Development of Software Defined Radio (SDR) Receiver

M.H.M.N.D. Herath[1*], M.K. Jayananda[2,]

[1]Department of Electrical and Computer Engineering, The Open University of Sri Lanka, Nawala, Nugegoda, Sri Lanka

[2]Department of Physics, University of Colombo, Colombo 03, Sri Lanka

*Corresponding Author: email: mhher@ou.ac.lk, Tele: +94112881272

Abstract – *Purpose of this study is making digital signal processing blocks and connections in radio systems by software and making digital analog boundary for transmitting and receiving signals needed to connect with digital signal processing blocks.*

Aim of this research is analysis of how to create digital signal modulation and demodulation using free and open software tool kit named GNU Radio which is a type of software defined radio tool kit and analysis of how to make customized.

Telecommunication field is now continuously improving and different types of radio networks and technologies are evolving in the world. It is very essential to making devices with interoperable facilities between various networks and technologies. It is very difficult to implement these facilities by traditional making method of telecommunication devices which is by making hardware circuits. Software Defined radio important with this problem. Most functioning features, such as bandwidth, modulation, coding rate, can be modified during runtime according to software configuration.

Keywords*: Software Defined Radio (SDR), Digital Signal Processing, FPGA, GNURadio*

1 INTRODUCTION

Today's voice, data and video transmission needs lots of flexible and reconfigurable radio systems for selecting various communication systems such as 3G, 4G and various modes such as video, audio. Multiple communication systems (3G,4G) or multiple functions (AM/FM) for a single system need computationally intensive signal processing algorithms and high data rates. Separate hardware resources for each of the function will increase the silicon area and complicate the design validation and compatibility (Gupta, 2010). Mobile communication services can be accessed by today's most of the mobile devices. But integrating various protocols and different IC chips into a small device is an important challenge in recent years (Chen, 2010).

With emerging digital signal processing (DSP) technologies, high speed micro controllers, microprocessors, computers and Field Programmable Gate Arrays (FPGA), SDR concept is becoming a new path to the telecommunication field. Analog transmitters and receivers can be converted to digital transmitters and receivers using SDR because most of the hardware components can be replaced with reconfigurable software.

Mitola's view of the SDR is radio is purely digital, except ADCs and DACs, (Bagheri, R. M., 2006). Highest degree of reconfigurability can be obtained by this concept.

Aim of the software defined radio is doing the entire signal processing functions digitally. All digital signal processing can be implemented by software. RF signals are analog and should be converted to digital levels for digital signal processing. For converting analog to digital and vice versa need ADCs and DACs. For entire digital signal processing, ADC and DAC should be at the antenna and everything else would be done by software (Tuutlebee, 2002). As technology progress, SDR is moving to Software Radio where the digitization can expanded to the (or close to the) antenna.

Ideal SDR block diagram is depicted in figure 1. SDR receiver gets signals from the antenna, amplifies using low noise amplifier, convert received analog signal to digital using ADC and send to the digital signal processor. Digital signal processor converts the signal to suitable format and send through DAC to get the required message.

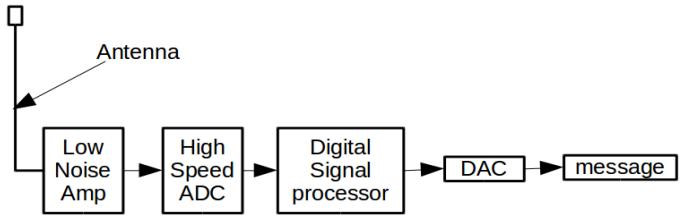

Figure 1: Ideal Software Defined Radio

ADC and DAC cannot be connected directly to DSP computers. Down conversion decimators after ADC and Up conversion interpolators before DAC must be implemented digitally between ADC/DAC and DSP computers. There must be a communication link between hardware and DSP computers. FPGA connected to ADC, DAC and serial communication link can be used for this purpose.

