Reconfigurable Coprocessors Synthesis in the MPEG-RVC Domain

2015 International Conference on ReConFigurable Computing and FPGA's

7-9 December 2015, Mayan Riviera, Mexico

Francesca Palumbo
Università degli Studi di Sassari
PolComIng – Information Eng. Unit

Carlo Sau, Luca Fanni, Paolo Meloni, Luigi Raffo
Università degli Studi di Cagliari
DIEE – Dept. of Electrical and Electronics Eng.

Outline

- Introduction:
  - Problem statement
  - Background
  - Goals
- Coprocessing units generation:
  - Coarse-Grained reconfiguration
  - Tool flow
  - Available coprocessing layers
- Performance assessment
  - Use-case scenario
  - Results
- Final remarks and future directions

Problem Statement

CONSUMER NEEDS

- **HIGH PERFORMANCES** real time applications:
  - Media players, video calling...
- **UP-TO-DATE SOLUTIONS**
  - Support for the last audio/video codecs, file formats...
- **MORE INTEGRATED FEATURES** in mobile devices:
  - MP3, Camera, Video, GPS...
- **PORTABILITY**
- **LONG BATTERY LIFE**
  - Convenient form factor, affordable price...

Problem Statement

CONSUMER NEEDS

- **HIGH PERFORMANCES** real time applications:
  - Media players, video calling...
- **UP-TO-DATE SOLUTIONS**
  - Support for the last audio/video codecs, file formats...
- **MORE INTEGRATED FEATURES** in mobile devices:
  - MP3, Camera, Video, GPS...
- **PORTABILITY**
- **LONG BATTERY LIFE**
  - Convenient form factor, affordable price...

POSSIBLE SOLUTIONS

- **DATAFLOW MODEL OF COMPUTATION**
  - Modularity and parallelism → **EASIER INTEGRATION AND FAVOURED RE-USABILITY**
- **COARSE-GRAINED RECONFIGURABILITY**
  - Flexibility and resource sharing → **MULTI-APPLICATION PORTABLE DEVICES**

Problem Statement

CONSUMER NEEDS

- **HIGH PERFORMANCES** real time applications:
  - Media players, video calling...
- **UP-TO-DATE SOLUTIONS**
  - Support for the latest media/video codecs, file formats

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.

  - Convenient form factor, affordable price...

POSSIBLE SOLUTIONS

- **DATAFLOW MODEL OF COMPUTATION**
  - Modularity and parallelism → **EASIER INTEGRATION AND FAVOURED RE-USABILITY**
- **COARSE-GRAINED RECONFIGURABILITY**
  - Flexibility and resource sharing → **MULTI-APPLICATION PORTABLE DEVICES**

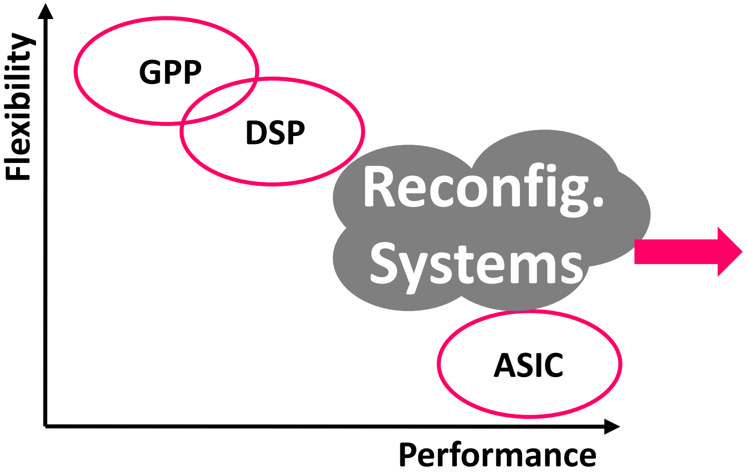
Background: Hw Reconfigurability

FINE- GRAINED (FG) ACCELERATORS

- High flexibility bit-level reconfiguration
- Slow and memory expensive configuration phase

COARSE-GRAINED (CG) ACCELERATORS

- Medium flexibility word-level reconfiguration
- Fast configuration phase



	FG	CG
	Bit-level	Word-level
Flexibility	☺	😐
Reconf. Speed	😐	☺
Config. Storage	☹	😐

Background: Hw Reconfigurability

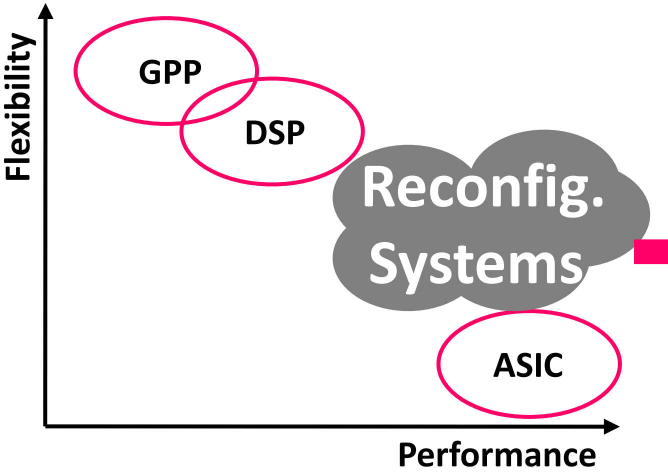
AUTOMATIC GENERATION

FINE- GRAINED (FG) ACCELERATORS → VIVADO (XILINX) NIOS II (ALTERA)

- High flexibility bit-level reconfiguration
- Slow and memory expensive configuration phase

COARSE-GRAINED (CG) ACCELERATORS → ???

- Medium flexibility word-level reconfiguration
- Fast configuration phase



	FG	CG
	Bit-level	Word-level
Flexibility	😊	😐
Reconf. Speed	😐	😊
Config. Storage	☹	😐

Background : The Dataflow MoC

DATAFLOW PROGRAM

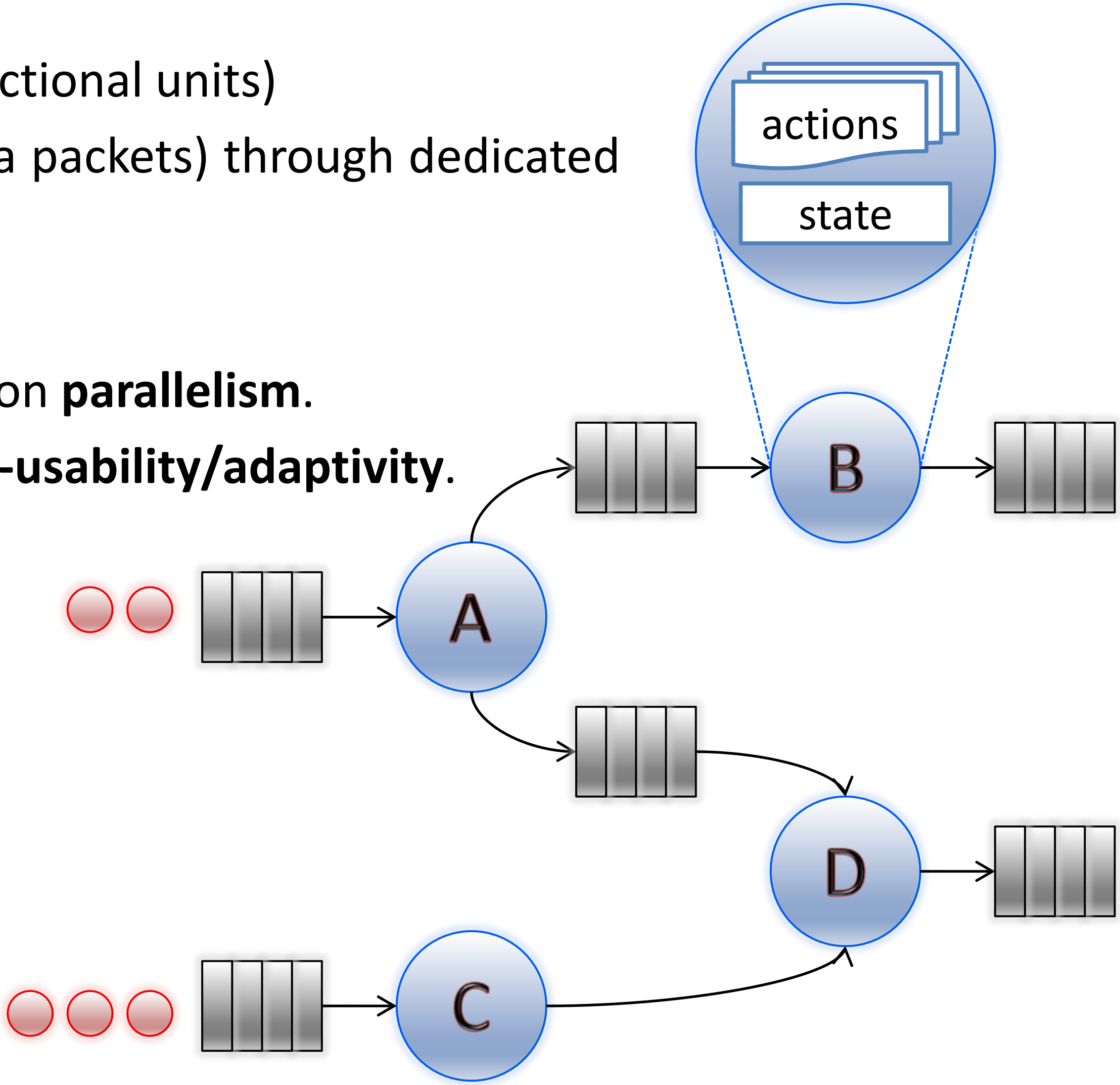
- Directed graph of **actors** (functional units)
- Actors exchange **tokens** (data packets) through dedicated channels

PECULIARITIES

- Explicit the intrinsic application **parallelism**.
- **Modularity** favours model **re-usability/adaptivity**.

EXTERNAL INTERFACE

- I/O ports **number**
- I/O ports **depth**
- I/O ports **burst** of tokens



Background : The Dataflow MoC

DATAFLOW PROGRAM

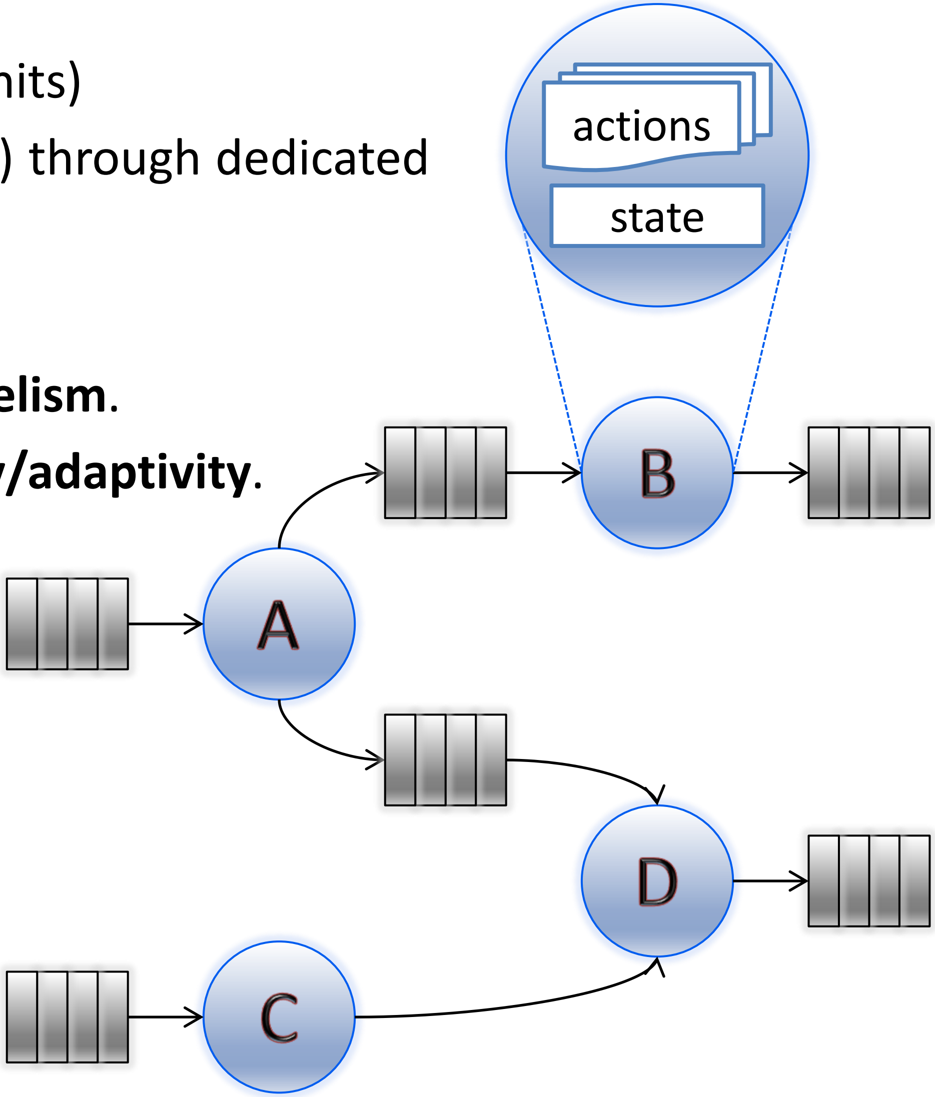
- Directed graph of **actors** (functional units)
- Actors exchange **tokens** (data packets) through dedicated channels

PECULIARITIES

- Explicit the intrinsic application **parallelism**.
- **Modularity** favours model **re-usability/adaptivity**.

