

The Study of the impact of architecture design on cognitive radio

Manel Hentati*, Amor Nafkha[†], Xun Zhang[†], Pierre Leray[†], Jean-Francois Nezan* and Mohamed Abid[‡]

* INSA/IETR, 20 ,av des Buttes de coesmes CS 14315 F-35043 RENNES, France

Email: {manel.hentati, jean-francois.nezan}@insa-rennes.fr

[†] SUPELEC/IETR Av. de la Boulaie CS 47601 F-35576 Cesson-Sévigne CEDEX, France

Email: {amor.nafkha, xun.zhang, pierre.leray}@supelec.fr

[‡]ENIS/CES, Route Soukra B.P. 1173, 3038 Sfax, Tunisie

Email: mohamed.abid@enis.rnu.tn

Abstract—Software Defined Radios (SDR) allow a dynamic reconfiguration technique of reusing hardware to implement the physical layer processing of a Cognitive Radio (CR) equipment. This paper exhibits three hardware implementation approaches for an SDR respectively called velcro, parametrization and dynamic partial reconfiguration. The main objective of this paper is to discuss the best way to design a flexible, high performed and a low power consumption CR equipment.

Index Terms—encoder, architecture, dynamic partial reconfiguration, performance

I. INTRODUCTION

For mobile communication systems concerning the third generation, it will be necessary to realize a smart wireless terminal that can operate under several communication systems. 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 systems can be easily supported by the same platform [1]. An SDR terminal changes adaptively its operation mode according to the type of the available wireless network. Although this concept is very interesting, using programmable hardware for SDR terminals has been unachievable because of their tight power budget and their 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 Programmability needed for the execution of various wireless protocols tends to increase the power of the hardware system efficiency. 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 consumption of energy. Many approaches and studies have addressed the hardware reconfigurability/programmability of the SDR design. The "Velcro" approach aims at supporting several communication standards through few self-contained complex communication components; each one is exclusively dedicated to a given standard. On the contrary, the pioneering work in [3, 4, 5] proposed a new design approach, named "Parameterization". This approach aims at designing

multi-standards systems in which operations can be modified by a simple parameter adjustment. This approach can be extended to lower level entities called common operators. Palicot and al [6], [14] proposes a very interesting technique based on graph theory for optimal determination of common operators/function. This approach is based on selecting the primitive operators and invoking them repeatedly to perform the various communication tasks as it is necessitated by each standard. More recently, FPGA Dynamic Partial Reconfiguration (DPR) has been widely studied in academia [11]. 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 and less reconfiguration time [12][13].

In this paper, we investigate performance, flexibility, and power consumption of the above design approaches (i.e. Velcro, parameterization, and DPR). As we know, there are not many papers published concerning the impact of the hardware design technique on power consumption in the context of SDR equipment design. This analysis helps us to take a very simple hardware block: the convolutional encoder. In the remainder of this paper, Section 2 explains the specificities of convolutional encoder. Section 3 exposes the different architectures to implement such processing block (Velcro, parameterization, Partial reconfiguration). Sections 4 and 5 give and detail the main implementation results followed by the conclusion in Section 6.

II. CONVOLUTIONAL ENCODER

Convolutional codes were introduced by Peter Elias [8] in 1955. They are the widely used channel codes in practical communication systems. The encoded bits depend not only on the current k input bits but also on past input bits. They are described by three integers, n , k and K , where the ratio k/n is called the code rate and K is a parameter known as the constraint length; it represents the number of k stages in the encoding shift register. The constraint length K determines the capability and complexity of the code. Several decoding algorithms have been proposed in the literature, but the most well-known is probably Viterbi algorithm [9]. As

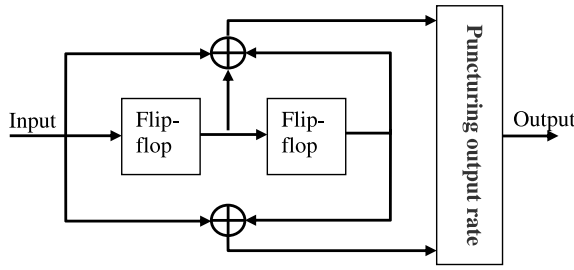


Fig. 1. Example of convolutional encoder with $r=1/2$ and $K=3$

with the majority of codes, convolutional codes typically use binary symbols. Figure 1 illustrates a simple $(7, 5)$ binary convolutional encoder of constraint length $K = 3$. 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.

