INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2012, VOL. 58, NO. 4, PP. 301–306
Manuscript received October 3, 2012; revised Decemer, 2012. DOI: 10.2478/v10177-012-0041-3

A Multichannel Receiver of the Experimental FM Based Passive Radar Using Software Defined Radio Technology

Bogusław Szlachetko and Andrzej Lewandowski

***Abstract*—In this paper we present results of research on multichannel receiver using Software Defined Radio technology. This receiver is a part of the experimental FM based passive radar being designed. The hardware platform of the receiver consists of the Universal Software Defined Radio Peripheral devices. In the paper we propose modifications of the USRP's FPGA configuration and GNU Radio code. These modifications allow for developing six/nine synchronous input channel receiver based on two/tree USRPs respectively. Issues of synchronization of separate USRP devices by assuring synchronous sampling were presented. We also propose the solution of the problem of the alignment of the data streams being sent from USRP devices via USB to the PC host.**

***Keywords*—SDR, passive radar, GNU Radio, USRP.**

I. Introduction

PASSIVE radar can be defined as the radar without a dedicated (active) transmitter sending electromagnetic waves in order to illuminate target objects. Instead of the transmitter part of the system, passive radars utilize third party transmissions (so called "illuminators of opportunity") such as FM radio signals, VHF/UHF, TV signals or GSM transmissions [1]. Hence for passive radars the transmitters and receiver are at different locations which gives the configuration known as the bistatic radar [2].

Using a single pair of transmitting/receiving antennas, it is not possible to determine the exact location of the target. The bistatic range of the radar is the ellipse with antennas at its focal points and the target can lie anywhere on this ellipse [3]. Using multiple pairs of transmitter/receiver antennas (so called multistatic passive radar), multiple bistatic ellipses can be determined. Theoretically all this ellipses have to cross one point – the exact target location. In Fig. 1 this ideal situation is plotted.

In a classical approach, an omnidirectional antenna simultaneously receives the signal arriving directly from the transmitter and the signal reflected by the target object. Hence, a critical limiting factor of such a system is the unwanted interference in the echo signal due to the direct reception of the FM radio signal. Moreover, the received direct signal is much stronger than the signal incoming from the target. One of the solutions of this problem is to employ digital beamforming algorithms. Given the positions of transmitters, the antenna pattern can be formed in a such way that the zero null depths will be located directly on the transmitters' azimuth angles. As a result, the unwanted direct signals will be suppressed.

B. Szlachetko and A. Lewandowski are with the Institute of Telecommunication, Teleinformatics and Acoustics, Wrocław University of Technology, Poland (e-mails: {boguslaw.szlachetko; andrzej.lewandowski}@pwr.wroc.pl).

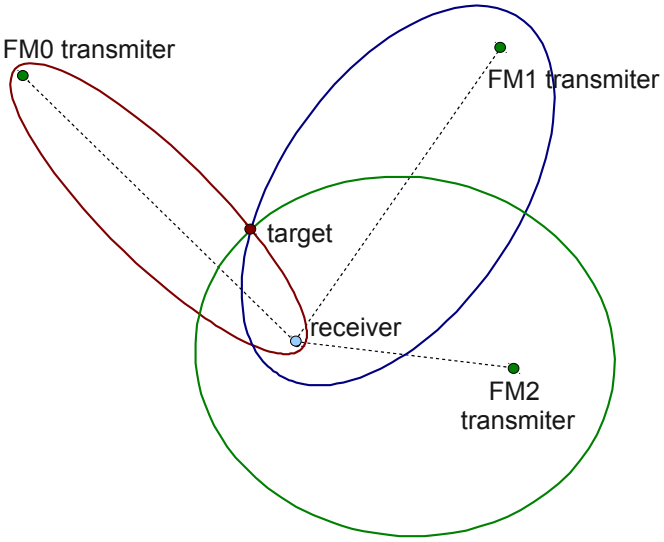

Fig. 1. Contours of constant bistatic ranges in a multistatic scenario

The research and development in radar technology is always an expensive and long process, mainly due to the project production and testing cost of the prototypes of transmitters and receivers, that have to provide the new functionality. Each consecutive experiment that involves applying, for example, different waveform modulation or bandwidth requires changes in the hardware. And this entails a long process of projecting, building and testing of the new hardware before it can be widely used. The use of the Software Defined Radio technology (SDR) in this process remarkably cuts down research and development time. In the SDR the functionality of the system is defined, for the most part, by the software, not the hardware. In the SDR systems necessary modules such as modulators/demodulators, down-converters, filters, etc. are software procedures implemented in a hardware. SDR provides the flexibility in adjusting the transmitter/receiver frequency band, modulation and many other essential parameters by only changing the software while the hardware of the unit remains without any modifications. Once designed and tested, the unit can be used in many different experiments and its properties can be changed in a very short time.

