Fall ’01 Project

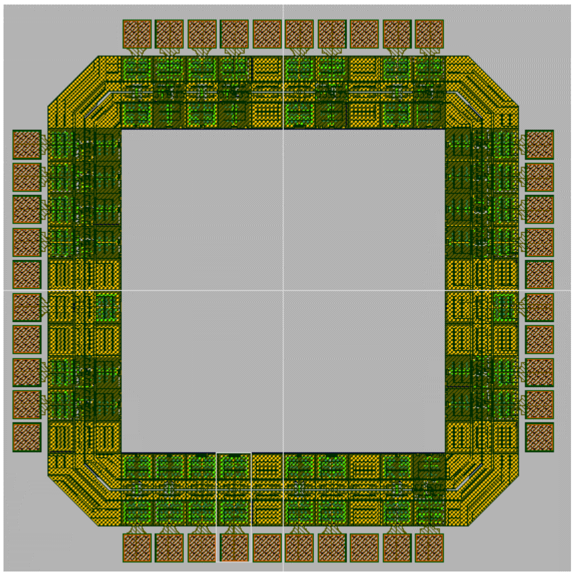
- The Fall ’00 project had each team design a small standard cell library for AMI 0.5u technology
  - Cadence tools to place/route the design
  - A MOSIS tinychip padframe (40 pin) was used for the IO
- W. Tan (a student from that class) took the best library from that effort, cleaned it up, and have since used it to fabricate a non-trivial design
- This library will be made available to you for this semester’s project
- For Fall ’01, you must do a design that combines custom layout + automated cell-based layout to create an ‘interesting’ VLSI system

Fall ’01 Project Requirements

- Must combine custom layout block(s) + automated placed/routed (blocks)
  - i.e., Standard cell block + ram array block
  - i.e., array multiplier + external standard cell control
  - Custom layout cannot be just adding new cells to library
- Final design must use at least ½ the available area in a MOSIS 0.5u Tinychip padframe
- IO requirements must be within constraints of Tinychip
- Must develop a testplan that would show how the chip would be tested if returned from MOSIS
- I would be willing to do a directed individual during the Spring semester for all who are willing to follow thru on fabrication/test of the design (submission date is February)
- Can be 2-person teams or individual effort.

Sample Project Ideas (not exhaustive)

- SSRAM – single cycle write/read SSRAM block
  - latched address, data in, control
  - unlatched data out
- Dual port register file – two read ports, one write port
  - latched address, data in, control
  - unlatched data out
- Carry Save Array (CSA) Multiplier
  - array tiled with custom cells, final vector merging adder either custom or standard cell (pg 412 of Rabaey book)
- Phased Logic Cell – do a custom cell for that implements a Phased logic function (4-input Lookup table + scan chain + other logic) – use standard cell place/route to create a design.
  - This project would require close cooperation with me and I would restrict it to only on-campus teams.



HP 0.5 tinychip Padframe (40 pin) with Tanner pads

Core area: 900µ x 900 µ

Need to use 2 pins for Vdd/Gnd.

Test Plan

- You will probably need to surround your design with some sort of ‘testbench’ that will feed inputs to the design and capture outputs within the limits of the Tinychip padframe
  - Tinychip has 40 pins – 38 IO’s, need 2 pins for Vdd/Gnd
  - testbench will have to multiplex design IO to limits of tinychip padframe
- E.G., assume your design is a 16 x 16 = 32 multiplier that accomplishes 1 multiply per clock cycle.
  - pinout could be Vdd, Gnd, Clk, Reset, Dir, D[31:0] (37 total pins)
  - When DIR = 1, then D[31:0] are inputs and latch two 16-bit operands on rising clock edge.
  - When DIR = 0, then D[31:0] are outputs and contain the 32-bit product

Tool Usage

- The standard cell library in Cadence format will be provided along with tutorials for producing placed/routed standard cell block
  - Verilog netlist used to specify the design
- For custom layout, you will need to use Cadence
  - You will probably be creating one more custom cells for your design
  - ‘Tiling’ of the cells into an array can either be done manually or automated via SKIL code
- Routing between the standard cell block(s) and custom block(s) can either be automated or done manually
- Routing between the padframe and the final design can either be automated or done manually
- You must make an attempt to do full chip transistor level simulation via IRSIM as final verification (more on this later).

Extra Points

- Extra points will be awarded if you use additional ECAD tool automation to produce your design above the minimum
  - SKIL code used to automate 'tiling' of array cells
  - Automated routing between standard cell block(s) and custom block(s)
  - Automated routing between core and padframe
- I will give you tutorials/methodology for standard cell place/route
  - To do the above 'extra' automation, you will have to read the Cadence documentation on your own and do experimentation.
- Project counts as 3rd test.

Simulations

- You must provide a gate level Verilog simulation of your COMPLETE chip (testbench + design)
  - Will provide Verilog models of standard cells
  - You will have to write Verilog models of any custom cells that you design (I can help in this, it is not hard).
  - This simulation is for functional verification only, not timing verification
- Once the layout is complete, you must provide a chip-level through-pad transistor-level simulation via IRSIM
  - IRSIM is switch-level simulator that works from an extracted transistor level netlist of the layout
  - Useful for finding fatal bugs like missing routes
  - Will talk about this later, you should practice IRSIM simulation on your custom cells first.
- Project is due at midnight on Sunday, December 2nd.

