Development of Algorithm, Architecture and FPGA Implementation of Demodulator for Processing Satellite Data Communication

K. R. Nataraj *, Dr S. Ramachandran and Dr B. S. Nagabushan *****

* M. G..R. University, Chennai, India
** National Academy of Excellence, Bangalore, 560060, India
*** Sanlab Technologies, Bangalore, 560038, India

Summary
This paper proposes a novel VLSI architecture for the demodulator for processing satellite data communication. The overall receiver algorithm is divided into two parts: one to be implemented on an FPGA and the other on a DSP processor. A new distributed arithmetic based architecture for implementing a Sampling Rate Converter is also proposed. The main advantage of this architecture is that it does not employ any MAC unit, whose operational speed is, generally, a bottleneck for high filter throughput. Instead, it makes extensive use of LUTs and hence is ideally suited for FPGA implementation. Architecture for Digital Frequency Synthesizer, which gives 60 dB spectral purity, is also presented. The developed FPGA core consists of a mixer and two numbers of 193 tap, RRC filters to accept modulated, 12-bit, signed ADC output at a sampling frequency of 1.536 MHz and convert it into In-phase (I) and Quadrature-phase (Q) channel outputs, each of size 16 bits, signed, at half the sampling frequency. The main design goals in this work were to maintain low system complexity and reduce power consumption and chip area requirements. These architectures were coded in Verilog HDL and implemented on Xilinx FPGA. The design was synthesized with XCV600-4 FPGA and occupies about 2360 slices with an equivalent gate count of about 45000 and operating at a maximum frequency of 19.8 MHz. The entire modulator and demodulator have been coded in Matlab in order to validate the hardware results. The hardware and MATLAB results compare favorably.

Key words:
Algorithm, Demodulator, Linear algebra, Distributed Arithmetic Architecture, Sampling Rate Converter, Digital Frequency Synthesizer, Field Programmable Gate Arrays.

1. Introduction

World demand for communication facilities carrying many different types of real-time and non-real-time signals such as voice, data, facsimile, and video has been growing by leaps and bounds during the past few decades. The increasing demand and the resulting large amount of world-wide communication traffic naturally calls for links with very large transmission bandwidth.

Manuscript received July 5, 2009
Manuscript revised July 20, 2009

A number of demodulator algorithms for data communication have been reported by researchers [1-12]. Digital Frequency Synthesizer (DFS) Algorithm and Architecture developed are available in the literature [13-15]. FPGA implementations of some of these architectures were also reported [16-18]. A two stage estimation scheme for demodulator for processing satellite data was proposed in our earlier work [19], where carrier frequency estimation was followed by timing recovery under training. Therein the receiver algorithm was partitioned into two parts, one to be implemented on FPGA and the other on DSP. An overview of the whole system architecture was also presented and its performance was evaluated. In this paper, we review the basic theory of distributed arithmetic and its modified versions to achieve a trade off between chip area and throughput. Then we present new architectures for sampling rate converter and digital frequency synthesizer, which results in the use of reduced memory. However, the DSP implementation of the remaining part, namely, frequency and timing offset, carrier recovery and LMS are out of scope of the present work.

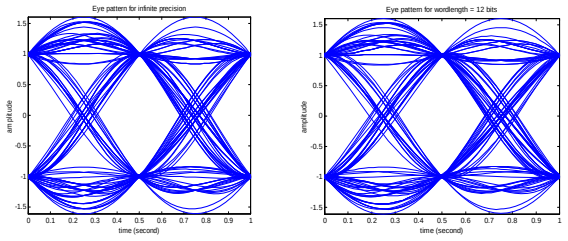
The rest of this paper is organized as follows: In the next section, sampling rate converter theory and optimization techniques for ROM are presented. This is followed by the detailed architecture of sampling rate converter. The DFS architecture is developed in Section 3. Section 4 presents the implementation of mixer and RRC filters using Verilog targeted on Xilinx FPGA. So also the results. Conclusions are presented in the last section.

2. Implementation of Sampling Rate Converter

2.1 Word-length issues

Eye patterns are often employed in the qualitative evaluation of receiver performance. These patterns may be

obtained using the BPSK system. The output of the matched filter in the receiver is fed to the vertical input of an oscilloscope and the symbol clock is fed to the external trigger of the oscilloscope. The transmitted digital signal is recovered by sampling the received analog signal and then making a threshold decision. In the optimal case, the decision point or sampling point is the point where the eye is most open. While implementing the filter in a fixed-point platform such as an FPGA, the effect of finite word length on the filter performance needs to be considered. Excessive word length increases the hardware cost and reduces the speed, whereas smaller word length reduces the precision of filter coefficients. For the present case of an RRC filter, we need to select the word length of the filter coefficients such that we get minimum inter-symbol interference at the output of matched filter at the receiver. The effects of the truncation of filter coefficients can be illustrated by way of an example. In this example there is no noise in the channel and a roll-off factor of 0.4 is used. The sample point is at t = 0.5. Fig. 1 a shows the eye pattern obtained using MATLAB, for infinite precision of RRC filter coefficients. It may be noted that at the sampling point, the ISI is negligible. We also plot the eye-patterns for filter coefficient word lengths of 12 bits. For 8 bits and 10 bits, there exists a large amount of ISI at the sampling points. This is due to the presence of large quantization noise, which manifests itself in the form of ISI, in the present case. Also, due to the finite



a b

Figure 1 Effect of filter coefficient word length on eye-pattern
a Eye-Pattern for infinite precision
b Eye-Pattern for 12-bit precision

word length of filter coefficients, the attenuation of the filter response in the stop-band reduces. As the word length for filter coefficients is increased, the ISI at the sampling points reduces. It may be noted that as the number of bits is increased from 12 to 16 bits, the ISI at the sampling points remains approximately constant, close to that due to infinite precision in Fig.1 a. Thus, no major gain is evident in increasing the word length beyond 12 bits. Therefore, we design our filter with the coefficient word length of 12 bits.

2.2 Distributed Arithmetic (DA)

Distributed Arithmetic is used to design bit-level architectures for vector to vector multiplications. In distributed arithmetic, each word in the vector is represented as a binary number; the multiplications are reordered and mixed, such that the arithmetic becomes "distributed" throughout the structure [11, 12]. Distributed arithmetic is commonly used for implementation of convolution operations and discrete cosine transform (DCT).