It is evident that signal processing algorithms of the experimental passive radar have be flexible and capable of

accommodating to changing electromagnetic environment of a radar's location. So the SDR approach seems to be the best solution, especially at the research stage.

Employing the SDR technology in communications and radars is not a new idea [4], [5], but it has become more popular in the last few years when the new high speed multichannel AD converters have become purchasable and the GNU Radio project was established. Some researchers try to utilize the UMTS (Universal Mobile Telecommunication System) or DVB (Digital Video Broadcaster) signals to develop passive radar [6]. Petri et al. [7] have performed some experiments with moving objects like cars and trucks obtaining promising results. They argue that the bandwidth of UMTS signal is high (5MHz) and this allows for achieving the better range resolution then utilizing narrower signals. The main disadvantages of their radar are: the short range (only few kilometers) and a lack of UMTS transmitters far away from big urban areas, which restricts the possible applications of this kind of a surveillance system. Other researchers try to utilize the analog television signals. Howland in [8] has shown that the range of his passive radar can exceed 100 kilometers, but at the cost of resolution. This system didn't use SDR architecture principles because in 1999 SDR was a brand new idea. Another passive radar concept utilizing WIFI systems was presented by Rzewuski and Kulpa [9]. Asset of such systems is wide signal bandwidth 20MHz, which provides theoretical bistatic resolution about 15m.

FM-based SDR systems are outstanding among others in the passive radar applications. Many publications present using analog radio broadcasting transmitters as the illuminators [10], [11], [12], [13], [14], [15]. The main advantages of using FM signals are: simplicity of building the receiver (even if one doesn't use SDR approach), wide accessibility of FM signals at all locations in almost all countries, strong power of transmitted signals and high range of detection. Howland in [13] has shown that it is possible to reach bistatic range of 150km (in the real world experiment the bistatic range of 10km with 1.5km resolution resolution was achieved). The research performed by O'Hagan [14] has demonstrated the useful target detection in bistatic range of 70km (20km experimentally).

Our research has been focused on building the multichannel receiver for the experimental, FM-based, passive radar. As the Commercial Off The Shelf (COTS) hardware platform for receiver, the Universal Software Radio Peripheral (USRP) from Ettus company was chosen. These hardware devices are widely used in many application. Friedman et al. have built the system for estimating the angle of arrival for relative interferometric localization [11]. This system consists of the specialized transmitter and the receiver both realized on the USRP devices. Berizzi et al. in [6] have demonstrated the using USRPs in the short range, high resolution surveillance radar system for detecting land vehicles. Heunis et al have presented the prototype of a passive radar utilizing FM transmitter as "illuminators of opportunity" [12]. This prototype allows them to capture the signals for only two synchronous channels, so there is a problem with separating the direct signal from the echo. The beamforming algorithm can't be used in this case because of a lack of sufficient number of synchronously captured signals. Our receiver solves this problem providing six or nine synchronous receiving channels.

II. FM Receiver in SDR Technology

Our receiver of the passive radar has to ensure multichannel acquisition of signals from radar's antennas and conversion of these signals to a digital form. The term multichannel refers to the synchronous sampling of at least 6 channels. This requirement was imposed by the digital beamforming algorithm in the next stage of the signal processing.

There are a lot of the receivers that conform to this specification. In our opinion the simplest solution that offers a high degree of reconfigurability is to take advantage of the SDR technology. The SDR approach is characterized by the flexibility and the reconfiguration capability that cannot be achieved in standard (hardware) radio systems. This is especially valuable feature for an experimental system such as the passive radar under research.

