Replacement Policies for a Function-based Instruction Memory: A Quantification of the Impact on Hardware Complexity and WCET Estimates

Stefan Metzlaff and Theo Ungerer

Department of Computer Science, University of Augsburg, Germany

ECRTS12
24th Euromicro Conference on Real-Time Systems
July 12, 2012
Motivation

- Safety-critical hard real-time systems
  - No deadline miss tolerable
  - System schedule requires safe and tight WCET bound per task
- Determination of the timing of the whole system, including
  - the executed software (task with all input data sets, OS)
  - the underlying hardware (CPU, memory hierarchy, bus, etc.)
- Speed gap between processor and main memory
  - Usage of fast, but small on-chip memories
Memory hierarchy has a major contribution to the system’s predictability and worst case performance.

- Approaches for on-chip memories in hard real-time systems
  - Scratchpads with fixed content
  - SW-managed scratchpads
  - Cache locking
  - Cache analysis
  - **Function-based instruction memories**
Function-Based Instruction Memories for Real-Time Systems

- Load a function completely into on-chip memory before executing
- Predictable and instantaneous execution of functions
- Examples: Method-Cache [Schoebel 04] and D–ISP

Examination of the impact of different replacement policies on WCET estimate and hardware complexity
FIFO replacement policy

- Keeping recently loaded functions
  - Eviction of the oldest function
- Implementable in hardware with low complexity
  - Queue with write and eviction pointer
Replacement Policies for Function-Based Memories

LRU replacement policy

► Keeping frequently accessed functions
  ▶ Eviction of the least recently used function
  ▶ Order depends on access history
► Too complex to implement in hardware, because
  1. functions of different size
  2. reordering of memory content
  3. memory fragmentation
Stack-based replacement policy

- Keep active branch of call tree
  - Definitely accessed again
  - At the expense of sibling function
  - Eviction of the function with largest stack distance

- Possible hardware implementation
  - Double ended queue with different write and eviction pointers for call and return
Dynamic Function-Based Instruction Scratchpad (D–ISP)

- Using function-based instruction memory D–ISP for comparison of the replacement policies

- Two operation phases: load function & execute function

- Dynamic content management
  - Is aware of D–ISP content
  - Load of complete function from off-chip memory
  - A function is always in in memory before its execution

- Instruction memory access
  - All fetches directed to D–ISP
  - Always hit for any fetch
Quantification of the impact of the replacement policy on WCET estimate

- Static WCET analysis tool
  - Dual-issue in-order processor
  - Timing model of D–ISP with FIFO, Stack-based, and LRU replacement policy
  - D–ISP analysis using collective semantics
  - No on-chip data memory
D–ISP Replacement Policy
Comparison for Sha

- Benchmark: Sha
- FIFO with high eviction rate for small memory sizes
- Stack-based with few evictions for small sizes, but non-optimal behavior for large sizes

![Graph showing memory size normalized to size of largest function and normalized WCET estimate for D–ISP FIFO, D–ISP LRU, and D–ISP STACK]
Hardware Estimation for D–ISP Replacement Policies

- Comparison of the hardware effort
- VHDL implementation of D–ISP controller with FIFO and stack-based replacement policy
- Synthesized for Altera Stratix II FPGA

<table>
<thead>
<tr>
<th>Replacement Policy</th>
<th>ALUTs</th>
<th>Registers</th>
<th>Max. f</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIFO</td>
<td>809</td>
<td>613</td>
<td>103.17 MHz</td>
</tr>
<tr>
<td>Stack-based</td>
<td>995</td>
<td>643</td>
<td>101.21 MHz</td>
</tr>
<tr>
<td>Overhead of stack-based</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>compared to FIFO (in %)</td>
<td>+23.0%</td>
<td>+4.9%</td>
<td>-1.9%</td>
</tr>
</tbody>
</table>

- Additional logic and register usage caused by different operation modes for call and return and their maintenance
Conclusions

- Function-based instruction memory for HRT systems and the implementation of different replacement policies
- Comparison regarding WCET estimate & hardware effort
  - LRU: usually best WCET estimates, but not implementable
  - FIFO: lowest implementation cost, but high WCET estimates
  - Stack-based: with additional hardware effort than FIFO, WCET estimates comparable to LRU
- Future work
  - Development of scalable analysis techniques for function-based replacement policies