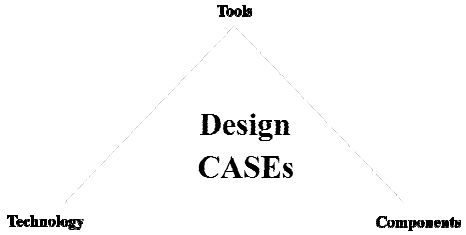
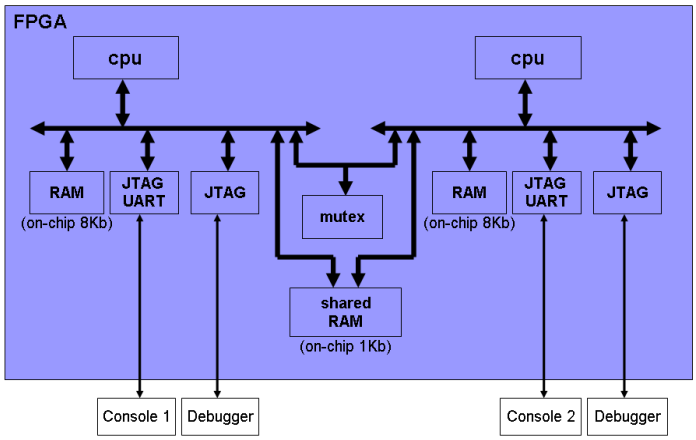
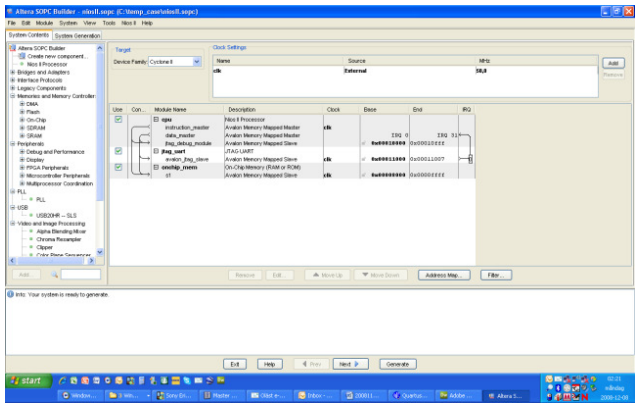


Lennart Lindh and Tommy Klevin

HW/SW Embedded System Design with FPGA Technology, 2009

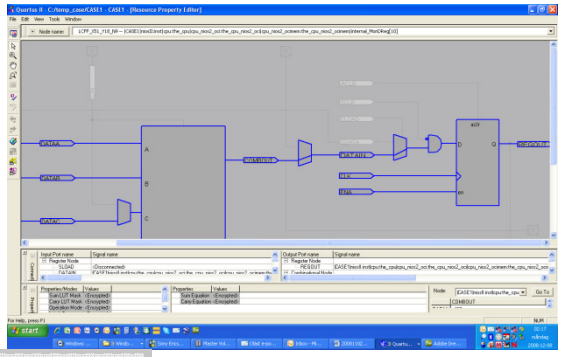
- Engineering approach -





Highlights:

- Flexible hardware and software
- HW/SW Component Design
- SW Device driver
- Multiprocessor System
- Hardware based Real-time Kernel
- Documentation
- Learn by doing
- Configurable Computing



Use complex components (CPU, busses etc) and simulation, design and debugging SW/HW PC tools for practical training.

Lennart Lindh and Tommy Klevin

HW/SW Embedded System Design
with FPGA Technology

Version 1, Swedish, 2005 programmerbara kretsar – Utveckling av inbyggda system, ISBN 91-44-03713-9, www.studentlitteratur.se

Version 2, 2008 HW/SW Embedded System Design with FPGA Technology, www.agstu.com.

- Translated version 1 to English and new practical training.

Version 3, 2009 HW/SW Embedded System Design with FPGA Technology, ISBN 978-91-977667-0-8, www.agstu.com.

- Updated the methodology
- Updated bus sections and component design
- Updated Sierra CASE and new RTK CASEs
- Updated device driver code
- Add new CASE, verifying the Timer CASE with ModelSim
- Mention Reconfigurable Computing, this will be more described in the next version.