EXTERNAL INTERFACE

- I/O ports **number**
- I/O ports **depth**
- I/O ports **burst** of tokens



Research Evolution and Objectives

DASIP 2010:

- High-level dataflow combination tool, front-end of the Multi-Dataflow Composer tool.

DASIP 2011:

- Concrete definition of the hardware template and of the dataflow-based mapping strategy.

ISCAS 2012:

- Integration of the complete synthesis flow.

SAMOS 2014 & SPS JOURNAL:

- Implementation of a coarse-grained multi-standard decoder.

Research Evolution and Objectives

GOAL: automatic deployment of **CG RECONFIGURABLE ACCELERATORS**, by means of **HIGH LEVEL SYNTHESIS** and **DATAFLOW-BASED CUSTOMIZATION** strategies.

DASIP 2010:

- High-level dataflow combination tool, front-end of the Multi-Dataflow Composer tool.

DASIP 2011:

- Concrete definition of the hardware template and of the dataflow-based mapping strategy.

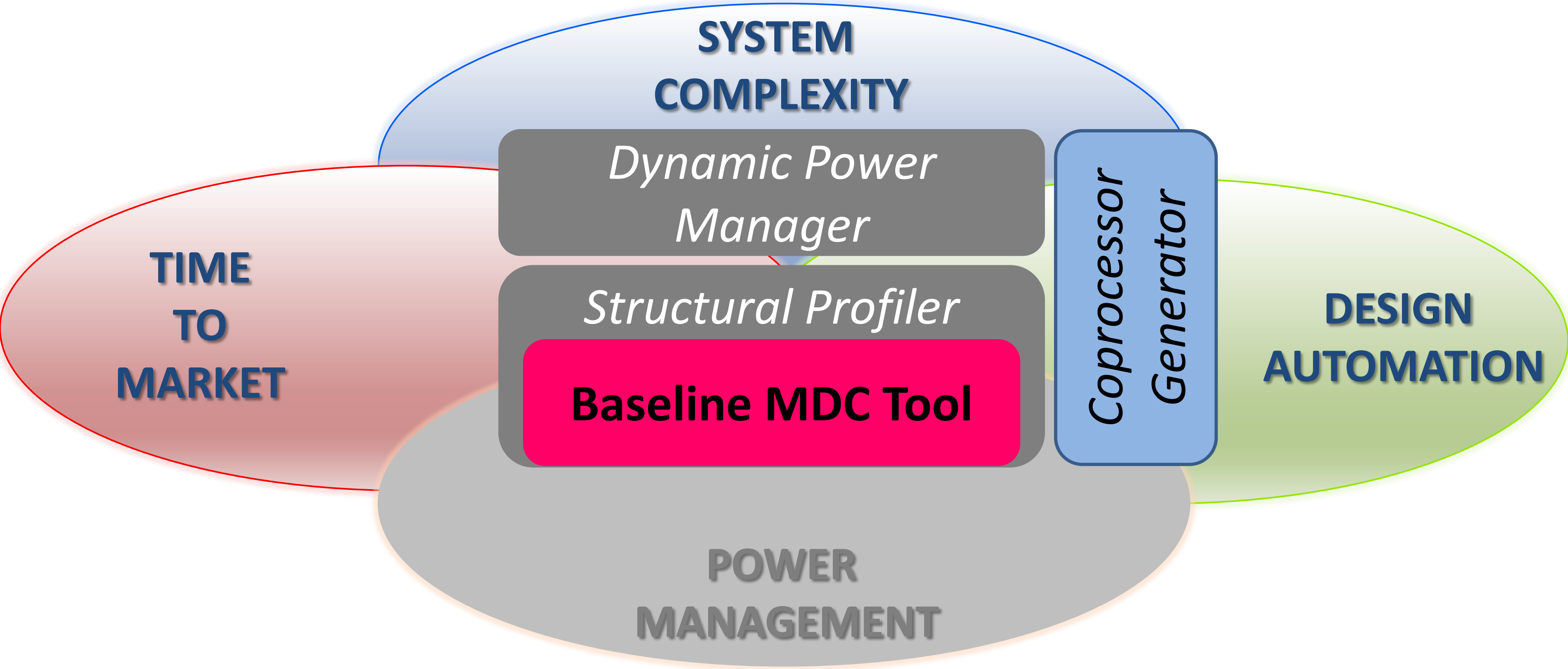
ISCAS 2012:

- Integration of the complete synthesis flow.

SAMOS 2014 & SPS JOURNAL:

- Implementation of a coarse-grained multi-standard decoder.

Multi-Dataflow Composer (MDC) tool



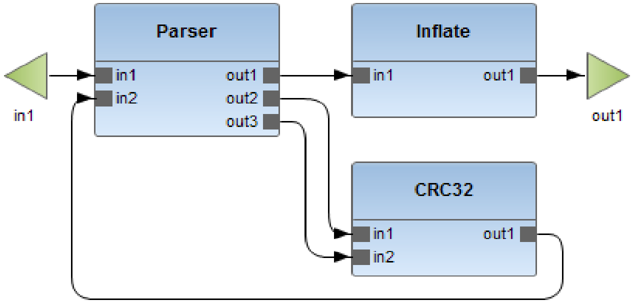
Reconfigurable Platform Composer Tool Project (L.R. 7/2007, CRP-18324)
January 2012 – December 2015
http://sites.unica.it/rpct/

Outline

- Introduction:
  - Problem statement
  - Background
  - Goals
- Co-processing units generation:
  - Coarse-Grained reconfiguration
  - Tool flow
  - Available co-processing layers
- Performance assessment
  - Use-case scenario
  - Results
- Final remarks and future directions

