INTEGRATED CIRCUITS

Small and medium scale integrated circuits
- up to 200 gates per circuit
- Most frequently used: 74xx circuits + gates, flip flop circuits, latch circuit, decoder circuit, counter circuit, register circuit etc.

Large scale integrated circuits
- from 200 to 200 000 gates per circuit
- small memories, programmable circuits

Very large scale integrated circuits
- over 200 000 gates per circuit
- the number of gates is often replaced by the number of transistors, because these circuits have an integrated memory etc.

Designing the systems using small and medium scale integrated circuits has the advantage of a simpler understanding of how the circuit functions but as a major disadvantage we cannot use it for elaborate projects without increasing the size of the board, the power consumption and the cost.
Notions about Digital Circuits

A good site to remember the notions about circuits:

http://www.play-hookey.com/digital/
A combinational circuit generally has $m$ inputs and $n$ outputs. In this case, the outputs are uniquely defined for every combination of the inputs.

\[ y_1 = f_1(x_1, x_2, \ldots, x_n) \]
\[ y_2 = f_2(x_1, x_2, \ldots, x_n) \]
\[ \vdots \]
\[ y_m = f_m(x_1, x_2, \ldots, x_n) \]

where $f_1, f_2, \ldots, f_m$ are boolean functions with the arguments $x_1, x_2, \ldots, x_n$.
RECONFIGURABLE TECHNOLOGIES

Definition:
• A set of configurable resources that can be interconnected in different ways.

Architectures:
• Achievements of the Boolean function by using the sum of products (PALs, PLAs, PLDs…)
• Array of cells (FPGA)

Programming Technologies:
• Fuse – through the use of fusible
• Antifuse – through the use of antifusible
• Static RAMs (SRAM)
Programmable Logic Device Families

Acronyms
SPLD = Simple Prog. Logic Device
PAL = Prog. Array of Logic
CPLD = Complex PLD
FPGA = Field Prog. Gate Array

Common Resources
Configurable Logic Blocks (CLB)
- Memory Look-Up Table
- AND-OR planes
- Simple gates
Input / Output Blocks (IOB)
- Bidirectional, latches, inverters
Methods of implementing the logic:

• Full Custom Logic – In this case every logical function is manually designed and optimized. We can obtain compact circuits, high speed and lowered power consumption. It is a technique used rarely because it leads to high costs and low productivity.

• The Standard Cell Design – uses predefined logic blocks (like the 74xx type circuits) that are at the disposal of the user, inside a library of cells. The design can be done schematically or by using the Hardware Description Languages (HDL). The placement and routing of cells is done automatically. This way we lower the design time as well as the cost.

• The Gate Array Design. The *Full Custom* and *Standard Cell* design means making the mask sets in order to produce the circuits, which proves to be an expensive method. In the case of circuits that use gate arrays, the masks are identical which will lead to a lower price.

• Designing with programmable circuits
• Designing with programable circuits

**ROM** – is a SOP-type common circuits (sum-of-products) with a fixed area of AND gates and a programmable array (area) of OR gates. We can implement $M$ outputs for $N$ inputs. Programming the ROM means specifying the truth table of the functions.

**PLA - Programmable Logic Array** – is a sum-of-products (SOP) type logic circuit with a programmable array of AND gates and a programmable array of OR gates. We can implement functions using the available mean terms that can be used by more functions.
PLAs: Programmable Logic Arrays

- AND-OR planes: Sum of products function generation

\[
F_0 = ABC \\
F_1 = ABC + \overline{AB} \\
F_2 = ABC + \overline{BC} + \overline{AC}
\]
PLAs: Examples

AND-OR plane

\[ S = A \cdot B + C \]

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
PLAs: Examples

**AND-OR plane**

\[ S = A \cdot B \cdot C \]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
PLAs : Examples

**AND-OR plane**

\[ S_1 = A \cdot B \cdot C \quad , \quad S_2 = A \cdot B + C \]