Dragverksg 138, S-724 74 Västerås, Sweden. Mail: publisher@agstu.com. www.agstu.com

Disclaim
All the information (including hardware, software, designs, text and files) are provided "as is" and without any warranties expressed or implied, including but not limited to implied warranties of merchantability and fitness for a particular purpose. In no event should the author be liable for any damages whatsoever (including without limitation, damages for loss of business profits, business interruption, loss of business information, or any other pecuniary loss) arising out of the use or inability to use information (including text, software, designs and files) provided in this book.

ISBN 978-91-977667-0-8

Preface

The aim of the book is to prepare you for real SW/HW embedded system design with FPGA technology, by walking step by step through entire designs together with adequate theory.

The software tools used in this book are professional and free to download. The educational FPGA board costs around 265 dollars, you can by a simpler board for 125 (DE1) or 79 dollar (DE0), at www.terasic.com . We hope you will enjoy our guided journey in this book.

The FPGA technology has a huge impact on today's development process, and also on how we conduct science and engineering. One of the limitations today is the insufficient knowledge of how to fully exploit FPGA technology. The design space for FPGA devices is large and fascinating with efficiencies that vary by orders of magnitude. A good engineer and researcher must know the fundamental tradeoffs to be able to re-evaluate solutions as the underlying technology changes, like FPGA technology. We will certainly see a dramatic change when FPGA technology will be further explored in the development and research process in the near future.

Many different areas of knowledge are merged together in the design process of embedded system on chip solutions. Subsequently, those who design on a high level of abstraction must have a very broad competence.

Organization of the book

The area described in the book spans over a wide range of knowledge, stretching from technology, operating systems, computer architecture, methods and different tools.

The book is divided into background, introduction, FPGA technology, tools and components and CASE (practical training). The theory is sometimes integrated into the CASEs, to get as short a distance as possible between practice and theory. The purpose of the CASE is "learn by doing".

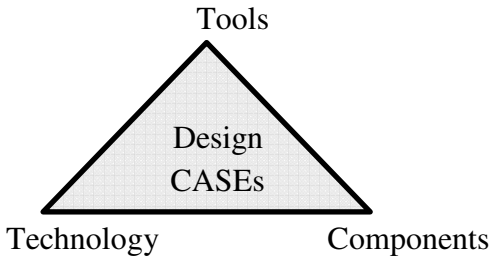


Figure 1: The three key areas and the implementation in the CASEs

The book concentrates on technologies, tools and reusable components for FPGA. Those three key objects are then analyzed and later used in an engineering way to guide you through theory and practical CASE studies.

Technology is the physical FPGA chip, consisting of the fundamental programmable small building blocks.

Components are reusable SW/HW components, which can be assembled like Lego bricks (usually called IP components, Intellectual Property).

Tools are the tools that make the construction work easier.

CASEs are examples of real designs.

Acknowledgement

We thank the following persons for small or larger contributions, comments and/or inspiration (and our apologies to those not mentioned here):

Graham McKenzie, Birger Egnér, Johan Furunäs, Susanna Nordström, Andreas Löfgren, Erik Käll, Jens Lindh, Kerstin Åberg, Johan Stärner, Katarina Myrehed, Farshid Atachi, Lars Johansson, Anders Karlström, Amir Sejdinovic, Tobias Skoglund, Björn Thor, Mathias Vall, Peter Westling, Johan Bard, Lukas Knapik, Trond Rogne Øyre, David Johansson, Jakob Lindgren, Johan Palm, Sigurd_Bårdsen, Anders Nesheim Vinje, Mats Karlsson, Gustaf Naeser, Martin Næsse, and last but not least all the hundreds of students we have had. The book could not have been made without all these inspiring people!

The book has its own web page: see www.agstu.com, there can you perhaps find extra material.

Table of Contents

1 BACKGROUND AND DEFINITIONS 8

1.1 DEFINITIONS 8
1.1.1 System and attribute 8
1.1.2 Embedded computer system 9
1.1.3 Real-time embedded system 10
1.1.3.1 Examples of real-time attributes: 11
1.1.4 FPGA and ASIC 11
1.1.5 Terms 12
1.2 ELECTRONICS- AND SEMICONDUCTOR HISTORY 14
1.3 HARDWARE VERSUS SOFTWARE AND CO-DESIGN 16
1.4 APPLICATION AND PLATFORM 18
1.4.1 Comparison between ASIC, One-chip computer and FPGA 20
1.4.2 Design space SW/HW platform 21
1.5 COMPLEXITY REDUCTION 21

