Hardware Development of the In-Vehicle System Modules for the EU Emergency Call

Majeed Maghdeed Nader

Majeed.nader2@gmail.com

Abstract: This paper presents the hardware design and implementation of the in-vehicle system (IVS) for the European Union (EU) emergency call (eCall) system. Modules of the IVS are 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. Benchtop 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 can be 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, CRC, modulator*

1. Introduction

Telematics has been widely used for vehicle safety and infotainment. Although a lot of effort has been made to reduce automotive accidents, road accident is still one of the leading causes of loss of life. In fact, over a million people die annually because of car accidents. Besides the loss of life, the cost of dealing with road incidents and the treatments of injuries reaches billions of dollars each year [1]. To reduce the fatalities caused by car accidents, the European Union has mandated installation of a built-in emergency call (eCall) system in vehicles to be sold in EU countries starting March 1, 2018 [2,3].

Fast response of emergency centers right after car accidents can reduce the loss of life by 11% and the disability probability by 12% [1]. The EU eCall system can play a vital role in shortening the arrival time of emergency treatment in traffic accidents [4]. There are three crucial parts in the EU eCall system, including the In-Vehicle System (IVS) to be installed in every vehicle, Public Safety Answering Point (PSAP), and the GSM cellular system. The IVS is an electronic hardware that collects data about the accident and the vehicle to build the Minimum Set of Data (MSD) that is required for emergency aids [4]. It uses a cellular module to activate a voice call with the PSAP. There will be multiple stations of the PASP around EU countries. The most appropriate PSAP will be chosen by the IVS according to its location and facilities. The mobile carriers are required to provide the dedicated 112 emergency channel for the eCall system so that the channel is always available [4].

When a car accident occurs, the IVS activates a voice call between the car involved in the accident and the PSAP, transmits the MSD using an in-band modem, and is required to make sure the PSAP can receive the MSD in 4 seconds [4]. The activation can be done manually through a specified button or automatically through the installed sensors in the vehicle [5]. The IVS modem contains multiple modules to process the signals. A cyclic redundancy check (CRC) module is employed to implement the CRC code on the MSD data. A Turbo encoder is used to encode the data before transmission of the MSD data from the IVS. A modulator generates modulated waveforms and transmits the MSD data through a cellular channel [4]. After the IVS activates the eCall uplink channel, it monitors the downlink channel to receive feedback messages from the PSAP. The PSAP detects the uplink channel, demodulates and decodes the MSD, and responds to the IVS with feedback messages in the downlink channel. The feedback messages are Acknowledgment (ACK), Not Acknowledgment (NACK), and START messages. The feedback messages control the status of the IVS transmissions. The IVS employs a receiver that contains a demodulator and a BCH decoder to demodulate and decode the downlink messages from the PSAP.

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 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 [6]. Hardware implementation has shown better performance compared to software implementations for many applications [7]. 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 design of digital electronic circuits and VLSI [8] [9]. FPGA devices can be used to perform the critical processes that are used in many hardware designs [10], sensor networking [11], and image processing [12]. 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 [13,14].

This work studies the design procedures of the CRC and modulator modules that are used in the IVS modem and implements the designed modules on an FPGA device. Verilog HDL is employed to describe the register transfer level (RTL) of the modules. By utilizing a state-of-the-art FPGA synthesis tools, the CRC and modulator modules are designed, simulated, and synthesized. The modules are loaded on an FPGA device. The developed modules are tested and verified for different clock frequencies. The effects of the higher frequencies on the developed modules are studied. The hardware architecture of the IVS modem, the interfaces, and the processing time are considered. The logic utilization of the developed modules is optimized. The goal of this work is to design and develop all the IVS modules on a single chip using FPGA technologies.

The rest of the paper is organized as follows. In Section II, a brief of related works is presented. Section III discusses the architecture of the IVS modem. In Section IV, the design procedure of the CRC and modulator modules is analyzed. In Section V, the FPGA implementation of the designed modules is presented. It also discusses the simulation of the modules. In Section VI, the test and verification procedures are discussed. Conclusions are provided in Section VII.

2. Related Work

At the time of submitting this paper, there are no known authors have published hardware implementation works related to the IVS of the EU eCall system. B. Jon et al. [3], however, have published an enhanced eCall system that provided a video channel to the eCall system. Our previous works [15,16] presented in conferences detail the FPGA implementations of other modules of the IVS. M. Werner et al. [5] presents 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 spread spectrum signaling receivers [17], receiver channelization [18], TV broadcast receivers [19], and embedded fuzzy controllers [20].

3. System Architecture and Interfaces

A hardware architecture is proposed for the EU eCall system. The different modules of the IVS are shown in Figure 1. A micro-controller as an electrical control unit (ECU) communicates with the sensors in a vehicle and builds the MSD to be transmitted. The GPS coordination of the vehicle, which is a crucial part of the MSD data, is determined by a built-in GPS receiver in a car. The ECU communicates with the GPS receiver. The interface between the micro-controller and the CRC module can be implemented through an SPI bus. The CRC module receives 140 bytes of the MSD data, encodes the MSD by generating a 28-bit CRC parity check, and appends the MSD with the parity check bits. The controller area network (CAN) bus is proposed as a bus configuration for the interface between the ECU and the MSD sources in cars [21].

The encoder module encodes the 1148 bits of the MSD. It employs a Turbo encoder that has a 1/3 coding rate. The modulator groups the encoded MSD into symbols and generates the corresponding uplink waveforms for each symbol in digital format. It modulates the encoded MSD data to be sent over a voice channel to the PSAP. The demodulator receives and demodulates the feedback messages that are sent by the PSAP to control the transmission status of the IVS. A BCH decoder is employed to decode the feedback message which is ACK, NACK, or START.

