



HWP1

Hardware Programming

Introduction to digital systems
And structured digital design

Goal of this course

Learn to describe **digital circuits**
with **VHDL** and implement them in
an **FPGA**.

Instructors

- Ron Verhagen
- Harry Broeders

VeRon@hr.nl

BroJZ@hr.nl

Organization

- Compulsory attendance
- HWP1 structure:
 - Lecture
 - Five lectures: Tuesday: 1.5-hours p/w
 - Lab exercise
 - Full-time: Friday: 2.5 hours Online p/w
 - Homework
 - 56.5 hours total
 - Theory from the book
 - “Circuit Design with VHDL” (Third Edition)
 - Volnei A. Pedroni
 - ISBN 978-0-262-04264-2
 - Preparing assignments / writing code
 - Making simulations at home / outside regular class times

Planning: theory

- First week
 - Introduction digital systems
 - Introduction to FPGAs
 - Structured digital Design
 - Modelling concepts in VHDL
- Second week
 - Introduction VHDL
 - Design verification
 - Code structure and data types
- Third week
 - Combinational versus sequential design
 - Concurrent and sequential code
 - Signals and variables
- Fourth week
 - Components
 - Generics
- Fifth week
 - Designing state machines

Earning your grade

In the course manual you will find the the following

Action		Completed
Step 1	Complete all 5 Labs before starting on your final assignment	Yes
Step 2	Final Assignment (FA) ingredients:	% (leerdoel)
2.1	Architecture (schematics and drawings) with basic ** functionality	20% (LD1)
2.2	VHDL code of all architectural functions and justification in report	40% (LD2,LD3)
2.3	A functional VHDL Test Bench	20% (LD4)
2.4	A working FPGA prototype with the basic** functionality	20% (LD5)
<i>** basic functionality (complexity) may be expanded to include additional features for a higher grade. This is agreed upon before starting this assignment. Basic functionality = 6</i>		
HT1	Retake	

Agenda

- **Introduction to digital systems and FPGAs**
- Structured digital design
- Modelling concepts in VHDL

Computing realization

- **Software programmed processors:** software flexibility, fixed instructions and performance is limited.
- **Application Specific Integrated Circuits (ASIC):** expensive, time consuming, best performance.
- **Reconfigurable computing:** high flexibility, good performance and fills the gap between hardware and software. This is the domain of FPGAs.

Digital Logic: 1980's vs today FPGA



Field-programmable Gate Array (FPGA)

Transistor count for key FPGA families

FPGA	Transistor count	Date of introduction	Manufacturer	Process
Virtex	~70,000,000	1997	Xilinx	
Virtex-E	~200,000,000	1998	Xilinx	
Virtex-II	~350,000,000	2000	Xilinx	130 nm
Virtex-II PRO	~430,000,000	2002	Xilinx	
Virtex-4	1,000,000,000	2004	Xilinx	90 nm
Virtex-5	1,100,000,000	2006	Xilinx	65 nm
Stratix IV	2,500,000,000	2008	Altera	40 nm
Stratix V	3,800,000,000	2011	Altera	28 nm
Arria 10	5,300,000,000	2014	Altera	20 nm
Virtex-7	6,800,000,000	2011	Xilinx	28 nm
Stratix 10 Family device, 10GX5500/10SX5500	17,000,000,000	2017	Intel (formally Altera)	14 nm
Virtex-Ultrascale XCVU440	20,000,000,000+	2014	Xilinx	20 nm
Everest	50,000,000,000	2018	Xilinx	7 nm

Source: <https://www.napatech.com/road-to-fpga-reconfigurable-computing/> 2018

Comparison

Field-programmable Gate Array (FPGA)

semiconductor versions of the Swiss Army Knife.

