CPS&IoT'2019

Summer School on Cyber-Physical Systems and Internet-of-Things Budva, Montenegro, June 10-14, 2019

Proceedings of the Summer School on Cyber-Physical Systems and Internet-of-Things

Vol. I

Editors

Lech Jóźwiak, Chairmen of the CPS&IoT'2019 Eindhoven University of Technology, The Netherlands and Radovan Stojanović University of Montenegro, Montenegro



MECOnet, Montenegro, June 2019

CPS&IoT'2019

Summer School on Cyber-Physical Systems and Internet-of-Things Budva, Montenegro, June 10-14, 2019

Chairmen Introduction

Cyber comes from Greek adjective kyberneticos (cybernetic) which means skilled in steering or governing. Already from ancient times people constructed various machines (physical systems) and their controllers (cyber systems). **Cyber-physical system (CPS)** is a compound system engineered through integration of cyber and physical sub-systems or components, so that it appears and operates as a single unit in relation to the external world (to other systems).

With the progress of time, machines and their controllers became more and more complex. Until the end of the 19th century the controllers (cyber systems) were implemented as mechanical, hydraulic and pneumatic systems. In the 20th century they started to be gradually replaced by the electric controllers, and later by the electronic controllers. Introduction of transistors and integrated circuit technologies in the years 1950s and 1960s, correspondingly, enabled the microelectronics and information technology revolution that is progressing according to the Moore's low till now. The revolutionary progress in computing platforms, communication, networking, sensors and actuators enabled much more effective and efficient CPS for traditional applications, and "smart", sophisticated and affordable CPS for numerous new applications, e.g. smart communicating robots, cars, wearable and implantable medical devices, etc.

Contemporary cyber-physical systems (**CPS**) are smart compound systems engineered through seamless integration of embedded information processing sub-systems and physical sub-systems. The modern smart collaborating CPS, that started to form the Cyber-Physical Systems of Systems (CPSoS) and Internet of Things (IoT), have important applications in virtually all economic and social segments, and their huge economic and societal impact rapidly increases. The CPS and IoT area undergoes a revolutionary development. There is however a common opinion that many more well-trained researchers and developers are needed in this rapidly developing area, as well as, more information exchange and collaboration among different projects and teams in the area.

This Summer School on Cyber-Physical Systems and Internet-of-Things (CPS&IoT'2019) aims at serving the following main purposes:

- advanced training of industrial and academic researchers, developers, engineers and decisionmakers; academic teachers, Ph.D. and M.Sc. students; entrepreneurs, investors, research funding agents, and policy makers; and other participants who want to learn about CPS and IoT engineering;
- dissemination, exchange and discussion of advanced knowledge and project results from numerous European R&D projects in CPS and IoT;
- promotion and facilitation of international contacts and collaboration among people working or interested in the CPS and IoT area.

The school is open to everybody, but previous knowledge or equivalent practical experience at least at the Bachelor level in engineering (e.g. system, computer, electronic, electrical, automotive, aviation, mechanical, or industrial engineering), computer science, informatics, applied physics or similar is recommended.

Industry Participation is encouraged. CPS&IoT'2019 Summer School is not only to follow courses and learn new knowledge on CPS and IoT from top professionals, but to meet people, interact and discuss with outstanding researchers, developers, academic lecturers, advanced students, and other

participants, collaborate or start collaborations, and meet many talented people who may become employees of your companies as well.

Distinguishing features of this advanced Summer School are that its *lectures, demonstrations, and practical hands-on sessions* will be given by *top European specialists* in particular CPS and IoT fields *form industry and academia*, and will deliver *very fresh advanced knowledge*. They are based on *results from numerous currently running or recently finished European R&D projects in CPS and IoT*, what gives an excellent opportunity to get acquainted with issues and challenges of CPS and IoT development; actual industrial problems, designs and case studies; and new concepts, advanced knowledge and modern design methods and tools created in the European R&D projects.

CPS&IoT'2019 Summer School is collocated with ECYPS2019 – 7th EUROMICRO/IEEE Workshop on Embedded and Cyber-Physical Systems, and MECO2019 – 8th Mediterranean Conference on Embedded Computing. The Summer School participants are encouraged to submit their papers to ECYPS2019 and MECO2019: <u>http://embeddedcomputing.me/en</u>.

The CPS&IoT'2019 Summer School Program is composed of four days of lectures, demonstrations, practical hands-on sessions, and discussions, as well as free participation in ECYPS 2019 and MECO 2019 sessions. The topics of the lectures, demonstrations, and practical hands-on sessions cover major CPS applications (focusing on modern mobile applications that require high-performance or low energy consumption, as well as, high reliability, security and safety), computing technology for modern CPS, CPS architectures, development problems and solutions, as well as, design methodologies and design tools for all CPS design phases. Detailed list of the CPS&IoT'2019 Presentations including the names of their authors and presenters is provided in the Schedule of the CPS&IoT'2019 Summer School.

Venue of CPS&IoT'2019 is <u>Hotel Budva</u>*****, Budva, Montenegro. **Budva** is a 3500 years old town located at the Adriatic Sea coast of Montenegro. It is a popular touristic destination, with its charming Old Town, beautiful natural environment, 35 clean sandy beaches, and proximity to many famous touristic attractions as Kotor, Boka Kotorska, Sveti Stefan, Dubrovnik, and several national parks. It is an excellent place to have a summer school in a relaxed and friendly atmospheer. For accomodation Hotel Budva***** and Hotel Slovenska Plaza**** are advised, but there are many other accommodation possibilities in Budva. Budva is very well accessible by plane. Podgorica Airport is about 65 km from Budva and it receives regular flights from Vienna, Paris, Rome, Zürich, Frankfurt, Warsaw, Ljubljana, Belgrade, and Instabul, while Tivat Airport (about 20km from Budva) and Dubrovnik Airport (65km from Budva) are very frequent vacation and charter flight destinations during the summer time.

More information can be found at the CPS&IoT'2019 Summer School web-site: <u>http://embeddedcomputing.me/en/cps-iot</u>.

This Summer School is possible thanks to involvement of many outstanding researchers and developers from multiple European countries, R&D projects and teams. The Chairmen of the CPS&IoT'2019 Summer School express their thanks to all authors and presenters of the CPS&IoT'2019 presentations, as well as, to all other people who contributed to the success of the Summer School, and wish a very effective and pleasant Summer School time to all the CPS&IoT'2019 participants.

Lech Jóźwiak Eindhoven University of Technology, The Netherlands and Radovan Stojanović University of Montenegro, Montenegro

Contents

Lech Jóźwiak, Introduction to Modern Cyber-Physical Systems and their Quality-Driven Design, 1

Henk Corporaal, Embedded Processors for Cyber-Physical Systems, 66

Jiří KadlecKadlec, *Zdeněk Pohl*, *Lukáš Lukáš Kohout*, Implementation of Implementation of HW - accelerated video -processing on industrial Zynq modules, 127

Francesca Palumbo, Claudio Rubattu, Carlo Sau, Tiziana Fanni, Luigi Raffo, Dataflow-Based Toolchain for Based Toolchain for Adaptive Accelerators, 149

Eugenio Villar, Megamodelingof complex, distributed, heterogeneous CPS systems, 288

Jesús Gorroñogoitia, Execution of software models, 308

Marc Geilen, Mladen Skelin, Hadi Alizadeh, Bram van der Sanden, João Bastos, Max plus linear models for cyber-physical systems, 326

Michael Paulweber and Andrea Leitner, Comprehensive modular V&v framework for automated cyber-physical systems, 372

Vittoriano Muttillo, Luigi Pomante, Design space exploration for Hypervisor-based mixed-criticality systems, 435

Axel Jantsch, Self-aware CPS, 517

Dip Goswami, Sajid Mohamed, Sayandip De, Tradeoff analysis between Quality-of-Control and degree of approximate computing for image based control systems, 622

Radek Fujdiak, Security of Embedded and Cyber Physical Systems, 652

Muhamed Shafique, Brain-Inspired Computing for Smart CPS and IoT, 677

Christoph Schmittner and Thomas Gruber, Safety & security engineering of automotive CPS, 744

Lorenzo Strigini and Peter Popov, Probabilistic modelling integrating concerns of safety, security, etc., 793

Dominique Blouin, Maysam Zoor and Ludovic Apvrille, Practical Embedded Systems Modeling and Safety, Security and Performance Verification with TTool, 846

Razi Seyyedi, Sören Schreiner Virtual Platforms for Low-power Mixed-criticality Embedded Systems Development and Validation 940



Introduction to Modern Cyber-Physical Systems and their Quality-Driven Design



Proceedings of CPS&IoT2019 page 1

1. Introduction

- 2. Modern cyber-physical systems (CPS)
- 3. Challenges of advanced CPS development
- 4. Computing technology for advanced CPS
- 5. Quality-driven design of advanced CPS
- 6. Conclusion

The main aim of this tutorial is to prepare the ground for the whole CPS&IoT'2019 Summer School

- □ This means in particular:
 - to introduce several basic definitions related to CPS
 - to sketch the CPS scene, what includes:
 - introduction to modern cyber-physical systems, their importance, their ongoing revolution, and challenges of their development, and
 - explanation of the necessity of their holistic multi-objective quality-driven design
 - to introduce the methodology of quality-driven model-based system design

Introduction: Further reading for this tutorial

- L. Jóźwiak: Advanced Mobile and Wearable Systems, Microprocessors and Microsystems, Elsevier, Vol. 50, May 2017, pp. 202-221
- L. Jóźwiak: Quality-driven Design in the System-on-a-Chip Era: Why and how?, Journal of Systems Architecture, vol. 47, no. 3-4, Apr. 2001, pp. 201-224
- L. Jóźwiak: Life-inspired Systems and Their Quality-driven Design, Lecture Notes in Computer Science, Vol. 3894, 2006, Springer, pp. 1-16
- Jóźwiak, L.; Lindwer, M.; Corvino, R.; Meloni, P.; Micconi, L.; Madsen, J.; Diken, E.; Gangadharan, D.; Jordans, R.; Pomata, S.; Pop, P.; Tuveri, G.; Raffo, L. and Notarangelo, G.: ASAM: Automatic Architecture Synthesis and Application Mapping, Microprocessors and Microsystems journal, Vol.37, No 8, pp. 1002-1019, 2013
- Jóźwiak, L. and Jan, Y.: Design of Massively Parallel Hardware Multi-Processors for Highly-Demanding Embedded Applications. Microprocessors and Microsystems, Volume 37, Issue 8, November 2013, pp. 1155-1172.
- L. Jóźwiak and S.-A. Ong: Quality-driven Model-based Architecture Synthesis for Real-time Embedded SoCs, Journal of Systems Architecture, Elsevier Science, Amsterdam, The Netherlands, ISSN 1383-7621, Vol. 54, No 3-4, March-April 2008, pp. 349-368
- Many other papers of myself and my former Ph.D. students; many of them referenced in the above papers 4

Introduction: What is a system?

- A system is a complex whole composed of interrelated, interdependent and/or interacting items (parts or elements of a system) that are so intimately connected that they appear and operate as a single unit in relation to the external world (to other systems)
- □ Three basic types of systems:
 - *unorganized system* a mechanical unsystematic conglomerate of objects
 - organized system a systematic, relatively stable and law-governed composition of parts which properties cannot be reduced to the simple sum of the properties of its parts, but involve some new emerging properties resulting from complex composition of the parts' properties (e.g. a molecule, crystal, circuit, computer), and
 - organic stem formed not as a composition of some ready-made parts, but being an integral whole with distinguishable parts that originate, develop and die together with the whole, and cannot preserve and demonstrate their complete quality without the whole (e.g. life organisms); the characteristic features of the organic systems are the self-development and self-reproduction
- In this presentation organized systems will be considered

Introduction: System organization and structure

- □ The system organization (composition) appropriately:
 - defines its parts
 - arranges the parts in relation to each other and to the whole, and
 - interconnects them to form the whole
- The term system structure designates the parts of a system arranged into a proper relation and appropriately interconnected according to a certain set of laws and/or rules in order to form a whole
- □ We will consider material systems
- Since matter is active and is in constant change, the material systems are in constant change, with only some relative and transient stability conditions
- Compositions of interrelated, interdependent or interacting single changes (transformations, actions) form processes
- Process is a relatively isolated composition of interrelated interdependent or interacting actions (transformations, changes)

Introduction: System = process © structure

- A given process can only perform (take place, occur) in particular relatively stabile conditions
- These conditions that make the process possible are created and guaranteed by the system structure
- The system structure is a relatively isolated, stable and slowly changing (in relation to the process) part of the universe in which a particular process (or a collection of co-operating processes) can take place
- A system is a unity of a process and structure in which this process takes place
- System design is an activity of defining an appropriate composition of the system process and structure

Introduction: What are cyber-physical systems?

- Cyber comes from Greek adjective kyberneticos (cybernetic) that means skilled in steering or governing
- Already in ancient times people constructed various systems: the oldest known artificial automatically controlled system is probably a water clock invented by Ktesibios (285–222 BC) in Alexandria
- Form those times, the construction of machines (physical systems) and their controllers (cyber systems) continued and developed through the centuries
- Until the end of 19th century the controllers (cyber systems) were implemented as mechanical, hydraulic and pneumatic systems
- In the 20th century they started to be gradually replaced by the electric controllers, and later by the electronic controllers
- Physical systems are systems in which matter or energy acquisition, processing and transfer take place according to the lows of physics
- Cyber systems are (parts of) control systems, i. e. information collecting, processing and communicating systems

Introduction : What are cyber-physical systems?

- Cyber-physical system (CPS) is a compound system engineered through integration of cyber and physical sub-systems or components and/or preexisting component cyber-physical systems, so that it appears and operates as a single unit in relation to the external world (to other systems)
- Introduction of the transistor and integrated circuit technologies in the years 1950s and 1960s, correspondingly, enabled the ongoing microelectronics and information technology revolution that is till now progressing according to the Moore's low
- The revolutionary progress in computing platforms, communication, networking, sensors and actuators enables:
 - much more effective and efficient CPS for traditional applications, and
 - "smart", sophisticated and affordable CPS for numerous new applications, e.g. smart robots, homes, cars, wearable and implantable medical devices, etc.

Introduction: very complex MPSoCs



Source: ANANDTECH (http://www.anandtech.com/show/7622/nvidia-tegra-k1)

- Modern nano-dimension semiconductor technology enables implementation of a very complex multiprocessor system on a single chip (MPSoC)
- □ This facilitates a rapid progress in:
 - global networking
 - (mobile) wire-less communication
 - (mobile autonomous) embedded computing

NVIDIA Tegra K1 massively parallel MPSoC for mobile applications CPU: (4+1) Cortex-A15 cores

Kepler GPU: 192 CUDA GPU cores

Introduction: cyber-physical technology revolution

- The recent rapid developments in:
 - system-on-a-chip technology
 - common global networking
 - wire-less communication
 - > mobile and autonomous computing
 - miniaturized sensors and actuators
 - material technology

created a large discrepancy between what is possible and what is used nowadays

- □ This discrepancy:
 - causes both a very strong technology push and market pull to create new or modified products and services, and
 - results in the cyber-physical technology revolution
- Recently, a revolutionary transition has been started from the internet of computers to the internet of smart (mobile) cyber-physical systems (CPS), called Internet of Things (IoT)

Examples of modern mobile CPS: autonomously-driving cars



Examples of modern mobile CPS: smart wearables













Examples of CPS: wearable virtual and augmented reality



Source: https://www.oculus.com

Examples of modern CPS: smart miniaturized implants and pill-size medical devices



modern 10 times smaller pace-makers

A new wave of the information technology revolution has arrived that creates much more coherent and fit to use CPS and connects them to form the IoT

Importance of modern mobile CPS

- Application areas of mobile CPS cover virtually all socially important application sectors, including:
 - *consummer applications*, e.g. mobile computing, communication, localization, navigation, gaming, entertainment, fashion, etc.
 - *extension or replacement of human capabilities*, e.g. tele-operation, personal assistance, artificial limbs, implants, etc.
 - social systems, e.g. smart health-care and other numerous health-care applications, assisted leaving, law enforcement, public safety, military, etc.
 - transportation and automotive, e.g. traffic control, navigation, tracking, communication, mobile fares and personalized customer service, assisted/autonomous driving, etc.
 - industrial, safety, security and military applications, e.g. mobile real-time in-the-field surveillance, monitoring, inspection, repair, robotics, instruction, assistance, etc.
 - *commercial applications*, e.g. mobile inventory tracking and customer service, wearable augmented reality and other systems for touristic applications, and many others
- The economic and societal importance of mobile CPS is very high and rapidly increases

Rapid growth of the mobile CPS and IoT markets

Worldwide car sales forecast by level of autonomy



Source: Canalys estimates, Autonomous Vehicle Analysis, December 2016

Rapid growth of the mobile CPS and IoT markets

Global unmanned aerial vehicle (UAV) market



- The fastest growing market of all mobile sectors is this of smart wearable devices:
 - \$14 billion and 123 million devices in 2016
 - \$34 billion and 411 million devices in 2020

(CCS Insight, February 2016)

Rapid growth of the chip market for mobile CPS and IoT

IC End-Use Markets (\$B) and Growth Rates



Source: IC Insights

The fastest-growing chip markets are automotive, IoT, medical and wearables

Proceedings of CPS&IoT2019 page 19

Rapid growth of the processor and MPSoC market for mobile CPS and IoT



MPUs for mobile systems account for almost 50% of MPU sales, and this share repidly increases

Proceedings of CPS&IoT2019 page 20

Challenges: unusual complexity and ultra-high demands

- The huge and rapidly developing markets of sophisticated mobile CPS represent great opportunities
- □ These opportunities come with a price of:
 - unusual system complexity and heterogeneity, resulting from *convergence* and combination of various applications and technologies in one system or even on one chip, and
 - stringent and difficult to satisfy requirements of modern applications
- □ Smart cars, drones and various wearable systems:
 - involve big instant data from multiple complex sensors (e.g. camera, radar, lidar, ultrasonic, sensor network tissues, etc.) and from other systems, used for mobile vision, imaging, virtual or augmented reality, etc.
 - are required to provide continuous autonomous service in a long time
 - are safety-critical
- In consequence, they demand a guaranteed (ultra-)high performance and/or (ultra-)low energy consumption, while requiring a high reliability, safety and security

Challenges: distribution of intelligence, computing resources, services and workloads in the IoT chierarchy

To transform the big data from multiple sensors to the information being directly used for decisions, while satisfying the stringent requirements of the modern mobile systems, a careful distribution of information delivery and computation services among the different layers of IoT is needed

□ For many reasons of primary importance, as:

- real-time availability of local information
- guaranteed real-time reaction
- security, safety, reliability
- minimization of communication traffic and energy, etc.

a majority of computing and decision making related to advanced CPS should be performed locally in the IoT edge devices, in collaboration among various local IoT edge devices or just above the edge nodes, and not in the higher levels of fog or in cloud

- □ The higher levels of fog and cloud should only be asked for services if:
 - necessary information or computing resources are not available locally, and
 - reaction-time, security, safety, etc. allow for this

Challenges: distribution of intelligence, computing resources, services and workloads in the IoT chierarchy

- This requires implementation of advanced intelligent computations and sophisticated powerful embedded computing technology:
 - directly in the IoT edge devices related to the complex sensors and actuators, or
 - just above the edge nodes, where the information from different sensors can be combined and based on the combined information the control decisions can be taken and subsequently actuated
- Sophisticated and powerful edge/dew computing has to be used requiring advanced intelligence, processing power and communication capabilities to be pushed towards the edge-nodes of IoT, where the data originates and information is used (i.e. to sensors, controllers and actuators)
- A very good example of the edge/dew computing necessity is the local vehicleto-vehicle and -infrastructure communication and collaboration necessary for autonomous driving
- □ In consequence, the IoT for advanced (mobile) CPS will be substantially different than Internet for other traditional targets Proceedings of CPS&IoT2019 page 23

Challenges: application parallelism and heterogeneity

- The modern complex applications that require ultra-high performance and/or ultra-low energy consumption:
 - are from their very nature heterogeneous
 - include numerous different algorithms involving various kinds of massive parallelism: data parallelism, and task-level, instruction-level and operation-level functional parallelism
- □ To adequately serve these applications:
 - heterogeneous computation platforms have to be exploited
 - processing engines with parallel multi-processor macro-architectures and parallel processor micro-architectures have to be constructed
 - different parts of complex applications involving different kinds of parallelism have to be implemented with corresponding different application-part specific parallel hardware
 - multiple different or identical processors, each operating on a (partly) different data sub-set, have to work concurrently to realize the ultra-high throughput and ultra-low energy consumption

Challenges: application complexity, parallelism and heterogeneity

To implement the highly-demanding complex heterogeneous CPS applications complex heterogeneous MPSoCs are needed



*Source: http://tweakers.net/reviews/3162/2/intels-atom-bay-trail-de-eerstenieuwe-atom-in-vijf-jaar-zes-verschillende-bay-trails.html +Source: http://www.anandtech.com/show/4144/lg-optimus-2x-nvidia-tegra-2-reviewthe-first-dual-core-smartphone/3 Proceedings of CPS&to T2019 page 25 25

CPU

Challenges: application complexity, parallelism and heterogeneity

NVIDIA's advanced massively parallel heterogeneous MPSoC for ADAS and similar mobile CPS applications



Source: Albert Y.C. Chen, Viscovery

The status of computing technology for advanced CPS

- Many advanced processors and heterogeneous parallel MPSoC architectures have been proposed in the recent years
- □ Many of them are useful for various advanced (mobile) CPS applications
- What is the problem?
- **The design methods and automated tools for:**
 - mapping of complex heterogeneous parallel applications to such hardware platforms
 - customization of such platforms and coherent HW/SW architecture co-development
 - parallel programming and code parallelization and compilation for such platforms
 - development and management of autonomous evolvable distributed systems and systems-of-systems collaborating through IoT
 - management of competing CPS applications, computing resources, services and workloads in the IoT hierarchy
 - modeling, analysis, development, verification, validation and certification of CPS involving combined diverse cyber and physical components or sub-systems
 - holistic development and multi-objective optimization of complex heterogeneous CPS
 - ensuring reliability, security and safety of critical CPS

are much less advanced

Challenges: criticality of applications

- Cyber-physical systems influence our life to a higher and higher degree
- □ Therefore, the society expectations regarding them grow rapidly
- Due to CPS common usage in various kinds of technical, social and biological applications, and their growing influence, we and the life on the Earth more and more depend and rely on these systems:
 - their quality is becoming more and more critical
 - many applications considered previously as non-critical are becoming critical
- Due to the rapidly growing share of the highly demanding embedded and CPS applications, *higher demands are becoming much more common*
- Due to the multiple reasons just discussed, and specifically, due to the rapidly growing system and silicon complexity and diversity, it will be more and more difficult to guarantee the systems' quality
- This is a new difficult situation that cannot be adequately addressed without an adequate design methodology and electronic design automation

Quality-driven Model-based Design

- When considering a system and design methodology adaptation to the situation in the field of modern CPS, we have first to ask: what general system approach and design approach seem to be adequate to solve the listed problems and overcome the challenges?
- Predicting the current situation, more than 20 years ago I proposed such system paradigm and design paradigm, i.e. the paradigms of:
 - Iife-inspired systems and quality-driven design, and
 - the methodology of quality-driven model-based system design based on them
- From that time my research team and our industrial and academic collaborators were researching the application of this methodology to the design and design automation of embedded processors, MPSoCs and CPS, and this research confirmed the adequacy of the quality-driven design methodology
- For "Outstanding Achievements and Contributions to Quality of Electronic Design" I was awarded the Honorary Fellow Award by the International Society for Quality Electronic Design (San Jose, CA, USA, 2008)

What is the quality-driven design? of CPS&IoT2019 page 29

Quality-driven design approach

- **System design is a** *definition of the required quality*, i.e. a satisfactory answer to the following two questions:
 - > What new (or modified) quality is required? and
 - How can it be achieved?
- Intuitively we feel that **quality** is here used in the sense of **the totality of the** (important) features the system has
- So, system design should define:
 - What is the required totality of the (important) system features? and
 - How to realize a system that has these all features?
- In other words:
 - What process must be realized in a certain system and what structural and parametric features must have the system?
 - How can we build a system that will be able to realize this process and will have the required structural and parametric features? 30

□ Actually, what is quality?

□ The most used and cited definitions of quality:

- > fitness for use (Juran)
- conformance to requirements (*Crosby*)
- quality is meeting the customers' expectations at a price they can afford (*Deming*)
- the loss of quality is the loss a product causes to society after being shipped, other than any losses caused by its intrinsic functions (*Taguchi*)
- The totality of features and characteristics of a product or service that bear on its ability to satisfy given needs (*American Society for Quality Control*)
- The totality of features and characteristics of a product or service that bear on its ability to satisfy stated or implied needs (*ISO8402: Quality Vocabulary Part 1, International terms, 1987*)

Problems with the existing definitions of quality

they focus exclusively on a product being designed, while the original problem is solved by designing, fabrication, usage and disposing of the system



Quality cannot be limited to the system itself, but it must account for the complete problem solution, related to complete system life-cycle 32
Problems with the existing definitions of quality

- none of these definitions is precise enough to enable the systematic consideration, measurement and comparison of quality
- the assumption of perfectly known and inviolable customer's requirements is not acceptable, because the customer may specify the requirements poorly and such requirements may result in system which will create danger, damage environment or squander scarce resources
- engineered systems solve certain real-life problems, serve certain purposes they are purposive systems
- quality of a purposive system can only be defined in relation to its purpose

Quality of a purposive systemic solution is its **total effectiveness and efficiency**

in solving of the real-life problem that defines the solution's purpose

- **Effectiveness** = the degree to which a solution attains its goals
- Efficiency = the degree to which a solution uses resources in order to realize its aims
- Effectiveness and efficiency of a systemic solution together decide its grade of excellence their aggregation expresses quality
- Effectiveness and efficiency can be expressed in terms of measurable parameters, and in this way, quality can be modeled and measured
- In particular, the quality can be modeled in the form of *multi-objective* decision models involving measurable design parameters
- The multi-objective decision models and design parameter estimators enable application of the multi-objective decision methods for construction, improvement and selection of the most promising solutions

Quality-driven Design - Difficulties



Interactions and trade-offs between various parts and aspects of the total systemic solution

Quality-driven Design - Difficulties



Interactions of a design project with its context

Quality-driven Design - Difficulties

- Design does not concern the reality as it is, but as it will possibly be realized
- Quality recognition and formulation, i.e. recognition of the problem, as well as of the nature of its solution are *subjective* to a high degree
- The contemporary system design problems are complex, multiaspectual, dynamic, and ill-structured:
 - there is no definitive formulation of the problem,
 - any problem formulation may be inconsistent,
 - formulations of the problem are solution dependent,
 - proposing and considering solutions is a means for understanding the problem, and
 - there is no definitive solution to the problem

- The complex design problems are ill-defined
- It is very difficult to find precise relations between various aspects of the system effectiveness and between the different forms of energy and matter used to attain the system's aim, and even more difficult to express them as one uniform measure
- There are trade-offs as well between effectiveness and efficiency as among different their aspects
- The required quality or its perception can change in time

quality cannot be well defined, but it can and should be modelled

Quality-driven Design - Design models

- Well-structured models of the required/delivered quality can serve to:
 - conceptualize, denote, analyse and communicate the customer's and designer's ideas
 - > show that the requirements and designs are meaningful and correct
 - > guide the design process
 - > enable the explicit and well-organized design decision making
 - > enable design automation



Quality-driven Design: Design problem-solving using models

- Since the system design problems are:
 - complex;
 - multi-aspect;
 - ill-defined,

to solve them, all human concepts for dealing with complexity, diversity and ill-structure have to be applied:

- abstraction;
- separation of concerns;
- decomposition and composition;
- generalization and specialization;
- modelling;
- simulation;
- prototyping;
-

A design problem has to be converted into a system of simpler subproblems

■ The solution to the original problem can then be achieved by solving the subproblems and composing the sub-problem solutions into an aggregate solution

Quality-driven Design: Design problem-solving using models

- The problem decomposition and design modelling are to some degree subjective
- The design decision processes are also to some degree subjective, as they are influenced by the **designers'** value systems, feelings, believes, intuition etc.
- The design problem solving activity is performed under uncertainty, inaccuracy, imprecision and risk conditions, and in a dynamic environment

₩

System design has to be an evolutionary process in which analysis and modelling of problems; proposing their solutions; analysis, testing and validation of the proposals; learning and adapting are very important

Main concepts of the quality-driven design

- Designing *top-quality systems is the aim* of a design process
- Quality is modelled and measured (in particular, in the form of the multiobjective decision models) to enable invention and selection of the best alternatives and quality improvement
- Quality models are considered to be heuristics for setting and controlling the course of design

The design process is evolutionary and it basically consists of:

- constructing the tentative quality models,
- using them for constructing, improving and selecting of the tentative solutions,
- analysing and estimating them directly and through analysis of the resulting solutions,
- improving the models, and using them again to get improved solutions, etc.

Main concepts of the quality driven design

- In the design process, a balance is sought for between the effectiveness and efficiency, and among all their important aspects; in particular between:
- the multiplicity of the life-cycle aspects considered in parallel and the amount of iteration
- between design reuse and innovation
- ➢ between art and science in the design
- between the designers subjective inventive thinking and objectivity increasing constraints and regulations
- > between the human designer's involvement and automation.
- Criterium for this balance is *total effectiveness and efficiency of a design process*

One of the main aims of using the well-defined models in design is:

Limiting the scope of subjective design decision making and enlarging the scope of reasoning-based decision making with clear and well-defined rational procedures which can be computerized

- Too much subjectivity in design may result in solutions that either do not solve the actual real-life problem or do not do it in a satisfactory manner
- Limiting the design subjectivity in an appropriate manner, when enabling the creativity exploitation at the same time, is necessary to arrive at the high-quality designs

Quality-driven Design: Limiting the design subjectivity

- □ The main means for limiting the design subjectivity is the design space exploration (DSE) with usage of the well-structured quality models
- Exploration of the abstract models of the required quality and more concrete solutions obtained with these models:
 - gives much and more objective information on the design problem, its possible and preferred solutions, and various models used in this process
 - enhances exploitation of the designer's imagination, creativity, knowledge and experience
- Other important means for limiting the design subjectivity include:
 - > appropriately organised team-work
 - benchmarking and comparison with both own previous designs and designs of competition
 - > design analysis and validation
 - design reuse
 - > government and branch regulations and standards

□ The general model of the required **system's** quality is represented by the *system (design) requirements*

Not "the conformance to requirements" (P.B. Crosby), but the solution of the actual real-life design problem with a satisfactorily high total effectiveness and efficiency is important

Requirements can only be treated as a non-perfect and tentative model of the required quality

The requirements and the solutions obtained with their use should be confronted with the actual up-to-date needs many times during the design process, and replaced or modified, if necessary

Requirements and any other quality models are not sacred and inviolable, but they are subject to design and change

- Design requirements represent a model of the required quality that models the design problem at a hand through the *imposition of a number of constraints and objectives in relation to the acceptable or preferred problem solutions*
- It is thus an *abstract model of a solution to the problem*.
- Since such a model limits the space of acceptable or preferred solutions to a certain degree only, it *models many solutions concurrently*.
- Each of the solutions fulfils all the hard constraints of the model, but different solutions can satisfy its objectives to various degrees.
- □ It is possible to distinguish three sorts of requirements:
 - ➤ functional,
 - *structural*, and
 - ▶ parametric

All the three sorts of requirements impose *limits on the structure of a required solution*, but they do it in different ways

- The structural requirements define the acceptable or preferred solution structures directly, by limiting them to a certain class or imposing a preference relation on them
- The parametric requirements define the structures indirectly, by requiring that the structure has such physical, economic or other properties (described by values of some parameters) as fulfil given constraints and satisfy stated objectives
- The *functional requirements* also define the structures indirectly, by requiring the structure to expose a certain externally observable behaviour that realizes the required behaviour

Quality-driven design space exploration (DSE)

- System design is an evolutionary quality engineering process in which the concepts of analysing and modelling problems, proposing their solutions, analysing and testing the proposals, learning and adapting are very important
- It starts with an *abstract*, and possibly *incomplete*, *imprecise*, and *contradictory*, *initial quality model* (initial requirements)
- It tries to transform the initial model into a concrete, precise, complete, coherent and directly implementable final model
- The initial abstract model mostly involves some *behavioural and* parametric characteristics and to a lesser extend the structure definition
- □ The final model defines the *system's structure explicitly*
- This structure supports the system's required behaviour and satisfies the parametric requirements

Quality-driven DSE

- During the design process the structural information is gradually added by the designers and synthesis tools to the created (partial) solutions.
- This evolutionary quality engineering processes applies the problem-solving framework of heuristic search and decomposes the total design problem into several issues.
- In this framework, the process of design problem solving can be represented by a *design search tree*:
 - the tree's nodes correspond to various design issues (sub-problems)
 - the tree's branches correspond to various design options (alternative solutions)
 - > for each issue, many various alternative solutions are typically possible.
- A design decision is a choice of a particular option, or the option chosen
- Each option chosen may recursively raise new issues, expanding the design search tree downwards until a final design will be obtained proceedings of CPS&IoT2019 page 50

Quality-driven design space exploration

- □ For each issue, many various alternative solutions are typically possible.
- For each issue, we can construct some issue's quality models, composed of some selected and abstracted functional, structural and parametric requirements extracted in an appropriate manner from the total quality model of the considered system.
- In particular, the issue's decision model can be constructed that is a base for decision making in the scope of a certain issue
- A decision model is a *partial* (reduced to only certain concerns) and *abstract* (reduced to the necessary and/or possible precision level) model of the required quality, *expressed in the decision-theoretical terms*.
- Decision models and design parameter estimators enable application of the multi-objective decision methods for construction, improvement and selection of the most promising solutions.

- The decision model of a given issue must account for all system characteristics substantially relevant to the issue
- It must specify preferences of values for all the characteristics, expressed by hard constraints, objectives, and trade-off information
- For each single characteristic, the preferences of its values can be characterized by specifying a utility (effectiveness or efficiency) function u_i(x_i) for the characteristic x_i
- Each utility function $u_i(x_i)$ describes the level of satisfaction from a particular value of the characteristic x_i
- Due to the multi-aspect nature of systems and possible trade-offs, the relative importance of different characteristics or the reference points in the utility space have to be specified

Quality-driven Design - Decision models

- This can be done in different ways dependent on the problem characteristics, for example by:
 - establishing an order for the objectives,
 - constructing a multi-objective utility function,
 - defining ranking information,
 - establishing local preferences for small changes in values of the objectives, or
 - defining some reference (aspiration) points in the utility or parameter space
- With such models the total system quality Q can be modelled as a function of utility levels of all the important system characteristics influencing the systems effectiveness or efficiency
- Such design decision models make it possible to apply the multiobjective decision methods for invention and selection of solutions that are "totally optimal"

Modeling quality Q as a (vector) function of utility levels of the system characteristics



Proceedings of CPS&IoT2019 page 54

Generic model of the quality-driven design space exploration



Quality-driven design space exploration

- □ The quality-driven design space exploration basically consists of the alternating phases of:
 - exploration of the space of abstract models of the required quality

and

exploration of the space of the more concrete issue's solutions obtained with the chosen quality models

In this way, both:

"let's make better things"

and

"let's make things better"

will be brought into effect.

Generic model of the quality-driven design space exploration



Quality-driven design space exploration

- In result of the design space exploration, the considered system is defined as an appropriate *decomposition into a network of sub-systems*
- Each sub-system solves a certain sub-problem
- All sub-systems cooperating together solve the system design problem by exposing the external aggregate behaviour and characteristics which match the required behaviour and characteristics
- The design process breaks down a complex system defined in *abstract* and non-precise terms into a structure of cooperating sub-systems defined in more concrete and precise terms, which are in turn further broken down to the simpler sub-systems that can be directly implemented with the elements and sub-systems at the designer's disposal

Example: Quality-driven model-based automated design of multi-ASIP MPSoCs (ARTEMIS ASAM project Grant No. 100265)

- To develop the required complex multi-ASIP MPSoCs, a sophisticated design space exploration is necessary in which only the most promising ASIP and MPSoC architectures will be efficiently constructed, and the best of these architectures will be selected for further analysis, refinement and actual implementation
- The ASAM multi-ASIP MPSoC design-space exploration implements the qualitydriven model-based system design methodology
- According to this methodology, to bring the quality-driven design into effect, quality has to be modeled, measured, and compared
- □ The quality of the multi-ASIP MPSoC required is modeled in the form of the:
 - demanded system behavior (application c-code)
 - structural constraints: generic ASIP and MPSoC architecture templates and their pre-characterized generic parts included in the IP library, and
 - parametric constraints and objectives to be satisfied by the MPSoC design
- Based on the analysis of the so modeled required quality, the generic architecture templates are adequately instantiated and used in design space exploration that constructs one or several most promising MPSoC designs supporting the required behavior and satisfying the demanded constraints and objectives 59

Example of Generic WLIW ASIP Architecture Template (Intel Benelux, used in ASAM project)



Example instance of the generic ASIP template: video processor



Quality-driven model-based automated design of multi-ASIP MPSoCs: Quality-driven DSE

- Based on the analysis of the so modeled required quality, the generic architecture template is adequately instantiated and used in design space exploration that aims at:
 - analysis of various architectural choices regarding:
 - processor micro-architectures and multi-processor macro-architecture
 - parallel memories architectures
 - parallel communication architectures
 - macro-/micro-architecture tradeoffs
 - processor, memory and communication tradeoffs,

and based on this analysis,

 construction of one or several most promising (sub-)system architectures supporting the required behavior and satisfying the demanded constraints and objectives.

Quality-driven DSE: General Organization



ASAM main result: quality-driven design method, flow and tools for the automated synthesis of heterogeneous ASIP-based MPSoCs



Conclusion

- Many modern CPS applications are complex, heterogeneous, involve big data and massive parallelism, and demand an (ultra-)high performance and/or (ultra-)low energy consumption, while requiring a high reliability, safety and security
- Many parallel processors and heterogeneous MPSoC architectures have been proposed in the recent years, and some of them are useful for various advanced CPS applications
- □ The related design methods and automated tools are much less advanced
- Although much research on various aspects of CPS, and design methods and automated tools for CPS has been performed, and a reasonably solid methodological base has been created, much more work has still to be done
- In this CPS&IoT Summer School you will have a unique occasion to be informed on and to discuss the most recent European R&D developments in CPS and IoT

CPS-IoT summerschool

Embedded Processors for Cyber-Physical Systems

Henk Corporaal www.ics.ele.tue.nl/~heco h.corporaal@tue.nl June 10-13, 2019

Proceedings of CPS&IoT2019 page 66

Lecture overview

- Some history
- Processor basics
- Memory hierarchy
- Advanced processor system design
 - the advanced concepts of today you'll see tomorrow in the low end CPS

Crash Course

- Example system: EEG monitoring system
- Interesting CPS development processors & boards
- Conclusions

A long history: Earliest computers

- Mechanical
 - Charles Babbage: Difference Engines
- Electro-Mechanical
 - Konrad Zuse's Z1, Z2, Z3
- First Electronic
 - ENIAC



Part of Difference Engine 1, 1832


5/23/2019

Frequency and Power development of processors

- Intel 80386 consumed ~ 2 W
- 3.3 GHz Intel Core i7 consumes 130 W
- Heat must be dissipated from 1.5 x 1.5 cm chip
- This is the limit of what can be cooled by air



Processor history

35 YEARS OF MICROPROCESSOR TREND DATA

Trends:

- #transistors follows Moore
 - but slows down somewhat
 - not clear what happens below 3-5 nm

• frequency and performance/core do not scale further



Original data collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond and C. Batten Dotted line extrapolations by C. Moore

A small processor board: Arduino MKR GSM 1400



- ARM Cortex M0 32-bit
- 256 KB flash
- 32 KB SRAM
- upto 48 MHz

68x25 mm, 32 gr

The other extreme: Summit, Nr 1 in the Top 500

- IBM Summit
 - 122.3 petaflops on Linpack
 - 4356 nodes
 - per node:
 - 2x 22-core Power9
 - 6 NVIDIA Tesla V100 GPUs





Proceedings of CPS&IoT2019 page 73

Processor basics: structure of any computer system How does it operate?



The Big Picture



see tutorial: https://www.cise.ufl.edu/~mssz/CompOrg/CDAmlang.htmlpage 75

Types of Instructions of any Computer

The core ISA is easy: we need only 3 types of instructions/operations:

- Arithmetic
 - Integer
 - Floating Point
- Memory access
 - Loads & Stores
- Control flow
 - Jump
 - Conditional Branch
 - Call & Return
 - Interrupts

Q: what's the difference between an operation and an instruction?

Instructions: Load and store

• Example:

C code: A[8] = h + A[8]; MIPS code: lw \$t0, 32(\$s3) add \$t0, \$s2, \$t0 sw \$t0, 32(\$s3)

- Store word operation has no destination (register) operand
- Remember: on a RISC processor arithmetic operands are in registers, not in memory!

Machine Language: Load & Store

- Example: lw \$t0, 32(\$s2)
- Introduce a new type of instruction format: I-type

Question: Where's the compromise?

A typical RISC: the MIPS

Instruction ullet

Meaning

add \$s1,\$s2,\$s3	\$s1 = \$s2 + \$s3
sub \$s1,\$s2,\$s3	\$s1 = \$s2 - \$s3
lw \$s1,100(\$s2)	\$s1 = Memory[\$s2+100]
sw \$s1,100(\$s2)	Memory[\$s2+100] = \$s1
bne \$s4,\$s5,L	Next instr. is at Label if $\$s4 \neq \$s5$
beq \$s4,\$s5,L	Next instr. is at Label if $\$s4 = \$s5$
j Label	Next instr. is at Label

• 3 Formats:

R	op	rs	rt	rd	shamt	funct
I	op	rs	rt	16 bit	addre	SS
J	op		26 bit	addre	SS	

Keep it simple: only 3 data addressing modes

1. Immediate addressing

ор	rs	rt	lm m ediate
----	----	----	-------------

2. Register addressing



3. Base addressing



Raise the frequency: Pipelining



5/23/2019 ECA H.Corporaal

Raise the frequency: Pipelining



5/23/2019

Raise the frequency: Pipelining details, with control



ECA H.Corporaal

Hazards: problems due to pipelining

Hazards: can not start next instruction in next cycle

3 types:

• Structure hazards

- A required resource is busy

• Data hazard

- Need to wait for previous instruction to complete its data read/write

- Control hazard
 - Do not know (directly) what the next instruction is

Data Hazards

- An instruction depends on completion of data access by a previous instruction
 - add \$s0, \$t0, \$t1

 sub
 \$t2, \$s0, \$t3
 // RaW dependence on \$s0



Forwarding (aka Bypassing)

- Use result when it is computed ASAP (as soon as possible)
 - Don't wait for it to be stored in a register
 - Requires extra connections in the datapath



Datapath with Forwarding



Memory Hierarchy, why?

- Users want large and fast memories! SRAM access times are 1 – 10 ns DRAM access times are 20 – 120 ns Disk access times are 5 to 10 million ns, but it's bits are very cheap
- Get best of both worlds: fast and large memories:
 build a memory hierarchy
- Q: why does it work, such a small level-1 memory ?





