Towards Runtime Adaptivity by using Models of Computation for Real-Time Embedded Systems Design

D. S. Loubach [1] E. G. O. Nóbrega [1] I. Sander [2] I. Söderquist [3] O. Saotome [4]

[1]University of Campinas - UNICAMP

[2]KTH Royal Institute of Technology

[3]Aeronautics, Saab AB

[4]Aeronautics Institute of Technology - ITA

ACCES Lab UNICAMP

Agenda

About Denis Loubach

Assistant Professor at UNICAMP, since 2014

Background

- PhD in Electronics and Computer Engineering, ITA, 2012
  Brazilian Aeronautics Institute of Technology - ITA
- MSc (ITA, 2007) and BSc in Computer Engineering (UMC, 2004)

Industry experience as development engineer (*e.g.*, Embraer 2011-2014)

Research Interests:

- Reconfigurable computing
- Embedded systems
- Real time systems
- Real time operating systems
- Model-based design

About Denis Loubach

Assistant Professor at UNICAMP, since 2014

Background

- PhD in Electronics and Computer Engineering, ITA, 2012
  Brazilian Aeronautics Institute of Technology - ITA
- MSc (ITA, 2007) and BSc in Computer Engineering (UMC, 2004)

Industry experience as development engineer (*e.g.*, Embraer 2011-2014)

Research Interests:

- Reconfigurable computing
- Embedded systems
- Real time systems
- Real time operating systems
- Model-based design

About Denis Loubach

Assistant Professor at UNICAMP, since 2014

Background

- PhD in Electronics and Computer Engineering, ITA, 2012
  Brazilian Aeronautics Institute of Technology - ITA
- MSc (ITA, 2007) and BSc in Computer Engineering (UMC, 2004)

Industry experience as development engineer (*e.g.*, Embraer 2011-2014)

Research Interests:

- Reconfigurable computing
- Embedded systems
- Real time systems
- Real time operating systems
- Model-based design

About Denis Loubach

Assistant Professor at UNICAMP, since 2014

Background

- PhD in Electronics and Computer Engineering, ITA, 2012
  Brazilian Aeronautics Institute of Technology - ITA
- MSc (ITA, 2007) and BSc in Computer Engineering (UMC, 2004)

Industry experience as development engineer (*e.g.*, Embraer 2011-2014)

Research Interests:

- Reconfigurable computing
- Embedded systems
- Real time systems
- Real time operating systems
- Model-based design

Introduction

Motivation: aeronautics current trend towards the second generation of **integrated modular avionics** (IMA2G), a.k.a. distributed IMA

Our paper shows that **runtime adaptivity** is feasible to be performed

Case study implementation on both software and hardware, using a **heterogeneous SoC** (uC + FPGA)

To capture and deal with the complexity of adaptivity, we modeled a case study using the theory of **formal models of computation** (MoC), mainly the synchronous MoC

ACCES Lab

Introduction

Motivation: aeronautics current trend towards the second generation of **integrated modular avionics** (IMA2G), a.k.a. distributed IMA

Our paper shows that **runtime adaptivity** is feasible to be performed

Case study implementation on both software and hardware, using a **heterogeneous SoC** (uC + FPGA)

To capture and deal with the complexity of adaptivity, we modeled a case study using the theory of **formal models of computation** (MoC), mainly the synchronous MoC

Introduction

Motivation: aeronautics current trend towards the second generation of **integrated modular avionics** (IMA2G), a.k.a. distributed IMA

Our paper shows that **runtime adaptivity** is feasible to be performed

Case study implementation on both software and hardware, using a **heterogeneous SoC** (uC + FPGA)

To capture and deal with the complexity of adaptivity, we modeled a case study using the theory of **formal models of computation** (MoC), mainly the synchronous MoC

Introduction

Motivation: aeronautics current trend towards the second generation of **integrated modular avionics** (IMA2G), a.k.a. distributed IMA

