Multi-standard OFDM transceiver for heterogeneous System-on-Chips

Pascal Cotret, Vipin Kizheppatt, Christophe Moy

SCEE/IETR Research group, CentraleSupélec Rennes

11 octobre 2016

pascal.cotret@centralesupelec.fr

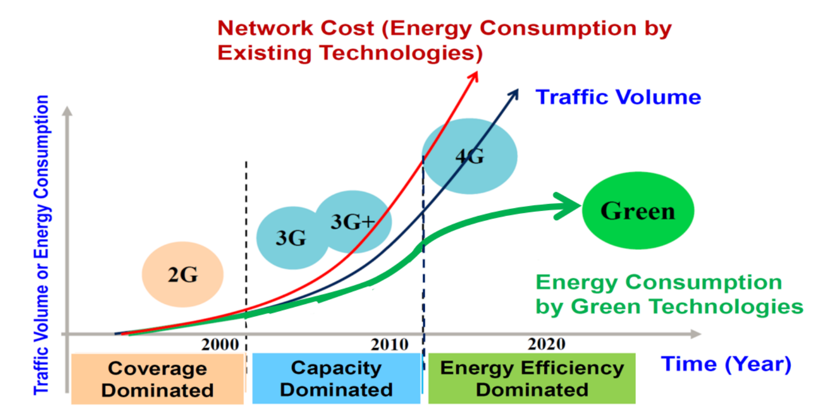
Table of contents

1. Background
2. HDCRAM on FPGA platform
3. Metrics on FPGA platform
4. OFDM transmitter and receiver example
5. Demonstration
6. Conclusion and perspectives

Background

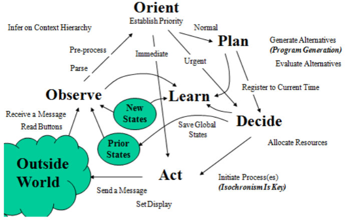
Energy efficiency

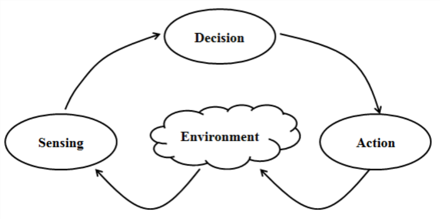
- Explosive mobile data traffic growth
- Leads to the rise of energy costs
- Consequently a significant growth of carbon emission



Cognitive radio

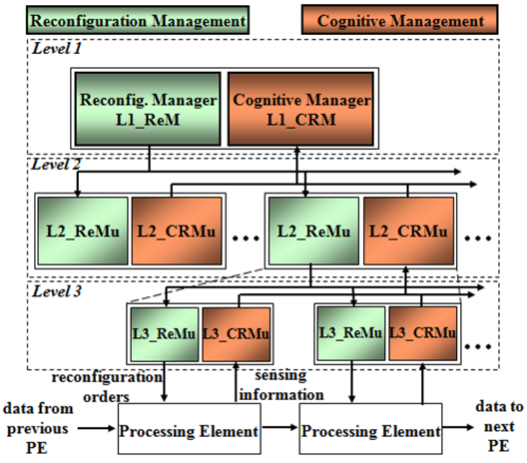
- Cognitive Radio (CR) has been considered as an enabling technology for green radio communications due to its ability to adapt its behavior to the changing environment
- The concept of cognitive radio (CR) has been first proposed by Mitola in 1999





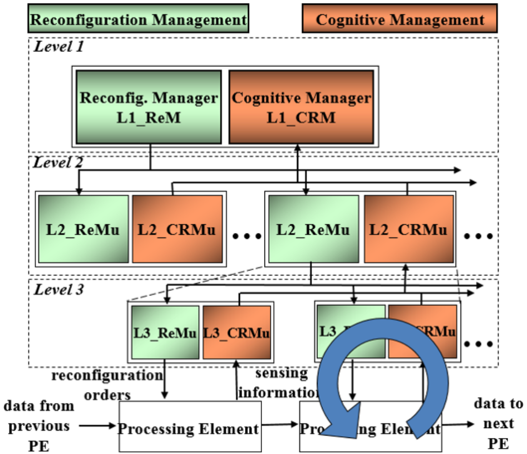
J. Palicot. *Cognitive radio : an enabling technology for the green radio communications concept*, IWCMC '09

