DarkMem: Fine-Grained Power Management of Local Memories for Accelerators in Embedded Systems

Christian Pilato
Università della Svizzera italiana, Lugano, Switzerland

Luca P. Carloni
Columbia University, New York, NY, USA
Hardware Accelerators in SoCs

• Key elements for energy-efficient high performance [DAC15]
  • Specialized microarchitecture for both computation and storage

• Almost 45% of power consumption is due to SRAM static power
  • Industrial 32nm CMOS technology

• No specific solutions for SRAM power management
  • Dual-rail SRAMs not sufficiently exploited

Embedded Scalable Platforms

- **Flexible architecture** for the integration of heterogeneous components
  - *Protocol & Shell paradigm* and *scalable communication infrastructure*
- **System-level design methodology** supported by
  - a mix of commercial and in-house CAD tools
  - a growing library of reusable IP blocks

---

Private Local Memories

- Dedicated local memories for storing part of the data
- Application-specific microarchitecture for fixed-latency accesses
- Alternative implementations with block partitioning [TCAD17]

How to Dynamically Control the Banks

• A scenario is a given configuration of the accelerator to execute a specific problem instance
  • E.g.: processing images of different size

• PLM units must be sized for the worst-case scenario, but they may not be entirely used in all scenarios
  • Possibility of fine-grained power savings

Let us assume an accelerator that can be executed in two scenarios (S1 and S2) with a 50% probability
Scenario-based Optimization

• Each accelerator can turn off the banks that are not entirely used in the current scenario

• Design-time partitioning of the banks to maximize the ones that are power gated

SRAM banks are always active even when partially used

8x32 SRAMs  1,024x32 SRAMs
Workload-based Optimization

• The system conditions can alter the execution dynamics
  • E.g.: System congestion when communicating with the external memory

• **Dynamic control of the logic/cell power gating** based on the execution phases
  • Three operating modes: *active, idle, deep-sleep*

![Diagram of load, compute, and store phases]

- **Load**:
  - A0 | A1 | A0 | A1

- **Compute**:
  - A0 → B0, A1 → B1, A0 → B0

- **Store**:
  - B0, B1

(b) System congestion can delay the data transfers

Unused banks are always power gated
DarkMem Architecture

• Each PLM unit is extended with power-control logic
  • **SMC** identifies the current execution scenario (based on the register values)
  • **OMC** manages the SRAM operating modes (based on signals from the accelerator logic)

• Fine-grained control of each SRAM bank through its power pins (PGL and PGM)
Scenario Memory Controller

- Analyzes the configuration registers (provided by the user with memory-mapped operations)
  - In each scenario, only the used banks are kept active, while the others are power gated
- Mask is one input of the OR gate for each power pin
Operating Mode Controller

- FSM to manage the transitions among operating modes
  - Latency-insensitive protocol with the accelerator logic so that no operations are performed during the transitions

- The supply voltage can be also reduced to DRV
  - Additional power savings in deep-sleep mode

- Resulting values are the other input of the OR gate

```
always @ (posedge clock)
begin
  ....
  case (state)
  ACTIVE:
    begin
      if (valid = 1'b1 && mode = DS) begin
        next_state = WAIT_DEEPSLEEP;
        PGL = 1'b1;
      end
    end
  endcase
end
```
DarkMem Methodology

• **HLS-based methodology** to generate:
  • Accelerator Logic: DarkMem API to specify operating modes of the data structures directly in SystemC
  • DarkMem units: Extension to **MNEMOSYNE** for multi-bank configuration and power-control logic for each PLM unit

• Additional information on the execution scenario
  • Estimated by the designer
  • Always possible to generate a feasible configuration

[MNEMOSYNE]: [http://github.com/chrpilat/mnemosyne](http://github.com/chrpilat/mnemosyne)
Determining the Bank Configuration

- **ILP formulation** to determine the number and type of banks for each PLM unit, based on:
  - List of scenarios and frequency of execution
  - Data to be stored (bitwidth and number of words) in each scenario
  - List of available memory IPs and corresponding active/gated static power configurations

\[
PLM_{\text{static}} = \sum_{s \in S} (PLM_{\text{static}}^{s} \cdot freq(s))
\]

- Used to determine the banks and accordingly configure the SMC modules to generate the proper masks
Experimental Results

• We improved the design of eight accelerators
  • SystemC specification extended with DarkMem API
  • MNEMOSYNE extended with generation of the DarkMem units

• Industrial 32nm CMOS technology at 1GHz
  • Cadence C-to-Silicon for HLS

• Memory library with 18 dual-rail SRAMs
  • Three variants (STD, LP, ULP): customized to have different power-gating characteristics (e.g., different leakage power in the deep-sleep mode)
Evaluation Methodology

- Logic synthesis and gate-level simulation to generate performance results and accurate SAIF backannotations

Different transition latencies to evaluate performance overhead

Accurate power analysis when varying the memory library

Logic synthesis to evaluate controllers overhead
Impact of Single Optimizations

- Reference designs are the ones without power-related optimizations

- Performance overhead is minimal (less than 1%)
Combined Results

- SRAM static power can be reduced up to 60%
  - On average, the total power is reduced by about 18%
Concluding Remarks

• Complete approach for fine-grained power management of accelerator’s SRAMs
  • DarkMem architecture identifies the execution scenario and dynamically varies the operating modes of the banks
  • DarkMem methodology automatically generates such architecture within a commercial HLS flow

• Significant power savings with almost no performance/area overhead

• Future Work: Explore the combination of this approach with processor-oriented solutions to extend the last-level cache
Questions?

Christian Pilato, USI Lugano

christian.pilato@usi.ch