

CHREC Tools (existing and upcoming)

NSF CHREC Center

Latest update: July 1, 2011

Tool name: Assorted core libraries, benchmarks, and applications

Project(s): Various projects, all four CHREC sites

Summary: A large and growing variety of kernel, benchmark, and application codes and cores have and continue to be produced from CHREC projects at all four of the university sites in support of various project goals and research. These codes and cores target a diverse range of processing devices (e.g. FPGA, TILE64, CPU, Cell, GPU, and DSP), for a wide range of science domains, such as signal and image processing, bioinformatics, cryptology, control systems, etc.

Tool name: Autonomy

Project(s): V3-09 (Virginia Tech)

Summary: This tool suite and API provides a different use-model for FPGAs that could be used to create new capabilities to members' products, and provide a potentially new method for fault recovery. This tool is intended to ultimately make a system that is to a degree "self-aware" and can configure itself autonomously at a fine granularity in response to external stimuli. A system with these capabilities could do system-level link maintenance (cognitive and software-defined radios), change the topology of a computation (HPC), universal modularity for in-system adaptation, or rework a system in response to a highly localized failure.

Tool name: Bitstream Relocator

Project(s): F4-08 (University of Florida)

Summary: Partial Reconfiguration (PR) of FPGAs presents many opportunities for application design flexibility, enabling tasks to dynamically swap in and out of the FPGA without entire system interruption. However, mapping a task to any available PR region (PRR) requires a unique partial bitstream for each PRR. This replication can introduce significant overheads in terms of bitstream storage and communication requirements. Bitstream relocation is a process which allows transforming a single partial bitstream to map to any available PRR. Bitstream relocation has high potential for usage on adaptive RFT (Reconfigurable Fault Tolerance) and runtime management of FPGA resources. The bitstream relocation tool is C-based and supports Xilinx Virtex-4 devices.

Tool name: CMD: Core-level Modeling and Design Framework and Toolset

Project(s): F1-08, F1-09, F1-10 (University of Florida)

Summary: A core-level modeling and analysis methodology has been developed to provide fast and accurate prediction of important parameters of RC designs on an FPGA. Core-level modeling maintains a level of abstraction of RC designs between the RCML abstraction level and the programming level (ESL or HDL). A graphical editor provides users the ability to drag-and-drop their model components and edit component attributes and is integrated into RCML. The CMD framework also includes an analysis tool to predict frequency, latency, and resource utilization of RC application design for a given FPGA. This analysis tool is integrated with RAT to provide a more accurate and comprehensive prediction capability for large RC applications and systems.

Tool name: DAPR+: Design Automation for Partially Reconfigurable FPGA Systems

Project(s): F4-09, F4-10, F4-11 (University of Florida)

Summary: Partial reconfiguration (PR) is a powerful feature of Xilinx FPGAs, which permits individual FPGA region reconfiguration at runtime while the remaining FPGA regions continue

operation. PR systems have a high potential for enhancing the performance, power, functionality, and/or size of applications that require hardware adaptation or that can time-multiplex the FPGA resources. However, current design and implementation of PR systems is a highly specialized process involving many manual and time-consuming steps. DAPR+ is a C- and Perl-based software tool set that aids PR system design by automatically generating an application's base platform hardware architecture (VHDL-based) from designer-specified parameters. Additionally, DAPR+ automatically performs PR system floorplan generation from a PR system description implemented by the designer using high-level PR-augmented VHDL. DAPR+'s output files are used to automatically run the standard Xilinx PR Design Flow to implement a candidate PR system. DAPR+ evaluates the generated candidate PR system's performance using a hardware profiler and if the candidate PR system meets designer-specified performance constraints, the candidate PR system's bitstreams are output, otherwise DAPR+ re-floorplans the PR system using simulated annealing and generates another candidate PR system. DAPR+ candidate system generation is an iterative process that can be controlled by the designer since it is possible that candidate PR systems may never meet the designer-specified performance constraints, which causes DAPR+ to run indefinitely. The tool currently supports Virtex-4 devices and expanding support to other FPGA families is future work.

Tool name: Decomposable Macroblocks Exploration Tools

Project(s): F2-10, F1-11 (University of Florida)

Summary: This tool consists of a C++ library and simple frontend for performing macroblock placement and routing with adaptive decomposition. The library can generate low-level (VQM) netlists and PAR constraint files fully specifying the implementation of in-memory or XML input netlists. Preliminary results show speedups > 100x over Quartus 9.1 for placement alone. Tools are also included to generate the macroblock implementation databases required by the PAR library from simple XML+VHDL core descriptions and device information available through QUIP. The tool currently supports Cyclone-III devices and in future will be extended to support other families.

