

CORA-MBAD FOR ZYNQ 7000

MODEL BASED DEFINITION AND IMPLEMENTATION OF
RECONFIGURABLE COTS AVIONICS

TEC-ED & TEC-SW Final Presentation Days

12/05/2020

CORA-MBAD FOR ZYNQ 7000

OBJECTIVES

Presenters:

Tiago da Silva Jorge

Rubén Domingo Torrijos

CORA-MBAD FOR ZYNQ 7000

OBJECTIVES

CoRA-MBAD “Compact Reconfigurable Avionics - Model Based Avionics Design”

HW/SW co-design toolchain providing functionality to easily deploy functional blocks in either HW or SW implementations, from identical source models.

TASTE + GR740 + BRAVE medium FPGA

CoRA-MBAD for ZynQ 7000

Adapt said toolchain to a ZynQ 7000 SoC target, motivated by low-cost missions that will use platforms based on COTS.

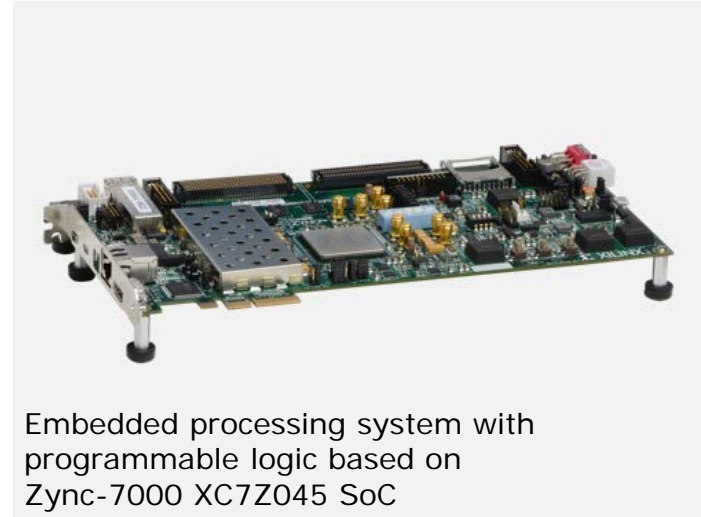
TASTE + Xilinx SoC (ARM + large FPGA)

CORA-MBAD FOR ZYNQ 7000

OBJECTIVES

To switch between HW and SW forms, the toolchain implements the **automatic transformation** of C source code (whether manually written or generated by a code generator like those in Matlab/Simulink) into Hardware (VHDL) source files.

To perform an automatic generation of the needed **consistent communication interfaces** supporting the exchange of commands and data between functional blocks executed on the processing system (PS) and on the programmable logic (PL) sides of the Xilinx SoC.



Embedded processing system with programmable logic based on Zync-7000 XC7Z045 SoC

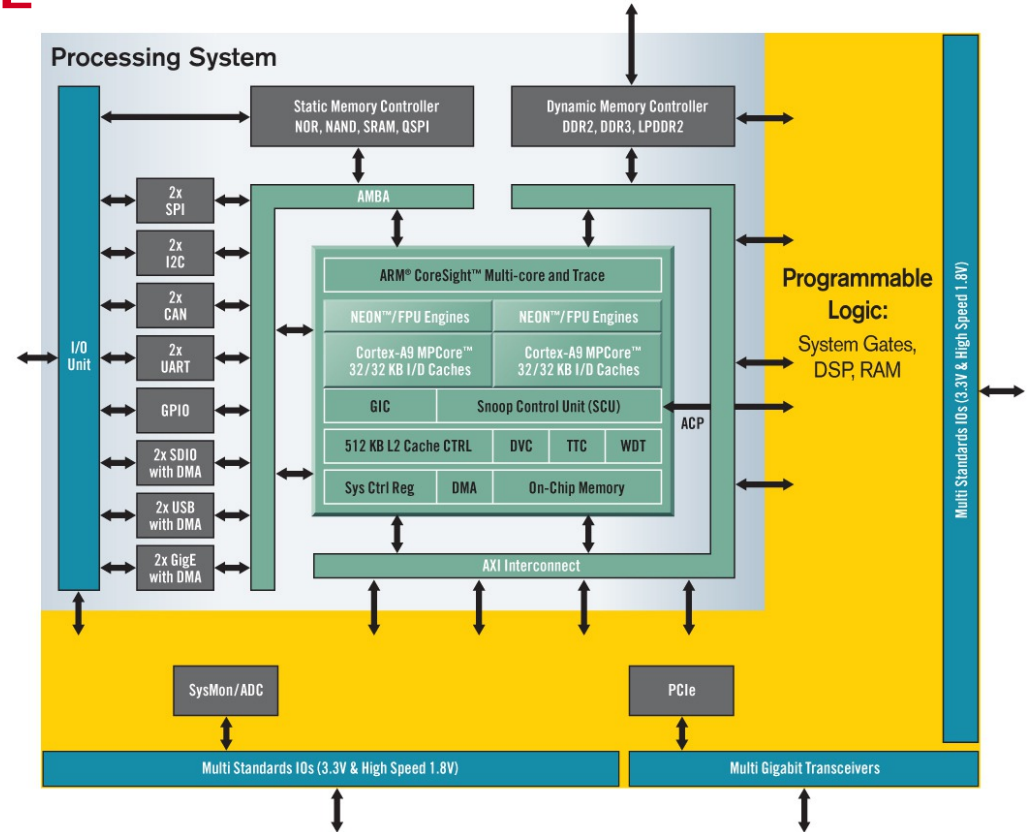
- PS: Dual ARM Cortex-A9 core processors
- PL: Artix-7 FPGA
- 1 GB of embedded DDR3 memory
- Transceivers
- FMC connector
- PCI Express, Ethernet, general purpose IO and UART interfaces

CORA-MBAD FOR ZYNQ 7000

ZYNQ-7000 ARCHITECTURE

SoC with programmable hardware and software parts

- Allow implement hardware/software co-design in a single device
- AMBA AXI interfaces between PS and PL
- 32 or 64 bits high bandwidth communication between PS and PL
- 32 or 64 bits direct high bandwidth communication between DDR3 memory controller and PL
- Specific low bandwidth and complexity communication with AXI_LITE (very useful for control registers)



CORA-MBAD FOR ZYNQ 7000

TOOLING

CORA-MBAD FOR ZYNQ 7000 TOOLING

TASTE provides heterogeneous application level modelling and implementation facilities, and importantly transparent and robust middleware level automation capabilities, in particular for communication aspects.