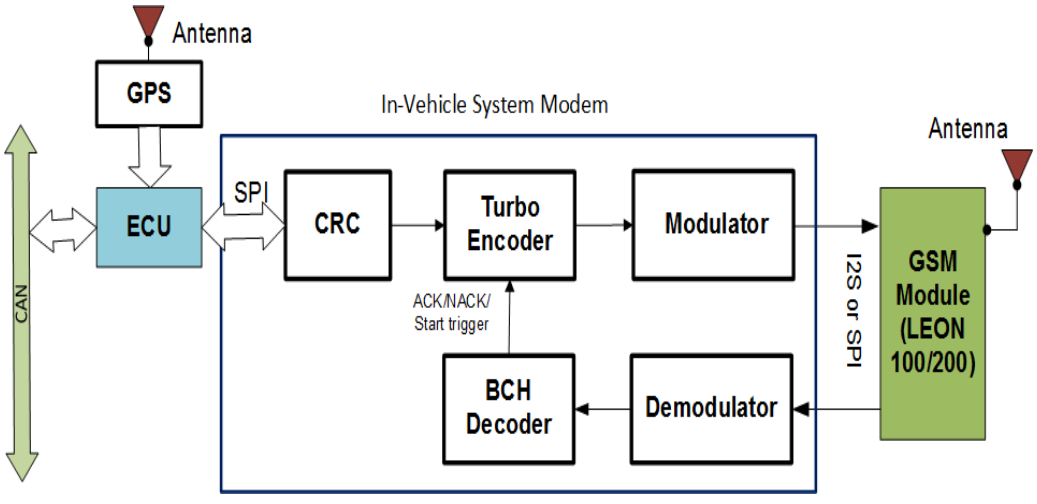


Figure 1. The IVS hardware architecture

The interface between the IVS and the GSM module can be implemented using the Inter-IC Sound (I^2S) bus. The I2S is a standard protocol for exchanging the digital audio data over a three-or four-wire serial link. The protocol was developed by Philips Semiconductor in 1986 and revised in 1996 [22]. It was developed to standardize the digital audio data transmission between different devices that process digital audio signals. The I2S uses three wires for one-directional transmission and four wires for bidirectional transmission.

The controller area network (CAN) bus in a vehicle can be read to obtain information to construct the MSD [21]. The updated data about the status of the sensors installed in the car are available on the CAN bus [21]. The coordination of the vehicle is determined through a global positioning system (GPS). An electronic control unit (ECU) constructs the MSD data and sends it to the IVS modem. The modulated uplink waveforms are sent to the GSM to be transmitted to PSAP. As soon as the GSM receives the downlink waveforms, it transfers the digital waveforms to the demodulator.

The modulator and demodulator are digital devices. The input to the modulator is the stream bits of the encoded MSD, $d = \{d_1, d_2, \ldots, d_L\}$, $L = 3456$. The output of the modulator is the generated uplink waveforms in the form of binary stream bits. Since each waveform is represented by 32 samples and each sample is a signed-16-bits, each uplink waveform consists of 512 bits. The modulated waveforms are sent to the GSM transmitter to be transmitted to the PSAP. As soon as the GSM receiver receives the downlink waveform, it sends the downlink waveform data in binary to the demodulator to be demodulated. The interface between the GSM module and IVS modem can be implemented using inter-IC sound (I2S) bus which is a vital solution to interface ICs with other digital audio processors [22]. The designed modulator module can interface with a GSM module that supports 12S bus. The utilized GSM module supports 12S bus for digital audio data transmission [23,24]. It has four 12S wires, the clock source (SCK), word select (WS), TX, and RX. The sampling frequency is 8 KHz, the word length is 16 bits, and the clock frequency is 256 KHz [24]. These parameters match the designed modules in this work.

The encoded MSD data consists of 1152 symbols, $\underline{S} = \{s_i\}$, $i = 0, 1, \ldots, 1151$. Each symbol is represented by a waveform that consists of 512 bits. Therefore, for each MSD there are 1152 x 512 = 589, 824 bits to be sent through the I^2S bus. With the employed GSM module clock frequency, each MSD data can be sent in 589, 824/256, 000 = 2.3 seconds regardless the chip delay which is only a few milliseconds in this design. Therefore, it can be verified that the design follows the 3GPP eCall standards. This work is an approach to implement the 3GPP EU eCall IVS on one FPGA chip.

The modules are developed to be compatible with multiple clock frequencies. The IVS modules process different amounts of data bits. Therefore, each of the modules has different processing times. Different clock frequencies and parallel processing are employed to optimize the total processing time of the IVS modem.

In order to put all of the IVS modules on a single chip, multiple clock sources and parallel processing between the modules can be employed so that the modem processing time can be optimized. However, the transmission data rate depends on the external cellular modem. In the proposed IVS architecture, the modulator needs 3456 clock cycles to read all the encoded MSD data and 589,824 clock cycles for the modulation process. Table 1 shows the corresponding processing time of the developed modules for each utilized clock frequency.

Table 1. The Clock Frequency and Processing Time

Clock Frequency	The Modulator Processing Time (ms)
256 KHz	2317.5
1 MHz	579.375
5 MHz	115.875
10 MHz	57.93
33 MHz	17.55
100 MHz	5.173

4. The Designed Modules

The CRC and modulator modules are developed. The modules are designed, synthesized, simulated, and implemented on an FPGA device. The employed algorithms are detailed and analyzed.

4.1. The Developed CRC Algorithm

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].

$$g(X) = X^{28} + X^{26} + X^{24} + X^{23} + X^{18} + X^{17} + X^{16} + X^{15} + X^{14} + X^{11} + X^8 + X^4 + X^3 + 1. \quad (1)$$

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

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

The developed CRC module is illustrated in Figure 2. All the functions shown in the figure are described in Verilog and are realized to implement the CRC algorithm. There are two registers; the MSD register stores the input MSD, and the CRC register stores the calculated 28 CRC parity bits. There are 1120 iterations for the 28-parity bit calculation because the MSD input consists of 1120 bits. The CRC-iteration sub-module that is shown in Figure 2 acquires the MSD and CRC data from the two registers and calls the CRC-serial sub-module for each bit of the MSD, and then it sends the calculated CRC parity bits to the CRC register.

The CRC-serial sub-module uses a single bit of the MSD and the previously calculated 28 parity check bits to calculate a new set of the 28 parity check bits and sends them back to the CRC-iteration sub-module. The CRC-serial sub-module is designed to implement the generator polynomial in the equation (1). Implementing the process for every single bit of the MSD, or for 1120

times, the final 28 parity check bits for the entire MSD data are calculated and stored in the CRC register.

