Proceedings of the SDR 11 Technical Conference and Product Exposition, Copyright © 2011 Wireless Innovation Forum All Rights Reserved

DEVELOPMENT OF LOW-COST PUBLIC SAFETY P25 WAVEFORM IN AN OSSIE ENVIRONMENT WITH USRP

Zhongren Cao, Jeff Cuenco, Anthony Nwokafor, Per Johansson, William Hodgkiss
(California Institute for Telecommunications and Information Technology – Calit2, University of California San Diego, La Jolla, CA 92093; zcao@soe.ucsd.edu, jcuenco@ucsd.edu, aanwokafor@ucsd.edu, pjohansson@soe.ucsd.edu, whodgkiss@ucsd.edu)

ABSTRACT

Low-cost software communications architecture (SCA) based waveform implementation and porting is a much sought-after feature in the software-defined radio (SDR) community. It not only reduces the acquisition cost to purchase systems and equipment, but also lowers the entry barrier, thus enabling a broader range of organizations to carry out SCA-based SDR research, development and training. This paper describes the development and implementation of a low-cost public safety Project 25 (P25) waveform that has been ported from the SDR-4000, a high-end surrogate JTRS SDR platform, to a low-cost PC and Universal Software Radio Peripheral (USRP) platform. The ported P25 waveform is implemented in an OSSIE environment, which is an open-source SDR core framework based on the JTRS SCA, using Linux as the operating system (OS). The choices of the SDR platform, the core framework and the OS enable a low-cost porting and implementation of the P25 waveform. In this paper, the porting process is described in detail and lessons learned are discussed.

1. INTRODUCTION

The Joint Tactical Radio System (JTRS) program was established to provide next generation radio systems for US military forces [1]. By adopting software defined radio (SDR) technology and leveraging on the inherent programmability using software, the JTRS program aims to provide affordable, high-capacity, and flexible waveforms for rapid field deployment. The cornerstone for achieving this objective is the use of a standardized open architecture – the software communication architecture (SCA), which defines the common interfaces of waveform components [2]. SCA enables software reuse, hence, reduces the development time and associated costs.

The prices of many commercially available SDR platforms and SCA software suites are prohibitively high for entry level players. Therefore, low-cost SCA based waveform implementation and porting is a much sought-after feature in the SDR community. It not only reduces the acquisition cost to purchase systems and equipment, but also lowers the entry barrier, thus enabling a broader range of organizations to carry out SCA-based SDR research, development and training.

In this paper, we describe the porting of the public safety Project 25 (P25) waveform that has been implemented in the SDR-4000, a high-end surrogate JTRS SDR platform, to a low-cost PC and Universal Software Radio Peripheral (USRP) platform [3]. The ported P25 waveform is implemented in an OSSIE (Open Source SCA Implementation Embedded) environment, which is an open-source SDR core framework based on the JTRS SCA, using Linux as the operating system (OS) [4]. This combination of a PC or laptop running Linux and USRP is a low cost generic platform. The USRP is the radio up-converter and down-converter for the PC or laptop.

The USRP family allows you to create SDR using any computer with a USB 2.0 or Gigabit Ethernet port. Various plug-on daughter boards allow the USRP and USRP2 SDRs to be used on different radio frequency bands. Daughter boards are available from DC to 5.9 GHz. The entire design of the USRP family is open source. The USRP product family works with GNU Radio, a free-software (open source) framework for the creation of SDR. In addition to GNU Radio, USRP family products can be driven by the universal hardware driver (UHD) on all major platforms. The UHD library contains a host driver and API for current and future USRP family products. The UHD is particular useful for SCA-based development as it can be simply encapsulated into the SCA device driver.

The OSSIE is developed by Wireless@VirginiaTech. It goal is to enable research and education in SDR and wireless communications. The OSSIE package includes: an SDR core framework based on the JTRS SCA; the Waveform Workshop, a set of tools for rapid development of SDR components and waveforms applications; and libraries of waveform applications and components, etc. The OSSIE uses the omniORB CORBA ORB, which is also openly available.

The choices of the SDR platform, the core framework and the OS enable a low-cost porting and implementation of

Proceedings of the SDR '10 Technical Conference and Product Exposition, Copyright © 2010 Wireless Innovation Forum, Inc. All Rights Reserved

the P25 waveform. The rest of this paper is organized as follows. In Section 2, we introduce the P25 waveform common air interface relevant to this paper. The porting and implementation of the P25 software on USRP are elaborated in Section 3. Conclusions follow in Section 4.

