



LUND
UNIVERSITY

EITF35: Introduction to Structured VLSI Design

Part 7.1.1: Wrap Up

Liang Liu
liang.liu@eit.lth.se



Outline

□ Conclusion

□ Next Step

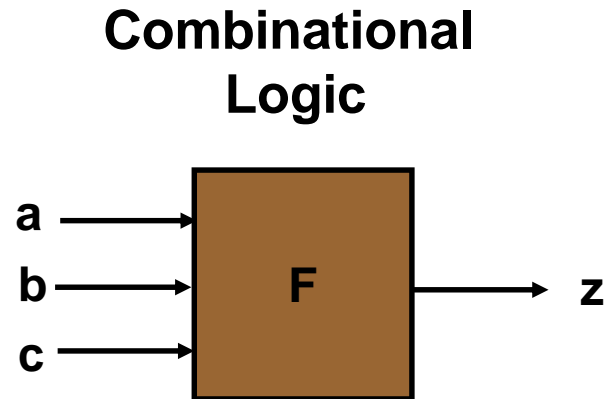
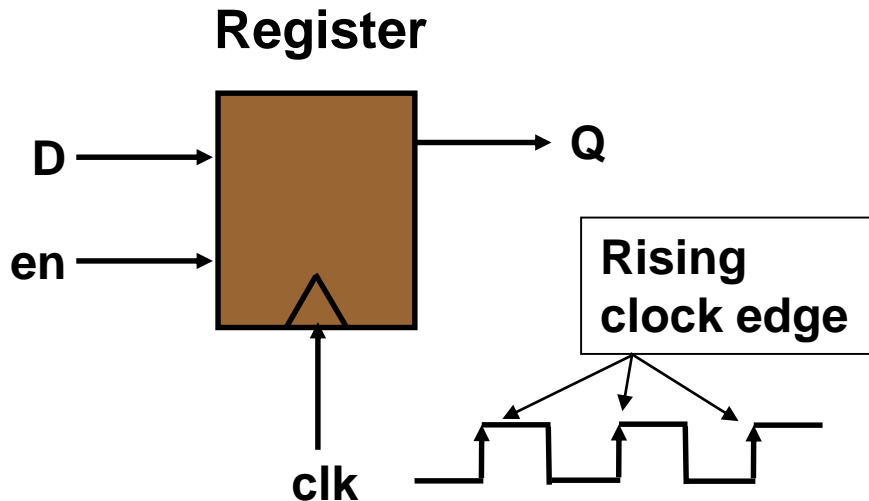
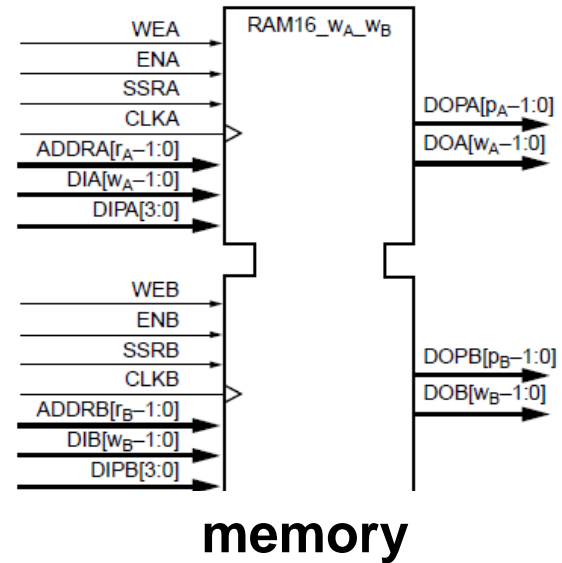


