DESIGN AND IMPLEMENTATION OF THE SDR DIGITAL CONTROL SYSTEM (SDR-DCS) FOR THE BASEBAND OF A MULTI STANDARD HANDSET TRANSCEIVER

Mr. Khalid Eltahir Mohamed (Department of Computer and Communications Systems, Faculty of Engineering, University Putra Malaysia (UPM), 43400 Serdang, Selangor, Malaysia, abu_ahmed_05@yahoo.com); Prof. Dr. Borhanuddin Mohd. Ali (MIMOS Berhad, Technology Park Malaysia 57000 Kuala Lumpur, Malaysia, borhan@mimos.my); Prof. Dr. S.S. Jamuar, Assoc. Prof. Dr. Sabira Khatun, Dr. Alyani Ismail (University Putra Malaysia (UPM), ssjamuar@eng.upm.edu.my, sabira@eng.upm.edu.my, alyani@eng.upm.edu.my)

ABSTRACT

In this paper, SDR Digital Control System (SDR-DCS) has been developed to perform the multi standard protocol of the handset using GSM and CDMA systems. This system has been designed for SDR baseband digital transceiver for the handset such that it could roam between different wireless systems. The SDR-DCS controls the download of the specific air interface environment, to the unique hardware accordingly. The Synopsys™ software has been used with combination of VHDL and Verilog languages. The simulation tools used are Model Sim and System Studio. Xilinx ISE 9.2i has been used as synthesis tool. The results of the simulated and synthesized top-level design files were downloaded into the Xilinx XSA-3S1000 FPGA board.

1. INTRODUCTION

One of the capabilities of SDR is to enable handset devices for the predefined air interfaces [1]. If a particular interface is not built into the handset, it is not possible for the handset to operate in that particular environment. In such cases, consumers will have to purchase a different unit.

In SDR system, the goal is to dynamically adapt to any of several air interfaces, such as GSM, GPRS, W-CDMA, cdma2000 and so on, through software detection and dynamic reconfiguration of the hardware [2]. It is similar to today's mobile devices but rather than using multiple RF interfaces, a single reconfigurable RF interface is used.

Any new wireless communication system that emerges in the future must have or support a set of standard rules – things like frame size, start/stop bit [3]. While different devices may support different protocols, they all need to, at the very least, support a small set of common control protocols. This is required to establish some sort of link between two devices. Once the link has been established more advance or newer protocols can be sent from one device to another so both can operate in the newer protocol arena.

For a SDR system to be viable there must be [4]:

- A single RF transceiver circuitry that can operate in wide frequency band to support the various frequencies used by future wireless systems.
- A digital signal processor that is able to extract the protocol information from the RF transceiver output and configure the mobile device to operate in a newer environment
- A set of control protocol transmitted by the service providers using a known protocol rule set.

In future system, when a mobile handset enters an area, it will first scan for a control channel. This control channel should be consistent across the globe. The information transmitted will be used to configure the handset so that it is able to operate in the new environment. This is similar to doing a ROM upgrade with current handsets except this will be done wirelessly in the future.

2. SDR DIGITAL CONTROL SYSTEM (SDR DCS) OVERVIEW

Today, the accepted design methodology for high-level design consists of capturing design intent with a hardware description language (HDL) at the register-transfer (RTL) or behavioral level. The design is then verified with an HDL simulator and synthesized to the gates. The design can then be fabricated using ASIC technology or be implemented on an FPGA.

In this research, FPGA has been chosen as the technology of choice for implementing the SDR digital control system (SDR DCS).

The following sections outline the steps that need to be taken to implement the SDR digital control system (SDR DCS) on an FPGA.

2.1 SDR DCS Design Specification

The first task is to describe the design's intended function. The specification for the SDR DCS is shown in Figure 2. The SDR DCS operates as follows:

2.1.1 Operation

- Scan band to see what network is available
- If known network is detected, send ID (details of handset – phone number, origin, capability etc)
  - Wait for action to be performed
  - When required send and/or receive data
  - When task complete go back to wait for action mode
- If unknown band, send control data requesting for configuration data
  - If configuration data is received, handset configures with new data. Handset is now ready to send/receive data
  - If configuration data is not received, request for data again unless a timeout is detected, then display error message to user indicating unsupported network.