III. CONVOLUTIONAL ENCODER ARCHITECTURES

In order to estimate and analyze the impact of the implementation method on power consumption, we consider here some examples of convolutional encoder used by GSM, UMTS and 802.11g standards. Table I illustrates the generators polynomials used by above standards.

TABLE I
GENERATORS POLYNOMIALS USED BY CHANNEL ENCODER

Conv. Coding	Rate	Generators polynomials
GSM TCH/FH TCH/HE	$1/2$	$G_0 = 31$ $G_1 = 33$
	$1/3$	$G_4 = 155$ $G_5 = 123$ $G_6 = 137$
UMTS BCH, PCH TCH/HE	$1/2$	$G_0 = 561$ $G_1 = 753$
	$1/3$	$G_0 = 557$ $G_1 = 663$ $G_2 = 711$
802.11 G (OFDM Mode)	$1/2$	$G_0 = 133$ $G_1 = 171$

What is conventional is that we can implement our channel encoder using a synchronous approach, 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) approach. In this system synchronous modules 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 block. 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 architectures Velcro, parameterization and partial dynamic reconfiguration

In Velcro approach, each standard is optimized and can be implemented as an independent Processing Element (PE)[15] shown in figure 2-a. The inter-change between these PEs can be made by using a select signal "select". 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.

The parameterization architecture [7] 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. This approach consists in the reusing of the same design to perform different standards. To design parameterization architecture the worst case must be adopted. This approach reduces occupied area but it's more complex if we increase the number of parameters; and this requires more than one cycle clock to change standard

Partial reconfiguration is one of the useful solutions to increase the chip operating rate. Furthermore, the partial reconfiguration helps to customize the size of one reconfigurable module that has enough logic cells for the appointed standard. It is useful to reduce the implementation area of the design. In this approach, to commute from one standard to another, we need to update or design our architecture using partial bit stream. A unit of control is needed to manage efficiently the partial reconfiguration. This architecture is composed of static region containing primitive operators (XOR and shift registers) and reconfigurable region. There are inter-connections between different shifts register and XOR operators through Look Up Table (LUT).

The work presented in this paper focuses on comparing the impact of the above design approaches in what concern performance, configuration overhead, and power consumption.

IV. TARGET PLATFORM

In order to demonstrate the propriety of three presented approaches, XILINX development board XC5FXV5 [10] is used; it is equipped with a Virtex-5 FPGA and 9MBits of extern SRAM, and is connected with a standard PC. The basic structure is shown in Figure IV. The main control over the microprocessor ,MicroBlaze, is implemented in the FPGA. The Configuration files, which are stored in on-board SRAM, are transferred via ICAP configuration interface. The above module is working with clock frequency $f_{req_clock} = 100MHz$. As seen in figure 3, a direct memory access engines to establish a direct transfer link between the extern on-board SRAM, in which the configuration files are stored. By comparing the original design where the ICAP controller

