Genesis Kernel on IXP1200

Stephen Chou
Michael Kounavis
Andrew Campbell
John Vicente
Columbia University
overview

- making network processor-based routers programmable.
- **NetBind:**
  - a binding tool for programming the data path in NPs
- **Genesis Kernel**
- “Genesis Box” testbed and status
programming a network processor-based router

- architectural constraints
- resource constraints
- performance constraints
- time scale
- binding policy/methods
- complexity
NPs and general-purpose processors
IXP1200 microengine

- 4 contexts
- 128 General Purpose Registers (GPRs) divided into Banks A and B
- 128 Address Registers
  - read and write transfer registers
  - SDRAM and SRAM
- addressing modes
  - absolute
  - context relative
programming the data path

Fine grain  Code Granularity  Coarse grain

Code Synthesis  Dynamic binding  Monolithic

Loader

Scripting

Complexity?  High  ??  Low
Flexibility?  High  High  Low
Reusability?  High  Low
dynamic binding issues

- headroom limitations
- register space and state management
- choice of the binding method
- data path admission control
- processor handoffs
- instruction store limitation
binding methods

• dispatch loop
  • MicroACE Extensions of the ACE framework in Intel IXP SDK 2.0
  • nested if-then-else on global binding state for determining the next code module

• vector table
  • table (stored in fast memory) of pointers to code modules

• code morphing
  • used by NetBind
Before Binding

After Binding
NetBind state space

argument registers (addresses: 9-12)

component #1

algorithm #1

pipeline registers (addresses: 0-8)

pipeline #1

component #2

algorithm #2

pipeline registers

pipeline #2

shared absolute registers (addresses: 13-15)

component #3

algorithm #3

pipeline #3

register space

memory space

Scratchpad

MIB, performance stats

SRAM

routing table

SDRAM

packet buffers

forwarding table

Scratchpad

MIB, performance stats

SRAM

routing table

SDRAM

packet buffers

forwarding table
binding overhead

Dynamic Binding Execution Overhead

- NetBind
- Dispatch Loop (MicroACE)
- Vector Table

Additional Exec Cycles

Number of components
throughout analysis
delay analysis

Per-Packet Execution Time

- No Binding
- NetBind
- Dispatch Loop (microACE)
- Vector Table
virtual networks on demand

virtual network

physical network
routelet: virtual router

programming environment

state

control plane

data path

binding interface base

metabus

routelet state

control unit

spawning controller

composition controller

allocation controller

datapath controller

input port

forwarding engine

output port
spawning a routelet

- Spawning controller
- Composition controller
- Allocation controller
- Datapath controller

Control plane

- Spawning
- VN Demultiplexer
- NetBind
- Admission controller
- Routing mgr
- IXP datapath

Data path

- Input port
- Forwarding engine
- Output port
- Input port
- Forwarding engine
- Output port
- Capacity arbitrator
data path specifications

- **input ports** ➔ **forwarding engines** ➔ **output ports**
- **virtual**: router specification
- **physical**: IXP1200 realization

- **components** ➔ **symbols**
- **entry point** ➔ **transport modules**
- **exit point**
- **global variable** ➔ **input argument**
dynamic binding in GK

- network processor related specification
- datapath constructor
- IXP1200 related specification
- binder
- microengines

StrongARM Core

transport modules (.tmd files)

datapath admission control
Genesis Developer Workbench
Genesis “Box” testbed
Genesis Kernel status

- **NetBind Code release**
  - www.comet.columbia.edu/genesis/netbind

- **Genesis Kernel (GK) v1.0 developed**
  - datapaths implemented on IXP1200
  - Genesis Developer Workbench front-end
  - Genesis testbed consisting of 15 IXP1200 evaluation platforms
  - Release March 2002
Genesis publications

thanks for listening