SH-X3
Flexible SuperH Multi-core for High-performance and Low-power Embedded Systems

Shinichi Shibahara\textsuperscript{1}, Masashi Takada\textsuperscript{2}, Tatsuya Kamei\textsuperscript{1}, Kiyoshi Hayase\textsuperscript{1}, Yutaka Yoshida\textsuperscript{1}, Osamu Nishii\textsuperscript{1}, Toshihiro Hattori\textsuperscript{1}

\textsuperscript{1} Renesas Technology Corp.
\textsuperscript{2} Hitachi Ltd.

Hot Chips 19, 2007/8/20
Requirement for Embedded Systems

● Trend
  ● Total scale is increasing by the introduction of advanced features.

![Mobile Phone Features](image)

- 2D / 3D
- Mail
- W-CDMA
- Camera
- Video
- Sound
- GSM
- JAVA

● Requirement
  ● High Performance (for advanced features)
  ● Small Area (for smaller gadgets)
  ● Low Power (for long duration of battery)

● Solution: On-chip Multi-processor
  ● Process technology allows to produce easily.
  ● MP can be performance and power effective.
Multi-processor Approaches for Embedded Systems

- Features
  - Application Specific: Optimize hardware & software for each system
  - Low Power: Less than 1W (in the case of battery-run)

- Approaches
  - Heterogeneous / Homogeneous AMP
    - Integration of sub-systems / Deterministic behavior
  - Homogeneous SMP
    - Relatively easy programming model / Performance oriented
  - Hybrid (Mixed system of AMP and SMP)
    - Automatic Parallelizing Compiler
SuperH Processor Core Roadmap

**SH-X**
- Superscalar 7-stage pipeline
- Released in 2003
- Products:
  - SH-Mobile3
  - SH-Navi1

**SH-X2**
- Superscalar 8-stage pipeline
- Released in 2005
- Products:
  - SH-Mobile G1, G2
  - SH-Navi2

**SH-X3**
- MP-ready (up to 4-CPU)
- Superscalar (each)
- 8-stage pipeline

- First Target: Car Information Systems

- >800MHz@90nm G
  - 6000MIPS/W@90nm LP
  - 7.2 MIPS/MHz (4-CPU)

- >800MHz@90nm G
  - 4500MIPS/W@130nm LP
  - 1.8 MIPS/MHz

- >400MHz@130nm G
  - 4500MIPS/W@130nm LP
  - 1.8 MIPS/MHz

- 5-stage pipeline up to 266MHz
SH-X3 Block Diagram

CPU #0

On-chip System Bus (SuperHyway Bus™ On-chip Interconnect)

D-LRAM: Also called XY-RAM in SH4AL-DSP
Pipeline Structure

- Eight-stage dual-issue superscalar pipeline (Inherited from SH-X2)
Specification Features

- Efficient for both SMP and AMP
  - Cache coherency (Snoop Controller) for SMP
  - Local memories (LRAM, URAM) and data transfer unit for AMP
  - Realization of hybrid MP model
- Fine power management for each CPU
  - Low-power modes according to workload (sleep, light sleep, standby etc.)
  - Flexible clock ratio (CPU Clock : System Bus Clock = m:n (m≥n), 1:n)
  - Hierarchical clock gating
- Configurable and synthesizable
  - Number of CPU (up to 4-CPU), Co-processor (DSP, FPU)
  - Cache (8KB~64KB/4way)
  - Local memory (LRAM: 4KB~128KB, URAM: 128KB~1MB)
Cache Coherency for Embedded Systems

- Problems in applying bus snooping (used in HPC servers)
  - Performance degradation by system bus occupation
  - Unnecessary power dissipation by snooping activity
  - Fixed write-cache mode: MESI (Copy-back) or ESI (Write-through)
    - MESI Fixed: HW accelerator on system bus cannot access the latest data.
    - ESI Fixed: CPU cannot run at the best performance due to store accesses.

