Scientific Research Publishing

Journal of Computer and Communications, 2018, 6, 66-82
http://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Chip Design for In-Vehicle System Transmitter

Majeed Nader, John Liu

Electrical and Computer Engineering Department, Wayne State University, Detroit, USA
Email: majeed@wayne.edu

How to cite this paper: Nader, M. and Liu, J. (2018) Chip Design for In-Vehicle System Transmitter. *Journal of Computer and Communications*, **6**, 66-82.
https://doi.org/10.4236/jcc.2018.65006

Received: January 27, 2018
Accepted: May 22, 2018
Published: May 25, 2018

Abstract

This paper presents embedded system design of the In-Vehicle System (IVS) for the European Union (EU) emergency call (eCall) system. The IVS transmitter modules are designed, developed and implemented on a field programmable gate array (FPGA) device. The modules are simulated, synthesized, and optimized to be loaded on a reconfigurable device as a system-on-chip (SoC) for the IVS electronic device. All the modules of the transmitter are designed as a single embedded module. The bench-top test is completed for testing and verification of the developed modules. The hardware architecture and interfaces are discussed. The IVS signal processing time is analyzed for multiple frequencies. A range of appropriate frequency and two hardware interfaces are proposed. A state-of-the-art FPGA design is employed as a first implementation approach for the IVS prototyping platform. This work is used as an initial step to implement all the modules of the IVS on a single SoC chip.

Keywords

EU Emergency-Call (eCall), FPGA, In-Vehicle System, Transmitter, System-on-Chip

1. Introduction

The traffic accidents are leading causes of human fatalities around the world. In 2014, the European Commission (EC) revealed that 25,900 people died in car accidents [1]. Also, the US National Highway Traffic Safety Administration has recorded 32,675 fatalities due to the road accidents in 2014 [2]. The car accidents have also resulted in losing billions of dollars each year [3]. Therefore, the transportation agencies have been concerned to decrease the human life loss in road incidents. The European Union (EU) has planned to develop and mandate the emergency call (eCall) system in all vehicles on the European roads after March 2018 [4] [5].

The eCall system is designed to reduce the arrival time of emergency aid to road accidents [6]. Providing the emergency aid in a short time can reduce the fatalities by 11% and the injured disability by 12% [3]. The eCall system includes In-Vehicle System (IVS), Public Safety Answering Point (PSAP) and the cellular network to connect the IVS and the PSAP [6]. During a car accident, the IVS initiate a data link between the vehicle and the PSAP. It collects the Minimum Set of Data (MSD) that includes the information about the accident and the vehicle that are necessary for emergency aid. The MSD contains the GSP location, airbag sensor data, vehicle VIN number and other crucial data for emergency aids. The IVS should transmit the MSD to the PSAP in less than 4 seconds [6]. As soon as the MSD is received by the PSAP, a feedback message is sent by the PSAP to acknowledge of receiving the MSD. The IVS activates the data link and voice channel between the vehicle automatically or manually through a button [6] [7].

The IVS employs multiple sophisticated modules to process the MSD data. The MSD data is read via CAN communication in the vehicle. The IVS transmitter uses a cyclic redundancy check (CRC) algorithm and a scrambler system. Then it encodes MSD data by a Turbo encoder module. The transmitter employs a modulator to modulate the encoded MSD in a Bipolar Pulse Position Modulation (BPPM) system [7]. The IVS transmitter needs the interface solutions to read the MSD data from the vehicle and to transmit the modulated signal via the GSM module. Developing all the modules of the IVS on a single module is challenging and needs a lot of effort to be optimized as a System-on-Chip transmitter.

Eventually, the EU eCall IVS is expected to be a chip. Designing such a chip is a challenging project because the IVS contains a complex state machine and needs to perform multiple sophisticated signal processing. Usually, the first stage of an application specific IC (ASIC) development is to design, implement and test all the functions on an FPGA. This approach is chosen by many researchers because it gives designers an excellent opportunity for optimizations [8]. Hardware implementation has shown better performance compared to software implementations for many applications [9]. Developing the IVS modules on a programmable device has not been implemented before, and it gives many advantages in terms of processing time and reliability. FPGA is highly recommended in the modern hardware design and VLSI [10] [11]. FPGA devices can be used to perform the critical processes that are used in many hardware designs [12], sensor networking [13], and image processing [14]. FPGAs give the flexibility to perform the designed modules and free designers from the cost of fabrication during the optimization process of new designs. Moreover, FPGA devices have good reconfigurability facility and allow performance tuning during the prototype phase of hardware designs [15] [16].

Recently, the eCall system has been the topic of researches by many scholars. However, at the time of publishing this work, no one has implemented the eCall

transmitter or receiver on a single chip. The authors in [2] analysis the updates and progress in the eCall system. They study the standardized parts of the eCall system, and they conclude that both 911 and 112 are suitable cellular links for eCall application. B. Jon *et al.* [5] have studied an enhanced eCall system that provided a video channel to the eCall system. Our previous published works [17] [18] [19] [20] detail the FPGA implementations of single modules of the IVS. M. Werner *et al.* [7] present an in-band modem to implement the EU eCall IVS. The approach that is employed in this work has shown good performance in hardware implementations. Researchers have employed FPGA technologies for many hardware implementations including receiver channelization [21] and embedded fuzzy controllers [22].

This work presents design and development of the IVS transmitter on a single FPGA chip. All the modules of the IVS transmitter are integrated into one module. The performance and the chip size of the designed module are optimized. Verilog HDL is employed to develop the RTL of the developed transmitter. Xilinx ISE and VIVADO tools are employed for the simulation and implementation. The results of the simulation are presented. The IVS transmitter is also developed in C code for validation purposes. The same algorithms of the developed modules are simulated in C [23]. The results that are obtained from the C code and the FPGA implementation are correlated.

The rest of this paper is organized as follows. Sections 2 details the different modules of the IVS transmitter and presents the modulation and multiplexing of the uplink waveform. In Section 3, the simulation results are presented. It also shows the hardware implementation of the IVS transmitter. In Section 4, the validation and system test results are illustrated. Section 5 discusses the employed interfaces between the IVS transmitter and the ECU as well as the GSM module. Finally, the conclusions are provided in Section 6.