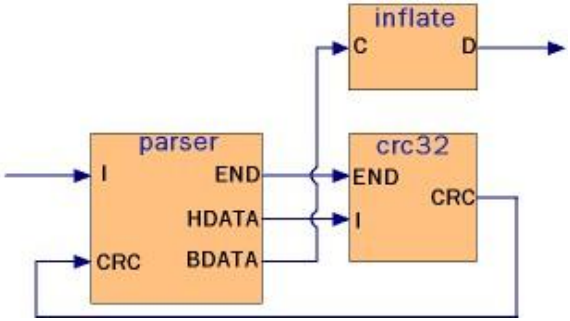
CG Datapath Merging

Dataflow Description

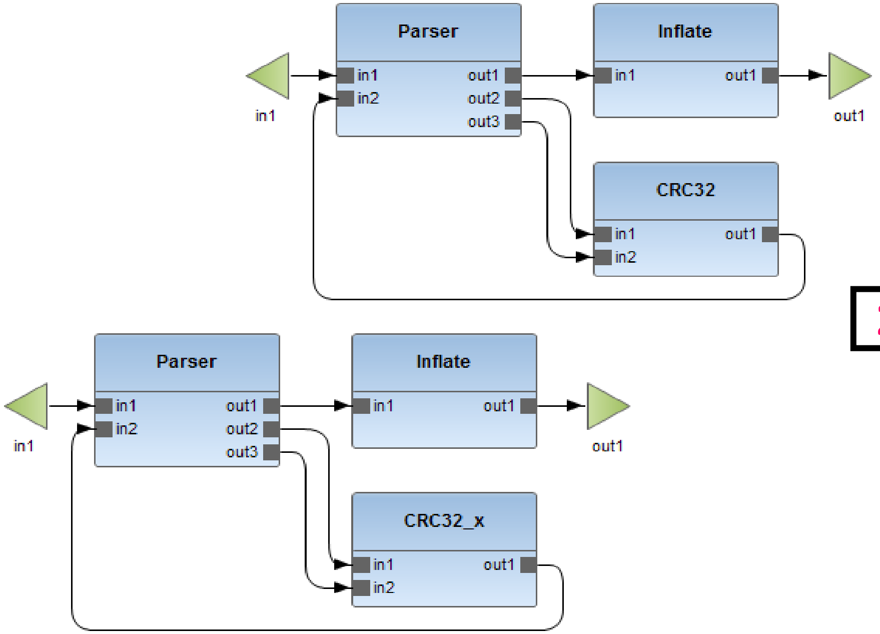


1:1

Coarse Grained Hardware Platform

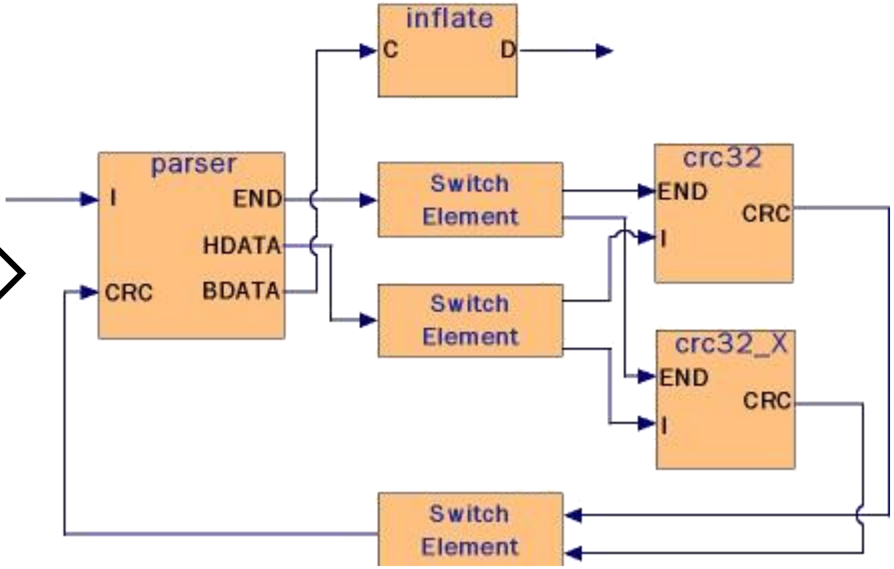


Dataflow Descriptions



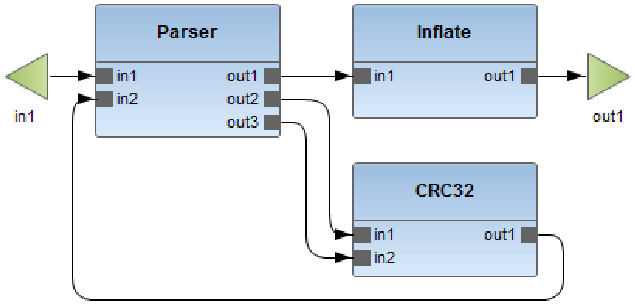
2:1

Coarse Grained Reconfigurable Hardware Platform



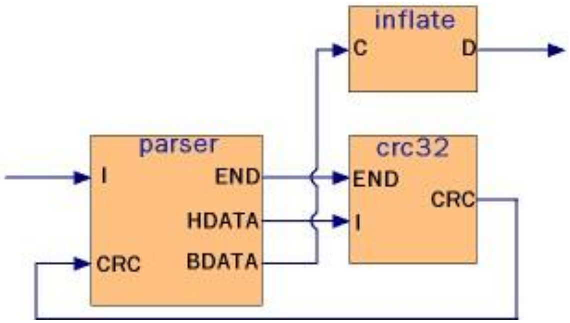
CG Datapath Merging

Dataflow Description

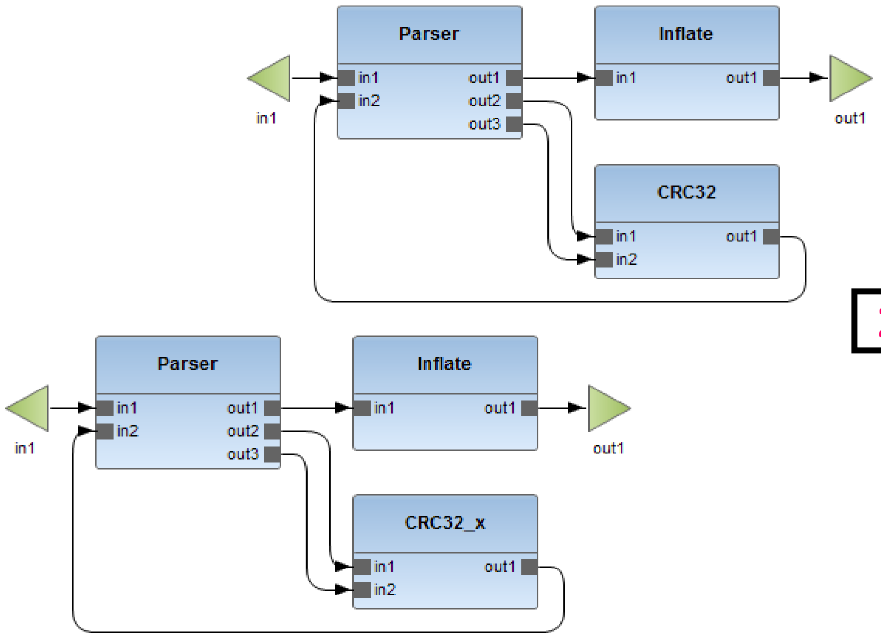


1:1

Coarse Grained Hardware Platform

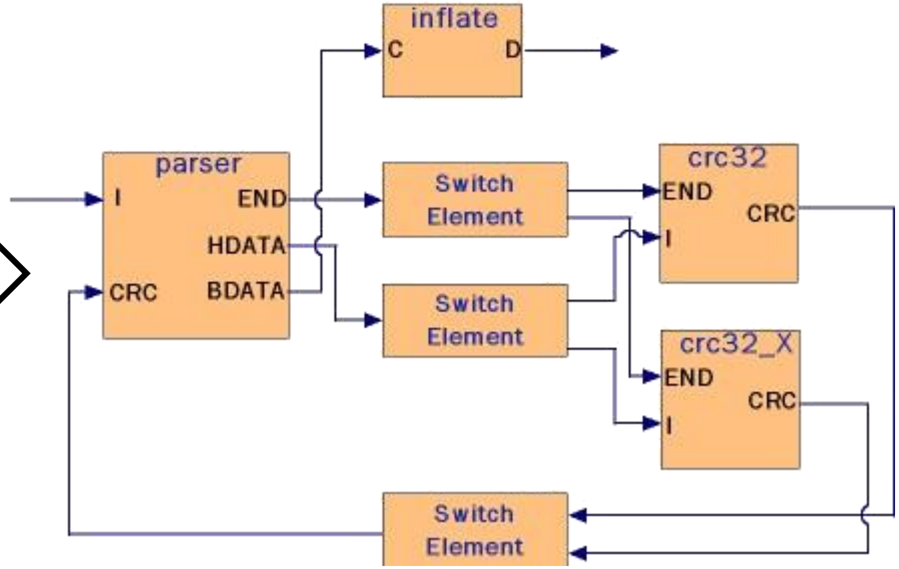


Dataflow Descriptions



2:1

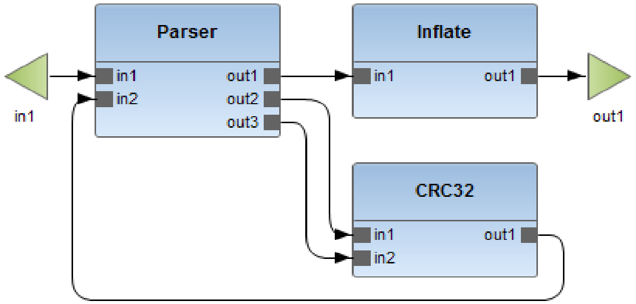
Coarse Grained Reconfigurable Hardware Platform



APPLICATION	# KERNEL	# ACTORS	# SBOX
zoom	7	87	54

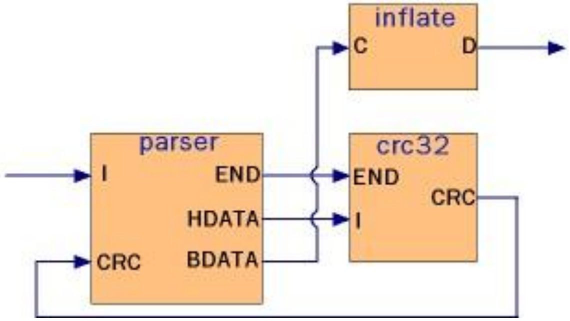
CG Datapath Merging

Dataflow Description

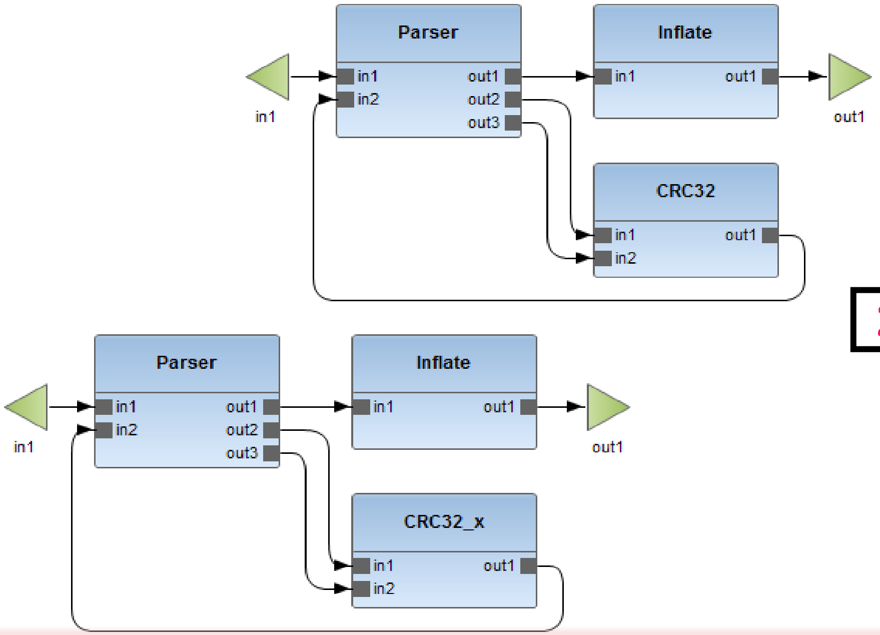


1:1

Coarse Grained Hardware Platform

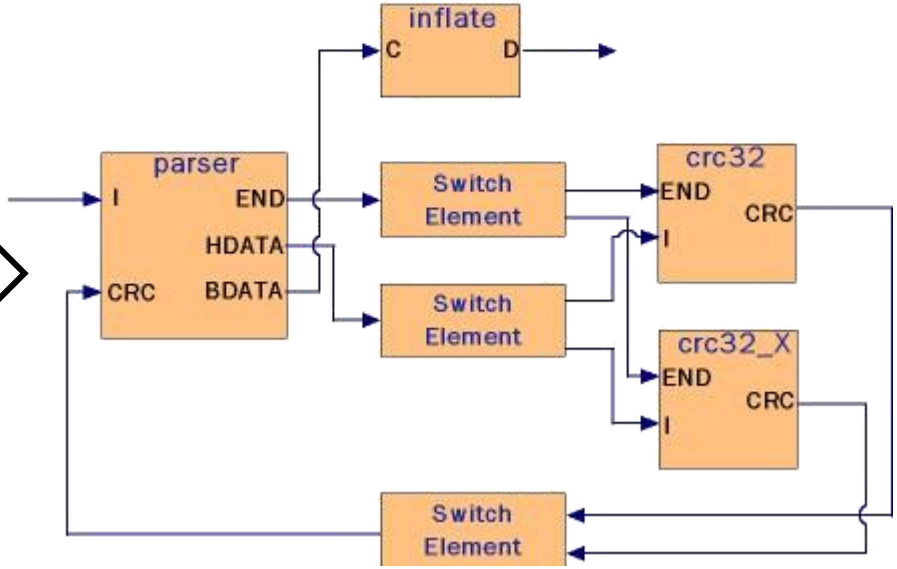


Dataflow Descriptions



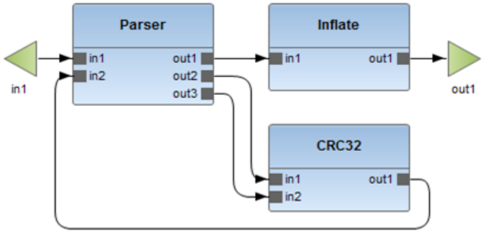
2:1

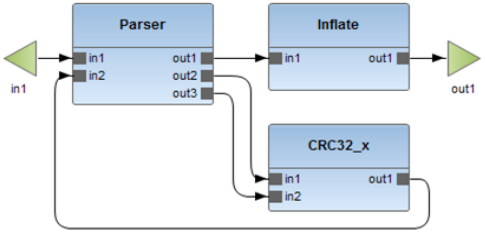
Coarse Grained Reconfigurable Hardware Platform



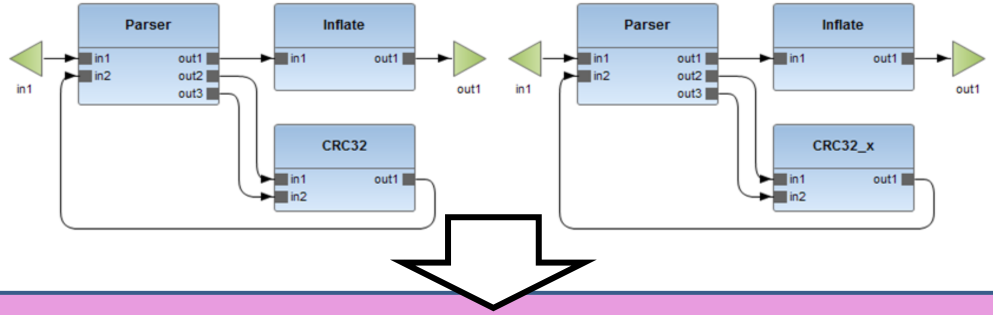
NEEDS: automatic management of the **CG SYSTEM RUNTIME CONFIGURABILITY** and of the custom **ACCELERATOR DEPLOYMENT**.