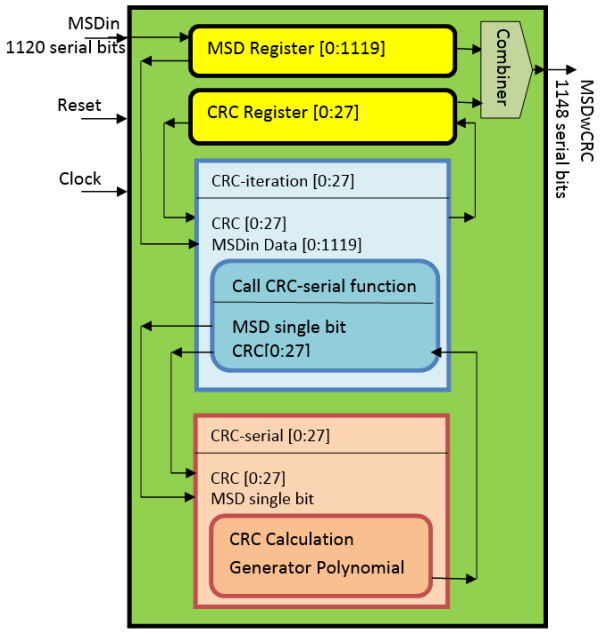


Figure 2. The CRC module structure

As soon as the MSD data bits are registered, and the CRC parity bits are calculated, the combiner generates the output data on the output pin of the module. The output is the appended MSD with CRC parity bits. The CRC module processing time is optimized by using the CRC parallel processing algorithm. It calculates the parity check bits while it reads the input bits. This work employs a parallel technique for the CRC parity bit generation. CRC parallel technique is used for many CRC applications [25,26]. However, employing the technique for data with 1120-bit length gives a significant benefit compared to serial CRC computation techniques.

In this design, the parity check bits for all the 1120 bits of the MSD are calculated in one clock cycle after reading all the inputs. The time for parity bits calculation is overlapped by using a parallel CRC technique. When a serial technique to calculate parity bit is used, 1120 clock cycles are needed to calculate the parity bits.

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

$$T_s = T_r + T_c + T_g = 1120 + 1120 + 28 = 2248 \quad (2)$$

$$T_p = T_r + T_g = 1120 + 28 = 1148 \quad (3)$$

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,

$$T_g = (28/1120)T_r, \text{ and } T_r = T_c.$$

Then one has,

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

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

$$T_p = 0.506T_s. \quad (6)$$

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 designed module works based on a developed algorithm with the flowchart shown in Figure 3. While the chip reads each bit of the serial input bits for the MSD data, it stores the data in the dedicated register. As soon as the 1120 data bits of MSD is built, the system implements the CRC parity bits calculation based on the dedicated generator polynomial in Eq. 1. The calculated CRC parity bits will be stored in another register that is dedicated to storing the 28 bits of the CRC parity check.

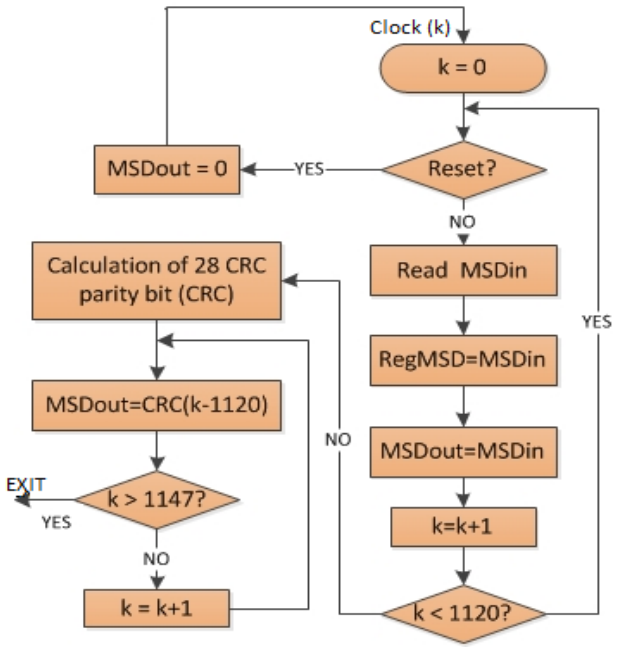


Figure 3. The CRC flowchart

In the flowchart, *k* represents the number of system clock cycles. For each clock, a process of reading one input bit and generating one output bit is performed. The entire process takes 1148 clock period time. As soon as all the MSD is read, the CRC of the last bit is calculated, and the 1148 bits of the output data are generated. The system exits from the flowchart processes and waits for the next MSD to be received and encoded with CRC.

Parallel to the input readings and building the MSD data in the dedicated register, the chip generates the output data which consists of the 1120 MSD bits and 28 parity bits. Therefore, the output MSD data is 1148 bits. For each bit of the total 1120 bits of the MSD input, the process of the CRC parity bits calculation will be repeated based on the designed circuit in Figure 4 which is the hardware implementation of Eq. 1. For the very first bit of the MSD, the initial 28-bits of the parity check is assumed to be zeros. After that for each round, the previously calculated CRC parity bits are used. The calculated CRC of the last bit of the MSD is the final parity bits of the CRC for each MSD.

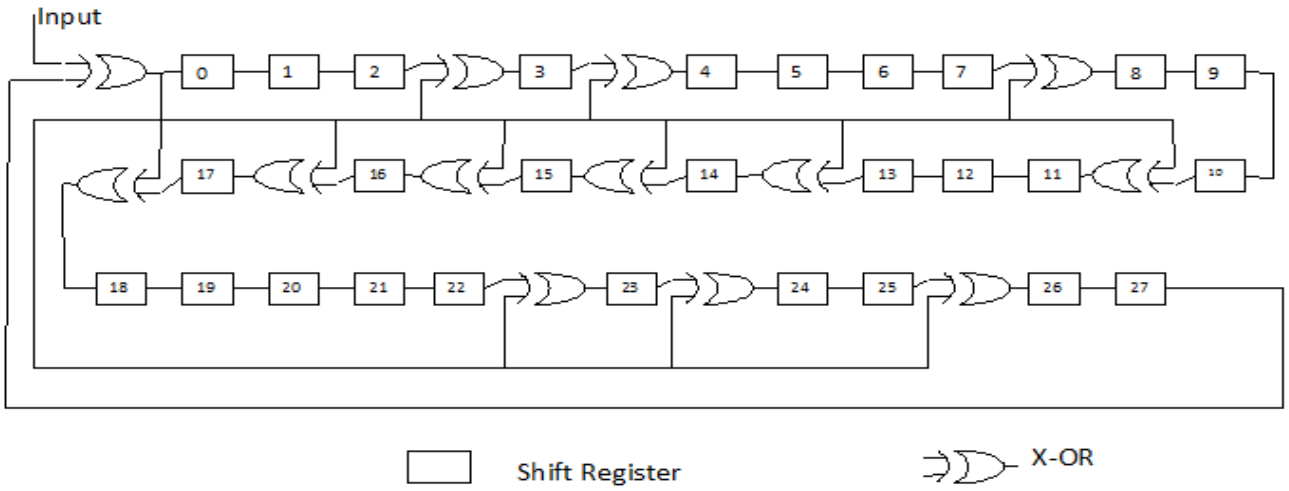


Figure 4. The 28-bit CRC parity check generator circuit

4.2. The Modulator Module

The EU eCall system employs a bipolar pulse position modulation (BPPM). The modulator input signal is the encoded bit stream from the turbo encoder which is a forward error correction (FEC) code. The length of the coded MSD at the turbo encoder output is 3456 bits [4]. The modulator takes three bits at a time and generates a corresponding waveform. There is a total of $2^3 = 8$ possible waveforms. The modulator output signal is a digital signal, and it is the input signal of the GSM module.