Our paper shows that **runtime adaptivity** is feasible to be performed

Case study implementation on both software and hardware, using a **heterogeneous SoC** (uC + FPGA)

To capture and deal with the complexity of adaptivity, we modeled a case study using the theory of **formal models of computation** (MoC), mainly the synchronous MoC

Background

Main concepts:

- Model of Computation (MoC)
- Event, Signal, Process
- Synchronous MoC
- ForSyDe
- Real-time embedded systems
- Adaptivity
  - Full *vs.* Partial reconfiguration
- Modeling of adaptivity
  - Signals carrying functions
  - Adaptive process

Background

Main concepts:

- Model of Computation (MoC)
- Event, Signal, Process
- Synchronous MoC
- ForSyDe
- Real-time embedded systems
- Adaptivity
  - Full *vs.* Partial reconfiguration
- Modeling of adaptivity
  - Signals carrying functions
  - Adaptive process

Background

Main concepts:

- Model of Computation (MoC)
- Event, Signal, Process
- Synchronous MoC
- ForSyDe
- Real-time embedded systems
- Adaptivity
  - Full *vs.* Partial reconfiguration
- Modeling of adaptivity
  - Signals carrying functions
  - Adaptive process

Background

Main concepts:

- Model of Computation (MoC)
- Event, Signal, Process
- Synchronous MoC
- ForSyDe
- Real-time embedded systems
- Adaptivity
  - Full *vs.* Partial reconfiguration
- Modeling of adaptivity
  - Signals carrying functions
  - Adaptive process

Background

Main concepts:

- Model of Computation (MoC)
- Event, Signal, Process
- Synchronous MoC
- ForSyDe
- Real-time embedded systems
- Adaptivity
  - Full *vs.* Partial reconfiguration
- Modeling of adaptivity
  - Signals carrying functions
  - Adaptive process

Background

Main concepts:

- Model of Computation (MoC)
- Event, Signal, Process
- Synchronous MoC
- ForSyDe
- Real-time embedded systems
- Adaptivity
  - Full *vs.* Partial reconfiguration
- Modeling of adaptivity
  - Signals carrying functions
  - Adaptive process

Background

Main concepts:

- Model of Computation (MoC)
- Event, Signal, Process
- Synchronous MoC
- ForSyDe
- Real-time embedded systems
- Adaptivity
    - Full *vs.* Partial reconfiguration
- Modeling of adaptivity
    - Signals carrying functions
    - Adaptive process

Background

Main concepts:

- Model of Computation (MoC)
- Event, Signal, Process
- Synchronous MoC
- ForSyDe
- Real-time embedded systems
- Adaptivity
  - Full *vs.* Partial reconfiguration
- Modeling of adaptivity
  - Signals carrying functions
  - Adaptive process

Background (cont'd)

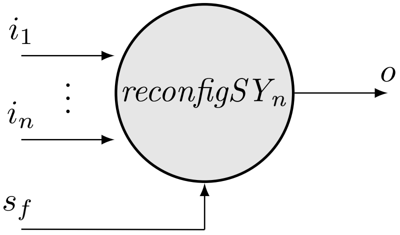

Figure: Process constructor for *reconfigSY* adaptive process

$$ap_f = reconfigSY_n$$

where:

$$o = ap_f(s_f, (i_1, \ldots, i_n))$$
$$o[k] = s_f[k](i_1[k], \ldots, i_n[k])$$

Case Study

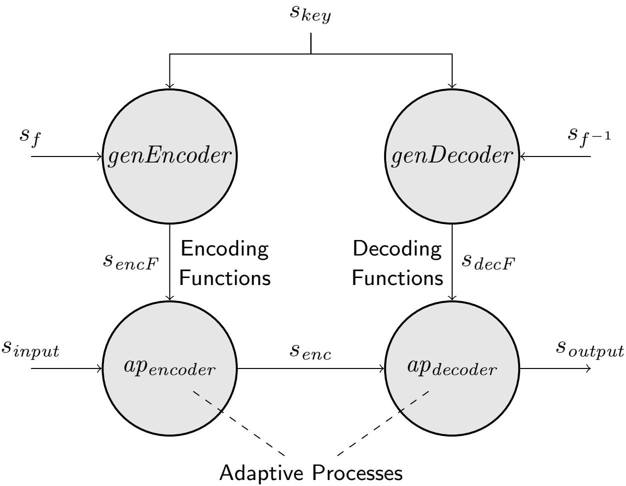

Figure: Specification model

Case Study (cont'd)

