


# Dynamic Cognitive Radios on the Xilinx Zynq Hybrid FPGA

Shanker Shreejith<sup>1</sup>, Bhaskar Banarjee<sup>1</sup>,  
Kizheppatt Vipin<sup>2</sup>, and Suhaib A. Fahmy<sup>1</sup>

<sup>1</sup> School of Computer Engineering,  
Nanyang Technological University, Singapore, Singapore  
{shreejit1,bhaskar.banarjee,sfahmy}@ntu.edu.sg

<sup>2</sup> Mahindra École Centrale, Hyderabad, India

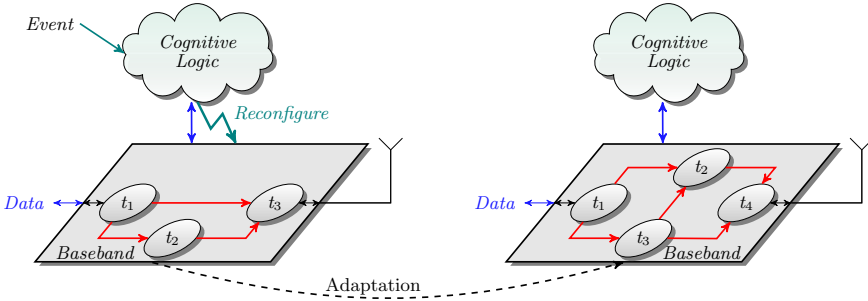
**Abstract.** Cognitive radios require an intelligent MAC layer coupled with a flexible PHY layer. Most implementations use software defined radio platforms where the MAC and PHY are both implemented in software, but this can result in long processing latency, and makes advanced baseband processing unattainable. While FPGA based SDR platforms do exist, they are difficult to use, requiring significant engineering expertise, and adding dynamic behaviour is even more difficult. Modern hybrid FPGAs tightly couple an FPGA fabric with a capable embedded processor, allowing the baseband to be implemented in hardware, and the MAC in software. We demonstrate a platform that enables radio designers to build dynamic cognitive radios using the Xilinx Zynq with partial reconfiguration, enabling truly dynamic, low-power, high-performance cognitive radios with abstracted software control.

**Keywords:** Cognitive radio platforms · Field programmable gate arrays

## 1 Introduction

Cognitive radios can adapt to channel conditions to effectively utilise available radio frequency spectrum. Their adoption is driven by increasing demand for precious frequency spectrum while statically allocated spectrum is often significantly under-utilised by primary users. Designing cognitive radio systems requires consideration on multiple fronts. High performance baseband processing is necessary to support advanced wireless standards with high data throughput. Yet, the baseband should be modifiable at runtime for a wide range of deployment scenarios. Additionally, an easily programmable software stack is necessary to provide higher level functions and programmability (cognitive logic) by application experts. A radio that combines these features can respond to environmental changes to maximise radio performance, as shown in Fig. 1. This performance and flexibility should be achieved within a low power budget to enable deployment in a range of scenarios.

The flexibility requirement has often meant general purpose processors are chosen for cognitive radio implementations. However, processors are not ideally



**Fig. 1.** Baseband adaptation under the control of software cognitive logic, in response to an external event.

suiting to the high throughput signal processing required for baseband processing, and as a result, experimental radios are often implemented on fully-functional desktop computers that consume significant power, precluding deployment in scenarios with restricted power, space, and portability requirements. As a result much cognitive radio systems research has been restricted to investigations in labs.

Field programmable gate arrays (FPGAs) have been used for signal processing for over two decades. They enable highly advanced baseband systems to be implemented within a low computational power budget, by exploiting the fine-grained parallelism found in such algorithms. FPGAs are also volatile devices that can be reprogrammed with different functionality at runtime. However, designing FPGA based systems has remained difficult for non-experts.

Recently, new platforms have emerged that couple high performance embedded processors with a flexible FPGA fabric on a single die. In such systems, the processor can host a fully functioning software stack while the baseband can be implemented in the reconfigurable fabric, with high throughput connectivity between them. This represents a promising platform for cognitive radio systems, offering both high computational performance and flexibility that can be leveraged from higher software layers.

