Software Defined Radio Equipment: What's the Best Design Approach to Reduce Power Consumption and Increase Reconfigurability?

Manel Hentati[1, 3], Amor Nafkha[2], Pierre Leray[2], Jean-François Nezan[1] , Mohamed Abid[3]

[1]INSA/IETR, 20 Av. des Buttes de coesmes CS 14315 F-35043 RENNES, France
[2]SUPELEC/IETR, 5 Av. de la Boulaie CS 47601 F-35576 Césson-Sévigne CEDEX, France
[3]ENIS/CES, Route Soukra B.P. 1173, 3038 Sfax, Tunisie

ABSTRACT
This article explores several hardware design methods used to implement a reconfigurable software defined radio system. The promise of software defined radios for rapidly changing the operating characteristics of radios suggests further an exciting new method to create opportunities and means for interoperability among and between any number of different radio systems. The possibilities of run-time reconfiguration techniques are explained and quantified. In this article, we are going to limit our discussion to examine the reconfigurability and low power trade-offs between: (*i*) building dedicated functional modules providing high performance at a high cost (*Velcro approach*), versus (*ii*) parameterizable function blocks used in FPGA-based system development, versus (*iii*) dynamic partial reconfiguration which is the ability to reconfigured a portion of the FPGA while the remainder is still in operation. The main objective here is to explore and discuss the best method to design a reconfigurable, a high performance and a low power consumption software defined equipment.

General Terms
Embedded Systems.

Keywords
Design approaches, low power consumption, software defined radio, convolutional encoder, FPGA, dynamic and partial reconfiguration.

1. INTRODUCTION
In future wireless mobile communication systems, it will be necessary to realize smart equipment that can operate under several communication scenarios. Software defined radio (SDR) is a technology that comes progressively to realize this objective. The SDR is a wireless communication system in which some function blocks are implemented using flexible software routines instead of fixed hardware. So various wireless communication scenarios and models can be easily supported by the same platform [1]. A SDR terminal changes adaptively its operation mode according to the type of the available wireless network. Although this concept is very interesting, using reconfigurable computing, it has been unachievable for the moment because of its tight power budget and its high demand on computation capability.

The power allowed for baseband signal processing should be lower than several hundred mW in order to be used for commercial purposes [2]. Furthermore, the reconfigurability needed by the execution core of various wireless protocols tends to increase the total power consumption of the hardware. Designing the digital baseband processing of such an extremely flexible system is a very challenging task. This step is very critical given that these dynamic reconfigurable radios are strictly limited by the battery capacity. Many techniques have addressed the hardware reconfigurability and power consumption of the SDR design.

To implement software defined radios, various technologies can be used, such as the application specific integrated circuit (ASIC), the digital signal processor (DSP), the field programmable gate array (FPGA), and the general purpose processor (GPP). The GPP offers the higher flexibility but has the lowest performance. However, the ASIC is the least flexible one but has the highest performance. ASIC is used to minimize power consumption and to maximize performance. The FPGA provides the best reconfigurable solution for high speed signal processing modules that are highly parallel. In software defined radio baseband, the GPP, DSP and FPGA are commonly used for baseband processing and the ASIC [21] can be only used for some common specific modules such as the up converters and/or down converters.

As indicated above, the FPGA provides the best balance between performance, low power consumption, and short design cycle. Beside, the new Xilinx FPGA series provides a dynamic and partial reconfiguration functionality which is the ability to dynamically modify a local region of logic by downloading partial reconfiguration files while the remaining logic continues to operate without interruption.

In this article, we propose to compare and analyze three hardware implementation methods to design software defined radio equipment. The first method is the Velcro approach which is the classical approach to implement the different radio standards and modes configuration inside equipment. This method uses a simple switch to execut the used mode. The duplication of each chain becomes unworkable from a technical and economical point of view and the crying need of new methodology to design terminal reflects that. On the contrary, in the pioneering work in [3, 4], authors present a second design approach, named Parameterization. This approach aims to design software defined radio equipment where each function can be changed just by a simple parameter adjustment and it can be extended to lower design level. Palicot and al [5] propose a very interesting technique based on the graph theory for optimal determination of common functions and optimal parameters. More recently, Xilinx introduces a third design approach called FPGA Dynamic and Partial Reconfiguration (DPR) which has been

widely studied in academia [9]. The DPR provides the modification of a portion of the device while the rest remains unchanged and active. It has prominent advantages such as the increase of the system performance, the ability to change hardware, hardware sharing, low power consumption, and very small reconfiguration time.

