BEE2: A High End Reconfigurable Computer

http://bwrc.eecs.berkeley.edu/Research/BEE

- Faculty in charge
  - John Wawrzynek
  - Bob W. Brodersen
- Industrial liaison
  - Bob Conn (Xilinx)
- Research associates
  - Andrei Vladimirescu
  - Vason Srini
- Graduate students
  - Chen Chang
  - Pierre-Yves Droz
  - Nan Zhou
  - Yury Markovskiy
  - Hayden So
  - Kevin Camera
  - Zohair Hyder
- Undergrad student
  - Alexander Krasnov
  - Greg Gibeling
  - Jeffry West
  - John Conner
- Technical staff
  - Brian Richards
  - Susan H. Mellers

Motivations

High-End Reconfigurable Computer

(HERC)

is a computer, whose hardware can be configured on a per-application-basis to match the requirements of the task by exploiting all levels parallelism in both temporal and spatial dimensions, to achieve supercomputer-like performance.

March 11, 2004 BWRC, UC Berkeley

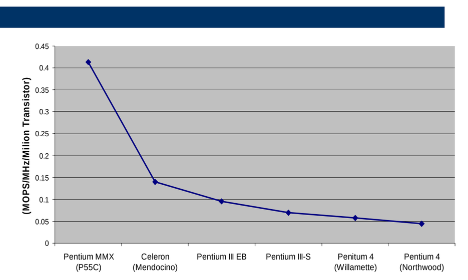
Conventional High-End Computers

Clusters of "commodity" microprocessors

- System performance in the 100's of GFLOPs to 10's of TFLOP range.
- Original intension is to use commodity microprocessors for scalability and low cost
  - Cost of Earth Simulator is around **$350M**, for peak performance of 40TFLOPS
  - **$8,500 per GFLOPs**

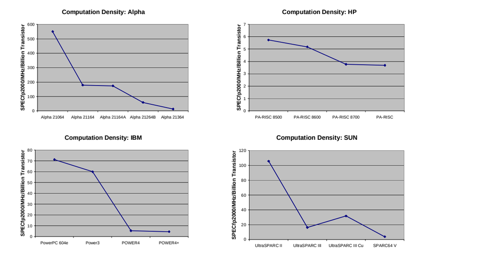
March 11, 2004 BWRC, UC Berkeley

Computation Density of Processors



March 11, 2004 BWRC, UC Berkeley

Computation Density (cont.)



March 11, 2004 BWRC, UC Berkeley

Existing multi-processor HEC system problems

- Memory organization
  - Inadequate memory bandwidth
  - Processor vs. memory speed gap
  - Cache inefficiencies
- Scalability
  - Power is the limiting factor
  - Global interconnect is the bottleneck

March 11, 2004 BWRC, UC Berkeley

Applications of Interest

- **High-performance DSP/communication systems**
  - Cognitive radio or SDR
  - Image processing and navigation
- **Scientific computation and simulation**
  - E & M simulation for antenna design
- **Others**
  - Large Ad-Hoc network simulation
  - Bioinformatics
  - CAD tool acceleration

March 11, 2004 BWRC, UC Berkeley

High-performance DSP

- "Stream-based" computation model
  - Usually hard real-time requirement
  - High-bandwidth data I/O
- Low numerical precision requirements
  - Mostly fix-point operations
  - Rarely needs reduced floating point
- Data flow processing dominated
  - few control branch points

March 11, 2004 BWRC, UC Berkeley

Scientific Computing

- Computationally demanding
  - Double-precision floating point
  - Large matrix operations, linear systems solvers (lapack), 2D/3D FFTs
- Traditionally not real-time processing, but real-time processing would offer new applications.
- Opportunities to innovate on the algorithm and mapping for reconfigurable computing

March 11, 2004 BWRC, UC Berkeley

Large Ad-hoc network simulation

- Simulate 1K ~ 1M nodes in the network, communicating with ad-hoc protocol, for environment sensing and actuation applications
- Massively coarse grain level parallelism
- Event-based concurrency makes the simulation inefficient on conventional processor based parallel computers
- Near real-time performance needed to validate network design

