Software Defined Radio Architecture Survey for Cognitive Testbeds

Mickaël Dardaillon, Kevin Marquet , Tanguy Risset, Antoine Scherrer
Université de Lyon, Inria,
INSA-Lyon, CITI,
F-69621, Villeurbanne, France
Emails : {mickael.dardaillon, kevin.marquet, tanguy.risset}@insa-lyon.fr, antoine.scherrer@inria.fr

Abstract—In this paper we present a survey of existing prototypes dedicated to software defined radio. We propose a classification related to the architectural organization of the prototypes and provide some conclusions about the most promising architectures. This study should be useful for cognitive radio testbed designers who have to choose between many possible computing platforms. We also introduce a new cognitive radio testbed currently under construction and explain how this study have influenced the test-bed designers choices.

Keywords-Software radio, Cognitive radio, Computer architecture, Reviews, Digital communications

I. INTRODUCTION

Radio technologies have been developed in a static paradigm: protocols, radio resources allocation and access network architecture were defined beforehand, providing non-adaptable radio systems. Nowadays, the saturation of radio frequency bands calls new era of radio networking which will be characterized by self-adaptive mechanisms. These mechanisms will rely on *software radio technologies*.

The concept of software radio has been coined by J. Mitola in his seminal work during the early 90's [1]. While implementing the whole radio node in software is still an utopia, many architectures now hitting the market include some degree of programmability. Unfortunately, there is no agreement on the hardware architecture embedded in a mobile terminal with SDR facilities. Various technologies are used: ASIC, FPGA, DSP, GPP, etc. These technologies are often mixed and sometimes the term configurable is more adequate than programmable for them.

Studying architectures for emerging SDR systems is of crucial importance because of the need to define the hardware abstraction layer of SDR systems: the *radio hardware abstraction layer* (R-HAL). We believe that it is also of first importance for testbed providers, as we explain later in the paper.

In 2010, two important surveys where published [2], [3]. In [3], Tore Ulversøy provides a very complete review of SDR challenges related to software architecture, computational requirements, security, certification and business for SDR systems. Some SDR architecture prototypes are mentioned but are not the main topic of the study, and many other

prototypes have been delivered since 2010. In [2], Palkovic et al. provide a precise comparative study between the Imec Bear Platform and other important SDR multi-core architectures. The comparison is made for architectures and programming flows. Our study focuses more precisely on architectures of SDR platforms available up to now. Programming models and programming tools used in these platforms are very important topics but much less mature today, most of these platforms being currently programmed "by hand". All the performances results presented in this paper are taken from bibliography.

Another goal of our contribution is to show how such a technological survey can be useful for cognitive radio testbed providers. In 2009, an NSF report [4] pointed out the lack of cognitive radio testbeds and urges *"The development of a set of cognitive networking testbeds that can be used to evaluate cognitive networks at various stages of their development"*. We are currently involved in the development of a cognitive radio testbed called *CortexLab* which is part of the FIT platform [5]. Our work is motivated by the development of this platform: what is the most adapted SDR node for a cognitive testbed accessed via Internet ?

The rest of the paper is organized as follows: section II provides a brief summary of radio and SDR technology. Section III describes the different platforms and gather them in categories. The analysis of the different categories is made in the section IV. Section V describes the choices made for the *CortexLab* testbed. We draw conclusions in section VI.
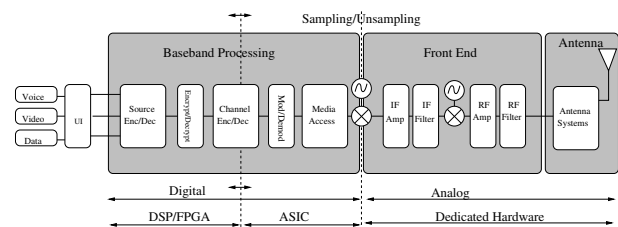
II. SDR TECHNOLOGY

Fig. 1. Radio Block Diagram, highlighting separation between digital and analog parts, as well as programmable, configurable and fixed hardware parts.