One of the most popular SDR projects is the GNU Radio [16]. This project contains a large library of functions written in Python/C++ language (for the PC computer of the SDR system) and the library of several basic components for programming the FPGA device implementing basic functions of a digital radio receiver. Moreover the GNU Radio project contains a detailed specification (schematics, PCBs and BOMs) of the required hardware of the system. The whole GNU Radio project is available under GNU license that does not restrict free use of GNU Radio components in research and commercial projects.

A. Universal Software Radio Peripheral – the Hardware Platform of the Receiver

Perhaps the most straightforward choice of a hardware platform for the GNU Radio-based passive radar is a device especially designed for the GNU Radio project, namely Universal Software Radio Peripheral – USRP. USRP's (ver. 2) mainboard contains:

- two AD9862 codecs – each of them contains two 64MS/s, 12-bit A/D converters and two 128 MS/s, 14-bit D/A converters
- an Altera Cyclone FPGA device
- a Cypress USB 2.0 controller
- four expansion slots for daughtercards – two receiving (RX) and two transmitting (TX) channels

There are many different models of daughtercards available (receivers, transmitters, and transceivers). Receivers offer the radio frequency range from 1-250MHz (Basic RX) up to 300MHz-4GHz (BURX). Because the passive radar being designed is FM based (RF range 88-108MHz), for our project we chose the Basic RX as the daughtercard modules.

Theoretically, USRP device allows for receiving any radio signal (using appropriate input circuits on daughtercards) with 32 MHz bandwidth. However USB 2.0 interface of the USRP device limits transfer speed of received signal to the outside of the device. The documentation of the GNU Radio project states that the highest data transfer rate between the USRP device and PC host computer is 32 MB/s. The USRP uses

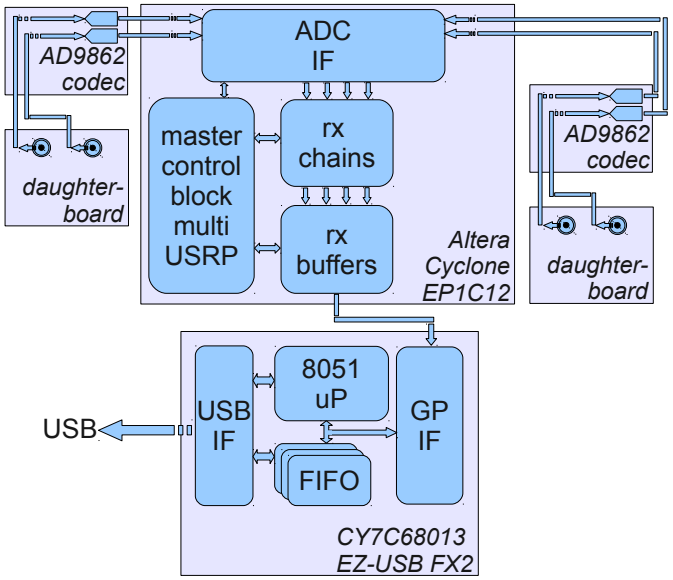

Fig. 2. Functional diagram of USRP device.

16-bit word length for each real-valued sample. Standard configuration of the FPGA device contains digital down conversion (DDC) block that for each real-valued input sample produces two 16-bit values – I (in phase) and Q (quadrature) representing complex IQ signal. As a result a single sample of complex signal (after down conversion) is a 32-bit value. Taking into account this fact the actual restriction of signal's bandwidth is as follows:

- 8MHz for one channel/USRP configuration
- 4MHz for two channels/USRP configuration
- 2MHz for four channels/USRP configuration

III. The Idea of Constructing a Multichannel FM Receiver Using USRP Devices

In the standard (factory) configuration of the USRP's FPGA device there are two receiving (RX) and two transmitting (TX) channels. These default settings can be found in the `config.vh` (the Verilog header file) in the GNU Radio project directory.

```
`include "../include/common_conf_2rxhb_2tx.vh"
//`include "../include/common_conf_4rx_0tx.vh"
```

In order to allow acquisition for all four A/D channels of the USRP device, one should uncomment the first line and comment the second one:

```
//`include "../i...e/common_conf_2rxhb_2tx.vh"
`include "../include/common_conf_4rx_0tx.vh"
```

As a result after building the GNU Radio project and programming of the USRP's FPGA circuit, we get the USRP device with four RX channels (there are no TX channels in this configuration). This configuration is shown in Fig. 3.

