Automatic Gain Control on FPGA for Software-Defined Radios

Khaled Sobaihi, Akram Hammoudeh, David Scammell
Faculty of Advanced Technology
University of Glamorgan, Pontypridd, UK
Email: {ksobaihi, amhammou, dscammel}@glam.ac.uk

Abstract—**This paper introduces an efficient implementation of the Automatic Gain Control (AGC) for an IEEE 802.15.3c compliant receiver developed using a Field-Programmable Gate Arrays (FPGAs), both feed-forward and feed-backward AGCs are designed, implemented and evaluated.**

Keywords—Automatic Gain Control (AGC); Feed-Forward AGC; Feed-Backward AGC; FPGAs; IEEE 802.15.3c; Software-Defined Radio.

I. INTRODUCTION

In wireless communication, the received signal has an unpredictable power level that varies over a wide dynamic range caused by multi-path fading and varying distances between the transmitter and receiver. To help mitigate this variation in the receiver power, an AGC loop is necessary to dynamically adjust the signal to minimize the quantization error and prevent the overflow/underflow of the analogue-to-digital converter and the digital back end. The Automatic Gain Control (AGC) is employed in both parts (analogue and digital parts). In this paper, only the digital AGC is analyzed and implemented [1].

With wireless devices becoming more and more sophisticated in terms of processing power, consumer demand for media rich applications is becoming the norm. To accommodate for these application requiring larger bandwidths new wireless standards are being implemented and explored to address the current spectrum shortfall. Future wireless devices will need to support multiple air interfaces and modulation formats that are outlined in these proposals. To accommodate for multiple standards, a flexible approach in design of wireless devices is becoming a key requirement for future hardware developments. The use of Software-Defined Radios (SDRs) has become a favoured approach due to the highly reconfigurable hardware platforms that provides the technology to realize many sophisticated signal processing tasks. SDRs offer the potential to revolutionize the way radios are designed, manufactured, deployed, and used. SDR promises to increase flexibility, extend hardware lifetime, lower costs, and reduce time to market [2].

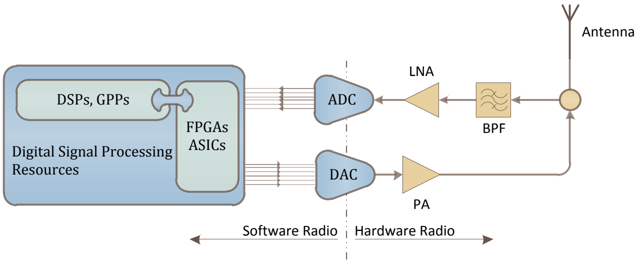

Fig. 1. Ideal Software-Defined Radio

Fig. 1 presents the ideal software radio architecture which consists of a digital system interfaced with the analogue front ends by means of ADCs and DACs. An ideal SDR has the majority of the design implemented digitally, with only the analogue aspect having the design elements that cannot be achieved through digital design. These elements being, the Power Amplifier (PA), the Low Noise Amplifier (LNA) and the antenna. The ideal placement of the ADCs/DACs should be as close as possible to the antenna for optimal performance of the SDR [3], [4].

A wideband ADC samples and digitizes the overall received signal spectrum. Depending upon the requirements of the system, in terms of speed and flexibility, the digital processing can be performed by a multitude of different devices; a General Purpose Processors (GPPs), Digital Signal Processors (DSPs), Field Programmable Gate Arrays (FPGAs), or Application Specific Integrated Circuit (ASICs).

FPGAs have become an attractive option to replace ASICs for prototyping and final product release. Nowadays FPGAs are not just a massive array of inter-connectable gates, but contain dedicated DSPs, RAMs and embedded processors to perform arithmetic operations that would be required for the third and the fourth generation wireless communication systems. FPGAs greatly relax the intrinsic constraints of off the shelf processor architectures by allowing the designer to customize a processing architecture to meet the demands of the application, while keeping the flexibility of processor's software. Using faster FPGAs along with a pipelined structure enable designers to produce high performance systems that can acquire and process a considerable amount of data in real time. This flexibility and performance coupled with high level system design tools utilizing an expanding array of Intellectual Property (IP) cores, performing complex DSP algorithms, allows for the implementation of SDRs on FPGA platforms [5], [6], [7].