The different components of a radio system are illustrated in Fig. 1. Of course, all of the digital components may not be programmable, but the bigger the programmable part

(DSP/FPGA part on Fig. 1), the more *software* the radio. Dedicated circuits are usually needed, for which the term configurable is more adapted than programmable. In a typical SDR, the analog part is limited to a frequency translation down to an intermediate band which is sampled, and all the signal processing is done digitally. To encourage a common meaning for the term "SDR", the SDR Forum (recently renamed Wireless Innovation Forum) proposes to distinguish five tiers. Tier 0 corresponds to hardware radio, Tier 1 corresponds to software controlled radio (only the control functions are implemented in software) and Tier 2 corresponds to software-defined Radio and is the most popular definition of SDR: the radio includes software control of modulation, bandwidth, frequency range and frequency bands. Tier 3 and 4 are not realistic today.

Building an SDR terminal includes choosing a computing platform for the digital part, a sampling frequency and a radio front-end. In addition to the careful choice of a computing platform, the designer must make a trade-of between sampling frequency and terminal complexity. For instance, sampling a signal at 4.9 GHz (hence with a 10 GHz sample rate) is today not available with reasonable power consumption. Even with an evolution to lower power ADC, a high bandwidth ADC would produce more samples, hence require a more powerful or specialized platform. In this paper, we focus on the digital part represented on the left side of Fig. 1.

Finally, *Cognitive Radio* is a wireless system that can sense the air, and decide to configure itself in a given mode. Tier 2 SDR platforms are natural candidates for cognitive radio implementation but cognitive radios do not have to be SDR.

The hardware platforms we review in the following are considered from a SDR point of view. They target the implementation of wireless communication protocol stacks from application down to physical layer (including baseband processing and intermediate frequency conversion), for emission (TX) and/or reception (RX).

III. SURVEY OF HARDWARE PLATFORMS FOR SDR

In order to classify the SDR platforms, we need to define objective criteria. Trying to define criteria based on used technology can be tricky, as most platform are heterogeneous. Moreover, the technology used may not be a relevant criterion for platform users. The user will mainly be interested in the four following features: programmability, flexibility, energy consumption and computing power. Choosing a computing platform for a given application is a trade-of between these cost functions.

However, from the programmer point of view, the architecture is of major importance because it will have a crucial impact on programming models and tools used on the platform. We finally end up with six categories:

- General-purpose CPU approach
- Co-processor approach
- Processor-centric approach
- Configurable units approach
- Programmable blocks approach
- Distributed approach

Each approach is described in their corresponding subsections.

A. General-purpose CPU approach

The general-purpose CPU approach uses familiar computer processor to provide a computing platform, it is depicted in Fig. 2. It offers a flexible and easy way to program the platform, but with a high energy consumption for a performance objective.
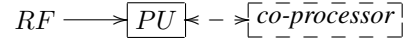
$$RF \longrightarrow \boxed{PU} \dashleftarrow \dashrightarrow \overline{\big[\,co\text{-}processor\,\big]}$$

Fig. 2. General-purpose CPU approach with optional co-processor

USRP: The Universal Software Radio Peripheral (USRP) [6] is representative of the General-purpose CPU approach. It is composed of high frequency ADC/DAC which sample the signal in intermediate frequency. A FPGA converts and stores baseband signal. Most of the signal processing is done by a CPU connected to the FPGA by a USB link (USRP1) or a ethernet link (USRP2). The platform is widespread and supported by third party software. It is aimed to work with GNU radio, but is also compatible with National Instruments LabView and Mathworks Matlab.

Quicksilver: The Quicksilver [7] module is similar in behaviour with the USRP. However, it is only able to receive RF signals.

Microsoft SORA: Recently, Microsoft developed SORA [8]. This platform is connected to the computer by a PCIe bus, which permits low latency and high throughput data transmission. It makes extensive use of modern CPU features to perform 802.11b/g processing in real time.

With the advance of Moore Law, one could imagine that future computers will be able to compute all protocols in real time. However, as shown in [3], the increase in data throughput is higher than the increase in computing power. Therefore, this kind of architecture will only be able to support past protocols, unless it can make use of higher parallelism.

B. Co-processor approach

In order to accelerate the signal processing, optimizations of the General-purpose CPU approach have been explored recently. They rely on the addition of a co-processor to perform heavy processing. It reduces the price to pay in terms of energy while keeping high programmability and flexibility.

The work presented in [9] uses a GPU as a co-processor in a GNU radio flow. It permits gains of a factor 3 to 4 in processing speed.