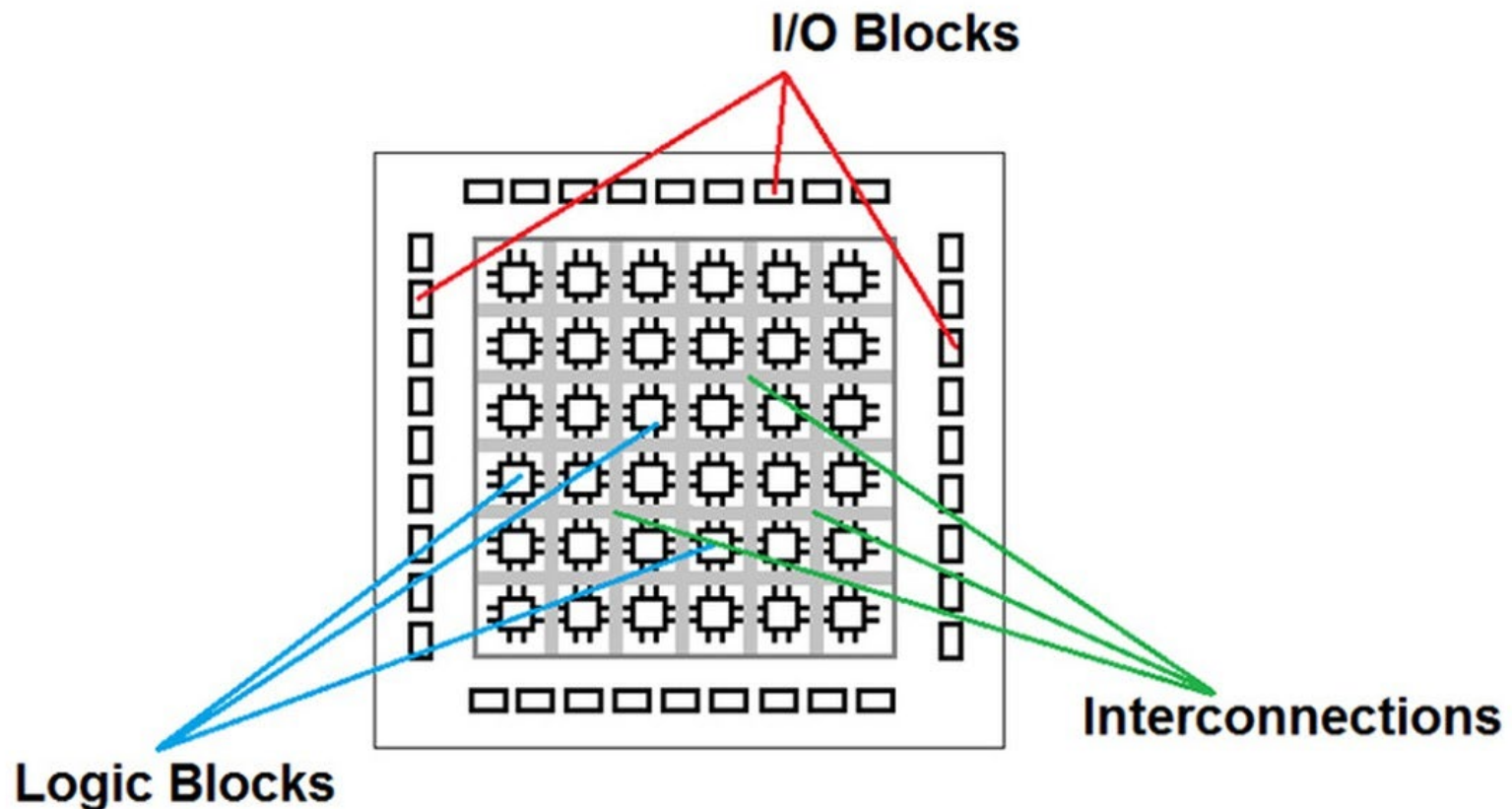
Microcontrollers vs. FPGA's

Microcontroller	FPGA
Programming languages (C, C++, Assembly) Optimized for general purpose computing	Programming Languages VHDL, Verilog, Open Computing Language; Or use Schematics
Control over the Software; 10's to 100's of cores.	Control over the Hardware; Hardware timed execution; millions of programmable digital logic cells.
Predefined instruction set and datapath widths	No predefined instruction set or datapath widths
Reprogrammable - unlimited	Reconfigurable - unlimited