What we have learned

Basic Building Blocks

- Combinational logic
- Registers
- Memory

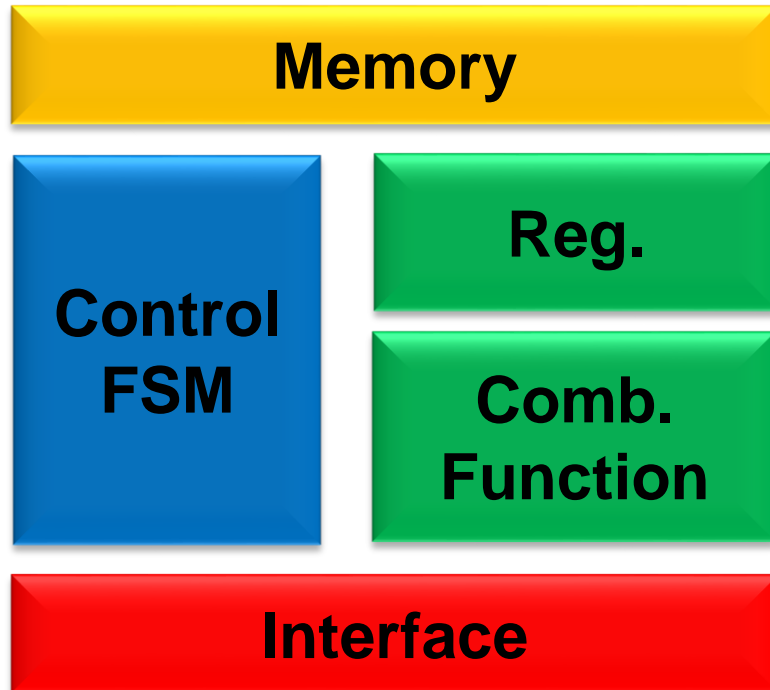
Construct VLSI Systems



What, How, When



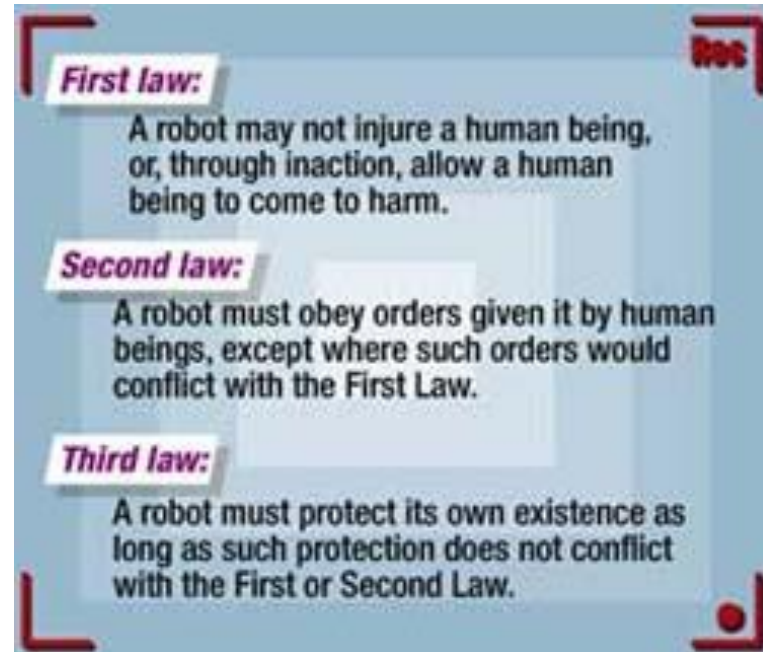
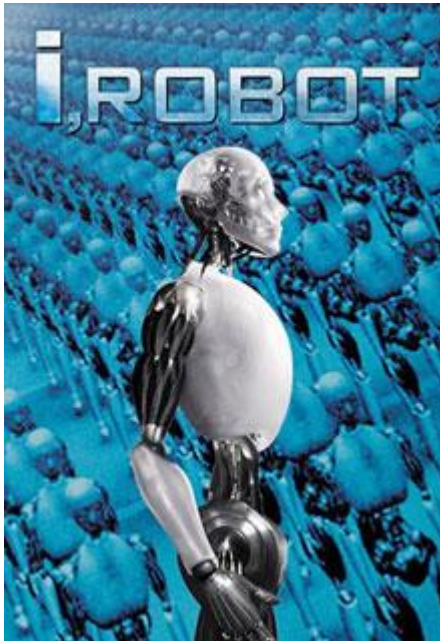
What we have learned



- ❑ **Assign.1**
 - Sequence Detector
- ❑ **Assign.2**
 - Keyboard controller
- ❑ **Assign.3**
 - ALU
- ❑ **Assign.4&5**
 - “Micro Processor”



VLSI 'Laws'



**Workable hardware meeting
the requirement**



Suggestion 1 – Design Flow

Specification

- ❑ Understand the **requirement**
 - Functionality & performance
 - Time to deadline



Rem or Mod?

