LATVIAN JOURNAL OF PHYSICS AND TECHNICAL SCIENCES
2013, N 1
DOI: 10.2478/lpts-2013-0006

SDPSK-SPECIFIC DEMODULATION METHOD FOR THE nanoRTU FIELD-PROGRAMMABLE GATE ARRAY

A. Skorodumovs

Engineering Research Institute
"Ventspils International Radio Astronomy Centre"
of Ventspils University College,
101 Inženieru Str. 101, Ventspils LV-3601, LATVIA

The software-defined radio (SDR) solutions inpart flexibility to the satellite applications when the devices are physically inaccessible after the launch. The nanoRTU FPGA-based controller (*AAC Microtec*) may be programmed to serve as a software-defined differential phase shift keying (SDPSK) modem backend to be used in satellites for communication with the Earth. The modem consists of two units – a modulator and a demodulator. A fully functional symmetric SDPSK modulator for nanoRTU FPGA has already been implemented. The next step of the modem implementation is the development of demodulator. In order to implement such facilities, the existing demodulation techniques should be reviewed in order to propose the appropriate method in which the demodulator would be capable of demodulating a signal, and, at the same time, would be resource-efficient. The author describes a valid method of specific SDPSK signal demodulation for the nanoRTU FPGA.

Key words: *SDR, SDPSK demodulation, nanoRTU, FPGA demodulator.*

1. INTRODUCTION

The purpose of this work was to develop a specific differential phase shift keying (SDPSK) demodulation method for the use in nanoRTU field-programmable gate array (FPGA). Based on this method, it is possible to find an FPGA resource saving solution that would successfully be used for demodulating the SDPSK-modulated signals [1]. This demodulation method would then be introduced into FPGA next to the SDPSK modulator [2] to form a nanoRTU-based software defined radio (SDR) SDPSK satellite modem. A major advantage of the nanoRTU is that it is a FPGA-based radiation tolerant and space-approved board developed for rapid cubesat deployment.

2. THEORY

Figure 1 illustrates the basic idea underlying the proposed SDPSK-specific demodulator: the data are encoded by the phase alteration rather than by the phase itself. The counter-clockwise phase movement would stand for an incoming 'one', while clockwise – for 'zero'. In other words – the data could be directly recovered from the sign of the phase derivative, provided that the sampling timing is correct (Fig. 1*e*). This also eliminates the need in phase offset compensation.

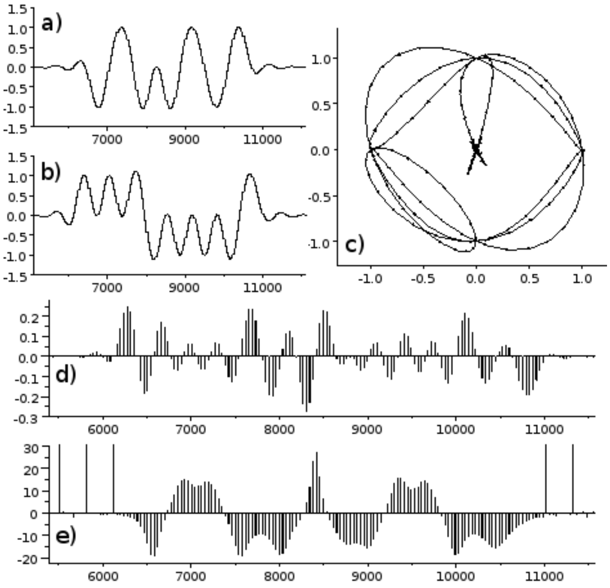

Fig. 1. SDPSK signal: *a*) I channel; *b*) Q channel; *c*) constellation; *d*) power derivative; *e*) phase derivative.

A certain amount of phase rotation resulting from Doppler's shift could also be neglected if the derivative is sampled precisely in the middle of a symbol. The phase rotation effectively shifts the derivative graph up or down by some degree, depending on the rotation direction and speed. The symbol timing may be restored from the derivative. When the phase derivative crosses zero the symbol value changes to the opposite, which means that the crossing signifies the end of the symbol. Knowing when symbols start and end, we can restore the symbol timing. Another important element of demodulation – the frame detection – can be done using the signal power. The start of a frame is accompanied by a rapid rise in the power level (Fig. 1*d*). A simple threshold cannot be used here, since the amplitude of a signal depends on the external amplifier. Scaling the sampled power with the averaged power calculated in some range around the derivation point would eliminate the input amplification and produce a pure ratio of the power values.