KUAR: The Kansas University Agile Radio (KUAR) [10] uses an embedded PC associated to a FPGA. The choice of the model of computation is left to the programmer, ranging from a full VHDL implementation (category described in subsection III-E) to a full processor implementation close to the GNU radio flow.

Other developments use generic DSP as central processor, which provides higher efficiency while keeping high programmability.

Texas Instruments: Texas instruments offers a three-core DSP with specialized symbol and chip rate accelerators. This product provides programming flexibility for WCDMA base cells, with support for up to 64 users and different protocols [11].

Imec ADRES: The ADRES (Architecture for Dynamically Reconfigurable Embedded Systems) [12] developed by Imec is a coarse grain reconfigurable architecture. It is built around a main CPU and the ADRES accelerator. The ADRES is seen by the processor as a VLIW, while being an array of 16 functional units (FU). Each FU is a SIMD processor, which leverages the data parallelism. The processor is programmed using the DRESC compiler [13], in ANSI C. The DRESC compiler generates code to unroll loops and compute them using the ADRES accelerator. The ADRES is aimed at telecommunications, with benchmarks on 802.11n up to 108 Mbps and LTE up to 18 Mbps, with an average consumption of 333 mW [12].

Hiveflex: Hiveflex [14] produces accelerators based on many small cores. These accelerators are scalable in number of cores, depending on the application. All wireless protocols are targeted, from 802.11 to LTE, but no details about computing power or energy consumption are given. The accelerators are sold as soft IP with HiveCC, the company SDK.

These architectures offer only limited task parallelism, which may reduce their efficiency. The next categories fill this gap using tailored architectures with heterogeneous types of processors.

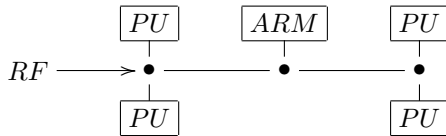### C. Processor-centric approach

Fig. 3.   Processor-centric approach

One approach to get efficient and specialized platforms is to use dedicated processors. In this approach, dedicated processors are used to compute signal processing. The main processor, usually an ARM, is used for control.

The processor-centric approach has a high programmability, but the flexibility of the platform is reduced by its specific architecture. The architecture concept is depicted on Fig. 3.

NXP EVP16: The NXP EVP16 [15], presented in 2005, is composed of several units. An ARM processor provides control and LINK/MAC layers. A conventional DSP, a vector processor and several hardware accelerators are used for signal processing. The vector processor is built as a vectorized pipeline and addressed as a VLIW. It performs UMTS for a 640 kbps bandwidth at 35 MHz, with a maximum of 300 MHz [15].

Infineon MuSIC: Infineon built the MuSIC [16] as a multi-DSP solution for SDR. The control is processed by an ARM processor. Signal computation is processed by 4 SIMD DSP and dedicated processors for filtering and channel encoding. Power consumption in WCDMA mode is 382 mW for the worst case and 280 mW for normal case. This chip is provided as

a commercial solution under the name X-GOLD SDR 20 by Infineon [17]. It is programmed using a mix of C code and assembly code for critical processing.

Sandblaster: The Sandblaster architecture [18] is built around 3 units, the fetch and branch, the integer and load/store and the SIMD vector unit. Task parallelism is managed by a Token Triggered Threading (T^3) component, which provides hardware support for multithreading. On the SB3011 [19], 4 sandblaster cores are integrated and controlled by an ARM processor. It is programmed in ANSI C with a dedicated compiler. Maximum consumption is 171 mW for WCDMA at 384 kbps [19]. The SB3500 is sold as an IP by Optimum Semiconductor Technologies (http://optimumsemi-tech.com).

University of Michigan ARDBEG: The University of Michigan at Ann Arbor developed the SODA [20] SDR platform, and its prototype version ARDBEG [21]. SODA was developed as a complete software SDR solution. It consists of an ARM for control and 4 SIMD DSP for signal processing. ARDBEG builds on that platform by adding hardware turbo decoder and optimizing DSP for signal processing. All programming is made using C code. Consumption results on ARDBEG for WCDMA and 802.11a are under 500 mW [21].

University of Dresden Tomahawk: The University of Dresden, Germany developed the Tomahawk SDR chip [22], aiming at LTE and WiMAX. It uses two Tensilica RISC processors for control, six vector DSP and two scalar DSP for signal processing, as well as ASIC accelerators for filtering and decoding. The scheduling is done by dedicated hardware and C code is used for programming. No protocol has been implemented yet on this platform. From the authors estimation, the platform consumption is about 1.5 W [22].

