Design Heirarchy and Analysis

September 14, 2006

Modern Digital Design

Tools:

- Design Hierarchy
- Top-down Design
- CAD (Computer Aided Design) Tools
- HDLs (HW Description Languages)
- Logic Synthesis

Logic Circuits

Two kinds of circuits: combinational or sequential.

Combinational

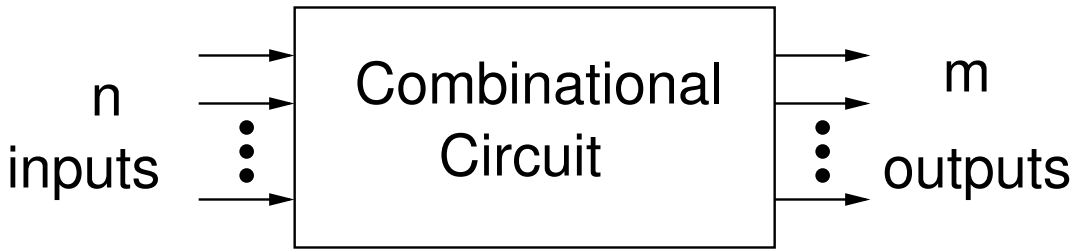
- Outputs determined by Inputs
- Specified by boolean equations.
- memory-less

Sequential

- Store bit values
- Outputs determined by inputs and stored values
- Past inputs as well as current determine behavior

Combinational Circuits

- n inputs variables, 2^n possible input combinations
- 1 output for each input combination
- Described by a Truth Table or Boolean Equation

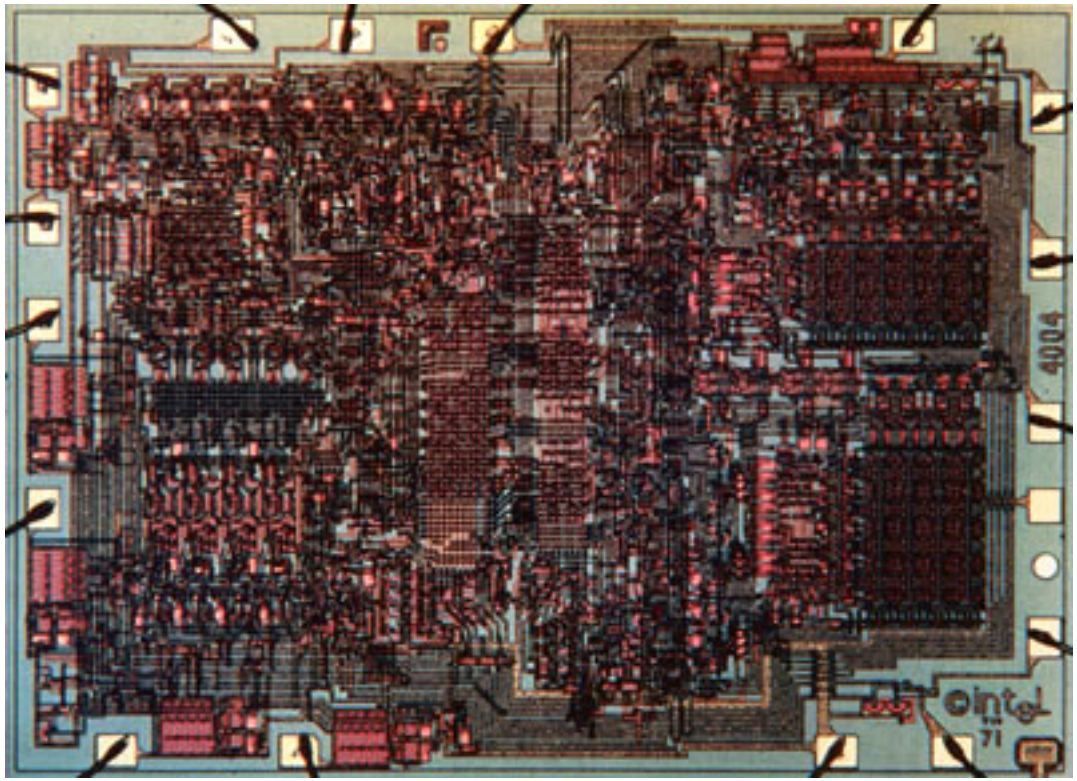


How do we design these things?