2. P25 COMMON AIR INTERFACE

The common air interface (CAI) [5] is the main P25 element ported in the project. It includes link level packetizing, modulation, source and error correction coding. The P25 voice frame format is depicted in Fig. 1. A voice message consists of a Header Data Unit (HDU) followed by pairs of Logical Data Units (LDU1 and LDU2) that carry a mixture of voice data, link control data, etc. The HDU has 792 bits corresponding to a length of 82.5ms. Each LDU has 1728 bits spanning 180 ms in time. Every pair of LDU1 and LDU2 forms a superframe with a time span of 360ms. A 144-bit Terminator Data Unit (TDU) is sent to signal the end of a message, when a voice session is terminated. The TDU can be 432 bits in length with link control information.

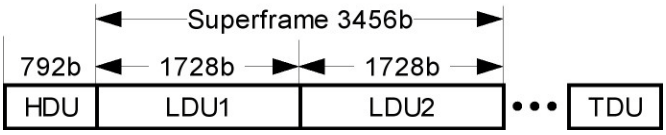

Figure 1: P25 voice frame format.

The main contents of LDUs are voice code words. Each superframe carries 18 voice code words (VC1-VC18), generated from the improved multi-band excitation (IMBE) voice codec, adopted in the P25 standard. The IMBE vocoder is a patented technology of Digital Voice System Incorporated (DVSI). Besides voice code words, link control and protocol information, such as the frame synchronization (FS) sequence, network identifier (NID), message indicators, talk-group ID, etc, are carried by the HDU, LDU and TDU. To ensure that information is delivered correctly, forward error correction (FEC) codes are used. Most link control and protocol information are coded with Reed-Solomon as the inner code and either Hamming or Golay as the outer code, except that the NID has only one layer of Golay coding and FS is a bit pattern without coding.

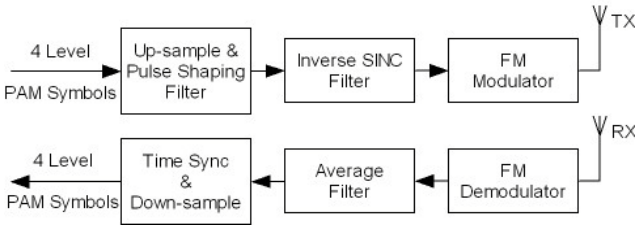

Figure 2: The C4FM modem architecture.

The Phase 1 P25 CAI uses continuous 4-level FM (C4FM) non-linear modulation and occupies 12.5 kHz bandwidth for analog, digital or mixed mode transmission. The P25 C4FM uses a 4-level pulse amplitude modulation (PAM) to map bits into symbols and has a baseband baud rate of 4800 symbols per second. The mapping of input bits to 4PAM symbols and C4FM frequency deviations are listed in Table 1. Every 4PAM symbol carries 2 input bits, so the raw bit rate in P25 is 9.6 kbps.

Table 1 : C4FM Symbol Mapping

Bits	4PAM Symbol	C4FM Frequency Deviation (kHz)
01	+3	+1.8
00	+1	+0.6
10	-1	-0.6
11	-3	-1.8

The implementation of the C4FM modem is illustrated in Fig. 2. The 4-level PAM symbols pass through two filters before the FM modulator. Those filters are defined by the standard [5]. The first filter is a Nyquist raised cosine filter for pulse shaping. If needed, this filer also can realize a sampling rate change, which is common in SDR implementations. The second filter is a truncated inverse SINC filter. The standard only defines the inverse SINC filter response within the cut-off frequency of the first low pass raised cosine filter. In the end, the over-sampled, pulse shaped and filtered 4PAM signals are used to modulate the phase of the RF carrier signal. The reverse process takes places in the receiver chain. The FM demodulator output is first filtered with a rectangular window, i.e., an averaging filter, to compensate for the inverse SINC filter at the transmitter. Then, time synchronization is acquired by correlation of the FS sequence. Down sampling is performed to extract 4PAM symbols for detection, decoding and packet processing.

3. PORTING P25 TO LINUX/USRP

This section describes the porting of a P25 waveform implementation to the Linux and USRP environment. The original P25 waveform was implemented on SDR-4000. As pointed out in [6], the porting process includes two steps. First, the waveform software is migrated from the SDR-4000 to the Linux/USRP environment without SCA. In this step, the key targets are ensuring accurate operation of the waveform in the new platform and maximizing the code reuse. In the second step, the waveform is partitioned into SCA-compatible resources and encapsulated into SCA components. In the following, the setup of the Linux/USRP platform is introduced first, following by the software archi-

Proceedings of the SDR '10 Technical Conference and Product Exposition, Copyright © 2010 Wireless Innovation Forum, Inc. All Rights Reserved

tecture of the P25 implementation. The SCA partitioning is described in the end of this section.