In each channel transmitting digital data via the USB interface to the PC host there are 16-bit in-phase (I) and quadrature (Q) components. Therefore in the Verilog code of the GNU Radio project there are eight channels with the USRP's output data: `ch0rx,...,ch7rx`.

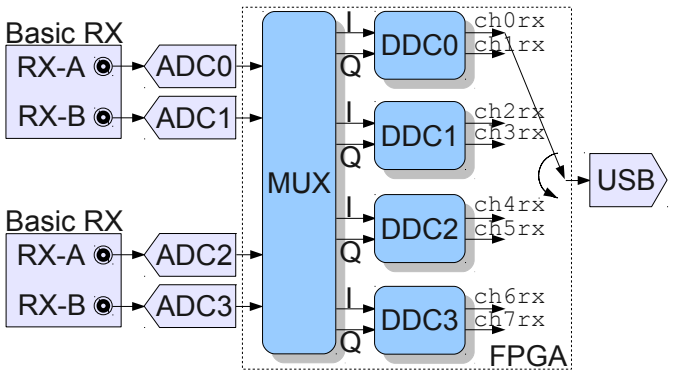

Fig. 3. Configuration of a single USRP device with four receiving channels.

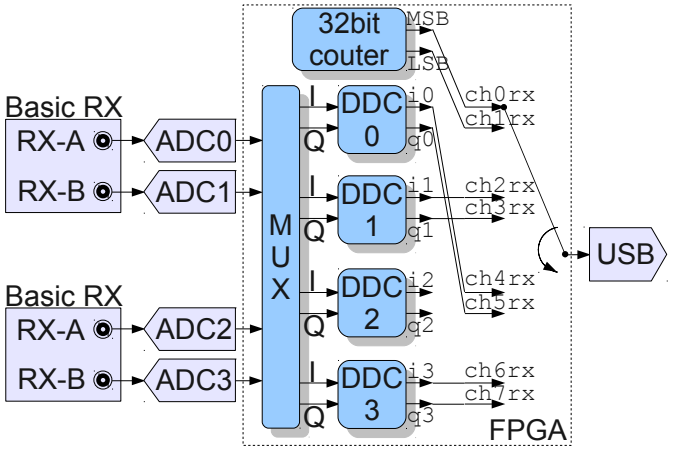

Fig. 4. Configuration of a single USRP in muliUSRP mode.

As mentioned previously the beamforming algorithm that will be used in the signal processing path requires simultaneous acquisition of signals in at least 6 channels. Hence more than one USRP device have to be used in the receiver of the radar.

In order to allow several USRP devices to work synchronously (so called multiUSRP mode), the GNU Radio project contains directory:

```
usrp/fpga/toplevel/usrp_multi
```

with the appropriate Verilog code for the FPGAs of the USRP devices. In the `usrp_multi.v` file `ch0rx` and `ch1rx` channels of each USRP device are not connected to the DDC components (down-converting and decimation of sampled input signals). Instead of this, they are connected to the output of the 32-bit counter. The GNU Radio software of the PC host computer uses this counter to synchronize independent data streams being transmitted via USB interfaces. These data from separate USRP devices are written to the PC memory buffers with different delays. Utilizing the sample counter solves the problem of data synchronization – the PC software creates one coherent data stream with signal samples aligned in time.

Configuration of the USRP's FPGA described above is presented in Fig. 4. Most significant 16 bits of the counter are sent as the `ch0rx` signal, whereas least significant 16 bits are sent as the `ch1rx` signal. The `ch4rx` and `ch5rx`

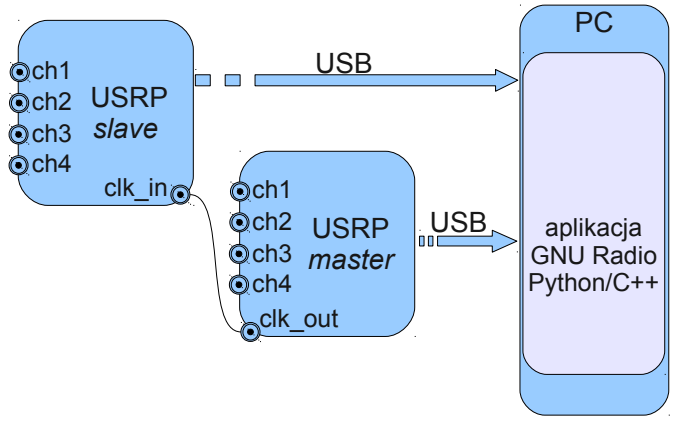

Fig. 5. Six-channels receiver using two USRPs (master-slave mode).