- Circuits are specified by symbols showing inputs/outputs
- All of these elements are decomposable
- Circuit $\longrightarrow$ gates $\longrightarrow$ transistors $\longrightarrow$ silicon, dopants, etc...
- VLSI Systems have 10-100s of millions of gates.

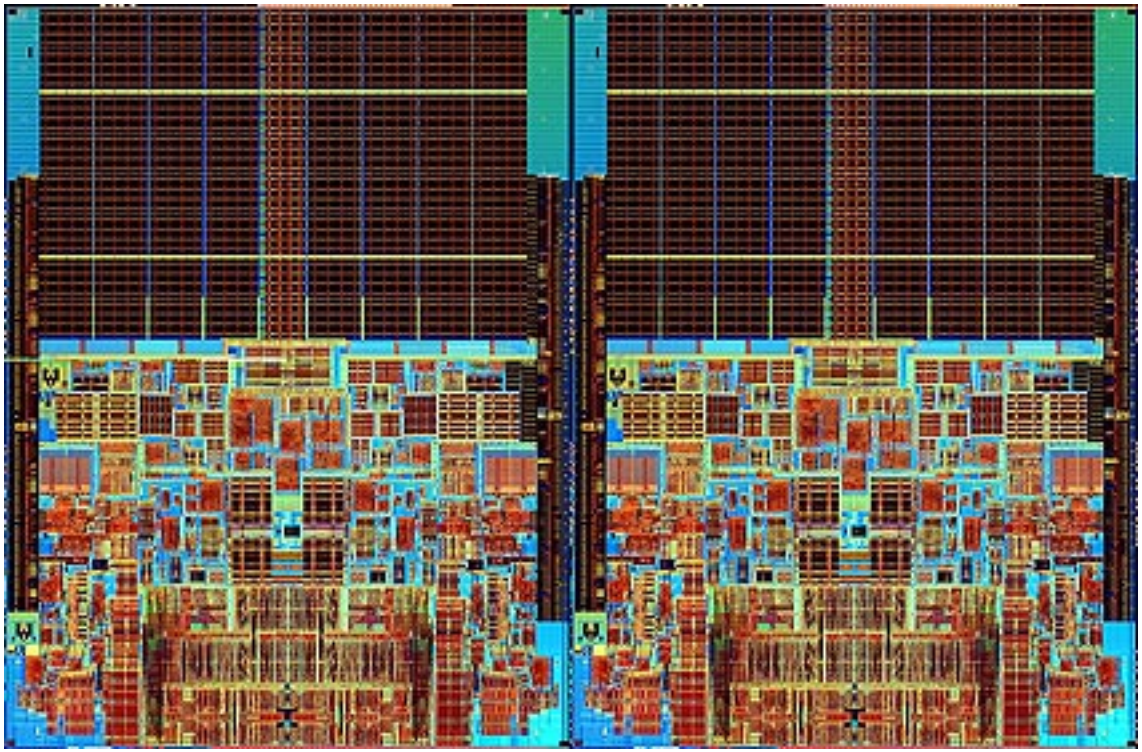
Example

1971 - Intel 4004 – 2000 transistors



Example