We have developed a prototyping system incorporating the Xilinx Zynq hybrid FPGA, allowing us to combine software programmability for upper layers of the radio with a high-performance flexible baseband implemented in hardware. The software portion of the radio has full control of baseband configuration, and we have abstracted control to enable radio experts to leverage advanced features like FPGA partial reconfiguration. In this paper, we present our platform and a case study, before characterising its dynamic properties. We show that with the proposed abstraction layer, it is possible to bring together the flexibility of software control with the performance of a hardware baseband.

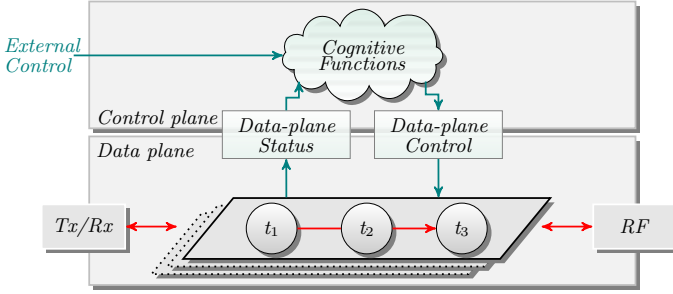
## 2 Background and Related Work

Modern radio protocols rely on flexibility to make efficient use of limited communication spectrum, resulting in the need for highly adaptable baseband and RF processing systems. Research on cognitive radio platforms has focussed primarily on software platforms with component level architectures to afford flexibility as in the case of GNU Radio [1] and Iris [2]. The use of software enables dynamic configurability of the baseband, coupled with an easily programmable MAC layer. While these platforms have been useful for prototyping and academic research, the overhead of implementing advanced baseband processing in software running on general purpose processors means prototyping advanced systems is unrealistic.

Field programmable gate arrays (FPGAs) are silicon devices with a programmable architecture that is flexible enough to implement arbitrary custom datapaths [3]. To implement a datapath circuit, the designer describes it using a hardware description language like Verilog. This design is synthesised and converted, using vendor tools, to a set of configurations that describe how the basic components are to be set up. This “bitstream” is loaded into the FPGA configuration memory to implement the described circuit. As FPGAs are ideally suited to parallel algorithms with large amounts of regular computation, they have been widely used in software radio systems [4]. What makes modern FPGAs attractive for cognitive radios is that besides the high performance of a static datapath implementation, they offer flexibility too.

Multiple radio test beds have leveraged FPGA capabilities for acceleration, like the WARP project from Rice University [5] and the SDC Testbed from Drexel [6]. Iris [2] was also extended with FPGA baseband processing support [7], demonstrating the ability to minimise power consumption in the baseband as channel conditions change [8]. KAUR [9] closely couples a general purpose processing platform, a Xilinx Virtex-II Pro FPGA, and RF front-end in a compact form factor with software and hardware API functions for managing computation. CRUSH [10] integrates a Xilinx Virtex 4 platform with GNU Radio and the USRP front end, offering the performance benefits of a custom baseband datapath, but none of the programmability benefits of FPGAs. CRKIT [11] also aims at integrating hardware baseband processing with software radio management in a system-on chip on an FPGA. It hosts multiple hardware baseband processing chains with support for switching between them and adapting parameters at run-time. Fundamentally, these platforms do offer the performance benefits of hardware, but in many cases, these are limited by the latencies of software-hardware communication due to distinct subsystems being used for the two components, or outdated embedded processor integration on older FPGAs.