	Microcontroller	FPGA	ASIC
Clock speed (frequency)	Can be very high (Up to 4 GHz)	Average (up to 0.7 GHz)	Can be very high (several GHz)
Throughput (data per second)	Limited	Can be very high	Highest
Power consumption	Low	High	Low
Cost	0.3 \$ ~ 2000 \$	3 \$ ~ 2000 \$	From \$1.000.000 NRE
HW flexibility (design time)	Low	High	High
HW flexibility (run time)	Low	High	Low
Peripherals	Limited to what IC manufacturers offer	Many, as long as pins and logic elements are available.	Many, as long as physically possible.
Behavior	Almost exclusively sequential	Sequential or concurrent	Sequential or concurrent
Potential parallelism	Low for most single-cores	High	High

Technology

- Digital component.
- 3 building blocks:



Technology – Logic Blocks

A basic logic building block is the Logic Element

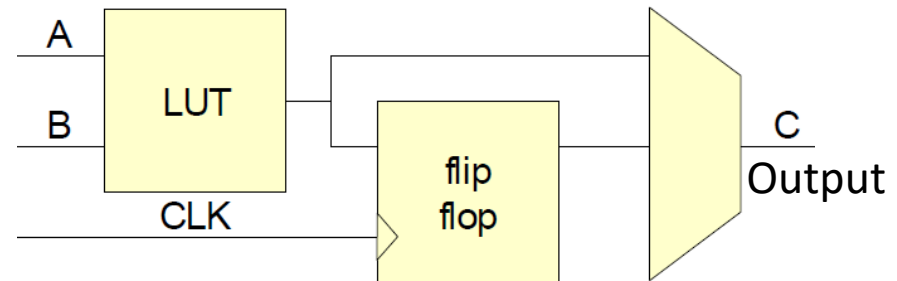
AND			NAND			OR			NOR			XOR		
In A	In B	Out	In A	In B	Out	In A	In B	Out	In A	In B	Out	In A	In B	Out
0	0	0	0	0	1	0	0	0	0	0	1	0	0	0
1	0	0	1	0	1	1	0	1	1	0	0	1	0	1
0	1	0	0	1	1	0	1	1	0	1	0	0	1	1
1	1	1	1	1	0	1	1	1	1	1	0	1	1	0



Synthesis (programmed into the LUT)

