Multithreaded Coprocessor Interface for Dual-Core Multimedia SoC

Student: Chih-Hung Cho
Advisor: Prof. Chih-Wei Liu
VLSI Signal Processing Group, DEE, NCTU
Outline

- Introduction
- Multithreaded Coprocessor Interface
- Simulation & Implementation Results
Introduction

- Dual-core/Multi-core SoC is the possible solution for modern mobile multimedia systems

- Task divergence in most embedded systems drives the heterogeneous computing platform
  - Control-oriented task vs. computation-intensive task
  - **RISC + DSP, “play the right thing in the right place”**[1]

- Example: TI OMAP
  - RISC: ARM9
  - DSP: TI C’5x

---

Low DSP Utilization Problem (1/2)

- Peak vs. delivered performance gap increasing
  - Instruction latency & data dependency
  - Limits to ILP seem to limit to 3~6-issue for practical options
  - **DSP utilization is below 50%** [2]
    - IPC (Inter-Processor Communication) latency
    - Memory latency
    - Pipeline latency

Performance beyond single thread ILP
- Explicitly parallel (DLP or TLP)
- TLP could be more cost-effective than to exploit ILP
- IMT (Fine-Grained) vs. BMT (Coarse-Grained)

Solutions to Pipeline Latency

- **Forwarding path**
  - Overheads on area, power, and even critical path
  - Non-causal path existence

- **Software optimization**
  - Overhead on code size

- **Hardware multithreading**
  - Explicit TLP exploration
    - Multiple thread context
    - Hardware-supported thread switch mechanism
Low DSP Utilization Problem (2/2)

- Inter-processor communication (IPC)
  - Become more complicated in multi-core/multithreaded computing model

Conventional

Enea’s OSEck (RTOS) provides full support for StarCore’s SC1000 families of DSP core
Outline

- Introduction
- Multithreaded Coprocessor Interface
- Implementation & Simulation Results
DSP Core

- 4 threads IMT
  - MIPS compatible ISA
  - 4 program counters
  - 32 32-bit register files (each thread)
  - 5-stage pipeline
Multiple tasks tend to use DSP concurrently

\[\Rightarrow\] DSP task management is required
Simulation Model: Dataflow Process Network

- Multimedia applications can be described as FIFO-communicated processes

- For simplicity, we assume *each process has only a single input & a single output (SISO)*
JPEG Encoding Example

- 4 processes

- RGB-to-YUV color space transform (CST)
- Discrete cosine transform (DCT)
- Quantization (Q)
- Zero run-length & variable-length coding (VLC)

Processes are mapped on a single (multithreaded) DSP

- Each process is assigned a unique priority
- FIFO channels (except those for I/O processes) are implemented on DSP local memory
- A process notifies its descendent that the data are ready, when it completes its computations
Simulation Platform (Concept)

MPU (ARM926) -> VIC -> MPU2DSP -> Shared Memory

0x0000_0000

0x0400_0000

0x1000_0000

DSP (ARM926) -> VIC -> DSP2MPU

0x0000_0000

0x0400_0000

0x1000_0000
Task Management on MPU

IPC Overhead ↑
Task Management on DSP

IPC Overhead ↓, but ….
Outline

- Introduction
- Multithreaded Coprocessor Interface with Hardware Queues
- Implementation & Simulation Results
Experiment Framework

- Prototyping on ARM Versatile
  - Multithreaded DSP core on Xilinx Virtex II-6000 (@35MHz)
  - Host processor: ARM926 @210MHz
  - AMBA AHB @35MHz

- Target application
  - JPEG encoding
    - 320*240 Lena image

- 3 cases
  - Case1: Software Process Management on Host
  - Case2: Software Process Management on DSP
Case I & Case II

Computation kernel

Interface

ARM

In
CST
Out
VLC
DCT
Q

In page #0
Out page #0
PC
Initiate #0

Thread #0

In page #7
Out page #7
PC
Initiate #7

Thread #7

Computation kernel

Interface

ARM

In
Out

In page #1
Out page #1
PC
Initiate #1

Thread #1

In page #7
Out page #7
PC
Initiate #7

Thread #7

17/27
Simulation Results

- Performance comparison
  - DSP utilization evaluation
  - DSP idle time comparison

- Q&A
  - Improvement?