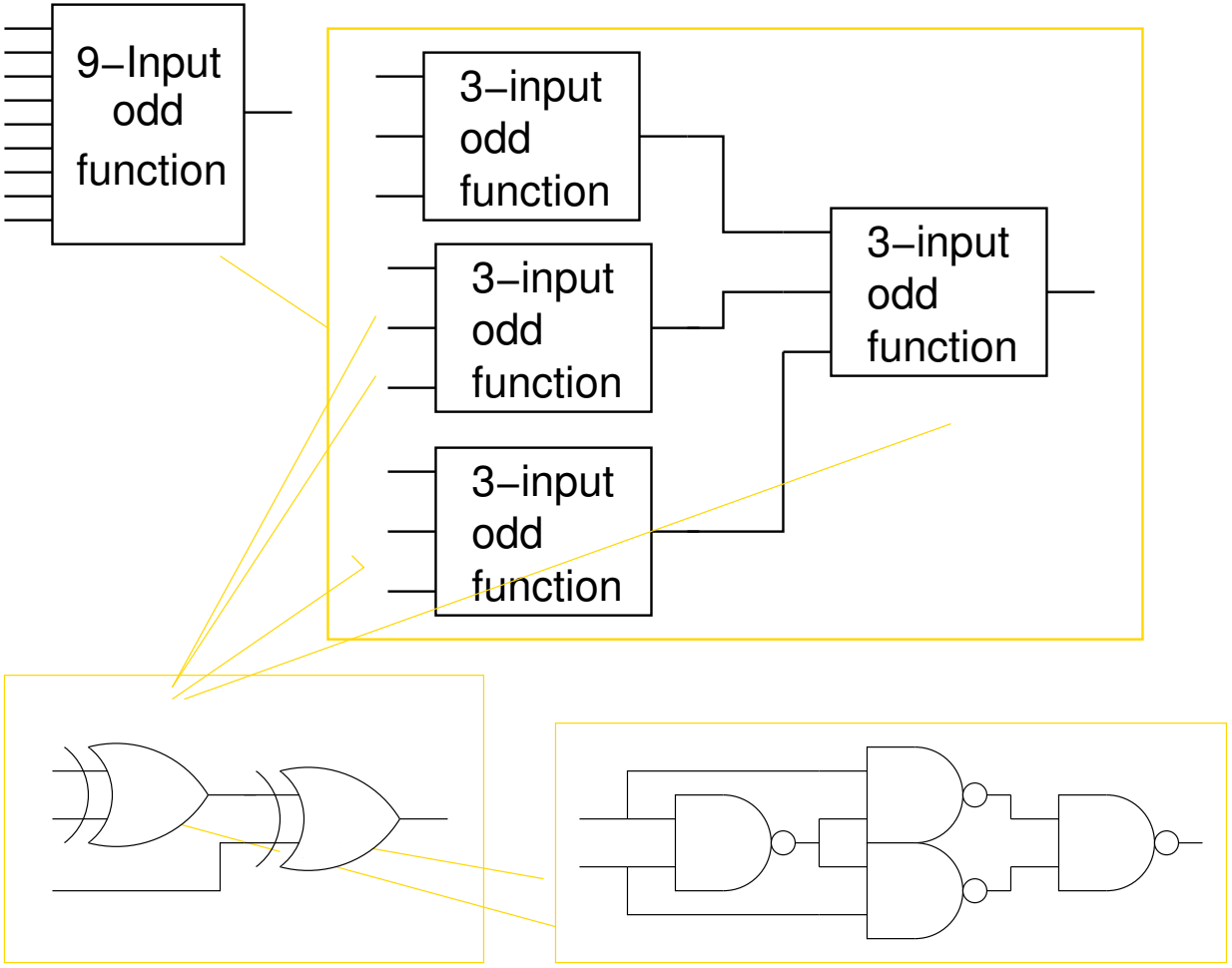
2006 - Intel Core 2 Quadro – 291 million transistors



Design Hierarchy

- Design complexity requires a *divide and conquer* approach
- Circuit $\longrightarrow$ *blocks*
- Each block is a distinct function
- Blocks are interconnected.
- Complex blocks are broken down into simpler blocks.
- Blocks are combined to form a *system*.