2. The IVS Transmitter Architecture

Figure 1 illustrates the IVS block diagram and system architecture of the IVS transmitter. The developed module is a single integrated chip that processes the functionality of the IVS transmitter. The 1120-bits of the MSD is appended with the 28 bits of the CRC parity bits. The encoded MSD data is 1148 bits. The MSD data is applied to a dedicated scrambler. The scrambler is designed based on the 3GPP standard for EU eCall system [6]. The scrambler improves the MSD data bits by reducing the long stream of zeros or ones in the MSD data. The output of the scrambler is the 1148 bits of the CRC encoded MSD. The output of the scrambler is applied to the Turbo encoder module.

The Turbo encoder encodes the 1148 bits of the MSD. The code rate of the Turbo encoder is 1/3. The output of the Turbo encoder is 3456 bits. The process of the encoder module is detailed in chapter three. The encoder employs a Hybrid Automatic Repeat-request (HARQ) technique to generate eight different versions of the encoded MSD. The data bits of the MSD revisions are then applied to the modulator.

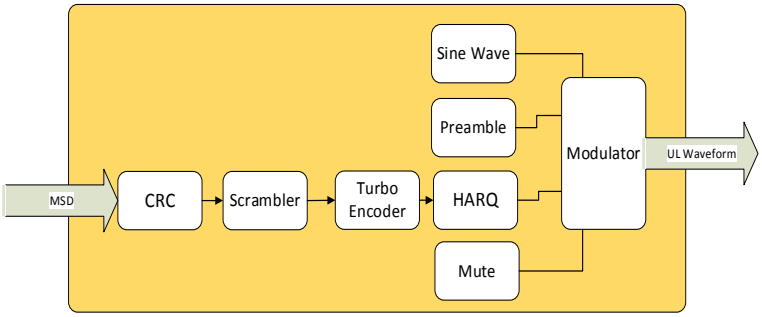

Figure 1. The In-Vehicle System transmitter structure.

The modulator groups the bits of the encoded MSD bits into three bits. Each three bit represents a symbol. There are eight ($2^3 = 8$) different symbols. The modulator modulates each symbol and generates a dedicated uplink waveform for it. The uplink waveform structure is detailed in [20]. The modulator starts the modulation with the synchronization signal and generates the output signal according to the 3GPP standard for the EU eCall system. The GSM module and the FPGA module are interfaced through the specific I2S that is built-in in the GSM module. The interfacing will be explained in the next sections.

2.1. The IVS CRC and Scrambler

Denote the MSD data bits by $a = (a_1, a_2, a_L)$, $L = 1120$. The cyclic generator polynomial that is used for generating the parity bits for the IVS is represented by Equation (1) [4] [18].

$$\begin{aligned} g(X) = {} & X^{28} + X^{26} + X^{24} + X^{23} + X^{18} + X^{17} + X^{16} \\ & + X^{15} + X^{14} + X^{11} + X^{8} + X^{4} + X^{3} + 1 \end{aligned} \tag{1}$$

Let the parity check bits be $p = (p_1, p_2, \cdots, p_{28})$. The generated polynomial of the appended MSD with the 28-bit CRC can be represented as:

$$a_1 X^{L+27} + a_2 X^{L+27} + \cdots + a_L X^{28} + p_1 X^{27} + p_2 X^{26} + p_3 X^{25} + \cdots + p_{27} X^{1} + p_{28}$$

To optimize the CRC processing time a CRC parallel algorithm is employed. The details of employed CRC are detailed in [18]. Denoting the module processing time (in clock cycles) by T_s for CRC serial technique and by T_p for CRC parallel technique, one has

$$T_s = T_r + T_c + T_g = 1120 + 1120 + 28 = 2248 \tag{2}$$

$$T_p = T_r + T_g = 1120 + 28 = 1148 \tag{3}$$

where T_r is the time for reading the 1120 bits of the MSD, T_c is the time for calculating the parity check bits for the MSD data, and T_g is the time for generating the 28 parity check bits on the output port.

And,

$$T_g = \frac{28}{1120} T_r \quad \text{and} \quad T_r = T_g$$

Then one has,

$$T_s = 2.0248T_r \quad (4)$$

$$T_p = 1.0248T_r \quad (5)$$

$$T_p = 0.506T_r \quad (6)$$

Equation (6) shows that almost half of the processing time is saved by using the CRC parallel technique compared with the CRC serial calculation.

The employed scrambler implements the scrambling scheme based on the 3GPP standard [6]. It uses a stored scrambled sequence to scramble the CRC encoded MSD. The scrambled sequence is 1148 bits which is the same length of the CRC encoded MSD. It uses an XOR operation between the bits of the scramble sequence and the MSD data. If the MSD data contains a long stream of zeros or ones, the scrambler reduces the number of length of the zeros and ones' streams in the data. The scrambler operation can be modeled as:

$$S_c(i) = S_{crc}(i)\ XOR\ S_{SCR}(i);\ \ i = 0,1,\cdots,1147 \quad (7)$$

where S_c is the output bit of the scrambler, S_{crc} is the MSD bit, and the S_{SCR} is the bit of scrambled sequence.

The scrambler sequence is stored in the FPGA device. The sequence is designed based on the 3GPP standards.

2.2. The Employed HARQ

The Turbo encoder employs an HARQ technique to build eight different versions, RV0, RV1, …, RV7, of the encoded MSD. Each version consists of 1380 bits that are selected from the 3456 bits of the encoded MSD. The selection of the revision bits is systematic and based on the 3GPP standards for the EU eCall system. The modulator modulates the encoded MSD starting from RV0, then RV1, and so on until it gets an Acknowledge feedback message from the PSAP. It is expected that in most cases the PSAP detects and demodulates the MSD from RV0.

The HARQ technique is designed and simulated in Xilinx software. The developed IVS transmitter implements the HARQ on the employed Turbo encoder on an FPGA device. The bit sequence selection technique is based on a stored sequence in the FPGA device. There is a specific sequence for each of the revisions of the encoded MSD.