Solving collisions: 4-Way Set Associative Cache



Splitting first level cache, 2nd level cache

- Split Instruction and Data cache:
 - Caches can be tuned differently
 - Avoids dual ported cache
- 2nd Level: reduce miss penalty



Example Intel: i7 Ivy Bridge, 22 nm

3rd Generation Intel® Core™ Processor: 22nm Process



-3 levels of cache; see next slide

i7 (Nehalem) Cache hierarchy

• Per core

– L3

- L1 instruction
- L1 data
- L2 instr. + data
- Per 4 cores/die



Going parallel



• Why

- power gets out of control; Power = α f C V² // *frequency Capacitance Voltage*²

- Many ways:
 - More pipeline stages
 - Multi-issue / Out-of-Order processing
 - Multi-threading
 - Vector or SIMD / SIMT GPU
 - Multi-core: MIMD

OoO + Speculative execution



Core i7 micro architecture



Proceedings of CPS&IoT2019 page 96

SIMD concept



✓ SIMD: low-power architecture

 \checkmark instruction fetch and decode overhead negligible

✓ massively-parallel: large number of PEs, high performance

✓ e.g. performing 128 'add' operations in 1 cycle

SIMD Sub-word Implementations

- Implementations:
 - Intel MMX (1996)
 - Eight 8-bit integer ops or four 16-bit integer ops
 - Streaming SIMD Extensions (SSE, 1999)
 - Eight 16-bit integer ops
 - Four 32-bit integer/fp ops or two 64-bit integer/fp ops
 - Advanced Vector Extensions (AVX, 2010)
 - Four 64-bit integer/fp ops or eight 32 bit integer/fp ops
 - ARM NEON (since v6 architecture)

Operands must be consecutive and aligned memory locations

E.g. 16 bytes in parallel:



Example 1: DAXPY, SIMD Code (MIPS)

• DAXPY = (double) $Y = a \cdot X + Y$ //used to rank top500 supercomputers

L.D	F0, a	;load scalar a
MOV	F1, F0	;copy a into F1 for SIMD MUL
MOV	F2, F0	;copy a into F2 for SIMD MUL
MOV	F3, F0	;copy a into F3 for SIMD MUL
DADDIU	R4,Rx,#512	;last address to load (Double word immediate unsigned add)
Loop:		
L.4D	F4,0[Rx]	;load X[i], X[i+1], X[i+2], X[i+3]
MUL.4D	F4,F4,F0	;a×X[i],a×X[i+1],a×X[i+2],a×X[i+3]
L.4D	F8,0[Ry]	;load Y[i], Y[i+1], Y[i+2], Y[i+3]
ADD.4D	F8,F8,F4	;a×X[i]+Y[i],, a×X[i+3]+Y[i+3]
S.4D	0[Ry],F8	<pre>;store in Y[i],Y[i+1],Y[i+2],Y[i+3]</pre>
DADDIU	Rx, Rx, # 32	; increment index to X
DADDIU	Ry,Ry, #32	; increment index to Y
DSUBU	R20,R4,Rx	;compute bound
BNEZ	R20,Loop	; check if rdome f CPS&10T2019 page 99

Multi-Threading Options



4 Threads- hiding latencies



Going Multi-Core: from 2 to ?



Major issues:

- 1. How to design
- 2. How to program?
- 3. How to communicate?

Intel Xeon Phi count the number of cores !



Proceedings of CPS&IoT2019 page 101

Intel Xeon Phi

- Knights Landing (2015):
- build on Silvermont (Atom) x86 cores
- with 512-bit (SIMD) AVX units
- up to 72 cores / chip => 3 TeraFlops
- 14 nm
- using Micron's DRAM 3D techn. (hybrid memory cube)



GPUs: NVIDIA Volta (2017)

Combining MIMD + SIMD + MultiThreading => multiple SMs, supporting SIMT



L1 Instruction Cache

<u>https://images.nvidia.com/content/pdf/tesla/whitepaper/p</u> ascal-architecture-whitepaper.pdf

Proceedings of CPS&IoT2019 page 103

1 SM core

- Units:
 - 8 tensor cores/SM
 - 64 Int units
 - 64 FP32
 - 32 FP64
 - 32 Ld/St
 - -4 SFUs
- 128 LB L1 Data \$
- 4 warp schedulers

							L1 Instruct	ion Cache								
		104	ostruct	tion C	ache			1		104	nstruc	tion C	ache			
_	War	p Sch	edule	r (32 ti	hread	/clk)			Warp Scheduler (32 thread/clk)							
	Dispatch Unit (32 thread/clk)						1	Dispatch Unit (32 thread/clk)								
	Reg	ister	File (1	16,384	4 x 32	!-bit)			Reg	ister	File ('	16,384	4 x 32	2-bit)		
FP64	INT	INT	FP32	FP32	-			FP64	INT	INT	FP32	FP32				
FP64	INT	INT	FP32	FP32				FP64	INT	INT	FP32	FP32				
FP64	INT	INT	FP32	FP32				FP64	INT	INT	FP32	FP32				
FP64	INT	INT	FP32	FP32	TEN	SOR	TENSOR	FP64	INT	INT	FP32	FP32	TEN	SOR	TENSOR	
FP64	INT	INT	FP32	FP32	CO	RE	CORE	FP64	INT	INT	FP32	FP32	CC	DRE	CORE	
FP64	INT	INT	FP32	FP32				FP64	INT	INT	FP32	FP32				
FP64	INT	INT	FP32	FP32				FP64	INT	INT	FP32	FP32				
FP64	INT	INT	FP32	FP32				FP64	INT	INT	FP32	FP32	\square			
.D/ LD/ ST ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	SFU	LD/ LD/ ST ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	SFU	
		L0 lr	nstruc	tion C	ache	_				L0 li	nstruc	tion C	ache			
	War	L0 Ir p Sch	nstruc! Iedulei	tion C: r (32 ti	ache hread	/clk)			War	L0 li p Sch	nstruc Iedule	tion C r (32 tl	ache hread	/clk)		
	War Dis	L0 Ir p Sch spatcl	nstruci Iedulei In Unit	tion C r (32 tl (32 th	ache hread/ read/c	/cik) :ik)			War Dis	L0 In p Sch spatcl	nstruc Iedule h Unit	tion C r (32 tl (32 th	ache hread read/o	/clk) clk)		
	War Dis Reg	L0 Ir p Sch spatcl ister	nstruci Iedulei h Unit File (1	tion C: r (32 ti (32 thi 16,384	ache hread read/c 1 x 32	/clk) :lk) ?-bit)			War Dis Reg	L0 li rp Sch spatcl ister	nstruc Iedule h Unit File ('	tion C r (32 th (32 th 16,384	ache hread read/o 4 x 32	/clk) clk) 2-bit)		
FP64	War Di: Reg INT	L0 Ir p Sch spatcl ister INT	nstruct eduler h Unit File (1 FP32	tion C: r (32 th (32 th (32 th (32 th) 16,384	ache hread read/c 1 x 32	/clk) :lk) !-bit)		FP64	War Dis Reg	L0 li rp Sch spatcl ister INT	nstruc Iedule h Unit File (1 FP32	tion C r (32 th (32 th 16,384 FP32	ache hread read/d 4 x 32	/clk) clk) 2-bit)		
FP64 FP64	War Dis Reg INT	L0 Ir p Sch spatcl ister INT INT	nstruct redule h Unit File (1 FP32 FP32	tion C (32 th (32 th (32 th (32 th (32 th (32 th (32 th) (32 t	ache hread/ read/c 1 x 32	/clk) :lk) ?-bit)		FP64 FP64	War Dis Reg INT INT	L0 II p Sch spatcl ister INT INT	nstruc nedule h Unit File (* FP32 FP32	tion C r (32 th (32 th 16,384 FP32 FP32	ache hread read/d 4 x 32	/clk) clk) 2-bit)		
FP64 FP64 FP64	War Dis Reg INT INT	L0 Ir p Sch spatcl ister INT INT	nstruct edule h Unit File (1 FP32 FP32 FP32	tion C (32 th (32 th (32 th (32 th) (32 th) (3	ache hread/c 1 x 32	/clk) :lk) !-bit)		FP64 FP64 FP64	War Dis Reg INT INT	L0 II p Sch spatcl ister INT INT	nstruc hedule h Unit File (* FP32 FP32 FP32	tion C r (32 th (32 th 16,384 FP32 FP32 FP32	ache hread/ 4 x 32	/clk) clk) 2-bit)		
FP64 FP64 FP64 FP64	War Dis Reg INT INT INT	L0 Ir p Sch spatcl ister INT INT INT	edule n Unit File (1 FP32 FP32 FP32 FP32	tion C (32 th (32 th (32 th (32 th (32 th (32 th) (32	ache hread/c 4 x 32	/clk) :lk) ?-bit)	TENSOR	FP64 FP64 FP64 FP64	War Dis Reg INT INT INT	L0 In p Sch spatcl ister INT INT INT	nstruc nedule h Unit File (* FP32 FP32 FP32	tion C r (32 th (32 th 16,384 FP32 FP32 FP32 FP32	ache hread/ 4 x 32	/clk) clk) 2-bit)	TENSOR	
FP64 FP64 FP64 FP64 FP64	War Dis Reg INT INT INT INT	L0 Ir p Sch spatcl ister INT INT INT INT	FP32 FP32 FP32 FP32 FP32 FP32 FP32	tion C (32 th (32 th (32 th (32 th (32 th (32 th (32 th (32 th) (32 th	ache hread/ read/c 1 x 32	/cik) :ik) :-bit) SOR PRE	TENSOR	FP64 FP64 FP64 FP64 FP64 FP64	War Dis Reg INT INT INT INT	L0 li p Sch spatcl ister INT INT INT INT	nstruc iedule h Unit File (* FP32 FP32 FP32 FP32 FP32	tion C r (32 th (32 th 16,384 FP32 FP32 FP32 FP32 FP32	ache hread read/4 4 x 32	/clk) slk) 2-bit) SOR DRE	TENSOR	
FP64 FP64 FP64 FP64 FP64 FP64	War Dis Reg INT INT INT INT INT	L0 Ir p Sch ister INT INT INT INT INT	nstruci edulei n Unit File (1 FP32 FP32 FP32 FP32 FP32 FP32	tion C: (32 th (32 th (32 th (32 th (32 th (32 th (32 th (32 th) (32 t	ache hread/ read/c 1 x 32	/clk) :lk) t-bit) SOR DRE	TENSOR	FP64 FP64 FP64 FP64 FP64 FP64 FP64	War Dis Reg INT INT INT INT INT	L0 in p Sch ister INT INT INT INT INT	nstruc nedule File (' FP32 FP32 FP32 FP32 FP32 FP32	tion C r (32 th (32 th 16,384 FP32 FP32 FP32 FP32 FP32 FP32 FP32	ache hread fread/ 4 x 32	/clk) clk) 2-bit) ISOR DRE	TENSOR	
FP64 FP64 FP64 FP64 FP64 FP64 FP64	War Dis Reg INT INT INT INT INT	L0 in p Sch spatcl ister INT INT INT INT INT	Istruct edule h Unit File (1 FP32 FP32 FP32 FP32 FP32 FP32 FP32	tion C (32 th (32 th) (32 th)(ache hread/c 1 x 32	/clk) clk) l-bit) SOR	TENSOR	FP64 FP64 FP64 FP64 FP64 FP64 FP64	War Dis Reg INT INT INT INT INT	L0 In p Sch spatcl ister INT INT INT INT INT	Istruc iedule h Unit File (* FP32 FP32 FP32 FP32 FP32 FP32	tion C r (32 th (32 th 16,384 FP32 FP32 FP32 FP32 FP32 FP32 FP32 FP32	ache hread/ 4 x 32	/clk) Clk) ?-bit) ISOR	TENSOR	
FP64 FP64 FP64 FP64 FP64 FP64 FP64 FP64	War Dia Reg INT INT INT INT INT INT	L0 Ir p Sch spatcl ister INT INT INT INT INT INT	Istrucional Instruction Instructio Instruc	tion C: (32 th (32 th) (32 th)	ache hread/c 4 x 32 TEN CC	/clk) :lk) t-bit) SOR PRE	TENSOR	FP64 FP64 FP64 FP64 FP64 FP64 FP64 FP64	War Dis Reg INT INT INT INT INT INT	L0 In p Sch spatc ister INT INT INT INT INT INT	nstruc nedule h Unit File (* FP32 FP32 FP32 FP32 FP32 FP32 FP32 FP32	tion C (32 th (32 th 16,384 FP32 FP32 FP32 FP32 FP32 FP32 FP32 FP32	ache hread/ 4 x 32 TEN CC	/clk) slk) 2-bit) SOR PRE	TENSOR	
FP64 FP64 FP64 FP64 FP64 FP64 FP64 FP64	War Dia Reg INT INT INT INT INT INT INT INT INT	L0 Ir p Sch spatcl ister INT INT INT INT INT INT INT INT INT INT	Istruct edulee h Unit File (1 FP32 FP32 FP32 FP32 FP32 FP32 FP32 FP32	tion C: (32 th (32 th) (32 th)	ache hread/c 4 x 32 TEN CO	/clk) :lk) t-bit) SOR DRE	TENSOR CORE	FP64 FP64 FP64 FP64 FP64 FP64 FP64 FP64	War Dis Reg INT INT INT INT INT INT INT INT INT	L0 II p Sch spatc ister INT INT INT INT INT INT INT INT INT	Instruction inedule in Unit File (* FP32 FP32 FP32 FP32 FP32 FP32 FP32 FP32	tion C (32 th (32 th (32 th (32 th (32 th (32 th) (57) (57) (57) (57) (57) (57) (57) (57	ache hread/ 4 x 32 TEN CC	/clk) clk)	TENSOR CORE	
FP64 FP64 FP64 FP64 FP64 FP64 FP64 FP64	War Dis Reg INT INT INT INT INT INT INT INT	L0 Ir p Sch spatc ister INT INT INT INT INT INT INT LD/ ST	Instruction in Unit File (1 FP32 FP32 FP32 FP32 FP32 FP32 FP32 FP32	tion C: (32 th (32 th) (32 th)	ache hread/c 4 x 32 TEN CO	/clk) l-bit) SOR RE LD/ ST	TENSOR CORE SFU	FP64 FP64 FP64 FP64 FP64 FP64 FP64 FP64	War Dis Reg INT INT INT INT INT INT INT INT INT INT	L0 In pp Sch spatcl ister INT INT INT INT INT INT INT INT INT	nstruc nedule h Unit File (* FP32 FP32 FP32 FP32 FP32 FP32 FP32 FP32	tion C (32 th (32 th 16,384 FP32 FP32 FP32 FP32 FP32 FP32 FP32 FP32	ache hread/ 4 x 32 TEN CC	/clk) slk) 2-bit) SSOR DRE	TENSOR CORE	
Tensor core operation

- D = AxB + C, all 4x4 matrices
- 64 floating point MAC operations per clock



Near-Memory Computing using Stacked DRAM: 2.5D vs 3D

- Stacked DRAMs in same package as processor
 - High Bandwidth Memory (HBM)



2.5 D: using Interposer



3D computing NeuroCube (ISCA 2016)





NeuroCube computing core

- Multiple MAC units and its temporal buffer
- Packet based NoC router
- Programmable neurosequence generator
- Buffer to handle out-of-order packet arrival

Does it help? Amdahl's Law

- Sequential part limits speedup
- Example: P = 100 processors, can we get 90× speedup?

$$- T_{\text{parallel}} = T_{\text{parallel}} / 100 + T_{\text{parallel}} \frac{1}{(1 - F_{\text{parallelizable}}) + F_{\text{parallelizable}} / 100} = 90$$
$$=> F_{\text{parallelizable}} = 0.999$$

• Need sequential part < 0.1% of original time





Luckily we have Gustafson's law

- Amdahl's law: Strong scaling:
- Gustafson's law: Weak scaling
 - Problem size grows proportional with number of processors. E.g.:
 - 10 processors, 10 × 10 matrix
 - 100 processors, 32 × 32 matrix
 - Constant performance in this example (derive the Speedup formula !!)
- Speedup = $(P-1)f_{parallizable} + 1$
- Effectively: Linear speedup !!



Proceedings of CPS&IoTNumber of Processors - P

Does it help? **Check the Roofline**



Attainable GPLOPs/sec = Min (Peak Memory BW × Arithmetic Intensity, Peak FP Performance)

Proceedings of CPS&IoT2019 page 113

Rooflines



1/8

1/4 1/2

Arithmetic Intensity (Flops/Byte) Proceedings of CPS&IoT2019 page 114

Example CPS: EEG monitoring



EEG Processing steps



Brainwave monitoring system – Overview



Brainwave monitoring system – Energy breakdown

- Three system scenarios:
- 1. Off-chip processing; raw EEG data is transmitted to the nearest coordinator
- 2. On-chip seizure detection with standalone AFE + ADC; only small alarm is send
- 3. Identical to scenario 2 but with power-optimized on-chip AFE + ADC **Putting a system together:**

Description	Component	Configuration 1 (off-chip processing)	Configuration 2 & 3 (on-chip processing)	
Micro-controller	STM32F401xB with ARM Cortex-M4	Not used	Active (Lei Wang et al. (2017))	
Data acquisition	 TI ADS1299 AFE with 24-bit ADC TI ADS1298 AFE with 24-bit ADC J. Yoo et al. (2013) AFE with 10-bit ADC 	Continuous sampling of 32 EEG channels @ 100Hz		
Radio – Tx	Dialog DA14580 SoC with BLE 4 stack	Send raw EEG data	Send alarm (256B per 20s)	
SD card (logging)	Sandisk microSD 16GB	Not used	Store raw EEG data	

BrainWave system overview energy breakdown

• Estimated system energy breakdown per epoch



Proceedings of CPS&IoT2019 page 119



5/23/20149DS 1299-based Opprc Handownsoundlifier from openbci.com: signal quality for EEG registration and SSVEP Based BCP

RISC-V runtime

EEG pipeline cycle breakdown on Pulpino (Ch = 1, N = 256)



Feature selection experiment

EEG seizure classification performance



EEG seizure classification performance



Proceedings of CPS&IoT2019 page 122

Energy-efficiency RISCV only evaluation



Proceedings of CPS&IoT2019 page 123

Energy-efficiency RISCV

• Energy/cycle roughly similar for all features on RISCV



Some CPS processors

- Arduino
- Rasberry-Pie
- RISC-V Pulpino
- Jetson nano
- Experimental: CGRA

- If you need advanced Deep Learning Networks, e.g.
 - Intel Movidius
 - Google: Edge TPU
 - NVIDIA Xavier
 - Newest Xilinx FPGA
 - NVIDIA DLA (Open Source Verilog)

Conclusions

- Many processor options, also for CPS
- Energy-efficiency is major problem
- All exploit parallelism
 - many options
- Do not forget the memory hierarchy
- Compiler, mapping and optimization research needed !
 - using local memories efficiently
 - hitting the roofline
- Many MPSoCs and Development boards to get started





Implementation of HW-accelerated video-processing on industrial Zynq modules

Jiří Kadlec, Zdeněk Pohl, Lukáš Kohout







// "From the cloud to the edge - smart IntegraTion and OPtimisation Technologies for highly efficient Image and VIdeo processing Systems" //



Jiří Kadlec, Zdeněk Pohl, Lukáš Kohout

ÚTIA AV ČR v.v.i.

Proceedings of CPS&IoT2019 page 128

// Presentation Overview

- Introduction to the Xilinx SDSoC design flow
- Boards supported by UTIA in FitOptiVis:
 - **ZynqBerry** 2x A9, .5GB DDR3, .1Gb Eth
 - UltraScale4x5 4x A53, 2x R5, 2GB DDR4, 1Gb Eth
 - UltraScale8x5 4x A53, 2x R5, 4GB DDR4, 1Gb Eth
- Example of acceleration LK Dense Optical Flow
- Demonstration of board support platform generation project for Zynq Ultrascale+ module: TE0820-03-4EV-1EA on TE0701-06 carrier
- HW is from: https://www.trenz-electronic.de/





// What is Xilinx SDSoC System Lev. Compiler?

LK Dense Optical Flow on Xilinx Zynq Ultrascale (SDSoC 2017.4)







ECSEL

oint Undertaking

8.5 cm x 5.5 cm x 2 cm





// ZynqBerry









// UltraScale8x5 7.6 cm x 5.2 cm x 4.0 cm







// Comparison: PL size, Acceleration by HW

Supported module	Logic Cells (K)	Memory (Mb)	DSP Slices	GPU	VCU	Ultra RAM
ZynqBerry	28	2.1	80			
<u>UltraScale4x5</u>	<u>192</u>	<u>18.5</u>	728	<u>YES</u>	<u>YES</u>	<u>YES</u>
UltraScale8x5	747	57.7	3528	YES		YES

int 32 Z[n,n] = A[n,n]*B[n,n]	Accelerated (max) [n,n]	ARM clock MHz	Accelerator clock MHz	Acceleration relative to ARM
ZynqBerry	[18,18]	650	100	5 x
UltraScale4x5	[80,80]	<u>1200</u>	_200	<u>_21 x</u>
UltraScale8x5	[250,250]	1050	187	101 x





// Supported UltraScale5x4 Modules

<u>UltraScale4x5</u> <u>TE0820</u> <u>module</u>	Xilinx UltraScale Device	Logic Cells (K)	PL Mem (Mb)	PL DSP Slices	A53, GPU, VCU	DDR4 (GB)	Mod. Price (Euro)
02-2CG-1EA	xczu2cg-sfvc784-1-e	103	5.3	240	2,n,n	1	269
02-3CG-1EA	xczu3cg-sfvc784-1-e	154	7.6	360	2,n,n	1	319
02-4CG-1EA	xczu4cg-sfvc784-1-e	192	18.5	728	2,n,n	1	549
02-2EG-1EA	xczu2eg-sfvc784-1-e	103	5.3	240	4,y,n	1	299
02-3EG-1EA	xczu3eg-sfvc784-1-e	154	7.6	360	4,y,n	1	369
<u>03-4EV-1EA</u>	<u>xczu4ev-sfvc784-1-e</u>	<u>192</u>	18.5	<u> 728</u>	<u>4,y,y</u>	<u>2</u>	<u>669</u>

Modules & PCBs are produced & distributed by:

- Trenz Electronic <u>https://www.trenz-electronic.de/</u>
- Carrier PCBs: TE0726-03, TE0701-06, TEBF0808-04A
- Immageon FMC adapter (HDMI I/O) is from Avnet





// LK Dense Optical Flow Full HD – PC hdmi







// LK Dense Optical Flow Full HD 495x faster







// Create and compile Vivado 2018.2 project

- Create Windows setup
- _create_win_setup.cmd
- Create virtual drive X:
- _use_virtual_drive.cmd
- Create initial design in Vivado 2018.2
 X:\zusys\vivado_create_project_guimode.cmd
- Compile design to bitstream and HDF file
 TE::hw_build_design -export_prebuilt





// Created initial Vivado 2018.2 design







// HDMI_in FULL HD HDMI video input






// HDMI_in FULL HD HDMI video output







// Video DMA to 8 video frame buffers







// SDSoC 2018.2 Platform and Design Flow

LK Dense Optical Flow on Xilinx Zynq Ultrascale (SDSoC 2017.4)



ECSEL

oint Undertaking

17





// Configuration of PetaLinux 2018.2

Use of Vivado HDF in PetaLinux 2018.2 petalinux-config --get-hw-description= /home/devel/work/TS82/TE0820/zusys/prebuilt/hardware/4ev_1e

Root filesystem type for Debian platform

earlycon clk_ignore_unused root=/dev/mmcblk1p2
rootfstype=ext4 rw rootwait quiet

Compile Petalinux 2018.2

petalinux-build

Make Debian 9.8 "Stretch"

sudo ./mkdebian.sh

> Zip created *te0820-debian.img* (7 GB)

zip te0820-debian te0820-debian.img





// Create SDSoC 2018.2 platform from Vivado

- Generate FSBL and BOOT.bin
 TE::sw_run_hsi
- Generate SDSoC platform
 TE::ADV::beta_util_sdsoc_project

In SDSoC 2018.2, compile example performing int32 matrix operation:

D[75,75] = A[75,75] * B[75,75] + C[75,75]

 $\texttt{X:} \texttt{4EV-1EA} zusys \texttt{samples} z_is_a_times_b_direct_connect \texttt{}$

In SDSoC 2018.2, compile example LK Dense Optical Flow





// Matrix Mult and Add [75x75]: 33x faster







// LK Dense Optical Flow Full HD 495x faster







// FitOptiVis Design Time Resources

- FitOptiVis www:
- https://fitoptivis.eu/
- Resources on UTIA FitOptiVis www:
- > http://sp.utia.cz/index.php?ids=projects/fitoptivis
- Detailed App. Note and Eval package: Jiři Kadlec, Zdeněk Pohl, Lukáš Kohout: Design Time and Run Time Resources for Zynq Ultrascale+ TE0820-03-4EV-1E with SDSoC 2018.2 Support
- Contact: Jiří Kadlec, UTIA AV CR v.v.i. <u>http://zs.utia.cas.cz/</u> kadlec@utia.cas.cz











Dataflow-Based Toolchain for Adaptive Hardware Accelerators

Francesca Palumbo¹, Claudio Rubattu^{1,2}, Carlo Sau³, Tiziana Fanni³, Luigi Raffo³

¹University of Sassari, Intelligent system DEsign and Application (IDEA) Group ²University of Rennes, INSA Group

³University of Cagliari, Diee – Microelectronics and Bioengineering (EOLAB) Group







MDC suite

Motivations and Overview

Proceedings of CPS&IoT2019 page 150

Who and Where



Who and Where



Who and Where



UNIVERSITY OF SASSARI

Modern Embedded Systems

Embedded Systems (*real-time* computing systems with a dedicated functionality) are pervasive (*98%* of computers are embedded) and may present *sensing* and *actuating* capabilities.



Modern Embedded Systems

Embedded Systems (*real-time* computing systems with a dedicated functionality) are pervasive (*98%* of computers are embedded) and may present *sensing* and *actuating* capabilities.

	Safety	Security	Certif.	Distrib.	IMH	Seamless	MPSoC	Energy
Automotive	x	x	x	x	x	x	x	
Aerospace	х	х	х	х	х		х	x
Healthcare	х	x	х	х	х	х	х	x
Consumer					х	х	х	
IDC - Design of Future ES								



Colliding technical requirements.

Complex functionalities.



Multimedia Domain



HIGH PERFORMANCES

real time, portability, long battery life

UP-TO-DATE SOLUTIONS

last audio/video codecs, file formats...

MORE INTEGRATED FEATURES

MP3, Camera, Video, GPS...

MARKET DEMAND

convenient form factor, affordable price, fashion



A Target & Technological Challenges

DATAFLOW MODEL OF COMPUTATION

- Modularity and parallelism → EASIER INTEGRATION AND FAVOURED RE-USABILITY
- COARSE-GRAINED RECONFIGURABILITY
 - − Flexibility and resource sharing → MULTI-APPLICATION PORTABLE DEVICES



Target & Technological Challenges

DATAFLOW MODEL OF COMPUTATION

- − Modularity and parallelism → EASIER INTEGRATION AND FAVOURED RE-USABILITY
- COARSE-GRAINED RECONFIGURABILITY
 - − Flexibility and resource sharing → MULTI-APPLICATION PORTABLE DEVICES

Reconfigurable Platform Composer Tool Project

Automated **DESIGN FLOW** are fundamental to guarantee **SHORTER TIME-TO-MARKET**. Dealing with **APPLICATION SPECIFIC MULTI-CONTEXT** systems, in particular for **KERNEL ACCELERATORS**, state of the art still lacks in providing a broadly accepted solution.



The RPCT project (2012-2015) has been funded by Sardinian Regional Government (L.R. 7/2007, CRP-18324). http://sites.unica.it/rpct/



Reasons for Coarser-Grain



Reasons for Coarser-Grain



	Fine Grained	Coarse Grained
	bit-level	word-level
Flexibility	\odot	e
Speed	æ	\odot
Memory	8	e

- Coarse Grained (CG):
 - both in ASIC and FPGA
 - 1 clock cycle switching, with dedicated switching blocks.
- Fine Grained (FG):
 - FPGA only
 - switching requires a new bitstream

Baseline MDC



MDC design suite http://sites.unica.it/rpct/

13/19



MDC design suite http://sites.unica.it/rpct/

14/19

Out1

Out1

С

E

G



Additional Features



MDC design suite http://sites.unica.it/rpct/

Structural Profiler:

low-level feedback (from synthesis) and DSE for topology optimization.

•(ASIC + FPGA)

Co-Processor Generator: generation of ready-to-use Xilinx Ips •(FPGA)

Power Manager:

automatic application of clock-gating and/or power-gating. •CG (ASIC + FPGA) •PG(ASIC)







MDC suite

Context of Application

Proceedings of CPS&IoT2019 page 165

What kinds of applications can be combined with MDC?

What kinds of applications can be combined with MDC?

 Different applications with common computational operations: it is achieved by considering applications from the same application field or small actor granularities.



What kinds of applications can be combined with MDC?

 Different applications with common computational operations: it is achieved by considering applications from the same application field or small actor granularities.



 Different working points of the same applications obtained through several strategies (e.g. actor parallelization, actor variants, granularity modification, approximate computing, ...)



What kinds of applications can be combined with MDC?

 Different applications with common computational operations: it is achieved by considering applications from the same application field or small actor granularities.



EXAMPLE: Neural Signal Decoding

 Different working points of the same applications obtained through several strategies (e.g. actor parallelization, actor variants, granularity modification, approximate computing, ...)



Resource Optimization



Implantable Devices: strict area & power requirements

Resource Optimization



Implantable Devices: strict area & power requirements

Neural Signal Decoding:

- Fast
- Low Area
- Low Power



Resource Optimization

Implantable Devices: strict area & power requirements

Neural Signal Decoding:

- Fast
- Low Area
- Low Power



D. Pani, et al., «Real-time processing of tflife neural signals on embedded dsp platforms: A case study» *Neural Engineering*, 2011.

MDC can be used to build the accelerators compliant to those constraints.





	# actors	#sbox	
12 networks (dec_filter, Thr, rec_filter, NEO, idx_max_abs, Avg, sqr_sum, weight_mul, dot_prod, idx_max, sync_avg, sync_wavg)	46	0	
MDC network	14	86	



	# actors	#sbox
12 networks (dec_filter, Thr, rec_filter, NEO, idx_max_abs, Avg, sqr_sum, weight_mul, dot_prod, idx_max, sync_avg, sync_wavg)	46	0
MDC network	14	86



	# actors	#sbox	
12 networks (dec_filter, Thr, rec_filter, NEO, idx_max_abs, Avg, sqr_sum, weight_mul, dot_prod, idx_max, sync_avg, sync_wavg)	46	0	
MDC network	14	86	



20/05/2019
Multiple Working Points

- Approximate Computing: trading a controlled quality degradation (# taps) for an increased energy efficiency
- **Software Implementation**: Erwan Raffin, et al., *"Low power HEVC software decoder for mobile devices"*, JRTIP 12(2): 495-507 (2016)

Multiple Working Points

- Approximate Computing: trading a controlled quality degradation (# taps) for an increased energy efficiency
- **Software Implementation**: Erwan Raffin, et al., *"Low power HEVC software decoder for mobile devices"*, JRTIP 12(2): 495-507 (2016)



Multiple Working Points

design @200 MHz Xilinx XC7Z020	LUT	FF	BRAM	DSP	Fmax [MHz]	tap	dP (Vivado) [mW]	dE [لبا]	time per block [cycles]	# interpolated pixels in a fixed time
legacy_luma	212	37	4	16	213	8	11	0.248	460	57957
	582 8 (+175%) (+13			16 (+0%)	200 (-6%)	8	12 (+9%)	0.270 (+9%)	460 (+0%)	57957 (+0%)
reconf_luma (vs legacy %)		85	4			7	11 (+0%)	0.245 (-1%)	395 (-14%)	59033 (+2%)
		(+130%)	(+0%)			5	10 (-9%)	0.217 (-12%)	265 (-42%)	61191 (+6%)
						3	10 (-9%)	0.211 (-15%)	135 (-71%)	63357 (+9%)
legacy_chroma	163	33	2	8	217	4	9	0.053	107	14753
	383 (+1 35%)	383 65 L 35%) (+97%)		8 (+0%)	200 (-12%)	4	9 (+0%)	0.053 (+0%)	107 (+0%)	14753 (+0%)
reconf_chroma (vs legacy %)			2 (+0%)			3	8 (-11%)	0.045 (-13%)	73 (-32%)	15293 (+4%)
						2	6 (-33%)	0.033 (- 37%)	39 (-64%)	15835 (+7%)



Multiple Working Points

design @200 MHz Xilinx XC7Z020	LUT	FF	BRAM	DSP	Fmax [MHz]	tap	dP (Vivado) [mW]	dE [لىم]	time per block [cycles]	# interpolated pixels in a fixed time
legacy_luma	212	37	4	16	213	8	11	0.248	460	57957
					200	8	12 (+9%)	0.270 (+9%)	460 (+0%)	57957 (+0%)
reconf_luma	582	85 4 16	16	7		11 (+0%)	0.245 (-1%)	395 (-14%)	59033 (+2%)	
(vs legacy %)	(+175%)	(+130%)	(+0%)	(+0%)	(-6%)	5	10 (-9%)	0.217 (-12%)	265 (-42%)	61191 (+6%)
						3	10 (-9%)	0.211 (-15%)	135 (-71%)	63357 (+9%)
legacy_chroma	163	33	2	8	217	4	9	0.053	107	14753
						4	9 (+0%)	0.053 (+0%)	107 (+0%)	14753 (+0%)
reconf_chroma (vs legacy %)	383 (+1 3 5%)	65) (+97%)	2 (+0%)	8 (+0%)	200 (- 12%)	3	8 (-11%)	0.045 (-13%)	73 (-32%)	15293 (+4%)
(vs legacy /0)		(, .,	(37-7)			2	6 (-33%)	0.033 (- 37%)	39 (-64%)	15835 (+7%)



Multiple Working Points

design @200 MHz Xilinx XC7Z020	LUT	FF	BRAM	DSP	Fmax [MHz]	tap	dP (Vivado) [mW]	dE [لىم]	time per block [cycles]	# interpolated pixels in a fixed time
legacy_luma	212	37	4	16	213	8	11	0.248	460	57957
						8	12 (+9%)	0.270 (+9%)	460 (+0%)	57957 (+0%)
reconf_luma	582	85	4	16	200	7	11 (+0%)	0.245 (-1%)	395 (-14%)	59033 (+2%)
(vs legacy %)	(+175%)	(+130%)	(+0%)	(+0%)	(-6%)	6) 5	10 (-9%)	0.217 (-12%)	265 (-42%)	61191 (+6%)
						3	10 (-9%)	0.211 (-15%)	135 (-71%)	63357 (+9%)
legacy_chroma	163	33	2	8	217	4	9	0.053	107	14753
					200	4	9 (+0%)	0.053 (+0%)	107 (+0%)	14753 (+0%)
reconf_chroma	reconf_chroma (vs legacy %) (+135%) (+97%) (+0%) (+0%)	65 (+97%)	2 (+0%)	8 (+0%)		3	8 (-11%)	0.045 (-13%)	73 (-32%)	15293 (+4%)
			2	6 (-33%)	0.033 (-37%)	39 (-64%)	15835 (+7%)			



Multiple Working Points

design @200 MHz Xilinx XC7Z020	LUT	FF	BRAM	DSP	Fmax [MHz]	tap	dP (Vivado) [mW]	dE [لىم]	time per block [cycles]	# interpolated pixels in a fixed time
legacy_luma	212	37	4	16	213	8	11	0.248	460	57957
						8	12 (+9%)	0.270 (+9%)	460 (+0%)	57957 (+0%)
reconf_luma	582	85	4	16	200 (-6%)	7	11 (+0%)	0.245 (-1%)	395 (-14%)	59033 (+2%)
(vs legacy %)	(+175%)	(+130%)	(+0%)	(+0%)		5	10 (-9%)	0.217 (-12%)	265 (-42%)	61191 (+6%)
						3	10 (-9%)	0.211 (-15%)	135 (-71%)	63357 (+9%)
legacy_chroma	163	33	2	8	217	4	9	0.053	107	14753
					\frown	4	9 (+0%)	0.053 (+0%)	107 (+0%)	14753 (+0%)
reconf_chroma (vs legacy %)	383 (+1 35%)	383 65 (+135%) (+97%)	2 (+0%)	8 (+0%)	200 (-1 2 %)	3	8 (-11%)	0.045 (-13%)	73 (-32%)	15293 (+4%)
						2	6 (- 33%)	0.033 (- 37%)	39 (-64%)	15835 (+7%)



Multiple Working Points

design @200 MHz Xilinx XC7Z020	LUT	FF	BRAM	DSP	Fmax [MHz]	tap	dP (Vivado) [mW]	dE [لىم]	time per block [cycles]	# interpolated pixels in a fixed time
legacy_luma	212	37	4	16	213	8	11	0.248	460	57957
						8	12 (+9%)	0.270 (+9%)	460 (+0%)	57957 (+0%)
reconf_luma	582	85	4	16	200 (-6%)	7	11 (+0%)	0.245 (-1%)	395 (-14%)	59033 (+2%)
(vs legacy %)	(+175%)	(+130%)	(+0%)	(+0%)		5	10 (-9%)	0.217 (-12%)	265 (-42%)	61191 (+6%)
						3	10 (- 9%)	0.211 (-15%)	135 (-71%)	63357 (+9%)
legacy_chroma	163	33	2	8	217	4	9	0.053	107	14753
						4	9 (+0%)	0.053 (+0%)	107 (+0%)	14753 (+0%)
reconf_chroma (vs legacy %)	383 (+1 3 5%)	65 %) (+97%)	2 (+0%)	8 (+0%)	200 (-1 2%)	3	8 (-11%)	0.045 (-13%)	73 (-32%)	15293 (+4%)
	<u> </u>	. ,	. ,			2	6 (- 33%)	0.033 (- 37%)	39 (-64%)	15835 (+7%)





Multi Dataflow Composer







Step 1: Getting Familiar with the Orcc Environment

Proceedings of CPS&IoT2019 page 184

Model Driven Desingn

Dataflow Models



Model Driven Desingn

Dataflow Models

Several Models depending on how actors process tokens e.g. SDF has fixed token rates for reading and writing



Model Driven Desingn

RVC-CAL Dataflow Formalism

XDF Networks



<?xml version="1.0" encoding="UTF-8"?> <XDF name="Testbench"> <Instance id="src"> <Class name="common.SourceImage"/> </Instance> <Instance id="dst"> <Class name="common.ShowImage"/> </Instance> <Instance id="dut"> <Class name="baseline.Sobel"/> </Instance> <Connection dst="dut" dst-port="y" src="src" src-port="Y"/> <Connection dst="dst" dst-port="SizeOfImage" src="src" src-port="SizeOfImage"/> <Connection dst="dut" dst-port="SOI" src="src" src-port="SizeOfImage"/> <Connection dst="dst" dst-port="Y" src="dut" src-port="edgeY"/> </XDF>

CAL Actors

de	el_1
dataIn	dataOut

package common;

```
actor Delay()
uint(size=8) dataIn ==>
uint(size=8) dataOut :
```

uint(size=8) dataReg := 0;

```
action dataIn:[dataNew] ==> dataOut:[data]
var uint(size=8) data
do
data := dataReg;
dataReg := dataNew;
end
```

end

Open XDF and CAL files

edgeY

SOI



right click \rightarrow Open With

- Network/CAL Editor (graphical)
- Text editor
- System Editor

Double click on an instance (subnetwork in yellow, actor in blue) on the network editor to open it

Y SizeOfImage

src

dst

SizeOfImage

A Test Case

Edge Detection



M Test Case

Sobel XDF









Explore Sobel XDF



20/05/2019

Proceedings of CPS&IoT2019 page 194











Edge Detection



Derive Roberts from Sobel

- 1. Duplicate Sobel.xdf dataflow
- 2. Remove all unnecessary actor instances
- 3. Rename actor instances to respect name conventions
- 4. Disconnect wrong connections
- 5. Connect unconnected actor instances
- Change entity of actor instances (fwd and aln from 3x3 to 2x2 kernel actor)

Derive Roberts from Sobel

1. File Duplication

- 1. right click on Sobel.xdf \rightarrow Copy
- 2. right click on package baseline \rightarrow Paste
- 2. Remove an actor instance
 - 1. click on actor instance on the xdf graphical editor
 - 2. move the cursor on the top right corner
 - 3. click on the trash bin

(or right click on actor instance \rightarrow Delete)

Derive Roberts from Sobel

3. Rename an actor instance

- 1. slow double click on actor instance
- 2. type a new name

(or right click on actor instance → Rename)

- 4. Disconnect wrong connections
 - 1. right click on connection \rightarrow Delete

Derive Roberts from Sobel

- 5. Connect unconnected actor instances
 - 1. click on Connections under the Connection folder on the Palette tab (right side of the screen)
 - 2. click on the desired output port of the source actor instance
 - 3. drag a line to the desired input port of the destination actor instance

Derive Roberts from Sobel

- 6. Change entity of actor instances
 - right click on actor instance → Set/Update refinement...
 - 2. select the new entity (cal file for actors, xdf file for sub-networks) from the list of available ones

Dataflow Simulation



- Possibility of simulating dataflow models before going ahead with their processing
- Testbench network should not present inputs or outputs, but source and destination actors emulate the surrounding environment

Testbench

A Dataflow Simulation

SourceImage Actor

```
sendY: action ==> Y:[val]
guard open
var uint(size=8) val
do
  val := source readByte();
  if x < WIDTH-1 then
    x := x+1;
  else
    if y < HEIGHT-1 then
      y := y+1;
    else
      y := 0;
      open := false;
    end
    x := 0;
  end
end
```

- Read Y component of source image from file byte by byte
- Fixed image size to 300 pixel widht and 255 pixel height
- Infinitely read the image: after one complete read it starts again reading the image from the beginning

Dataflow Simulation

ShowImage Actor

```
recvY: action Y:[val] ==>
guard init and open
do
  pictBuffY[x+y*width] := val;
  if x < width-1 then
    x := x+1;
  else
    if y < height-1 then
      y := y+1;
    else
      y := 0;
      open := false;
      fpsPrintNewPicDecoded();
    end
    x := 0;
  end
```

- Display image on a window
- Print the time required for computation on the system console

```
displayYUV displayPicture(pictBuffY, pictBuffU, pictBuffV, width, height);
```

end

Dataflow Simulation

Device Under Test (DUT)



- DUT is a network instance to be simulated
- The Testbench.xdf file provides a ready-to-use testbench with the Sobel.xdf network as DUT
- It is possible to change the DUT in the same way we changed instance entity from one actor to another

Simulate Sobel

- Click on Run → Run Configurations... on the main menu
- 2. Double click on Orcc Simulation to create a new run configuration
- 3. Choose a Name for the new run configuration (e.g. Tutorial Simulation 1)
- 4. Select Tutorials as project referenced by the new run configuration

Simulate Sobel

- 5. Choose Visitor interpreter and debugger as simulator
- 6. Choose Testbench as xdf network to be simulated
- Choose gear.bin as input stimuli for the simulation (this is the file subjected to edge detection)

Simulate Sobel

8. Select an output file directory (no output will be generated in this case)

9. Click Run to launch the simulation

Name: Tutorial Simulation 1		
🕐 Simulation settings 🛛 🕐 Simulation options 🔲 Common		
Project:		
Tutorials	Browse	
Simulator:		
Select a simulator: Visitor interpreter and debugger $\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \$		
Options:		
XDF name: baseline.Testbench	Browse	
Input stimulus: C:\Users\cassa\Desktop\unica\ricerca\dev_MDC\repo_mdc\mdc\test\Tutorials\reference\input\gear.bin	Browse	
Output file directory: C:\Users\cassa\Desktop\unica\ricerca\dev_MDC\repo_mdc\mdc\test\Tutorials\reference\input	Browse	
Golden Reference		
Type Resizer		
Enable profiling		
Enable Traces		
	Revert	Apply
	Run	Close



Simulate Roberts

- 1. Change the DUT from Sobel to Roberts
- 2. Run again the simulation and see the difference with respect to the Sobel simulation in terms of:
 - 1. Detected edges
 - 2. Frame rate


Multi Dataflow Composer





MDC suite

Baseline MDC Datapath Merging

Proceedings of CPS&IoT2019 page 213

Baseline: Dataflow to HW



MDC design suite http://sites.unica.it/rpct/

Baseline: Dataflow to HW

Dataflow to Hardware



Baseline: Dataflow to HW

Dataflow to Hardware



Multi-Dataflow Generation



Datapath Merging Problem: Graph Model



Datapath Merging Problem: Graph Model

GRAPHS



LABELING $\pi_i : \vee_i \rightarrow T$



Datapath Merging Problem: Graph Model



Datapath Merging Problem: Graph Model



PROBLEM STATEMENT: find a **Reconfigurable Graph G** (V,E) with the minimum costs (**min|V|** and **min |E|**)

 $\forall T \in \mathbf{T}, V^{T} = \{v : \pi(v) = T\} \qquad \rightarrow \qquad |V^{T}| = \max |V_{i}^{T}|, V_{i}^{T} = \{v_{i} : \pi_{i}(v_{i}) = T\}$

Datapath Merging Problem: Graph Model



PROBLEM STATEMENT: find a Reconfigurable Graph G (V.E) with the minimum **NP-complete problem**: N. Moreano, et al., "Datapath merging and interconnection sharing for reconfigurable architectures", Symp. On System Synthesis, 2002.

