DESIGN OF BASIC RECEIVING FUNCTIONS FOR AN SDR BASED QPSK BASE BAND DEMODULATOR FOR A RECONFIGURABLE DATA-LINK

Angelo Manco (a.manco@cira.it)
Ivan Iudice (i.iudice@cira.it)
Vittorio Ugo Castrillo (v.castrillo@cira.it)
Embedded Systems and Communications
CIRA – Italian Aerospace Research Centre, Capua, Caserta, Italy

ABSTRACT

An Unmanned Aerial Vehicle (UAV) needs "talk" to the ground control station (GCS) about mission and service information like the flight plan and telemetry. This paper focuses on the design and implementation of the base-band basic receiving functions, for a Quadrature Phase Shift-Keying demodulator case study, as independent modules of a complete Reconfigurable Data-Link (RDL) for UAVs communications. A model-based approach and Software Defined Radio (SDR) paradigm are used for the design. The implementation will be executed on Field-Programmable Gate Array (FPGA) based hardware.

1. INTRODUCTION

UAVs could operate in different scenarios that require multi-standard operations and environment link adaption, so flexibility and scalability are critical. These requirements can be addressed developing a Reconfigurable Data-Link (RDL), which is a system able to provide different communication functions without changing hardware. To design such a system, a model-based approach and SDR paradigm are investigated.
Model-based approach is helpful to obtain modular architectures with stand-alone blocks which can be used afterwards for other systems designs simply changing their interconnections. Therefore one of the main goals is the development of a library of blocks (each one with its own close hardware model) useful to implement a RDL.
Pushing the analog-to-digital conversion as close as possible to the antenna, SDR based systems allow to realize multiple radio features on the same hardware platform and to adapt the designed communications system to the radio context changes. In this panorama, Field-Programmable Gate Arrays (FPGAs) show good balance among computational power, configurability and design costs, so they are suitable to realize the advanced signal processing tasks necessary in the physical layer of a SDR based system.

On the basis of the above considerations, this paper focuses on the design of basic receiving functions as part of a complete RDL using a model-based approach. Since Phase Shifting-Keying modulations play an important role in the area of aeronautical communications, in particular in the satellite field, the receiving functions are investigated for a differential encoded QPSK demodulator case.
Digital signal processing can be executed at high or intermediate frequencies, but baseband processing is much more suitable by hardware performance and cost point of view; therefore in this work all operations are done on in-phase (I) and on quadrature (Q) components coming from a down conversion stage that is not described here. Synchronization is taken into account and feedback solutions are adopted. In particular, both timing and carrier recovery are implemented as described in sections 3 and 4.

2. BACKGROUND

A QPSK modulated signal is described by the following equation:

$$s_m(t) = \sqrt{\frac{2E_s}{T}} g(t) \cos\left(2\pi f_c t + \frac{\pi}{4}(2m-1)\right) = \sqrt{\frac{2E_s}{T}} g(t) \cos\left(\frac{\pi}{4}(2m-1)\right) \cos(2\pi f_c t) + \ldots - \sqrt{\frac{2E_s}{T}} g(t) \sin\left(\frac{\pi}{4}(2m-1)\right) \sin(2\pi f_c t) \quad (1)$$

where E_s is the symbol energy, T the symbol period, f_c the carrier frequency, $g(t)$ is the pulse shape and $m=1,2,3,4$ depends on the transmitted symbol.

An ideal SDR communication system has Analog-to-Digital Converter (ADC) pushed close to the antenna, so there is minimum or null signal conditioning prior to sampling. Today technologies allow us to have something very close to the ideal case only for not too high frequencies

and data-rates. However we consider the case where the down-conversion stage is in the analog domain and the output of this stage is the complex baseband signal represented through its I/Q components, neglecting the noise component:

$$I_m = \sqrt{E_s}\, g(t)\cos\left(\frac{\pi}{4}(2m-1)+\phi+\Delta f\right)$$
$$Q_m = \sqrt{E_s}\, g(t)\sin\left(\frac{\pi}{4}(2m-1)+\phi+\Delta f\right) \quad (2)$$