2.2.1 Convolutional Distributed Arithmetic

Let us consider the inner product of two length-*N* vectors *C* and *X*:

$$Y = \sum_{i=0}^{N-1} c_i x_i \qquad (1)$$

where,

$\{c_i\}$*'s are* $M-$*bit constants,* No

$\{x_i\}$*'s are coded as* $W-$*bit 2's complement numbers.*

w, x_i can be written as

$$x_i = -x_{i,W-1} + \sum_{j=1}^{W-1} x_{i,W-1-j} 2^{-j} \qquad (2)$$

Substituting (2) in (1), we get

$$Y = \sum_{j=0}^{W-1} C_{W-1-j} 2^{-j} \qquad (3)$$

Therefore, by interchanging the summing order of i and j, the initial multiplications in equation (1) are *distributed* to another computation pattern.

Since the term C_j depends upon $x_{i,j}$, which has only 2^N possible values, it is possible to pre-compute them and store them in a read only memory (ROM). An input set of N bits ($x_{0,j}, x_{1,j}, \ldots, x_{N-1,j}$) is used as the address to retrieve the corresponding C_j values. These intermediate results are accumulated in W clock cycles to produce one Y value. This leads to multiplier-free realization of vector multiplication. Table 1 shows the contents of the ROM for N = 4. Fig. 2 shows a typical architecture for the computation of the inner product of two length-N vectors. The shift accumulator is a bit-parallel carry propagate adder that adds the ROM contents to the previously accumulated result. The inverter and the MUX are used for inverting the output of the ROM in order to compute C_{W-1}. The control signal S is 1 when

$j = W-1$ and, 0 otherwise. The computation runs from j = 0 to $j = W-1$ and the result is available in bit parallel form after W clock cycles. This approach corresponds to bit-serial distributed arithmetic.

Table 1 Contents of ROM for N = 4

$x_{0,j}$	$x_{1,j}$	$x_{2,j}$	$x_{3,j}$	*Contents of ROM*
0	0	0	0	0
0	0	0	1	c_3
0	0	1	0	c_2
0	0	1	1	$c_2 + c_3$
0	1	0	0	c_1
0	1	0	1	$c_1 + c_3$
0	1	1	0	$c_1 + c_2$
0	1	1	1	$c_1 + c_2 + c_3$
1	0	0	0	c_0
1	0	0	1	$c_0 + c_3$
1	0	1	0	$c_0 + c_2$
1	0	1	1	$c_0 + c_2 + c_3$
1	1	0	0	$c_0 + c_1$
1	1	0	1	$c_0 + c_1 + c_3$
1	1	1	0	$c_0 + c_1 + c_2$
1	1	1	1	$c_0 + c_1 + c_2 + c_3$

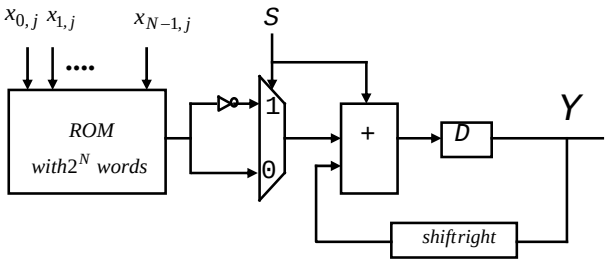


Figure 2 Architecture for computing inner product of two length - N vectors using Distributed Arithmetic

2.2.2 Distributed Arithmetic with offset-binary coding (DA-OBC)

In this section, the offset-binary coding (OBC) is introduced, which can reduce the ROM size by a factor of 2, i.e., down to 2^{N-1}.

Equation (2) can be rewritten as

$$x_i = \frac{1}{2}[x_i - (-x_i)]$$

$$= \frac{1}{2}[-(x_{i,W-1} - \overline{x_{i,W-1}}) + \sum_{j=1}^{W-1}(x_{i,W-1-j} - \overline{x_{i,W-1-j}})2^{-j} - 2^{-(W-1)}] \quad (4)$$

where

$$-x_i = -\overline{x_{i,W-1}} + \sum_{j=1}^{W-1}\overline{x_{i,W-1-j}}\,2^{-j} + 2^{-(W-1)}$$

Defining

$$d_{i,j} = \begin{cases} x_{i,j} - \overline{x_{i,j}}, & for\ j \neq W-1 \\ -(x_{i,W-1} - \overline{x_{i,W-1}}), & for\ j = W-1 \end{cases}$$

and $d_{i,j} \in \{-1, 1\}$.

Equation (4) can be rewritten as

$$x_i = \frac{1}{2}[\sum_{j=0}^{W-1} d_{i,W-1-j}\,2^{-j} - 2^{-(W-1)}] \quad (5)$$

Using (5), (1) can be rewritten as

$$= \sum_{j=0}^{W-1}(\sum_{i=0}^{N-1}\frac{1}{2}c_i d_{i,W-1-j})2^{-j} - (\frac{1}{2}\sum_{i=0}^{N-1}c_i)2^{-(W-1)}$$

Now defining

$$D_j = \sum_{i=0}^{N-1}\frac{1}{2}c_i d_{i,j}, \quad for\ 0 \le j \le W-1$$

and $$D_{extra} = -\frac{1}{2}\sum_{i=0}^{N-1}c_i$$

we have,

$$Y = \sum_{j=0}^{W-1} D_{W-1-j}\,2^{-j} + D_{extra}\,2^{-(W-1)} \quad (6)$$

Equations (4) to (6) characterize the OBC scheme.

It was observed that the contents of the ROM are mirrored across the line between the eighth and the ninth rows in the ROM table. Therefore, it is possible to reduce the ROM size by a factor of two. Table 2 illustrates the new ROM table.

