SYNCHROINFO JOURNAL

www.media-publisher.eu/synchroinfo-journal

DESIGN AND MODELING OF UFMC SYSTEMS

Anastasia V. Ermakova[1], Vu Sy Dao[2]

[1] Moscow Technical University of Communications and Informatics, Moscow, Russia;
msikisylia@gmail.com

[2] Le Quy Don Technical University, Hanoi, Vietnam

ABSTRACT

This paper presents a step-by-step design and implementation of the transmitter section of a universal filtered multicarrier modulation (UFMC) system targeted for use in next-generation wireless communication systems. The development process includes four sequential stages: mathematical modeling and simulation of the UFMC system in MATLAB, system-level modeling in Simulink, hardware-oriented design using the Xilinx System Generator, and design verification using hardware co-simulation. The paper describes in detail the main functional blocks of the UFMC transmitter, including random data generation, QPSK modulation, serial-to-parallel conversions, zero padding, inverse discrete Fourier transform, digital upconversion, and subband filtering. Special attention is paid to the implementation of the system on a Xilinx Spartan-6 FPGA, taking into account the limitations of hardware resources and fixed-point representation accuracy. An elliptic filter and a Chebyshev filter of the second kind are studied for subband formation. Simulation results obtained in MATLAB and Simulink show that the use of an elliptic filter provides better spectral localization and more effective sidelobe suppression compared to a Chebyshev filter. These results confirm the practical applicability of the proposed UFMC architecture and its potential for further hardware implementation in 5G wireless communication systems.

DOI: 10.36724/2664-066X-2026-12-1-15-30

Received: 20.11.2025
Accepted: 27.01.2026

Citation: A.V. Ermakova, V.S. Dao, "Design and modeling of UFMC systems," *Synchroinfo Journal* **2026**, vol. 12, no. 1, pp. 15-30.

KEYWORDS: *UFMC; multicarrier modulation; 5G communication systems; digital signal processing; FPGA; MATLAB; Simulink; Xilinx System Generator; QPSK modulation; subband filtering; elliptic filter; Chebyshev filter; fixed point; hardware cosimulation.*

Licensee IRIS, Vienna, Austria.

1 Introduction

Modern fifth-generation wireless communication systems place increased demands on spectral efficiency, resistance to intersymbol interference, and flexibility in radio resource allocation. Traditional orthogonal frequency division multiplexing (OFDM), widely used in LTE networks and early versions of 5G, offers several advantages, including ease of implementation and compatibility with multiple-input multiple-output (MIMO) technologies [10-13]. However, OFDM is characterized by high spectral sidelobes and sensitivity to synchronization errors, limiting its effectiveness when transmitting short packets and in dense wavelength-division multiplexing (WDM) environments.

As an alternative to OFDM, new signal forms such as filtered bank multicarrier (FBMC) and universal filtered multicarrier (UFMC) [1-5] have been actively explored in recent years. UFMC is considered a compromise between OFDM and FBMC, combining the simplicity of OFDM with the improved spectral characteristics of FBMC. In UFMC, filtering is performed at the subband level, which significantly reduces filter length, improves spectral localization, and reduces intersymbol interference without significantly increasing receiver complexity.

Despite the potential of UFMC, practical implementation of this technology requires careful consideration of the transmitter architecture and the limitations of hardware platforms such as field-programmable gate arrays (FPGAs) [14-17]. Particularly important are the selection of a filter prototype, optimization of the fixed-point representation, and balancing system performance with hardware resource utilization.

The objective of this work is to develop and implement, step-by-step, the transmit portion of a UFMC system using MATLAB and Simulink modeling tools, as well as the hardware-specific Xilinx System Generator tools [18-20]. This study examines in detail the UFMC modulation processes, from digital data generation to subband filtering, and also provides a comparative analysis of an elliptic filter and a Chebyshev filter of the second kind [6-9]. The obtained simulation results confirm the effectiveness of the proposed approach and substantiate the possibility of further implementation of the system on FPGA for use in next-generation wireless communication networks.

