Design of a Multiband Full-Rate Ultra-Wideband Receiver in FPGA

Mário Véstias
INESC-ID/ISEL/IPL
Lisbon, Portugal
Email: mvestias@deetc.isel.ipl.pt

Horácio Neto
INESC-ID/IST/UTL
Lisbon, Portugal
Email: hcn@inesc-id.pt

Helena Sarmento
INESC-ID/IST/UTL
Lisbon, Portugal
Email: hsarmento@inesc-id.pt

***Abstract*—MultiBand OFDM (MB-OFDM) UWB [1] is a short-range promising wireless technology for high data rate communications up to 480 Mbps. In this paper, we have designed and implemented in an Virtex-6 FPGA an MB-OFDM UWB receiver for the highest data rate of 480 Mbps. To test the system, we have also implemented an MB-OFDM transmitter and an AWGN generator in VHDL and determined the bit error rates at the receiver running in an FPGA.**

***Index Terms*—UWB, MB-OFDM, Receiver, FPGA**

I. Introduction

MultiBand OFDM (MB-OFDM) UWB [1] is a short-range wireless technology that permits data transfers at rates between 53.3 and 480 Mbps. MB-OFDM uses the already licensed radio spectrum, between 3.1 GHz - 10.6 GHz, in an unlicensed manner, i.e. without a licensing cost or control. MB-OFDM divides the spectrum allocated to UWB into 14 bands of 528 MHz. Each OFDM symbol is transmitted across a band.

A total of 100 data sub-carriers and 10 guard sub-carriers are used per symbol. In addition, 12 pilot sub-carriers allow for coherent detection. The time to process an OFDM symbol is 242.42 ns (1/4.125 μs).

Data subcarriers transport information modulated by QPSK or dual carrier modulation (DCM). DCM exploits channel diversity, by adding an alternative form of redundancy for the higher data rates (from 320 to 480 Mb/s). The additional redundancy is introduced by mapping the same four bits onto two individual subcarriers of the same OFDM symbol. The probability of coded information to experiment a deep fade is extremely small if two tones with the same information are separated by a large bandwidth.

Depending on the modulation used on data sub-carriers and code rate adopted, MB-OFDM supports data rates of 53.3, 80, 106.7, 160, 200, 320, 400 and 480 Mbps (see table I).

MB-OFDM receivers are typically implemented with ASICs. However, field-programmable gate arrays (FPGAs) have become extremely popular because they offer a combination of low cost, very fast turnaround and reconfigurability. Their flexibility, high density and considerable operating frequencies turned FPGA-based systems a good choice for digital processing. The latest generations of FPGAs include DSP capabilities and special features for I/O streaming turning them powerful enough to be used for an efficient replacement of ASICs in wireless applications [2].

978-1-4673-4900-0/13/$31.00 © 2013 IEEE

TABLE I. MB-OFDM Parameters

Data rate	Modulation	Code rate
53.3 Mbps	QPSK	1/3
80 Mbps	QPSK	1/2
106.7 Mbps	QPSK	1/3
160 Mbps	QPSK	1/2
200 Mbps	QPSK	5/8
320 Mbps	DCM	1/2
400 Mbps	DCM	5/8
480 Mbps	DCM	3/4

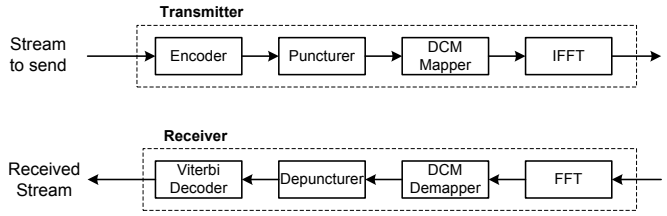

Fig. 1. Transmitter and receiver of an MB-OFDM system

A major challenge in the implementation of an MB-OFDM receiver in FPGA is the performance requirements over the Fast Fourier Transform (FFT) and the Viterbi decoder (VD) for the highest data rate (480 Mbps). This paper proposes an implementation for an MB-OFDM receiver focusing on the 480 Mbps data rate. We have designed and implemented the MB-OFDM UWB receiver in FPGA, together with a transmitter and model of the channel to test the circuit.

