum T_s$ over all $s$?
  - Minimize some aspect of delay
  - Assign signals to pins taking into consideration placement, assuming placement is known
  - Use a two pass system, where you first assign signals to edges, and later to pins.
  - Etc.
Input Netlist: Placement

- For your placement tool, the input is a little different since I/O pins are already assigned.
- The netlist changes are as following:
  - INPUT_SIGNAL(X) N .. where N indicates the pin which is assigned to signal X
  - OUTPUT_SIGNAL(Y) N .. where N indicates the pin which is assigned to signal Y
Input Netlist: Placement Example

- \text{INPUT\_SIGNAL}(G1) \quad 12
- \text{INPUT\_SIGNAL}(G2) \quad 24
- \text{INPUT\_SIGNAL}(G3) \quad 23
- \text{INPUT\_SIGNAL}(G4) \quad 56
- \text{INPUT\_SIGNAL}(G5) \quad 13

- \text{OUTPUT\_SIGNAL}(G16) \quad 1
- \text{OUTPUT\_SIGNAL}(G17) \quad 19

- G8 = \text{NAND}(G1,G3)
- G9 = \text{NAND}(G3,G4)
- G12 = \text{NAND}(G2,G9)
- G15 = \text{NAND}(G9,G5)
- G16 = \text{NAND}(G8,G12)
- G17 = \text{NAND}(G12,G15)
Placement Example: Input

I/O pins and associated signals are fixed

Logic blocks are movable
Intra-FPGA Routing Input Netlist

- The same as before for the design flow “A”
Inter-FPGA Structure
Clockwise I/O Pin Numbering
Numbering System

Counter Clockwise pin counting

Clockwise pin counting

Note: The Li’s all have N1 pins, the Si’s have N2 pins, and SA and SB have N3 pins.
Each FPGA Task

Gates will be put in these FPGAs

Routing FPGAs

Routing and Primary I/O FPGAs

(c) M.A. Breuer
FPGA Structure of the Li’s’s

- The Li’s pin numbering is clockwise
- They are all numbered the same way
- Each quarter of their pins connects to each S1-S4 routing-FPGA.
FPGA Structure of the Li’s

- FPGA Li has n pins, with n/4 pins on each side
- Each Li has 4 symmetric sides. Li-N, Li-E, Li-S, Li-W
Si Hardwired Interconnect to Lj

- The Si FPGAs have counter-clockwise I/O pin numbering
- The *hardwired* interconnections between the Si’s and Lj’s are shown below. Assume Lj has n pins, and Si has 2n pins.
- Here, for example, pins 1 to n/4 of S1 are connected to pins 1 to n/4 of L1; pins 1 to n/4 of S2 to pins n/4+1 to n/2 of L1, etc.

Therefore, if L1 has n pins, n/4 pins on each side, then S1-S4 will have n/2 pins on each side. S1-S4 each will have 2n pins.
Si Hardwired Interconnect to SA & SB

- The *hardwired* interconnections between the Si’s and SA and SB is as below
- Pins on the East side of the Si’s are not connected to the Lj’s
SA and SB Primary I/O FPGA

One half (East + Right Half of North + Right Half of south) of the pins of SA and SB are used for I/O; the others are hardwired to the Si’s.

If L1 has n pins, then S1-S4 will have 2n pins. SA and SB will also have a total of 2n pins with n/2 pins on each side just like S1-S4.

The pins on SA and SB are numbered clockwise.
Intra-FPGA Structure
General View of One FPGA
Pin Locations