<https://taste.tools>



Hands-on videos: Simulink

A nice way to see what TASTE does, is to watch it applied hands-on, in building an application from scratch.

In this video, you'll see it being used to automatically integrate C code with a Simulink design.

Hands-on videos:
FPGAs

This video showcases how TASTE automatically creates device drivers and VHDL skeletons for the HW parts of a system.



CORA-MBAD FOR ZYNQ 7000 TOOLING

The toolchain was adapted to leverage the latest TASTE enhancements: The **TASTE's Kazoo** tool was adapted to build the modelled systems with significantly increased build performance, especially in rebuilds. It efficiently produces derived models, code and scripts using AdaCore's "templates-parser" for templates processing and files generation.

TASTE's Kazoo allows for simple expansion and update of the supported targets.

<http://taste.tuxfamily.org/wiki/index.php?title=Kazoo>

Basic usage

Kazoo is integrated in TASTE as one of the main tools for generating and building the code of the system under development. The calls to kazoo are normally a project using Kazoo, user normally run a single command:

```
$ taste
```

If user would like to use kazoo directly, or tweak it's behaviour, the following example Makefile presents the calls to kazoo that generate the code skeletons and

```
KAZOO?=kazoo

all:    c

c:      work/glue_built
        $(MAKE) -C work

skeletons:    InterfaceView.aadl DataView.aadl
              $(KAZOO) --gw -o work
              $(MAKE) -C work dataview

work/glue_built:    InterfaceView.aadl DeploymentView.aadl DataView.aadl
                  $(KAZOO) -p --glue --gw -o work
                  touch work/glue_built

clean:
        $(MAKE) -C work clean

.PHONY: clean skeletons c
```

Kazoo will read the default input AADL models and generate code to the selected output directory. Parameter `--output` chooses the output directory for generation, `--glue` enables glue code generation and `-p` or `--polyorb-hi-c` turns on use of PolyORB-HI-C runtime.

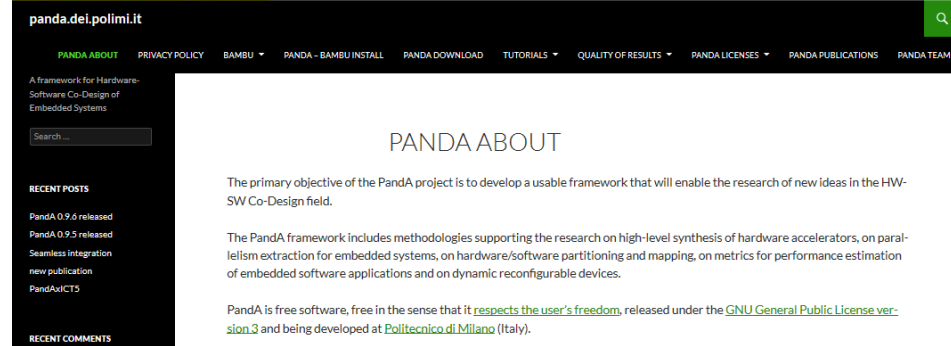
In the result directory `work` will be created and filled with generated models and source files.

CORA-MBAD FOR ZYNQ 7000 TOOLING

For Matlab/Simulink models, the MBAD System relies on model-to-code transformation performed by MathWorks **Embedded Coder** and on high-level synthesis of C code performed by **Bambu**.

*Note that both **TASTE** and **Bambu** are open-source SW tools, so subsystems built in pure C can be synthesized and executed on the FPGA with no external dependencies. **Bambu** is FPGA vendor independent, hence it can be used with minor adaptations needed for each FPGA specific component.*

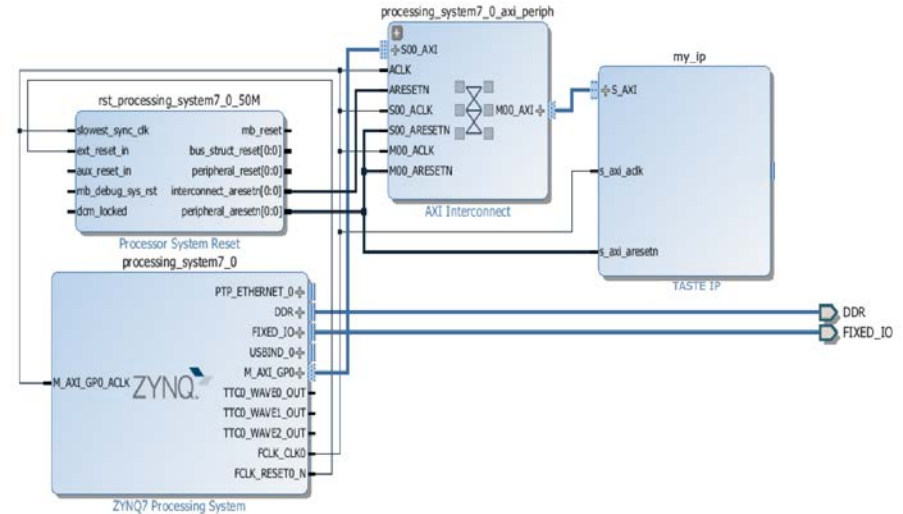
<https://www.mathworks.com/products/embedded-coder.html>
<https://panda.dei.polimi.it/>



CORA-MBAD FOR ZYNQ 7000 TOOLING

Vivado is an EDA (Electronic Design Automation) tool for FPGA and SoC, developed by Xilinx, with capabilities for low and high level synthesis, bitstream generation, timing analysis, simulation, etc.