Aim of this research is developing open source analog digital hardware and driver for open source SDR software tool kit named GNURadio tool kit. ADC, FPGA and communication IC can be used for developing SDR analog digital boundary.

2 LITERATURE SURVEY

2.1 GNURadio tool kit

GNURadio is a world wide open source project which is published on the Internet through www.GNURadio.org web address. It is a free & open-source software development toolkit that provides signal processing blocks to implement software radios. GNURadio basically includes flow graphs and blocks. Sampled signals values pass through flow graphs. Flow graphs are graphs. Nodes of such a graph are called blocks, and the signal values pass along the edges of blocks (BLOSSOM).Signal blocks, edges and flow of signals are shown in figure 2.

GNURadio can be used to create flow graphs and digital signal processing blocks. Various signal processing block collection is already developed by GNURadio developer group. Graphical presentation GRC is easy to use and lots of simulations and real time experiments can be done. When you want to create your own signal processing modules, OOT gr_modtool can be used. When you become highly expert, your own modules

without gr_modtool can be created because these all are python and C++ programming.

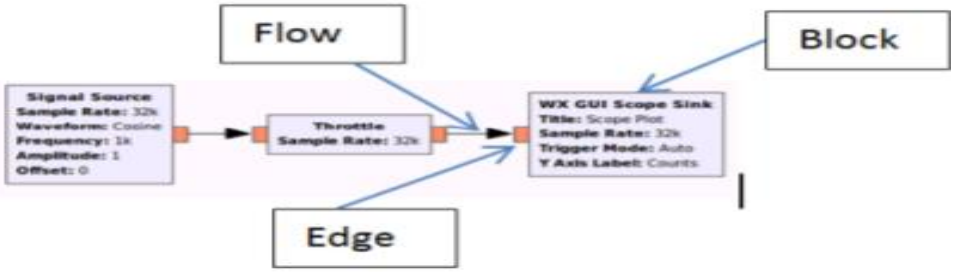

Figure 2: blocks, flow and edges of a GNURadio flow graph

GNURadio has following features

1. Graphical user interface called GNURadio Companion (GRC)
2. A mixed of C++ and python programming languages
3. Real time running
4. Changeable and able to create new signal processing blocks
5. Message passing facility

Message passing blocks are used for implementing feedback and non-synchronous signals. Message passing uses a common data type called Polymorphic Types (PMTs). Other data types such as int, float, char should be converted to PMT type when using message passing blocks.

2.2 Architecture of GNURadio

Figure 3 shows the architecture of GNU Radio. Flow graphs are written by python high level language. Signal processing blocks are written by lower level C++ blocks. Middle interface between python and C++ is Simplified Wrapper Interface Generator (SWIG).

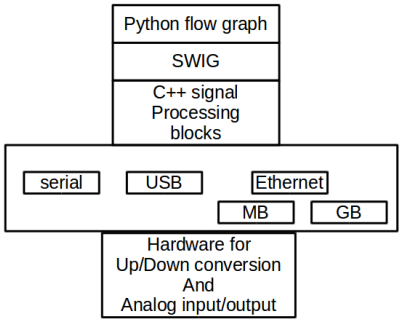

Figure 3: Architecture of GNURadio

GNURadio tool kit is an open source software but analog digital boundary hardware products available in the market are very expensive. Most hardware products available in the market have developed by using FPGAs. ADC is needed for converting analog signal to digital.

2.3 Analog digital boundary hardware for GNURadio

A separate hardware named Universal Software Radio Peripheral (USRP) is manufactured by Ettus Research team that can be used with specified DSP software including GNU Radio installed in personal computers. USRP Hardware Driver (UHD) in GNURadio tool kit is available for connecting hardware, developed by Ettus Research. The USRP devices can be connected to PC via serial, USB or Ethernet. Each USRP device has a separate device address that has to be entered in the UHD.

