Rapid System-Level Performance Evaluation and Optimization for Application Mapping onto SoC Architectures

Sumit Mohanty and Viktor K. Prasanna @ University of Southern California
The 15th Annual IEEE International ASIC/SOC Conference, 2002

Presented by Yang Qu, VTT Electronics
OUTLINES

• Introductions
  • SoC architectures integrate heterogeneous components such as GPP, DSP, ASIC, reconfigurable logic.
  • huge design space
    • dynamic voltage scaling, reconfiguration, multiple clock domain, …
• Problems
  • How to efficiently perform the system-level performance estimation to support the design space exploration?
    • evaluate design decisions, find bottleneck and other optimization possibilities at early phase of the design!
    • accuracy <= abstraction => simulation speed
• Solutions
  • A Generic Model (GenM): identifies key architecture features
  • A High-level Performance Estimator (HiPerE): evaluate performance at the system level
• Case Studies
Generic Model (1)

- Target is a reconfigurable SoC:
  - a processor + a reconfigurable logic (RL) + a memory
  - use state models and cost models to define the design space
- Model each component with a number of discrete states
  - \( V: [V_0 \ldots V_{v-1}] \) states for the processor (voltage)
  - \( C: [C_0 \ldots C_{c-1}] \) states for the RL (reconfiguration)
  - \( M: [M_0 \ldots M_{m-1}] \) states for the memory (power)
  - \( S: V \times C \times M \) (total system states)
- Parameters of energy/time costs of state transitions
  - \( (V_{ij}^t, V_{ij}^e), (C_{ij}^t, C_{ij}^e), (M_{ij}^t, M_{ij}^e) \)
- Parameters associated with tasks, \( T \)
  - \( t_{ij}, e_{ij} : \) time/energy of task \( T_i \) mapped onto processor state \( V_j \)
  - \( t'_{ij}, e'_{ij} : \) time/energy of task \( T_i \) mapped onto RL state \( C_j \)
Generic Model (2)

- The GenM can be used for
  - a) rapid performance estimation (energy + latency)
  - b) development of efficient application designs using high-level abstraction
  - c) development of optimization techniques for mapping and scheduling of the tasks onto SoC architectures

- An example of optimization technique
  - A linear pipeline of tasks with the assumption that at any time only one task can be executed either on the processor or the RL
  - The goal is to find a mapping that can achieve minimum total energy consumption
    - total number of system states: \((v - 1) + (c - 1)\)
    - total energy consumption: \(E_{\text{total}} = \sum_{i=1}^{n}(E_{i,x_i} + q_{x_{i-1},x_i})\)
    - dynamic programming is used (result showed in a different paper)
High-Level Performance Estimator (HiPerE) (1)

• Problems in system-level simulation:
  • lack of standard interface to integrate component specific simulators
  • time consuming to simulate a full system at low level
• Solutions:
  • use "component specific performance estimation" to capture the
    GenM performance parameters (tasks => processor/RL states)

MILAN: Model based
Integrated simuLAtioN framework
High-Level Performance Estimator (HiPerE) (2)

• Solutions
  • use "interpretive simulation" to derive system-level performance
  • design decision is used as input (brute force Vs. subsets from optimization techniques)
  • "interpretive simulation": resource constraint scheduling

• Example

\[
\text{for } k \leftarrow 1 \text{ to } n \text{ do}
\]

Let \( \beta \) be the set of immediate predecessors of \( T_{\pi_k} \).
The earliest start time for \( T_{\pi_k} \) is
\[
\tau = \max \{ \max_{T_i \in \beta} \{ t_i \}, A_{MP_{t_k}} \}
\]
Set \( t_{\pi_k} = \tau + \Gamma C_{\pi_k} \) and \( A_{MP_{\pi_k}} = t_{\pi_k} \)
Desing Space Exploration Framework

1. define the system states and roughly estimate the parameters in the GenM model
2. use optimization techniques to reduce the design space or use brute force method
3. use HiPerE to evaluate the initial system performance
4. use the component specific estimation to refine the parameters, and use the HiPerE to get more accurate performance estimates
5. integrate low-level simulators to performance complete system simulation
Case Studies (1)

- Azimuth-Elevation application

- Two architectures (each contains only a processor model) + four designs
  - MIPS @ 600 MHz \{SimpleScalar + Wattch\}
  - StrongARM @ 206, 162 and 59 MHz \{JouleTrack\}

- The simple architectures reduce the need to consider the transition parameters

---

TABLE 1
Part of the GenM model for AzimuthElevation

<table>
<thead>
<tr>
<th>Latency values</th>
<th>MIPS</th>
<th>SA</th>
<th>SA</th>
<th>SA</th>
</tr>
</thead>
<tbody>
<tr>
<td>in milliseconds</td>
<td>0600</td>
<td>0206</td>
<td>0162</td>
<td>059</td>
</tr>
<tr>
<td>BandPassFilter</td>
<td>0.22</td>
<td>2095</td>
<td>2312</td>
<td>6808</td>
</tr>
<tr>
<td>Threshold</td>
<td>0.02</td>
<td>01</td>
<td>120</td>
<td>330</td>
</tr>
<tr>
<td>LowPassFilter</td>
<td>0.44</td>
<td>3718</td>
<td>3050</td>
<td>9040</td>
</tr>
<tr>
<td>SignalGenerator</td>
<td>0.63</td>
<td>5618</td>
<td>6188</td>
<td>18393</td>
</tr>
</tbody>
</table>
Case Studies (2)

- Three simulations are carried out for each design
  - full system simulation using low-level simulators as reference data
  - HiPerE + Rough Estimates (energy/time costs are derived from experimental equations)
  - HiPerE + MILAN estimates

- Errors
  - MIPS
    - HiPerE + RE: >20%
    - HiPerE + MILAN: <3%
  - ARM
    - HiPerE + RE: <11%
    - HiPerE + MILAN: <7%
Conclusions

• Generic Model (GenM): A high-level abstraction of SoC architecture
• High-level Performance Estimator (HiPerE): A high-level interpreter for design space exploration
• Case studies to partially validate the approach
• Future work will be applied for the reconfigurable logic part