Algorithm Acceleration with Reconfigurable Hardware

Maya Gokhale

maya@lanl.gov

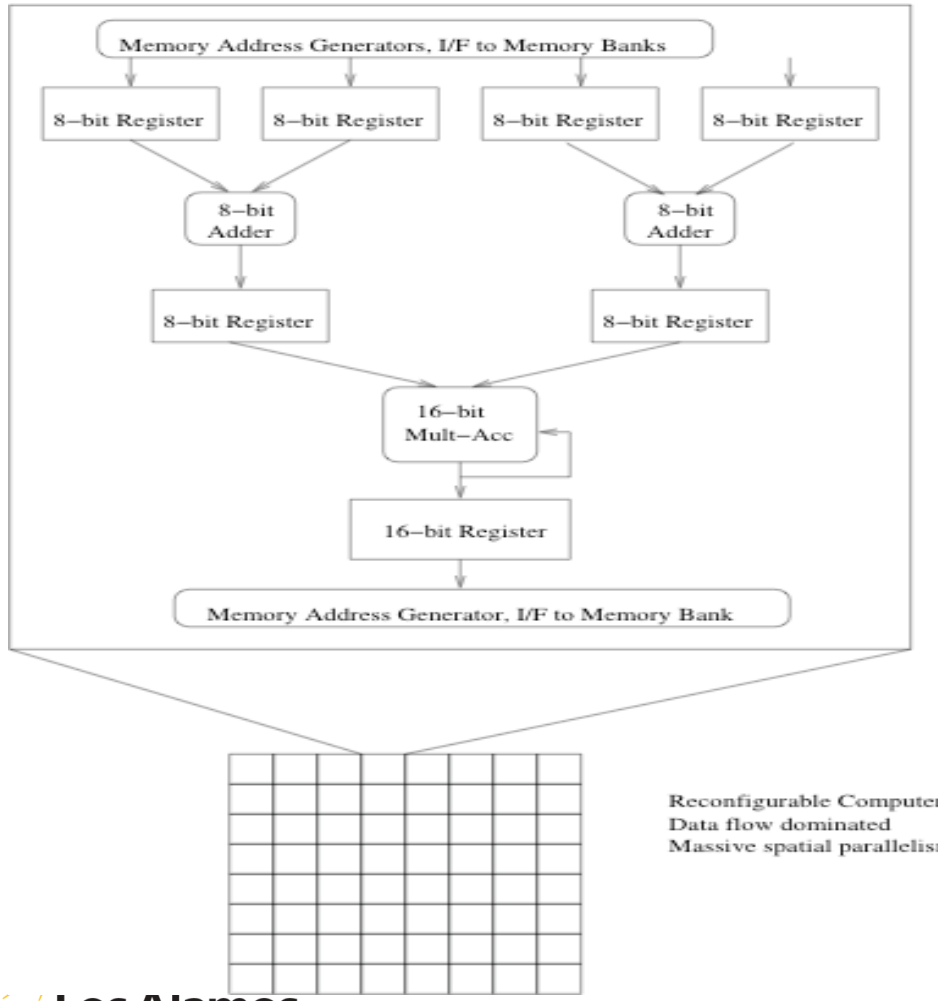
Outline

- What is reconfigurable hardware?
- How is reconfigurable hardware incorporated into high performance systems?
- What applications or application classes can benefit from reconfigurable hardware?
- What programming models are suitable for reconfigurable hardware?
- What languages and compilers exist?

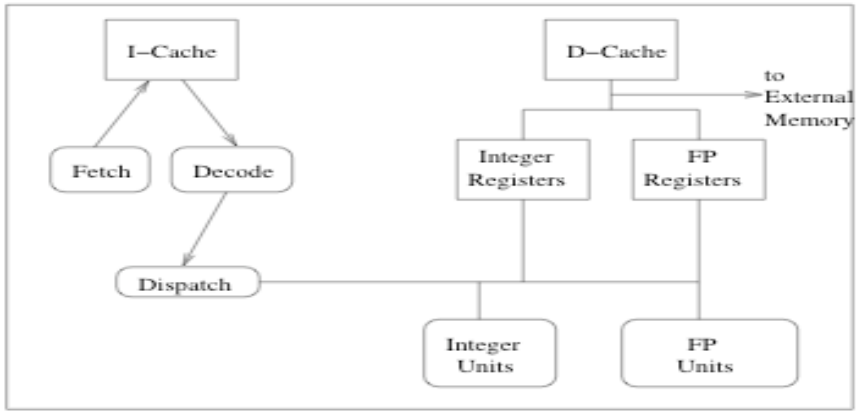
Much of this material is taken from the book:

Reconfigurable Computing: Accelerating Computation with Field-Programmable Gate Arrays, by **Gokhale**, Maya B., **Graham**, Paul S., ISBN: 0-387-26105-2