March 11, 2004 BWRC, UC Berkeley

Bioinformatics

- Implicitly parallel algorithms
  - Stream-like data processing
  - Integer operations sufficient
- History of success with reconfigurable/ASIC architectures. (TimeLogic, Paracell)
- High-capacity persistent storage devices required for matching large database

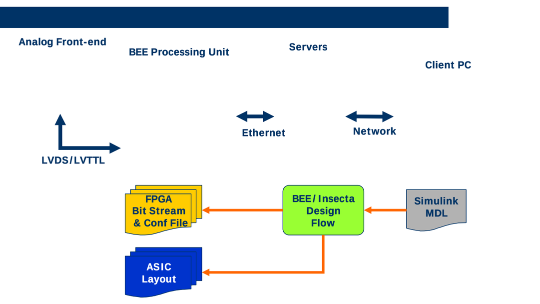
March 11, 2004 BWRC, UC Berkeley

Berkeley Emulation Engine

- FPGA-based system for real-time hardware emulation:
  - Emulation speeds ***up to 60 MHz***
  - Emulation capacity of ***10 Million*** ASIC gate-equivalents, corresponding to ***600 GOPS*** (16-bit adds)
  - ***2400*** external parallel I/Os providing ***192 Gbps*** raw bandwidth.

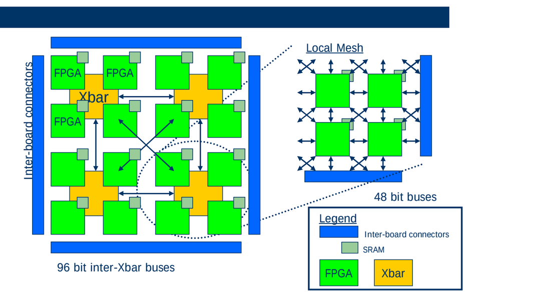
March 11, 2004 BWRC, UC Berkeley

BEE System Overview



March 11, 2004 BWRC, UC Berkeley

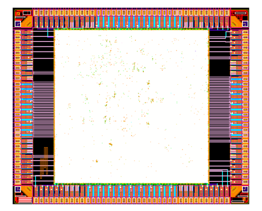
Interconnect Architecture



March 11, 2004 BWRC, UC Berkeley

Current BEE Status

- Four BEE processing units built
- Three in continuous "production" use
- Supported universities
  - CMU, USC, Tampere, UMass, Stanford
- Successful tapeout of:
  - 3.2M transistor pico-radio chip
  - 1.8M transistor LDPC decoder chip
- System emulated:
  - QPSK radio transceiver
  - BCJR decoder
  - MPEG IDCT
- On-going projects
  - UWB mix-signal SOC
  - MPEG transcoder
  - Pico radio multi-node system
  - Infineon SIMD processor for SDR



March 11, 2004 BWRC, UC Berkeley

Lessons from BEE

- Simulink based tool-flow very effective FPGA programming model in DSP domain.
- **Many system emulation tasks are significant computations in their own right – high-performance emulation hardware makes for high-performance general computing.**
- ***Is this the right way to build supercomputers?***

March 11, 2004 BWRC, UC Berkeley

Hardware

The HERC Ideas

- A cluster of FPGAs
- Match the hardware to the application
- Do as much computation in hardware as possible, leave processor cores to handle the rest
- Intelligent memory and network architecture
- High-level parallel programming model
- A complete computer solution

March 11, 2004 BWRC, UC Berkeley

Characteristics of HERC

- High computation density, lower clock rate, low power conosumption.
- No cache, computing elements operate at the same speed as memory
  - Multiple independently addressed memory banks per node
  - Internal FPGA SRAM can be user-controlled cache if needed.
- Flexible interconnection network (circuit/packet switching).
- Predictable memory and network latency permit static scheduling in real-time applications, and performance tuning.