Tool name: DECIDE: Device Entry, Comparison, & Integrated Decision Environment

Project(s): F5-11 (University of Florida)

Summary: This tool is a web-based application that expands on the features provided by F5-10's prior tool, the Device Metrics Analysis Tool. The supported metrics include computational density (CD) and CD per Watt (CD/W) as well as the newer metrics for I/O bandwidth (IOB), internal memory bandwidth (IMB), and external memory bandwidth (EMB). Users can now quickly select which devices and metrics they would like to view for comparison and immediately generate graphs for efficient device analysis. For the CD and CD/W metrics, a sliding bar allows users to quickly adjust the operations ratio and immediately see the effects in the dynamically adjusting graphs. Like the previous F5 tool, the expansive database of devices (CPUs, DSPs, FPGAs, GPUs, etc.) is available for use through the tool, and users are welcome to share with the CHREC community by submitting data for their own devices as well.

Tool name: EDIF TMR Tools

Project(s): B3-08 (Brigham Young University)

Summary: This software provides several APIs for parsing and manipulating EDIF netlists. A key component of this API is the ability to automatically triplicate EDIF circuit designs and insert voters to support Triple Modular Redundancy (TMR). A variety of other reliability tools are included with this package as well as a user guide. The original development of this tool at BYU predates CHREC, but was updated in CHREC and posted as open source via project B3 in 2008.

Tool name: Extensible Middleware for RC Platforms

Project(s): F1-10, F3-11 (University of Florida)

Summary: The objective of this tool is to enable seamless application migration between RC platforms. It will provide a set of abstractions away from platform-specific board and API details, creating a virtual environment for vendor-independent application execution. This platform-independent virtual environment features a virtualized board and API, allowing a single application source to be executed on multiple platforms with minimal change and user effort. An XML-based *Platform Description Framework* will be used to encapsulate platform-specific details, enabling the generation of the virtual environment. This XML framework will allow for easily extending support for additional platforms. In addition to assisting with application migration, the standardized virtual interface provided by the middleware will also simplify and help accelerate the deployment of high-level synthesis tools.

Tool name: FoRSE: Formulation-level PR Design Space Exploration

Project(s): F4-11 (University of Florida)

Summary: Run-time partial reconfiguration (PR) provides many benefits for FPGA-based applications; however, exploiting these benefits requires applications to be developed as PR-capable with an associated PR-architecture. Designing PR-capable applications and PR-architectures are challenging tasks due to many competing implementation metrics (e.g., area, power, operating frequency, etc.), which results in large design spaces (there is one design space per pair of competing implementation metrics). In order to assist designers in efficiently and effectively selecting an appropriate design space based on the application's requirements, rapid PR design space exploration (DSE) techniques and tools are required. FoRSE is the first formulation-level PR DSE tool to rapidly prune the application's design space. FoRSE leverages the application's PR-architecture and mathematical models of FPGA device- and vendor-specified PR technology to generate Pareto-optimal sets of PR-floorplans and devices based on designer-designated implementation metrics. FoRSE operates at the application formulation level, which affords very fast runtimes, allowing FoRSE to prune the application's implementation design space by 3-4 orders of magnitude in approximately 15 seconds.

Tool name: High-Level-Language Productivity Analysis Tool

Project(s): G4-07 (George Washington University)

Summary: The tool enables comparative analyses of high-level languages for developing reconfigurable computing applications. The tool identifies numerous language attributes and enables the user to do a weighted, comparative evaluation. The weights enable the user to highlight features of interest and diminish the effect of undesirable attributes on the final outcome.

Tool name: Intermediate Fabrics Exploration Tool

Project(s): F2-09, F2-10, F1-11 (University of Florida)

Summary: This tool consists of a C++ library and simple frontend for performing placement and routing on intermediate fabrics (IFs), coarse-grain virtual reconfigurable fabrics that serve as a translation layer on top of COTS FPGAs to enable circuit portability and fast PAR. The library can generate bitfiles for programming IFs from in-memory or XML input netlists. Results show an average PAR speedup of 550x over ISE 10.1 across a number of case studies. Tools are also included to generate individual IFs required by the PAR library from XML+VHDL core and fabric topology descriptions. The tools are vendor-agnostic (i.e. useful for Altera, Xilinx, and more).

