

# CSCE 491

## Computer Engineering Design

---

2005/8/23

### Week 1

## Systems Design Using VLSI Custom Logic

© 2004 Dr. James P. Davis

---

## CSCE 491 Week 1 - Outline

---

- Introduction
  - ✓ Drivers for VLSI Systems-On-a-Chip (SOC)
  - ✓ Example: Wireless Communications
  - ✓ The Widening Productivity Gap: Capacity vs. Capability
- Hardware Systems Design with VLSI
  - ✓ VLSI-based Design Space (Y-chart)
  - ✓ Designing with CPUs versus Custom Logic
  - ✓ Abstraction Levels and Representation Domains
- Example – 802.11 WLAN Project
  - ✓ Hardware Design Representation
  - ✓ The CSCE 491 Design Process

# Introduction

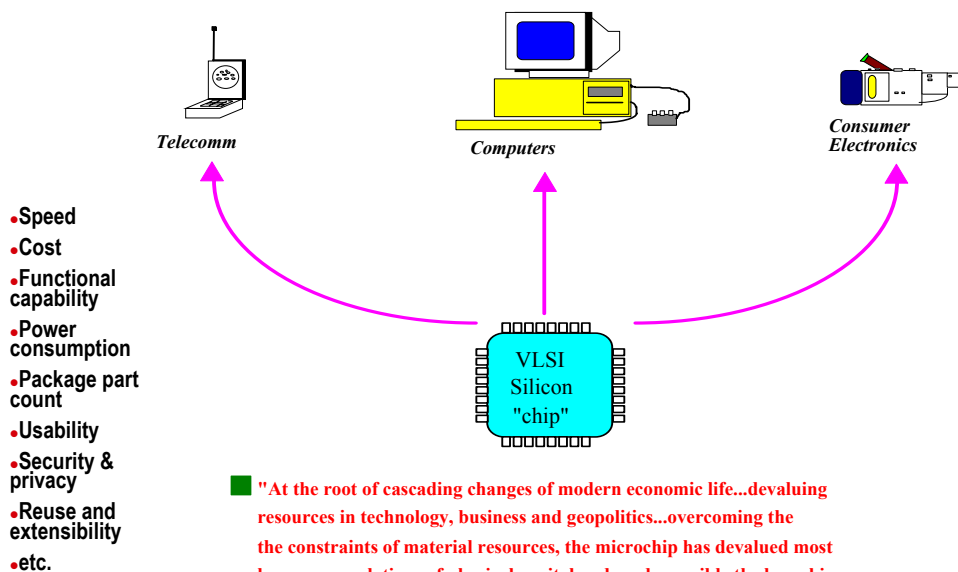
or

***What's so important about this stuff anyway and why should we care?***



© 2004 Dr. James P. Davis Page 3

## Technology Convergence



■ "At the root of cascading changes of modern economic life...devaluing resources in technology, business and geopolitics...overcoming the constraints of material resources, the microchip has devalued most large accumulations of physical capital and made possible the launching of global economic enterprises...microchips find their value not in their substance but in their intellectual content: their design..."

*George Gilder, Microcosm, 1989*



© 2004 Dr. James P. Davis Page 4

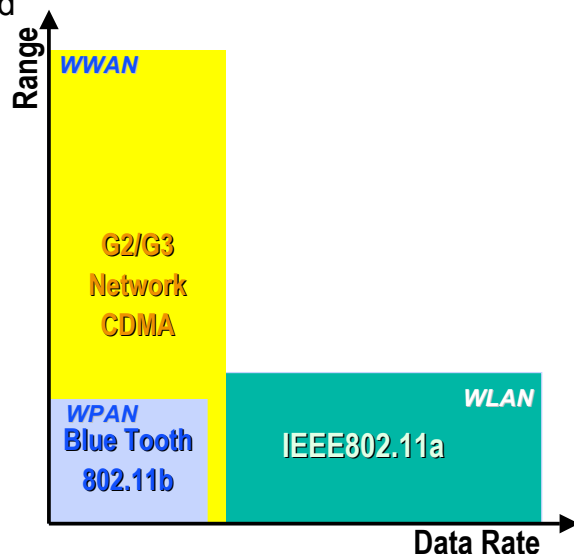
# Introduction - VLSI System “Drivers”

- Many market and technology factors coming together to create pressure on electronics product engineering organizations worldwide.
  - ✓ Increasing global competition and new markets.
  - ✓ Increasing rate of product innovations and new product introductions.
  - ✓ Decreasing time-to-market windows.
  - ✓ Decreasing “shelf life” for products in many categories.
  - ✓ Increasing pressure on competitive cost containment, profit margins.
  - ✓ Increasing convergence: integrated functionality in single electronics devices and product packages.
  - ✓ Increasing quality expectations: means for containing distribution and support costs.
  - ✓ Increasing innovation in silicon process technology and wafer scale integration densities.
- ✓ Increasing disparity: capacity of the underlying technologies versus capability of designers to manage increasing design complexity.



## Example – Wireless Communications

- The market is seeking product technology options to cover different geography ranges and data rates.
  - ✓ Bluetooth – **WPAN**.
  - ✓ IEEE 802.11 - **WLAN**.
  - ✓ 2/3G Network – **WWAN**.
- The opportunity for creating value chains encompassing product offerings, distribution and new service offerings hinges on the ability to get low cost solutions to market quickly.
  - ✓ Deliver content to wireless handheld devices.
  - ✓ Function convergence in the handset and at the base station.
  - ✓ Requires large cross-functional design teams in varied disciplines.