Proposed Flow



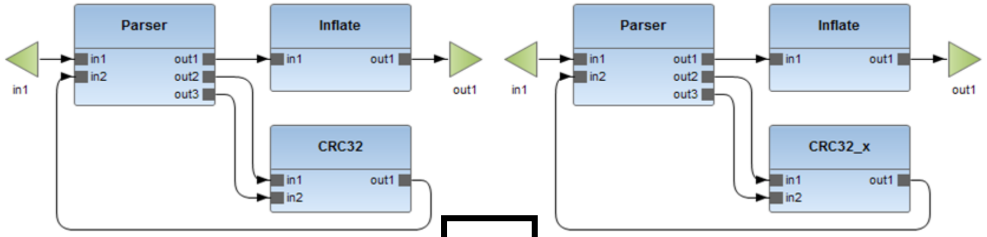


Proposed Flow



PHASE 1: HIGH-LEVEL SPECIFICATION COMPOSITION

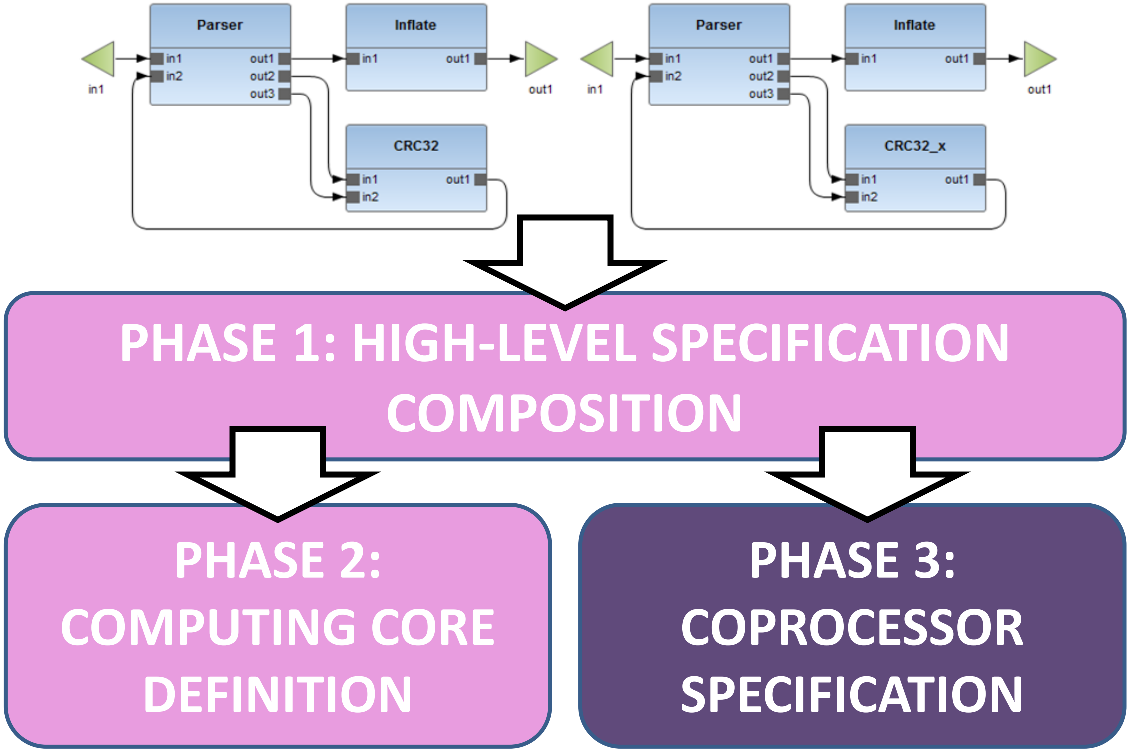
Proposed Flow



PHASE 1: HIGH-LEVEL SPECIFICATION COMPOSITION

PHASE 2: COMPUTING CORE DEFINITION

Proposed Flow

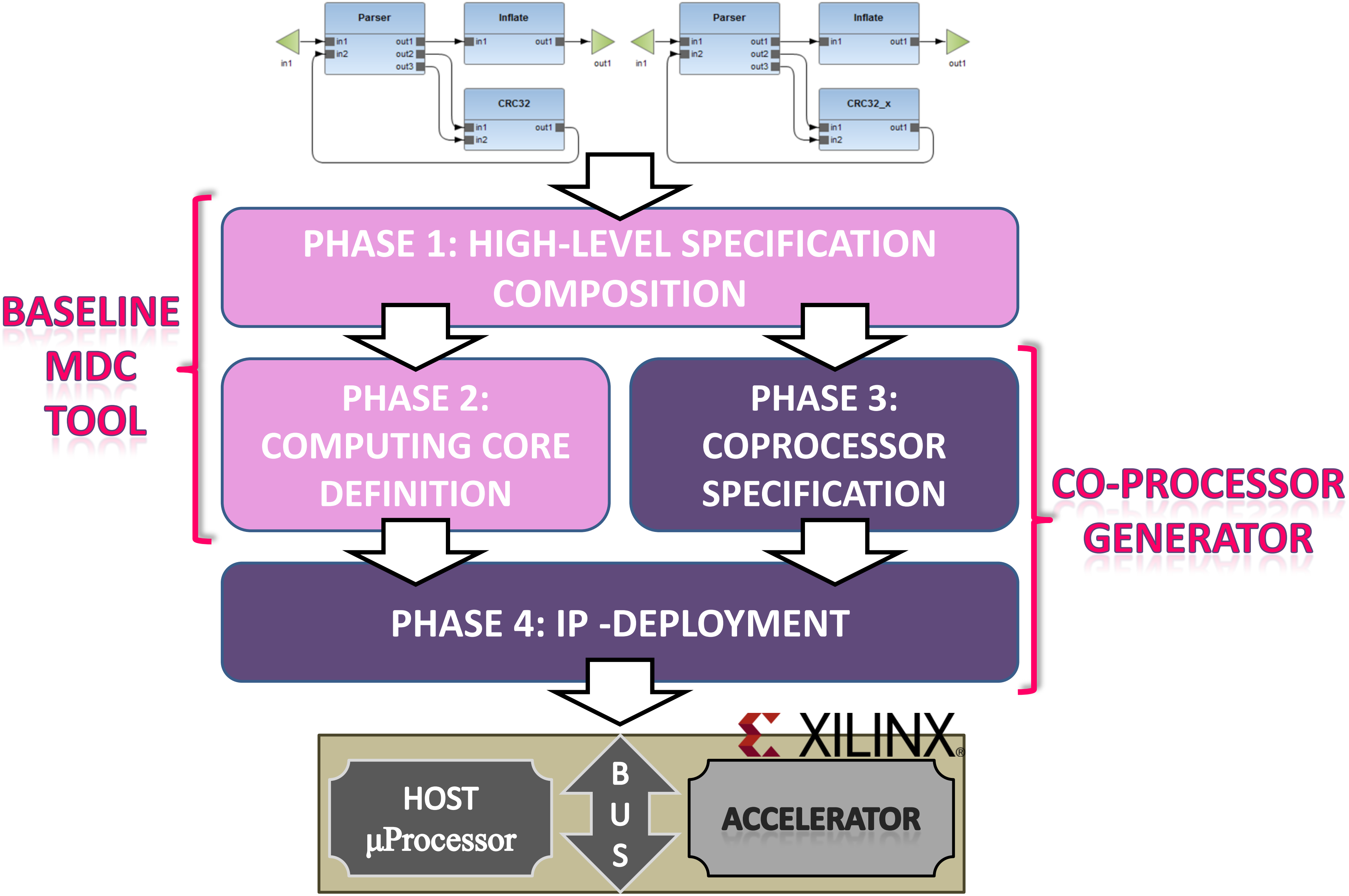


PHASE 1: HIGH-LEVEL SPECIFICATION COMPOSITION

PHASE 2: COMPUTING CORE DEFINITION

PHASE 3: COPROCESSOR SPECIFICATION

Proposed Flow



PHASE 1: HIGH-LEVEL SPECIFICATION COMPOSITION

PHASE 2: COMPUTING CORE DEFINITION

PHASE 3: COPROCESSOR SPECIFICATION

PHASE 4: IP -DEPLOYMENT

BASELINE MDC TOOL

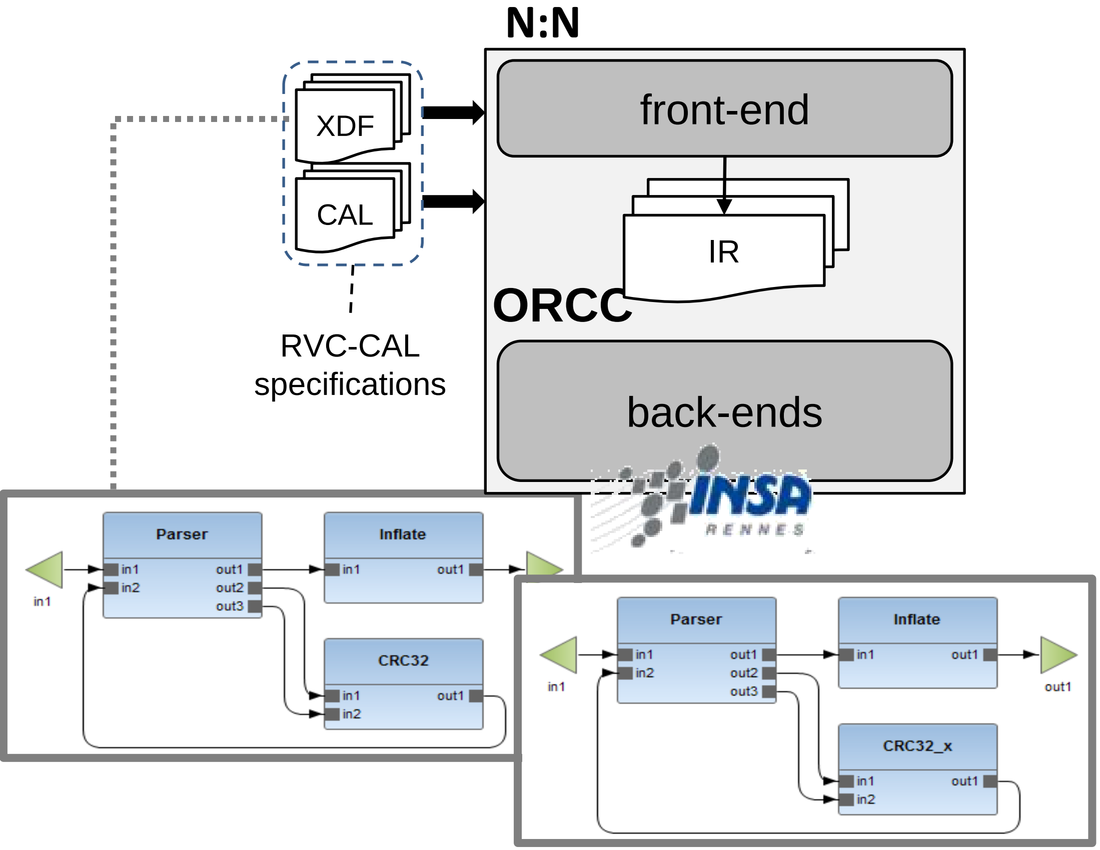
CO-PROCESSOR GENERATOR

HOST µProcessor

ACCELERATOR

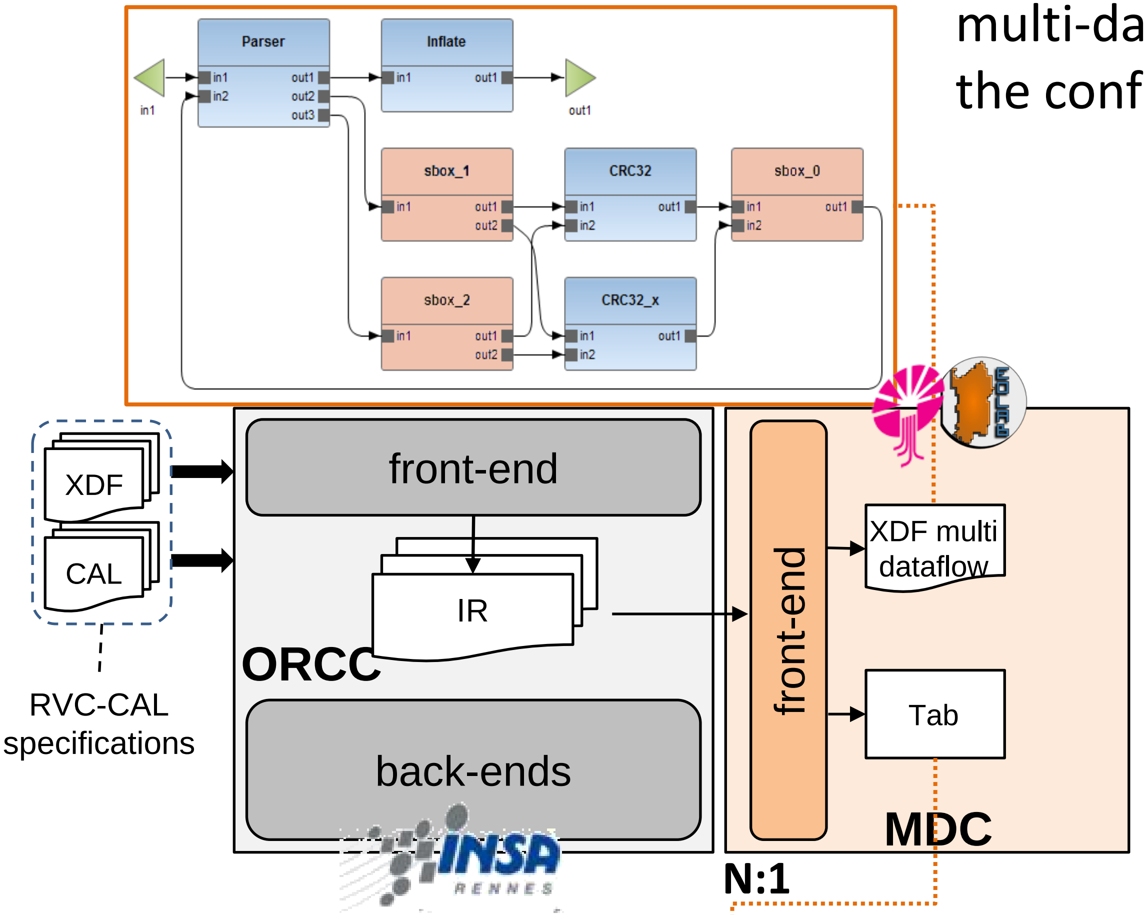
Baseline MDC: High-Level Specif. Composition

PHASE 1.a: ORCC acquires all the input dataflow specifications, one by one, and transform them into java intermediate representations.