Table 2 Contents of ROM with DA – OBC Coding (N = 4)

$x_{1,j}$	$x_{2,j}$	$x_{3,j}$	*Contents of ROM*
0	0	0	$-(c_0 + c_1 + c_2 + c_3)/2$
0	0	1	$-(c_0 + c_1 + c_2 - c_3)/2$
0	1	0	$-(c_0 + c_1 - c_2 + c_3)/2$
0	1	1	$-(c_0 + c_1 - c_2 - c_3)/2$
1	0	0	$-(c_0 - c_1 + c_2 + c_3)/2$
1	0	1	$-(c_0 - c_1 + c_2 - c_3)/2$
1	1	0	$-(c_0 - c_1 - c_2 + c_3)/2$
1	1	1	$-(c_0 - c_1 - c_2 - c_3)/2$

2.3 Modified DA-OBC Architecture

It can be observed from Table 2 that the ROM values except c_o term are mirrored along the line between the 4th and the 5th rows. Therefore, the ROM table of OBC scheme can be further reduced by a factor of two [16]. Table 3 illustrates the new ROM table and Fig. 3 shows the architecture for the computation of inner product using this method. By repeated application of this method, the ROM size can be reduced up to 2 words.

To achieve the size reduction of the whole system, the reduction in the number of ROM cells and the decoder circuit inside the ROM should be larger than the hardware increase for control circuits.

Table 3 Modified DA-OBC ROM Contents (N = 4)

$x_{1,j}$	$x_{2,j}$	$x_{3,j}$	*Contents of ROM*
0	0	0	$-(c_0 + c_1 + c_2 + c_3)/2$
0	0	1	$-(c_0 + c_1 + c_2 - c_3)/2$
0	1	0	$-(c_0 + c_1 - c_2 + c_3)/2$
0	1	1	$-(c_0 + c_1 - c_2 - c_3)/2$

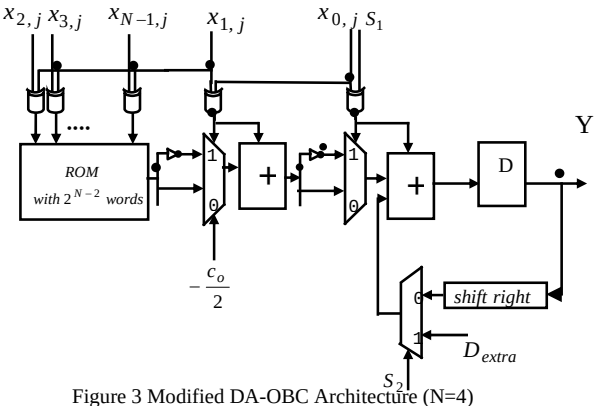


Figure 3 Modified DA-OBC Architecture (N=4)

2.4 ROM Decomposition for Distributed Arithmetic

The ROM size of the conventional distributed arithmetic increases exponentially with N. Generally, ROM access time can be a bottleneck for the speed of the whole system, especially when the ROM size is large. Therefore, reducing the ROM size is very important and is of great practical concern. Exploiting the linearity of equation (3), one possible solution to this problem is to divide the N address bits of the ROM into N/K groups of K bits, i.e., to implement the ROM of size 2^N with N/K ROMs of size 2^K and add the outputs of these ROMs using a multi-input accumulator. Fig. 4 illustrates the architecture for computing an N-input inner product using conventional distributed arithmetic with ROM decomposition. The total size of storage is now reduced from 2^N to $(N/K)2^K$ which increases linearly with N. The ROM access time is also reduced along with the ROM size. This reduction of the storage size is balanced by a linear increase of the computational complexity of the accumulator.

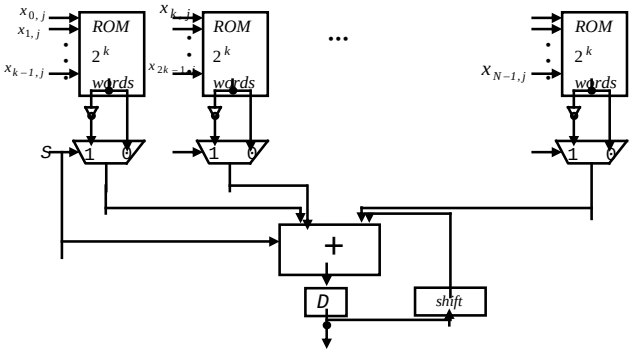


Figure 4 Decomposing 2^N sized ROM into N/K ROMs of size 2^K

2.5 Sampling Rate Converter Architecture

Having reviewed various distributed arithmetic based techniques for vector inner product implementation, we will now discuss the hardware architecture of sampling rate converter, which makes use of these techniques extensively. The input to the sampling rate converter is the output of the mixer, which then gets multiplied by $\cos(n\pi/2)$ and $\sin(n\pi/2)$ to bring the signal to baseband [19]. Then the image signals are removed by the following LPF, which in the present case is the RRC filter. After that, the signal is down-sampled by a factor of two, so as to make it suitable for carrier recovery.

Mathematically, the output of the RRC, in the I-channel, can be written as

$$I(n) = h_{LP}(n) \otimes \left[y(n) \cdot \cos(\frac{\pi}{2} n) \right]$$

$$= \sum_{m=0}^{N-1} h_{LP}(m) \cdot y(n-m) \cdot \cos(\frac{\pi}{2}(n-m))$$

Decimating by 2,

$$I(2n) = \sum_{m=0}^{N-1} h_{LP}(m).y(2n-m).\cos(\frac{\pi}{2}(2n-m))$$

Substituting m by $2k$, we get

$$I(2n) = \sum_{k=0}^{K_1} h_{LP}(2k).y(2n-2k).\cos(\frac{\pi}{2}(2n-2k))$$
$$\text{where } K_1 = (N-1)/2, \text{ if } N \text{ is odd}$$
$$\text{and } K_1 = (N/2)-1, \text{ if } N \text{ is even}$$
$$= \sum_{k=0}^{K_1} h_{LP}(2k).y(2(n-k)).\cos(\pi(n-k))$$
$$= \sum_{k=0}^{K_1} h_{LP}(2k).y(2(n-k)).(-1)^{(n-k)}$$
$$= h_{LPI}(n) \otimes [(-1)^n y(2n)], \text{ where } h_{LPI} = h_{LP}(2n),$$
$$\text{for } n = 0,1,2,3,4....K_1 \quad (7)$$

Effectively, the input data is decimated by two and sign changes are applied to alternate remaining samples. The resulting data stream is filtered by a new low-pass filter. The impulse response of the new in-phase low-pass filter $h_{LPI}(n)$ is given by

$$h_{LPI}(n) = h_{LP}(2n) \quad \text{for } n = 0,1,2,\ldots, K_I \quad (8)$$

It may be noted that the new filter processing speed is half of the input data rate. Similarly,

$$Q(n) = h_{LP}(n) \otimes [y(n).-\sin(\frac{\pi}{2}n)]$$
$$= \sum_{m=0}^{N-1} h_{LP}(m).y(n-m).\sin(-\frac{\pi}{2}(n-m))$$

and

$$Q(2n) = h_{LPQ}(n) \otimes [(-1)^n y(2n-1)],$$

where $h_{LPQ} = h_{LP}(2n+1)$.

