An heterogeneous compiler of dataflow programs for Zynq Platforms

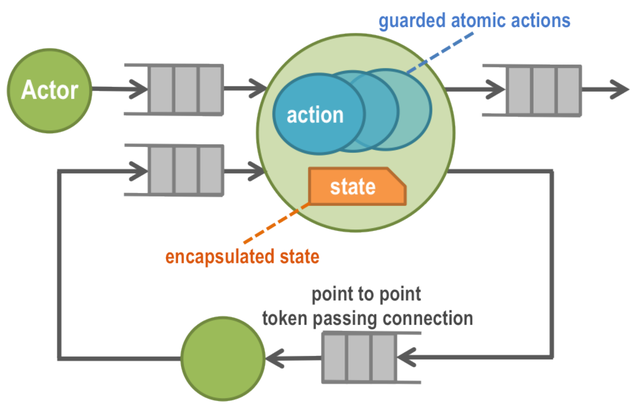
Endri Bezati[1] Simone Casale-Brunet[2] Romuald Mosqueron[3] Marco Mattavelli[2]

[1]EPFL VLSC, [2]EPFL SCI STI MM, École Polytechnique Fédérale de Lausanne, Switzerland
[3]School of Business and Engineering Vaud HES-SO, REDS Institute at HEIG-VD,
University of Applied Sciences Western Switzerland, Yverdon-les-Bains, Switzerland

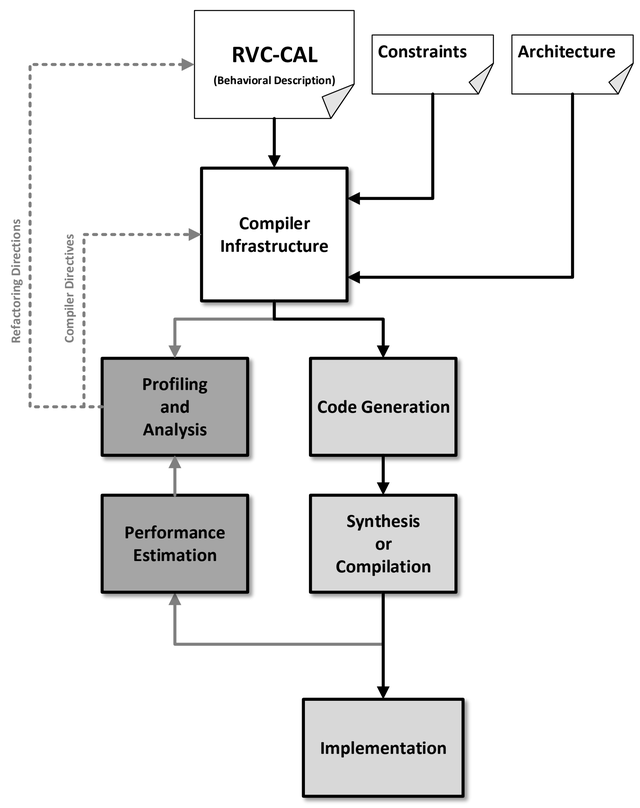
In recent years, the number and variety of heterogeneous multiprocessor system-on-chip MPSoCs, such as for instance Zynq platforms, has sensibly increased. However, today all design flow solutions capable of programming the different components of such platforms require to the designer either to modify the software or hardware based designs to obtain higher performance implementations. Thus, the developer needs to either rewrite functional blocks in HDL or to use high-level synthesis of C-like sequential languages with platform locked extensions. In this paper, a compiler infrastructure that takes as input a single behavioral representation, expressed in high-level dataflow RVC-CAL language, is proposed for programming any of the components of heterogeneous Zynq MPSoCs platforms without the need of modifying any line of code on the design.

The RVC-CAL Dataflow Language

RVC-CAL is a high level dataflow language adopted by MPEG for its Reconfigurable Video Coding Initiative. It is structured around the concepts of: **Actors**, **Actions**, **Finite State Machines** and **Procedures**.