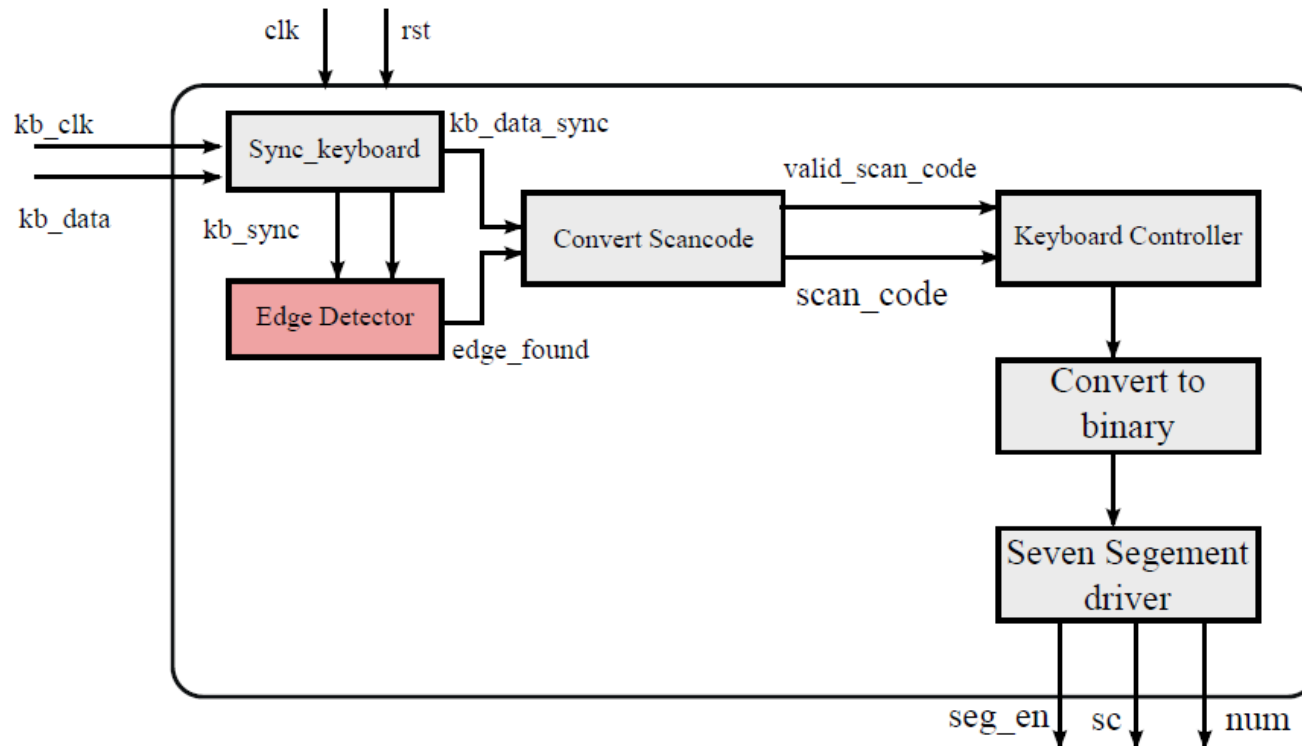


Suggestion 1 – Design Flow

Specification

Block Diagram

- ❑ Understand the **requirement**
 - Functionality & performance
- ❑ Draw a **block diagram**



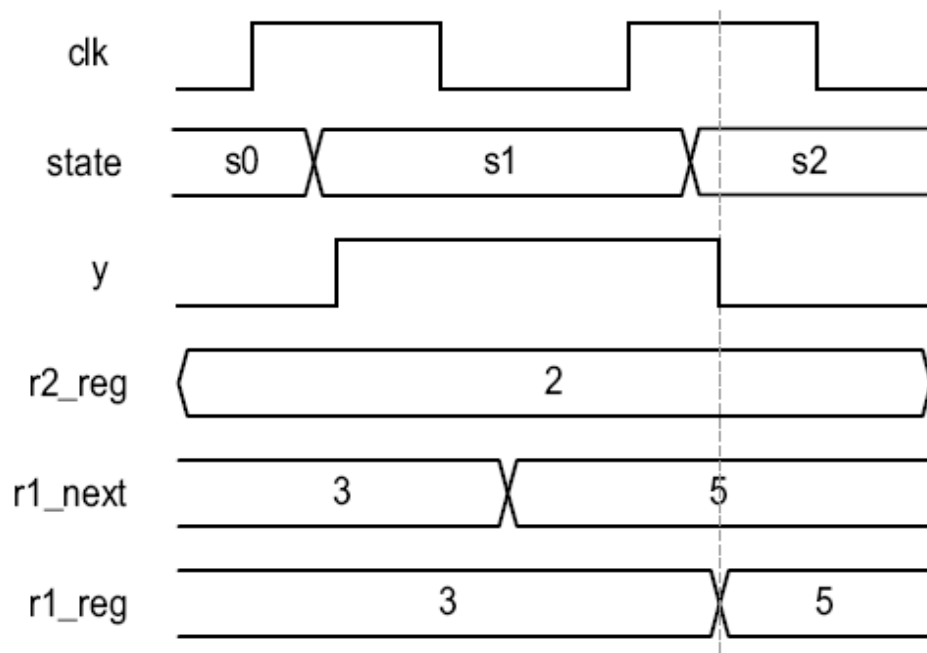
Suggestion 1 – Design Flow

Specification

Block Diagram

Timing Diagram

- Understand the **requirement**
 - Functionality & performance
- Draw a **block diagram**
- Understand the **timing**



(c) Timing diagram



Suggestion 1 – Design Flow

Specification

Block Diagram

Timing Diagram

VHDL Coding

Implementation

- Understand the **requirement**
 - Functionality & performance
- Draw a **block diagram**
- Understand the **timing**
- VHDL is only a way to **ask for component**
- Implement your design



**You are a
architecture
designer**



Suggestion 2 – Verification

- ❑ Verification is strictly required at each design stage
- ❑ Verify as much as possible, especially at early stage
 - Simulate before implementation
 - Verify blocks before integration

The cost of fixing a bug grows exponentially with the stage!!!



Suggestion 3 – Debug

□ There **MUST** be an error!

