Preface

Single-core chip is no longer able to offer increasing performance. Hardware manufactures moved their attention from increasing clock frequency and instruction-level parallelism to thread-level parallelism by designing chips with a number of multiple interconnected cores.

The multi-core technologies are going to rapidly move towards massively parallel computing elements (hundreds or thousands of cores), becoming more and more pervasive in all fields of the embedded system industry, thanks to the improvements of the performance per watt ratio. However, it is clear that increasing the number of cores does not automatically translate into increasing performance.

Old sequential code will get no performance benefit from new multi-core chip. Indeed, since the single-core complexity and the clock frequency are typically lower with respect to old chips, performance of sequential code may get even worse.

The only way to increase performance on multi-core platforms is by squeezing their real power: using thread-level parallelism. However, new multi-threaded parallel code may not scale well with the number of cores due to the overhead introduced for data movement, scheduling and synchronisations. Even parallel code well tuned for maximising performance for a given platform may face the problem of achieving a good trade-off between development costs and time to solution. Parallel software engineering has engaged this challenge mainly by way of designing new tools and tool chains, introducing high-level sequential language extensions and reusable, well-known, parallel patterns.

This book discusses both basic concepts on parallel programming and advanced topics and issues related to the use of real embedded applications. The book derives from the experience and results obtained in the 3-year European ARTEMIS project SMECY (Smart Multi-core Embedded Systems, project number 100230) involving 27 partners in nine European countries. This represents a concrete experience of work in the embedded system area, where application, tool and platform providers worked together in a coordinated way with the goal to obtain new high-level programming tool chains for current and forthcoming embedded many-core platforms. The aim of the book is to describe a "new way" for programming complex applications for embedded many-core systems.

The partners involved in SMECY project were:

France:
  Commissariat à l'énergie Atomique et aux 'energies alternatives,
  THOMSON Video Networks SA,
  Silkan,
  STMicroelectronics (Grenoble 2) SAS,
  Thales Research & Technology,
  Université Joseph Fourier Grenoble 1
The Netherlands:
  ACE Associated Compiler Experts bv,
  Technische Universiteit Delft
Greece:
  Aristotle University of Thessaloniki,
  Hellenic Aerospace Industry S.A.,
  University of Ioannina
Czech Republic:
  Brno University of Technology,
  CIP plus s.r.o.,
  Ústav Teorie Informace a Automatizace AV ČR, v.v.i.
Denmark:
  Danmarks Tekniske Universitet
Sweden:
  Free2Move AB,
  Hogskolan i Halmstad,
  Realtime Embedded AB,
  Saab Microwave Systems
Finland:
  Nethawk Oyj,
  Valtion Teknillinen Tutkimuskeskus (VTT)
Italy:
  Politecnico di Milano Dipartimento di Elettronica e Informazione,
  Politecnico di Torino,
  SELEX E S,
  STMicroelectronics S.r.l.,
  Alma Mater Studiorum - Universita di Bologna
United Kingdom:
  United Kingdom: Thales Research & Technology

SMECY envisioned that multi-core technology will affect and shape the whole business landscape, thus having the ambitious mission to develop and propose new parallel programming technologies enabling the exploitation of embedded many-core architectures. The conceptual approach proposed by SMECY is based on the simple assumption that a tool chain should take both the application requirements and the platform specificities into account in order to be efficient. One of the main outcomes of the project was the definition of an intermediate representation

among front-end and back-end tools defining three different tool chains, one for each application domain.

The book contents are structured as follows:

In Chap. 1 we give a survey on the different parallel programming models available today. These models are suitable not only for general-purpose computing but also for programming specialised embedded systems that offer the required facilities.

Chapter 2 presents the big pictures of the SMECY project with the interaction of the different tools from the applications used as case studies to the targeted hardware platforms. The two-level intermediate representation allowing the interoperability of the components of the tool chains while providing a frame that drastically decreases the amount of work that would be required to add a new tool for further enrichment of the project tool chains is also presented. The intermediate representation designed within the SMECY project represents a bridge between a number of front-end and back-end tools.

Chapters 3 and 4 present in detail the two target platforms considered in the SMECY project, the STHORM platform by STMicroelectronics and the ASVP platform by UTIA.

Chapter 5 covers fault tolerance aspects in the context of embedded systems, whereas Chap. 6 proposes a lightweight code generation process that enables the capability to perform data-dependent optimisations, at runtime, for processing kernels.

Chapters 7, 8 and 9 describe the experiences made developing real-world applications on the SMECY target platform: a radar signal processing application, an object recognition image processing application and a foreground recognition video processing application, respectively.

Our sincere hope is that the reader could find value in the topics covered in this book.

Pisa, Italy   Massimo Torquati
Delft, The Netherlands   Koen Bertels
Lyngby, Denmark   Sven Karlsson
Grenoble, France   François Pacull