3. RESULTS AND DISCUSSION

3.1. General demodulator structure

The proposed SDPSK demodulator consists of several large blocks (Fig. 2):

1) ADC driver for nanoRTU onboard ADC chip control and data acquisition;
2) P&P block for conversion of I and Q data into a power and Δphase sample set;
3) filter A for filtering the power samples and calculating the average power;
4) filter B for filtering the Δphase and detecting the phase rotation direction changes;
5) pattern recognition for the frame start and end detection;
6) buffer to store samples to compensate the output delays of other blocks;
7) timer to generate and adjust the sampling timing signal;
8) storage and output driver for serial output of the demodulated data.

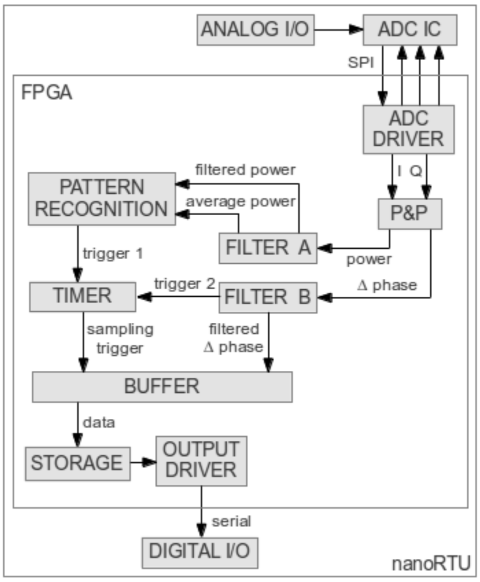

Fig. 2. Demodulator structure.

3.2. ADC driver

The ADC driver block controls nanoRTU onboard ADC IC via SPI interface. Since the nanoRTU has a single four-channel ADC, both I and Q cannot be sampled simultaneously. Hence, a delay between I and Q samples is unavoidable. The necessity of taking an additional sample also lowers the overall sampling rate. For optimal timing performance this block was hardcoded in a state-machine fashion for a particular ADC chip.

The results are: 115clk delay between channels, and 300clk delay between consecutive channel samples. With 16MHz clock sampling a 9600 symbol per second signal, this results in 5.56 samples per symbol and an interchannel delay of 6.9% per symbol.

3.3. P&P

The principle of P&P block operation is illustrated by Fig. 3. The block converts an I/Q sample set into the power and phase derivative. The power calculation is quite simple, and utilizes one multiplier and one adder, calculating I^2 and Q^2 consecutively and then adding them together. Alternatively, squaring can be done using a LUT (look-up table) and iterative search.

The algorithm of phase approximation is shown in Fig. 3*a*. This is basically an iterative approximation of the arctangent function for FPGA. The full 360°circle can be divided into four quadrants depending on the I and Q signs. Furthermore, the quadrants may be split into halves (octants), since |I| ≤ |Q| is a mirror of |I| ≥ |Q|. The idea behind the algorithm is to use a line equation Y = KX to determine whether the point (X, Y) is above or below the line (see Fig. 3*b*). In this way we can conduct a binary search until the required precision is achieved. The derivative of a phase is calculated by subtracting a newly acquired value from the previous one.

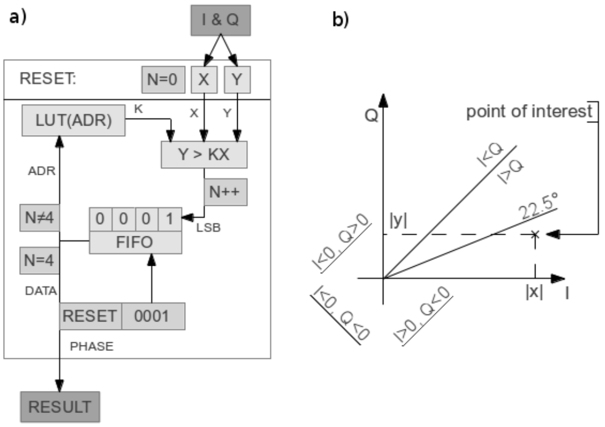

Fig. 3. a) arctangent approximation algorithm; *b*) the use of line equation.

In the reset state, FIFO (first-in–first-out) is loaded with an initial binary address 0b0001, so that in first iteration LUT(1) = tg(22.5°) is loaded into K. When comparison is done, the iteration counter is incremented and the result (1 if higher, 0 if lower or equal) is pushed into the address FIFO, providing new K for the next iteration. After the fourth iteration, FIFO would contain a binary representation of the approximated value. Then a true phase value is constructed using the octant number.