Beyond performance, FPGAs offer the advantage of flexibility as they can be reprogrammed with different hardware depending on requirements. Partial Reconfiguration (PR) is an advanced technique that allows parts of the hardware to be modified at runtime while other parts continue to run, enabling designers to swap modules at any given time. Though PR provides adaptability and power



**Fig. 2.** Conceptual view of intelligent cognitive radio separated into control and data planes.

savings, it requires explicit management of reconfiguration and synchronisation, which is difficult in heterogeneous radio systems. Designing efficient PR systems is non-trivial and is an area explored only by FPGA experts. Iris has explored the use of partial reconfiguration, but the software portions of the radio are deployed on a PowerPC hard processor on the Virtex 5 FPGA, resulting in low performance [8]. Furthermore, the use of soft and hard processors in FPGAs remains difficult for anyone other than FPGA designers.

In this paper, we present a radio platform based on the hybrid Xilinx Zynq FPGA that offers a strongly integrated processing system and reconfigurable fabric. It enables low latency data movement and close integration between higher layers of the radio stack and the computational baseband processing with predictable performance. In [12], the authors showed that the tight coupling and predictable latency afforded by moving even MAC layer functions into hardware can improve radio performance. We believe hybrid FPGAs offer an ideal architecture for integrating the computational capabilities of hardware processing with high level management of dynamic radio behaviour. However, design complexity must be addressed if such platforms are to be adopted by the radio community. The team behind Iris demonstrated an initial attempt at using the Zynq processor to run a radio management system, but with no hardware support [13].

We present the first platform to demonstrate interacting software on the ARM processor and a reconfigurable hardware baseband in programmable logic on the Xilinx Zynq. We abstract the low-level baseband management operations, allowing the software radio designer to use high-level function calls to cause parametric and structural reconfiguration of the baseband, simplifying the management process. Our platform integrates a high speed partial reconfiguration controller to allow reconfiguration of the radio baseband with very low latency.

### 3 System Architecture

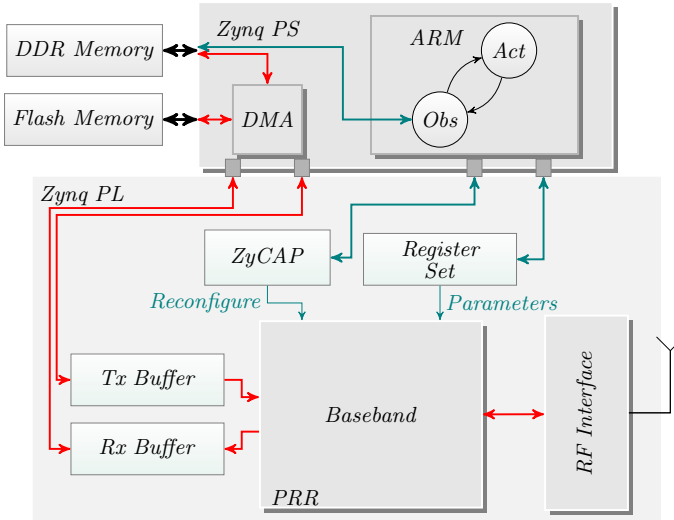
Intelligent cognitive radio designs can be conceptualised as two independent planes, as shown in Fig. 2: the *data plane* which performs the baseband modulation/demodulation and the *control plane* that performs radio control functions; the *cognitive* part. In essence, the control plane implements medium access schemes, enabling the same channel to be used by multiple devices with collision avoidance or detection, and higher layer protocols that ensure reliable transmission. With cognitive radios, the control plane takes on more complex tasks like monitoring channel conditions, triggering sensing, and modifying the configuration of the baseband in response to varying conditions, e.g., by switching the modulation scheme, modifying coding, or changing the baseband transmission standard entirely. The data plane responds to such requests by altering its functions, and should hence support all required types of signal processing to support the various possible modes the system may operate in.

The control plane may use complex intelligent algorithms. Ideally, this cognitive part of the radio should offer maximum flexibility, easy programmability, and being control-intensive, is suited to implementation in software running on a processor. The data plane, however, deals with heavy computational processing on streams of data samples, and so when implemented in software, suffers from long computational latency, and a reliance on powerful processors for advanced baseband schemes or radio standards.