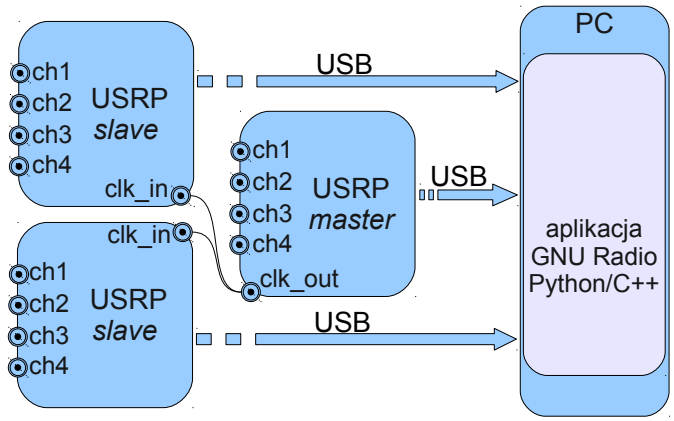

Fig. 6. Nine-channels receiver using three USRPs (master-slave mode).

signals contain data from the DDC0 block while data from DDC2 block are being lost. As a result the PC host receives via its USB interfaces data from three (not four) ADC channels (ADC0, ADC1, ADC3) and the 32-bit counter.

A. USRP Clock and 32-bit Counter Synchronization

In order to construct the receiver with more than four RX channels, two or more USRP devices must work synchronously (the multiUSRP mode). Therefore a stable source of a digital clock signal have to be used for synchronous sampling in all USRP receivers. According to the documentation [17], the synchronous acquisition of signals by a number of the USRP devices can be achieved with master/slave mode of operation. In this case one of the USRP devices works as the master providing the clock signal for remaining USRP receivers working as the slaves as presented in Fig. 5 and in Fig. 6.

In order to enable master/slave mode of operation some hardware modifications have to be made (these modifications are USRP's revision-dependent). In the case of REV. 4.5 the procedure is as follows (see Fig. 7) [18]:

- on the master USRP board
  - solder an SMA connector (the J2002 point) – it will be the master clock output `clk_out`
- on the slave USRP board
  - solder an SMA connector (the J2001 point) – it will be the slave clock input `clk_in`
  - remove R2029 (0 [Ohm]) and solder it in the R2030's place. This blocks internal clock of the slave device
  - remove C925 and solder it in the C926's place
  - remove C924

The slave devices can be connected in daisy chain manner (J2002 to J2001).

After applying modifications described above we have performed experiments (two USRPs in master/slave mode) in order to evaluate the quality of the synchronization of USRP devices. Our measurements showed that in the master/slave mode of operation the clock signal arriving at `clk_in` pin is delayed approximately by 6ns. It means that for 44MHz clock the slave devices' clock is delayed for about 1/4 of the period.

Deeper investigation of this issue revealed that this problem is inherent in the USRP's design. The clock signal of the master device is connected through the clock distribution circuit AD9513-CLK (U702) to the J2002 connector, then through the SMA cable to the J2001 of the slave device and finally to its AD9513-CLK circuit (Fig. 7). Hence, the propagation times of these two AD9513 ICs contribute to the total clock delay, which gives the value of 6ns. It means that, in fact, in order to get synchronous sampling in the multiUSRP mode, the external, stable source of the clock signal for all the USRP devices has to be provided.

In order to solve this problem we propose modification of the multiUSRP master/slave mode of operation by applying an external, common, stable clock source as shown in Fig. 8.