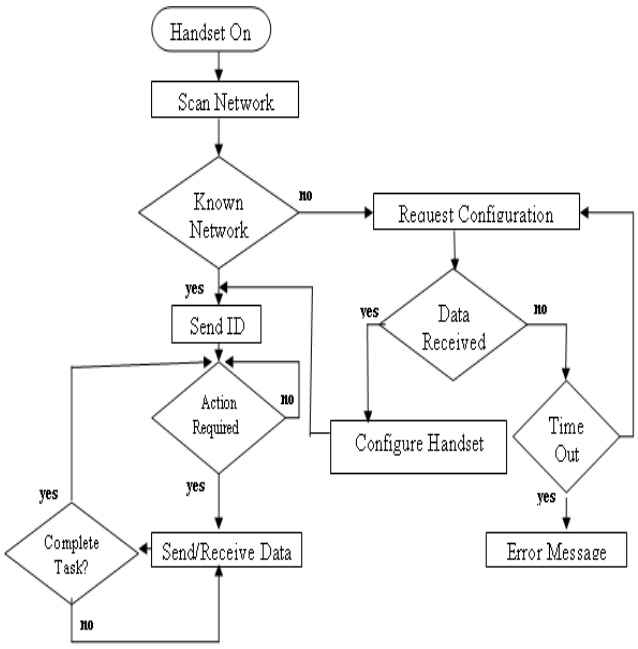

Figure 2: SDR DCS flow chart

2.1.2 Network Detection

- Check network ID
- If network ID matches that stored in system ROM, known network detected, send handset ID to establish connection
- If network ID does not match any stored IDs, request for configuration information
- Network should respond by sending configuration information
- Device should extract configuration data, and store in system flash memory for future use. Once configured, device responds by sending handset ID to establish connection

The diagram of the system is shown in Figure 3 and Table 1 represents the input and output function for the SDR DCS.

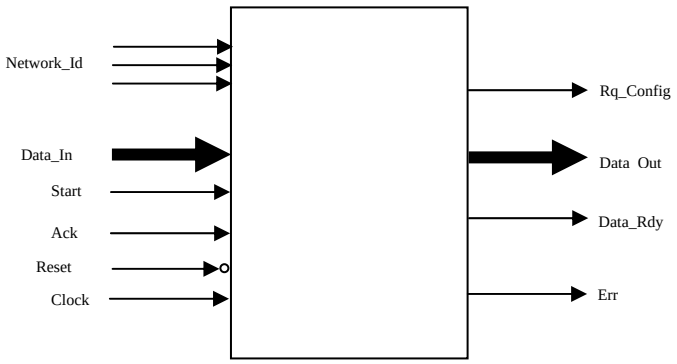

Figure 3: SDR DCS block diagram

Table 1: SDR DCS input/output (IO) pin functions

Input/Output (IO) pins	functions
Clock	system wide clock
Reset	active low reset
Network_Id	3 bit value to identify the type of network. This is used to mimic actual mobile telephone SID (system identification code)
Ack	active when Network ready to send configuration data, else held low
Data_In	user data input, data stored in internal memory
Start	active when Network sends data
Rq_Config	when unknown network present, signal held high to indicate device requires configuration data
Err	held high when device unable to receive configuration data
Data_Out	data output, data stored in internal memory sent out
Data_Rdy	held high when sending user data out, else held 0

The specification above was created to model the behavior of SDR DCS in a typical SDR handset. Only details pertaining to the control mode is defined. This is done to meet timing and budget constraints.

2.2 SDR DCS Design

As this design involves many different states such as scan, configure, send/receive, we use Finite State Machine (FSM) to model the system. The states and their logic values are shown in Table 2.

Table 2: FSM with logic values

State with logic value	function
OFF = 0	Main reset state. When no operation is performed the system goes into this default state
SCAN = 1	This is where the network scanning operation is done
WAIT = 2	Once the correct network ID is received, the system enters this state waiting for data to be transmitted or received
TX_DATA = 3	The system remains in this state when data is sent or received
ERROR = 4	When no network ID is received, the system enters an error state
RCONF = 5	Request for configuration data is done in this state
RCD = 6	This is where the system configures itself with the new system data

This state description is based on the flow chart shown in Figure 2.

2.3 SDR DCS Verification

To verify the correct functionality of the SDR DCS, a full testbench was developed to deliver realistic stimuli for the data processing and the control part of the device. Verification was done using Synopsys System Studio and VCS verification tool suite.