- Small & simple



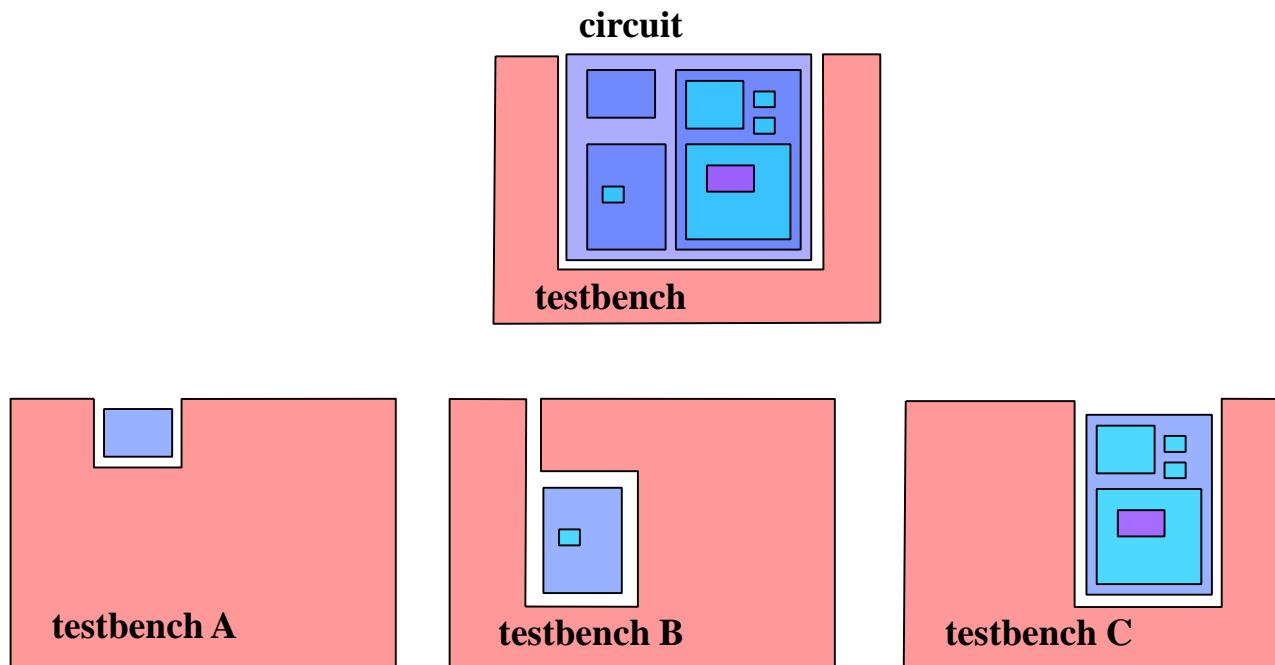
Suggestion 3 – Debug

❑ There **MUST** be an error

- Small & simple

❑ Locate the error

- Divide and conquer



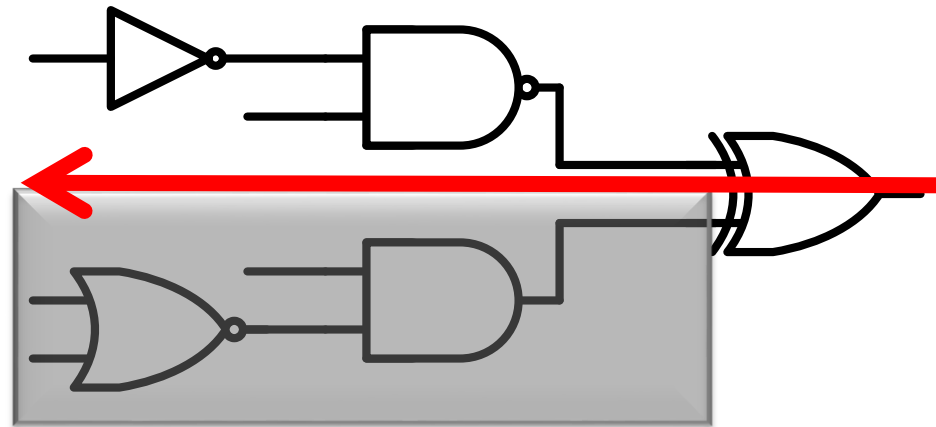
Suggestion 3 – Debug

❑ There **MUST** be an error

- Small & simple

❑ Locate the error

- Divide and conquer
- Error isolation
- Trace back
- Special stimuli

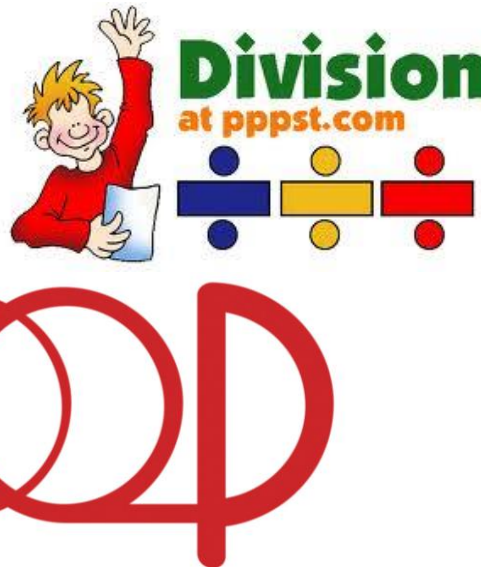
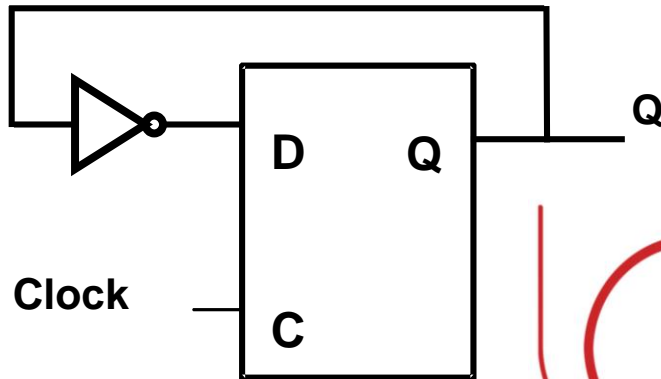


Suggestion 4 – Synthesizable VHDL

- ❑ RTL simulation pass \neq implementation success
- ❑ Typical un-controllable codes