Design Hierarchy



Design Hierarchy

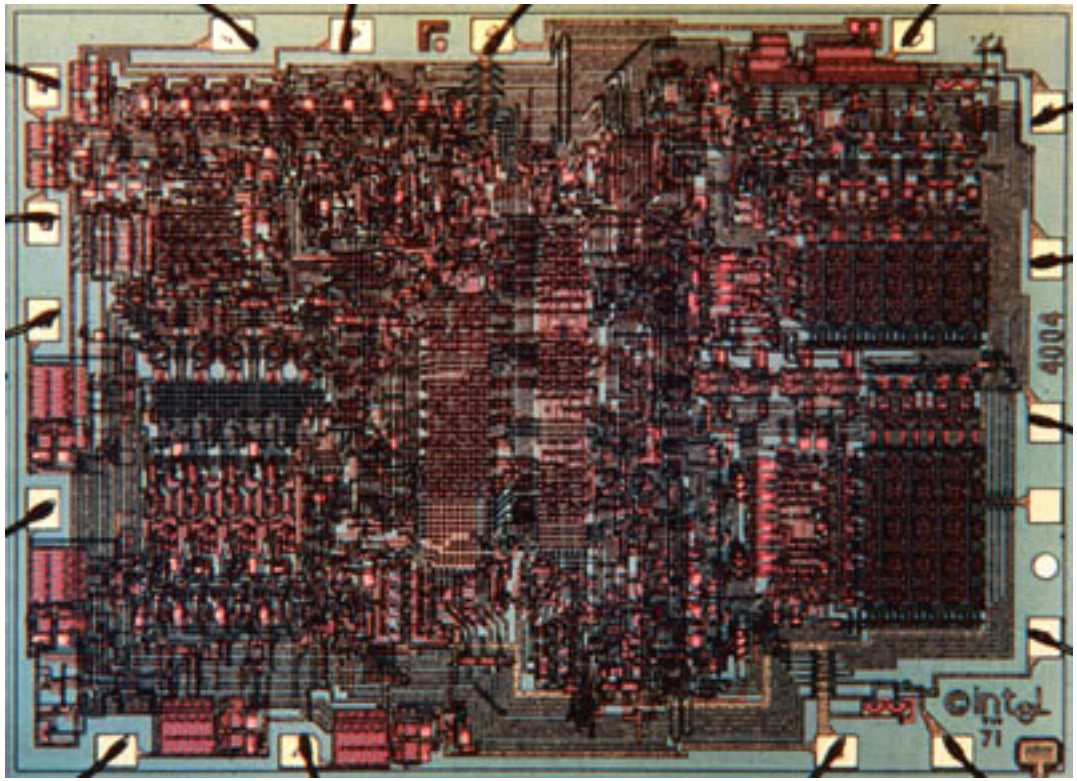
- Functions are built up of smaller blocks
- *Leaves* make up the lowest – the smallest object in the hierarchy
- Blocks can be reused to more efficient and quickly create larger structures
- Testing all of the inputs of the smaller blocks, ensures that you have an expectation of behavior when combined into larger blocks.
- However, no guarantee...