Effectively, the input data is decimated by two and sign changes are applied to alternate remaining samples. It may be noted that there is one sample relative delay between the in-phase and quadrature channels. The resulting data stream is filtered by a new low-pass filter. The impulse response of the new quadrature low-pass filter $h_{LPQ}(n)$ is given by

$$h_{LPQ}(n) = h_{LP}(2n+1) \quad \text{for } n = 0,1,2,\ldots, K_Q \quad (9)$$

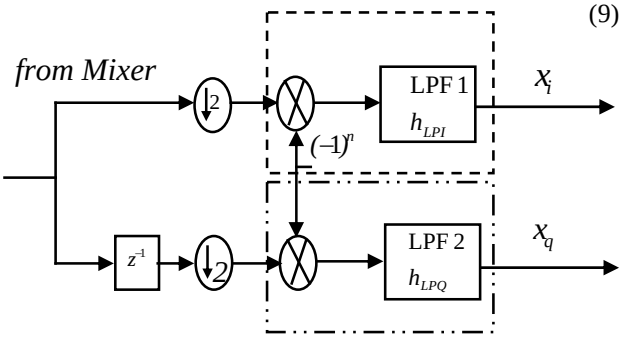


Figure 5 Combined Simplified Structure

Again, the processing rate is half of the input data rate. The overall digital quadrature demodulator is shown in Fig. 5.

It may be noted that all samples are not passed to both digital filters. The "even" samples are passed to the in-phase filter with every other sample undergoing a sign change. Similarly, the "odd" samples are passed to the quadrature filter, and again, every other sample also undergoes a sign change. The two blocks are very similar from the hardware architectural point of view. Therefore, we will discuss the hardware architecture of only one of these blocks in the next section.

2.5.1 Detailed design description of I-channel Block

Substituting values for n in equation (7), we get

$I(0) = h(0)y(0)$

$I(2) = -h(0)y(2) + h(2)y(0)$

.

.

$$I(200) = \sum_{k=0}^{96} h(2k)\, y(200-2k)\, (-1)^{100-k}$$
$$= h(0)y(200) - h(2)y(198) + h(4)y(196) - \ldots + h(96)y(104) - \ldots + h(188)y(12) - h(190)y(10) + h(192)y(8)$$

$$= h(0)[y(8) + y(200)] - h(2)[y(10) + y(198)] + h(4)[y(12) + y(196)] - \ldots + h(96)y(104)$$

$$I(202) = -h(0)[y(10) + y(202)] + h(2)[y(12) + y(200)] - h(4)[y(14) + y(198)] + \ldots - h(96)y(106)$$

$$I(204) = h(0)[y(12) + y(204)] - h(2)[y(14) + y(202)] + h(4)[y(16) + y(200)] - \ldots + h(96)y(108)$$

and so on.

It can be seen that the sign of all the terms containing coefficients *h(0), h(4), h(8), … , h(96)* is the same. Similarly, all the terms containing coefficients *h(2), h(6), h(10), … , h(94)* bear the same sign. Again, the signs get inverted in each succeeding even value of *I(2n)*. Also, due to the symmetry of the FIR filter, the number of filter coefficients reduces to 49 from 97, effectively. However, implementing these 49 coefficients with Distributed arithmetic will take a huge amount of memory (2^{49} words). Thus, on breaking the ROM into smaller parts, using improved form of modified DA-OBC and taking into consideration the symmetry of coefficients and the

inversion of their sign in every succeeding value of *I(2n)*, we get the I – channel filter architecture as shown in Fig. 6. An extra signal *sgn_pos* is introduced to take care of sign inversion. A high on *sgn_pos* gives a positive sign for the terms containing the coefficients *h(0), h(4), h(8), … ,h(96)*, whereas a low on this signal produces a negative sign for terms containing coefficients *h(2), h(6), h(10), … ,h(94)*. These terms are then appropriately mixed to produce values of *I(2n)*.

The coefficients of the smaller modules are as follows:

A_1 : *h(0), h(4), h(8), h(12), h(16), h(20)*
A_2 : *h(2), h(6), h(10), h(14), h(18), h(22)*
A_3 : *h(24), h(28), h(32), h(36), h(40), h(44)*
A_4 : *h(26), h(30), h(34), h(38), h(42), h(46)*
A_5 : *h(48), h(52), h(56), h(60), h(64), h(68)*
A_6 : *h(50), h(54), h(58), h(62), h(66), h(70)*
A_7 : *h(72), h(76), h(80), h(84), h(88), h(92)*
A_8 : *h(74), h(78), h(82), h(86), h(90), h(94)*

Thus, the total ROM size needed reduces to $8*2^4 + 1$ words instead of 2^{49} words, as in the case of the simple DA.

2.6 Results

To verify the arithmetic based architectural design, we compare the impulse response of the design implemented in VERILOG with that coded in MATLAB. A random input sequence is fed to the functional block and results of MATLAB and VERILOG implementations are presented in Fig. 7a and b for comparison. It can be seen that the two outputs closely follow each other. The main advantage of this architecture is that it does not employ any MAC unit, whose operational speed is, generally, a

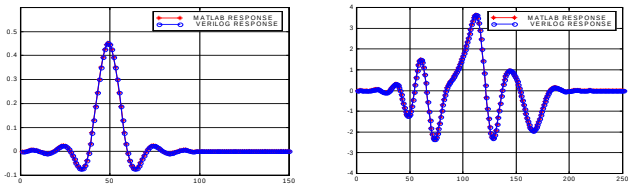


Figure 7 Response of I-Channel Block in MATLAB and VERILOG
a Impulse input b Random Input Sequence

bottleneck in filter throughput. Again, it makes extensive use of LUTs and hence is ideally suited for FPGA implementation.

