Evolution of Implementation Technologies

- Discrete devices: relays, transistors (1940s-50s)
- Discrete logic gates (1950s-60s)
- Integrated circuits (1960s-70s)
  - e.g. TTL packages: data book for 300s of different parts
- Gate Arrays (IBM, 1970s)
  - "Custom" integrated circuit chips
  - Design using a library (like TTL)
  - Transistors are already on the chip
  - Place and route software puts the chip together automatically
- Large circuits on a chip
- Automatic design tools (no tedious custom layout)
- Only good if you want 300s of parts

Gate Array Technology (IBM - 1970s)

- Simple logic gates
- Use transistors to implement combinational and sequential logic
- Interconnect
  - Wires to connect inputs and outputs to logic blocks
- I/O blocks
  - Special blocks at periphery for external connections
- Add wires to make connections
- Done when chips is fabricated
  - "mask-programmed" block
  - Construct any circuit

Programmable Logic

- Disadvantages of the Data Book method
  - Constrained to parts in the data book
  - Parts are not necessarily small and standard
  - Need to stock many different parts
- Programmable logic
  - Use a single chip (for a small number of chips)
  - Program it for the circuit you want
  - No reason for the circuit to be small

Programmable Logic Technologies

- Fuse and anti-fuse
  - Fuse makes or breaks link between 2 wires
  - Fuse is 50-300 ohm
  - One-time programmable (no erasing before programming?)
  - EPROM and EEPROM
  - High power consumption
  - Typical connections are 32x4 kOhm
  - Fairly high density
  - RAW-based
    - memory bit controls a switch that connects/disconnects two wires
    - Typical connections are 5k-10k ohm
    - Can be programmed and re-programmed in the circuit
- Low density

Programmable Logic

- Program a connection
  - Connect two wires
  - Set bit to 0 or 1
- Regular structures for two-level logic (1960s-70s)
  - All rely on two-level logic minimization
  - PROM: connections - permanent
  - EPROM: connections - erase with UV light
  - EEPROM: connections - erase electrically
- PROMs
  - Program connected in the _________ plane
- PALs
  - Program the connect in the _________ plane
- PALs
  - Program the connect in the _________ plane

Making Large Programmable Logic Circuits

- Alternative 1: "PALD"
  - Put a lot of PLDs on a chip
  - Add wires between them whose connections can be programmed
  - Use fuse/EPROM technology
- Alternative 2: "FPGA"
  - Emulate gate array technology
  - Hence: Field Programmable Gate Array
  - You need:
    - A way to implement logic gate
    - A way to connect them together
Field-Programmable Gate Arrays

- PALs, PLAs = 10 - 100 Gate Equivalents
- Field Programmable Gate Arrays (FPGAs)
- Xilinx Logic Array
- 100 - 1000(s) of Gate Equivalents

Tradeoffs in FPGAs

- Logic block - how are functions implemented? fixed functions (multiply inputs) or programmable?
- Support simple functions, need fewer blocks, but they are bigger in size than them on chip
- Support simple functions need more blocks, but they are smaller in size than them on chip

Interconnect

- How are logic blocks arranged?
- How many wires will be needed between them?
- Are wires evenly distributed across chip?
- Programmability of buses wires down are some wires specialized to long distances?
- How many inputs/outputs must be routed to/from each logic block?
- What utilization are we willing to accept? 50%, 20%, 90%

Altera EPLD (Erasable Programmable Logic Devices)

- Historical Perspective
  - PALs: Some technology is programmed once block from EPROM activated by UV light
  - Altera building block = MACROCELL

Altera EPLD

- Altera EPLDs contain 8 to 64 independent macrocells
- Synchronous Mode
  - Flip-flops are controlled by global clock signal
- Asynchronous Mode
  - Flip-flops are controlled by local clock signal

Altera Multiple Array Matrix (MAX)

- AND-OR structure is limited to one or limited gates
- Product terms for product term implementation
- Global Routing: Programmable Interconnect Array