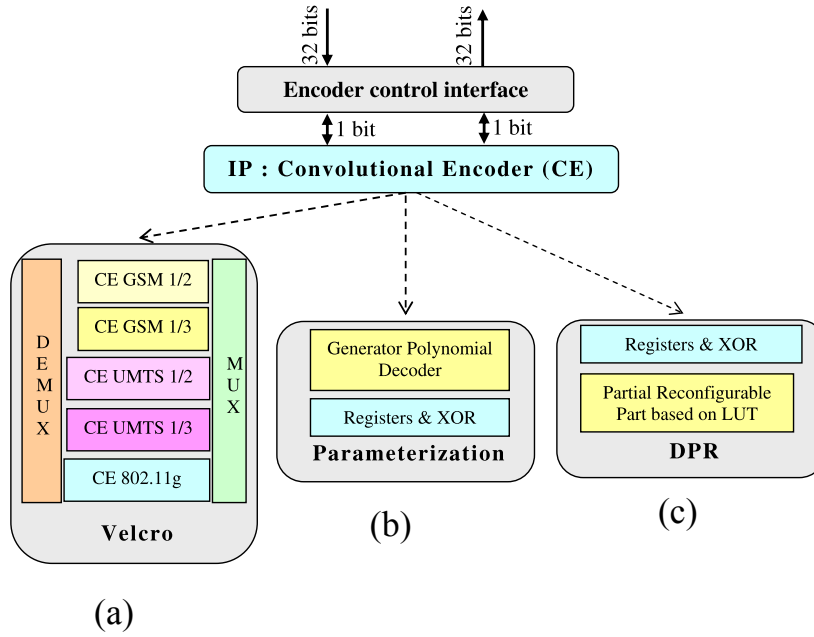


Fig. 2. Architecture for Channel encoder using velcro, parameterization, reconfigurable approaches

contains only the ICAP and ICAP FSM, our new ICAP controller [8] provides high efficient with transfer throughput 400Mbytes=s. It helps to reduce the reconfiguration overhead during the reconfiguration processing.

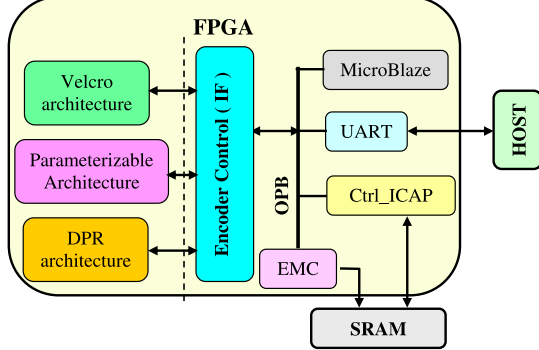


Fig. 3. Adopted reconfigurable architecture

V. IMPLEMENTATION & RESULTS

In this section, we present separately the implementation results of three channel encoder architectures. In order to implement the channel encoders, we employed the Xilinx ISE development tool [10]. All the three architectures are being tested with the same data. Table II shows the optimized value of the area required for the different architectures: Velcro, Parameterization and Partial Reconfiguration. we notice that the Partial Reconfiguration design method has significantly optimized the implementation area and the highest performance value (maximum frequency). The potential savings of the optimized area are around 49%.

TABLE II
ON-CHIP AREA BY USING DIFFERENT DESIGN METHOD AND THE MAXIMUM RUNNING FREQUENCY

	LUTs	FlipFlop	I/O	Freq_max (MHZ)
Velcro	77	140	39/4	359,409
parametrization	57	78	39/5	360,021
Partial reconfiguration	38	51	37/5	360,888

By making a comparison with other architectures in terms of hardware resource complexity, the partial reconfigurable approach has the highest processing rate. In fact, Velcro architecture wastes on-chip logic cell for the idle function module more than the parameterization architecture and more than Partial reconfigurable architecture.

Table III gives the experimental results in term of power consumption of the three approaches. The over high total power is for the total FPGA fabric in which include the leakage power and dynamic power. The results are obtained via XPower, which is the power estimation tool of Xilinx. We find that the saving area on chip indicates directly the optimized power consumption.

TABLE III
POWER CONSUMPTION ESTIMATION OF THREE KINDS OF DESIGN

	Dynamic power (mw)	Total power (mw)
Velcro	0.053	1.341
Parametrization	0.041	1.329
Partial reconfiguration	0.027	1.313

The power consumption of those architectures is on a regular pattern which is identical to the optimization of the area shown in table II. As we can see from table III. Static power of Velcro approach is higher than the others because

the logic cell is required to implement simultaneously all the independent modes. The dynamic power system is also higher because the synchronous "inactive" modes still draw power on each clock cycle. The third design approach implements only the correct circuit for the current operating mode. Furthermore, the static power is lower than the others. Dynamic power is also reduced because of the lower toggle rate at each clock period. The partial reconfiguration method has savings of the power consumption which are around 50% than Velcro and around 35% than parametrization.

