GRLIB Open-Source VHDL IP Library

Jiri Gaisler

Gaisler Research

jiri@gaisler.com

Introduction

- High device density (ASIC & FPGA) has led to a larger number of new SOC designs
- An improved design methodolgy is needed to allow cost-efficient development of complex SOC systems
- For this pupose, Gaisler Research has developed a open-source VHDL IP library for both commercial and space-based applications.
- This presentation will describe the concept of the IP library and provide details on some of its cores, including the LEON3 SPARC processor.

Common SOC design problems

- Merging of 3-party IP cores may cause several problems:
    - Harmonisation of interfaces (on-chip buses, irq ...)
    - Merging of synthesis and simulation scripts
    - Mapping of technology specific cells (RAM, pads)
    - Name space conflicts, CAD tool specific syntax
    - Licensing issues
- Problems for space applications
    - SEU hardening
    - Portability and long-term support

GRLIB design goals

- Efficient and unified SOC design IP library with:
  - Common interfaces
  - Unified synthesis and simulation scripts
  - Target technology independent
  - IP-vendor independent
  - CAD tool independent
  - Open and extensible format
  - (SEU tolerance for space applications)

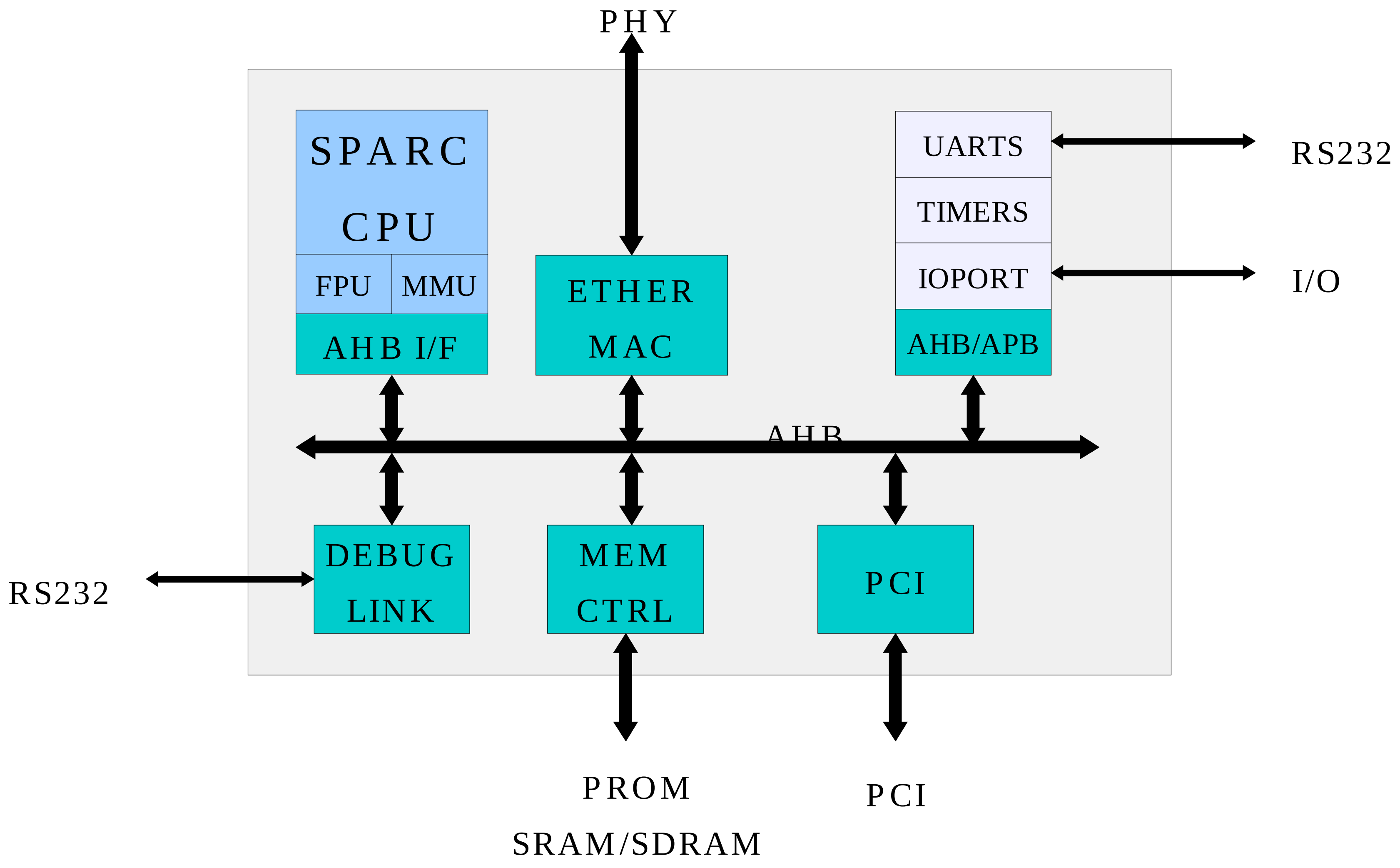
GRLIB implementation overview

- Based around AMBA-2.0 on-chip bus (ARM)
- PCI-style plug&play support for AMBA configuration:
    - Device & vendor identification
    - Address and interrupt configuration
- Vendors and cores isolated through use of VHDL libraries
- Portability achieved through RAM and pad wrappers
- Automatic generation of synthesis and simulation scripts
- Supported tools: Mentor, Cadence, Synopsys, Synplify, ISE
- New cores, CAD tool scripts or tech wrappers easily added

GRLIB IP Cores

- 32-bit LEON3 SPARC processor
- GRFPU IEEE-754 floating-point unit
- 32-bit PCI bridge with FIFO and DMA, PCI trace buffer
- Ethernet 10/100 Mbit Ethernet Controller
- PC133 32-bit SDRAM controller
- 32-bit PROM/SRAM controller
- AHB controller and APB bridge with plug&play support