As mentioned previously, configuration of the FPGAs of the USRP devices working in the multiUSRP mode (the code in the `usrp/fpga/toplevel/usrp_multi` directory) introduce 32-bit sample counter. In the PC software, in the `multi_usrp` object (the `usrp_multi.py` file) there is the special method used for resetting this counter to zero for each USRP device in prior to start the acquisition. This procedure would be useless if it was based only on commands sent via USB because the operating system can't guarantee that reset commands will arrive at the USRP devices at the same time. This procedure is complemented by additional hardware line between the USRP devices [18]. Documentation states that the IO15 pins of the BasicRX cards of master and slaves devices have to be wired in order to properly synchronize the counter reset moment. The command of resetting of the counter to zero is sent only to the master USRP – it also sets IO15 output port to logical "0". Then the logical "1" is set and the rising edge of this signal resets the counters in all USRP devices.

B. Synchronously Working USRP Devices – the Configuration of the Host PC Application

In order to receive data being sent via USB by USRP devices to the PC host, the code for the multiUSRP configuration from `gr-usrp/src` of the GNU Radio project should be used. In Fig. 9 the dataflow in the `usrp_multi` object of the PC host application is presented.

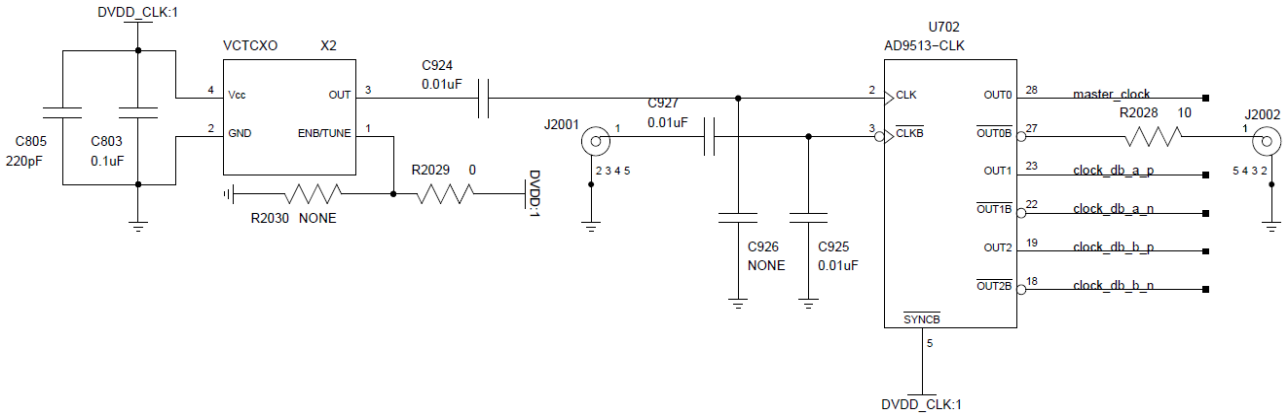

Fig. 7. USRP internal clock distribution circuit [18].

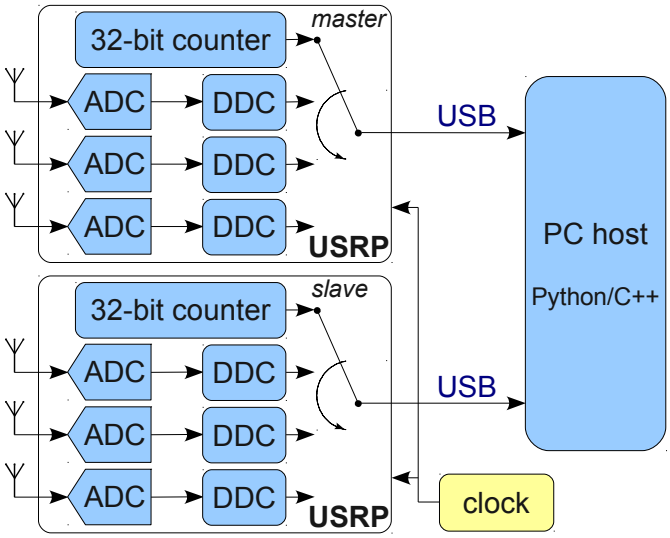

Fig. 8. Master-slave configuration with external clock.