20/05/2019

Moreano Algorithm

merging $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$

FEASIBLE EDGE MAPPING between $\{e_1(u,v),e_2(w,z)\}$ in E_1xE_2 , where $u,v \in V_1$ and $w,z \in V_2$, if: $\pi_1(u) = \pi_2(w)$ and $\pi_1(v) = \pi_2(z)$





Moreano Algorithm

merging $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$

FEASIBLE EDGE MAPPING between $\{e_1(u,v),e_2(w,z)\}$ in E_1xE_2 , where $u,v \in V_1$ and $w,z \in V_2$, if: $\pi_1(u) = \pi_2(w)$ and $\pi_1(v) = \pi_2(z)$



GRAPHS

Moreano Algorithm

merging $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$

FEASIBLE EDGE MAPPING between $\{e_1(u,v),e_2(w,z)\}$ in E_1xE_2 , where $u,v \in V_1$ and $w,z \in V_2$, if: $\pi_1(u) = \pi_2(w)$ and $\pi_1(v) = \pi_2(z)$



20/05/2019

Moreano Algorithm

merging $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$

FEASIBLE EDGE MAPPING between $\{e_1(u,v), e_2(w,z)\}$ in E_1xE_2 , where $u,v \in V_1$ and $w,z \in V_2$, if: $\pi_1(u) = \pi_2(w)$ and $\pi_1(v) = \pi_2(z)$



Moreano Algorithm

merging $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$

FEASIBLE EDGE MAPPING between $\{e_1(u,v), e_2(w,z)\}$ in E_1xE_2 , where $u,v \in V_1$ and $w,z \in V_2$, if: $\pi_1(u) = \pi_2(w)$ and $\pi_1(v) = \pi_2(z)$



Moreano Algorithm

merging $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$

FEASIBLE EDGE MAPPING between $\{e_1(u,v), e_2(w,z)\}$ in E_1xE_2 , where $u,v \in V_1$ and $w,z \in V_2$, if: $\pi_1(u) = \pi_2(w)$ and $\pi_1(v) = \pi_2(z)$



MDC Back-End

Platform Composer





Multi Dataflow Composer





MDC suite Baseline functionalities

High Level Synthesis (HLS) support

Proceedings of CPS&IoT2019 page 230

Communication Protocol <protocol> <sys signals> <signal id="0" net_port="clock" is_clock=""...></signal> B </sys signals> <actor> <sys signals> <signal id="0" port="clk" net_port="clock" ...></signal> </sys signals> <comm signals> <signal id="0" port="din" channel="data"...></signal> <signal id="1" port="dout" channel="data"...></signal> <signal id="2" port="wr" channel="en"...></signal> <comm signals> </actor> <predecessor> <sys signals>...</sys signals> <comm signals>...<comm signals> </predecessor> R <successor> <sys_signals>...</sys_signals> <comm signals>...<comm signals> </successor> </protocol>

CGR substrate





```
<sys_signals>
```

```
<signal id="0" net_port="clock" is_clock=""...></signal>
```

...

</sys_signals>

<actor>

<sys_signals>

<signal id="0" port="clk" net_port="clock" ...></signal>

....

</sys_signals>

<comm_signals>

<signal id="0" port="din" channel="data"...></signal> <signal id="1" port="dout" channel="data"...></signal> <signal id="2" port="wr" channel="en"...></signal>

...

<comm_signals>

</actor>

<predecessor>

<sys_signals>...</sys_signals> <comm_signals>...<comm_signals>

</predecessor>

<sys_signals>...</sys_signals>

<comm_signals>...<comm_signals>

</successor>

</protocol>

Communication Protocol





^{.....}



Xronos and Turnus for MPEG-RVC



Xronos and Turnus for MPEG-RVC



•High-Level Synthesis supports only FPGAs from one specific FPGA vendor (Xilinx)





 Platform Agnostic High-Level Synthesis: it supports any kind of FPGA from any vendor, as well as ASIC design flows



Multi Dataflow Composer





Tutorial

Step 2: Baseline MDC Datapath Merging

Proceedings of CPS&IoT2019 page 239

MDC Datapath Merging

Merging Expectations



Sobel dataflow



actor	Sobel	Roberts	NS	S
Forward2x2	0	1	1	0
Forward3x3	1	0	1	0
Delay	6	2	4	2
LineBuffer	2	1	1	1
LeftShifter	4	0	4	0
Subtractor	6	2	4	2
Adder3x1	2	0	2	0
Multiplier	2	2	0	2
Adder2x1	1	1	0	1
Sqrt	1	1	0	1
Align2x2	0	1	1	0
Align3x3	1	0	1	0
Total	26	11	19	9

NS = Non Shareable, **S** = Shareable

Merge Sobel and Roberts

- Click on Run → Run Configurations... on the main menu
- 2. Double click on Orcc Compilation to create a new run configuration
- 3. Choose a Name for the new run configuration (e.g. Tutorial Merging 1)
- 4. Select Tutorials as project referenced by the new run configuration

Merge Sobel and Roberts

- 5. Choose MDC as backend for the run configuration
- 6. Select an output file directory where generated outputs will be stored
- 7. Check the List of Networks to be Compiled and Merged box
- Make sure that the Number of Networks is 2 (we are going to merge Sobel and Roberts)

Merge Sobel and Roberts

9. Add Sobel and Roberts to the XDF List of Files
 10. Select MOREANO as Merging Algorithm
 11. Check the Generate RVC-CAL multi-dataflow

		-	
Tutorials		Browse	
Backend:			
Select a backend:	MDC ~		
Output folder:	C:\Users\cassa\Desktop\unica\ricerca\dev_MDC\outputs	Browse	
Options:			
List of Network	s to be Compiled and Merged		
Number of Net	works: 2		
XDF List of File	baseline.Sobel, baseline.Roberts	Add	
Merging Algorit	ากา		
MOREANO N			
	⊐ C-CAL multi-dataflow		
Generate RV	L multi-dataflow		
Generate RV			
Generate RV Generate HI Enable Profi	ing		
Generate RV ☐ Generate HI ☐ Enable Profi	ing		
i Generate RV Generate HI Enable Profi	ing		
i Generate RV Generate HE Enable Profi	ing		

box

RVC-CAL Multi-Dataflow



Multi-Dataflow Composer report

input networks composition

- network Sobel number of actors: 26
- network Roberts number of actors: 11
 - --> input networks total number of actors: 37

multi-dataflow network composition: 00_Sobel1Roberts1

- number of merged networks: 2,0
- number of actors: 49
 - -- original actors: 28 (57.14286%)*
 - -- sbox actors: 21 (42.857143%)*
 - -- shared actors 9 (24.324326%)**

* with respect to the multi-dataflow network number of actors
** with respect to the input networks total number of actors

Sbox Actor

```
@sbox actor Sbox1x2int16 (int ID=0, int SIZE=16)
int(size=SIZE) in1
==>
int(size=SIZE) out1,
int(size=SIZE) out2
```

```
forward0: action in1: [a] ==> out1: [a]
guard not SEL[ID]
end
```

```
forward1: action in1: [a] ==> out2: [a]
guard SEL[ID]
end
end
```

- Sbox CAL special actors are created
- One different Sbox cal actor per data size and kind of Sbox (1x2 or 2x1)

Configurator Unit

unit Configurator:

bool SEL[21] = SEL2;

// ID = 1 Sobel
bool SEL1[21] = [
 false, fal

```
// ID = 2 Roberts
bool SEL2[21] = [
    true, tru
```

- Special CAL file capable of configuring Sboxes and then the multi-dataflow
- By changing the SEL (SEL1 or SEL2) it is possible to execute
 Sobel or Roberts

Multi-Dataflow Simulation

- 1. Change the Testbench network to set the Multi-Dataflow as DUT
- 2. Run the simulation with the new DUT
- 3. Change the SEL variable in the Configurator unit from SEL1 (Sobel) to SEL2 (Roberts)
- Check results in terms of detected edges (on the display) and processing time (on the console)

Generate Platform

- 1. Open Run Configurations
- 2. Select the merging run configuration previously created (Tutorial Merging 1)
- 3. Uncheck Generate RVC-CAL multi-dataflow
- 4. Check Generate HDL multi-dataflow
- 5. Put input protocol.xml as Protocol File
- 6. Put input/lib folder as HDL component library
Check Platform Interface

// Multi-Dataflow Netwo	ork module	<protocol></protocol>
// Date: 2019/05/08 16:	03:48	<predecessor></predecessor>
//		<name>fifo_small</name> <sys_signals> <signal id="0" net_port="clock" port="clk" size="1"></signal></sys_signals>
module multi_dataflow	(<signal id="1" net_port="reset" port="rst" size="1"></signal> <comm_parameters> <parameter id="0" name="depth" value="bufferSize"></parameter></comm_parameters>
input [7 : 0] y_data, input y_wr, output y_full, input [15 : 0] SOI_data,	y_data,	<pre><pre><pre></pre></pre></pre>
output SOI_V	_full,	<actor> <sys_signals> <signal id="0" net_port="clock" port="clock" size="1"></signal> </sys_signals></actor>
output [7 : output edg input edge	0] edgeY_data, eY_wr, Y_full,	<pre><signal id="1" net_port="reset" port="reset" size="1"> <comm_signals> <signal channel="data" dir="direct" id="0" kind="input" port="" size="variable"></signal> <signal channel="data" dir="direct" id="1" kind="output" port="" size="variable"></signal> <signal channel="data" dir="direct" id="1" kind="output" port="" size="1"></signal> <signal channel="rd" dir="direct" id="2" kind="output" port="rd" size="1"></signal> <signal channel="rd" dir="direct" id="3" kind="output" port="vr" size="1"></signal></comm_signals></signal></pre>
input [7:0] I	D,	<pre><signal channel="empty" dir="direct" id="4" kind="input" port="empty" size="1"></signal> <signal channel="full" dir="reverse" id="5" kind="input" port="full" size="1"></signal> </pre>
input clock, input reset		<pre> </pre>
<i>)</i> ;		

// -

Check P	latform	Actors
---------	---------	--------

	<pre><pre>cyrotocol> </pre></pre>
Delay actor_Delay_0 (<actor> <pre> <sys_signals> <signal id="0" net_port="clock" port="clock" size="1"></signal> <signal id="0" net_port="reset" port="clock" size="1"></signal> <signal id="1" net_port="reset" port="reset" size="1"></signal> <signal id="1" net_port="reset" port="reset" size="1"></signal> </sys_signals> <comm_signals> <signal channel="data" dir="direct" id="0" kind="input" port="" size="variable"></signal> <signal channel="data" dir="direct" id="0" kind="output" port="" size="variable"></signal> <signal channel="data" dir="direct" id="1" kind="output" port="" size="1"></signal> <signal channel="rd" dir="direct" id="2" kind="output" port="rd" size="1"></signal> <signal channel="wr" dir="direct" id="3" kind="output" port="wr" size="1"></signal> <signal channel="mr" dir="direct" id="3" kind="input" port="wr" size="1"></signal> <signal channel="mr" dir="direct" id="3" kind="input" port="wr" size="1"></signal> <signal channel="mr" dir="direct" id="3" kind="input" port="mrt" size="1"></signal> <signal channel="mr" dir="direct" id="3" kind="input" port="mrt" size="1"></signal> <signal dir="direct" id="0" kind="input" net_port="clock" size="1"></signal> </comm_signals></pre> </actor>

fifo small #(

Check Platform Connections

assign fifo_small_Adder2x1_0_opA_data = Multiplier_0_prod_data; assign fifo_small_Adder2x1_0_opA_wr = Multiplier_0_prod_wr; assign Multiplier_0_prod_full = fifo_small_Adder2x1_0_opA_full;

assign fifo_small_Adder2x1_0_opB_data = Multiplier_1_prod_data; assign fifo_small_Adder2x1_0_opB_wr = Multiplier_1_prod_wr; assign Multiplier_1_prod_full = fifo_small_Adder2x1_0_opB_full;

assign fifo_small_Sqrt_0_inY_data = Adder2x1_0_sum_data; assign fifo_small_Sqrt_0_inY_wr = Adder2x1_0_sum_wr; assign Adder2x1_0_sum_full = fifo_small_Sqrt_0_inY_full;

assign sbox_0_in1_data = y_data; assign sbox_0_in1_wr = y_wr; assign y_full = sbox_0_in1_full;

assign fifo_small_Forward2x2_0_inY_data = sbox_0_out2_data; assign fifo_small_Forward2x2_0_inY_wr = sbox_0_out2_wr; assign sbox_0_out2_full = fifo_small_Forward2x2_0_inY_full;



Check Platform Configurator

// Configura // Date: 20	ator module 19/05/08 16	:03:48		Cal Configurator
//				unit Configurator:
module cor	nfigurator(input [7:0]	ID,		bool SEL[21] = SEL2;
);	ουτρυτ τεε	[20:0] sei		<pre>// ID = 1 Sobel bool SEL1[21] = [false false false false</pre>
always@(IE	0) case(ID) 8'd1: sel[0]=1'b0	begin ;	// Sobel	false, false, false, false, false, false, false, false, false, false, false, false, false, false, false, false];
	sel[20]=1'b 8'd2:	0; end begin	// Roberts	<pre>// ID = 2 Roberts bool SEL2[21] = [true, true, true, true,</pre>
	sel[0]=1'b1 sel[20]=1'b	; 1; end		true, true, true, true, true, true, true, true, true, true, true, true, true, true, true,
endcase	default:	sel= <mark>21'b</mark> x;		true]; end

20/05/2019

11 __



Multi Dataflow Composer





MDC tool Additional features

Coprocessor Generator

Proceedings of CPS&IoT2019 page 253

Ready to use Xilinx IPs



MDC design suite http://sites.unica.it/rpct/









HARDWARE ACCELERATOR/CO-PROCESSOR













Multi Dataflow Composer





Tutorial Step 3

Co-Processor Generator

Proceedings of CPS&IoT2019 page 264

- Click on Run → Run Configurations... on the main menu
- Right clock on the Tutorial Merging 1 run configuration → Duplicate
- 3. Select a new name for the duplicated run configuration (e.g. Tutorial Coprocessor 1)
- 4. Under the Generate HDL multi-dataflow section check Generate Accelerator IP box

- 5. Select MEMORY-MAPPED as Processor Coprocessor Coupling
- 6. Select ARM as Host Processor
- 7. Check the Adopt DMA box
- Leave the default Board Part and Partname (em.avnet.com:zed:part0:1.0 and xc7z020clg484-1) that are referred to the Avnet Zedboard Development board

Merge Sobel and Roberts

me: Tutorial Cop	processor 1			
Compilation sett	tings 🔍 🕐 Compilation options 🕐 Mapping) 🔲 Common			
Project:				
Tutorials		Browse		
Backend				
Select a backend	MDC			
Output folder:	C:\Users\cassa\Desktop\unica\ricerca\tutorial\codice\out_copr	Browse		
Options:				
List of Network	s to be Compiled and Merged			
Number of Net	works: 2			
XDF List of Files	baseline.sobel, baseline.Koberts	Add		
Generate RV	C-CAL multi-dataflow			
🗹 Generate HD	DL multi-dataflow			
Protocol File	C:\Users\cassa\Desktop\unica\ricerca\MDC\protocols\protocol_CAPH.xml	Browse		
HDL Compo	nent Library: C:\Users\cassa\Desktop\unica\ricerca\tutorial\codice\hcl	Browse		
Compute	Logic Regions			
Import Bu	iffer Size File List			
Import Cl	ock Domain File List			
Generate .	Accelerator IP			
Processor-	Coprocessor Coupling			
MEMORY	Y-MAPPED 🗸			
Host Proce	essor			
ARM	~			
🗹 Adopt	DMA			
Board Par	t: em.avnet.com:zed:part0:1.0			
	· ·			
Partname	: xc7z020clg484-1			
Enable Profil	ing			
			Revert	Apply
			Pum	Class
		L	Kun	Close

Check Generated HW Code

mm_accelerator.v

Top module of the accelerator instantiating all the other modules



Check Generated HW Code

mm_accelerator.v config_regs.v

AXI lite slave bus interface and configuration registers



Check Generated HW Code

mm_accelerator.v config_regs.v axi_full_ipif.v

AXI full slave bus interface (only for memory mapped coupling)



Check Generated HW Code

mm_accelerator.v config_regs.v axi_full_ipif.v local_memory.v

Local memory banks to locally store input and output data (only for memory mapped coupling)



Check Generated HW Code

mm_accelerator.v config_regs.v axi_full_ipif.v local_memory.v front_end.v

Front-end to feed the CGR datapath inputs (only for memory mapped coupling)



Check Generated HW Code

mm_accelerator.v config_regs.v axi_full_ipif.v local_memory.v front_end.v multi_dataflow.v

The CGR datapath generated by the baseline MDC feature



Check Generated HW Code

mm_accelerator.v config_regs.v axi_full_ipif.v local_memory.v front_end.v multi_dataflow.v back_end.v

Back-end to retrieve results from the CGR datapath outputs (only for memory mapped coupling)



Check Generated Drivers

#include "mm_accelerator.h"

```
int mm_accelerator_Sobel(
    // port edgeY
    int size_edgeY, int* data_edgeY,
    // port SOI
    int size_SOI, int* data_SOI,
    // port y
    int size_y, int* data_y
) {
```

two parameters for each I/O port

- number of data
- data pointer

volatile int* config = (int*) XPAR_MM_ACCELERATOR_0_CFG_BASEADDR;

```
// configure I/O
*(config + 1) = size_y - 1;
*(config + 3) = size_edgeY - 1;
*(config + 2) = size_SOI - 1;
```

 configure registers with number of data for each I/O port

Check Generated Drivers

// send data port y

((volatile int) XPAR_AXI_CDMA_0_BASEADDR + (0x04>>2)) = 0x00000002; // verify idle
//*((volatile int*) XPAR_AXI_CDMA_0_BASEADDR + (0x00>>2)) = 0x00001000; // irq en (optional)
((volatile int) XPAR_AXI_CDMA_0_BASEADDR + (0x18>>2)) = (int) data_y; // src
((volatile int) XPAR_AXI_CDMA_0_BASEADDR + (0x20>>2)) =

XPAR_MM_ACCELERATOR_0_MEM_BASEADDR + MM_ACCELERATOR_MEM_1_OFFSET; *((volatile int*) XPAR_AXI_CDMA_0_BASEADDR + (0x28>>2)) = size_y*4; // size [B] while((*((volatile int*) XPAR_AXI_CDMA_0_BASEADDR + (0x04>>2)) & 0x2) != 0x2);

// send data port SOI

((volatile int) XPAR_AXI_CDMA_0_BASEADDR + (0x04>>2)) = 0x00000002; // verify idle //*((volatile int*) XPAR_AXI_CDMA_0_BASEADDR + (0x00>>2)) = 0x00001000; // irq en (optional) *((volatile int*) XPAR_AXI_CDMA_0_BASEADDR + (0x18>>2)) = (int) data_SOI; // src *((volatile int*) XPAR_AXI_CDMA_0_BASEADDR + (0x20>>2)) = XPAR_MM_ACCELERATOR_0_MEM_BASEADDR + MM_ACCELERATOR_MEM_2_OFFSET;

((volatile int) XPAR_AXI_CDMA_0_BASEADDR + (0x28>>2)) = size_SOI*4; // size [B]

while((*((volatile int*) XPAR_AXI_CDMA_0_BASEADDR + (0x04>>2)) & 0x2) != 0x2);

send input data by means of DMA

Check Generated Drivers

// start execution (check matching ID
*(config) = 0x1000001;

// wait for completion
while(((*(config)) & 0x4) != 0x4);

launch execution and poll ready register

return 0;

}

receive output data by means of DMA

Run Generated Scripts

- 1. Open Vivado
- 2. Change directory to the output of the MDC co-processor generator run

Tcl Console	? _ 🗆 🖒 X
Q, X, ♦ II 🗉 🖩 🖻	
start_gui	
<	>
cd C:/Users/cassa/Desktop/unica/ricerca/tutorial/codice/out_copr/	\otimes

Run Generated Scripts

Run the generate_ip.tcl script to create
 Vivado project and package the accelerator
 into a Xilinx compliant IP



Run Generated Scripts

 Run the generate_top.tcl script to generate a Vivado project integrating the MDC accelerator with the desired processor



Run Generated Scripts

5. Check the system with the Block Design view



Example of Usage

Run Demo



A Example of Usage

Run Demo



Exploitation of MDC

CERBERO H2020 Project

EU Commission for funding the **CERBERO** (Cross-layer modEl-based fRamework for multi-oBjective dEsign of Reconfigurable systems in unceRtain hybRid envirOnments) project as part of the H2020 Programme under grant agreement No 732105.

Coordinator:

Michal Masin (IBM), michaelm@il.ibm.com Scientific Coordinator:

Francesca Palumbo (UniSS), fpalumbo@uniss.it

Innovation Manager:

Katiuscia Zedda (Abinsula), katiuscia.zedda@abinsula.com

Dissemination-Communication Manager:

Francesco Regazzoni (USI), francesco.regazzoni@usi.ch


Exploitation of MDC

FitOptiVis H2020 Project

EU Commission for funding the *FitOptiVis* (*From the cloud to the edge: smart IntegraTion and OPtimization Technologies for highly efficient Image and VIdeo processing Systems*) project as part of the H2020 Programme under grant agreement No XXX.

Coordinator:

. . .

...

. . .

Scientific Coordinator:

Innovation Manager:

Dissemination-Communication Manager:



- 1. Sau C, et al., "Challenging the Best HEVC Fractional Pixel FPGA Interpolators With Reconfigurable and Multi-frequency Approximate Computing", IEEE ESL 2017
- 2. Palumbo F., et al., "Power-Awarness in Coarse-Grained Reconfigurable Multi-Functional Architectures: a Dataflow Based Strategy", JSPS 2017
- 3. Sau C., et al., "Automated Design Flow for Multi-Functional Dataflow-Based Platforms", JSPS 2015
- 4. Sau C. et al., "*Reconfigurable Coprocessors Synthesis in the MPEG-RVC Domain*", ReConFig Conf. 2015
- 5. Palumbo F., et al., *"The multi-dataflow composer tool: generation of on-the-fly reconfigurable platforms"*, JRTIP 2014



Multi Dataflow Composer





http://sites.unica.it/rpct/ **Thank you** <u>crubattu@uniss.it</u> carlo.sau@diee.unica.it

Francesca Palumbo¹, Claudio Rubattu^{1,2}, Carlo Sau³, Tiziana Fanni³, Luigi Raffo³

¹University of Sassari, Intelligent system DEsign and Application (IDEA) Group ²University of Rennes, INSA Group

³University of Cagliari, Diee – Microelectronics and Bioengineering (EOLAB) Group





Megamodeling of complex, distributed, heterogeneous CPS systems

Eugenio Villar University of Cantabria





Agenda

- Introduction
- Single-Source Design Approach
- Model-driven Analysis and Design for the IoE
- Conclusions





Introduction

- Model-Driven Design (MDD)
 - High-abstraction level
 - Mature SW engineering methodology
- State-of-the-Art
 - Matlab-Simulink
 - Proprietary, only one MoC, M language
 - CoFluent
 - Proprietary, a few MoCs, C/C++ language
 - Ptolemy II
 - Academic, any MoC, C/C++ inside a Java block





Introduction

- UML
 - Standard, any (user-defined) MoC, any language
 - Natural way to capture system architecture



- Semantic lacks
- Domain-specific profiles
- MetaMorph
 - OpenSource, any (user-defined) MoC, language agnostic







Introduction





Single-Source Embedded System Design

• S3D





Proceedings of CPS&IoT2019 page 293





Model-Driven Analysis and Design of IoT Systems

- Programming the Internet of Everything
- Services provided on computing platforms of many kind







- Programming the Internet of Everything
- Services provided on computing platforms of many kind





- Programming the Internet of Everything
- Services provided on computing platforms of many kind





- UML/MARTE System Modeling Methodology
- Platform-Independent
- Component-Based
 - Supporting
 - Object-Orientation
 - Actor-Orientation







- UML/MARTE System Modeling Methodology
- Platform-Independent
- Component-Based





- UML/MARTE System Modeling Methodology
- Platform-Independent
- Component-Based
 - Supporting
 - Object-Orientation
 - Actor-Orientation







Model-Driven Analysis of IoE Services

- Properties of the Provided Port
 - NotAttendedService
 - Retry
- Properties of the Interface Methods
 - concurrency
 - exekind
 - syncKind
- Properties of the Required Port
 - queueSize
 - FullPoolPolicy





Model-Driven Analysis of IoE Services

Function Call/RPC/RMI

Required Port		RtService			Provided Port		
NotAttendedService	retry	concurrency	exekind	syncKind	queueSize	FullPoolPolicy	МоС
infiniteWait	none	G or C	rem.lm.	sync.	none	none	exactly once
infiniteWait	none	G or C	rem.lm.	async.	none	none	at most once
dynamic	none	G or C	rem.lm.	sync.	none	none	exactly once
dynamic	none	G or C	rem.lm.	async.	none	none	at most once
timedWait	0	G or C	rem.lm.	sync.	none	none	exactly once
timedWait	0	G or C	rem.lm.	async.	none	none	at most once
timedWait	> 0	G or C	rem.lm.	sync.	none	none	at least once
timedWait	> 0	G or C	rem.lm.	async.	none	none	maybe once

Rendezvous

Required Port		RtService			Provided Port			
	NotAttendedService	retry	concurrency	exekind	syncKind	queueSize	FullPoolPolicy	МоС
	infiniteWait	none	G or C	rem.lm.	rendezvous	none	none	CSP
	timedWait	0	G or C	rem.lm.	rendezvous	none	none	RV
	timedWait	> 0	G or C	rem.lm.	rendezvous	none	none	RV







Model-Driven Analysis of IoE Services

Data-Flow

Required Port		RtService			Provided Port			
	NotAttendedService	retry	concurrency	exekind	syncKind	queueSize	FullPoolPolicy	MoC
	infiniteWait	none	G or C	deferred	async.	> 0	block	KPN/SDF
	infiniteWait	none	G or C	deferred	async.	> 0	(any other)	DF
	dynamic	none	G or C	deferred	async.	> 0	any	DF
	timedWait	0	G or C	deferred	async.	> 0	any	DF
	timedWait	> 0	G or C	deferred	async.	> 0	any	DF

Discrete-Event/Time-Triggered/Timed Data-Flow

Required Port		RtService			Provided Port		
NotAttendedService	retry	concurrency	exekind	syncKind	queueSize	FullPoolPolicy	МоС
dynamic	none	G or C	rem.lm.	async.	none	none	DE/TT/TDF







Performance Analysis

- Problem Statement
 - Fast Simulation & Performance Analysis
 - Before full SW Development
- Native Simulation
 - Host-Compiled







Proceedings of CPS&IoT2019 page 303



Performance Analysis

$T_{B}()$ is a function of Native Simulation # of binary instructions Global variable type of instructions int Sim_Time = 0; Overflow = 0; # of cache misses s = 1L; + Sim Time += $T_B()$; frequency for (i = 0; i < L subfr; i++) { Carry = 0;s = L_macNs(s, xn[i], y1[i]); even Sim Time += $T_B()$; if (Overflow != 0) { data dependencies \rightarrow Sim Time += T_B(); break; }} if (Overflow == 0) { exp_xy = norm_l(s); \rightarrow Sim Time += T_B(); if (exp xy <= 0) $T_{SYS}()$ is a function of xy = round(L_shr (s, -exp_xy)); preemptions \rightarrow Sim Time += T_B(); else conflicts in the bus xy = round(L_shl (s, exp_xy));|} → Sim_Time += T_B(); mutex_lock(mutex_name); wait included \rightarrow Sim Time += T_{SYS}(); **ECSEL JU** MegaM@Rt²

Proceedings of CPS&IoT2019 page 304



SW Synthesis

- Functional synthesis
 - 'main' functions
 - Static concurrency
 - Platform-Specific code
 - Optimized C code for DSPs
 - OpenCL/GL for GPUs
 - C/C++ & OpenMP for SMPs...









SW Synthesis

- Communication synthesis
 - Architectural mapping
 - Memory space
 - OS
 - Processing node
 - Benefits / Drawbacks
 - Communication Speed
 - Memory protection
 - Memory/cache use

ECSEL JU

- Scheduling
- Parallelism...







Conclusions

- The IoE demands new CPSoS design methods and tools
- Model-Driven system design is a powerful candidate
 - A CPSoS system modeling language is required
 - Supporting Mega-Modeling
 - Analysis & design of the whole IoE service
- Single-Source Approach
- S3D Demo





Execution of software models

Jesús Gorroñogoitia Atos Research & Innovation (ARI)

CPS&IoT 2019 Summer School



Outline

- Models, Metamodels & Model Driven Engineering
- Executable models
- fUML (Foundational UML)
- Papyrus MOKA
- Co-Simulation with MOKA
- Demo



Models

- Abstraction: capability of finding the commonality in many different observations of a physical domain:
 - generalize specific features of real objects (generalization)
 - classify the objects into coherent clusters (classification)
 - aggregate objects into more complex ones (*aggregation*)...
- Model: a simplified or partial representation of reality, defined to accomplish a task or to reach an agreement



Ref: M.Brambilla, J.Cabot, M. Wimmer. Model-driven Software Engineering in practice. Morgan & Claypool. 2017

FCSFL .

Metamodels

- Is a model of a model in a particular **domain**
- A conceptual model which defines concepts, relationships, and semantics and enables creation of concrete models



 A metamodel is a representation (a model) of a modeling language; it formalizes the aspects and the concepts used by a modeling language, and models the domain in question



OMG Metamodel Architecture





Ref: https://www.omg.org/ocup-2/documents/Meta-ModelingAndtheMOF.pdf



Model Driven Engineering

ECSEL JU

• Model is the central artifact of software development





Ref: M.Brambilla, J.Cabot, M. Wimmer. Model-driven Software Engineering in practice. Morgan & Claypool. 2017

Simulation-driven design (SDD) process





Ref: Glidden, P. (1993) Simulation driven: board design process automation, Conference Record Northcon, 12-14 October, pp.22-26.

ECSEL JU

UML metamodel

Structural vs **behavioral** representations

- *Structural*: class, component, deployment diagrams, etc.
- *Behavioral*: sequence, activity, state machine, etc.
- 4+1 Architecture View Model



Model vs Program (I)

- A computer **program** is a collection of instructions¹ that performs a specific task when executed by a computer
- A program is a model
 - A **program** is a **model** instance of an specific programming language (**metamodel**)
 - Python, Perl, Java, C++ are general purpose modeling languages
- A UML model is also a executable program







Proceedings of CPS&IoT2019 page 317

Executable models (I)

A UML model can be executed!, thanks to:

- fUML¹ (Foundational UML): is an executable subset of standard UML that can be used to define, in an operational style, the structural and behavioral semantics of systems, and
- ALF² (Action Language for Foundational UML): provides a textual notation, and
- And specific UML model execution standards:
 - PSCS³: Precise Semantics of UML Composite Structures
 - PSSM⁴: Precise Semantics of UML State Machines



Executable models (II)



Executable models (II)

Benefits of executing models:

- No pre-compilation/encoding required
- **High level abstraction execution** (not platform specific architecture)
- Visual execution
 - understandable for non-technical stuff
 - usefull for communication purposes
- Fast prototyping (at design-time)
- Post code generation (program framework)


fUML/ALF implementations

• fUML reference implementation:

http://portal.modeldriven.org/content/fuml-reference-implementationdownload

Alf reference implementation:

https://modeldriven.github.io/Alf-Reference-Implementation/

- fUML/Alf based tools:
 - Cameo Simulation Toolkit (Commercial):

https://www.magicdraw.com/simulation

• Papyrus/MOKA (OSS, EPL):

http://www.papyrusuml.org/

 IBM Rational Software Architect Simulation Toolkit (Commercial): <u>https://www-</u>

01.ibm.com/software/rational/products/swarchitect/simulation/

Ref: https://modeling-languages.com/list-of-executable-uml-tools/





Papyrus/MOKA

- Papyrus is an open-source Eclipse UML modeling environment
- MOKA is its fUML execution engine that offers:
 - a simulation-driven design process: (model/execute/observe/refine)+
 - Model execution control and debug
 - Execution **observation** (diagram animation, objects inspection)
- It is compliant with fUML and PSCS/PSSM standards
- It supports for FMI¹ Co-Simulation standards:
 - fUML to FMU²s export
 - FMU import, co-simulate, and visualize



Co-Simulation with MOKA

- Co-simulation¹ is an approach for the joint simulation of models developed with different tools (tool coupling) where each tool treats one part of a modular coupled problem
- FMI (Functional Mockup Interface):
 - is an emerging standard for co-simulation

ECSEL JU

- permit the inteoperability of compliant modeling/simulation tools
- UML/MARTE can be applied to design parts of CPS
- Papyrus/MOKA provides FMI support (incubating)



[Ref]: S. Guermazi, et al. "Papyrus tool support for FMI. Tutorial", Modprod2016, 2017

[1]: Bastian, J., Clauß, C., Wolf, S., & Schneider, P. (2011, June). Master for co-simulation using FMI. In Proceedings of the 8th International Modelica Conference; March 20th-22nd; Technical University; Dresden; Germany (No. 63, pp. 115-120). Linköping University Electronic Press.



DEMO









Thank you

Questions?

Megam@Rt² web-site: <u>https://megamart2-ecsel.eu/</u> contact: jesus.gorronogoitia@atos.net





max-plus-linear models for cyber-physical and embedded systems

CPS&IoT'2019, Budva, Montenegro, June 10-14, 2019

Marc Geilen, Mladen Skelin, Hadi Alizadeh, Bram van der Sanden, João Bastos

Dept. Electrical Engineering, Electronic Systems Group

"'T is hard, he cries, to bring to sudden sight ideas that have winged their distant flight", Homer, The Odyssey

modelling timed discrete event systems

- modelling, analysis, synthesis for timed discrete
 event systems are hard
 - they are complex, concurrent, non-deterministic
 - state-space explosion
 - modelling time is complicated
- exploit linearity to improve scalability
 - max-plus linear algebra
 - may good properties that are rarely exploited
- see the work of Bram van der Sanden and João Bastos on wafer logistics



overview

- linear systems
- discrete-event systems
- linear discrete-event systems
- models with (some) non-determinism
- performance analysis and controller synthesis

linear systems

where life is good

example linear system: a spring

ал.М. Кляг L-47/14Ю. Я.И. 372 А.М. Кляг L-47/14Ю. Я.И. 372



more examples of linear systems



electronic components

mechanical components



linear systems

a system S is linear if

$$u_1[k] \xrightarrow{s} y_1[k]$$
 and $u_2[k] \xrightarrow{s} y_2[k]$

implies

$$c \cdot u_1[k] \xrightarrow{S} c \cdot y_1[k]$$
 (homogeneity)

$$u_1[k] + u_2[k] \xrightarrow{s} y_1[k] + y_2[k]$$
 (superposition)

why are linear systems so great?



the good stuff...

impulse response analysis



the good stuff...

spectral analysis



linear systems, canonical form



$$\mathbf{x}[k+1] = \mathbf{A}\mathbf{x}[k] + \mathbf{B}\mathbf{u}[k],$$
$$\mathbf{y}[k] = \mathbf{C}\mathbf{x}[k] + \mathbf{D}\mathbf{u}[k]$$





$$v[k+1] = \frac{RC - T}{RC} v[k] + \frac{T}{C} i[k]$$

stability

$$\mathbf{x}[k+1] = \mathbf{A}\mathbf{x}[k] + \mathbf{B}\mathbf{u}[k],$$
$$\mathbf{y}[k] = \mathbf{C}\mathbf{x}[k] + \mathbf{D}\mathbf{u}[k]$$



stability

the system is stable iff all eigenvalues λ of **A** are such that $|\lambda| \leq 1$

the good stuff...

- we know everything there is to know
 - most analysis is efficient
- if real life is not linear, engineers will make it, approximately, linear
 - *"in engineering, model fidelity is a two-way street"* Edward A. Lee, EMSOFT 2015,
 - make the physical world fit your model, instead of making your model fit reality

discrete-event systems

where life is so hard on us



modelling trains as discrete events





a 'self-timed execution'

AMS T_1 0 trains are leaving as soon as possible 1 3 $\circ T_3$ T_2 THG 0 $\mathbf{0}$ $O{}_{T_4}$ 2 2 10 15 20 5 0 T1 T2 MST 0 Т3 T4

linear discrete-event systems

our models can be linear too!

a mathematical model for discrete events

$$w[k] = x[k] + 3$$
$$z[k] = \max(x[k] + 3, y[k] + 5) + 4$$
$$= \max(7 + x[k], 9 + y[k])$$

- systems with deterministic synchronization and deterministic delay
- quite restrictive
 - but both can be relaxed...
- input/output behavior described with max and addition



max plus algebra

we know classical linear algebra with $(\mathbb{R}, +, \times)$

$$a \times (b + c) = a \times b + a \times c$$

$$a + 0 = a, \qquad a \times 0 = 0$$

$$a \times 1 = a$$

multiplicative zero-element '0' multiplicative unit-element '1'

multiplicative zero-element ' $-\infty$ '

multiplicative unit-element '0'

another algebraic semi-ring structure: max-plus algebra replace + with max, × with +, add $-\infty$: ($\mathbb{R} \cup \{-\infty\}, \max, +$)

 $a + \max(b, c) = \max(a + b, a + c)$ $\max(a, -\infty) = a, \qquad a + (-\infty) \coloneqq -\infty$

a + 0 = a

many properties of linear algebra carry over to max plus linear algebra (not everything, max lacks inverse elements)

max-plus linear algebra

notation

 $x = b \otimes u_1 \oplus c \otimes u_2$ $x = \max(b + u_1, c + u_2)$

- there are various max-plus (or min-plus) based models in use
 - timed dataflow graphs
 - some timed Petri-nets
 - network calculus and real-time calculus
 - resource models, latency-rate
 - activity model
- extended to matrices and vectors as usual

max-plus linear systems

if

$$u_1[k] \xrightarrow{s} y_1[k]$$
 and $u_2[k] \xrightarrow{s} y_2[k]$

then

$$c \otimes u_1[k] \xrightarrow{s} c \otimes y_1[k]$$
 (homogeneity)

$$u_1[k] \oplus u_2[k] \xrightarrow{s} y_1[k] \oplus y_2[k]$$
 (superposition)

linear systems

 $c \otimes u_1[k] \xrightarrow{s} c \otimes y_1[k]$ (homogeneity)







$u_1[k] \oplus u_2[k] \xrightarrow{s} y_1[k] \oplus y_2[k]$ (superposition)



superposition

in a linear system, the response to the sum of two stimuli is the sum of the responses to the individual stimuli

in a max-plus linear system, the response to the maximum of two stimuli is the maximum of the responses to the individual stimuli

I didn't believe it, at first, when I learned about linear systems

I didn't believe it, again, when I learned about discrete event systems

• *"the trains interact, you can't analyze them separately"?*



spectral analysis

how about eigenvalues and eigenvectors?

$$\begin{bmatrix} -\infty & 1 & 3 & -\infty \\ 1 & -\infty & -\infty & -\infty \\ 4 & -\infty & -\infty & 4 \\ 4 & -\infty & -\infty & 4 \end{bmatrix} \begin{bmatrix} -1 \\ -4 \\ 0 \\ 0 \end{bmatrix} = 4 \otimes \begin{bmatrix} -1 \\ -4 \\ 0 \\ 0 \end{bmatrix}$$

a periodic time table

max. 'throughput', period is 4 (bottleneck Maastricht-Eindhoven)



impulse response

- how about impulse response analysis?
- the impulse: one train from Liège at midnight
 - no other constraints

$$\begin{bmatrix} -\infty \\ -\infty \\ -\infty \\ -\infty \\ -\infty \end{bmatrix}, \begin{bmatrix} -\infty \\ -\infty \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 5 \\ -\infty \\ 6 \\ 6 \\ 6 \end{bmatrix}, \begin{bmatrix} 9 \\ 6 \\ 10 \\ 10 \\ 10 \end{bmatrix}, \begin{bmatrix} 13 \\ 10 \\ 14 \\ 14 \end{bmatrix}, \dots$$





superposition

how about superposition?

stimulus 1: a periodic time table

stimulus 2: the second train from Liège has two hours delay

superposition says: the actual schedule must be the maxium of both schedules



impulse response





long behaviors

$$\mathbf{x}[k+1] = \mathbf{G} \otimes \mathbf{x}[k]$$
$$\mathbf{G} = \begin{bmatrix} 1 & -\infty & -\infty \\ 7 & 4 & -\infty \\ 11 & 8 & 2 \end{bmatrix}$$

repetition of the behavior is very easy:

$$\mathbf{x}[k+1000] = \mathbf{G}^{1000} \otimes \mathbf{x}[k]$$

 \mathbf{G}^N can be computed in $O(\log N)$ time


a convolution filter

- filtering video stream of 1024 by 1024 pixels
 - with different behavior for the border pixels
- about four million actor firings per frame
- states before and after a frame can be related by the matrix

 $\mathbf{F} = \mathbf{A}^{2048} (\mathbf{A}^2 \mathbf{B}^{1022} \mathbf{C}^2)^{1022} \mathbf{C}^{2048}$

- final state can be determined by some form of operational model, but that would take a very long time
- can be computed in **linear time** in the length of the expression





linear systems and stability

can the system explode? how robust is it against disturbances?

stability

- follows from eigenvalue analysis
 - the eigenvalue of our matrix was 4
- for a time table with a period μ , the eigenvalue is

 $\lambda = 4 - \mu$

• stability requires

 $\lambda \leq 0$

• compare traditional linear systems: $|\lambda| \leq 1$

delays

- the 3 hour time table is **unstable**, $\lambda = 4 3 > 0$
 - only one train every 4 hours
 - time table cannot be kept





AMS

0

 OT_3

OT

MST

2

2

 T_1

 T_2

THG

delays

- the 4 hour time table is marginally stable
- the second train from Liège is delayed by 2 hours
 - the schedule cannot recover





delays

- the 5 hour time table is **stable**
- the second train from Liège is delayed by two hours
 - the schedule does recover





non-deterministic systems

because many systems are not deterministic (linear) after all

Proceedings of CPS&IoT2019 page 363

modelling non-determinism

- fast analysis of deterministic systems is nice, but many systems are not deterministic
- among those, however, many are largely, but not entirely, deterministic
 - in a video decoder, frames may use **arbitrary encodings**
 - in a Wifi receiver frames may have arbitrary length
 - in a production line products or materials may arrive in arbitrary order

the switched linear system model

- the traditional solution in linear systems theory
 - the switched linear model
- non-deterministic choice between deterministic (linear) behaviors
- a finite state automaton defines which sequences of linear behaviors are valid



wafer logistics

- classical finite state automata describe the nondeterministic sequences of activities
- every atomic transition abstracts an entire activity
 - as a linear transformation
 - represented by a matrix



wafer logistics

- resources, peripherals and actions
- actions
- max-plus linear state vector includes the times that resources become available for the next activity



activity sequence

Gantt chart of the sequential execution of the activities $C\cdot A\cdot D$













L.J. van der Sanden, "Performance Analysis and Optimization of Supervisory Controllers", Ph.D. thesis, Eindhoven University of Technology, to be published Proceedings of CPS&IoT2019 page 368

common optimization techniques

- all standard techniques can be used on the automata
- synchronous composition
- verification, model-checking
- standard optimizations
 - for instance, partial order reduction
 - preserving performance: throughput and latency
- supervisory controller synthesis
- on a much smaller state-space
 - see theses of Bram and João





wafer logistics example

- synthesized supervisory controller for a wafer logistics system
- 2190 states and 6969 transitions



conclusion

- linearity is an often overlooked property of many timed discrete-event systems
- based on max-plus algebra
- many concepts from linear system theory have an interesting application for timed discrete event systems
- non-deterministic behavior can be incorporated in the form of switched linear system
 - even time-dependent behavior can be incorporated





Comprehensive Modular V&v Framework For Automated Cyber-physical Systems

CPS&IOT'2019 | Summer school | Tutorial

Michael Paulweber & Andrea Leitner



Proceedings of CPS&IoT2019 page 372





ADAS / AD VALIDATION CHALLENGES

VALIDATION FRAMEWORK FOR HIGHLY AUTOMATED VEHICLES | CPS&IOT Proceedings Buff@BS&IoT2019 page 373

How To Avoid This? And Achieve That?