We propose that the strength of FPGAs in data processing be leveraged as in the case of some of the platforms discussed in Section 2, and the data plane be implemented in custom hardware. The key novelty in our platform is to enable dynamic modification of the data plane from the control plane without the need for detailed FPGA knowledge. Previous attempts at building such platforms have used software running on a soft processor on the FPGA fabric with the data plane also in the FPGA as custom hardware blocks, but radio control and reconfiguration still required low-level FPGA knowledge. Essentially, the designer would need to prepare a set of valid hardware configurations for the data plane, store this configuration information in off-chip memory, then manage the loading of the required configurations at runtime through the low-level driver provided by the FPGA vendor. This meant that only FPGA designers could use these systems, and the software programmability was still at a very low-level.

Hybrid FPGAs like the Xilinx Zynq present a compact and efficient architecture for building such software/hardware systems. They tightly couple a highly capable dual-core ARM processing system with a reconfigurable fabric, providing computational capability and flexibility for both the control and data planes. These platforms offer the benefit of a fully functional software side with the ability to add hardware processing in the FPGA fabric. However, managing hardware adaptation on FPGAs at runtime remains difficult and is achieved by a complex sequence of operations, requiring low-level control and knowledge of the underlying hardware.

The unique feature of our proposed framework is that it abstracts such low-level details from the user while also integrating efficient high-speed dynamic



**Fig. 3.** Proposed cognitive radio architecture on the Zynq.

partial reconfiguration for hardware-level support of baseband adaptation. The simplified architecture of our platform is shown in Fig. 3. The Zynq allows a clear partition between the control plane and data plane with its hybrid architecture comprising the processing system (PS) and the programmable logic (PL). A high speed datapath enables data to be moved to/from the external interfaces (like DDR memory or Ethernet in the PS region) to the data plane in the PL region using dedicated direct memory access (DMA).

The logic in the PL implements the baseband, antenna interfaces, buffers and control/status registers for efficient interaction with the control software running on the ARM core. A key benefit of our platform is that the RF interface is directly connected to the baseband chain, avoiding the need for a round-trip in software as required by some other platforms. This ensures minimal latency and high throughput. The *Tx* and *Rx* buffers form the high-speed data interface between the PS and baseband processing system in the PL. DMA-based data movement allows high speed full-duplex data streaming to the Tx/Rx buffers from the software or other interfaces within the PS, like Ethernet. Baseband control and status monitoring is established via the *Register Set*, that provides configuration and status information for both *Tx* and *Rx* interfaces. The register set also sets the parameters of the RF interface, providing a unified view of all the parameters a radio designer may wish to modify at runtime.

A baseband radio chain with tunable parameters is loaded into the PL at system start-up. The control registers in the register set configure the parameters of the baseband, allowing them to be altered at run-time. This provides fast adaptation without requiring any changes in the physical design of the baseband block, and is called parametric reconfiguration. This works for small changes like modifying the carrier modulation scheme or selecting a new coding scheme.

The antenna interface provides the interface to off-the-shelf RF boards, and we initially support the Analog Devices AD-FMCOMMS4-EBZ (based on the AD9364) that interfaces with the FPGA using an FPGA Mezzanine Card (FMC) connector.

The control plane implements the *Observe (Obs)–Decide–Act (Act)* loop, which observes events, either through changes in values in the register set, or external events triggered from software. If modification of the baseband is required, it can trigger parametric and/or physical reconfiguration. Higher layer protocols may be integrated on top of this to provide a complete network stack.

While parametric reconfiguration allows us to modify some aspects of the baseband, any significant changes, e.g. changing from sensing mode to transmission, require more significant changes in hardware. This is achieved using partial reconfiguration, which is standard across the entire range of Zynq devices and other 7-series FPGAs from Xilinx. The baseband processing chain is implemented in a partially reconfigurable region (PRR) within the PL, enabling its physical implementation (and thus function) to be modified at runtime beyond just parametric changes. Effectively, the whole PRR can be replaced with new hardware blocks at runtime by writing new configuration bits into the FPGA’s configuration memory.