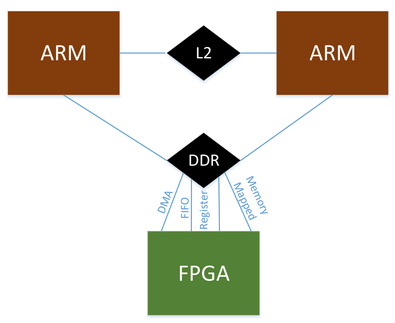


Design Flow



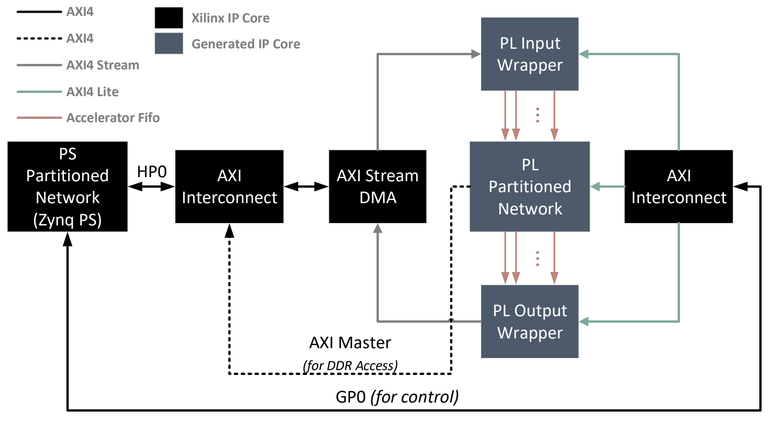
Architecture

The architecture defines on which kind of platform the design is implemented. The architecture contains **operators**, **media**, and **links**, which are described in the XCF file.

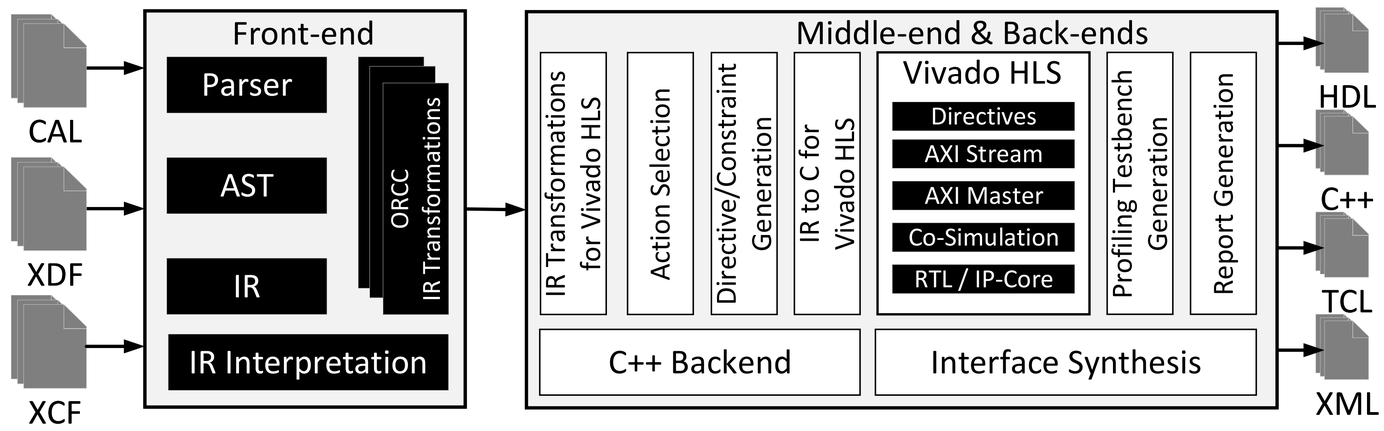


HW-SW Interfacing

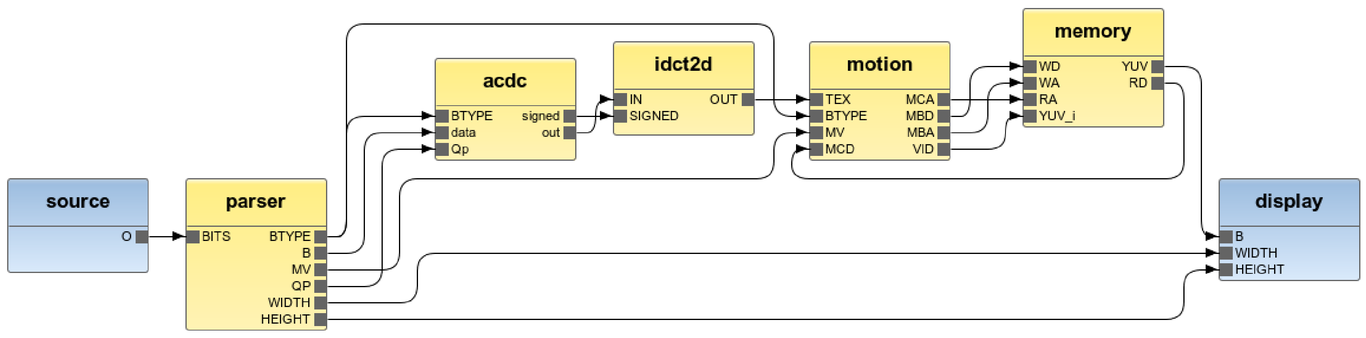
How actors communicate with each other, when partitioned between the Processing System (PS) and Processing Logic (PL) of the Zynq MPSoC.