In this article, we study performance, reconfigurability, and power consumption of the three design approaches. In the literature, there are not many papers published, according the best of our knowledge, that concerning the impact of the hardware design technique on power consumption in the context of SDR equipment design. The analysis is done using a classical convolutional encoder block. This is a simple example but, the concept can easily be extended to provide more useful and complex signal processing functionality. In this work, we did not take in consideration various power minimization techniques like clock gating, dynamic voltage and frequency scaling, and multiple voltage islands.

In the remainder of this paper, Section 2 explains the specificities of a convolutional channel encoder. Section 3 presents the different design approaches to implement the channel decoder block using Velcro, parameterization, and DPR techniques. The experimental results are reported in Section 4. Section 5 concludes the paper and discusses some future directions.

2. CONVOLUTIONAL ENCODER

Convolutional codes were introduced by Peter Elias [6] in 1955. They are the widely used channel codes in practical communication systems. Convolutionally encoding the data is achieved using a shift register and associated combinatorial logic that is often in the form of cascaded exclusive-or gates. The encoded bits depend on the current k input bits and on past input bits. They are described by three integers, (n, k, K), where the ratio k/n is called the code rate and K is a parameter presents the constraint length; it represents the number of k stages in the encoding shift register. The constraint length K determines the capability and the complexity of the code. Several decoding algorithms have been proposed in the literature, but the most well-known is probably Viterbi algorithm [7]. As with the majority of codes, convolutional codes typically use binary symbols. Figure 1 illustrates a simple (7; 5) binary convolutional encoder of constraint length K = 3.

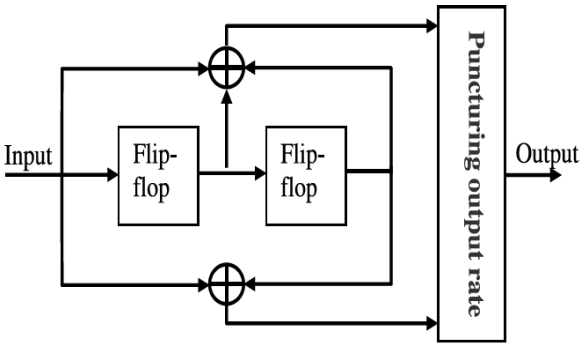

Fig 1: Example of convolutional encoder with r=1/2 and K=3

The octal numbers 7 and 5 represent the code generator polynomials, which read in binary (111_2 and 101_2). It corresponds to the shift register connections to the respectively upper and lower modulo-two adders. The choice of the connections between the adders and shift register gives rise to the characteristics of the code. At each input bit time, a bit is shifted into the leftmost stage and the bits in the register are shifted one position to the right. The output switch presents the sample of the output of each modulo-2 adder, forming the code symbol pair associated with the present input bit. The sampling is repeated for each inputted bit.

In order to estimate and analyze the impact of the implantation method on power consumption, we consider here some examples of convolutional encoder used by the Global System for Mobile communications (GSM), the Universal Mobile Telecommunication System (UMTS) and 802.11g wireless standards. Table 1 presentes the polynomials generator used by above standards.

Table 1: Generators polynomials used by channel encoder

Conv. Coding	Rate	Generators polynomials
GSM TCH/FH TCH/HE	1/2	G0 = 31 G1 = 33
	1/3	G4 = 155 G5 = 123 G6 = 137
UMTS BCH, PCH TCH/HE	1/2	G0 = 561 G1 = 753
	1/3	G0 = 557 G1 = 663 G2 = 711
802.11G (OFDM Mode)	1/2	G0 = 133 G1 = 171

We can implement our channel encoder using a synchronous model, in which all blocks are assumed to have finished computation when a clock edge arrives. In this paper, we will use a globally asynchronous locally synchronous (GALS) model [14]. In this model synchronous functional blocks with locally generated clocks are used with asynchronous connections between them. Therefore, we retain the advantages of synchronous circuits, and we can exploit the advantages of asynchronous routing at the same time. The designed system is composed of blocks which can be individually optimized and the timing of one block does not affect the other blocks. The channel encoder might work on different standards such as GSM, UMTS and in different modes for example GSM-TCH, UMTS-BCH. We implement the channel encoder using three approaches: Velcro, Parameterization and DPR.