The standard PR flow supported by Xilinx requires extensive understanding of FPGA architecture and programming such reconfiguration is complex, requiring low-level access to memory addresses, and understanding of configuration *bitstream* details, making it difficult for radio designers. Furthermore, the reconfiguration speed attainable with the supported flow is very slow, resulting in considerable latency when switching between different baseband modes.

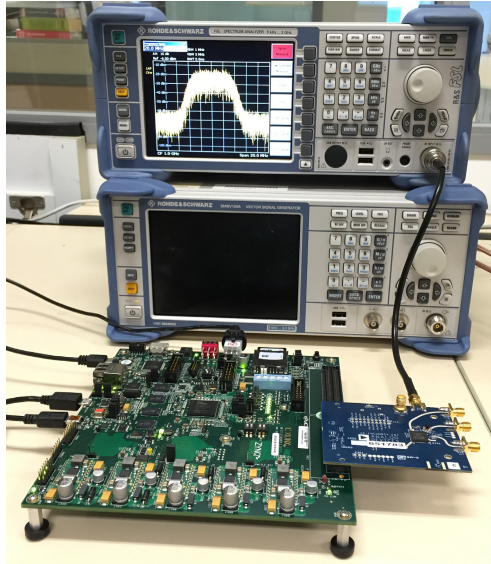
To address this challenge, we use ZyCAP, a custom-designed reconfiguration manager that enables seamless management of partial reconfiguration from the ARM processor via its own software/libraries [14]. ZyCAP is added as a peripheral to the PS and connected to the internal configuration circuitry of the FPGA. At the software level, the ZyCAP driver manages the low level commands for reconfiguration, memory organisation for the different bitstreams, and performance enhancements like bitstream caching, all of which are abstracted from the system designer through high-level API functions.

The application designer is able to call different physical configurations of the baseband using function calls like `set_baseband(receive1)`. The driver handles all the steps required for physical reconfiguration. ZyCAP provides non-blocking operation, which returns from the reconfiguration call immediately, allowing processor load to be minimised, and hence supporting more complex cognitive algorithms. By using DMA-based bitstream reconfiguration, ZyCAP also minimises reconfiguration time by a factor of 20 times or more.

## 4 Case Study

To evaluate the capabilities of our platform, we have implemented a dynamically modifiable radio on a Xilinx ZC702 evaluation board. We have created a





**Fig. 4.** Test setup for DVB-S/C case study.

hypothetical scenario with two baseband standards based on the digital video broadcasting (DVB) cable (DVB-C) and satellite (DVB-S) standards. The two baseband processing chains are distinct and implemented as custom hardware designed in Verilog. Each configuration supports a number of parameters that modify the coding mechanism (like modes of convolutional/differential coding in DVB-S/C) or a variation in the code rate ( $2/3$ ,  $3/4$ , or  $7/8$ ). The parameter changes for both baseband configurations are present as multiplexed hardware and the active path is chosen by setting multiplexer control signals in the register set. This allows low latency parametric adaptation, representing system changes that may be required to adapt to instantaneous channel conditions using the same baseband scheme. Switching between baseband schemes requires partial reconfiguration of the FPGA.

The baseband output is interfaced to the Analog Devices AD-FMCOMMS4-EBZ FMC module with a tunable operating frequency, providing a highly flexible air interface. The transceiver is configured over an SPI interface from the PS providing complete software control over the data-plane (from baseband to RF). Fig. 4 shows the laboratory setup for evaluating the case study.