When the IVS is activated, it starts sending MSD as an uplink message to the PSAP. After sending MSD, IVS expects acknowledgment from the PSAP. If the IVS receives the NACK message from the PSAP, it re-transmits the MSD until it receives the ACK message. The IVS also can reuse the downlink message to activate the connection between the IVS and the PSAP. In this case, the IVS uses the downlink message to push the PSAP to send a START message and monitor the received signal to detect the START message from the PSAP [4].

The MSD consists of 1120 bits. After implementing the CRC on the MSD and appending it with the 28 parity check bits, 1148 bits are encoded through a turbo encoder which has a 1/3 code rate. The output of the encoder, including the trellis bits, is a binary stream of 3456 bits.

The modulator groups the encoded bit streams into symbols and generates the corresponding waveform for each symbol. Each symbol is represented by 3 bits, so there are eight possible symbols.

Denote the encoded MSD bits as $d = \{d_1, d_2, \ldots, d_L\}$, L = 3456. The symbol sequence is $\underline{S} = \{s_1, s_2, \ldots, s_i\}$, i = 1152. The uplink waveforms are mapped based on the basic waveform $S_{UL}(k)$, k = 0, 1, ..., 31 [4]. The basic waveform is:

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

Figure 5 shows the Matlab simulation of the basic waveform. There are eight different uplink waveforms for the eight different possible symbols. BPPM is used as the modulation scheme [5].

There are 3 cyclic right shift versions of the positive or negative basic waveform to represent the different symbols. The uplink waveforms of the eight symbols are simulated in Matlab and shown in Figure 6.

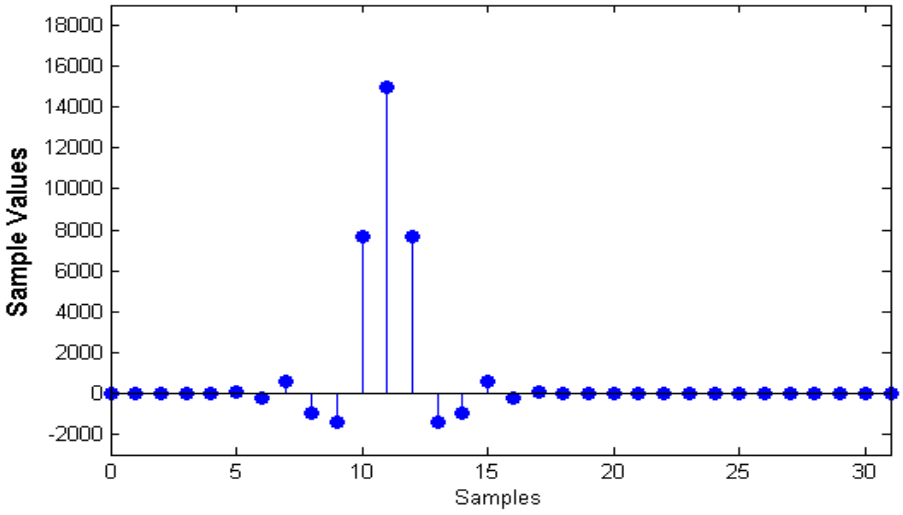


Figure 5. The basic uplink waveform, the waveform is represented by 32 samples

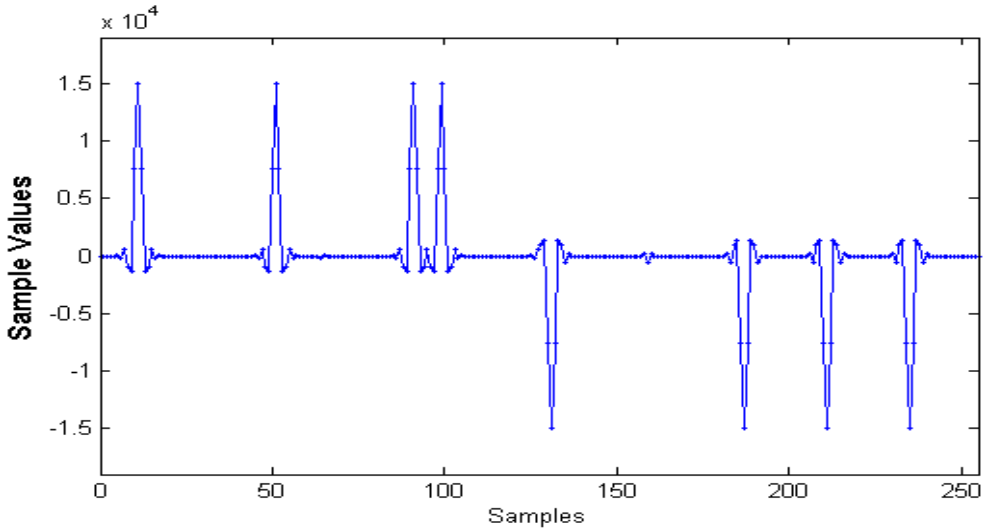


Figure 6. The uplink waveforms of the eight possible symbols are simulated in Matlab

5. FPGA Implementation

The designed modules are implemented on an FPGA device. The RTL of the modules is developed in Verilog HDL. In order to study the performance of the designed modules, two test benches are developed to simulate the CRC and modulator modules separately. One of the powerful FPGA device and the latest RTL software are employed to simulate and implement the modules.