Baseline MDC: High-Level Specif. Composition

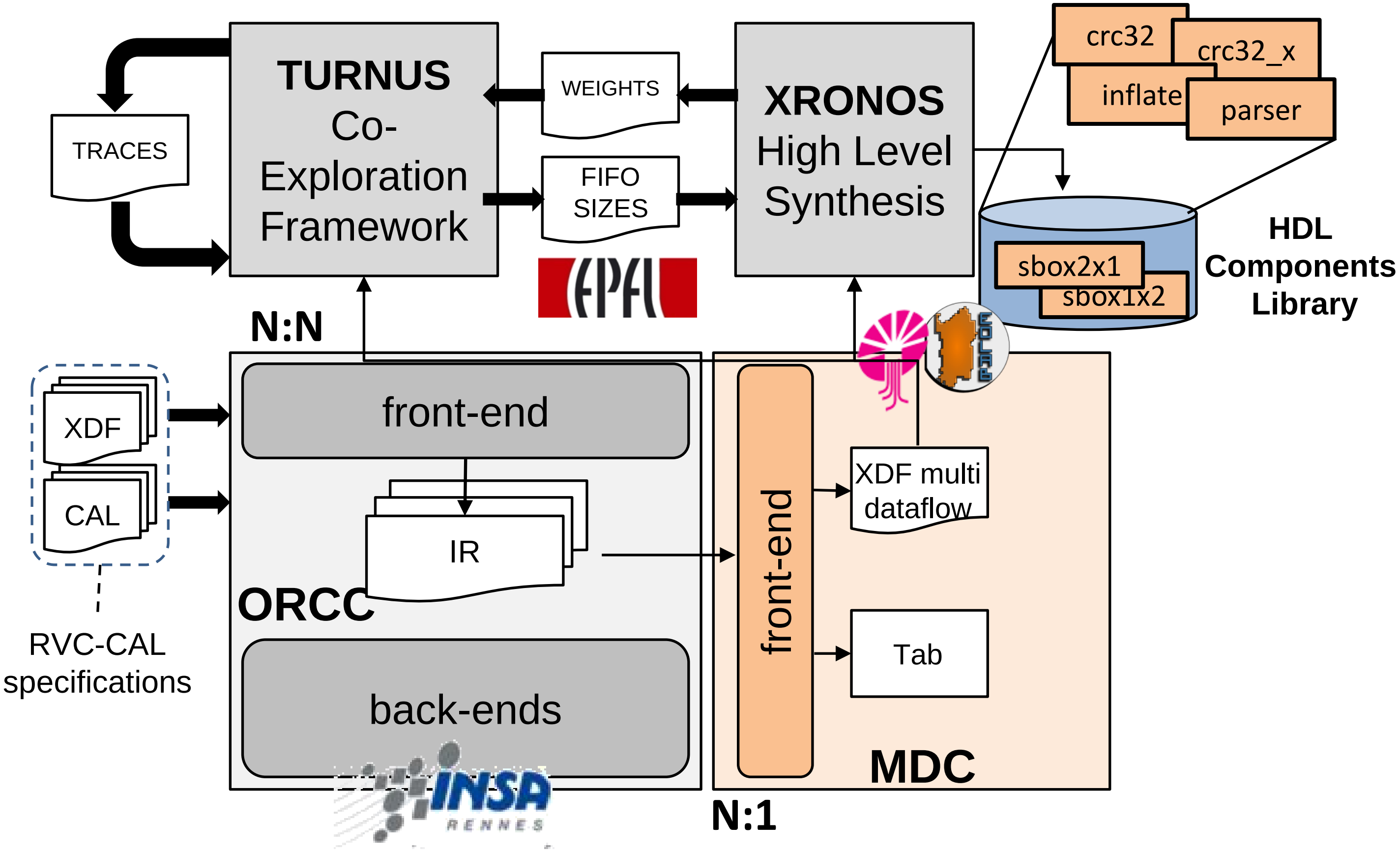
PHASE 1.b: MDC front-end performs the datapath merging. It outputs the multi-dataflow network and the configuration table (Tab).



SBox	D1	D2
S0	0	1
S1	0	1
S2	0	1

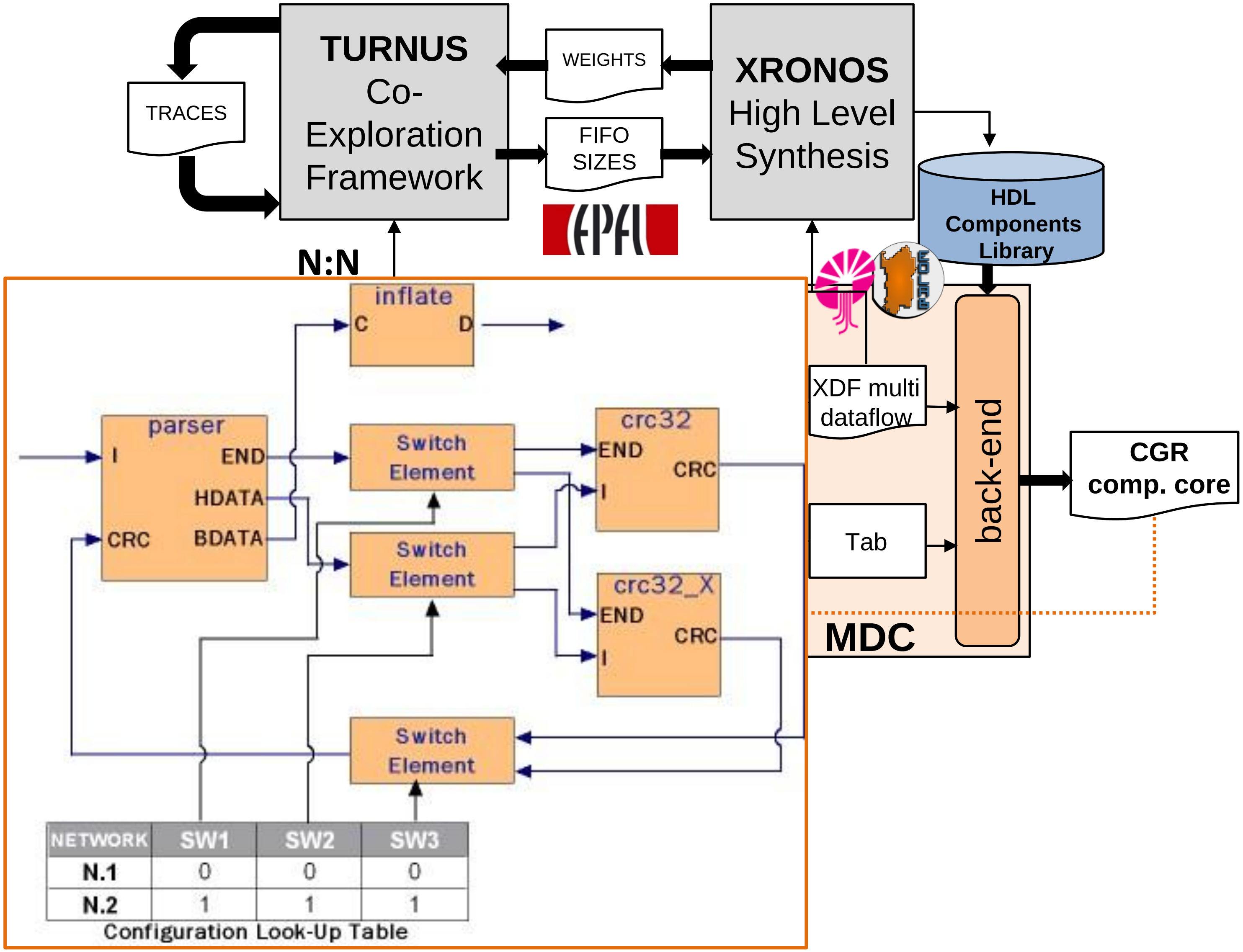
Baseline MDC: Computing Core Definition

PHASE 2.a: Xronos and Turnus are used to create the library of HDL components, which implement the actors functionalities.



Baseline MDC: Computing Core Definition

PHASE 2.b: MDC backend assembles the Coarse-Grained Reconfigurable computing core.