Our experiments aim at quantifying overall data-path latency and the delay incurred for data-plane adaptation (both parametric and full reconfiguration adaptation). For our experiments, we have simple software control in C that initialises the baseband modules and initialises transmission of data: no medium access control is implemented. Baseband adaptation is managed from software by modifying the transmit and receive status registers, that can trigger a parametric reconfiguration or a physical hardware reconfiguration. In a full cognitive radio



**Table 1.** Resource utilisation on ZC-7020.

| Function | LUTs  | FFs   | BRAMs | DSPs  |
|----------|-------|-------|-------|-------|
| DVB-S    | 1487  | 2812  | 0     | 24    |
| DVB-C    | 1109  | 2580  | 0     | 24    |
| PRR      | 5400  | 8000  | 50    | 40    |
| RF I/F   | 13389 | 21086 | 15    | 69    |
| Reconfig | 806   | 620   | 0     | 0     |
| Total    | 15682 | 24518 | 15    | 93    |
| (%)      | 29.5% | 23%   | 10.6% | 42.3% |

implementation, more complex software can be used to decide on the correct configuration, and would use the same abstracted interface. We keep the software simple to provide us with meaningful latency numbers in our experiments.

Table 1 shows the resource utilisation of the different modules in the case study. The PRR is large enough to include all resources required for the DVB-S or DVB-C baseband scheme, with PR used to switch between them. It still consumes a minimal amount of resources considering the simplicity of the baseband in this case. The entire design does not consume more than 42% of the resources (DSPs) in the relatively small Zynq XC7020 device. More complex baseband schemes based on OFDM would consume more resources, but our initial experiments have shown that a flexible OFDM baseband consumes just over half the resources on this same device.

Table 2 shows the end-to-end latency of the data-plane for transmitting one complete frame of 188 Bytes. The path delay is composed of two components: the delay for loading data from external memory (DDR) into the internal buffers and the processing delay of the baseband logic. The packet from the external DDR memory is loaded into (or read from) the *Tx* (*Rx*) buffer through a dedicated DMA into the baseband, enabling high speed uninterrupted data flow. The path latency of the processing chain depends on the different baseband configurations and different parameter settings chosen by the control software at runtime. We can see that overall latency is dominated by the baseband logic and is largest in case of DVB-S with the 1/2 coding rate. These latencies are  $400\times$  less than what can be achieved by implementing the baseband in software (in C) running on the ARM processor in the Zynq, with the DVB-S baseband consuming 39.18 ms to encoding each frame at 1/2 code rate. It is also worth noting that data movement from external memory to the hardware baseband consumes only a fraction of the total time, and can be effectively hidden by overlapping data movement with baseband operation.

To determine the latency incurred during parametric and physical reconfiguration, the Tx/Rx status registers were used to trigger changes in the system from the control plane. Parametric adaptation incurred a delay of 150 ns from the time the register values were changed and hence detected by the software

**Table 2.** End-to-end latency of the Data Plane at 100 MHz.

| DDR Latency  | Baseband Code Rate | Latency           |
|--------------|--------------------|-------------------|
| 3.70 $\mu$ s | DVB-S              | 1/2 96.25 $\mu$ s |
|              |                    | 2/3 72.19 $\mu$ s |
|              |                    | 3/4 64.19 $\mu$ s |
|              |                    | 5/6 57.73 $\mu$ s |
|              |                    | 7/8 54.98 $\mu$ s |
|              | DVB-C              | NA 24.06 $\mu$ s  |

(periodic polling). The delay incurred accounts only for the write path delay from the processing system to the Tx/Rx control registers in the register set as these controls are directly wired to multiplexers controlling the different paths. A complete baseband adaptation using PR incurred a delay of 786  $\mu$ s from status decode, primarily due to ZyCAP achieving a reconfiguration throughput of 380 MB/s — nearly 95 percent of the theoretical bandwidth. This is more than  $3\times$  faster than the normal blocking reconfiguration control possible in the Zynq.

## 5 Conclusion

Cognitive radio systems require highly flexible hardware support for implementing adaptive and computationally complex baseband functions, with further constraints on the power budget for mobile applications. Hybrid FPGAs like the Xilinx Zynq show promise for such platforms as they closely integrate adaptability at the hardware level with a computationally capable processing system. However, managing runtime adaptation through reconfiguration and exploiting the benefits of partial reconfiguration on the Zynq is generally too difficult for radio designers used to software, instead requiring experienced FPGA engineers. In this paper, we have presented a cognitive radio prototyping platform based on the Xilinx Zynq which uses a high level reconfiguration management system to abstract low level details of hardware management from the application designers.