Building blocks of automated driving







Vehicle and environment are interacting







Vehicle and environment are interacting









Check is promised safety level above driving by humans is reached

- Evidence is needed that risk does not exceed today level of risk (reference)
- But what is the safety reference for validation?

Source: Prof.Winner (TUD Darmstadt), TRB Annual Meeting, Sunday Workshop, Washington D.C.; Jan. 8, 2017



References for Safety Validation from Manual Driving



Reference variants:

- Possible safety references vary by several orders of magnitude, both far above and below today's reference safety values.
- Progress in safety validation for automated vehicles must be measured in comparison with today's risk values.
- At least two relevant metrics must to be measured:
 - accidents with personal injuries
 - accidents with fatalities

- Source: Prof.Winner (TUD Darmstadt), TRB Annual Meeting, Sunday Workshop, Washington D.C.; Jan. 8, 2017
- Present day driving tests are far from collecting enough data to cover the reference risk figures

Numbers for Autobahn in Germany 2014

Accident category	Distance between accidents	Test-drive distance
with injuries	12·10 ⁶ km	240·10 ⁶ km
with fatalities	660 · 10 ⁶ km	13.2·10 ⁹ km



Validation Challenges of Automated Vehicles



- Automated systems are most complex cyber physical systems
- Environment is part of technical system
- Uncountable number of scenarios
- Critical scenarios occur only rarely
- Sensors to detect environment imperfect (e.g. in rough weather conditions)



- Only testing of physical systems not enough
- Virtual and physical testing required
- Artificial intelligence requires new validation methods







Potential Acceleration measures for ADAS/AD System Validation:

- 1. Virtual Validation: Perform tests in virtual environment using high performance parallel computing
- 2. Select relevant Scenarios: Test only relevant scenario from real world driving (which may case safety issues)



Acceleration measures :

- 3. Identify edge-cases in virtual environment
- 4. Test edge-cases using real sensors
- Use road testing to validate virtual tests (models, scenarios)

Problem:

- Excellent simulation models of vehicle, driver, sensors as well as replica of ADAS/AD SW strategy required
- Otherwise "another" vehicle is validated





Scenario Based Validation: Building Blocks







Scenarios and Scenario Classes





Scenario: all parameters instantiated – e.g specific velocities and distances
Scenario class: parameter ranges – e.g. velocity and distance ranges





1) 🐟

17











11-1 <

HEALTH RAIL MARITIME AEROSPACE AUTOMOTIVE FARMING

ENABLE S3

















Scenario based ADAS/AD Validation Tool Chain







ADAS/AD Validation Tool Chain Architecture









SENSOR MODELING AND PARAMETRIZATION

VALIDATION FRAMEWORK FOR HIGHLY AUTOMATED VEHICLES | CPS&IOT Proceedings Buff@BS&IoT2019 page 392

M.PAULWEBER, A.LEITNER | 22
Scenario Simulation Uses Open Simulation Interface (OSI)





OpenSimulationInterface (OSI)

- Standardize ground truth data
- Standardize sensor data exchanged by different simulation models





Difficult to validate such scenarios in real driving tests

Robust





Critical Driving Scenarios

- Lane-cutting of other vehicles
- Road and traffic obstacles
- Non-compliant driver

Critical Weather Scenarios

- Backlight conditions
- Rain conditions
- Snow conditions

Source: RobustSense research project funded by European Commission and National funding authortities





Transfer to Simulation Can Lose Relevant Details required for Sensor Performance







Source: RobustSense research project funded by European Commission and National funding authortities



Challenges for Sensor Simulation



Transfer to Simulation Can Lose Relevant Details required for Sensor Performance





Source: RobustSense research project funded by European Commission and National funding authortities



Connect virtual world with real world to overcome sensor model weaknesses



Virtual world:

Simulation model

Real world:

- Interface to stimuli
- Conversion: virtual signal to physical signal (Stimulus)
- Real component
- IO bus connects real to virtual world





Sensor Model (M) Types, Sensor Stimulus (St)



Sensor Model Types

SM1	Ideal Sensor model
SM2a	Phenological Sensor model (no Sensor SW available)
SM2b	Phenological Sensor model (Sensor SW available)
SM3	Physical based sensor model (Sensor SW available)

Sensor Stimulus

St Sensor stimulus (Real physical sensor available)



VALIDATION FRAMEWORK FOR HIGHLY AUTOMATED VEHICLES | CPS&IOT Proceedings of 1028 & 1072019 page 399

Sensor Model Parameterization







Source: https://github.com/OpenSimulationInterface

Open Simulation Interface

- Consists of two individual interfaces (entry points) for object data.
- Implementation based on protocol buffers library.

osi::GroundTruth

- Generic object output of the simulation framework.
- World / global reference frame.
- Comprehensive description of the virtual environment including all relevant object data required by statistical sensor models.

osi::SensorData

- Direct input and output of statistical sensor model(s).
- Input for the environment model.
- Sensor reference frame.
- Description of the sensor output including uncertainties.

Source: https://www.hot.ei.tum.de/fileadmin/tueihot/www/Forschung/OSI_description.pdf





SCENARIO GENERATION

VALIDATION FRAMEWORK FOR HIGHLY AUTOMATED VEHICLES | CPS&IOT Proceedings Buff@BS&IoT2019 page 402

M.PAULWEBER, A.LEITNER | 37



Definitions



Ego Vehicle: Unit under test with ADAS and/or AD capabilities

Scenario: Formal description of real world traffic situation together with drive order for vehicle (ego vehicle)

Scenario may contain scenario parameters (having default values and variation range)

Scenario consists of static part (road lanes, traffic signs and lights, barriers, ...) and dynamic part (moving objects around ego vehicle)

- **Test Case:** Scenario with defined values for all scenario parameters
- **Test:** Testcase with KPI and required simulation models
- **Experiment:** Sequence of tests (Test Sequence) with potential parallel eventhandlers

All necessary simulation models and sensor stimuli

Exact definition of UUT

Exact description of test environment (MIL/SIL, Cloud, HIL, VIL, Proving ground, Road, test tools)

Tests Consists of Three Parts



1. Simulation models

- Environment model (OpenDRIVE*, OpenCRG**, OpenFLIGHT***
- Vehicle model
- Sensor models

2. Test scenario definition (OpenSCENARIO**)

- Scenario description (object types, positions, movements)
- Object parameters
- Scenario parameters (e.g. velocity of closest object, weather conditions, ...)

3. Key Performance Indicators (KPI): evaluation script

- Performance KPIs
- Objective Safety KPIs
- Perceived Safety KPIs
- Comfort KPIs (objective)

- * ASAM Standard
- ** potential ASAM Standard
- *** 3D geometry model format, administered by PRESAGIS



Scenario Generation for SW development teams:



Coarse scenarios



Scenario Generation for Vehicle Validation:









ADAS/AD SCENARIO EXTRACTION

VALIDATION FRAMEWORK FOR HIGHLY AUTOMATED VEHICLES | CPS&IOT Proceedings Buff@BS&IoT2019 page 407

M.PAULWEBER, A.LEITNER | 42



































TEST SEQUENCE GENERATION

VALIDATION FRAMEWORK FOR HIGHLY AUTOMATED VEHICLES | CPS&IOT Proceedings Buff@BS&IoT2019 page 413

M.PAULWEBER, A.LEITNER | 48

Sources for Validation Scenarios





High number of scenarios with parameter variations MIL/SIL

Requirements for Test Scenarios



Phase	Test Scope	Scenario Details	Input
Sensor development	Neural network trainingTest of sensors	HD videos with ground truth labeling	HD reference real world dataObject labeling
Sensor model parametrization	 Adapt model parameters to match sensor behavior Compare sensor with sensor stimulation 	HD videos with ground truth labeling	 HD reference real world data Object labeling Sensor testbed Sensor stimulus
Function verification	 Test of all indented scenarios (Function Test coverage) Real world scenario coverage Identify corner cases 	 Low detail (lanes, traffic signs, vehicles, pedestrians, cyclists) 	 Synthetic tests Coverage of all real world situations
System (Vehicle) validation	 Test (quasi) legislative regulations Test safety critical situations Test of corner case situations Analyze/fix detected problems from road testing 	 HD 3D virtual world Low detail (lanes, traffic signs, vehicles, pedestrians, cyclists) Dynamic objects 	 3D virtual world from Static ground truth Relevant Real world scenarios from dynamic ground truth (Quasi) Standards, e.g. Euro- NCAP

ADAS/AD Function and System Validation Workflow

Scenarios

All requirements fulfilled?

Tests to stimulate behavior defined in reg.

AVL 00

KPIs

KPIs defined in requirements

Selection of all scenarios in real world Check behavior in traffic Comfort, safety, perceived safety operation in designated area of UUT situations Detect pareto lines of edge Critical scenarios with variation of scenario Area of safe operation inside cases (corner cases) required operational area parameters Critical cases of functional safety **Check functional safety** Safetv analysis (ISO 26262) **Check behavior in potential** Synthetic critical scenarios Comfort, safety, perceived safety new critical traffic situations Check fail operational Test cases with sensor failures, critical Functionality under not normal behavior critical conditions situations, other failure conditions



List of

scenarios with

variations

- Environment conditions (daylight, fog..)

Experiment generator

Scenarios relevant for ADAS

Scenario parameter ranges

fog, rain, traffic type, ..)

Environment conditions (daylight,

function

Road





TEST ENVIRONMENTS

VALIDATION FRAMEWORK FOR HIGHLY AUTOMATED VEHICLES | CPS&IOT Page 418

M.PAULWEBER, A.LEITNER | 53





Sensor Testing:

- Sensor performance testbeds
- Sensor model parametrization

Software testing:

- Model in the loop (MiL) in the office,
- Model in the loop using high performance parallel cloud computing

Software and Hardware testing: Control unit in the loop (HiL)

Usability testing: Driving simulator with Driver in the loop (DiL)

ADAS/AD system qualification: Vehicle in the loop (ViL)

Safety tests: Tools for Tests on private proving ground (PG)

Performance tests: Tools for Tests on public roads (PRT)





VALIDATION IN THE CLOUD, MIL, SIL

VALIDATION FRAMEWORK FOR HIGHLY AUTOMATED VEHICLES | CPS&IOT Proceedings of CPS&IOT Proceed

M.PAULWEBER, A.LEITNER | 55



MIL/SIL Office / Cloud Environment





	Virtual or real
Vehicle	Virtual vehicle on virtual road
Sensors	Sensor models
Traffic objects	Virtual objects around ego-vehicle
Weather conditions	Simulated weather



Create virtual test drives

MIL Environment Simulation

- Interactive Road Network Editor allows to design road and rail networks in full detail with unlimited numbers of lanes, complex intersections, comprehensive signs and signaling.
- It links and exports logic and graphic data consistently from a single source.
- Virtual worlds can be designed from scratch or compiled from existing database tiles.
- Dynamic content is defined with an interactive scenario editor. It visualizes the underlying static virtual drive database and allows the user to specify traffic as individual objects and as autonomous swarms around key entities.
- Both, left- and right-hand driving environments need to be supported.
- Library of vehicles, pedestrians and driver properties are necessary.
- Optional real-time monitoring and command injection during the simulation phase.
- User may take full control over the execution of the simulation, specify varying time steps and consume object, image and sensor data via a whole range of interfaces
- Environment simulation tool may be operated from a single computer up to a full-scale HPC parallel cloud cluster.









to

Configuration of virtual worlds

Simulation of virtual worlds

Source: Vires VTD; https://vires.com/vtd-viresvirtual-test-drive/#configuration



Cloud Simulation to Identify Critical Scenarios





- Definition of System under test
- Controllers, Sensors, Actuator models, Environment
- Creation of input parameters
- Definition of steps of parameters or/and scenarios

- Creation of full scale full factorial amount of scenarios
- Optional send them to a DoE Tool like AVL CAMEO
- Create input file for the cloud
- Start up multi instance infrastructure
- Implicit rating or pass/fail evaluation
- Automated reporting and deep dive analysis

Cloud Simulation to Identify Critical Scenarios





Define test sequence with parameter variations (experiment)





Scalable simulation environment

Identify critical regions

- Simulation models never completely match behavior of real components

 e.g. behavior of sensor models in severe weather conditions
- Results of high performance parallel simulation only indication of ADAS/AD performance in critical situations
- Additional tests with real components required (Verification of model performance)

Cloud Simulation to Identify Critical Scenarios





Validation of system performance in critical scenarios using more detailed models or real components





VEHICLE IN THE LOOP (VIL) TESTING

VALIDATION FRAMEWORK FOR HIGHLY AUTOMATED VEHICLES | CPS&IOT Page 426

M.PAULWEBER, A.LEITNER | 61



VIL test in different Weather Conditions: Virginia Tech Smart Road



	Virtual or real
Vehicle	Real vehicle on model road
Sensors	Real sensors on vehicle
Traffic objects	UFOs or real objects
Weather conditions	Limited weather stimulation (fog, rain)







Source: Virginia Tech University; https://www.vtti.vt.edu/facilities/virginia-smart-road.html



VIL test in different Weather Conditions: TNO VeHiL



	Virtual or real
Vehicle	Real vehicle on chassis dyno TB
Sensors	Real sensors on vehicle
Traffic objects	UFOs or real objects
Weather conditions	Limited weather stimulation (fog, rain)



Source: TNO; https://www.youtube.com/watch?v=BW3_tsf48WY


Dangerous Scenarios in Save Environment: AVL Driving Cube











Camera and Radar Sensor Stimuli







Camera and Radar Sensor Stimuli





New config file is loaded!



Steering Force Stimulus







Steering Force Stimulus







Sensor Stimulation and Simulation









Summer School on Cyber-Physical Systems and Internet-of-Things

Budva, Montenegro, June 11, 2019



Design space exploration for Hypervisor-based mixed-criticality systems

Authors: Vittoriano Muttillo, Luigi Pomante

vittoriano.muttillo@univaq.it, luigi.pomante@univaq.it



University of L' Aquila Center of Excellence **DEVVS** Department of Information Engineering, Computer Science and Mathematics (**DISIM)**

Proceedings of CPS&IoT2019 page 435





- 1. Introduction
- 2. Safety Assurance Standards
- 3. Mixed Criticality Systems Analysis
- 4. Mixed-Criticality Classification
- 5. Mixed-Criticality HW/SW Co-Design
- 6. Proposed Methodology
- 7. ESL Methodology Main Elements
- 8. HepsyCode-MC
- 9. Case Studies
- 10. Hepsycode Ecosystem
- 11. Publications and European Projects
- 12 Conclusion and Future Works







Introduction

"Brief Introduction to Mixed Criticality and Cyber Physical Systems"

MECO Context: Cyber-Physical System



- A cyber-physical system (CPS) is an integration of computation with physical processes whose behavior is defined by both cyber and physical parts of the system.
- Embedded computers and networks monitor and control the physical processes, usually with feedback loops where physical processes affect computations and vice versa.
- As an intellectual challenge, CPS is about the intersection, not the union, of the physical and the cyber.

[1] Lee, E. A., Seshia, S. A.,: Introduction to Embedded Systems, a Cyber-Physical Systems approach, Second Edition, LeeSeshia.org, 2015



MECO Mixed-Criticality Embedded Systems

- The growing complexity of embedded digital systems based on modern System-on-Chip (SoC) adopting explicit heterogeneous parallel architectures has radically changed the common design methodologies.
- HW/SW co-design methodologies are of renovated relevance
- A growing trend in embedded systems domain is the development of mixed-criticality systems where multiple embedded applications with different levels of criticality are executed on a shared hardware platform (i.e. Mixed-Criticality Embedded Systems)





- A mixed criticality system is "an integrated suite of HW, OS, middleware services and application software that supports the concurrent execution of safety-critical, mission-critical, and noncritical software within a single, secure computing platform"
- MAIN GOALS: development of EDA tools, mainly oriented to support the designer of Mixed-Criticality and Cyber-Physical systems based on heterogeneous multi/many-core platforms, considering Hypervisor-Based SW partitions

introduces the possibility to add real-time and mixed-criticality requirements to the set of non-functional ones Focus on **Design Space Exploration** considering

Exploration considering HPV-based SW Partitions

Official Web Site: www.hepsycode.com



In the context of real-time

design, this work starts

HEPSYCODE: HW/SW CO-

DEsign of **HE**terogeneous

SYstems), based on an

existing System-Level

а

systems

specific

Dedicated

Co-Design

and

(called

embedded

methodology

from

Parallel

HW/SW

methodology,



epsy

ode







2.

Safety Assurance Standards

"Criticality is a designation of the level of assurance against failure needed for a system component"

MECO Safety Related Standards



- Industry has shown a growing interest in integrating and running independentlydeveloped applications of different "criticalities" in the same (often multicore) platform. Such integrated systems are commonly referred to as mixed-criticality systems (MCS).
- Most of the MCS-related research cite the safety-related standards associated to each application domain (e.g. aeronautics, space, railway, automotive) to justify their methods and results. However, those standards are not freely available and do not always clearly and explicitly specify the requirements for mixed-criticality
- New MC task model is in essence the result of combining the standard hard real-time requirements (studied by the real-time research community since the 70's) with the notion of "criticality" of execution.



System Design and Development Assurance Process



- During a typical development life cycle of a safety-critical system, the behavior and characteristics that are expected from the system are expressed in the form of a list of requirements
 - based on the system operational requirements (what the system is expected to do) and also considering non-functional properties related to safety, security and performance, including timing and energy constraints.
- System safety assessment process must be carried out as part of the development life cycle to determine and categorize the failure conditions of the system (e.g. through a hazard analysis).
 - safety-related requirements are derived as a result of the system safety assessment process, which may include functional, integrity, dependability requirements and design constraints.
- > Safety-related requirements are allocated to hardware and software components, thereby specifying the mechanisms required to prevent the faults or to mitigate their effects and avoid the propagation of failures.





- Most safety standards use the concept of an integrity level, which is assigned to a system or a function. This level will be based on an initial analysis of the consequences of software going wrong. Both standards have clear guidance on how to identify integrity level.
 - DO-178C has Software Development Assurance Level (DAL), which are assigned based on the outcome of "anomalous behavior" of a software component Level A for "Catastrophic Outcome", Level E for "No Safety Effect".
 - ISO26262 has ASIL (Automotive Safety Integrity Level), based on the exposure to issues affecting the controllability of the vehicle. ASILs range from D for the highest severity/most probable exposure, and A as the least.







- GENERAL (IEC-61508) based on SIL (Safety Integrity Level): Functional safety standards (of electrical, electronic, and programmable electronic)
 - AUTOMOTIVE (ISO26262) based on ASIL (Automotive Safety Integrity Level) (Road vehicles Functional safety)
 - NUCLEAR POWER (IEC 60880-2)
 - MEDICAL ELECTRIC (IEC 60601-1)
 - PROCESS INDUSTRIES (IEC 61511)
 - RAILWAY (CENELEC EN 50126/128/129])
 - MACHINERY (IEC 62061)

> AVIONIC based on DAL (Development Assurance Level) related to ARP4761 and ARP4754

- DO-178B (Software Considerations in Airborne Systems and Equipment Certification)
- DO-178C (Software Considerations in Airborne Systems and Equipment Certification, replace DO-178B)
- DO-254 (Airborne Design), similar to DO-178B, but for hardware
- DO-160F (Airborne Test)

> MEDICAL DEVICE

- FDA-21CFR
- IEC-62304



3. Mixed Criticality Systems Analysis

"The more confidence one needs in a task execution time bound (the less tolerant one is of missed deadlines), the larger and more conservative that bound tends to become in practice"

MEC^O MCS state-of-the-art Model (1)



- Almost 200 papers treating of the scheduling of MCS have been referenced in Burns and Davis* paper, and tens of related papers are still published every year. Most of the works about MCS published by the real-time scheduling research community are based on a model proposed by Vestal*
 - System has several modes of execution, say modes $\{1, 2, ..., L\}$. The application system is a set of real-time tasks, where each task τ_i is characterized by a period T_i and a deadline D_i (as in the usual real-time task model), an assurance level I_i and a set of worst-case computational estimates $\{C_{i,1}, C_{i,2}, ..., C_{i,l_i}\}$, under the assumption that $C_{i,1} \leq C_{i,2} \leq ... \leq C_{i,l_i}$
- > The different WCET estimates are meant to model estimations of the WCET at different assurance levels. The worst time observed during tests of normal operational scenarios might be used as $C_{i,1}$ whereas at each higher assurance level the subsequent estimates $C_{i,2}, \ldots, C_{i,l_i}$ are assumed to be obtained by more conservative WCET analysis techniques.

* Burns, A, Davis, RI.: "Mixed Oriticality Systems - A Review", University of York, 4 March 2016. ** S. Vestal, "Preemptive Scheduling of Multi-criticality Systems with Varying Degrees of Execution Time Assurance," Real-Time Systems Symposium (RTSS) 28th IEEE International on, Tucson, AZ, 2007, pp. 239-243.

MEC^O MCS state-of-the-art Model (2)



- The system starts its execution in mode 1 and all tasks are scheduled to execute on the core[s]. Then at runtime, if the system is running in mode k then each time the execution budget $C_{i,k}$ of a task τ_i is overshot, the system switches to mode k+1 It results from this transition from mode k to mode k+1 that all the tasks of criticality not greater than k (i.e., $l_i \ge k$) are suspended. Mechanisms have also been proposed to eventually re-activate the dropped tasks at some later points in time*.
 - one of the simplifications of this model is the Vestal's model with only two modes, usually referred to as LO and HI modes (which stand for Low- and High-criticality modes).
- Multiple variations of that scheduling scheme exist, some for single-core, others for multicore architectures. In the case of multicore, both global and partitioned scheduling techniques have been studied and solutions for fixed priority scheduling (RM), Earliest Deadline First (EDF) and time triggered scheduling have been proposed in literature.
 - some works also propose to change the priorities or the periods of the tasks during a mode change rather than simply stopping the less critical ones.
 - Note that some works also propose to change the priorities or the periods of the tasks during a mode change rather than simply stopping the less critical ones.

* F. Santy, G. Raravi, G. Nelissen, V. Nelis, P. Kumar, J. Goossens, and E. Tovar. Two protocols to reduce the criticality level of multiprocessor mixed-criticality systems. In RTNS 2013, RTNS' 13, pages 183–192. ACM, 2013.

MCS Design - OFFIS FCO education & training



access on res. X

(decoupled)

Mixed-criticality principles applied to application layer modelling objects:





- Performance-critical domain T_3, T_4, T_5, S_3 : low-critical tasks and shared objects
- mixed-critical shared objects: S1, S2



Connections to shared object interfaces

- Deadline
- might affect scheduling decisions

Mixed-Criticality aware properties:

- Task periods
- Safety-critical tasks might have increased activation frequencies in high criticality levels
- Vector of computation times
- ▶ platform-dependent, → later ► One for each criticality level (e.g. LO, HI)
- Criticality level
- statically defined at design time

Task $T_i \in \tau$. $(\vec{T}_i, D_i, \vec{C}_i, \pi_i, L_i)$

- a vector of periods T (minimum arrival interval)
- D: deadline
- C: vector of computation times (one for each criticality level)
- m: ports for connecting to communication objects
- Li: criticality level (e.g. LO, HI)

Possible Shared Object realisation should include:

- Cached internal state Σ. Σ'
- ▶ before performing the call, create copy: $\Sigma' \leftarrow \Sigma$

Application Layer

2 (Shared) Object

Virtual Target Architecture Layer

/ Task

- active calls work on state copy Σ'
- after the call, **update** original state: $\Sigma \leftarrow \Sigma'$
- Preemptible scheduling of access. policy defined by attribute Φ (e.g. round-robin, fixed-prio)
- ▶ if criticality level L increases, abort active calls < L</p>
- ► discard Σ'
- only allow calls from tasks with criticality level > L





Core 0

Runtim

- Σ_{0,1}: inner states (containing) abstract data types),
- L: current criticality level (e.g. LO, HI)
- $M \subseteq \Sigma \times \Sigma$: a set of methods or services (e.g. read(), write())
- ▶ $I \subseteq \mathcal{P}(M)$: Interfaces for grouping methods
- policy



🛉 arrival	running	X	access res. X (sync required)
prosmotion	blocked	×	access on res.)

····· preemption __





Proceedings of CPS&IoT2019 page 4 CPCI-LEON4-N2X

REFERENCE

SOFTWARF DEMO PLATFORM

SPACEWIRE

PERIPHERAL

DEVICE 1

landing!

Flight

Data

Mining

Predictability





Research – Industrial Domain Misunderstandings

- Academic papers: "system criticality" as a mode of execution of software tasks (e.g. high or low criticality). Mode change allowed
- Industrial domain: "system criticality" refers to the level of assurance (e.g. DAL, SIL or ASIL) applied in the development of a software application that implements critical system functionalities (i.e. safety functions)

Mixed-Criticality Challenges

 Scheduling (Priority Vs Safety), Partition (Isolation), Performance (WCET estimation), Predictability (Graceful Degradation), Manufactory Cost, Fault-tolerance, Power-consumption, Networking





4. Mixed-Criticality Classification

"Amajor industrial challenge arises from the need to face cost efficient integration of different applications with different levels of safety and security on a single computing platform in an open context"





Separation technique:

- **Timing separation**: scheduling policy, temporal partitioning with HVP, NoC
- Spatial separation: one task per core, one task on HW ad hoc (DSP, FPGA), spatial partition with HVP, NoC, MMU, MPU etc.

≻HW:

- Temporal isolation: Scheduling HW
- Spatial isolation: separated Task on dedicated components (HW ad hoc, FPGA etc.)

Single core:

- Temporal isolation: Scheduling policy with SO o RTOS, Scheduling policy with HVP
- Spatial isolation : MMU, MPU, HVP Partitioning

≻Multi-core

- Architecture: shared memory systems, Uniform Memory Architecture, UMA (SMP), Not Uniform Memory Architecture, NUMA, distributed systems, NoC
- Temporal isolation: Scheduling policy with SO o RTOS, Scheduling policy with HVP
- Spatial isolation: MMU, MPU, HVP partitioning

≻Many-core

Work in progress



Tecnologies:

- Hardware: HW ad hoc, FPGA, DSP, Processor
 > Processor: LEON3, ARM, MICROBLAZE etc.
- Software: Bare-metal, OS, RTOS, HVP
 OS: Linux etc.
 - > OS: Linux etc.
 - **RTOS**: eCos, RTEMS, FreeRTOS, Threadx, VxWorks, EriKa etc.
 - > HVP: PikeOS, Xtratum, Xen etc.





Proceedings of CPS&IoT2019 page 455



MECO MC Implementation Solutions (M-CIS)



Tecnologies:

- Hardware: HW ad hoc, FPGA, DSP, Processor.
 - Processor: LEON3, ARM, MICROBLAZE, etc.
- Software: Bare-metal, OS, RTOS, HVP
 - OS: Linux etc.
 - RTOS: eCos, RTEMS, FreeRTOS, Threadx, VxWorks, EriKa etc.
 - HVP: PikeOS, Xtratum, Xen etc
- Many-core: WIP

Separation Technique	нพ	Single core	Multi-core	Many-core	
0-l Spatial sche [125],[0-levels scheduling [7],[129],[130],[131]	0-levels scheduling [132],[124],[125],[131]	[124],[125], [126],[127],[128]	
	0-levels scheduling [125],[129],[130]	1-level scheduling [97],[133],[134],[135] [141],[142],[143]	1-level scheduling [7],[136],[137],[125],[138],[139],[140] ,[133],[144],[141],[145],[146],[61],[14 2],[147],[131]		
	-	2-levels scheduling [141],[148]	2-levels scheduling [149],[125],[126],[150],[151],[128]		
		3-levels scheduling [154]	3-levels scheduling [154]		
Temporal	0-levels scheduling [125],[129],[130]	0-levels scheduling [7],[129],[130],[131]	0-levels scheduling [132],[124],[125],[129],[130],[128]		
		1-level scheduling [133],[134],[135], [142],[143],[131]	1-level scheduling [7],[136],[137],[155],[125],[138],[140] [133],[144],[146],[61],[156],[142]	[124],[125],	
		2-levels scheduling [141],[148]	2-levels scheduling [149],[125],[126],[152],[141],[145], [153],[148],[49],[128]	[126],[127],[128]	
		3-levels scheduling [154]	3-levels scheduling [154]		







[7] Henning Schlender, S"oren Schreiner, Malte Metzdorf, Kim Gr"uttner, and Wolfgang Nebel. Teaching mixed-criticality: Multi-rotor flight control and payload processing on a single chip. In Proceedings of the WESE'15: Workshop on Embedded and Cyber-Physical Systems Education, WESE'15, pages 9:1–9:8, New York, NY, USA, 2015. ACM.

[49] Gernot Heiser and Ben Leslie. The OKL4 microvisor: Convergence point of microkernels and hypervisors. In Asia-Pacific Workshop on Systems (APSys), pages 19–24, New Delhi, India, August 2010

[61] Vittoriano Muttillo, Giacomo Valente, and Luigi Pomante. Criticality-driven design space exploration for mixed-criticality heterogeneous parallel embedded systems. In Proceedings of the 9th Workshop and 7th Workshop on Parallel Programming and RunTime Management Techniques for Manycore Architectures and Design Tools and Architectures for Multicore Embedded Computing Platforms, PARMA-DITAM '18, pages 63–68, New York, NY, USA, 2018. ACM.

[97] S. Vestal. Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. In 28th IEEE International Real-Time Systems Symposium (RTSS 2007), pages 239–243, Dec 2007.

[124] H. Isakovic and R. Grosu. A heterogeneous time-triggered architecture on a hybrid system-on-a-chip platform. In 2016 IEEE 25th International Symposium on Industrial Electronics (ISIE), pages 244–253, June 2016.

[125] Haris Isakovic, Radu Grosu, Denise Ratasich, Jiri Kadlec, Zdenek Pohl, Steve Kerrison, Kyriakos Georgiou, Kerstin Eder, Norbert Druml, Lillian Tadros, Flemming Christensen, Emilie Wheatley, Bastian Farkas, Rolf Meyer, and Mladen Berekovic. Asurvey of hardware technologies for mixed-critical integration explored in the project em c2. In Stefano Tonetta, Erwin Schoitsch, and Friedemann Bitsch, editors, Computer Safety, Reliability, and Security, pages 127–140, Cham, 2017. Springer International Publishing.

[126] S. Saidi, R. Ernst, S. Uhrig, H. Theiling, and B. D. de Dinechin. The shift to multicores in real-time and safety-critical systems. In 2015 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS), pages 220–229, Oct 2015.

[127] F. Federici, M. Micozzi, V. Muttillo, L. Pomante, and G. Valente. Simulationbased analysis of a hardware mechanism to support isolation in mixed-criticality network on chip. In 2017 European Modelling Symposium (EMS), pages 185–190, Nov 2017.

[128] Martin Schoeberl, Sahar Abbaspour, Benny Akesson, Neil Audsley, Ra¢aele Capasso, Jamie Garside, Kees Goossens, Sven Goossens, Scott Hansen, Reinhold Heckmann, Stefan Hepp, Benedikt Huber, Alexander Jordan, Evangelia Kasapaki, Jens Knoop, Yonghui Li, Daniel Prokesch, Wolfgang Pu'tsch, Peter Puschner, Andr Rocha, Cludio Silva, Jens Spars, and Alessandro Tocchi. Tcrest: Time-predictable multi-core architecture for embedded systems. Journal of Systems Architecture, 61(9):449 – 471, 2015.

[129] Rodolfo Pellizzoni, Patrick Meredith, Min-Young Nam, Mu Sun, Marco Caccamo, and Lui Sha. Handling mixed-criticality in soc-based real-time embedded systems. In Proceedings of the Seventh ACM International Conference on Embedded Software, EMSOFT '09, pages 235–244, New York, NY, USA, 2009. ACM

[130] M. Zimmer, D. Broman, C. Shaver, and E. A Lee. Flexpret: Aprocessor platform for mixed-criticality systems. In 2014 IEEE 19th Real-Time and Embedded Technology and Applications Symposium (RTAS), pages 101–110, April 2014.







[131] W. Weber, A. Hoess, J. v. Deventer, F. Oppenheimer, R. Ernst, A. Kostrzewa, P. Dor, T. Goubier, H. Isakovic, N. Druml, E. Wuchner, D. Schneider, E. Schoitsch, E. Armengaud, T. Sderqvist, M. Traversone, S. Uhrig, J. C. Prez-Corts, S. Saez, J. Kuusela, M. v. Helvoort, X. Cai, B. Nordmoen, G. Y. Paulsen, H. P. Dahle, M. Geissel, J. Salecker, and P. Tummeltshammer. The emc2 project on embedded microcontrollers: Technical progress after two years. In 2016 Euromicro Conference on Digital System Design (DSD), pages 524–531, Aug 2016.

[132] I. Sourdis, D. A. Khan, A. Malek, S. Tzilis, G. Smaragdos, and C. Strydis. Resilient chip multiprocessors with mixed-grained reconfigurability. IEEE Micro, 36(1):35–45, Jan 2016.

[133] M. Paulitsch, O. M. Duarte, H. Karray, K. Mueller, D. Muench, and J. Nowotsch. Mixed-criticality embedded systems – a balance ensuring partitioning and performance. In 2015 Euromicro Conference on Digital System Design, pages 453–461, Aug 2015.

[134] Andreas Gerstinger, Heinz Kantz, and Christoph Scherrer. Tas control platform : Aplatform for safety-critical railway applications. ERCIM News, 2008, 2008.

[135] M. G. Hill and T. W. Lake. Non-interference analysis for mixed criticality code in avionics systems. In Proceedings ASE 2000. Fifteenth IEEE International Conference on Automated Software Engineering, pages 257–260, Sept 2000.

[136] J. Steinbaeck, A. Tengg, G. Holweg, and N. Druml. A3d time-of-flight mixed criticality system for environment perception. In 2017 Euromicro Conference on Digital System Design (DSD), pages 368–374, Aug 2017.

[137] G. Macher, A. Hller, E. Armengaud, and C. Kreiner. Automotive embedded software: Migration challenges to multi-core computing platforms. In 2015 IEEE 13th International Conference on Industrial Informatics (INDIN), pages 1386–1393, July 2015.

[138] T. Bijlsma, M. Kwakkernaat, and M. Mnatsakanyan. Areal-time multi-sensor fusion platform for automated driving application development. In 2015 IEEE 13th International Conference on Industrial Informatics (INDIN), pages 1372–1377, July 2015.

[139] D. Muench, O. Isfort, K. Mueller, M. Paulitsch, and A. Herkersdorf. Hardwarebased i/o virtualization for mixed criticality real-time systems using pcie sr-iov.

In 2013 IEEE 16th International Conference on Computational Science and Engineering, pages 706–713, Dec 2013.

[140] Z. Mynar, L Vesely, and P. Vaclavek. Pmsm model predictive control with field-weakening implementation. IEEE Transactions on Industrial Electronics, 63(8):5156–5166, Aug 2016

[141] J. Kim, M. Yoon, S. Im, R. Bradford, and L. Sha. Optimized scheduling of multi-ima partitions with exclusive region for synchronized real-time multi-core systems. In 2013 Design, Automation Test in Europe Conference Exhibition (DATE), pages 970–975, March 2013.

[142] M. Pagani, A. Balsini, A. Biondi, M. Marinoni, and G. Buttazzo. Alinux-based support for developing real-time applications on heterogeneous platforms with dynamic fpga reconfiguration. In 2017 30th IEEE International System-on-Chip Conference (SOCC), pages 96–101, Sept 2017.







[143] Kees Goossens, Arnaldo Azevedo, Karthik Chandrasekar, Manil Dev Gomony, Sven Goossens, Martijn Koedam, Yonghui Li, Davit Mirzoyan, Anca Molnos, Ashkan Beyranvand Nejad, Andrew Nelson, and Shubhendu Sinha. Virtual execution platforms for mixed-time-criticality systems: The compsoc architecture and design flow. SIGBED Rev., 10(3):23–34, October 2013.

[144] Ankit Agrawal, Gerhard Fohler, Johannes Freitag, Jan Nowotsch, Sascha Uhrig, and Michael Paulitsch. Contention-aware dynamic memory bandwidth isolation with predictability in cots multicores: An avionics case study. In 29th Euromicro Conference on Real-Time Systems (ECRTS 2017), volume 76 of Leibniz International Proceedings in Informatics (LIPIcs), pages 2:1–2:22, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[145] J. Kim, M. Yoon, R. Bradford, and L Sha. Integrated modular avionics (ima) partition scheduling with conflict-free i/o for multicore avionics systems. In 2014 IEEE 38th Annual Computer Software and Applications Conference, pages 321–331, July 2014.

[146] B. Huber, C. El Salloum, and R. Obermaisser. A resource management framework for mixed-criticality embedded systems. In 2008 34th Annual Conference of IEEE Industrial Electronics, pages 2425–2431, Nov 2008.

[147] M. S. Mollison, J. P. Erickson, J. H. Anderson, S. K. Baruah, and J. A Scoredos. Mixed-criticality real-time scheduling for multicore systems. In 2010 10th IEEE International Conference on Computer and Information Technology, pages 1864–1871, June 2010.

[148] Y. D. Bock, J. Broeckhove, and P. Hellinckx. Hierarchical real-time multicore scheduling through virtualization: A survey. In 2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), pages 611–616, Nov 2015.

[149] C. E. Salloum, M. Elshuber, O. Hftberger, H. Isakovic, and A. Wasicek. The across mpsoc – a new generation of multi-core processors designed for safetycritical embedded systems. In 2012 15th Euromicro Conference on Digital System Design, pages 105–113, Sept 2012.

[150] S. Avramenko, S. Esposito, M. Violante, M. Sozzi, M. Traversone, M. Binello, and M. Terrone. An hybrid architecture for consolidating mixed criticality applications on multicore systems. In 2015 IEEE 2 lst International On-Line Testing Symposium (IOLTS), pages 26–29, July 2015.

[151] S. Esposito, S. Avramenko, and M. Violante. On the consolidation of mixed criticalities applications on multicore architectures. In 2016 17th Latin-American Test Symposium (LATS), pages 57–62, April 2016.

[152] F. Federici, V. Muttillo, L Pomante, G. Valente, D. Andreetti, and D. Pascucci. Implementing mixed-critical applications on next generation multicore aerospace platforms. In EMC2 Summit at CPS Week, 2016.

[153] P. Modica, A. Biondi, G. Buttazzo, and A. Patel. Supporting temporal and spatial isolation in a hypervisor for arm multicore platforms. In 2018 IEEE International Conference on Industrial Technology (ICII), pages 1651–1657, Feb 2018.

[154] G. Cicero, A. Biondi, G. Buttazzo, and A. Patel. Reconciling security with virtualization: Adual-hypervisor design for arm trustzone. In 2018 IEEE International Conference on Industrial Technology (ICIT), pages 1628–1633, Feb 2018.

Proceedings of CPS&IoT2019 page 461

28%



ISOLATION TECHNIQUES





MECO MC Implementation Solutions (M-CIS)



SCHEDULING LEVELS









5. Mixed-Criticality HW/ SW Co-Design

"Multi-core and manycore computing platforms have to significantly improve system (and application) integration, efficiency and performance"

MEC^O HW/ SW Co-Design Methodologies



Proceedings of CPS&IoT2019 page 463





Electronic System Level (ESL) is the utilization of appropriate abstractions in order to increase comprehension about a system, and to enhance the probability of a successful implementation of functionality in a cost-effective manner, targeting design methodologies for electronic digital HW/SW systems


MEC^{CO} Mixed-Criticality HW/ SW Co-Design



Classification of different ESL Methodology Approaches (considering Mixed-Criticality Issues)

		Specification		Specification	Implementation			Decision Making		Refinement	
ESL Approach	$Gen.^1$	Appl. ²	Arch. ³	Language	MoS^4	MoP^5	DSE ⁶ $ $	Comp. ⁷	Comm. ⁸	Comp. ⁷	Comm. ⁸
AUTOFOCUS3	4.0	PN	HeMPES	Custom	Component	TAPM	•	•	0	•	-
CONTREP	4.0	UML, SDF	HeMPES	MARTE & SysML & SystemC	TLM	TAPM	•	•	0	•	0
DeSyDe	3.0/4.0	SDF	HeMPSoC	XML	-	-	•	•	0	•	0
Combined-DSE	4.0	CP	HeMPES	MiniZinc	TLM	TAPM	•	•	-	•	-
OSSS-MC	4.0	OSSS/MC	HeMPSoC	SystemC	TLM	T/ISAPM	-	•	-	•	-
MultiPARTES	4.0	UML	HoMPES	MARTE	TLM	TAPM	•	•	-	•	-
HEPSYCODE	4.0	CSP	HeMPES	SystemC	TLM	T/ISAPM	•	•	•	•	0

¹ Gen.: HW/SW Co-Design Generation; ² Appl.: Application Model; ³ Arch.: Architecture Model;

⁴ MoS: Model of Structure; ⁵ MoP: Model of Performance; ⁶ DSE: Design Space Exploration; ⁷ Comp.: Computation; ⁸ Comm.: Communication; CP: Constraint Programming;

PN: Process Network; CSP: communicating sequential processes; UML: Unified Modeling Language; SDF: Synchronous Dataflow;

He/HoMPS: Heterogeneous/Homogeneous Multi-processor Systems;

He/HoMPSoC: Heterogeneous/Homogeneous Multi-processor System on chip;

He/HoMPES: Heterogeneous/Homogeneous Multi-processor Embedded System;

TLM: Transaction-Level Model:

TAPM: Task Accurate Performance Model: ISAPM: Instruction Set Accurate Performance Model: CAPM: Cycle-Accurate Performance Model:





6.