2 FPGA design process

To design and implement a UFMC system, four stages are followed according to the design flow. At each stage, the UFMC system is modeled and simulated to obtain simulation results that ensure high performance before moving on to the next stage. The first two stages of the design process are modeling and simulation in MATLAB and Simulink. The UFMC system design process consists of four stages. The first stage is to model and simulate UFMC systems with different filters in MATLAB. Based on the mathematical foundation of the UFMC system described in Chapter 3, two UFMC systems are modeled in MATLAB using LTE parameters, where the two systems are tested in a fading channel to demonstrate the differences between the UFMC systems.

In the second stage, the UFMC system is modeled using LTE downlink parameters and simulated in Simulink. Unlike in MATLAB, many parameters, such as the sampling period, must be defined in Simulink. Because the UFMC model has different filters, the OFDM-based design is divided into multi-rate subsystems. Simulink is essentially a MATLAB graphical extension for modeling and simulating a multi-rate system with different simulation time steps. Using the Simulink communications library, the UFMC system is modeled based on a MATLAB model. Each Simulink block represents a mathematical formulation already implemented in MATLAB. This Simulink UFMC model will serve as the basis for a Xilinx-based implementation by replacing Simulink blocks with Xilinx blocks. However, some Simulink blocks, such as the data source and receivers, are still required for hardware co-simulation.

Designing a UFMC system using Xilinx blocks is the third stage of the design process. In this step, Simulink blocks are replaced with Xilinx blocks. Some Xilinx subsystems are designed to have the same functionality as Simulink blocks. Using this representation leads to a tradeoff between system performance and the size of the Xilinx UFMC design, as the number of dedicated bits affects system performance. The greater the number of dedicated bits, the more complex the hardware design. The number of dedicated bits is chosen to ensure the hardware size is suitable for implementing a high-performance Xilinx Spartan-6 FPGA.

The final step of the design process is validating the Xilinx-based design using co-simulation. To perform tasks in the fields of digital communications and digital signal processing, MATLAB software was effectively used for modeling, testing, and evaluating system performance. The UFMC system design must be tested before it can be implemented to avoid any design errors. Therefore, our UFMC system is modeled in MATLAB and Simulink before the VHDL code is generated and loaded onto the Spartan-6 board for implementation. To better understand the signal modulation processes, this chapter describes and discusses each stage of the UFMC modulation process in detail. Since the receiver performs the inverse processes to the transmitter, only the transmitter blocks are described in detail.

As part of the System Editions ISE® Design Suite, the Xilinx System Generator for Digital Signal Processing is a Simulink plug-in design tool that enables designers to design, simulate, and develop high-performance digital signal processing systems. This design tool consists of a set of Xilinx blocks in the form of a Simulink library that can be mapped directly to target FPGA hardware. Using the Xilinx System Generator for Digital Signal Processing, you can not only design and simulate digital signal processing algorithms but also generate synthesizable hardware description language (HDL) code using the Xilinx FPGA coder. Therefore, the Xilinx System Generator for Digital Signal Processing is considered a high-level design tool for high-performance digital signal processing systems, providing system simulation and automatic code generation from MATLAB or Simulink. Using Xilinx block libraries in the Simulink model-based design environment does not require previous experience with register-level design and Xilinx FPGAs, as subsequent FPGA implementation steps, such as place, route, and synthesize, are automatically performed to generate the FPGA bitmap. The Xilinx System Generator also provides hardware co-simulation functionality when the model-based design is executed in Simulink and the FPGA. Chapter 4 describes how MATLAB and Simulink were initially used to simulate a QPSK-UFMC system, including a transmitter, channel, and receiver. In this chapter, the Simulink model is converted into hardware blocks using the Xilinx System Generator. The Xilinx design is then synthesized, mapped, and routed to generate the bitstream that will be loaded into the FPGA. When hardware co-simulation is performed, MATLAB actually communicates with the FPGA through internal RAM. An overview of each block used to design the transmitter, receiver, or wireless channel is provided, describing its functions and characteristics. Xilinx UFMC transmitter, wireless fading channel, and receiver blocks are shown respectively.