The paper is organized as follows. In section II, the baseband processing steps of the UWB transceiver are briefly described. In section III, we describe the implementation of the MB-OFDM receiver. Section IV describes the testbed scenario and presents the implementation results of the receiver in an FPGA. Finally, section V concludes the paper.

II. Baseband Processing of MB-OFDM

In the transmitter of an MB-OFDM system (see figure 1) the input stream is first encoded with a convolutional encoder. The convolutional code has a code rate of 1/3 and a constraint length of 7 defined by the polynomial generators $g_0 = 133_O$, $g_1 = 171_O$ and $g_2 = 165_O$.

The encoded bits are sent to be punctured. For the data

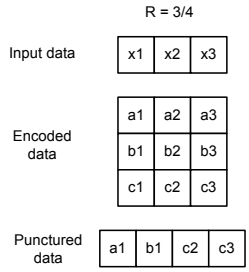

Fig. 2. Puncturer behaviour

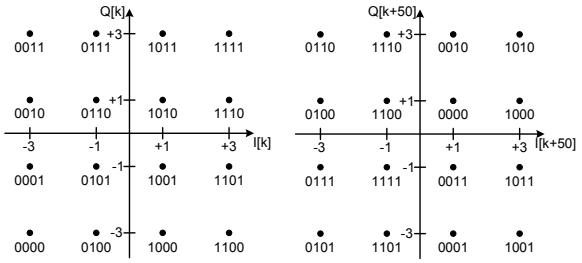

Fig. 3. DCM QAM Constellation

rate of 480 Mbps, the puncturer generates the 3/4 code rate according to table I. To apply code rate 3/4 to the highest data rate the puncturer removes five bits from nine (see figure 2).

The punctured data is then mapped. For data rates of 480 Mbps, data sub-carriers of an OFDM symbol are modulated, using dual carrier modulation (DCM). DCM employs a 16 QAM (quadrature amplitude modulation) technique to encode four bits. By mapping the same four bits in two different sub-carriers of the same OFDM symbol, separated by 50 sub-carriers, DCM adds frequency diversity and thereby reduces the impact of frequency-selective fading.

The input sequence bit b[i], where i = 0, 1, 2, ..., at the input of the DCM modulator is divided into groups of 200 bits. The 200 bits are grouped into 50 groups of 4 bits, reordered according to equation (1), where k $\in$ [0, 49] and g(k) is represented by equation (2). The DCM symbol holds four bits.

$$(b[g(k)], b[g(k)+1], b[g(k)+50], b[g(k)+51]) \quad (1)$$

$$g(k) = \begin{cases} 2k & \text{if k} \in [0, 24], \\ 2k+50 & \text{if k} \in [25, 49]. \end{cases} \quad (2)$$

Each group of 4 bits is converted into two complex numbers d[k] = I[k] + jQ[k] and d[k + 50] = I[k+50] + jQ[k+50], representing the points of two 16 QAM constellation (see figure 3). The complex numbers are normalized, using a normalization factor $1/\sqrt{10}$. The two DCM symbols (d[k], d[k + 50]) are allocated to two individual subcarriers with 50 subcarriers separation. The symbols are then transformed into the time-domain using an Inverse FFT (IFFT).

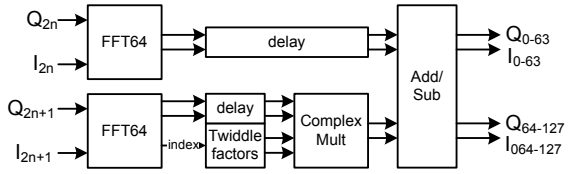

Fig. 4. Parallel FFT implementation

The receiver performs the inverse operations (see figure 1). DCM symbols are recovered from the time domain using an FFT. Two complex numbers transmitted over two sub-carriers separated by 50 sub-carriers are combined to demap four soft bits (equations (3-6)) [7].

$$b_1 = 2I_{R_n} + I_{R_{n+50}} \quad (3)$$

$$b_2 = I_{R_n} - 2I_{R_{n+50}} \quad (4)$$

$$b_3 = 2Q_{R_n} + Q_{R_{n+50}} \quad (5)$$

$$b_4 = Q_{R_n} - 2Q_{R_{n+50}} \quad (6)$$

Bits b_1, b_2, b_3 and b_4 correspond to the groups of four bits generated at the transmitter (see equations (1) and (2)).

