Graduate course on FPGA design

Andreas Ehliar, Oscar Gustafsson

February 15, 2010

Course planning

- Lectures
- Laboration
  - 4 HP for actively attending lectures and doing individual laboration
- Project
  - 2-6 HP (4 nominally) for design project

Lectures

No.	Topic	Responsible
1	Introduction and more	Oscar and Andreas
2	Logic blocks and memory	Andreas
3	DSP blocks	Oscar
4	I/O and clocking	Andreas
5	Routing and architectures	Oscar
6	Processors and partial reconfiguration	Andreas
7	IP blocks and high-level design	Kent
8	Benchmarking and projects	Andreas and Oscar

Laboration

- Run complete design flow including power estimation
- Design using an FFT and a FIR decimation filter
- Optimize the designs for FPGAs
- Written report on the optimization and comparison with initial results

Project

- Project preferrably related to (your) research
- Nominally individual and 4 HP (can be discussed)
- Written report (plus paper?) and oral presentation
- Decide on deadline(s): April 30, May 28
- Reviewer/opponent for one project
- All M.Sc. students should have the same number of credits

FPGA

- A Field-Programmable Gate Array (FPGA) is an integrated circuit designed to be configured by the customer or designer after manufacturing
- Easy to program and evaluate – short turn-around time
- Flexible – Can change during run-time
- High(?) performance – parallel, many different sizes available
- Often cost-effective

FPGA design starts

Use cases

- Glue logic
- DSP
- Reconfigurable computing
- ASIC verification
- High-performance computing
- Evolutionary computing

Glue logic

- "Glue logic" is a common name for smaller parts of logic required to connect more complex circuits
- Interfacing
  - Generating control signals
  - Reformatting/retiming data
- Simple control

DSP

- Many FPGAs have a large set of multipliers
- Can be used to map complex DSP algorithms
- High degree of parallelism

DSP

- Spectrometers for radio astronomy
  - As high sample rate as the ADCs allow
  - As many points as the FFT allows
  - Currently about 5 GSa/s and 32k points

DSP

- Radar signal processing
- Cable TV modulators
-

DSP

- Software Defined Radio
  - Multi-standard radio
  - Reconfigure the FPGA

Reconfigurable computing

- Reconfigurable computing is computing where not only the software but also the hardware can be adapted
- "Flexibility of software, performance of hardware"
- Run-time reconfiguration – adapt hardware when running certain parts of the program

ASIC emulation

- FPGAs are commonly used to verify ASIC designs
- "Real-time" operation
- Up to 30 million gates

ASIC emulation

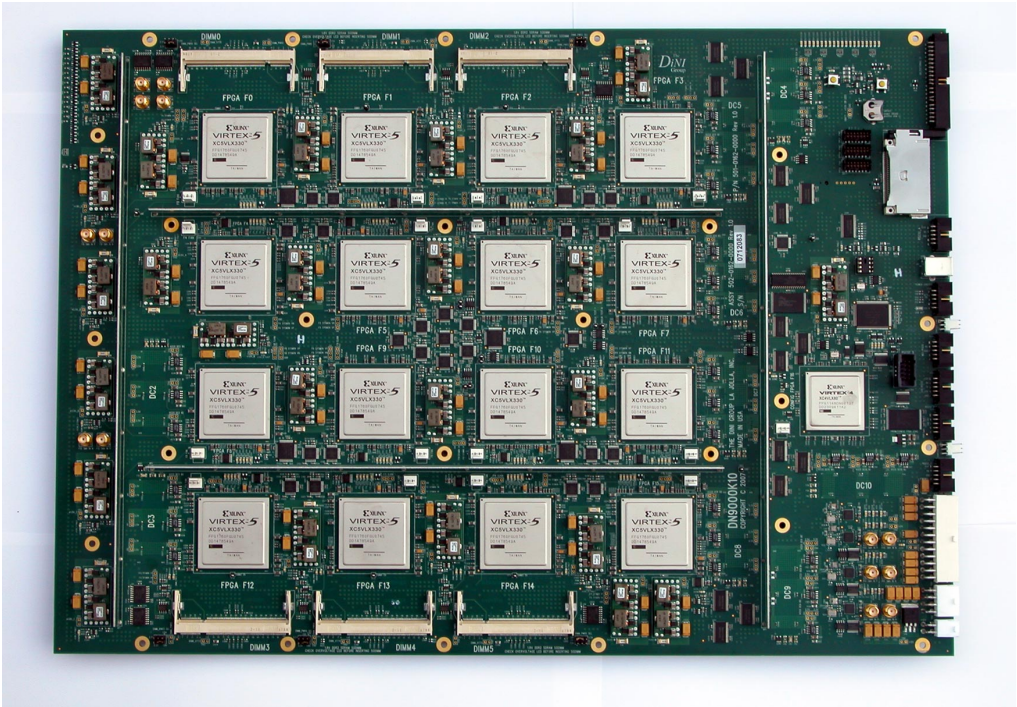


Photo from Dini Group

ASIC emulation

- AMD was claimed to have complete FPGA versions of the processors
- Intel Pentiums available in an FPGA – used for testing accelerators