LUT			
A	B	C...F	OUTPUT
0	0	X	0
0	1	X	1
1	0	X	1
1	1	X	0

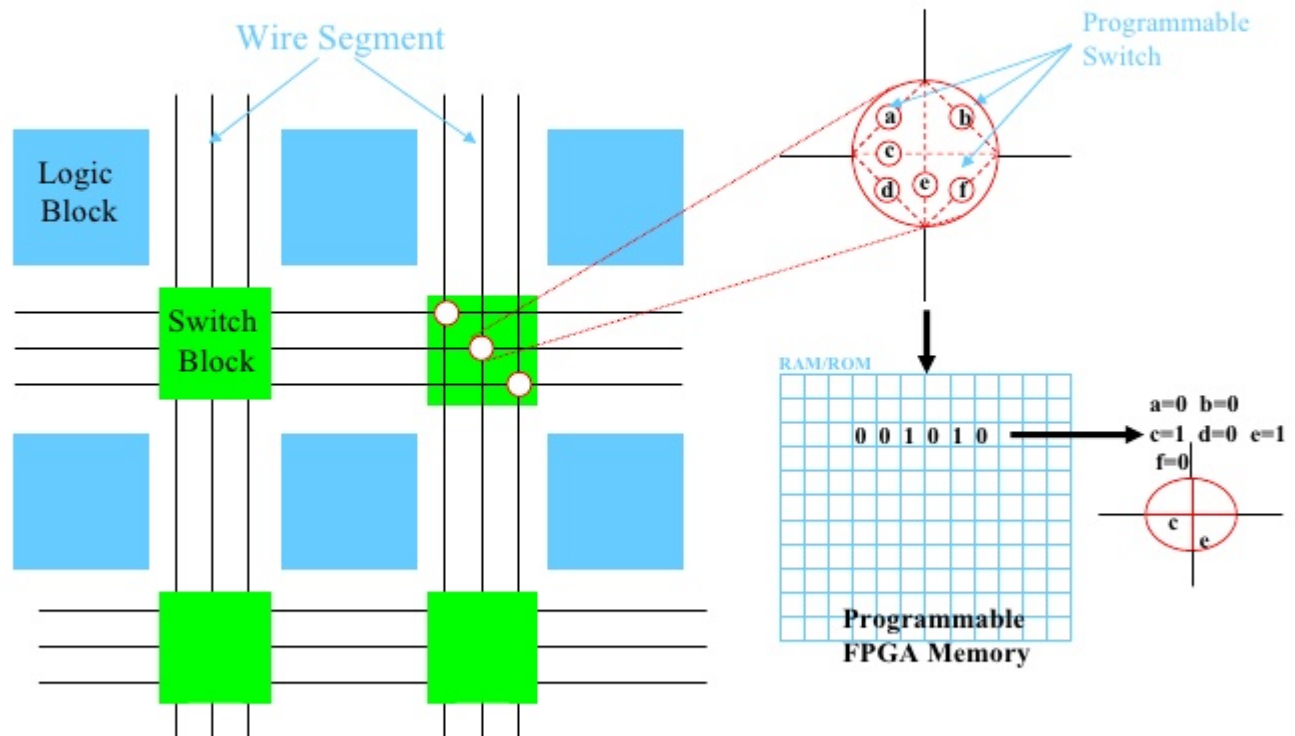
Logic Element



If we bypass the flip-flop using the multiplexer, what is the function of this Logic Element?

Technology - Interconnections

FPGA interconnect



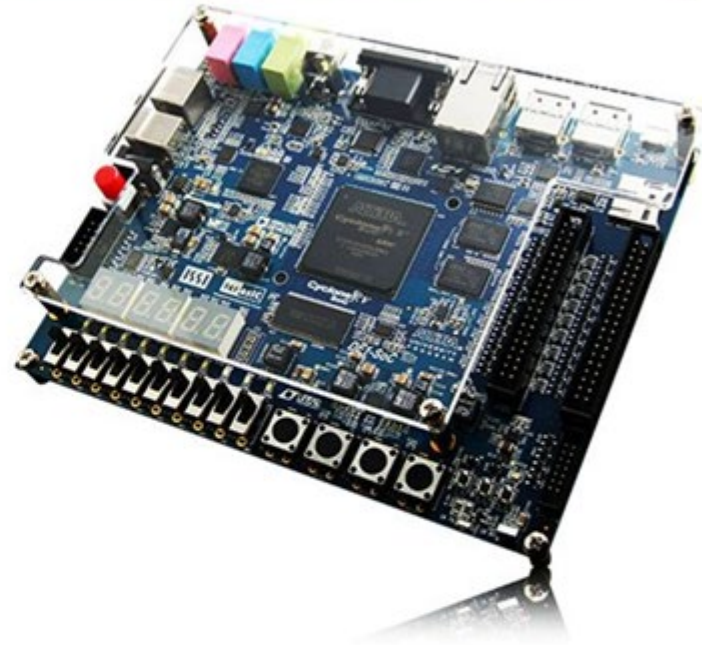
Dec 26, 2010

50

THE DE1 DEVELOPMENT BOARD

DE1-SoC

- **\$377 USD**
- **Cyclone V SoC FPGA**
- **Dual-core ARM Cortex-A9**
 - 1GB DDR 3 SDRAM, MicroSD
 - USB, Triple-speed Ethernet
 - ADC, Accelerometer
 - LED, Pushbutton
- **FPGA**
 - 85K Programmable Logic Elements
 - 64 MB SDRAM
 - DVD-quality audio in/out, Video in/VGA out
 - PS/2, IrDA
 - 4 debounced pushbuttons, 10 slider switches, 10 red LEDs, six 7-segment displays
 - Expansion headers
- **Built-in USB Blaster for FPGA programming**