Reconfigurable Hardware: Example

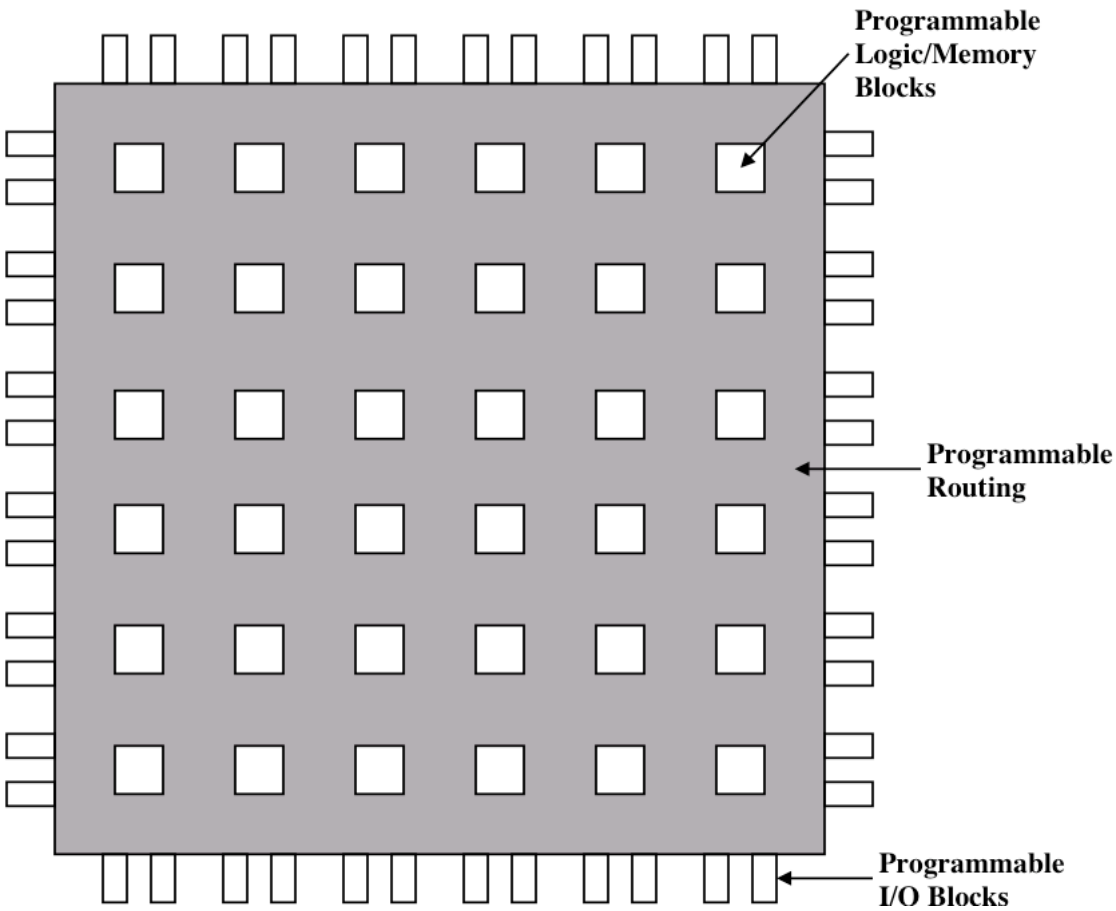


Microprocessor:
Control flow dominated
Sequential instruction stream



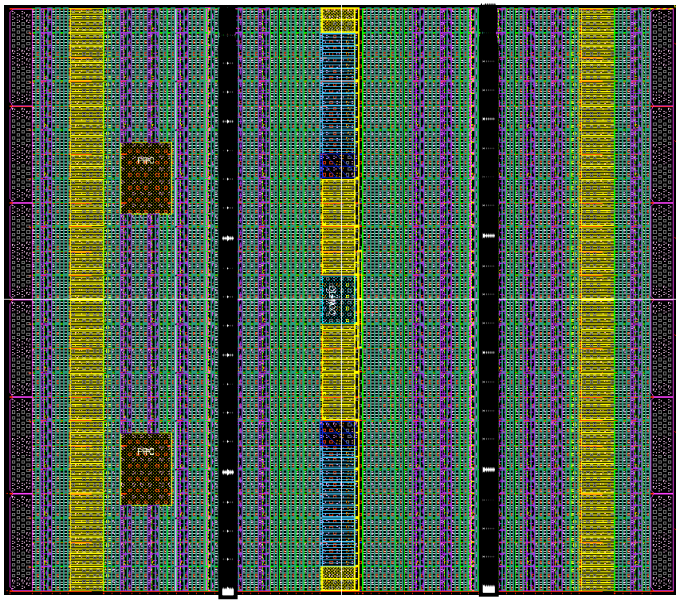
Reconfigurable Computer:
Data flow dominated
Massive spatial parallelism

What is an FPGA

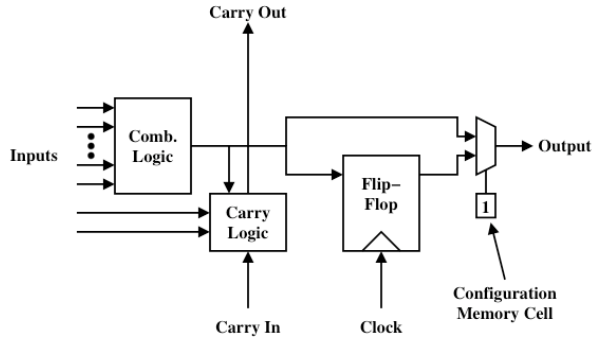


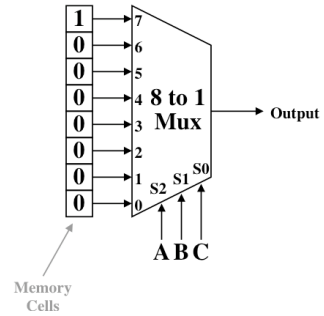
- Function Units are synthesized from Logic/Memory Blocks.
- Function Units are interconnected with programmable routing wires.
- Communication to outside uses I/O blocks

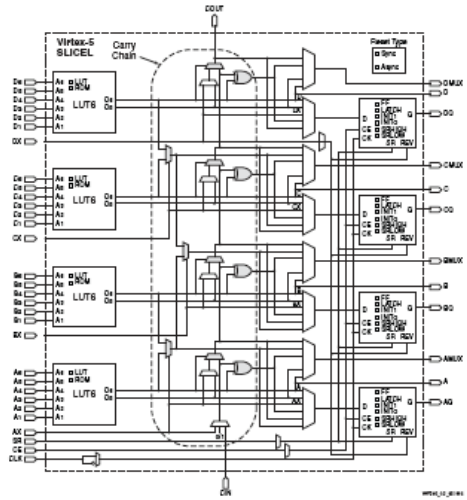
A Closer Look



Modern FPGA architecture combines logic, memory blocks, DSP blocks, processors

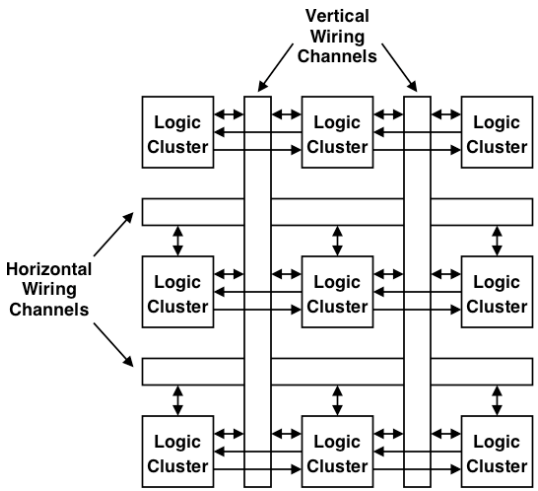




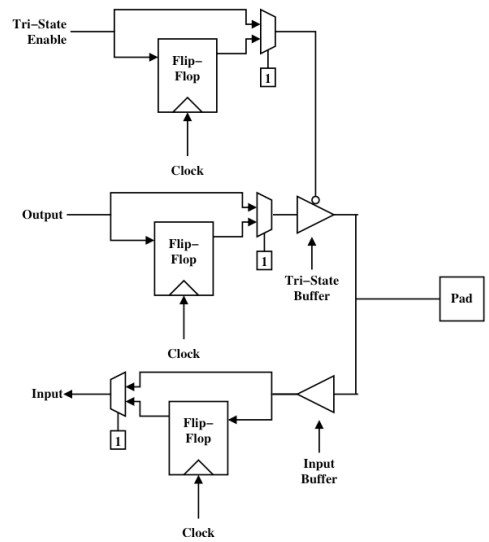


An **FPGA slice** contains flip-flops, Look-Up Tables and some associated mux, carry, and control logic.

