ASIC Design Flow
How to design your own chip

Frank K. Gürkaynak
Integrated Systems Laboratory
10. October 2006
Outline of the talk

1. Introduction
2. About design
3. First steps to design
4. Front-end
5. Understanding timing constraints
6. Logic synthesis
7. Back-end
Frank K. Gürkaynak

- Originally from Turkey
- Working on IC design since 1994
- Studied and gave IC design courses at:
  - İstanbul Teknik Üniversitesi (İTÜ)
  - Ecole Polytechnique Fédérale de Lausanne (EPFL)
  - Worcester Polytechnic Institute (WPI)
  - Eidgenössiche Technische Hochschule Zürich (ETHZ)
- Worked on IC design at Motorola and IBM, worked for Philips
Who am I?

Frank K. Gürkaynak

- Originally from Turkey
- Working on IC design since 1994
- Studied and gave IC design courses at:
  - İstanbul Teknik Üniversitesi (İTÜ)
  - Ecole Polytechnique Fédérale de Lausanne (EPFL)
  - Worcester Polytechnic Institute (WPI)
  - Eidgenössische Technische Hochschule Zürich (ETHZ)
- Worked on IC design at Motorola and IBM, worked for Philips
- Made enough errors designing chips to last 3 careers.
What is Hardware Design?

*Physically implementing an idea, a function, a system in hardware.*
What is Hardware Design?

Physically implementing an idea, a function, a system in hardware.

Find the optimal balance between:

- Cost / Area
- Speed / Throughput
- Energy Consumption / Power Density
- Design Time
Trade-offs in Design

No free lunch

Depending on the design, some parameters are more important. You can generally sacrifice one parameter to improve the other:
Trade-offs in Design

No free lunch

Depending on the design, some parameters are more important. You can generally sacrifice one parameter to improve the other:

- **Speed vs Area**
  
  It is possible to speed up a circuit by using larger transistors, parallel computation blocks.
No free lunch

Depending on the design, some parameters are more important. You can generally sacrifice one parameter to improve the other:

- **Speed vs Area**
  It is possible to speed up a circuit by using larger transistors, parallel computation blocks.

- **Design time vs Performance**
  Given enough time, the circuits can be optimized for higher performance
What is Important for the Following Designs?
Application Specific Integrated Circuits

When to use ASICs?

The good:
- Highest performance
- Cheap for mass production

The bad:
- Long development time
- Not very configurable
- Requires specialization
Design Specification

A list of requirements

- **Function**
  What is expected from the ASIC?

- **Performance**
  What speed, power, area?

- **I/O requirements**
  How will the ASIC fit together with the system?
We have specifications, but can we do it?

Feasibility?

For most of the projects:

We have never done it, so we don’t know exactly!

We may:

- Have experience from earlier projects
- Make small experiments to estimate performance
- Choose appropriate technology
Always Start with a Block Diagram

Iterative Process

- Identify blocks
  What do we need to perform the functionality?
Always Start with a Block Diagram

Iterative Process

- **Identify blocks**
  What do we need to perform the functionality?

- **Visualize structure**
  How are blocks connected?

```
Always Start with a Block Diagram

Iterative Process

- **Identify blocks**
  What do we need to perform the functionality?

- **Visualize structure**
  How are blocks connected?

```
Always Start with a Block Diagram

Iterative Process

- **Identify blocks**
  What do we need to perform the functionality?

- **Visualize structure**
  How are blocks connected?

- **Find critical paths**
  Which block is most critical (speed, area, power)?
Always Start with a Block Diagram

Iterative Process

- **Identify blocks**
  What do we need to perform the functionality?

- **Visualize structure**
  How are blocks connected?

- **Find critical paths**
  Which block is most critical (speed, area, power)?