![Logic Diagram](image)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>S1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>S2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
**PAL - Programmable Array Logic** – Is a sum-of-products (SOP) type logic circuit with a programmable array of AND gates and a fixed array of OR gates. Functions can be implemented using the available mean terms for every output function. (PAL is a registered trademark of *Advanced Micro Devices*)

**PALs : Programmable Arrays of Logic**

\[
F_0 = A \cdot B (1 + F_1) \\
F_1 = \overline{A} \cdot C (1 + F_0)
\]
SPLDs: Simple Programmable Logic Devices

Feedback terms
Every type of programmable circuit contains:

- *Interconnections* — connection resources.

- programmable switches — allows the connection of the logical elements to the (link) wires or to the (link) wires between them.

- The *Logic Block* — a base circuit that is multiplied in the programmable array. When a circuit is implemented in a programmable array, it is firstly split into smaller circuits that are mapped in the logic block.
CPLD : Complex Programmable Logic Devices

Macrocells: Uses the sum of the products + the sequential, flip-flop type elements.

The architecture based on the sum of products is similar to the SPLD. They are actually made up of multiple SPLD blocks.

It is made up of a number of programmable logic blocks (PLA, PAL and less ROM) connected by a programmable array of interconnected wires.
CPLDs: Complex Programmable Logic Devices

Several PAL-like block interconnected via a switch matrix
Typical applications for CPLD type circuits:

In the CPLD type circuits we can put together complex projects, such as a graphic controller, LAN controller, UART etc. As a general rule: these circuits shall be used for projects that require AND and OR gates but do not require a large number of flip-flip circuits. Because they are reprogrammable, it is possible for them to be reconfigured *in system* (an example would be to change the protocol for a circuit that implements a communication protocol) while it is still supplied with power.

The design is divided into SPLD blocks that make up the CPLD, this way we can obtain predictable results in regards to the speed performances. An advantage to the CPLD circuits is the predictability of the implementation.
CPLDs: Complex Programmable Logic Devices

Altera MAX 7000

Global Routing: Programmable Interconnect Array

EPM5128:
- 8 Fixed Inputs
- 52 I/O Pins
- 8 LABs
- 16 Macrocells/LAB
- 32 Expanders/LAB
CPLDs: Complex Programmable Logic Devices

Altera MAX 7000 LAB structure

LAB =
Logic Array Block
CPLDs: Complex Programmable Logic Devices

Altera MAX 7000 Macrocell structure
CPLDs: Complex Programmable Logic Devices

Xilinx CFB structure
CPLDs: Complex Programmable Logic Devices

Xilinx XC9500 architecture

Block diagram
CPLDs: Complex Programmable Logic Devices

Xilinx XC9500 architecture

Switch matrix
CPLDs: Complex Programmable Logic Devices

Xilinx XC9500 architecture

Function block
FPGA — Field-Programmable Gate Array

It is a structure that allows the implementation of high capacity logic.
FPGA-type circuits are an improved version of the gate array technology they also offer a lower cost for prototyping.

Types of circuits:

- Nonvolatile, one time programmable (anti-fuse)
- Nonvolatile, reprogrammable (flash)
- Volatile (SRAM)

But this type of circuit is more than a array of gates. This contains I/O Cells, logic cells, memories, microprocessors, clock management circuits, high speed transceiver, programmable routing resources, the latter being the main improvement compared to the standard gate arrays.