ALTERA
MEASURABLE ADVANTAGE™

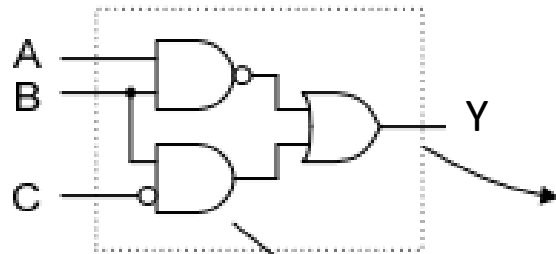
Agenda

- **Introduction to digital systems and FPGAs**
- **Structured digital design**
- **Modelling concepts in VHDL**

Structured digital design

- Determine the function you want to perform
 - **Architecture**
- Find a method to implement and test the function
 - **Implementation**
 - **Simulation (TDD)**
- Use tools to materialize the method
 - **Realization**
- Verify your design
 - **Evaluation**

Example VHDL code



Bounding box
represented by entity
statement; behavior by
architecture statement

```
library ieee;  
use ieee.std_logic_1164.all;
```

```
entity Example is  
  port (A,B,C   : in  STD_LOGIC;  
        Y       : out STD_LOGIC);  
end Example
```

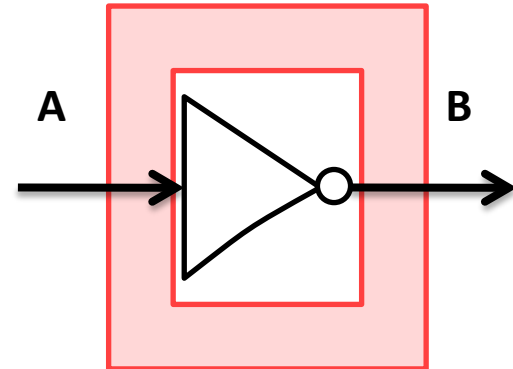
```
architecture behavioral of Example is  
begin  
  Y <= (not (A and B) or (B and not C));  
end behavioral;
```

1. Import the necessary libraries
2. Create an entity block.
 1. One per design
 2. Interface in and out of design unit
3. Describe architecture block
 1. Defines how the entity operates using Structural or Behavioral code.

From VHDL to FPGA

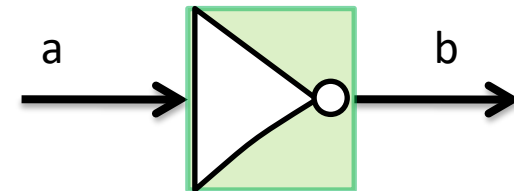
```
ENTITY inverter IS  
  PORT (a: IN  STD_LOGIC;  
        b: OUT STD_LOGIC);  
END inverter;
```

=



```
ARCHITECTURE voorbeeld OF inverter IS  
BEGIN  
  b <= NOT(a);  
END voorbeeld;
```