978-1-4577-0580-9/12/$26.00 ©2012 IEEE

II. IEEE 802.15.3C PHY FRAME FORMAT

The IEEE 802.15.3c is the first wireless standard from the IEEE that addresses the 60GHz frequency (millimetre wave) band; the proposal is to achieve a multi-Gb/s Wireless Personal Area Network (WPAN). In this standard, Golay complementary sequences have been proposed for use in the preamble, to determine synchronization and channel estimation. Golay sequences have been widely used to detect a signal immersed in noise. A pair of Golay sequences [a_N, b_N] have an attractive property in that the addition of their a-periodic auto-correlation functions equals to zero for all nonzero time shifts, in other words, the sum of their auto-correlations has maximum peak and no side-lobes [8]. Let [a_N, b_N] be the pair of complementary Golay sequences of length N, *where* $N=2^M$ (M natural number) and [R_a, R_b] the auto-correlation of a_N and b_N respectively, the Golay sequences are defined by the following auto-correlation property:

$$\mathrm{R_{ab}}\,(i) = R_a(i) + R_b(i) = 2\mathrm{N}\delta(i - N) \tag{1}$$

Where δ(i) is the Kronecker delta function.

Fig. 2 shows the frame format of the IEEE 802.15.3c millimetre-wave standard. A PHY preamble shall be added at the head of the data frame to aid receiver algorithms related to Automatic Gain Control (AGC), timing acquisition, carrier recovery, frame synchronization and channel estimation. The data is mapped into sub-blocks separated by a Guard Interval (GI) [9].

The frame synchronization (SYNC) field is used for frame detection and uses 14 repetitions of a128 Golay code, the SYNC sequences is long and robust enough to perform the AGC, carrier and timing recovery. Start Frame Delimiter (SFD) is used to establish frame timing and header rate, and consists of [a128 -a128 a128 -a128] or [a128 a128 -a128 -a128] for high rate or medium rate respectively. The Channel Estimation (CES) field consists of [a256 b256 a256 b256 b128]. For its constant envelope and low side-lobes spectrum, the $\frac{\pi}{2}BPSK$ modulation is used to modulate to PHY preamble [9].

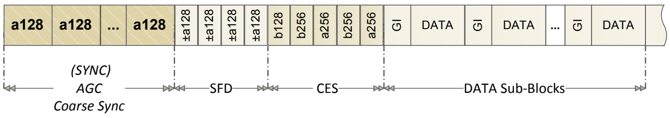

Fig. 2. IEEE 802.15.3c Frame Structure

III. DIGITAL AUTOMATIC GAIN CONTROL

Basically, AGCs can be implemented in two structures: feed-forward and feed-backward. In Fig. 3.a, the feed-forward AGC, shows the input voltage controls the output voltage by continually adjusting the output gain to keep a constant output. Feed-forward (i.e. open loop) AGC has the advantage to be intrinsically stable because it does not incorporate feedback.

In the feed-backward structure (i.e. closed loop), the estimated output voltage is compared with a reference to generate an error signal, this error is sent to adjust the input signal until the error is cancelled, Fig. 3.b. The loop filter guarantees the stability of the feedback loop, but precautions should be taken to avoid instability.

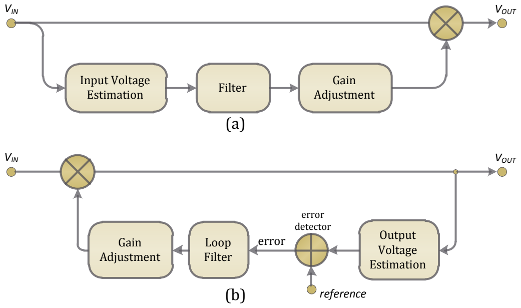

Fig. 3. Digital Feed-Forward (a) and Feed-Backward (b) AGC

IV. FPGA IMPLEMENTATION

For the rapid FPGA prototyping of the design, the Xilinx System Generator for Matlab Simulink is used. The design is captured using the DSP friendly Simulink modelling environment which uses a Xilinx specific blockset. All of the downstream FPGA implementation steps including synthesis and place and route are automatically performed to generate an FPGA programming file.