As a starting point and initial reference for this development we used the Synopsys UMTS/Mobile telephony reference-design kit. Based on this reference, a complete and detailed simulation executable specification of our complete system was captured in System Studio. It includes a basestation model, transmission channels and receiver models.

System Studio provides a Direct Kernel Interface (DKI) to VCS so that a simple and fast co-simulation with VHDL and Verilog implementation of the device-under-design, are possible. Thus, the RTL implementation can be tested against the overall SDR DCS specification using the complete set of existing test cases developed during the design process. The sample test frame that is used for SDR DCS is shown in Figure 4.

The VHDL or Verilog which describes the design function, is read into the simulator along with a set of input vectors created by the designer. The SDR DCS test frame represents the input vectors into the device. The simulator generates output vectors that are captured and evaluated against a set of expected values. If the output values match the expected values, then the simulation passes; if the output values differ, then the simulation is said to fail and the design needs to be corrected. Most simulators generate output in two forms: numerically, as 0's and 1's in a file for comparison purposes, and graphically, as waveforms that depict the transition of signals from 0 to 1 and vice-versa.

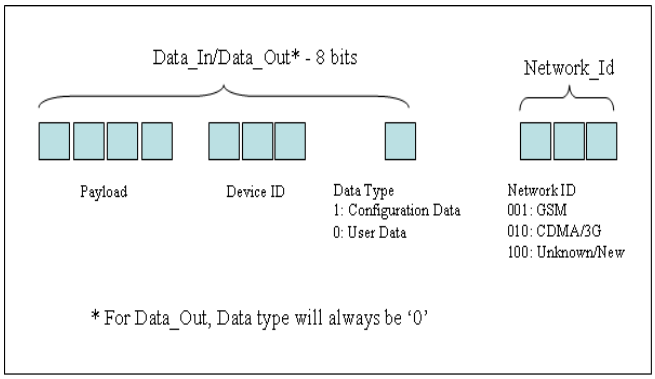

Figure 4: SDR DCS sample test frame

Once SDR DCS verification is completed, HDL source code is ready for implementation.

Figure 5 discusses the conceptual scheme of the handheld SDR terminal including the broadband RF stage [7]. The key subjects of this proposal considered for putting the SDR to practical use as a handheld terminal are the applications program, radio function library, software specification language, digital radio processor, and broadband RF stage.

The application program is written in some software specification language that describes the radio architecture and performance using a radio function library. Application programs are prepared for a specific radio standard, for example, one for GSM, one for cdma-One, one for IMT-2000 and so on. The performance of the handheld SDR is easily reconfigured by rewriting the application program to change the use of the library, to follow up this methodology of application program; SDR DCS is designed to reconfigure the handset with current and future protocols same as GSM and CDMA. The In this case downloading is done over the air from a central or base station to the SDR terminal as shown in Figure 5. SDR DCS reconfigure the handset with the new protocol stored inside the handset memory automatically when the handset switches on and network detected. The network ID is the key here in this design. Each protocol stored inside the handset memory is assigned a certain network ID. When detected network ID matches with the stored one, the handset will automatically

reconfigure with this protocol. Only in case of network ID detected and there is no protocol stored then Hiroshi Tsurumi method will be applied.

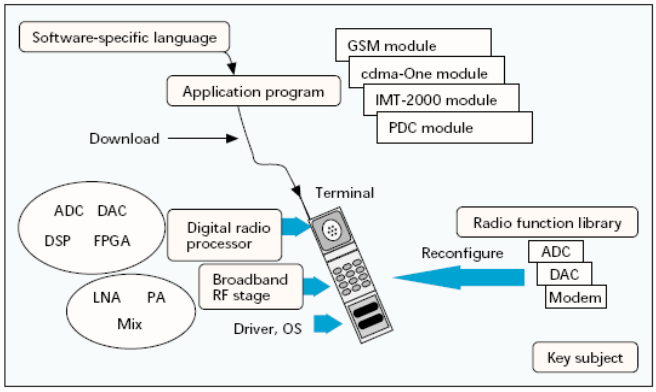

Figure 5: Handheld SDR terminal

3 SDR DCS SIMULATION AND IMPLEMENTATION RESULTS

The waveforms shown below are the results of simulation run done on the SDR DCS. The following series of test were performed in sequence to test the SDR DCS:

1. check on reset
2. test protocol 001 (emulate GSM)
3. test protocol 010 (emulate CDMA)
4. test new protocol (100)

The reset signal has the highest priority, when the reset pin is held low; the output will go to 0.