- Latches
- Mixed reg. with comb.
- “wait” & “after”
- Division...
- Loop & variable

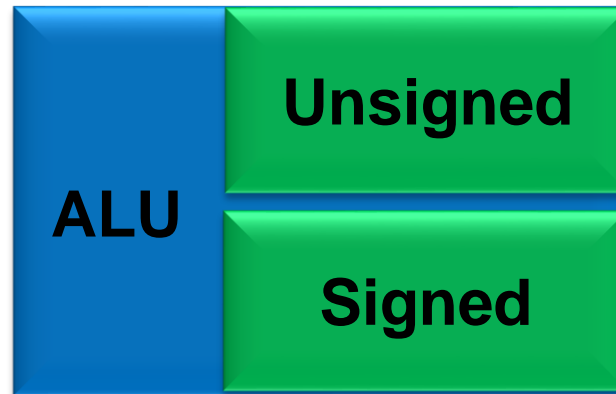
```
process (a,b)
begin
    c<=c+a+b;
end process;
```



Suggestion 5 – Hierarchical Design

□ Hierarchical design

- **Divided-and-conquer** strategy
- Divide a system into *smaller parts*
- Constructs each module *independently*



Suggestion 6 – Optimize at early stage

	Opt.	Ave.	Raw
Speed	200MHz	80MHz	20MHz
Area (Slice)	75	190	310



Suggestion 6 – Optimize at early stage

Algorithm

```
x mod n = x - n · floor(x/n)
```

Architecture

Function Block

```
x = (x >> 6) + (x & 0x3f);  
x = (x >> 4) + (x & 0xf);  
x = (x >> 2) + (x & 0x3);  
x = (x >> 2) + (x & 0x3);  
if (x == 3) x = 0;
```

Implementation

	Slices	4-input LUT
8-bit Multiplier	37	70
8-bit Adder	4	8



Suggestion 7 – Design Management

❑ Folder

- Project_name (A3_ALU)
- Algorithm, RTL, Sim, Synthesis, P&R...
- Synthesis: Netlist, Script, Report, **Readme.txt**...

❑ File Name

- “stage_design”
- Design module or entity (“**m_alu**” or “**ent_alu**”)
- Test bench (“**tb_alu**”)

❑ Signal Name

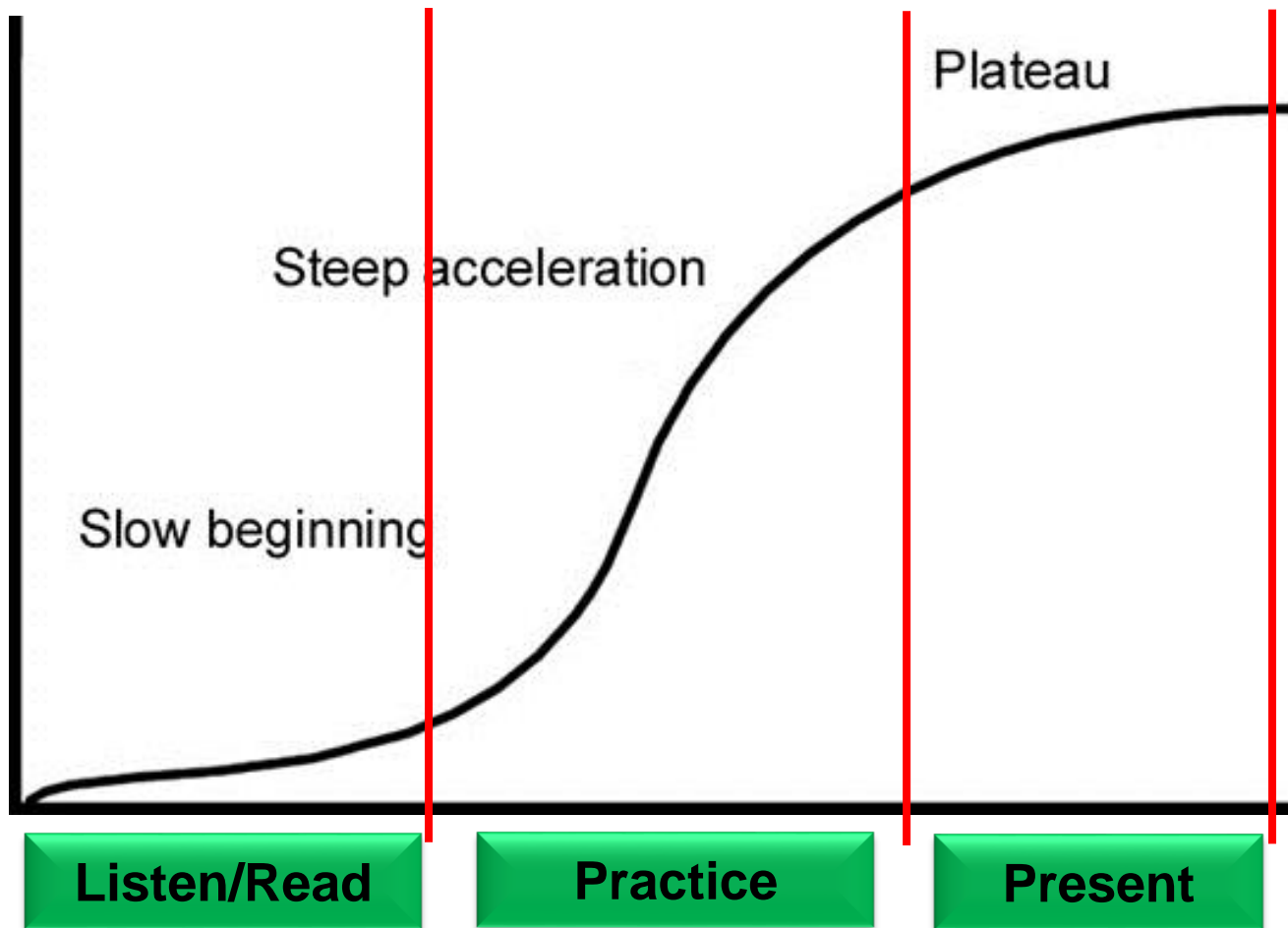
- “direction_function_feature”
- Pos-edge clock (**i_clk_p**)
- Active low reset (**i_rst_n**)