5.1. 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 [27]. The FPGA platform has a high capacity and a large FPGA chip. The platform has 15,850 logic slices with 6-input Look-Up Tables (LUT) that includes 101,440 logic cells. It is an excellent choice to host designed chips ranging from simple combinational circuits to many sophisticated embedded processors. 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 CRC and modulator modules are 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 7 shows the number of utilized Flip Flops, Look-Up Tables (LUTs), and In/Out pins (I0Bs) for the implementation of the designed CRC module. It also shows that only a small portion of the device is used to implement the developed CRC. Therefore, the device can be employed to implement multiple modules of the IVS modem as a System-On-Chip (SoC).

The modulator module is designed, synthesized, and simulated for different inputs. The stream bits of the output uplink waveforms are generated. The logic cells utilization for the design is optimized to use the minimum amount of the available logic cells as it is shown in Figure 8.

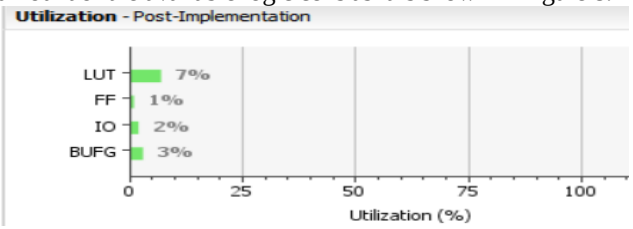


Figure 7. The logic cells utilization for the CRC module on Nexys4 DDR

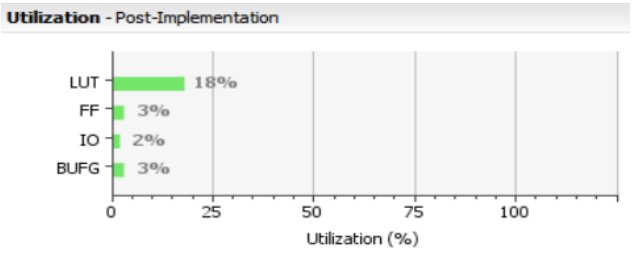


Figure 8. The logic cells utilization for the modulator on Nexys4 DDR

5.2. Simulation of the Designed Modules

A test bench is developed by using Verilog HDL for each of the designed modules. The modules are simulated for multiple input patterns. An FPGA RTL design software is employed to simulate and study the modules.

1) The CRC Module Simulation: Multiple MSD patterns are used for the simulation. A few MSDs are shown as examples to explain the simulation. Two samples of the MSD patterns are named as MSD One and MSD Two. MSD One consists of 1120 bits of all ones. MSD Two is a train of 1120 of alternating ones and zeros (101010...10). These two sets of MSD data are chosen as they are easy to generate in a function generator and can evaluate the system accurately.

The designed digital circuit for the CRC module is evaluated and tested by comparison with the generated CRC parity bits in a C code for the CRC developed

program. The C code program uses the same generator polynomial and MSD data to generate the CRC parity bits. The C code is developed and evaluated based on the 3GPP ANSI code [28].

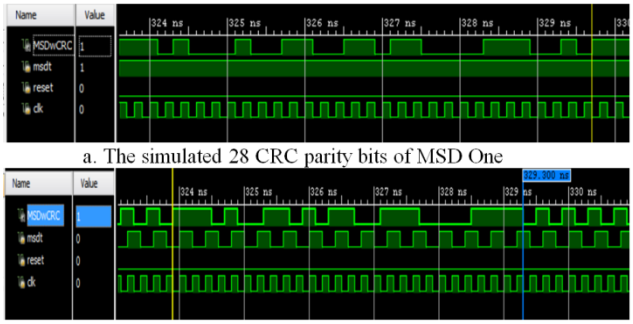
For MSD One, it can be shown in the C code that the 28 CRC parity bits using the generator polynomial in equation 1 are:

$$(0100\ 0100\ 1100\ 1101\ 1000\ 0111\ 0010).$$

For MSD Two, the CRC parity bits are:

$$(1001\ 0111\ 1001\ 0100\ 0100\ 0100\ 1111).$$

After developing, compiling, and synthesizing the CRC module in Verilog HDL, the system is simulated using a powerful FPGA design tool. Employing the Verilog HDL, a test bench is developed to simulate the designed system for the mentioned MSD One and MSD Two. The 28 CRC parity bits of the simulation is shown in Figure 9. Note that the 28 CRC bits for both MSD One and MSD Two in the simulation results are similar to the results of the C code program. Therefore, it is verified that the designed system after the RTL level works accurately.

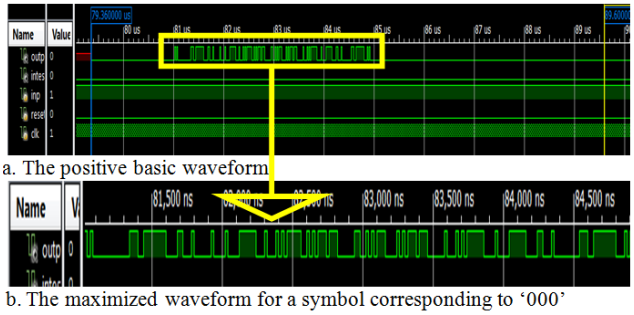


a. The simulated 28 CRC parity bits of MSD One

b. The simulated 28 CRC parity bits of MSD Two

Figure 9. The CRC parity bits from the simulation

2) The Modulator Simulation: The modulator module is simulated for all possible inputs. In all cases, it was verified that the module generates the desired uplink waveform. The test bench code is written in Verilog to simulate the designed modulator. All the possible symbols were used to generate the uplink waveforms. For example, Figure 10 shows the binary format of the uplink waveform for the 000 symbol which is a positive waveform. The simulation is based on 50 MHz for the clock frequency. The clock cycle is 20 ns. Therefore, the uplink waveforms, which consist of 512 bits, are generated in 512X20ns = 10240ns.



a. The positive basic waveform

b. The maximized waveform for a symbol corresponding to '000'

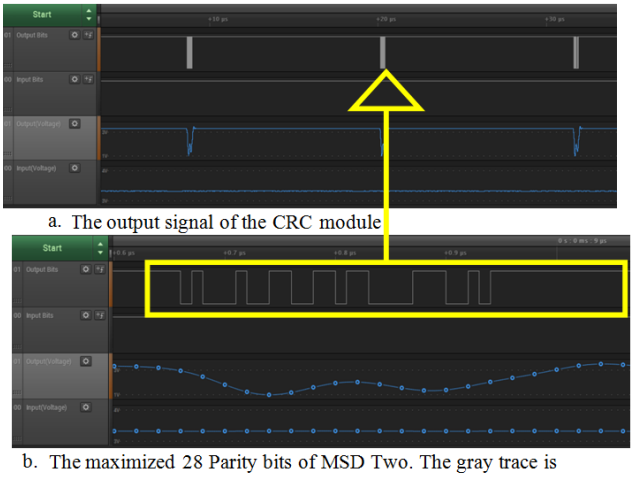
Figure 10. The simulation result of the modulator module, it is a sample of the uplink waveform

6. Test and Verification

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.