The start signal acts like a chip enable, only when it is 1 would any operation other than reset can take place.

The debug_cs signal shows the states the design is under. This signal is used for debugging purposes and is not related to handling of protocol.

3.1 RESET CHECK

In this test as shown in Figure 6, the reset pin, rst is held low for three cycles, the output pin, dout goes to 0, indicating the reset works as expected.

3.2 Test protocol 001

This protocol emulates a GSM signal being received as shown in Figure 7. Since this is a known protocol, the controller should only respond if the transmitted handset id matches the controller's id which is set to 111. The handset id is transmitted as part of datain bit 6 to 4. As can be seen in the waveform, if the handset id (din[6:4]) is different from 111, the input data (din[3:0]) is rejected, and if it matches then din[7:0] is sent out through dout_y[7:0]. As long as the netid and handset id is correct, data received would send out via dout_y. We perform this test with several sets of value for din [3:0].

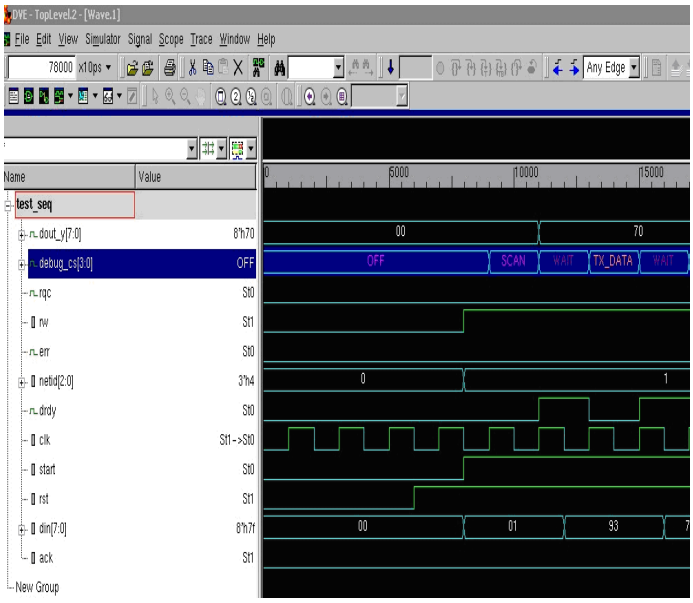

Figure 6: SDR DCS Reset test

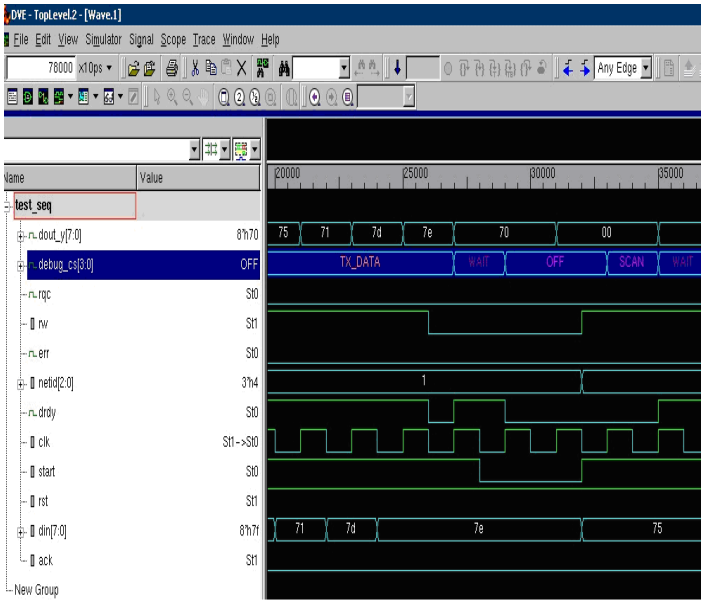

Figure 7: SDR DCS GSM test

Figure 8 shows the waveform of the GSM protocol after the final step of the downloading in the Xilinx XSA-3S1000 FPGA board. The board was tested with real data from the PC RAM. This design looks at the baseband of the handset transceiver only. So the waveform shown represents the GSM protocol. The 7-segment display was used to show the first letter of the protocol as seen C for CDMA here G for GSM.

