

FPGA introduction

2008

eCos is a registered trademark of eCosCentric Limited

Øyvind Harboe, General Manager, Zylin AS



What is an FPGA?

- Field Programmable Gate Array
- Not necessarily reprogrammable (anti-fuse, encryption)
- Contains programmable logic
- A “software PCB”



Why?

- Improve performance
- Some tasks can not be solved in software
- Some interfaces/protocols not supported/possible in hard CPU's
- Replace glue logic
- Reduce number of PCB spins
- Reduce number of parts on PCB



What's inside an FPGA?

- Logic
- Clock resources
- Extremely high bandwidth RAM (10-100x more bandwidth than a PC)
- Multipliers
- High-speed serial interfaces
- Lots of wires that can be connected in any fashion



FPGA vs. Microcontroller

- **FPGA advantages:**
 - Include only peripheral you need/use
 - Implement “any” functionality
 - 10-100x performance for suitable tasks
 - IP cores available
 - Drivers, code, example...
 - Avoid CPU obsolescence
- **Microcontr. advantages:**
 - Complete out-of-the box working solution
 - Datasheet available
 - Drivers, code, examples available
 - Easy to implement difficult algorithms



FPGA vs. ASIC

- **FPGA advantages:**
 - Faster time-to-market
 - No upfront NRE
 - Simpler design cycle
 - More predictable project cycle. (No wafer re-spin)
 - Field reprogrammability (Upgrade product with new firmware)
- **ASIC advantages:**
 - Full custom capabilities
 - Lower unit cost
 - Lower power consumption
 - Smaller form factor
 - Higher performance



How is an FPGA programmed?

- Write a program in HDL (VHDL, Verilog, schematics)
- Simulate
- Synthesis
- Place and route
- Timing – maximum clockrate is determined by tools
- Resulting .bin file to be programmed into the FPGA



Can an FPGA do anything?

- Yes and no
- Like a microprocessor it needs a program: IP
- IP – intellectual property
- That program can be simple or complex
- You can buy parts of that program or write it entirely yourself
- Free IP (www.opencores.org, FPGA vendors)
- FPGAs can be big enough that it can fit many man-years of engineering
- To fill a big FPGA you probably need to get IP somewhere



Challenges in FPGA design

- In software when you are 80% complete, you're 80% done. With FPGAs when your feature set is 80% done you're 20% through the project
- FPGAs are like diving from 10m, you have to get it just right or go up and try all over again
- Unlike software you can't “tweak” FPGA code, you have to get it right
- With software things get a little bit slower and clunkier when you add kludges, with FPGAs things fall apart
- Can be hard to convince management to have the nerve to get things absolutely right before moving on



How do I know I need an FPGA?

- Microprocessors and FPGAs overlap
Are you using a microprocessor?
- You may need an FPGA if:
 - You need very fast interrupts
 - You need many interrupts >1000-10000's interrupts/s
 - Many interrupt sources, that should be served in parallel
 - You have many small operations that need to happen at very precise times
 - Tough (impossible) real time requirements
 - Very large bandwidth requirements



How do I know I need an FPGA?

- Do you have lots of glue logic on your PCB?
 - Replace lots of small components with an FPGA
 - Reduce BOM w/an FPGA
 - Fewer PCB spins



How do I know I need an FPGA?

- Some operations are impossible in a microprocessor
- Have interfaces/protocols that you need to support and are not available on standard CPU's
- You may search in vain to find a chip that is right for you as others are using FPGAs for such applications



Soft CPUs

- An FPGA can implement a CPU ARM7/9 performance
- This is offered by all FPGA vendors
- This is not an attempt to replace microcontrollers but FPGA vendors are happy to take away any part of the BOM they can of course
- Why? The answer is different for everyone who uses it
 - Reduce BOM?
 - Reduce cost?
 - Improve performance?
 - Reduce time to market?
 - Reduce development cost?



Switch to hard CPU?

- What could reasons be to switch to hard CPU?
 - Reduce power
 - Get lots of tested and documented peripherals
 - Reduce usage of FPGA pins, easier to route FPGA
 - Reduce cost
 - Toolchains(GCC) are available from official GCC tree
 - Easier to divide project into software and hardware domains
 - Easier to upgrade to real Linux
 - Faster
 - Much wider choice of parts and suppliers



Switch to soft CPU?

- If you are using a hard CPU, what might reasons be to switch to a soft CPU?
 - Reduce part count
 - Avoid part obsolescence
 - FPGA vendors have tools that can be a good fit
 - Availability of engineering resources and skills?



FPGA applications

- Replace glue logic
- DSP (large bandwidth + multiplications)
- Communication
- Video
- Non-standard interfaces
- Non-standard protocols



How to get started

- Read up on the promises from FPGA vendors
- Create a wish-list
- Get hold of a seasoned FPGA person
- Re-adjust expectations and ambitions something realistic
- It takes years of experience to master the art of developing FPGAs



FPGA introduction

2008

eCos is a registered trademark of eCosCentric Limited

Øyvind Harboe, General Manager, Zylin AS