6.1. Test and Verification of the CRC Module

After the module is loaded on the FPGA device, two function generators are used, one for generating the clock to run the FPGA chip and the other for generating the MSD data bits. By utilizing a multi-channel logic analyzer, the output bits of the FPGA device is displayed which are the coded MSD with CRC. As an illustration of the test procedure, the MSD One is applied to the input CRC module. The actual CRC parity bits for MSD One are shown in Figure 11. A logic analyzer is employed to display the output signals. The obtained results of Case One are similar to the results from the C code and the simulation. Although the analog version of the signal is not clear due to the high clock frequency, the digital bits are generated accordingly. A range of frequency up to 100 MHz is used for the clock frequency



a. The output signal of the CRC module

b. The maximized 28 Parity bits of MSD Two. The gray trace is the digital format. The blue trace is the same signal in analog format

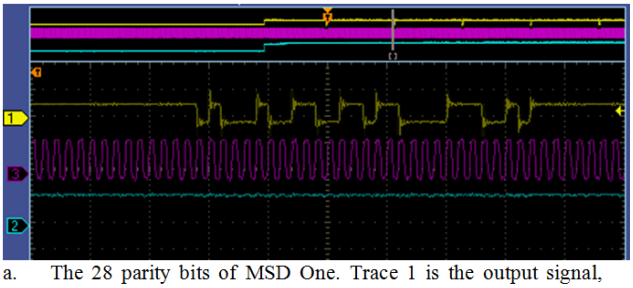
Figure 11. The generated output signal of the implemented CRC module on an FPGA device

An oscilloscope is employed to show the output signal in the lower frequency to see the actual bits streams. The clock frequency for Case Two is 5 MHz. The generated 28 CRC parity bits for MSD Two is shown in Figure 12. The signal in yellow, which is the trace 1, is the output, while the trace 2 is the input signal. The clock is the trace 3 in purple.

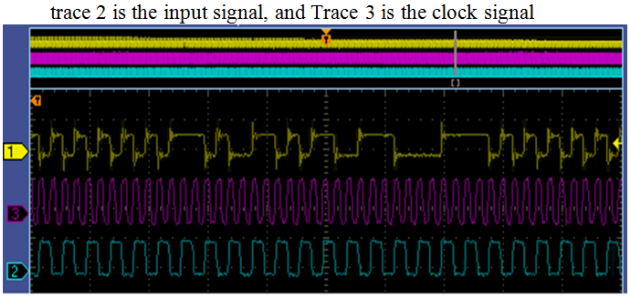
Note that the results on the oscilloscope have risen up and down at the edge of bit toggling. This is due to the high frequency of the system clock. When the signal was showed on a logic analyzer, a clean result is received as illustrated in Figure 12.

Also, 10 MHz is used as a clock frequency, and MSD Two is applied to the system. Although the results contain more distortion at the edge of toggling bits, the FPGA

works properly, and the bits can be read and distinguished as illustrated in Figure 13. Note that in all cases for both MSD One and Two, the generated 28 CRC parity bits are the same as those of the C code program and the FPGA simulation. This is verification of the performance of the CRC module for the IVS of the eCall system that is designed and implemented in FPGA.



a. The 28 parity bits of MSD One. Trace 1 is the output signal, trace 2 is the input signal, and Trace 3 is the clock signal



b. The 28 parity bits of MSD Two. The 28 parity bits of MSD One. Trace 1 is the output signal, trace 2 is the input signal, and Trace 3 is the clock signal

Figure 12. The generated CRC parity bits of the MSD at the FPGA device output. The employed clock frequency is 5 MHz

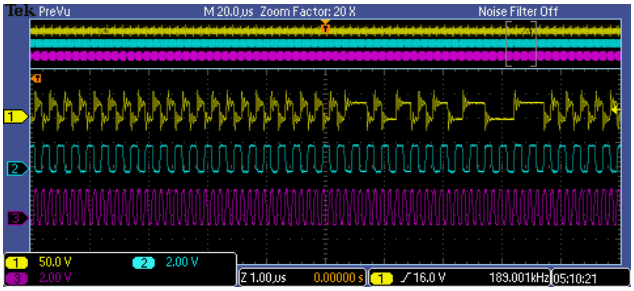


Figure 13. The generated MSD appended with 28 CRC parity bits at the output of the FPGA device. The clock frequency is 10 MHz. Trace 1 is the output signal, Trace 2 is the input signal, and Trace 3 is the clock signal

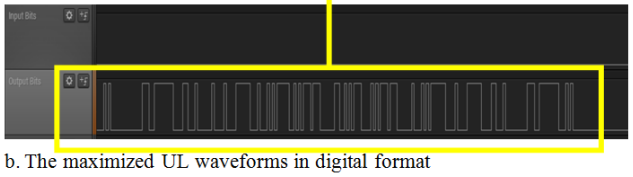
6.2. Test and Verification of the Modulator

A sub-module is designed to generate symbol bits $d = \{d_1, d_2, ..., d_k\}$, k = 3456 to be applied to the input pin of the designed modulator for testing purposes. Different frequencies for the tests and verifications are used. By using different frequencies, the effects of different clock frequencies are examined. The selected frequencies are 1 MHz, 5 MHz, 10 MHz, and 33 MHz. A separate FPGA development kit generates these clock frequencies. An Open-Sparc LM509 evaluation FPGA kit [29] is employed to generate the clock frequencies. These frequencies also can be generated by a function generator. For the 100 MHz, the built-in oscillator on the FPGA kit was employed.

The digital modulator is tested by applying a complete set of symbols to the input pin and examining the generated waveforms. As the interface between the modulator and the GSM module can be done through 12S, the generated output waveforms are in binary. Each waveform is represented by 512 bits.