=



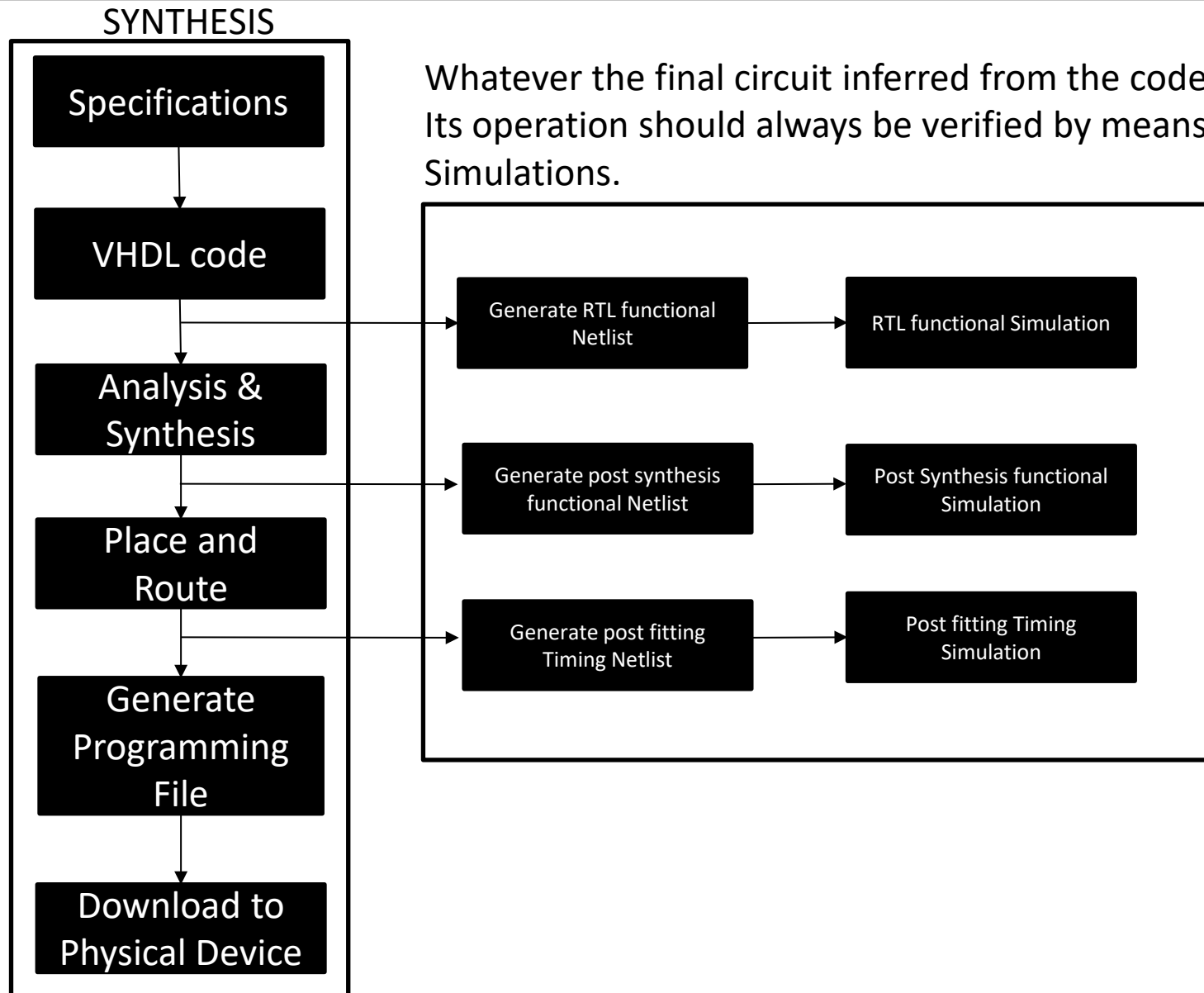
Structural decomposition

- Top-down design: no constraints on the availability of hardware
- Bottom-up design: design is conditioned by what's available (e.g., the number of gates)
- Typical dimensions for the Digital Design Space: speed, chip area (cost), power

Agenda

- Introduction to digital systems and FPGAs
- Structured digital design
- **Modelling concepts in VHDL**