3. DESIGN APPROACHES

The main objective of this work is to study the impact of design approaches in the performance of the system. We propose three different implementation methods, which are Velcro, Parameterization and DPR.

3.1 Velcro approach

In Velcro approach [12], each standard is optimized and can be implemented as an independent processing unit (PU) as shown in figure 2.a. The inter-change between this PEs can be made by using a multiplexer. Switching from one standard to another can be done during one clock cycle. However, all standards must be implemented on chip which can increase the global power consumption of the design..

3.2 Parameterization approach

The parameterization approach [12] is based on using parameters to switch from one standard to another. In this approach standards share resources as seen in figure 2.b. To commute between different standards, we should only modify the polynomial generator parameters for example. This approach consists in the reusing of the same design to perform different standards. To design parameterization architecture the worst case must be implemented. This approach reduces the area utilization but it becomes more complex if the number of parameters is increased. Moreover, the design usually requires more than one cycle clock to commute between two standards.

3.3 DPR approach

Dynamic partial reconfiguration is the ability to change the configuration of part of an FPGA device while other processes continue in the rest of the device [10]. Xilinx suggests in two basic techniques of DPR on a single FPGA: the module-based and the difference-based partial reconfiguration. The difference-based [16] can be used when a small change is made to the design. It is especially useful in case of changing look-up table (LUT) equations or memory blocks content. The generated partial bitstream contains only information about differences between the current design and the new content of an FPGA. Switching the configuration of a module from one implementation to another is very quick, as the bitstream differences can be extremely small. The module-based [17] uses modular design concepts to reconfigure blocks of logic. The distinct portions of the design to be reconfigured are known as reconfigurable modules. Because specific properties and specific layout criteria must be met with respect to a reconfigurable module, any FPGA design intending to use partial reconfiguration must be planned.

A partial reconfigurable design typically comprises an area for static modules and one or more partial reconfigurable modules (RM). The static modules contain logic that will remain constant during partial reconfiguration. Partial reconfiguration module is the design module that can be swapped in the device on the real-time multiple modules can be defined for a specific FPGA region. In the early Virtex family devices like Virtex-II and Virtex-II Pro devices, we must partially reconfigure whole columns. Recently, in Virtex -4/5/6 devices, the partial reconfigurable region (PRR) is rectangular of arbitrary size and may be located anywhere with no overlapping.

During Partial reconfiguration process, the routing signals used for inter-modules communication should be unchanged when the module is reconfigured. This fixed routing bridge of communication is achieved by using bus macros. Bus macros are hard macros, which is located at the edge of boundaries separating dynamic and static regions.

To load the partial bitstream, designer should use an internal configuration access port (ICAP). The Virtex-II series are the first architectures that support ICAP which is a subset of the SelectMAP interface having fewer signals. The ICAP only deals with partial configurations and does not have to support different configuration modes. It gives an 8 input/output bits data bus. While with the Virtex-4 and Virtex-5 series, the ICAP interface has been updated with 32 input/output bits data bus to increase its bandwidth and speed up the reconfiguration time. The intuitive benefits of using DPR are: the improvement of FPGA area efficiency [11], the augmentation of architectures flexibility, the decrease of power consumption [18] and smaller FPGAs can be used to run an application because to commute between different standards, we can just load the partial reconfiguration bitstream. However, the implementation process for DPR is still complex and tedious, and the designer should have a thorough understanding of the underlying device and design methodology.

In our case study, the convolutional encoder architecture is composed of a static module containing primitive operators as XOR gates and shift registers, and a dynamic module. The partial reconfigurable module can be seen as a connection matrix with limited connectivity where each matrix element corresponding to the connection or not between registers outputs and XOR gates inputs as shown in figure 2.c

4 HARDWARE SETUP

Based on the Xilinx ML505 development board with a Virtex-5 FX FPGA, experiments have been done to investigate the performance of various design structures. We use the system architecture shown in figure 3.