2.4 Field Programmable Gate Arrays (FPGA)

FPGA is programmable logic device that can be used for implementing large logic circuit. It has logic blocks for implementing required functions. It has 3 main resource types. Logic blocks, input output ports and wires and switches (Brown, 2007). Largest FPGA venders are Altera and Xilinx. More flexible and reconfigurable application specific DSP methods can be implemented than using microcontrollers (Bdti, 2007).

Altera is developing flexible soft processor called NIOS for implementing in Altera FPGA. User can modify NIOS processor with user specified features. i.e. bus width, number of general I/O port, connection modules called Intellectual Property (IP) cores to specific ports like Ethernet, USB, serial. Altera develop and sell 4 main categories of FPGA chips, Cyclone, MAX, Arria and Stratix. Application developer has to check specifications of FPGAs to check they are supported for their applications before buying. For example, 10GBit Ethernet and FIR II IP core is not support for cyclone ii devices (www.altera.com).

Software required to implement digital modules in Altera FPGA are Quartus II, Mage wizad plug in manager, QSYS, NIOS II Software Builder Tools (SBT) for Eclipse, programmer, QSYS is a fully automated GUI system for configuring processor features and generating hardware design in the FPGA. SBT for Eclipse is used as a GNU C/C++ programming tool for NIOS II processor (Altera, 2014).

NIOS II processor is a soft processor IP core designed for Altera FPGA as opposed to a fixed hardware processor. Therefore it has more flexible than other general purpose processor. It is a Reduced Instruction Set Computer (RISC) 32 bit processor, optional Memory Management Unit (MMU) and optional Memory Protection Unit (MPU) (Altera, 2014).

2.5 Pipelined Analog to Digital Converter

Figure 4 shows a block diagram of k bit pipelined ADC. Each stage contains a sample and hold (S/H) circuit, A/D sub converter and Digital to Analog (D/A) converter (Stephen, 1987).

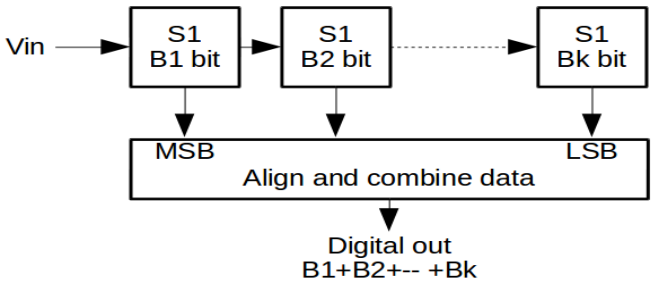

Figure 4: Pipelined Analog to Digital converter

2.6 Proposed analog digital boundary hardware and driver

Proposed analog digital boundary and driver is developing open source analog digital hardware and driver for the hardware. ADC, FPGA and communication IC can be used for developing SDR analog digital boundary. Serial communication can be implemented by using python serial communication and it can be connected to GNURadio by using message passing and message converting methods in GNURadio tool kit.

3 METHODOLOGY

The proposed system is shown in figure 5. FPGA board is depicted in figure 5.a. and CYCLONE II EP2C8Q208C8N altera FPGA is depicted in (a2) in figure 5. Serial USB conversion cable is depicted in figure 5.b. All the software needed to run SDR and driver is depicted in figure 5.c.

