



3L Diamond 4.0 – Multiprocessor Tool Suite

The 3L Diamond multiprocessor tool suite delivers highly efficient multiprocessor applications in the quickest possible time.

Benefiting from 22 years of 3L's multiprocessor design experience, 3L Diamond is distinct in being created from the ground up for multiprocessor architectures featuring high performance DSP and FPGA.

It supports industry-standard design languages and design entry tool flows, and allows IP reuse through block-based design.

3L Diamond is supplied with a comprehensive development environment that is easy-to-use, flexible and scalable.

3L Diamond includes:

- a proven model for multicore and distributed multiprocessing
- a multiprocessor compiler to process the model
- a compact, robust, high-performance RTOS optimised for distributed multicore systems
- seamless integration of FPGAs
- an Eclipse-based multiprocessor IDE.

Benefits

Ease of use

Diamond uses the standard, unaltered C language to program DSPs. It does not require you to alter or replace standard compilers or to learn a new programming language. It provides a simple and easy-to-use message-passing API that results in simpler and more maintainable code.

High-level and hardware independence

Diamond reduces low-level parallel programming effort while improving portability and safety with a high-level API that abstracts low-level hardware parallelism mechanisms. It targets distributed multiprocessor systems and multicore architectures simultaneously.

Forward-scaling

You can re-target a Diamond application to multiple multicore processors and hardware with different architectures, instruction sets, cache architectures and core counts without having to re-write your code.

Safety

Diamond minimises parallel programming bugs such as memory sharing races and deadlocks by



design. Diamond guards against these problems by providing a robust design methodology based on the model of processors, tasks and channels which guides you throughout the development. The 3L Multiprocessor Compiler, which processes the model, automates most of the management of the system (memory, RTOS, device drivers, ...) thus minimising the risk of errors and allowing you to focus on your product.

Proven Multiprocessor Model

Diamond allows you to separate the implementation of your application into two largely independent sections: a software section based around communicating *tasks* and a hardware section built from processors communicating over data-transfer links.

At the heart of Diamond is the *configurer*, a multiprocessor compiler that combines your software and hardware descriptions into a complete application ready to run. The configurer performs the final build step and so has access to your complete application, allowing it to perform many system-wide optimisations that cannot be attempted by conventional tools.

You develop your application in three steps:

1. Describe your application as a number of independent software tasks that are connected by logical communication channels. You write DSP tasks in C and build them with standard tools. The configurer will combine them with the Diamond RTOS, run-time library and appropriate device drivers. You write FPGA tasks in VHDL or using high level design entry tools such as Impulse-C or Xilinx System Generator. They will be automatically instantiated in a top level component alongside the various Diamond IP blocks to create a working system (DSP/FPGA interconnect, clock domains, ...). The FPGA is built using standard tools to produce a bitstream.
2. Describe your target hardware as a network of processors (DSP or FPGA) connected by communication devices.
3. Finally, map the software onto the hardware by describing on which processor each task should run.

The 3L Diamond Multiprocessor Compiler takes these descriptions and constructs your application by choosing the most efficient implementation for your channels and adding the necessary extra software and firmware.

The result of the compilation is a single file containing everything needed for the host Diamond server to load your application onto the target hardware and get it running.

This model is the same whether you are running on one processor or any number of them, and whether the processors are DSPs, FPGAs, or a combination of DSPs and FPGAs.

Changing task placement is almost trivial: update the description of where tasks are placed and then rebuild the application; you change nothing else.



Real-Time Performance for DSPs

The compact and robust Diamond real time operating system (RTOS) is ideal for distributed multicore systems. It allows developers to focus on their product while confident the OS will not negatively impact overall performance. 3L Diamond OS comprises a small kernel, device drivers, libraries, and a simple, easy to use API.

Real-time Response

The Diamond RTOS provides fast interrupt response and fast context switching. The real time reliability and performance of the 3L Diamond RTOS is supported by a set of features including fast context switches and low interrupt latencies, to achieve the best possible response time.

Simple and Easy to Use API

Powerful and simple channel communication functions provide you with a high-level, straightforward interface which can be simply described as tasks connected by channels.

Functions for thread creation, destruction and priority control are available.

The Diamond API supports semaphores, events and timers.

Scalable for Distributed Systems

The Diamond 'channel-based' API is specifically designed for distributed multiprocessor systems.

A channel is an abstract mechanism that Diamond uses to transfer messages (data) efficiently from one task to exactly one other task. The two communicating tasks may be on the same processor, on processors directly connected by a physical link, or on processors that must be reached via a sequence of links.

The Diamond configurer automatically chooses the implementation of the channels when it builds the application.

Built-in Support for FPGAs

The Diamond FPGA co-design tool lets you add FPGAs to your application.

Diamond FPGA automatically generates HDL frameworks for the FPGAs in your system and provides communication links to support communication between DSP and FPGA.

Diamond FPGA seamlessly integrates with standard HDL design flows allowing users to leverage existing investments.



Kane Computing Ltd
7 Theatre Court, London Road,
Northwich, Cheshire, CW9 5HB, UK.
Tel: +44(0)1606 351006
Fax: +44(0)1606 351007/8
Email: sales@kanecomputing.com
Web: www.kanecomputing.co.uk