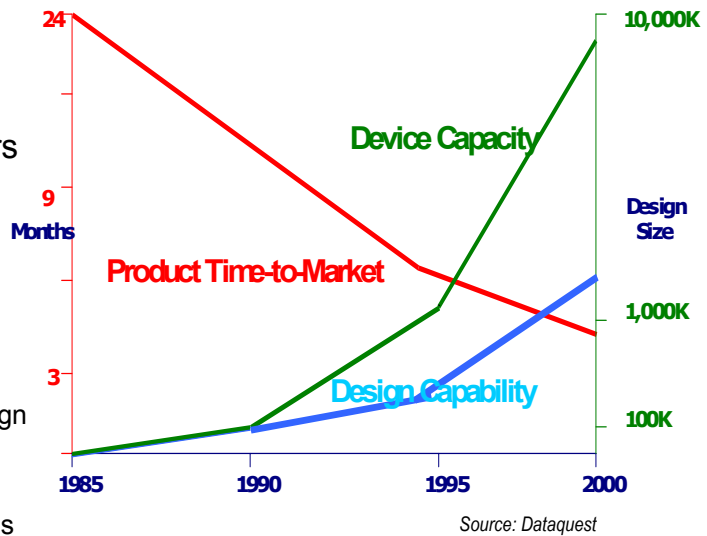


Source: Knowledge Edge KK



# The Capacity vs. Capability Gap

- Increasing capacity of the technology:
  - ✓ The rate of new technology and associated silicon process changes has continued to follow Moore's Law.
- The capability of designers and design teams to use this capacity isn't keeping up.
  - ✓ The Capacity versus Capability Gap is widening.
  - ✓ Each set of technology and process changes requires designers to manage ever more complexity in the design process.
  - ✓ New architectures, abstractions, methods and tools are required to address this increasing complexity.