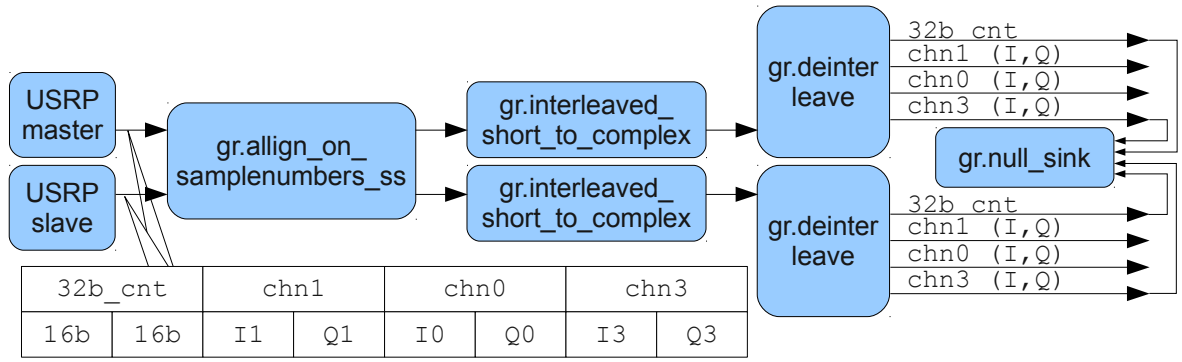

Fig. 9. Dataflow in usrp_multi object for multi USRP configuration (GNU Radio PC software).

USRP master and USRP slave are the objects collecting data being sent by the USRP devices via USB ports. As shown in Fig. 9, the data streams consist of four complex channels (eight 16-bit numbers) – the first one is the 32-bit counter. Remaining three channels contain data with sampled input signals. The data streams from USRP master and USRP slave are independent so in order to achieve mutual synchronization they have to be aligned.

This operation is performed by the object called `gr.allign_on_samplenumbers_ss` that, based on the 32-bit counter, produces output data frames with identical values of this counter for all data streams from USRP devices. Each frame is then converted by the object named `gr.interleaved_short_to_complex` so that at its outputs we get the frames consisting of four 32-bit numbers (I and Q components). These frames are then sent to the

`gr.deinterleave` objects deinterleaving data from the streams so that at these objects' output we get four signals per USRP device (32-bit counter – `32b_cnt`, complex, input data from channels 1 – `chn1`, complex, input data from channels 0 – `chn0` and complex, input data from channels 3 – `chn3`).

As it can be seen in Fig. 9, the `32b_cnt` signal is being lost in the `gr.null_sink` object. Moreover in the original GNU Radio code the `chn3` signal with useful input data from ADC3 A/D converter is also sent to the `gr.null_sink` object. Therefore only `chn0` and `chn1` signals are available to the user. It means that after described changes in the FPGA's configuration that enable multiUSRP mode and using original PC software of the GNU Radio project, the receiver consisting of two USRP devices allows users to perform acquisition only in four channels – the same number of channels is offered by a single USRP device.

In order to increase the number of input channels in the receiver consisting of two USRP devices working in the multiUSRP mode the code of the `usrp_multi` object was modified, so that the `chn3` signal (see Fig. 9) instead of being sent to the `gr.null_sink` object is driven out along with the `chn0` and `chn1` signals.

In the case of the receiver utilizing three USRP devices additional source code modifications have to be made to get it work. The `multi_usrp` object's code (written in Python) takes into account only connection of two USRP devices. Therefore we had to modify this code. We created a new instance of the USRP slave object working as additional signal sample source for the `gr.allign_on_sample_numbers_ss` object (this object, implemented in C++, correctly aligns two and more input streams). In addition we modified the method of synchronization 32-bit counters for each USRP device.

As a result, all described modifications allow us to construct the receiver utilizing two and more USRP devices with three input channels per USRP and synchronous sampling in all input channels.

IV. Conclusion

In this paper we proposed a Software Defined Radio approach to building the multichannel, synchronous receiver of the experimental FM based passive radar. Our system uses the low-cost Universal Software Radio Peripheral devices as the hardware platform of the receiver. The software part of the system is built on the GNU Radio project.

It should be noted that the high cost of multichannel receivers seems to be one of the bottlenecks of current research on passive radars and the SDR technology is one of the ways of alleviating this problem. Moreover the SDR approach allows one to use the same hardware in order to perform research for various RF bands and types of transmitters which is not possible using the traditional receivers.