- Solution
  - Separation of system bus and snoop bus (Reduce bus occupation)
  - Centralized coherency control by snoop controller (Reduce bus activity)
  - Support of mixed cache coherency protocol (Each CPU can select mode)
Cache Coherency Maintenance (Conventional: Bus Snooping)

All transactions appear on bus → Bus occupation, Wasting power

Cache Operation (Fill, WB, …)

Monitor All Transactions (Bus Snooping)

System Bus
Cache Coherency Maintenance (MESI Protocol)

1. Operand Access
2. Search
3. Fill Request
4. Search & Update
5. Request
6. Search & Update
7. Response
8. Response
9. Update

Snoop Bus
Reduce bus occupation
No bus monitoring

BIC: Bus I/F Controller, DAA: Duplicated Address Array
Snoop Latency Optimization (ESI Protocol)

1. WT Request
2. Search & Update (Erase the stale)
3. WT Request

Initiator Target Initiator Target

Operand Cache Cache Controller 0
Controller

Operand Cache Cache Controller 3
Controller

(3) Request (Erase the stale)

Snoop Controller

DAA DAA DAA DAA

BIC 0 Initiator

BIC 3 Initiator

System Bus

Latency

Cache Controller 0

Cache Controller 3

Snoop Controller

No need to wait for snoop response
Optimized

No affection to WT request → Execute on background
Mixed Coherency Protocol

Need to consider the latest in other CPUs

Write-through

Copy-back

Cache Controller 0

Operand Cache

Initiator Target

Controller

Operand Cache

Initiator Target

Controller

Operand Cache

Initiator Target

Copy-back

Copy-back

Merge

Dirty Data

Snoop Controller

BIC 0

Initiator

DAA (CPU0)

DAA (CPU1)

DAA (CPU2)

DAA (CPU3)

BIC 3

Initiator

WB Request

00 01 AA 03

System Bus

(1) WT Request

AA

(2) Request

(3) Response (with dirty)

00 01 02 03

Colored Variations

(1) WT Request

AA

(2) Request

(3) Response (with dirty)

00 01 02 03
Difficulty of SMP OS Development

- Cache operation after process migration (Caused by time sharing processing)
  - Conventional measure
    - Flushing cache entries, accessed before, via inter-processor interrupt after process releases memory or virtual-physical address map is changed.

- Synonym problem (Caused by more than one virtual-physical address maps)
  - Conventional measure
    - Flushing the cache of synonym page during page allocation
    - Preventing the synonym occurrence by using page coloring

**Problem in conventional measure**

- Complicated to implement software
- Large software overhead

**Solution**

- Broadcast of operand cache operating instructions (OCBI, OCBP, OCBWB)
- Hardware implementation of synonym detection and eviction
Operand Cache Operating Instructions (Conventional)

Conventional Specification
Operate only my own cache line → Cannot operate other CPUs’ cache line
→ Need inter-processor interrupt to operate others’

Example: Process Migration

Process Migration

CPU #0

MEM

CPU #3

MEM

Snoop Controller

System Bus

Process X

MOV.L Rm, @Rn

VOID

OCBWB @Rn

Not executed yet

Cannot write back the dirty line

D$

Dirty

D$

Cannot access the latest

Memory

DMAC

Memory Access after Process X

OCBWB: Write-back Cache Block
Broadcast of Operand Cache Operating Instructions

Extended Specification
Operate all CPUs’ cache by broadcast → No need inter-processor interrupt to operate
→ Reduce software overhead of using interrupt

Example: Process Migration

Broadcast via Snoop Controller

CPU #0

D$

Dirty

CPU #3

D$

System Bus

Write Back

Available the latest

Memory

DMAC

Process X

MOV.L Rm, @Rn

OCBWB @Rn

Executed

Memory Access after Process X

OCBWB: Write-back Cache Block
Synonym Detection and Eviction (In the case of 4KB/Page)

VPN[31:12] → PPN[31:12]

D$: 32KB/4way
(Virtual Index - Physical Tag)

V-Index [12:5]  P-Tag[31:10]

DAA
(Physical Index - Physical Tag)


1. Read Miss Adr(A) 0x00

2. Read Miss Adr(B) 0x00
   Suppose
   Physical Address (A) 0x80
   = Physical Address (B)
   Synonym (Differs in VPN[12])

3. Request from SNC 0x00

Suppose
Physical Address (A) 0x80
= Physical Address (B)
Synonym (Differs in VPN[12])

Purge Request from SNC

SNC: SNoop Controller
**RP1: Experimental Chip**

**Process Technology** 90-nm, 8-layer, Triple-Vth, Generic CMOS, 1.0V

**Area** 3.88mm² (Each CPU excluding all memories), 7.28mm² (Each CPU)

**I/D Cache** 32KB/4way set-associative (Each)

**Local Memory** I-LRAM 8KB, D-LRAM 16KB, URAM 128KB (Each CPU)

**Performance** 1.8 MIPS/MHz/CPU (Dhrystone 2.1)
4320 MIPS @ 600MHz (4-CPU Total)

**Power Consumption** 0.6 mW/MHz/CPU @ 600MHz
Application of Synonym-related Function to SMP OS

- **OS Enhancement for SMP**
  - Measurement: When kernel detects a synonym page, it flushes all entries of the page.