HDCRAM : Hierarchical and Distributed Cognitive Radio Architecture Management

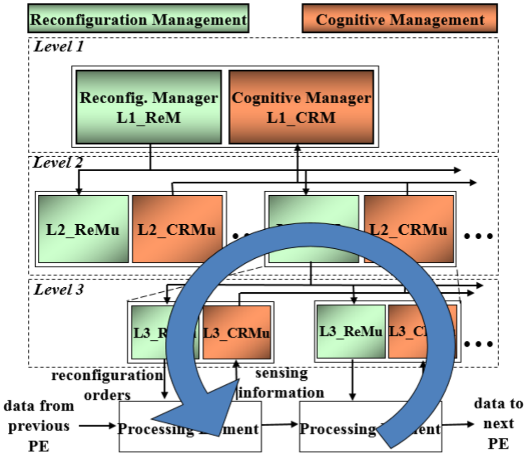


L. Godard, C. Moy, J. Palicot. *From a Configuration Management to a Cognitive Radio Management of SDR Systems*, Crowncom '06

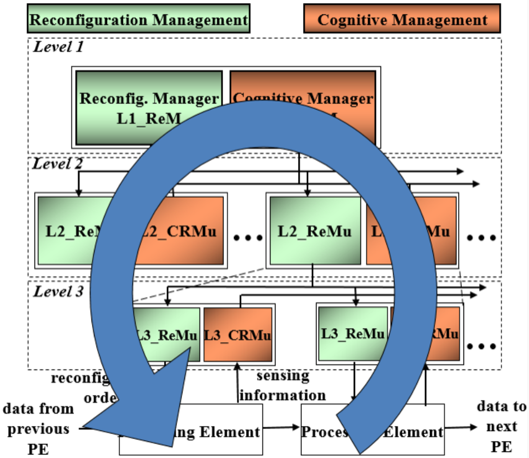
Cognitive cycles



Cognitive cycles



Cognitive cycles



Software radio architectures

Architectures	Development language	FPGA support	Runtime reconfiguration	Dynamic partial reconfiguration
GNU Radio	C++/Python	Partially supported	Partially supported	Not supported
GNU Radio + RFNoC	C++/Python/Verilog	Supported	Partially supported	Not supported
Iris	C++	Supported	Supported	Not supported
HDCRAM	C++/VHDL	Supported	Supported	Supported

M. Braun. *RFNoC : RF Network on Chip*, GNU Radio Conference 2015

P. Sutton. *Iris : an architecture for cognitive radio networking testbeds*, IEEE Communications Magazine, December 2010

Contributions

- Study of HDCRAM for CR management in green context
- Implementation of HDCRAM on two hardware platforms
- Introduction of some useful metrics on hardware platform
- Development of a hardware UDP core, which works at 125MBytes/s, to provide a high speed transmission of data and partial bitstreams
- Proof of concept management scenarios and demonstrations of HDCRAM with software/hardware co-design

HDCRAM on FPGA platform

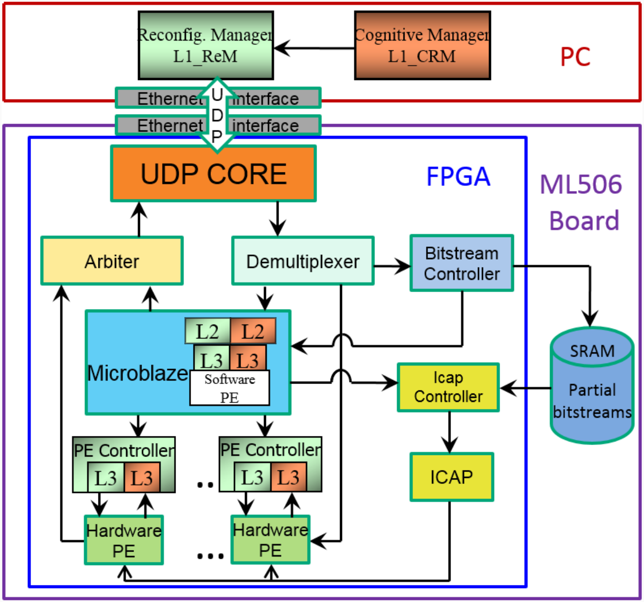
Partial reconfiguration

Partial Reconfiguration (PR) is the ability to dynamically reprogram a subset of logic in an operating FPGA by downloading a partial configuration file while the remaining logic continues to operate without interruption.