Routing and I/O

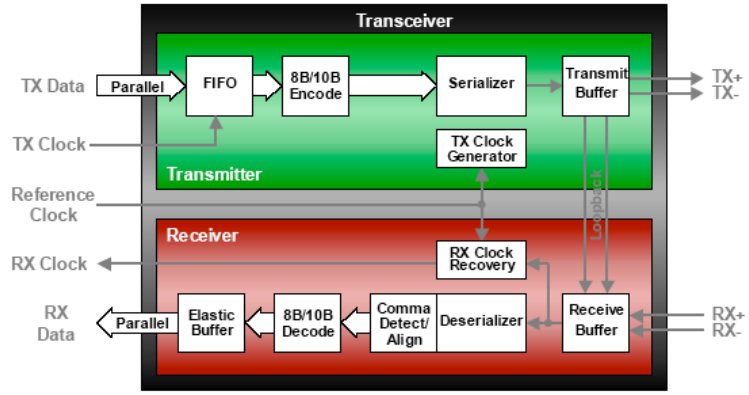


Programmable Routing Fabric

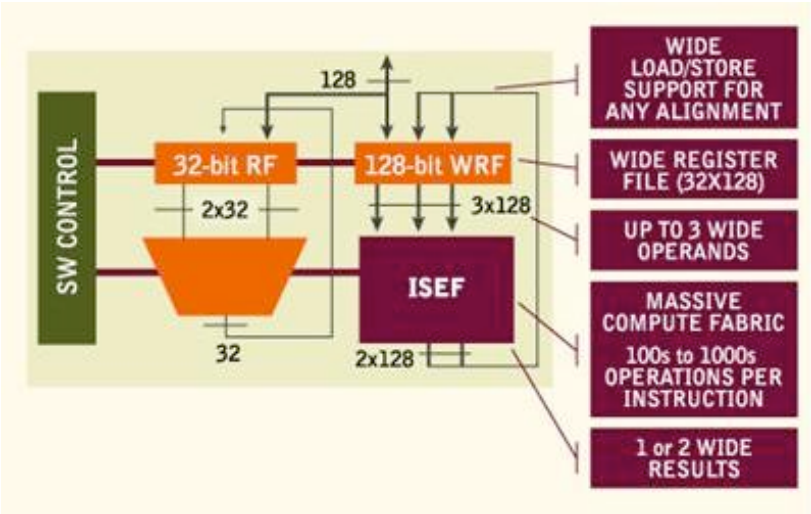


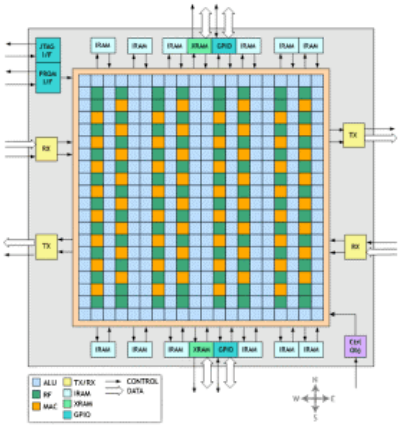
IO Block

Multi-Gb Serial IO



New SoC architectures





- Stretch: Tensilica processor + FPGA
- Configurable processor
- Closely integrated fabric interface

- Mathstar: Field Programmable Object Array
- Application-class-specific arithmetic units - ALU, MAC, RF
- Distributed memory blocks
- Programmable routing

Ambric: Globally asynchronous, locally synchronous architecture

On-chip

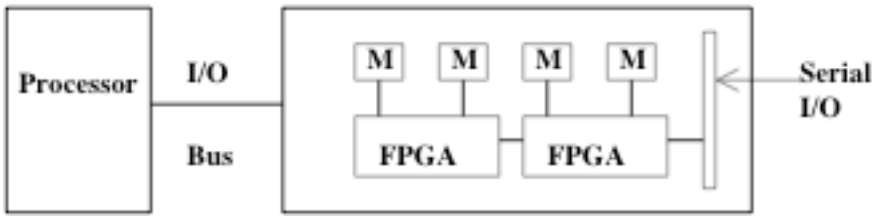
- Hundreds of 32-bit RISC processors
  - SRD processor: a 32-bit streaming RISC processor with DSP extensions. It has local memory for its 32-bit instructions, and can directly execute more code from the RAM Unit next door.
  - SR processor: a simpler 32-bit streaming RISC CPU, used mainly for managing channel traffic, generating complex address streams, and other utility tasks which enable sustained high throughput for the SRDs.
- Hundreds of distributed memories
  - RAM Units (or RUs): the main on-chip memories that can stream addresses and data over channels
- Hardware "channels" for communication between processors
  - 32-bit, unidirectional, point-to-point
  - carry data and control tokens
  - simple or structured messages
  - dynamically synchronized at run-time
  - avoids global synchronization

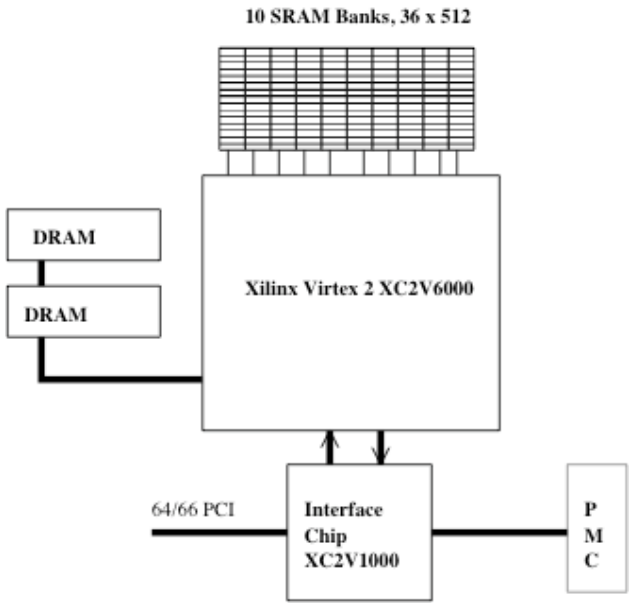
Ambric chip

- 130nm standard cell ASIC
- 360 32-bit processors
  - 1 - 333MHz
  - 45 “brics” each with 8 CPUs and 13KB local SRAM
- .585MB SRAM (4.6Mb)
  - largest Xilinx Virtex 5 has 10Mb
- channels run at 1.32GB/s (10.6Gb/s)