3 UFMC signal generation in Xilinx System Generator

The UFMC transmitter is designed as shown in Figure 1.

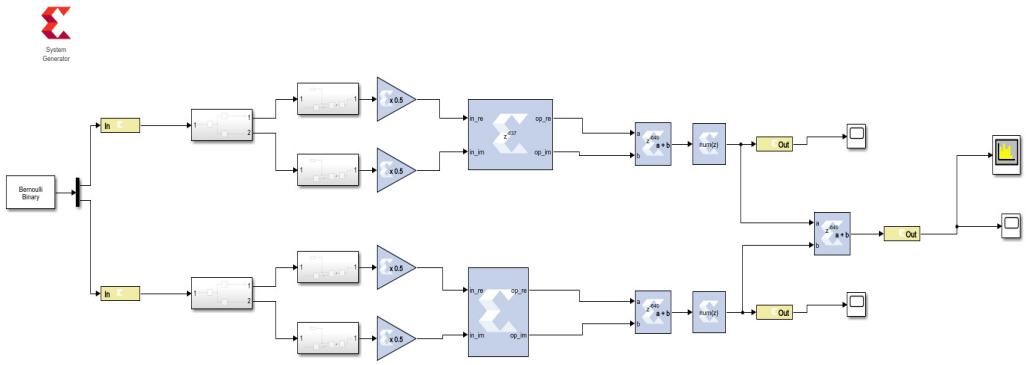

Figure 1. Block diagram of the Xilinx UFMC dual-band transmitter

Data is generated from the Simulink environment and transferred to the Xilinx design. The digital signal is modulated with a QPSK baseband modulator before being modulated with an FFT. Typically, the wireless channel model is applied to the signal after it has been upconverted.

To apply a multipath fading channel to a signal consisting of I and Q components, the components must be separated as I and Q signals for complex multiplication. For this reason, the multipath fading channel in our design is used before the I and Q signals are modulated. This makes the design more efficient and less complex.

4 Generating random data

In the first stage, a digital random data matrix of size m × n is generated with the condition that it should have only scalar values 0 or 1 with the same probability to be similar to the original digital signal in digital communication. To determine the matrix size, many factors must be taken into account, such as the number of UFMC symbols, the FFT size, the number of zero-padded symbols, and the modulation method. A large input matrix size may lead to a shortage of memory on the computer, depending on the computer RAM. In our design, the FFT is 512, the number of UFMC symbols is 105, and QPSK modulation is used. Assuming that 256 zeros are padded in one UFMC symbol, the number of bits per UFMC symbol is represented as (512 - 256) * k = 512, where k = 2 is the modulation index for QPSK.

For simplicity, the number of columns is assumed to be 1, so the number of rows is $6*10^7$. In fact, this definition of the random matrix size should ensure that the number of QPSK symbols, after zero padding, can be converted to a multiple of the FFT size without remainder. However, the number of UFMC symbols per matrix can be as small as one integer, but achieving this number of UFMC symbols requires generating multiple matrices.

In model-based design, random bits are generated using the Bernoulli binary generator provided by the Communications System Toolkit library. Input data is generated with a zero probability of 0.5 and a sampling rate of 20 Mbps. The Bernoulli Binary Generator output is sample-based. To convert Simulink integer, fixed-point, or double input data types to the System Generator fixed-point type, the Gateway-In block is used as an input to the Xilinx portion of the Simulink model. When converting from a floating-point to a fixed-point data type, the Gateway-In block uses some options for overflow and quantization. In the event of an overflow, it can be used to saturate or wrap the input value or mark it as an error. Saturation is essentially the process of converting an overflowed value to its largest positive or smallest negative value. Wrapping involves discarding the overflowed value, that is, the bits to the left of the most significant bit. Marking is only intended to indicate an error whenever an overflow occurs. The Gateway-In block also provides quantization options: either rounding the value to the nearest representable value or truncating it by discarding the bits to the right of the least significant bit.