Processes definitions:

$$genEncoder : s_f[k](s_{key}[k]) = s_{encF}[k] \tag{1}$$

$$ap_{encoder} : s_{encF}[k](s_{input}[k]) = s_{enc}[k] \tag{2}$$

$$genDecoder : s_{f^{-1}}[k](s_{key}[k]) = s_{decF}[k] \tag{3}$$

$$ap_{decoder} : s_{decF}[k](s_{enc}[k]) = s_{output}[k] \tag{4}$$

Case Study (cont'd)

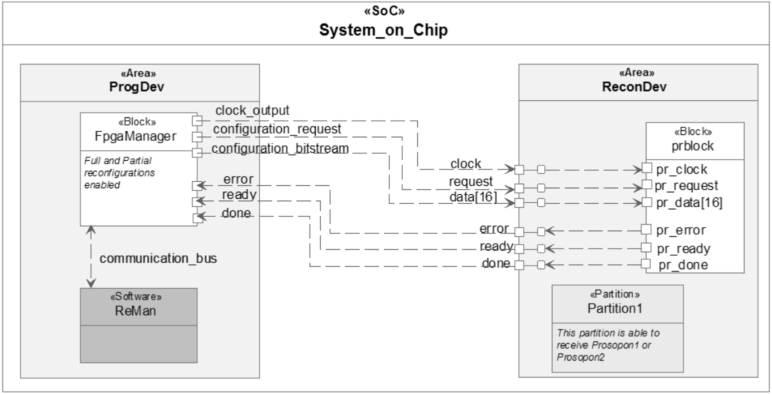

Figure: SoC with the `ProgDev` and `ReconDev` areas overview. It also contains the `ReMan` software application inside the `ProgDev`

Case Study (cont'd)

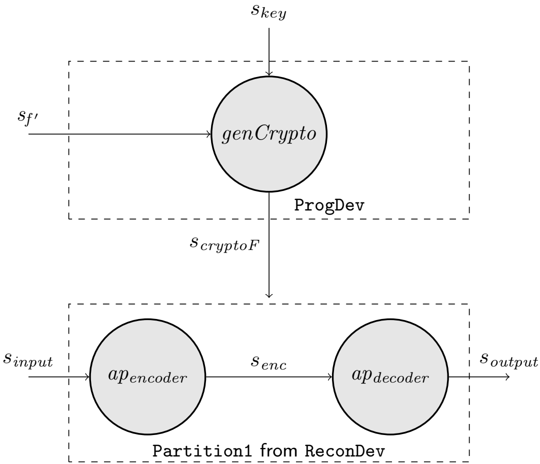

Figure: Encoder and decoder implementation model showing both SoC areas: `ProgDev` and `ReconDev` with one partition named `Partition1`

Case Study (cont'd)

Signals containing the encoding and decoding functions:

$$s_f = \langle (enc_{AES}), (enc_{DES}), \ldots \rangle$$
$$s_f^{-1} = \langle (dec_{AES}), (dec_{DES}), \ldots \rangle$$

s_f and s_f^{-1} signals turned in $s_{f'}$ (for simplicity):

$$s_{f'} = \langle (f_{AES}), (f_{DES}), \ldots \rangle \quad (5)$$

where:
f_{AES} stands for AES cryptography (encoder/decoder) function;
f_{DES} for the DES cryptography (encoder/decoder) function

Case Study (cont'd)