Compiler Infrastructure - Code Generation



RVC-CAL dataflow MPEG-4 Simple Profile decoder



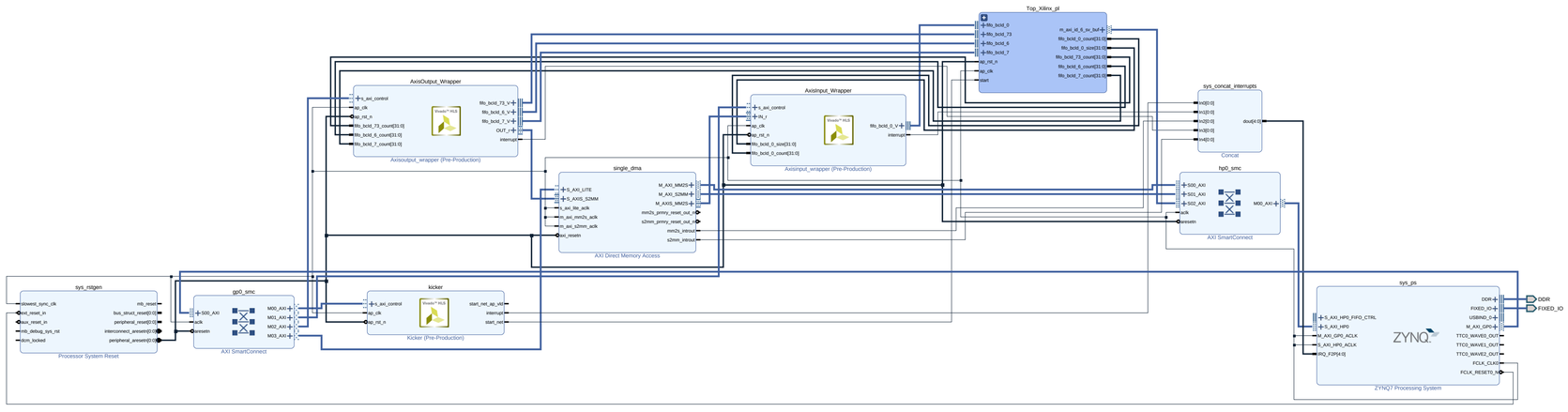
MPEG-4 SP XDF or dataflow program. Actors are represented in blue, and actors composition/networks in yellow.

	src	parser	acdc	idct2d	motion	memory	display	PL Connections	
Actors	1	5	7	12	6	2	1	**Inputs**	**Outputs**
P1	PS	PS	PS	PS	PS	PS	PS	0	0
P2	PS	PL	PS	PS	PS	PS	PS	1	6
P3	PS	PL	PL	PS	PS	PS	PS	1	6
P4	PS	PL	PL	PL	PS	PS	PS	1	5
P5	PS	PL	PL	PL	PL	PS	PS	2	6
P6	PS	PS	PL	PL	PL	PL	PS	4	1
P7	PS	PS/PL	PL	PL	PL	PL	PS	7	1
P8	PS	PL	PL	PL	PL	PL	PS	1	3

	Resources %								
Paritition	P1	P2	P3	P4	P5	P6	P7	P8	Available
LUT	0	18	22.03	27.81	31.83	29.64	32.23	34.89	53200
LUTRAM	0	4	4.66	4.63	4.82	6.37	6.4	6.4	14700
FF	0	9	10.78	14.19	16.07	16.47	17.25	17.84	106400
BRAM	0	3.62	12.44	30.8	38.57	34.93	48.93	54.29	140
DSP	0	0	1.36	4.55	4.55	4.55	4.55	4.55	220

Partitioning configuration of actors and networks between PL and PS,
and used FPGA Resources on ZC702 for different partitioning configurations.

IP Blocks implementation in Xilinx Vivado



Vivado block design representation of the P8 partition, all generated actors and interface IP blocks are instantiated automatically.

Decoder performance on the ZedBoard