Comparison of VLSI architectures for a WLAN OFDM transmitter with interpolation filters

Pavlos Kolovos, Eleni Fotopoulou and Thanos Stouraitis
Department of Electrical and Computer Engineering,
University of Patras, Greece
helenf@ece.upatras.gr

***Abstract*—This paper introduces two efficient computational techniques for Orthogonal Frequency Division Multiplexing (OFDM) based transmitter design with interpolation filters and discusses the corresponding VLSI architecture issues. The proposed technique is demonstrated by detailing the implementation of a 64-point, radix-2^4 SDF FFT in combination with digital interpolation filters. By exploiting the redundancy into the cyclic prefix part of the OFDM symbol, the computational load of the transmitter is reduced by approximately 20% for the IEEE 802.11 WLAN standard. The first technique outperforms the second one regarding the number of operations, while the second one is more appealing for applications where required area is an issue.**

I. Introduction

The current increase of demand for wireless communications applications, dictates a necessity for high-speed and spectral-efficient communications over wireless highly dispersive channels. Orthogonal Frequency Division Multiplexing (OFDM) has been selected as an efficient solution for high-speed digital wireless communications and is employed in current wireless communication standards, such as IEEE 802.11b in the 5GHz band and 802.11g in the 2.4GHz band, for wireless LANs, DVB for digital terrestrial TV broadcasting, WiMax for wireless MANs *etc.*

In the 802.11 WLAN standard [1] the baseband signal is generated by using the IFFT, then a guard interval (also called a cyclic-prefix-CP) is added to make the system robust to multipath. The IFFT modulation produces a spectra of overlapping subcarriers, where each subcarrier is orthogonal to all other subcarriers. Respectively, the transmitted data are demodulated by an FFT in the receiver.

In this paper, the study and implementation of interpolation filters in the digital part of an OFDM transmitter is presented. We propose two architectures where the filtered cyclic prefix part is not explicitly computed; instead it is constructed from the filtering of the useful OFDM symbol part. In this way the proposed algorithm, avoids the filtering of the total OFDM symbol, thus significantly reducing the number of multiplications needed by approximately 20%, for the case of a 64-point IFFT with cyclic prefix of one fourth of the OFDM symbol. The digital filter is implemented as a polyphase interpolator described by Vaidyanathan [2], thus avoiding data oversampling prior to filtering.

We thank the European Social Fund (ESF), Operational Program for Educational and Vocational Training II (EPEAEK II), and particularly the Program PYTHAGORAS II, for funding the above work.

In previous work upsampling with interpolation filters has been discussed. A technique called circular time domain interpolation is applied where the 64-point IFFT result is both pre- and post- extended and then upsampled and filtered requiring additional computations, while a radix-2^2 pipeline FFT structure is used [3]. The IFFT structure used there is outperformed by the proposed architectures of a modified radix-2^4 FFT . Construction of the CP after filtering has already been presented in [4]. It is based on a combination of a radix-4 Multipath-Delay Commutator FFT architecture [5] with 4 parallel filters. In this paper we propose architectures which outperform the previous designs in terms of area, number of computations, and utilization.

The remainder of the paper is organized as follows: In Section II, a brief overview of the OFDM transmitter is presented. Section III describes the proposed architecture. Specifically in Section III-A, the 64-point, radix-2^4 Decimation-in-Time Single Delay Feedback (SDF) FFT architecture is presented, while Section III-B details the first proposed algorithm for the convolution of the OFDM symbol with the interpolation filter and Section III-C describes the second proposed algorithm. Results of implementation are discussed in Section IV and conclusions are given in Section V.

II. Overview of the OFDM Transmitter

A brief description of the organization of an OFDM transmitter [6], shown in Fig. 1, follows. The binary input data are encoded by a forward error correction scheme. The encoded data are subsequently interleaved and mapped onto QAM values. After the pilot tones are inserted, the IFFT modulation is performed to produce the useful OFDM symbol part.

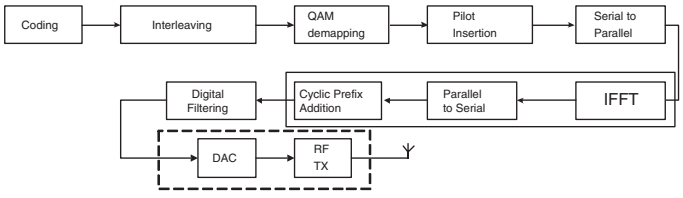


Fig. 1. Block diagram of an OFDM transmitter

Low-pass interpolation filters need to be included in the transmitter in order to double the sample rate and simplify

Authorized licensed use limited to: New Jersey Institute of Technology. Downloaded on March 24,2010 at 12:20:35 EDT from IEEE Xplore. Restrictions apply.

the analog reconstruction filters [7]. This can be achieved by using a double-in-size IFFT, but this enlarges the required area, requires faster D-A converters, and does not achieve reduction of complexity in the analog filter. Thus, there are advantages in the use of interpolation filters after the IFFT.