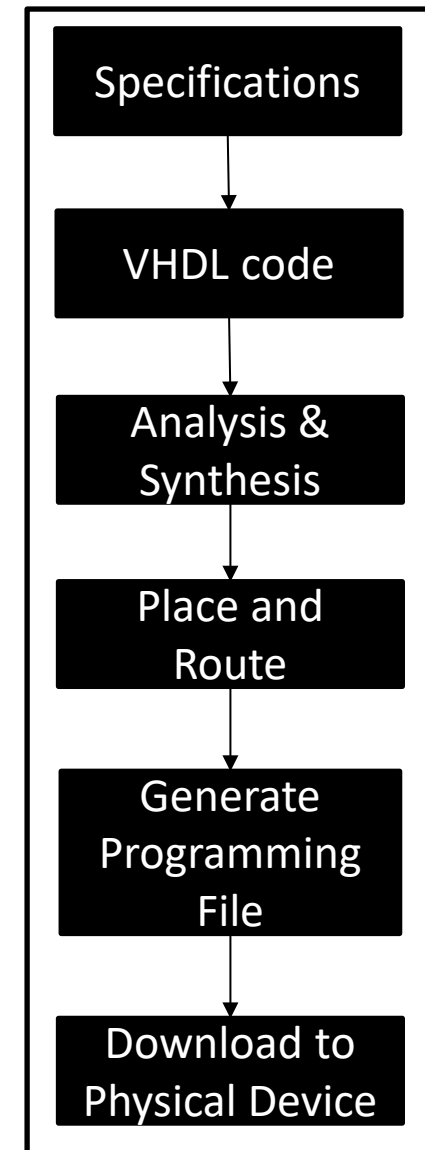
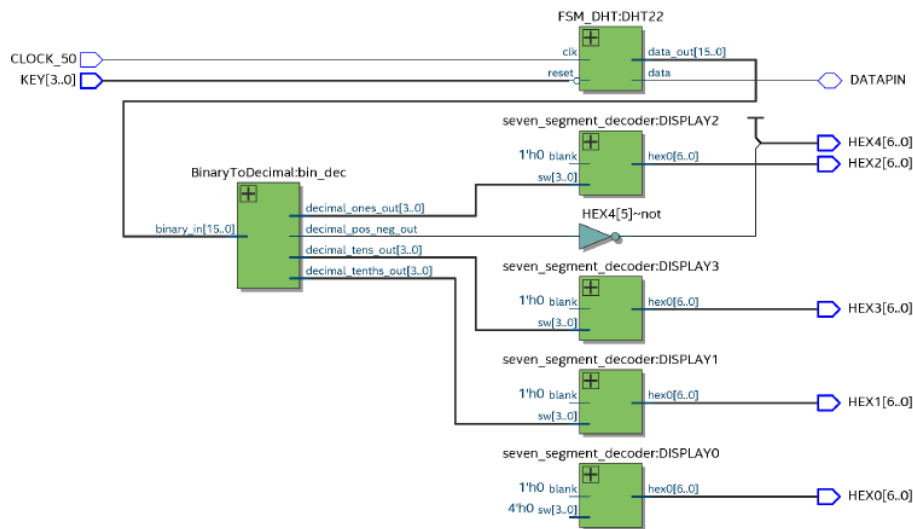
Simplified VHDL Design Flow



From VHDL to FPGA

1. Synthesize

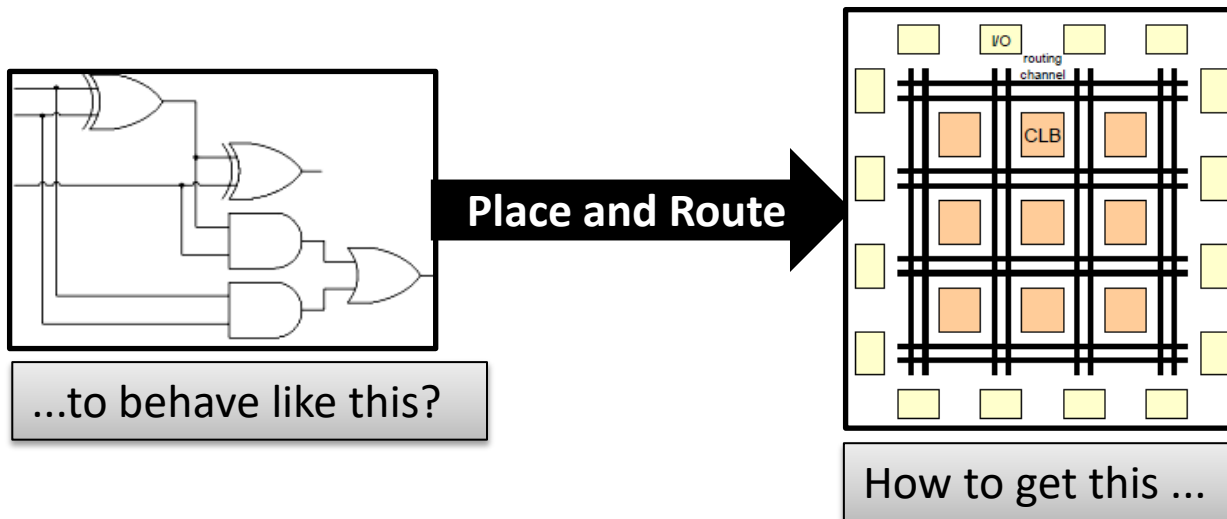
- Simulate (Analysis & Elaboration)
- Compiles code into RTL (Register Transfer Level) schematic. Always take a look at this



