Faouzi Derbel, Nabil Derbel, Olfa Kanoun (Eds.)
Communication, Signal Processing & Information Technology

Advances in Systems, Signals and Devices

Edited by
Olfa Kanoun, University of Chemnitz, Germany

Volume 4

Communication, Signal Processing & Information Technology

Edited by
Faouzi Derbel, Nabil Derbel, Olfa Kanoun

DE GRUYTER
OLDENBOURG

Editors of this Volume
Prof. Dr.-Ing. Faouzi Derbel
Leipzig University of Applied Sciences
Chair of Smart Diagnostic and Online Monitoring
Wächterstrasse 13
04107 Leipzig, Germany
faouzi.derbel@htwk-leipzig.de

Prof. Dr.-Eng. Nabil Derbel
University of Sfax
Sfax National Engineering School
Control & Energy Management Laboratory
1173 BP, 3038 SFAX, Tunisia
n.derbel@enis.rnu.tn

Prof. Dr.-Ing. Olfa Kanoun
Technische Universität Chemnitz
Chair of Measurement and Sensor Technology
Reichenhainer Strasse 70
09126 Chemnitz
olfa.kanoun@etit.tu-chemnitz.de

ISBN 978-3-11-044616-6
e-ISBN (PDF) 978-3-11-044839-9
e-ISBN (EPUB) 978-3-11-043618-1
Set-ISBN 978-3-11-044840-5
ISSN 2365-7493
e-ISSN 2365-7507

Library of Congress Cataloging-in-Publication Data
A CIP catalog record for this book has been applied for at the Library of Congress.

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available on the Internet at http://dnb.dnb.de.

Typesetting: Konvertus, Haarlem
Printing and binding: CPI books GmbH, Leck
♾ Printed on acid-free paper
Printed in Germany

www.degruyter.com

Preface of the Volume Editor

The fourth volume of the Series "Advances in Systems, Signals and Devices" (**ASSD**), contains international scientific articles devoted to the field of communication, signal processing and information technology. The scope of the volume encompasses all aspects of research, development and applications of the science and technology in these fields. The topics include information technology, communications systems, digital signal processing, image processing, video processing, image and video compression, modulation and signal design, content-based video retrieval, wireless and optical communication, technologies for wireless communication systems, biometry and medical imaging, adaptive and smart antennas, data fusion and pattern recognition, coding compression, communication for e-mobility, microwave active and passive components and circuits, cognitive and software defined radio, vision systems and algorithms.

These fields are addressed by a separate volume of the series. All volumes are edited by a special editorial board made up by renowned scientist from all over the world.

Authors are encouraged to submit novel contributions which include results of research or experimental work discussing new developments in the field of communication, signal processing and information technology. The series can be also addressed for editing special issues for novel developments in specific fields. Guest editors are encouraged to make proposals to the editor in chief of the corresponding main field.

The aim of this international series is to promote the international scientific progress in the fields of systems, signals and devices. It provides at the same time an opportunity to be informed about interesting results that were reported during the international SSD conferences.

It is a big pleasure of ours to work together with the international editorial board consisting of renowned scientists in the field of communication, signal processing and information technology.

The Editors
Faouzi Derbel, Nabil Derbel and Olfa Kanoun

Advances in Systems, Signals and Devices

Series Editor:

Prof. Dr.-Ing. Olfa Kanoun
Technische Universität Chemnitz, Germany.
olfa.kanoun@etit.tu-chemnitz.de

Editors in Chief:

Systems, Automation & Control

Prof. Dr.-Eng. Nabil Derbel
ENIS, University of Sfax, Tunisia
n.derbel@enis.rnu.tn

Power Systems & Smart Energies

Prof. Dr.-Ing. Faouzi Derbel
Leipzig Univ. of Applied Sciences, Germany
faouzi.derbel@htwk-leipzig.de

Communication, Signal Processing & Information Technology

Prof. Dr.-Ing. Faouzi Derbel
Leipzig Univ. of Applied Sciences, Germany
faouzi.derbel@htwk-leipzig.de

Sensors, Circuits & Instrumentation Systems

Prof. Dr.-Ing. Olfa Kanoun
Technische Universität Chemnitz, Germany
olfa.kanoun@etit.tu-chemnitz.de

Editorial Board Members:

Systems, Automation & Control

Power Systems & Smart Energies

Communication, Signal Processing & Information Technology

Til Aach, Achen University, Germany
Kasim Al-Aubidy, Philadelphia Univ., Amman, Jordan
Adel Alimi, Engineering School of Sfax, Tunisia
Najoua Benamara, Engineering School of Sousse, Tunisia
Ridha Bouallegue, Engineering School of Sousse, Tunisia
Dominique Dallet, ENSEIRB, Bordeaux, France
Mohamed Deriche, King Fahd University, Saudi Arabia
Khalifa Djemal, Université d'Evry, Val d'Essonne, France
Daniela Dragomirescu, LAAS, CNRS, Toulouse, France
Khalil Drira, LAAS, CNRS, Toulouse, France
Noureddine Ellouze, Engineering School of Tunis, Tunisia
Faouzi Ghorbel, ENSI, Tunis, Tunisia
Karl Holger, University of Paderborn, Germany
Berthold Lankl, Univ. Bundeswehr, München, Germany
George Moschytz, ETH Zürich, Switzerland
Radu Popescu-Zeletin, Fraunhofer Inst. Fokus, Berlin, Germany
Basel Solimane, ENST, Bretagne, France
Philippe Vanheeghe, Ecole Centrale de Lille France

Sensors, Circuits & Instrumentation Systems

Ali Boukabache, Univ. Paul, Sabatier, Toulouse, France
Georg Brasseur, Graz University of Technology, Austria
Serge Demidenko, Monash University, Selangor, Malaysia
Gerhard Fischerauer, Universität Bayreuth, Germany
Patrick Garda, Univ. Pierre & Marie Curie, Paris, France
P. M. B. Silva Girão, Inst. Superior Técnico, Lisboa, Portugal
Voicu Groza, University of Ottawa, Ottawa, Canada
Volker Hans, University of Essen, Germany
Aimé Lay Ekuakille, Università degli Studi di Lecce, Italy
Mourad Loulou, Engineering School of Sfax, Tunisia
Mohamed Masmoudi, Engineering School of Sfax, Tunisia
Subha Mukhopadhyay, Massey University Turitea, New Zealand
Fernando Puente León, Technical Univ. of München, Germany
Leonard Reindl, Inst. Mikrosystemtec., Freiburg Germany
Pavel Ripka, Tech. Univ. Praha, Czech Republic
Abdulmotaleb El Saddik, SITE, Univ. Ottawa, Ontario, Canada
Gordon Silverman, Manhattan College Riverdale, NY, USA
Rached Tourki, Faculty of Sciences, Monastir, Tunisia
Bernhard Zagar, Johannes Kepler Univ. of Linz, Austria

Contents

M. Rößler, J. Langer and U. Heinkel

Finding an Optimal Set of Breakpoint Locations in a Control Flow Graph

Abstract: With the advance of high-level synthesis methodologies it has become possible to transform software tasks, typically running on a processor, to hardware tasks running on a FPGA device. Furthermore, dynamic reconfiguration techniques allow dynamic scheduling of hardware tasks on an FPGA area at runtime. Combining these techniques allows dynamic scheduling across the hardware-software boundary. However, to interrupt and resume a task, its context has to be identified and stored. We propose a method to find an optimal set of breakpoints in the control flow of a hardware task, such that the introduced resource overhead for context access is minimized and a maximum latency between interrupt request and the end of the context storing is guaranteed. This set of breakpoints allows the context to be restricted to the essential subset of data. Our method opens the door to flexible task scheduling not only on one reconfigurable device but also between different devices and even software instances of the same task.

Keywords: High level synthesis, Hardware software codesign, Preemption, Heterogeneous systems.

1 Introduction

This paper contributes to the vision of a system consisting of many different computing devices, including various types of processing units (generic multicore processors, digital signal processors or graphics processing units) running software tasks and reconfigurable FPGA devices running hardware tasks. Today, the advance of high-level synthesis (HLS) techniques blur the distinction between the two worlds as it becomes possible to create implementations of a single design specification for all types of devices mentioned before. Typically, the input specification is an algorithmic program written in common programming languages such as C or derived variants.

M. Rößler, J. Langer and U. Heinkel: Chair Circuit and System Design, Department of Electrical Engineering and Information Technology, Chemnitz University of Technology, Chemnitz, Germany, emails: marko.roessler@etit.tu-chemnitz.de, jan.langer@etit.tu-chemnitz.de, ulrich.heinkel@etit.tu-chemnitz.de.