The subcarrier orthogonality in OFDM is retained, when the signal is transmitted over a non-dispersive channel. To deal with the time dispersion, a guard interval is introduced between successive OFDM symbols. In most cases, the guard interval is a cyclic repetition of a number of the last samples of the IFFT output, pre-pended to the symbol part, called the cyclic prefix (CP). Due to bandwidth restrictions the CP is usually limited to less then one quarter of the useful OFDM symbol part. The OFDM symbol, after filtering, is sent to the Digital-to-Analog converter and passes on to the RF part of the modem.

III. The Proposed Architecture

In Figure 2, the basic architecture for constructing the OFDM signal is presented. The basic architecture consists of the basic IFFT block, the FIR interpolation filter, which upsamples the signal by 2, and a buffer that is used to construct the cyclic suffix.

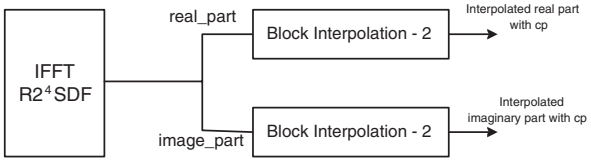


Fig. 2. Basic architecture for constructing the OFDM signal

The study is based on the IEEE 802.11a standard [1] and thus a 64-point FFT is implemented under the required specifications based on 16-QAM modulation and coding rate 3/4. Simulation results have shown that the optimal interpolatin filter implementation is a 15-tap FIR filter.

A. The IFFT block

The 64-point IFFT block is based on [8]. The algorithm is based on a Decimation in Frequency (DIF) algorithm, and so the output data are in bit-reversed order. In ordered for the interpolation filter to process the IFFT output in real-time, we need the IFFT output to be produced in normal order. The radix-2^4-Single Delay Feedback (SDF) algorithm needs 128 clock cycles to complete the transform. For the reordering procedure, a buffer is used for storage of the output samples. Specifically, the first output is obtained at clock cycle 64 and stored at position 0 of the buffer. After cycle 64, a new, output is obtained and stored in the buffer. Finally, the reordered sequence is obtained in 128 cycles. As the filter processing may commence only after clock cycle 128, and not in an overlapping manner with the IFFT output, an extra 64 clock cycles are required for the completion of the whole procedure.

From the above, computation of the suffix requires a total of 128+64+16=208 clock cycles, which according to the standard, have to be executed within 4 μsec. This results in a clock frequency at least 52 MHz, which leads to high power consumption especially by taking into account that application of the interpolation filter will result in double sampling frequency of the output data. Additionally, according to the standard, the input data samples arrive with a frequency of 20 MHz. In most implementations of the HIPERLAN 2 or IEEE802.11b standard, the sampling frequency is double or quadruple, hence 40-80 MHz, which makes this architecture unsuitable for implementation.

By organizing the IFFT output to be in normal order, processing by the interpolation filter can be achieved in real time. This reduces the required clock cycles to 151 and the minimum required clock operation frequency of the system to 37,75 MHz, as explained in detail in Section III-D. It is known ([9]) that for each decimation-in-time (DIT) FFT algorithm, there exists a DIF FFT algorithm that corresponds to interchanging the input and output and reversing the direction of all the arrows in the flow graph. In the proposed architecture the radix-2^4-SDF algorithm [8] is used but based on the DIT signal flow graph of the FFT algorithm. The block diagram of the FFT block implemented is shown in Fig. 3.

B. Architecture with two interpolation filters

In this paper, two different architectures are proposed to construct the final OFDM symbol. These schemes differ from the straightforward technique, regarding the creation of the cyclic prefix part, as the extension by $N/4$ of the input data is not necessary. In contrary, the input data sequence remains the same and the convolution of the first $N/4$ samples is used to compute the cyclic suffix needed to construct the final OFDM symbol. The method used to compute the convolution with the FIR filter is based on the *overlap-add* method ([9]), in which the input sequence is divided in sub-blocks and each sub-block is convolved with the filter coefficients. The final filtered output is produced by adding the corresponding appropriate output samples.

The data coming from the IFFT output are in normal order: $x(0), x(1), \ldots, x(63)$

To apply the upsampling filter, zeros are interpolated between the samples.