Benefits of DPR

- Flexibility
- Smart devices
- Less ressources
- Reduce reconfiguration time
- Reduce power consumption

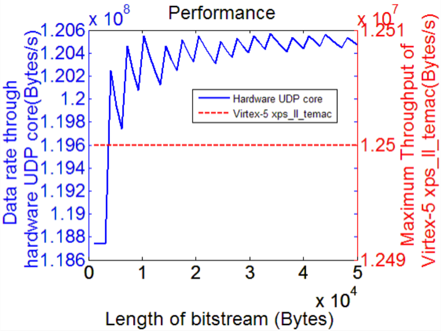
Virtex-5 platform



Downloading Speed of Partial Bitstreams through UDP

$$R_{UDP} = \frac{n}{\frac{n}{1472} \times 1526 + rem(n, 1472) + 54} \times 125 \times 10^6 \ Bytes/s$$

- n : partial bitstream size
- $rem(n, 1472)$: remainder of dividing 1472 into n
- Limit data rate : 120.6 Mbytes/s



X. Wu, P. Leray, J. Palicot. *A High Speed Approach of Downloading FPGA Partial Bitstreams through UDP for Reconfigurable SDR*, 8th Karlsruhe Workshop on Software Radios, March 2014

A. Sarangi et al. *LightWeight IP Application Examples*, November 2014

Zynq-7000 Platform

Virtex-5 ML506

- standalone application without OS
- codes on Microblaze are hardware dependent
- hard to migrate
- high power consumption, etc

Zynq platform

- running Linux on ARM
- easy to upgrade
- portable
- lower power consumption, etc

Zynq-7000 Platform

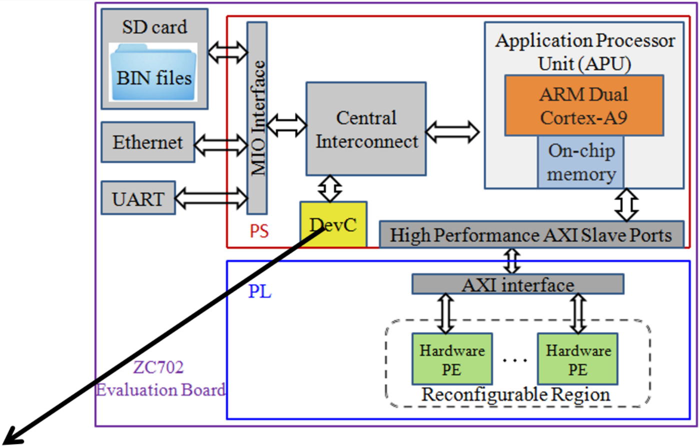
Dual-core ARM Cortex-A9 ⇔ Processing System (PS)
Xilinx FPGA Artix-7 ⇔ Xilinx FPGA Artix-7

Therefore provide both :

- Good performance
- Flexibility

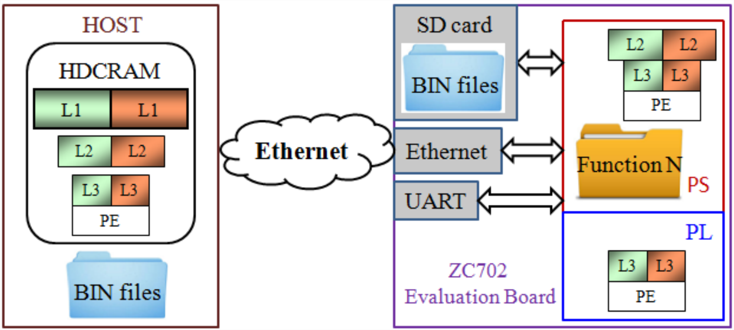
Especially : **Flexible Full and Dynamic Partial Reconfiguration (DPR) technique**

Zynq-7000 Platform



Device Configuration (DevC) / Processor Configuration Access Port (PCAP) interface

Zynq-7000 Platform



The HDCRAM implementation on the ZC702 evaluation board. Bitstreams stored on the Host or on the board depending on the level

Case study : FIR filter

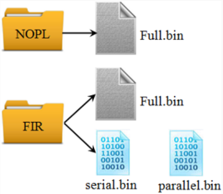
32-tap FIR filter :