D. Configurable units approach

In order to offer lower energy consumption, some platforms substitute DSP for configurable units. The difference between specialized DSP and configurable units is very thin, however we think that there is a frontier between these two types of devices.

Fujitsu SDR LSI: Fujitsu developed the SDR LSI [23] in 2005. The platform makes extensive use of hardware accelerators, associated to reconfigurable processors. All these components are connected to a crossbar data network, and controlled by a central ARM processor. The chip was able to run 802.11a/b with a maximum throughput of 43 Mbps [23].

Imec BEAR: The BEAR SDR platform [24] is the evolution of the ADRES from Imec. It is constituted of an ARM processor for control and three ASIPs for coarse time synchronisation on different front ends. Two ADRES coarse grain configurable architectures, as described in subsection III-B, are used for baseband processing with a Viterbi accelerator. The platform can be programmed with C or Matlab code, using the Imec development chain. In terms of energy consumption, BEAR achieves 2x2 MIMO OFDM at 108 Mbps for 231 mW [25]. Imec is licensing the BEAR platform as an IP block.

CEA Magali: The Magali SDR chip [26] is developed by the CEA as a telecommunication demonstration platform. It is

built on a network on chip, each peripheral having an access to the network, with an ARM processor controlling configurations. Computation is done by coarse grain reconfigurable cores called Mephisto and reconfigurable IP for OFDM, decoding and deinterleaving. Smart memory engines are distributed on the NoC and act like DMA, while also providing data rearrangement. The chip performs 4x2 MIMO LTE reception in the most demanding scenario with a consumption of 236 mW [27].

EURECOM ExpressMIMO: The ExpressMIMO is developed as a configurable units approach on a FPGA by EURECOM [28]. All the configurable units share a common network interface, DMA engine and microcontroller, and each as a specific configurable IP for data processing. The board targets OFDM MIMO implementation and uses the open-source OpenAirInterface framework [29].

University of Twente Annabelle: University of Twente, Netherlands developed the Annabelle SDR chip. It is also built on a network on chip, using coarse grain reconfigurable cores. An ARM processor is used for control, and accelerator modules (Viterbi, etc.) are connected to the ARM through an AMBA bus. Only OFDM specific benchmarks have been published at the time of submission.

E. Programmable blocks approach

The last approach uses programmable blocks and is mainly constituted of FPGAs. It doesn't provide programmability as it is, but great flexibility to create tailored architectures. Programmable blocks offer high computing power for moderate energy consumption.

XiSystem: The XiSystem [30] is a VLIW architecture featuring 3 concurrents datapaths, including a PiCoGA (Pipelined Configurable Gate Array). The PiCoGA is an oriented datapath FPGA which executes specific instructions for the processor at run-time. The development is made with C to provide code for both the VLIW and the PiCoGA. It is aimed at embedded signal processing in general, with a benchmark on MPEG2 encoding and an average consumption of 300 mW [30].

Rice University WARP: The Rice University has developed WARP [31], an open SDR platform. The computation is done by a Xilinx Virtex FPGA. Programming uses VHDL language. An open source community is led by the Rice University to offer open source implementations on the platform.

Rutgers University WINC2R: WINC2R is an original platform for SDR developed by the Rutgers University. The platform is built on a FPGA, with softcore processors and accelerators. Softcore processors can be programmed with GNU radio. Computation flow can be balanced on processors or accelerators, depending on the constraints. Moreover, by using an FPGA, accelerators can be chosen and tuned during development. 802.11a has been implemented on the platform [32].

Lyrtech: The Lyrtech company [33] offers development tools and platforms for SDR based on FPGA. Development is done using Simulink model-based approach. The platform is presented as supporting MIMO WiMAX. Many other companies offer similar products based on FPGA ([34], [35] for instance).

F. Distributed approach

The distributed approach has only few elements, distributed control representing a challenge in terms of programmability. We present here SDR platforms using distributed computing.

Picochip: Picochip [36] approaches signal processing using many small cores. These cores are mapped on a deterministic matrix. A C based development tool flow is provided by the company. No benchmark is provided for this chip. However, the company is annoucing OFDM and 4G base stations as reference applications on its website.