2 INTRODUCTION; FPGA, COMPONENTS AND TOOLS 25

2.1 FPGA TECHNOLOGY 25
2.2 REUSABLE COMPONENTS 27
2.2.1 Architecture 28
2.2.2 Hardware components 29
2.2.2.1 IP – Intellectual Property components 31
2.2.2.2 Hardness classes for electronic IP 32
2.2.3 Software components 34
2.2.4 Buses – standard connection between HW components 35
2.2.4.1 Memories and memory map 37
2.2.4.2 A bus example – AMBA 37
2.2.5 Interface between hardware and software 38
2.2.5.1 Device drivers and libraries 41
2.2.6 Move functions from software to hardware 48
2.2.7 Accelerators in hardware 49
2.3 DEVELOPMENT TOOLS 56
2.3.1 Hardware development tools 56
2.3.2 Software Development tools 56

3 CASES 59

3.1 CASE ONE A - SIMPLE COMPUTER PLATFORM - 59
3.1.1 The tools and simple methods 60
3.1.2 Presentation of the board and components 61
3.1.3 Theory "Components" 61
3.1.3.1 Theory " FPGA technology" 62
3.1.3.2 Theory "CPU Nios II" 71
3.1.3.3 Theory "Avalon Bus" 74
3.1.3.4 Theory "JTAG UART" 76
3.1.4 Designing Case one A 80
3.1.4.1 Create a new project in Quartus II 81
3.1.4.2 Add the Nios II System (subsystem) 86
3.1.4.3 Map the Pins on the FPGA 87
3.1.4.4 Verify the result and generate a bit file 88
3.1.4.5 Configuration of the FPGA 89
3.1.4.6 Construction of the Software 91
3.2 CASE ONE B – DEBUGGING APPLICATION SOFTWARE 94
3.2.1 Theory "Debugging, monitoring and simulation" 94
3.2.1.1 Traditional software debugging 95
3.2.1.2 Other Debugging tools 100
3.2.1.3 Hardware/Software Co-Simulation of Embedded Systems Abstract 106
3.2.2 Design Case one B 115
3.3 CASE TWO: ADDING AN EXTERNAL RAM TO THE SYSTEM 117
3.3.1 Theory "Memory" 117
3.3.1.1 RAM 117
3.3.1.2 Types of RAM 117

3.3.1.3 VHDL syntax for RAM and ROM 119
3.3.2 Design Case two *122*
3.3.2.1 Facts and background 122
3.3.2.2 Hardware Design 123
3.3.2.3 Software Design 124
3.4 CASE THREE: DESIGN A TIMER IP COMPONENT 126
3.4.1 Specification *126*
3.4.2 Define all interfaces *128*
3.4.2.1 Driver definitions (specification) 128
3.4.2.2 Register Definition (specification) 128
3.4.3 Theory "HAL (Hardware Abstraction Layer)" *130*
3.4.4 Design CASE 3, Design process for SW driver *131*
3.4.5 Design CASE 3 "Bus interface design" *132*
3.4.5.1 Extra Component Verification CASE 3 with ModelSim 136
3.4.5.2 Integrate the component into the SOPC builder 139
3.4.5.3 Use the timer and reflections 142
3.4.6 Theory Time measurement techniques *143*
3.5 CASE FOUR: USE A PUSHBUTTON TO CONTROL 4 LEDS 145
3.5.1 Design CASE 4: Polling system *146*
3.5.1.1 Adding two new PIOs to the system 146
3.5.1.2 Software development; polling the button 147
3.5.2 Theory "Interrupts" *149*
3.5.2.1 Timing model of interrupt 149
3.5.2.2 Interrupts in the Nios II processor 150
3.5.2.3 Interrupts and the Hardware Abstraction Layer 150
3.5.3 Design CASE 4 B: Interrupt system *151*
3.6 CASE FIVE: MOVE SOFTWARE TO HARDWARE TO REDUCE RESPONSE TIME 153
3.6.1 Design CASE 5: SW 2 HW *153*
3.6.1.1 Redesigning the hardware 153
3.6.1.2 Software Design 157
3.7 CASE SIX, MULTIPROCESSOR SYSTEM 158
3.7.1 Theory "Multiprocessor" *159*
3.7.1.1 Bus Arbiter 162
3.7.1.2 Hardware mutex 162
3.7.1.3 Dual-port on chip memory 163
3.7.1.4 Processing Architectures 163
3.7.1.5 Software Languages 166
3.7.2 Design CASE 6: Hardware *167*
3.7.2.1 Add components to your system 167
3.7.3 Design CASE 6: Software *167*
3.7.3.1 Create projects for each CPU 170
3.7.3.2 Setting System Library Properties 171
3.7.3.3 Add application software files 172
3.7.3.4 Insert the code 172
3.7.3.5 Build the software projects 172
3.7.3.6 Create a "multiprocessor collection" 173
3.7.3.7 Running your software 174
3.7.3.8 Debugging your software 175
3.8 CASE SEVEN: REAL-TIME KERNEL 177
3.8.1 Theory "Real-time kernel –connection between SW components" *179*
3.8.1.1 Scheduling tasks 182
3.8.1.2 Communication 190
3.8.1.3 Synchronization 196
3.8.2 Theory "Operating system accelerator in hardware" *200*
3.8.2.1 General description of operating systems and tasks 201
3.8.2.2 Description of a hardware operating system 203
3.8.2.3 Interrupt handling 204
3.8.2.4 Hardware operating system in practice 205
3.8.2.5 Handshake protocol 210
3.8.2.6 Service calls 210
3.8.2.7 Task switch interrupts 211
3.8.2.8 Interface timing 213
3.8.2.9 Increasing performance 215
3.8.3 Design CASE 7, Hardware design: Implement Sierra *216*