- **Divide and Conquer**
  Draw sub-block diagrams
Architectural Transformations

Efficiency

Performance parameters:
- Area (mm²)
- Clock rate (MHz)
- Throughput (data/sec)
- Latency (num clock cycles)

![Graph showing architectural transformations](chart.png)

- Faster
- Smaller

Parallelization

Pipelining

Iteration

Area

Time (to complete operation)
Parallelization

More computation
If we use 2 parallel blocks:
- Area doubles
- Clock stays same
- Throughput doubles
- Latency stays same
Faster computation

if we introduce one pipeline stage:

- Area increases a little
- Clock doubles
- Throughput doubles
- Latency doubles
Iterative Decomposition

More clock cycles
If we can perform the operation in two iterations:
- Area halves
- Clock stays same
- Throughput halves
- Latency doubles
Make sure your system works

There are many languages that can be used for modelling:

- C, C++, SystemC
- Perl, Java, Tcl
- Matlab
- Verilog, VHDL

This is not a religion, there is not one definitive answer. Choose whatever is suitable, not always whatever you are comfortable with.
### Key words in modelling

- **bit-true**
  The model mimicks the hardware at bit level. Numbers are actually computed at the same accuracy as the hardware.

- **cycle-true**
  The model accurately replicates how the hardware works for every clock cycle.

- **transaction-based**
  A high level model that works on blocks of data. It calculates the end result of the computation, intermediate steps are not available.

The model is an important part of simulation environment.
Describing Hardware

Next Lecture

We will discuss this topic in a second lecture

- How to turn an idea into an architecture
- How to come up with a block diagram,
- Converting block diagram into VHDL code
Simulating your Design

Does it actually work?

- **Behavioral simulation**
  - no delays for processing

- **No performance parameters**
  - only functionality is verified

- **Hardware description**
  - Not every construct in VHDL or Verilog can be implemented in hardware. This simulation will not show this.
Testbenches

Stimuli Generator → DUT → Golden Model

Clock Generator → DUT

Comparator

stimuli

response

expected response

simulation report
Verification

Bug hunting

- **Time consuming**
The majority of design is verification.

- **Exhaustive tests are not feasible**
A 32 bit adder has $2^{64}$ possible input combinations. If we check 1,000,000,000 inputs per second it will take 200 days!!

- **Every line we write, has a potential for error**
People talk about 1 bug every 20 lines of code.

- **Golden models can be wrong**
Sometimes, your hardware description is correct, but your model is wrong.
Now that we have a working code, convert into hardware.
Standard Cell Libraries

Collection of pre-designed gates

- simple logic functions and, or, xor, not, nand, nor
- complex logic functions adders, and-or-invert
- sequential elements latches, flip-flops
- different drive strengths Each cell has at least 2-4 variations with different current driving capabilities
CMOS timing basics

Goal: to charge the capacitor as fast as possible

- Decrease load capacitance
- Increase driving (strength) current
MOS Transistor Equations

\[ I_{D,\text{lin}} = \mu \cdot C_{ox} \cdot \frac{W}{L} \cdot \left[ (V_{GS} - V_T) \cdot V_{DS} - \frac{V_{DS}^2}{2} \right] \]

\[ I_{D,\text{sat}} = \mu \cdot C_{ox} \cdot \frac{W}{L} \cdot \frac{(V_{GS} - V_T)^2}{2} \]

Driving Current

- The geometric parameters directly determine the amount of current that can flow through the transistor.
- If length \((L)\) is constant, transistors that are wider \((W)\), have more current. Larger transistors are faster.
Capacitive Loads

- Input capacitance of gates
- Interconnect capacitance
Input Capacitance

Mos capacitance

■ The dominant capacitance is the gate capacitance.
■ The capacitance is proportional to the gate area.
■ The wider the transistor the more capacitance it has.

The dominant capacitance is the gate capacitance. The capacitance is proportional to the gate area. The wider the transistor the more capacitance it has.