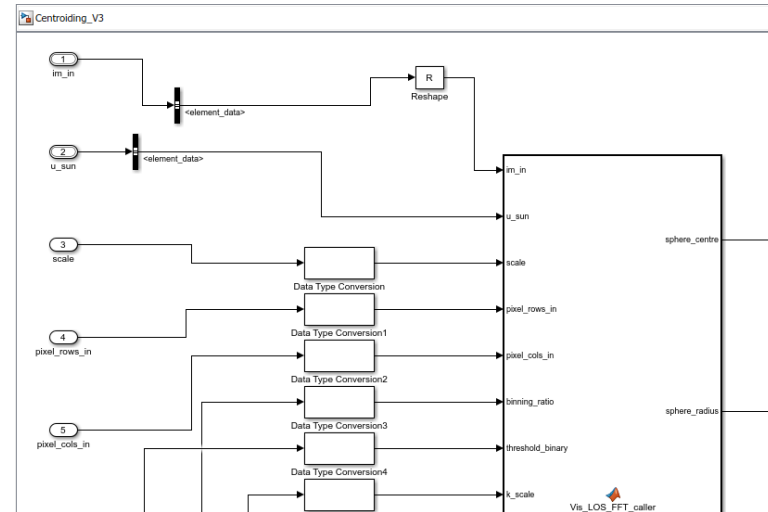
<https://www.xilinx.com/products/design-tools/vivado.html>



CORA-MBAD FOR ZYNQ 7000 TOOLING

Matlab Simulink commonly used by domain engineers to design dynamic systems, e.g. the control and guidance of satellites designed by a GNC team, producing cyclically actuation data from sensor data, are best modelled with mathematics, data flows or functional models. This environment is extensible to e.g. incorporate as well autocoding facilities such as Embedded Coder.

<https://www.mathworks.com/products/simulink.html>

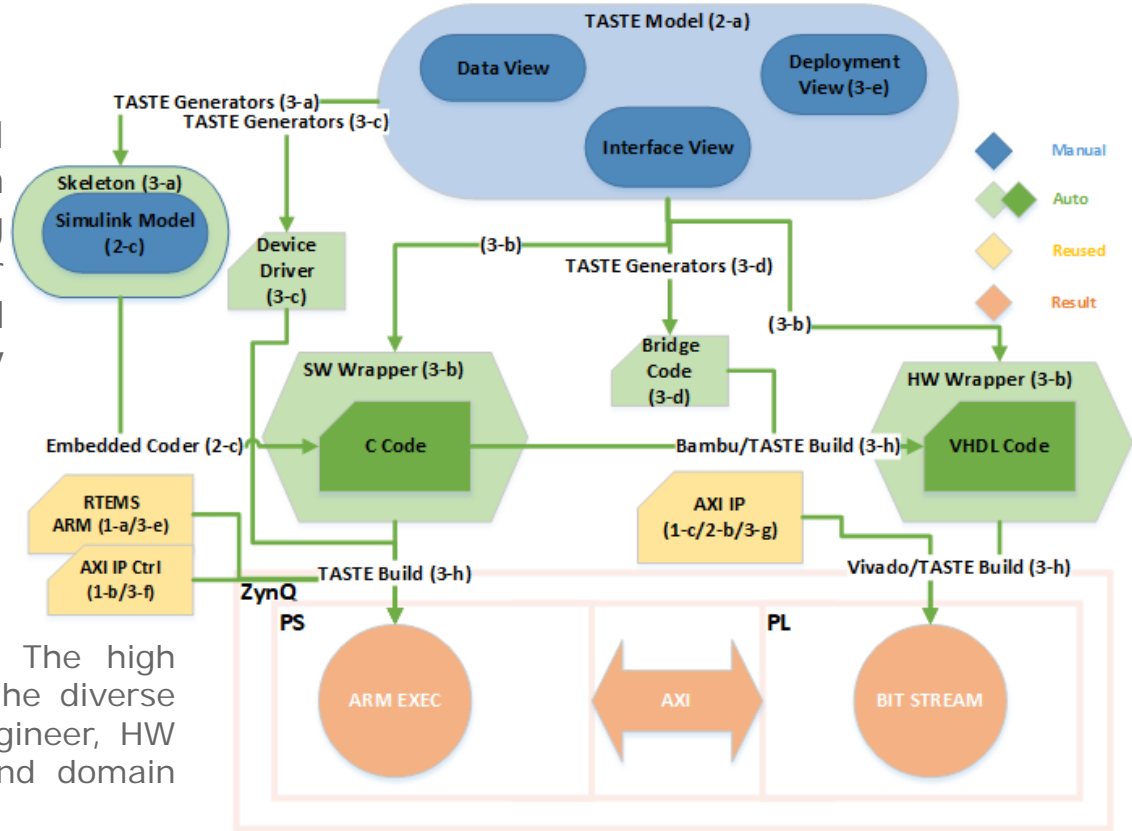


CORA-MBAD FOR ZYNQ 7000 TOOLING

A **pivot open toolchain** gluing all elements together: **TASTE**, being an open framework targeting heterogeneous systems, is particular suitable to **integrate and orchestrate** all the other necessary elements.



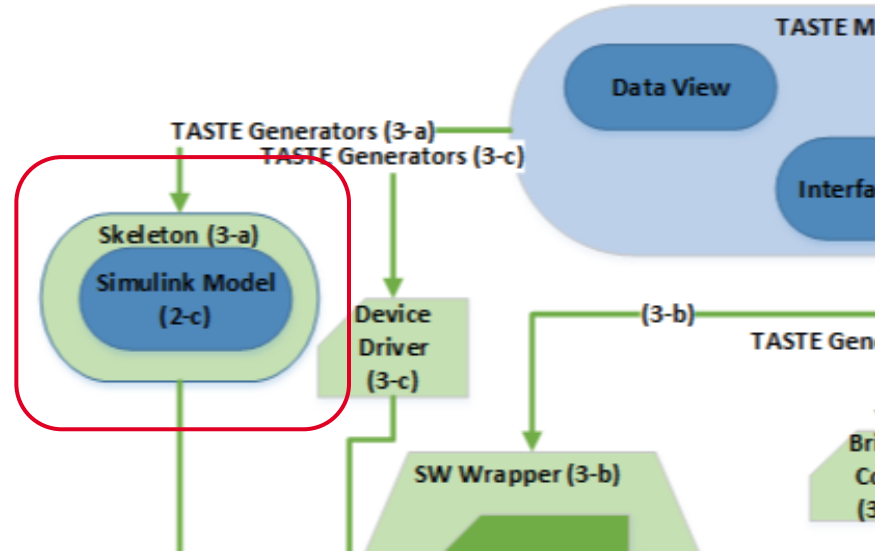
Multifaceted team in co-engineering: The high technical degree of the activity required the diverse skills and close collaboration of a: SW engineer, HW engineer, design environment engineer, and domain engineer.