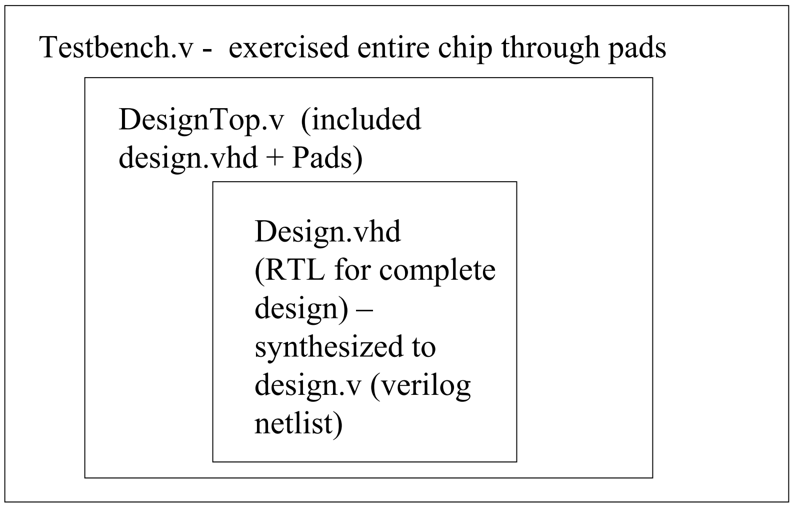
Project Proposal

- Project proposal due on Friday, October 26th at class time (hardcopy)
- 2-page description that contains a complete description of custom blocks + standard cell blocks
  - Block diagram of design, plus description of custom cells
  - Need to roughly indicate how your testbench will exercise the circuit within padframe contraints.

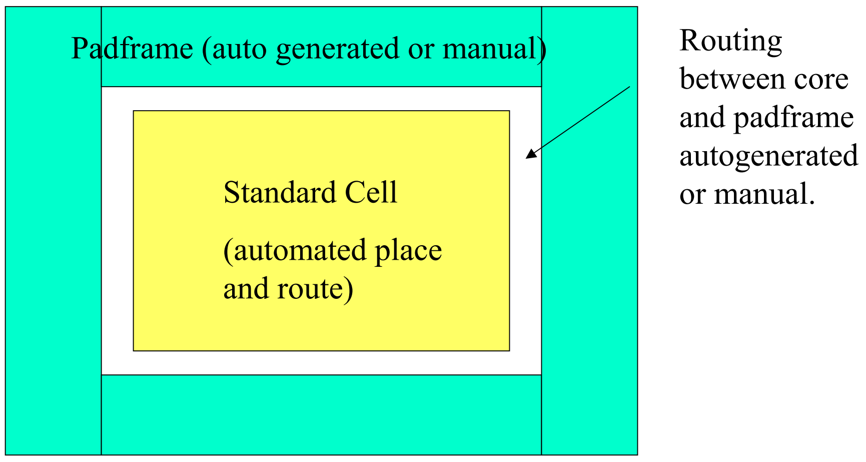
Checkoffs to complete project

- Simulation (Modelsim or Cadence Verilog simulator)
  - Verify that you can simulate both an RTL level design and a gatelist level netlist
  - The tutorial uses VHDL for the RTL level and Verilog for gate level. You can use Verilog for the RTL level if you wish
- Synthesis (Synopsys or Cadence Synthesis)
  - Verify that you synthesize RTL to a netlist of gates using the supplied library.
- Layout
  - Verify that you can create a standard cell layout, and import that layout back into Cadence icfb.
  - Verify that you can create a padframe (manual or automated)
  - Verify that you integrate your standard cell layout with the padframe and route between the two (manual or automated).

Model Hierarchy (last year)


Testbench.v - exercised entire chip through pads
DesignTop.v (included design.vhd + Pads)
Design.vhd (RTL for complete design) – synthesized to design.v (verilog netlist)

Layout Blocks (last year


Padframe (auto generated or manual)
Standard Cell
(automated place and route)
Routing between core and padframe autogenerated or manual.

Sample Model Hierarchy (this year)

testbench.v – model that exercises entire chip through pads

Designalltop.v – model that includes pads with Designall.v

Designall.v – model that ties control.v + design.v together

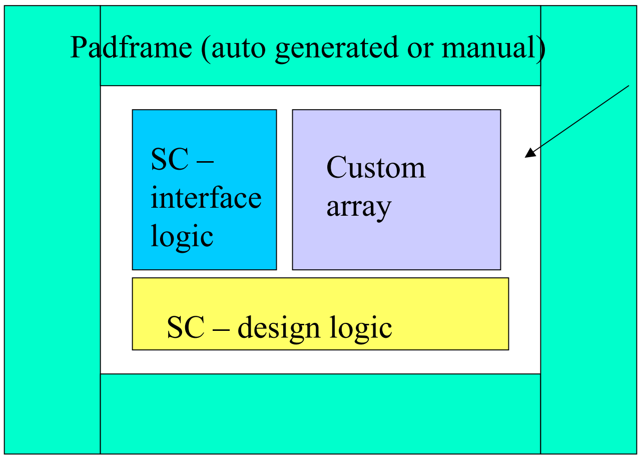
Control.vhd (.v) Logic that interfaces limited external pins to design – synthesized to standard cells.

Design.v – model that ties scell + custom together

Design_scell.vhd or .v (standard cell portion of design, synthesized to design_scell.v

Design_custom.v – model for custom part of design

Layout Blocks (this year)



Routing between blocks, and between padframe and blocks autogenerated or manual.