Putting FPGAs into systems

Workstation accelerator: I/O card



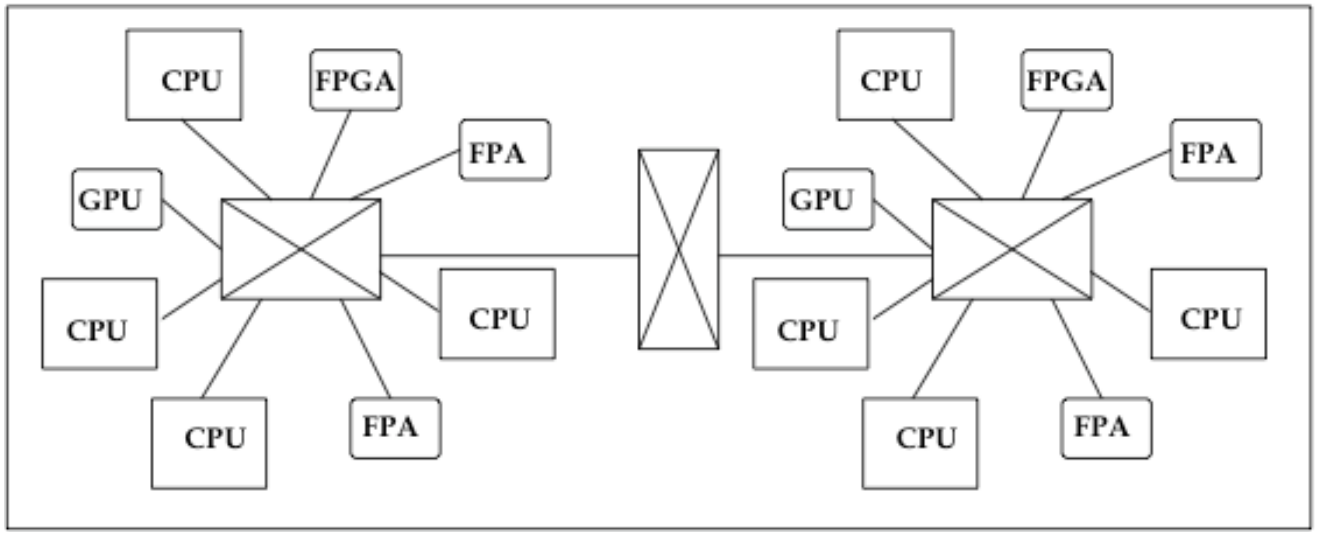


- collection of inter-connected SoC processors.
- each processor has dedicated (or shared) memory subsystem
- accelerator system attaches to “host” workstation via an I/O bus
- data acquisition channel for direct access to real-time data streams, eg. sensors or disk array
- enables “personal supercomputer”

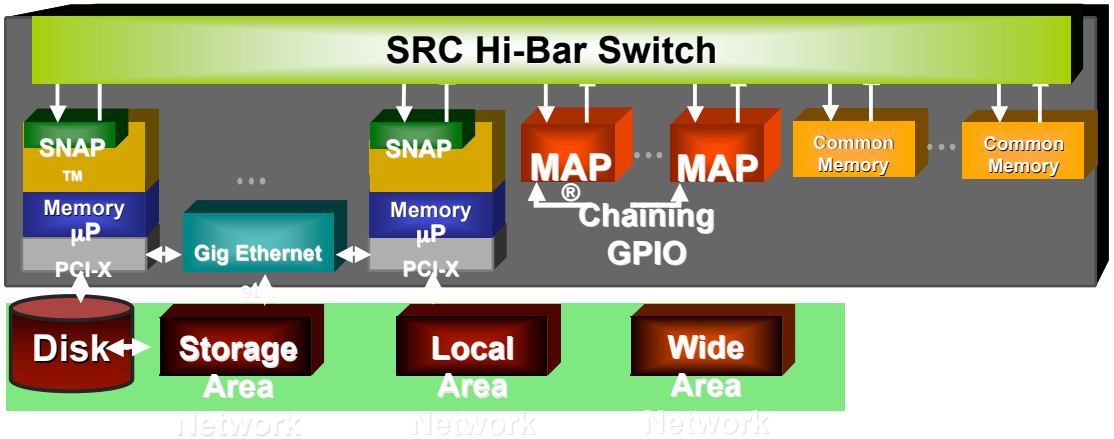
Clusters with accelerators

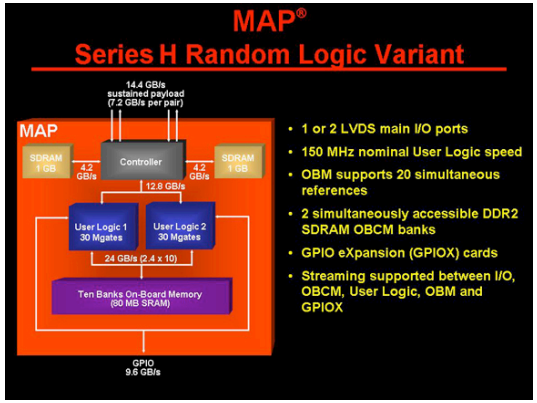
- High performance, multi-level interconnect
  - Infiniband, Myrinet, GigE
- High performance microprocessors
  - 64-bit, multi-core, multi-socket
- co-processors
  - Graphics boards, floating point arrays, FPGAs

- accelerator can be peer to microprocessor on network
- peer to microprocessor across sockets on hypertransport
- on I/O bus
- on memory bus



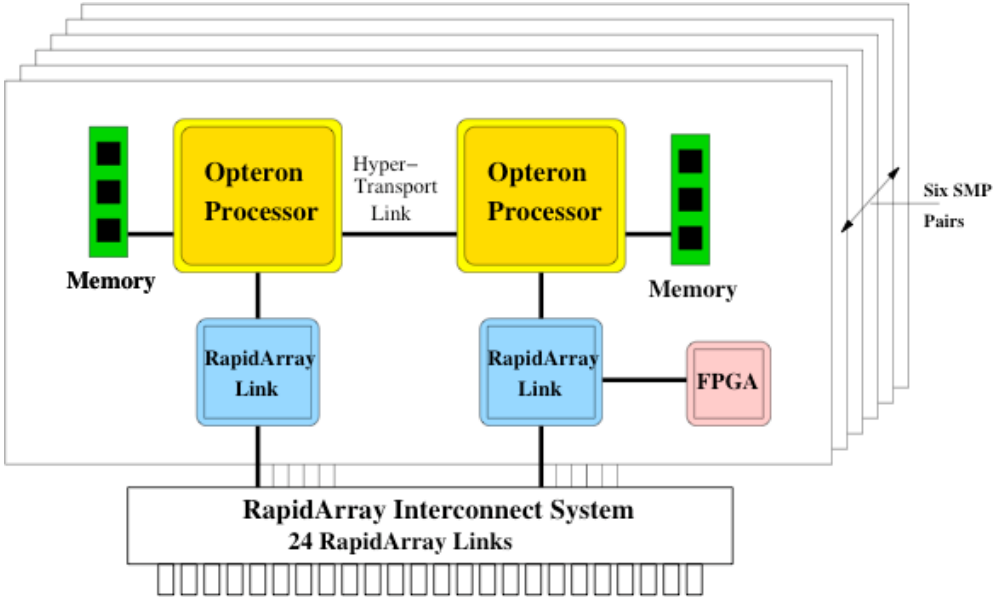
SRC Computer

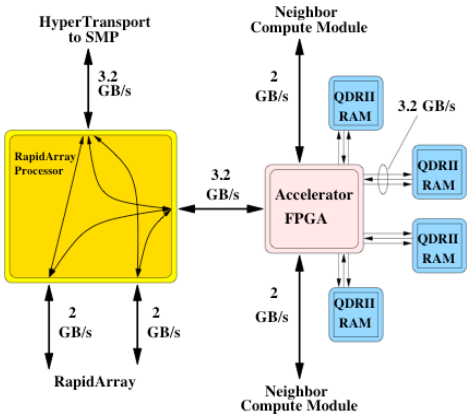




- FPGA board augments microprocessor
- "MAP" on DIMM interface 2.8 GB/s
- 2 large FPGAs
- multiple banks of on board SRAM
- on board DRAM
- provides for 20 simultaneous memory accesses @ 150MHZ
- FPGAs can be interconnected independently of microprocessor

