Tools and dataflow-based programming models for heterogeneous MPSoCs

Jeronimo Castrillon
Chair for Compiler Construction
TU Dresden
jeronimo.castrillon@tu-dresden.de

PEGPUM Workshop. HiPEAC Conference
Amsterdam, January 21st 2015
Acknowledgements

- German Cluster of Excellence: Center for Advancing Electronics Dresden (www.cfaed.tu-dresden.de)

- Collaborative research center (CRC): Highly Adaptive Energy-Efficient Computing (HAEC)

- Silexica Software Solutions GmbH
Multi-Processor on Systems on Chip (MPSoCs)

- **HW complexity**
  - Increasing number of cores
  - Increasing heterogeneity
  - Multi-cores everywhere
  - Ex.: Smartphones, tablets and e-readers

---

**ITRS Trend: PE Count**

![Graph showing thePE Count trend from 2011 to 2024.](image)

**Number of PEs**

<table>
<thead>
<tr>
<th>Year</th>
<th>Number of PEs</th>
</tr>
</thead>
<tbody>
<tr>
<td>2011</td>
<td>133</td>
</tr>
<tr>
<td>2012</td>
<td>185</td>
</tr>
<tr>
<td>2013</td>
<td>266</td>
</tr>
<tr>
<td>2014</td>
<td>343</td>
</tr>
<tr>
<td>2015</td>
<td>447</td>
</tr>
<tr>
<td>2016</td>
<td>558</td>
</tr>
<tr>
<td>2017</td>
<td>709</td>
</tr>
<tr>
<td>2018</td>
<td>899</td>
</tr>
<tr>
<td>2019</td>
<td>1137</td>
</tr>
<tr>
<td>2020</td>
<td>1460</td>
</tr>
<tr>
<td>2021</td>
<td>1851</td>
</tr>
<tr>
<td>2022</td>
<td>2317</td>
</tr>
<tr>
<td>2023</td>
<td>2974</td>
</tr>
</tbody>
</table>

**PE Count in SoCs**

- **OMAP Family**
  - OMAP1
  - OMAP2
  - OMAP3530
  - OMAP3640
  - OMAP4430
  - OMAP4470
  - OMAP5430

- **Snapdragon Family**
  - S1 QSD8650
  - S2 MSM8255
  - S3 MSM8060
  - S4 MSM8960
  - S4 APQ8064
  - S4 APQ8084

---

**System Integrations**

![Graph showing the System Integrations trend from 2010 to 2015.](image)

**Number of Systems**

<table>
<thead>
<tr>
<th>Year</th>
<th>Number of Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010</td>
<td>100000</td>
</tr>
<tr>
<td>2011</td>
<td>300000</td>
</tr>
<tr>
<td>2012</td>
<td>500000</td>
</tr>
<tr>
<td>2013</td>
<td>700000</td>
</tr>
<tr>
<td>2014</td>
<td>900000</td>
</tr>
</tbody>
</table>

---

© J. Castrillon. Pegpum workshop, Jan. 2015
MPSoCs: SW productivity gap

- SW-productivity gap: complex SW for ever-increasing complex HW
  - Cannot keep pace with requirements
  - Cannot leverage available parallelism
- Difficult to reason about time constraints
  - Even more difficult about energy consumption
- Need domain-specific programming tools and methodologies!
  - In this presentation: Dataflow/process networks for signal processing and multimedia
Dataflow and process networks

- Graph representation of applications
  - Implicit repetitive execution of tasks
  - Good model for streaming applications
  - Good match for signal processing & multi-media applications
- Stereo digital audio filter
Dataflow models: static vs. dynamic

- **Static: Synchronous dataflow models**
  - KPNs (& DDFs)
    - No “hardcoded” rates
    - More expressiveness
    - More difficult to analyze
  - Synchronous Dataflow (SDF) spec.
  - Unfolded cycle
  - "Blocked" DAG scheduling
Dealing with dynamic behavior: tracing

- **White model of processes**: source code analysis and tracing

... for (;i < x;i++) {
    write(&c2);
    f1(...);
    read(&c1);
    f2(...);
    read(&c1);
    ...
}
Trace-based mapping and scheduling

- Mapping: Trace-based heuristics
  - Mapping & scheduling: Analyze traces and propose mapping
  - Iterate: Improve mapping (if required)
Event traces can be represented as large dependence graphs.
Sample trace graph

Possible to reason about

- Channel sizes and memory allocation
- Mapping and scheduling onto heterogeneous processors

size(chan. 2) = 2, size(chan. 1,3) = 1
Dealing with heterogeneity: group-based mapping (GBM)

1) Initialize: All to all
2) Select element: Trace graph critical path
3) Reduce group
4) Assess & propagate
5) Quasi-homogeneous

[99x457] Dealing with heterogeneity: group-based mapping (GBM)

1) Initialize: All to all
2) Select element: Trace graph critical path
3) Reduce group
4) Assess & propagate
5) Quasi-homogeneous

[224x436] Apps. elements

[686x136] Resources

[99x278] Trace graph

[325x156] [DAC12]
Dealing with real-time applications

- Idea: intelligently add resources until a valid mapping is found
  - Resources: memories and processors

- Graphically
Dealing with SW/HW acceleration

- Specialized platforms: HW/SW acceleration
  - Compilation and source-based performance estimation **not applicable**
- Approach
  - Framework for marking accelerated routines
  - Tool flow to select and configure accelerators