High-performance computing

- Customized architectures
- High-degree of parallelism
- Many arithmetic units
- Adaptive accuracy/number representation
- Stand-alone or cards for direct connection over a bus

High-performance computing

- Financial simulation
  - Simulate different stock/bond setups
  - Monte-Carlo type simulations
  - Very computationally intensive as many runs are required
  - Automatic generation of FPGA code from modelling languague
  - Reductions shown 60 to 100 times over software (4 CPUs)
  - In practice from several weeks to a few hours

High-performance computing

- Seismic simulation
  - Shown four to seven times speed-up
  - 48 times possible with faster I/O

High-performance computing

- SAT/optimization solvers
  - Generate parallel constraint evaluation
  - 17 times MiniSAT
  - Up to 10k variables and 280k clauses

High-performance computing

- Silicon Graphics have "supercomputers" with FPGA accelerators built in
- Scottish supercomputer Maxwell is based on 32 Xeons and 64 Virtex4 FPGAs

Programmable devices/logic

- Memories like PROMs, EPROMs and even SRAMs can be used to emulate logic

Gate-arrays

- Fairchild 4500
- 32 gates
- Programmed using two layers of metal

FPLA - Signetics 82S100

- Introduced in 1975
- 16 inputs
- Programmable AND plane (48 product terms)
- Programmable OR plane (8 outputs)
- No flip-flops

PLA, PAL, PLD, etc

- Typical example: PAL16R8
- 8 inputs, 8 outputs (with registers)
- Feedback possible from the 8 registers
- Programmable AND plane with 16 inputs, fixed or plane

CPLD

- Basically many PLDs put together in one chip
- Some sort of internal routing network is used to interconnect the different "PLDs"
- Example I'm very familiar with: XC9572
  - 72 macrocells divided into four groups of "PLDs"
  - One macrocell can be either D flip-flop, T flip-flop or combinational
  - Programmable via JTAG
  - NOTE: Today it may be difficult to tell a CPLD from an FPGA

XC2064

- First successful FPGA
- Released in 1985
- 64 Complex logic block (CLB)
  - Two 3-input LUTs or one 4-input LUT
  - One flip-flop
- Configuration memory: 12kbit
- Not really any special features
- Hardware design using schematic entry

XC4000 series

- Released in 1991
- Spawned a wide family of parts
- Original XC4000 contained 100 to 1024 CLBs
  - Two 4 input LUTs
  - Two flip-flops
  - LUTs can be used as 16x1 memory to improve memory densities!
  - Carry-chain to speed up adders

XC6200 series

- Dynamically reconfigurable
- Documented configuration bitstream
- Beloved by academia, for example in evolvable hardware
- Scorned by industry

Virtex

- Built in memory blocks (512 byte each)
- (One block RAM can contain as much memory as 256 LUTs)

Virtex-II

- Built-in multiplier blocks (18x18)
- One LUT can be used as a 16 bit shift register

Virtex-II Pro

- Built in processor (PPC 405)
- Built in high speed SERDES blocks

Virtex-4

- Multiply-accumulate combined into DSP blocks

Virtex-5

- Routing delay is getting larger compared to logic delay
- Solution: 6-input LUTs instead of 4-input LUTs

Current status

- Major suppliers:
  - Xilinx
  - Altera
- Various other established vendors:
  - Actel
  - Lattice
- Newcomers:
  - SiliconBlue (Optimized for power)
  - Aboundlogic (Aim for extremely large devices)
  - Achronix (Asynchronous FPGAs optimized for throughput)

Different configuration technologies

- SRAM
- Flash
- Flash + SRAM
- Antifuse (One-time programmable)

Other architectures of note

- eASIC
  - Cross between gate array and SRAM based FPGA
  - SRAM programmed LUT to configure logic
  - Routing is programmed using custom masks or via FIB

Other architectures of note

- ORGA - Optically Reconfigurable Gate Array
  - Instead of using SRAM cells for configuration bits, a photodiode is used.
- LPGA
  - All possible connections are connected by default
  - A laser is used to disable all unneeded connections
  - Fast turn-around prototyping with gate-array performance

Quick introduction to all parts of the flow

- You will learn more during the lab about this

Synthesis using xst

- INPUT: file.v, file.vhd
- OUTPUT: design.ngc
- Read in VHDL/Verilog code
- Outputs FPGA primitives in a netlist

ngdbuild

- INPUT: design.ngc, design.ucf
- OUTPUT: design.ngd
- Works like a linker

map

- INPUT: design.ngd
- OUTPUT: design.ncd design.pcf
- Tries to map FPGA primitives into slices in a clever fashion
- May also place components

par (Place and route)