Cray XD1



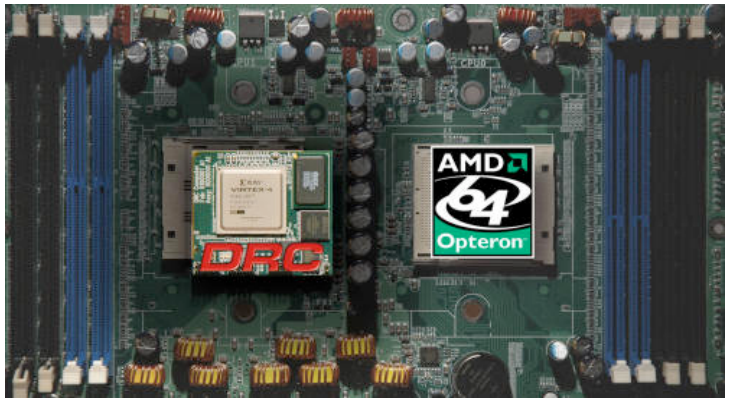


- special ASIC to talk hypertransport
- Only one small FPGA co-processor
- 16MB QDR SRAM - 12 GB/s BW

FPGA co-processors

- Opteron motherboards
- Hypertransport connection between Opteron and FPGA
- Use DIMM slots on MB for FPGA memory
- Include additional off-chip SRAM
- Two companies
  - DRC, XtremeData
- current generation HT implementations give uni-directional bandwidths of 260MB/s - 500MB/s (8-bit)

DRC →



XtremeData ↓

Applications

- Data Stream processing
  - hardware block on FPGA acquires sample from A/D or memory buffer
  - data packet flows directly through processing pipeline
  - processed data streamed to microprocessor or storage device
- Kernels of compute-intensive pipelines
  - one or more pipeline mapped onto FPGA
  - input from microprocessor
  - result back to microprocessor for further processing
- FPGAs used as “hidden” acceleration engines
  - Exegy - high speed data search for finance, etc.
  - Netezza - SQL database server backed by FPGAs for fast compare

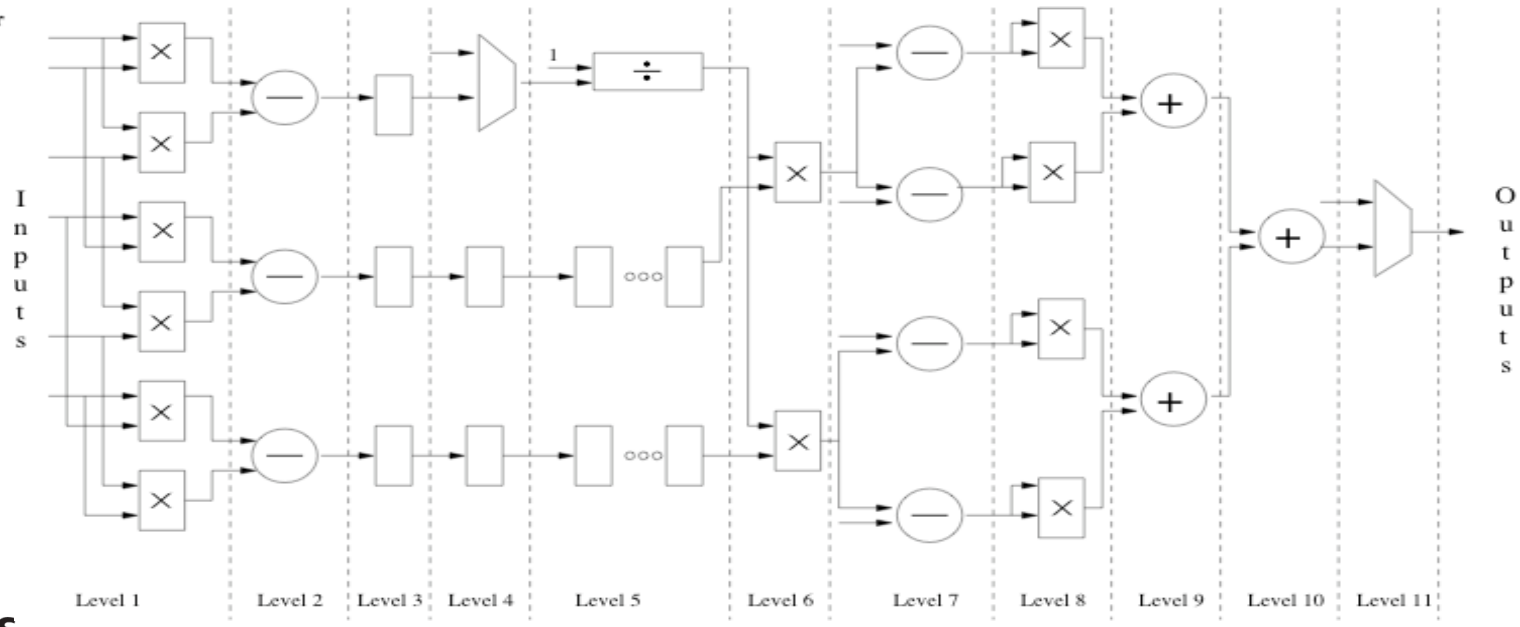
Application Design Principles

- Maximize parallelism
  - multiple customized function units
  - pipelining within function unit
  - customized interconnect among function units
    - systolic flow
- Maximize memory bandwidth
  - application-controlled memory hierarchy
  - on-chip
    - flipflops in CLB configured to be part of a register
    - Embedded on-chip RAM blocks, configurable in width/depth
  - off-chip
    - multiple SRAM banks
    - DRAM

Kernel Processing

```
a  for each surface in the m–surface polygon
b    for each of N photons emitted from this surface
       Emit a photon with random characteristics from this surface
c      while the photon is not absorbed, transmitted or lost
d        for each side in the polygon
           if the current side is not the emitted side
             Check if the photon intersected with this side
           end if
         end for
         Randomly determine if the photon is absorbed, transmitted, reflected or lost
       end while
     end for
   end for
```