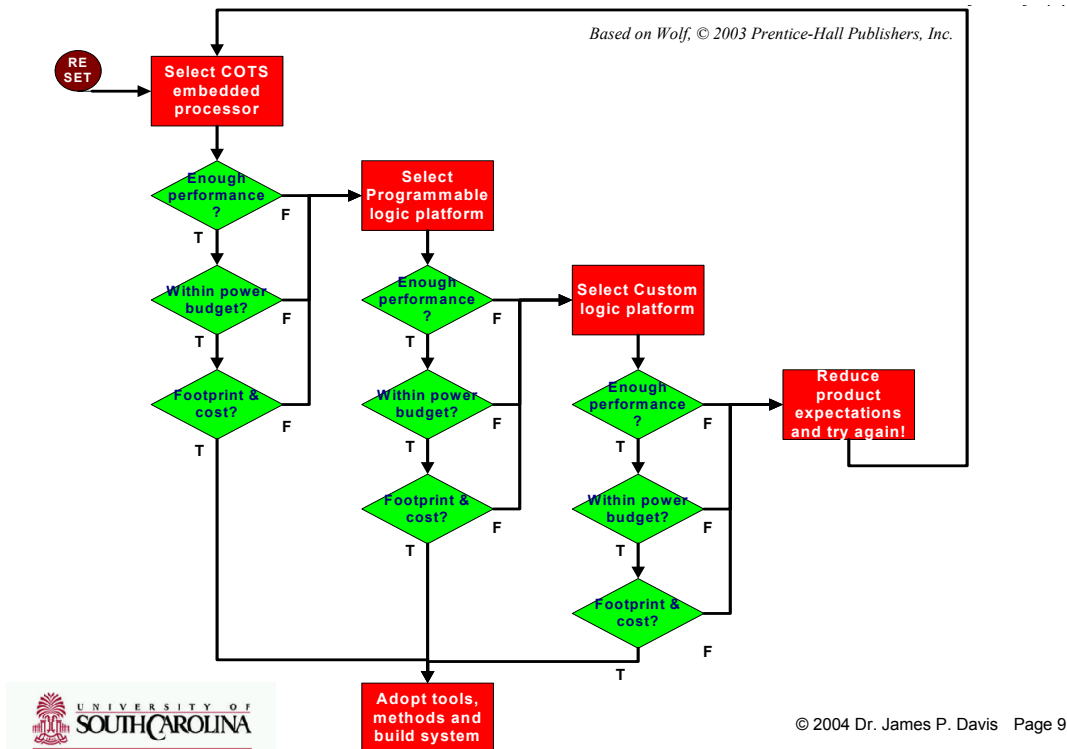
## Course of Study

or

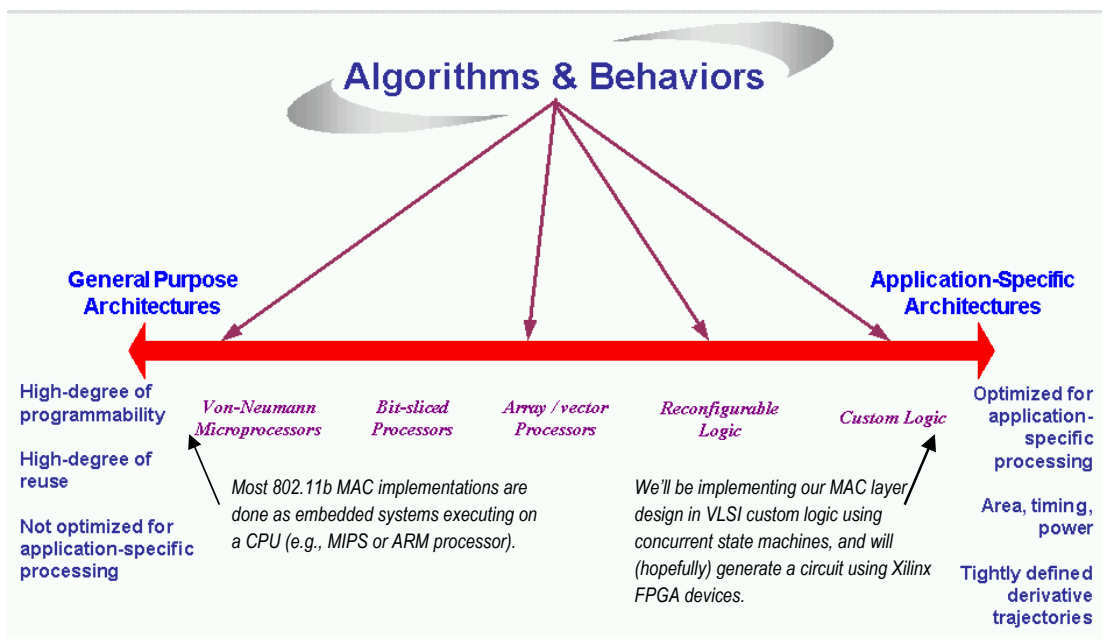
***What are we going to do about it?***



# The Systems IC Development Decision Tree

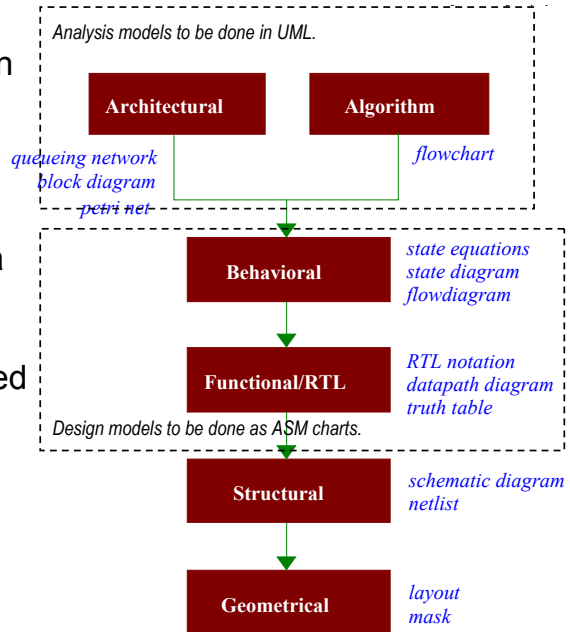


## Behavior to Architecture Mapping



# Levels of Abstraction in VLSI System Design

- A design transforms from "concept" to "implementation" in a series of ordered levels.
- From the highest level to lower levels of design "abstraction", a design is iteratively refined.
- The design description is verified and validated at each level, often cycling between levels of abstraction.
- Design descriptions are described using one or more domain representations (Behavior, Structure, Physical).



## Application Design Project

*or*

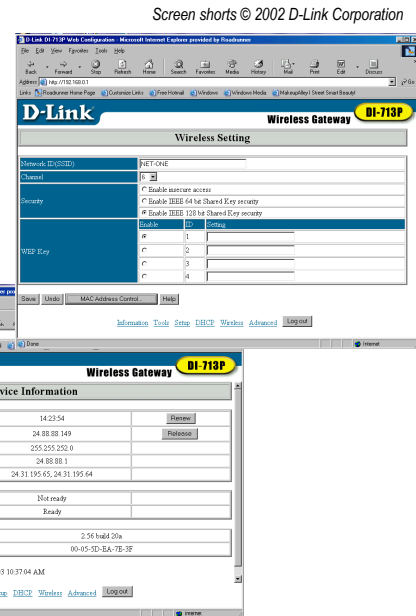
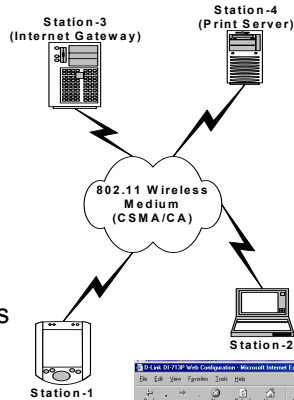
*How are we going to do it?*



# Mobile Computing - 802.11b/g WLAN MAC

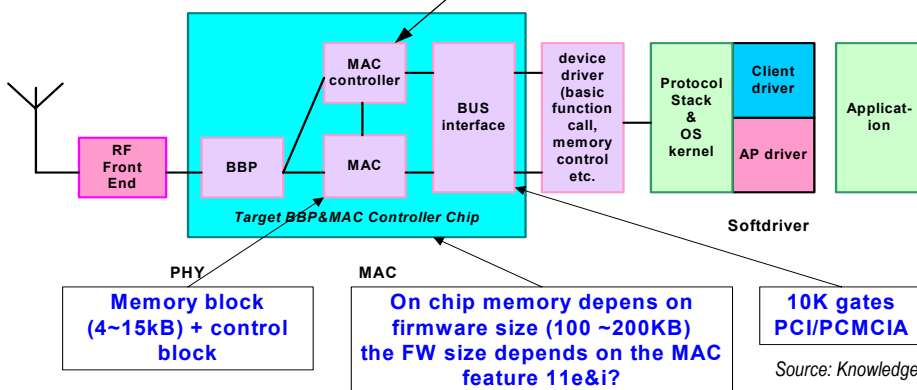
## Wireless Local Networking (PCF)

- ✓ Supports pervasive computing on PCs, laptops, PDAs and other Internet-enabled devices.
- ✓ Most Access Points support broadband access to a wired network.
- ✓ Personal network management screens access functions that control behavior of the MAC Layer.
- ✓ Example: wireless router from D-Link®.



## Typical 802.11 Product Architecture

CPU core	Die area	Peak Power Consumption (mW/MHz)	Memory System	Clock frequency & MIPS performance
ARM9E / ARM946E-S cached processor with tightly coupled memory interfaces	4.9mm <sup>2</sup> on 0.18μm estimated size with 16KB instruction & 4KB data caches and no TCMs Using Artisan cell library & RAM compiler	1.1 mW/MHz @ 1.8V (estimated)	Selectable I & D cache sizes: 0, 4K, 8K... 1M Selectable I & D TCM sizes: 0, 4K, 8K... 1M	150MHz on TSMC 0.18μm (worst case) 230MHz on TSMC 0.18μm (typical)