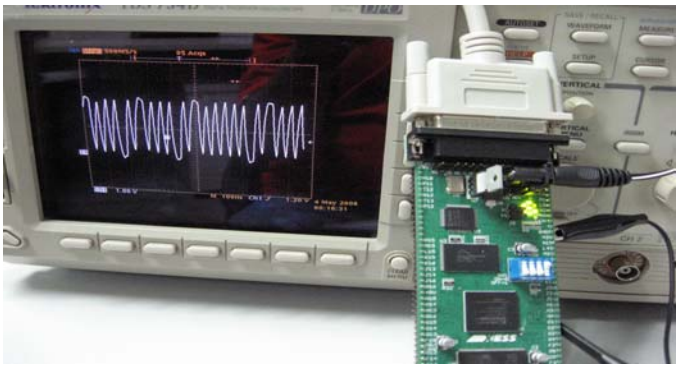

Figure 8: SDR DCS implementation for GSM protocol using FPGA

3.3 Test Protocol 010

This protocol emulates a CDMA signal being received as shown in Figure 9. Just like the GSM signal, this is a known protocol. As can be seen from the simulation results, input data is sent out correctly as long as the handset ID matches. When the handset ID is changed, the output goes low as expected.

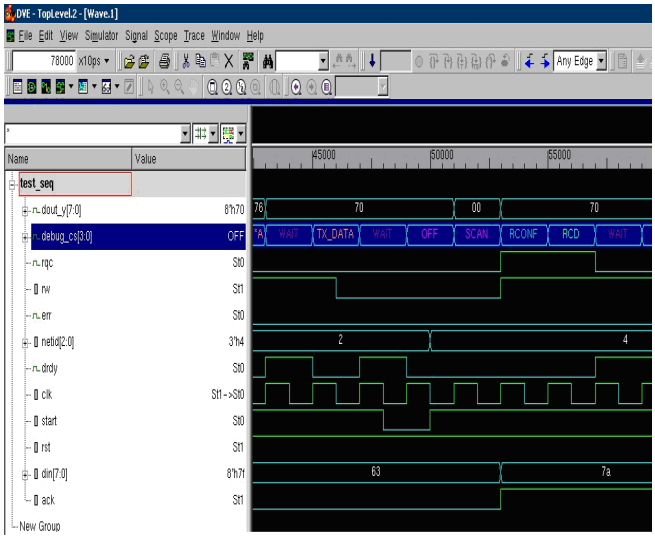

Figure 9: SDR DCS CDMA test

The results of the simulated and synthesized top-level design file for CDMA protocol were downloaded into the Xilinx XSA-3S1000 FPGA board. As shown in Figure 10. Synthesis allows timing factors and other influencing factors of FPGA devices; thereby a thorough checking could be implemented before the design is committed to the FPGA or a similar device. Xilinx XSA-3S1000 FPGA board was used with 7-segment display to show the status of the protocol. Letter C represent CDMA as shown in the FPGA and the waveform represent the CDMA protocol. Letter C represent CDMA as shown in the FPGA and the waveform represent the CDMA protocol.

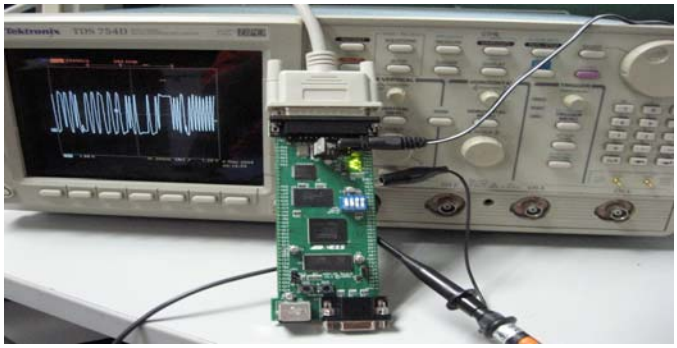

Figure10: SDR DCS implementation for CDMA protocol using FPGA

3.4 Test new protocol 100

When the SDR DCS receives a new protocol it should first enter state RCONF and RCD as shown by the debug_cs signal. This is where the SDR DCS configures itself with the new protocol. At this point no user data is allowed to go out. The simulation results show this operation being performed as intended.

Next we set the start signal to go low, indication no operation to be performed as shown in Figure 11. Then we must test the SDR DCS to respond to the new protocol 100 that was just configured. At this point the SDR DCS controller should already have the configuration data; hence it should directly go into data transfer mode and bypass state RCONF and RCD. As can be seen from the waveform, the controller correctly goes into data transfer mode, and data at the input port din, is sent out via dout_y.