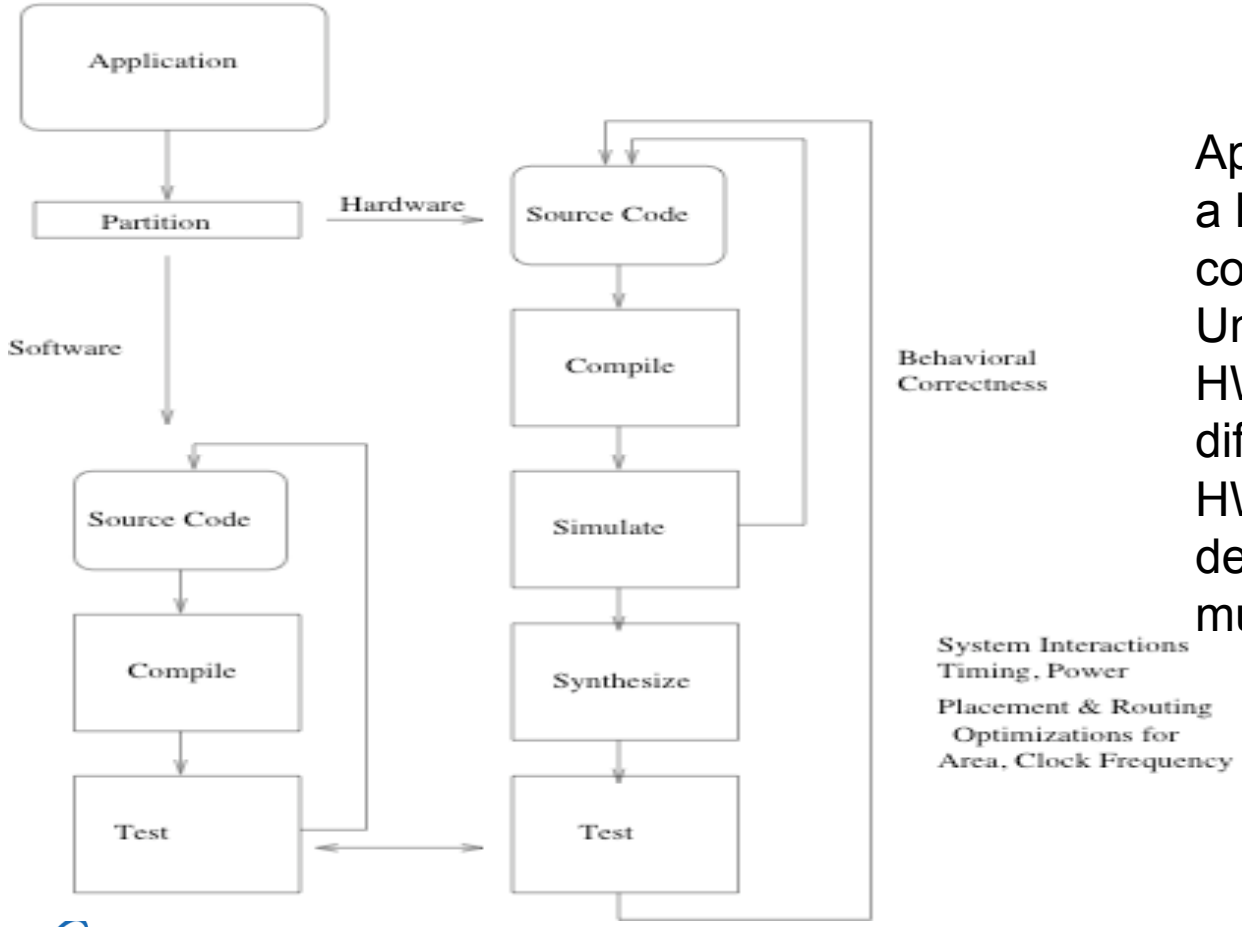
- FPGA circuit for innermost loop
- FP operators are pipelined
- Expression eval is pipelined



Design Tools and Compiling for FPGAs

- Select functions to run on FPGA
- Translate from algorithm to circuit
- Interface software and hardware

Design Cycle



Application has a hardware and software component.
Unlike SPMD model, HW and SW perform different functions.
HW and SW are compiled, debugged differently, yet must work together.

Levels of Description

- Algorithmic
  - True software level of description
  - Function calls, pointers, …
- Behavioral
  - Describe behavior of a circuit
  - Timing behavior is synthesized
  - Alternative modules automatically chosen
  - C = (A + A) * (B+C)
- RTL
  - Registers, interconnection between registers is defined by programmer
  - Timing behavior is defined by programmer
  - Control structure is defined by programmer using state machine
  - Synthesizer builds control logic for register transfers
  - Seq(Par(T1 = B+C; T2 = A+A;) C = T1*T2)
- Structural
  - Arithmetic or control modules explicitly instantiated
  - Control behavior explicitly specified
  - M1: adder(B, C, T1, clk, enable); M2: adder (a, a, T2,…); M3: mult(T1, T2, C,…)

Algorithmic Languages

- C/Fortran
  - SRC Computer C/Fortran
  - Impulse C (Impulse Accelerated Technologies) based on LANL Streams-C
  - Catapult C (Mentor Graphics)
- Matlab
  - Accelchip
  - System Generator
- Dataflow
  - Mitrion
- CSP
  - Celoxica
- Lower Level
  - System C
  - Behavioral VHDL, Verilog

RC: multiplicity of programming models

- Data parallel
- Streaming
- Dataflow
- Co-processor

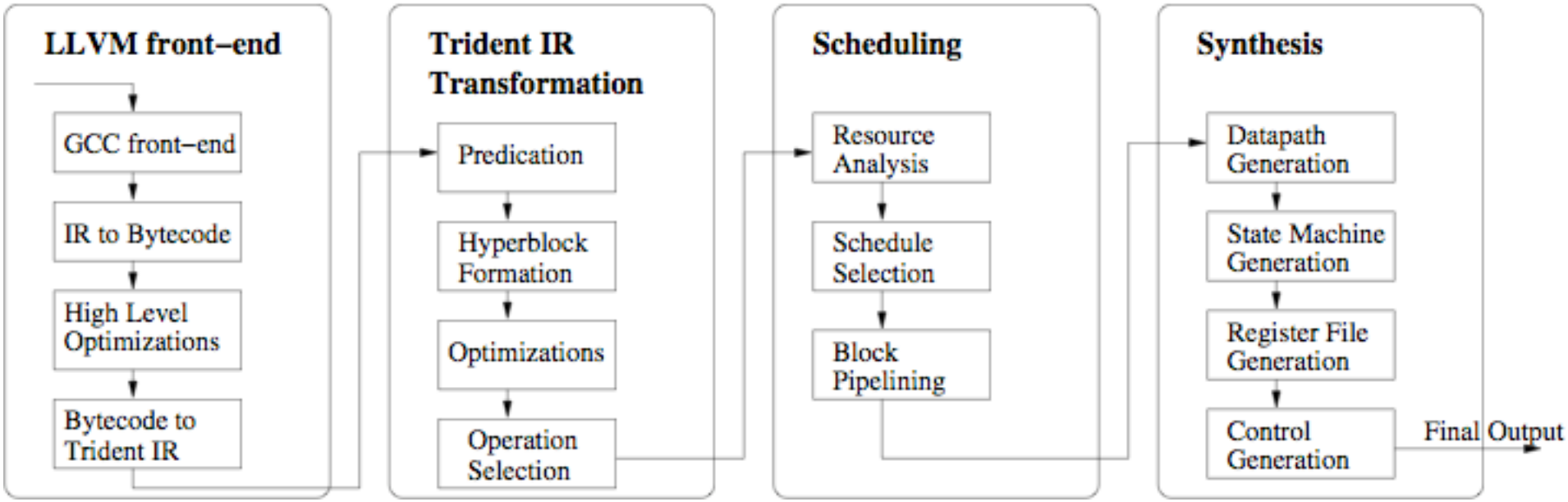
Data parallel: dbC (data parallel Bit C)