where ϕ, Δf are phase and frequency offset due to mismatch between carrier and local oscillator of the quadrature mixer in the down conversion stage. The I/Q signals are sampled by an ADC, with a sampling period T_s equal to T/N, where N is the number of samples for symbol, and then processed by the demodulator. A Raised Cosine $g(t)$ pulse shape filter mitigates the Inter Symbol Interference (ISI) over a band limited channel. A Decision Directed Phase Locked Loop (DD-PLL) is adopted to track the initial phase and the frequency drift. In order to resolve the phase ambiguity introduced by the DD-PLL, a differential encoding is adopted and so the data are mapped to the phase shifts of the modulated carrier.

thanks to a Decision-Directed Phase-Locked Loop. It is constituted by a Phase Error Detector (PED), a Loop Filter (LF), a NCO and a Rotational CORDIC, as depicted in Fig. 1.

The phase error θ_k, computed by the PED, is showed in the following equation

$$\theta_k = sign(I_k)\cdot Q_k - sign(Q_k)\cdot I_k \quad (3)$$

The PED works at the sampling rate $1/T_s$ and its output is accumulated after filtering operation. PI loop filter constants K_p and K_i can be calculated considering a certain dumping factor ζ and an equivalent noise bandwidth B_n expressed in terms of maximum frequency drift Δf_{max} loop can track, using the following formula [8]:

$$B_n \approx \frac{\Delta f_{max}}{2\pi\sqrt{2}\zeta} \quad (4)$$

Anyway tracking errors are proportional to the equivalent noise bandwidth, so the optimum choice for the right value of B_n has to be based on a trade-off between fast acquisition and good tracking.

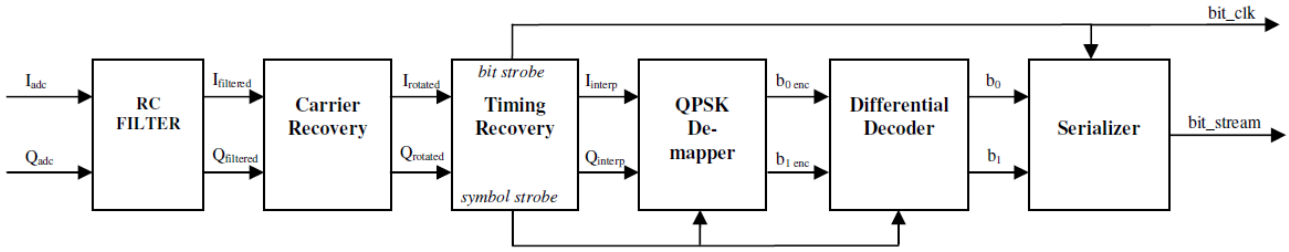

Fig. 1 – Functional architecture of the QPSK Demodulator

3. CARRIER RECOVERY

The phase and the frequency offset are acquired and tracked

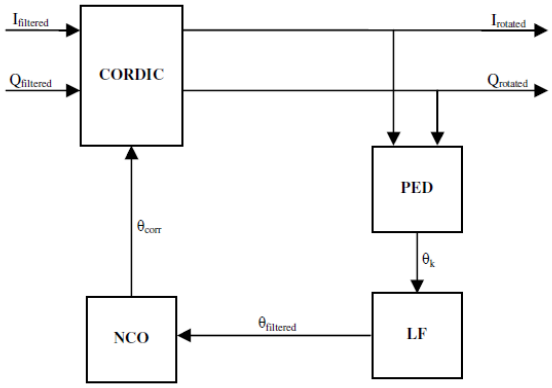

Fig. 2 – Carrier Recovery Unit

The filtered and accumulated error is used to adjust the phase and the frequency of the complex baseband signal by means of rotation of the vector with components I/Q. For this purpose, a CORDIC (COordinate Rotation DIgital Computer) algorithm in *rotational mode* is used [1]. It is an iterative procedure that only needs of adders, barrel shifters and a look-up table with off-line calculated values and, therefore, well suited for low cost FPGAs where dedicated multiplier resources are limited. Since the algorithm doesn't converge for input vectors having phase (in module) greater than 99°, a range extension is used [2].