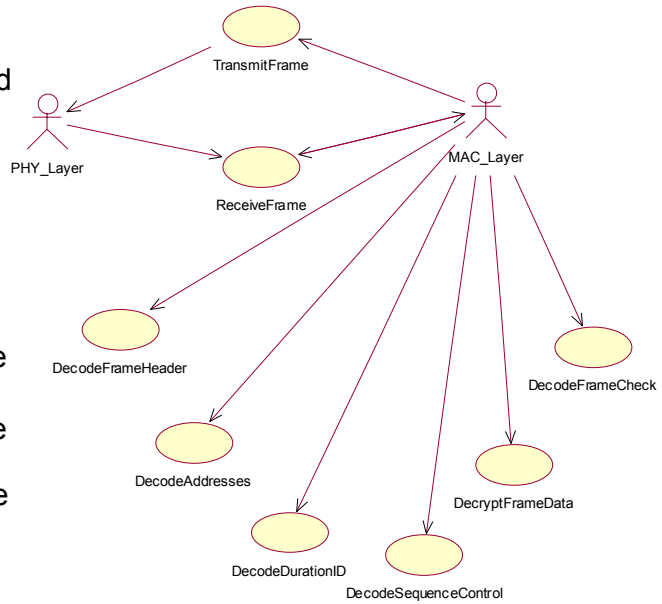
Source: Knowledge Edge KK



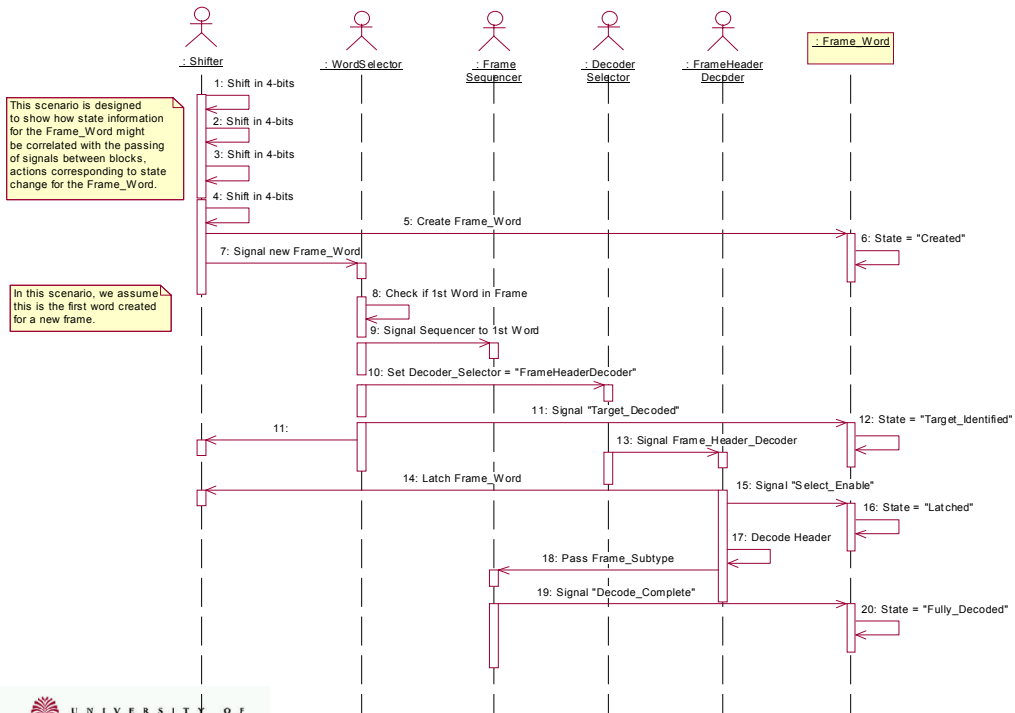
# 802.11 WLAN – Example Use Case Diagram

## Operations

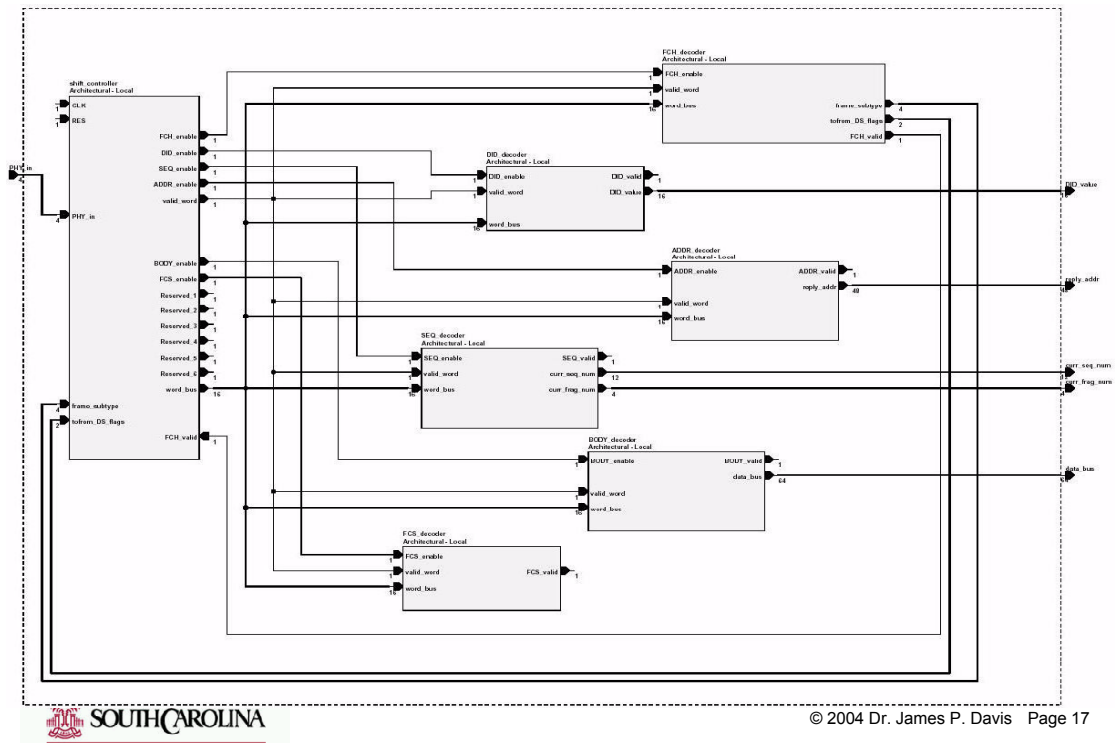
- ✓ Inventory of basic functions to be supported in our MAC layer Receiver architecture.
- ✓ Interaction at the system boundary with the PHY layer
- ✓ Each Use Case will be iterated using Sequence diagrams, then hardware block diagrams.
- ✓ Each Use Case will have a set of behaviors associated with it that we will want to model as an RTL hardware description.