The first revision of the encoded MSD, Rv0, contains the entire 1148 bits of the MSD + CRC data. In a good transmission channel, it is expected that the PSAP detects and demodulates the MSD based on RV0. The IVS transmitter transmits eight revisions of the MSD in a row until it receives the "Acknowledge" feedback message from the PSAP. **Figure 2** shows the structure of the implemented revisions of the encoded MSD. The synchronization frame (SF) is regenerated after every eight revisions of the MSD until the PSAP detects and receives the MSD data.

2.3. The Modulated Signal

The modulator modulates the encoded MSD bits that consist of the RVs. Each

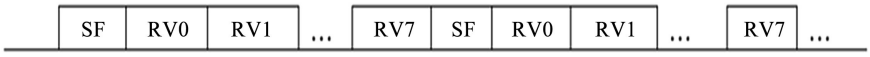

Figure 2. The eight revisions of the encoded MSD. SF is the synchronization frame.

revision consists of 1380 bits. Each symbol represents 3 bits, so the bits of each revision are grouped into 460 symbols. The modulator modulates RV0 by generating 460 uplink waveforms. The uplink waveform starts with synchronization frame. The synchronization frame is proceeded by the data frame. The data frame is multiplexed with synchronization fragments and mute signals. The Synchronization frame, synchronization fragments, and muting signals are utilized for tracking purposes in the PSAP during detection and demodulation process. The modulation and multiplexing are designed to meet the standard 3GPP protocol for the UE eCall system.

The uplink waveforms are represented by 32 samples, and each sample is 16 signed bits. The sampling rate is 8 KHz. The clock frequency is 16 × 8 KHz = 128 KHz. Denote the uplink waveform as UP_{bit}, one has:

$$UP_{bits} = 16 \text{ bits} \times 32 \text{ samples} = 512 \text{ bits}.$$

The uplink waveforms are mapped based on the basic waveform $S_{UL}(k)$, $k = 0,1,\cdots,31$ [6]. The basic waveform is:

$$S_{UL} = \left(0,0,0,0,0,40,-200,560,-991,-1400,7636,15000,7636,-1400,-991,\right.$$
$$\left.560,-200,560,-991,560,-200,0,0,0,0,0,0,0,0,0,0,0,0,0,0\right)$$

The modulation frame contains one uplink waveform. Therefore, the modulation frame is 4 ms at 128 KHz clock frequency. The speech frame which is 20 ms contains five modulation frames.

The uplink waveform starts with the synchronization frame. The Synchronization frame starts with the synchronization tone proceeded by the synchronization preamble signal. Both the synchronization tone and preamble are developed according to the 3GPP standards for the EU eCall system.

The synchronization tone is a sine wave with 800 Hz. The wave structure is based on the proposed sine wave in the 3GPP standard for the EU eCall system it can be expressed as:

$$Syn_{SIN}(n) = \left(0,11756,19021,19021,11756,0,-11756,-19021,-19021,-11756\right)$$

where $n = 0,1,\cdots,9$. Each sample of the sine wave is represented in 16 signed bits. See **Table 1**.

The basis of the synchronization preamble is a pseudo noise (PN) sequence of length 15 that takes values of (1, 1, 1, 1, −1, 1, −1, 1, 1, −1, −1, 1, −1, −1, −1). MATLAB is used to regenerate the PN sequence as is shown in **Figure 3**.

Five periods of the PN build the pulse sequence of the preamble synchronization. The first and last periods are inverted. The repeated samples between the origin PN sequence and inverted ones are transmitted once only. The result is 69 samples of the preamble sequence [6] [20]. **Figure 4** shows the generated pulses of the preamble sequence.

Table 1. Sine wave in analog and digital samples.

Sample Value	Digital Value
0	0000 0000 0000 0000
11756	0010 1101 1110 1100
−11756	1101 0010 0001 0100
19021	0100 1010 0100 1101
−19021	1011 0101 1011 0011

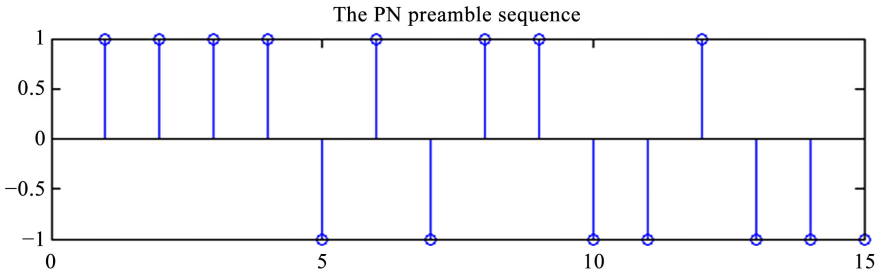

Figure 3. The employed PN code is generated in MATLAB.

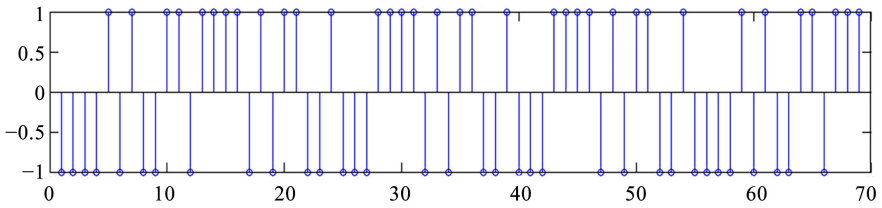

Figure 4. The pulse sequence of the preamble synchronization.

The preamble signals consist of positive and negative versions of the same value which is 20,000. The positive preamble signal (1) is represented by 20,000. The negative preamble signals (−1) are represented by −20000. The preamble values are separated by 21 zero samples. The preamble signal starts with 71 zero samples. Therefore, the preamble signal, including the zero samples, consists of 1568 samples. Note that 71 + 69 + (68 × 21) = 1568. The preamble samples are represented in 16 signed bits. See Table 2.