4. BIT TIMING RECOVERY

The sampling period of analog-to-digital converter in the down-conversion stage is not aligned to the symbol period, i.e. it is asynchronous with symbols, and so timing errors affect demodulation performance in presence of noise. Real-world symbol pulse shapes have a peak in the centre of the symbol period. Sampling the symbol at this peak means to have the best signal-to-noise-ratio and it will mitigate interference from other symbols. There are possible several approaches to fix this issue depending on the characteristics of the receiver. We consider the case where the timing correction is all digital. This means that it is not possible to adjust the sampling frequency/phase of the ADC (it is fixed), passing from digital domain to analog one. A feedback scheme is adopted and in particular a Non-Data-Aided (NDA) timing recovery technique is used [3][5][6]. The structure is depicted in Fig. 3.

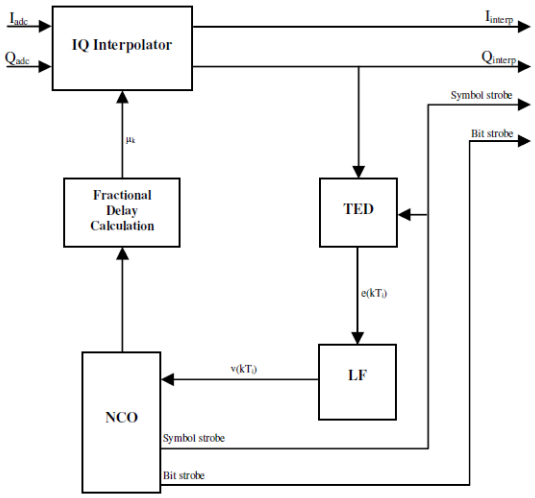

Fig. 3 – Bit Timing Recovery Unit

Let are {..., (k-1)T, kT, (k-1)T, ...} the desired interpolation instants and {..., (n-1)T_s, nT_s, (n-1)T_s, ...} the ADC sampling instants with n ≈ kN. The right sample for the k-th symbol is computed by means of an interpolation of ADC I/Q output samples on the basis of the fractional delay μ_k and on the last L samples available from ADC, where μ_k is the distance between the desired optimum sample (i.e. the sample at the time kT) and the closest ADC preceding sample (whose index, indicated with mk, is named base-point index) and L depends on the interpolation order. For example, the interpolated Q component sample is:

$$q(kT_i) = q((m_k + \mu_k)T_s) \qquad (5)$$

A Lagrange polynomial interpolation is considered. Since the Lagrange coefficients are expressed as a polynomial in μ, the interpolation is implemented as Farrow structure [8]. This approach allows the fractional delay to vary in a continuous way. A Timing Error Detector (TED) is used to estimate the error between the right sampling instant and the current one. In particular the Gardner algorithm is used. It is based on finding zero crossing between two consecutive symbols. It uses two samples per symbol and it generates the following error signal:

$$\hat{e}(kT_i) = \left(y(kT_i) - y((k-1)T_i)\right) \cdot y((k-\tfrac{1}{2})T_i) \qquad (6)$$

In this case the TED works at the sampling rate $1/T_s$ and so its output is decimated according to the base-point index m_k to extract the right error.

The TED output is filtered in order to have the control signal for the timing adjustment. A first order Proportional-plus-Integrator (PI) loop filter is used for this purpose. In particular it consists of two paths. The proportional path multiplies the error signal by the proportional gain K_p. It is able to track out a phase step error. An integral path multiplies the error signal by the integral gain K_i in order to track out a ramp phase error (i.e. a frequency error). Constants K_p and K_i can be calculated considering a certain dumping factor ζ and an equivalent noise bandwidth B_n using a procedure similar to the one of a PLL design as explained in [8]. Note that the gain of the linearized TED and the NCO gain must be known to make the above procedure.