❑ Comments

- At least 30% of the codes

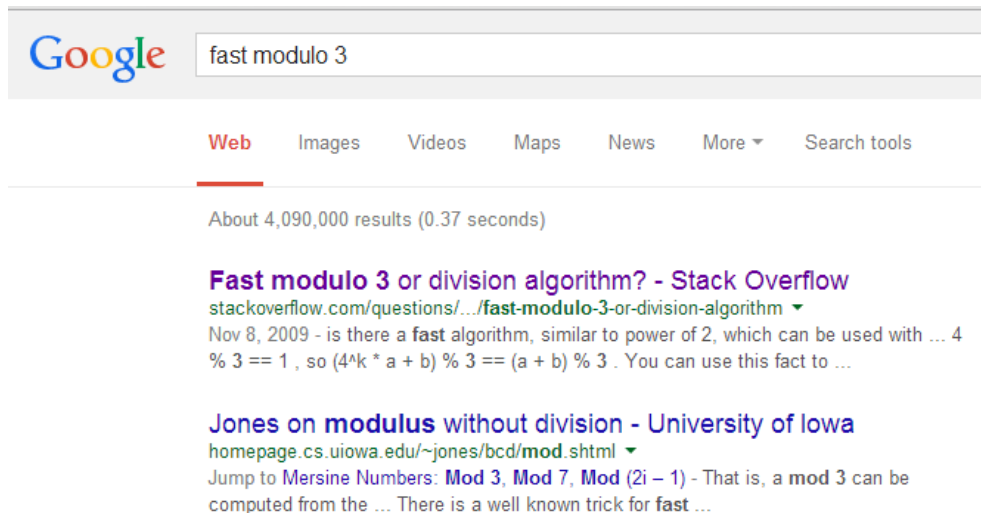


Suggestion 8 – Practice makes perfect



Suggestion 9 – Source finding

<http://www.eit.lth.se/index.php?ciuid=774&coursepage=4569&L=1>



The screenshot shows a Google search interface. The search bar contains the text "fast modulo 3". Below the search bar, there are tabs for "Web", "Images", "Videos", "Maps", "News", "More", and "Search tools". The "Web" tab is selected. Below the tabs, it says "About 4,090,000 results (0.37 seconds)". There are two search results listed:

- Fast modulo 3 or division algorithm? - Stack Overflow**
stackoverflow.com/questions/.../fast-modulo-3-or-division-algorithm
Nov 8, 2009 - is there a **fast** algorithm, similar to power of 2, which can be used with ... $4 \% 3 == 1$, so $(4^k * a + b) \% 3 == (a + b) \% 3$. You can use this fact to ...
- Jones on modulus without division - University of Iowa**
homepage.cs.uiowa.edu/~jones/bcd/mod.shtml
Jump to Mersine Numbers: **Mod 3**, **Mod 7**, **Mod (2i - 1)** - That is, a mod 3 can be computed from the ... There is a well known trick for fast ...



