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 100s of different parts
  - Map your circuit to the Data Book 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 1000s of parts

Programmable Logic

- Disadvantages of the Data Book method
  - Constrained to parts in the Data Book
  - Parts are 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 two wires
  - Typical connections are 50-300 ohm
  - One-time programmable (testing before programming?)

- EPROM and EEPROM
  - High power consumption
  - Typical connections are 2K-4K ohm
  - Fairly high density

- RAM-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

PAL Logic Building Block

- Programmable AND gates
- Fixed OR/ANOR gate
- Flipflop/Registered Output
- Feedback to Array
- Tri-state Output
**XOR PALS**

- Useful for comparator logic, arithmetic sums, etc.
- Use of XOR gates can dramatically reduce the number of AND plane inputs needed to realize certain functions.

**XOR PAL**

- And/Or/XOR Logic
- Feedback
- Registered Outputs
- Tri-State Outputs

**Another Variation: Synchronous vs. Asynchronous Outputs**

**Making Large Programmable Logic Circuits**

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

**Field-Programmable Gate Arrays**

- PALs, PLAs = 10s - 100s Gate Equivalents
- Field Programmable Gate Arrays = FPGAs
- Altera MAX Family
- Actel Programmable Gate Array
- Xilinx Logical Cell Array
- 1000s - 100000(s) of Gate Equivalents!

**Field-Programmable Gate Arrays**

- Logic blocks
  - To implement combinational and sequential logic
- Interconnect
  - Wires to connect inputs and outputs to logic blocks
- I/O blocks
  - Special logic blocks at periphery of device for external connections
- Key questions:
  - How to make logic blocks programmable?
  - How to connect the wires?
  - After the chip has been fabbed
Tradeoffs in FPGAs

- Logic block: how are functions implemented (fixed functions or programmable)?
  - Support complex functions need fewer blocks, but they are bigger so less of them on chip
  - Support simple functions need more blocks, but they are smaller so more of them on chip
- Interconnect
  - How are logic blocks arranged?
  - How many wires will be needed between them?
  - Are wires evenly distributed across chip?
  - Programmability/size/wires drawn: 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 technologists programmed once bipolar ROM
  - EPLD: CMOS erasable programmable ROM (EPROM) erased by UV light
  - Altera building block = MACROCELL

Altera EPLD: Synchronous vs. Asynchronous Mode

- Altera EPLD's contain 10s-100s of independent macrocells
- Synchronous Mode
  - Flip flop controlled by global clock signal
  - Output enable local (input output is enabled)
- Asynchronous Mode
  - Flip flop controlled by local clock generated signal

- MACROCELL
  - MUX's
  - INCREMENT
  - EQUALIZER
  - EXPANDER

Altera Multiple Array Matrix (MAX)

- AND-OR Structures are relatively limited
- Cannot share logic within product terms among macrocell

LAB Architecture

- MACROCELL
  - EXPANDER
  - MUX
  - I/O Pad
  - LAB Architecture

P22V10 PAL

- Supports large number of product terms per output
- Latches and multi-asserted with output pin
- EFF is key to use AND/OR intercell connections
**Actel Programmable Gate Arrays**

- Rows of programmable logic building blocks
- Rows of interconnect
- Anti-fuse Technology: Program Once

Use Anti-fuse to build up long wiring runs from short segments

**Actel Logic Module**

- Example: Implementation of SR Latch

**Actel Interconnect**

- Logic Module
- Horizontal Track
- Vertical Track
- Anti-fuse

**Actel Routing Example**

- Jogs cross an anti-fuse
- Minimize the # of jogs for speed critical circuits
- 2 - 3 hops for most interconnections

**Actel’s Next Generation: Accelerator**

- C-Cell
  - Basic multiplexer logic plus more inputs and support for fast carry addition
  - Carry connections are “direct” and do not require propagation through the programmable interconnect

- R-Cell
  - Core is D flip-flop
  - Muxes for altering the clock and selecting an input
  - Direct connection from one C-cell output of logic module to an R-cell input; eliminates need to use the programmable interconnect