Drain Source
Gate
Length
Width
n+n+n+n+p-p-

The wider the transistor the more capacitance it has.
Timing Basics

Summary

- If the load stays constant, making a transistor wider, will make it faster.
Timing Basics

Summary

- If the load stays constant, making a transistor wider, will make it faster.
- A larger transistor will have a higher input capacitance.
Summary

- If the load stays constant, making a transistor wider, will make it faster.
- A larger transistor will have a higher input capacitance.
- It will be harder to drive this transistor.
Summary

- If the load stays constant, making a transistor wider, will make it faster.
- A larger transistor will have a higher input capacitance
- It will be harder to drive this transistor
- **Find a balance between driving strength and input capacitance**
Summary

- If the load stays constant, making a transistor wider, will make it faster.
- A larger transistor will have a higher input capacitance.
- It will be harder to drive this transistor.
- **Find a balance between driving strength and input capacitance.**
- The fanout (number of driven gates) of a transistor and the length of the interconnections determines the switching speed.
- Exact contribution of interconnect is not known at early stages, has to be estimated.
The synthesizer is "lazy", if you don’t set the proper constraints it will select constraints that will make him work less.

Always set proper constraints

**Synthesis Constraints**

- max delay combinational delay
- max area total circuit area
- setting the constraint does not guarantee the result
Sequential Timing

In a sequential circuit there are 4 different timing paths:

- Register to register
- Input to register
- Register to output
- Input to output
Sequential Timing

In a sequential circuit there are 4 different timing paths:

- Register to register
- Input to register
Sequential Timing

In a sequential circuit there are 4 different timing paths:

- Register to register
- Input to register
- Register to output
Sequential Timing

Timing paths
In a sequential circuit there are 4 different timing paths:

- Register to register
- Input to register
- Register to output
- Input to output
Sequential Timing

In a sequential circuit there are 4 different timing paths:
- Register to register
- Input to register
- Register to output
- Input to output

One of these paths will limit the performance of the system.
Logic Synthesis Results

Synthesis is just a tool

- Synthesis tools do not magically generate circuits
- They are supposed to generate exactly the circuit that you want
- You must have a good idea of what the synthesis result will be
**Synthesis is just a tool**

- Synthesis tools do not magically generate circuits
- They are supposed to generate exactly the circuit that you want
- You must have a good idea of what the synthesis result will be
- If the result is not as you expect, you should convince the synthesizer to produce the _correct_ result.
Different constraints, different circuits

Area vs Propagation Delay

- **Look-up Table**: X
- **Algorithmic Decomposition**: +

- **ROM solution space**

- **Constant AT product**
  - 0.0
  - 5.0k
  - 10.0k
  - 15.0k
  - 20.0k
  - 25.0k
  - 30.0k
  - 35.0k
  - 40.0k

- **Area [squm]**
  - 30.000 µm²
  - 40.000 µm²

- **Propagation Delay [ns]**
  - 1
  - 1.5
  - 2
  - 2.5
  - 3
  - 3.5
  - 4
  - 4.5
  - 5
Don’t trust the synthesizer too much

Area vs Propagation Delay (AES 8-bit Look-up table)

1 x 256 byte look-up
Don’t trust the synthesizer too much

Area vs Propagation Delay (AES 8-bit Look-up table)

Don’t trust the synthesizer too much
Don’t trust the synthesizer too much

Area vs Propagation Delay (AES 8-bit Look-up table)
Don’t trust the synthesizer too much

Area vs Propagation Delay (AES 8-bit Look-up table)

1 x 256 byte look-up

4 x 64 byte look-up

8 x 32 byte look-up

16 x 16 byte look-up

Area [squum]

10.0k

5.0k

0.0

1.2 1.4 1.6 1.8 2.0 2.2 2.4 2.6 2.8 3.0

Propagation Delay [ns]
Now that we have a netlist, let us convert it to a physical layout.
Floorplanning

How will the chip look

- Determine the total area/geometry of the chip
- Place the I/O cells
- Place pre-designed macro blocks
- Leave room for routing, optimizations, power connections
Floorplanning

How will the chip look

- Determine the total area/geometry of the chip
- Place the I/O cells
- Place pre-designed macro blocks
- Leave room for routing, optimizations, power connections
- **iterative process**, can not determine the *perfect* floorplan from the beginning

Integrated Systems Laboratory (kgf)
Standard Cells - 2

Schematic

Layout

Abstract
Standard Cell Rows
Power Planning

- Standard Cell Row
- I/O and Corner Pads Placed on the Padframe
- Macro Cell (RAM)
- Power Stripe
- Block Power Ring
- Block Halo
- I/O to Core Spacing
- Power Pad Connections
- Standard Cells
- VDD
- GND
Placing Standard Cells

NP hard problem

What is the best way of placing the cells within a given area so that:

- **Critical path is minimum**
  Long interconnections on the critical path add capacitance

- **The design is routable**
  Not all placements can be routed.