- PS : software
- PL : hardware (serial, parallel)

Resource	Serial	Parallel
LUT	868	1096
FD_LD	1516	3108
SLICEL	141	288
SLICEM	91	187
DSP48E1	2	64
RAMBFIFO36E1	8	8

Each time we sent 4096 32-bit integers and then repeat 2000 times

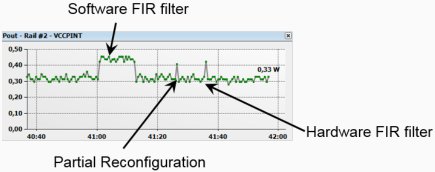
Software	Hardware (μs)	
(μs)	Serial	Parallel
12 229 279	281 315	279 026



Power consumption of PL

Function	NOPL	Serial	Parallel
Power (W)	0.06	0.095	0.101

Power consumption of PS



Software FIR filter

Hardware FIR filter

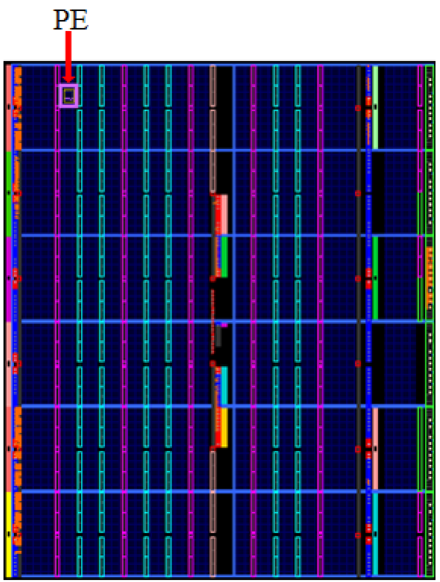
Partial Reconfiguration

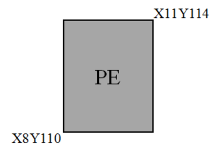
Metrics on FPGA platform

Discussion About the Metrics

Metrics	Self changeability	Configurability	Green impact	At which level	Susceptibility
Voltage	static	medium	strong	system	low
Current	dynamic	unconfigurable	strong	system	medium
Frequency	static	easy	strong	PE	low
Temperature	dynamic	unconfigurable	strong	system	high
Area	static	medium	medium	PE & system	low
Position	static	medium	weak	PE	low
Resource	static	difficult	strong	PE & system	low
Activity rate	dynamic	unconfigurable	medium	PE	medium
Serial /parallel	static	easy	medium	PE	low
Power consumption	dynamic	unconfigurable	strong	PE & system	medium
Performance to power consumption ratio	dynamic	unconfigurable	strong	PE	medium
Working mode	static	easy	strong	system	low

Area, Position, and Resource





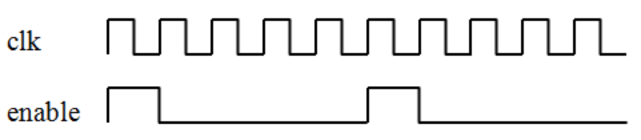
Ressource	Available	Required	% Utility
LUT	80	57	72
FD_LD	80	32	40
SLICEL	10	8	80
SLICEM	10	8	80

Activity Rate

$$activity\ rate = \frac{en \times N}{c} \times 100\%$$

Where

- c : number of clock cycles
- en : number of clock cycles the enable signal lasts during c clock cycles
- N : constant that indicates, given an input, how many clock cycles are required to generate an output