3. Implementation of Digital Frequency Synthesizer

We use a digital frequency synthesizer in our system to generate a sampled sinusoidal wave of frequency 71 KHz ± estimated carrier frequency offset.

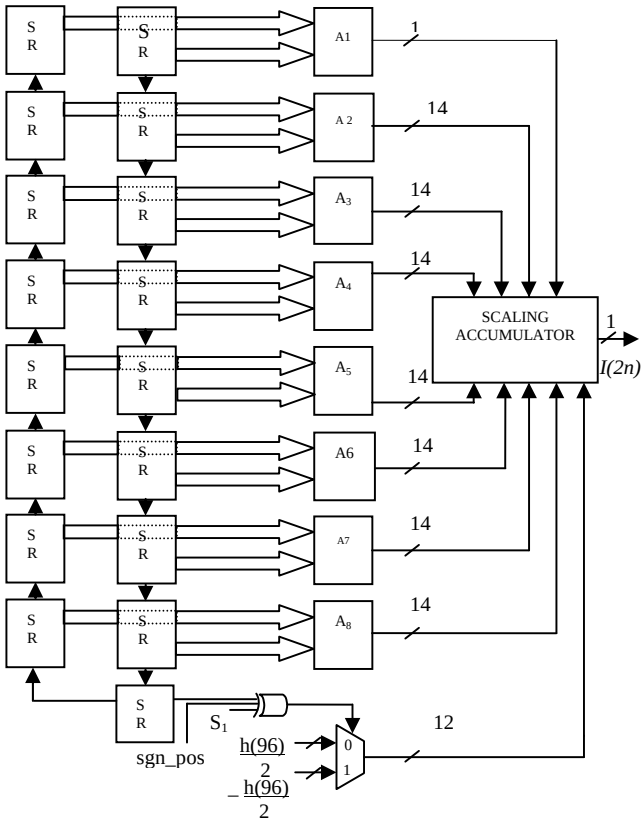


Figure 6 Telescopic View of I-Channel Block Implementation

3.1 Introduction

The major advantage of Digital Frequency Synthesizer is that its output frequency, phase and amplitude can be precisely and rapidly manipulated under the control of a DSP. Other inherent DFS attributes include the ability to tune with extremely fine frequency and phase resolution and to rapidly “hop” between the frequencies. These combined characteristics have made this technology popular in military, radar and communications systems.

3.2 Analysis

Various techniques are available in the literature for quarter wave memory compression, such as Sine-phase difference algorithm, Taylor series expansion, Modified Sunderland Architecture, Nicholas’ Architecture, CORDIC (Coordinate Rotation Digital Computer) Algorithm, etc. The implicit goal of these phase-to-sine conversion techniques is to reduce the maximum amplitude error for any phase angle, in effect mimicking the behavior of a LUT. In pursuing this goal, all architectures become complex in one way or the other. Also, the ROM size becomes fairly large as it grows exponentially with the width of the phase accumulator,

whereas a large phase accumulator width is desirable in order to achieve fine frequency tuning. Truncating the phase accumulator output, on the other hand, introduces spurious harmonics.

3.3 Concept of the Architecture used

Instead of a ROM LUT, a hardware-optimized phase-to-sine amplitude converter approximates the first quadrant of the sine function with eight equal-length piecewise linear segments [8]. The main goal is to maintain low system complexity and reduce power consumption and chip area requirements. The second aim is to achieve a specified spectral purity, which is defined as the ratio of the power in the desired frequency to the power in the greatest harmonic, across the synthesizer's tuning bandwidth. Spectral purity is an essential design parameter for synthesizer used in communication systems, ensuring that undesired in-band signals remain below a given threshold and are not detected.

In order to achieve the first goal, we approximate a sinusoid as a series of eight equal-length piecewise continuous linear segments,

$$s_i(x) = m_i \times (x - \frac{i}{8}) + y_i, \qquad i \in [0,7]$$

where m_i is the slope of each segment and is carefully selected to eliminate the requirement for multiplication by representing each one as a sum of at the most two powers of two. This is well known and often used technique [15]. We also restrict the precision of slope representation, i.e., the difference between the smallest and the largest powers of two used; in effect putting an upper bound on the adder's width. Equal length segments are selected to reduce the control system circuitry costs. In order to achieve a desired spectral purity, different sets of m_i and y_i coefficients are evaluated and the best one meeting the requirements is selected.

3.4 Description of the architecture

The new DFS architecture is shown in Fig. 8. It corresponds to a set of coefficients yielding 60dB purity. The coefficients are given in Table 4.

The phase to sine amplitude converter block includes a 1's complement to exploit quarter wave symmetry, as previously seen in other structures. Clearly, this architecture is significantly less complex than those of the other methods discussed previously. It does not include a ROM. No multipliers or squaring circuits are required. Equal length segments are used to simplify the control circuitry. Only three integers need to be added and multiplexers shown in Fig. 8 have been optimized by combining similar inputs and implemented in combinational logic.

The phase accumulator is 20 bits wide, truncated to 12 bits. The two MSBs are used for quadrant symmetry. The next three bits identify the segment. The remaining seven bits identify different sub-angles. The two upper multiplexers shift these remaining seven bits according to the slopes m_i, listed in Table 4.

In Fig. 8, the notation {>>n} signifies a right shift by n bits, or equivalently, division by 2^n. The lower multiplexer selects the appropriate y_i approximation listed in the table. The output from the multiplexers is 13 bits wide, to account for the whole dynamic range of possible values. The three-operand adder sums the multiplexer outputs together and rounds the result to 7 bits.

Table 4 Linear segment coefficients for 60 dB purity

i	m_i	y_i
0	1 + ½	2/1024
1	1 + ½	191/1024
2	1 + ¼	384/1024
3	1 + 1/8	552/1024
4	1	697/1024
5	½ + ¼	819/1024
6	½	909/1024
7	1/8	971/1024

3.5 Results of DFS

To verify the architecture, the design was coded in Verilog. The spectrum was obtained by taking the DFT of one grand repetition of the system output data. It can be easily seen that all the spurs are at least approx. 60 dB below the fundamental. Spectra for other odd frequency control words are similar, with spurs no greater than those shown here. It has been shown [13] that the spectrum corresponding to two frequency control words that are relatively prime to the phase accumulator's overflow values are permutations of each other. The amplitudes shown in the spectra of Fig. 9 are, therefore, exact in amplitude to the spurs for any other odd frequency control word. Only their locations change. The spectral purity of the DFS output is sufficient for the present system requirement.