- Utility cores: uart, timer, interrupt control, GPIO, ...
- Memory and pad wrappers for FPGAs and ASIC

Sample GRLIB SOC design



AMBA plug&play support

- Inspired by PCI plug&play method
- Distributed address decoding
    - => AHB/APB cores can be inserted/removed without modifications to arbiter or address decoder/multiplexer (!)
- Configuration table automatically built and readable from bus
- Address mapping and interrupt assignment through generics
- Pure VHDL implementation, no external SOC tools needed

SOC design VHDL code

```
ahb0 : ahbctrl       -- AHB arbiter/multiplexer
port map (rstn, clkm, ahbmi, ahbmo, ahbsi, ahbso);

u0 : leon3s                -- LEON3 processor
generic map (ahbndx => i, fabtech => FABTECH, memtech => MEMTECH, isetsize => 1, dsetsize => 1)
  port map (clkm, rstn, ahbmi, ahbmo(i), ahbsi, leon3i(i), leon3o(i));

sd0 : mctrl                -- PROM/SRAM/SDRAM memory controller
generic map (ahbndx => 0, apbndx => 0, apbaddr => 0, sden => 1)
port map (rstn, clkm, memi, memo, ahbsi, ahbso(0), apbi, apbo(0), wpo, sdo);
end generate;

apb0 : apbmst                    -- AHB/APB bridge
generic map (ahbndx => 1, memaddr => 16#800#)
port map (rstn, clkm, ahbsi, ahbso(1), apbi, apbo );

uart0 : apbuart            -- UART 1
generic map (apbndx => 1, apbaddr => 1,  irq => 2)
port map (rstn, clkm, apbi, apbo(1), u1i, u1o);

irqctrl0 : apbictrl              -- interrupt controller
generic map (apbndx => 2, apbaddr => 2, ncpu => NCPU)
port map (rstn, clkm, apbi, apbo(2), irqi, irqo);

timer0 : gptimer                 -- timer unit
generic map (apbndx => 3, apbaddr => 3, irq => 8)
port map (rstn, clkm, apbi, apbo(3), gpti, open);

pci0 : pci_target generic map (ahbndx => 1, device_id => 16#0210#, vendor_id => 16#16E3#)
port map (rstn, clkm, pciclk, pcii, pcio, ahbmi, ahbmo(1));

eth0 : eth_oc
generic map (mstndx => 2, slvndx => 5, ioaddr => 16#B00#, irq => 12)
port map ( rst => rstn, clk => clkm, ahbsi => ahbsi, ahbso => ahbso(5),
ahbmi => ahbmi, ahbmo => ahbmo(NCPU+dbg+pci), ethi => ethi, etho => etho);
```

