



The **Microelectronics  
Training Center**

*The MTC is an initiative within the INVOMECE division*  
Industrialization &  
Training in  
Microelectronics



# The Microelectronics Training Center

## IMEC v.z.w .

### Teacher Guidelines

### Lab Exercises on the Virtex™-II Pro XUP board

Version: 1.5  
Date: 26 Oct 2007  
Author : Bart De Mey  
Modified by: BDM, Levi Geenen, Geert vwb

This material was developed with support of the European Social Fund. ESF: Prevent and combat unemployment by promoting employability, entrepreneurship, adaptability and equal opportunities between women and men, and by investment in people.

<http://www.esf-agentschap.be>



# Introduction

Since 1983, the Microelectronics Training Center has been supporting and life-long training teaching staff of higher technical schools and universities. This activity started as an independent not-for-profit organization: INVOMECE – a Dutch acronym for Industrial Training in Microelectronics. After a short time INVOMECE was integrated into IMEC, as one of its divisions.

The Microelectronics Training Center provided EDA tools to the academia, and started a Multi-Project Wafer service in 1985. Today this service is manufacturing some 500 different ASIC prototypes and small volume batches per year: [www.imec.be/europractice](http://www.imec.be/europractice).

Together with the Flemish polytechnic schools and Flemish universities, it was decided to develop a set of hands-on lab exercises to be used throughout the teaching curriculum of electronic engineers, starting from the first bachelor up to the final master year.

The goal was to develop hands-on exercises on a limited set of EDA tools and on one single hardware implementation platform to allow a student to get acquainted with the tools and the platform as early as possible in the curriculum. As a result teaching can focus upon concepts, rather than having to spend time in introducing new EDA tools and new implementation platforms.

The Virtex-II Pro based Xilinx® University Program Board was selected, because it can be used for simple exercises as well as to implement complex systems even beyond the needs and resources available within the educational context.

The main purpose of the labs developed at IMEC is to increase the general digital design knowledge of the students without putting too much emphasis on the tools or development board being used. This is the main difference between the IMEC labs and the excellent and detailed Xilinx labs introducing the tools and the design environment.

## Objectives

This document gives an overview of the rationale behind the lab exercises developed by the Micro-electronics Training Center of IMEC and should allow a teacher to understand the exercise portfolio and to make a selection of labs and modules fitting into the curriculum he has in mind.

Each lab-exercise covers the implementation of a function or (sub) system. Most labs are subdivided into modules. Each lab consists of an overview document in PDF, and a varying number of modules. For each of the modules there is an assignment document, a folder with input files (configuration files, templates, ...) and another folder with the results, for reference for the teacher.

Different modules can be combined into clusters. Each cluster of modules is coping with the teaching of specific skills.

## Overview of the different labs and modules

No	Title	Level	Duration	Cluster
<b>Lab 1</b>	<b>10 ways to model a rising edge detector in VHDL</b>	1	2.0 h	1
<b>Lab 2</b>	<b>Resource sharing between 2 processors</b>	2	2.0 h	10
<b>Lab 4</b>	<b>Design of a microprocessor</b>	1	0.5 h	
Mod 1a	VHDL basics – microprocessor specifications	1	0.33 h	1
Mod 1b	VHDL basics – registers	1	0.25 h	1
Mod 1c	VHDL basics – counters	1	0.5 h	1
Mod 1d	VHDL basics – multiplexers	1	0.5 h	1
Mod 1e	VHDL basics – register file	2	0.75 h	1
Mod 2a	Design of the ALU	3	1.5 h	1
Mod 2b	Testbench ALU	3	1.5 h	1
Mod 3a	Design of the memory unit	2	0.5 h	2
Mod 3b	Design of the memory traffic controller	3	1.0 h	1
Mod 3c	Design of the stack	3	1.5 h	1
Mod 4a	Design of the decode unit	3	2.0 h	1
Mod 4b	Design of the decode-execute unit	3	1.5 h	1
Mod 4c	Design of the fetch unit	2	2.0 h	1
Mod 5a	Design of the CPU	2	0.5 h	1
Mod 5b	Building a basic system	2	0.5 h	1
Mod 5c	Transfer assembly code into ram contents	2	1.0 h	1
Mod 5d	Simulation of the basic HW-SW system	2	1.0 h	1
Mod 6a	Build and simulate a complete system	2	2.0 h	1
Mod 7a	Implement and verify the complete system	2	1.0 h	2
<b>Lab 6</b>	<b>Driving a VGA screen</b>			
<i>Phase 1</i>	<i>Building the hardware controller</i>			
Mod 1a	Video timing hardware	2	2.0 h	1
Mod 1b	Reusable Testprocedures	3	3.0 h	1
Mod 1c	Video timing simulation	4	4.0 h	1
Mod 2a	Implementation of a video timing generator	1	1.5 h	2
Mod 3a	Implementation of a moving block	4	10.0 h	1,2
<i>Phase 2</i>	<i>Convert hardware controller to an IP block on the OPB</i>			
Mod 4a	Generation of a dual port RAM	1	0.5 h	2
Mod 4b	Driving the screen from a dual port RAM	4	5.0 h	1,2
Mod 5a	OPB-interface	3	6.0 h	1
Mod 6a	VGA-IP	3	2.5 h	1,8
<i>Phase 3</i>	<i>Integrating the IP block in a HW/SW system</i>			
Mod 7a	Integration of the VGA-IP	1	1.0 h	2,8
<i>Phase 4</i>	<i>Optimization of the software and hardware</i>			
Mod 7b	Software driver for driving a VGA screen	3	4.0 h	4
Mod 7c	Software optimization	2	2.0 h	4
Mod 7d	Hardware-Software communication	3	2.0 h	8