Proposed Methodology

"You will never strike oil by drilling through the map! -Solomon Wolf Golomb"

MECO Proposed HW/ SW Co-Design Methodology

DEFINITION OF A GENERAL METHODOLOGY ABLE TO ABSTRACT CLASSICAL SYSTEM DESIGN FLOW (APPLICABLE TO DIFFERENT HW/SW CO-DESIGN FLOW)

System Description: Introduction of a partition layer to model HPV SW partitions

Metrics Evaluation and Estimation: Definition of different metrics (with related benchmarking arctivites in order to extract as-much-as-possible system informations)

Search Methods: Meta-heuristic algorithm refinement (GA improvements)

Timing Simulator: improvement introducing Hierarchical scheduling feature





Procee

EC **HEPSYCODE HW/ SW Co-Design Flow**



MEC⁽²⁾ HW/ SW Co-Design Framework

Definition of a general framework able to automate system design flow (Implemented using different SW technologies)

System Behavioral model: Realization of a GUI to model application using the specification language defined in the system behavioral specification step

Functional Simulation: automatic generation of a SystemC code implementing Hoare's CSP model of Computation from GUI

Co-Analysis&Co-Estimation: definition of a extensible activity step to evaluate system metrics:

- Affinity
- Concurrency
- Communication
- Size
- Load
- Power (WIP)

DSE: implementation of an automatic (extensible) DSE to make analysis and propose solutions in an HW/SW Co-simulation environment (**HEPSIM**)







7. ESL Methodology Main Elements

"All models are wrong but some are useful-*George E P. Box*"







- Concurrency is the decomposability property of a program, algorithm, or problem into order-independent or partially-ordered components or units.
- Even if the concurrent units of the program, algorithm, or problem are executed out-of-order or in partial order, the final outcome will remain the same. This allows for parallel execution of the concurrent units, which can significantly improve overall speed of the execution in multi-processor and multi-core systems.
- A number of mathematical models have been developed for general concurrent computation (Petri nets, process calculi, the Parallel Random Access Machine model, the Actor model etc.).

MEC^O Model Of Computation



- A *Model of Computation (MoC)* is a set of operational elements used to describe the behavior of an application (or a system). The set of operational elements and the set of relations among them are called the semantics of a MoC.
- MoC can be classified into Timed or Untimed, when introducing a totally or partially ordered events respectively.
 - □ Untimed MoC:
 - Rendezvous of Sequential Processes: applications are modeled with sequential processes that reach a particular point at which they have to synchronize each other (i.e., CSP by Hoare, 1975).
 - Kahn Process Networks: a process network where processes communicates using channels, which are unbounded point-to-point FIFO queues, sending fixed amount of data, called tokens
 - Dataflow: a special case of Kahn process networks, where processes (called actors) consume data exchanged between channels with a fixed firing rate

MECO Process Calculi and CSP



- Process Calculi (or Process Algebras) are a diverse family of related approaches for formally modelling concurrent systems. Process calculi provide a tool for the high-level description of interactions, communications, and synchronizations between a collection of independent agents or processes. They also provide algebric laws that allow process descriptions to be manipulated and analyzed, and permit formal reasoning about equivalences between processes (e.g., using bisimulation).
- Process Calculi include the Communicating Sequential Processes (CSP), the Calculus of Communicating Systems (CCS), the Algebra of Communicating Processes (ACP) and so on.
- CSP is based on message passing via channels and was highly influential in the design of the OCCAM programming language.
- CSP was first described in a 1978 paper by Tony Hoare [8], but has since evolved substantially. CSP has been practically applied in industry as a tool for specifying and verifying the concurrent aspects of a variety of different systems as well as a secure ecommerce system. The theory of CSP itself is also still the subject of active research, including work to increase its range of practical applicability (e.g., increasing the scale of the systems that can be tractably analyzed).

MECO Modelling Language



- The system behavior modeling language introduced in HEPSYCODE named HML (HEPSY Modeling Language), is based on the well-known Communicating Sequential Processes (CSP) Model of Computation (MoC)
- > By means of HML it is possible to specify the System Behavior Model (SBM)

SBM = {PS, CH} is a CSP-based executable/simulatable model of the system behaviour based on a Concurrent Processes MoC that explicitly defines also a model of communication) among processes (PS) using unidirectional point-to-point blocking channels (CH) for data exchange (i.e. CSP channels).

 $PS = \{ps_1 ps_2, ..., ps_n\}$ is a set of concurrent processes that communicate each others exclusively by means of channels and use only local variables. Each process is described by means of a sequence of statements (an init section followed by a neverending loop) by using a suitable modeling language. Each process can have a priority p: 1(lower) to 100 (higher) imposed by the designer

 $CH = \{ch_1 ch_2, ..., ch_n\}$ is a set of channels where each channel is characterized by source and destination processes, and some details (i.e. size, type) about transferred data. Each channel can have also a priority p: 1(lower) to 100 (higher) imposed by the designer

MECO System Description Models

- > **HEPSYCODE** Modeling Language (HML):
 - Process Network connected via synchronous channels
- \triangleright G = {PS; CH} is the graph of the specification, where the graph nodes are the processes and the graph edges are the channel



The initial HML model is then transformed into an executable SystemC model based on the Communicating Sequential Processes (CSP) Model of Computation (MoC)

MECO HEPSYCODE Functional Language



- > **HEPSYCODE** Functional Language:
 - Reference languages is the SystemC, C++ class library able to capture and define system specifications
 - CSP processes are modelled by exploiting basic SC_THREAD (implemented as an infinite loop with an initialization step)
 - CSP channels have been modeled by introducing a proper SC_CSP_CHANNEL in the SystemC library.
 - System behavior is enclosed into a single SC MODULE, containing all the CSP processes and channels
 - Other SC MODULE and SC CSP CHANNEL are then used to model the Test-Bench and connected to the system by means of proper SC PORT



MECO System Behaviour



An example of a possible SBM in shown in Figure, where the process $PS = \{ps_1, ..., ps_4\}$ exchange data using channel $CH = \{ch_1, ..., ch_7\}$





> Non-Functional Constraints

- ✓ Timing Constraints (TC)
 - Time-To-Completion Constraint (TTC)
- ✓ Real-Time Constraints (RTC)
 - Time-To-Reaction Constraint (TTR)
- ✓ Mixed-Criticality Constraints (MCC)
 - Constraint in the DSE cost function
- ✓ Architectural Constraints
 - Target Form Factor (TFF)
 - On-chip: ASIC, FPGA, SO(P)C
 - On-Board: SOB (PCB)
 - Target Template Architecture (TTA) (related to type of available Basic Blocks BB)
- ✓ Scheduling Directives (SD) Available scheduling policies for SW processors:
 - First-Come First-Served (FCFS), FCFS(no overhead), FCFS (Time Stretching)
 - Fixed Priority (FP)
 - Hypervisor (HVP WIP)



MECO SBM with Real-Time Constraints



Communicating sequential processes (CSP) System Behavioral Model (CSP-SBM): network of concurrent processes model

Process Implementation Model (PIM): split processes into several "dependent" tasks.

Process Task Graph Model (PTM): Directed Acyclic Graph Task model (data flow)



MECO SBM with Real-Time Constraints



With respect to the SBM model, it is now possible to identify two class of CSP processes: classical CSP process and real-time CSP processes



MECO Timing Constraints



- Time-to-Completion (TTC): time available to complete the SBM execution from the first input trigger to complete output generation. This constrain should be satisfied by each (input_i, output_i) couple.
- Time-to-Reaction (TTR): real-time constraints related to the time available for the execution of leaf CSP processes (i.e. the time available to execute the statements inside the input/output pair that delimits the never-ending loop of a CSP process). This constrain should be satisfied by each input and output



Proceedings of CPS&IoT2019 page 482

MECO Target Architecture

- The target HW architectures are composed of different basic HW components. This components are collected into a Technologies Library (72). TL can be considered a generic "database" that provides the characterization of all the available technologies used in industry and academic world.
- TL ={PU, MU, EL}, where PU = {pu₁ pu₂, ..., pu_p} is a set of Processing Units, MU = {mu₁ mu₂, ..., mu_m} is a set of Memory Units and EL = {il₁ il₂, ..., il₀} is a set of External Interconnection Links.
- Blocks built by the designer starting from the TL are called Basic Blocks (BB)
- They are the basic components available during DSE step to automatically define the HW architecture. A generic BB is composed of a set of Processing Units (PU), a set of Memories Units (MU), an Internal Interconnection (II) and a Communication Unit (CU). and a Communication Unit (CU). CU represents the set of EL that can be managed by a BB.







Proposed HW/ SW Co-Design Methodology education & training



HEPSYCODE Metrics



CC4CS: early stage metric to estimate in a HW/SW unified view process execution time

Statistical Analisys: Evaluate metric accuracy

S4CS (HW/SW): size metric to evaluate software (in terms of bytes in RAM/ROM) and hardware (in terms of gate or LUT count) size (WIP)

Other WIP metrics: Affinity, Power/Energy, Monitorability, Security







MECO HEPSYCODE Design Space Exploration



► INPUT:

- **Application Model:** CSP model injected with safety requirements.
- Platform Model: subset of HW solution (also in a multi-core scenario)
- Metrics: results from the Evaluation&Estimation activity
- **Constraints:** F/NF constraints (depending on application domain)

▶OUTPUT:

- **Physical Links:** Possible optimal links and topology.
- Mapping: Process to BBs.
- Basic Blocks: Processors, architecture and number of cores.



MECO HEPSYCODE Design Space Exploration



► INPUT:

- **Application Model:** CSP model injected with safety requirements.
- Platform Model: subset of HW solution (also in a multi-core scenario)
- Metrics: results from the Evaluation&Estimation activity
- **Constraints:** F/NF constraints (depending on application domain)

▶OUTPUT:

- **Physical Links:** Possible optimal links and topology.
- Mapping: Process to BBs.
- Basic Blocks: Processors, architecture and number of cores.



MECO HEPSYCODE Design Space Exploration



► INPUT:

- **Application Model:** CSP model injected with safety requirements.
- Platform Model: subset of HW solution (also in a multi-core scenario)
- Metrics: results from the Evaluation&Estimation activity
- **Constraints:** F/NF constraints (depending on application domain)

▶OUTPUT:

- **Physical Links:** Possible optimal links and topology.
- **Mapping:** Process to BBs.
- Basic Blocks: Processors, architecture and number of cores.









MECO System Description Models

- HEPSIM (HEPSYCODE SIMulator), the extended SystemC simulator used for HW/SW Co-Simulations in HEPSYCODE:
 - The SystemC library has been extended with a SC_CSP_CHANNEL template class to implement the point-to-point CSP channel semantic



Implemented an Hierarchical Scheduling manager (2-Levels Scheduling)





Example of a Possible Deployment platform



Proceedings of CPS&IoT2019 page 492



"The fundamental issue with MCS is how to reconcile the differing needs of separation (for safety) and sharing (for efficient resource usage)"



8. HepsyCode-MC

MECO education & training HEPSYCODE Multi-Objective Optimization Problem



Multi-Objective Design Space Exploration Optimization Problem

 $\min_{\bar{x}} \quad \bar{F}(\bar{x}) = [f_1(\bar{x}), f_2(\bar{x}), \dots, f_k(\bar{x})]$ subject to $\bar{x} \in \Omega = \{\bar{x} \in \mathbb{N}_{>0}^n : x_i \le (b-r) + r * p_{max}\}$

where $\bar{x} = \{x_1, \ldots, x_n\}$ is an n-dimensional decision variable vector representing processes in the solution space Ω (which refers to a feasible search space, feasible set of decision vectors) and $\bar{F}(\bar{x}) = [f_1(\bar{x}), f_2(\bar{x}), \ldots, f_k(\bar{x})] \in \mathbb{R}^k$ consists of k ≥ 2 real-valued objective functions (\mathbb{R}^k refers to the objective space). The value b is the total number of BBs, r is the number of BBs that have processor type equal to GPP, and p_{max} is the maximum number of HPV-based SW Partition instances for each GPP processor.

Linearization of Multi-objective Design Space Exploration Optimization Problem

$$\min_{\bar{x}} \qquad U(\bar{x}) = \sum_{k} \omega_k \cdot f_k(\bar{x}) = \sum_{k} \omega_k \cdot f_k(x_1, x_2, \dots, x_n)$$

subject to $\bar{x} \in \Omega = \{ \bar{x} \in \mathbb{N}_{>0}^n : 0 < x_i \le (b-r) + r * p_{max} \}$

 $U(\bar{x})$ is the utility function evaluated at each iteration of the GA for each individual $\bar{x} \in \Omega$. f_k represents the value of the objective function (or metric) k for each individual \bar{x} , while ω_k is the weight associated to each objective function or metric. Proceedings of CPS&IoT2019 page 494

MECO education & training HEPSYCODE Multi-Objective Optimization Problem



- The introduction of mixed-criticality requirements reduces the decision space due to the fact that applications with different criticality can not share the same HW block or the same SW partition
 - Introducing *l* criticality levels assigned to each process (with some kind of risk analysis driven by standards and certifications), the decision variable space (without HPV-based software partition) could be divided into different clusters



▶ Decision Variable Space Size (No HPV SW Partitions): $P(b, l) = \frac{b!}{(b-l)!}, \ l \le b$

• *r* processors able to support HPV, *t* processors not able to support HPV

61

▶ Decision Variable Space Size: $P([t + r \cdot p_{max}], l) = \frac{[t + r \cdot p_{max}]!}{([t + r \cdot p_{max}] - l)!}, \ l \leq [t + r \cdot p_{max}]$ Proceedings of CPS&IoT2019 page 495

MECO education & training HEPSYCODE Multi-Objective Optimization Problem



Multi-objective optimization problem: Introduction of HPV-based software partitions into the decision variable space.

Pareto analysis with mixed-criticality constraints: The introduction of Hypervisor technologies increase the number of feasible solutions decreasing the global cost.





Proceedings of CPS&IoT2019 page 496



DSE Approach



11.3 Processes Communication Index

The **Processes Communication Index** is based on the Communication Matrix, calculated in the Co-Estimation step:

$$CM = \begin{bmatrix} cm_{1,1} & cm_{1,2} & \cdots & cm_{1,n} \\ cm_{2,1} & cm_{2,2} & \cdots & cm_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ cm_{n,1} & cm_{n,2} & \cdots & cm_{n,n} \end{bmatrix}$$
(48)

CM is expressed by the number of bits sent/received over each channel. So, for each individual \bar{x} , it is possible to define a *Processes Communication Selection* Matrix, $S^{cm}(\bar{x}) \in \mathbb{R}^{n \times n}$, as listed below:

$$S^{cm}(\bar{x}) = \begin{cases} s^{cm}_{i,j}(\bar{x}) = 1, & \text{if } ps_i \in pu_x \land ps_j \in pu_y \land pu_x \neq pu_y \\ s^{cm}_{i,j}(\bar{x}) = 0.5, & \text{if } ps_i \in pt_x \land ps_j \in pt_y \land pt_x \neq pt_y \\ s^{cm}_{i,j}(\bar{x}) = 0, & \text{otherwise} \end{cases}$$
(49)

So, for each individual \bar{x} , the *Inter Cluster Communication Cost*, $ICCC(\bar{x}) \in \mathbb{R}^{n \times n}$, represents the cost associated to process communication if processes are allocated on different processors:

$$UCCC(\bar{x}) = CM \cdot S^{cm}(\bar{x}) \tag{50}$$

Starting from ICCC matrix, the Normalized Total Communication Cost index is:

$$f_{NTCC}(\bar{x}) = \frac{\sum_{j=1}^{n} \sum_{k=1}^{n} iccc_{j,k}(\bar{x})}{max_{NTCC}}$$

$$max_{NTCC} = \sum_{j=1}^{n} \sum_{k=1}^{n} cm_{j,k}$$
(51)

11.7 Criticality Index

The metric specifically introduced in [29] [30] and extended in this paper to consider HPV-based SW partition is the **Criticality Index**, related to the criticality level associated to each process ps_j . In particular, defined the array $CRIT = \{[crit_1, crit_2, ..., crit_j, ..., crit_n] : crit_j \in \mathbb{R} \text{ is the criticality level associated to process } ps_j\}$, then it is possible to define the *Criticality Index* as:

$$f_{CRIT}(\bar{x}) = \frac{\sum_{j=1}^{n} \sum_{k=j+1}^{n} mc_{j,k}(\bar{x})}{\frac{n\cdot(n-1)}{2}}$$

$$MC(\bar{x}) = \begin{cases} mc_{j,k}(\bar{x}) = 1 & if \ |crit_j - crit_k| > 0 \ \land \ ps_j \in pu_x \ \land \ ps_k \in pu_y \ \land \ pu_x = pu_y \\ mc_{j,k}(\bar{x}) = 1 & if \ |crit_j - crit_k| > 0 \ \land \ ps_j \in pt_j \in pu_x \ \land \ ps_k \in pt_k \in pu_y \ \land \ pt_j = pt_k \ \land \ pu_x = pu_y \end{cases}$$

$$(62)$$

The goal behind this metric is to avoid having processes with different criticality levels on the same (shared) partition/processor/core resource. If the constraint is not satisfied, the index value becomes 1, so the final cost function has a higher value (in term of utility function) if an individual doesn't satisfy criticality constraint.

Proceedings of CPS&IoT2019 page 497





9. Case Studies

"In the future, everyone will be world-famous for 15 minutes – Andy Warhol"







Proceedings of CPS&IoT2019 page 499






V. Muttillo, G. Valente, D. Ciambrone, V. Stoico, and L. Pomante. **HEPSYCODE-RT: a Real-Time Extension for an ESL HW/SW Co-Design Methodology**. In Proceedings of the 10th Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools (RAPIDO '18). ACM, New York, NY, USA, 2018 Proceedings of CPS&IoT2019 page 502









10. Hepsycode Ecosystem

"In the long history of humankind (and animal kind, too) those who learned to collaborate and improvise most effectively have prevailed – *Charles Darwin*"







11.

Publications and European Projects

"Framework Programmes for Research and Technological Development"



Pubblications



Journals

- 1. Muttillo, V., Valente, G., Federici, F., Pomante, L., Faccio, M., Tieri, C., Ferri, S.: "A design methodology for soft-core platforms on FPGA with SMP Linux, OpenMP support, and distributed hardware profiling system", In: EURASIP Journal on Embedded Systems
- Pomante, L., Muttillo, V., Krena, B., Vojnar, T., Veljkovic, F., Pacome, M., Matschnig, M., Fischer, B., Martinez, J., Gruber, T.: "The AQUAS ECSEL Project - Aggregated Quality Assurance for Systems: Co-Engineering Inside and Across the Product Life Cycle", In Microprocessors and Microsystems: Embedded Hardware Design (MICPRO) MINOR REVISION
- 3. Pomante, L., Muttillo, V., Santic, M., Serri, P.: "SystemC -based Electronic System-Level Design Space Exploration Environment for Dedicated Heterogeneous Multi-Processor Systems", In: Microprocessors and Microsystems: Embedded Hardware Design (MICPRO) **SUBMITTED**
- 4. Muttillo, V., Tiberi, L., Pomante, L.: "Benchmarking Analysis of Hypervisor Technologies for Aerospace Multi-core Systems", In: Journal of Aerospace Information Systems (JAIS) **SUBMITTED**

Conferences

- 1. V. Muttillo, L. Pomante, P. Balbastre, J. Simo. HW/SW Co-Design Framework for Mixed-Criticality Embedded Systems considering Xtratum-based SW Partitions. Euromicro Conference on Digital System Design. 2019 **SUBMITTED**
- 2. V. Muttillo. Hw/sw co-design methodology for mixed-criticality and real-time embedded systems. In Design, Automation and Test in Europe (DATE 2019), Ph.D. Forum, Florence, Italy, Mar. 2019.
- 3. V. Muttillo, G. Fiorilli, . Di Mascio. Tuning dse for heterogeneous multi-processor embedded systems by means of a self-equalized weighted sum method. In Workshop on Parallel Programming and RunTime Management Techniques for Manycore Architectures and Design Tools and Architectures for Multicore Embedded Computing Platforms, 2019
- 4. V. Muttillo, G. Valente and L. Pomante, "Design Space Exploration for Mixed-Criticality Embedded Systems considering Hypervisor-based SW Partitions", Euromicro Conference on Digital System Design, Prague, 2018. **BEST POSTER AWARD**
- 5. D. Ciambrone, V. Muttillo, L. Pomante and G. Valente, "HEPSIM: An ESL HW/SW co-simulator/analysis tool for heterogeneous parallel embedded systems" 7th Mediterranean Conference on Embedded Computing, Budva, 2018. **BEST PAPER AWARD**



Pubblications



Conferences

- 6. V. Muttillo and G. Valente, "Injecting hypervisor-based software partitions into Design Space Exploration activities considering mixed-criticality requirements", 2018 7th Mediterranean Conference on Embedded Computing (MECO), Budva, 2018
- 7. V. Muttillo, G. Valente, L. Pomante. "Criticality-aware Design Space Exploration for Mixed Criticality Embedded Systems". In Proceedings of the 9th ACM/SPEC on International Conference on Performance Engineering (ICPE '18), ACM, New York, NY, USA, 2018
- 8. V. Muttillo, G. Valente, L. Pomante, V. Stoico, F. D'Antonio, F. Salice. "CC4CS: an Off-the-Shelf Unifying Statement-Level Performance Metric for HW/SW Technologies". In ACM/SPEC International Conference on Performance Engineering (ICPE '18), 2018, pp. 119-122
- 9. V. Muttillo, G. Valente, L. Pomante. "Criticality-driven Design Space Exploration for Mixed-Criticality Heterogeneous Parallel Embedded Systems". In 9th Workshop and 7th Workshop on Parallel Programming and RunTime Management Techniques for Many-core Architectures and Design Tools and Architectures for Multicore Embedded Computing Platforms (PARMA-DITAM '18), 2018
- 10. V. Muttillo, G. Valente, D. Ciambrone, V. Stoico, L. Pomante. "HEPSYCODE-RT: a Real-Time Extension for an ESL HW/SW Co-Design Methodology. 10th Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools (RAPIDO'18), 2018
- 11. Di Pompeo, D., Incerto, E., Muttillo, V., Pomante, L., Valente, G.: "An Efficient Performance-Driven Approach for HW/SW Co-Design", In: Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering (ICPE)., L'Aquila, Italy, 22-27 Apr. 2017
- 12. Faccio, M., Federici, F., Marini, G., Muttillo, V., Pomante, L., Valente, G.: "Design and validation of multi-core embedded systems under time-toprototype and high-performance constraints", In: Research and Technologies for Society and Industry (RTSI), Bologna, Italy, 7-9 Sep. 2016
- 13. Valente, G., Muttillo, V., Pomante, L., Federici, F., Faccio, M., Moro, A., Ferri, S., Tieri, C.: "A Flexible Profiling Sub-System for Reconfigurable Logic Architectures", In: Parallel, Distributed, and Network-Based Processing (PDP), pp. 373-376, Heraklion Crete, Greece, 17-19 Feb 2016



European Projects



ARTEMIS-JU AIPP 2013-621429 EMC2 (Embedded Multi-Core systems for Mixed Criticality applications in dynamic and changeable real-time environments):

Deliverable D2.3: Design, implementation, prototyping and verification approach for mixed-critical and parallel applications, Sep. 2015

Deliverable D2.4: Intermediate validation report based on selected living labs scenarios, Mar. 2016

Deliverable D2.5: Complete modelling and analysis framework, Oct. 2016

Deliverable D2.6: Comprehense validation report for the modelling frameworks and offline tools, based on refined living labs results, Apr. 2017

H2020 ECSEL RIA 2016-737494 MegaM@rt2 (MegaModelling at Runtime - scalable model-based framework for continuous development and runtime validation of complex systems):

Deliverable D1.2: Architecture specification and roadmap - initial version, Oct. 2017 Deliverable D1.4: Architecture specification and roadmap – final Version, Jun. 2018 Deliverable D2.2: Design Tool Set Specification, Feb. 2018 Deliverable D2.3: Design Tool Set – Initial Version, Jun. 2018 Deliverable D6.3: Dissemination and Exploitation Report – initial release, Feb. 2018

H2020 ECSEL RIA 2016-737475 AQUAS (Aggregated Quality Assurance for Systems):

Deliverable D2.1.1: Domain Environment – Air Traffic Management, Oct. 2017
Deliverable D2.1.5: Domain Environment – Space Multicore Architecture, Oct. 2017
Deliverable D2.2.1: Demonstrator Architecture - Air Traffic Management, Apr. 2018
Deliverable D2.2.5: Demonstrator Architecture - Space Multicore Architecture, Apr. 2018
Deliverable D3.1: Specification of Safety, Security and Performance Analysis and Assessment Techniques, Apr. 2018







HEPSYCODE Repository

HW/SW CO-DEsign of HEterogeneous Parallel dedicated SYstems:

Tool available for free on a git repository under GPL2 for testing, improvements, collaborations etc. Web Site: <u>www.hepsycode.com</u>

You can download the HEPSYCODE tool on this page:

https://bitbucket.org/vittorianomuttillo87/tool-hepsycode/src/master/





12.

Conclusions and Future Work



"Embedded systems are the key innovation driver to improve mechatronic products with cheaper and even new functionalities. They support today's information society as inter-system communication enabler. Consequently, boundaries of application domains are alleviated and ad-hoc connections and interoperability play an increasing role'

MECO Conclusions and future work



- ESL HW/SW Co-Design approach able to take into account mixed-criticality constraints
- The methodology, design flow and framework drive the designer from the input specification to the final implementation solution, while offering timing simulation capabilities, design space exploration activities with the support of analysis tool
- It is possible to integrate this approach with other external tools (like Xamber, but other tools are under evaluation)
- FUTURE WORKS:
 - Consider multi-core scenario while introducing schedulability and RT analysis
 - Combine PAM1 and PAM2 activities into a unique DSE approach
 - Exploit parallel programming techniques (parallel meta-heuristics)
 - Analysis and tests in PAM2 considering also mixed-criticality index
 - Introduce fixed WCET values (taken from external tools)
 - Integrate other external tools to enhance HEPSYCODE functionality
 - Improve the hierarchical scheduling implementation

MECO Conclusions and future work



- Model GR-CPCI-LEON4-N2X Quad-Core 32-bit LEON4 SPARC V8 processor with MMU, IOMMU
- Model TASI/ UNIVAQ Satellite Applications
- Contributed to benchmarking of fully-open Aeroflex Gaisler quad-LEON3 system on FPGA with Xtratum and PikeOS
- Improve case studies example, exploit works into some European Projects



MEC[®] HW/ SW Co-Design Methodologies







THANKS!

Any questions?





TECHNISCHE UNIVERSITÄT WIEN Vienna | Austria

Institut für Computertechnik Institute of Computer Technology

Self-Aware Cyber-physical Systems

Axel Jantsch

TU Wien, Vienna, Austria

Summer School on Cyber-Physical Systems and Internet-of-Things Budva, Montenegro, June 2019

Acknowledgment

Axel Jantsch, David Juhasz, Hedyeh Kholerdi, Amir Rahmani, Nima Taherinejad

TU Wien, Vienna, Austria

Nikil Dutt, Kasra Moazzemi, Amir Rahmani UC Irvine, California

Anil Kanduri, Arman Anzanpour, Elham Shamsa, Iman Azimi, Maximilian Götzinger, Pasi Liljeberg

University of Turku, Finland

Outline



- 2 Architecture for Awareness
- 3 Comprehensive Observation
- 4 Goal Management

5 Conclusion



Outline



- 2 Architecture for Awareness
- 3 Comprehensive Observation
- 4 Goal Management
- 5 Conclusion







 Large number of resources





- Large number of resources
- Many tight constraints





- Large number of resources
- Many tight constraints
- Varying application demands, both within and between applications;





- Large number of resources
- Many tight constraints
- Varying application demands, both within and between applications;
- Functional Aberrations:
 - Design errors or omissions:
 - Malicious attacks;
 - Aging;
 - Soft errors:
- Non-functional Aberrations:
 - Performance;



Power consumption: Proceedings of CPS&IoT2019 page 525



Santanu Sarma et al. "On-Chip Self-Awareness Using Cyberphysical-Systems-On-Chip (CPSoC)". In: Proceedings of the 12th International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS). New Delhi, India, Oct. 2014







Proceedings of CPS&IoT2019 page 527





















Proceedings of CPS&IoT2019 page 532

Autonomy and Adaptivity

Autonomy is the ability to operate independently, without external control.

Adaptivity is the ability to effect run-time changes and handle unexpected events.



Outline



2 Architecture for Awareness

3 Comprehensive Observation

4 Goal Management

5 Conclusion



Self-Awareness Architecture





Proceedings of CPS&IoT2019 page 535

Cyber-Physical SoC


Cyber-Physical SoC





Cyber-Physical SoC





Cyber-Physical SoC





CPSoC - A Sensor Rich SoC Platform



Santanu Sarma et al. "CyberPhysical-System-On-Chip (CPSoC): A Self-Aware MPSoC Paradigm with Cross-Layer Virtual Sensing and Actuation". In: *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE)*. Grenoble, France, Mar. 2015



CPSoC - A Sensor Rich SoC Platform



Nikil Dutt, Axel Jantsch, and Santanu Sarma. "Self-Aware Cyber-Physical Systems-on-Chip". In: Proceedings of the International Conference for Computer Aided Design. invited. Austin, Texas, USA, Nov. 2015

TU

Thermal-Aware Performance



TU

Throughput improvement by 70%-300% for same power and temperature.

Benefit is due to accurate and fine-grain measurement and tight tracking.

Santanu Sarma et al. CyberPhysical-System-On-Chip (CPSoC): Sensor-Actuator Rich Self-Aware Computational Platform. Tech. rep. CECS Technical Report No; CECS TR-13-06. Invine; CA 92697-2620, USA: Center for Embedded Computer Systems University 0162410048, or University 0172119 page 542

Motivation

2 Architecture for Awareness

3 Comprehensive Observation

4 Goal Management

5 Conclusion



Observation Pipeline









Proceedings of CFS&1012019 page 545



Accuracy Systematic errors, a measure of statistical bias.



Accuracy Systematic errors, a measure of statistical bias. Precision Random errors, a measure of statistical variability.



Accuracy Systematic errors, a measure of statistical bias. Precision Random errors, a measure of statistical variability. Data Reliability The extent to which a measuring procedure yields the same results on repeated trials.



Accuracy Systematic errors, a measure of statistical bias.
Precision Random errors, a measure of statistical variability.
Data Reliability The extent to which a measuring procedure yields the same results on repeated trials.
Relevance The quality of being important for the matter at hand.









24

Correct value

High accuracy, high precision











Correct value High accuracy, high precision High accuracy, low precision Low accuracy, high precision

Proceedings of CPS&IoT2019 page 554





Comprehensive Observation



Nima TaheriNejad, Axel Jantsch, and David Pollreisz. "Comprehensive Observation and its Role in Self-Awareness -An Emotion Recognition System Example". In: *Proceedings of the Federated Conference on Computer Science and Information Systems*. Gdansk, Poland, Sept. 2016





TU

Early Warning Score

Score	3	2	1	0	1	2	3
Heart rate ¹	<40	40–51	51–60	60–100	100–110	110–129	>129
Systolic BP ²	<70	70–81	81–101	101–149	149–169	169–179	>179
Breath rate ³		<9		9–14	14–20	20–29	>29
SPO ₂ (%)	<85	85–90	90–95	>95			
Body temp.4	<28	28–32	32–35	35–38		38–39.5	>39.5

 $^1\text{beats}$ per minute, $^2\text{mmHg},\,^3\text{breaths}$ per minute, $^4\ ^\circ\text{C}$





EWS Improvement

- Data reliability:
 - Values in reasonable scope
 - Changes in reasonable scope
 - Consistency between sensors
- Situation awareness
- Power efficiency



Enhanced Early Warning Score



Arman Anzanpour et al. "Self-Awareness in Remote Health Monitoring Systems using Wearable Electronics". In: Proceedings of Design and Test Europe Conference (DATE). Lausanne, Switzerland, Mar. 2017



Enhanced Early Warning Score - Data Reliability

- 1 Check on the reliability of sensed values
- 2 Check on the reliability of value changes
- 3 Check on consistency between sensor data



Enhanced Early Warning Score - Data Reliability

- 1 Check on the reliability of sensed values
- 2 Check on the reliability of value changes
- 3 Check on consistency between sensor data





Enhanced Early Warning Score - Situation Awareness

- 1 Consider the activity mode of person
- 2 Consider time of day
- 3 Consider location



Enhanced Early Warning Score - Situation Awareness

- 1 Consider the activity mode of person
- 2 Consider time of day
- 3 Consider location



Enhanced Early Warning Score - Power Efficiency Prioritize different situations **Priority Score** Indoor-Day Indoor-Night **Outdoor-Day Outdoor-Night** 100 80 High 60 Medium 40 Low 20 Normal Walking Resting Running Sleeping logging



Enhanced Early Warning Score - Power Efficiency

- 1 Prioritize different situations
- 2 Distinguish different modes of urgency

Emergency Level:	Score:0 Normal				Score:1-3 Low				Score:4-6 Medium				Score>6 High						
	Indoor		Outdoor			Indoor		Outdoor		Indoor		Outdoor			Indoor O		Out	Dutdoor	
	Day	Night	Day	Night		Day	Night	Day	Night	Day	Night	Day	Night		Day	Night	Day	Night	
Sleeping	Ε	E	E	Ε		С	D	D	D	в	С	С	С		Α	Α	В	В	
Resting	D	D	D	D		С	С	С	С	в	в	В	В		Α	Α	В	В	
Walking	С	С	С	С	[В	С	С	С	в	В	В	В		Α	Α	Α	В	
Jogging	С	С	С	С		В	В	В	С	в	В	В	В		Α	Α	Α	В	
Running	С	С	С	С		В	В	В	В	В	В	В	В		Α	Α	A	Α	



Enhanced Early Warning Score - Power Efficiency

- 1 Prioritize different situations
- 2 Distinguish different modes of urgency
- 3 Define sensing activity for each mode

State	Respiration Rate Activity	Blood Pressure	Heart Rate, SpO2, and Body Temp.	Transmission Power Consumption
Α	Continuous	Every hour in day Disabled in night	Every sec.	29 mW
в	2 min continuous 8 min OFF	Every hour in day Disabled in night	Every sec.	26.8 mW
С	2 min continuous 3 min OFF	Every 3 hours in day Disabled in night	Every min.	12.5 mW
D	2 min continuous 8 min OFF	Every 3 hours in day Disabled in night	Every min.	7 mW
E	2 min continuous 18 min OFF	Disabled	Every min.	4.3 mW



Enhanced Early Warning Score - Power Efficiency

Over a day half the energy can be saved.





Enhanced Early Warning Score Summary

- Considering data reliability improves quality of observation;
- Considering sitation improves quality of observation;
- Collecting needed data only improves efficiency.



Attention Based Temperature Measurement



Attention Based Temperature Measurement

- How many temperature measurements are required in an MPSoC?
- It varies over several orders of magnitude depending on activity and current temperature.



Attention Based Temperature Measurement

- How many temperature measurements are required in an MPSoC?
- It varies over several orders of magnitude depending on activity and current temperature.



Nima TaheriNejad, M. Ali Shami, and Sai Manoj P. D. "Self-aware sensing and attention-based data collection in Multi-Processor System-on-Chips". In: 15th IEEE International New Circuits and Systems Conference (NEWCAS). June 2017, pp. 81–84




Intel Nehalem processor, running Barnes from SPLASH-2 Benchmarks, using Snipersim and Hotspot.



- When only differences $> \Delta = 1, 2, 5^{\circ}C$ are reported, 7 out of 10 sensors send only 1 value in this experiment.
- Reduction of temperature reports for Memory, ALU and D-Cache:

Unit	$\Delta = 1$	Imp.	$\Delta = 2$	Imp.	$\Delta = 5$	Imp.
Memory	13	35%	9	55%	4	80%
ALU	4	80%	2	90%	1	95%
D-Cache	2	90%	2	90%	1	95%
All others	1	95%	1	95%	1	95%







• Rate of temperature reporting can be significantly reduced and fine tuned;

- Rate of temperature reporting can be significantly reduced and fine tuned;
- Can depend on
 - relative difference,
 - absolute difference,
 - absolute value,
 - system level mode;

- Rate of temperature reporting can be significantly reduced and fine tuned;
- Can depend on
 - relative difference,
 - absolute difference,
 - absolute value,
 - system level mode;
- Potential benefits:
 - reduced processing,
 - reduced communication,
 - reduced measurements.



Motivation

- 2 Architecture for Awareness
- 3 Comprehensive Observation
- 4 Goal Management

5 Conclusion



Goals for Dynamic Task Mapping





Dynamic Task Mapping





Example 1: Performance Driven Task Mapping





MapPro prefers compact and contiguous regions.

Mohammad-Hashem Haghbayan et al. "MapPro: Proactive Runtime Mapping for Dynamic Workloads by Quantifying Ripple Effect of Applications on Networks-on-Chip". In: *Proceedings of the International Symposium on Networks* on Chip. Vancouver, Canada, Sept. 2015



Example 2: Throughput- and Power-Constrained Task Mapping



The patterning algorithm disperses mapped cores to maximize the Thermal Safe Power budget.

Anil Kanduri et al. "Dark Silicon Aware Runtime Mapping for Many-core Systems: A Patterning Approach". In: *Proceedings of the International Conference on Computer Design (ICCD)*. New York City, USA, Oct. 2015, pp. 610–617



Example 3: Lifetime-Reliability-Driven Task Mapping





MapPro: lifetime=5.52 years

Reliability aware mapping: lifetime=12 years

The plots show the reliability of cores at the end of the system's lifetime. The end of the system's life is reached when the reliability of one core drops below 30%.

M. H. Haghbayan et al. "A lifetime-aware runtime mapping approach for many-core systems in the dark silicon era". In: Design, Automation Test in Europe Conference Exhibition (DATE). Mar. 2016, pp. 854–857



1 Single objective; Design time;



1 Single objective; Design time;

2 Multiple objectives; Design time;



- 1 Single objective; Design time;
- 2 Multiple objectives; Design time;
- 3 Multiple objectives; Run time;



- 1 Single objective; Design time;
- 2 Multiple objectives; Design time;
- 3 Multiple objectives; Run time;
- 4 Multiple, hierarchical objectives; Run time;



Hiararchical Goal Management





Supervisory Control



Amir M. Rahmani et al. "SPECTR - Formal Supervisory Control and Coordination for Many-core Systems Resource Management". In: *Proceedings of the 23rd ACM International Conference on Architectural Support for Programming Languages and Operating Systems*. Williamsburg, VA, USA, Mar. 2018; T. R. Mück et al. "Design Methodology for Responsive and Robust MIMO Control of Heterogeneous Multicores". In: *IEEE Transactions on Multi-Scale Computing Systems* PP.99 (2018), pp. 1–1

















55

Hierarchical Goal Mangement



- The system's requirements changes over its lifetime.
- Different objectives are invoked at different time.



Goal Driven Autonomy





State Detection

State vector:

- Power: Violation: TDP < pPotential Violation: $0.8 TDP \le p \le TDP$ No Violation: $p \le 0.8 TDP$
- User Command: High Performance Low Power
- Performance per application: [Min run time, Max run time]



www.ict.tuwien.ac.at

Priority Assignment

- Primary goals: thermal safety
- Secondary goals: User experience
- Tertiary goals: Application requirements



Priority Assignment - Urgency

Urgency is the extent of a violation of a parameter:

$$U_{Pow} = rac{P_{cur}}{P_{ref}}$$

 P_{cur} is the instantaneous power consumption; P_{ref} is the fixed upper bound on power (TDP)



Priority Assignment - Urgency

$$U_{perf} = rac{perf_{max} - perf_{curr}}{perf_{max} - perf_{ref}}$$

perf_{max} the maximum required application performance;

perf_{curr} the instantaneous measured performance;

 $perf_{ref} \begin{cases} \frac{perf_{max} + perf_{min}}{2} & \text{if User Command} = \text{High Performance} \\ perf_{min} & \text{if User Command} = \text{Low Power} \end{cases}$



Goal Enforcement

- Selects action that most likely will satisfy the highest priority goal;
- Action = Resource allocation policy;
- Initial action is randomly selection;
- Actions are assessed in a reinforcement learning loop;
- Reinforcement learning is based on a reward function.



Reward Calculation

 $\textit{Reward} = \textit{W}_0 \times \textit{R}_0 + \textit{W}_1 \times \textit{R}_1 + \textit{W}_2 \times \textit{R}_2 + ... + \textit{W}_n \times \textit{R}_n$

E.g. with two goals for power and performance:

$$egin{aligned} {Reward} &= W_{Power} imes R_{Power} + W_{Perf} imes R_{Perf} \ R_{Power} &= rac{P_{ref} - P_{curr}}{P_{ref}} \ R_{Perf} &= rac{1}{n} \sum_{i=1}^n rac{Perf_i - Perf_{min}}{Perf_{max} - Perf_{min}} \end{aligned}$$

Perf_i Perf_{min}, Perf_{max} n W_i the measured performance of the i_{th} application minimum and maximum required performance the total number of applications running assigned by the priority re-assigner.



Experiments



Experiments with a set of microkernel benchmarks; Hardkernel Odroid XU3 board, with two clusters (4 big (A15) and 4 little (A7) CPU cores; Performance in heartbeats/sec.