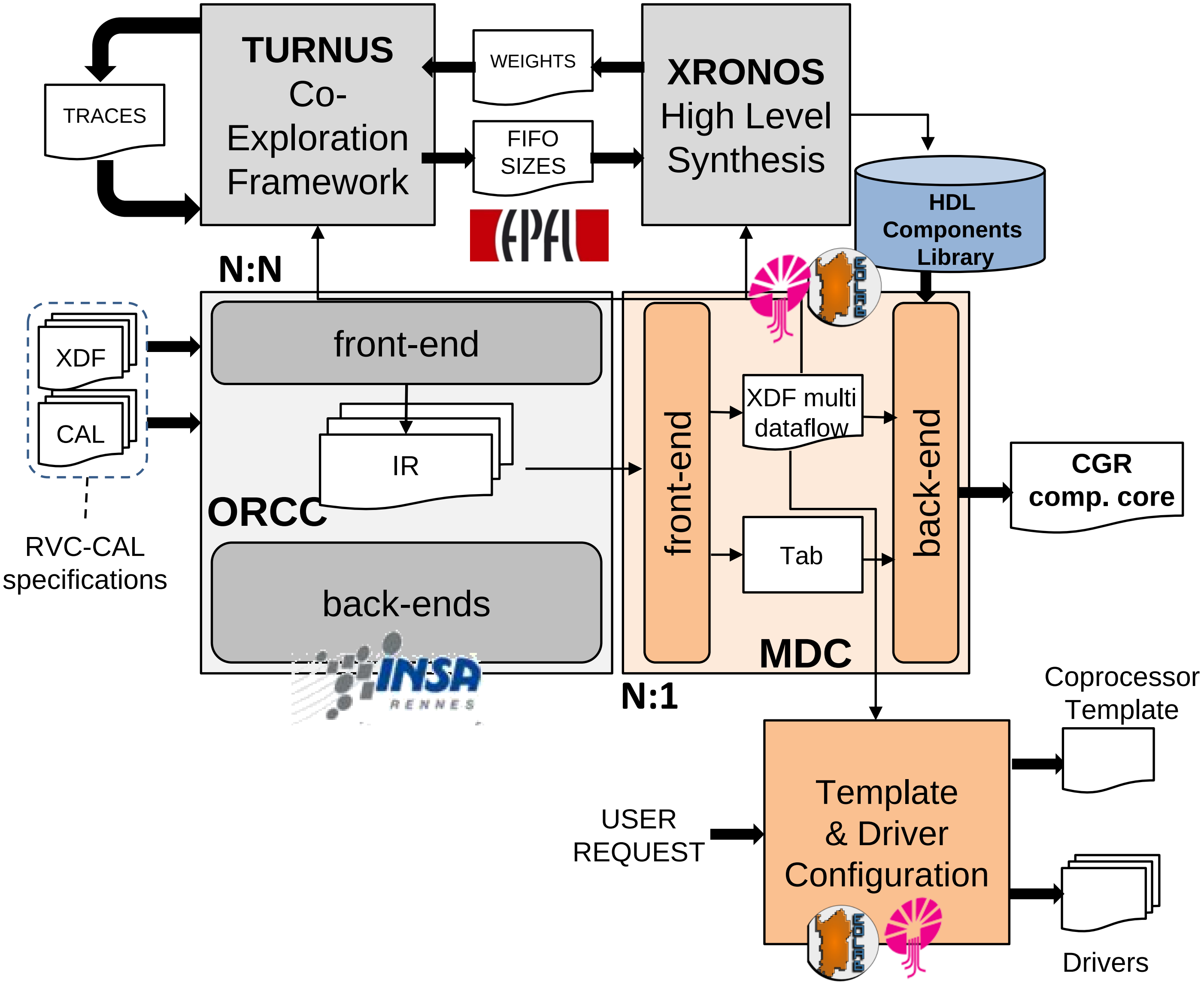


NETWORK	SW1	SW2	SW3
N.1	0	0	0
N.2	1	1	1

Configuration Look-Up Table

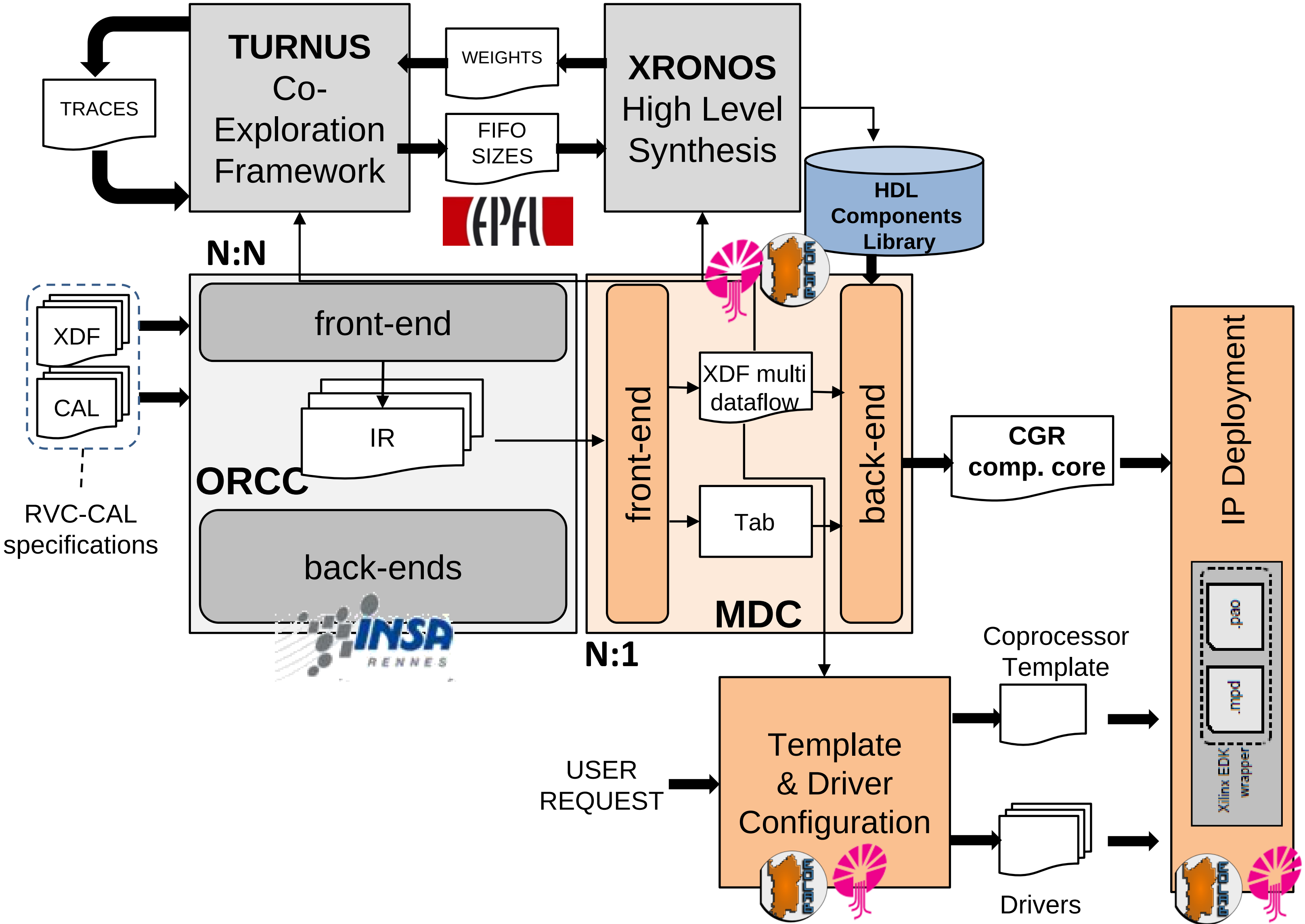
MDC Coprocessor Generator: Specification

PHASE 3: Template and drivers are characterized according to the user selected coprocessor and derived from the multi-dataflow network analysis.



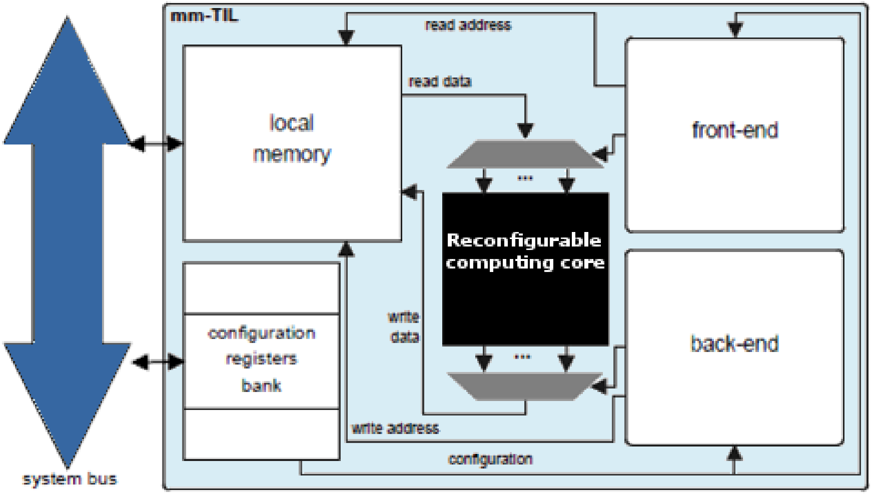
MDC Coprocessor Generator: Deployment

PHASE 4: The processor-accelerator system is assembled and a Xilinx compliant IP is released.



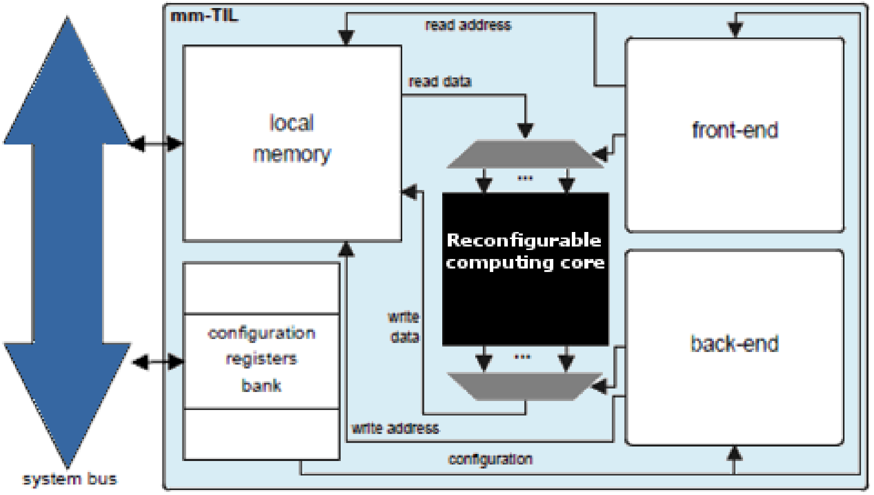
Template Interface Layer Architecture

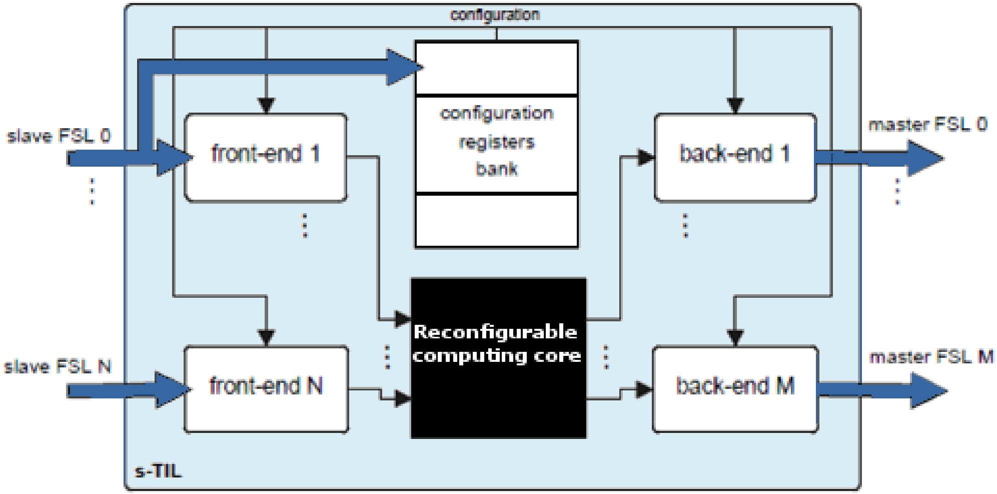
Template Interface Layer Architecture



- Memory-mapped loosely coupled coprocessor: accessible through the system bus as a memory-mapped IP.

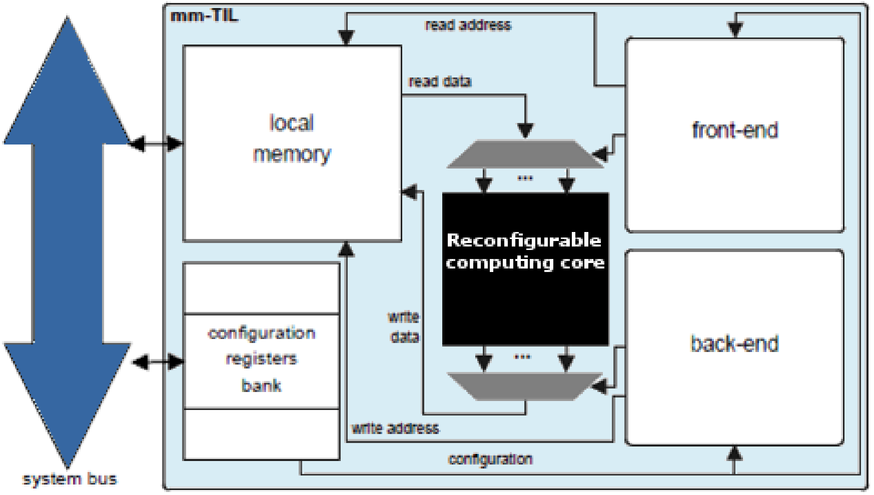
Template Interface Layer Architecture

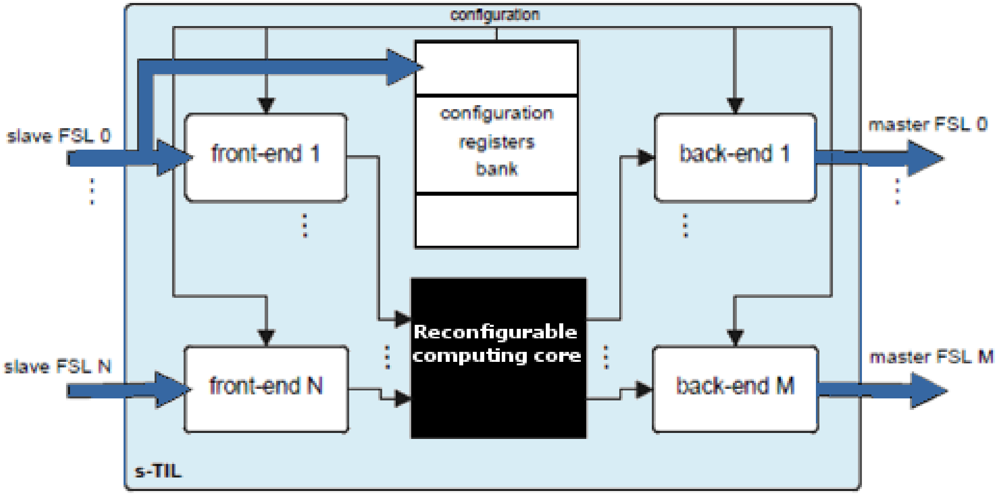




- Memory-mapped loosely coupled coprocessor: accessible through the system bus as a memory-mapped IP.
- Stream-based tightly coupled coprocessor: accessible through different full duplex links, one for each I/O port.