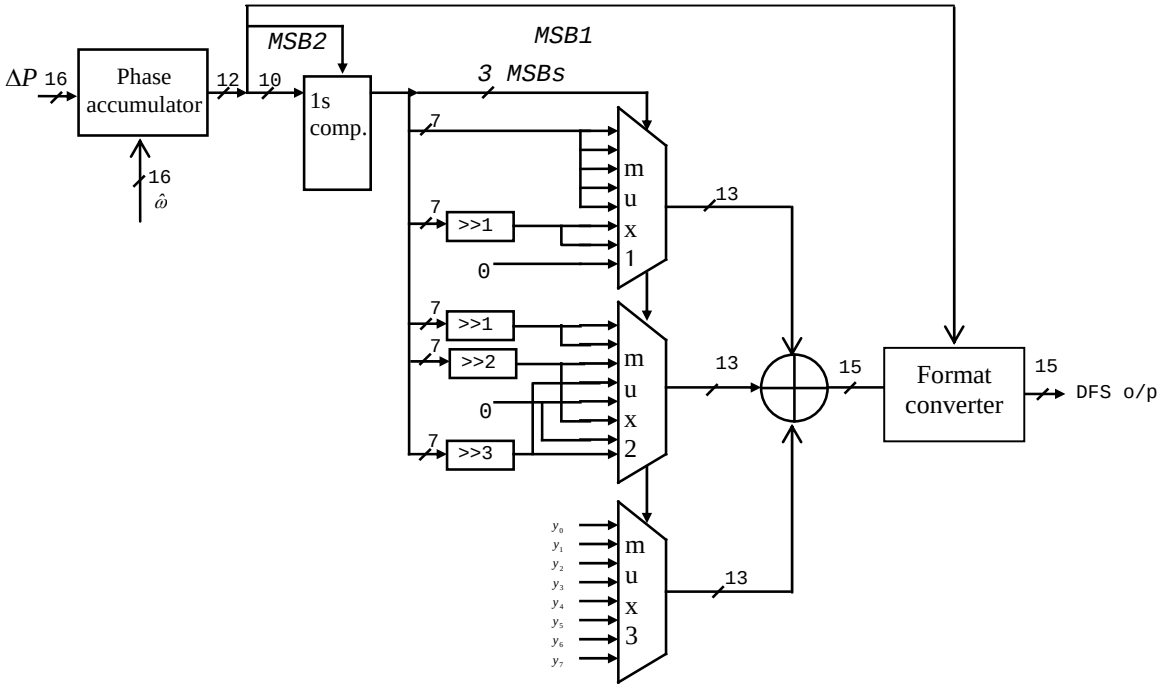


Figure 8 DFS Architecture

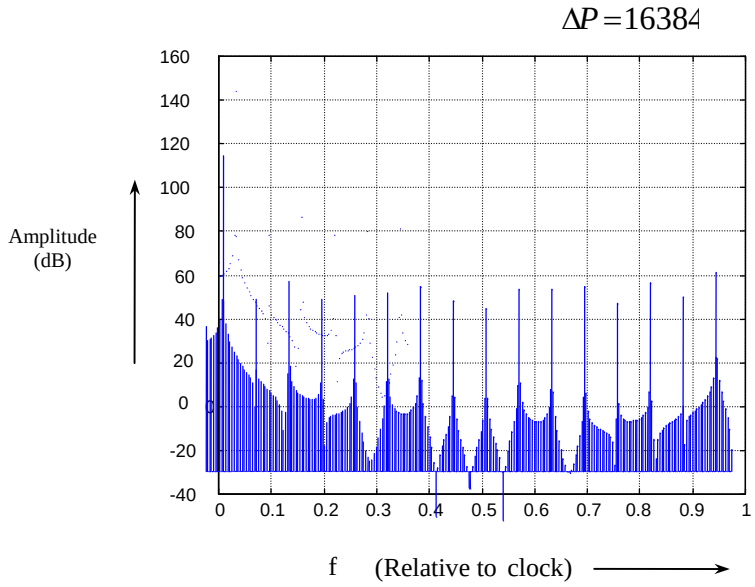


Figure 9 Output Spectrum

4. Implementation of FPGA part of the demodulator

4.1 Introduction

In our earlier paper [19], synchronization techniques and receiver structure were discussed. Also, the task of partitioning the receiver algorithm into two parts, one to be implemented on FPGA and the other on DSP, was performed. We presented a new architecture for sampling rate converter implementation in Section 2, which results in the use of reduced memory utilization. The DSP implementation of the remaining part is out of scope of the present work. In this section, we will describe the hardware implementation details of the FPGA part of the demodulator.

4.2 Implementation of mixer and RRC filters using Verilog on Xilinx FPGA

A block diagram of the FPGA part of the demodulator as implemented using Verilog is shown in Fig. 10. The developed core consists of a mixer and two numbers of 193 tap, RRC filters to accept modulated, 12-bit, signed ADC output at a sampling frequency of 1.536 MHz and convert it into in-phase (I) and quadrature-phase (Q) channel outputs, each of size 16 bits, signed, at half the sampling frequency. It may be noted that the core requires two input clocks for successful operation: 'clk_bit' must be 8.5 times the frequency of 'clk_word' frequency. These clocks may be easily generated from the sampling frequency, 'clk_word' (1.536 MHz) using a PLL and three numbers of frequency dividers.
Fig. 11 shows the RTL View of FPGA part of the demodulator after running the synthesis using Synplify. The synchronous clock, 'clk_wordby2', for the output channels, I and Q, is generated by the core and, the outputs may be registered at the positive edge of this clock when I_Q_valid is high. EDF file generated by Synplify tool is input to the Xilinx place and route and the results for the same are tabulated in Table 5. This was followed by running Xilinx back annotation and simulation using Modelsim. The results obtained after back annotation were verified to be correct.

Table 5 FPGA Implementation details

FPGA DEVICE	LOGIC GATES	PERFORMANCE (MHz) *	
		clk_word	clk_bit
Xilinx Virtex XCV600hq240-4	44,652	54.3 #	19.8 $

* Maximum operating frequency
# 1.536 MHz and $ 13.056 MHz are what is required for normal operation.