As defined in the standard, the $N/4$ first samples of that sequence is used to calculate the cyclic suffix. Thus, using the overlap-add method, the input sequence $x(n)$ can be represented as $x(n) = \sum_{k=0}^{1} x_k(n)$, where

$$x_0(n) = \begin{cases} x(n), & 0 \le n \le N/4 - 1 \\ 0, & \text{otherwise} \end{cases}, \quad (1)$$

$$x_1(n) = \begin{cases} x(n), & N/4 - 1 \le n \le N - 1 \\ 0, & \text{otherwise} \end{cases},$$

The linear convolution $y(n)$ of the sequence $x(n)$ is then $y(n) = x(n) * h(n) = (\sum_{k=0}^{1} x_k(n)) * h(n) = x_0(n) * h(n) + x_1(n) * h(n)$

If the impulse response of the filter is P samples and the input sequence is N samples (in this case N=128), then the convolution output will be $N + P - 1$. Thus there is a region of $P - 1$ samples over which the first convolution overlaps

Authorized licensed use limited to: New Jersey Institute of Technology. Downloaded on March 24,2010 at 12:20:35 EDT from IEEE Xplore. Restrictions apply.

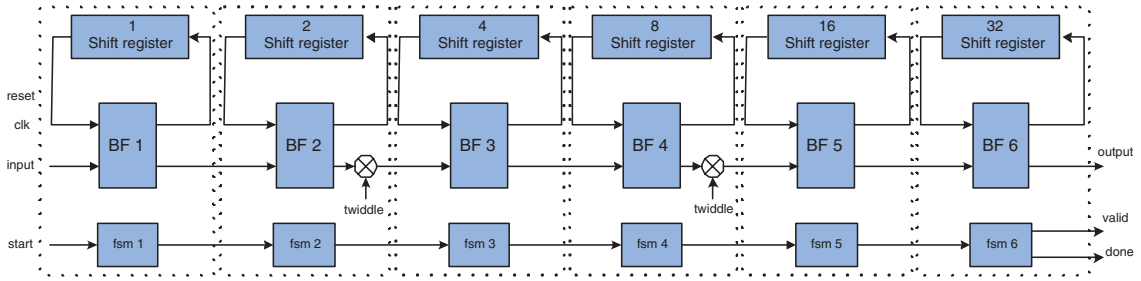


Fig. 3. Block diagram of the r2^{4}SDF algorithm based on DIT

with the second convolution and the corresponding appropriate output samples must be added resulting in $y(n),\ \ n = 0...N+P-1$. The convolution $y_0(n)$ of the first sequence $x_0(n)$ is actually the convolution of the cyclic suffix and may be used to construct the filtered OFDM symbol y_{OFDM}, by adding the appropriate overlapping samples of the sequence generated by $y(n)$.

The construction of the filtered OFDM symbol y_{OFDM} is given by

$$y_{\text{OFDM}}(n) = \begin{cases} y(n), & 0 \le n \le N-1 \\ y(n-N) + y_0(n-N), & N \le n \le P+N-1 \\ y_0(n-N), & P+N \le n \le P+(N+N/4)-1 \end{cases} \quad (2)$$

The computational complexity of the above method is PN multiplications and $(N+2)(P-1)$ additions.

In the straightforward approach, the input sequence to the filter is extended by $N/4$ and the computational load is $P(N+N/4)$ multiplications and $(P-1)(N+N/4)$ additions.

Thus, the gain in computational complexity is $\frac{P(N+N/4)-PN}{P(N+N/4)} = 20\%$, for multiplications, and $\frac{(P-1)(N+N/4)-(N+2)(P-1)}{(P-1)(N+N/4)} = \frac{N-8}{5N}$ for additions. For $N = 128$, in this implementation, the gain for additions is also 20%.

Fig. 4 shows the proposed architecture for implementing the above method.

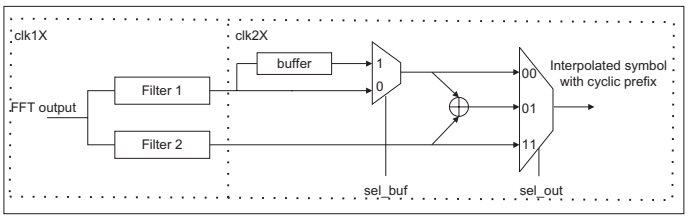


Fig. 4. Architecture for computing a filtered OFDM symbol

The output data from the IFFT block, appear in normal order, are sampled by clock $clk1X$, and are fed as inputs to Filters 1 and 2 of Fig 4. In contrary, the filter outputs are sampled by clock $clk2X$ that operates in double frequency of $clk1X$. All references to clock cycles from this point forward are in correspondence to $clk2X$. The control signals `sel_buf` and `sel_out` determine the input and output of the adder, respectively. Filter 1 is used to compute the convolution of sequence $x_0, ..., x_{15}$, while Filter 2 is used for the sequence $x_{16}, ..., x_{63}$. The description and timing for the 1st method are summarized in Table I.