Demapped symbols are depunctured. The depuncturer considers omitted bits to be zero and recovers the original encoded rate of 1/3 following the inverse process illustrated in figure 2. Finally, the stream of soft bits is decoded with a VD.

III. MB-OFDM Receiver Implementation

The MB-OFDM receiver contains four main blocks: FFT, demapper, depuncturer and Viterbi decoder. In the following sections, we describe the design of each of these blocks to guarantee a data rate of 480 Mbps.

A. Fast Fourier Transform

An OFDM symbol has a bandwith of 528 Mhz and, therefore, arrives into the 128-FFT every 242.42ns. The required clock frequency for 128-FFT is 528 MHz. To achieve this time requirement for OFDM demodulation, we have parallelized the FFT calculation, decomposing the 128-point input in two 64-point sequences and process them in parallel with two 64-point FFT blocks. The outputs of these blocks are then combined with a radix-2 butterfly to obtain the final values, maintaining continuous data processing (see figure 4).

As illustrated in the figure, even and odd index data are computed separately and then combined with radix-2 block to compute the final values. The real and imaginary twiddle factors are stored in RAMs. The index signal, from one of the 64-point FFT block is used to address twiddle factor values for the complex multiplication.

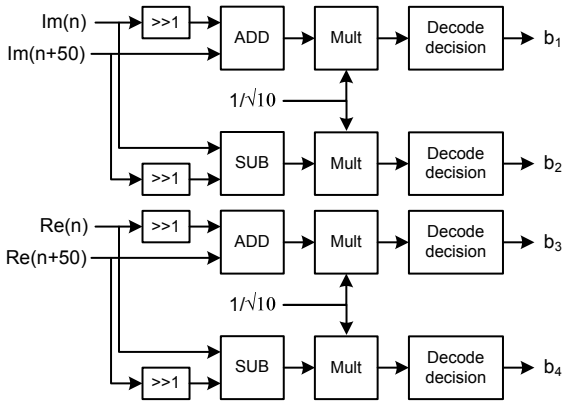

Fig. 5. DCM demapper implementation

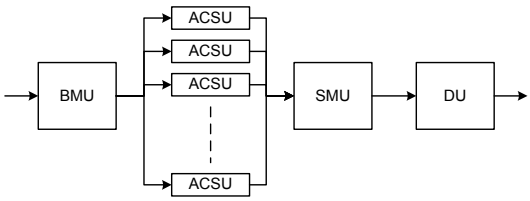

Fig. 6. Architecture of a parallel Viterbi Decoder

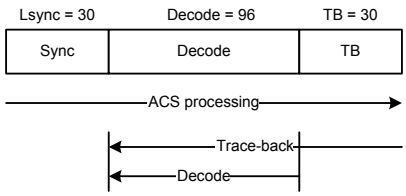

Fig. 7. Sliding Block Method for Viterbi Decoder

B. DCM Demodulator and Depuncturer

The DCM demodulator receives serially 100 complex numbers (guard and pilot symbols are not processed by the DCM demodulator) from the FFT distributed in two groups of complex numbers: one group covers the complex numbers from 0 to 49 (I[k] + jQ[k]) and the other from 50 to 99 (I[k+50] + jQ[k+50]). The demapper combines two complex numbers separated by 50 sub-carriers at a time to demap groups of four soft bits (equations (3-6)) (see figure 5).

The values generated by the demapper are between -1.6 and 1.6. These complex numbers are normalized, using a factor $1/\sqrt{10}$ generating values between -5 and 5. Fixed point values at the outputs of the multipliers are then converted to soft bit numbers with three bits by the decode decision blocks.

The groups of four soft bits are then depunctured and converted into a set of nine soft bits following the puncturing process (see figure 2 in section 2). The bits removed in the transmitter are now assumed to be ”000” (the most uncertain value in the soft bit generation).