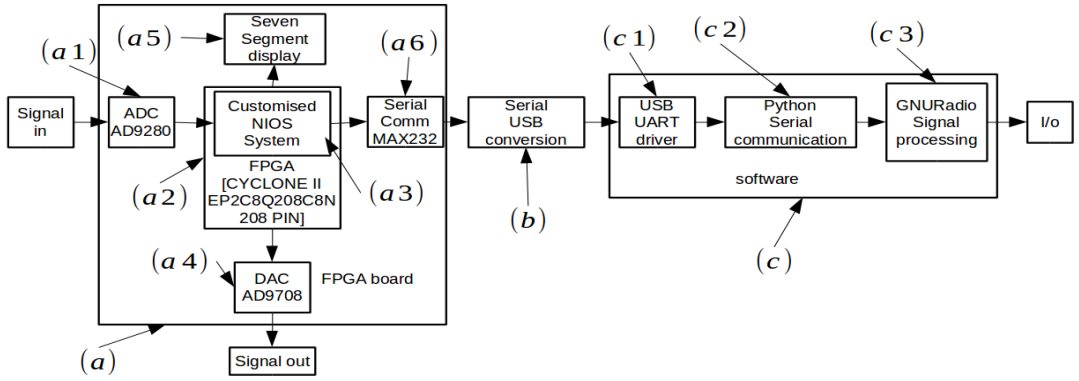

Figure 5: SDR hardware, GNURadio software and communication

Signals received to the ADC (a1) are converted to digital format and send to the serial communication port (a6) through NIOS system (a3). Digitized signal values can be displayed via seven segment display (a5). DAC (a4) can be used for transmitting purpose. Data send to computer is obtained by USB UART driver (libusb) (c1). Serial data obtained by USB UART can be sent to GNURadio by using python serial communication. (c3) indicates the GNURadio blocks and connection paths.

3.1 Equipment and software support for developing GNURadio receiver

1. Computer – Intel Core i3 CPU 2.3 GHz, Memory – 2GB, operating system- Ubuntu 14.04.

2. GNURadio tool kit and supporting software - Latest installed GNU Radio version is 3.7.6. Supported operating system is 64bit Ubuntu 14.04. Installing methods-Easy method "apt-get install GNURadio ".

3. FPGA board ordered for making open source hardware part to communicate with GNU Radio toolkit named as YG_V2.1. I assumed it is good for DSP purposes. This board has FPGA-ALTERA – CYCLONE II -EP2C8Q208C8N-208 PIN, Ethernet- ENC28J60, SERIAL CONTROLLER - MAX232, DAC- AD9708, pipelined multistage ADC- AD9280, USB - CH376, 8 seven segment displays, 9 keys for input. 5 LEDs and LCD display.

4. Windows base Altera Quartus II 32 bit Version 12.0 Web edition.

3.2 NIOS system implementation

Figure 6 shows custom based NIOS system for communication between computer and ADC through FPGA. Modules of this system are 32 bit NIOS II/e processor,32 bit On chip memory, 8 bit input port for ADC, 8 bit output port for display seven segment display, 8 bit UART for communication with computer through usb serial port. All components need clock signals. These modules are connected via Avalon memory mapped interface. Components in the "Altera QSys" can easily connect through Avalon memory mapped interface. It is an address based read-write interface with master-slave connections. Interface master side sends data to slave and vice versa.

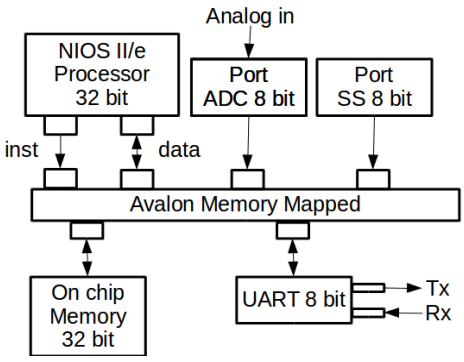

Figure 6: Customized NIOS II/e system

Experimental NIOS system designed for receiving from ADC to the computer through UART in QSYS is shown in figure 7. This system has 7 components- (1) Clock, (2) clock bridge, (3) memory, (4) cpu, (5) pin_adc, (6) pout_ss, (7) uart. These components are interconnected as necessary. Avalon memory mapped master slave interfaces are used to connect above components. Master is used to send data and slave is for receive data. Conduit endpoint interface is used to connect with external devices.