TABLE I
TIMING AND OPERATION OF 1ST PROPOSED ARCHITECTURE

Number of cycles	Description
1-32	Filter 1 is activated, output data are stored in a 46-word buffer, while the same elements constitute the output of the system (`sel\_buf=0` and `sel_out==00`)
33-46	Filter 2 is also activated (the two filters are working in parallel), while the output consists of the sum of the outputs of the first and second filter (overlap in time (`sel\_buf=0` and `sel_out==01`). After cycle 46, Filter 1 is deactivated.
47-96	The output of the system is the output of Filter 2
97-110	The output of the buffer is added to the output of filter2 (`sel\_buf=1` and `sel_out==01`)
111-174	The output of the system is provided by the output of the buffer.

C. Architecture with one interpolation filter

The second proposed architecture with one interpolation filter is shown in Fig 5.

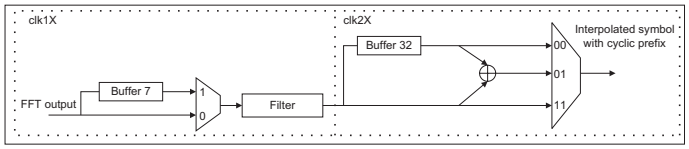


Fig. 5. Second architecture for computing final OFDM symbol

In the second proposed architecture the initial input sequence $x(n)$ is not divided in two subsequences. In contrary, $x(n)$ is convolved as is with the filter coefficients, while the appropriate number of the first output samples are stored in a buffer.

As in the first architecture, the system consists of two clocks. The clock cycles in the following description refer to clk2X, while the input sequence length is 128 and the filter taps are 15. The control signals `sel\_inbuf` and `sel_out` determine the circuits filter input and the system output, respectively. The description of the operation and the timing of the second method is summarized in Table II.

The computational complexity of the above method is $(N+P-1)P$ multiplications and $(N+P)(P-1)$ additions.

Thus, the gain in computational complexity, compared to a straightforward implementation for multiplications and additions, is $\frac{N-4(P-1)}{5N}$ and $\frac{N-4P}{5N}$ respectively.

D. Comparison of the two proposed architectures

In Tables III and IV, the results of comparing the two proposed architectures are shown

The first method is better than the second one regarding computational complexity. But it requires double area, which

Authorized licensed use limited to: New Jersey Institute of Technology. Downloaded on March 24,2010 at 12:20:35 EDT from IEEE Xplore. Restrictions apply.

TABLE II
TIMING AND OPERATION OF 2ND PROPOSED ARCHITECTURE

Number of cycles	Description
1-32	The filter output is stored in Buffer32 for later use during computation
19-32	The input samples are stored in Buffer7, for later use during computation
1-128	The input data is convolved with, the filter coefficients and the output is the systems output
129-142	There is overlap in time and the first 14 samples of Buffer32 are added to the filter output (`sel\_buf=0` and `sel_out==01`)
143-160	The stored data in Buffer32 are the system output. The 14 samples remaining for the completion of the computation have to be reconstructed by the filter. Thus, in time cycle 147, data stored in Buffer7 are serially passed to the filter, disregarding the output samples.
161-174	The final $P-1$ output samples of the system are actually the filter outputs.

TABLE III
COMPARISON OF COMPLEXITY OF THE TWO ARCHITECTURES

Method	Gain Mult/tion complex.	IEEE 802.11b (N=128 P=15)	Gain Sum/tion complex.	IEEE 802.11b (N=128 P=15)
1	0.2	20%	$\frac{N-8}{5N}$	18,75 %
2	$\frac{N-4(P-1)}{5N}$	11,25 %	$\frac{N-4P}{5N}$	10,63 %

is important, especially for the implementation of the design in FPGA, where the available resources are limited. Additionally, the proposed architecture of the first method, requires a temporary memory (buffer) of $N/4+P-1$ words, while, for the second one the required memory is of length $N/4+(P-1)/2$ words.

In the case where the available hardware resources are not limited and the double required area is not an issue, the first architecture is better than the second one, as the computational complexity is independent of the filter length and depends only on the input sequence. Also, even though the required hardware is double, the power consumption does not increase significantly as the two filters work in parallel only for $P-1$ cycles.