This simple four-iteration block allows a ±1.4° precision to be achieved. Practically, this design is limited by the K-values. In order to avoid the use of floating point variables, the LUT contains approximated integer values, for instance tg(2.8125°) ≈ 0.049 ≈ 13/256, so Y > KX will look like Y*256 > X*13, where Y*256 can be found by shifting Y value left by 8 bits. This approach however affects precision of the resulting value.

3.4. Filter A

Filter A block is used for filtering the power samples and calculating the average power of a signal. As shown in Fig. 4*a*, the input power sample is pushed both into FIFO 2 and FIFO 1. The values of the latter are used for calculating the average power within a range of three symbols around sample #16. FIFO 1 is used as a smoothing filter for power values. After the filtered power sample has been calculated, it is instantly pushed into FIFO 3. The single purpose of FIFO 3 is synchronisation of FIFO 1 and FIFO 2 outputs, since they have different latency. The output values of filter A are the smoothed power samples and the average power in a three-symbol range around it.

3.5 Filter B

Filter B (Fig. 4*b*) is used to smooth the phase derivative and detect changes in the phase movement direction. FIFO 1 produces averaged Δphase samples. The delay block and comparators generate trigger 2 signal when Δphase crosses zero.

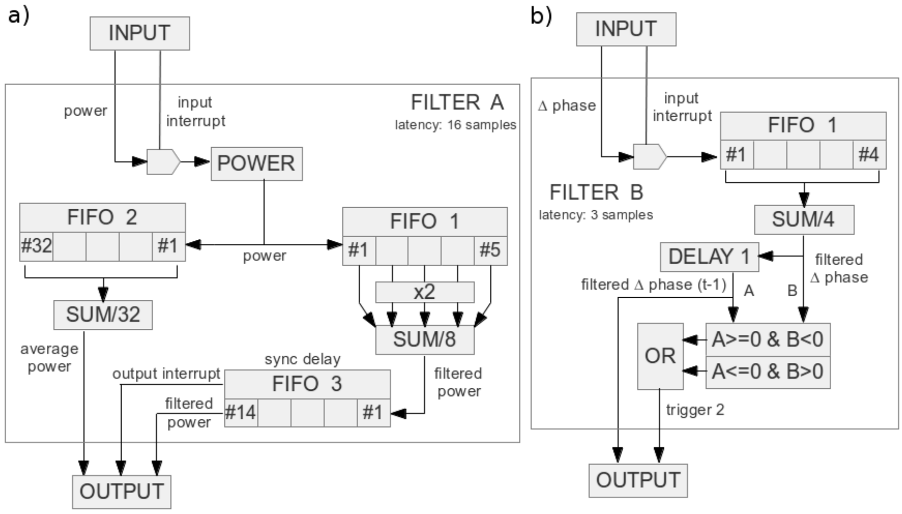

Fig. 4. Structure of: filter A (*a*); filter B (*b*).

3.6. Pattern recognition block

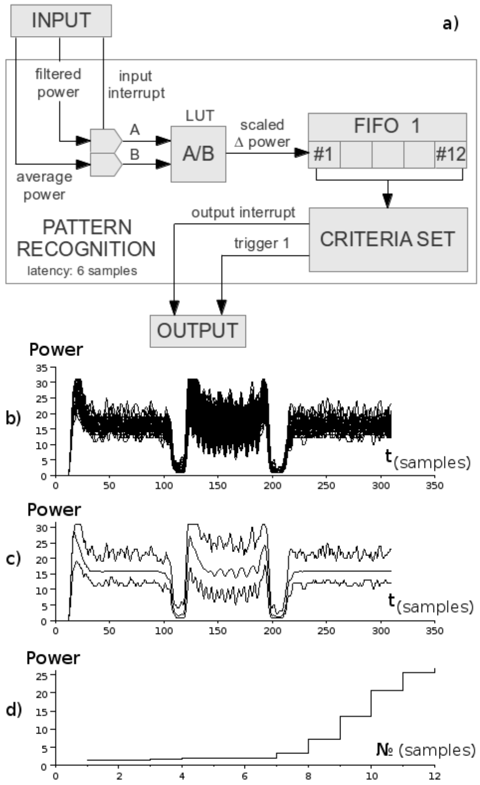

Fig. 5. *a*) pattern recognition block; *b*) scaled power for a simulation set;
c) averaged signal with minimum and maximum deviation; *d*) averaged pattern.

The pattern recognition block (Fig. 5*a*) provides detection of the frame start and end. As noted before, the incoming frame starts with a rapid power increase. However, the real received signal will be contaminated with Rician fading, so sudden power changes are possible after the frame start. Since the power pulse shape will vary greatly, a pattern recognition block is necessary to detect the signal front.