The sine wave (Syn_{sin_t}) is repeated for 64 ms:

$$Syn_{SIN\_t} = 64 \text{ ms}$$

And the synchronization frame is followed by 196 ms of preamble signal (*Synpreamble_t*),

$$Syn_{preamble\_t} = 196 \text{ ms}$$

The entire duration of the synchronization frame (*Synsignal_t*) is 260 ms.

$$Syn_{signal\_t} = Syn_{SIN\_t} + Syn_{preamble\_t} = 64 \text{ ms} + 196 \text{ ms} = 260 \text{ ms}$$

2.4. The Modulation and Multiplexing

The modulation and multiplexing are designed and developed based on the

3GPP standard of the eCall system. The uplink waveform starts with the synchronization frame to be used as a wake-up signal. The duration of the frame which includes the synchronization tone and preamble signal is 260 ms at 128 KHz of the clock frequency. The modulator starts the modulation of the MSD symbols. Each revision of the encoded MSD is 460 symbols.

The MSD symbols are modulated in three separated parts. The modulation starts with 20 ms of mute signal (M). There are also mute signals (M) and fragment synchronization (F) frames between the modulated data (D). The duration of the mute signals and fragment synchronizations are illustrated in Figure 5. The mute signal and fragment signal is used for tracking purpose during detection and demodulation in the PSAP. The fragment synchronization frames are the last 576 samples of the synchronization preamble which is prepended by 64 zero samples.

Figure 5 shows the modulated uplink waveform of RV0 prepended by the synchronization frame. The design module repeats the modulated waveform for the rest of the eight RVs excluding the synchronization waveform. There is a START message input to the designed module. The transmission does not start until the module receives the START message from the PSAP. The start message is a single digit input in this design. The transmission is activated when the START message is 1.

3. FPGA Design and Implementation

The modules of the IVS transmitter are designed as a single module to process the sophisticated signal processing of the MSD transmission. The module implements the CRC, the scrambler, the Turbo encoder, the HARQ technique, the multiplexer, and the modulator on a single module. The module is the IVS transmitter on a single chip that processes as a System-on-Chip (SoC). The RTL of the modules is developed in Verilog HDL. Xilinx ISE 14.7 and VIVADO 2016 are employed to simulate and implement the IVS transmitter.

Table 2. Preamble samples in analog and digital formats.

Sample Value	Digital Value
0	0000 0000 0000 0000
20000	0100 1110 0010 0000
−20000	1011 0001 1110 0000

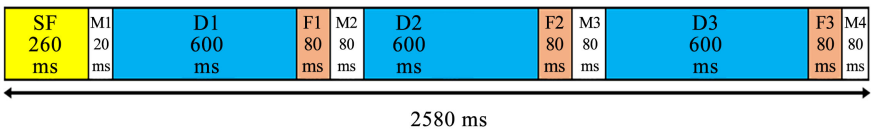

Figure 5. The uplink modulation frames. SF is the synchronization frame, M is the mute gape, D the modulated MSD data, and F is the fragment synchronization signal. The total duration of one revision MSD modulation is 2580 ms at 125 KHz clock frequency.

3.1. Simulation

The simulation of the designed IVS transmitter is presented in this section. The designed module is simulated in Xilinx ISE 14.7 which is the latest ISE version. The simulation of the single modules is detailed in our published works [15], [16] and [17]. This work has developed all the IVS modules as a single embedded module on an FPGA device. The simulation is based on the frequency that is supported by the I2S interface port of the GSM module. The clock frequency is 256 KHz, and the frequency of the WA signal is 8 KHz. The transmitter chip is developed to be compatible with the I2S of the employed GSM. The START message should be activated (high) so that the transmitter module starts the data transmission. The samples of the uplink waveform are transmitted when the WA signal is high.

The uplink waveform of RV0 is simulated, and the result is shown in **Figure 6**. Refer to the labeling of the figure to see the different parts of the uplink waveform. **Figure 6** shows that all the parts of the uplink waveform that is explained in **Figure 5** are generated. The parts of the waveform include the synchronization frame (Sine + Preamble), Mute (M) signals, Fragment (F) signals, and the Modulated Data (D). Note that the waveform is simulated in 2580 ms. According to the 3GPP standard, the waveform should be transmitted in less than 4 seconds.

The signal that is shown in **Figure 6** which is the uplink waveform of RV0 (starting from M1 to M3) is repeated for eight times. Then the synchronization frames retransmitted which will be followed by another eight uplink waveforms of RV0. **Figure 7** illustrates the modulated waveform of the eight revisions.

To look at the simulated waveform in more details, multiple parts of the simulated output signal are examined. **Figure 8** illustrates the synchronization frame which includes 64 ms of the synchronization tone (sine wave at 800 Hz), and 196 ms of the preamble signal. The entire duration of the synchronization frame is 260 ms as is shown in **Figure 9**.

The sine wave $Syn_{SIN}(n)$ is generated based on Section 2.3. The output is in digital format. **Figure 8** shows the ten samples of a complete sine wave.

3.2. Hardware Implementation

This design employs an FPGA platform to implement the designed modules. The platform is a ready-to-use FPGA platform based on the newest FPGA technology [24]. The FPGA platform has a high capacity and a good performance. The platform has 15,850 logic slices with 6-input Look-Up Tables (LUT) that includes 101,440 logic cells. It is a good choice to host designed chips ranging from simple combinational circuits to many sophisticated embedded processors. The functionality of the Nexys4 DDR and the FPGA device is detailed in [24]. The utilized FPGA kit is shown in **Figure 10**. The Verilog HDL is employed to design the modulator module. The most updated hardware development software is used to compile and synthesize the designed modules.

The developed transmitter module is optimized to be implemented on the FPGA device using the optimization features of the employed hardware development tools. This work employs the latest versions of the FPGA tools and Verilog HDL to design, simulate, synthesize, and implement the developed modules of the IVS modem.

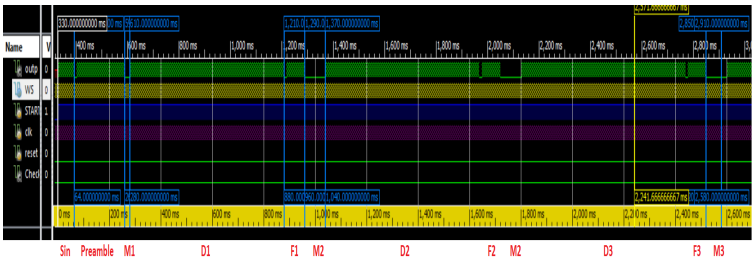

Figure 6. The simulated uplink waveform for RV0. The uplink waveform (outp) is the green trace, WA signal (WS) is the yellow trace, START signal is the blue trace, and reset is also in green.