- **Experiment (On evaluation board)**
  - Enhanced for hardware implementation of synonym-related function
  - Executed shell command “find” for each CPU in parallel (Whole is stored in DDR)
    - Not Enhanced (Using original synonym measurement)
    - **Enhanced**

<table>
<thead>
<tr>
<th>CPU</th>
<th>Execution Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU0</td>
<td>31.88</td>
</tr>
<tr>
<td>CPU1</td>
<td>30.34</td>
</tr>
<tr>
<td>CPU2</td>
<td>29.31</td>
</tr>
<tr>
<td>CPU3</td>
<td>30.46</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CPU</th>
<th>Execution Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU0</td>
<td>14.39</td>
</tr>
<tr>
<td>CPU1</td>
<td>14.36</td>
</tr>
<tr>
<td>CPU2</td>
<td>14.10</td>
</tr>
<tr>
<td>CPU3</td>
<td>14.90</td>
</tr>
</tbody>
</table>

53% Performance Improvement
Summary

- SH-X3: SuperH multi-core for high-performance and low-power systems
  - Efficient for both SMP and AMP

- Specification features for cache coherency
  - Separation of system bus and snoop bus
  - Centralized cache coherency by snoop controller
  - Support of mixed cache coherency protocol

- Specification features for SMP OS development
  - Broadcast of operand cache operating instructions (OCBI, OCBP, OCBWB)
  - Hardware implementation for synonym problem (53% performance improved)

Acknowledgement

This work was supported by NEDO (New Energy and Industrial Technology Development Organization) P03022, a joint project of Renesas Technology Corp., Hitachi Ltd., and Waseda University.
Backup Slides
Synonym Detection and Eviction (In the case of 4KB/Page)

VPN[31:12] → PPN[31:12]

D$: 32KB/4way
(Virtual Index - Physical Tag)

V-Index [12:5] P-Tag[31:10]

Way X

1. Read Miss Adr(A)
   0x00
   A

2. Read Miss Adr(B)
   0x00
   A
   0x80
   B

3. Request from SNC
   0x00
   A
   0x80
   B

DAA
(Physical Index - Physical Tag)


Way X

0x80
A
0

0x00
A
B
0 1

Overwrite

Must keep 1:1 correspondence

Different Detection by SNC

Purge Request from SNC

SNC: SNoop Controller
Synonym Detection and Eviction (In the case of 4KB/Page)

VPN[31:12] → PPN[31:12]

D$: 32KB/4way (Virtual Index - Physical Tag)


1. Read Miss Adr(A)

| P-Tag[31:10] | 0x80 | A | 0 |

2. Read Miss Adr(B)

| Overwrite | P-Tag[31:10] | 0x80 | A | B | 0 | 1 |

Different Detection by Cache Controller

Must keep 1:1 correspondence

3. Request from SNC

| P-Tag[31:10] | 0x80 | B | 1 |

Invalidate Request to SNC

SNC: SNoop Controller
Application of Synonym-related Function to SMP OS

- **OS Enhancement for SMP**
  - Measurement: When kernel detects a synonym page, it flushes all entries of the page.

- **Experiment (On evaluation board)**
  - Enhanced for hardware implementation of synonym-related function
  - Executed shell command “find” for each CPU in parallel (Whole is stored in DDR)
    - Not Enhanced (Using original synonym measurement)
      
<table>
<thead>
<tr>
<th>CPU</th>
<th>Activation Frequency of Function</th>
<th>Execution Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU0</td>
<td>0.06%</td>
<td>31.88</td>
</tr>
<tr>
<td>CPU1</td>
<td>0.05%</td>
<td>30.34</td>
</tr>
<tr>
<td>CPU2</td>
<td>0.05%</td>
<td>29.31</td>
</tr>
<tr>
<td>CPU3</td>
<td>0.05%</td>
<td>30.46</td>
</tr>
</tbody>
</table>

- **Enhanced**

<table>
<thead>
<tr>
<th>CPU</th>
<th>Activation Frequency of Function</th>
<th>Execution Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU0</td>
<td>0.16%</td>
<td>14.39</td>
</tr>
<tr>
<td>CPU1</td>
<td>0.13%</td>
<td>14.36</td>
</tr>
<tr>
<td>CPU2</td>
<td>0.17%</td>
<td>14.10</td>
</tr>
<tr>
<td>CPU3</td>
<td>0.13%</td>
<td>14.90</td>
</tr>
</tbody>
</table>

**Performance Improvement:** 53%