Comparison

pwr	

Tech.	Obj	Cmd	Pwr viol.	Perf. viol.	Avg. pwr
LP policy	Power	Х	3%	65%	2.99
HP policy	Perf.	Х	67%	0%	3.8
GDA	Dynamic	\checkmark	20%	34%	3.2



Goal Driven Autonomy



Elham Shamsa et al. "Goal-Driven Autonomy for Efficient On-chip Resource Management: Transforming Objectives to Goals". In: *Proceedings of the Design and Test Europe Conference (DATE)*. Florence, Italy, Mar. 2019

Axel Jantsch et al. "Hierarchical Dynamic Goal Management for IoT Systems". In: Proceedings of the IEEE International Symposium on Quality Electronic Design (ISQED 2018). USA, Mar. 2018

Amir M. Rahmani, Axel Jantsch, and Nikil Dutt. "HDGM: Hierarchical Dynamic Goal Management for Many-Core Resource Allocation". In: IEEE Embedded Systems letters 10.3 (Sept. 2018) Proceedings of CPS&IoT2019 page 605

Motivation

- 2 Architecture for Awareness
- 3 Comprehensive Observation
- 4 Goal Management





Self-Aware Control Loop





Let's Get Out

PROACTIVE COMPUTING

Human-in-the-loop computing has its limits. What must we do differently to prepare for the networking of thousands of embedded processors per person? And how do we move from human-centered to human-supervised computing?

David Tennenhouse. "Proactive Computing". In: Communications of the ACM 43.5 (May 2000), pp. 43–50


Let's Get Out

Let's get physical

PROACTIVE COMPUTING

Human-in-the-loop computing has its limits. What must we do differently to prepare for the networking of thousands of embedded processors per person? And how do we move from human-centered to human-supervised computing?

David Tennenhouse. "Proactive Computing". In: Communications of the ACM 43.5 (May 2000), pp. 43–50



Let's Get Out

- Let's get physical
- · Let's get real

PROACTIVE COMPUTING

Human-in-the-loop computing has its limits. What must we do differently to prepare for the networking of thousands of embedded processors per person? And how do we move from human-centered to human-supervised computing?

David Tennenhouse. "Proactive Computing". In: Communications of the ACM 43.5 (May 2000), pp. 43–50



Let's Get Out

- Let's get physical
- · Let's get real
- Let's get out

PROACTIVE COMPUTING

Human-in the-loop computing has its limits. What must we do differently to prepare for the networking of thousands of embedded processors per person? And how do we move from human-centered to human-supervised computing?

David Tennenhouse. "Proactive Computing". In: Communications of the ACM 43.5 (May 2000), pp. 43–50





Design of Self-Aware Chips





Design of Self-Aware Chips







References I

Robin Arbaud, Dávid Juhász, and Axel Jantsch. "Management of Resources for Mixed-Critical Systems on Multi-Core Platforms with explicit consideration of Communication". In: *Proceedings of the Euromicro Conference on Digital System Design (DSD)*. invited tutorial. Sept. 2018.

Arman Anzanpour et al. "Self-Awareness in Remote Health Monitoring Systems using Wearable Electronics". In: *Proceedings of Design and Test Europe Conference (DATE)*. Lausanne, Switzerland, Mar. 2017.

Nikil Dutt, Axel Jantsch, and Santanu Sarma. "Self-Aware Cyber-Physical Systems-on-Chip". In: *Proceedings of the International Conference for Computer Aided Design*. invited. Austin, Texas, USA, Nov. 2015.

Nikil Dutt, Axel Jantsch, and Santanu Sarma. "Towards Smart Embedded Systems: A Self-Aware System-on-Chip Perspective". In: ACM Transactions on Embedded Computing Systems, Special Issue on Innovative Design Methods for Smart Embedded Systems 15.2 (Feb. 2016). invited, pp. 22–27.

ī.

Nikil Dutt, Amir M. Rahmani, and Axel Jantsch. "Empowering Autonomy through Self-awareness in MPSoCs". In: *Proceedings of the IEEE NEWCAS Conference*. Strasbourg, France, June 2017.

Maximilian Götzinger et al. "Model-free Condition Monitoring with Confidence". In: International Journal of Computer Integrated Manufacturing (2019).

References II

	_

Mohammad-Hashem Haghbayan et al. "MapPro: Proactive Runtime Mapping for Dynamic Workloads by Quantifying Ripple Effect of Applications on Networks-on-Chip". In: *Proceedings* of the International Symposium on Networks on Chip. Vancouver, Canada, Sept. 2015.



Axel Jantsch et al. "Hierarchical Dynamic Goal Management for IoT Systems". In: Proceedings of the IEEE International Symposium on Quality Electronic Design (ISQED 2018). USA, Mar. 2018.

Axel Jantsch, Nikil Dutt, and Amir M. Rahmani. "Self-Awareness in Systems on Chip – A Survey". In: *IEEE Design Test* 34.6 (Dec. 2017), pp. 1–19.

Axel Jantsch and Kalle Tammemäe. "A Framework of Awareness for Artificial Subjects". In: Proceedings of the 2014 International Conference on Hardware/Software Codesign and System Synthesis. CODES '14. New Delhi, India: ACM, 2014, 20:1–20:3.

Anil Kanduri et al. "Dark Silicon Aware Runtime Mapping for Many-core Systems: A Patterning Approach". In: *Proceedings of the International Conference on Computer Design (ICCD)*. New York City, USA, Oct. 2015, pp. 610–617.

S. Kounev et al., eds. Self-Aware Computing Systems. Springer, 2017.

References III

Hedyeh A. Kholerdi, Nima TaheriNejad, and Axel Jantsch. "Enhancement of Classification of Small Data Sets Using Self-awareness - An Iris Flower Case-Study". In: *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*. Florence, Italy, May 2018.

Peter R. Lewis et al. "Architectural Aspects of Self-aware and Self-expressive Computing Systems". In: *IEEE Computer* (Aug. 2015).

Peter R. Lewis et al., eds. Self-Aware Computing Systems: An Engineering Approach. Springer, 2016.

Kasra Moazzemi et al. "Trends in On-Chip Dynamic Resource Management". In: *Proceedings* of the Euromicro Conference on Digital System Design (DSD). invited. Prague, Czech Republic, Sept. 2018.

T. R. Mück et al. "Design Methodology for Responsive and Robust MIMO Control of Heterogeneous Multicores". In: *IEEE Transactions on Multi-Scale Computing Systems* PP.99 (2018), pp. 1–1.

Amir M. Rahmani et al. "SPECTR - Formal Supervisory Control and Coordination for Many-core Systems Resource Management". In: *Proceedings of the 23rd ACM International Conference on Architectural Support for Programming Languages and Operating Systems*. Williamsburg, VA, USA, Mar. 2018.

Amir M. Rahmani, Axel Jantsch, and Nikil Dutt. "HDGM: Hierarchical Dynamic Goal Management for Many-Core Resource Allocation". In: *IEEE Embedded Systems letters* 10.3 (Sept. 2018).

Proceedings of CPS&IoT2019 page 618

References IV

		Σ.
	_	
Ξ		

1

Santanu Sarma et al. CyberPhysical-System-On-Chip (CPSoC): Sensor-Actuator Rich Self-Aware Computational Platform. Tech. rep. CECS Technical Report No: CECS TR-13-06. Irvine, CA 92697-2620, USA: Center for Embedded Computer Systems University of California, Irvine, May 2013.

Santanu Sarma et al. "On-Chip Self-Awareness Using Cyberphysical-Systems-On-Chip (CPSoC)". In: *Proceedings of the 12th International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. New Delhi, India, Oct. 2014.

Santanu Sarma et al. "CyberPhysical-System-On-Chip (CPSoC): A Self-Aware MPSoC Paradigm with Cross-Layer Virtual Sensing and Actuation". In: *Proceedings of the Design*, *Automation and Test in Europe Conference and Exhibition (DATE)*. Grenoble, France, Mar. 2015.

Elham Shamsa et al. "Goal-Driven Autonomy for Efficient On-chip Resource Management: Transforming Objectives to Goals". In: *Proceedings of the Design and Test Europe Conference* (*DATE*). Florence, Italy, Mar. 2019.

David Tennenhouse. "Proactive Computing". In: *Communications of the ACM* 43.5 (May 2000), pp. 43–50.

Nima TaheriNejad, Axel Jantsch, and David Pollreisz. "Comprehensive Observation and its Role in Self-Awareness - An Emotion Recognition System Example". In: Proceedings of the Federated Conference on Computer Science and Information Systems. Gdansk, Poland, Sept. 2016.

References V



Nima TaheriNejad, M. Ali Shami, and Sai Manoj P. D. "Self-aware sensing and attention-based data collection in Multi-Processor System-on-Chips". In: *15th IEEE International New Circuits and Systems Conference (NEWCAS)*. June 2017, pp. 81–84.



Proceedings of CPS&IoT2019 page 621

.







Platform-aware Model-driven Optimization of Cyber-Physical Systems

Tradeoff analysis between Quality-of-Control and degree of approximate computing for image based control systems

Dip Goswami, Sajid Mohamed, Sayandip De Eindhoven University of Technology, The Netherlands



Motivation – image-based control (IBC)



Image-based control – embedded processing







Image-based control – timing challenges

- Camera framerate is much higher that the rate at which frames are processed; several frames are not processed;
- For 60 fps(frames per second) camera, f_h = 16.67 ms;
- Sensing, computing and actuating tasks have WCETs τ_{s},τ_{c} and τ_{a}
- $\tau_s \gg \tau_c + \tau_a$



Image-based control – example

- For 60 fps(frames per second) camera, f_h = 16.67 ms;
- Sensing, computing and actuating tasks have WCETs τ_s , τ_c and τ_a
- $\tau_t = \tau_s + \tau_c + \tau_a$
- $\tau = \left[\frac{\tau_t}{f_h}\right] f_h$
- E.g., τ_t =84 ms; τ =100 ms
- h= τ=100 ms



Response with h= τ **=100 ms**



Measurement range

Proceedings of CPS&IoT2019 page

Actuation (T_a)



What can we do?

- Pipelining sensing task \rightarrow shorten sampling period h and unaltered long delay τ
- Parallelizing sensing task → shorten both sampling period h and delay τ; limited by degree of parallelism;
- Approximation of the sensing block \rightarrow shorten both sampling period h and delay τ



Background – Approximate computing





Particular Technology Node







































Error Resilient Applications



Image Processing & Compression



Navigation



Biometric Security



Web Browsing

OCPS

No single accurate result!!!

IBC with image approximation in the loop

 Image processing applications work as part of a bigger closed-loop system



What is the impact of image approximation on the bigger closed-loop system? How to analyse this impact?



Tutorial coverage

- Basics of image-based control design and challenges
- Background on image approximation
- Integration and analysis of image approximation in an IBC
- Performance evaluation and trade-off analysis
- Case-study: Lane Keeping Assist System (LKAS)
- Evaluation framework IMACS
- Results and (possibly) demo



Image Approximation: tuning knobs





IMACS framework: overview





IMACS: Software-in-the-Loop simulator


IMACS: Hardware-in-the-Loop simulator





Results: Profiling

- Intel i7 processor @2.6GHz
- ISP stage is the most compute-intensive \rightarrow approximate





Results: Profiling

Output of accurate_{ISP}



Output of approximate_{ISP} knob1

Output of approximate_{ISP} knob2







Proceedings of CPS&IoT2019 page 647

Results: Control Performance

- Without considering improved timing
- Performance deteriorates for approximated images (v1 v7)
 - Still acceptable for control





Results: Performance

- Considering improved timir
- Performance improves





Conclusions

- Image-based control suffers from long processing delay;
- Image-approximation is one primising approach to deal with long delay and save compute energy;
- There are several knobs that decides the performance of the overall IBC system;
- Extensive design space exploration is required design sweetspots;







D.Goswami@tue.nl
S.Mohamed@tue.nl
Sayandip.De@tue.nl



Proceedings of CPS&IoT2019 page 651



Security of Embedded and Cyber-Physical Systems

Radek Fujdiak

fujdiak@vutbr.cz

Brno University of Technology, Czech Republic

Proceedings of CPS&IoT2019 page 652



- Embedded system is electronic device or product with certain components:
 - microprocessor(s),
 - software,
 - peripherals,
 - $\circ\,$ and several optional parts such as external memory.
- There are plenty of use-cases for these, such as:
 - aircraft industry,
 - automobile industry,
 - network devices,
 - and many others.



 Today more than ever, the cyber-criminality, hacktivism and other illegal activities in the cyber-space are growing exponentially.

- Several examples around the world:
 - Everything start with Stuxnet? Of course not, ...
 - 2007, Syrian radar was hacked (temporally disabled) before serious military operation
 - 2012, back-door in computer chip was found in Boeing 787s
 - and we can continue with many different examples from today cyberwarfare such as Estonian cyberattack in 2007, South Korea cyber attack in 2009, Burma cyberattack in 2010, Singapore cyberattack in 2013, and many others.



- There are also non-security aspects, which must be considered when speaking about security such as:
 - type of network,
 - type of communication,
 - bandwidth,
 - maximum coupling loss,
 - frequency bands,
 - maximum downlink and uplink data rates,
 - daily downlink and uplink throughput,
 - cost.



- Security ≠ Safety !
- Security might be defined in several ways lets stick with this definition:

"Security is the ability of an entity to protect assets"

- Embedded system security is nothing else than previously defined security in the context of embedded systems.
- There many different levels such as: software security, hardware security, network security and many others.



- In the beginning, there were no specific (crypto-/cyber-)security assets.
- Then come so called CIA Triad <u>C</u>onfidentiality, <u>Integrity</u> and <u>A</u>vailability.
- However, this approach was not sufficient with more and more difficult cyber-environment. Therefore, "Security Star" was introduced, which brought also authentication and nonrepudiation.
- In 2002, the Parkerian Hexad was established, which bring another three additional components to the CIA Triad: possession, authenticity, and utility.
- However, today is not enymore that simple.



- Credentials such as keys, initial vectors, salt, pepper and others.
- The basic of all algorithms relies on credentials and their lifecycle:
 - Their creation random number generators their entropy, randomness and other parameters,
 - Their process delivery, saving, exchange and more,
 - Their destruction each credentials end with its destruction.



• There are different protection of identity, we will divide it to static and dynamic ones.

- Extra techniques might be such as privacy-preserving measures to minimise the use of permanently allocated identifiers which could be intercepted and correlated with device activity over time.
- An example of this is the TMSI (Temporary Mobile Subscriber Identity) allocated by 3GPP networks to address the mobile device instead of the IMSI (International Mobile Subscriber Identity) which is only used once each time the device is powered on.



- Authentication is process of verifying/identifying the source/party/individual/device/...
- There are various parties involved in the establishment of network connectivity, each of which may desire to authenticate themselves to the other parties
- Authentication should be considered from point of:
 - device identificators, certificates,
 - network identificators, certificates, ...
 - message (H)MAC,
 - and subscriber handled by "own", "know" and "are" methods.



- Integrity protection ensures that any tampering with the content of the communication by a "man in the middle" can be detected by the intended recipient.
- With layered network architectures, control information (for example, routing addresses) is processed at different layers.
- Integrity protection applied at higher layers may protect application data but not the lower level control information, so we should consider control and data integrity protection separately.
- Data integrity is mostly handled via simple CRCs, hashes, etc.



- Data confidentiality means protecting the information from disclosure to unauthorized parties.
- Data confidentiality is almost always achieved by encryption of the data.
- Although a lot of attention is paid to the strength of encryption algorithms and the length of encryption keys. However, in practice these are not the most important factors in the effectiveness of the security.



E2M/E2E SECURITY

 End-to-Middle security means that security is handled only partially on the whole chain. On the other hand, end-to-end security means that security is handled on the whole communication chain.

• End-to-Middle Security and End-to-End Security is also very important aspect of the architecture.

• End-to-Middle security is mostly provided by third-parties such as operators.

• End-to-End security must be provided mostly by the user.



- Forward Secrecy is just simple protection against historical breaks of security, which would cause future security breaches.
- This is often handled via so called one-time credentials such as a "session key" for the encryption and others.
- Typical example might be an ephemeral Diffie-Hellman Key Exchange, which is a public key-based algorithm, but relatively computationally intensive, thus perhaps unsuitable for many low power devices.



 Replay protection is a security property of a protocol such that messages recorded by an attacker will not be accepted by their recipient as legitimate if they are reinserted into the communications link later.

- This is important in scenarios where the content of the message is linked to some kind of commercial transaction, or, for example, where an attacker wishes to evade detection by a surveillance device by disabling it and replacing its transmissions with previously recorded normal activity.
- The protection is often made via counters, timestamps and other similar methods.



- Reliable delivery is a service, which ensures the availability.
- It indirectly affects other security characteristics as, without reliable delivery of messages, attackers could potentially block delivery of certain messages without the device and/or the network being aware of it.

• The main risk come from selectively blocking a few messages, which might be unnoticed and benefit an attacker, for example avoiding their being detected by a surveillance device.



• The prioritization is when tasks, processes, messages and other entities involved might get prioritized over others in case of need.

• This is mostly crucial in critical application, critical infrastructures, safety-hazard applications, etc.

• Most common solutions are QoS, flow controls, control functions and other similar methods.



 Updatability is an ability to renew the software, firmware, credentials, algorithms, and other important part involved in the processes.

- New vulnerabilities are discovered every day, and the most effective response to such vulnerabilities is to patch affected devices with updates.
- The OTAA is always preferred method as there might be already tens, hundreds, thousands and more devices deployed and it is very inefficient to try update each one-by-one.



 Network Monitoring and Filtering is the one real-time service, which might prevent real attacks from their impact on the assets.

 Not a main attribute as previous, but very important part of the "whole" conceptual solution.

• There are systems such as IPS or IDS, which tried to detect or even prevent the systems from being attacked.



- Algorithm Negotiation is an important surveillance services similar to updatability.
- Algorithm negotiation provide possibility to select between the algorithms, i.e., when one of the algorithms get broken as over time, the use of certain cryptographic algorithms may be deprecated, either due to advances in computing power or to flaws being discovered in their mathematical basis or design.
- Simple example might be with selecting between RC4, MD5 and SHA.



- Class Break Resistance helps in situations where there are a large number of devices of the same type deployed, the risk of a class break occurs when the system design is such that an attacker who finds a way to compromise one device is then able to easily use the same method to compromise other devices of the same type.
- This risk frequently arises when devices share the same secret or private keys, so that if the key from one device is exposed, perhaps by a lengthy or difficult attack, it can then be easily used to compromise other devices.
- Best practice is to ensure that secret and private keys are unique to each device.



• There are very few certification authorities for the end-devices from the point of security.

- Mostly you will find institutions, which tries to ensure that security requirements are met if needed (i.e., medical devices).
- However, mostly you will face proprietary devices made by regular engineers not security experts.
- Always ask for security documentations without black holes, risk-analysis and security optimization made, standards which are fulfilled or used algorithms, ...



• IP Network is just last example what all types of parameters might be involved in the security.

• IP network is most probably the most comfortable and known environment in case of cyberspace.

 Considering many different proprietary solutions, networks, communication technologies – IP Network will be also the most dangerous environment with most of the criminal activity (compared for example with proprietary radio solution, where attacker would need to first study the environment).



- You will most probably not protect your yard with M1A2 SEP (USA tank) or Challenger 2 (UK tank) - even if you of course could.
- Thus very similarly it works with the cybersecurity.
- Most importantly, you need to establish the use-case specifics.
- Then you should define attack vectors, potential vulnerabilities and risks.
- ... and finally, select the right defense.



Thank you for your attention.

fujdiak@vutbr.cz





- <u>https://www.slideshare.net/adelbarkam/introduction-to-</u> <u>embedded-system-security</u>
- <u>https://www.slideshare.net/MalachiJones/embedded-systems-security-54730736</u>
- <u>https://slideplayer.com/slide/10206947/</u>
- <u>https://fhcouk.files.wordpress.com/2017/05/lpwa-security-</u> white-paper-1_0_1.pdf





Brain-Inspired Computing for Smart CPS and IoT

M. Shafique



Smart Cyber Physical Systems & Internet-of-Things



Smart Traffic Control https://www.emaze.com/@ACIOWOWR/IMSA-Slide-Show



Smart Health Care



Smart Automobiles http://www.it5g.com/latest-softwareenhancements-in-the-auto-industry/



Smart Transport Systems

https://www.automotiveworld.com/analysis/automotivecyber-physical-systems-next-computing-revolution/

Applications



Smart Robots http://alpha-smart.com/alphaboten



Smart Grids

Smart Houses

1:

Q

6

https://www.linkedin.com/pulse/smart-homesprivate-sepure-future-intelligent-home-tripti-iha

6

Q

Ŷ

http://solutions.3m.com/wps/portal/3M/en EU/Sma rtGrid/EU-Smart-Grid/

CP Factory Wireless communication via RFID, NFC and WLAN

Industry 4.0: **Smart Industrial Automation** https://vimeo.com/145877805

3

ML Applications => requiring High Efficiency Gains

Autonomous Driving

Image Classification



Object Detection & Localization





Proceedings of CPS&IoT2019 page 679

11.60 52.35

5,8

Smart CPS & IoT => The Big Data Processing Challenge!



AI / ML is inevitable, we have to efficiently infer knowledge from the big data, and derive predictions
Smart CPS & IoT => The Big Data Processing Challenge!

... should consider

Performance

ThroughputLatency

Robustness

ReliabilitySecurity



Smart Healthcare (Energy and time constraints)



Norwegian C-130 crash (2012)

https://en.wikipedia.org/wiki/2012 Norwegian_C-130_crash



Satellite imagery of the Northeastern United-States taken before and during the blackout



Failure of F-22 Raptor (2007)

http://www.dailytech.com/Lockheeds +F22+Raptor+Gets+Zapped+by+Inte mational+Date+Line/article6225.htm



Toronto, on the evening of August 14, 2003

Northeast blackout of 2003

https://en.wikipedia.org/wiki/Northeast_blackout_of_2003

Others

- □ Adaptability
- Safety
- Privacy
- □ Interoperability

Hacking Jeep Cherokee 4x4 (2015)

Sent the instructions through Entertainment systems

- Change the in-car temperature
- Control the steering
- Control the braking system

https://www.ophtek.com/4-real-life-examples-iot-hacked/ https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/ Proceedings of CPS&IoT2019 page 681



Smart CPS & IoT => The Big Data Processing Challenge!



Challenging Question

How to process such huge amount of data in power/energy efficient way, while providing robustness?

PrivacyInteroperability

Hacking Jeep Cherokee 4x4 (2015)

Sent the instructions through Entertainment systems

- Change the in-car temperature
- · Control the steering
- Control the braking system

https://www.ophtek.com/4-real-life-examples-iot-hacked/ https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/





Why to care about Low-Power Computing? Power is the Limiting Factor for Technology Scaling

Power

- Power wall vs. core count
- ❑ Leakage → significant part

Leads to high Temperature

Aggravates reliability

High energy => Reduced Battery Lifetime







Power density continues to get worse



Proceedings of CPS&IoT2019 page 683 Intel's microchip processor families. (Source: Intel)

8

Why to care about Low-Power Computing? Power is the Limiting Factor for Technology Scaling

8.0

7.0

6.0

€^{5.0} €4.0

3.0

1.0

Static

DPower

- Power wall vs. core count
- **D** Leakage \rightarrow significant part
- Leads to high Temperature
 - Aggravates reliability

All Computing is Low-Power!



(src: A. Cuomo, ST Micro, Stockholm, Sept.8, 2004)

Proceedings of CPS&IoT2019 page 684 Intel's microchip processor families. (Source: Intel)

(source: ITRS Roadmap 2010)

Dynamic

9

Why to care about Low-Power Computing? High Power => High Cost and CO₂ Emissions

2014: \$143 Billion investment worldwide for new data centers

By 2030: Power consumption of Data Canters will be **7.5%** (i.e., **2967 TW/h**) of the total global electric power supply.



2010 2012 2014 2016 2018 2020 2022 2024 2026 2028 2030

Power requirement for Yole Data Center in France

Components/Systems	Power (KW)
Cooling System	52.5
Computational Load	60
Lighting and other overheads	7.5
UPS and Power Distribution	30



High Power/Energy Requirement => Bad CO_2 Balance Computing systems => Appx. 2% of global CO_2 emissions

Why to care about Low-Power Computing? High Power => High Cost and CO₂ Emissions



High Power/Energy Requirement => Bad CO_2 Balance Computing systems => Appx. 2% of global CO_2 emissions





Brain-Inspired Computing (BrISC)



Machine Learning and the Computing Efficiency Gap!



IBM Blue Gene supercomputer Proceedings of CPS&IoT2019 page 688

Machine Learning and the Computing Efficiency Gap!



We need 10,000x Gains to Bridge this Gap!!!





IBM Blue Gene supercomputer Proceedings of CPS&IoT2019 page 689

Complexity of Neural Networks

Different DNN architectures and their resource requirements



Framework	fps (NVIDIA Jetson TXI)	IOU/mAP.
Fast YOLO	17.85 [1]	52.7 [2]
O-YOLOv2	11.8 [1]	65.1 [1]
YOLOv2	5.4 [1]	67.2 [1]

Sources:

[1]: Shafiee et al. @ arXiv:1709.05943 (2017)

[2]: Wei@ECCV1'6

Complexity of Neural Networks



Sources:

[1]: Shafiee et al. @ arXiv:1709.05943 (2017)

YOI Ov2

[2]: Wei@ECCV1'6

67.2 [1]

5.4 [1]

Brain-Inspired Computing: Research@CARE-Tech.



Approximate Computing: A Motivating Example

Adder Accuracy	100%	98.58%	85.85%	74.16%	74.9%
PSNR (1 st Row)	Inf	42	35.53	28.37	19.18
PSNR (2 nd Row)	Inf	40.61	34.45	28.95	20.56
Power*	1	0.63	0.53	0.18	0

Proceedings MPACT Configurations [A. Ragunathan and K. Roy@TCAD'13] 18

Approximate Computing: Our Cross-Layer Approach



Proceedings of CPS&IoT2019 page 694

Low-Power Approximate 1-Bit Full Adders

Ι	Inputs		AccuFA A		Apx	ApxFA ₁		ApxFA ₂		ApxFA ₃		ApxFA ₄		ApxFA ₅	
A	В	C_{in}	Sum	Cout	Sum	C_{out}	Sum	Cout	Sum	Cout	Sum	Cout	Sum	C_{out}	
0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	
0	0	1	1	0	1	0	1	0	1	0	1	0	0	0	
0	1	0	1	0	0	1	1	0	0	1	0	0	1	0	
0	1	1	0	1	0	1	0	1	0	1	1	0	1	0	
1	0	0	1	0	0	0	1	0	1	0	0	1	0	1	
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	
1	1	0	0	1	0	1	0	1	0	1	0	1	1	1	
1	1	1	1	1	1	1	0	1	0	1	1	1	1	1	
Area [GE]		4.4	41	4.2	23	1.9	94	1.	59	1.76		0			
Роч	<i>Power [nW]</i> 1130		771		29	294		98	416		0				
#Er	#Error Cases 0)	12	2	12	2	3		3		2	4		

IMPACT Configurations [A. Ragunathan and K. Roy@TCAD'13]

Low-Power Approximate 1-Bit Full Adders



IMPACT Configurations [A. Ragunathan and K. Roy@TCAD'13]

Low-Power Approximate Multi-Bit Adders

	Area [GE]	Latency [ns]	Power [nW]	Number of Error Cases	Max Error Magnitude	Occ. of Max Error
AccuAdd	48.157	1.03	3250	0	0	0
ApproxAdd ₁ , 4 LSBs	38.279	0.93	2040	89600	15	512
ApproxAdd ₁ , 8 LSBs	28.400	0.89	1680	117950	255	2
ApproxAdd ₂ , 4 LSBs	36.868	0.72	1730	108288	15	768
ApproxAdd ₂ , 8 LSBs	25.578	0.49	1350	126891	255	3
ApproxAdd ₄ , 4 LSBs	29.988	0.52	926	122880	8	8192
ApproxAdd ₄ , 8 LSBs	12.348	0.06	480	130560	128	512

2x2 Approximate Multipliers

	ApxMul _{SoA}				ApxMul _{our}					ſ	LEGEND		
	00	01	10	11		00	01	10	11		AccMul	Accurate Multiplier	
00	000	000	000	000	00	0000	0000	0000	0000		ApxMul _{SoA}	State-of-the-Art (SoA)	
01	000	001	010	011	01	0000	0000	0010	0010			P Kulkarni @ VI SID'11	
10	000	010	100	110	10	0000	0010	0100	0110		CfgMulson	SoA Configurable Multiplier	
11	000	011	110	111	11	0000	<u>0010</u>	0110	1001		ApxMul _{Our}	Our Approximate Multiplier	
Correction: Adder Corre					orrec	tion	Inv	erter		CfgMul _{Our}	Our Configurable Multiplier		

	AccMul	ApxMul _{SoA}	CfgMul _{SoA}	ApxMul _{Our}	CfgMul _{Our}
Area [GE]	6.880	3.704	7.232	4.939	6.350
Power [nW]	542.9	363	525	262	379
No. of Error Cases	0	1	-	3	-
Max. Error Value	0	2	-	1	-

All Tested Approximate Multipliers: Circuits



DSE Results for 8x8 Multipliers





□ 3x10¹² Possible Different Configurations

□ 19 Selected Design-Points Filtered to:

□ A, B3 and B6, C1 and C3, D6, D7, and D8

Inherent Resilience of the HEVC Motion Estimation: Case Study for Low-Power Approximate Accelerator

□ SAD Value of 4 Candidates

□ S1 = 565 => 540 □ S2 = 600 => 610 □ S3 = 475 => 460

□ S4 = 560 **=> 550**

Choosing the minimum → S3 will be selected



Approximate Architecture for High Efficiency Video Coding (HEVC)



 Tiles with Arrays of Heterogeneous
 Approximate
 Variants of SAD to support highthroughput

LUT for Variant Selection based on User Constaints

Unused Tiles are Power-Gated

Approximate Architecture for High Efficiency Video Coding (HEVC)



Approximations for Energy-Efficient Machine Learning

Loop Perforation, Pruning, SW/Algo Level Quantization, Data Decimation, Linux Skipping Computation, Variability-aware Software, Compiler for AC Approximate memory cells, Approximate functional units, Analog processing unit Approximate Circuits Computing for Machine Learning

Approximate Accelerators, Relaxed communication, Approximate cache, Approximate SPMs, Stochastic processors



[Hanif, et al @DATE'48,SJOLPE'48,0Springer'19]

Approximations for Energy-Efficient ML

Our Cross-Layer Approximation Methodology

> Model Compression

Quantization

Hardware-Level Approximations



[Hanif, et al @JOLPE'18]

Results using Alexnet on ImageNet Dataset

1. Structured Pruning

2. Quantization



3. Hardware Approximation



[Hanif, et al @JOLPE'18]

Proceedings of CPS&IoT2019 page 706

DNN Challenges for GPUs



- □ Our goal: reduce memory requirements and efficiency at inference stage.
- Drawback: increase training time (but this penalty is typically affordable to achieve a better DNN design).
- ⇒ POSITIVE EFFECT: pruning has a regularizing effect, then at the first stages the accuracy is higher than the baseline.

DNN Pruning: Methods Comparison

- Class-Distribution (CD): a certain threshold T is selected and, for every layer, all the parameters below σT are pruned, where σ is the standard deviation.
- Class-Uniform (CU): a certain percentage x is selected and, for every layer, the smallest x% parameters are pruned.
- Class-Blind (CB): a certain percentage x is selected and the smallest x% parameters of the entire model are pruned, without keeping uniform sparsity for each layer.



PruNet Methodology

□ Hyper-parameters:

- Reduce learning rate.
- Set pruning percentage at every iteration.
- Set threshold for maximum accuracy loss.
- Pruning process (at each) iteration):
 - Sort weight according to the absolute value.
 - Prune the lowest ones.
 - Apply a mask for each weighted layer.
 - Retrain the network.
- Terminate the process when the accuracy loss falls above the threshold.





Memory Saving Ratio Proceedings of CPS&IoT2019 page 710

35

Case study for Error Resilience analysis

□ Monte-Carlo simulation for analyzing error resilience



Adder	Approximation applied to individual Feature Maps											
Configuration	FM ₁	FM ₂	FM ₃	FM ₄	FM ₅	FM ₆	Average					
α1	92%	92%	92%	92%	90%	92%	92.00%					
α_2	56%	92%	60%	84%	92%	94%	80.00%					
α3	56%	92%	60%	88%	93%	94%	80.67%					
$lpha_4$	92%	92%	92%	92%	91%	92%	92.00%					
α_5	94%	92%	92%	94%	92%	94%	93.34%					
Average	78%	92%	79%	90%	92%	93%	87.60%					

Case study for Error Resilience analysis

Example for employing approximation in two data-paths (Best vs Worst)

Adder	Approximation applied to individual Feature Maps										
Configuration	FM_1	FM ₂	FM ₃	FM ₄	FM ₅	FM ₆	Average				
α1	92%	92%	92%	92%	90%	92%	92.00%				
α_2	56%	92%	60%	84%	92%	94%	80.00%				
α3	56%	92%	60%	88%	93%	94%	80.67%				
$lpha_4$	92%	92%	92%	92%	91%	92%	92.00%				
α_5	94%	92%	92%	94%	92%	94%	93.34%				
Average	78%	92%	79%	90%	92%	93%	87.60%				

■ FM1, FM3 ■ FM5, FM6



Case study for Error Resilience analysis

Example for employing approximation in two data-paths (Best vs Worst)



DNN Systolic Array with Curable Approximations

Modified systolic array architecture based on curable approximate modules

Conventional Systolic Array (with Accurate Modules) Proposed Systolic Array (based on Curable Approximation)

Weights

Weights

1.5x PDP reduction @ NO Accuracy Loss



[Hanif, et al. @DAC'19]

DNN Systolic Array with Curable Approximations

Accurate Module

Consider an example system composed of N stages



Proposed MAC Unit Designs

Based on

- Baugh-Wooley algorithm for performing signed multiplication
- □ Wallace tree reduction
- Truncation of the final addition stage for building curable approximate modules


DNN Systolic Array with Curable Approximations

- □ Cadence Genus, TSMC 65 nm
- Our Systolic Array offers better Latency and PDP

Appx. and Curable MAC Units

	Latency [<i>ps</i>]	Area [cell area]	Power [<i>µW</i>]
Accurate_MAC	1871.1	746	66.56
DAx_MAC	1214.2	744	66.3
C&DAx_MAC	1214.2	746	68.13



43

Evaluation of Systolic Arrays: Accurate, Approximate and Curable



Energy-Efficient Deep Learning Architectures

Deep Learning Applications (CNN, CapsNets)



Efficient Computing Array



Efficient Accelerators



Efficient Hardware Components



Analysis & Optimization



Proceedings of CPS&IoT2019 page 718

Energy-Efficient Deep Learning Architectures



Proceedings of CPS&IoT2019 page 719

What is a CapsuleNet?



Proceedings of CPS&IoT2019 page 720

Analysis: CapsuleNet complexity



CapsAcc Architecture Design



Data-Flow Mapping

- 1: for(l=0; l<L; l++) //output capsules
- for(k=0; k<K; k++) //output channels 2:
- 3: for(j=0; j<J; j++) //input capsules

ဖ Ш.

I

- 4: for(i=0; i<I; i++) //input channels
- 5: for(g=0; g<G; G++) //output columns in a feature map
- for(f=0; f<F; f++) //output rows in a feature map 6:
- for(c=0; c<C; c++) //kernel/input columns 7:
- for(r=0; r<R; r++) //kernel/input rows 8:
- Sum += Weight Data //multiply and accumulate 9:

Weight Memory Control Weight Routing Unit **Buffer Buffer** Systolic Array Memory Data Buffer Data Accumulator , Si



Bind Sigepaps:

Activation: squash

Experimental results



Open-Source Libraries of Approximate Modules

Library 1 and 2: Low-Power Approximate Adders and Multipliers

https://sourceforge.net/projects/lpaclib/

Libi

https

Home / Browse / SEALPAA IpACLib and SEALPAA An Open-Source Library for Low-Power Approximate Computing Modules Brought to you by: kamranayub 1100 oht to you by: swahlah, welharouni Summary Files Reviews Support Code Tickets Wiki Discussion Summary Files Reviews Support Wiki Git Tickets Discussion SVN * Add a Review * Add a Review 1 Download cosus ✤ 30 Downloads (This Wee 1 Last Update: 6 days ago fft Last Update: 2016-04-03 Browse All Files Geel 0 afe Like O Browse All Files Tweet G+1 0 1 Like 23 Description Test Cases 2

https://sourceforge.net/projects/sealpaa/

Enables research and development at higher system layers and easy reproducible comparisons

Summary Files Reviews Support Wiki Code Tickets Discussion	Summary Files Reviews Support Code Tickets Wiki
★ Add a Review ↓ 3 Downloads (The West) GL Last Update 2015-05-29	★ Add a Review ↓ 0 Downloads (This Week) ⑤ Last Update: 2017-05-11
🕑 Tirect 🚱 🕡 👔 🕼 🔕 Browse All Files	Tweet G1 0 14 Like 0 Browse All Files
Description We provide MATLAB and Veniog Models of GeAr, and previously proposed adders (ACA-I, ETAII, ACA-II and GDA) at http://sourcelorge.nel/projects/agrosuddentb/ GeAr is a low latency Generic Accuracy Configurable Adder that provides a higher number of potential configurations compared to state-of-the-at approximate adders, thus enabling a high degree of flexibility and trade-off between performance and output quality. These MATALB and Veniog models can allow software programmer as well as hardware designers to evaluate their code and design. To the best of our knowledge, this is the first open-source library of	Image Image <th< td=""></th<>
Shafique et al. @ DAC'15	Hanif et al. @ DAC'17

Reliability and Security for Machine Learning Systems



- M. A. Hanif, F. Khalid, R. V. W. Putra, S. Rehman, M. Shafique, "Robust Machine Learning Systems: Reliability and Security for Deep Neural Networks", in IOLTS-2018, Platja d'Aro, Spain, pp. 257 - 260.
- F. Kriebel, S. Rehman, M. A. Hanif, F. Khalid, M. Shafique, "Robustness for Smart Cyber-Physical Systems and Internet-of-Things: From Adaptive Robustness Methods to Reliability and Security for Machine Learningedirs VLSP-2018;0Hong Kong, China, pp. 581-586.

Impact of Memory Bit-Flip Errors on Accuracy



On-going Analysis with reduced-precision, fixed-point, and compressed networks

Security Vulnerabilities in Machine Learning



Attacker' Goals: Confidence Reduction, Random Misclassification, Targeted Misclassification, IP Stealing

Data Poisoning Attacks during Training

Attack 1: In this attack certain portion of the original dataset is intruded which is then used for training.



Attack 2: Instead of perturbing the original dataset, in this attack we extend the dataset with intruded samples which is then used for training.



Data Poisoning Attacks: Training



Data Poisoning Attacks: Training



To develop a successful (less impact on validation accuracy) Data Poisoning Attack, append the intruded samples in original dataset instead of changing the original dataset

Limitations:

- 1. Require the access to the complete training dataset
- 2. The attack Noise is perceptible.



Imperceptible Data Poisoning Attacks during Inference



F. Khalid, M. A. Hanif, S. Rehman, <u>M. Shafique</u>, "TrISec: Training Data-Unaware Imperceptible Security Attacks on Machine Learning Modules of Autonomous Vehicles", IOLTS2019. [Link: <u>https://arxiv.org/abs/1811_010311]</u> s of CPS&IoT2019 page 732

Imperceptible Data Poisoning Attacks during Inference



Key Insights:

- Instead of training data samples, the probability vector at the output can be used to optimize the attack noise.
- To ensure the imperceptibility, additional similarity and correlation checks are required

Practical Implications of Adversarial Perturbations

Introducing the Noise into the Camera Feed



Challenges

- Impractical to intrude Environment
 - e.g., How to introduce the imperceptible noise to real human face?
- Handling the impact of environmental Noise

Practical Implications of Adversarial Perturbations

Attack Model I: Attacker has the access to output of the Noise Filter.



Attack Model II: Attacker donot has the access to output of the Noise Filter.



FAdeML: Adversarial Perturbations without Filters



Adversarial Perturbations: Attack Model I



Adversarial Perturbations: Attack Model II



FAdeML: Adversarial Perturbations with Filters



Adversarial Perturbations with Filters: Attack Model II



CapsAttacks: Robust and Imperceptible Adv. Attacks



My Research Team and Collaborators

Post-Docs and PhDs



MS/BS Students



Collaborators







CARE <u>Computer</u> <u>A</u>rchitecture and <u>R</u>obust, <u>Energy-Efficient</u> <u>Tech</u>nologies Group

Thank you! Questions?

M. Shafique swahlah1979@gmail.com





SAFETY & SECURITY ENGINEERING OF AUTOMOTIVE CPS

CPS&IoT'2019

Summer School on Cyber-Physical Systems and Internet-of-Things

Christoph Schmittner & Thomas Gruber





PRESENTER – CHRISTOPH SCHMITTNER

- Member of
 - Austrian Standards Institute (ASI)
 - Das Österreichische Elektrotechnische Komitee (OEK)
- ISO/TC 22 Road vehicles
 - Participation in the development of ISO26262:2018
 - Safety and Cybersecurity Topic Group
 - Coordination of the Austrian group in the development of ISO/SAE 21434
- Safety and Security engineering in automotive and industrial







PREAMBLE

- Tutorial requires basic understanding of
 - Automotive engineering
 - Safety
 - Security
- Methods & Process will be explained on a Use Case
 - We will try an interactive process
- Focus on analysis & system level



AGENDA





AIT – AUSTRIAN INSTITUTE OF TECHNOLOGY Overview, Dependable Systems Engineering Group





bmvit Austria's largest 1.370 **RTO** Centers employees System **Applied Research** Infrastructure Systems Competence Federation of **Austrian Industries** Subsidiary **Next Generation** Enterprises (through VFFI) Solutions LKR, NES, SL, Profactor 51% 162,9 **Tomorrow Today M** EUR total revenue



AIT AUSTRIAN INSTITUTE OF TECHNOLOGY

OWNERSHIP STRUCTURE 49.54 % **FEDERATION OF AUSTRIAN INDUSTRIES** (through VFFI)

50.46 %

REPUBLIC OF AUSTRIA (through the Federal Ministry for Transport, Innovation and Technology)

1.370

EMPLOYEES

162,9 M EUR

TOTAL REVENUES as of YE 2018

87,1 m EUR Contract research revenues (incl. grants) 50,4 m EUR bmvit funding **21,3 m EUR** Other operating income, incl. Nuclear Engineering Seibersdorf **4,1 m EUR** Profactor (51% of 8 m EUR)



Nuclear

Engineering

Seibersdorf

GmbH

AIT AUSTRIAN INSTITUTE OF TECHNOLOGY

AIT Austrian Institute of Technology

Seibersdorf Labor GmbH

Energy	Health &	Digital Safety &	Vision, Automation &
	Bioresources	Security	Control
Mobility Systems	Low-Emission	Technology	Innovation Systems &
	Transport	Experience	Policy



FOCUS OF THE CENTERS

Energy	Health &	Digital Safety &	Vision, Automation &
	Bioresources	Security	Control
 Sustainable Thermal Energy Systems Electric Energy Systems Sustainable Buildings and Cities Photovoltaic Systems 	 Bioresources Molecular Diagnostics Biomedical Systems Digital Health Information Systems 	 Security & Communication Technologies Visual Surveillance and Insight Smart Sensor Solutions Dependable Systems Engineering Information Management 	 High-Performance Image Processing Autonomous Systems Complex Dynamical Systems
Mobility	Low-Emission	Technology	Innovation Systems &
Systems	Transport	Experience	Policy
DEPENDABLE SYSTEMS ENGINEERING **RESILIENCE AGAINST FAULTS AND ATTACKS**







AQUAS - AGGREGATED QUALITY ASSURANCE FOR SYSTEMS

Overview





AQUAS PARTNER



Proceedings of CPS&IoT2019 page 755



MISSION STATEMENT

Co-Engineering into mainstream practices:

We are investigating Co-Engineering techniques for safety, security and performance of critical and complex embedded systems







SAFETY & SECURITY Starting Point





- Engineers
- Familiar with models, statistics, calculations
- Focus on failures and traits of materials and construction
- Risk analysis: FMEA et al

- InfoSec People
- Familiar with attacks, exploits, controls

- Focus on intentional, human opponents (and their automated tools)
- Risk analysis: Rule-of-thumb



• Risk analysis: FMEA et al

Fault Error Failure Hazard • Risk analysis: Rule-of-thumb





Mental Defense Model



- Predictable, unintentional events
- Targeted and static protection

Mental Defense Model



- Evovling, intentional attackers
- Enclosing and adaptive protection



- Remote hack of a level 2+ car
- Attacker gains control of braking system

- Brake system failure
- Hazards
 - Loss of control (of vehicle)
 - Risk of collision

- System compromise
 - Loss of control (IT system)
 - Risk of replacment costs



CHALLENGE – NETWORKED SYSTEMS

- Traditional View
 - Safety, Security Separate Issues.
- BUT
 - Safety cannot be guaranteed without security, and security can be influenced by safety requirements.