The design starts by fixed-point simulation and validation using Simulink as the modelling interface. Once the design is verified and validated, it is translated into efficient HDL code along with the integration of the IP Cores before synthesis and place and route.

Fig. 4 shows the proposed schematic of feed-forward AGC. To avoid the square-root and multiplication functions (those functions require huge FPGA resources), an approximation is used to estimate the magnitude of the input voltage, as follows:

$$V_{est} = \sqrt{I^2 + Q^2} \approx |I| + |Q| \tag{2}$$

The absolute value function can be realized with a multiplexer (MUX), where sign bit of the input switches the MUX between the two inputs –I and +I. The estimated voltage is then smoothed to reduce noise and glitches, the smooth function is realised with a moving average over 64 samples. The gain values are stored in a lookup table (i.e. implemented on the FPGA by a Read Only Memory ROM), replacing the division function by a ROM saves on the FPGA fabric, but requires more memory blocks.

The Dynamic Range (DR) is directly related to the number of bits used in the division B (size of the lookup table) as follows:

$$DR_{dB} = 10\log_{10} 2^{B-1} = 3(B - 1) \tag{3}$$

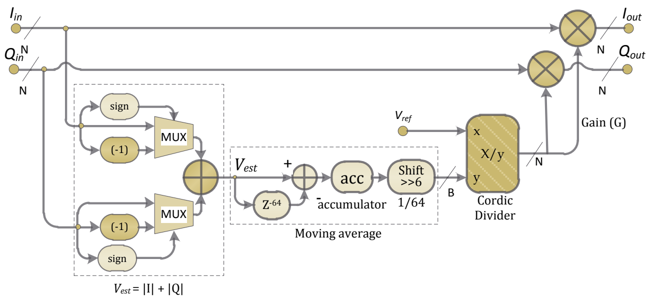

Fig. 4. Schematic Block Diagram of the Implemented Feed-Forward AGC

Fig. 5 illustrates the block diagram of the implemented feed-backward AGC; the output amplitude is calculated and smoothed with the same approach as the previous feed-forward AGC. The estimated amplitude is then compared with a reference to generate an error signal that will increment or decrement the gain accumulator with a factor of 2^{-K} (easily achieved using shift operations). When the output is locked to the reference, V_{ref}, the output fluctuates around the locked value by:

$$\frac{\Delta V}{V} = \frac{V_n - V_{n-1}}{V_n} \approx \left(\frac{1}{2}\right)^{K-1} \tag{4}$$

If B is the number of bits in the accumulator, the dynamic range of the AGC [10]:

$$\mathrm{DR_{dB}} = 10\log(2^{\mathrm{B}} - 2^{\mathrm{K}-1}) \approx 6(\mathrm{B} - \mathrm{K} + 1) \tag{5}$$

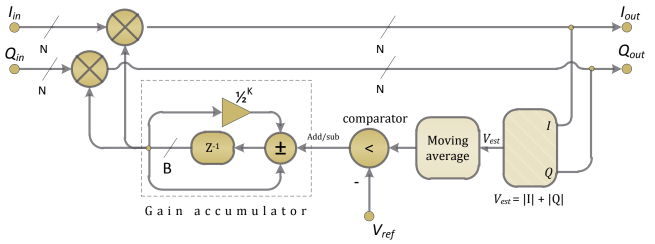

Fig. 5. Schematic Block Diagram of the Implemented Feed-Backward AGC

Table I shows the resources used to implement the feed-forward and feed-backward AGCs. The results show that replacing the division by a ROM memory, filled with pre-calculated values reduces the utilised FPGA resources by more than 50%. The feed-forward AGC's implementation requires less resource and is the more favourable option, due to the intrinsically stable feed-forward structure which eliminates the risk of the AGC becoming unstable causing the SDR to fail.