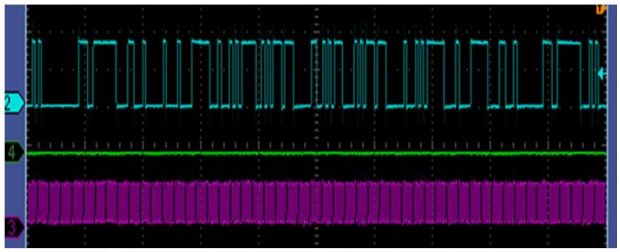
1) Case One: For this case, the "000" symbols are used as the input of the modulator. 100 MHz is employed as a clock source to run the designed chip. Figure 14 illustrates the generated uplink waveforms in Case One by using 100 MHz as a clock source. There is a group of similar uplink waveforms in series (see the output bits waveform) because all the input signals are "000" symbols.

a. Multiple UL waveforms. The input symbol is “000”

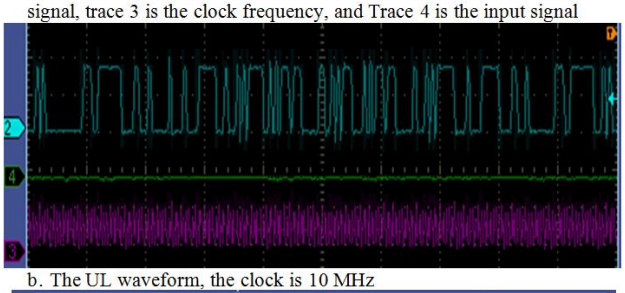


b. The maximized UL waveforms in digital format

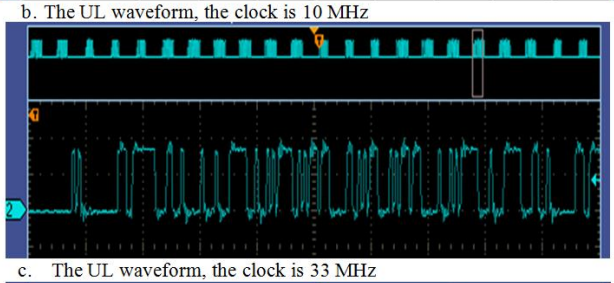
Figure 14. The generated UL waveforms in digital format for the “000” symbol; the clock frequency is 100 MHz



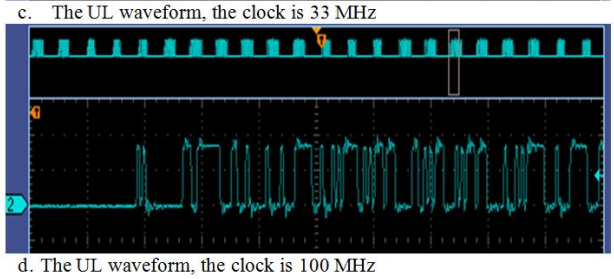
a. The UL waveform, the clock is 1 MHz. Trace 1 is the output signal, trace 3 is the clock frequency, and Trace 4 is the input signal



b. The UL waveform, the clock is 10 MHz



c. The UL waveform, the clock is 33 MHz



d. The UL waveform, the clock is 100 MHz

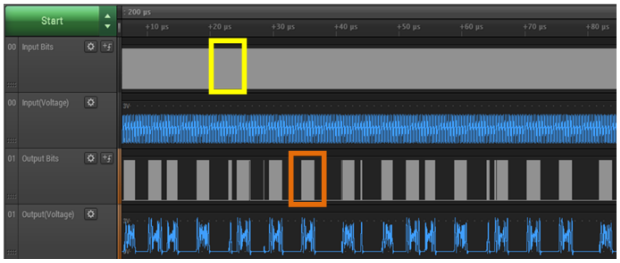
Figure 15. The generated UL waveforms for Case One with different clock frequencies

Figure 15 shows the waveform for different utilized clock frequencies on an oscilloscope. It can be seen that it is the positive uplink waveform, which is the corresponding waveform for "000" symbols. To verify that the uplink waveform is generated accordingly, note that the generated waveform in Figure 16-a is the same simulated uplink waveform in Figure 11. The lower waveform (purple) is the clock frequency, and the middle waveform (green) is the input data.

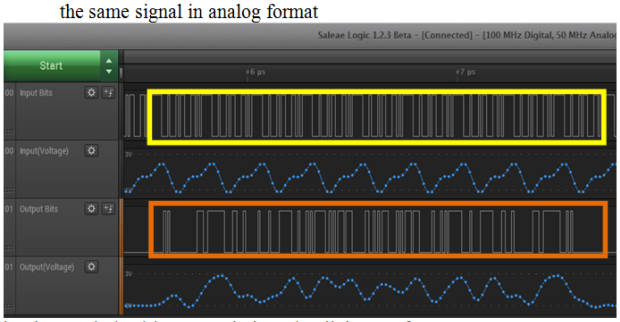
For the same case, different clock frequencies were employed. The generated uplink waveforms are examined. It is noted that the signal is clearer with lower frequencies. Distortion can be seen at 33 MHz or 100 MHz. The distortion was caused by the utilized FPGA kit at higher frequencies. However, the uplink waveform is still detectable and recognizable. By using a logic analyzer, all the generated signals can be seen clearly. In the verification of the demodulator, it will be seen that the FPGA device is able to detect the downlink waveform at all frequencies. Therefore, it is verified that the designed chip can perform the modulation process of IVS by using all the clock frequencies from 1 MHz to 100 MHz.

2) Case Two: A complete set of the symbols are applied to the input pin of the modulator. The eight input symbols include 000, 001, ..., 111 in a row. The modulator groups the input binary bit stream into three bits for each symbol and generates the corresponding waveform accordingly. The output generated waveforms are a series of the basic waveform with different signs and shifts. Figure 16 shows the generated waveform of Case Two. By using the logic analyzer, both the analog signal and digital bits of the output are illustrated.

Note that the generated uplink waveforms are repeated periodically. This is because the input binary stream is a collection of the same eight possible symbols, so the generated uplink waveforms are the eight different possible waveforms which are similar in a period of eight waveforms.



a. The uplink waveforms for all the possible input symbols. The gray traces are the digital format of the signals, and the blue traces are the same signal in analog format



b. The maximized input symbols and uplink waveforms

Figure 16. The uplink waveforms for Case Three, all the possible input symbols are applied to the modulator input, the clock frequency is 100 MHz

The modulator module 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.

7. Conclusions

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 modulator modules of the IVS transmitter are designed, synthesized, and simulated for the EU eCall system application. They are implemented on the latest FPGA device. The CRC parallel computation for the IVS modem is implemented to reduce the CRC module processing time by 50% compared to CRC serial computation.