- data parallel language based on C*
  - linear SIMD array
- added arbitrary bit length integers
- used Compass compiler technology
- mapped to CM-2 and Terasys
  - SIMD processor in memory array (IEEE Computer, vol. 28, pp. 23--31, April 1995)
- mapped to Splash
  - FPGA array (Chapter 7 in Splash 2: FPGAs in a Custom Computing Machine, Wiley, 1996)
  - generated VHDL to describe each processor's behavior
  - library included hardware for global reduction operations across processing elements on 16 FPGAs

Streaming: Streams-C

- Streams-C language and compiler; *Stream-oriented FPGA computing in the Streams-C high level language*. In IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM), pages 49--56, 2000; Evaluation of the streams-C C-to-FPGA compiler:/\an applications perspective, ACM Symposium on FPGAs, 2001
- bounded buffer communication channels between processes
- a process can be mapped to software or to hardware
- 1D arrays of processes can be instantiated and interconnected by streams
- language based on C with annotations in comment lines to mark the parallel processes and stream interconnections
- compiler based on SUIF 1.3, source available at www.streams-c.lanl.gov
- compiler commercialized by Impulse Accelerated Technologies

Trident overview



- compiles C function to FPGA - user specified floating point modules
- GPL source code at http://sourceforge.net/projects/trident/

Compiling for Reconfigurable Systems

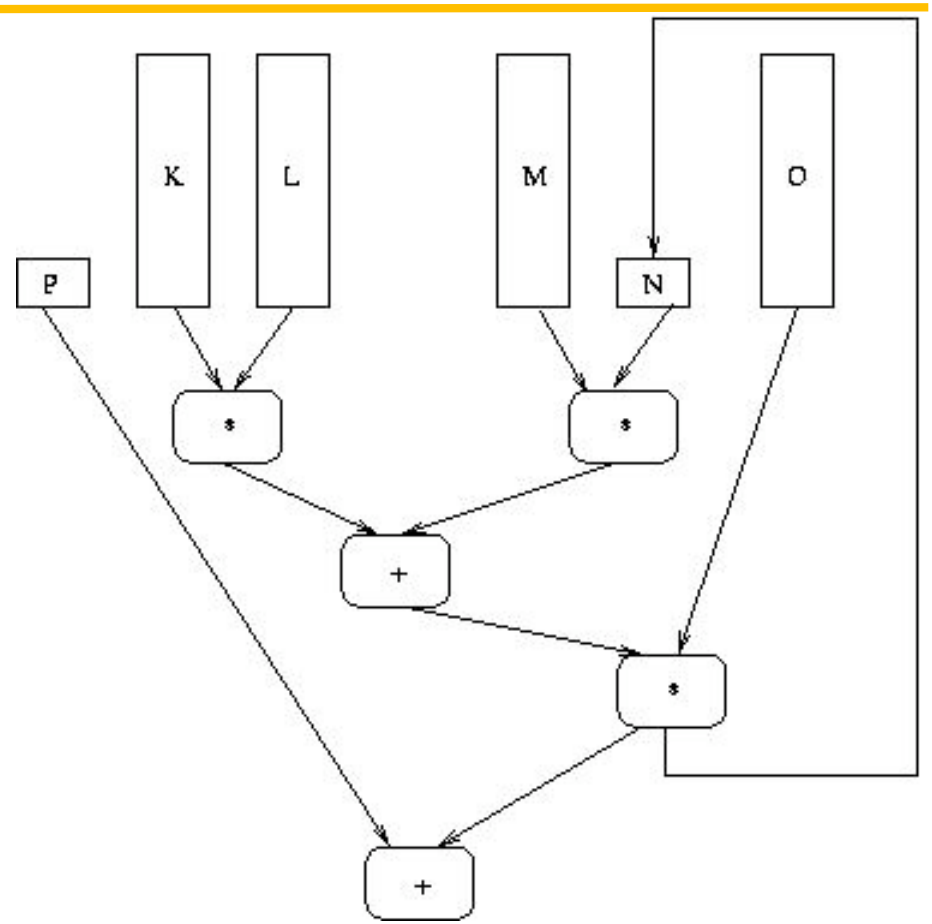
- Algorithmic/Behavioral
  - Control/Dataflow analysis
  - Dependency analysis and loop scheduling
  - High level pipelining
  - Target-independent expression-level optimizations
  - Predication
  - Operation selection
  - Memory access scheduling
  - Low level scheduling and pipelining
  - State machine synthesis
- RTL
  - Resource binding (function units, registers)
- Structural
  - Gate-level code generation
- Mapping
  - Gates are packed into (virtual) logic blocks of target FPGA
- Placement
  - Virtual logic blocks are mapped to physical logic blocks
- Routing
  - Connections between physical logic blocks assigned to routing resource on chip

Steps in Hardware Synthesis

- Predication
  - If (a) then X else Y → [(a) X; (~a) Y]
  - Eliminates control flow
  - Useful transformation for convention microprocessors also
- Operation selection
  - Size, eg. 4/8/12/14/16 … 64 bit
  - Data type, eg. [unsigned] int or float
  - Implementation, eg. Combinational, pipelined, bit serial, …
- Scheduling
  - Transform the control/data flow graph into a timed execution flow
  - Decide on operations that occur at every clock cycle
    - Function units
    - Memory access
    - Registers
  - Specify finite state machine to sequence through the sets of operations at each clock cycle
  - Synthesize pipelines for loops that can be pipelined

Example

```
n=0
p=0
for i
    n = (K[i]*L[i]+M[i]*n)*O[i]
    p = n+p
```



- Many choices for instruction level parallelism
  - multiplies/adds in parallel or sequential?
    - area/speed tradeoff
    - affects loop-level pipelining
  - Number of clock cycles to compute n & p
    - affects clock speed

Pipeline Loop Alternatives

- Pipelined: One memory for all arrays
- Initiate a new loop iteration every 4 clock cycles.
- 8 stage pipeline

Stage 0: Update array pointers; issue read of M

Stage 1: Increment i; issue read of K; save M in register

Stage 2: temp1 = M*n; issue read of L; save K in register

Stage 3: issue read of O; save L in register

Stage 4: temp2 = K*L; save O in register

Stage 5: temp3 = temp1 + temp2

Stage 6: n = temp3*O

Stage 7: p = p+n

- Pipelined: Four memories
- Initiate a new loop iteration every 2 clock cycles
- 6 stage pipeline

Stage 0: Issue reads of K, M, L, O

Stage 1: Increment i; save M, L, O in registers

Stage 2: temp1 = K*L; temp2 = M*n

Stage 3:temp3 = temp1 + temp2

Stage 4: n = temp3*O

Stage 5: p = p+n

- Combinational: Four memories
- Initiate a new loop iteration every clock cycle
- 1 stage pipeline