Figure 7: Receiving from ADC to the computer through UART in QSYS

3.3 Making communication through NIOS II software build tools for eclipse

NIOS II Software Build tools for Eclipse (SBT)

SBT is a GUI that runs NIOS II SBT utilities. User can create, edit, build, run, debug, and compile programs. User should have good hardware related knowledge of c programming. (Altera, 2014)

Program flow chart for the communication

Figure 8 shows the flow chart for sending data to the computer through uart port. First put the value of PORTADC to the variable "i" and it check whether the uart has received complete previous data to the uart. If previous data has sent to the uart next value is stored in the uart[1] i.e. for saving transmission data.

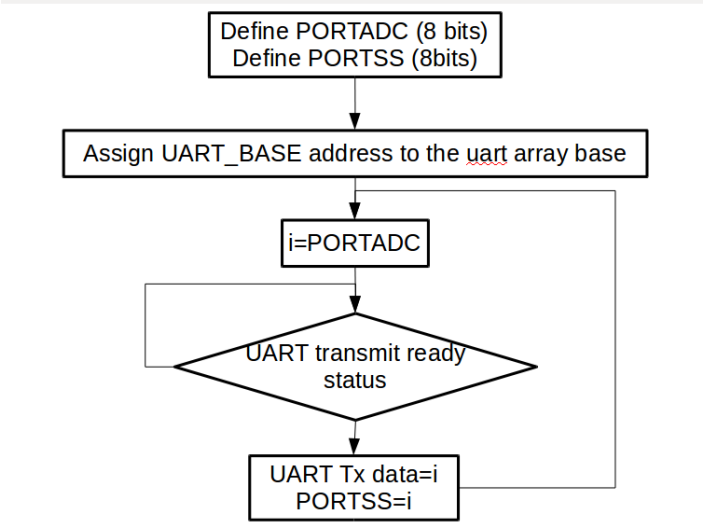

Figure 8: Sending data to the computer though UART port

3.4 Connecting serial port and GNURadio

To make connection between serial port and GNURadio, python serial access module (programmed by python language) called "serial. Serial" and message passing method of the GNU Radio tool kit are needed. Python serial module makes connection and read/writes operations between serial port and computer.

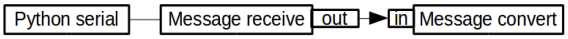

Figure 9: Python serial connection, message receiver and message convert blocks

Figure 9 shows python serial connection and 2 modules called Message_receiver and Message_convert. Python aided by special module belongs to the GNURadio is required to develop 2 modules called "PMT". These modules are imported to the python application program.

Message passing-Message passing method provides receiving and sending messages non synchronously. It uses common variable type called Polymorphic Types (PMTs). Originally only synchronous bit streams are used for sending data between signal processing blocks. There are no ways to communicate from upstream to downstream (feedback). Message passing method is used to connect external applications to GNU Radio.

Python serial connection-This is used for accessing serial port of the FPGA board trough usb port of the computer.

Message receiver block-Input side of the message receiver block is serial data and output side of the block is message port. Message passing blocks should write under the gr.basic_block. Streaming ports of the block must be null when making the message passing port. Therefore, both sides do not have streaming ports. Therefore, in_sig=[] and out_sig=[]. Output port is a message port.

Message converter block- Message converter block convert PMT data to the generic type. These data are not passing synchronously.

4 RESULTS AND DISCUSSION

There are 3 ways of communicating between FPGA and computer. I.e. USB, Ethernet and serial. USB has four transfer modes named "control, interrupt, bulk and isochronous". Best mode to transfer audio or video data is isochronous transfer mode because one way communication and no handshake controlling. USB connection of YG_V2.1 board is configured only for bulk transfer. USB controller in the board is CH376. It is not supported for isochronous transfer mode. MAX232 serial controller is used as serial port communication. Maximum baud rate for this IC is 112500. Serial communication is used to test communication for this research.