UC Davis AsAP: The University of California at Davis developed the Asynchronous Array of Simple Processors [37] (AsAP). This project aims at providing signal processing computation using small processors. All processors can communicate with their nearest neighbours, in a grid like array. The version 2 adds hardware accelerators for FFT, Viterbi and video motion estimation, while increasing the total number of cores to 167. Complete 802.11a/g is processed at 54 Mbps using 198 mW [37].

CEA Genepy: CEA Genepy [27] is using coarser grain for its distributed approach. It is based on Magali [26] technology, using the Network on Chip and the coarse grain configurable cores presented in subsection III-D. The control carried out by the ARM processor is undertaken by distributed small RISC processors. Each cell on the network is composed of two Mephisto cores, one Smart Memory Engine and a RISC controller. The platform is purely homogeneous, with no hardware accelerators. In terms of computing power, 4x2 MIMO LTE reception is processed with a total consumption of 192 mW [27].

IV. ANALYSIS

In order to better understand each category, we summarize the main characteristics for key-platforms that use different approaches in Table I. Energy consumption is not defined for FPGA-based platforms because it is heavily dependent on the configuration. Based on these key platforms, we draw conclusions on the application fields of each category.

	availability	application	prog.	cons.
USRP	commercial	N/A	C++	≈ PC
TI C64+	commercial	base station	C/ASM	6 W
MuSIC	commercial	WCDMA	C/ASM	≤ 382 mW
Sandblaster	IP licence	WCDMA	C	171 mW
ARDBEG	prototype	WCDMA	C	≤ 500 mW
BEAR	IP licence	MIMO OFDM	matlab/C	231 mW
Magali	prototype	MIMO OFDM	C/ASM	236 mW
ExpressMIMO	prototype	MIMO OFDM	C	N/A
WARP	commercial	MIMO OFDM	VHDL	N/A
Lyrtech	commercial	N/A	matlab/VHDL	N/A
ASAP	prototype	802.11a/g	N/A	198 mW
Genepy	prototype	MIMO OFDM	C/ASM	192 mW

TABLE I
MAIN CHARACTERISTICS OF KEY PLATFORMS

If you don't want to study energy consumption nor architecture algorithm adequacy, the general-purpose CPU approach is the easiest way to go. However, if you intend to study

energy consumption or computing power impact, this approach is not recommended. Indeed, dedicated hardware platforms have very different behaviours compared to generic processors. This makes it difficult to establish a relationship between computing power and energy consumption for the generic approach and others. As an example, for a given protocol, computing requirements may vary with a factor of 100 in the literature, depending on the architecture granularity.

In order to study computing power and have the lowest energy consumption, a heterogeneous approach which exploits hardware acceleration is a better starting point. In this family, using DSPs as in Imec's solution [24] or configurable blocks as in Magali [26] is clearly a pragmatic and efficient approach.

Unfortunately, using such a solution makes you heavily dependent on the platform architecture, and porting a waveform to a different architecture can be tricky. Providing a common HAL is a real challenging but promising way to develop practical multi platform SDR.

Alternatively, the programmable blocks approach provides a flexible and efficient platform for prototyping. It can be versatile in the architecture choice, see the radically different approaches from [31] and [32] for example.

From these perspectives, we are now going to address the problem of building a cognitive SDR testbed.
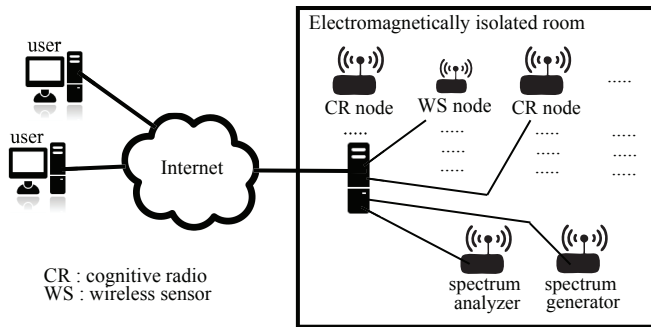
V. COGNITIVE RADIO TESTBED

Fig. 4.   Overview of the CortexLab testbed

Before going into details on the architectural choices, we briefly review existing work on SDR testbeds.