Tool name: **OpenCL High-Level Synthesis Library**

Project(s): F2-10, F1-11 (University of Florida)

Summary: This tool consists of a proof-of-concept C++ library and simple frontend for performing high-level synthesis from C-like languages generally and OpenCL kernels specifically, supporting rapid runtime synthesis of accelerator circuits through the use of IFs and decomposable macroblock PAR (by integration with those libraries).

Tool name: **PFIF: Portable Framework Interface**

Project(s): G5-08 (George Washington University)

Summary: This tool is an Eclipse-based IDE for developing applications on various reconfigurable computers. The user chooses the targeted reconfigurable platform (Cray XD1, SGI RC100) and also specifies memory interfaces to be used from a drop-down list. The tool guides the user through a simple wizard and generates the required template files in a high-level language (Impulse-C), to assist the user in developing the application. The tool also generates the necessary wrapper files as well as PFIF files specific to the selected platform, and carries out the required synthesis, PAR, and bitstream generation.

Tool name: **Profiling and Scheduling IDE**

Project(s): G1-07, G6-08 (George Washington University)

Summary: This tool provides a graphical front-end for profiling and analyzing applications written in C, for visualizing and identifying computation- and data-intensive functions. The application developer can also view the call graph of the application. The call graph is treated as a DFG and partitioned/scheduled across multiple FPGA configurations, and the corresponding task clusters are indicated in the call graph. Instead of using C source files, the user also has the option of providing the task graph of the application as an input to the tool directly, by using a text file that describes the graph.

Tool name: **QAReEM Design-Space Exploration Tool**

Project(s): G7-09, G7-10 (George Washington University)

Summary: QAReEM (Queuing Analysis of Reconfigurable Execution Models) is a MATLAB-based tool developed for design-space exploration of system virtualization solutions that exploit full/partial reconfiguration in order to share FPGA resources among multiple user processes in high-performance reconfigurable computers (HPRCs). The tool is built on an analytical model that is based on a discrete-time Markov chain representation of the system. Various system scenarios can be simulated, such as deterministic (periodic) or stochastic generation of tasks from the user processes, single or multiple I/O channels for communicating with the FPGA resources, shortest job first or first-come first-served scheduling for task processing, etc.

Tool name: **QFlow**

Project(s): V2-11 (Virginia Tech)

Summary: When compared to other computing solutions, FPGA application development environments have suffered by requiring a higher degree of expertise and longer compilation times, leading to lower overall productivity. There has been a great deal of attention paid to front-end productivity enhancements for FPGAs with new languages, graphical entry tools, and core generators. Very little attention has been paid to back-end compilation, where the time to produce a downloadable bit file can take as much as 20-30 hours for a moderate-size FPGA. This back-end domain has been considered something that only the vendor can control. QFlow is an alternative back-end compilation environment that greatly accelerates the compilation of Xilinx FPGAs,

reducing compile times by one or two orders of magnitude with the expense of some area efficiency. Furthermore, QFlow is compatible with most existing front-end design entry environments, preserving a user's existing entry tool-chain.

Tool name: **RAT: RC Amenability Test**

Project(s): F3-07, F3-08, F1-09 (University of Florida)

Summary: This tool is an Excel worksheet that, using the RAT performance model, enables analytical performance modeling of a specific RC application targeting a specific RC platform. Users provide key application and platform parameters as defined by the tool, which are used to calculate the performance predictions. RAT encompasses both a device-level model for analysis of single-FPGA systems (F3-07, F3-08) and modeling aspects being leveraged and added (F1-09) from LogGP to provide system-level prediction for scalable, multi-FPGA platforms. The goal is to maintain efficient and reasonably accurate prediction prior to expensive implementation iterations.

Tool name: **RapidSmith**

Project(s): B1-10, B1-11 (Brigham Young University)

Summary: XDL is an intermediate language supported by Xilinx for the description of FPGA designs. It is equivalent to NCD in that it can represent designs which have been mapped, placed, and/or routed. The RapidSmith tool set is a low-level XDL manipulation library, written in Java, which can manipulate Xilinx designs described in XDL. Using it, a variety of manipulations can be done such as replicating circuit modules, changing circuit connections, changing LUT contents, etc. This package grew out of the B1-10 project, which focused on creating a rapid prototyping flow based on XDL and which bypasses most of the Xilinx tool flow. Both the B1-10 and B1-11 projects use RapidSmith as their foundation to manipulate Xilinx designs in XDL.