CORA-MBAD FOR ZYNQ 7000 TOOLING

From a common ASN.1 data model and an AADL minimalistic component interface model it consistently and automatically exports:

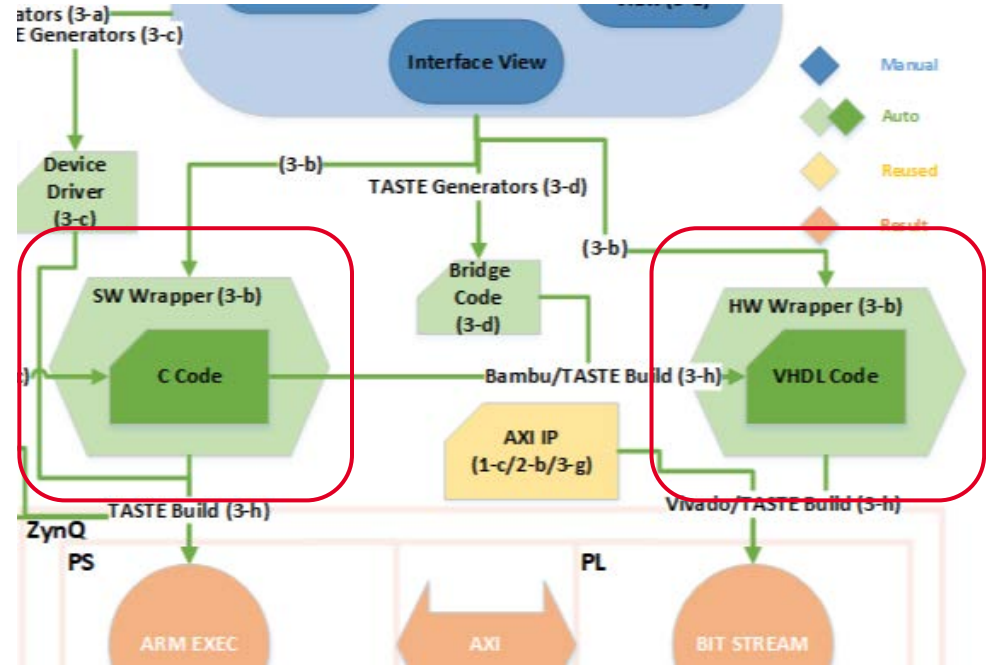
- **Interface definition** in the target language of choice with consistent inputs and outputs (in our demonstrator a Simulink model)



CORA-MBAD FOR ZYNQ 7000 TOOLING

From a common ASN.1 data model and an AADL minimalistic component interface model it consistently and automatically exports:

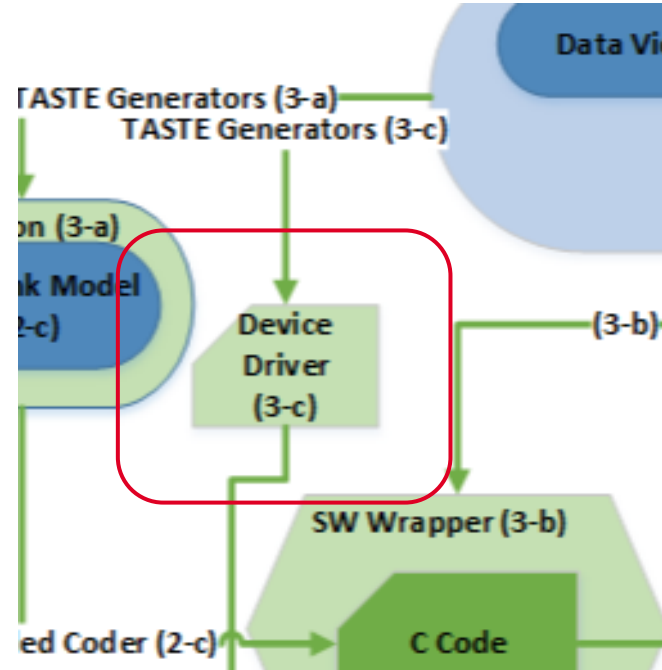
- **SW and HW wrapper** interface code that transparently guarantees the correct communication between the target's functions. Importantly these interface wrappers automatically grow or shrink according to the number and type of inputs and outputs.



CORA-MBAD FOR ZYNQ 7000 TOOLING

From a common ASN.1 data model and an AADL minimalistic component interface model it consistently and automatically exports:

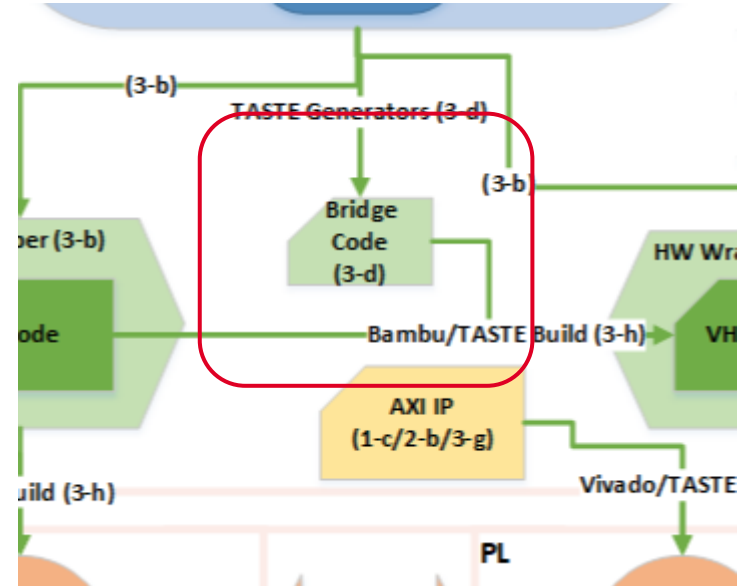
- **SW device driver** to provide SW-HW communication with the HW implementation of the target function.