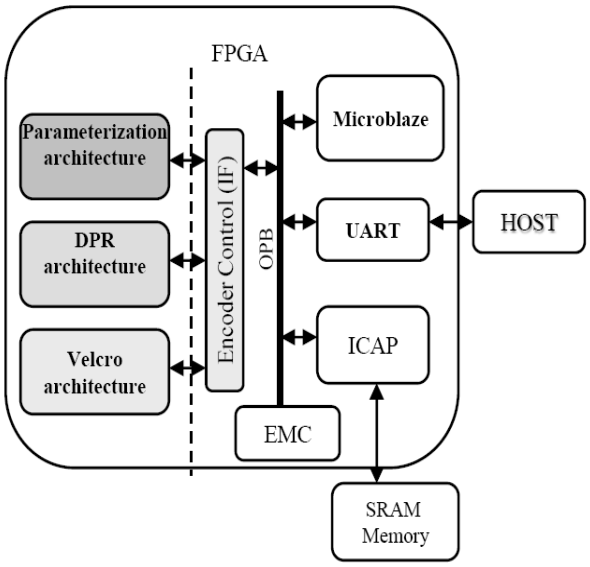

Fig3: An embedded architecture for the design approach study

This architecture is composed of:

- A Microblaze, which is a 32-bit RISC soft processor core [19] to manage the configuration of different standards. The processor programs are executed in DDR memory (32-bit, 100 MHz).
- An UART provides serial communication between PC and FPGA.

- A ICAP core has been utilized in this paper for adaptive computing study. To improve the reconfiguration speed, authors in [13] have used a direct memory access technique to directly transfer configuration data from external SRAM to the ICAP port which can run at 100 MHz. So by using this memory access technique with 4 bytes ICAP width, we will be able to reach 400 megabytes per second (MB/s).
- A SRAM memory to store the partial bitsreams.
- A bus OPB to connected all the peripherals.
- A direct memory access engine (EMC) to establish a direct transfer link between the external SRAM and the ICAP controller. So by reducing the partial reconfiguration time, we are able to perform real-time switching between standards. Moreover, we can also reduce the partial reconfiguration power overhead.

A C code is developed to control the ICAP. It provides a basic set of memory and instruction registers which are all accessible and addressable from processor cores on the FPGA. These processors are mainly MicroBlaze processor or hard wired PowerPC cores within Xilinx FPGAs

5. EXPERIMENTS AND RESULTS

In this section, we present separately the implementation results of the convolutional encoder based on Velcro, Parametrization and DPR design approaches. We have employed a set of Xilinx tools version 11.1 [20]:

-The ISE tools is used for the synthesis, it can automatically interpret the HDL syntax, synthesize the description, to generate a ngc file.

-The PlanAhead visual floorplanning tool for iterative design and placement. PlanAhead has partial reconfiguration options which make it a required tool in the partial reconfiguration tool flow

-Xilinx EDK : provides a framework for design of hardware/software components of the embedded processor systems on programmable logic. Appropriate tools for each stage of the design in addition to IBM PowerPC and Xilinx MicroBlaze processor cores infrastructure and peripheral IP cores facilitate hardware/software partitioning and design reuse.

-The IMPACT tool is used to download bitsreams in the FPGA using JTEG connector.

All the above design approaches have been tested with the same standardized tests that are used to measure and to verify the convolutional channel encoder associated to different standards. Table 2 shows area costs of the three design techniques at the frequency of 360 MHz.

Table 2: On-chip area costs using different design approaches

	Velcro	Parameterization	DPR
LUTs	77	57	38
FlipFlop	140	78	51
I\O	9/3	14/3	5/3
Freq max (MHZ)	359.409	360.021	360.888

The Dynamic and Partial Reconfiguration technique reduces area by decreasing the number of design parameters and input/output ports. We can conclude that the DPR approach leads to more than 49% average reduction in area costs for Virtex-5.

The comparison between the three above architectures in terms of hardware resource shows that, the partial reconfigurable approach has the highest processing rate. In fact, the Velcro method requires more area than the parameterization and dynamic and partial reconfigurable architectures**.**