Signals containing the encoding and decoding functions:

$$s_f = \langle (enc_{AES}), (enc_{DES}), \ldots \rangle$$
$$s_f^{-1} = \langle (dec_{AES}), (dec_{DES}), \ldots \rangle$$

s_f and s_f^{-1} signals turned in $s_{f'}$ (for simplicity):

$$s_{f'} = \langle (f_{AES}), (f_{DES}), \ldots \rangle \quad (5)$$

where:
f_{AES} stands for AES cryptography (encoder/decoder) function;
f_{DES} for the DES cryptography (encoder/decoder) function

Results

Table: Full reconfiguration static data

FR	Measure	Number
	ALM	2,896
	Bytes	7,007,204

Table: `Prosopons` and the number of ALM needed to implement them

`Prosopon`	ALM
$prosopon_{AES}$	2,849
$prosopon_{DES}$	1,019

Results (cont'd)

Table: Prosopons bitstream modes and sizes. (*)Data given in Bytes

Bitstream mode	$prosopon_{AES}$*	$prosopon_{DES}$*
Scrub	1,873,272	1,873,272
And/or	3,081,512	3,000,628

Results (cont'd)

- `ReconDev` clock: 50 MHz (20 nano seconds period)
- `ProgDev` clock: 925 MHz

Table: Functions and sub functions runtime clock cycles and computation time

Main function	Sub function	Clock cycles	Time [ns]
f_{AES}	encode	63	1,260
f_{AES}	decode	63	1,260
f_{DES}	encode	18	360
f_{DES}	decode	19	380

Results (cont'd)

Partial and full reconfigurations

Table: Partial reconfiguration measured times. (*)Time is given in mili seconds [ms]

Bitstream mode	$prosopon_{AES}$*	$prosopon_{DES}$*
Scrub	7.76	7.76
And/or	12.7	12.4

Table: Full reconfiguration with no data compression measured time. (*)Time is given in mili seconds [ms]

Bitstream	Time*
FR	29.6

Conclusion

We introduce in this paper a **runtime adaptivity case study** using **formal models of computation** (MoC) for real-time embedded systems design

We modeled an encoder/decoder system in a high-level of abstraction (*system specification*) using ForSyDe

Next, we refined that model and manually transformed it into an *implementation specification* to be designed in software/hardware

Conclusion

We introduce in this paper a **runtime adaptivity case study** using **formal models of computation** (MoC) for real-time embedded systems design

We modeled an encoder/decoder system in a high-level of abstraction (*system specification*) using ForSyDe

Next, we refined that model and manually transformed it into an *implementation specification* to be designed in software/hardware

Conclusion

We introduce in this paper a **runtime adaptivity case study** using **formal models of computation** (MoC) for real-time embedded systems design

We modeled an encoder/decoder system in a high-level of abstraction (*system specification*) using ForSyDe

Next, we refined that model and manually transformed it into an *implementation specification* to be designed in software/hardware

Conclusion (cont'd)

Results showed that one partial reconfiguration takes less time to complete than a full reconfiguration

This enables the possibility to **runtime reconfiguration using partial reconfiguration techniques**, then leading to runtime adaptivity feasibility

Conclusion (cont'd)

Results showed that one partial reconfiguration takes less time to complete than a full reconfiguration

This enables the possibility to **runtime reconfiguration using partial reconfiguration techniques**, then leading to runtime adaptivity feasibility

Acknowledgments

This research work is supported by:

- Swedish-Brazilian Research and Innovation Centre (CISB) – Project CISB ID 68-2015-A
- Regular Research Awards grant #2014/24855-8
  São Paulo Research Foundation - FAPESP
- Altera University Program

Questions & Answers

Thank you for your attention!

Denis Loubach
Assistant Professor, UNICAMP
Email: `dloubach@fem.unicamp.br`
Academic page: `http://www.fem.unicamp.br/~dloubach/`
Lab page: `http://www.fem.unicamp.br/~acceslab/`