The Interpolation Control provides the base-point index and the fractional delay on the basis of the filtered error signal $v(kT_i)$. It is performed by a Numerically Controlled Oscillator (NCO) [5]. In this case, the NCO is constituted by an accumulator, operating at the sampling rate $1/T_s$, that overflows every N samples. The overflow in the k-th period indicates the base-point m_k and a symbol strobe is generated. This signal gives the timing for the right decimation at the de-mapper and at the differential decoder that must work at symbol time. Another strobe signal at a rate of two times the symbol rate, that gives the timing for the serializer unit, is necessary. This signal, named bit strobe, must depend on the m_k and so it is derivate from it.

The loop filter output $v(kT_i)$ adjusts the amount by which the accumulator increments. The fractional delay is computed using the content η of the accumulator as showed in the following equation

$$\mu(m_k) \approx N \cdot \eta(m_k) \qquad (7)$$

The fractional delay is updated at symbol rate.

4. DEMODULATION

The demodulation is done by means of a de-mapper and of a differential decoder. Then a serialization process take place in order to have a serial output stream at a frequency of 2*symbol rate. The timing, as seen previously, is provided by the Interpolation Control Unit. De-mapper works at the symbol rate and it compares the input I/Q with thresholds (depending on the source code chosen, e.g. Gray code) and outputs two parallel bits. The DD-PLL introduces a phase ambiguity multiple of $\pi/2$ due to the rotational symmetry of the constellation. In order to resolve this issue, a differential encoding is adopted and the data are mapped to the phase shifts of the modulated carrier instead of the phase. Therefore a differential decoding, implemented through a look-up table, takes place. The last process is relative to the transition from a parallel data to serial one, whose timing is "slaved" to the previous units one.

6. DESIGN FLOW

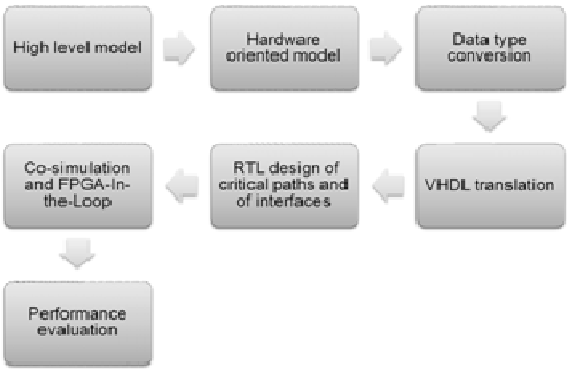

Fig. 4 – Design flow for QPSK Demodulator

QPSK base-band demodulator design is composed by the following steps. First of all the demodulator is high-level modelled and tested in MATLAB/SIMULINK environment. Required functions are implemented using standard maths functions (as the sine and cosine functions) and all signals are represented with a double data type. In the second step a close hardware model is designed. The general maths functions are calculated with algorithms whose hardware implementations are convenient (for example, vector rotation is performed by CORDIC algortihm). During the third step the data type is changed to fixed-point in order to obtain blocks models structure well suited to get a VHDL code. At this stage, comparing the simulation results of the double data type blocks models and the ones related to fixed-point models, it is possible to determinate the best choice for words and fractional parts lengths, choosing a good trade-off between accuracy and resources utilization. At the fourth step a VHDL code is designed for each block and for the overall system, taking into account the FPGA target. The fifth step is relative to simulation of demodulator VHDL code using SIMULINK and co-simulation tools; Next, an hardware test using signal generator and Bit Error Rate Tester (BERT). In a final step, performance tests will be executed to estimate the demodulator quality in presence of carrier frequency and timing errors, and in presence of noise tracing the BER vs E_b/N_0 curve.

7. SIMULATION

The overall system is simulated in MATLAB/SIMULINK environment. The aim of the simulation is to verify that the demodulator, in particular the synchronization functions, works properly. Timing and carrier errors are considered separately.