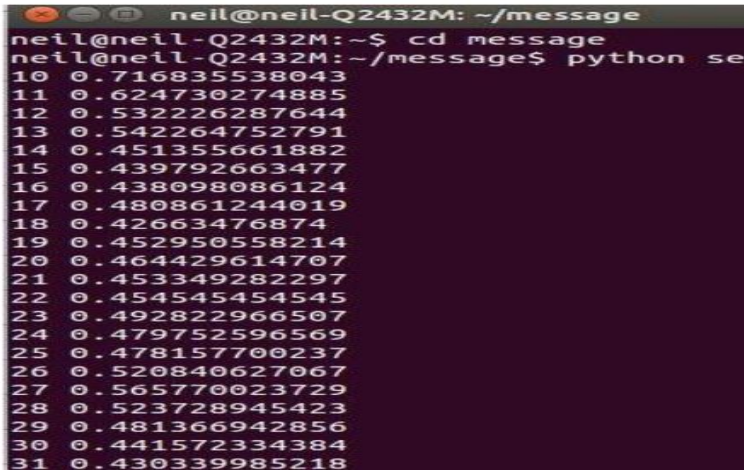

Figure 10: Serial communication output

10Hz sine wave output is shown in figure 10. There are 2 columns 1. Sample number, 2nd column is complex amplitude value of the signal.

GNURadio toolkit can be used to simulate, learn and testing digital signal processing in real time environment easily. With new high speed CPUs, high speed-high capacity FPGAs and continuously developing software modules in GNURadio, Software Defined Radio concept which is brining ADC/DAC closer to the antenna is becoming SDR from dream to reality.

5 CONCLUSION

Mobile communication world is changing from one device for one communication application to one device for all communication applications with Software defined Radio technology.

GNURadio (www.GNURadio.org) is a continuously developing free and open source toolkit for developing software defined radio. It is a large software project with many people in the world is joining through the Internet.

Making open source hardware part need to analyse how to make drivers for GNURadio, operating system based drivers like libusb (for USB connection) or serial port communication or Ethernet stack, selecting FPGA, selecting ADC, DAC, how to make digital up down conversion in FPGA, digital connection modules for USB, serial port, Ethernet and transmitting antenna.

Lots of theories of digital signal processing can be studied from beginning to expert level by simulating and can practice real time environments. Thus real results can be obtained. Users should have prior knowledge of C++, python, object oriented paradigm and Linux environment.

REFERENCES

1. Gupta, P. G, (2010), Radio Implemented in Software, Electronics for you, pp 111-114.
2. Chen, C. Y. (2010), Reconfigurable Software Defined Radio and Its Application. Tamkang Journal of Science and Engineering, Volume 13, No 1, pp 29-38.
3. Bagheri, R. M. (2006, August), Software Defined Radio Receiver: Dream to Reality, IEEE Communication magazine, pp 111-118.
4. Tuutlebee, W. (2002), Software Defined Radio. John Wiley & Sons, Ltd.
5. Blossom, E. (n.d.)., The Free And Open Source Radio Eco System. [Online] Available at: http://GNU Radio.org/redmine/projects/GNU Radio/wiki, [Accessed 26 01 2015].
6. Brown, S. V. (2007), Fundamentals of Digital Logic Design with VHDL ,Second edition. New York: McGraw-Hill Companies.
7. Bdti, B. T. (, An In2007), An independent Analysis: The Evolving Role of FPGAs in DSP Applications.
8. Altera, (2014), NIOS II Processor Reference. [Online] Available at: http://www. altera.com/literature/hb/nios2/n2cpu_nii5v1.pdf. [Accessed 02. 02. 2015].
9. Stephen, H. L. (1987), A Pipelined 5-Msample/s 9-bit Analog-to-Digital Converter. IEEE Journal of Solid-State circuits, Vol. Sc-22, No. 6, pp. 954-961.