TABLE I
RESOURCES CONSUMED IN FEED-FORWARD AND FEED-BACKWARD AGCS ON VIRTEX-4 FPGA

Resources	**Feed-Forward**		**Feed-Backward**
	Using Division	***Using LUT***	
Flip Flops (FFs)	1,898	701	897
Lookup Tables (LUTs)	1,706	662	896
Logic Slices	1,062	379	515
DSP48s (Embedded Mults)	1	0	0
RAMB16s	0	64	0

V. HARDWARE TESTING

To evaluate the performance of the implemented AGCs, the Error Vector Magnitude (EVM) of the received constellation is calculated, this quantifies how accurate the received constellation matches the ideal locations. In these measurements the AGC module is activated only during the SYNC part of the preamble, defined by the IEEE 802.15.3c as shown in Fig. 2. The EVM is calculated as follows:

$$EVM_{RMS} = \left[\frac{\frac{1}{N}\sum_N[(I_{ideal} - I_{measure})^2 + (Q_{ideal} - Q_{measure})^2]}{\frac{1}{N}\sum_N(I_{ideal}^2 + Q_{ideal}^2)}\right]^{1/2} \tag{6}$$

Fig. 6 shows the measured EVM vs. input power in both AGCs schemes; 16 bits are used in the division and gain accumulator for both feed-forward and feed-backward. The measurements show that the two AGCs demonstrate similar dynamic range, around 45 dB.

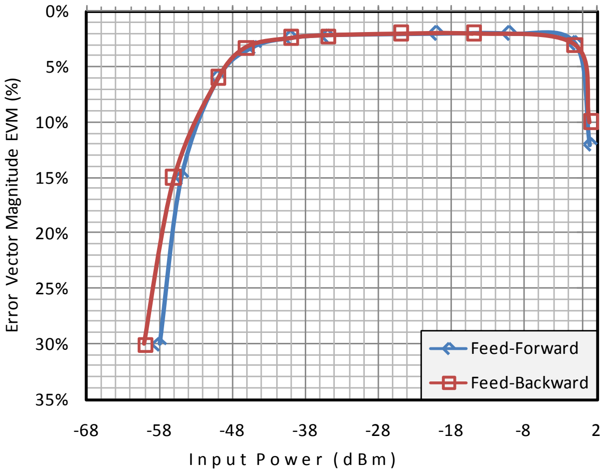

Fig. 6. EVM vs. Input Power in Feed-Forward and Feed-Backward AGC

VI. CONCLUSION

This paper proposed the design and the FPGA implementation of digital feed-forward and feed-backward AGCs, those AGCs are suitable for a software-defined radio receiver and compatible with the IEEE 802.15.3c millimetre-wave preamble. For the sake of stability, the feed-forward AGC was optimized and made even better resource savings than the feed-backward implementation with a similar dynamic range.

REFERENCES

[1] D. Qiang, J. Meilong, L. Guangquan, and S. Ning, “ALL-digital AGC in CDMA base station receiver”, International conference on communication technology proceedings, pp. 1037-1041, vol. 2, Apr. 2003.

[2] P. Burns, “Software defined radio for 3G”, Artech House, 2002.

[3] E. Buracchini, “The software radio concept”, IEEE Communications Magazine, pp. 138 – 143, Vol. 38, Sep. 2000.

[4] J. Mitola, “The software radio architecture”, object oriented approach, John Wiley & Sons, 2000.

[5] M. Cummings and S. Haruyama, “FPGA in the software radio” IEEE communications magazine, Vol. 37, No. 2, pp. 108-112, Feb. 1999.

[6] M. Devlin and D. Shand, “Scaling FPGA systems for software radio”, IEE colloquium on DSP enabled radio, pp. 1-7, Jan. 2011

[7] COTS Journal, The journal of military electronics & computing, Vol. 12, Num. 10, Oct. 2010.

[8] B.M. Popovic, “Efficient golay correlator”, Electronics letters, Vol. 35, pp. 1427-1428, Aug. 1999.

[9] IEEE 802.15.3c Standard - “Wireless medium access control (MAC) and physical layer (PHY) specifications for high rate wireless personal area networks (WPANs) amendment 2: millimetre-wave-based alternative physical layer extension”, IEEE Oct. 2009

[10] K. Chonghoon and I. Sungbin “Digital automatic gain control for software radio W-CDMA base stations” Electronics letters, Vol. 39, pp 318-320, Feb 2003.