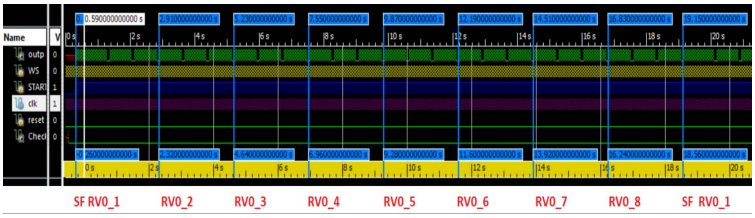

Figure 7. The simulated uplink waveforms of the eight revisions. The uplink waveform (outp) is the green trace, WA signal (WS) is the yellow trace, START signal is the blue trace, and reset is also in green.

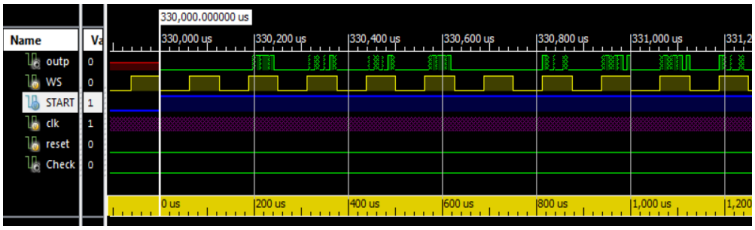

Figure 8. The simulated sine waves. The ten samples of the sine wave as it is shown in Equation (11) are presented. This figure shows that the samples are generated only when WA = 1, and after START is activated (high).

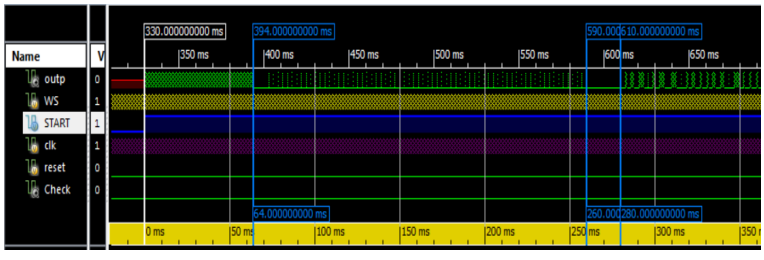

Figure 9. The simulated synchronization frames. The duration of the frame is 260 ms. The uplink waveform (outp) is the green trace, WA signal (WS) is the yellow trace, START signal is the blue trace, and reset is also in green.

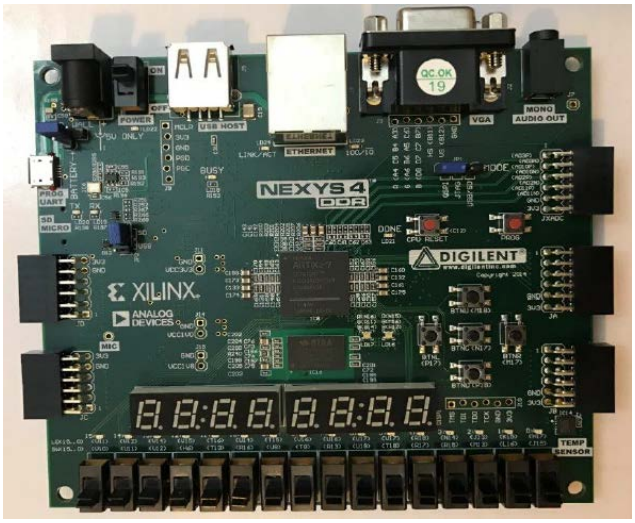

Figure 10. The Nexys4 DDR platform evaluation FPGA kit.

Figure 11 shows the number of utilized Flip Flops, Look-Up Tables (LUTs), and In/Out pins (I0Bs) for the implementation of the designed IVS transmitter modules on the employed FPGA chip. The modules of the IVS transmitter are developed in Verilog. Xilinx VIVADO is used to optimize the developed module. Figure 11 also shows that all the modules of the IVS transmitter are optimized to be designed on the employed FPGA device. Therefore, the device can be employed to implement multiple modules of the IVS modem as a System-On-Chip (SoC).

4. Verification and Test Results

The IVS transmitter processes the MSD signal in multiple stages before the eight revisions of the encoded MSD are generated. The first stage is the CRC encoding of the MSD. Then the MSD + CRC goes through a scrambling process before it gets into the Turbo encoder. The output of the Turbo encoder is the data bits of the eight revisions (RVs). A validation method is developed to validate the designed module.

The IVS transmitter is developed in C code. The C code implements the exact algorithms that are used in the FPGA designed module. Both systems, the C code and FPGA modules, are developed according to the 3GPP standards of the EU eCall system. The outputs of both systems are correlated to validate the generated revisions of the MSD. Figure 12 illustrates the validation module that is used in this design.

If the generated encoded MSD data of the FPGA module is the same of the C code module, the output of the circuit is zero. If there is a difference between the two encoded MSD, the output of the circuit is one. The validation system is a built-in module in the developed FPGA. The RV0 stream data of a specific MSD data is generated in C Code and stored in the FPGA device. The developed FPGA module processes the same MSD, generates RV0, and correlates the result

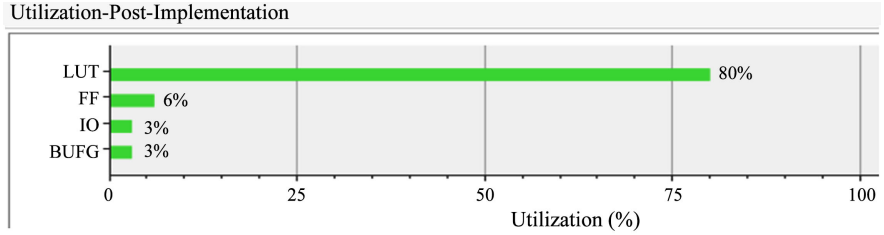

Figure 11. The logic cells utilization for the IVS transmitter on Nexys4 DDR.