C. Viterbi Decoder

The Viterbi decoder (VD) generates one bit for each three soft bits. The Viterbi decoder finds the path in the trellis diagram whose sequence of output symbols best matches the received sequence. Its functionality is implemented by four functional units: the branch metric unit (BMU), the add-compare select unit (ACSU), the survivor memory unit (SMU) and the decision unit (DU) (see figure 6). The BMU calculates the distance (metric) between the received noisy symbol and the output symbol of the state transition (branch). The ACSU computes the accumulated metric associated with the sequence of transitions (path) to reach a state (the Viterbi decoder has $2^{(CL-1)}$ states, where CL is the constraint length of the convolutional coder). The SMU stores the information that permits to trace-back from a state to the previous one. Trace-back processing is managed by the decision unit (DU). The MB-OFDM specification considers the utilization of puncturing which forces theoretical trace-back lengths of at least $12 \times CL$.

To achieve high data rates, the sliding block technique proposed in [3] was combined with the Viterbi algorithm in [4] reducing the decode of a continuous input stream to independent decoding of overlapping blocks.

The method is based on two main observations. One is that the survivor paths from all possible starting states merge with high probability L iterations back into the trellis. The parameter is the survivor path length and is typically $5 \times (CL-1)$ [6]. The other is that when starting with unknown initial state metrics, the state metrics after K trellis iterations are independent of the initial metrics. The parameter is the synchronization length and is also typically $5 \times (CL-1)$ [5].

Based on this characteristics, a state at time *n* can be decoded using only information from the interval n - K to n + L. Generally, K = L is considered. The sliding block VD method applies the Viterbi algorithm to the interval [n - M/2 - L, n + M/2 + L] to decode the interval [n - M/2, n + M/2]. To guarantee correct initial and final states, the method considers an initial synchronization block and a final trace-back block without decoding for each parallel VD (see the behavior of each VD for the SBVD in figure 7).

With this technique, we can use independent parallel VD to increase the decoding rate (DR). The DR of an SBVD is:

$$DR = nVD \times \frac{M}{M+2L} \times drVD$$

where *nVD* is the number of parallel VD and *drVD* is the decoding rate of a single VD. The bit error rate of the decoding process of the VD depends on the length of the initial synchronization step, the length of the final trace-back block and of the trace-back decode length (see [8] for a detailed study about using SBVD to achieve DCM data rates in MB-OFDM UWB technology).

In our particular case, to achieve a decoding rate of 480 Mbps, we will use three parallel Viterbi decoders.

D. UWB Receiver

The final implementation of the UWB receiver contains the four blocks described previously and three FIFOs between the depuncturer and the SBVD for synchronization (see figure 8.

The softbits generated by the depuncturer are sent in order to the parallel VD whose outputs are then multiplexed to generate the final decoded bit stream.

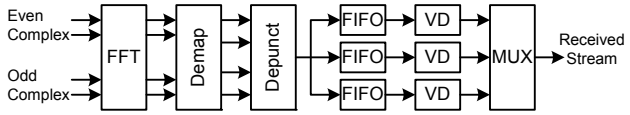

Fig. 8. Block diagram of the UWB receiver

IV. Results

To test the receiver, a complete UWB transceiver was implemented including an AWGN (*Additive White Gaussian Noise*) channel between the transmitter and the receiver. An AWGN generator was used to emulate an AWGN channel by adding noise to transmitted streams. The amplitude of the noise is programmed according to emulated real noise conditions. Hence, we can emulate an AWGN channel in which signals are transmitted with different SNRs. We have implemented an AWGN generator in hardware based on the polar method ([9]) and a Mersenne twister [10] for pseudorandom number generation of 32-bit word length numbers.

The system was designed and implemented with the following configurations:

- Data rate = 480 Mbps; Code rate (CR): 3/4;
- Soft bits: 3 bits;
- Synchronization length (SL): 40;
- Trace-back length (TB): 120.

The configuration of the SBVD follows the results in [8] to guarantee that there is no BER degradation compared to the BER obtained with a single VD ignoring the data rate.