A. Related work

Large-scale cognitive radio testbeds are mandatory to develop and evaluate the performance of upcoming PHY/MAC layers and future cognitive radio algorithms. Whereas numerous testbeds are available in the field of wireless communications (sensor or 802.11-oriented, see for instance Orbit developed at Winlab, Rutgers University), only a few large-scale testbeds have been developed in the SDR and cognitive radio field. Appart from on-going projects such as CREW [38] or TRIAL [39] and some small testbeds involving less than 10 nodes, we found only one testbed developed at Virginia Tech., CORNET [40], where 48 USRP2 with custom RF front-ends have been dispatched in the ceilings of a building, spanning 4 floors. The registered users can remotely program and run experiments on the USRPs. Nodes can be programmed using the OSSIE framework [40] also developed at Virginia Tech.
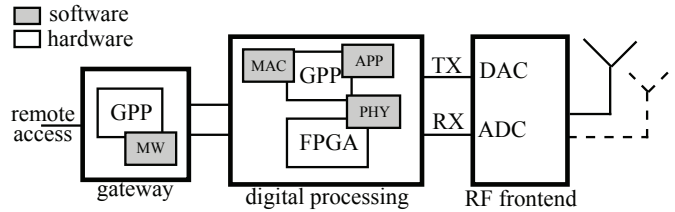

Fig. 5.   Hardware/software requirements for the *CortexLab* cognitive radio node.

B. Cortexlab

We are currently building a new cognitive radio testbed in Lyon named *CortexLab* [41], as part of the Future Internet of Things [5] french funding, we will deploy about 50 cognitive radio nodes together with 50 wireless sensor nodes in a electromagnetically isolated room so as to bring radio propagation under control. The testbed will be open to the scientific community within two years and will allow academics and industrials to conduct real-life cognitive radio experiments. Nodes will be remotely programmable just as if users had them on their desk. Our approach differs from CORNET in that 1) the topology and the room have been selected to target reproducibility and control over the radio propagation and 2) the nodes will have the computing power to run WiFi/LTE in real-time at standard rates and using 2x2 MIMO. We believe USRP2, even with a powerful host PC, cannot achieve such a computing power. The organization of *CortexLab* testbed is illustrated on Fig. 4.

Our main objective is to enable users to run real-time communications with custom APP (application such as traffic generation), MAC (medium access control) and PHY layers implementing state of the art (WiFi, Zigbee) and upcoming (LTE, LTE adv.) standards. The programmability of the platform is a key factor since this has to be done easily and remotely.

Following the conclusions of the previous section, we chose to mix two types of nodes in the testbed: general-purpose CPU nodes and programmable blocks nodes. The general-purpose CPU nodes should be able to run an open source environment (GNU radio or Open Air interface for instance) allowing rapid prototyping at slow data rates, and the programmable ones should be able to run advanced and MIMO PHY layers. Fig. 5 shows a block diagram of a node. The difference between the general-purpose CPU and the programmable node is the size of the FPGA and the function of the PHY layer that are assigned to it. Note that the programming of FPGA is much different from software programming, involving different skills and most of the time different people. The challenge is therefore to abstract those pieces of hardware such that they can be derived from higher-level specifications. We are currently investigating two approaches in this field: high-level synthesis and System-generator coupled with Matlab.

VI. Conclusion

We have reviewed existing platforms for software-defined radio and classified them with respect to their programmability, flexibility, energy consumption and computing power. Although the classification we proposed is clearly arbitrary and based on our experience, we believe this survey gives an up-to-date global view of the available solutions. Based on our study, we saw that some platforms trends are emerging. In our case, as we intend to study computing power and energy consumption while keeping programmability, we chose a mixed FPGA/general-purpose processor platform.

A promising research direction we are investigating at the moment is the design of a software layer able to abstract from the different categories we have seen in this paper.

References

[1] J. Mitola, "Software Radios Survey, Critical Evaluation and Future Directions," in *Telesystems Conference, NTC. National*, (Washington, DC , USA), pp. 13/15 –13/23, IEEE, 1992.

[2] M. Palkovic, P. Raghavan, M. Li, A. Dejonghe, L. Van der Perre, and F. Catthoor, "Future Software-Defined Radio Platforms and Mapping Flows," *Signal Processing Magazine, IEEE*, vol. 27, no. 2, pp. 22–33, 2010.