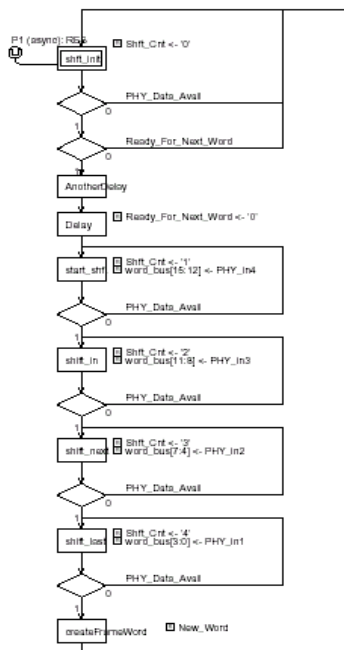
# 802.11 WLAN – Example Sequence Diagram



# 802.11 WLAN – Example Block Diagram



# 802.11 WLAN – Example ASM Thread 1

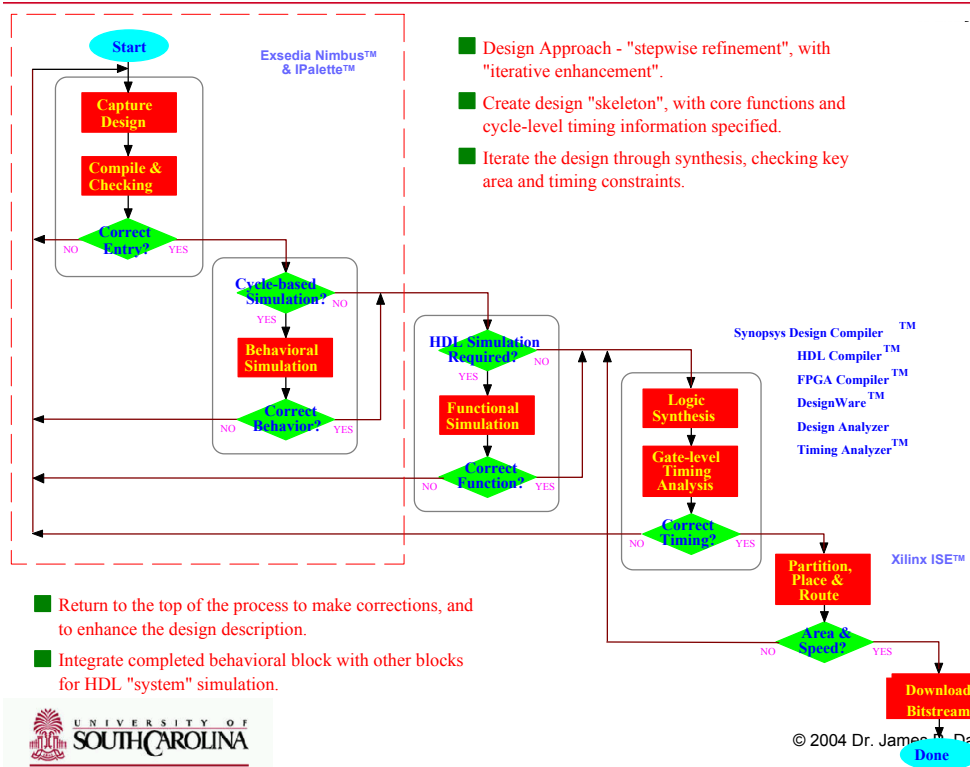


- A model for a MAC-layer functional thread.
  - ✓ We use Executable ASM diagrams to model state machine behavior and datapath operations for the hardware.
  - ✓ Executable ASM models have a graphical symbol set that looks like a flowchart.
  - ✓ Algorithm structure can be easily modeled using the ASM graphics.
  - ✓ The diagrams are annotated with register transfer notation (RTN) expressing operations and events.
  - ✓ Executable ASM models are directly executable in Nimbus™, are “correct by construction” and result in VHDL/Verilog code optimized for circuit synthesis.