3.8.4 Design CASE 7, Software design *217*
3.8.4.1 Design CASE 7- A 217
3.8.4.2 Design CASE 7- B 221
3.8.4.3 Design CASE 7- C 224
3.8.5 Software design inspiration examples *226*
3.9 CASE – DOCUMENTATION 228
3.9.1 Theory "Project methodology" *228*
3.9.2 CASE - Project report for component design *236*
3.9.3 CASE - Review of the documentation *240*

4 APPENDIX 242

4.1 APPENDIX; AN INTRODUCTION TO C 242
4.2 APPENDIX; VIDEO GRAPHICS ARRAY 248
4.3 APPENDIX; AMERICAN STANDARD CODE FOR INFORMATION INTERCHANGE (ASCII) 256

5 INDEX 258

Lennart Lindh works at the university and in the industry. For many years he has been engaged in research and development of products with FPGA technology. His most well-known result is real-time kernels for single- and multiprocessor systems in hardware. He has written several books about FPGA, HW/SW systems, VHDL and real-time systems.

Tommy Klevin is employed in the industry at present. He has previously done academic work as well and has, in cooperation with Lennart Lindh and a team of researchers, worked a lot with issues concerning performance in real-time systems, and a large part of this focused on moving parts of operating systems from software to hardware (ASIC/FPGA). Everything from a small real-time OS up to Linux has been part of the research work.

HW/SW Embedded System Design with FPGA Technology

The motivation for reading this book is the increase of implementations of FPGA (Field Programmable Application Gate Arrays) devices in embedded products. The aim of the book is to prepare you for real embedded FPGA system projects by walking step by step through an entire design of an embedded system together with adequate theory.

The FPGA technology has a huge impact on the development process, and also on how we conduct science and engineering. One of the limitations today is the insufficient knowledge of how to fully exploit the FPGA technology. The design space for FPGA devices is large and fascinating with efficiencies that vary by orders of magnitude. Further, the limitations of our technology are changing, so the "right answers" of the past will almost certainly become outdated. A good engineer and researcher must know the fundamental tradeoffs to be able to re-evaluate solutions as the underlying technology changes, like FPGA technology. We will certainly see a change in the development process and research topics in the near future in the area of embedded systems.

Many questions arise, such as: how and where do we draw the boundary between hardware and software? How do we organize a component design paradigm with enormous power of parallel processing? What is most important to optimize when the price is close to zero for one gate?

The book concentrates on **technologies, tools** and **reusable components** for FPGA. Those three key objects are then analyzed and later used in an engineering way to guide you through theory and practical CASE studies.

FPGA technology is relatively new and very important in many applications, in terms of development cost, performance, power consumption etc. The well-known friend to FPGA technology is “Moore’s law”, which still predicts that the number of gates will increase and the cost decrease.

Tools are important for optimizing the development process; archiving short development time, cost optimal solutions, minimizing the risks of bugs etc.

Reusable components have the purpose to decrease the risks, shorten development time etc.

The book is divided into background, introduction and CASE (practical training). The theory is sometimes integrated in the CASEs, to get a short distance between practice and theory. The CASE purpose is to “learn by doing”.

The book can be order on www.agstu.com

AGSTU AB, Dragverksgatan 138, S-724 74 Västerås, Sweden,

ISBN 978-91-977667-0-8