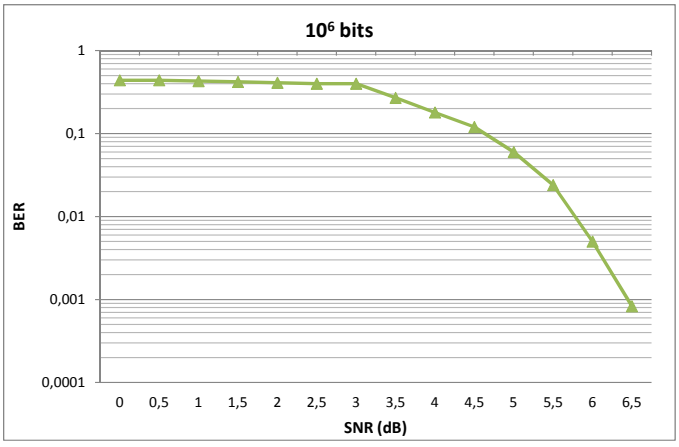

Fig. 9. BER curve of the designed UWB receiver

The hardware implementation of the testing scenario was simulated (see figure 9) and the results compared with a MATLAB simulation of the UWB transceiver fully confirming the correctness of the hardware implementation of the receiver (MATLAB results are not shown because they are practically identical to those from the FPGA implementation).

The MB-OFDM receiver was synthesized, placed and routed in a Virtex-6 FPGA (speed grade -2) using ISE13.4 from Xilinx (see implementation results in table II).

TABLE II. Implementation results

Block	LUTs	BRAM	DSP48	Freq
FFT	4140	2	12	321 MHz
Demapper+Depuncturer	108	0	0	700 MHz
SBVD	7905	3	0	285 MHz
Complete Receiver	**12601**	**5**	**12**	**275 MHz**

The system operates at a frequency of 275 MHz enough to decode a data rate of 480 Mbps due to the parallel processing of data at the FFT and at the SBVD. The most demanding block is the SBVD followed by the FFT. The complete system utilizes about 25% of the smallest Virtex-6 FPGA.

V. Conclusion

The highest data rate of the MB-OFDM UWB technology can be achieved in a Virtex-6 FPGA using about 25% of the reources of the smallest Virtex-6 FPGA. The most demanding block in terms of resources and timing requirements is the Viterbi decoder. A sliding block Viterbi decoder was used with three parallel Viterbi decoders to achieve a data rate of 480 Mbps. We have configured the SBVD so that the BER is not degradated due to the use of the sliding block technique.

The system is being completed with the integration of the scrambler and the interleaver at the transmitter and the de-scrambler and de-interleaver at the receiver. Preliminary results indicate that these blocks do not influence the operating frequency of the circuit and consume a small percentage of those consumed by the four reported blocks.

Acknowledgment

This work was supported by national funds through FCT Fundação para a Ciência e Tecnologia, under project PEst-OE/EEI/LA0021/2011.

References

[1] High Rate Ultra Wideband PHY and MAC Standard, ECMA International ECMA-368, 2nd Edition Dec. 2008.

[2] Gamba D., "Using FPGAs in Wireless Base Station Designs", Xcell Journal, Issue 52 Xilinx, spring 2005.

[3] Tzu, K.-H. and Dunham, J., "Sliding Block Decoding of Convolutional Codes", in IEEE Transactions Communication, vol. COM-29, pp. 1401-1403, Sep.-1981.

[4] P. Black and T. Meng, "A 1-Gb/s, Four-State, Sliding Block Viterbi Decoder", IEEE Journal of Solid State Circuits, Vol. 32, No. 6, June 1997.

[5] A. J. Viterbi and J. K. Omura, Principles of Digital Communication and Coding. McGraw-Hill, 1979, pp. 258-261.

[6] G. C. Clark and J. B. Cain, "Error-Correction Coding for Digital Communications", New York: Plenum, 1981, pp. 227-264.

[7] R. Yang, R. S. Sherratt and O. Cadenas, *FPGA Based Dual Carrier Modulation Soft Mapper and Demapper for the MB-OFDM UWB Platform*, Proc. of the Annual Postgraduate Symposium, June 2007, Liverpool.

[8] Mário Véstias and Helena Sarmento, "Tradeoffs in the Design of Sliding Block Viterbi Decoders for MB-OFDM UWB Systems", in International Conference on Consumer Electronics, September 2012, pp. 173-177.

[9] A convenient method for generating normal variables, G. Marsaglia and T. A. Bray, SIAM Rev. 6, 260-264, 1964

[10] Matsumoto, M.; Nishimura, T. (1998). "Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator". ACM Transactions on Modeling and Computer Simulation 8 (1): 3-30