[3] T. Ulversoy, "Software defined radio: Challenges and opportunities," *Communications Surveys & Tutorials, IEEE*, vol. 12, no. 4, pp. 531–550, 2010.

[4] P. Steenkiste, D. Sicker, G. Minden, and D. Raychaudhuri, 2009. NSF Workshop Report.

[5] "Equipex fit: Future internet of things." http://fit-equipex.fr.

[6] "Universal software radio peripheral (usrp)." http://www.ettus.com.

[7] "Quicksilver." http://www.philcovington.com/QuickSilver.

[8] K. Tan, H. Liu, J. Zhang, Y. Zhang, J. Fang, and G. M. Voelker, "Sora: high-performance software radio using general-purpose multi-core processors," *Communications of the ACM*, vol. 54, no. 1, pp. 99–107, 2011.

[9] P.-H. Horrein, C. Hennebert, and F. Pétrot, "Adapting a SDR environment to GPU architectures," in *Wireless Innovation Forum (SDR Forum)*, (Brussels, Belgium), June 2011.

[10] Minden et al., "KUAR: A Flexible Software-Defined Radio Development Platform," in *2nd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks*, (Dublin, Ireland), pp. 428–439, IEEE, Apr. 2007.

[11] S. Agarwala, A. Rajagopal, A. Hill, M. Joshi, S. Mullinnix, T. Anderson, R. Damodaran, L. Nardini, P. Wiley, P. Groves, and Others, "A 65nm C64x+ multi-core DSP platform for communications infrastructure," in *Solid-State Circuits Conference, ISSCC. Digest of Technical Papers. IEEE International*, pp. 262–601, IEEE, 2007.

[12] B. Bougard, B. D. Sutter, and D. Verkest, "A coarse-grained array accelerator for software-defined radio," *IEEE Micro*, pp. 41–50, 2008.

[13] B. Mei, S. Vernalde, D. Verkest, H. De Man, and R. Lauwereins, "DRESC: A retargetable compiler for coarse-grained reconfigurable architectures," in *Field-Programmable Technology (FPT), IEEE International Conference on*, pp. 166–173, 2002.

[14] "Hiveflex." http://www.siliconhive.com.

[15] K. V. Berkel, F. Heinle, P. P. E. Meuwissen, K. Moerman, and M. Weiss, "Vector Processing as an Enabler for Software-Defined Radio in Handheld Devices," *EURASIP Journal on Advances in Signal Processing*, no. 16, pp. 2613–2625, 2005.

[16] U. Ramacher, "Software-Defined Radio Prospects for Multistandard Mobile Phones," *Computer*, vol. 40, no. 10, pp. 62–69, 2007.

[17] U. Ramacher et al., "Architecture and implementation of a Software-Defined Radio baseband processor," in *International Symposium on Circuits and Systems (ISCAS)*, (Rio de Janeiro, Brazil), pp. 2193–2196, May 2011.

[18] M. J. Schulte, J. Glossner, S. Mamidi, M. Moudgill, and S. Vassiliadis, "A low-power multithreaded processor for baseband communication systems," *Computer Systems: Architectures, Modeling, and Simulation*, pp. 333–346, 2004.

[19] J. Glossner, D. Iancu, M. Moudgill, G. Nacer, S. Jinturkar, S. Stanley, and M. Schulte, "The Sandbridge SB3011 Platform," *EURASIP Journal on Embedded Systems*, pp. 1–16, 2007.

[20] M. Woh, Y. Harel, S. Mahlke, T. Mudge, C. Chakrabarti, and K. Flautner, "SODA: A Low-power Architecture For Software Radio," *33rd International Symposium on Computer Architecture (ISCA)*, pp. 89–101, 2006.

[21] M. Woh, Y. Lin, S. Seo, S. Mahlke, T. Mudge, C. Chakrabarti, R. Bruce, D. Kershaw, A. Reid, M. Wilder, and Others, "From SODA to scotch: The evolution of a wireless baseband processor," in *Microarchitecture, MICRO. 41st IEEE/ACM International Symposium on*, (Como, Italy), pp. 152–163, IEEE, Nov. 2008.

[22] T. Limberg, M. Winter, M. Bimberg, R. Klemm, E. Matus, M. Tavares, G. Fettweis, H. Ahlendorf, and P. Robelly, "A fully programmable 40 GOPS SDR single chip baseband for LTE/WiMAX terminals," in *Solid-State Circuits Conference, ESSCIRC. 34th European*, (Edinburgh, Scotland), pp. 466–469, IEEE, Sept. 2008.