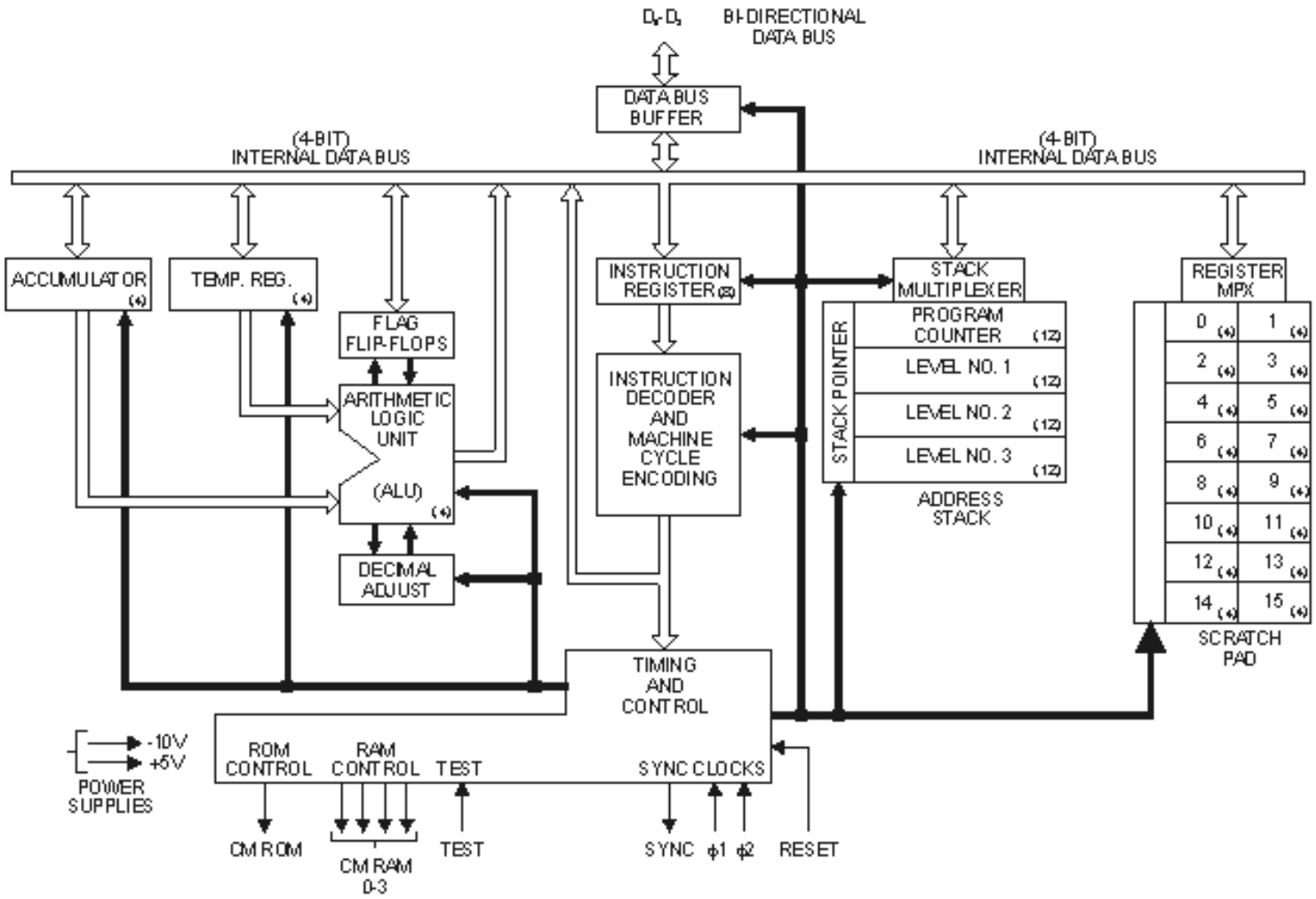
Top-Down Design

- Ideally, designs are created in a top-down fashion.
- Specifications for a design are created in a HDL
- High-level parts of the design are divided into blocks
- HDL descriptions are created for each and every block and function.

Example

1971 - Intel 4004 – 2000 transistors




D0-D3
BI-DIRECTIONAL
DATA BUS
DATA BUS
BUFFER
(4-BIT)
INTERNAL DATA BUS
(4-BIT)
INTERNAL DATA BUS
ACCUMULATOR
(4)
TEMP. REG.
(4)
FLAG
FLIP-FLOPS
ARITHMETIC
LOGIC
UNIT
(ALU)
(4)
DECIMAL
ADJUST
INSTRUCTION
REGISTER (8)
INSTRUCTION
DECODER
AND
MACHINE
CYCLE
ENCODING
STACK
MULTIPLEXER
STACK POINTER
PROGRAM
COUNTER
(12)
LEVEL NO. 1
(12)
LEVEL NO. 2
(12)
LEVEL NO. 3
(12)
ADDRESS
STACK
REGISTER
MPX
0 (4)
1 (4)
2 (4)
3 (4)
4 (4)
5 (4)
6 (4)
7 (4)
8 (4)
9 (4)
10 (4)
11 (4)
12 (4)
13 (4)
14 (4)
15 (4)
SCRATCH
PAD
TIMING
AND
CONTROL
-10V
+5V
POWER
SUPPLIES
ROM
CONTROL
RAM
CONTROL
TEST
SYNC
CLOCKS
CM ROM
CM RAM
0-3
TEST
SYNC
φ1
φ2
RESET

Computer Aided Design

- Schematic Capture (Drawing circuits and blocks from primitives)
- Logic Simulators (like ModelSim) allow for functional and timing verification
- Logic Synthesis tools (Synplicity, XST in Xilinx ISE, etc) take the primitive blocks and convert them into usable hardware.

Hardware Description Languages

- VHDL and Verilog (SystemC, HandelC, ...)
- Include structural and behavioral descriptions
- HDLs allow for simulation and synthesis
- The structural allows for top-down and hierarchical design