However, from our point view, the partial reconfiguration approach is not always perfect. The reconfiguration overhead is the major parameter to limit its use. As we know, the time to change one standard to another, in Velcro approach, is one clock cycle. The time to reprogram the standard in the architecture parameterization needs also several clock cycles. An added reconfiguration time is necessary to use partial reconfiguration. This time is dependent on the size of the configuration file and on the way to download it into FPGA. In our experimentation, For a configuration file with a size 13312 words (32 bits), the global reconfiguration time is $T_{global} = T_{MicroBlaze\ overhead} + T_{SRAM-to-ICAP} = 1.2us + (13312/freq_{clock})us = 134,32us$. The configuration time is obtained by a hardware timer implemented in FPGA. The power required in the partial reconfiguration processing is not indicated in this work. It could be small with respect to the operating power of a circuit. Because the operating modes of a circuit are not frequently changed, the power consumption time is small for a long processing time.

The propriety of three different design methods are concluded in table IV resting on four aspects: processing rate (area), performance, power consumption and configuration overhead.

TABLE IV
PROPRIETY OF DESIGN METHODS

	Area	Performance	Power	Conf-overhead
Velcro	—	—	—	+
Parametrization	-	+	-	-
Partial reconfiguration	+	+	+	—

We can see in table IV, velcro design approach provides the lowest configuration overhead because the switching time between different standard is holding only one clock period. However, the highest area waste makes the system power inefficient. Parameterization approach has less area waste than Velcro according to these attached parameters for reprogramming between different standard. Finally, Partial reconfiguration helps to increase power efficiency and FPGA logic cell use rate. However, an added configuration time must be consisted in the operating process and also, it is an unwanted power consumption

VI. CONCLUSION

In this paper, three design methods are implemented and analyzed through comparing its on-chip area occupation, performance, and power consumption. The implementation

result demonstrates the partial reconfiguration as it is an effective way to increase the processing rate of logic cell and accordingly to reduce 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 lonidle 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

REFERENCES

- [1] J. H. Reed *Software Radio A Modern Approach to Radio Engineering*, Prentice Hall, Upper Saddle River, NJ 13-18, 2002.
- [2] H Lee and a.l., *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] L. Alaus, D. Noguét and J. Palicot *A Reconfigurable Linear Feedback Shift Register Operator for Software Defined Radio Terminal*, ISWPC, Santorini, Greece, May 2008.
- [5] J. Palicot, C. Roland *FFT: a basic Function for a Reconfigurable Receiver*, ICT'2003, Papeete, Tahiti, Feb. 2003.
- [6] V. Rodriguez, C. Moy and J. Palicot *Install or invoke?: The optimal trade-off between performance and cost in the design of multi-standard reconfigurable radios*, *Wirel. Commun Mob Comput* 6:1-14 2006.
- [7] Sufi Tabassum GUL, Christophe MOY, Jacques PALICOT *Two Scenarios of flexible multi-standard architecture designs using a multi-granularity exploration*, PIMRC'07, 2007.
- [8] P. Elias *Coding for Noisy Channels*, IRE Conv. Rec., Part 4, pp. 37-47, 1955.
- [9] 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.
- [10] www.xilinx.com.
- [11] 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.
- [12] P. Lysaght, B. Blodgett, 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.
- [13] Cindy Kao *Benefits of Partial Reconfiguration Take advantage of even more capabilities in your FPGA*, Xcell Journal Xilinx, vol. I, pp. 65-67, 2005.
- [14] C. Moy, J. Palicot, C. Giri, V. Rodriguez, C. Roland *Optimal Determination of Common Operators for Multi-Standards Software-Defined Radio*, (WSR'06), Karlsruhe (Germany), March 22-23, 2006.
- [15] V. Rodriguez, C. Moy, 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.