3.1 Using Linux/USRP as SDR Platform

The setup of the Linux/USRP platform we used for one P25 radio node is illustrated in Fig. 3. It consists of a laptop running Linux OS and a USRP N210 kit. The laptop and the USRP are connected with a Gigabit Ethernet cable.

This is a genuine SDR platform in the sense that the waveform is fully implemented using software in Linux, a generic operating system, environment. The USRP is essentially a radio frequency up-converter and down-converter for the laptop. Complex baseband signal samples are transferred between the USRP and the laptop over the Gigabit Ethernet cable. In additional, the laptop can control the behavior of the USRP, such as center frequency, power, sampling rate, through the Ethernet.

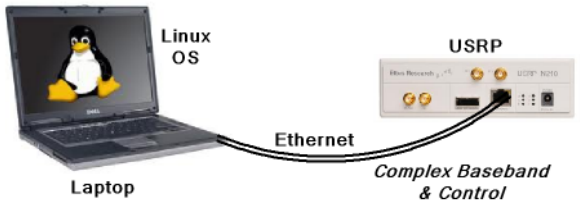

Figure 3: Linux/USRP as SDR platform.

3.2 Software Architecture

The P25 waveform was originally implemented in the SDR-4000, a high-end surrogate JTRS SDR platform consisting of devices such as GPP, DSP and FPGA. The P25 waveform implementation in the SDR-4000 is thus divided into three parts, as shown in Fig. 4. Supporting waveform porting to another SDR platform was one key requirement during the development of the P25 waveform implement on SDR-4000. As a result, the C code written for packet processing in GPP and baseband processing in DSP can be directly reused.

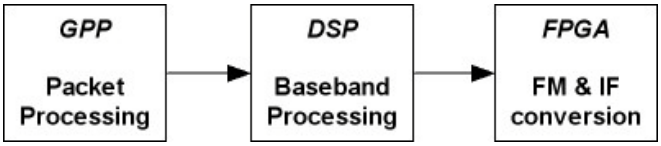

Figure 4: Partition of P25 waveform in SDR-4000.

Different from the implementation on the SDR-4000, waveform developers for the Linux/USRP environment no long need to handle different computing devices such as GPP, DSP and FPGA, as well as the partition of waveform among those devices.

In order to speed up the porting process and maximize code reuse, the initial software architecture of the P25 waveform in the Linux/USRP consists of three independent processes. They are: 1) the packet processing process, corresponding to the software implemented in the SDR-4000 GPP; 2) baseband processing process, corresponding to the software implemented in the SDR-4000 DSP; and 3) the FM modem process, as shown in Fig. 6. These three processes correspond to the P25 waveform partition in the SDR-4000, as shown in Fig. 5. The three processes communicate with each other using the Unix Domain Socket, which is a widely used inter-process communication (IPC) scheme on the Linux/UNIX platform. The FM modem process also interfaces with the USRP through the UHD library.

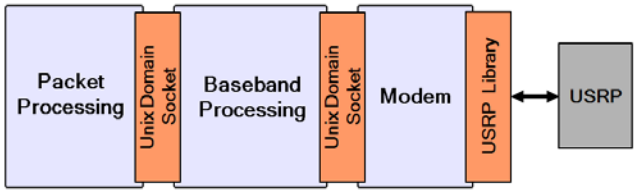

Figure 6: Non-SCA P25 software architecture on Linux.

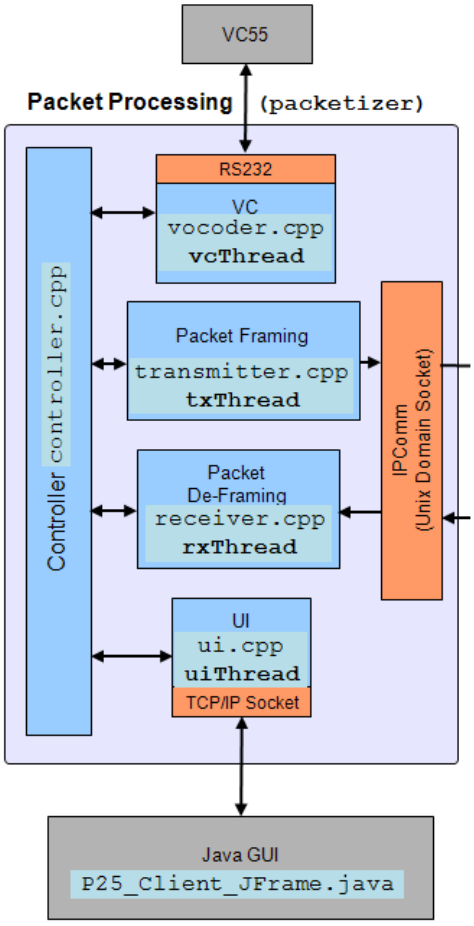

Figure 7: Software architecture for the P25 packet processing.