• Examples

. . .

- Hackers Remotely Kill a Jeep on the Highway With Me in It, wired.com, 21/07/2015
- Hacking a Tesla Model S: What we found and what we learned, blog.lookout.com, 07/08/2015
- Hackers Cut a Corvette's Brakes Via a Common Car Gadget, wired.com, 11/08/2015
- Hackers pop grease monkeys' laptops to disable Audi airbags, theregister.co.uk, 23/10/2015





STATE OF THE ART Automotive: Functional Safety & Cybersecurity





- ISO 26262
 - Road vehicles Functional Safety
 - First edition published in 2011
 - Second edition published in 2018

- SAE J3061
 - Cybersecurity Guidebook for Cyber-Physical Vehicle Systems
 - Published in 2016
 - Not longer available
- ISO/SAE 21434
 - Road Vehicles Cybersecurity Engineering
 - Currently in Development



WHAT IS SAFETY (IN AUTOMOTIVE)

- Passive Safety:
 - aim of passive safety mechanisms is to minimize the severity of that accident.
 - Examples: Seatbelts, crumple zones, etc.
- Active Safety:
 - aim of active safety mechanisms is to avoid accidents altogether
 - Examples: predictive emergency braking, Electronic Stability Control, etc.
- Functional Safety:

Focus of ISO 26262

- Aim to prevent accidents due to failures in electric / electronic / programmable electronic systems
 - Examples: Power supplies, sensors, communication networks, actuators, etc.
- Also other aspects of safety exist







 IEC
61508
 IEC
61508-
ED2
 ISO
26262
 ISO
26262
 ISO
26262-
ED2

 IEC
61508
 IEC
61508-
ED2
 ISO
26262-
ED2
 ISO
26262-
ED2

Proceedings of CPS&IoT2019 page 767

ISO 26262 EXTENSION OF SCOPE



	Classof vehicle	In scope?	Status
June 1	L1/L2	Excluded	
O SO	L3/L4/L5	In scope	Integration in Ed2
	L6/L7	Not defined	
	M1	In scope	Ed1
	M2/M3	In scope	Integration in Ed2
	N1/N2/N3	In scope	Integration in Ed2
	01/02/03	In scope	Integration in Ed2
	Other categories	Not defined	

20/05/2019 Source: Internet

ISO 26262 OVERVIEW





• 2018





AUSTRIAN INSTITUTE

ISO 26262 OVERVIEW



Proceedings of CPS&IoT2019 page 770



- New part 11 & 12
- Other Changes
 - Shifting of contents
 - Rework of specific sections

27

 Adapted / enhanced guidance

ISO 26262 **CYBERSECURITY**

Guidance on the interaction of safety with cybersecurity ۲







WHAT IS SECURITY (IN AUTOMOTIVE)

- Physical Security:
 - safeguards to deny access to unauthorized persons from physically access
 - Examples: locked door, badge access controls
- Information / Cyber Security:

Focus

- protection of an IT-system from the attack or damage to its hardware, software or information, as well as from disruption or misdirection of the services it provides
 - Examples: tamper protection, encryption, authentication, firewall
- Also other aspects of security exist

AUTOMOTIVE CYBERSECURITY HISTORY







Proceedings of CPS&IoT2019 page 773

AUTOMOTIVE CYBERSECURITY REGULATION



- UNECE (United Nations Economic Commission for Europe)
 - World Forum for Harmonization of Vehicle Regulations (WP.29)
 - 62 States follow UNECE Type Approval regulation



AUTOMOTIVE CYBERSECURITY REGULATION



• UN Task Force on Cyber Security and Over the Air



20/05/2019

AUTOMOTIVE CYBERSECURITY REGULATION



• Recommendation on Cyber Security



20/05/2019

Proceedings of CPS&IoT2019 page 776

AUTOMOTIVE CYBERSECURITY REGULATION ⇔ STANDARDS



- Process can be based on SAE J3061, ISO/SAE 21434 or other guidance
 - Need to fulfill requirements for cybersecurity management
 - => compliant process
- Operation requires approval of vehicle type cybersecurity
 - => approved cybersecurity case



CASE STUDY Methodology & Application



20/05/2019

CASE STUDY STEPS





- Case Study presents only selected steps
- Focus on risk management for safety & security
- Implementation of mitigation measures and testing is out of scope



USE CASE

- Steering column lock with keyless go
 - Steering column should be locked to protect unauthorized vehicle operation
 - Steering column lock should be deactivated if the key is in the vehicle and start button is pressed



SAHARA



- Identification of
 - Potential Hazards (with HARA)
 - Potential Threats (with STRIDE)
- Evaluation of Threats based on
 - Required Resources
 - Available Know-How
 - Impact of a successful attack



SAHARA

• Application in the Tutorial



ARCHITECTURE



- Electronic Steering Column Lock
 - receives key signal from Gateway
 - Receives information for wheel speed from ABS



FMVEA



- Combination of
 - Failure Mode and Effect analysis
 - Threat Modeling



FMVEA

• Application in the Tutorial



DESIGN

• Application in the Tutorial



THANK YOU!

Christoph Schmittner, June 14, 2019





STRIDE

Threat	Desired property
Spoofing	Authenticity
Tampering	Integrity
Repudiation	Non-repudiability
Information disclosure	Confidentiality
Denial of Service	Availability
Elevation of Privilege	Authorization


THREATS TO VEHICLES

- Spoofing of messages or data received by the vehicle;
- Communication channels used to conduct unauthorized manipulation, deletion or other amendments to vehicle held code/data;
- Communication channels permit untrusted/unreliable messages to be accepted or are vulnerable to session hijacking/replay attacks;
- Viruses embedded in communication media are able to infect vehicle systems;
- Messages received by the vehicle (for example X2V or diagnostic messages), or transmitted within it, contain malicious content;
- Information can be readily disclosed. For example through eavesdropping on communications or through allowing unauthorized access to sensitive files or folders;
- Denial of service attacks via communication channels to disrupt vehicle functions;
- An unprivileged user is able to gain privileged access to vehicle systems;
- Misuse or compromise of update procedures;
- It is possible to deny legitimate updates;
- Misconfiguration of equipment or systems by legitimate actor, e.g. owner or maintenance community;
- Legitimate actors are able to take actions that would unwittingly facilitate a cyber-attack;
- Manipulation of the connectivity of vehicle functions enables a cyber-attack, this can include telematics; systems that permit remote operations; and systems using short range wireless communications;
- Hosted 3rd party software, e.g. entertainment applications, used as a means to attack vehicle systems;
- Devices connected to external interfaces e.g. USB ports, OBD port, used as a means to attack vehicle systems.

POTENTIAL TARGETS OF, OR MOTIVATIONS FOR, AN ATTACK:



- Extraction of vehicle data/code;
- Manipulation of vehicle data/code;
- Erasure of data/code;
- Introduction of malware;
- Introduction of new software or overwrite existing software;
- Disruption of systems or operations;
- Manipulation of vehicle parameters.

POTENTIAL VULNERABILITIES THAT COULD BE EXPLOITED IF NOT SUFFICIENTLY PROTECTED OR HARDENED:



- Encryption methods can be compromised or are insufficiently applied;
- Parts or supplies could be compromised to permit vehicles to be attacked;
- Software or hardware development permits vulnerabilities;
- Network design introduces vulnerabilities;
- Physical loss of data can occur;
- Unintended transfer of data can occur;
- Physical manipulation of systems can enable an attack.



POSSIBLE MITIGATIONS

- The principle of security by design shall be adopted to minimise the impact of an attack on the vehicle ecosystem
- Access control techniques and designs shall be applied to protect system data/code
- Through system design and access control it should not be possible for unauthorized personnel to access personal or system critical data
- · Measures to prevent and detect unauthorized access shall be employed
- The vehicle shall verify the authenticity and integrity of messages it receives
- Security controls shall be implemented for storing cryptographic keys
- Confidential data transmitted to or from the vehicle shall be protected
- Measures to detect and recover from a denial of service attack should be considered
- Measures to protect systems against embedded viruses/malware should be considered
- Measures to detect malicious internal messages or activity should be considered
- Secure software update procedures shall be employed
- Measures shall be implemented for defining and controlling maintenance procedures
- Measures shall be implemented for defining and controlling user roles and access privileges, based on the principle of least access privilege
- Organizations shall ensure security procedures are defined and followed
- Security controls shall be applied to systems that have remote access
- Software shall be security assessed, authenticated and integrity protected
- Security controls shall be applied to external interfaces
- Cybersecurity best practices for software and hardware development shall be followed
- Data protection best practices shall be followed for storing private and sensitive data
- Systems should be designed to respond appropriately if an attack on a vehicle is detected.



Probabilistic modelling integrating concerns of safety, security, etc.

Lorenzo Strigini and Peter Popov Strigini@csr.city.ac.uk, P.T.Popov@city.ac.uk Centre for Software Reliability City, University of London, U.K.

Preliminary slides for CPS&IoT'2019 Summer School on Cyber-Physical Systems and Internet-of-Things Budva, Montenegro, June 10-14, 2019

CSR Building confidence in a computerised world www.csr.city.ac.uk

Proceedings of CPS&IoT2019 page 793



Structure of the seminar

This seminar will address three topics:

- 1. the need of probabilistic reasoning for security
- 2. an application example for simple combinatorial models
- 3. an application example for state-based models
- in contexts where security affects safety



Part1: probabilities and security

- premise: probabilistic reasoning is popular for reasoning about system reliability, safety but much less so about security
- *thesis*: this is an artificial, erroneous, detrimental divide between technical communities:
 - probabilities are the correct language for *specifying* many important properties in security
 - the difficulties of *reasoning with* probabilities to support decisions are not qualitatively different between security and other fields
- ... similar to the divide previosuly claimed between "hardware reliability" and "software reliability"

Why do we need probabilities?

what does one mean by

"my system A installation is more secure than my system B installation, in environment X"?

 do we mean some simple deterministic ordering, like: "the attacks that can defeat A are a proper subset of those that can defeat B"?

 or rather something less restrictive like "A will be penetrated less often than B"

B

(although its vulnerabilities may not be a proper subset of B's)?



- or perhaps "less total damage happens on A than on B", etc

Owners need measures of the *effects* **of vulnerabilities**

e.g.

- how long before the next attack requiring interruption of service?
- how much damage/expense due to intrusion over a year?

Such qualities of "real" interest are subject to uncertainty: they "naturally" require quantitative statistical/probabilistic language

- "more secure so far": measurement and statistics
- "more secure for the future": probability

Probabilities for decision making

in design:

is this hypothetical design better than that?

- in acceptance, integration: is this implemented system good enough? How good?
- questions that require *both* deterministic *and* probabilistic reasoning

an example from the history of "replicated subsystems with voting" redundancy:

Deterministic: "to outvote one faulty/crooked subsystem you need a group of *four* individuals and many message rounds (if certain non-obvious behaviours are possible)" Probabilistic: "and this setup may be *less reliable* than simpler, 3-subsystem setup (if rare failure behaviours rare enough, failure rates high enough, messages take long enough)"

Why not probabilities?

Common objection:

- probabilities work well for physical hardware failures because they are random events which can be observed in great numbers
- [while software-caused failures are deterministic, *systematic* failures]
- [and security failures are also *intentional* and *rare*]
- hence probabilitic reasoning is useless for {software, security, ...}

No!

confusion between *causal mechanism* and *process*

we don't know when the next failure/intrusion will occur... even if God or the attacker knows

Two objections: determinism and intentionality

- "a software flaw means that the software will deterministically fail given the "appropriate" inputs" where is the randomness?
- for system owners, failures are a random process, because they do not know:
 - the values of the "appropriate" inputs (the faults/vulnerabilities)
 - when these will be presented to the system by the external world

- "the attacker will attack when he pleases, and adapting attacks as vulnerabilities/defences evolve" where is the randomness?
- for the defender this is a random process
 - perhaps a complex one, of course
 - more or less easy to describe so as to give good predictions

So... yes, there are difficulties with probabilities

... in security, *just as* in reliability and safety. For *some* problems, *some* probabilistic approaches prove useful

Yet,

- decisions must be (and *are*) taken under uncertainty
- probability calculus describes reasoning under uncertainty in formal, auditable fashion
 - explicit assumptions
 - consistency checks
 - no magical increase of factual knowledge
- are there any advantages in not making one's reasoning explicit?

"Yes, probabilities are necessary... but unusable: how do I estimate the probabilistic parameters?"

- a very legitimate concern
- yet, no reason for avoiding probabilities a priori
- the difficulties are similar to those in *reliability* assessment (their *magnitude* varies with the circumstances)
 - extrapolating from one environment to another one, from the past to the future
 - estimating probabilities of one-off events
 - unpredictable individuals (people, chips), each one different from all others
 - using population statistics to estimate probabilities for individuals
- yet useful results are obtained for reliability, although
 - probability estimates will prove more or less accurate predictors depending on circumnstances
 - + cf opinion polls, predicting election results ...
 - and for one-off events cannot be verified empirically

In conclusion ...

- probabilities are a common language for all "dependability" -incl. security-(or all "security" - incl dependability) properties
- trusting the formalism more would *help* reasoning about [some] security problems
- and allow a more unified view of both techniques and decision issues
- [just as trusting the *numbers* less would help reasoning about reliability/safety problems]
- How useful they will be on a specific problem? It's a matter of trying





Part 2: insight from a simple model of diversity for security

Proceedings of CPS&IoT2019 page 804



Diversity, Safety and Security in Embedded Systems: modelling adversary effort and supply chain risks

Ilir Gashi, Andrey Povyakalo, Lorenzo Strigini Centre for Software Reliability City University London, U.K.

funded by EU ARTEMIS SeSaMo (Security and Safety Modelling) U.K. EPSRC D3S "Diversity and Defence in Depth for Security" *with thanks to* Martin Matschnig, Thomas Hinterstoisser, Bernhard Fischer, Peter Ryan



www.csr.city.ac.uk

Proceedings of CPS&IoT2019 page 805

Motivations

ever-growing security concerns for embedded, safety critical systems

 most security concerns are about *design faults* surrounded by great *uncertainty*

- design trade-offs arise
- so, designers need quantitative, *probabilistic* reasoning
 - though many claim that this is of little use regarding security; we have long argued the opposite ...
 - here we show new examples of probabilistic reasoning for *insight* into design decisions
 - motivated by a practical problem





The example: industrial drive control

- inspired by a "use case" in the SeSaMo project <u>http://sesamo-project.eu/content/industrial-drive</u>
- electric motor under computer control
 - generic control unit for motors that could drive a variety of loads



The example: industrial drive control

- electric motor under computer control
 - generic control unit for motors that could drive a variety of loads



- attackers may want to perturb motor operation
 - through physical access to *communication*
- communications are replicated and encrypted For security, should this be pure replication or diversity? What kind of diversity, and what gain will it achieve?

Details, threats, uncertainties

- communication is replicated (triple, voted) for reliability, safety
- and encrypted because attackers may want to
 - decipher signals to/from the controller
 - + by reading any one channel, they can steal secrets and/or engineer better attacks (violation of confidentiality)
 - craft and insert forged signals
 - + by "highjacking" a majority of control channels, they can cause accident/loss (violation of integrity)

... is *identical* encryption on all channels sufficient?

Why diversity?

- ... is *identical* encryption on all channels enough?

- state-of-the-art encryption ensures that brute force attacks need a very long time to succeed
- but a great concern is faults in the "implementation" of crypto systems
 - in hardware, software, operation, management, ...
 - replication of channels will replicate the flaws
 - and the cryptosystem is hardwired: adversaries can study it for years, and then strike
- obvious remedy: diversity between the replicated communication channels
- But there are questions:
 - how much will it help? (is it worth doing?)
 - how will improving integrity harm confidentiality?

We study two scenarios of "cryptography implementation flaws"

• "affordable cryptanalysis"

• shortcut to penetration through "supply-chain flaw"

Scenario 1: "Affordable cryptanalysis"

- adversaries only need to search a *reduced* key space
 - offline, over days or years, trying one key at a time
 - so that some "reasonable" chance of breaking the cipher is affordable, even though possibly expensive
- design question: how useful is it to use diverse keys on the redundant communication channels?
 - given that the adversary has a large but finite effort budget

other details:

•we analyse a design in which the same cipher is used for both confidentiality and authentication – we considered others

- •the keys are still equiprobable within the reduced space, and assigned independently
- finding the key for one channel does not change the effort required for the next key
 attackers try the most efficient attack strategy given these conditions and their own budget of effort
 - in this case, budget of computing cycles for breaking ciphers
- •the defender does not detect attack or respond

Scenario 1: probability of attack success

.. an attacker spending a certain amount of effort (keys tried) to find a single key gets what probability of success (violation)?



Scenario 1: effect of using diverse keys

- .. an attacker spending a certain amount of effort (keys tried) gets what probability of success?
 - for confidentiality violation: same as with identical keys
 - for integrity violation:



Scenario 1: how much do I gain by using diverse keys?

... how much more effort do attackers need to attain a given probability of integrity violation (i.e., of breaking 2 channels)?



Proceedings of CPS&IoT2019 page 815

Scenario 1: summary observations

- diversifying keys has no effect on risk to confidentiality so no trade-offs
- for integrity

-probability of attack success varies with square of effort (fraction of reduced key space searched)

–if the crypto is "reasonably" strong
 (even though weaker than in theory – reduced key space!)
 diverse keys add substantial protection

-if it's very weak, they add much less

–as a designer, if I trust that the crypto on one channel is "reasonably" strong

+ diverse keys are worth using!

-the adversary's viewpoint matters:

if my system is worth attacking even if attack budget offers low probability of success, diverse keys are good protection

details vary with the architecture and the controlled system

- they affect best attack strategy and probability of attacker success
- see paper

Scenario 2: "Supply chain flaws"

- the cryptosystem that I buy may have a *fatal flaw*, such that an attacker can read/forge messages with little effort
 - e.g. some kind of intentional "backdoor"
 - or keys leaked directly from the vendor organisation to attackers
 - unintentional flaw (say use of obsolete component), discovered/used by adversary.
 - … [see paper]
- **design question**: how useful is it to use three *diverse implementations* for the three redundant channels?

if the market offers *n* possible versions of the cryptosystem, *k* of which have some fatal flaw available to adversaries

– and won't diversifying *increase* the probability of buying a flawed implementation – how to handle the trade-offs?

other details:

- diversify vendors, designs, even ciphers, ... to match feared threats
- each of the *n* looks "as good as the others"
- designers choose one at random for each replicated channel
- each has probability q=k/n of being flawed

Scenario 2: effects of using 3 diverse implementations

on probability of the assembling system having *integrity* or *confidentiality* **system** *flaws*



Proceedings of CPS&IoT2019 page 818

Scenario 2: effects of using 3 diverse implementations

on integrity and on confidentiality system flaws



Scenario 2: effects of using 3 diverse implementations



Scenario 2 : the trade-off

if diversity buys integrity at a cost in confidentiality...

... under which circumstances is it the right solution?

- consider the two values of *expected loss*
 - from having a system that is "integrity-flawed" (adversary can highjack control and cause accidents) vs
 - from having a system that is "confidentiality-flawed" (adversary can steal message content, learn design secrets, devise more devious attacks..)
 - both affected by the specific physical system driven, the probability of adversaries deciding to attack, their preferred goals,...
- call these L_l and L_c ...

Scenario 2 : the trade-off: diversity improves ...

... expected loss if, for the given q, L_l is at least *this much* larger than L_c



Proceedings of CPS&IoT2019 page 822

Scenario 2 : summary observations

- If many of the available versions are likely to be "flawed", I should *not* choose diversity
 - better to put all my eggs in one basket
- if this scenario is unlikely at first deployment, it may be likely in the long run..
 - I may want to shorten my warranty
 - suggest massive hardening of physical security after a few years
 - or redesign radically to allow secure updates of crypto..
- but if (or as long as) k is quite small
 - (that is, small probability *q* of a randomly chosen version being flawed)
 - it is appropriate to choose diversity
 - + that is, the expected reduction in risk through integrity breaches compensates for the expected increase in risk through confidentiality breaches ...

... if the expected loss L_l from an "integrity flaw" (2 flawed channels) exceeds *twice* the expected loss L_c from a "confidentiality flaw" (1 flawed channel)

+ for higher *k*, diversity is justified if this ratio is higher

Some general conclusions

A new, and we believe useful, way of modelling the problem

- quantitative trade-offs
 - for given budget of adversary effort
 - or prevalence of flawed implementations
- useful insight from simple modelling
 - what-if reasoning
 - input for game-theoretical view: what would the adversary do?
 - some solutions are similar to those for textbook reliability scenarios, but not quite the same
- simple approach, covering general classes of scenarios
 - e.g. attacks on safe shutdown ability of safety system
 - breaking into two user accounts

-
General conclusions - 2

A small step forward on a hard problem

future work:

- modelling other realistic, more complex, attack scenarios (the "easy" part)
- dependencies between successes on two channels
 - causal and epistemic
- estimating plausible model parameters from evidence
 - the full decision problem can be modelled by treating the parameters as random variables





Part 3: Example of state-based modelling

Proceedings of CPS&IoT2019 page 826





Stochastic modeling of safety and security of the e-Motor, an ASIL-D device

Peter Popov SAFECOMP 2015, Delft, The Netherlands Friday, 25th Sept. 2015

Proceedings of CPS&IoT2019 page 827





Outline

- Background of the problem domain
 - ISO 26262 ASIL-D concept
 - Dealing with epistemic uncertainty in probabilistic modelling of Adversary
- Motivation/Problem Statement
 - The E-motor Safe state and a Cyber attack on the *safe state*
- The Stochastic model (stochastic Activity Networks)
- Findings related to sensitivity analysis
- Conclusions and future work





Acknowledgement

- This work has been, in part, supported by th ARTEMIS Joint Undertaking, via the SESAMO project (grant agreement number 295354).
- http://sesamo-project.eu/documents











ISO 26262 Safety requirements for software based functions

- ISO 26262 recommends that *qualitative* analysis be applied to systematic faults.
 - Functions that enable the system to achieve or maintain a *safe state*;
- Functions related to the *detection, indication and handling* of faults of safety-related *hardware elements*;
- Functions related to the detection, notification and mitigation of faults in the *software itself*;
 - self-monitoring of the software in the *operating system* and
 - *application-specific self-monitoring* of the software to detect, indicate and handle systematic faults in the application software.
- Functions related to *on-board and off-board tests*;
 - On-board tests can be carried out by the system itself or through other systems within the vehicle network during *operation* and during the pre-run and post-run phase of the vehicle.
 - Off-board tests refer to the testing of the safety-related functions or properties during production or in service.
- Functions that allow *modifications* of the software during production and service; and
- Functions related to *performance* or *time-critical operations*.
 - _ ___ Used for attacks not discussed in the paper.





ASIL-D and software

- Safety analysis shall be carried out at the software *architectural level* in order to:
 - identify or confirm the safety-related parts of software support the specification and *verify the efficiency of the safety mechanisms*.
- Mechanisms for error detection

	Methods	А	В	С	D
1a	Range checks of input and output data	++	++	++	++
1b	Plausibility checks	+	+	+	++
1c	Detection of data errors	+	+	+	+
1d	External monitoring facility	0	+	+	++
1e	Control flow monitoring	0	+	++	++
1f	Diverse software design	0	0	+ (++





- This is an **AUTOSAR software module** for controlling an **electric motor** meant to be integrated in a variety of potential car systems, such as:
- *Electric Power Steering*: support the steering actuation by superposing torque
 - power steering system has to fail **silently. A silent failure means that the shaft of** the electric motor runs freely and its presence cannot be recognized by the working environment any longer as soon as a failure occurs. This is important since a faulted electric motor can deter the driver from direct mechanical steering.
- Dynamic S speed and What if the safe state dynam

ing on vehicle

SR Building confidence in

www.csr.city.ac.uk

run fre is altered by an attacker?

d rather than er's steering

- Steer-by-Wire: steering actuation solely without mechanical link
 - Steer-by-Wire system must neither fail safely nor silently. In any case, it has to continue operating after fault occurrence at least *in a degraded mode*, which means a fail operational ability. In other words, electrical drives for Steer-by-Wire systems have to be fault tolerant, at least against single point failures.

In other words the E-motor will have several safe states, one of which will be "active" at any point in time.

25/09/2015

comma



An attacker can eliminate the safe state







Probabilistic modelling of Adversary

• *Issue 1:* Level of abstraction is important. The known probabilistic models of an Adversary range between:

The paper advocates the "Cyber-physical perspective", i.e. the system model be *sufficiently detailed* to capture adequately the impact of a successful attack in a specific system context.

• This makes it difficult for the results of the analysis to be communicated to the stakeholders in a specific context.

TOTTIGHT





- *Issue 2*: Epistemic uncertainty
 - The "standard way" of dealing with this issue in safety and reliability is Bayesian analysis: one would define one's belief about the uncertain model parameters and let the

The paper advocates *sensitivity analysis* on the range of "plausible" values.

 for large models this may be difficult as the space of plausible values is likely to be very large.

be "inaccurate"

• The likelihood of attacks (i.e. Adversary profile) are likely to be changing continuously, hence the posterior at any point of time is likely to remain a poor reflection of the likelihood of attacks.

Stochastic Activity Network (SAN) model

•



In case somebody is interested in "playing" with the model (and probably finding bugs ☺) it is available at:

http://openaccess.city.ac.uk/1 1980/

25/09/2015

Stochastic modeling of safety and security of the e-Motor, SAFECOMP 2015

Stochastic Activity Network (SAN) model



in

SAN model (2): SW channel model



25/09/2015





SAN model (3): Adjudicator





SAN model (4): Safe State Configuration







Measure of interest

- Time to system failure (safe or unsafe) is the measure of interest.
- System failure states are "absorbing", i.e. if the model reaches a failure state it will remain in this state forever.
- At any point of time the model will be in one of the three possible system states:
 - ОК
 - Safe failure
 - Unsafe failure
- The system is judges by the adjudicator as OK or failed *safely* or *unsafely*.
- The model is solved via simulation.
 - Many simulations (2,000,000+) were used to measure the time to system failure.
- The simulation campaigns were *long enough*, so that at the end of the campaign the system fails with a probability of 1 (or nearly 1).
- For any length of simulation campaigns, X ("mission duration"), the probability is computed that the system reaches a failure state (safe or unsafe) at the end of the mission.





Findings: An interesting observation

Global variable name		Exp 2	Exp 3	Exp 4	
AttackRate [hour ⁻¹]		0	.001		
CC_failure_rate [hour ⁻¹]		1.00E-04			
Config_repair_success			0.6		
Configuration_validation_Duration [hour]			0.1		
SS_repair_rate [hour ⁻¹]	36	360	3.6	36	
USF_repair_rate [hour ^{_1}]		36	36	360	
attack_CH1_success_pr		1	0.2		
attack_CH2_success_pr			0.1		
attack_count			10		
channel_failure_rate [hour-1]		0	.001		
failure_coverage			0.8		

Stochastic modeling of safety and security of the e-Motor, SAFECOMP 2015



Findings: Sensitivity Analysis (2)

		Probability of mission Success/Safe failure/Unsafe failure			s/Safe
Mission duration [hours]		Exp 1	Exp 2	Exp 3	Exp 4
1000	Success	0.999120	0.999245	0.999660	0.999175
	Safe failure	0.000880	0.000755	0.000340	0.000825
	Unsafe failure	0 (0	0 (0
7000	Success	0.994180	0.995060	0.008180	0.995015
	Safe failure	0.005815	0.004940	0.991820	0.004985
	Unsafe failure	5.00E-06	0	0	0
8000	Success	0.993410	0.019340	0.007930	0.019285
	Safe failure	0.006585	0.980660	0.992070	0.980715
	Unsafe failure	5.00E-06	0	0	0
11,000	Success	0.991455	0.017560	0.007290	0.017650
	Safe failure	0.008540	0.982440	0.992710	0.982350
	Unsafe failure	5.00E-06	0	0	0
12,000	Success	0.020310	0.017005	0.007100	0.017070
	Safe failure	0.979685	0.982995	0.992900	0.982930
	Unsafe failure	5.00E-06	0	0	0

25/09/2015

Stochastic modeling of safety and security of the e-Motor, SAFECOMP 2015





Discussion

- The paper advocates that a probabilistic model should be *detailed enough* to account for *all hazards* identified in safety analysis
 - Impact of attacks must be spelled out in the *specific* system context.
- The epistemic uncertainty related to model parameters (e.g. in deciding the values of the parameters related to the attacks) should be addressed by sensitivity analysis exploring the space of *plausible values*.





Future Work

- The model of the e-Motor already includes 3 types of attacks (identified by the safety analysis) and allows for different Adversary profiles to be used. In this paper only the "most interesting" of the three attacks is analysed.
 - Quantifying the impact of the other two attack types related to data integrity (torque request attacks and attack on the channels control loop parameters) will be dealt with in the future.
- In the particular context of AUTOSAR an interesting design trade-off exists between protecting the individual devices against cyber attacks specific to the device vs. using a generic intrusion detection/protection system (IDS).
 - Probabilistic modeling seems particularly suitable for addressing this problem and the trade-off will be studied in the future.
- The model of accidental dependent failures is simplistic, possibly unrealistic.
 - In the future this model of dependent accidental failures will be replaced with better alternatives;



Une école de l'IMT

Practical Embedded Systems Modeling and Safety, Security and Performance Verification with TTool

Dominique Blouin, Maysam Zoor and Ludovic Apvrille

{firstname.lastname}@telecomparistech.fr



CPS&IoT Summer School 2019, Budva, Montenegro

Outline

Installation of Tools

Introduction

Model-based Engineering The TTool Toolkit

Designing a Simple Smart Card with SysML-Sec

A Simple Smart Card System Modeling the Smart Card Application Safety Verifications of the Behavior of the Smart Card Specifying Hardware Architectures for the Smart Card Mapping the Smart Card Application to Hardware Architectures

Simulating and Verifying the Smart Card System Securing the Smart Card System with SysML-Sec

Going Further

References

Proceedings of CPS&IoT2019 page 847

Installation from Provided Virtual Machine (recommended for this tutorial)

- Download and install VirtualBox for your operating system from https://www.virtualbox.org/
- Copy the TTool virtual machine file from the provided USB stick
- Start VirtualBox and import the provided TTool virtual machine:
 - Select menu File » Import Appliance
 - Browse to the downloaded SysML-Sec_Tutorial.ova file
 - Follow the wizard using the default settings
 - Once the import is finished, start the machine (password = sysmlsec)
 - Note that it may happen that the machine hangs while starting. If so then just restart it from VirtualBox
- Double-click the TTool icon on the desktop to launch TTool

SysML-Sec

Installation from Official Releases

- Graphical installer or archive
- Follow the instructions from the TTool website:
 - https://ttool.telecom-paristech.fr/ installation.html
- Note that this may require installing companion tools...

Slides of this Tutorial

Open the SysML-Sec_Tutorial.pdf file located on the desktop

6.000-

Installation of Tools Introduction Designing a Simple Smart Card with SysML-Sec Going Fi

Increasing Systems Complexity

Number of embedded Software Lines of Code (SLOC) in Aircrafts has doubled every 4 years!



Estimated Onboard SLOC Growth

Airbus data source: J.P. Potocki de Montalk, Computer Software in Civil Aircraft, 6th Annual Conference on Software Assurance, (COMPASS 1991) Boeing data source: J.J. Chilenski, 2009.

6.000-

Installation of Tools Introduction Designing a Simple Smart Card with SysML-Sec Going Fi

Impacts of Late Faults Discovery



Source: Feiler et al., 2010

7/77

From Embedded to Cyber-Physical Systems

Embedded systems

		1100000		ParisTech
8/77	June 2019	Institut Mines-Telecom	SýsML-Sec	- % 10

Proceedings of CPS&IoT2010 page 853

\$ 535mm

From Embedded to Cyber-Physical Systems



8/77	June 2019	Institut Mines-Telecom	SvsML-Sec	ParisTech
0/11	June 2019	montal wines-relecom	Sysme-Sec	- Sait

Proceedings of CPS&IoT2019 page 854

6 535---

From Embedded to Cyber-Physical Systems



		11000000		Purchalingh
8/77	June 2019	Institut Mines-Telecom	SýsML-Sec	- 察職

Proceedings of CPS&IoT2010 page 855

\$ 535mm

From Embedded to Cyber-Physical Systems



8/77	June 2019	Institut Mines-Telecom	SysML-Sec	Parister
				2218117

Proceedings of CPS&IoT2010 page 856

6.000-

From Embedded to Cyber-Physical Systems



			mgs of CL Stero 12019 page 037	ParisTech
8/77	June 2019	Institut Mines-Telecom	SysmL-Sec	18課

Dessedings of CDC & LaT2010 mags 957

6 535---

Installation of Tools Introduction Designing a Simple Smart Card with SysML-Sec Going Fi

From Embedded to Cyber-Physical Systems



		Proceed	ings of CPS&IoT2019 page 858
8/77	June 2019	Institut Mines-Telecom	SysML-Sec





Installation of Tools
Introduction
Designing a Simple Smart Card with SysML-Sec
Going Fi

000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
<td

Definitions

Safety

Se-

curity



The practice of defending computers, servers, mobile devices, electronic systems, networks, and data from malicious attacks [4]

Performance

Ability to accomplish a specific task and respond to events within a specified time [1] [2]

Proceedings of CPS&IoT2019 page 860
Examples of Threats

Transport Systems

- Use of exploits in Flight Management System (FMS) to control ADS-B/ACARS [Teso 2013]
- Remote control of a car through Wifi [Miller 2015] [Tencent 2017]

Medical Appliances

Infusion pump vulnerability, April 2015. http://www.scip.ch/en/ ?vuldb.75158



Wired - ABC News



Hospira

Vulnerability Identification

Investigations

- Testing ports (JTAG interface, UART, ...)
- Firmware analysis
- Memory dump
- Side-channel analysis (e.g. power consumption, electromagnetic waves)
- Fault injection

Secure your systems!

Design with security in mind from the very beginning

Interdependency between Properties

Impact of Security on Safety: To have a safe system, security must be guaranteed. Safety can be compromised by cyber-attacks

 Security
 Safety

Installation of Tools Introduction Designing a Simple Smart Card with SysML-Sec Going Fi

Interdependency between Properties

Impact of Security on Safety: To have a safe system, security must be guaranteed. Safety can be compromised by cyber-attacks
 Security
 Safety
 Impact of Security on Performance: security measures require extra processing time which might degrade system performance
 Security
 Perf.

Interdependency between Properties



Modeling is not really a new Technique...

... and it is not limited to Software!



Proceedings of CPS&IoT2019 page 866



Installation of Tools Introduction Designing a Simple Smart Card with SysML-Sec Going Fi

Modeling is not Really a New Technique...

"If you fail to plan, you are planning to fail!"

SvsML-Sec

Painting by Duplessis. Source: Wikipedia



Installation of Tools Introduction Designing a Simple Smart Card with SysML-Sec Going Fr

Model-Based Systems Engineering

 Systems engineering: a multidisciplinary approach to develop a balanced system solution in response to stakeholder needs

Model-Based Systems Engineering

- Systems engineering: a multidisciplinary approach to develop a balanced system solution in response to stakeholder needs
- Systems engineering can be performed by using a document-based or model-based approach or combined approach

Model-Based Systems Engineering

- Systems engineering: a multidisciplinary approach to develop a balanced system solution in response to stakeholder needs
- Systems engineering can be performed by using a document-based or model-based approach or combined approach



Figure: Document-Based Figure: Model-Based

Model-Based Systems Engineering

- Systems engineering: a multidisciplinary approach to develop a balanced system solution in response to stakeholder needs
- Systems engineering can be performed by using a document-based or model-based approach or combined approach



Model-Based Systems Engineering

- Systems engineering: a multidisciplinary approach to develop a balanced system solution in response to stakeholder needs
- Systems engineering can be performed by using a document-based or model-based approach or combined approach



Model-Based Systems Engineering

- Systems engineering: a multidisciplinary approach to develop a balanced system solution in response to stakeholder needs
- Systems engineering can be performed by using a document-based or model-based approach or combined approach



Model-Based Systems Engineering

- Systems engineering: a multidisciplinary approach to develop a balanced system solution in response to stakeholder needs
- Systems engineering can be performed by using a document-based or model-based approach or combined approach



Proceedings of CPS&IoT2019 page 874

The TTool Toolkit

Started in 2004

- PhD thesis of Ludovic APVRILLE
- Team of about 10 developers (researchers, PhD students, engineers...)
- Several research projects and grants: Texas Instruments, Freescale, FP7 Evita, VEDECOM, H2020 AQUAS, Nokia, ...
- Several academic and industry users: ISAE, Nokia, Trusport, Siemens, Thalès, Continental, ...



SysML-Sec Method



TELECO

SysML-Sec Method



6 000---

TELEC

SysML-Sec Method



TELECO

SysML-Sec Method



TELECO

SysML-Sec Method



Simple Smart Card System

Main functions:

- Authenticate with the terminal
- Receive data from network
- Process data with a specific application (e.g.: process banking transactions)
- Send data to network

Requirements:

 Security: The data transferred between the terminal and the smart card shall be confidential



^{► ...}



TTeel		magneti bab			п		10:12 💠
0	File Edit V&V Code Generation View Tool Help	· 新 新 州 人 东东东 相相是 建度 3 3	2 7	115	8	6	8.8
	Create a new	Button tab					
	Ready						

Proceedings of CPS&IoT2019 page 882

6 535----

Hands-on: Create a SysML-Sec Methodology Diagram

TTool: /home/dblouin/temp	/smart_card.ttool/smart_card.xml	• • = ×
File Edit V&V Code Generation View Tool	Help	
♥ Tra ∰ Syntax analysis ∰ R. Graphs ⊨ m mariants Search result	Move to the left Move to the right Rename Remove Sort Clone New Network Calculus Design New DIPLODOCUS Methodology New AVATAR Methodology New SystML-Sec Methodology New Partitioning - Functional view New Partitioning - Communication Pattern	
	New Partitioning - Architecture and Mapping New SystemC-AMS Block Diagram	
	New Modeling Assumptions Diagram New Requirement Diagrams New Attack Tree New Fault Tree New Analysis	

SysML-Sec Methodology Diagram





Hands-on: Create an Application Diagram

TT 🔷	ool: /home/dblouin/temp/smart_c	ard.ttool/smart_card.xml	0 - EL		
File Edit V&V Code Genera	tion View Tool Help				
R, R, 66 68 28 48 18	X & B B + B D	C 4 100% 4 5 91 X X	A. A. +		
TTool: /home/dblauin/temp/s SyaMLSec_Methodolegy (! Syntax analysis W R. Graphs Minor analysis W M. Search result	SysMLSec_Methodology				
	SysMLSec_Methodology	Move to the left Move to the right			
	DY	Rename	-		
		Remove	-		
		Sort			
		Clone			
		New Network Calculus Design			
		New AVATAR Methodology			
		New SysML-Sec Methodology			
		New Pactitioning - Functional view			
		New Partitioning - Communication Pattern New Partitioning - Architecture and Mapping New SystemC-AMS Block Diagram			
		New Modeling Assumptions Diagram			
	Assumptions	New Requirement Diagrams			
		New Fault Tree			
		New Analysis			
		New Design			
		Attacks 🔩			
	A				

Hands-on: Create Application Components

- Create a primitive component for the smart card terminal
- Create a composite component for the smart card system
- Inside the smart card system:

23/77

- Create a primitive component for the communication interface
- Create a composite component for the TCP/IP module
- Create a primitive component for the smart card specific application



Communication Mechanisms

- Requests: used to spawn the execution of primitive components (operation calls)
- Events: used to synchronize control-flows
- Channels: used to exchange data
 - Blocking Read / Non Blocking Write: the emitter can write infinite times while the receiver task is blocked when attempting to read from an empty channel. Data are read in a FIFO with an infinite capacity
 - Non Blocking Read Non Blocking Write: the emitter can write an infinite number of times and the receiver is never blocked when attempting to read an empty channel. Equivalent to shared memory of infinite size
 - Blocking Read Blocking Write: Emitter blocked when attempting to write to a full channel and receiver blocked when attempting to read from an empty one. Equivalent to a finite FIFO buffer

Hands-on: Specify the communication between the Smart Card and the Terminal

- Create the appropriate ports to model each communication between the *smart card terminal* and the *communication interface* primitive components
- Connect the created ports





Task's Activity Diagrams

- Specialized UML activity diagrams
- Describe sequences of actions performed by the component of the diagram
 - Send / receive events to / from components
 - Send / receive requests (incl. parameters) to / from components
 - Send / receive data to / from components
 - Computation actions (exec i, exec c, etc.)
- Control flow nodes (conditional branching)





Hands-on: Describe the Behavior of the Smart Card Terminal



Create a send request action for activating the smart card

- Create a send event action to reset the smart card
- Create a wait for event action to receive a notification from the smart card when it has reset

SvsML-Sec



Hands-on: Describe the Behavior of the Smart Card Terminal (cont'd)

Model a non-deterministic behavior for sending or receiving data where half of the time data is received and sent for the other half:

- Create a non deterministic choice action
- On one branch:
 - Create a write channel action to send some data to the smart card
 - Create a send event action to notify the smart card that some data has been sent

Hands-on: Describe the Behavior of the Smart Card Terminal (cont'd)

On another branch:

- Create a notified event action to be notified that the smart card terminal has sent some data. In this action, assign the received value to variable x
- Create a *choice* action to determine what to do depending on the value of x
- On one branch, create a *condition* for when x = 0
- Add a stop action to end the flow of that branch
- On another branch, create a *condition* for when x > 0
- Following that branch, create a wait for event action to receive a notification from the smart card that it has sent some data
- Create a read in channel action to receive the sent data
- Add a stop action to end the flow

Verification of Behavior without Knowledge of the Hardware Architecture

- At this level of description, some analyses can already be performed without knowing the hardware execution platform
- For example, can we check that the terminal ever sends or receives data to or from the smart card?