VHDL

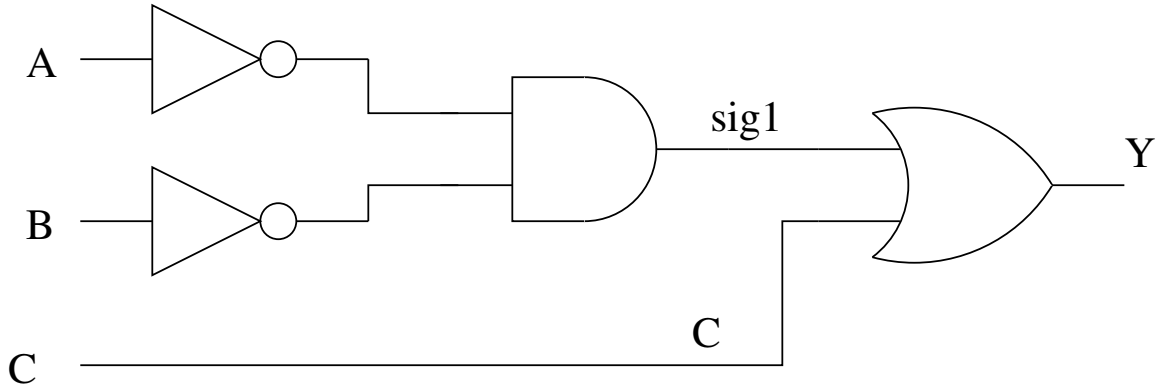
VHDL - **V**ery **H**igh Speed Integrated Circuit **H**ardware **D**escription **L**anguage

VHDL is the standard way for describing circuits, either programmable or fixed logic.

You are able to describe the circuit you want, test its behavior, and then synthesize this to the implementation technology of your choice. (Well, there are lots of caveats, ...)

Basic VHDL building Blocks

Consider the following circuit:

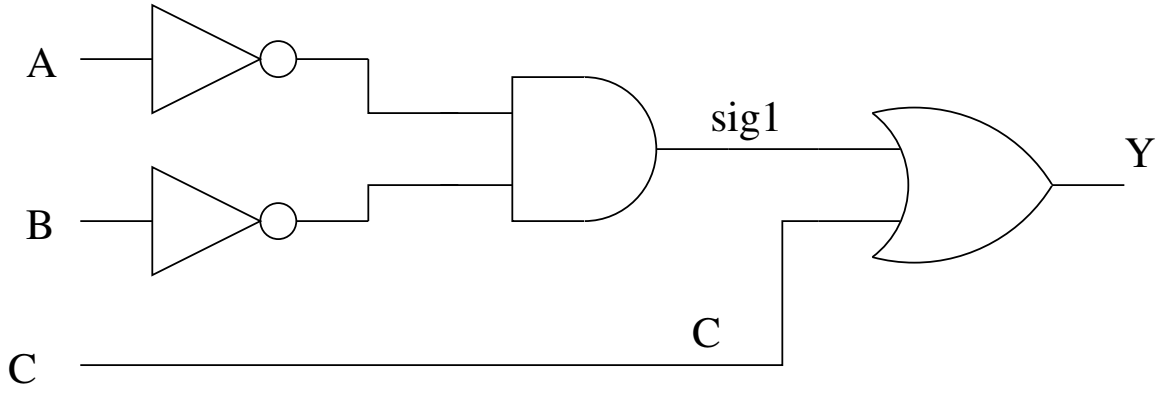



```
-- ENTITY
entity few_gates is
  port(
     a : in std_logic;
     b : in std_logic;
     c : in std_logic;
     y : out std_logic
    );
end fewgates;
```

```
-- ARCHITECTURE
architecture behavior of fewgates is
  signal sig1 : std_logic;
 begin
    process(a,b,c)
    begin
       sig1 <= (not a) and (not b);
       y <= c or sig1;
    end process;
end behavior;
```

Basic VHDL building Blocks

Consider the following circuit:




```
-- ENTITY
entity few_gates is
  port(
     a : in std_logic;
     b : in std_logic;
     c : in std_logic;
     y : out std_logic
    );
end fewgates;
```

```
-- ARCHITECTURE
architecture behavior of fewgates is
  signal sig1 : std_logic;
 begin
     sig1 <= (not a) and (not b);
     y <= c or sig1;
end behavior;
```

Review ISA.vhd

Use editor...

Hardware Description Languages

When simulating ...