- **The area is minimum**
  The routing overhead increases area.
Clock Distribution

Clock is the most critical signal

- Standard digital systems rely on the clock signal being present everywhere on the chip at the same time: skew
- Clock signal has to be connected to all flip-flops: high fan out
- Specialized tools insert multi level buffers (to drive the load) and balance the timing by ensuring the same wirelength for all connection.

The following example is a 200 MHz 3D image renderer with roughly 3 million transistors. The clock distribution has:

- 10.928 flip-flops
- 9 level clock tree
- 478 buffers in the clock tree
- 34 cm total clock wiring
Clock Distribution

Clock is the most critical signal

- Standard digital systems **rely** on the clock signal being present everywhere on the chip at the same time: **skew**

- Clock signal has to be connected to **all** flip-flops: **high fan out**

- Specialized tools insert multi level buffers (to drive the load) and balance the timing by ensuring the same wirelength for all connection.

- The following example is a 200 MHz 3D image renderer with roughly 3 million transistors. The clock distribution has:
  - 10.928 flip-flops
  - 9 level clock tree
  - 478 buffers in the clock tree
  - 34 cm total clock wiring
ASIC Design Flow

Integrated Systems Laboratory (kgf)
Determine interconnection

- Multiple (3-9) metal routing layers. Signals on different layers do not intersect.
- Vias to interconnect metals on adjacent layers.
- The longer the interconnection:
  - The more the capacitance
  - The slower the connection
  - The more the power consumption
- Thin wires, vias add resistance
Timing information depends on interconnect

Once the physical design is complete, everything about the interconnection network is known. It is possible to extract parasitic capacitance for the interconnection:

- **Wire capacitance**
- **Wire to wire capacitance**
- **Wire - via resistance**

The extracted information can be used to extract timing information. It can be stored in special files (SDF, SPEF) and can be used by circuit simulators.
Final timing with parasitics

Parasitics may influence timing severely. It is possible to make local optimizations to combat additional capacitance:

- Buffers are added to long connections
- Driver strength of the standard cells is adjusted
- Incremental change of placement
- Critical paths are resynthesized
Design Rule Check

Every physical layer has limits that are determined by the production flow. These include:

- Minimum spacing
- Minimum width
- Minimum coverage
- Minimum area
Layout Versus Schematic

Is the layout equivalent to schematic

- The physical layout contains only geometric information.
- The devices (transistors) and interconnections are extracted. This is the extracted netlist.
- The extracted netlist is compared to the initial netlist that we started with.
- Shorts between layers can be detected in this stage.
- **Very important step**, this step tells us that the chip is good to go.
Chip is Finished, Now Time For Fun
Tape-out

**Final stage of the design**

- In old times, the design data was transferred using magnetic tapes, hence the name.
- The geometric data is electronically transferred to the production site.
- Usually there will be an independent DRC check if problems are found, we get asked to correct the problems.
- Then we **wait 10-14** weeks for production.
- Tape-out dates are well-known in advance, and are not negotiable. You **have to finish** by the given date, no mercy!
Your Very Own Chip
Now that we have our chip back

- Does it really work?
- The topic of a following lecture
Summary of the Design Flow

- **RTL Design**
  - Synthesis
  - Test Insertion
  - RTL Simulation
  - Gate-level Sim.

- **Synopsys**
  - Placement
  - ClockTree

- **Silicon Encounter**
  - Routing

- **Modelsim**
  - Fault Grading

- **TetramaxPearl**
  - Timing

- **Calibre**
  - DRC
  - LVS

Integrated Systems Laboratory (kgf)  ASIC Design Flow
A Good Looking Chip Will Always Work