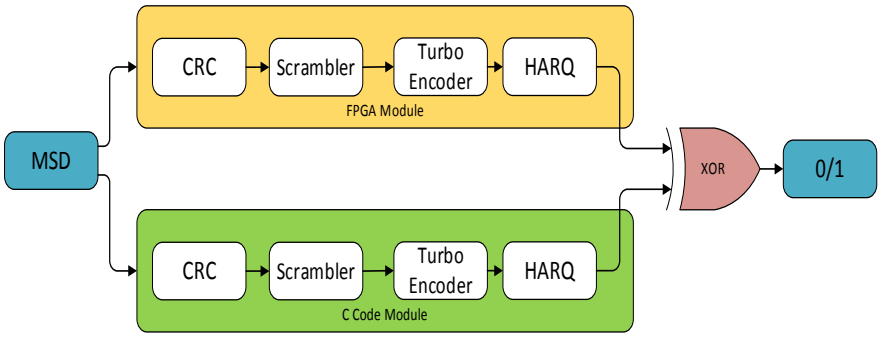

Figure 12. The validation block diagram of the designed IVS transmitter. Both C code and FPGA design are employed for the verification purpose.

with the stored RV0. The validation has shown that the FPGA module has the same result of the C code module.

The designed modules are tested and verified for multiple input patterns. The bench-top testing method is employed to test the developed modules. The input signals are generated and applied to the designed modules. Multiple frequencies are considered for the module testing and verification. The modules are verified to be an embedded part of the IVS modem. The developed transmitter is also tested and verified as a single embedded module.

The developed IVS transmitter is tested by applying the clock frequency and WA to examine the generated output. A function generator is used to generate the clock frequency, which is 256 KHz, and WA signal, which is 8 KHz. The clock and WA signals are used to simulate the I2S of the GSM module. The output of the module is read by using a logic analyzer. The results of the test show that the module is developed as a complete IVS transmitter system. Figure 13 shows that the output of the module which is the uplink waveform is generated when WA is activated (high).

The uplink waveform starts with synchronization frame. Figure 14 shows the frame which includes the sine wave and preamble signal.

Figure 15 shows most of the uplink waveform parts including the sine wave, preamble, the Mute signal (M), the Fragment signal (F), and the Modulated Data (D).

Also by maximizing the uplink waveform that is generated on the output pin, one modulated waveform is shown as it can be seen in Figure 16.

The IVS transmitter is designed and implemented on an FPGA device. A

complete set of the possible input signals is applied to the module for test and verification. Multiple frequencies are used to verify the developed module. It is verified that the modulator modulates all possible input symbols and generates corresponding waveforms accordingly.

5. The Interface Solutions

The IVS has two primary interfaces, one for reading the MSD and another for data transmission between the IVS and the GSM module. **Figure 17** illustrates the structural block diagram of the IVS. The interface between the GSM module and the IVS is achieved through the I2S. The MSD can be read from an ECU through the SPI.

5.1. The IVS/GSM Interface

The designed module employs the I2S interface between the GSM and the FPGA module. The implemented I2S interface is explained in [20] [25]. However, each device has a specified I2S configuration. This design employs the I2S that is supported by the GSM module. The GSM module is LEON G200 [26] [27]. The port of the physical I2S consists of 4 pins.

- I2S_WA (Word Alignment): It is an output signal which is used for synchronization. The frequency of the I2S_WA is always 8 KHz.
- I2S_CLK (Clock Signal): It is the clock frequency of the transmission between the GSM module and the interfaced device. It is an output signal. The frequency is 256 KHz in normal modes.
- I2S_RXD (Received Data): It is an input signal. The stream bits of the data are received through this pin. The Most Significant Bit (MSB) is received first. The length of the received word is 16 bits, in 2nd complement format at 8 KHz sampling frequency.
- I2S_TXD (Transmitted Data): This is an output signal. MSB is transmitted first.

The GSM module supports multiple modes of the I2S configuration. The FPGA module is designed to support two modes: Mode 8 and Mode 9 of the GSM module [25] [26]. In Mode 8, the RXD receives data on the rising edge of the clock. In Mode 9, the data is received on the falling edge of the clock cycle. The data is transmitted on the opposite edges of the clock cycles. In both modes, there is no delay between the WA signal and the TXD signal. The RXD is half a bit delayed with respect to the TXD. The transmission is activated when WA is activated. **Figure 18** illustrates the transmitted bit alignments with the WA signal.

The GSM module only works in Master mode. Therefore, the FPGA is designed to be in the slave mode configuration. The CLK pin is the clock signal of the FPGA device. The transmission time of the FPGA device is designed to work under 128 KHz to meet the 8 KHz sampling frequency requirement of the 3GPP standard. Each sample is represented by 16 signed bits, so the 8 KHz sampling rate is equal to 128 KHz clock frequency.

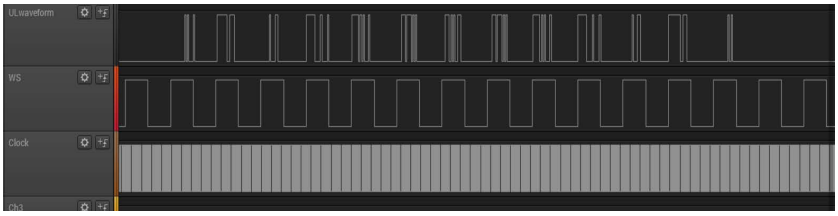

Figure 13. The test result shows the uplink waveform, WA signal, and the clock frequency. The clock frequency is 256 KHz, and WA is 8 KHz.