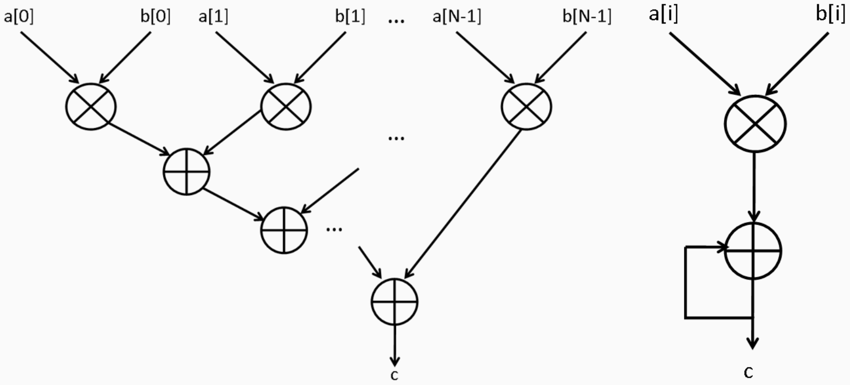


if N = 1, during c = 10 clock cycles, the activity rate = 20 %

if N = 5, the activity rate = 100 %

Serial / Parallel

$$c = \sum_{i=0}^{N-1} a_i \times b_i$$

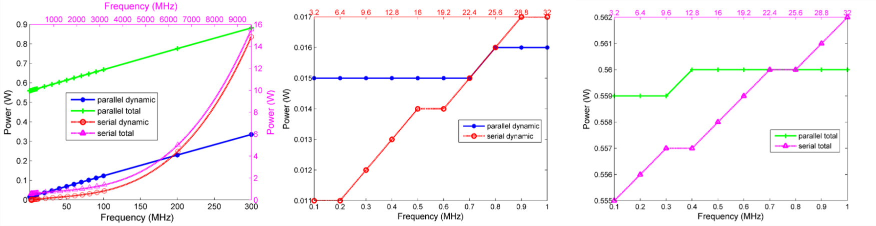


Case study : FIR filter

32-tap FIR filter on Virtex-5 platform

Involved metrics : Serial / Parallel, Frequency, Power Consumption, and Resource

Estimated by XPower Analyzer

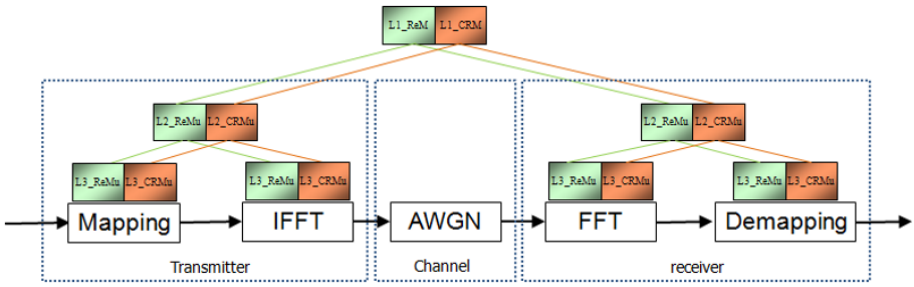


If the throughput < 0.8 MHz, it is better to work in serial mode. Otherwise, the parallel method is recommended.

X. Wu, J. Palicot, P. Leray. *Metrics on Energy Efficiency for Cognitive Green Equipment Based on FPGA Platform*, IEEE Systems Journal, July 2015

