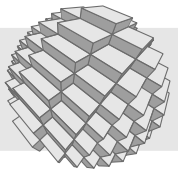

dagobert SoC [FACT SHEET]

netpp on chip 'processor'

The *dagobert* System on Chip (SoC) is the successor of the cranach networking-on-FPGA SoC.

It is part of the cCAP (configureable Custom Application Processor) family concept by section5.ch. The cCAP excel in high configureability and low resource usage on FPGA silicon. Independent of the FPGA technology, peripherals and IP cores can be tailored flexibly, starting with low-level register addresses up to number of peripheral instances (such as UART or SPI interfaces) or custom IP cores.

The cCAP family was particularely developed to meet specific requirements of robust and proveable functionality (like for safety relevant environments) while being easy to handle for application programmers at the same time, in order to facilitate handling of FPGA technology. The compact instruction set of the core architecture enables very compact program code as well as customer specific micro code extensions with DSP emulation functionality.

1 Functional overview 'dagobert'

- 32 Bit processor 'ZPUng' v1.1, three stage pipeline @54 Mhz
- Wishbone-Bus for address decoding and peripherals
- Autobuffer DMA for high speed data I/O
- Programmable in C (GCC)
- Hardware Debugger (JTAG), GDB
- Ethernet, UDP/IP stack

For technical specifications of the standard configuration, see Table 1.

Tab. 1: dagobert standard configuration

Technical specifications		
Memory configuration		
ROM	256kB	Overlay program memory (SPI) read only
L1RAM	16kB	Level one program and data segment
L1CACHE	16kB	Program memory or ROM cache (configureable)
SRAM	2kB	Dedicated Stack Memory
SPAD	8kB	Two ScratchPad RAM buffers for peripheral data I/O
Peripherals		
SCACHE	1	SPI flash cache
SIC	1	System Interrupt Controller with four peripheral channels and configureable priorities
DMAA	2	Autobuffer DMA for high speed RX/TX
UART	1	UART console
TWI	1	Two wire interface (I2C compatible)
MAC	1	Media Access Control Core (MII/MDIO), 100M

2 Reference setup

The reference environment for the dagobert SoC is the netpp node evaluation kit platform (Fig. 1)

This evaluation is supplied with a 'netpp bare metal' setup which demonstrates GPIO I/O and analog measurement using property access via the *netpp* library. This can happen in various ways:

- Command line tool
- Python scripting language
- National Instruments LabView
- GUI control via pvbrowser, wxWidgets wrappers, custom DLL

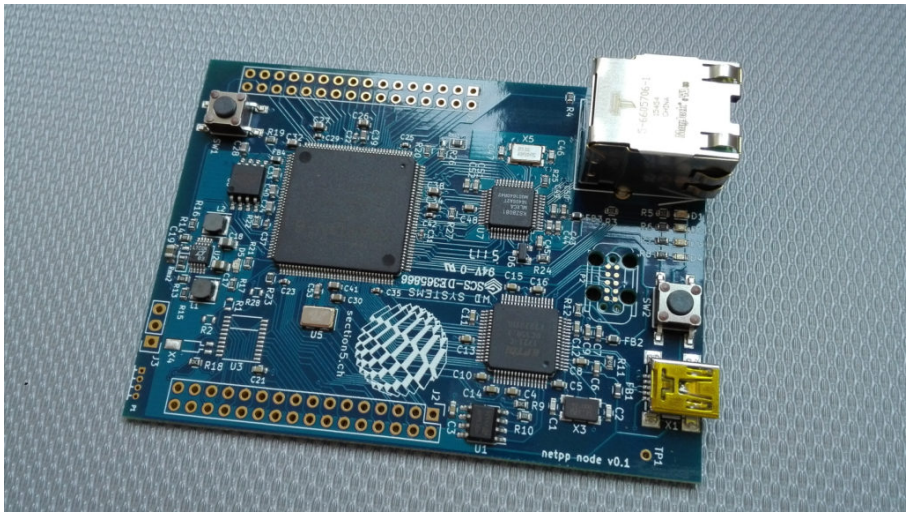

Fig. 1: netpp node eval kit

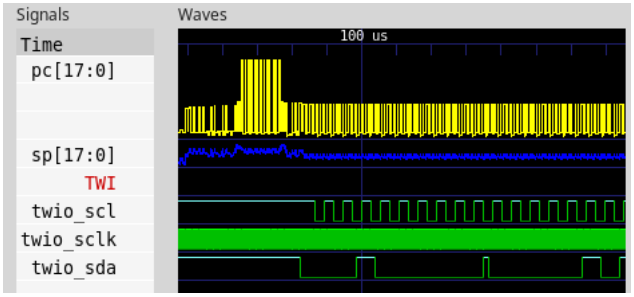

Fig. 2: i2c example trace

3 Simulation

The entire *dagobert* SoC can be co-simulated in a virtual machine. Advantages:

- Cycle-accurate verification of correct functionality, including code profiling, coverage, detection of uninitialized values
- Efficient visual debugging in trace waveforms (Fig. 2), optional on-chip trace debugging via Ethernet
- Co-Simulation with Python-Scripts for functional verification

4 Example applications

- Measurement and control
- Real-Time Network stress testing
- RTP applications (90 kHz time base)

5 Reference-Platforms

The dagobert SoC is a specific version for the netpp node evaluation board. Functionality overview:

- Read ADC values or control GPIOs, PWM, etc. over the network (100M ethernet)
- Reprogramming over network
- Extendable and flexible

The netpp node is available at:
section5.ch/index.php/product/netpp-node-v0-1/
A few resource usage examples are displayed in the tables below. The number of used EBR (Block RAM instances) can be reduced if no extended buffering is required.

Tab. 2: Resource usage netpp node LX9

	Spartan6-LX9
Registers	2961 (25%)
LUT	3972 (69%)
RAMB16	30 (93%)

6 Further pointers

dagobert Hardware Reference `soc-dagobert.pdf`
On request only (NDA required)

netpp library
section5.ch/netpp