An FPGA is actually a programmable logic circuit that has a multitude of programmable connecting wires.
FPGAs: Field Programmable Gate Arrays

Arrays of reconfigurable blocks connected by Programmables interconnections

The block architecture differs from vendors, but is always LUT-based.
Lookup table principle (LUT)

- A four-input lookup table

$2^n$ possible combinational functions
**Logic-cells**

A logic cell is made up of LUT (Look-up table) – lookup table, D flip-flop
FPGAs: Field Programmable Gate Arrays

4-input "look up table"
FPGAs: Field Programmable Gate Arrays

inputs → Combinational logic → outputs

set by configuration bit-stream

4-input "look up table"

Logic Block

4-LUT

INPUTS → OUTPUT
FPGAs: Field Programmable Gate Arrays

Logic Block

4-input "look up table"
FPGAs: Field Programmable Gate Arrays

- Inputs
- Combinational logic
- Outputs
- Set by configuration bit-stream

4-input "look up table"

Logic Block

- 4-LUT
- Latch

Route to others logic blocks
FPGAs vs. CPLDs

Both are digitally programmable circuits, but each has its own characteristics.

· FPGAs are "fine-grain" circuits. This translates into them having a lot of small logic blocks and flip-flops (up to 100000). CPLDs are "coarse-grain" circuits. They contain a relatively small number (100 max) of large logic blocks and flip-flops.

· FPGAs are based on RAM memory. They have to be configured at each time they are powered-up. CPLD use EEPROM technology. They will be active when powered-up if they were programmed at least once.

· CPLD have smaller input/output times – and this is provided by the coarse-grained architecture, e.g. A single logic bloc will store bigger equations.

· FPGA have special routing resources in order to efficiently implement binary counting and arithmetic functions (adders, comparators) and RAM. CPLD does not have these facilities.

FPGA can contain large designs, while the CPLD can contain only small designs.
Main Parameters for programmable circuits:

1. *Logic Capacity* — among of logic that can be mapped in a single circuit, and it’s measured in a number of equivalent gates with a traditional implementation, or the number of NAND gates with 2 inputs.
2. *Logic Density* — among of logic per unit area.
3. *Speed-Performance* — parameters that measures the maximum speed at which a circuit, implemented in the programmable array, can operate.
   To illustrate, for combinational circuits (circuits whose output depends only on the current inputs, the output being a combination of the inputs - comparators, decoders, multiplexors, demultiplexers etc) the performance is provided by the highest delay along any path. For sequential circuits (logic circuits whose outputs at one specified time are a function of the inputs at that time, along with a finite number of previous outputs) the performance is given by the maximum clock frequency for which the implemented circuit functions correctly.
4. Internal available resources.
Example: Xilinx Spartan-3E – family of circuits

The Spartan circuit family has a low cost, and larger volume.

• Spartan FPGA circuit family:
  – Spartan-II, Spartan-IIE
  – Spartan-3 Generation
• Spartan-3 (high density)
• Spartan-3E (optimized for logic cost)
• Spartan-3A (optimized for pin cost)
• Spartan-3AN (improved with nonvolatile flash)
• Spartan-3ADSP (improved for signal processing)

  – pin compatibility for the family of circuits.
  – on-chip memories and clock management units
  – up to 1 600 000 gates per system.

The following table will comparatively present different circuits from the Spartan3 family.
<table>
<thead>
<tr>
<th>Device</th>
<th>XC3S100E</th>
<th>XC3S250E</th>
<th>XC3S500E</th>
<th>XC3S1200E</th>
<th>XC3S1600E</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Gates</td>
<td>100K</td>
<td>250K</td>
<td>500K</td>
<td>1200K</td>
<td>1600K</td>
</tr>
<tr>
<td>Logic Cells</td>
<td>2,160</td>
<td>5,508</td>
<td>10,476</td>
<td>19,512</td>
<td>33,192</td>
</tr>
<tr>
<td>Slices</td>
<td>960</td>
<td>2,448</td>
<td>4,656</td>
<td>8,672</td>
<td>14,752</td>
</tr>
<tr>
<td>Dedicated Multipliers</td>
<td>4</td>
<td>12</td>
<td>20</td>
<td>28</td>
<td>36</td>
</tr>
<tr>
<td>Block RAM Blocks</td>
<td>4</td>
<td>12</td>
<td>20</td>
<td>28</td>
<td>36</td>
</tr>
<tr>
<td>Block RAM Bits</td>
<td>72K</td>
<td>216K</td>
<td>360K</td>
<td>504K</td>
<td>648K</td>
</tr>
<tr>
<td>Distributed RAM Bits</td>
<td>15K</td>
<td>38K</td>
<td>73K</td>
<td>136K</td>
<td>231K</td>
</tr>
<tr>
<td>DCMs</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Max Single Ended I/O</td>
<td>108</td>
<td>172</td>
<td>232</td>
<td>304</td>
<td>376</td>
</tr>
</tbody>
</table>
Programming technologies:

- **OTP (One-time programmable)** – programming can be done only once and is achieved through the following techniques:
  1) fuse
  2) antifuse

- **Limited number of reconfigurations**($x \times 10^3$)
  (E)EPROM

- Configuration can be done at each power-up

**SRAM**

<table>
<thead>
<tr>
<th>Name</th>
<th>Re-programmable</th>
<th>Volatile</th>
<th>Technology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fuse</td>
<td>no</td>
<td>no</td>
<td>Bipolar</td>
</tr>
<tr>
<td>EPROM</td>
<td>yes out of circuit</td>
<td>no</td>
<td>UVCMOS</td>
</tr>
<tr>
<td>EEPROM</td>
<td>yes in circuit</td>
<td>no</td>
<td>EE CMOS</td>
</tr>
<tr>
<td>SRAM</td>
<td>yes in circuit</td>
<td>yes</td>
<td>CMOS</td>
</tr>
<tr>
<td>Antifuse</td>
<td>no in circuit</td>
<td>no</td>
<td>CMOS+</td>
</tr>
</tbody>
</table>

Antifuses are opened circuits positioned between two connecting wires. When they are programmed they become low resistance circuits, allowing the connections between wires.
Fuse (PLDs, CPLDs)

One time programming
Antifuse (PLDs, CPLDs)

One time programming

- Métal 1
- Dielectric
- Métal 2

A short circuit will be created between the two metal layers. The dielectric between the two metal layers will melt.
Programming technologies: (E)EPROM (PLDs, CPLDs, FPGAs)
Electrically Erasable Programmable Read-Only Memory

Limited number of programming

The application of a potential on the upper gate causes the transfer of charges from the channel through the thin oxide layer, which charges the floating gate.

Configuration erased by UV or electrically
Programming technologies: (E)EPROM (PLDs, CPLDs, FPGAs)
Electrically Erasable Programmable Read-Only Memory

Limited number of programming
Programming technologies: SRAM (FPGAs)

Unlimited number of programming

Programming must be done at each power-up
FPGAs: Generic reconfigurable block

- Inputs
- Look-Up Table (LUT)
- Clock
- Enable
- Out
- State
FPGAs: Xilinx interconnect

- Fast Direct Interconnect - CLB to CLB
- General Purpose Interconnect - Uses switch matrix

- Long Lines
  - Segmented across chip
  - Global clocks, lowest skew
  - 2 Tri-states per CLB for busses
FPGAs: Xilinx interconnect
FPGAs: Xilinx Switch matrix

Double

Singles

Double

Six Pass Transistors
Per Switch Matrix
Interconnect Point
FPGAs: Switch matrix architecture

- Multiplexors and pass transistors implement routing.
- Switch matrix contains configurable clusters of pass transistors.
  - provides wide variety of routing options
FPGAs: Xilinx Interconnect

- **Single**
  Flexible connection between 2 adjacent CLBs
- **Doubles**
  To fast connect two hops away CLBs
- **Long**
  Travel all the way across
- **Global clock**
  Optimized to connect F.F. clocks in CLBs

Programmable switches are used for:

- Extend or isolate wire segments
- Connecting horizontal and vertical wires
FPGAs: Xilinx Interconnect
FPGAs: Xilinx Virtex-II family architecture