- INPUT: design.ncd, design.pcf
- OUTPUT: routeddesign.ncd
- Places the components (if map hasn't done that)
- Routes all connections (if possible)
- Performs static timing analysis

bitgen

- INPUT: design.ncd
- OUTPUT: design.bit
- Converts the netlist into a configuration bitstream

Constraints:

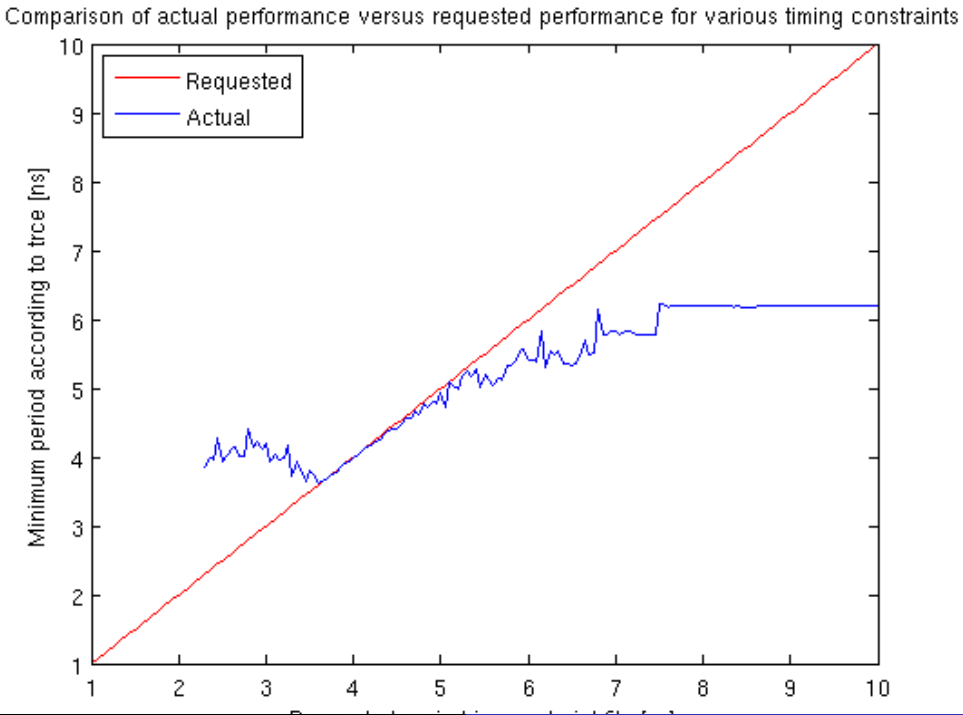
- You will need a way to tell the CAD flow what your design constraints are such as:
  - Clock frequency requirements
  - I/O placement
  - Placement of individual LUTs, BlockRAMs, etc...
  - This is done in a UCF file (User Constraint File)

Advanced and esoteric features

- So far there should have been few surprises I hope

Getting the maximum clock frequency from a design

- You may have to vary your timing constraints and optimization settings. An element of luck may be required as well.



Other synthesis tools

- Precision
- Synplify
- Little market for other backend tools though

trce: Static Timing Analysis

- May not be needed since the router does this as well
- Useful if you cannot meet timing since par will not tell you the failing paths

Back annotation

- You can create a netlist usable for simulation by using netgen
- Useful if you are suspecting bugs in the synthesis tool
- You can also simulate based on the routed netlist which will include timing information

Power Simulation

- For best results
  - Create a VCD file based on a simulation of a post place and route netlist
  - Load this together with your design into XPower
- You can also get a rough estimate without a VCD file
- The VCD file can also be used to optimize the design for reduced power consumption

Modifying the content of BlockRAMs

- You may have to update BlockRAMs much more often than the rest of the design
- First way: Modify content, rerun entire flow
  - Drawback: May be extremely slow!
- Better way: data2mem
  - Define a .BMM-file that tells the flow how to handle your memories
  - Run data2mem to update only the memories in an already routed design

Smartguide

- For a large design, map, place, and route may take hours
- Using smartguide you can reuse as much placement as possible from a previous iteration for much shorter runtimes

Floorplanning

- You can inspect the placement
- You can modify the placement to improve the timing

Fpga Editor

- You can inspect and modify routing
- You can inspect and modify logic
- You can inspect and modify placement
- Not so trivial to use though...

XDL

- Ever wanted to write your own CAD tools for the FPGA flow?
- XDL is an ascii description of the NCD netlists
- With XDL you can write a tool that can do almost anything:
  - Inspect a design
  - Modify a design
  - Write your own placer
  - Write your own router
  - etc...

What do you want to get out of this course?

- Optimize a design for FPGA primitives like LUTs, memories, and DSP blocks
- Investigate and improve the power consumption of FPGA designs
- Learn how to deal with clock generators and multiple global clocks
- Floorplan FPGA designs
- Design your own FPGA boards
- Write your own CAD tools for FPGA design
- Design your own FPGA