CORA-MBAD FOR ZYNQ 7000 TOOLING

From a common ASN.1 data model and an AADL minimalistic component interface model it consistently and automatically exports:

- **“Bridge” code** with the necessary adaptations and extra inputs needed in the transition between two autocoding tools, in this case between Embedded Coder and Bambu.

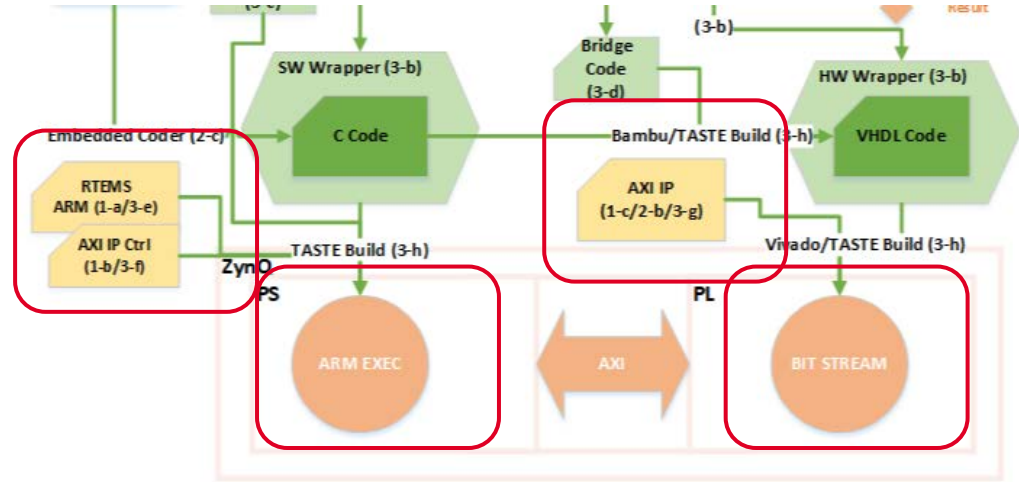


CORA-MBAD FOR ZYNQ 7000

TOOLING

Additionally, TASTE:

- integrates the custom **cross-compiler** as part of a new deployment target
- links with the **BSP** to use the necessary **HW drivers**
- orchestrates the calls to all needed **autocode and compilation tooling** - e.g. forwarding the Embedded Coder output as a Bambu input together with the generated consistent bridge
- calls the **synthesis, placement and routing** facilities of Vivado and generates FPGA bitfiles



CORA-MBAD FOR ZYNQ 7000

ARM SUPPORT

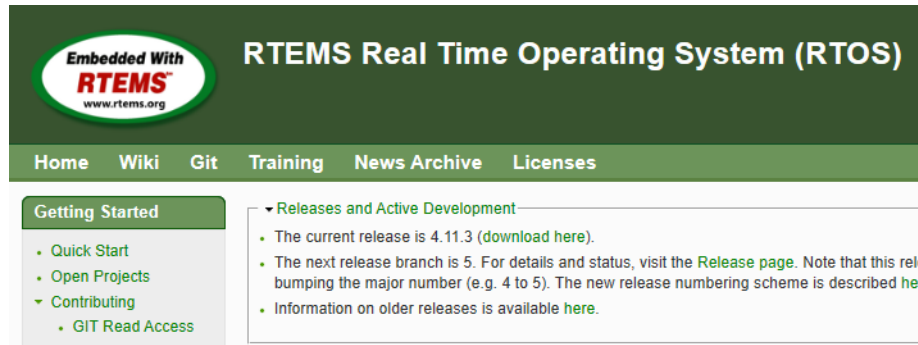
CORA-MBAD FOR ZYNQ 7000

ARM SUPPORT

Support for the **ARM Cortex A9** development toolchain (RTEMS ARM support)

The **RTEMS 5.1** custom built cross compiler for ARM Cortex A9, equipped with the needed BSPs and validated on target. Testing revealed timing issues, so **BSP customization** had to take place to properly configure clock frequency.

<https://lists.rtems.org>
<https://www.rtems.org/>



Embedded With RTEMS
www.rtems.org

RTEMS Real Time Operating System (RTOS)

Home Wiki Git Training News Archive Licenses

Getting Started

- Quick Start
- Open Projects
- Contributing
 - GIT Read Access

Releases and Active Development

- The current release is 4.11.3 ([download here](#)).
- The next release branch is 5. For details and status, visit the [Release page](#). Note that this release bumps the major number (e.g. 4 to 5). The new release numbering scheme is described [here](#).
- Information on older releases is available [here](#).

CORA-MBAD FOR ZYNQ 7000

ARM/FPGA COMMUNICATION SUPPORT

CORA-MBAD FOR ZYNQ 7000

ARM/FPGA COMMUNICATION SUPPORT

Support for the **Advanced eXtensible Interface (AXI)** communication interface for on-chip communication.

(SW) AXI IP core control driver providing the necessarily interface configuration, initialization and read/write access to the SW applications.

Support for the **Xilinx FPGA** development toolchain (**Vivado**)

(HW) An AXI interconnect IP Core to manage and connect the AXI ports in the PS with the AXI ports of each of the modules/IPs implemented in the PL. At the same time, an AXI DMA IP controller to manage stream data transmissions between PS and PL.