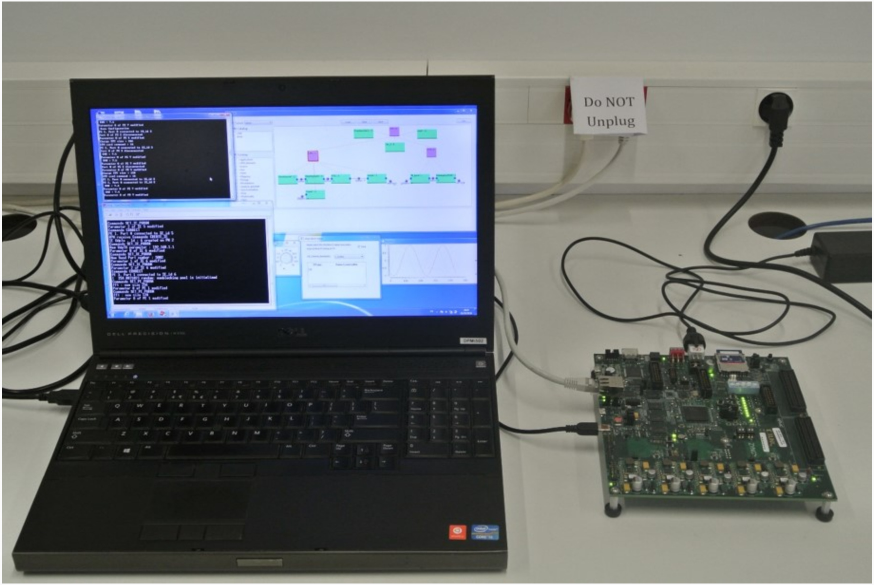
OFDM transmitter and receiver example

OFDM system model

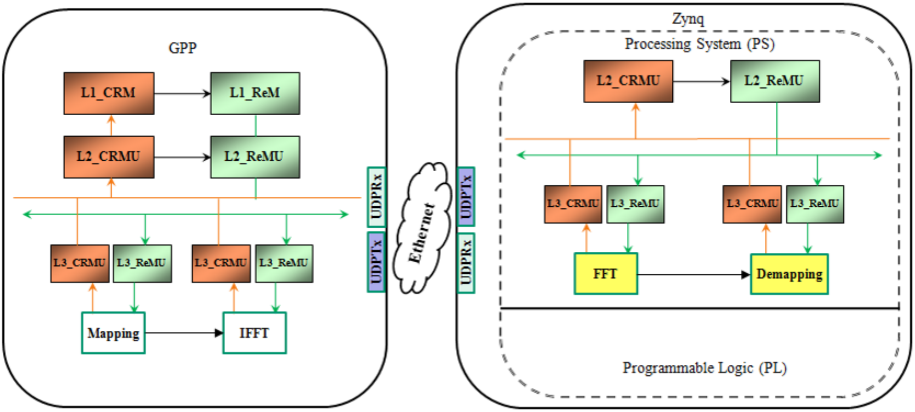


- Transmitter : Mapping and IFFT
- Receiver : FFT and Demapping
- AWGN Channel : additive white Gaussian noise