From VHDL to FPGA

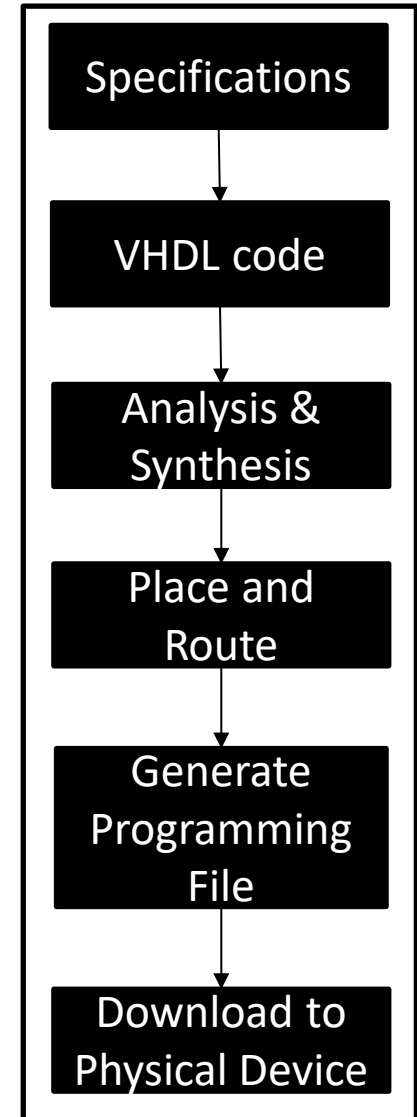
2. Place and Route

- How to configure the FPGA elements to become the functional equivalent of the RTL schematic?
- Find the optimal paths through the FPGA
- Timing analysis



3. Generate bitstream

- Contains the setting of every switch in the FPGA



Modeling concepts in VHDL

- Two abstract models to describe your design in VHDL
- *Behavioral Model:*
 - Defines how outputs are computed as function of inputs
 - Can make use of a “process”
- *Structural Model:*
 - Uses pre-defined components to make a structure
 - Wider use of “portmap”

Design verification

- Check if your realization matches your expectations.
- In VHDL we use test benches to verify the results.
- In this course you will create test benches to verify your designs.
- In (simple) simulations we don't know if the actual design meets timing constraints. We will discuss this later.

Homework

- Covered today:
 - Introduction to digital design and FPGAs
 - Structured design
 - Modeling concepts in VHDL
- Team up with another student
- Start working on the laboratory
- <https://nandland.com/fpga-101/>
- <https://fpgatutorial.com/vhdl/>
- Useful:
 - Chapter 5 “Introduction to VHDL”
 - Chapter 6 “Code Structure and Composition”
- Next week:
 - Chapter 7 and 8 “Predefined and User-Defined Data Types”
 - Chapter 9 “Operators and Attributes”
 - Chapter 18 “Introduction to Simulation with Testbenches”

First week: Lab

- Introductory Assignment 1
 - Getting to know Quartus
 - Getting to know ModelSim
 - Getting to know the DE1 SoC development and Education Board