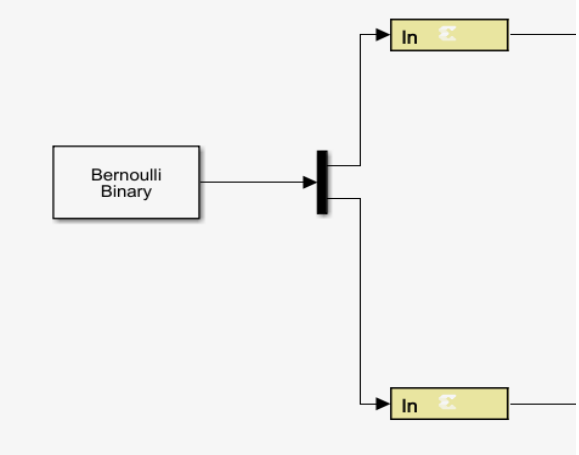

Figure 2. Random data generation blocks

5 QPSK Modulation

QPSK modulation was used for the subcarrier modulation. Every two consecutive bits of the original digital data are mapped to a corresponding QPSK symbol, which differs in phase angles of 0, 90, 180, or 270 degrees. Each group of two bits is encoded along the in-phase or quadrature axis.

Therefore, each QPSK symbol can be represented as a complex number S_I+jS_Q if a constellation is used for mapping. In our implementation, a constellation with zero phase shift is used. The original data matrix is modulated into a matrix of complex numbers in the form of QPSK symbols, and the size is equal to half the size of the original matrix.

As shown in the figure below, the modulator is implemented as a combination of two ROMs and a serial-to-parallel converter [10]. The serial-to-parallel block accepts serial unsigned data, represented as a fixed-point representation of a single bit with a binary digit of zero, and produces a single output from two consecutive input bits. In other words, it combines every two bits that will later be mapped in ROM to the corresponding QPSK symbol. Serial input is ordered with the most significant word first.

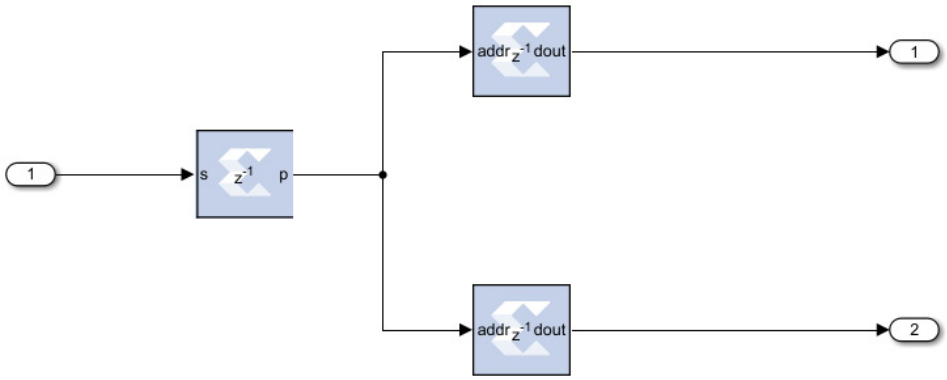

Figure 3. Xilinx Modulator

The Xilinx block ROM is a single-port read-only memory that stores four words corresponding to "00," "01," "10," and "11." The word values are specified in the block parameters as a seed vector. Since a QPSK symbol consists of quadrature and in-phase values, two ROMs are used for both the I and Q channels. Both have the same input and depth, but they differ in the seed vector corresponding to the gray-coded QPSK constellation. To reduce the size of the fixed-point representation, the QPSK symbol power is not normalized. Since the Xilinx design will be loaded into an FPGA, the design area is very important and should be as small as possible. This area is affected by the number of used bits in the fixed-point representation. Therefore, the in-phase and quadrature QPSK values are set to 1 or -1 to reduce the number of bits for their fixed-point representation.