March 11, 2004 BWRC, UC Berkeley

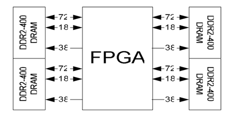
BEE2 Design Goals

- Successor of BEE as a bigger faster hardware emulator and simulation accelerator
- As a flexible platform for experimenting with "reconfigurable computing" programming models and applications
- Demonstrate supercomputer level computation at fraction of cost and size

March 11, 2004 BWRC, UC Berkeley

Basic Processing Element

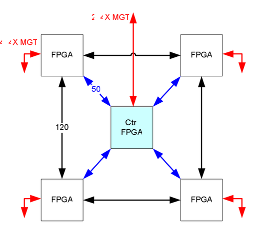
- Single Xilinx Virtex 2 Pro 100 FPGA
  - ~100K logic cells
  - 2 PowerPC405 cores
  - 444 dedicated multipliers (18-bit)
  - 1MB SRAM on-chip
  - 20X 3.125-Gbit/s duplex serial communication links (MGTs)
- 4 physical DDR2-400 banks
  - Each banks has 72 data bits with ECC
  - Independently addressed with 16 banks total
  - Up to 12.8 GBps memory bandwidth, with maximum 8 GB capacity
  - Up to 4 GBps/GFLOPs



March 12, 2004 BWRC, UC Berkeley

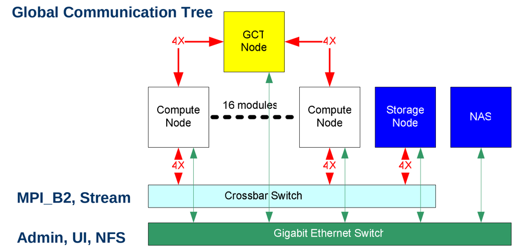
Computing Node

- 4 processing element + 1 control element
- Direct mesh connection between computing nodes
  - 120 bit 200MHz DDR
  - 48 Gbps per link
- Star connection from control node to computing nodes
  - 50 bit 200 MHZ DDR
  - 20 Gbps per link



March 11, 2004 BWRC, UC Berkeley

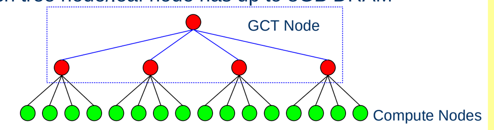
Internode Connections



March 11, 2004 BWRC, UC Berkeley

Global Communication Tree

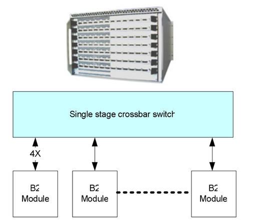
- 4-ary tree configuration
  - Off-module 4X Infiniband connection @ 10 Gbps duplex
  - On-module 50-bit 200DDR connection @ 20 Gbps
- Every 16th B2 modules act as a tree node
- Each tree node/leaf node has up to 8GB DRAM



March 11, 2004 BWRC, UC Berkeley

Crossbar Implementation #1

- Specialized crossbar switch implemented in ASIC (Mellanox, Voltaire)
- 24 ~ 96 4X ports
- 200~1000 ns switch latency
- 400~1200 ns FPGA to FPGA latency
- 480Gbps ~ 1.92Tbps full duplex constant cross section bandwidth
- Good for larger systems with 16 or more compute nodes
- ~$400 per port



March 12, 2004 BWRC, UC Berkeley

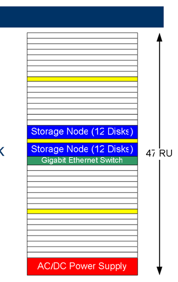
Disk Storage Schemes

- Compute node local working storage
  - OS, scratch and swap space
  - 1~2 SATA disks per node
  - 250 ~ 500 GB capacity per node, RAID 0
- Multi-node shared working storage
  - User work area
  - implemented with storage nodes
  - Up to 12 SATA disks (3TB capacity) per storage nodes
  - Any number of storage node in the system, RAID 5
- External persistent storage
  - User home directories
  - External SAN or NAS

March 11, 2004 BWRC, UC Berkeley

Rack Capacity

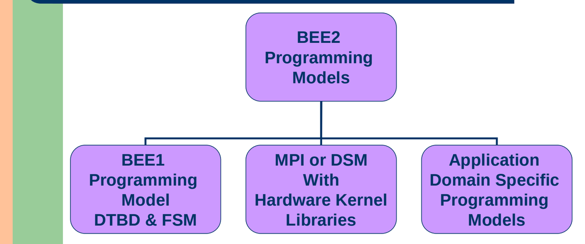
- 32 compute nodes, 3 DHSM modules, 2 Storage node
- Peak performance: 25 TOPS or 1 TFLOPS
- 512 GB local memory, 128 GB global shared memory
- 16 TB local disk, 6 TB shared work storage
- 5 Tbps I/O bandwidth
- 12 Kwatt power consumption
- Cost: ~ $250K-500K
- Performance/Price: ~$250-$500/Gflops