Number of pins is $4kN$
Tracks Specifications: 4 Types

- T1 tracks
- T2 tracks
- T3 tracks
- T4 tracks
Switch Specifications

1: no cut
0: a cut

0: no via
1: a via
Some Example Interconnections
CLB Connections

I/O pad

R or J or D
S or K
Gate or F/F
## CLB Lookup Table

<table>
<thead>
<tr>
<th>PROGRAMMABLE CLB</th>
<th>CONFIGURATION DATA</th>
<th>FUNCTION OF CELL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 0 0 0</td>
<td>RS flip flop</td>
</tr>
<tr>
<td></td>
<td>0 0 0 1</td>
<td>JK flip flop</td>
</tr>
<tr>
<td></td>
<td>0 0 1 0</td>
<td>D flip flop</td>
</tr>
<tr>
<td></td>
<td>0 0 1 1</td>
<td>inverter</td>
</tr>
<tr>
<td></td>
<td>0 1 0 0</td>
<td>buffer</td>
</tr>
<tr>
<td></td>
<td>0 1 0 1</td>
<td>NAND2</td>
</tr>
<tr>
<td></td>
<td>0 1 1 0</td>
<td>NAND3</td>
</tr>
<tr>
<td></td>
<td>0 1 1 1</td>
<td>NAND4</td>
</tr>
<tr>
<td></td>
<td>1 0 0 0</td>
<td>NOR2</td>
</tr>
<tr>
<td></td>
<td>1 0 0 1</td>
<td>NOR3</td>
</tr>
<tr>
<td></td>
<td>1 0 1 0</td>
<td>NOR4</td>
</tr>
<tr>
<td></td>
<td>1 0 1 1</td>
<td>EOR2</td>
</tr>
<tr>
<td></td>
<td>1 1 0 0</td>
<td>NOT-EOR2</td>
</tr>
<tr>
<td></td>
<td>1 1 0 1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 1 1 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 1 1 1</td>
<td></td>
</tr>
</tbody>
</table>
I/O Pin Connections to the Tracks
Intra-FPGA Routing Template 1
Template 1 Characteristics

- Template 1 is shown by the matrix Temp1.
- There are just 4 Temp1 matrices in any FPGA:
  - Temp1(1,1)
  - Temp1(1,2)
  - Temp1(2,1)
  - Temp1(2,2)

Example: \( \text{Temp1}(1,1) = \begin{bmatrix} 11101 & \ldots & 10101 \\ \ldots & \ldots & \ldots \\ 11010 & \ldots & 11011 \end{bmatrix} \)
Intra-FPGA Routing: Template 2

N=4
Template 2 Characteristics

- Template 2 is defined by the matrix Temp2
  - It has the same format as Temp1
  - But it can have a different number of columns
- There are 2*(N-1) Temp2 matrix in any FPGA
  - Temp2(1, 2);…; Temp2(1, N)
  - Temp2(N+1,2);…; Temp2(N+1,N)
Intra-FPGA Routing: Template 3
Template 3 Characteristics

- Template 3 is defined by the matrix Temp3.
- There are $2N$ Temp3 matrices in any FPGA.
  - $\text{Temp3}(1,1); \ldots; \text{Temp3}(1,N)$
  - $\text{Temp3}(N+1,1); \ldots; \text{Temp3}(N+1,N)$
Intra-FPGA Routing: Template 4
Template 4 Characteristics

- Template 4 is defined by the matrix Temp4.
- There are N Temp4 matrices per FPGA, one corresponding to each row of CLBs
  - Temp4(1);…; Temp4(N)
  - The number of columns is a function of the number of I/O’s
  - The number of rows is 4+k
  - The numbering is tricky and will be discussed next
k is always odd, and 0<k<10. The switches are enumerated as before, from left to right (columns), starting at the top row and working down.
In the routing FPGA blocks, namely S1-S4, SA and SB, the CLBs are not used. Also these FPGAs have twice as many pins as the logic FPGAs (k is even). Therefore Template 4 is a little different in the sense that the CLB input lines have been deleted.

This would provide less switches for the intra-router. But since there are no gates in those FPGAs, routing should not be a problem.
Intra-FPGA Routing: Template 5
Here we assume the CLB has two output signals, such as Q and Qbar, but we ignore one (bottom) of the signals. The interleaving of the CLB outputs and the I/O pins is similar to that use in Template 4.
Template 5 Characteristics

- In the routing FPGA blocks, namely S1-S4, SA and SB, the CLBs are not used. Also these FPGAs have twice as many pins as the logic FPGAs (k is even). Therefore Template 5 is a little different in the sense that the CLB output lines have been deleted.
- This would provide less switches for the intra-router. But since there are no gates in those FPGAs, routing should not be a problem.
Template 5 Characteristics

- Template 5 is defined by the matrix Temp5.
- There are N Temp5 matrices in any FPGA.
  - Temp5(1);...; Temp5(N)
Intra-FPGA Routing: Template 6
When $T_2=3$, the template is illustrated below (note the interleaving order of the output pins attached to the left CLB and the input pins attached to the right CLB):
In the routing FPGA blocks, namely S1-S4, SA and SB, the CLBs are not used. Also these FPGAs have twice as many pins as the logic FPGAs \((k \text{ is even})\). Therefore Template 6 does not exist in the routing FPGAs. What used to be Template 6 with interleaving CLB inputs and outputs are not just straight vertical channels with no switches.