As the pulse is quite wide, a 12-sample (~2 symbols long) FIFO is used. These values are then accessed and checked using criteria based on a set of simulations. Figure 5*b* shows a scaled power for 200 signals, while Fig. 5*c* – an averaged signal with the maximum deviation. Figure 5*d* displays a pattern used for the frame recognition. Averaging the power input is necessary for scaling the input power sample. For finding the pattern within a signal there exist diversified methods and even more possible implementations (since the structure of our demodulator does not depend on such a pattern, no specific criterion set is proposed for its recognition).

As seen in the scheme of Fig. 4, the pattern recognition block sends the signal via trigger_1 to the timer block about the frame start or end.

3.7. Buffer

The buffer is a sample storage block triggered by timer. Since different parts of the system have differing latency, a sample buffer is required to compensate the delays.

The timer block can only start countdown for sampling when a signal from the pattern recognition block is received, which, in turn, receives samples from filter A. The filter B block, which provides buffer with samples, has a three-sample latency. The pattern recognition and filter A blocks have the total latency of 22 samples (with a 22–3+1 = 20 sample buffer block used).

When such a block is triggered, the sign (positive or negative) of the 18th, 19th and 20th sample sum is checked. The positive value means the counter-clockwise phase rotation and, therefore, stands for 1 in the SDPSK modulation scheme, while the negative – for 0, respectively.

3.8. Timer

The timer block shown in Fig. 6 is used to produce a timing signal necessary to sample symbols as close as possible to the centre of a symbol. Since the samples per symbol rate is 5.56 and may vary during the transmission, the timer countdown is driven by FPGA clock rather than by the incoming samples.

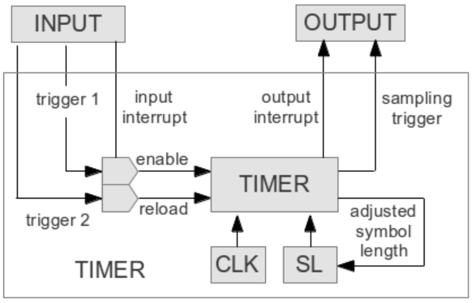

Fig. 6. Timer.

A necessary feature of the timer block is automatic adjustment of a symbol's length (SL). When the timer is enabled, SL is reset to a default value of 1667 – the length of a single symbol expressed in clk's at the data rate of 9600 symbols per second. While running, the timer collects data – how many samples were taken and how much time this took. After receiving the reload signal, a new symbol length is calculated. Previous SL values and boundaries are taken into account.

3.9. Storage and output driver

The storage block is a shift register collecting a byte from the demodulated bits. When a byte is received, the data are forwarded to the output driver. The output driver is an ordinary UART-like transmitter hardcoded to 8N1, 19200 baud settings.

4. CONCLUSIONS

The proposed specific SDPSK signal demodulation method for nanoRTU allows for SDPSK modulated data reception without phase and frequency recovery.

The FPGA-friendly algorithms have been proved to minimize the use of its resources while maintaining the optimal performance and a decent precision of calculations.

To detect the frame start and end patterns, Scilab link simulation has successfully been used to produce a series of modulated signals contaminated with Rician fading, with a random phase offset and a slight phase drift.

ACKNOWLEDGEMENTS

The research was supported by the European Regional Development Fund within the project "Development of the Software defined radio (SDR) satellite communication model" (No. 2010/ 0266/ 2DP/ 2.1.1.1.0/ 10/ APIA /VIAA/ 117).

The hardware and documentation were provided by AAC Microtec staff, namely Fredrick Bruhn and Jan Schulte.

REFERENCES

1. Prösch, R., & Daskalaki-Prösch, A. (2011). *Technical Handbook for Radio Monitoring VHF/UHF*. Books on Demand GmbH, pp. 245–246.
2. Skorodumovs, A. (2012). *SDPSK Modulator for Radiation Tolerant Satellite Modem.* Latv. J. Phys. Techn. Sci., 6 (II), 59–66.

SDPSK-SPECIFISKA DEMODULĀCIJAS METODE nanoRTU^{TM} PROGRAMMĒTAJĀM LOĢISKAJĀM INTEGRĀCIJAS SHĒMĀM

A. Skorodumovs

Kopsavilkums

Rakstā piedāvāts SDPSK modulēta signāla apstrādes veids, kas ir optimizēts izmantošanai FPGA, kā arī demodulatora realizācija ÅAC nanoRTU radiācijas noturīgajai aparatūras platformai. Galvenais darba mērķis ir maksimāli efektīva aparatūras resursu izmantošana.