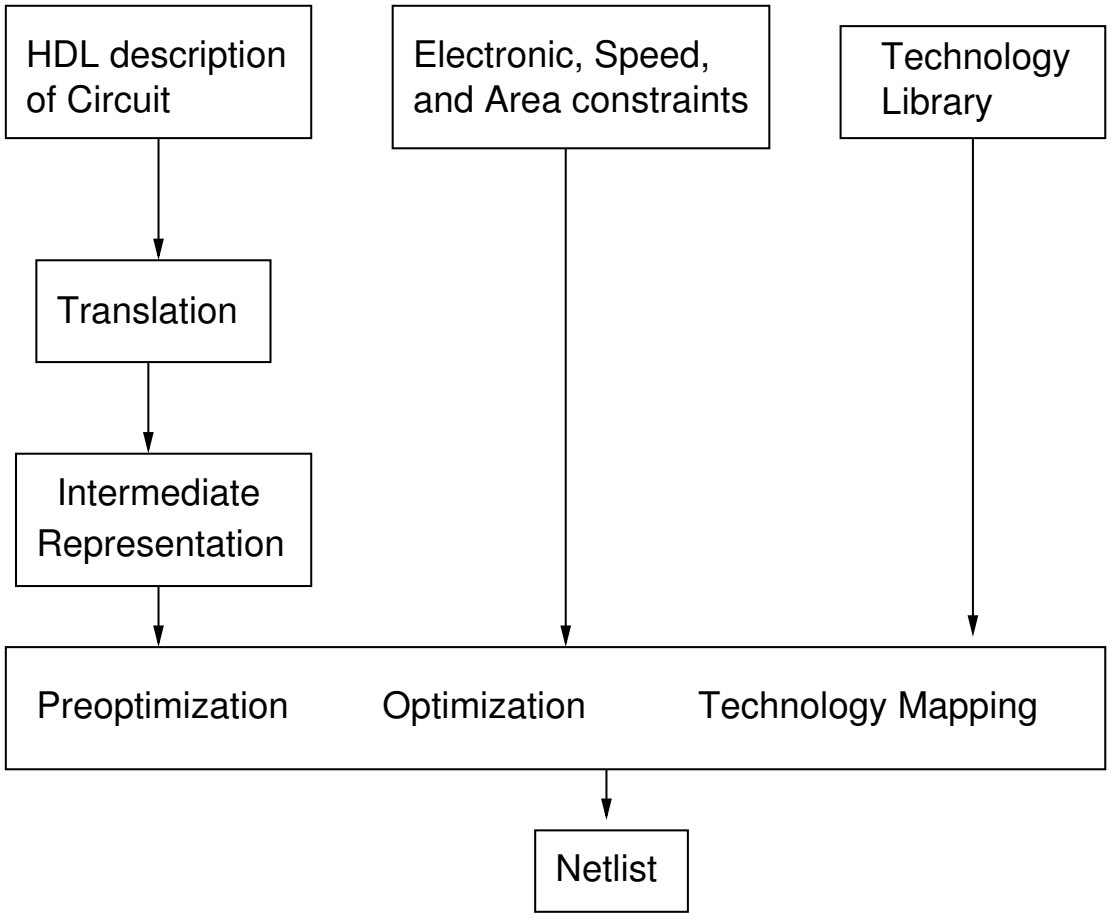
- Analysis – syntax and semantics.
- Elaboration – *builds* blocks
- Initialization – sets variables and default values
- Simulation – executes simulation model

Testbenches

- Special HDL structure for testings inputs and outputs.
- Powerful repeatable form of test
- Testbenches can be used to test multiple levels of the top down process

Logic Synthesis

High-Level Flow



Design Procedure

- Specification
- Formulation - create truth table or boolean equations.
- Optimization - reduce requirements to achieve goal
- Technology Mapping - transform logic diagram to a new diagram or netlist using available technology
- Verification - check the correctness of the final design

Specification

Create a BCD to Excess-3 Code Converter.

The excess-3 code is a decimal digit plus three converted into binary. 0 is 0011, 1 is 0100, etc.

So, let's create a truth table....

Formulation – Truth table

Decimal Digit	Input BCD				Output Excess-3			
	A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0

Formulation

Since we are using BCD input, there are inputs for A, B, C, and D, that we don't care about. e.g. 10-15 (1010 – 1111).

We can put don't cares (X's) in our K-maps and use the to reduce the required inputs.

Optimization

$$W = A + BC + BD$$

$$X = B'C + B'D + BC'D'$$

$$Y = CD + C'D'$$

$$Z = D'$$

Multi-level Optimization

$$T_1 = C + D$$

$$W = A + BT_1$$

$$X = B'T_1 + BC'D'$$

$$Y = CD + C'D'$$

$$Z = D'$$