Stage 0: Perform all multiplies and add, store results in registers

For XC2V2000-6

153/10752 slices

5/56 multipliers

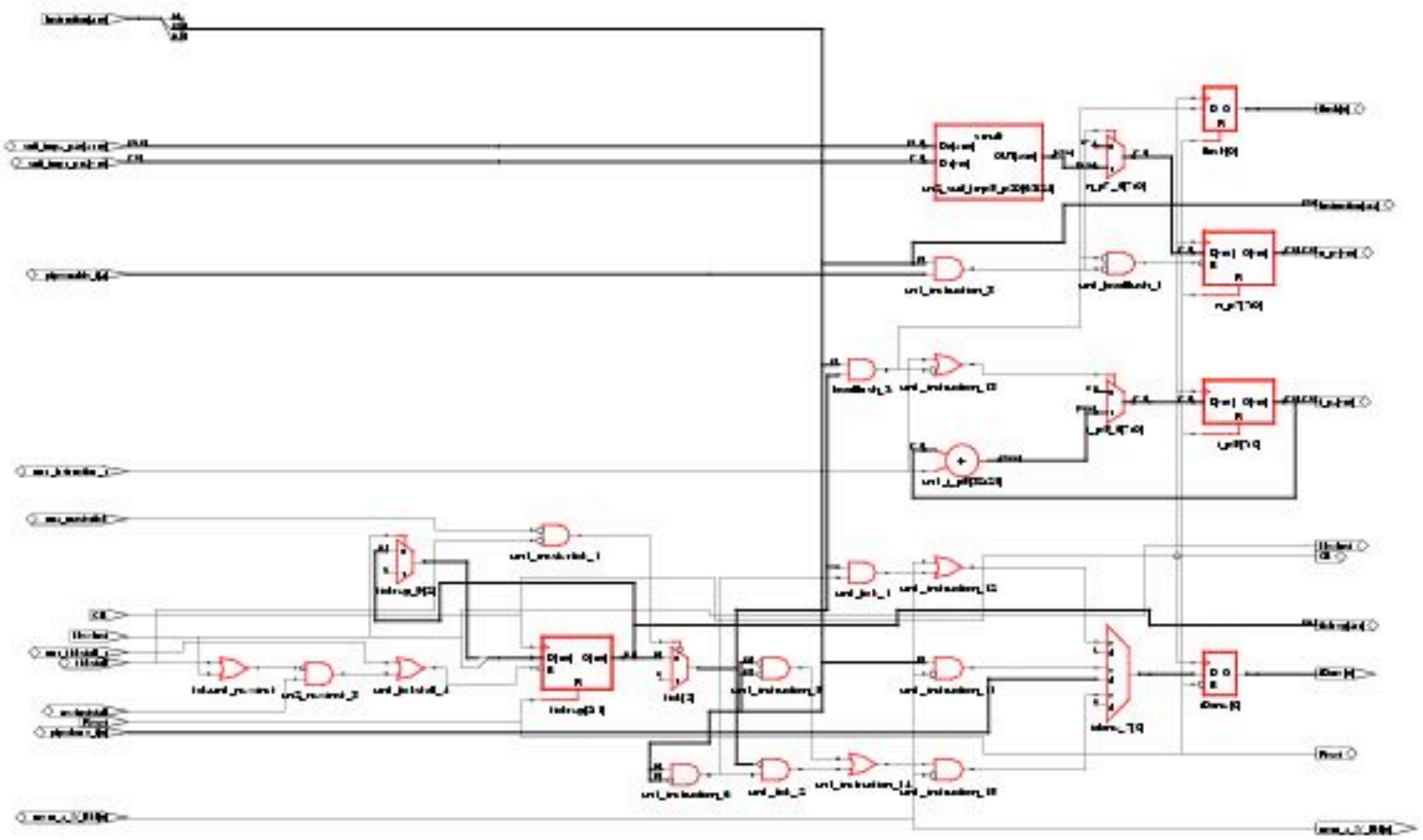
66 MHz

131/10752 slices

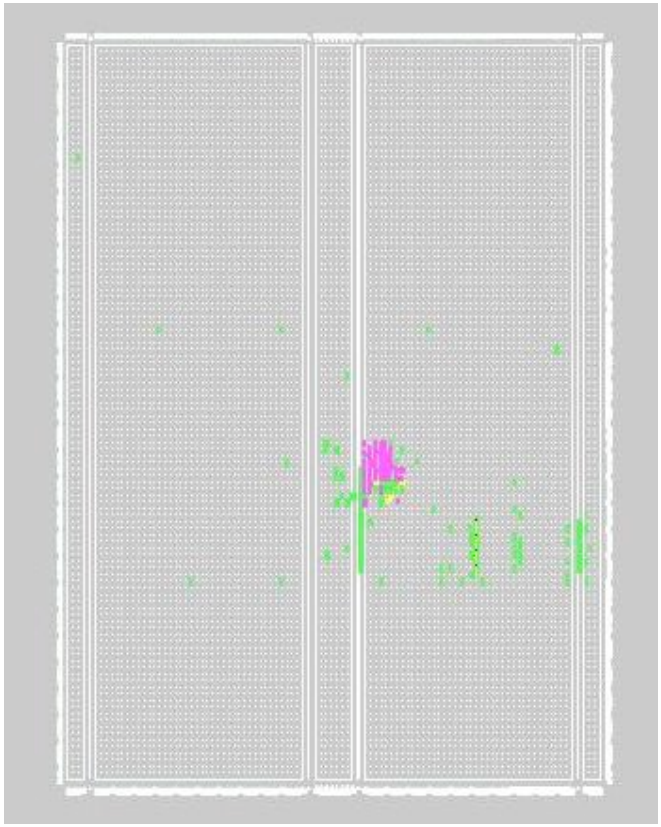
3/56 multipliers

219 MHz

Schematic View from Synthesis



Schematic View from Place&Route



Pipeline controller

Datapath: adders/multipliers/registers

Sequence controller

Device speed data version: PRODUCTION 1.118 2004-03-12.
Device utilization summary:

Number of External IOBs	442 out of 624	70%
Number of LOCed External IOBs	0 out of 442	0%
Number of MULT18X18s	3 out of 56	5%
Number of SLICEs	131 out of 10752	1%
Number of BUFGMUXs	1 out of 16	6%

The AVERAGE CONNECTION DELAY for this design is: 1.217
The MAXIMUM PIN DELAY IS: 4.558
The AVERAGE CONNECTION DELAY on the 10 WORST NETS is:4.005

Summary

- FPGAs have become system on a chip architectures, optimized to broad application classes
- Reconfigurable systems range from embedded processors to supercomputers
- Mapping algorithms onto RCs is a parallel processing problem
- Many language approaches for reconfigurable computers
- Compilers face a daunting task - extract ILP, pipeline loops, unroll, trade-off area/speed

Design Tools and Libraries

- Justin L. Tripp et. al. , Trident: An FPGA Compiler Framework for Scientific Computing
- Keith D. Underwood and K. Scott Hemmert, Implications of FPGAs for Floating-Point HPC Systems