March 11, 2004 BWRC, UC Berkeley

Programming

Programming Model Overview



March 11, 2004 BWRC, UC Berkeley

Programming Model (1) General Purpose Parallel Programs

- Most commonly used for distributed memory computer systems
  - Message passing interface (MPI)
  - Distributed shared memory (DSM)
- BEE2 can support both through common MPI-like communication layer in hardware
- Use GCT structure to optimize collective communications and more efficient global memory sharing

March 11, 2004 BWRC, UC Berkeley

Programming Model (2) Hardware Kernel Library

- Current BEE programming model
  - Discrete Time Block Diagram with FSM
- Abstraction layer extensions
  - Virtual memory and I/O
  - Stream data access and processing
  - Shared memory model with on-die processor cores

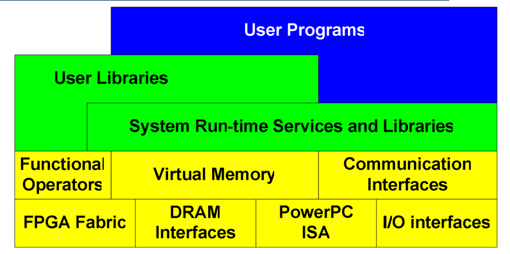
March 11, 2004 BWRC, UC Berkeley

Programming Model (3) Special Application Domains

- For particular application domain of interest
  - Specialized computation model, programming model, specification language, debugging, and visualization
  - To maximize BEE2 hardware system performance, and increase user productivity
- Example applications domains
  - Regular grid/mesh finite element simulation
  - Network simulation

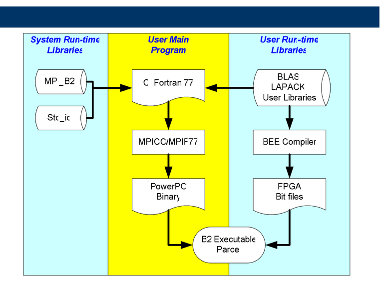
March 12, 2004 BWRC, UC Berkeley

Program Abstraction Layers



March 11, 2004 BWRC, UC Berkeley

Compiler Tool Chain Overview



March 11, 2004 BWRC, UC Berkeley

System Run-time Libraries

- MPI_B2
  - Based on MPIch
  - Collective communication implemented with direct hardware support, and common reduction functions in hardware
  - Includes some MPI2 extensions, such as single-sided communication, collective I/O.
- Std_io
  - Leverage standard Linux std_io library.

March 11, 2004 BWRC, UC Berkeley

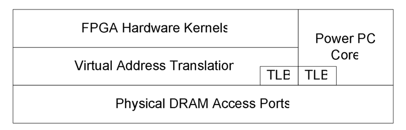
User Libraries

- Application domain specific libraries ported to BEE2 FPGA hardware implementation
  - BLAS
  - LAPACK
- Performance tunable at installation time to specific machine parameters

March 11, 2004 BWRC, UC Berkeley

Memory Access Model

- Global distributed memory, communication between exclusive memory through MPI
- Local memory shared between the processor and the FPGA hardware kernel
- Virtual address needed for protection



March 11, 2004 BWRC, UC Berkeley

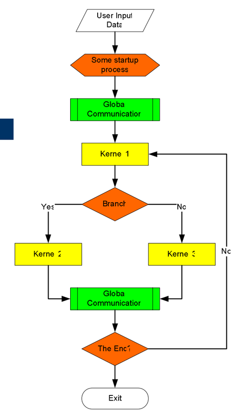
Program Execution Model

- Parallel program level
  - Single Program Multiple Data (SPMD)
  - Communicating through MPI
- Individual process level
  - Sequential execution with predefined library functions
  - Library kernels implemented on FPGA fabric and dynamically loaded at run-time
- Library kernel level
  - Data parallel execution of streaming data