[SDR10, ALOG11]
Dealing with SW/HW acceleration (2)

- “Nucleus” tool flow

[MILCOM09, Castrillon14]
Implementation: C extension for KPNs

- **FIFO Channels**
  ```c
  typedef struct { int i; double d; } my_struct_t;
  __PNchannel my_struct_t C;
  __PNchannel int A = {1, 2, 3}; /* Initialization */
  __PNchannel short C[2], D[2], F[2], G[2];
  ```

- **Processes & networks**
  ```c
  __PNkpn AudioAmp __PNin(short A[2]) __PNout(short B[2])
              __PNparam(short boost){
      while (1)
          __PNin(A) __PNout(B) {
              for (int i = 0; i < 2; i++)
                  B[i] = A[i]*boost;
          }
  }
  __PNprocess Amp1 = AudioAmp __PNin(C) __PNout(F) __PNparam(3);
  __PNprocess Amp2 = AudioAmp __PNin(D) __PNout(G) __PNparam(10);
  ```
Implementation: platform model

- Example: Texas Instruments Keystone

```xml
<Platform>
  <Processors List="ds0 ds1 ds2 ds3 ds4 ds5 ds6 ds7"/>
  <Memories List="local_mem_ds0 L2 local_mem_ds1 L2 local_mem_ds2 L2 local_mem_ds3 L2 local_mem_ds4 L2 local_mem_ds5 L2 local_mem_ds6 L2 local_mem_ds7 L2 local_mem_dsp L2 local_mem_dsp1 L2 local_mem_dsp2 L2 local_mem_dsp3 L2 local_mem_dsp4 L2 local_mem_dsp5 L2 local_mem_dsp6 L2 local_mem_dsp7 L2 local_mem_dma L2 local_mem_dma1 L2 local_mem_dma2 L2 local_mem_dma3 L2 local_mem_dma4 L2 local_mem_dma5 L2 local_mem_dma6 L2 local_mem_dma7 L2">
  <CommPrimitives List="PCI_SL2 PCI_DDR EDMA3 SLI EDMA3_DDR EDMA1">
  </CommPrimitives>
</Platform>

<Processor Name="ds0" CoreRef="DSPC66">
  ...
</Processor>

<Processor Name="ds7" CoreRef="DSPC66">
  ...
</Processor>

<Memory>
  <LocalMemory Name="local_mem_ds0" Size="512488" BaseAddr="">
  ...
</LocalMemory>
</Memory>

<CommPrimitive>
  <CPDMA Name="EDMA3_DDR">
    <Description>EDMA over DDR</Description>
    <Costs>
      <Cost End="800" Function="11442.60163-0.15775*x"/>
      <Cost Start="801" Function="11204.94186+0.316134*x"/>
    </Costs>
    <DMAs List="local_mem_ds0_DDR local_mem_ds1_DDR local_mem_ds2_DDR local_mem_ds3_DDR local_mem_ds4_DDR local_mem_ds5_DDR local_mem_ds6_DDR local_mem_ds7_DDR local_mem_dma1_DDR local_mem_dma2_DDR local_mem_dma3_DDR local_mem_dma4_DDR local_mem_dma5_DDR local_mem_dma6_DDR local_mem_dma7_DDR">
    </DMAs>
  </CPDMA>
</CommPrimitive>

<Core Name="DSPC66" CoreType="DSP">
  <MultiTaskingInfo MaxNumberOfTasks="1">
    <ContextSwitchInfo StoreTime="1000" LoadTime="1000"/>
    <SchedulingPolicies List="FIFO PriorityBased"/>
  </MultiTaskingInfo>
</Core>

<CostTable>
  <Operation Name="Load">
    <VariableType Name="Char">
      <Cost>1</Cost>
    </VariableType>
  </Operation>
</CostTable>
```

[SOC13]
Example: multi-media applications

- Platform: 2 RISCs, 4 VLIW, 7 Memories

![Graph showing makespan relative to GBM for real applications and random KPNs.](image)

- Makespan rel. to GBM

- LP-AF, JPEG, MJPEG

- Low comm., High comm.
Example: multi-media applications (2)

- **Dealing with real-time constraints**

  - Tool: \(\sim 1\) min. for LP-AF, \(\sim 7\) min. for MJPEG
  - Sim.: \(\sim 6\) days for LP-AF, \(\sim 24\) for MJPEG

  \[ \sim 10\ \text{min.} \]

  \[ \sim 10\ \text{days} \ (\sim x10^3) \]
SDR results: portable performance

- **Application**: MIMO OFDM receiver
- **Hardware**
  - **Platform 1**: Baseline software
  - **Platform 2**: Optimized software
  - **Platform 3**: Optimized SW + HW

Achieved rate @ 100 MHz

- **1)** bsp1 (sw, unoptimized) - 7680 bps
- **2)** bsp2 (sw, optimized) - 128000 bps
- **3)** bsp3 (hw) - 1758241,758 bps
Summary

- Programming methodology for dataflow applications
  - Programming model adapted to application domain
- Energy-efficiency
  - Aware of heterogeneous platforms with hardware acceleration
  - Do not use more resources than needed

- Outlook – Adaptability within HAEC CRC
  - Adapt HW to SW needs (e.g., wireless on-chip interconnect)
  - Adapt parallelism (with implicit parallel constructs) for scalability
  - More abstract application-specific software synthesis
References


References (2)


Thanks!
Questions?