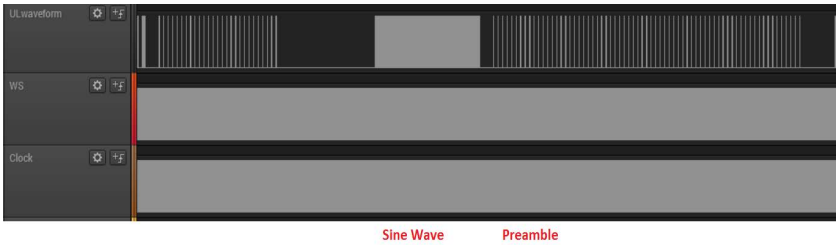

Figure 14. The generated synchronization frames. The signal is recorded on the output of the FPGA kit by using an oscilloscope.

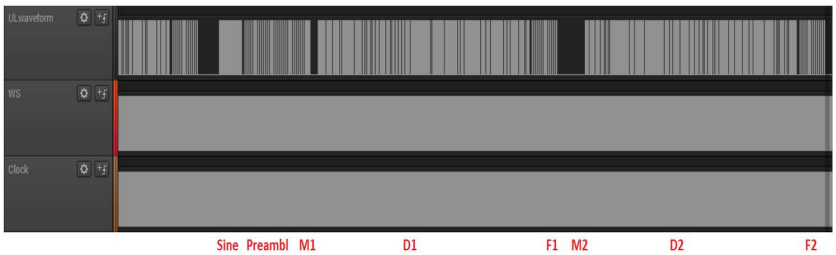

Figure 15. The generated uplink waveform.

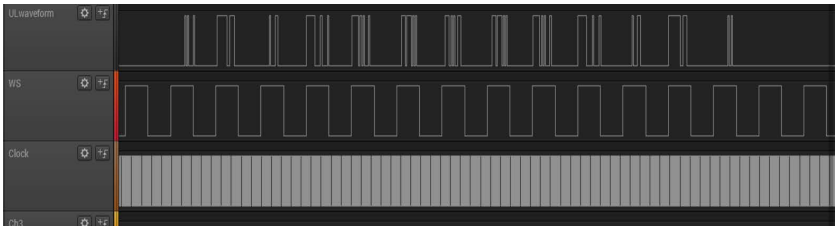

Figure 16. The uplink waveform is generated. It is verified that the generated uplink waveform is the same of the simulated waveform and the designed waveform according to the 3GPP standard for the EU eCall system.

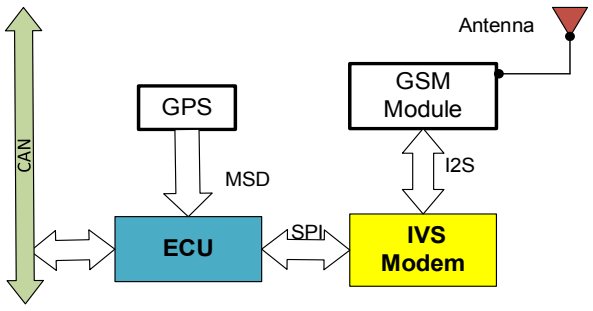

Figure 17. The IVS structure and interfaces.

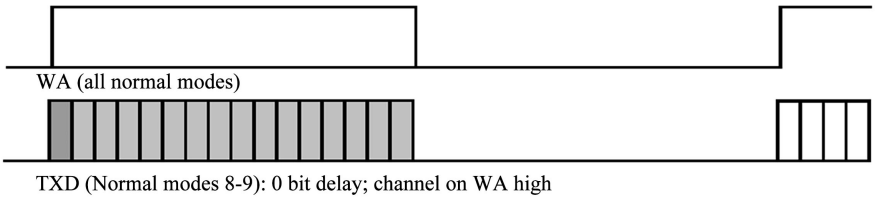

Figure 18. The bit alignment and configuration of mode 8 and mode 9 of GSM LEON G200. The dark gray represents the most significant bit. The transmission is active only when WA = 1.

The frequency of the CLK of the GSM is 256 KHz. This is a compatible frequency with the 128 KHz transmission frequency of the FPGA module because the transmission is activated only when the WA is high. Therefore, the FPGA transmission is activated only half of the time of the CLK of the GSM.

As the GSM receives the MSB first, the FPGA module is designed to transmit the MSB bits of the signal samples. The FPGA is designed to represent each sample of the uplink waveform, the synchronization tone, and the synchronization preamble in 16 signed bits. The FPGA device transmission is activated if and only if the START message and WA signal are high.

5.2. The IVS/ECU Interface

The current design uses a build-in MSD. The MSD that is used for the test is already implemented on the test bench in a laboratory. It consists of 1120 bits. The MSD is appended with CRC, scrambled, encoded by the Turbo encoder, and then the RVs are generated. The modulator modulates the encoded bits and multiplexes the uplink signal with the synchronization frames and the mute signals. The output of the designed module is the modulation of the build-in MSD. If the PSAP receiver receives the signal, it should be able to demodulate and decode the MSD that is known for the test.

The module is also able to read the MSD through a single digital bit as serial data. For standardization, an SPI interface is designed to read the MSD from an ECU.

6. Conclusions

The IVS transmitter is designed and developed as a single system on an FPGA device. The designed modules of the IVS transmitter are analyzed. A hardware architecture is proposed for the IVS of the EU eCall System. The CAN bus is employed for the IVS to communicate with sensors in a vehicle. The 12S bus is proposed to interface the IVS chip with the GSM radio. The CRC and scrambler, the Turbo encoder and the HARQ module, and modulator module of the IVS transmitter are designed, synthesized, and simulated for the EU eCall system application. They are implemented on the latest FPGA device as a single embedded system. The CRC parallel computation for the IVS modem is implemented to reduce the CRC module processing time by 50% compared to CRC serial com-

putation. Equation (6) reveals the improvement.

Based on multiple tests and experiments, it is shown that the designed module is verified for multiple cases. The modules are designed to implement the IVS modem on a single chip as a system-on-chip (SoC). By analyzing the functionality of the developed modules, testing has shown that all the modules have good performances. A complete set of input signals are employed to test and verify the IVS modules. Benchtop test is employed as a method for testing and verification.

All the modules for the IVS of the EU eCall system have been designed, synthesized, and simulated. They will be integrated on one chip and reported in the future papers.