CORA-MBAD FOR ZYNQ 7000

ARM/FPGA COMMUNICATION SUPPORT

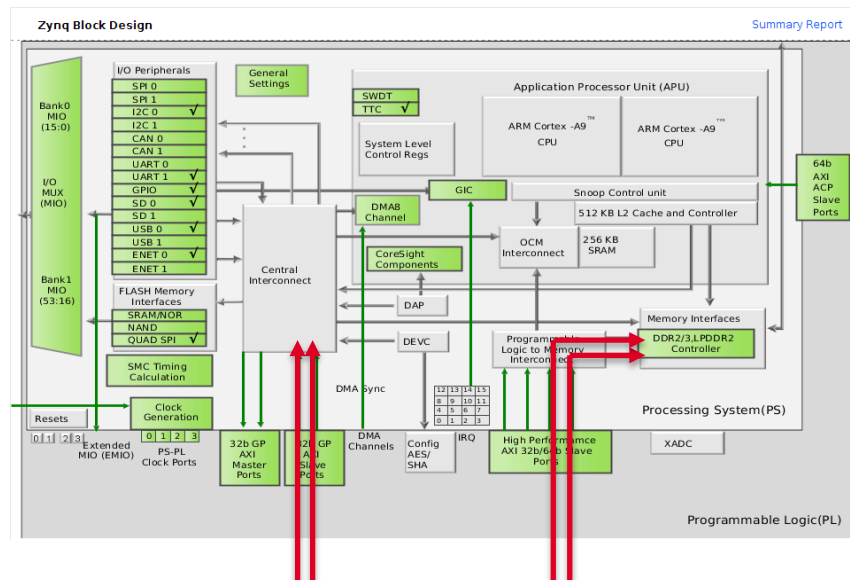
CoRA-ZynQ makes use of AXI bridges of Zynq-7000 architecture to connect PS with PL.

Three independent interfaces are implemented in order to provide different capabilities:

- one AXI interface used to write and read configuration registers
- one AXI interface fully devoted to write and read large blocks of memory inside FPGA
- one AXI stream interface to support stream data processing

The number of registers or memories addressed through AXI interfaces can be configured to optimize the resource allocation of the FPGA.

In addition, AXI stream transmission can be directed to achieve up to 32 different destinations through the same interface.



TASTE WRAPPER

CORA-MBAD FOR ZYNQ 7000

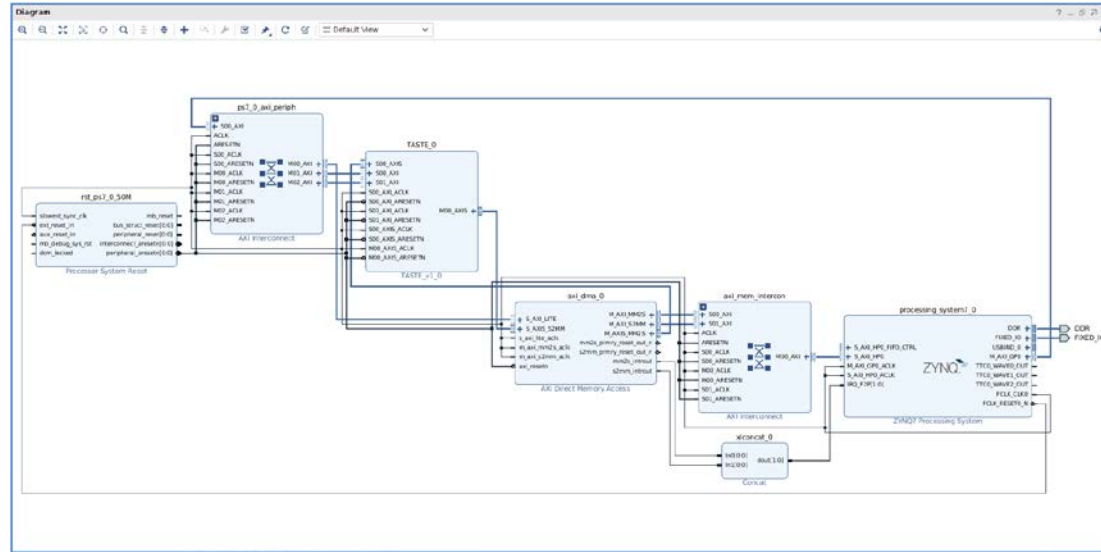
ARM/FPGA COMMUNICATION SUPPORT

The TASTE wrapper presents these capabilities:

- Used for IP configuration, operations with different type variables and stream processing
- 2 AXI-LITE Slave interface
- 1 AXI-STREAM Slave interface
- 1 AXI-STREAM Master interface
- Support for 1k memories of 1Mx32 bits size used to implement arrays (limited by FPGA resources)
- Support for arrays of different sizes
- Support up to 32 input FIFOs and 32 output FIFOs used to implement pointers (limited by FPGA resources)
- Support stream processing through DMA

Constraints:

- Arrays limited by code to 1048576 elements of 32 bits
- Pointers limited by code to use elements of 32 bits
- Transmission burst limited by code to 1024 elements



CORA-MBAD FOR ZYNQ 7000 USE CASES

CORA-MBAD FOR ZYNQ 7000

USE CASES

The use cases implemented have as prime objectives to:

- 1) demonstrate the toolchain new target support
- 2) support a space representative application

Objective 1 was fully achieved with simple use cases.

Objective 2 is presently partially achieved with work still ongoing.

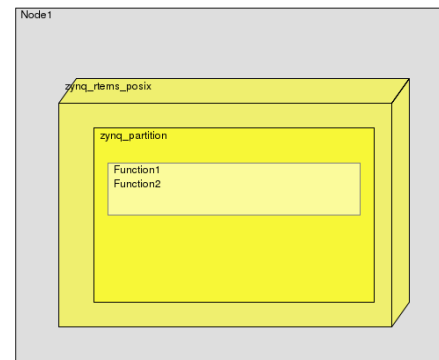
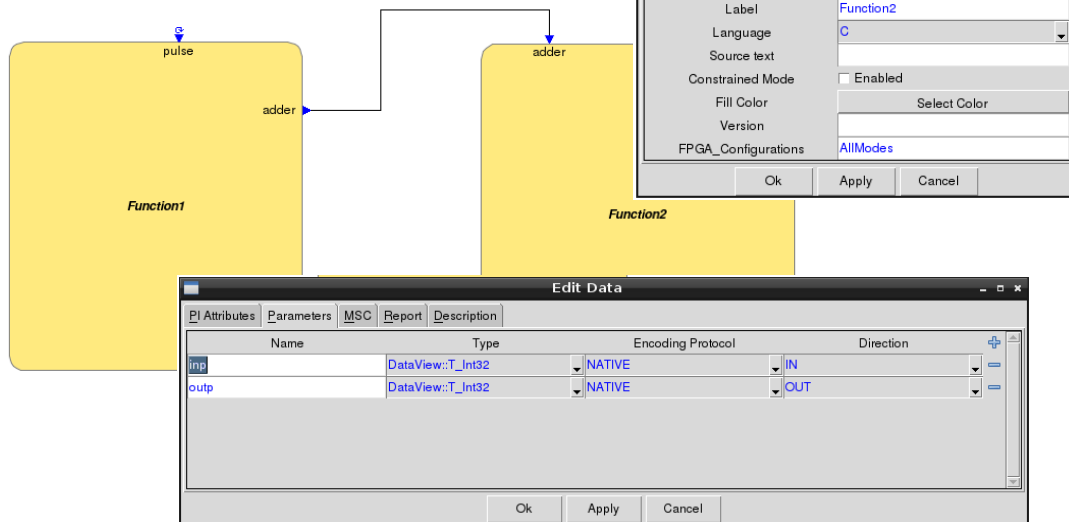
A preliminary version of the HERA mission computer-vision Lambertian sphere matching of asteroid body algorithm was reused in this context.