SOC design simulation

```
VSIM 1> run
# LEON3 Demonstration design
# GRLIB Version 1.0
# Target technology: infered,  memory library: infered
# ahbctrl: AHB arbiter/multiplexer rev 1
# ahbctrl: Common I/O area at 0xfff00000, 1 Mbyte
# ahbctrl: Configuration area at 0xfffff000, 4 kbyte
# ahbctrl: mst0: Gaisler Research        Leon3 SPARC V8 Processor
# ahbctrl: slv0: Gaisler Research        PROM/SRAM/SDRAM Controller
# ahbctrl:       memory at 0x00000000, size 16 Mbyte, cacheable, prefetch
# ahbctrl:       memory at 0x40000000, size 16 Mbyte, cacheable, prefetch
# ahbctrl: slv1: Gaisler Research        AHB/APB Bridge
# ahbctrl:       memory at 0x80000000, size 16 Mbyte
# apbmst: APB Bridge at 0x80000000 rev 1
# apbmst: slv1: Gaisler Research        Generic UART
# apbmst:       I/O ports at 0x80000100, size 256 byte
# apbmst: slv2: Gaisler Research        Multi-processor Interrupt Ctrl.
# apbmst:       I/O ports at 0x80000200, size 256 byte
# apbmst: slv3: Gaisler Research        Modular Timer Unit
# apbmst:       I/O ports at 0x80000300, size 256 byte
# apbmst: slv7: Gaisler Research        AHB Debug UART
# apbmst:       I/O ports at 0x80000700, size 256 byte
# eth_oc5: Opencores 10/100 Mbit ethernet MAC, rev 0, irq 12
# gptimer3: GR Timer Unit rev 1, 16-bit scaler, 2 32-bit timers, irq 8
# apbictrl: Multi-processor Interrupt Controller rev 1, #cpu 2
# apbuart1: Generic UART rev 1, irq 3
# ahbuart7: AHB Debug UART rev 0
# leon3_0: LEON3 SPARC V8 processor rev 0
# leon3_0: icache 1*2 kbyte, dcache 1*1 kbyte

# cpu0: 0x00000000    flush  0x0000
# cpu0: 0x00000004    sethi  %hi(0x00001000), %g1  [0x00001000]
# cpu0: 0x00000008    or   %g1, 0x00c0, %g1  [0x000010c0]
# cpu0: 0x0000000c    mov  %g1, %psr
# cpu0: 0x00000010    mov  0, %wim
# cpu0: 0x00000014    mov  0, %tbr
# cpu0: 0x00000018    mov  0, %y
```

LEON3 SPARC V8 Processor

- 7-stage pipeline, multi-processor support
- On-chip debug support unit with instrcution trace buffer
- 250/400 MHz on 0.18/0.13 um, 250/400 MIPS
- Virtex2pro: 125 MHz, Actel RTAX: 33 MHz
- SEU tolerance by design for space applications
- All on-chip ram protected against SEU:
  - 136x32 bit register file: 4-bit parity and duplication
  - Cache rams use 4-bit parity and forced miss on error
  - No timing penalty, < 0.5% area overhead (on RTAX)

LEON3 Advanced floating-point unit

- High-performance single/double precision FPU (GRFPU)
  - IEEE-754, fully pipelined, 4 clock latency
  - ADD/SUB/MUL/DIV/SQRT/COMP/CONV
  - Dual execution units, parallel processor interface
  - Fault-tolerance against SEU effects
- 150/250 MHz, 150/250 MFLOPS on 0.18/0.13 um, 100 Kgates
- 40 MHz on Virtex-II, 9,000 LUTs
- Too large to fit on RTAX devices
- Can be used for DSP designs (custom or processor-based)

GRLIB Master/Target PCI

- Implements PCI 2.1 standard (32-bit, 33 MHz)
- Configurable FIFO depth
- DMA channel for independent block transfers
- 45/75 MHz, 9% area of RTAX2000S
- Full SEU protection through 4-bit parity and duplication
- No timing penalty, 4 RAM blocks overhead on RTAX

Synthesis results

Core	Cells	% of RTAX2000	Mhz
LEON3 + caches	3650	15.00%	35
PCI, master/target + DMA	2750	9.00%	45/70
10/100 Mbit Ethernet MAC	2200	7.00%	65
PROM/SRAM controller	500	2.00%	75
SDRAM controller	550	2.00%	75
LEON3 SOC system with: PCI, memory ctrl, timers, uarts, Irq ctrl, GPIO, ethernet MAC	16250	51.00%	33