The coefficients of the loop filter for the timing recovery are designed considering an unitary damping factor and a single-sideband noise bandwidth Bn of 0.5% of the symbol rate. A PN11 bit sequence is generated through a Linear Feedback Shift Register and used, after modulation, as data input of the system. Whereas the sampling period of analog-to-digital converter in the down-conversion stage is not aligned to the symbol period, a step timing error is firstly considered. No carrier errors are introduced. Fig. 5 and Fig. 6 show the

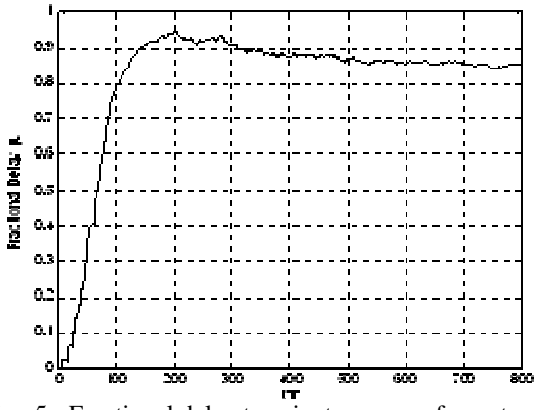

Fig. 5 - Fractional delay transient response for a step timing error

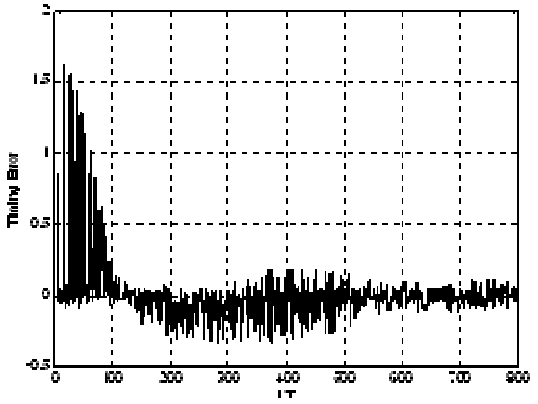

Fig. 6 – TED transient response for a step timing error

transient responses of the fractional delay μ_k and of the TED error signal. The fractional delay μ_k reaches a steady-state value of 0.85 (i.e. the target value) after about 500 symbols. Response to a timing error ramp of 1% of symbol period is shown in Fig. 7 and Fig. 8. The TED error signal goes to

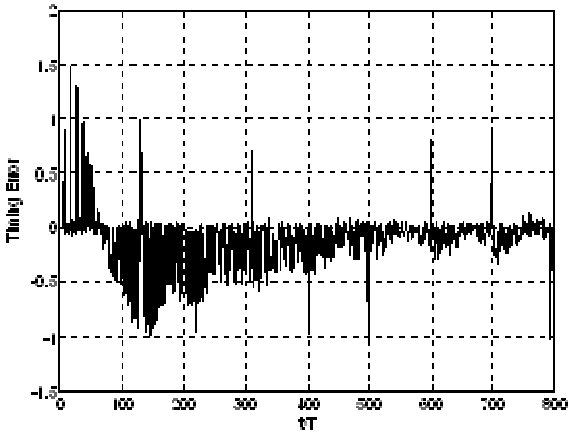

Fig. 7 - TED transient response for a ramp timing error

zero thanks to the loop filter that is capable to track out a frequency error. Because a residual timing error accumulates, the fractional delay μ_k decreases with time [8]. When the accumulated residual timing error exceeds a

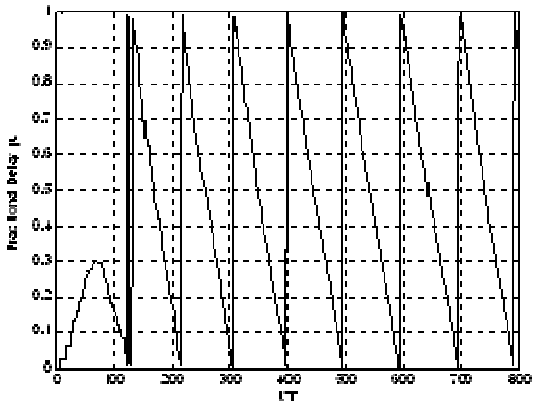

Fig. 8 - Fractional delay transient response for a step timing error

sample period, μ_k wraps around to 1. It happens every 100 symbols, accordingly to the introduced error.

