Novel Task Migration Framework on Configurable Heterogeneous MPSoC Platforms

Hao Shen     Frédéric Pétrot*
System Level Synthesis Group, TIMA Laboratory
CNRS/Grenoble INP/UJF
43, Avenue Félix Viallet, 38031, Grenoble, France
hao.shen@imag.fr     frederic.petrot@imag.fr
Outline

Introduction
Task Migration Definition and Implementation
Task Migration Algorithms
Experimental Results
Conclusion and Future Works
Trends (Power and Cost Constraints for Embedded Systems)

**Multiple Processor System-on-Chip (MPSoC)**
- Provide high Thread Level Parallelism (TLP)
- Provide high performance
- Power consumption and cost advantages

**Configurable processor**
- Different extended instruction set for different processors
- Dedicated compiler for each extended instruction set
- Provide high Instruction Level Parallelism (ILP)
- Power consumption and cost advantages

**Heterogeneity**
- Different configurable processor in a MPSoC for different kinds of applications
- Power consumption and cost advantages
## Configurable Processors

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Specific Instruction-set Processors (ASIP)</td>
<td>Configuration for one application or a group of applications</td>
</tr>
<tr>
<td>Provide high Instruction Level Parallelism (ILP)</td>
<td>SIMD instructions, MIMD instructions, specific instructions (such as Multiply-accumulate)</td>
</tr>
<tr>
<td>Configurable processors VS. RISC processors</td>
<td>Provide higher performance</td>
</tr>
<tr>
<td>Configurable processors VS. ASIC</td>
<td>More flexibility and shorter time-to-market</td>
</tr>
</tbody>
</table>
One Existing Heterogeneous Multiple Configurable SoC Example

General printer solution
6 heterogeneous Xtensa processors
Fixed task mapping
Communication with FIFOs
What is the Problem of Configurable Processors?

**Instruction Set**
- Core instruction set
- Extended instruction set

**Register File**
- Core registers
- Extended registers

Task

Processor A 0  Processor A 1  Processor B 0  Processor B 1

Global Communication

Shared Memory
Contribution of This Work

Task migration framework

- Support heterogeneous multiple configurable processor SoC
- Present realization details of this framework
- Add some formal description

Task migration algorithms

- Compare the efficiency of each algorithm
- Compare the migration cost of each algorithm
Outline

- Introduction
- Task Migration Theory and Implementation
- Task Migration Algorithms
- Experimental Results
- Conclusion and Future Works
Instruction Set Relationship

$S = \{S_{\text{core}}, S_A, S_B, S_C, S_D\}$

$S_{\text{core}} \subseteq S_A, \ldots, S_{\text{core}} \subseteq S_D, S_A \subseteq S_C, S_A \subseteq S_D, S_C \subseteq S_D, S_B \subseteq S_D$
Relationship Between Tasks and Processors

Task 1
Set Core (Score)

Task 2
Set A (SA)

Task 3
Set B (SB)

Task 4
Set C (SC)

Task 5
Set D (SD)

Processor 1
Set A (SA)

Processor 2
Set B (SB)

Processor 3
Set C (SC)

Processor 4
Set D (SD)
Instruction Set Identification

Assign ID for each CPU type (CPU_ISA_ID)
- Indicate the instruction set which it realizes

Assign ID for each task type (TASK_ISA_ID)
- Indicate the instruction set which it uses

Use bit operation to accelerate scheduling
- Use one bit to represent a standalone instruction set
- Test compatibility by using bit operations
- Save storage space for the OS realization
## Scheduler Realization