6 Serial to parallel conversion

Serial data is converted to parallel data. The modulated matrix S, as the input matrix, is transformed into a 256-column matrix. In this case, each row of 256 QPSK symbols is considered parallel data. These 256 QPSK symbols are grouped for modulation to create a UFMC symbol. The actual purpose of the shape change is to form an S-matrix that is ready for UFMC modulation using an IFFT.

For better performance, the frequency spacing between subcarriers in the UFMC frequency domain can be reduced as the sampling rate increases. To increase the sampling rate, interpolation is used by appending zeros to the end of the original data sequence. As zeros are appended to the signal, the number of samples in the time domain increases, which also increases the FFT size. By expanding the FFT samples, the UFMC symbol will have a higher resolution, which is required for digital signal processing such as digital-to-analog and analog-to-digital conversion. The interpolation process must comply with the Nyquist sampling theorem to avoid aliasing, which can occur in the frequency domain. Therefore, the Nyquist frequency must be at least twice the highest frequency of the sampled signal [11]. Since the number of samples for the FFT is 256, the number of appended zeros must be at least 256 to comply with the Nyquist theorem. To ensure that non-zero data is mapped to subcarriers near the zero frequency, and zero data is mapped to subcarriers with high positive/negative frequencies, these zeros must be padded in the middle of each parallel IFFT data input.

Zero padding in the Xilinx design is not used for the same purposes as in MATLAB and Simulink. In fact, zero padding is used for upsampling and creating time space between groups of QPSK symbols to form UFMC symbols. These spaces are actually used to add a cyclic prefix. To insert a cyclic prefix into a UFMC symbol using the Xilinx IFFT block with a pipelined streaming I/O implementation, the time space must be equal to the length of the cyclic prefix to avoid data loss during transmission.

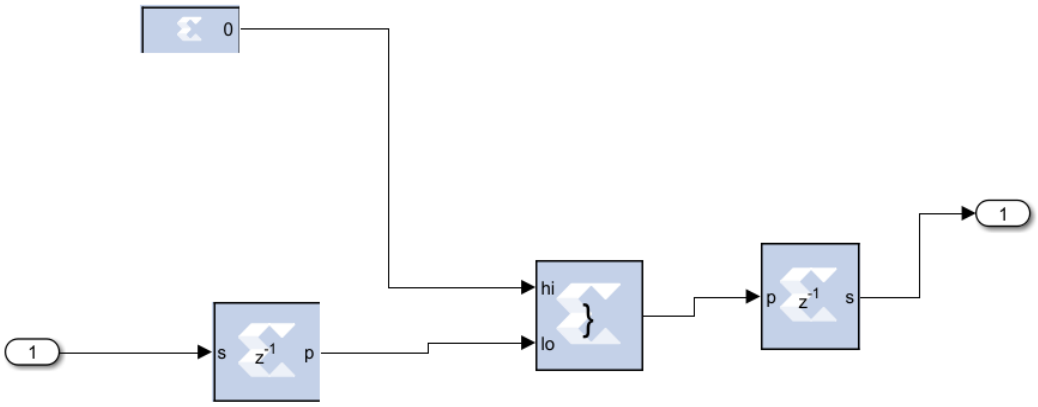

Figure 4. Xilinx Zero-Fill Design

As shown in Figure 4, the zero-padding design consists of Xilinx serial-to-parallel and parallel-to-serial converters, constant blocks, and constant blocks. The first serial-to-parallel block groups 1024 consecutive bits, which represent 512 QPSK symbols, with the most significant word coming first, to create a single unsigned output. A 256-bit unsigned fixed-point representation is used to represent the zero value as the constant block output. In fact, the number of unsigned fixed-point bits is related to the length of the cyclic prefix. Therefore, the time interval between OFDM symbols required to insert the cyclic prefix is equal to the length of 128 QPSK symbols. When the in-phase and quadrature-phase QPSK values are represented by two bits, 256 bits are required for each of the I and Q values to represent 128 QPSK symbols.