March 11, 2004 BWRC, UC Berkeley

Individual Process

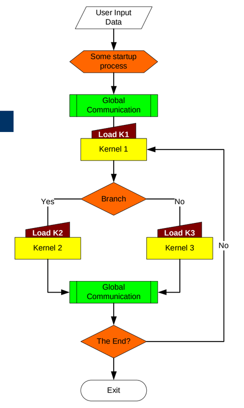
- Single process per PowerPC core (maximum 2 per FPGA)
- Kernels implemented on FPGA hardware fabric
- Global communication through MPI, and might be overlapped with computation



March 11, 2004 BWRC, UC Berkeley

Dynamic Kernel Loading

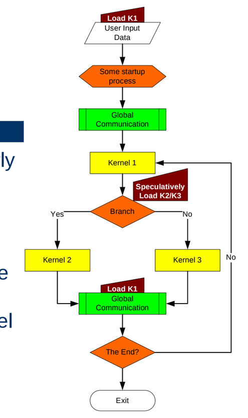
- Naïve way: just-in-time (JIT) loading right before using the kernel
- Easy to implement
- Load latency of 25~50ms always visible
- Degrade performance



March 11, 2004 BWRC, UC Berkeley

Dynamic Kernel Loading (cont.)

- Smarter way: pre-load as early as possible, so overlap load time with useful sequential computation and/or global communication
- Speculate when kernel choice depend on branch outcome
- Check if has the correct kernel before running it (kernel tagging)



March 11, 2004 BWRC, UC Berkeley

Amdahl's Law Modified

- Comparing BEE2 global performance to common micro-processor based cluster solutions
- Additional term is the sequential processing slow-down factor S
- The on-chip PowerPC core on FPGA is about 5~10 times slower than fastest sequential processors

$$Speedup = \frac{T_{sequential}}{T_{parallel}} = \lim_{P \to \infty} \frac{1}{\frac{(1-f)}{P} + f \cdot S} \approx \frac{1}{f \cdot S}$$

March 11, 2004 BWRC, UC Berkeley

Performance Modeling

- Computation Time: $\frac{t_{seq}}{P} + 2(d-1)t_{ck}$
  - Sequential processing time: tseq
  - Pipeline latency: d
  - Hardware clock period: tck
  - Parallelization factor: P
- Communication Time: $\alpha + \beta \cdot m$
  - Message overhead: α
  - Network bandwidth: β
  - Message size: m

March 11, 2004 BWRC, UC Berkeley

OS

Operating System Overview

- Each computing node runs one copy of Linux on the control FPGA as the master, the rest processing FPGAs act as slaves
- A small micro kernel resides in the kernel memory space of each processing FPGA for program loading/unload and exception handling

March 11, 2004 BWRC, UC Berkeley

User Access Management

- Work queue based multi-user environment running batch jobs on separate compute nodes
- Single compute node is the minimum job allocation unit
- Relative position of processes per FPGA can be specified on both intra and inter node levels

March 11, 2004 BWRC, UC Berkeley

Resource allocation and protection

- FPGA resource allocated per user job, and spatially isolated from other user jobs
- Global communication and I/O resources are shared between jobs, and protected by the user access privilege table maintained by the global OS
- Local memory allocated through local OS, and protected through virtual address translation

March 11, 2004 BWRC, UC Berkeley

Exception Handling

- Each hardware kernel has an exception status register (ESR)
- On error/faults:
  - stall pipeline, write error code to ESR
  - Release control to PowerPC, jump to exception handler
  - Exception handler choose to recover in-place, restart, or abort
- Page faults
  - On TLB miss, OS fixes the TLB in virtual memory manager.

March 11, 2004 BWRC, UC Berkeley

Project Status

- Currently under development
  - PCB board schematics and layout
  - MPI_B2 API, compiler, and libraries
  - Linux OS modification and micro-kernel
  - Hardware kernel library generation framework
- First prototype system of 5 compute nodes completes in second half of 2004
- Initial performance results in early 2005
- Production system in service by Q3 of 2005

March 11, 2004 BWRC, UC Berkeley