- Global Routing
- B Fixed Inputs
- 8 I/O Pins
- 6 Macros/4 LAB
- 32 Expendable LAB
**LAB Architecture**

- Diagram showing LAB architecture with labeled components:
  - Logic Module
  - Output
  - Input
  - Logic Module Wiring Tracks
  - Support for large number of product terms per output
  - Latched and reset at off-the-shelf output pins

**Actel Programmable Gate Arrays**

- Description of programmable gate arrays:
  - Block of programmable logic building blocks
  - Block of interconnect
  - Anti-fuse technology: Program Once
  - Use Anti-fuses to build up long wiring routes from short segments

**Actel Logic Module**

- Example implementation of 3:4 latch

**Actel Interconnect**

- Diagram of interconnection fabric:
  - Logic Module
  - Switches
  - Jogs cross anti-fuses

**Actel Routing Example**

- Diagram showing routing example:
  - Jogs cross anti-fuses
  - Minimize the number of jogs for speed-critical circuits
  - 2 - 3 hops for most interconnections
Xilinx Programmable Gate Arrays

- CLB - Configurable Logic Block
  - 5-input, 1-output function
  - or 2,4-input, 1-output functions
  - optional register on outputs
- Built-in fast carry logic
- Can be used as memory
- Three types of routing
  - direct
  - general-purpose
  - long lines of various lengths
- RAM-programmable
  - can be reconfigured

The Xilinx 4000 CLB

Two 4-input functions, registered output

5-input function, combinational output

CLB Used as RAM
Xilinx FPGA Combinational Logic Examples

- Key: General functions are limited to 5 inputs
  - Even better - 1/2 CLB
  - No limitation on function complexity
- Example
  - 2-bit comparator: A ≤ B ≤ C ≤ D and A ≤ B ≤ C ≤ D implemented with 1 CLB
  - (GT) \( F = A' B' C' + A B' D' + B C' D' \)
- Can implement some functions of > 5 inputs

Xilinx FPGA Adder Example

- Example
  - 2-bit binary adder - inputs: A0, A1, B0, B1, CIN
  - outputs: S0, S1, COUT
- Full Adder: 4 CLB circuit to find carry out
- 2 x Two-bit Adders (3 CLB each) yields 2 CLBs to find carry out

Xilinx FPGA Combinational Logic

- Examples
  - N-input majority function: 1 whenever n/2 or more inputs are 1
  - N-input parity functions: 5 inputs/1 CLB; 2 levels yield 25 inputs

Computer-Aided Design

- Can't design FPGAs by hand
  - Too much logic to manage, hard to make changes
- Hardware description languages
  - Specify functionality of logic at a high level
- Validation: high-level simulation to catch specification errors
- Verify pin-outs and connections to other system components
- Low-level to verify mapping and check performance
- Logic synthesis
  - Process of compiling HDL program into logic gates and flip-flops
- Technology mapping
  - Maps the logic elements available in the implementation technology (LUTs for Xilinx FPGAs)

CAD Tool Path (cont'd)

- Placement and routing
  - Assign logic blocks to functions
  - Make wiring connections
- Timing analysis
  - Verify paths
  - Determine delays as routed
- Look at critical paths and ways to improve
- Partitioning and constraining
  - If design does not fit or is unsuitable as placed split into multiple chips
  - If design it too slow prioritize critical paths, fix placement of cells, etc.
- Faults to help with these tasks exist today
- Generate programming files - bits to be loaded into chip for configuration

Xilinx CAD Tools

- Verilog (or VHDL) use to specify logic at a high-level
  - Combine with schematics, library components
- Synopsys
  - Completes Verilog to logic
  - Maps logic to the FPGA cells
- Xilinx APR - automatic place and route (simulated annealing)
  - Provides controllability through constraints
  - Handles global signals
- Xilinx Xact - measure delay properties of mapping and aid in iteration
- Xilinx XACT - design editor to view final mapping results