LEON3 multi-processor support

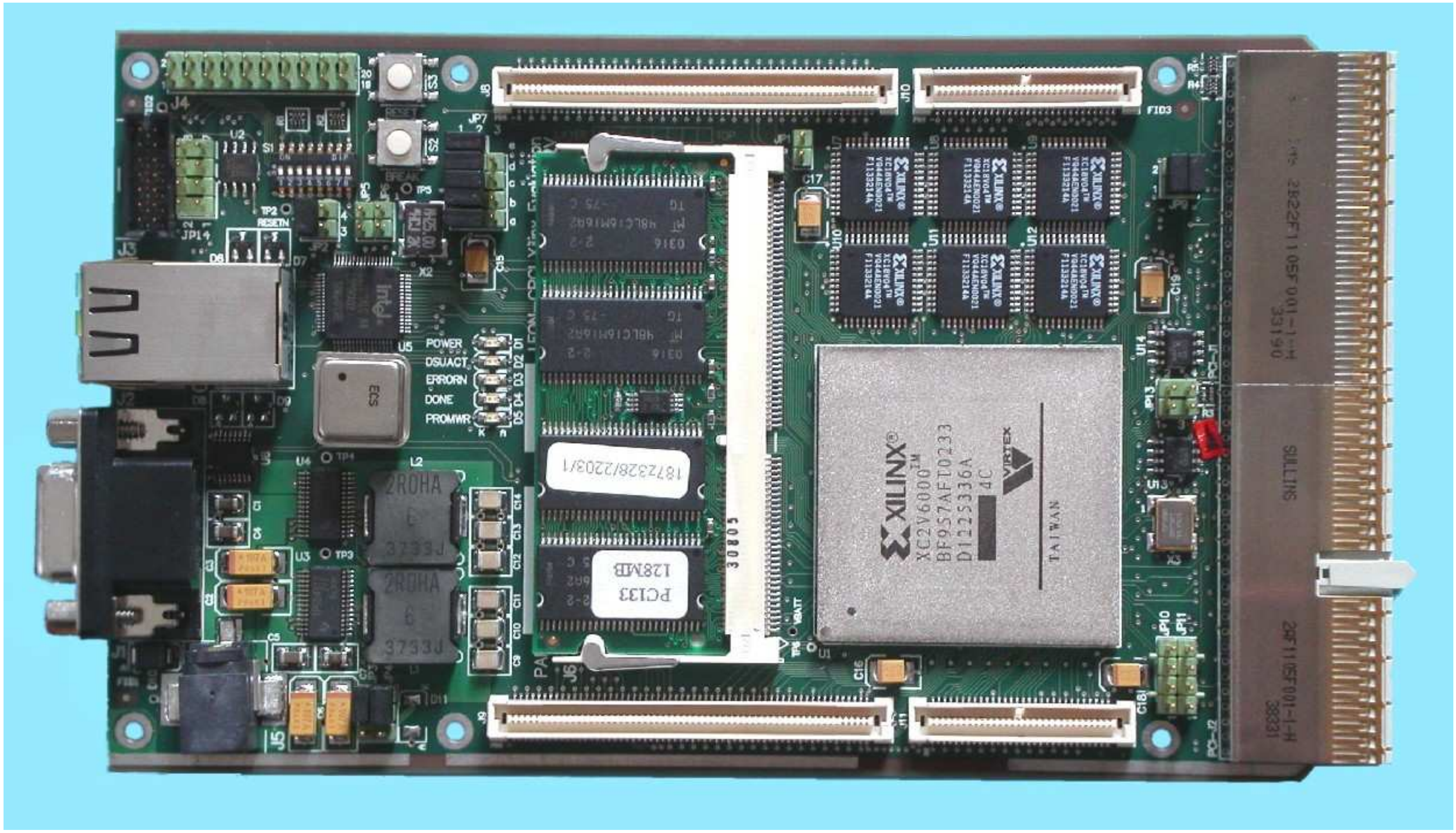
- LEON3 processor core + caches = 3 mm2 on 0.18 process
- Multi-processor system possible without area problems
- More than 4 cores not practical due to memory bandwidth
- Asymmetric configuration possible, e.g.2 'main' processors with FPU/MMU + 1 I/O (DMA/Interrupt) processor
- Multi-processor DSU and interrupt controller available
- 4-processor system fits on XC2V3000 FPGA @ 80 MHz
- 4-processor system fits on a RTAX2000 @ 25 MHz

GRLIB Support tools

- GRMON plug&play debug monitor
  - Debug 'drivers' for each specific IP core
  - Modules allow IP vendors to provide own drivers
- GRSIM modular simulator
  - Modular, re-entrant simulator based on TSIM
  - Can simulate any number of buses, cores or cpu's
  - Vendor independent models
  - Allows hardware/software co-simulation!

LEON3/SOC Development board

- Low-cost LEON CPCI FPGA development board available with **XC2V6000**, SDRAM, Flash, SRAM, 100-Mbit Ethernet



GRLIB availability

- Freely available in source code under GNU GPL
  - Valuable tool for academic research
  - Improves test-coverage due to large user-base
  - Allows early prototyping and try-before-buy
- Initial release September 2004
- Commercial licensing possible without restrictions
- The fault-tolerant version of the cores and the FPU are not initially released in open-source, but the long-term strategy is to release all cores under GPL.