## **Compiler Assisted Energy Reduction Techniques for Embedded Processors**



## Lovic Gauthier and Tohru Ishihara Kyushu University / System LSI Research Center



#### Context

Memory accesses: major source of energy consumption in embedded systems

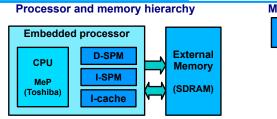
SDRAM

I-SPM

D-SPM

SDRAM

## Target system and memory



#### Memory mapping Kinds of memories

External memory (here, SDRAM):

- Large content
- Slow and energy consuming

#### Cache:

- Small content
- Fast and energy efficient but extra lookup cost in delay and energy
- Implicit to the software but low determinism

#### Scratch-pad memory (SPM):

- Small content
- Faster and more energy efficient than cache (ration for MeP ~ 25%)
- Explicit to the software but high determinism

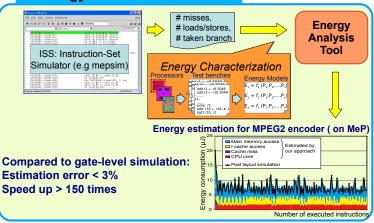
#### Contribution

## A set of tools for optimizing the usage of the SPM in order to reduce the energy consumption:

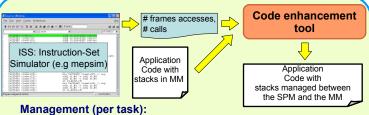
- 1. Tool for fast energy characterization of the processor running an application
- 2. Tool for modifying the application code to manage the tasks' stacks between the SPM and the external memory (MM)
- 3. Tool and additional OS context switch procedure for sharing the SPM among the tasks of a multi-task system

I-cache

### 1. Energy characterization tool



## 2. Management of the stack between SPM and MM



- Two sub-stacks instead of one, one in the SPM and one in the MM
  - Frames in SPM when accessed a lot
  - Top of SPM frame copied to MM when not accessed
- Operations for management: based on the stack pointer (SP)
  - Inserted before and after call instructions (decision ILP-based) Warp: move SP to SPM top, Unwarp: move SP to MM top

  - Store: copy part of SPM top to MM, Load: copy part of MM top to SPM Examples of the proposed stack management

SPM

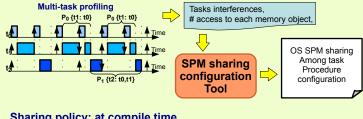
MM

SPM

## SPM Fr. 2<sub>1</sub> MM MM MM

Add Fr

# 3. Multi-task sharing of the SPM

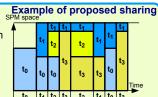


#### Sharing policy: at compile time

- Assign to each task a block in the SPM
  - Blocks can overlap
  - Blocks can contain static memory objects or stacks (sub-stacks)
- Optimization (ILP-based):
  - Select memory objects to place in blocks
  - Select size of SPM sub-stacks if stack management in SPM also used
  - Maximize blocks sizes
  - Minimize overlaps between blocks whose tasks can interfere at run time

### Sharing policy: at run time

- Our new context switch procedure:
  - Saves to the MM the parts of the SPM blocks in overlap with the block of the next active task
  - Loads the parts of the block of the next active task with are not present in the SPM



MM

#### Results

69% of energy consumption reduction for data accesses with 1Kb D-SPM

63% of energy consumption reduction for code accesses with 8Kb I-SPM (compared to I-cache)

