A Hardware/Software Partitioning and Scheduling Approach for Embedded Systems with Low-Power and High Performance Requirements

Javier Resano, Daniel Mozos, Elena Pérez, Hortensia Mecha, and Julio Septién

Dept. de Arquitectura de Computadores, Facultad de Informática, UCM, Madrid

Abstract. Hardware/software (hw/sw) partitioning largely affects the system cost, performance, and power consumption. Most of the previous hw/sw partitioning approaches are focused on either optimising the hw area, or the performance. Thus, they ignore the influence of the partitioning process on the energy consumption. However, during this process the designer still has the maximum flexibility, hence, it is clearly the best moment to analyse the energy consumption. We have developed a new hw/sw partitioning and scheduling tool that reduces the energy consumption of an embedded system while meeting high performance constraints. We have applied it to two current multimedia applications saving up to 30% of the system energy without reducing the performance.

1 Introduction

Low-power has become one of the major design concerns. First of all, the designer must guarantee that his design does not exceed the power constraints of the target platform, since it will generate heating problems. Moreover, due to the proliferation of portable, battery-dependent devices, low-energy consumption has become one of the key features for the success of a design.

The current trend for portable embedded systems is to create heterogeneous systems, with one or more low-power processors, some additional hardware (hw) logic (ASICs and/or FPGAs), and some memory hierarchy. Current technologies allow creating the whole system in a single chip (SoC).

One of the most important steps to carry out in order to implement an application over such a system is to partition the application functionality among the different processing elements. This process drastically influences both the energy consumption and performance of the system. Figure 1 presents a simple example where the partitioning process can lead to energy savings. If the designer selects the fastest solution (sch1), the execution time is 139 time-units and the energy 21 energy-units. However, if the deadline for the application is 150, the designer can try to find a slower solution that meets this constraint while consuming less energy. In this case sch2 would be selected since its execution time is less than the deadline and its energy consumption is 16. Thus, the energy consumption decreases 25%.
Since our partitioning tool is still under construction, currently we just support a software (sw) processor, an FPGA, a system bus and one or several memory blocks. However, partitioning an application to such a system is still a NP-complete problem. Moreover, there are several existing prototype platforms as well as commercial platforms that follow this scheme providing a sw processor and some reconfigurable hw resources e.g. Garp [1], Morphosys [2] and the Virtex II-Pro XC2VP4 and VP7 [3].

The system bus and the memory blocks require a careful study, since both elements can significantly affect the system performance and energy consumption, especially because both hw and sw performance are improving much faster than communication channels and memories do. In order to estimate accurately the impact of the memories and buses in the system performance and energy consumption their physical features must be taken into account. Ideally the vendor should provide either estimators or at least time and power models, but unfortunately, this is not always the case, then, time and power models are needed, some examples of existing useful models are [4] for USB, and PCI buses (just timing considerations), and [5,6] for memories.

However, even after accurately estimating all the tasks, communications and memory accesses, computing the overall execution time it is not trivial, since it involves a scheduling that must take into account data and control dependencies as well as the accesses to the shared resources. Thus, we have developed a tool that schedules the tasks and the accesses to the system bus, and the shared memories during the partitioning process. This scheduling is the only way to accurately evaluate a solution, since otherwise, it is impossible to determine the impact of the communications or the delays introduced due to the conflicts on the accesses to shared resources (In [7] this problem is explained in detail). In addition, this scheduling prevents the need for arbitration logic in the bus controllers. Since the scheduler is integrated in a partitioning tool that must evaluate a great amount of different partitions one of our major concerns was to achieve near-optimal scheduling without increasing significantly the execution time of the partitioning tool.

The rest of the paper is structured as follows: section 2 presents an overview of the related work; section 3 explains in detail the format of the initial specification for our partitioning tool; section 4 describes the cost function that steers the design space exploration; sections 5, 6, and 7 explain how the energy, execution-time and hardware area are estimated for a given partitioning. Section 8 presents the experimental results and finally section 9 remarks some conclusions as well as future work to be done.

---

**Fig. 1.** Partitioning example. Two nodes must be partitioned between two Processing Elements (PE). T means time. E means energy. Sch1 and Sch2 are two selected solutions.

<table>
<thead>
<tr>
<th>Node1</th>
<th>Node2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PE1</td>
<td>78 13</td>
</tr>
<tr>
<td>PE2</td>
<td>88 8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node1</th>
<th>Node2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PE1</td>
<td>61 8</td>
</tr>
<tr>
<td>PE2</td>
<td>73 9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sch1</th>
<th>Sch2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PE1</td>
<td>PE1</td>
</tr>
<tr>
<td>PE2</td>
<td>PE2</td>
</tr>
</tbody>
</table>

Deadline

---