

CHECKERS

Retargetable Instruction-Set Simulator

Checkers is a **retargetable instruction-set simulator** (ISS) for application-specific processors (ASIPs) that is part of the **IP Designer™** tool suite. It offers the following features:

- A unique approach to **ISS retargetability**, based on the processor modeling language **nML**. nML is a high-level language at the abstraction level of a programmer's manual of a processor. The user supplies bit-accurate simulation models of the processor's primitive operations, in **C++**.
- **Cycle-accurate** simulation, monitoring the full instruction pipeline. Fast **compiled-code instruction-accurate** simulation.
- Loading of Elf executable files, optionally containing source-level debug information in Dwarf-2.0 format.
- Customizable **graphical debugger**, displaying binary, assembly and C source code. Display of values in memories, registers and on busses. Through an application programming interface, this graphical debugger can also connect to the processor hardware for **on-chip debugging** (e.g. via a JTAG port).
- **Source-level debugging**, showing correspondence between executed instructions and C statements, and between register or memory locations and C variables.
- Support of **breakpoints** on instructions and on C statements, and of **watchpoints** on register and memory locations.
- Instruction and storage **profiling** capabilities.
- Waveform plotting and analysis capabilities.
- Application programming interface to **3rd-party simulators**, for co-simulation of the processor and its environment.
- Easy integration in **SystemC**.
- Application programming interface to **3rd party integrated design environments (IDEs)**.

Graphical debugger of the Checkers ISS, while simulating an MPEG4 motion estimation function on an ASIP.

The screenshot displays the Checkers graphical debugger interface. The main window is titled 'base (base)' and contains several panels:

- Micro-code:** Shows assembly instructions for an ASIP, such as `0008 3000 // movb r0,0` and `000C 5F00 // addb sp,-2`.
- Source-code:** Shows the corresponding C code for a search function, including `static int sad_16x16(unsigned char* search,` and `for (int i = 0; i < 16; i++) {`.
- PC history:** Shows the current instruction address (000001) and the next instruction address (000002).
- Instruction history:** Shows the current instruction being executed (`<nvald>`).
- Instruction/Cycle count:** Shows the number of instructions (1321), cycles (user: 14110, system: 14110), and the direction of execution (Up).
- Stack info:** Shows the current stack pointer (Low: 000000, High: 000024).
- Profiling information:** A separate window titled 'Profiling information' shows a bar chart for 'sad_16x16_motion'. The chart displays cycle counts (red bars) and instruction counts (green bars) for each instruction. The x-axis represents instruction addresses from 14 to 30, and the y-axis represents counts from 0 to 8640.
- Registers and Memories:** A panel at the bottom shows the current state of registers (P, PC, SP, LR, SR, LF) and loop control registers (LC, LS, LE).

GO

RTL Generator

Go is a retargetable **register-transfer level (RTL) hardware generator** for ASIPs that is part of the **IP Designer** tool suite. Once an ASIP has been optimized using Chess and Checkers, Go provides a quick and efficient route to hardware for the new ASIP, including the following features:

- Automatic translation of the processor's nML description into synthesizable **VHDL** or **Verilog** code.
- Supports a structural design style, using synchronous logic.
- The generated RTL description can be synthesized efficiently with standard, commercially available **ASIC** or **FPGA** synthesis tools. A synthesis script is automatically generated.
- Existing hardware blocks can be integrated in the RTL design.
- Many configuration parameters can be defined by the user, to influence the RTL style.
- Supports **low-power design optimizations**, such as selective clock gating per register and operand isolation.
- Automatic **test-bench generation** for simulation.

RISK

Test Program Generator

Risk is a **retargetable test program generator** for ASIPs that is part of the **IP Designer** tool suite. Risk allows to quickly generate a large number of assembly-level test-programs for the ASIP, which can then be executed both in the ISS and in the RTL model of the ASIP, to check for consistency of both models. This is especially relevant if the user introduces custom hardware modules in the design. Risk offers the following features:

- Using **template files**, the user can influence the basic structure of the generated test programs.
- Efficient **random generator** functions are available, to automatically select instruction sub-classes and bit-patterns in the generated code. Random generators can be **biased** to favor error-prone patterns.
- Includes customizable **fault coverage analysis**.
- Users can deploy selective test strategies for intensive verification of specific processor sub-systems, with a high fault coverage.
- Easy interfacing to the ISS generated by Checkers and to RTL simulators.

IP Designer is available on RedHat Linux and Windows (2000, XP, Vista).

For more details go to www.retarget.com

EUROPE/HQ

Target Compiler Technologies
Haasrode Research Park
Technologielaan 11-0002
B-3001 Leuven
Belgium
+32-16-38 10 30
info@retarget.com

NORTH AMERICA

Target Compiler Technologies
1004 Grant Place
Boulder, CO 80302
U.S.A.
+1-303-459-4337
info@retarget.com

JAPAN

Innotech Corporation
3-17-6 Shin-Yokohama
Kouhoku-ku
Yokohama-Shi
Kanagawa 222-8580
Japan
+81-45-474 2293
target@innotech.co.jp

Copyright © 2008 by Target Compiler Technologies™ NV. All rights reserved.

Target Compiler Technologies, The ASIP Company, IP Designer, and IP Programmer are trademarks of Target Compiler Technologies NV.

All other trademarks are owned by their respective owners.

Target may make changes to specifications, product descriptions, and plans at any time, without notice.