**Rakesh
Gangarajaiah**



**Oskar
Andersson**



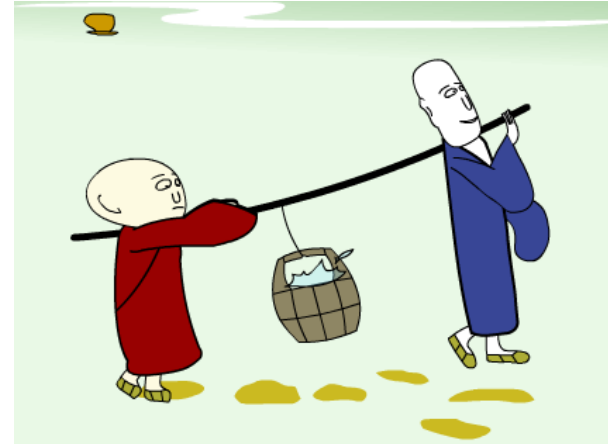
**Hemanth
Prabhu**



**Steffen
Malkowsky**



Suggestion 10 – Team work



Outline

Conclusion

Next Step

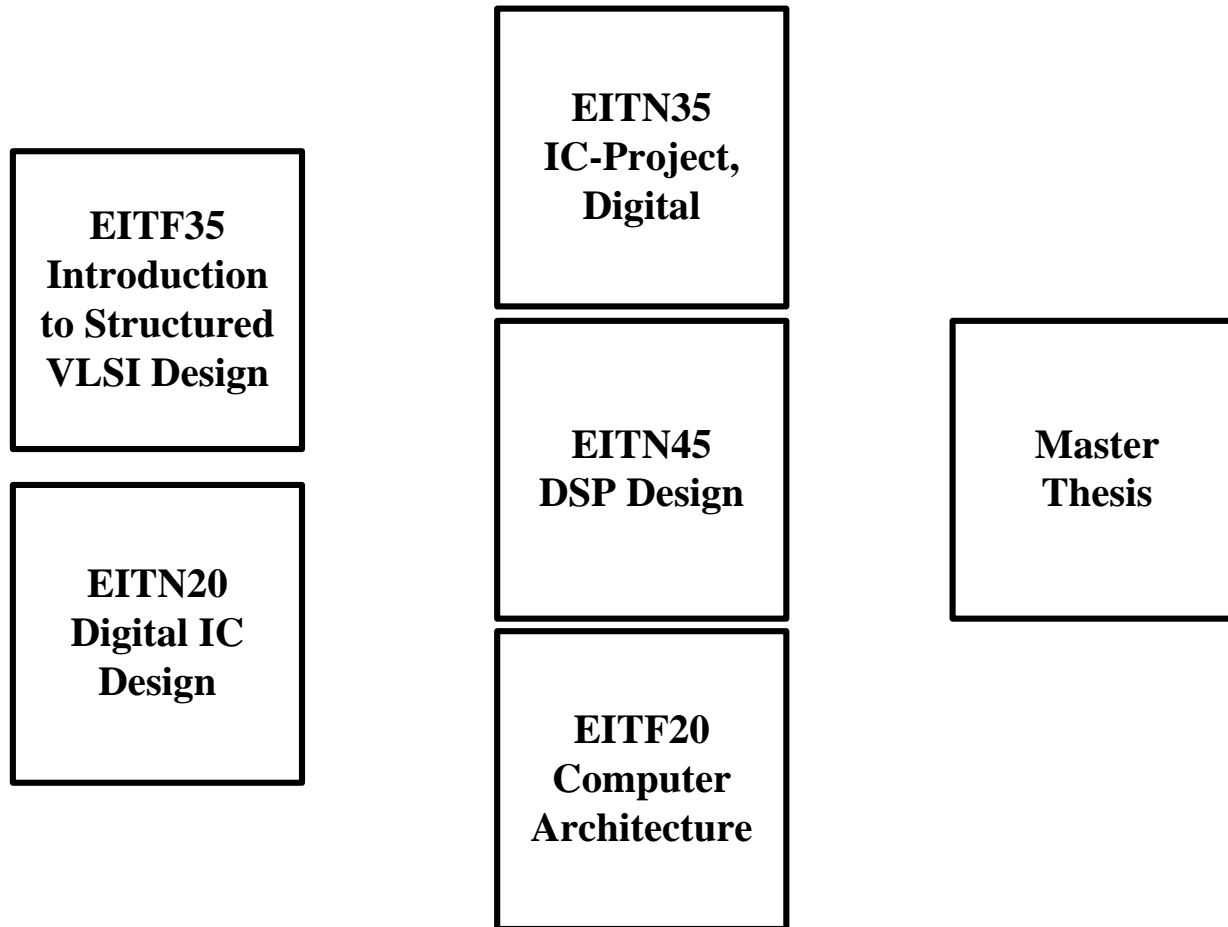


This Course

- ❑ TA assistant ends at 17th Oct. (deadline of project 4)
- ❑ Need self problem-solving capability for project 5



Digital Path



Encourage Publication

NORCHIP 2014

The Nordic Microelectronics event

32nd Norchip Conference
27-28 October 2014, Tampere, Finland

General Scope of the Conference

The NORCHIP conference is the main microelectronics event of the Nordic countries. The annual IEEE CAS sponsored conference covers all areas of microelectronics, spanning from large digital systems to simple analog circuits. The wide scope of NORCHIP is intentional promoting cross-field collaboration. NORCHIP is a well established conference with representation from both academia and industry. Papers of the highest scientific and technical quality are presented together with selected invited speakers and pre-conference tutorial sessions.



Norchip

Conference secretariat:



Technoconsult ApS
Agern Allé 3
DK-2970 Hørsholm
Denmark

Tel: +45 22 12 52 44

Fax: +45 45 76 57 08

E-Mail: info@norchip.org

WWW: www.norchip.org



Digital Asic Group - Projects

Massive MIMO: solution beyond LTE-A (4G)

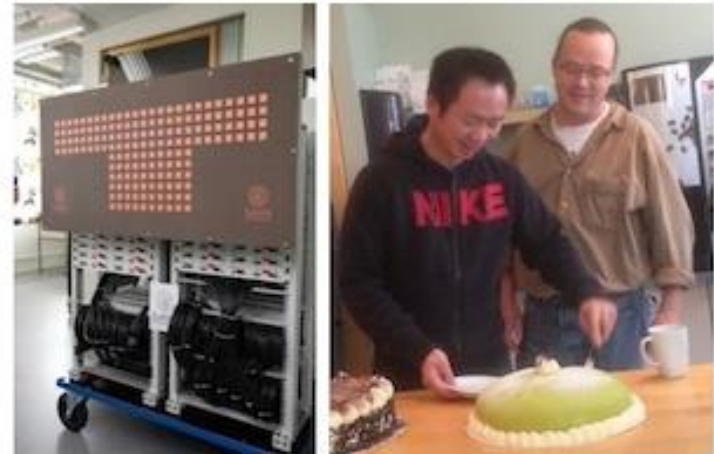
News archive

MIMO wheels-up

LUMAMI-testbed for massive-MIMO is up and running after hard work by especially Liang Liu, Joao Vieira och Steffen Malkowsky. The head of EIT supervises that Liang is as able in cake-cutting as in the work with the testbed.

