Prototype of a Spectrum Sniffer Software Implementation for ISM Bands

Ermano Picco, *ST Microelectronics*

E-mail: ermano.picco@st.com

***Abstract*—In a Software Defined Cognitive Radio the goal to exploit at maximum the under-utilized spectrum portions can be achieved only performing reliable spectrum sensing. This is obtained through a tunable full software implementation of cyclostationary spectrum sensing algorithms running on a reconfigurable, programmable SoC that embeds a core subsistem dedicated to the sensing routines. The engine will be used as a proof of concept to test on the field the baseband sensing algorithms.**

Furthermore the demonstrator will be part of ST contribution to the EU-Funded Project ARAGORN [1] – Adaptive Reconfigurable Access and Generic Interfaces for Optimization in Radio Networks.

***Index Terms*—Software Defined Radio, Cognitive Radio, Multicore Embedded Systems, Reconfigurable SoC, Spectrum Sensing.**

I. Introduction

In a home scenario, as long as we see growing the number of devices operating in the ISM bands, efficient use of the spectrum at frequencies that can be used for commercial wireless communications has become a key factor for the deployment of cohexisting new applications able to offer performing services. In such conditions secondary users must sense the presence of a primary user before using the spectrum; hence their primary task will be to measure the spectrum energy so to reliably determine whether a portion of spectrum is currently used by other devices.

II. Spectrum Sniffer Implementation

A. Reliable Spectrum Sensing

Among the several techniques useful to sense the spectrum energy, we chose to implement a cyclostationary algorithm for its low complexity, system independence and noise burst robustness.

Complexity is a main issue to consider when implementing a spectrum sniffer over a reconfigurable programmable architecture, so cyclostationary techniques performing basic arithmetic operations are best suited to this purpose.

System independence is advisable for a device operating in an ever changing environment so, as long as most of the wireless communication systems are based on cyclostationary waveforms, a cyclostationary technique will be able to detect such primary users in Rayleigh multipath fading channel conditions. Moreover the flexibility of a software implementation gives the advantage of fine tune the sniffer parameters adapting it to the particular environment without modifying the hardware.

Robustness to noise bursts is important to properly detect the presence of primary users even in presence of AWGN at low SNR. Again cyclostationary algorithms have shown to be less sensible to uncorrelated noise bursts. Then it is important to be able to fine tune the sniffer parameters in order to minimize the missing probability, hence the interference with primary users and obtain an fair false detection probability.

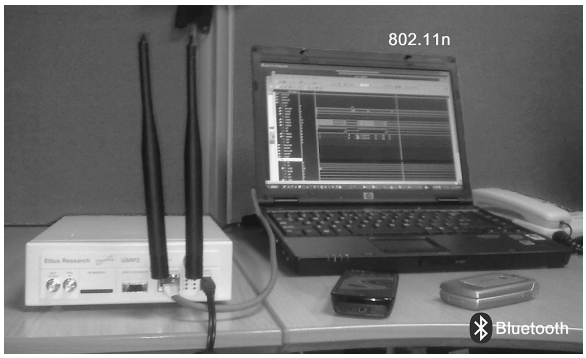

Fig. 1 The demonstrator operating in a typical home scenario

The challenge of cyclostationary detection in cognitive radio involves effects of time-varying multipath channel. In this section we derive effects of time-varying multipath channel on cyclic autocorrelation and spectral correlation function. We assume that cyclostationary features of primary user signal are known to cognitive radio user which is reasonable in this case. Thus some issues dealing with selecting proper set of cyclostationary features to be detected can be obtained.

jp).

B. Cyclostationary Based Spectrum Sensing Algorithm

Algorithm Arithmetic Description

We define signal model

$$y(t) = \int_{-\infty}^{+\infty} h(\tau;t)x(t-\tau)d\tau$$

where $y(t)$ and $x(t)$ are the received and transmitted signal and

$$h(\tau;t) = \sum_{n=1}^{N} \alpha_n(t) e^{-j\varphi_n(t)} \delta(\tau - \tau_n(t)) \quad [2]$$

is the time-varying channel impulse response where N represents the number of multipath components, $\varphi_n(t)$ the phase shift, $\tau_n(t)$ the path delay and $\alpha_n(t)$ the amplitude. Let's assume that the channel model has a Rayleigh-distributed amplitude and uniform phase without LOS components and that its in-phase and quadrature components are independent Gaussian processes with same auto-correlation, zero mean and zero cross-correlation. Let's also assume that the delay spread and Doppler spread are separable. In such hypothesis, the channel autocorrelation can be written as