The contents under “Template 6” for the routing FPGAs should be left blank in the output file of the intra-routing module.
Template 6 Characteristics

- Template 6 is defined by the matrix Temp6.
- There are (N-1)*N Temp3 matrices in any FPGA.
  - Temp6(1,2);…; Temp6(1,N)
  - Temp6(N,2);…; Temp6(N,N)
Template 7 Characteristics

- When T2=3 T4=4, the template is illustrated below
Intra-FPGA Routing: Template 8
Template 8 Characteristics

- Template 8 is very similar to Template 7, except that the vertical track is now T1 instead of T2.
- When T1=3 T4=4, the template is illustrated below:
Output Results
Partitioning

- Output
  - The number of gates/F-F’s in each FPGA
  - The number of I/O pins used in each FPGA
  - The distribution of a signal’s fan-out in each FPGA
  - The signal name associated with each I/O pin in each FPGA, if applicable
  - The name of the signals associated with each FPGA
  - For each signal, all I/O pins on all FPGA’s associate with that signal, if appropriate
  - The gates (signals) that are connected to a specific signal (gate) and their distribution numbers. For example, signal X is buried in L3; or signal Y is associated with 3 Li’s
  - *Other measures of your choice that quantify the quality of your partitioning*
  - *A discussion of how your partitioning procedure was influenced by the other aspects of this project*
  - *A discussion of how you handle feedback from other modules*
If two CLBs or Pins have a signal, \( s \), in common and are on different sides of an arbitrary line, \( c \), increment the “cut count” of the line by 1.
Placement

Output

- For any line that may have a cut value different from another cut lines
  - The “cut count”
  - The average value of the cut count of all such lines
  - The maximum value of the cut count over all vertical lines
  - The minimum value of the cut count over all vertical lines
  - The maximum value of the cut count over all horizontal lines
  - The minimum value of the cut count over all horizontal lines
  - The variance of the values of the cut count over all lines
- The distribution of the half parameter lengths
- A distribution of module placement density
- Any other measures that you think will quantify the quality of your placement
- A discussion of how your placement procedure was influenced by the other aspects of this project
- A discussion of how you handled feedback from other modules
- A discussion regarding your measures and routability (wait until the router is running)
Intra-FPGA Routing

- **Output**
  - The distribution of net length for each chip, and over all chips. Think about min., max., histograms, etc.
  - A list of nets that were not completely routed
  - The matrix format of the Templates for our graphics tool
  - *Any other measures that you think will quantify the quality of your router*
  - *A discussion of how your routing procedure was influenced by the other aspects of this project*
  - *A discussion of how you handled feedback from other modules*
Inter-FPGA Routing

- Output
  - For any specific net that was not buried in a single FPGA, how you route it among the FPGAs
  - The distribution of the number of FPGAs used for routing each non-buried net
  - The number of I/O pin used for any specific FPGA
  - Any other measures that you think will quantify the quality of your router
  - A discussion of how your routing procedure was influenced by the other aspects of this project
  - A discussion of how you handled feedback from other modules
Feedback
Feedback Introduction

- There are a number of ways for the modules to perform various types of feedback. The goal of feedback is to increase the robustness of the modules, therefore allowing the entire FPGA compiler to handle more complex circuits. It also shows that the individual modules are programmed with the whole picture in mind. Remember, this is a team project, you do need to cooperate and coordinate with your teammates.

- There is no provided standard or fixed format for the feedback information that is to be passed along modules. Be precise, be logical, be creative.
Feedback Flow

- This is one of the many feedback flows that could be achieved. It is only listed as an example.