Tool name: **RCML: RC Modeling Language Editor and Analysis Toolset**

Project(s): F1-08, F1-09, F1-10, F1-11 (University of Florida)

Summary: This tool is an Eclipse plug-in that allows users to create, save, and edit models based upon the RC Modeling Language (RCML) created in CHREC in 2008. The graphical editor provides users the ability to drag-and-drop model components and edit models (application and machine) through graphical interfaces and dialogs. Analysis tools are integrated into the RCML editor allowing users to analyze performance and behavior of their RCML model using system-level methodologies and tools developed in CHREC, namely RAT and RCSE. Methods for automated mapping and design-space exploration of RCML system models have also been integrated into this tool.

Tool name: **RCSE: RC Simulation Environment**

Project(s): F1-07, F1-08 (University of Florida)

Summary: This tool is a library of discrete-event models built within the commercial simulation environment called Mission-Level Designer (MLD) to support the rapid simulative analysis of RC systems, comprised of applications and machines and their mappings. The application models take as input a script representing a trace of the application under study. Scripts represent the application as a sequence of major events to drive simulative analysis. The architecture models in the library are highly generic components that use parameter files to tune the generic models to behave as a specific device technology.

Tool name: **ReCAP: Reconfigurable Computing Application Performance**

Project(s): F2-07, F2-08, F2-09, F2-10 (University of Florida)

Summary: ReCAP provides the ability to automatically instrument, measure, and present performance data from a reconfigurable application (e.g., multi-CPU/FPGA application) via a major extension to the Parallel Performance Wizard (PPW) tool that provides performance analysis for conventional parallel systems. The hardware portion of the application may be written using a hardware description language such as VHDL or a high-level language such as Impulse-C or Carte-C. ReCAP includes an instrumentation framework to gain access to application data as well as a hardware measurement module (HMM) to record, store, and permit runtime access of application performance data on the FPGA. ReCAP also provides RC-targeted visualization and platform-aware, knowledge-based bottleneck detection capabilities for quickly locating common performance bottlenecks across diverse systems as well as providing a mechanism for users to easily port ReCAP to other platforms. ReCAP also supports runtime verification and debug features such as assertion synthesis (including support for VHDL- and C-based assertions, timing-based assertions, and synchronous OVL 2.0 assertions), code coverage, and testbench generation that are particularly useful to large-scale (e.g., multi-FPGA) applications.

Tool name: RFT Time-Varying Fault-Rate Estimator

Project(s): F6-10, F6-11 (University of Florida)

Summary: Reconfigurable Fault Tolerance (RFT) aims to improve overall system capability by dynamic tradeoff of performance and fault-tolerance in environments (e.g. Earth orbit) with varying fault rates. The RFT Fault-Rate Estimator uses orbital data for existing satellites and other space-based systems (TLEs) to generate orbital paths. These paths are then correlated with data from the International Geomagnetic Reference Field (IGRF) model to estimate the magnetic field strength at the satellite's position over time. The estimated field strength is then correlated to device vulnerability using the CREME96 model to provide realistic time-varying fault-rate profiles.

Tool name: SCF: System Coordination Framework Toolset

Project(s): F1-09, F1-10 (University of Florida)

Summary: This framework provides a simplified mechanism for establishing communication in parallel RC applications using a message-passing programming model. SCF features an Eclipse plugin that allows users to describe applications as task graphs and define a mapping of tasks onto resources. A backend parser reads the task graph and mapping information to generate communication infrastructure. It auto-generates communication constructs for all the tasks in the application which are written in one of the supported IDEs and for an SCF-compliant target system.

Tool name: SHMEM+ Communication Library

Project(s): F1-09, F1-10 (University of Florida)

Summary: The SHMEM+ communication library, based upon an adaptation of the partitioned global-address-space (PGAS) programming model, provides RC application developers with a productive mechanism for developing parallel RC applications spanning multiple boards and servers. This library enables distributed, shared memory with direct transfers between any two devices of a scalable RC system (e.g. a transfer from the CPU on one board or server to an FPGA on another). A baseline set of SHMEM functions has been developed and evaluated on Novo-G.