De Gruyter Oldenbourg, ASSD – Advances in Systems, Signals and Devices, Volume 4, 2017, pp. 1–16.
DOI 10.1515/9783110448399-001

The big advantage of modern computing systems is concurrency. In order to gain efficiency the operating system can interrupt the execution of a task, save its context to the memory, restore it on another processor and resume its execution. This approach introduces flexibility to systems that can react to changing task priorities and environment requirements at runtime while consuming fewer resources than they virtually provide.

In order to implement such scheduling schemes not only on homogeneous systems of one or more equal processors but also in heterogeneous systems of different processors and reconfigurable hardware devices, it is necessary to interrupt hardware tasks during execution and save their context in an device-independent way while minimizing the required resource overhead. In this work we do not focus on possible implementations of such systems as SoC or HPC including their respective pros and cons. The focus is rather on how to find an optimal set of states in the central control flow of a hardware task that represent locations to interrupt the execution. We call those states *breakpoints* (the terms preemption point or switch point are also used in the literature).

When an interrupt occurs, the control flow progresses until it reaches a breakpoint. Then it stores the position in the control flow (the state itself) and all variables and structured memories that have been assigned a defined value (write access) and might be needed in the future (read access). This is called the context of a breakpoint and will later be used to initialize a software or hardware instance of the same task that is targeted on a different device. The set of *live* variables and structured memories are that part of the context that differs in size from breakpoint to breakpoint. In addition to minimizing the resource overhead of a set of breakpoints, we want to guarantee a maximum latency of the interrupt request, i.e. the time between the interrupt and the completion of storing the context does not exceed a certain number of cycles.

It is important to stress that interruption in the context of this work addresses breakpoints in the control flow that are common in the software and hardware implementation of a task. The traditional way of interrupting/restoring task execution on a processor at almost any position in the control flow is not applicable. Our definition of maximum latency assumes a continuous task execution for the software world until a breakpoint or the end is reached. In fact this assumption does hold for the addressed heterogeneous systems. With the contrast of multiple orders of magnitude between current reconfiguration latencies and the switching latency of a general purpose processor running a thin operating system layer it remains a reasonable approximation.

2 Related work

Current research focuses not only on the problem of converting classic C-style programs into corresponding hardware instances, but also develops new languages

and methodologies to write specifications that target many devices ranging from multi-core processors over GPUs to hardware [1, 2] as an enabling technology for a single source design environment to heterogeneous systems.

Preempting hardware tasks has long been proposed in the context of FPGAs or application specific programmable processors (ASPP). With FPGAs the read back approach for saving and storing the context was deployed in different ways, for example in [3, 4]. These methods operate on bitstream level and involve no additional hardware or design costs as the built-in hardware structure of the circuit is employed. [5] adresses the drawback of using the entire bitstream by filtering the state information from the stream. This requires very detailed information on the circuitry and introduces large dependencies on the FPGA vendors. However, all bitstream based methods do not allow context switching between software and hardware tasks.

There has been effort to introduce specific structures to hold state information on higher levels. In [6] a register scan chain approach including a preemption controller is proposed. It is shown that preemptive flip flops introduce low overhead in hardware but a high latency while transferring context because of the chaining. The search for an optimal set of states in which a scheduled control flow can be preempted is considered in [7] on the example of ASPP. Valid states are greedily chosen and refined in a second step to minimize a cost function describing the hardware overhead. Nevertheless, it remains unclear whether this approach can handle general control flow including indefinite loops and branches.

The problem of interrupting hardware tasks at selected breakpoints, saving its context and resuming its operation either as a software task or as a hardware task at a different device has been studied in [8]. However, this approach requires the developer to explicitly define all breakpoint locations and determine the variables of the breakpoint's context in the C code using specific annotations. Optimizing the set of living variables by rescheduling HLS designs, has been addressed in [9]. They minimize state retention registers that are related to power optimizations.

3 Control flow graph

A high-level description of the design is given as an algorithmic program. This program is transformed into hardware by a HLS tool resulting in the generation of a scheduled control flow graph (CFG) representing the finite state machine of the hardware and a data flow graph (DFG) connecting operations and variables. With the operations of the DFG being scheduled to control states and allocated to hardware resources, we know exactly which variable is read or written in which control state. For the sake of simplicity structured data and buffered input/output ports are treated as a single variable. By means for any future read access the respective memory resource is