4.3 Verification

Fig. 12 shows the blocks of the part of the demodulator implemented using Verilog and, the necessary Matlab modules required for its verification. Matlab generates the input signal for the Modulator. For the sake of testing, two sine wave frequencies, 2 KHz and 20 KHz, one at a time, are used as input. The Matlab Modulator output is signed, 16 bits wide. This serves as the input for both the Matlab and Verilog demodulators as shown.

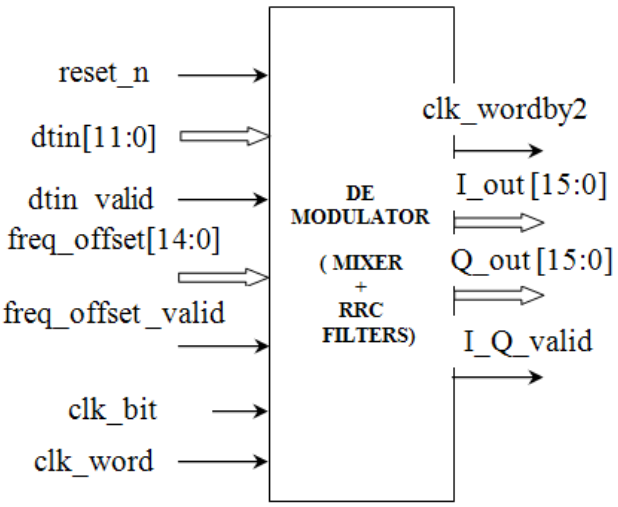


Figure 10 Block diagram of FPGA part of the demodulator

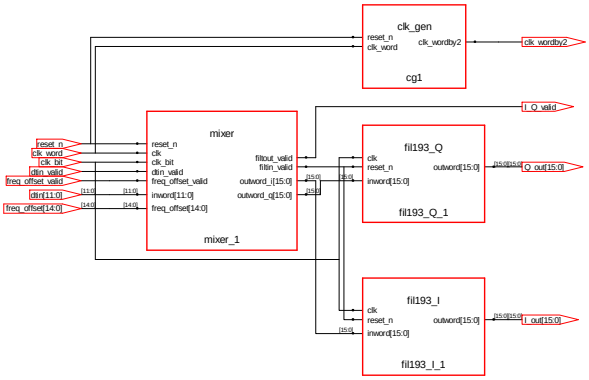


Fig. 11 Synthesis (Synplify) RTL View of FPGA part of the demodulator

It may be noted that the input to the Verilog demodulator is MSB 12 bits (and not the entire 16 bits) so as to keep the same precision as the ADC output in the existing ADSP implementation. 16-bit precision has been retained for the input to the Matlab demodulator since it serves as a better standard reference for verifying the Verilog implementation. The modulated signal, thus produced, is applied to both the Matlab and Verilog demodulators. The

I and Q channels are the required outputs, and are each of width 16 bits, signed. The Verilog design requires two clocks, clk_bit (13.056 MHz) and clk_word (1.536 MHz) for its operation. The I and Q outputs from Verilog demodulator compare favorably with the corresponding outputs of Matlab demodulator, which has served as the reference for verification of the hardware implemented on FPGA. I and Q results are shown in Fig. 13 for 20 KHz as an example.

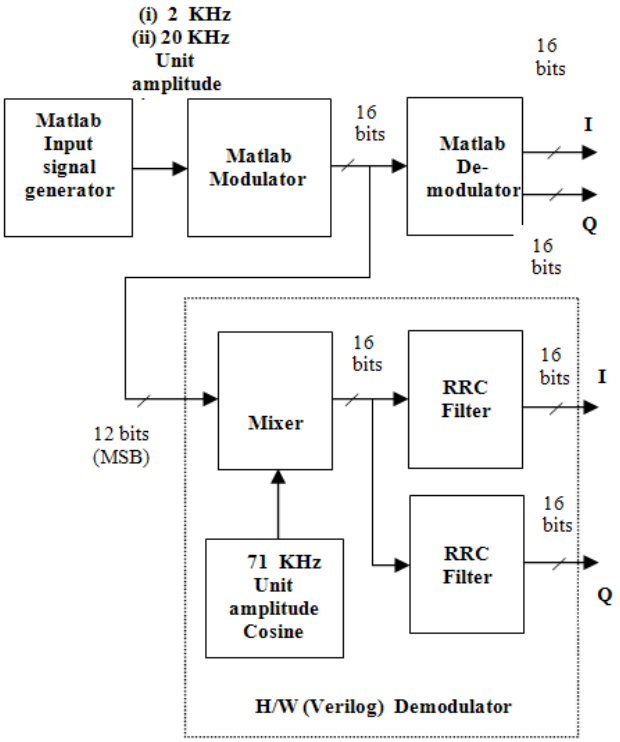


Fig. 12 Blocks of the part of the demodulator implemented using Verilog and the Matlab modules for its verification

5. Conclusions

This paper proposed a new distributed arithmetic based architecture for implementing a Sampling Rate Converter. The main advantage of this architecture is that it does not employ any MAC unit, whose operation speed is, generally, a bottleneck for high filter throughput. It makes extensive use of LUTs and hence is ideally suited for FPGA implementation. An architecture for Digital Frequency Synthesizer, which gives 60 dB spectral purity was also presented. Both the architectures were coded in Verilog HDL and implemented on XILINX FPGA. The hardware and MATLAB results compare favorably.