The behaviour of the carrier recovery loop is also examined. The coefficients of the loop filter are designed considering a damping factor $\zeta = 1/\sqrt{2}$ and a single-sideband noise bandwidth B_n of 5% of the symbol rate. the phase error is $\pi/6$ and the frequency error is 0.1% of the symbol rate.

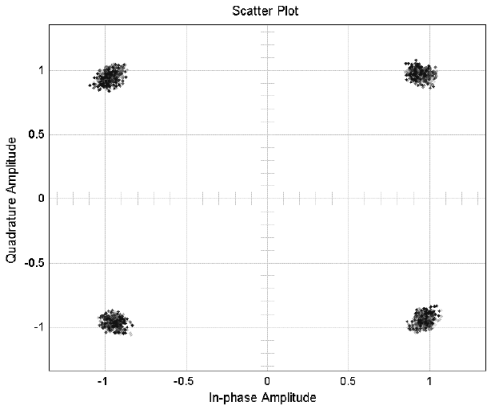

Fig. 9 – Corrected constellation

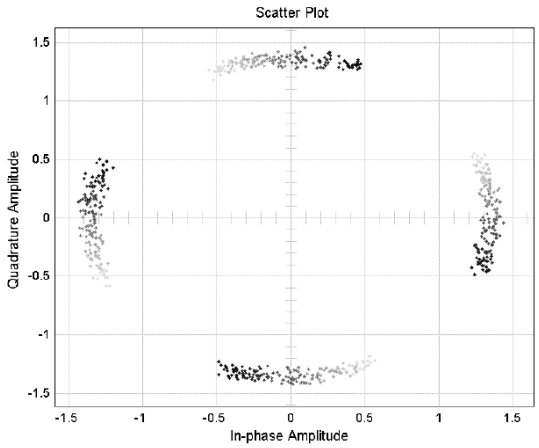

Fig. 10 – Frequency error implies constellation rotation

8. REFERENCES

In this paper, basic receiving functions of a communication system were investigated for a QPSK base-band demodulator case with the aim to create a library of modular blocks that can be used to implement a Reconfigurable Data-Link.

A model-based approach was used in the design flow. In such way the transition from a high level blocks models to a close hardware ones was simplified and the development time was considerably reduced. The obtained models were

tested with software simulation showing a proper operation. The VHDL code will be generated and implemented on low cost FPGA. The demodulator will be tested with laboratory instrumentation.

9. REFERENCES

[1] Volder Jack, "The CORDIC trigonometric computing technique", IRE Transactions on Electronic Computers, vol. 8, no. 3, September 1959, pp. 330-334

[2] Andraka Ray, "A survey of CORDIC algorithms for FPGA based computers", Proceedings of the ACM/SIGDA sixth international symposium on Field programmable gate arrays, Monterey, CA, February 22-24 1998, pp. 191-200

[3] Meyr Heinrich, Moeneclaey Marc, Fechtel Stefan A., "Digital Communication Receivers, Synchronization, Channel Estimation and Signal Processing", vol. 2, Wiley, NY, November 1997

[4] Navjot Singh, "Design and implementation of optimum interpolation filter using Farrow structure", International Journal of Engineering Science and technology, vol. 3, no. 5, May 2011, pp. 4108-4113

[5] Gardner Floyd, "Interpolation in Digital Modems – Part I: Fundamentals, IEEE Transactions on Communications, vol. 41, no. 3, March 1993, pp. 501 – 507

[6] Erup Lars, Gardner Floyd, Harris Robert, "Interpolation in Digital Modems – Part II: Implementation and Performances", IEEE Transactions on Communications, vol. 41, no. 6, June 1993, pp. 998 – 1008

[7] Harris Frederic, Rice Micheal, "Multirate Digital Filters for symbol Timing Synchronization in Software Defined Radios", IEEE Journal on selected areas in Communications, vol. 19, no. 12, December 2001, pp. 2346 – 2357

[8] Rice Michael, "Digital Communications: A Discrete-Time Approach", Pearson Prentice-Hall, Upper Saddle River, NJ, 2009