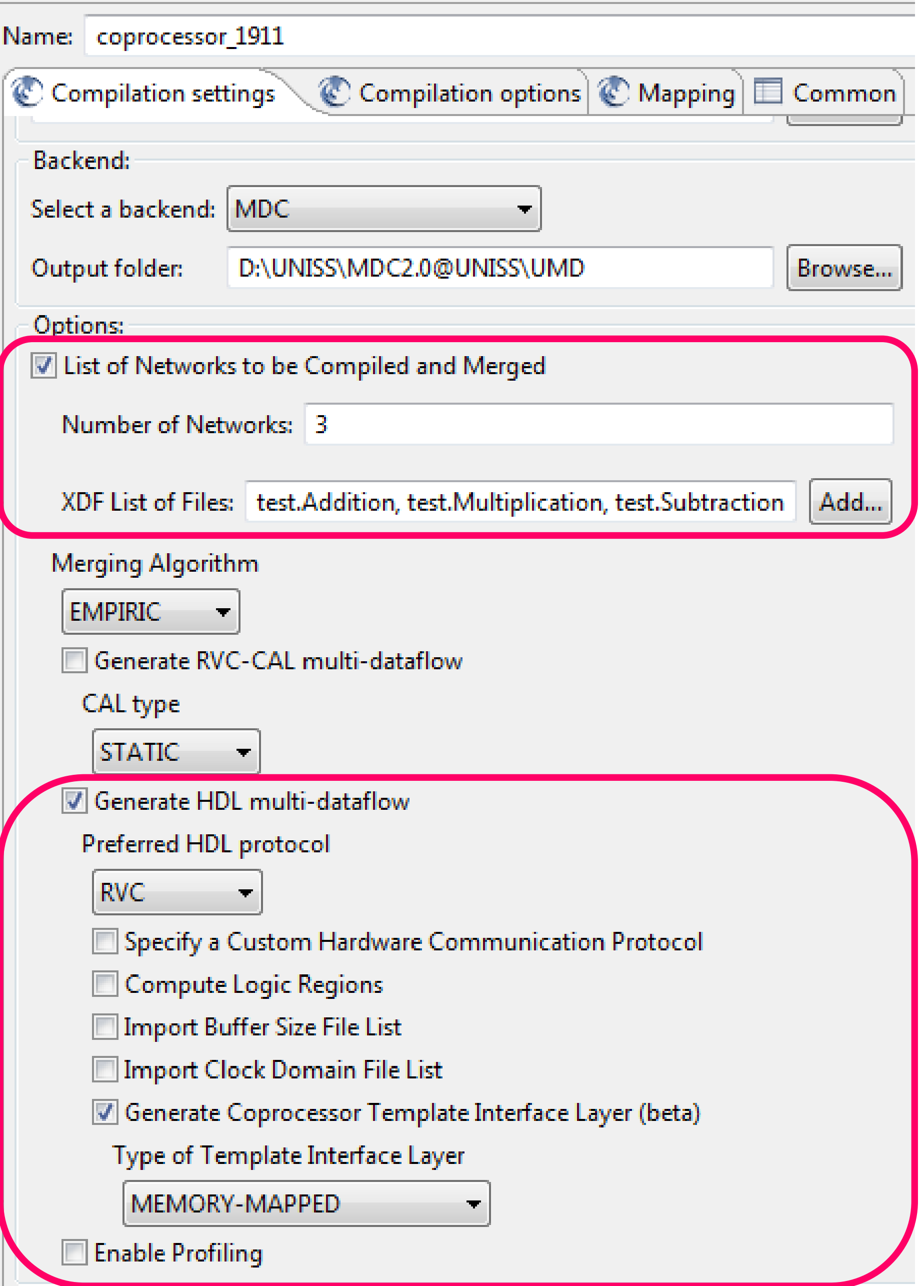
Template Interface Layer Architecture





- Memory-mapped loosely coupled coprocessor: accessible through the system bus as a memory-mapped IP.
- Stream-based tightly coupled coprocessor: accessible through different full duplex links, one for each I/O port.
- In both cases the Template Interface Layer:
  - integrates a bank of configuration registers, to store the desired configuration;
  - One (or more) front-end(s), to load data into the reconfigurable computing core;
  - one (or more) back-end(s), to read the computed data from the reconfigurable computing core.

MDC settings



Name: coprocessor_1911

Backend:

Select a backend: MDC

Output folder: D:\UNISS\MDC2.0@UNISS\UMD

Options:

☑ List of Networks to be Compiled and Merged

Number of Networks: 3

XDF List of Files: test.Addition, test.Multiplication, test.Subtraction

Merging Algorithm

EMPIRIC

☐ Generate RVC-CAL multi-dataflow

CAL type

STATIC

☑ Generate HDL multi-dataflow

Preferred HDL protocol

RVC

☐ Specify a Custom Hardware Communication Protocol

☐ Compute Logic Regions

☐ Import Buffer Size File List

☐ Import Clock Domain File List

☑ Generate Coprocessor Template Interface Layer (beta)

Type of Template Interface Layer

MEMORY-MAPPED

☐ Enable Profiling

LIST OF INPUT SPECIFICATIONS

TICK TO ENABLE COPROCESSOR GENERATION.
REQUESTED INPUT: TIL to be generated.

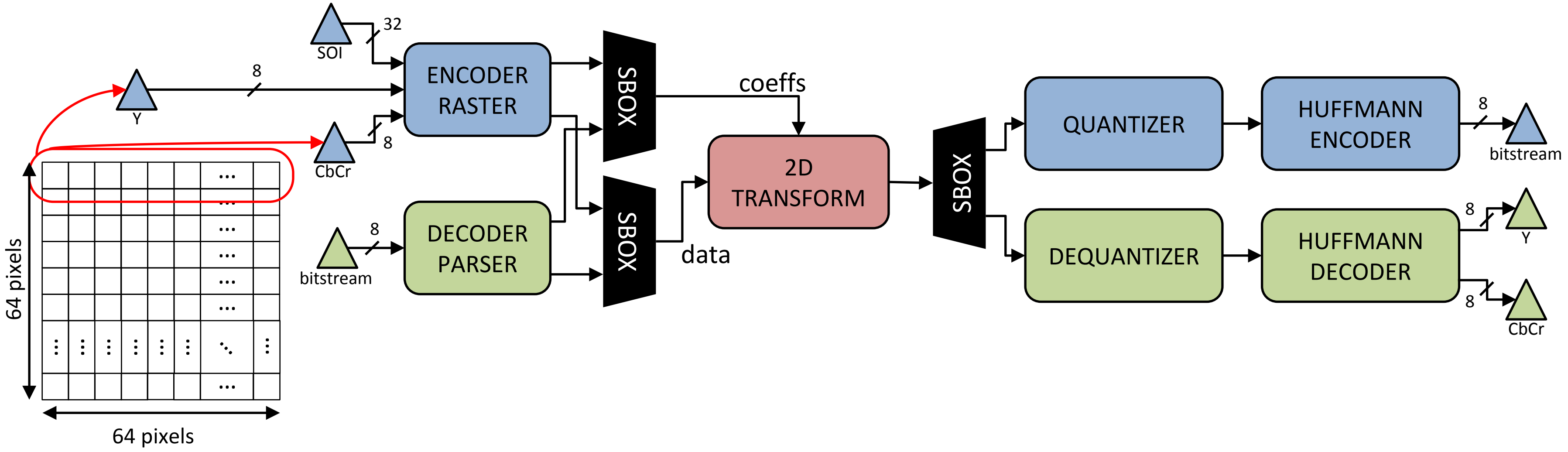
Outline

- Introduction:
  - Problem statement
  - Background
  - Goals
- Coprocessing units generation:
  - Coarse-Grained reconfiguration
  - Tool flow
  - Available coprocessing layers
- Performance assessment
  - Use-case scenario
  - Results
- Final remarks and future directions

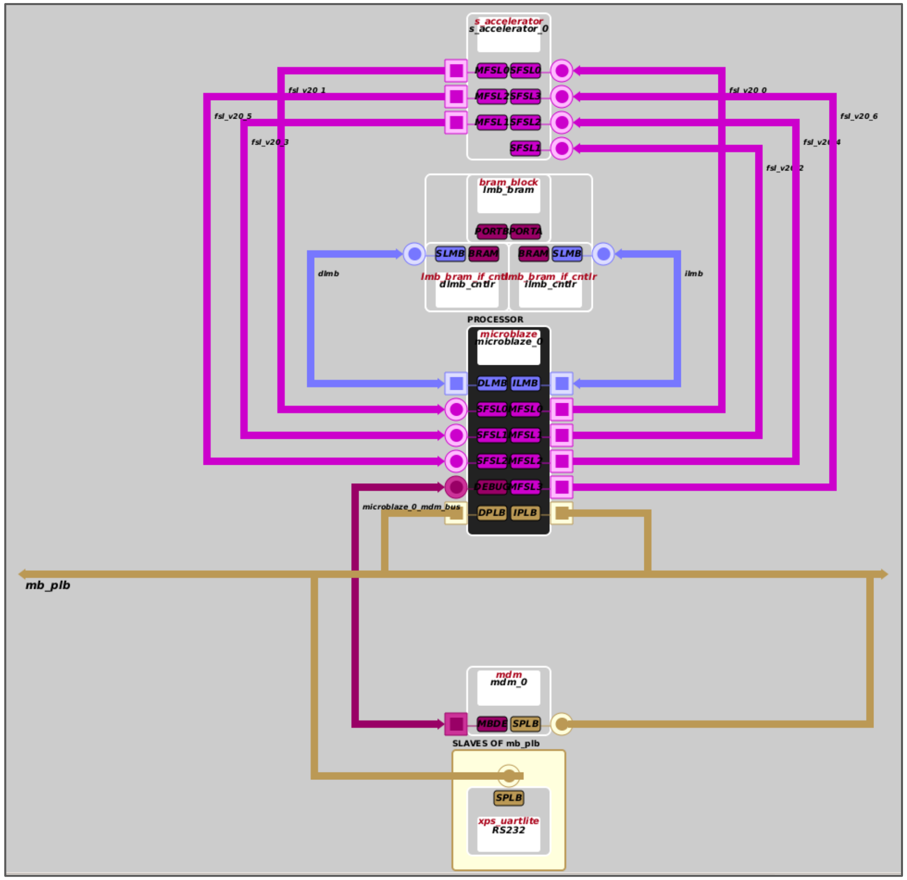
Use-Case: JPEG Codec

- Based on the simple profile ITU-T.IS 1091 standard.
- I/O footprint of the multi-dataflow system:
  - 3 input ports and 1 output port for the encoder and 1 input and 2 outputs for the decoder;
  - data channel depths vary from 8 to 32 bit;
  - token patterns less than or equal to 64.

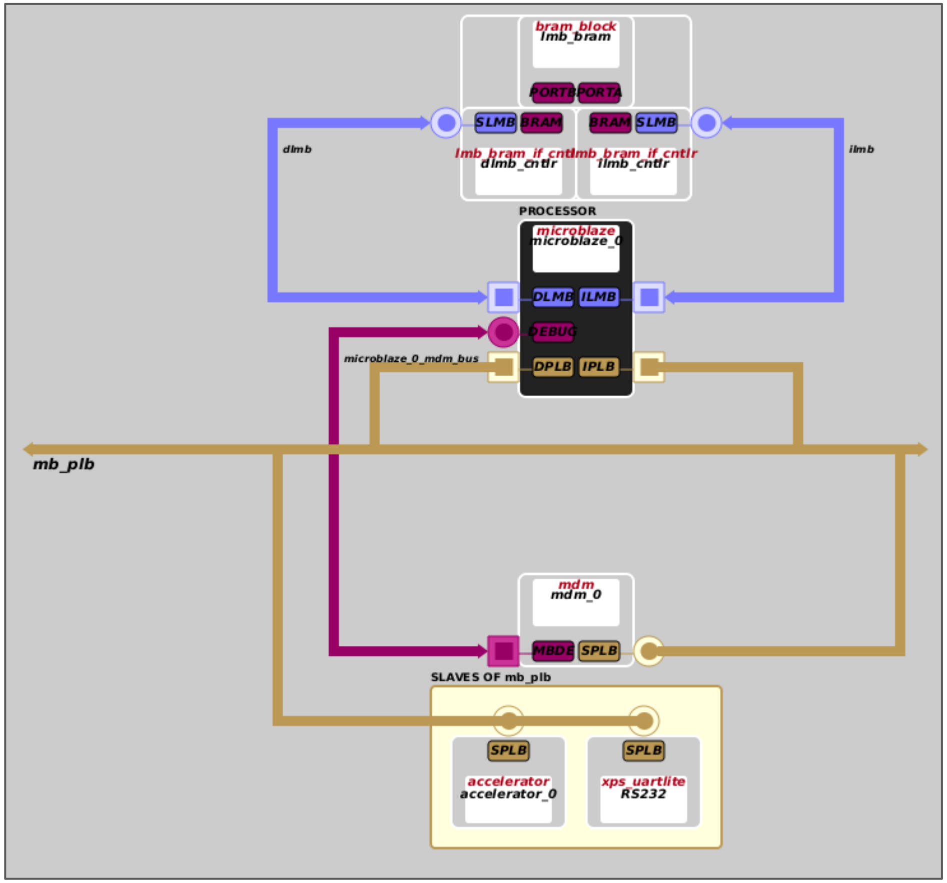
🟦 JPEG encoder 🟩 JPEG decoder 🟥 Shared actors



Designs Under Tests: Xilinx Virtex-5 330 board



Microblaze + 7 (4 inputs and 3 outputs) point-to-point links (Fast Simplex Links, FSLs) + Stream-Based Coprocessor (s-sys) + Local Bus (to access memory and other peripherals)



Microblaze + Memory-Mapped Coprocessor (mm-sys) + Local Bus (to access memory and peripherals, including mm-sys)