- Interconnection Fabric
  - Partitioned wires
  - Special long wires

---

**Logic Module**

- 4:1 Multiplexer

**Example: Implementation of SR Latch**

- MUX
- D0, D1
- D2, D3
- SOA, SOB

**Logic Module**

- 2:1 MUX
- S0, Y
- S1

**Logic Module**

- 2:1 MUX
- '0', '1'

**Logic Module**

- 2:1 MUX
- Q

**Logic Module**

- 2:1 MUX
- Input

---

**Interconnection Fabric**

- Partitioned wires
- Special long wires
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 routing types:
    - 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
Fast Carry Logic

Switch Matrix

Xilinx 4000 Interconnect

Xilinx 4000 Interconnect Details

Global Signals - Clock, Reset, Control

Xilinx 4000 IOB
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:
    - $A = C\overline{D}$ and $A + C\overline{D}$ implemented with 1 CLB
    - $G = A'BC + A'B'CD + A'BC' + A'C'D + A'BC' + A'B'C'D'
  - Can implement some functions of > 5 inputs

Xilinx FPGA Adder Example

- Example
  - 2-bit binary adder - inputs: $A_1, A_0, B_1, B_0, C_{IN}$
  - outputs: $S_0, S_1, C_{Out}$

Xilinx FPGA Combinational Logic

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

Xilinx Vertex-II Family

- 88-1000+ pins
- 64-10000+ CLBs
- Combinational and sequential logic using lookup tables and flip-flops
- Random-access memory
- Shift registers for use as better storage
- Multipliers regularly placed throughout the CLB array to accelerate digital signal processing applications
- E.g., the XC2V8000: 11,648 CLBs, 1108 IOBs, 90,000+ FFs, 3Mbits RAM (168 x 18Kbit blocks), 168 multipliers
  - Equivalent to eight million two-input gates

Xilinx Vertex-II Family IOB

- Tri-state/bidirectional driver
- Registers for each of three signals involved: input, output, tri-state enable
- Two registers to latch values with separate clocks
- For large pinouts, separate clocks stagger changes to avoid large current spikes
- FFs used for synchronization as well as latching

Xilinx Vertex-II Family CLB

- Four basic slices in two groups
- Each has a fast carry-chain
- Local interconnect to wire logic of each slice and connect to the CLB array; switch matrix is large collection of programmable switches
Xilinx Vertex-II Family CLB Internals

- Just ½ of one slice!
- 4-input LUT + FF
- Fast carry logic
- Many programmable interconnections for sync vs. async operation

Xilinx Vertex-II Family CLB

- Sequential Portion
  - Two positive edge-triggered flip-flops
  - Tri-state output vs. high or low
  - Asynchronous or synchronous set and reset
  - Initialize to different values of power-up
  - Clocks and load one bit complemented or not

Xilinx Vertex-II Family Slice Personality

- 4-input function generator
- OR 16 bits of dual-ported random-access memory (with separate address inputs for read G1 to G4 and write W1 to W6)
- OR a 16-bit variable-tap shift register
- With muxes, CLB can implement any function of 8 inputs and some functions of 9 inputs
- Registered and unregistered versions of function block outputs

Xilinx Vertex-II Family Interconnections

- Methods of interconnecting CLBs and IOBs:
  1. Direct fast connections within CLB
  2. Direct connections between adjacent CLBs
  3. Double lines to fanout signals to CLBs one or two away
  4. Hex lines to connect to CLBs three or six away
  5. Long lines that span the entire chip

- Fast access to neighbors vertically and horizontally with direct connections
- Double and hex lines provide a slightly larger range
- Long lines saved for time-critical signals with signal skew

Programmable Logic Summary

- Discrete Gates
- Packaged Logic
- PALs and PLAs
- Ever more general architectures of programmable combinational + sequential logic and interconnect

- Altera
- Actel
- Xilinx-4000 series to Vertex
  - CLBs implement logic function generators, RAMs, Shift registers, fast carry logic
  - Local, Inter-CLB, and long line interconnections

- Altera
- Actel
- Xilinx-4000 series to Vertex
  - CLBs implement logic function generators, RAMs, Shift registers, fast carry logic
  - Local, Inter-CLB, and long line interconnections