$$A_n(\tau;\Delta t) = E\left[h^*(\tau;t)h(\tau;t+\Delta t)\right]$$

From the definition presented previously, cyclic autocorrelation function of a received signal is defined as

$$R_y^{\alpha}(\Delta t) = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} E\left[y(t)y^*(t+\Delta t)\right] e^{-j2\pi\alpha t} dt$$

$$\lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} E\left[y(t)y^*(t+\Delta t)\right] e^{-j2\pi\alpha t} dt = \int_{-\infty}^{+\infty} A_n^*(\tau;\Delta\tau) R_x^{\alpha}(\Delta t) e^{-j2\pi\alpha\tau} d\tau$$

where

$$R_x^{\alpha}(\Delta t) = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} E\left[x(\tau)x^*(\tau+\Delta t)\right] e^{-j2\pi\alpha\tau} d\tau$$

is the cyclic autocorrelation function of the transmitted signal over the cyclic frequency α and $A_h^*(\tau;\Delta\tau)$ is the complex conjugate of autocorrelation function of $h(t)$.

Now, since the channel response $A_h^*(\tau;\Delta\tau)$ is approximately independent at frequency separations $\Delta t > B_c$, coherence bandwidth, and the time-varying channel decorrelates after approximately channel coherence time T_c, properly choosing $\alpha < B_c$ and $\Delta t < T_c$ for the primary user cyclostationary signals, we will have $R_y^{\alpha}(\Delta t) \to 0$ when received signals don't match these conditions.

Finally we identify

$$S(f,\alpha) = \sum_{\tau=-\infty}^{+\infty} R_y^{\alpha}(\tau) e^{-j2\pi f\tau} \quad [3]$$

as the Cyclic Spectral Density function of the received signal as the main test to detect whether a primary user is occupying the channel.

Algorithm Implementation and examples

The above specified algorithm applies to the typical signals that can be found in the ISM Band like Bluetooth or 802.11a/b/g/n.

In particular we can see that all this kind of signals have a periodic symbol structure which we can exploit to detect the presence of active devices in the neighborhood.

Let's consider for example an 802.11n transmission: as we can see in figure 2 (where a legacy 802.11a/g packet is considered for simplicity) the structure of this stream is the typical Orthogonal Frequency-Division Multiplexing (OFDM) based packet subdivided in OFDM symbols of fixed length. The packet starts with a preamble containing a periodic training sequence made of n cyclic symbols which is followed by a signal field containing information on the transmission scheme and then by data symbols. Each information symbol is built mixing data with a midamble constituted by pilot tones that are cyclically placed within each symbol.

So, for the considered Band, let's analyze the Cyclic Autocorrelation Function of a received stream over a period equal to an OFDM symbol duration and with a lag Δt of an OFDM symbol length. We can see that it tends to zero as long as we are presence of additive white Gaussian noise only. Instead when receiving a packet preamble intentionally built as a periodic signal, a strong autocorrelation value ramps up. The preamble is then followed by data symbols that, as far as we consider them uncorrelated, show anyway to be cyclically correlated due to the inserted midamble.

The corresponding average of the Cyclic Autocorrelation

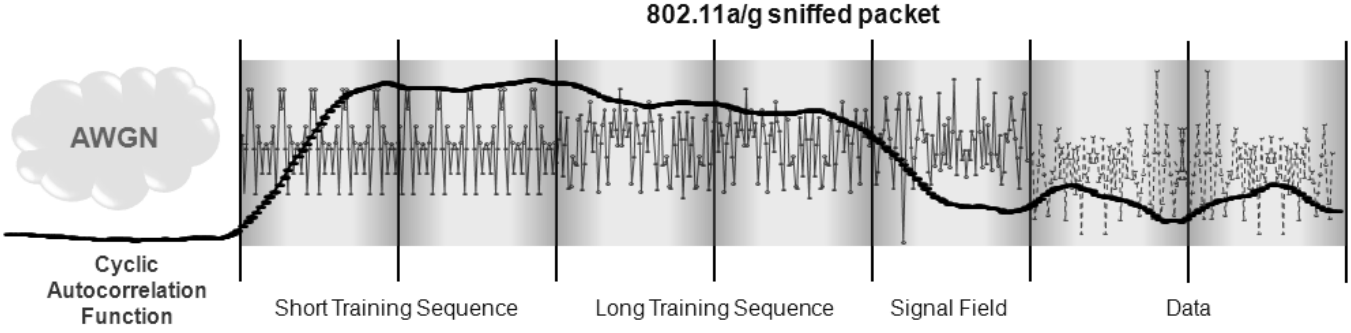

Fig. 2Example: cyclostationary detection of a 802.11a/g primary user

Function over an observation window of values will give us a reliable estimate of the in area Cyclic Spectral Density for the considered channel within the ISM Band. Given this, at upper layers of a Cognitive Radio System a Cognitive Radio Manager will be able to collect these estimates and analyze them to properly set thresholds to decide with a certain confidence interval whether we are in presence of a primary user or not.