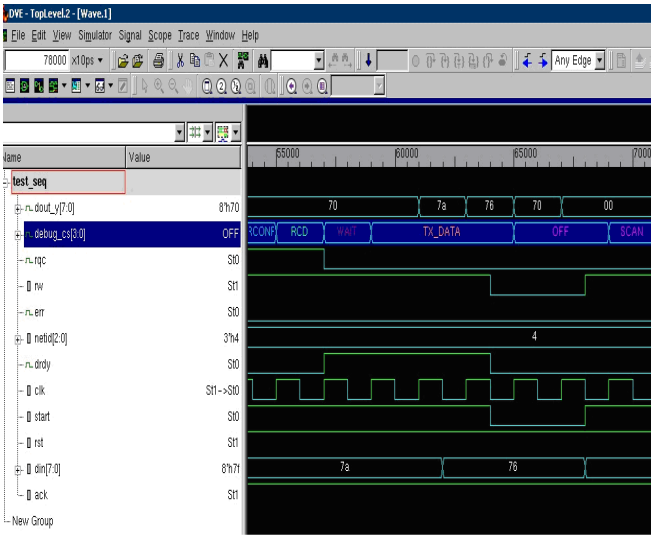

Figure 11: SDR DCS new protocol test

For the new protocol detected, no signal can be displayed on the oscilloscope, the system here request for configuration from the basestation. As stated in section 2.1.1 there is an error message appears at the output of the FPGA. The error message will appear on the 7-segment display which titled with letter E as appear in the FPGA as shown in Figure 12.

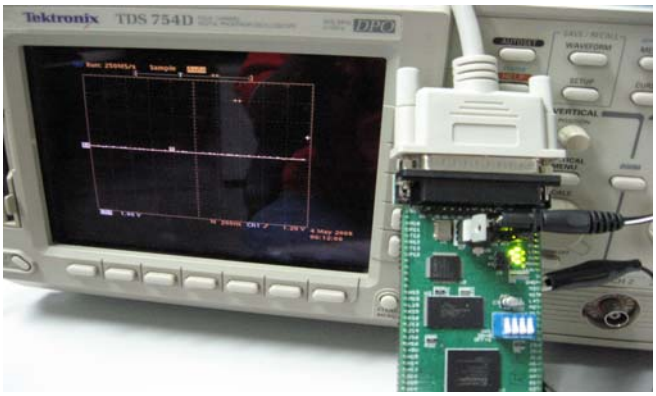

Figure 12: SDR DCS implementation for new protocol using FPGA

4 CONCLUSION

The simulation and implementation results show the proposed SDR DCS unit work as per specification. It is able to handle known protocol built into its memory. When a new protocol is observed, the SDR DCS is able to request for the configuration data and update its internal database with the new protocol. This will enable the SDR DCS controller to operate on the new protocol just like any other in its internal memory. The automatic switching between the different protocols and the reconfiguration for the handset with new protocol, will allow the handset to roam easily between different networks

5 REFERENCES

[1] W. Koenig, B. Haberland, A. Pascht, D. Wiegner, K. Nolte, "New Architectures and Technologies for Software-Defined Radio Base Stations", Technology White Paper © 04 2006 Alcatel.

[2] HY SDR research center, Hanyang University South Korea, "From silicon to software, SDR overview and technology roadmap", SDR Forum 19 April 2003.

[3] Luo Zhigang, Li Wei, Zhang Yan, Guan Wei, "A Multi-standard SDR Base Band Platform", Proceedings of the 2003 International Conference on Computer Networks and Mobile Computing (ICCNMC'03).

[4] SDR forum, Base Station Working Group, "An Adaptive Hardware Platform for SDR", (2001).

[5] Che, H.; Hajian, M.; Ligthart, L.P.; Prasad, R. "Software Radio Is Walking into Implementation Stage." Published in Del Re, Enrico. Software Radio: Technologies and Services. London: Springer – Verlag, 2001; 84.

[6] Zwolinski, M., "Digital System Design with VHDL", Pearson Education Ltd., England, 2003.

[7] Hiroshi Tsurumi and Yasuo Suzuki, IEICE "Broadband RF Stage Architecture for Software-Defined Radio in Handheld Terminal Applications" , IEEE Communications Magazine, February 1999.