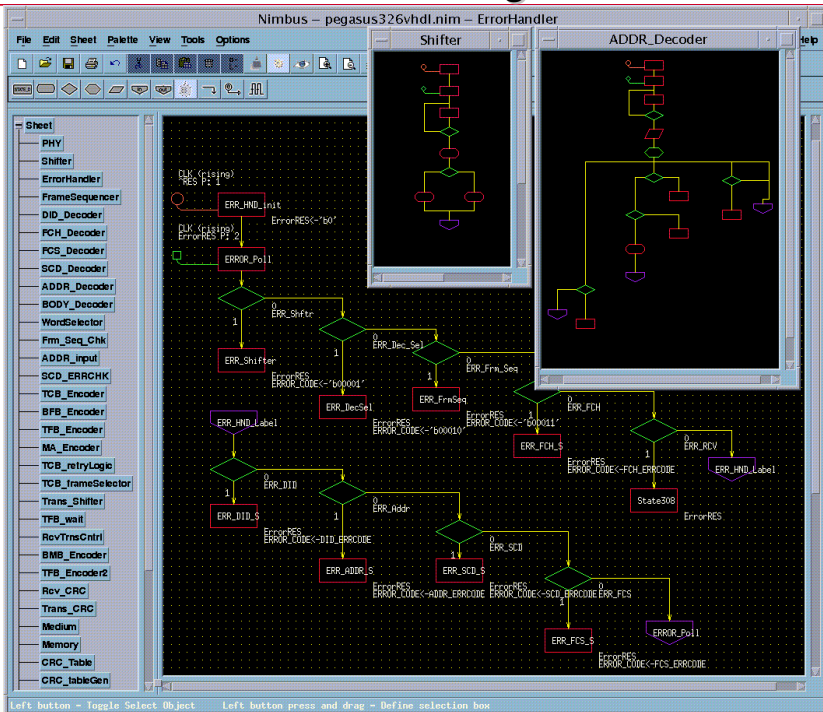




# Exploration Process - Methods & Tools

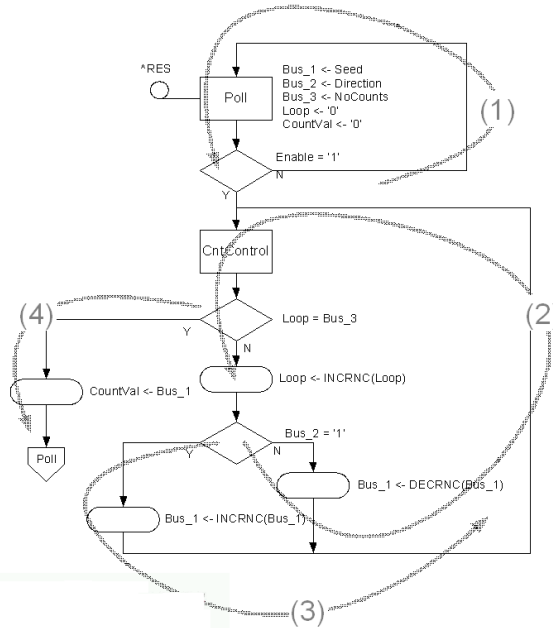


## Model Creation using Nimbus™



# Model Verification Planning – ASM Model

- In examining an ASM “thread”, we want to evaluate which logic paths we can test by defining a set of input stimuli to excite the design, so that each run allows different logic to be tested.



- Test points:
  - ✓ We start by looking at specific points allowing us to check different logic paths through the design.
  - ✓ We look at the signals and the value ranges that will cause our design to choose a different logic path.
  - ✓ We then create a set of stimulus “steps” that will allow us to trace the design in simulation.
  - ✓ Use McCabe’s *cyclomatic complexity* as additional metric.



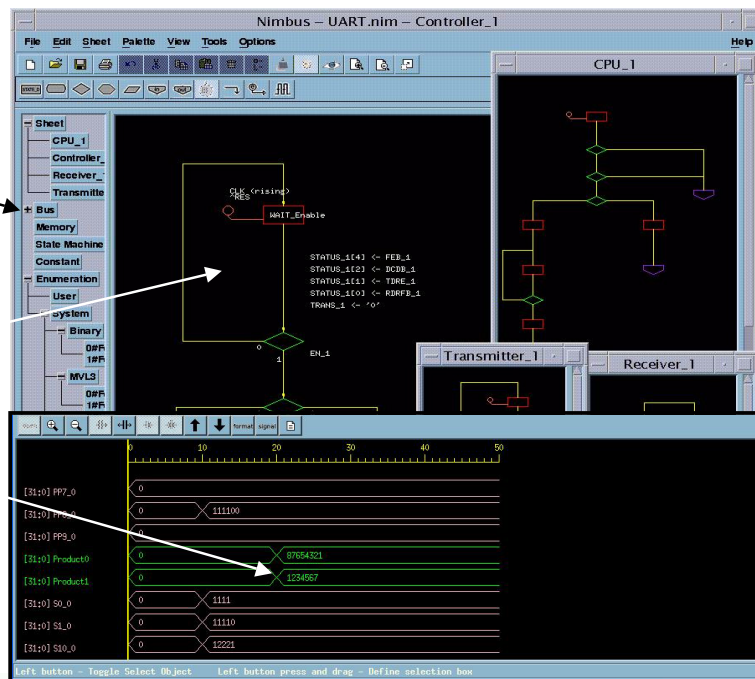
# Model Debugging & Verification in Nimbus™

Step through executing model.

Set stimulus.

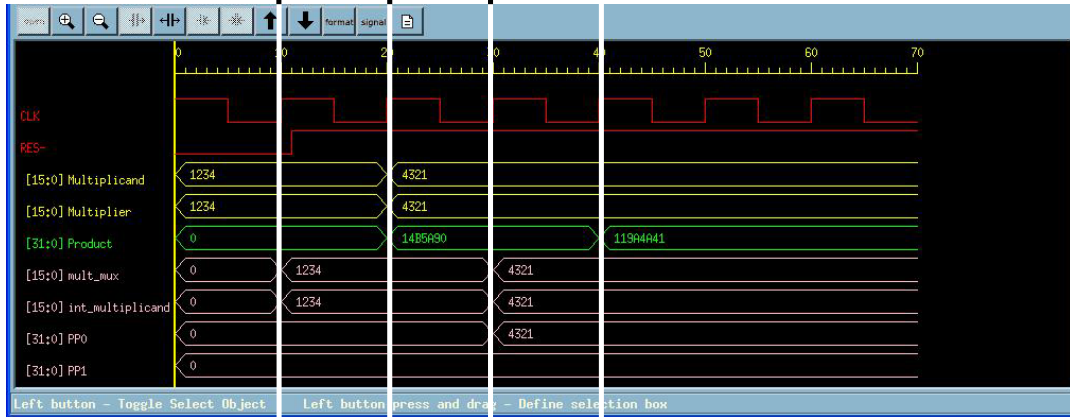
Animation of executing threads (control flow).

Display of data and cycle timing values (data).