Proceedings of the SDR '10 Technical Conference and Product Exposition, Copyright © 2010 Wireless Innovation Forum, Inc. All Rights Reserved

3.2.1 The Packet Processing Process

The packet processing process handles framing and deframing of P25 waveform. It interacts with the user interface to accept commands and display waveform operational information. Besides, the packet processing process also interfaces the voice codec devices for voice encoding and decoding. The software architecture for this process is illustrated in Fig.7. It consists of following four threads.

- **User interface thread** (uiThread in Fig. 7) interacts with the P25 java GUI through TCP/IP socket. It accepts user commands, such as "push to talk", and displays information to P25 users, such as channel ID, channel status.
- **Voice codec thread** (vcThread in Fig. 7) interfaces the external voice codec device to accept coded voice to be transmitted and send received voice code words.
- **Receiving thread** (rxThread in Fig. 7) extracts voice code words from the received P25 frames.
- **Transmitting thread** (txThread in Fig. 7) packages voice code words to be transmitted into P25 frames.

All four threads are controlled by the packet processing controller, which creates the four threads when the P25 waveform is initialized. The receiving thread and transmitting thread interact with the baseband process discussed in the following through the UNIX domain socket.

3.2.2 The Baseband Process

The baseband process performs the 4PAM modulation and demodulation. In the transmitter part, this process also filters the generated 4PAM signal with the raised cosine filter and the inverse SINC filter. In the receiver chain, the baseband process first performs the averaging filtering first and then correlates the received 4-PAM signal with prestored FS sequence to synchronize the signal and demaps the symbols to bits. The bandband process communicates with both the packet process and the modem process through the UNIX domain socket.

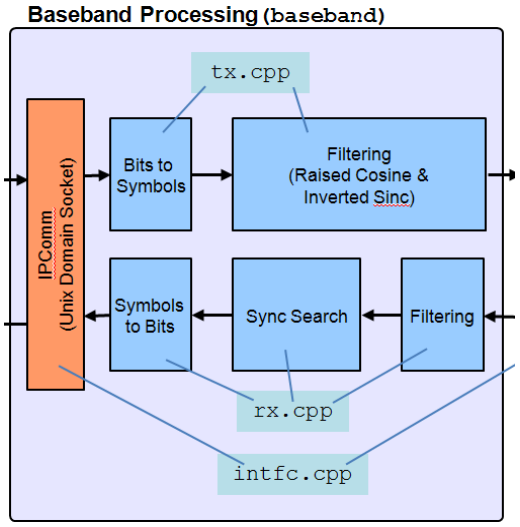

Figure 8: Software architecture for the P25 baseband process.

3.2.3 The FM Modem Process

The FM modem process performs the frequency modulation and demodulation, as well as interfaces the USRP through the UHD library. The software architecture of the modem process is depicted in Fig. 9.

The sampling rate of the USRP ADC/DAC is configured to be 200 ksps (kilo sample per second), while the sampling rate of the baseband processing is 48 ksps. In order to match the sampling rate of the baseband processing and that of the USRP ADC/DAC, the modem process also re-samples the transmitted signal from the bandband process and the received signal from the USRP.

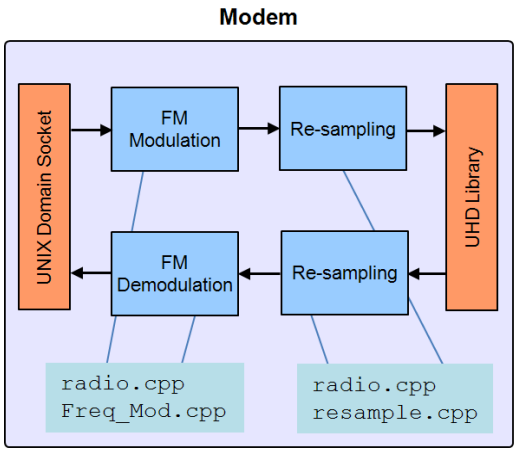

Figure 9: Software architecture for the modem process.