<b>Lab 7</b>	<b>Hardware Design with GPIO and UART</b>			2
Mod 1	Using the Xilinx Platform Studio	1	2.0 h	2
Mod 2	Add a reset unit and write a print function	2	1.0 h	2
<b>Lab 8</b>	<b>PowerPC interrupts</b>	3	2.0 h	2
<b>Lab 11</b>	<b>Serial communication – XMODEM</b>	3	4.0 h	4
<b>Lab 14</b>	<b>Implementation of a gray-code counter</b>			
Mod 1	4-bit gray counter	1	0.5 h	1
Mod 2	N-bit gray counter	2	2.0 h	1
Mod 3	Driving the gray counter	1	1.0 h	1
Mod 4	Implementation of a 4-bit gray counter	1	1.0 h	1
<b>Lab 15</b>	<b>Sound effects – ECHO</b>	3	2.0 h	4 + 6
<b>Lab 16</b>	<b>Software implementation of an MP3 player</b>			
Mod 1	Building the mp3-player hardware system	1	0.5 h	2
Mod 2	Reading a file from compact flash	3	2.0 h	4
Mod 3	Reading data from compact flash to AC97	3	2.0 h	8
Mod 4	MP3 decoder	2	2.0 h	8
<b>Lab 20</b>	<b>RTOS basic concepts</b>			9
Mod 1	Threads	2	2.0 h	9
Mod 2	Mutexes	2	2.0 h	9
Mod 3	Semaphores	2	2.0 h	9
Mod 4	Message Queues	2	2.0 h	9
Mod 5	Interrupts	2	2.0 h	9
<b>Lab 21</b>	<b>Introduction to the use of ChipScope™ Pro in EDK</b>	3	2.0 h	2
<b>Lab 22</b>	<b>Use of the EDK Base System Builder</b>	1	1.0 h	2
<b>Lab 23</b>	<b>Hardware implementation of “Simon says”</b>			1
Mod 1	Basic VHDL blocks	1	0.5 h	1
Mod 2	Using a FIFO	3	1.0 h	1
Mod 3	Pulse Generator	1	0.5 h	1
Mod 4	Using Buttons	1	0.5 h	1
Mod 5	Led Driver and Control Block	2	1.5 h	1
Mod 6	Implementation	2	1.0 h	1
<b>Lab 24</b>	<b>Development of PPC system including various IP</b>			2
Mod 1	Simple hardware design	1	1.0 h	2
Mod 2	Adding IP to a hardware design	1	1.0 h	2
Mod 3	Creating and adding custom IP	3	2.5 h	2
Mod 4	Utilizing timers and interrupts	3	2.5 h	2