We demonstrated a case study with a DVB baseband, showing that hardware level adaptation (including parametric and full baseband reconfiguration) can be achieved with minimal latency, while still being abstracted. This opens the door to radio designers with no FPGA experience to benefit from the capabilities of new hybrid architectures like the Zynq to build dynamic radios with minimal latency, high baseband performance and true hardware reconfigurability.

We are working on a public release of our framework and developing a library of baseband blocks for flexible OFDM cognitive radios in the hope that more radio designers will be able to benefit from this technology.

## References

1. GNU Radio. <http://www.gnuradio.org/> (accessed March 2015)
2. Sutton, P.D., Lotze, J., Lahlou, H., Fahmy, S.A., Nolan, K.E., Ozgul, B., Rondeau, T.W., Noguera, J., Doyle, L.E.: Iris: An architecture for cognitive radio networking testbeds. *IEEE Communications Magazine* **48**(9), 114–122 (2010)
3. Kuon, I., Tessier, R., Rose, J.: FPGA architecture: Survey and challenges. *Foundations and Trends in Electronic Design Automation* **2**(2), 135–253 (2008)
4. Cummings, M., Haruyama, S.: FPGA in the software radio. *IEEE Communications Magazine* **37**(2), 108–112 (1999)
5. Amiri, K., Sun, Y., Murphy, P., Hunter, C., Cavallaro, J.R., Sabharwal, A.: WARP, a unified wireless network testbed for education and research. In: *IEEE International Conference on Microelectronic Systems Education*, pp. 53–54 (2007)
6. Shishkin, B., Pfeil, D., Nguyen, D., Wanuga, K., Chacko, J., Johnson, J., Kandasamy, N., Kurzweg, T.P., Dandekar, K.R.: SDC testbed: software defined communications testbed for wireless radio and optical networking. In: *International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, pp. 300–306 (2011)
7. Lotze, J., Fahmy, S.A., Noguera, J., Ozgul, B., Doyle, L.E., Esser, R.: Development framework for implementing FPGA-based cognitive network nodes. In: *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)* (2009)
8. Fahmy, S.A., Lotze, J., Noguera, J., Doyle, L.E., Esser, R.: Generic software framework for adaptive applications on FPGAs. In: *IEEE Symposium on Field Programmable Custom Computing Machines*, pp. 55–62 (2009)
9. Minden, G.J., Evans, J.B., Searl, L., DePardo, D., Petty, V.R., Rajbanshi, R., Newman, T., Chen, Q., Weidling, F., Guffey, J., Datla, D., Barker, B., Peck, M., Cordill, B., Wyglinski, A.M., Agah, A.: KUAR: a flexible software-defined radio development platform. In: *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, pp. 428–439 (2007)
10. Eichinger, G., Chowdhury, K., Leeser, M.: Crush: cognitive radio universal software hardware. In: *International Conference on Field Programmable Logic and Applications (FPL)*, pp. 26–32 (2012)
11. Le, K., Maddala, P., Gutterman, C., Soska, K., Dutta, A., Saha, D., Wolniansky, P., Grunwald, D., Seskar, I.: Cognitive radio kit framework: experimental platform for dynamic spectrum research. *ACM SIGMOBILE Mobile Computing and Communications Review* **17**(1), 30–39, January
12. Di Francesco, P., McGettrick, S., Anyanwu, U.K., O’Sullivan, J.C., MacKenzie, A.B., DaSilva, L.A.: A split MAC approach for SDR platforms. *IEEE Transactions on Computers* **64**(4), 912–924 (2015)
13. van de Belt, J., Sutton, P.D., Doyle, L.E.: Accelerating software radio: iris on the synq SoC. In: *IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, pp. 294–295 (2013)
14. Vipin, K., Fahmy, S.A.: ZyCAP: Efficient partial reconfiguration management on the Xilinx Zynq. *IEEE Embedded Systems Letters* **6**(3), 41–44 (2014)