3.2.4 Architecture Evolution

The software architecture described previously retains the legacy of the SDR-4000 version of the P25 waveform implementation. In particular, the two steps in the C4FM modulation, 4-level PAM and FM, are split into two different processes, while the FM is combined with the re-sampling operation and interfacing with the USRP device.

A better division among multiple tasks is achieved if the FM is moved from the modem process to the baseband process. Thus, the generic P25 waveform implement will only have two processes – the packet processing and the baseband processing. The software of this architecture evolution is more portable. If the radio front end is changed from the USRP to another one, only the code in the modem process needs to be changed in order to match the requirement of the new radio front end. The code of the packet processing and baseband processing can be fully reused.

Proceedings of the SDR '10 Technical Conference and Product Exposition, Copyright © 2010 Wireless Innovation Forum, Inc. All Rights Reserved

3.3 P25 SCA Component Partitioning in Linux/OSSIE

After the software is migrated from the SDR-4000 to the Linux/USRP environment without SCA and the P25 functionalities are verified, the Linux P25 software implementation is partitioned into multiple SCA resources. Each SCA resource is added with SCA ports and SCA properties to formulate an SCA component.

The strategy currently adopted is to partition the packet process into SCA components while keep the baseband and modem processes as an external device. This is another carrier over legacy from the SCA based P25 implementation on the SDR-4000, since only the packet processing is implemented in the GPP. A modem device is created, which interfaces SCA resources using CORBA and communicates with the baseband process through TCP/IP.

The packet processing is thus partitioned into 5 SCA components, among them 3 are SCA devices. They are

- User Interface Device
- Vocoder Device
- Modem Device
- Transmitter
- Receiver.

SCA device is an SCA component that prescribes to the SCA device state machine. Since each SCA component is an independent executable, the four threads described in Section 2.2.1 are converted into processes. The function of packet processing controller is absorbed into the User Interface Device for the OSSIE version, but was a separate component in the SDR4000 version.

4. SUMMARY

In this paper, we implement the public safety P25 waveform in a low cost SDR platform using Linux and USRP with OSSIE environment. It demonstrates that it is feasible to use Linux and USRP as the entry level system to enable a broader range of organizations to carry out SCA-based SDR research, development and training.

5. ACKNOWLEDGMENT

This work was supported by the Joint Program Executive Office of the Joint Tactical Radio System (JPEO JTRS) through SPAWAR Systems Center – Pacific under contract no. N66001-08-D-0155/T.O.0001.

6. REFERENCES

[1] http://jpeojtrs.mil [Online].

[2] “Software Communication Architecture Specifications,” Version 2.2.2., May16, 2006.

[3] http://www.ettus.com/ [Online].

[4] http://ossie.wireless.vt.edu/ [Online].

[5] “Project 25 FDMA Common Air Interface”, TIA Std. TIA-102.BAAA-A, September 2003.

[6] P. Johansson, Z. Cao, and W. Hodgkiss, "Rapid Porting of an SCA-compliant FM3TR waveform," in Proceedings of the SDR Forum Technical Conference, Washington, DC, 2009.

[7] Z. Cao, and et. al., "SCA-compliant Public Safety P25-FM3TR-VoIP Bridge," in Proceedings of the SDR Forum Technical Conference, Washington, DC, 2010

[8] Z. Cao, and et.al., “Rapid Development of a JTRA P25 Waveform,”, in Proceedings of the 2010 IEEE Military Communication Conference, San Jose, CA, Oct. 2010.

Proceedings of the SDR ’10 Technical Conference and Product Exposition, Copyright © 2010 Wireless Innovation Forum, Inc. All Rights Reserved

Copyright Transfer Agreement: The following Copyright Transfer Agreement must be included on the cover sheet for the paper (either email or fax)—not on the paper itself.

"The authors represent that the work is original and they are the author or authors of the work, except for material quoted and referenced as text passages. Authors acknowledge that they are willing to transfer the copyright of the abstract and the completed paper to the SDR Forum for purposes of publication in the SDR Forum Conference Proceedings, on associated CD ROMS, on SDR Forum Web pages, and compilations and derivative works related to this conference, should the paper be accepted for the conference. Authors are permitted to reproduce their work, and to reuse material in whole or in part from their work; for derivative works, however, such authors may not grant third party requests for reprints or republishing."

Government employees whose work is not subject to copyright should so certify. For work performed under a U.S. Government contract, the U.S. Government has royalty-free permission to reproduce the author's work for official U.S. Government purposes.

Proceedings of the SDR '10 Technical Conference and Product Exposition, Copyright © 2010 Wireless Innovation Forum, Inc. All Rights Reserved