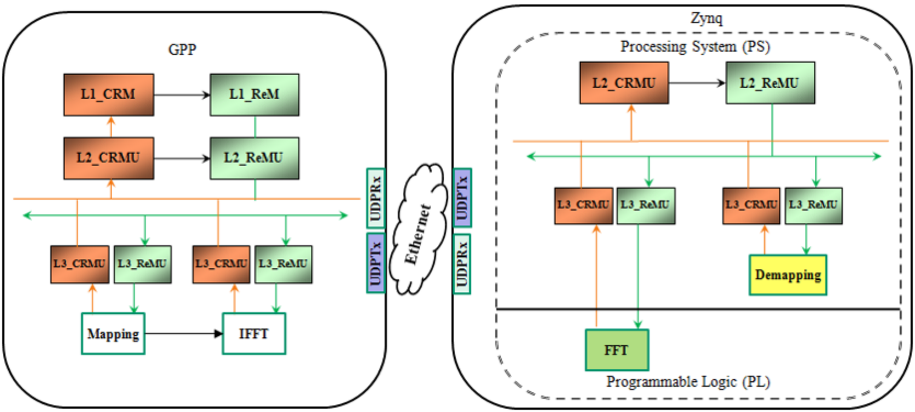
Implementation Platform



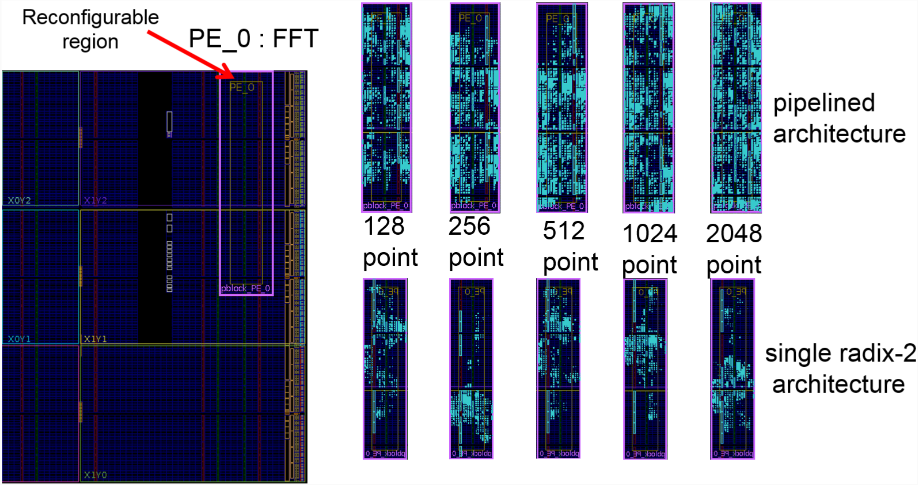
Implementation Platform



Implementation Platform



Implementation Platform



Reconfigurable region

PE_0 : FFT

128 point

256 point

512 point

1024 point

2048 point

pipelined architecture

single radix-2 architecture

Different FFT implementations

Transform length	Resource available	LUT	Register	Slices	DSP48E1	BRAM
		5184	10368	1359	32	48
128	pipelined	2806	3196	702	9	9
	radix-2	1067	1316	268	3	11
256	pipelined	3175	3578	795	9	10
	radix-2	1101	1361	276	3	11
512	pipelined	3589	4113	898	12	12
	radix-2	1154	1392	289	3	11
1024	pipelined	3993	4507	999	12	14
	radix-2	1153	1425	289	3	11
2048	pipelined	4455	5086	1114	15	19
	radix-2	1194	1491	299	3	13
Traditional reconfigurable FFT	pipelined	5741	6056	1435	15	19

Different FFT implementations

Transform time of different FFT implementations

Transform length	Software (μs)	Hardware (μs)		Traditional reconfigurable FFT (μs)
		Pipelined	Radix-2	
128	166	4.81	8.29	4.92
256	364	8.71	16.77	8.93
512	798	16.49	34.85	16.61
1024	1751	31.91	73.41	32.13
2048	3867	62.73	155.49	62.85

Full and partial configuration time of the FFT design

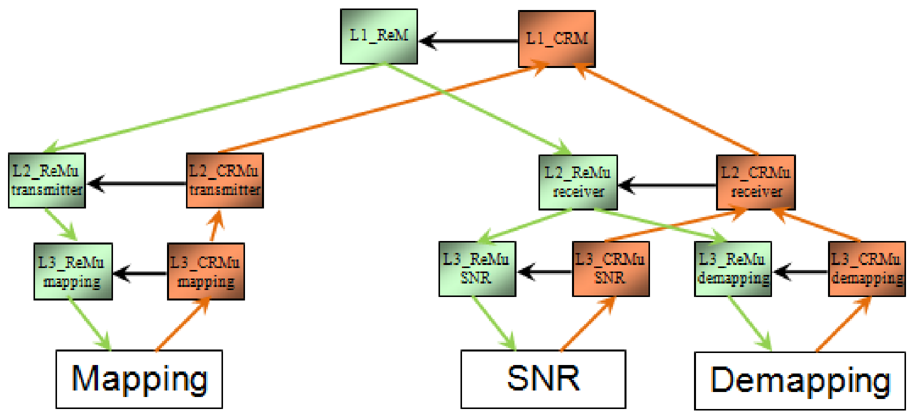
Type	Size (bytes)	Time (ms)
Full	4,045,564	211,413
Partial	384,512	35,122

Power consumption of different FFT implementations

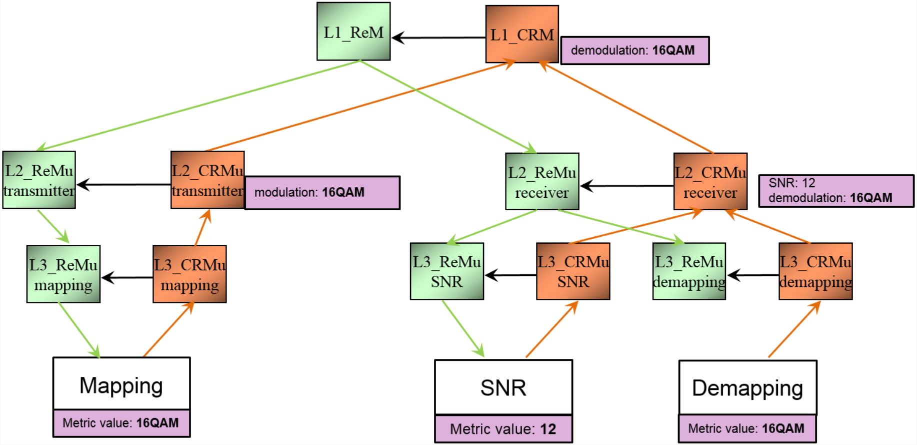
Transform	Power consumption (W)	
length	Pipelined	Radix-2
128	0.103	0.096
256	0.105	0.097
512	0.108	0.098
1024	0.113	0.099
2048	0.121	0.101
Software FFT	0.12	
Traditional reconfigurable FFT	0.135	