Model Checking



Source: Lohmann 2013

		Proceed	lings of CPS&IoT2019 page 894	TELECOM	6 1755
31/77	June 2019	Institut Mines-Telecom	SysML-Sec	ParisTech	

Hands-on: Verifying Reachability with UPPAAL

- TTool uses UPPAAL to perform safety analyses (http://www.uppaal.org/)
- Completing the specification in order to perform such verification would take too much time so a model is provided
- Open the model provided for this tutorial
- Open the smart card terminal activity diagram
- Right-click the send data action and right click check Reachability / Liveness
- Note the RL? decoration added next to the action



Hands-on: Verifying reachability with UPPAAL (cont'd)

- Check the syntax of the application model by clicking the Syntax analysis button 4
- Launch safety verification by clicking the Safety verification button X
- From the dialog window that opens, ensure *Reachability of* selected states is checked

*	Formal Verificatio	n with UPPAAL	• • ×		
Verify with UPP Options of UPP Size of infinite	AAL: options AAL Specification FIFO =	1024			
No deadlocks	5?				
Reachability:	O None	Selected	🔾 All		
Liveness:	O None	Selected	IIA O 🖌		
Leads to:	None	Selected			
🗌 Generate sin	ulation trace				
Show verifica	ation details				
Select options and then, click on 'start' to start generation of RG Session id on launcher—1 Sending UPPAAL specification data Reachability of: Write 1 sample(s) in channel(s): AppC_fromDtoSC -> property is satisfied All Done					
4					
·					
Star	t Stop	Close	<u>Î</u> Del		
Evaluation and Back Tracing

- Start the verification from the dialog window
- Once finished, check that the property is satisfied (lower part of the dialog window)
- Back tracing: return to the activity diagram. What happened to the RL? reachability annotation?

*	Formal Verificatio	n with UPPAAL	•		
Verify with UPP Options of UPP	AL: options AAL Specification				
Size of infinite I	FIFO =	1024			
No deadlocks	?				
Reachability:	O None	Selected	🔾 All		
Liveness:	O None	Selected	ali O 🖌		
Leads to:	None	Selected			
🔲 Generate sim	ulation trace				
🔲 Show verifica	tion details				
Select options and then, click on 'start' to start generation of RG Session id on launcher-1 Sending UPPAL specification data Reachability of: Write 1 sample(s) in channel(s): AppC_fromDtoSC -> property is satisfied					
All Done					
Star	t Stop	Close	🗍 Del		

Hardware Architecture Modeling

SysML-Sec provides a set of predefined component types to model hardware architectures:

- CPUs
- Hardware accelerators
- DMAs
- Memories
- Buses
- Bridges
- FPGAs (new)

Attributes of Hardware Components

CPUs:

- Scheduling policy
- Cycles for execution actions
- Data size for execution actions
- Miss branch
- Cass miss rate
- Etc.
- Buses
 - Arbitration policy
 - Data size per cycle
 - Pipeline
 - Privacy
 - Etc.



Hands-on: Creating a Minimal Hardware Architecture for the Smart Card

- Create an architecture mapping diagram
- Add a CPU, Bus and Memory components
- Connect the CPU and the memory to the bus



Multi-processor Hardware Architecture for the Smart Card



Mapping Model

To constitute a mapping model:

- Application components must be mapped to CPUs or hardware accelerators
- Component connections must be mapped to buses and memories
- Communication patterns (not introduced in this tutorial) must be mapped to CPUs, hardware accelerators, buses and memories
- Note that in the absence of mapping specifications for channels, TTool will compute a default mapping when there is no ambiguity (e.g. only one bus connects the processor onto which the application components are mapped)

Hands-on: Mapping the Smart Card Application to the Minimal Architecture

- First clone the minimal architecture diagram
- On the cloned diagram use the diagram palette to map all tasks onto the single CPU
- Communications mapping will be determined automatically for this simple architecture



40/77

Mapping Model for the Multi-processor Hardware Architecture



		Proceed	lings of CPS&IoT2019 page 904	TELECOM	9.000
41/77	June 2019	Institut Mines-Telecom	SysML-Sec	ParisTech	
				25 8 10	

The SysML-Sec Simulator

Based on transactions

- Represents a computation or communication action involving one or more hardware components
- Dedicated model of computation matching the abstraction level of SysML-Sec models
- Fast simulation
 - Much faster than SystemC
- Allows for performance evaluations
 - Loads of hardware components
 - Latencies
 - Etc.

42/77

June 2019

The SysML-Sec Simulator (cont'd)

- C++ code generated from the mapping model
- User interface and interactive debugger provided from TTool
- References: Knorreck D., Apvrille L., Pacalet R. Fast Simulation Techniques for Design Space Exploration. In: Objects, Components, Models and Patterns. TOOLS EUROPE 2009

Live Demo: Simulation of the Minimal Architecture

- Generate the code and compile the simulator
- Interactive execution of simulator
- Debugger overview
- Breakpoints
- Execution trace generation and latency analysis
- Reachability graph generation and safety properties verification

Live Demo: Simulation of the Multi-Processor Architecture

- Generate the code and compile the simulator
- Interactive execution of simulator
- Debugger overview
- Execution trace generation and latency analysis
- Reachability graph generation and safety properties verification

Discussion: Comparison of Estimated Properties for both Architectures

- Performance: the latency for the multi-processor architecture is significantly reduced
- Safety:
 - Due to multi-processor, some safety properties may not be verified
 - Reachability graph supports the automated analysis of several properties
 - Future work to better support these analyses
- Support for early design space exploration...

Security Requirements



6 535----

Attack Trees and Countermeasures



		Proceed	lings of CPS&IoT2019 page 911	TELECOM	8 1757
48/77	June 2019	Institut Mines-Telecom	SysML-Sec	ParisTech	

Public and Private Hardware



Institut Mines-Telecom

SysML-Sec

Verifications: ProVerif in a Nutshell

- Cryptographic protocol verifier via automated reasoning
- Handles many cryptographic primitives:
 - Shared- and public-key cryptography
 - Hash functions
 - Diffie-Hellman key agreements
- Capable of attack reconstruction: when a property cannot be proved, an execution trace which falsifies the desired property is constructed
- Can prove the following properties:
 - Secrecy (the adversary cannot obtain the secret)
 - Authentication (the adversary does not see the difference when the value of the secret changes)
 - Equivalences between processes that differ only by terms
- Developed by INRIA (french national institute for computer science and applied mathematics)

Proceedings of CPS&IoT2019 page 913

Hands-on: Securing the Channel between the Smart Card and the Terminal

- Request confidentiality of the data channel between the terminal and the smart card:
 - On the application diagram, double-click the data channel port of the terminal
 - On the dialog window, activate the Check for confidentiality option

	😪 Port pro	operties 🔶 💿	×	
/ese	Name, type and par	ameters		
	Name:	fromDtoSC		
e <u>activ</u>	Туре:	Channel	-	
b data	Origin:	Origin	-	
I.	Type #1	<unset></unset>	-	
trom	Type #2	<unset></unset>		
froms	Туре: #3	<unset></unset>	-	
	Type: #4	<unset></unset>	-	
Cara_Re	Type: #5	<unset></unset>	-	
	Blocking?	Non-blocking FIFO	-	
	Finite?	Infinite FIFO		
	Width (in Byte)=	4		
	Capacity=	B	_	
	Reference Requirem	ent ConfidentialityWi		
	nererence nequirem	cine contracticutery training		
	Check Confidentiality			
	Check Authenticit	y		
	Code generation			

Hands-on: Launching ProVerif

- Open the multi-processor architecture mapping diagram
- Launch the syntax analysis
- Launch ProVerif
- On the dialog window, select none for the computation of states reachability
- Start the analysis
- The results should show that the communication is not secured

RP .	Security verific	cation with Pr	oVerif	• E ×
Security	Verification Auto	mated Securi	ty	
Verificat	ion options			
Generate	e ProVerif code in:	/proverit/		
Execute I	ProVerif as:	/home/dbloui	n/Produits/proveri	t/proverif
Compute	state reachability:	O all	Selecte	d 🖲 none
Allow me	ssage duplication in	private chan	nels: 🖲 Yes	O No
🖌 Gener	rate typed Pi calculu	5		
Limit on I	loop iterations:	1		
				13
Results				
Non conf	idential Data:			
Interface	Device fromDtoSC c	bData		
merrate	orence.nonordsc_c	nodta		
_	distant and the		-	
	Start	Stop	Close	
			_	92-
LoT2	019 nage 915			TELEO
				Pari

Hands-on: Making the Communication Bus Private

- Open the multi-processor architecture mapping diagram
- Double-click the bus to edit its properties
- Change the bus privacy to *Private*. This means that the bus cannot be probed (e.g. it is located in a chip)
- Save the modification
- The bus should now be annotated with a privacy flag
- Re-execute the syntax and security analyses
- The data should now be found to be confidential





Hands-on: Automatically adding Security Mechanisms

- Open the multi-processor architecture mapping diagram
- Reset the bus privacy to Public
- Re-execute the syntax and security analyses
- On the dialog window, select the Automated Security tab
- Check the Add security and Add security (confidentiality) options
- Start the process and wait for its completion. This may take some time...



Hands-on: Automatically Adding Security Mechanisms (cont'd)

- The result is a duplication of the application and mapping models (named with the suffix _enc) into which security elements should have been added
- Notice the encryption and decryption blocks that have been added on the duplicated activity diagrams of the terminal and the smart card communication interface
- Re-execute the syntax and security analyses
- The data should now be found to be confidential



Live Demo: Security-Performance Analysis

Simplified model



		Proceed	ings of CPS&IoT2019 page 919	TELECOM	\$ 170r
56/77	June 2019	Institut Mines-Telecom	SysML-Sec	ParisTech	

Security-Performance Analysis (cont'd)





Security-Performance Analysis (cont'd)



58/77

Security-Performance Analysis (cont'd)



Security-Performance Analysis (cont'd)





Proceedings of CPS&IoT2019 page 923

6.000-

60/77 J	June 2019	Institut Mines-Telecom	SysML-Sec	ParisTech
---------	-----------	------------------------	-----------	-----------

Security-Performance Analysis (cont'd)

Scheduling for device: CPU1_1_core_0			
	40 90 99 100 100 110 113 120 125	138 135 148 143 126 125 148 145 176 15	5 MB MR 198 101 200 301 200 103 200
App:_smpided_kierkaedenue			
Average load is 0.156762			
d'an-			
e soci contrast, e de los			
Scheduling for device: CPU0_1_core_0			
	85 96 95 109 100 119 115 129 125	120 120 140 143 120 125 140 145 129 17	2 140 145 150 146 200 201 150 215 220
App: complified TOPP App: complified Application App: complified SmarkCold			
Average load is 0.530781			
a sud, susting, a sud register, sud, susting, a sud, susting, a sud, susting, a site too justing			
Scheduling for device: Bus0_0			
* * * * * * * * * * * * * * * * * *	85 99 95 100 105 110 115 120 125	130 130 140 145 130 155 166 165 156 1	5 180 185 190 191 200 205 110 215 220
App: supplied_hiteches/app: supplied_91999 App: supplied_hiteches/			
Average lead is 0.196213			



61/77





Adding Security



		Proceed	lings of CPS&IoT2019 page 926	TELECOM	See
63/77	June 2019	Institut Mines-Telecom	SysML-Sec	ParisTech	

Designing a Simple Smart Card with SysML-Sec Installation of Tools Introduction Going Fi

Adding Security





Proceedings of CPS&IoT2019 page 927

6 535----



Adding Security



Security-Performance Analysis: Simplified Model

The little	000 1305 1400 1405 1416 1415 1420 1425 143	e juich laam laat juich lakt jame jame juich ju	on take take take take tike tike tike tits in	a 105 10a 100 10a 10a 10a 10a 100 10
100 1105 1340 1345 1350 1355 1380 1345 1375 1375 1380 1387 1	100 100 100 100 101 101 101 102 102 102	0 14/5 1445 1445 5450 1455 1400 1465 14 ⁵ 0 14	17 1440 1445 1400 1401 1500 1505 1515 15	0 1525 1530 1530 1540 1545 1550 1555 1
308 1225 1348 1345 1226 125 1340 1345 1275 1275 1396 1385 1	280 1280 1880 1485 1418 1415 2420 2425 343	0 1425 1440 1445 1450 1455 1466 1465 1450 14	19 146 146 146 146 146 136 136 136 136 136 131 13	e 1525 155e 1556 154e 1545 150e 3555 15

		Proceed	lings of CPS&IoT2019 page 929	TELECOM
66/77	June 2019	Institut Mines-Telecom	SysML-Sec	AnnaTech

Adding Security and Hardware components





Proceedings of CPS&IoT2019 page 930

6 535----

67/77	June 2019	Institut Mines-Telecom	SysML-Sec	PartsTech

Adding Security and Hardware components



68/77 June 2019 Institut Mines-Telecom SysML-Sec

Proceedings of CPS&IoT2019 page 931

6.000-

Security-Performance Analysis: Simplified Model

		Proceed	$\frac{100}{100}$ of CPS $\frac{1}{100}$ [70] 9 $\frac{100}{100}$ [70] 9 $\frac{100}{100}$ [70]	TELECOM
69/77	June 2019	Institut Mines-Telecom	SysML-Sec	ParisTech
Security-Performance Analysis: Summary

Method	Terminate	Min Latency	Max Latency	Scheduling
Non-Secure	7563	736	846	-
AES	10803	1060	1170	Effected
AES+ CPU	9978	974	1128	Effected

6.000-

TTool-SSDLC Interaction





Proceedings of CPS&IoT2019 page 934

71/77	June 2019	Institut Mines-Telecom	SysML-Sec	■※翻
-------	-----------	------------------------	-----------	-----



Other capabilities of TTool:

- Modeling communication patterns
- Modeling fault trees
- Detailed software design with Avatar
- Rapid prototyping (code generation)

Conclusion

- Model-based design of embedded systems is an essential support for the specification, verification, optimization and synthesis of embedded systems
- Discover errors earlier to reduce costs
- Improved designs at lower costs
- Important to take into account safety, security and performances at once
- TTool nicely supports these features

Ongoing Work

- Impact of security mechanisms on safety (VEDECOM Autonomous Vehicle)
- Security-safety-performance interactions with Trustport (European project AQUAS)
- Efficient programming of digital communication infrastructures from models (Nokia)
- Performance tracing through abstraction levels (LIP6)
- Security of Cyber Physical Systems (INRIA duality Capella / TTool)
- Generation of test sequences (ISAE)
- Model and code generation for autonomous drone systems
- Analog components

74/77

SysML-Sec

Acknowledgements

- AQUAS project (Aggregated Quality Assurance for Systems)
- ECSEL program (Electronic Components and Systems for European Leadership)
- Partners:
 - THALES (coordination)
 - Integrasys SA
 - RGB Medical Devices
 - City University
 - Siemens
 - SYSGO
 - University of Aquila
 - CEA
 - TrustPort
 - Telecom ParisTech
 - Etc...

75/77

June 2019

Proceedings of CPS&IoT2019 page 938

SvsML-Sec

References

TTool Website:

https://ttool.telecom-paristech.fr/

- Ludovic Apvrille, Letitia Li, Annie Bracquemond, "Design and Verification of Secure Autonomous Vehicles", Proceedings of the 12th European ITS Congress, Strasbourg, France, June 2017
- Ludovic Apvrille, Yves Roudier, "Designing Safe and Secure Embedded and Cyber-Physical Systems with SysML-Sec", Chapter in Model-Driven Engineering and Software Development, p293–308, Springer International Publishing, 2015



References (cont'd)

- P. H. Feiler, "Model-based Validation of Safety-critical Embedded Systems", Aerospace Conference, 2010.
- J. Hansson, S. Helton, P. H. Feiler, "ROI Analysis of the System Architecture Virtual Integration Initiative", Tech. Report, Software Engineering Institute, 2018.
- N. Lohmann, "Pragmatic model checking: from theory to implementations", Invited presentation at the Business Process Compliance course at the Hasso Plattner Institute, 2013.





Dr. Kim Grüttner Principle Scientist

gruettner@offis.de

+ 49 4419722- 282



Dr. **Maher Fakih** Senior Scientist

<u>fakih@offis.de</u> + 49 4419722- 287 Virtual Platforms for Low-power Mixed-criticality Embedded Systems Development and Validation

TUTORIAL

CPS&IoT'2019 Summer School

Budva, Montenegro, June 10-14, 2019

Joint work with:

Razi Seyyedi, Sören Schreiner OFFIS - Institute for Information Technology, Oldenburg, Germany







Why Low-power Computing?

Introduction

More power-hungry high-performance embedded computational platforms

Low-power essential especially for mobile battery powered systems

- 1. Reliability:
 - > If power consumption and heat are reduced, the positive impact on reliability is doubled:
 - > the negative influence on the ageing of hardware elements is lowered,
 - > Avoid the use of cooling systems (e.g., ventilators) in the HW design

2. Availability:

> A low power consumption allows extending the operation of a system in special situations such as blackouts and energy disruptions.

3. Ecology:

> Power consumption reduction is also a desired feature towards *near-zero emission* in systems with tens/hundreds of electronic control units (ECUs).







Why validation of Low-power management?

Introduction



Low-power computing is a challenge for industrial dependable systems...

> On safety:

- 1. Active-cooling systems are too faulty
- 2. The use of SW low-power techniques should be certification cognizant
- 3. The system should be able to reach a safe-state

> On security

- Counter-measures (e.g., encryption) are power costly
- **CLKSCREW**: Exposing the Perils of Security- Oblivious Energy Management [1]
 - > Attack on ARM Trustzone:
 - > needs root access, but no physical manipulation of hardware
 - > Exploits DVFS features of the System-on-Chip:
 - operates target core in unstable state
 - generates predictable bit flips in the key generator
 - > Result: Access Trustzone information and introduce own code



CLKSCREW: Exposing the Perils of Security-Oblivious Energy Management

Adrian Tang, Simha Sethumadhavan, and Salvatore Stolfo, *Columbia University* https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/tang

Why Virtual Prototype based Solution?



Introduction

Problem

Power becomes Achilles Heel for software and mobile device success

Challenges

- 1. Complex, distributed SW functions
- 2. Executes on heterogeneous HW platforms
- 3. Expensive lab equipment for measurement
- 4. Debugging distorts power measurement
- → Virtual Prototype based Solution



Enhance Virtual Platforms with Low-Power Validation features for MCS



Scope of this talk







- 1. Motivation
- 2. Part I: Background
- 3. Part II: Time Triggered Model in Instruction-Accurate Virtual Platforms
- 4. Part III: Power Models in Instruction-Accurate Virtual Platforms
- 5. Part IV: Power Management Verification in Instruction-Accurate Virtual Platforms
- 6. Conclusion & Questions





Part I

BACKGROUND: MIXED-CRITICAL SYSTEMS (MCS)

Assurance Levels in Aerospace and Railway

IEC 61508 and DO-178

Criticality



Assign System Integrity levels (SIL – IEC 61508) or Design Assurance Levels (DAL – DO178) depending on the required failure probability

The SIL or DAL determine the development processes that should be followed.

Design Assurance Levels (DAL – DO178)	Safety Integrity Level (SIL –IEC 61508)	Probability of dangerous failure per hour
Level A — Catastrophic	SIL-4	>=10 ⁻⁹ to < 10 ⁻⁸
Level B — Hazardous/Severe-Major	SIL-3	>=10 ⁻⁸ to < 10 ⁻⁷
Level C – Major	SIL-2	>=10 ⁻⁷ to < 10 ⁻⁶
Level D – Minor	SIL-1	>=10 ⁻⁶ to < 10 ⁻⁵
Level E – No Effect	SIL-0 (non-SIL)	_

Mixed-Criticality System– What's it?



Multiple safety criticalities on the same platform

- > Long tradition in avionic architectures: Integrated Modular Avionic (IMA) systems
- > Spatial and temporal separation of functionally independent components



Criticality categories for Mixed-Criticality Multi-Rotor Avionics





Extra processor for payload processing?





Microcontroller for flight and navigation tasks

High-performance processor for payload tasks



- Soft doadlings (o.g. d
- Soft deadlines (e.g. d_{4->5})
- Hard power constraints (power budget)
- Hard temperature constraints (weight for cooler)

Single chip solution!



Future mixed-criticality systems: Applications with different criticalities implemented on a (general purpose, COTS) multi-core hardware/software platform that enables temporal and spatial segregation.

→ Size, Weight & Power Reduction (SWaP)







<u>Video</u> (Min 3:04)

MIXED CRITICAL SYSTEM

[13] CONTREX: Design of embedded mixed-criticality CONTRol systems under consideration of EXtra-functional properties





Part I

BACKGROUND: LOW-POWER

Power vs. Energy



Power: instantaneous energy consumption per unit time

- > In Architecture, implies conversion of electricity to heat
- > Need heat sinks and fans
 - > Thermal failures even when fans OK
- Power reduction can <u>reduce costs</u> and can <u>increase</u> reliability and availability

Energy: usage of power for some time

- > Battery life directly depends on energy consumption
- → Energy reduction can increase battery life for mobile systems





Static power *P*_{static}

- > "leakage" related to current leaking from a transistor even if doing nothing
- > Leakage power in the old days not an issue (about 10 % of total processor power)
 - > Up to over 50% of the total power in next-generation processors [c.f. ITRS technology roadmap]

Dynamic power *P*_{dynamic}

- > related to "switching" activity of transistors.
- > Dynamic power density keeps increasing:
 - > About 50-70 % of total processor power

Total power

- $\boldsymbol{P}(\boldsymbol{t}) = \boldsymbol{P}(\boldsymbol{t})_{dynamic} + \boldsymbol{P}(\boldsymbol{t})_{static} = V_{dd}^2(t) \cdot f(t) \cdot \bar{C}(t) + V_{dd}^2(t) \cdot G(\vartheta(t))) \text{ where }$
 - > $\overline{C}(t)$ average switching capacitance (or equivalent) per cycle (usually derived by dynamic annotation or performance counters)
 - > $G(\vartheta(t))$ leakage conductance depending on temperature $\vartheta(t)$ (dynamic temperature model required)
 - > $V_{dd}(t)$, f(t) supply voltage and frequency that can be changed over time (Dynamic Voltage and Frequency Scaling)

Low-power Management Techniques (PMT)



- **1.** Clock gating targets $P_{dynamic}$
 - > Stop switching in unused components
 - > Low latency to achieve low-power mode
- 2. Power gating targets *P*_{static}
 - > Turn off power to unused HW components> High latency for achieve low-power mode
- **3.** Voltage Scaling: both $P_{static} \& P_{dynamic}$
 - > Voltage limitations should be taken into consideration
 - > High latency for achieve low-power mode
- 4. Dynamic Voltage/Frequency scaling (DVFS): both *P*_{static} & *P*_{dynamic}
 - > Limited to retention voltage/frequency.
 - > Finding lowest possible operating frequency and supply voltage is not easy
 - > Increasing application execution time. Can cause possible deadline missing





1. Functional:

- 1. The recovery from Low-power state to normal state must be guaranteed.
 - > E.g. if processors are switched off, then it must be guaranteed that they are also switched off again.
- 2. PMTs side effects on the functional behavior (e.g. loss of data, system halt), must be prohibited.

2. Timing:

- 1. The time/duration required for the application of PMTs must be predictable.
- 2. PMTs are executed at a specified time and must have bounded duration.

3. Power

- 1. The available energy has to be shared by all running applications, regardless if they are critical or not.
- 2. The maximum power consumption of a SoC is effectively limited by its waste heat discharge capabilities and expected lifetime

→ Validation of the correct usage of Power Management Techniques in MCS is vital!

[17] Hypervisor Architecture for Low-Power Real-Time Embedded Systems





Part I

BACKGROUND: VIRTUAL-PLATFORMS









Virtual-platforms - Abstraction and Time





Instruction accurate Virtual Platform Based Software Development



Instruction Accurate Simulation

Pros

- > Complete observability and traceability
- > Non-invasive debugging
- > Deterministic repetition of experiments
- > Execute unmodified target binaries



Cons

- > No support for power models
- > No support for power management validations
- > No support for accurate timing models for MCS
 - > Simple MIPS timing model

Imperas ISS (OVPSim) environment provided by Imperas [2]

Enhance Virtual Platforms with Low-Power Validation features for MCS



Scope of this talk







Part II

TIME TRIGGERED MODEL IN IA VIRTUAL-PLATFORMS

Time-triggered Architecture



Many safety-critical and especially MCS are realized as a time-triggered (TT) system.

Time-Triggered Architecture (TTA)

- 1. TT System has deterministic timing behavior
- 2. Communication and computation according to a pre-determined schedule
- 3. Schedule is defined and executed based on a global time base
- 4. Global time guarantee the timeliness of real-time applications
- 5. Simplifies the verification of the timing behavior

→Virtual-prototyping of MCS requires TT Support

- [3] Functional Test Environment for Time-Triggered Control Systems in Complex MPSoCs using GALI
- [11] Towards Virtual Prototyping of Synchronous Real-time Systems on NoC-based MPSoCs



Why not cycle-accurate Simulation?

GALI (Globally Accurate Locally Inaccurate) simulation




Problem: Instruction Accurate Time Model



GALI: Globally Accurate, Locally Inaccurate

- IA: Globally Inaccurate, Locally Inaccurate
- CA: Globally Accurate, Locally Accurate



GALI: Instruction Accurate improvement



GALI: Globally Accurate, Locally Inaccurate

- IA: Globally Inaccurate, Locally Inaccurate
- CA: Globally Accurate, Locally Accurate







Part II

TIME TRIGGERED MODEL IN IA VIRTUAL-PLATFORMS- MULTIROTOR UC





Time Triggered Schedule

Multirotor UC





Safety-critical flight_algorithm (red task blocks) Video processing application (green task)

Function	Task WCET [ms]
T1: Read Gyroscope Accelerometers	0.541
T2a: Read Magnetometer (start measurement)	0.091
T2b: Read Magnetometer (read sensor data)	0.273
T3: Read Barometer	0.212
T4: Read Remote Control	0.049
T5: Read Battery Guards and Temperature	0.011
T6: Write Remote Control Display	0.100
T7: Voting	0.100
T8: Set Motor Values	0.851
T9: Sensor Processing	0.075
T10: Remote Processing	0.019
T11: Flight Controller	0.035
T12: Non-critical	1.5

Estimated with QCA simulator (see [4])

Multirotor-Mixed Critical Platform

Multirotor UC





[10] SAFEPOWER project: Architecture for Safe and Power-Efficient Mixed-Criticality Systems

Experimental Setup

Multirotor UC







160 times faster than quasi cycle accurate simulation (averaged relative accuracy error of only 0.71%)2.3 to 47 times faster than Approximate Cycle Accurate (ACA) simulation







Part III

POWER MODEL EXTENSION OF IA VIRTUAL-PLATFORMS

Virtual Platform with Power Model









Run-Time Power Management









[12] Integrating Power Models into Instruction Accurate Virtual Platforms for ARM-based MPSoCs



Building blocks for flexible power model

> Static design parameters

> Dynamic annotation/monitoring

Overall power consumption can be computed from static parameters and observations

Hierarchical total power processing possible: $P(t) = P(t)_{dynamic} + P(t)_{static}$



Approach & Methodology

Extra-Functional Properties – State based Power Models and Virtual Power Rails

Simplified example for MicroBlaze



State based power model, similar Benini et al. [5]

- > Each power state has corresponding automata state
- > Transition: valid change between power states



Virtual power rails, Schreiner et al. [6]

- > Consists of two parts
 - > Power rail platform models with PMBus interface to communicate with user application
 - > Power rail observer to provide power and voltage data of power models
- > Enables user application to retrieve power values and react on these like in physical system

OFFIS





Part III

POWER MODEL EXTENSION OF VIRTUAL-PLATFORMS – MULTIROTOR UC



P(t) =	$V_{dd}^2(t) \cdot \mathbf{j}$	$\overline{C}(t) \cdot \overline{C}(t)$	$+V_{dd}^2(t)$	G (fixed	temperature)
	L	J			

<pre>double ps_dynpower_est(int nCores, double clk_cpu, double load_cpu,</pre>
<pre>double clk_mem, double readrate_mem, double writerate_mem,</pre>
<pre>double clk_axi, double usage_axi, int axi_bw,</pre>
<pre>double clk_io)</pre>

Int nCores	number of active cores [0-2]
double clk_cpu	clock frequency of CPU in [Mhz]
double load_cpu	load of processors [0-1]
double clk_mem	clock frequency of memory in [Mhz]
double readrate_mem	read rate of external DDR3 memory [0-1]
double writerate_mem	write rate of external DDR3 memory [0-1]
double clk_axi	AXI dock frequency in [Mhz]
double usage_axi	usage rate of AXI interface [0-1]
int axi_bw	bit width of AXI interface [32 or 64]
double clk_io	clock frequency of IO in [Mhz]
1.06.2010	Virtual Platforms for Low power MCS



 $P(t) = V_{dd}^{2}(t) \cdot \boldsymbol{f}(t) \cdot \overline{\boldsymbol{C}}(t) + V_{dd}^{2}(t) \cdot \boldsymbol{G} \text{ (fixed temperature)}$

- double clk_cpu: clock frequency of CPU in [Mhz]
 - vmirtAddWriteCallback (vmiProcessorP processor, Addr lowAddr, Addr highAddr, vmirtMemWatchFn readC, Bvoid* userData) : void
 - Monitor access to clock speed register interface
- double load_cpu: load of processors [0-1]
 - vmirtGetlCount (vmiProcessorP processor) : Uns64
 - Load: numer of non-empty instructions / (time interval * clk_cpu)
- **double** readrate_mem: read rate of external DDR3 memory [0-1]
 - vmirtAddReadCallback (vmiProcessorP processor, Addr lowAddr, Addr highAddr, vmirtMemWatchFn readC, Bvoid* userData) : void
 - Read rate: number of read instructions / (time interval * clk_axi)
- double writerate_mem: write rate of external DDR3 memory [0-1]
 - vmirtAddWriteCallback (vmiProcessorP processor, Addr lowAddr, Addr highAddr, vmirtMemWatchFn readC, Bvoid* userData) : void
 - Write rate: number of write instructions / (time interval * clk_axi)

Performance Counter based power model for the Xilinx Zynq ARM-based system



$$P(t) = \underbrace{V_{dd}^{2}(t) \cdot f(t) \cdot \overline{C}(t)}_{V_{dd}} + V_{dd}(t) \cdot G$$

```
double ps_dynpower_est(int nCores, double clk_cpu, double load_cpu,
                      double clk mem, double readrate mem, double writerate mem,
                      double clk axi, double usage axi, int axi bw,
                      double clk io) {
P Processor = VCCPINT / 1000.0 * (nCores * clk cpu * load cpu * 0.415 +
               (load_cpu<0.5 ? (0.5-load_cpu)*nCores*clk_cpu*0.1515 : 0));</pre>
P_Processor_PLL = (((clk_cpu*2 > 0) ? 15.0 : 0) + clk_cpu*2*0.02)*VCCPAUX/1000.0;
P AXI = (VCCPINT/1000.0)*clk axi*usage axi*0.010417*axi bw/8;
P Logic = P Processor + P Processor PLL + P AXI;
 P DDR = P Memory + P Memory PLL;
 . . .
 P Interfaces = P USB + P SD + 2*P UART + P I2C + P SPI + 5*P GPIO;
 . . .
P IO = P Interfaces + P Interfaces PLL;
P_total = P_Logic + P_DDR + P_IO;
 return P total;
```

ARM Cortex-A9 - Power Model Overview





11.06.2019

Virtual Platforms for Low-power MCS









DVFS Bare Metal Example



Application Binary is executed bare metal on ARM cores

Switch frequencies and voltages for:

> ARM cores

> DDR memory

Power Model recognizes changes



VCD Traces of DVFS example





<u>Video</u> (Min 2:55)

POWER INTERCEPTION TRACING – DEMO

- 1. Booting Linux
- 2. DVFS on the ARM Cortex-A9
- 3. Power monitor in action





Part IV

POWER MANAGEMENT VERIFICATION IN IA VIRTUAL PLATFORMS

Approach & Methodology

Power Management Techniques

OFFIS

Power state observers on functional level

- > Check if registered application function...
 - > ... enters within a valid set A of power states
 - > ... changes the power state within a valid set B
 - > ... leaves within a valid set *C* of power states
- > Sets are defined through formal power state definitions, which fulfill system's power management specification
- > Function tracing engine: register callbacks on entering and leaving application functions
- > Processor model observer: provides current power state



Approach & Methodology

Power Management Techniques – Formal definition of Power State Observer

Simplified example: Function level power state observers, inputs frequency and voltage

- > Set of frequencies: F ::= [0.0, 400.0] MHz
- > Set of voltages: *V* ::= [0.0, 1.0] V
- > Set of inputs: $I ::= V \times F$
- > Power states: $PS ::= \{ ps \in I | ps = (f_0, v_0) \lor \dots \lor ps = (f_n, v_m) \} \cup \{ (\bot, \bot) \}$

> Transitions:

 $T ::= \{t \in PS \times I \times PS | t = (ps_0, i_0, ps_1) \lor \dots \lor t = (ps_k, i_n, ps_m)\}$ $T^c ::= \{t = (ps, i, ps') \in PS \times I \times PS | t \notin T \land ps' = (\bot, \bot) \land ps \neq (\bot, \bot)\}$

> Observer: $PSObs ::= \{I, PS, T \cup T^c, (f_x, v_y), PS \setminus (\bot, \bot)\}$

Creates an automata, where all transitions to invalid states lead to a fault state





Verification flow



Power state Specification Observers Applications UPPAAL Virtual Platform PMT Traces PMT Traces Power State Observer (s) TO VOC. FIN PE VCC. PNT PE ANN, CPU T ASM 0 Suspend T ASM 1 Suspend T ASS 1 Suspend T ASS 0 Click, Galaci T AS 1 Click, Galaci Trace Comparison matches? mismatches? Finish

Towards Power Management Verification of TTA using VP [7]

Implementation of Power State Observer at Virtual Platform level





[14] Advanced SystemC Tracing and Analysis Framework for Extra-Functional Properties





Part IV

POWER MANAGEMENT VERIFICATION – MULTIROTOR UC

PMTs Verification in OVP

Multirotor UC







Software running on Virtual Platform (OVP)



Internal Overview of Power Intercept Library



Multirotor UC



Observers for ARM and MicroBlaze cores

- > Get characteristic data of ARM and MicroBlaze cores CPU, RAM r/w and bus utilizations, frequencies, voltages suspend and sleep states, clock- and power gating
- > Timing models: improve accuracy of OVP's MIPS model
- > Power models: for each power rail a component belongs to

Observers for the power states

- > Checking if low power techniques of applied correctly and meet the power management specification of the use case
- > DVFS, different suspend and idle states, clock- and power-gating

Power rails observer

- > Use case is able to change voltages and to request present power data of virtual power rails
- > Power data is used in applications, e.g. to react in emergency cases

VCD sinks to store all relevant data for post simulation analyzes

PMTs Verification in OVP

Multirotor UC





Timed Automata Model with power mode variables (UPPAAL)





Software running on Virtual Platform (OVP)



PMTs Verification in OVP

Specification Trace (Timed Automata)

Simulation Trace

(OVP)

Multirotor UC





Virtual Platform in the Loop (VPIL) & Hardware in the Loop (HIL) Simulation



Mixed-Criticality Multi-Rotor Avionics



Virtual Platform in the Loop Simulation

- > Binary compatible platform of physical avionics
- > TMR on ARM 0, MicroBlaze 1 & 2, Voting on MicroBlaze 0
- > Includes intercept libraries
 - > Power Intercept Library
 - > Interface Intercept Library
 - > TCP Intercept Library to connect to flight simulator as environmental model
 - > Fault Injection Intercept Library to evaluate systems behavior in error states (TMR, power management, etc.)

Hardware in the Loop Simulation

- > Evaluation reference system of my approach & methodologies
- > Connects to same flight simulator as VPIL via adapter board





<u>Video</u> (Min 1:38)

MULTIROTOR UC DEMO – HIL W/O LOW POWER
Low-power Multirotor

Full Version



[9] Experimental Evaluation of SAFEPOWER Architecture for Safe and Power-Efficient Mixed-Criticality Systems



Conclusion



Presented methodologies enhancing current Instruction-accurate virtual-platforms to validate:

- **1.** power management techniques
- 2. extra-functional properties
 - > timing (Time-triggered Model)
 - > power (power models)

Evaluated on the use case of a heterogeneous MPSoC based mixed-criticality multi-rotor avionics showing:

> About 33 % of power saving while successfully integrated PMTs, without jeopardizing safety.





This work was partially funded by the European Union's Horizon 2020 research and innovation programme under the grant agreement No. 687902.

More information at: <u>http://www.safepower-project.eu</u>

SAFEPOWER

Thanks for your attention Questions are welcomed ©





[1] A. Tang, S. Sethumadhavan, and S. Stolfo, "CLKSCREW: Exposing the Perils of Security-Oblivious Energy Management," in 26th USENIX Security Symposium (USENIX Security 17), Vancouver, BC, 2017, pp. 1057–1074.

[2] ISS - The Imperas Instruction Set Simulator, Website: <u>http://imperas.com/iss-the-imperas-instruction-set-simulator</u>, last viewed: April 7th 2018

[3] <u>Functional Test Environment for Time-Triggered Control Systems in Complex MPSoCs using GALI</u> Razi Seyyedi and Sören Schreiner and Maher Fakih and Kim Grüttner and Wolfgang Nebel; Euromicro Conference on Digital System Design, DSD 2018, Prague, Czech Republic, August 29 – 31, 2018; 2018

[4] S. Schreiner, R. Görgen, K. Grüttner, and W. Nebel, "A quasi-cycle accurate timing model for binary translation based instruction set simulators," in 2016 International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation (SAMOS), 2016, pp. 348–353.

[5] L. Benini, R. Hodgson, and P. Siegel, "System-level Power Estimation and Optimization," in Proceedings of the 1998 International Symposium on Low Power Electronics and Design, New York, NY, USA, 1998, pp. 173–178.

[6] S. Schreiner, M. Fakih, K. Grüttner, D. Graham, W. Nebel, and S. P. Frasquet, "A Functional Test Framework to Observe MPSoC Power Management Techniques in Virtual Platforms," in 2017 Euromicro Conference on Digital System Design (DSD), 2017, pp. 315–322.

[7] <u>Towards Power Management Verification of Time-Triggered Systems using Virtual Platforms</u> Sören Schreiner and Razi Seyyedi and Maher Fakih and Kim Grüttner and Wolfgang Nebel; International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation (SAMOS) XVIII, Samos Island, Greece, July 15-19, 2018; 7 / 2018





[8] Empowering Mixed-Criticality System Engineers in the Dark Silicon Era: Towards Power and Temperature Analysis of Heterogeneous MPSoCs at System-Level, Kim Grüttner; Model-Implementation Fidelity in Cyber Physical System Design; 2 / 2017

[9] Experimental Evaluation of SAFEPOWER Architecture for Safe and Power-Efficient Mixed-Criticality Systems, Maher Fakih, Kim Grüttner, Sören Schreiner, Razi Seyyedi, Patricia Balbastre, Mikel Azkarate-askatsua, Peio Onaindia, Poggi Tomaso, Alina Lenz, Roman Obermaisser, Adele Maleki, Yosab Bebawy, Duncan Graham, Nera González Romero, Elena Quesada Gonzalez, Johnny Öberg, Tage Mohammadat, Timmy Sundström, Salvador Peiró Frasquet; Journal of Low Power Electronics and Applications; 2019

[10] SAFEPOWER project: Architecture for Safe and Power-Efficient Mixed-Criticality Systems, Maher Fakih, Alina Lenz, Mikel Azkarate-Askasua, Javier Coronel, Alfons Crespo, Simon Davidmann, Juan Carlos Diaz Garcia, Nera González Romero, Kim Grüttner, Sören Schreiner, Razi Seyyedi, Roman Obermaisser, Adele Maleki, Johnny Öberg, Mohamed Tagelsir Mohammadat, Jon Pérez-Cerrolaza, Ingo Sander, Ingemar Söderquist; Microprocessors and Microsystems; May / 2017

[11] Towards Virtual Prototyping of Synchronous Real-time Systems on NoC-based MPSoCs, Razi Seyyedi, M. T. Mohammadat, Maher Fakih, Kim Grüttner, Johnny Öberg and Duncan Graham; 12th IEEE International Symposium on Industrial Embedded Systems (SIES); 6 / 2017

[12] Integrating Power Models into Instruction Accurate Virtual Platforms for ARM-based MPSoCs, Ralph Görgen and Duncan Graham and Kim Grüttner and Larry Lapides and Sören Schreiner; 10 / 2016

[13] CONTREX: Design of embedded mixed-criticality CONTRol systems under consideration of EXtra-functional properties, Ralph Görgen and Kim Grüttner and Fernando Herrera and Pablo Penil and Julio Medina and Eugenio Villar and Gianluca Palermo and William Fornaciari and Carlo Brandolese and Davide Gadioli and Sara Bocchio and Luca Ceva and Paolo Azzoni and Massimo Poncino and Sara Vinco and Enrico Macii and Salvatore Cusenza and John Favaro and Raul Valencia and Ingo Sander and Kathrin Rosvall and Davide Quaglia; Euromicro Conference on Digital System Design (DSD); 8 / 2016





[14] Advanced SystemC Tracing and Analysis Framework for Extra-Functional Properties, Philipp A. Hartmann and Kim Grüttner and Wolfgang Nebel; The 11th International Symposium on Applied Reconfigurable Computing (ARC'15); 4 / 2015

[15] Teaching Mixed-Criticality: Multi-Rotor Flight Control and Payload Processing on a Single Chip, Henning Schlender and Sören Schreiner and Malte Metzdorf and Kim Grüttner and Wolfgang Nebel; Proceedings of the 2015 Workshop on Embedded and Cyber-Physical Systems Education (WESE); 10 / 2015

[16] Autonomous flight control meets custom payload processing: A mixed-critical avionics architecture approach for civilian UAVs, Sören Schreiner and Kim Grüttner and Sven Rosinger; Proceedings of the 5th IEEE Workshop on Self-Organizing Real-Time Systems; 6 / 2014

[17] Hypervisor Architecture for Low-Power Real-Time Embedded Systems, Peio Onaindia and Tomaso Poggi and Mikel Azkarate-askatsua and Kim Grüttner and Maher Fakih and Salvador Peiro and Patricia Balbastre; Euromicro Conference on Digital System Design, DSD 2018, Prague, Czech Republic, August 29 – 31, 2018 ; 2018

[18] An Integration Flow for Mixed-Critical Embedded Systems on a Flexible Time-Triggered Platform, Philipp Ittershagen and Kim Grüttner and Wolfgang Nebel; ACM Transactions on Design Automation of Electronic Systems (TODAES); 5 / 2018

[19] Data- and State-Dependent Power Characterisation and Simulation of Black-Box RTL IP Components at System-Level, Daniel Lorenz and Kim Grüttner and Wolfgang Nebel; 17th Euromicro Conference on Digital Systems Design (DSD 2014); 8 / 2014

CPS&IoT'2019

Summer School on Cyber-Physical Systems and Internet-of-Things Budva, Montenegro, June 10-14, 2019

Citation:

Lech Jóźwiak, Radovan Stojanović, Proceedings of Summer School on Cyber-Physical Systems and Internet-of-Things, Vol. I, MECOnet, 2019

Publishers:

MECOnet Institute (Mediterranean Excellence in Computing and Ontology), <u>www.meconet.me</u>

Montenegrin Association for New Technologies - MANT, www.mant



Year/Place: Montenegro, Podgorica, 2019