Featuring 5 main components:

- **IOB**: Configurable as input, output or bidirectional
  - DCI = Digitally Controlled Impedance

- **CLB**: Elements for combinatorial/sequential logic, memory element, 3-state buffer
  - Fast carry propagation inside the CLB constituting elements

- **SRAM block**: 18Kbit dual-port RAM

- **Multiplier block**: 18 x 18 bit in Hardware

- **DCM**: Digital clock management, max. 12 / FPGA
  - Connected with 16 global clock-Multiplexers, 3 Clocks / Quadrant

Routing resources:

- **GRM**: General Routing Matrix
- Uniformized GRM connection to GRM (routing-switches)
- Programmable Active Interconnect Technology (new feature)
FPGAs: Xilinx Virtex-II family architecture
FPGAs: Xilinx Virtex-II CLB architecture

- Features 4 identical slices
- 2 three state-buffers / CLB
- Connection to a Switch-matrix
- 1 common shift line
- 2 separate carry-chains
FPGAs: Xilinx Virtex-II Slice architecture

- **Features**
  - 2 function generators (LUT)
  - 2 storage elements
  - Dedicated Arithmetic logic
  - 1 common shift line
  - 2 separate carry-chains
  - ORCY gate for SOP (Sum Of Product)
- **Function generator (4 inputs):**
  - Can be also used as a 16 bit shift-register
  - Or 16 bit memory
  - In a CLB there are 16x8 bit of memory
- **Storage elements:**
  - Usable as Latch (level-sensitive) or flip-flop (edge-sensitive)
FPGAs: Xilinx Virtex-II architecture

- SOP (Sum Of Products) using dedicated ORCY gate (horizontal cascade chain)
- Also usable to implement other large combinatorial functions
FPGAs: Xilinx Virtex-II Routing resources
Modern FPGA

- 65nm technology, 40-nm gate length (Poly)
- 1.6nm oxide thickness (16 Angstrom)
  - ~5 atomic layers
- Triple-Oxide Technology
  - 3 oxide thicknesses for optimum power and performance
- 1.0 Vcc core
  - Lower dynamic power
- 12 layer copper
- Strained silicon transistor
  - Maximum performance at lowest AC power

Over 1 Billion Transistors

Source: MPSOC’06 Keynote, Ivo Bolsen, Xilinx
FPGAs: Virtex 5 logic architecture

True 6-input Lookup Table (LUT) with dual 5-input LUT option

64-bit RAM per M-LUT about half of all LUTs

32-bit or 16-bit x 2 shift register per M-LUT

Source: MPSOC’06 Keynote, Ivo Bolsen, Xilinx
FPGAs: Virtex 5 routing architecture

Virtex-5 Routing

Symmetric pattern, connecting CLBs

Same pattern for all outputs

Source: MPSOC'06 Keynote, Ivo Bolsen, Xilinx
Domain Optimized Platforms
One Family – Multiple Platforms

Column based features

- Logic
- Memory
- DSP
- Processing
- High-speed I/O

LX
Logic Domain
Highest logic density

SX
DSP Domain
Highest DSP performance

FX
Connectivity Domain
Embedded Processors
High-speed Serial I/O

Enables “Dial-In” hard IP Mix
Logic, DSP, BRAM, I/O, MGT, DCM, PowerPC
Enabled by Flip-Chip Packaging
I/O Columns Distributed Throughout the Device

Source: MPSOC’06 Keynote, Ivo Bolsen, Xilinx
FPGAs: Implementing a system on a FPGA

Source: MPSOC’06 Keynote, Ivo Bolsen, Xilinx
FPGAs: 8 HD MPEG4 decoders on a Virtex 5 FPGA

Source: MPSOC’06 Keynote, Ivo Bolsen, Xilinx
FPGAs: The MicroBlaze RISC CPU

- 1400 LUT6
- 230 Dhrystone Mips
- > 200 fit in V5