Tool name: SPFI: Simple Portable Fault Injector

Project(s): F6-09, F6-10, F6-11 (University of Florida)

Summary: SPFI is a comprehensive tool suite that emulates SEUs onto a variety of devices. It enables estimation of the susceptibility of a design/program to radiation-induced effects. Supported devices include: Xilinx Virtex-4 and Virtex-5 FPGAs (SPFI-FPGA); embedded PPC 405 and 440

on the FX series of Xilinx devices (SPFI-ePPC); standalone PPC devices (SPFI-PPC); and Tiler TILE64 and Maestro (SPFI-TILE). With SPFI-FPGA, the fault-injection (FI) process can be performed spatially or temporally. Spatial FI, addresses the static configuration memory of the device where the temporal FI, targets the dynamic components (BRAM, FF, SLR). The tool architecture is divided into four major components. The *Campaign Generator* uses a debug bitstream file generated by Xilinx ISE tools to determine random locations of errors to be injected. The *Management Engine* creates crafted partial bitstream files that are then programmed over JTAG to inject and remove faults; this process is further accelerated by testing multiple bits simultaneously. The *Test Generator* is a user-provided program that verifies the operation of the design of the FPGA and provides that information to the *Logging Engine*, which combines and saves all the injection details into a log file for post-mortem analysis. With SPFI-TILE, SPFI-ePPC, and SPFI-PPC, fault-injection processes are performed by using GDB to modify contents of memory or registers. Software under test is randomly paused during runtime and an error inserted, then the program is resumed and executed until completion or expiration of the deadline. Results are recorded into a detailed log for subsequent analysis.

Tool name: System-Level Virtualization Support for HPRCs

Project(s): G7-10 (George Washington University)

Summary: System-level virtualization techniques for HPRCs have been explored in the G7-10 project and a proof of concept has been developed for the Cray XD1 reconfigurable computer. The developed virtualization solution is based on run-time reconfiguration of FPGA on the hardware side, and kernel-space based resource management incorporated into the operating system on the software side. The developed infrastructure is complemented by software stack that provide access to the available services at a high level of abstraction, thereby hiding from the application developer all the details of multiplicity of FPGAs, scheduling techniques, node architecture, and hardware implementation.

Tool name: UPC Compiler for Tiler

Project(s): G8-09, G8-10 (George Washington University)

Summary: This tool provides UPC compile and runtime environment for the Tiler Tile64 processor. It uses the existing Berkeley compiler and runtime system which have been ported over on the Tiler platform. Two specific runtime conduits have been ported, (i) pThreads-based GASNet runtime conduit, and (ii) MPI-based GASNet runtime conduit. The tool adds various language and runtime level optimizations to the existing environment for the Tile64 processor such as address translation, thread placement, and optimized synchronization primitives and collectives.

Tool name: VAPRES System Builder

Project(s): F4-08, F4-09, F4-10 (University of Florida)

Summary: VAPRES (Virtual Architecture for Partially Reconfigurable Embedded Systems) is a multipurpose ready-to-use base platform for rapid-deployment of partially reconfigurable embedded systems. VAPRES is created around the Xilinx MicroBlaze soft-core processor and currently targets Virtex-4 devices. Multiple hardware coprocessors are connected to the MicroBlaze through FSL (Fast Simplex Link) streaming interfaces. Special features of VAPRES include: 1) independently clocked hardware coprocessors; 2) runtime reconfiguration of hardware coprocessors; and 3) inter-coprocessor communication support via SCORES (Scalable Communication Architecture for Modular Reconfigurable Systems). VAPRES includes architectural parameters to enable application customization. The VAPRES System Builder tool is a Linux program written in Python/GTK that

automates the process of building VAPRES-based systems and applications. VAPRES System Builder includes a VAPRES API for high-level access to VAPRES architectural features.

Tool name: **Virtualization through FPGA Resource Sharing**

Project(s): G7-08, G7-09 (George Washington University)

Summary: This virtualization solution provides a framework for sharing FPGA resources among the several microprocessor cores prevalent in contemporary high-performance reconfigurable computers (HPRCs). The solution uses partial reconfiguration to split the FPGA into multiple regions which are assigned to different processors. The software component of this framework, consisting of a daemon running in the user space, handles requests from multiple user processes that need to use the FPGA resources. Any process running on a microprocessor is therefore provided a virtual view of a dedicated FPGA attached to it. This user-space prototype is developed for the Cray XD1 reconfigurable computer.