CORA-MBAD FOR ZYNQ 7000

USE CASES

Simple Use Case – Prime numbers

- Compute prime numbers from C code, on Zynq-7000:
 - Caller function in C, running on ARM.
 - Compute function in C, running on FPGA.



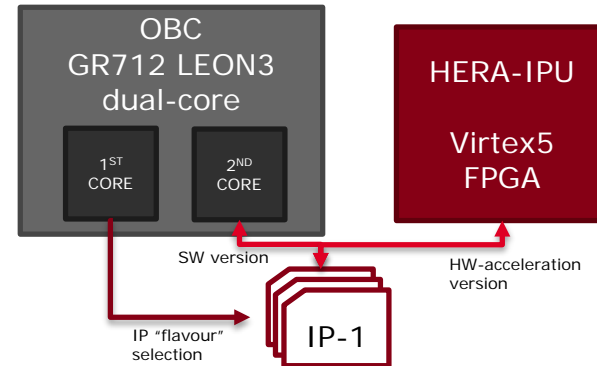
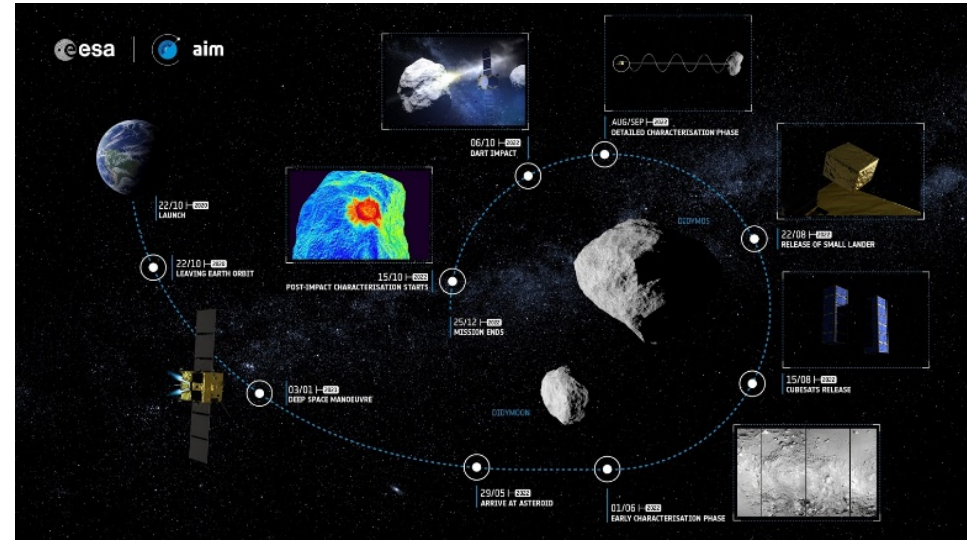
```
void function2_PI_adder
(const asnlSccT_Int32 *IN_inp,
 asnlSccT_Int32 *OUT_outp)
{
    // Write your code here
    if( *IN_inp % 2 == 0){
        *OUT_outp = 2;
        return;
    } else {
        for (int i=3; i <= *IN_inp / 2; i+=2){
            if(0 == (*IN_inp % i)){
                *OUT_outp = i;
                return;
            }
        }
        *OUT_outp = *IN_inp;
    }
}
```

```
[function1_PI_pulse] Sent 8, Got 2
[function1_PI_pulse] Sent 9, Got 3
[function1_PI_pulse] Sent 10, Got 2
[function1_PI_pulse] Sent 11, Got 11
[function1_PI_pulse] Sent 12, Got 2
[function1_PI_pulse] Sent 13, Got 13
[function1_PI_pulse] Sent 14, Got 2
[function1_PI_pulse] Sent 15, Got 3
[function1_PI_pulse] Sent 16, Got 2
[function1_PI_pulse] Sent 17, Got 17
[function1_PI_pulse] Sent 18, Got 2
[function1_PI_pulse] Sent 19, Got 19
```

CORA-MBAD FOR ZYNQ 7000 USE CASES

HERA mission

- Autonomous Vision-Based Navigation
- Proximity operations
- Computer-vision algorithms are computationally costive in terms of execution time and memory
 - Offload SW Processor
 - Acceleration by means of FPGA
 - Large data to be processed 1024*1024 pixels
 - Real-Time high-performances
- HW-accelerated functions
 - Computer-vision Lambertian sphere matching of asteroid body
 - Feature Detection and Tracking surface terrain of asteroid
 - Interchange SW functions with FPGA ones and among them