References

[1] European Commission (2016) Mobility and Transport, Road Safety, Statistics Accidents Data. http://ec.europa.eu/transport/road_safety/specialist/statistics/index_en.htm

[2] Öörni, R. and Goulart, A. (2017) In-Vehicle Emergency Call Services: eCall and beyond. *IEEE Communications Magazine*, **55**, 159-165. https://doi.org/10.1109/MCOM.2017.1600289CM

[3] Biox, E. (2014) Definition of a Protocol for Automatic Identification and Notification of Road Accidents and Development of an Advanced eCall System. SAE Technical Paper. https://doi.org/10.4271/2014-01-2029

[4] European Commission (2016) eCall: Time Saved = Lives Saved. Press Release, Brussels. https://ec.europa.eu/digital-single-market/en/ecall-time-saved-lives-saved

[5] Jon, B., *et al.* (2016) eCall++: An Enhanced Emergency Call System for Improved Road Safety. *IEEE*, *Vehicular Networking Conference* (*VNC*), Columbus, 8-10 December 2016, 1-8.

[6] eCall Data Transfer—In-Band Modem Solution—General Description. TS26.267, 3GPP.

[7] Werner, M., *et al.* (2019) Cellular In-band Modem Solution for eCall Emergency Data Transmission. *IEEE*, *Vehicular Technology Conference*, Barcelona, 26-29 April 2009, 1-6.

[8] Engelhardt, D., *et al.* (2014) FPGA Implementation of a Full HD Real-Time HEVC Main Profile Decoder. *IEEE Transaction on Consumer Electronics*, **60**, 476-484. https://doi.org/10.1109/TCE.2014.6937333

[9] Zafarifar, B., Kerkhof, T. and With P. (2014) Texture-Adaptive Skin Detection for TV and Its Real-time Implementation on DSP and FPGA. *IEEE Transaction on Consumer Electronics*, **58**, 476-484.

[10] Muralikrishna, B., Madhumati, G.L., Khan, H. and Deepika, K.G. (2014) Reconfigurable System-on-Chip Design Using FPGA. 2*nd International Conference on Devices*, *Circuits and Systems* (*ICDCS*), Combiatore, 6-8 March 2014, 1-6.

[11] Pan, J., Qi, N., Xue, B. and Ding, Q. (2011) Design and Hardware Implementation of FPGA & Chaotic Encryption-Based Wireless. *IEEE International Conference on Instrumentation*, *Measurement*, *Computer*, *Communication and Control*, Beijing, 21-23 October 2011, 691-695.

[12] Fusella, E., Cilardo, A. and Mazzeo, A. (2015) Scheduling-Aware Interconnect Synthesis for FPGA-Based Multi-Processor Systems-on-Chip. *IEEE* 25*th International Conference on Field Programmable Logic and Applications* (*FPL*), London, 2-4

September 2015, 1-2.

[13] Lu, S., *et al.* (2009) Design and Implementation of an ASIC-Based Sensor Device for WSN Applications. *IEEE Transaction on Consumer Electronics*, **55**, 1959-1967. https://doi.org/10.1109/TCE.2009.5373756

[14] Su, C. and Sie, Y. (2010) An FPGA Implementation of Chaotic and Edge Enhanced Error Diffusion. *IEEE Transaction on Consumer Electronics*, **56**, 1755-1762. https://doi.org/10.1109/TCE.2010.5606322

[15] Kiokes, G., *et al.* (2011) Design and Implementation of an OFDM System for Vehicular Communication with FPGA Technologies. *IEEE* 6*th International Conference on Design and Technology of Integrated Systems in Nanoscale Era* (*DTIS*), Athens, 6-8 April 2011, 1-6.

[16] Yao, C., *et al.* (2012) VLSI Implementation of a Real-Time Vision Based Lane Departure Warning Systems. *IEEE* 12*th International Conference on ITS Telecommunication*, Taipei, 5-8 November 2012, 170-174.

[17] Nader, M. and Liu, J. (2018) Hardware Development of the In-Vehicle System Modules for the EU Emergency Call. *American Journal of Electrical and Electronic Engineering*, **6**, 1-10. https://doi.org/10.12691/ajeee-6-1-1

[18] Nader, M. and Liu, J. (2016) Developing Modulator and Demodulator for the EU eCall In-Vehicle System in FPGAs. *IEEE*, 2016 *International Conference on Computing, Networking and Communications* (*ICNC*), Hawaii, 24 March 2016, 1-5. https://doi.org/10.1109/ICCNC.2016.7440593

[19] Nader, M. and Liu, J. (2015) Design and Implementation of CRC Module of eCall In-Vehicle System on FPGA. SAE Technical Paper 2015-01-2844, *SAE* 2015 *Commercial Vehicle Engineering Congress*, Rosemont, 6-8 October 2015. https://doi.org/10.4271/2015-01-2844

[20] Nader, M. and Liu, J. (2015) FPGA Design and Implementation of Demodulator/Decoder Module for eCall In-Vehicle System. *IEEE*, 2016 *International Conference on Embedded Systems and Applications* (*ESA*15), Vegas, 27-30 July 2015, 3-9.

[21] Tietche, B., Romain, O. and Denby, B. (2015) Sparse Channelizer for FPGA-Based Simultaneous Multichannel DRM30 Receiver. *IEEE Transaction on Consumer Electronics*, **61**, 151-159. https://doi.org/10.1109/TCE.2015.7150568

[22] Sanchez-Solano, S., *et al.* (2017) FPGA Implementation of Embedded Fuzzy Controllers for Robotic Applications. *IEEE Transactions on Industrial Electronics*, **54**, 1937-1945.

[23] eCall Data Transfer—In-Band Modem Solution—ANSI-C Reference Code. TS26.268, 3GPP.

[24] Xilinx (2016) Nexys 4 DDR FPGA Board Reference Manual. Document No. 502-292.

[25] Texas Instruments (2013) Using the I2S Audio Interface of DS90Ux92x FPD-Link III Device. Application Report, SNLA221.

[26] u-blox (2009) LEON-G100/G200. Data Sheet GSM.G1-HW-09001-B.

[27] u-blox (2011) Wireless Modules, Data and Voice Modules. AT Commands Manual, WLS-SW-11000-2.