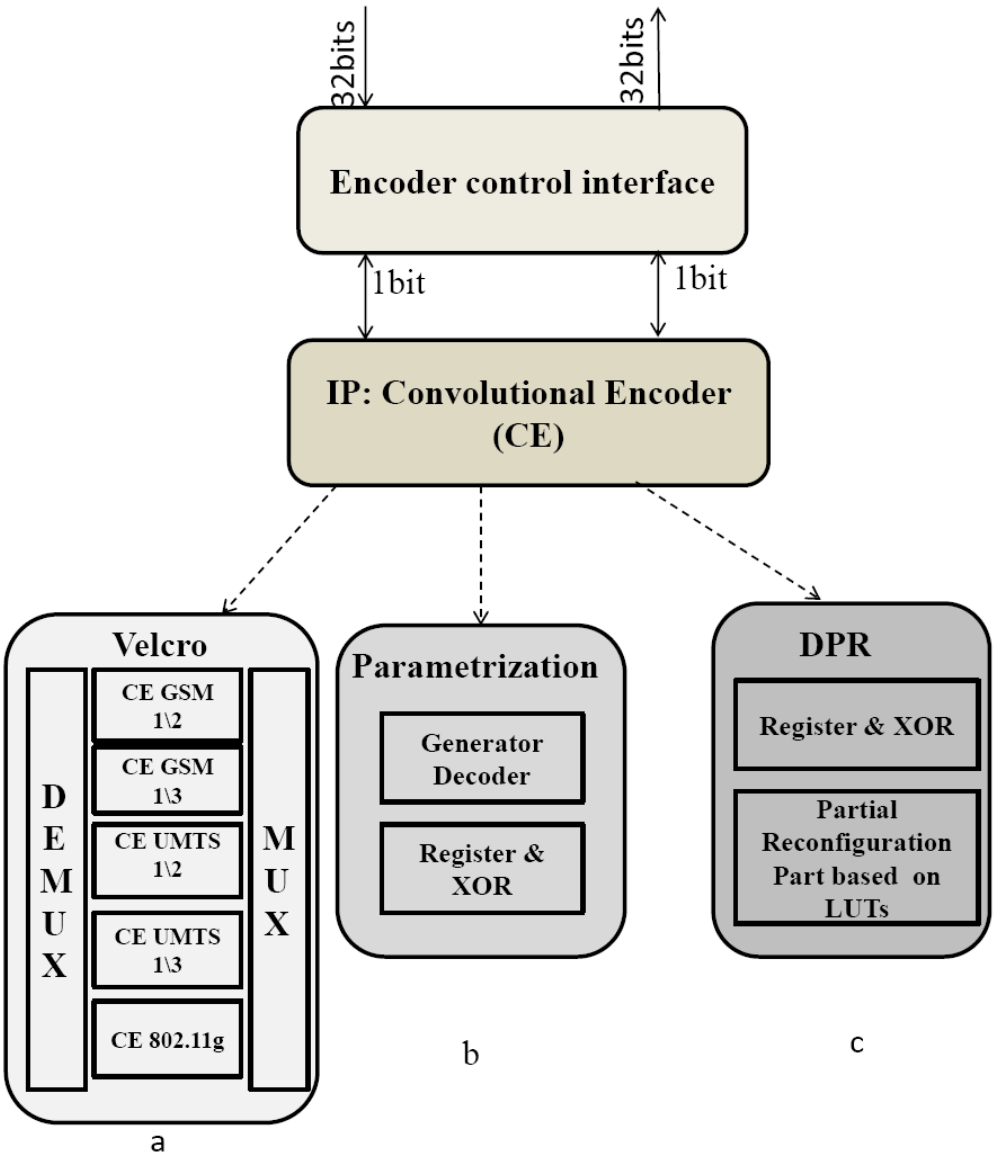

Fig 2: Architectures of channel encoder using respectively: Velcro, Parameterization, and DPR approaches

In table 3, we give the total power consumption estimation for the above three design approaches. To get an available estimation of static and dynamic power consumption of the used FPGA device, we used Xilinx Power Estimator (XPE) tool. The approach used by the Xilinx Power Estimator consists of providing information including the number of LUTs, flip-flops, DCM, DSP, the average switching within the FPGA, and the clock frequency to determine the power consumed by the FPGA for a given temperature [8]. The XPE tool is based on an excel spreadsheet. The advantage of these tools is that results are obtained very fast. Moreover, the results are accurate for dynamic and static power consumption.

Table 3: Total power consumption estimation for different design approaches

	Leakage power (w)	Dynamic power (w)	Total power (w)
Velcro	0.053	1.288	1.341
Parameterization	0.041	1.288	1.329
DPR	0.027	1.286	1.313

The total power of any CMOS circuit can be divided into the dynamic power and the leakage power. Dynamic power is dissipated due to switching of the circuit nodes, whereas

leakage power in a transistor is consumed when it is not switching. During the active mode of a system, some parts of the circuit consume dynamic power and the rest leakage power. According table 3, we note that, any area cost minimization of the processing block can reduce very significantly the total power consumption. An effective way to reduce leakage power consumption can be performed by using small FPGA devices. Another way consists of using the DPR technique to dynamically reconfigure only the needed blocks. So, the number of cells logics used will be reduced as well as the dynamic power.