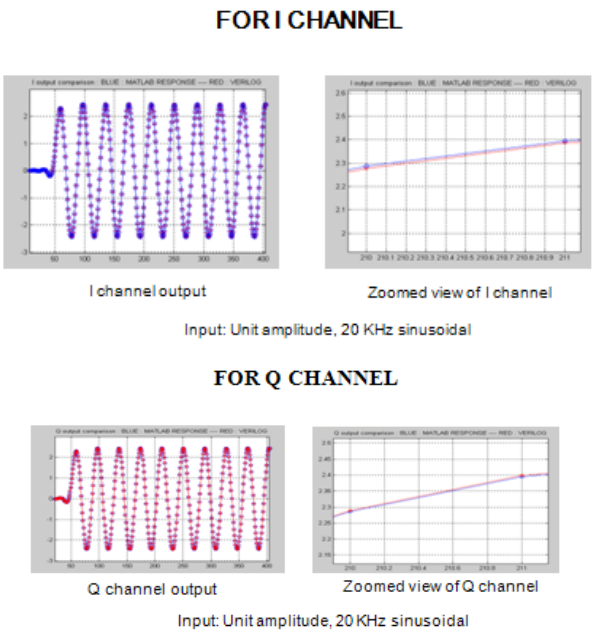


Figure 13 Matlab and Verilog responses of I and Q channels at 20 KHz

References

[1] G. Strang: Introduction to Applied Mathematics, Wellesly Cambridge Press (1986).
[2] J. M. Tribolet: A New Unwrapping Algorithm, IEEE Trans. on Acoustic, Speech and Signal Processing, Vol. ASSP-25, No-2, pp. 170-177 (1977).
[3] Mathew P. Joseph: DSP Algorithms for On-Board Satellite Trans-multiplexer and Receiver, MS Thesis, IIT Madras, India (2000).
[4] S. Haykin: Adaptive Filter Theory, Prentice-Hall, 2nd Edition (1991).
[5] M. E. Frerking: Digital Signal Processing in Communication Systems, Van Nostrand Reinhold, NY (1993).
[6] E. A. Lee and D. G. Messerschmitt, Digital Communication, Second Edition, Allied Publishers Limited, 1994.
[7] J. Proakis: Digital Communications, Third Edition, International Edition, McGraw Hill (1995).
[8] J.M.P. Langlois, D. Al-Khalili, R.J. Inkol: A High Performance, Wide bandwidth, Low cost FPGA based Quadrature Demodulator, Proceedings of IEEE Canadian Conference on Electrical and Computer Engineering (1999).
[9] Henry Samueli, Bennet C. Wong: A VLSI Architecture for a High Speed, All Digital, Quadrature Modulator and Demodulator for Digital Radio Applications, IEEE Journal on Selected Areas in Communication , Vol. 8, No. 8 (1990).
[10] Sanjeev Dua: Algorithms and Architectural Design of an Onboard Satellite QPSK Receiver. MS Thesis, IIT Madras, India (2003).
[11] P.P. Vaidyanathan: Multirate Systems and Filter Banks,

Prentice Hall Inc., Eagle-woods Cliffs, N.J. (1993).
[12] Keshab K. Parhi: VLSI Digital Signal Processing Systems, John Wiley & Sons Inc. (1999).
[13] H. T. Nicholas, H. Samueli , and B. Kim, The Optimization of Direct Digital Frequency Synthesizer in the Presence of Finite Word Length Effects Performance, in Proc. Of 42nd Annual Frequency Control Symposium, June 1988, pp. 357-363.
[14] J. M. P Langlois and D. Al-Khalili, Hardware Optimized Direct Digital Frequency Synthesizer Architecture with 60 dB Spectral Purity, Proc. IEEE International Symposium On Circuits and Systems, May 2002.
[15] S. I. Liu, T. B. Yu and H. W. Tsao, Pipelined Direct Digital Frequency Synthesizer Using Decomposition Method, IEEE Proceedings on Circuits, Devices and Systems, Vol. 148, No. 3, June 2001, pp. 141-144.
[16] Fubing Yu: FPGA implementation of a fully digital FM demodulator, Communications Systems (ICCS), The Ninth International Conference, pp. 446-450 (2004).
[17] Charoensak, C., Abeysekera, S. S.: FPGA implementation of efficient Kalman band-pass sigma-delta filter for application in FM demodulation, SOC Conference Proceedings, IEEE International Volume, pp. 137-138, 12-15 Sept. (2004).
[18] Zarifi, M.H.; Frounchi, J.; Asgarifar, S.; Baradaran Nia, M, FPGA implementation of a fully digital demodulation technique for biomedical application, Proceedings of IEEE Canadian Conference on Electrical and Computer Engineering, pp. 1265-1268, 4-7 May (2008).
[19] K. R. Nataraj, S. Ramachandran, B. S. Nagbushana, Development of Algorithm for Demodulator for Processing Satellite Data Communication, IJCSNS, Vol. 9, No. 6, June 30, 2009, pp. 233-243.

K. R. Nataraj obtained his ME degree from Bangalore University, India in 2000. He worked as Professor and Head of the Department during 2000-2008 and currently he is the Post Graduate Coordinator in the Department of Electronics and Communication in SJB Institute of Technology, Bangalore. Presently, he is pursuing his Ph. D. degree in Dr MGR University, Chennai. His research interests include Wireless communication, FPGA implementation, Microcontroller and Embedded systems design. He is a member of MIE, MISTE and IETE.

Dr S. Ramachandran obtained his M. Tech. and Ph. D. degrees from the Indian Institute of Technology, Kanpur and Madras respectively. He has wide academic as well as industrial experience of over 30 years, having worked as Professor in various engineering colleges as well as design engineer in industries in India and USA, designing systems and teaching/guiding students. His research interests include developing algorithms, architectures and implementations on FPGAs/ASICs for Video Processing, DSP applications, reconfigurable computing, open loop control systems, etc. He is the recipient of the Best Design Award at VLSI Design 2000, International Conference held at Calcutta, India and the Best Paper Award of the Session at WMSCI 2006, Orlando, Florida, USA. He has completed a video course on Digital VLSI System Design at the Indian Institute of Technology Madras, India for broadcast on TV by National Programme on Technology on Enhanced Learning (NPTEL) and is being broadcast in You Tube as well. He has also written a book on Digital VLSI Systems Design, published by Springer Verlag, Netherlands (www.springer.com).

Dr B. S. Nagabushana obtained his M. Tech. and Ph. D. degrees from Mysore University and Indian Institute of Science, Bangalore respectively. He has wide academic as well as software industrial experience for over 25 years. He has worked as Professor invarious engineering colleges as well as consultant to software industries like OMED (Software), Japan, BFL, CG-Smith Software Pvt. Ltd, KPIT Cummins Infosystems (Bangalore), Pvt. Limited, San Lab Technologies etc. His research interests include Wireless Communication, Neural Network, Fuzzy Logic and Embedded systems. He is the recipient of NRDC Independence Day award for the year 1992, Best Project Execution award for the year 2000 from M/s BMC Software, USA.