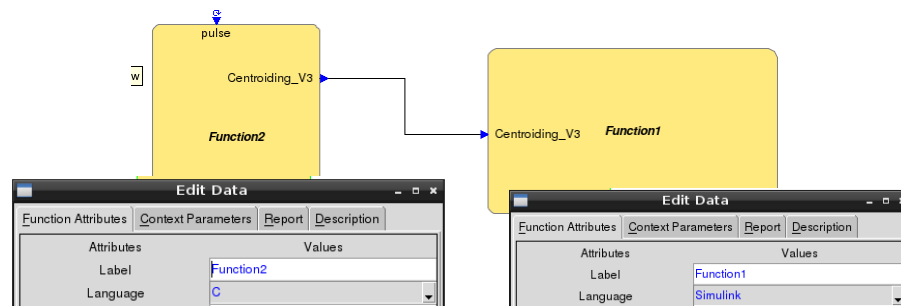
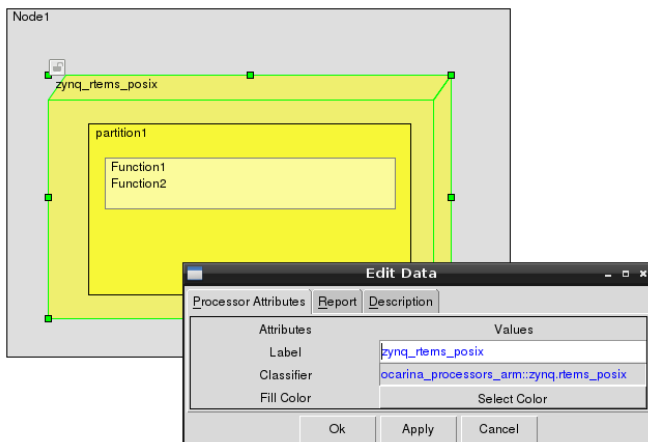


CORA-MBAD FOR ZYNQ 7000 USE CASES

HERA mission

TASTE model designed for Use Case algorithm integration:

- ASN.1 Data types tailored to use case IOs.
- Interface View with a calling interface to the Simulink generated algorithm.
- Deployment on Zynq-7000 target.



The screenshot shows the 'Edit Data' dialog box with the 'Parameters' tab selected. The table below lists the parameters of the function block.

Name	Type	Encoding Protocol	Direction
in_in	DataView:MyOctStr	NATIVE	IN
u_sun	DataView:MyRealSeq3	NATIVE	IN
scale	DataView:MyReal	NATIVE	IN
pixel_rows_in	DataView:MyReal	NATIVE	IN
pixel_cols_in	DataView:MyReal	NATIVE	IN
binning_ratio	DataView:MyReal	NATIVE	IN
threshold_binary	DataView:MyReal	NATIVE	IN
k_scale	DataView:MyRealSeq5	NATIVE	IN
k_scale_step	DataView:MyReal	NATIVE	IN
auto_k_scale	DataView:MyReal	NATIVE	IN
radius_peak	DataView:MyReal	NATIVE	IN
margin_radius_rel	DataView:MyReal	NATIVE	IN
COB_diff_thr	DataView:MyReal	NATIVE	IN
CAM_FOV	DataView:MyReal	NATIVE	IN
asteroid_real_diameter	DataView:MyReal	NATIVE	IN
sphere_centre	DataView:MyRealSeq2	NATIVE	OUT
sphere_radius	DataView:MyReal	NATIVE	OUT
cent_valid_flg	DataView:T_Boolean	NATIVE	OUT
apparent_altitude	DataView:MyReal	NATIVE	OUT

CORA-MBAD FOR ZYNQ 7000

USE CASES

HERA mission

The Matlab design reused is not tailored for a HW implementation (e.g. no parallel nor fixed-point design) which naturally represents some challenges to the autocoding facilities and HW resource usage.

Such tailoring was not yet performed due to project scope and time availability.

SW RUN



```
[[Function2] Startup...  
[function1] Centroiding_V3_initialize  
[Function2] PI_pulse  
[function1] Centroiding_V3_step  
[Function2] PI_pulse done!  
[Function2] Elapsed Time - 2.005620539 s  
Centroiding_V3 done with iteration 0  
    sphere_centre      {520.000000, 500.000000}  
    sphere_radius      201.791794  
    cent_valid_flg     TRUE  
    apparent_altitude  22115.820312
```

Expected sphere_radius 201.791794 (0x40693956605ee569), got 201.791794 (0x4069395660000000)

Expected apparent_altitude 22115.820313 (0x40d598f4800218df), got 22115.820312 (0x40d598f480000000)

CORA-MBAD FOR ZYNQ 7000

USE CASES

HERA mission

Targeting **prototyping** activities, the present approach is instead leveraging to the maximum possible extent the **configurability and autocoding strengths of the toolchain**, avoiding any manual work, e.g. by exploring:

- the rich Embedded Coder and Bambu **options**
- types of possible SW-HW **interfaces** generated (e.g. external memory access, streaming type parameters) and resulting HW resource allocation.

HW synthesis successfully achieved

some Bambu options relevant:

bambu --compiler=I386_CLANG4 --experimental-setup=BAMBU-TASTE --no-iob --clock-period=10 -O2 ...

some Embedded Coder options relevant:

long long support
reference interfaces

Some metrics of the Bambu generated HW kernel:

```
<item stringID="XILINX_SLICE" value="30202"/>
<item stringID="XILINX_SLICE_REGISTERS" value="53786"/>
<item stringID="XILINX_SLICE_LUTS" value="98319"/>
<item stringID="XILINX_BLOCK_RAMFIFO" value="83"/>
<item stringID="XILINX_IOPIN" value="0"/>
<item stringID="XILINX_DSPS" value="92"/>
<item stringID="XILINX_POWER" value="0.416"/>
<item stringID="XILINX_DESIGN_DELAY" value="24.028"/>
```

Integration and testing on-going

Conclusions

- TASTE's Kazoo facilitates toolchain evolution
- Role of Design Environment Engineer becoming increasingly important
- Increased collaboration between SW and HW engineers
- Functions can be fully automatically deployed in SW and HW from a high-level domain model (e.g. Simulink)
- Generated SW-HW interface and HW interface with Bambu kernel for computation is otherwise very challenging - CoRA-MBAD allows for rapid implementation iterations
- The configuration of the autocoding tools goes a long way in making integration easier and resulting in more efficient designs

CORA-MBAD FOR ZYNQ 7000

Future work

- Address runtime reconfiguration
- Improve autocoding and synthesis results
- Trade-off analysis with manual implementations
- Address more use cases, targetting diversified purposes (e.g. speed, memory, interfaces, etc.)
- Stress test TASTE translators for robustness
- Support advanced interfaces for modern EGSE integration
- and other

Thank you

Tiago da Silva Jorge
tmdasilva@gmv.com

12/05/2020