# Clusters

<b>Cluster 1</b>		<b>Understanding VHDL</b>
Lab 1		10 ways to model a rising edge detector in VHDL
Lab 4		Design of a simple microprocessor system
Lab 6	Mod 1-6	Driving an SVGA screen
Lab 14		Implementation of a gray-code counter
Lab 23		Hardware Implementation of "Simon says"
<b>Cluster 2</b>		<b>Using the EDK and ISE™ Foundation™ tools set</b>
Lab 4	Mod 3a, 7a	Design of a simple microprocessor system
Lab 6	Mod 2,3,4,7	Driving an SVGA screen
Lab 7	Mod 1,2	Hardware Design with GPIO and UART
Lab 8		PowerPC interrupts
Lab 16	Mod 1	Software implementation of an MP3 player
Lab 21		Introduction to the use of ChipScope Pro in EDK
Lab 22		Use of the EDK Base System Builder
Lab 24		Development of PPC system including various IP
<b>Cluster 4</b>		<b>Software</b>
Lab 6	Mod 7	Driving an SVGA screen
Lab 11		File-transfer via the hyperterminal of the PC to external memory on the XUP board, using the XMODEM protocol.
Lab 16	Mod 2,3,4	Software implementation of an MP3 player
<b>Cluster 6</b>		<b>DSP</b>
Lab 15		Sound effects -ECHO
<b>Cluster 8</b>		<b>Designing HW/SW systems</b>
Lab 6	Mod 6,7	Driving an SVGA screen
Lab 16	Mod 3-4	Software implementation of an MP3 player
<b>Cluster 9</b>		<b>Real-time Operating Systems</b>
Lab 20		RTOS basic concepts
<b>Cluster 10</b>		<b>Multi-processor SoC</b>
Lab 2		Resource sharing between 2 processors

## Labs listed by engineering curriculum

<b>Digital Logic (VHDL design with ISE Foundation)</b>			
<b>Getting started with VHDL</b>			
Lab 1			Design and simulate 10 versions of rising edge detector
<b>Basic digital logic design concepts</b>	<b>1 design concepts</b>		
Lab 4	Mod 1a..7a		Design of a simple microprocessor system
<b>Digital Applications</b>			
Lab 6	Mod 1a..4b		Building a Video Output controller design to drive an SVGA screen
Lab 14	Mod 1..4		Implementation and simulation of a gray-code counter
Lab 23	Mod 1..6		Hardware Implementation of the game "Simon says"
<b>Embedded Systems (Hardware Design with EDK)</b>			
<b>Create an embedded hardware system</b>			
Lab 22			Use of the EDK Base System Builder
Lab 24	Mod 1,2		Development of PPC system including various IP
Lab 7	Mod 1		Build a system with GPIO and Uart
<b>Create and Add your own custom peripheral (IP)</b>			
Lab 6	Mod 5a .. 7a		Add a bus interface to the Video Output controller and integrate the IP block in a HW/SW system
Lab 24	Mod 3,4		Development of PPC system including various IP
<b>Hardware Verification</b>			
Lab 6	Mod 7d		Simulate the hardware system with Modelsim®
Lab 21			Perform on-chip verification with ChipScope
<b>Embedded Systems (Software Development with EDK)</b>			
<b>Hardware/Software communication</b>			
Lab 6	Mod 7a .. 7d		Driving an SVGA screen
Lab 7	Mod 2		Add a reset unit and write a print function
<b>Interrupt Service Routines</b>			
Lab 8			Display switch settings on LEDs
<b>Software Verification</b>			
Lab 6	Mod 7b		Use the GNU Debugger to verify SVGA screen driver
<b>Embedded Systems (Applications)</b>			
<b>MP3 System</b>			
Lab 16	Mod 1		Generate the system hardware using Base System Builder
Lab 16	Mod 2		Reading the Compact Flash
Lab 16	Mod 3		Reading/writing samples from the audio CODEC
Lab 16	Mod 4		Implementing an MP3 decoder
<b>Echo System</b>			
Lab 15	Mod 1		Create an embedded system with audio interface and buffer for audio samples
Lab 15	Mod 2		Create software application to add echo to audio samples
<b>File transfer from PC to external memory on the XUP board</b>			
Lab 11			Implementation of the XMODEM protocol

<b>Advanced Embedded Topics</b>			
	<b>Real-Time Operating Systems( Basic Concepts)</b>		
	Lab 20	Mod 1	Threads
	Lab 20	Mod 2	Mutexes
	Lab 20	Mod 3	Semaphores
	Lab 20	Mod 4	Message Queues
	Lab 20	Mod 5	Interrupts
	<b>Multi-processor SoC</b>		
	Lab 2		Resource sharing between 2 processors