The second method requires much less hardware, which makes it a more appealing choice for implementation. Also the utilization is pretty high, as in the case where $N=128$, as required by the standard, and for the chosen $P=15$, utilization is 97,7%. In contrast, for the first method utilization of Filter1 is 26,4% and of Filter2 63,2%. The number of clock cycles of clk1X required for the completion of the procedure is 151. This is a result of the fact that 64 cycles of clock clk1X are required for the input data to be fed to the filter and an additional 174/2 cycles (174 cycles for clk2X of double frequency than clk1X)are required for the completion of the whole procedure.

The common advantage of the two proposed architectures is the reduction of computational complexity in comparison to a straightforward implementation, where the cyclic prefix is appended to the input sequence prior to being passed to the interpolation filter. Reduction of computational complexity results in reduction in power consumption.

TABLE IV
COMPARISON OF UTILIZATION OF THE TWO ARCHITECTURES

Method	Utilization	
	Filter1	Filter2
1st	Is used for $N/4+P-1$ cycles Utilization $\frac{N/4+P-1}{5N/4+P-1}$	Is used for $3N/4+P-1$ cycles Utilization $\frac{3N/4+P-1}{5N/4+P-1}$
2nd	Is used for $N+3(P-1)$ cycles Utilization $\frac{N+3(P-1)}{5N/4+P-1}$	

IV. IMPLEMENTATION

Synthesis of the proposed architectures was realized with the Xilinx^{TM} Ise 5.2i tool and the FPGA device that the design was targeted on was Virtex XCV300 package 352g and speed grade -6. The results of synthesis for the first and second architecture are summarized in Table V.

TABLE V
FPGA RESOURCE USE AND PERFORMANCE OF THE PROPOSED BLOCKS

Virtex XCV 300	**1st Arch.**	**2nd Arch.**	**IFFT (Optimized)**
Number of Slices	73%	62%	51%
Number of Flip-Flops	16%	11%	9%
Number of 4-input LUTs	66%	56%	47%
Number of bonded IOBs	20%	20%	19%
Number of BRAMs	12%	25%	-
Number of GCLKs	50%	50%	25%
Max. Frequency (MHz)	53	40.77	55.86

V. CONCLUSIONS

In this paper we have introduced two architecture of a 64-point radix-2^4 SDF FFT architecture followed by digital interpolation filters. By taking advantage of the cyclic repetition of the guard interval of the OFDM symbol, a significant reduction of approximately 20% of the required arithmetic operations, for the operation of the convolution is achieved for a 802.11 WLAN implementation. The first implementation is better in terms of computational complexity while the second requires less chip area.

REFERENCES

[1] "Wireless LAN MAC and PHY specifications High-Speed Physical Layer in the 5 ghz Band," IEEE, Tech. Rep. IEEE Std. 802.11a-1999, 2000.

[2] P. Vaidyanathan, *Multirate Systems and Filter Banks*. Prentice Hall Inc., 1993.

[3] S. Leyonhjelm and M. Faulkner, "Comparison of three interpolator designs for use in OFDM transmitters," *Electronics Letters*, vol. 42, no. 5, pp. 289– 291, Mar. 2006.

[4] E. Fotopoulou, V. Paliouras, and T. Stouraitis, "A computational technique and a VLSI architecture for digital pulse shaping in OFDM modems," in *Proceedings of the 2003 IEEE Symposium on Circuits and Systems (ISCAS 2003)*, vol. 2, May 2003, pp. 125–128.

[5] S. He and M. Torkelson, "Design and implementation of a 1024-point pipeline FFT processor," in *Proceedings of the IEEE Custom Integrated Circuits Conference*, May 1998, pp. 131–134.

[6] R. Van Nee and R. Prasad, *OFDM Wireless Multimedia Communications*. Artech House Publishers, 2000.

[7] M. Faulkner, "The effect of filtering on the performance of OFDM systems," *IEEE Transactions on Vehicular Technology*, vol. 49, no. 5, pp. 1877–1884, Sep. 2000.

[8] H.-Y. Oh and M.-S. Lim, "Area and power efficient pipeline FFT algorithm," in *Signal Processing Systems Design and Implementation, 2005. IEEE Workshop on*, Nov. 2005, pp. 520– 525.

[9] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*. Prentice Hall Inc., 1999.

Authorized licensed use limited to: New Jersey Institute of Technology. Downloaded on March 24,2010 at 12:20:35 EDT from IEEE Xplore. Restrictions apply.