### Instruction set compatibility

$\bullet (\text{CPU\_ISA\_ID} \mid \text{TASK\_ISA\_ID}) == \text{CPU\_ISA\_ID}$

<table>
<thead>
<tr>
<th>Set</th>
<th>CPU_ISA_ID</th>
<th>Compatible Tasks</th>
<th>TASK_ISA_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core</td>
<td>0x0000</td>
<td>Core</td>
<td>0x0000</td>
</tr>
<tr>
<td>A</td>
<td>0x0001</td>
<td>Core and A</td>
<td>0x0000, 0x0001</td>
</tr>
<tr>
<td>B</td>
<td>0x0010</td>
<td>Core and B</td>
<td>0x0000, 0x0010</td>
</tr>
<tr>
<td>C</td>
<td>0x0101</td>
<td>Core, A and C</td>
<td>0x0000, 0x0001, 0x0101</td>
</tr>
<tr>
<td>D</td>
<td>0x1111</td>
<td>Core, A, B, C and D</td>
<td>0x0000, 0x001, 0x0101, 0x1111</td>
</tr>
</tbody>
</table>
Outline

- Introduction
- Task Migration Definition and Implementation
- Task Migration Algorithms
- Experimental Results
- Conclusion and Future Works
First Match First Serve Algorithm

For $j=1$ to $|T_{queue}|$ in the FIFO order

If $c(T_j, P_i) \in C$ // $T_j$ is compatible with $P_i$

Choose the task $T_j$
Most Compatible Algorithm

For all $T_j \in T_{queue}$

If $c(T_j, P_i) \in C$ // $T_j$ is compatible with $P_i$

$T_{candidate} = T_{candidate} \cup T_j$ // Add $T_j$ to candidate set

If $T_{candidate} \neq \emptyset$

choose the task $T = \min(D(T_j \in T_{candidate}, P_i))$
Outline

Introduction
Task Migration Definition and Implementation
Task Migration Algorithms
Experimental Results
Conclusion and Future Works
Motion-JPEG Decoder Example and the Optimization
Heterogeneous MPSoC Architecture

- IDCT
- DEMUX
- VLD
- LIBU
- IQ/ZZ
- CONV
- Local Sched
- Local Task Scheduling
- Simple Xtensa Bus
- TG
- Memory
- TTYs
- RAMDAC

Core ISA
Extended ISA for IDCT
Extended ISA for CONV
## Performance and Cost Advantages

<table>
<thead>
<tr>
<th></th>
<th>Fixed Task Assignment</th>
<th>FMFS Algorithm</th>
<th>Most Comp Algorithm</th>
<th>SMP Task Scheduling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame/s</td>
<td>1.44</td>
<td>2.88</td>
<td>2.70</td>
<td>2.88</td>
</tr>
<tr>
<td>CPUs Gate number</td>
<td>341,773</td>
<td>341,773</td>
<td>341,773</td>
<td>611,295</td>
</tr>
<tr>
<td>Perf/Cost</td>
<td>0.50</td>
<td>1.00</td>
<td>0.94</td>
<td>0.56</td>
</tr>
</tbody>
</table>

### Heterogeneous architecture VS. homogeneous architecture

- May achieve the same performance
- May need smaller chip size (higher performance/cost ratio)

### Task migration VS. fixed task assignment

- Need the same chip size
- Provide higher system performance (shorten the processor waiting time)

### Different task migration algorithms have different performance

- FMFS requires less computation resource during migration
Outline

Introduction
Task Migration Definition and Implementation
Task Migration Algorithms
Experimental Results
Conclusion and Future Works
Conclusion and Future Works

A task migration framework

• Support configurable processors
• Support heterogeneous MPSoC architectures
• Support several migration algorithms

Future works

• Formalization this framework
• More complex case studies (benchmarks)
Questions & Answers

Hao.Shen@imag.fr  Frederic.Petrot@imag.fr
Priority Based Most Compatible Algorithm

\[
\begin{align*}
\text{For } k = 1 \text{ to } n & \text{ // } n \text{ queues with different priorities} \\
\text{Forall } T_j \in T_{\text{queue}} & \\
\text{If } c(T_j, P_i) \in C & \text{ // } T_j \text{ is compatible with } P_i \\
T_{\text{candidate}} = T_{\text{candidate}} \cup T_j & \\
\text{If } T_{\text{candidate}} \neq \emptyset & \\
\text{choose the task } T= \min(D(T_j \in T_{\text{candidate}}, P_i)) &
\end{align*}
\]
Time

• 20’ presentation
• 5’ question