Based on multiple tests and experiments, it is shown that the designed modules can run with a frequency range

from 256 KHz up to 100 MHz. The modules are designed to implement the IVS modem on a single chip as a system-on-chip (SoC). The functionality of the developed modules is analyzed. Test results have shown that all the modules have good performance. 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] E. Biox, "Definition of a protocol of automatic identification and notification of road accidents and development of an advanced eCall system," *SAE Technical Paper*, doi:10.4271/2014- 01 -2029. Mar. 2014.

[2] European Commission, "eCall: Time saved = lives saved," Press Release, Brussels, January 26, 2016. Website: *https://ec.europa.eu/digital-single-market/en/ecall-time-saved-lives-saved.*

[3] B. Jon, *et al.*, "eCall++: An enhanced emergency call system for improved road safety" in *IEEE, Vehicular Networking Conference (VNC),* Columbus, OH, 2016, pp. 1-8.

[4] "eCall data transfer; in-band modem solution; general description," 3GPP, Tech. Rep. TS26.267.

[5] M. Werner, *et al.*, "Cellular in-band modem solution for eCall emergency data transmission" in *IEEE, Vehicular Technology Conference,* Barcelona, Spain, 2009, pp. 1-6.

[6] D. Engelhardt, *et al.*, "FPGA implementation of a full HD real-time HEVC main profile decoder," *IEEE Trans. on Consumer Electronics*, vol. 60, no.3, pp. 476-484, Nov. 2014.

[7] B. Zafarifar, T. Kerkhof, and P With, "Texture-adaptive skin detection for TV and its real-time implementation on DSP and FPGA," *IEEE Trans. on Consumer Electronics*, vol. 58, no. 1, pp. 476-484, Mar. 2014.

[8] B. Muralikrishna, G.L. Madhumati, H. Khan, and K.G. Deepika, "Reconfigurable system-on-chip design using FPGA," in *2nd International Conference on Devices, Circuits and Systems (ICDCS),* 2014, pp.1-6.

[9] J. Pan, N. Qi, B. Xue, and Q.Ding, "Design and hardware implementation of FPGA & chaotic encryption-based wireless," in *IEEE International Conference on Instrumentation, Measurement, Computer, Communication and Control*, 2011, pp. 691-695.

[10] E. Fusella, A. Cilardo, and A. Mazzeo, "Scheduling-aware interconnect synthesis for FPGA-based Multi-Processor Systems-on-Chip," in *IEEE 25th International Conference on Field Programmable Logic and Applications (FPL),* London, 2015, pp.1-2.

[11] S. Lu, *et al.*, "Design and Implementation of an ASIC-based Sensor Device for WSN Applications," *IEEE Trans. on Consumer Electronics, vol.* 55, no. 4, pp. 1959-1967, Nov. 2009.

[12] C. Su and Y. Sie, "An FPGA implementation of chaotic and edge enhanced error diffusion," *IEEE Trans. on Consumer Electronics*, vol. 56, no. 3, pp. 1755-1762, Aug. 2010.

[13] G. Kiokes, *et al.*, "Design and implementation of an OFDM system for vehicular communication with FPGA technologies," in *IEEE 6th International Conference on Design and Technology of Integrated Systems in Nanoscale Era (DTIS),* Athens, Greece, April 6-8, 2011, pp. 1-6.

[14] C. Yao, *et al.*, "VLSI implementation of a real-time vision based lane departure warning systems," in *IEEE 12th International Conference on ITS Telecommunication,* Taipei, Taiwan, 2012, pp. 170-174.

[15] M. Nader and J. Liu. "FPGA design and implementation of demodulator/decoder module for eCall In-Vehicle System" in *IEEE, 2016 International Conference on Embedded Systems and Applications (ESA15),* Vegas, USA, Jul. 27-30, 2015, pp. 3-9.

[16] M. Nader and J. Liu. "Developing modulator and demodulator for the EU eCall in-vehicle system in FPGAs" in *IEEE, 2016 International Conference on Computing, Networking and Communications (ICNC),* Hawaii, USA, 2016, pp. 1-5.

[17] S.K.S. Chan and V.C.M Leung, "An FPGA Receiver for CPSK spread spectrum," *IEEE Trans. on Consumer Electronics,* vol. 45, no. 1, pp. 181-191, Aug. 2002.

[18] B. Tietche, O. Romain, and B. Denby, "Sparse channelizer for FPGA based simultaneous multichannel DRM30 receiver," *IEEE Trans. on Consumer Electronics*, vol. 61, no. 2, pp. 151-159, Jul. 2015.

[19] P. Juarez, *et al.*, "A DVB-H Receiver and Gateway Implementation on an FPGA and DSP based platform," *IEEE Trans. on Consumer Electronics, vol.* 57, no. 2, pp. 372-378, Jul. 2011.

[20] S. Sanchez-Solano, *et al.*, "FPGA implementation of embedded fuzzy controllers for robotic applications," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 19371945, Aug. 2007.

[21] Texas Instruments, "Introduction to the controller area network (CAN)," Application Report, SLOA101A, August 2002.

[22] Texas Instruments, "Using the I2S audio interface od DS90Ux92x FPD Link III Device," Application Report, SNLA221, June 2013.

[23] u-blox AG, "LEON-G100/G200 - Data Sheet" GSM.G1-HW-09001-B, 2009.

[24] u-blox AG, "Wireless modules, data and voice modules, AT commands manual" WLS-SW-11000-2, 2011.

[25] C. Cheng and K. K. Parhi, "High-Speed parallel CRC implementation based on unfolding, pipelining, and retiming," *IEEE Transactions on Circuits and Systems II: Express Briefs,vol.* 53, no. 10, pp. 1017-1021, 2006.

[26] S.M. Sait and W. Hasan, "Hardware design and VLSI implementation of a byte-wise CRC generator chip" in *IEEE Tran. on Consumer Electronics,* vol. 58, no. 1, pp. 195-200, Aug. 2002.

[27] Xilinx, "Nexys4 DDR FPGA Board Reference Manual," Document Number: 502-292. April 11, 2016.

[28] "eCall data transfer; in-band modem solution; ANSI-C reference code," 3GPP, Tech. Rep. TS26.268.

[29] Xilinx, "ML505/ML506/ML507 evaluation platform user guide" UG347 (v3.1.2), May 16, 2011.