[23] V. Saito and V. Sugiyama, "Single-Chip Baseband Signal Processor for Software-Defined Radio," *FUJITSU Sci. Tech. J*, vol. 42, no. 2, pp. 240–247, 2006.

[24] M. Palkovic, P. Raghavan, M. Li, A. Dejonghe, L. Van der Perre, and F. Catthoor, "Future Software-Defined Radio Platforms and Mapping Flows," *Signal Processing Magazine, IEEE*, vol. 27, no. 2, pp. 22–33, 2010.

[25] V. Derudder, B. Bougard, A. Couvreur, A. Dewilde, S. Dupont, L. Folens, L. Hollevoet, F. Naessens, D. Novo, P. Raghavan, and Others, "A 200Mbps+ 2.14 nJ/b digital baseband multi processor system-on-chip for SDRs," in *VLSI Circuits, Symposium on*, (Kyoto, Japan), pp. 292–293, IEEE, 2009.

[26] F. Clermidy, R. Lemaire, X. Popon, D. Ktenas, and Y. Thonnart, "An Open and Reconfigurable Platform for 4G Telecommunication: Concepts and Application," in *Euromicro Conference on Digital System Design, Architectures, Methods and Tools*, (Patras, Greece), pp. 449–456, IEEE, Aug. 2009.

[27] C. Jalier, D. Lattard, A. Jerraya, G. Sassatelli, P. Benoit, and L. Torres, "Heterogeneous vs homogeneous MPSoC approaches for a mobile LTE modem," in *Conference on Design, Automation and Test in Europe*, (Dresden, Germany), pp. 184–189, Mar. 2010.

[28] D. Nussbaum, K. Kalfallah, C. Moy, A. Nafkha, P. Lerary, J. Delorme, J. Palicot, J. Martin, F. Clermidy, B. Mercier, and R. Pacalet, "Open Platform for Prototyping of Advanced Software Defined Radio and Cognitive Radio Techniques," in *12th Euromicro Conference on Digital System Design, Architectures, Methods and Tools*, (Patras, Greece), pp. 435–440, IEEE, Aug. 2009.

[29] "Open air interface." http://www.openairinterface.org.

[30] A. Lodi, A. Cappelli, M. Bocchi, C. Mucci, M. Innocenti, C. De Bartolomeis, L. Ciccarelli, R. Giansante, A. Deledda, F. Campi, and Others, "XiSystem: a XiRisc-based SoC with reconfigurable IO module," *Solid-State Circuits, IEEE Journal of*, vol. 41, no. 1, pp. 85–96, 2006.

[31] "Warp." http://warp.rice.edu.

[32] S. Satarkar, *Performance analysis of the WiNC2R platform*. PhD thesis, 2009.

[33] "Lyrtech." http://www.lyrtech.com.

[34] "Sundance." http://www.sundance.com.

[35] "Pentek." http://www.pentek.com.

[36] D. Pulley and R. Baines, "Software defined baseband processing for 3G base stations," in *3G Mobile Communication Technologies, 3G. 4th International Conference on (Conf. Publ. No. 494)*, (London, UK), pp. 123–127, IET, June 2003.

[37] D. Truong, W. Cheng, T. Mohsenin, Z. Yu, A. Jacobson, G. Landge, M. Meeuwsen, C. Watnik, A. Tran, Z. Xiao, E. Work, J. Webb, P. Mejia, and B. Baas, "A 167-processor computational platform in 65 nm CMOS," *Solid-State Circuits, IEEE Journal of*, vol. 44, no. 4, pp. 123–127, 2009.

[38] "Cognitive radio experimentation world." http://www.crew-project.eu/.

[39] "Trial environment for cognitive radio and networks." http://www.tekes.fi/programmes/trial.

[40] C. R. A. González, C. B. Dietrich, S. Sayed, H. I. Volos, J. D. Gaeddert, P. M. Robert, J. H. Reed, and F. E. Kragh, "Open-source sca-based core framework and rapid development tools enable software-defined radio education and research," *Comm. Mag.*, vol. 47, pp. 48–55, October 2009.

[41] "Cortexlab: Cognitive radio testbed." http://www.cortexlab.fr.