Architectural Results

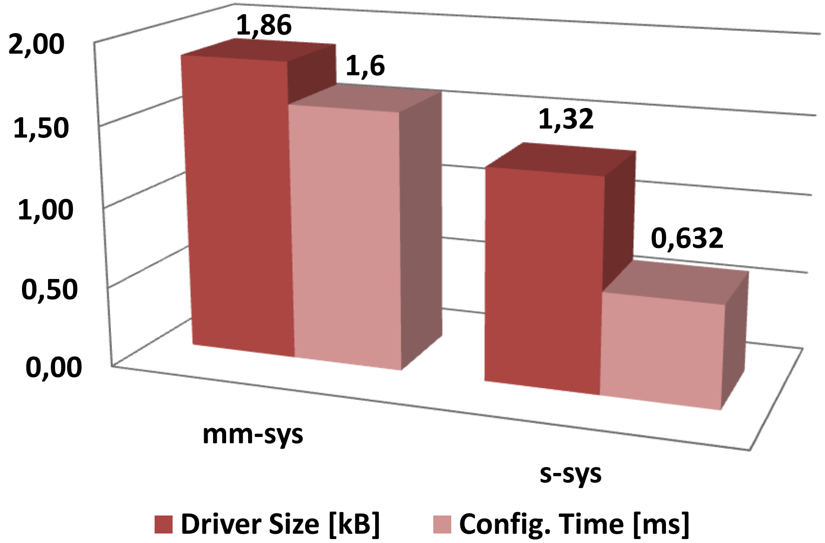
s-sys and mm-sys:
Frequency 57.8 MHz

⇨

critical path determined by the coarse-grained reconfigurable computing core

Architectural Results

s-sys and mm-sys: Frequency 57.8 MHz



critical path determined by the coarse-grained reconfigurable computing core

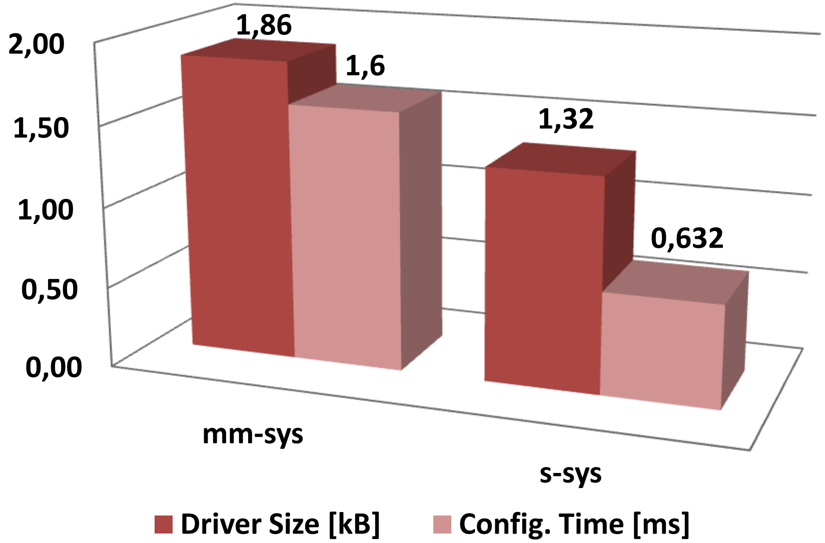
s-sys: no need for the I/O address configuration phase

less information have to be accessed and managed

Architectural Results

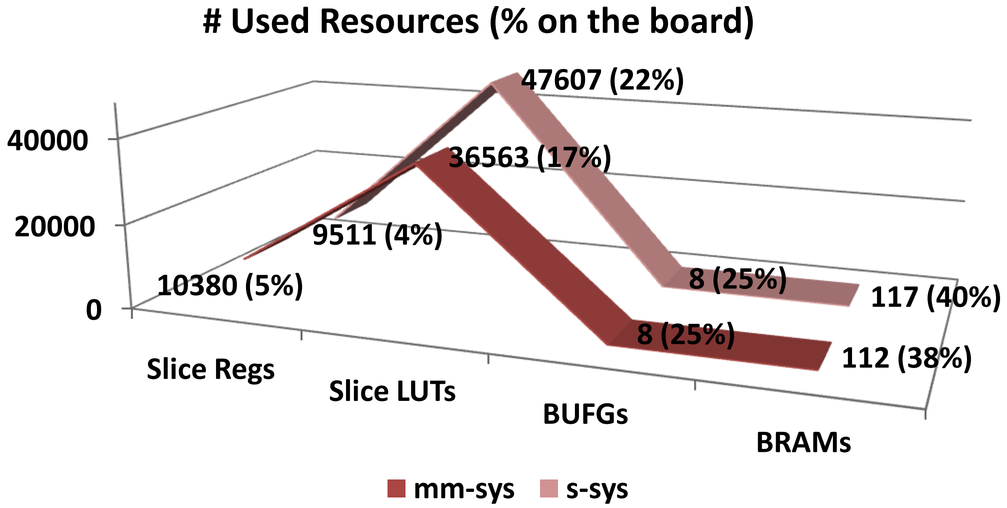
s-sys and mm-sys: Frequency 57.8 MHz

critical path determined by the coarse-grained reconfigurable computing core



s-sys: no need for the I/O address configuration phase

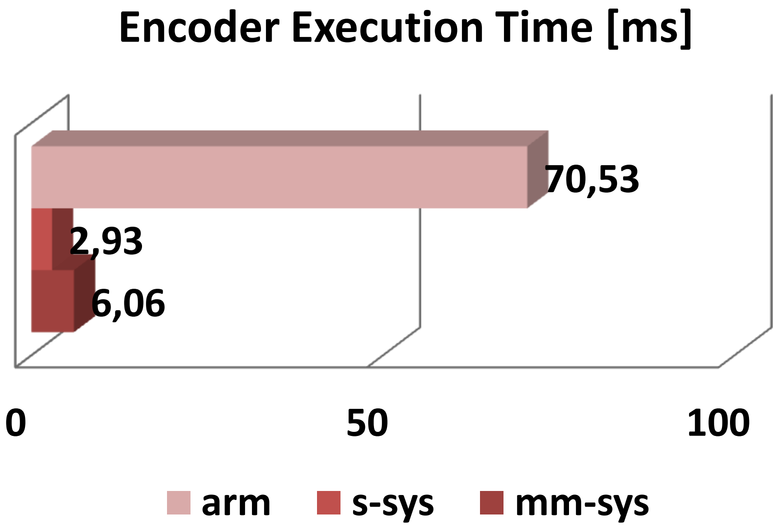
less information have to be accessed and managed

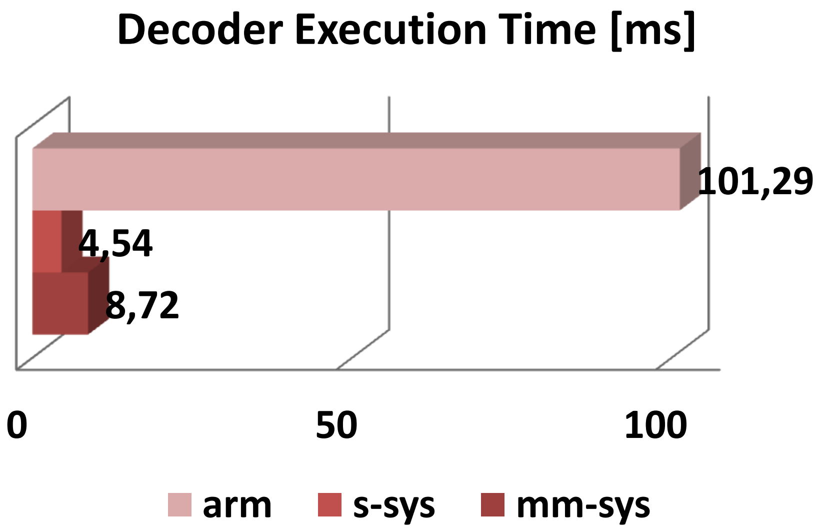


s-sys: 7 dedicated communication channels

necessary resource overhead

Performance Results



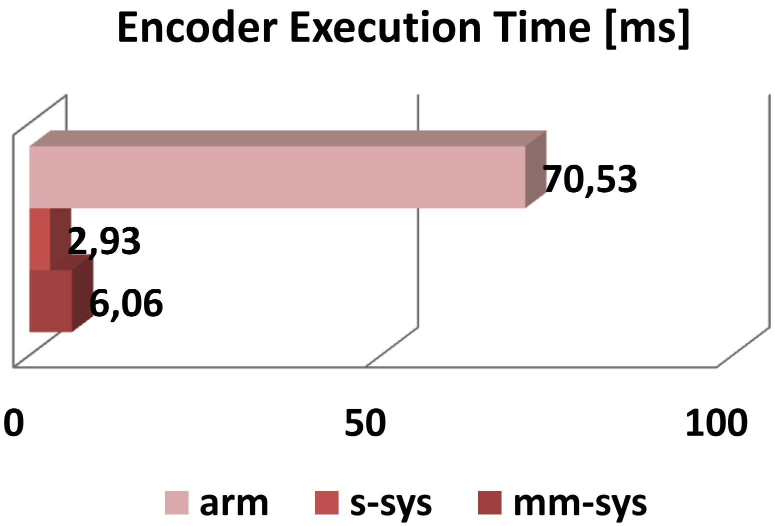


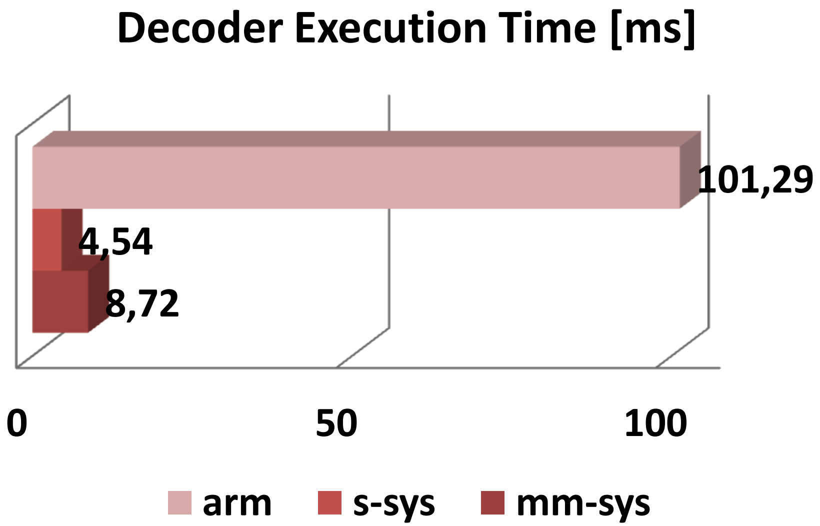
s-sys vs. mm-sys: parallel loading and storing of the I/O ports

⇩

halved execution latency

Performance Results





s-sys vs. mm-sys: parallel loading and storing of the I/O ports

⇩

halved execution latency

arm: C++ code automatically synthesized from the MPEG-RVC networks of the JPEG codec with Xronos

⇩

mm-sys and the s-sys: consistent speed-up, despite the smaller operating frequency [57.8 MHz vs 666.67 MHz]

Outline

- Introduction:
  - Problem statement
  - Background
  - Goals
- Co-processing units generation:
  - Coarse-Grained reconfiguration
  - Tool flow
  - Available co-processing layers
- Performance assessment
  - Use-case scenario
  - Results
- Final remarks and future directions

Conclusions and Perspectives

- Coarse-grained reconfigurable coprocessors are valuable and viable solutions to achieve flexibility and high efficiency, but:
  - mapping different computational requirements over the same substrate it is not straightforward;
  - debug and design effort increment with the number of requested kernels to successfully deploy an efficient multi-functional IP.

Conclusions and Perspectives

- Coarse-grained reconfigurable coprocessors are valuable and viable solutions to achieve flexibility and high efficiency, but:
  - mapping different computational requirements over the same substrate it is not straightforward;
  - debug and design effort increment with the number of requested kernels to successfully deploy an efficient multi-functional IP.
- Targeting a Xilinx FPGA technology, we proposed an automated flow to accomplish:
  - the automatic mapping of the different high-level specifications into a unique multi-functional one (***MDC baseline***);
  - the high-level-synthesis and composition of a coarse-grained reconfigurable datapath capable of executing the different kernels (***MDC baseline***);
  - the easy integration of a custom stand-alone IP and its drivers, to be used on the vendor environment (***MDC coprocessor generator extension***).

Results and Perspectives

- Experimental results highlighted the peculiarities of the available coprocessing units.

	loosely	tightly
Infrastructure constraints	☺	☹
Resource footprint	☺	😐
Performance	😐	☺

Results and Perspectives

- Experimental results highlighted the peculiarities of the available coprocessing units.

	loosely	tightly
Infrastructure constraints	☺	☹
Resource footprint	☺	😐
Performance	😐	☺

- Future developments
  - @ the framework level:
    - High-level analysis methods for the identification, at the application level, of the different kernels to be accelerated.
    - Automatic identification of the proper coupling level that will optimally serve the selected kernel.

Results and Perspectives

- Experimental results highlighted the peculiarities of the available coprocessing units.

	loosely	tightly
Infrastructure constraints	☺	☹
Resource footprint	☺	😐
Performance	😐	☺

- Future developments
  - @ the framework level:
    - High-level analysis methods for the identification, at the application level, of the different kernels to be accelerated.
    - Automatic identification of the proper coupling level that will optimally serve the selected kernel.
  - @ the architecture level:
    - Deployment of multi/hybrid accelerator environments.

Reconfigurable Coprocessors Synthesis in the MPEG-RVC Domain

Reconfigurable Platform Composer Tool Project (L.R. 7/2007, CRP-18324)
January 2012 – December 2015
http://sites.unica.it/rpct/

Francesca Palumbo
University of Sassari
PolComIng Dept. – Information Eng. Unit