Scenario 1 : Modulation adaptation

- Channel conditions : when SNR $\leq$ 10 dB, modulation 16-QAM $\Rightarrow$ QPSK
- Three PEs involved : Mapping, SNR and demapping
- The metric SNR is used in this scenario

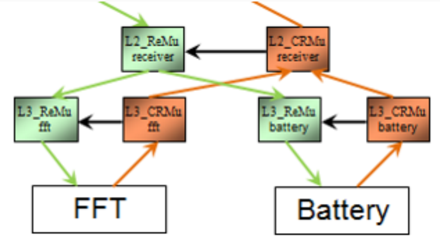


Scenario 1 : Modulation adaptation



Scenario 2

Management of FFT implementation type depending on the battery level



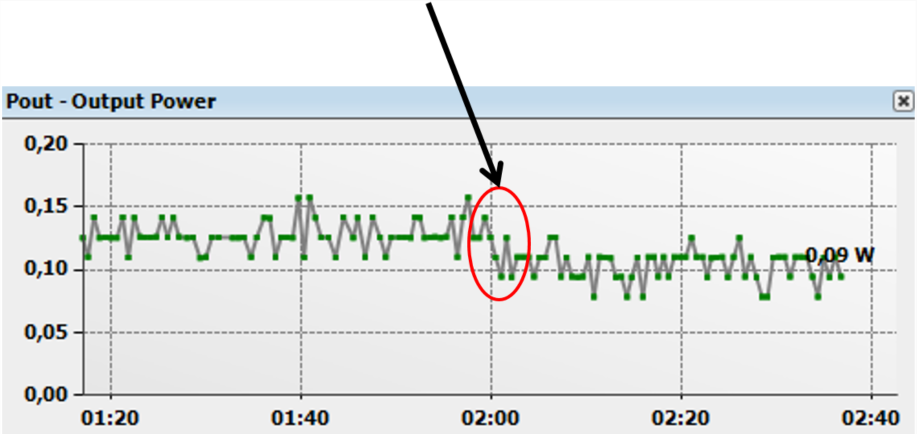
Metrics

- FFT type
- Battery level
- Power consumption

- If Battery Level = high and FFT type ≠ hardware pipelined, higher performance can be achieved. the FFT should be implemented with hardware pipelined architecture
- if Battery Level = low and FFT type ≠ hardware radix-2, the FFT should be implemented with low power consumption architecture hardware radix-2

Scenario 2

The power changes when changing the FFT from pipelined 2048 to radix-2 2048



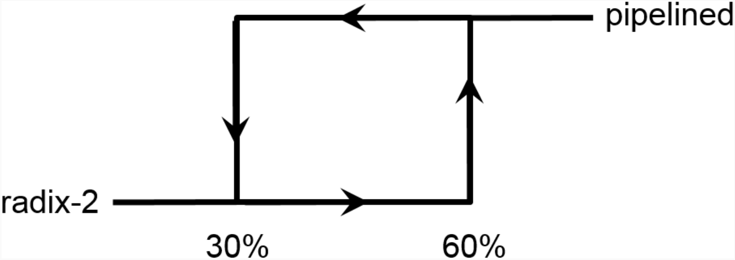
Around 0.02W power reduction

Demonstration

Demonstration

HDCRAM management handles automatic FFT implementation adaptation according to the battery level :

- If Battery Level $\geq 60\%$, hardware pipelined architecture is used to achieve higher performance
- if Battery Level $< 30\%$, hardware radix-2 architecture is chosen to save power



Conclusion and perspectives

Conclusion

- Study of HDCRAM for CR management in green context
- Implementation of HDCRAM on two hardware platforms
- Introduction of some useful metrics on hardware platform
- Development of a hardware UDP core, which works at 125MBytes/s, to provide a high speed transmission of data and partial bitstreams
- Taken advantage of dynamic partial reconfiguration for the runtime reconfiguration of hardware PEs
- Proof of concept management scenarios and demonstrations of HDCRAM with software/hardware co-design
- Proof of a gain in energy consumption

Perspectives

- Enrichment of the PE library
- Development of a hardware UDP core on PL
- Inclusion of complex learning algorithms in the CRMus
- Study of High level modeling methodology and integration tools to automatically generate codes, e.g., C++, VHDL

Any questions ?

pascal.cotret@centralesupelec.fr