Total power consumption of Velcro approach is higher than the others because we require a high number, as compared to other approaches, of logic cells to implement simultaneously all standards. The DPR design approach implements only the needed circuit logic gates so the dynamic power is less as comparing to the two other approaches. Dynamic power is also reduced because of the lower toggle rate at each clock period. Using the DPR design approach, the designer is now able to save around 50% of the dynamic power as compared to Velcro, and around 35% as regards to parameterization approach.

The main two limits of the DPR design approach are that: timing and reconfiguration power overhead. Both of these overheads are directly related to the size of the physical partial reconfiguration file. As reconfiguration time is highly dependent on the size and organization of the partial reconfigurable region, in this paper, we have designed the simplest and smallest partial reconfiguration module. The size of the produced partial reconfiguration file is about 8 kBytes, so we are able to calculate the partial reconfiguration time, based on the following equation:

$$T_{DPR} = T_{inst} + T_{eff}$$

$$= \frac{10}{f_{clk}} + \frac{bitstreams\ size}{4 * f_{clk}}$$

Where T_{inst} is the time required by the MicroBlaze to start the partial reconfiguration process, and T_{eff} is the speed by the data transfer from the external 32 bits SRAM to internal reconfiguration port (ICAP). The FPGA clock frequency (f_{clk}) is equal to 100 MHZ in our case, the size of partial bitstream is 8 kBytes, so the partial reconfiguration time values is 20.58us. The power consumption overhead required in the partial reconfiguration processing is not indicated in this work. However, Xilinx indicates that this overhead is small with respect to the total consumed power. Moreover, in wireless communications, standards are not frequently changed and updated as a consequence the partial reconfiguration is rarely performed comparing to processing frequency. The ICAP throughput is critical to the performance of partial reconfiguration: a high ICAP throughput enables partial configuration to reduce energy consumption even if the reconfiguration time is in the range of milliseconds as presented in [11]. Table 4 illustrates the design time spent to develop convolutionnal channel encoder using above design approaches. This time present the time required to develop an application on FPGA.

Table 4: Design time

	Design time
Velcro	1 day
Parameterization	1 day
DPR	2 days

We note that, DPR architecture has the highest design time. This is mainly due to the complexity of the design flow used for the DPR technique. Indeed, in comparison to a static system, the design of a partially reconfigurable system requires additional design steps, such as partitioning the application into static and dynamic modules, adding the ICAP, develop a C code to manage the partial reconfiguration bitstreams, etc.

The design properties of the three proposed design methods are concluded in table 5 resting on four aspects: processing rate (area), performance, power consumption and configuration time overhead.

Table 5: Properties of design approaches

	Velcro	Parameterization	DPR
area	-	+	++
Performance	+	+	+
power	-	+	++
Conf-overhead	++	+	-

We can see in table 5 that, the Velcro design approach has the lowest configuration time overhead because the switching time between different standard is holding only one clock period. However, its highest area waste makes the system power inefficient. Parameterization approach ensured less area waste than Velcro according to these attached parameters for reprogramming between different standard. If we use a Virtex-II FPGA, the ICAP will operate at 50 MHz and 8 bits data width port, the reconfiguration time will be highly increased.

4. CONCLUSIONS

In this paper, three design approaches have been implemented and analyzed through comparing their on-chip area occupations, performances, and power consumptions. We have focused in the convotuionnal encoder as a case study But, the design approaches can easily use for large variety of FPGA Application.

The implementation results demonstrate that the Dynamic and Partial Reconfiguration design approach is an effective way to decrease the hardware resources utilization and the power consumption. The intervallic area-processing rate shows the advantage of using partial reconfiguration than Velcro, even supposing that the latter has shortest configuration time. As we know an added configuration, overhead is appended when we use the partial reconfiguration; and it is insignificant thanks to the novel ICAP controller. Especially, in the real telecommunication example, these systems always have a long idle time for a processing period. Therefore, the number of economic logic cells is more remarkable here. Added to that, when we reduce the number of logic cell, the power

consumption is reduced at the same time. Therefore, based on this experimentation, the results of the extrapolation study indicate that the use of partial reconfiguration is more beneficial when the configuration overhead is not considered in the system. Therefore, dynamic partial reconfiguration seems to be a promising solution to develop an applications demanding adaptive and flexible hardware.