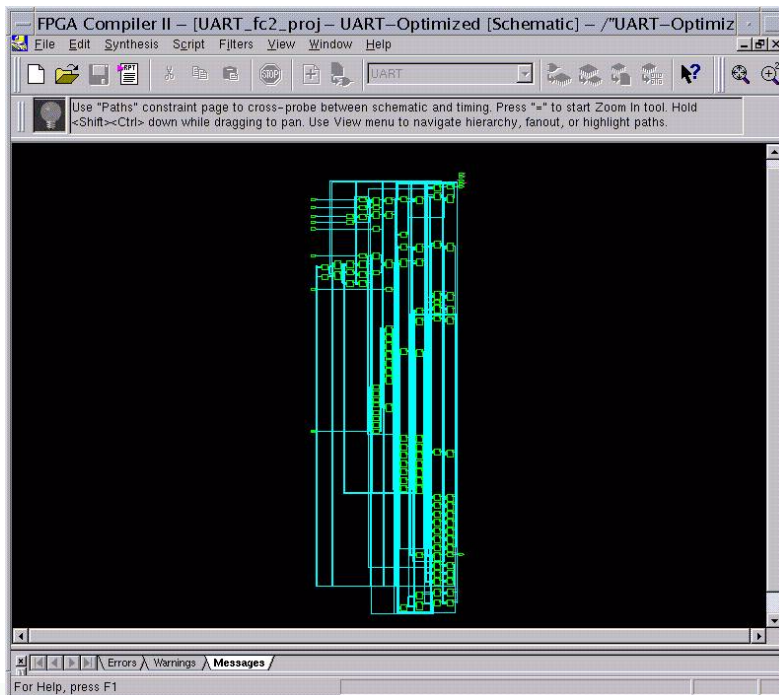


# Checking Model Latency in Nimbus™

1. On start of simulation, operands are sampled at inputs.
2. After total 2-cycle latency, product appears at output.
3. As product from 1<sup>st</sup> operation appears on output, new operands appear on the data inputs (for pipelined operation of multiplication for streaming data).
4. After a cycle latency, 2<sup>nd</sup> operand set is sampled at inputs.
5. After a 2<sup>nd</sup> cycle, 2<sup>nd</sup> product generated at output.



# Synthesis & Netlist in Synopsys® FC2



# Architecture Evaluation using Xilinx® ISE

```

mul_shiftadd32.par - Notepad
File Edit Format View Help
Release 4.2i - Par E.35 Copyright (c) 1995-2001 Xilinx, Inc. All rights reserved.
Mon May 24 18:01:25 2004
par -f _par.rsp
Constraints file: mul_shiftadd32.pcf
Loading design for application par from file par_temp.ncd. *MUL_ShiftAdd32* is an NCD, version 2.37, device xc4062xla, package bg432,
speed-08
Loading device for application par from file '4062xla.nph' in environment
C:\Xilinx\ISE
Device speed data version: PRELIMINARY 1.15 200-12-19.

Device utilization summary:

Number of External IOBs      133 out of 384      37%
Flops:                       32
Latches:                     0
Number of IOBs driving Global Buffers 1 out of 8          12%
Number of CLBs              200 out of 2304     8%
Total Latches:              0 out of 4608     0%
Total CLB Flops:            169 out of 4608     3%
4 input LUTs:               324 out of 4608     7%
3 input LUTs:               70 out of 2304     3%
Number of BUFIOs           1 out of 8          12%
Number of STARTUPs         1 out of 1          100%

[...]

Generating PAR statistics.

The Delay Summary Report. The Score for this design is: 1000
The Number of signals not completely routed for this design is: 0

The Average Connection Delay for this design is: 5.473 ns
The Maximum Pin Delay is: 45.094 ns
The Average Connection Delay on the 10 Worst Nets is: 22.677 ns
Listing Pin Delays by value: (ns)  d < 9.00 < d < 18.00 < d < 27.00 < d < 36.00 < d < 46.00 d >= 46.00
-----
1216  111  27  35  22  0

Dumping design to file mul_shiftadd32.ncd.
All signals are completely routed.
Total REAL time to PAR completion: 42 secs
Total CPU time to PAR completion: 39 secs
Placement: Completed - No errors found.Routing: Completed - No errors found.
PAR done.
    
```

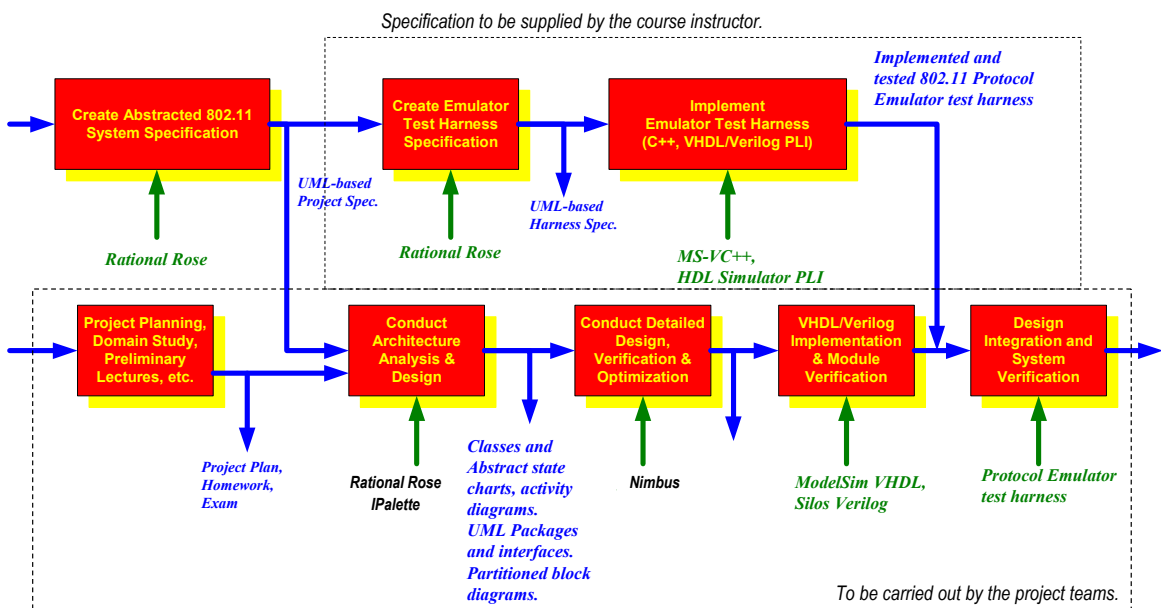
Resource usage: does the design fit?