Proposed modification of USRP's and GNU Radio software allowed us to build the low-cost radio receiver utilizing Software Defined Radio technology. The receiver features synchronous sampling and processing of radio signals with bandwidth up to 2MHz. Thanks to the described hardware and software modifications, we were able to increase the number of input channels per one USRP device working in the multiUSRP mode up to three. Hence, using N USRP devices in the multiUSRP mode we can construct a synchronous, 3N-channels receiver. This is a valuable result since it provides a method of building the low cost, multichannel receiver platform for further research of our experimental, FM based, passive radar being designed.

Our further research will focus on building the complete signal processing path for the passive radar utilizing the GNU Radio in order to achieve the passive object detection and tracking. The second line of research will center on how to obtain four synchronous (instead of three) input channels per one USRP device in the multiUSRP mode, which will reduce the number of the USRPs in the receiver of the passive radar.

References

[1] C. J. Baker and H. D. Griffiths, "Bistatic and multistatic radar sensors for homeland security," July 2005. [Online]. Available: http://www.prometheus-inc.com/asi/sensors2005/papers/baker.pdf

[2] N. J. Willis, *Bistatic Radar*. SciTech Publishing, 2005.

[3] T. Johnsen and K. E. Olsen, "Bi-and multistatic radar," *Advanced Radar Signal and Data Processing*, no. 4, pp. 1–34, 2006.

[4] A. Haghighat and J. Fitton, "Enabling technologies for software defined radio," in *Proceedings of SDR'02 Technical Conference and Product Exposition*, 2002.

[5] J. Mitola, "The software radio architecture," *IEEE Communications Magazine*, vol. 33, no. 5, pp. 26–38, 1995.

[6] F. Berizzi, M. Martorella, D. Petri, M. Conti, and A. Capria, "USRP technology for multiband passive radar," in *IEEE Proc. of Radar Conference*, 2010, pp. 225–229.

[7] D. Petri, F. Berizzi, M. Martorella, E. Dalle, and A. Capria, "A software defined UMTS passive radar demonstrator," in *Proc. of 11-th International Radar Symposium IRS*, 2010, pp. 1–4.

[8] P. E. Howland, "Target tracking using television-based bistatic radar," in *IET Radar Sonar and Navigation*, no. 146(3), 1999, pp. 166–174.

[9] S. Rzewuski and K. Kulpa, "System concept of wifi based passive radar," *JET Intl. Journal of Electronics and Telecommunications*, vol. 57, no. 40, pp. 447–450, 2011.

[10] C. Bongioanni, F. Colone, and P. Lombardo, "Performance analysis of a multi-frequency FM based passive bistatic radar," in *Proc of Radar Conference, RADAR'08*. IEEE, 2008, pp. 1–6.

[11] J. Friedman, A. Davitian, D. Torres, D. Cabric, and M. Srivastava, "Angle-of-arrival-assisted relative interferometric localization using software defined radios," in *Proc of Military Communications Conference MILCOM 2009*. IEEE, 2009, pp. 18–21.

[12] S. Heunis, Y. Paichard, and M. Inggs, "Passive radar using a software-defined radio platform and opensource software tools," in *Proc of IEEE Radar Conference (RADAR 2011)*, 2011, pp. 879–884.

[13] P. E. Howland, D. Maksimiuk, and G. Reitsma, "Fm radio based bistatic radar," in *IET Radar, Sonar and Navigation*, 2005, pp. 107–115.

[14] D. W. O'Hagan and C. J. Baker, "Passive bistatic radar (PBR) using FM radio illuminators of opportunity," in *New Trends for Environmental Monitoring Using Passive Systems*, 2008, pp. 1–6.

[15] A. S. Tasdelen and H. Koymen, "Range resolution improvement in passive coherent location radar systems using multiple FM radio channels," in *Waveform Diversity and Design in Communications, Radar and Sonar*. The Institution of Engineering and Technology Forum, 2006, pp. 23–31.

[16] "The GNU Radio," Project website, 2011. [Online]. Available: http://gnuradio.org

[17] "MultiUsrp or mimo use of the USRP," The GNU Radio Wiki, 2011. [Online]. Available: http://gnuradio.org/redmine/projects/gnuradio/wiki/MultiUsrp

[18] "USRP clocking notes," The GNU Radio Wiki, 2011. [Online]. Available: http://gnuradio.org/redmine/projects/gnuradio/wiki/USRPClockingNotes