C. *Sniffer Architecture*

The main idea in designing the architecture of the sniffer has been to apply the concept of software defined radio down to the lower layer of the stack. From one side this leads to have a complete general purpose architecture avoiding so to embed a dedicated hardware sniffer.

From the other side, the need to cyclically perform sniffing routines in order to be as much reactive as possible to adapt to the spectrum changes can be an issue in case we run all the computation on a single core.

To prove how to overcome this issue we adopted as testbed a commercial off-the-shelves USRP2 (*Universal Software Radio Peripheral*) board including an FPGA to map the whole GNU Radio system; starting from there we have embedded a programmable core dedicated to perform the sniffer routines avoiding so the other core to become the system bottleneck and leaving it to cope with the upper layer computation.

D. *Embedding the Sniffer Subsystem*

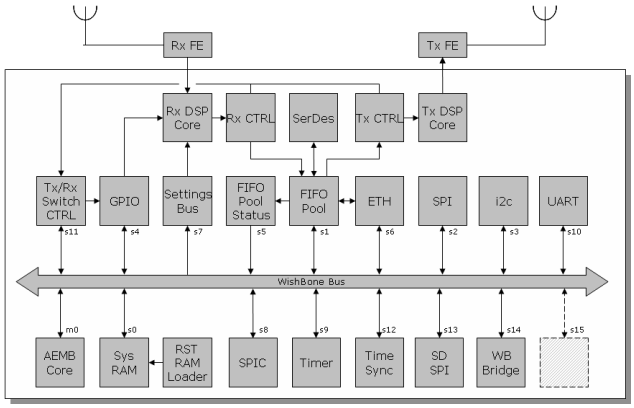

Fig. 3 Original USRP2 SoC Architecture

As we can see from fig. 2, the original USRP2 architecture is a Wishbone Bus based system with a aeMB microprocessor mastering all the peripherals available on-board. As a dedicated coprocessor, the sniffer subsystem can be considered as another on-board device interacting as a slave. So the aeMB master can see its tightly coupled memories (TCM) and read/write data into them. So, for example during the boot phase, the aeMB will be able to initialize the ST Core providing its TCPM (Tightly Coupled Program Memory) and TCDM (Tightly Coupled Data Memory) the firmware code to be run. Another task that can be easily performed this way is passing to the Sniffer the configuration parameters to start a sweep over the spectrum.

In general, as the memory space addressable by the aeMB is tighter than the total Sniffer TCM (Tightly Coupled Memory) space, there can be an issue addressing the whole TCM space. This issue has been overcome by setting the subsystem memory addressing to a fixed logic address seen by the aeMB and using some additional registers that can be read/written in order to displace the logic address by an offset so to properly address the physical address of the TCM.

To avoid doubling the instructions needed to operate on a contiguous memory space, the additional registers are incremented at each memory access. So for example, to read/write n values, a total amount of n+1 instructions is required just to properly set the initial address offset.

The Sniffer core has a proprietary bus interface connected to its own proprietary bus lane for which a bridge has been developed to properly interface Wishbone protocol based devices. Then a dedicated Wishbone bus lane has been connected to let the Sniffer core access the peripherals on board. Among the peripherals there are a Gigabit Ethernet connected to the host, the receiver DSP core and the FIFOs where the baseband captured data are stored to be elaborated.

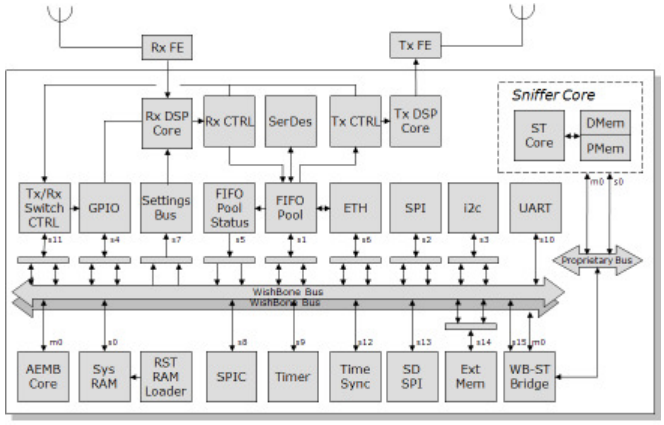

Fig. 4 Modified USRP2 SoC Architecture embedding the Spectrum Sniffer

The two concurrent Wishbone bus lanes have been implemented to allow the maximum degree of parallel access to the on-board resources by the two cores. When a concurrent request of service is presented by both the cores to the same peripheral, a Wishbone compliant arbiter will manage the request.