Worst case delay: can we clock it fast enough?

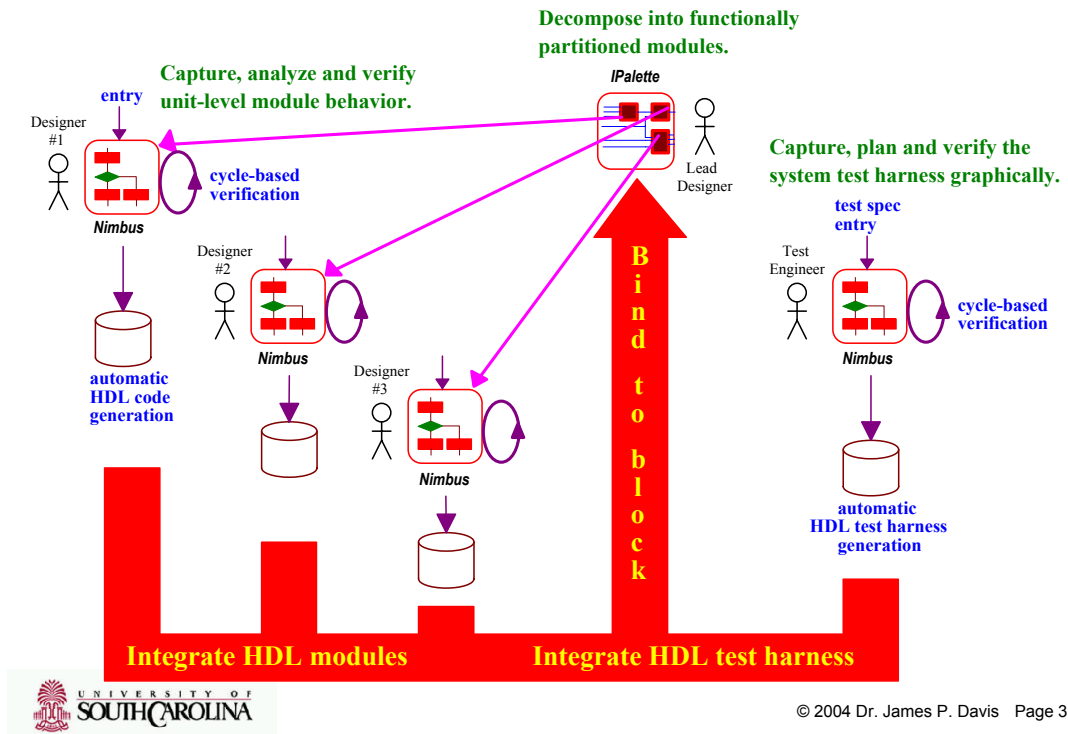
Will we have stable outputs within timing tolerances of resources?



## The CSCE 491 Design Process-1

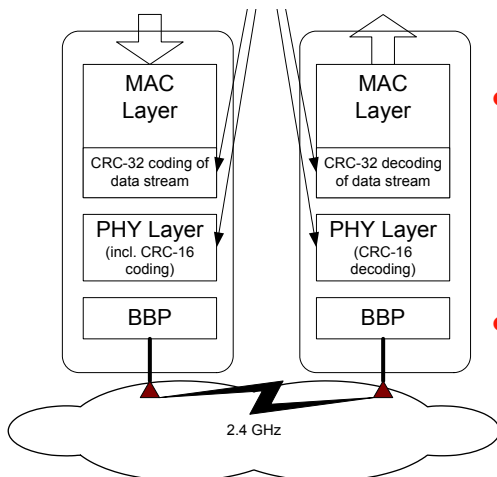


# The CSCE 491 Design Process-2



# The CSCE 491 Design Process-3

We focus on MAC layer design and performance optimization, but we hide PHY layer in the test harness.



- Project teams of 2 or 4
  - ✓ Team of 4 works on both MAC Tx and Rcv blocks.
  - ✓ Team of 2 works on a single side—either Tx or Rcv.
- Team organization
  - ✓ Team pair #1 carries out design on Tx block, and test development for Rcv block.
  - ✓ Team pair #2 carries out design of Rcv block, and test development for Tx block.
- Outcomes:
  - ✓ All team members will get “down and dirty” with 802.11 protocol.
  - ✓ You will have digital systems and ad-hoc wireless networking protocol design experience.
  - ✓ THIS MAKES YOUR EXTREMELY MARKETABLE IN THE JOB MARKET.

# Homework Assignment #1

---

- **Topic:**

- ✓ Introduction to ASM modeling using the Nimbus tool set.

- **Objective:**

- ✓ To learn to use the Nimbus tool set to carry out design entry and simulation activities.
- ✓ To ramp the learning curve so you can be proficient using the ASM method as quickly as possible.

- **Task:**

- ✓ Read the Nimbus Tutorial Guide, located at the link on my Tools web page:  
✓ <http://www.cse.sc.edu/~jimdavis/Tools/exsedia/tutorial.pdf>
- ✓ Start Nimbus on a Sun workstation (lab 1D39, 1D43), follow the steps in the tutorial, and turn in the printout of the design file you have created. Keep this file, as we will use it again in a later assignment. Do your own work!
- ✓ Starting Nimbus, follow these steps explicitly: (1) logon to a Sun workstation, (2) cd \$HOME, (3) mkdir csce491; mkdir nimbus, (4) /usr/local/nimbus/setup\_user (specify Printer as 'l39' and Acrobat Reader as 'acroread'), (5) cd \$HOME/csce491/nimbus, (6) ./run\_nimbus& (note the './' in front of the command, and the '&').
- ✓ Once you have completed the Tutorial: (1) put your name, "assignment #1" and date in fields accessed from Options -> Design Information menu, and (2) print "All Sheets" option from the File -> Print menu.

- .

