FPGA introduction

Øyvind Harboe, General Manager, Zylin AS

What is an FPGA?

- How many have used an FPGA?
- Field Programmable Gate Array
- Not necessarily reprogrammable (anti-fuse, encryption)
- Contains programmable logic
- A “software PCB”

Why?

- Some tasks can only be solved with an FPGA
- Realtime performance in sub us range
- 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. SoC/MCU/DSP

FPGA:

- Implement “any” functionality
- 10-100x performance for suitable tasks
- IP cores available
  - Drivers, code, example...
- Avoid CPU obsolescence

SoC/MCU/DSP:

- Complete out-of-the box working solution
- Datasheet available
- Drivers, code, examples available
- Easy to implement difficult algorithms
- Cost, power

FPGA vs. ASIC

FPGA advantages:

- Faster time-to-market
- No upfront NRE
- Simpler design cycle
- More predictable project cycle. (No wafer re-spin)
- Reprogrammable
- FPGA is winning

ASIC advantages:

- Full custom capabilities
- Lower unit cost
- Lower power consumption
- Smaller form factor
- Higher performance
- Analogue capabilities

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
- 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 from the outset
- 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 (sub us)
  - You need many interrupts >1000-10000's interrupts/s
  - Many interrupt sources, that should be served in parallel
  - 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
- Offered by all FPGA vendors
- This is not an attempt to replace MCUs, but FPGA vendors are happy to take away any part of the BOM they can of course

Why Soft CPU?

- 10000 different MCU's available.
- Resource allocation: is there a **chance** that you can use the same CPU on two projects?
- If any CPU will do, then a soft CPU may be a better choice.
- 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 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

Øyvind Harboe, General Manager, Zylin A