As perspectives, we propose to implement a modulation chain for multi standard communication based on DPR technique. In the first, we should partition the application on static modules and dynamic modules. Then, we apply the DPR in order to enhance current Software Radio and future Cognitive Radio system.

5. REFERENCES

[1] J. H. Reed, "Software Radio Modern Approach to Radio Engineering ", Prentice Hall, Upper Saddle River,NJ 13-18, 2002.

[2] H Lee and all, "Software Defined Radio - A High Performance Embedded Challenge ", , Intl. Conference on High Performance Embedded Architecture and Compiler.2005.

[3] F.~Jondral Software Defined Radio Enabling Technologie (by Walter Tuttlebee), book, chapter Parametrization - A technique for SDR Implementation. Wiley, 2002..

[4] J. Palicot, C.~Roland FFT: a basic Function for a Reconfigurable Receiver, ICT'2003, Papeete, Tahiti,Feb. 2003..

[5] V. Rodriguez, C. Moy, J. Palicot, Install or invoke?: The optimal trade-off between performance and cost in the design of multi-standard reconfigurable radios, Wiley InterScience, Wireless Communications and Mobile Computing Journal, to appear, 2007.

[6] P. Elias Coding for Noisy Channels, IRE Conv. Rec., Part 4, pp. 37-47, 1955.

[7] A. J. Viterbi Error bounds for convolutional codes and an asymptotically optimum decoding algorithm, IEEE Trans. Inf. Theory, vol. IT-13, pp. 260-269, Apr 1967.

[8] Diana Gohringer, Jonathan Obie, André, L. S. Braga, Michael Hubner, Carlos H. Llanos, and Jurgen Becker "Exploration of the Power-Performance Tradeoff through Parameterization of FPGA-Based Multiprocessor Systems", International Journal of Reconfigurable Computing Volume 2011.

[9] P. Manet, and all An Evaluation of Dynamic Partial Reconfiguration for Signal and Image Processing in Professional Electronics Applications, EURASIP Journal on Embedded Systems, 2008..

[10] P. Lysaght, B. Blodget, J. Mason, B. Bridgford, and J. Young Enhanced Architectures, Design Methodologies and CAD Tools for dynamic reconfiguration of XILINX FPGAs, in 16th Int. Conf. on Field Programmable Logic and Applications (FPL2006), pp. 12-17, 2006..

[11] Cindy Kao Benefits of Partial Reconfiguration Take advantage of even more capabilities in your FPGA, Xcell Journal Xilinx, vol. I, pp. 65-67, 2005..

[12] V. Rodriguez, C. Moy, and J. Palicot An optimal architecture for a multi-standard reconfigurable radio: Cost-minimising common operators under latency constraints}, IST Mobile Summit'06, Mykonos : Greece, 2006..

[13] J. Delorme, A. Nafkha, P. Leray, and C. Moy , New OPBHWICAP interface for real-time partial reconfiguration of FPGA}, ReConFig'09, Cancun : Mexico. 2009.

[14] Y.Thonnart, E.Beigne, and P.Vivet, Design and Implementation of a GALS Adapter for ANoC Based Architectures, ASYNC'09, pp.13-22, 2009.

[15] S. Liu, R. N Pittman, and A. Forin, Energy reduction with run-time partial reconfiguration, in Proc. FPGA, 2010.

[16] R.V. Kshirsagar and S. Sharma, difference based partial reconfiguration, International Journal of Advances in Engineering and Technology, Vol. 1,Issue 2,pp.194-197, May 2011..

[17] Two flows for Partial Reconfiguration: Module Based or Difference based, xilinx Application Note XAPP290 (v2.0) December 3, 2007.

[18] B. Krill, A. Ahmad, A. Amira, and H. Rabah, An efficient FPGA-based dynamic partial reconfiguration design flow and environment for image and signal processing IP cores}, Journal of Signal Proc.: Image Comm., vol. 25(5), pp. 377-387, 2010.

[19] MicroBlaze Processor Reference Guide Embedded Development Kit EDK 10.1i, UG081 (v9.0), 2008.

[20] www.xilinx.com.

[21] Wemekamp, John, Emerging Trends in Mil/Aerospace Embedded Systems, Electronic Component News, pp. 27-29, May 2007.