In the sniffer subsystem the data is moved from and to the FIFOs to the internal data memory directly accessible to the core where effectively the computation takes place.

Once a data block is copied inside the tightly coupled data memory, the core manipulates it computing a cyclostationary based estimate of the spectral energy density for the given

frequency. The estimate is then performed sweeping the whole spectrum according to the RF parameters (maximum and minimum frequency, filter bandwidth, step bandwidth) passed by the aeMB core during the configuration phase or directly by the host.

The estimated energy spectral density is then copied back to the external memory to be accessible to the other layers.

E. *Mapping on board*

As stated above, the Sniffer subsystem architecture has been designed to be embedded on the existing USRP2 system on-chip. The off-the-shelves solution maps the USRP2 SoC on a Xilinx SPARTAN3 XC3S2000 FPGA placed on a board tied to a dual-band RF daughterboard operating at both 2.4GHz and 5GHz thus covering all the wireless LAN frequencies.

Xilinx Spartan 3 XC3S2000	**USRP2 SoC**	
Resource	*Usage*	*%*
Registers	8993	21%
Single Port RAM RAM16X1S	16	
Double Port RAM RAM16X1D	208	
Single Port RAM RAM64X1S	108	
ROM ROM128X1	10	
Block RAM	24	60%
Block MULT	8	20%
Clock Buffers	6	75%
Kgates	740	37%

Table 1 USRP2 SoC FPGA Mapping

Xilinx Spartan 3 XC3S2000	**Modified USRP2 SoC**	
Resource	*Usage*	*%*
Registers	12808	31%
Single Port RAM RAM16X1S	16	
Double Port RAM RAM16X1D	4336	
Single Port RAM RAM64X1S	108	
ROM ROM128X1	10	
Block RAM	40	100%
Block MULT	8	20%
Clock Buffers	6	75%
Kgates	1740	87%

Table 2 Spectrum Sniffer SoC FPGA Mapping

As we can see from the tables 1 and 2, the most critical resource available on the FPGA for the placement of the SoC, are the block RAMs. Block RAMs are used to implement both the aeMB TCM and the Sniffer core TCMs. They are also the main component of the FIFO Buffer Pool used to handle data coming from the Ethernet, from the Tx DSP, the Rx DSP and from the WB Bus. With regard to this, since the total amount of needed Block RAMs by each component would have been greater than the available number, a cut from 8 down to 2 of the FIFOs within the Buffer Pool has been necessary to fit the design on board. This is reasonable as both cores TCMs have been sized to properly store the firmware and the data for this application and cannot be further cut down. Furthermore less available FIFOs can be handled by the firmware/software that for this application, will store the captured data with a ping-pong policy. Congestion of the design mapped on the fpga is about 87% with no further memories available, so there's almost no space to fit other components unless they are simple logic nets. Clocks are tied to the limit as well due to the strong timing requirements needed for peripherals like the Gigabit Ethernet 125 MHz clock.

III. CONCLUSION

In this paper we have presented a demonstrator implementing a full-software Spectrum Sniffer as main part of a Cognitive Radio Physical Layer. This allows us to run it on a fully reconfigurable, programmable SoC embedding a Sniffer core subsystem where the sniffing routines can be run in a timely manner joining the flexibility of a software implementation and the performances of off-the-shelves sniffer equipment.

Limitations of this work are mostly due to the almost saturated physical resources available on board.

Future extensions could consider an expansion to a more capable hardware so to allow the extension of the multi-core system with other cores and more receiver daughterboards to be able to explore benefits from sniffing the spectrum from multiple antennas.

REFERENCES

[1] http://www.ict-aragorn.eu

[2] A. Goldsmith, Wireless Communication, Cambridge, 2005.

[3] U. Gardner, WA, "Exploitation of spectral redundancy in cyclostationary signals," IEEE Signal Processing Mag., vol. 8, no. 2, pp. 14–36, 1991.