Text/Photo: Anders Borgström

2014-10-06



Digital Asic Group - Projects

□ DARE: Digital Assistant Radio Evolution

- Everything now is digitalized
- Analog interface is still needed
- We are trying to **digitalize the interface** as much as possible



	Price
16-bit ADC	185\$
10-bit ADC	50\$



Digital Asic Group - Projects

□ Adaptive Processing with Smart Circuit

- Changing environment & application
- Processor adaptive to requirement
- Energy efficient with satisfied QoS

□ *Changing environment*



□ *Various applications*



**Smart
Processor**



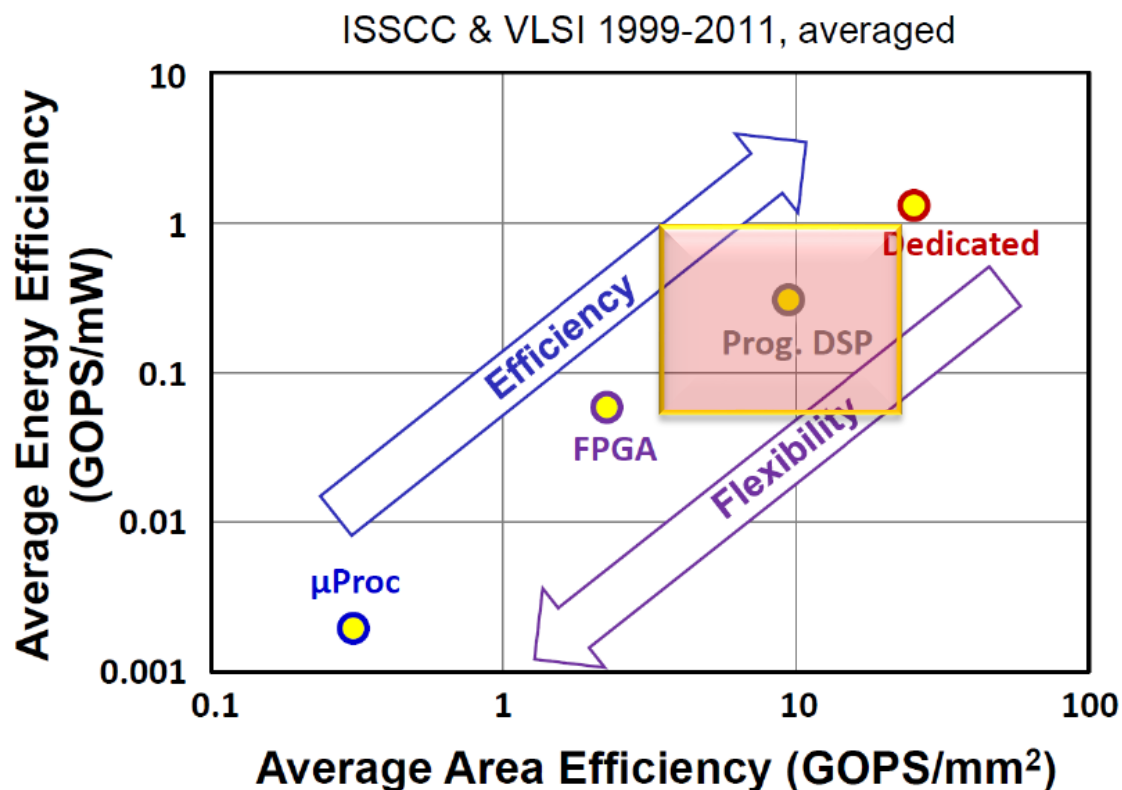
□ *Low Power Consumption*



Digital Asic Group - Projects

□ HiPEC: High Performance Embedded Computing

- Flexible to support multi-standard, multi-version, multi-mode
- Area & energy efficient to be adopted in portable device
- Application-specific processor (vector processor)



Possible Internship



Stefan.Lundberg@axis.com



Internship in the GPU modelling team in Lund (Part time 1-2 days a week)

Job Description:

The model team develop software models of the GPU hardware. The work intended for this intern position is to produce a model from existing RTL by compiling it into a x86 library and wrap it into a systemC/TLM2.0 environment. We would like to study the feasibility of this process for our GPU cores and also the properties of the end product.

Internship in the GPU modelling team in Lund (Part time 1-2 days a week)

Job Description:

The model team develops a bit-exact and cycle approximate model of the GPU HW. This model is used for a number of use-cases including verification and performance predictions. The work intended for this intern position is to improve the host performance of the GPU modelling framework. Since both graphics content and our GPU architecture is becoming more complicated simulation speed is becoming more and more important.



Good Luck !