The unsigned outputs of the constant and serial-to-parallel blocks are combined to create a single unsigned integer bit vector. Data input to the upper port of the concat block occupies the most significant bits of the output, while data input to the lower port occupies the least significant bits [12]. Therefore, the output of the concat block is an unsigned integer represented by a 1280-bit vector. This process of combining two inputs with the same data rate creates a single output with a higher data rate. In the parallel-to-serial block, the input word is divided into 640 parts, so every two bits form one signed word with a binary position of zero and least significant bit order first. As a result of zero padding, the output data rate is higher than the input rate by a ratio of 5/4.

7 Inverse Discrete Fourier Transform

To generate multiple orthogonal subcarrier signals overlapping in spectrum, the discrete Fourier transform and inverse discrete Fourier transform processes should be used. In MATLAB, the FFT and IFFT were used to implement the DFT and IDFT processes. In fact, the FFT function in MATLAB uses a combination of several algorithms, including Cooley-Tukey [12] and prime factorization algorithms [13].

To use time-decimation, the number of IFFT points N must be an integer of power 2, and therefore the modulated matrix is transformed to have 256 columns and then filled with zeros to have 512 columns, which meets the requirement of time-decimation algorithms. As a result, the transpose of the IFFT output matrix is represented as a matrix X. Each row of matrix X represents a UFMC symbol, which is periodic with a period of 512 samples. The Xilinx FFT block supports the Virtex-5 and other FPGAs, such as the Virtex-7, Virtex-6, Virtex-5, Virtex-4, Spartana-6, and so on. It calculates the forward and inverse DFT of an N-point vector of complex values.

The FFT size can be fixed-point or floating-point numbers. Since the FPGA area is limited, the real and imaginary components are represented in fixed-point with the minimum possible number of bits without affecting system performance. In other words, the number of bits required to represent values is inversely proportional to the FPGA area, since calculating the DFT requires a lot of mathematical calculations.

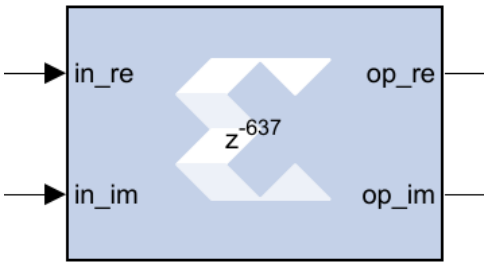

Figure 5. Xilinx FFT block with unscaled output

However, each real and imaginary FFT input can be represented as a 2's complement with a number of bits ranging from 8 to 34 bits inclusive [13]. The FFT block uses either block RAM or distributed RAM. The output order can be either in reverse or natural order. Experience shows that scaling has a significant impact on system performance, and the best way to achieve high performance is to scale the output signal immediately after the FFT block. Using the Xilinx CMult block, scaling can be performed and fixed-point precision can be specified. Since the complex-valued input vector must have a size in the range 8 to 34, two CMult blocks with a scale of 0.5 are applied to the I and Q channels before the FFT block to change the number of bits with fixed-point precision to 8 bits. As with the DFT calculation, the IFFT result must be divided by the IFFT size, which in our case is 512.

The FFT output is scaled to 1/256, since the input is already scaled from 1/2. However, Xilinx constant blocks with constant values 0 and 128 are used to specify the type of forward or reverse FFT operation and determine the length of the cyclic prefix, respectively. Control signals, such as the START, cyclic prefix, and forward-reverse write enable signals, can also be provided by a Xilinx constant block with a true Boolean value.

Using two Xilinx Mult4 multipliers, two sine and cosine digital signals are then multiplied by the quadrature and in-phase signals emanating from the fading channel circuit, as shown in the UFMC transmitter design figure. The Xilinx Mult4 block implements a multiplier that calculates the product of two inputs. Each Mult4 multiplier has a latency of 3 clock cycles as a default setting, but the fixed-point accuracy is determined for domain optimization. The Mult4 block latency means that the Mult4 block requires 3 sample periods to show its output.

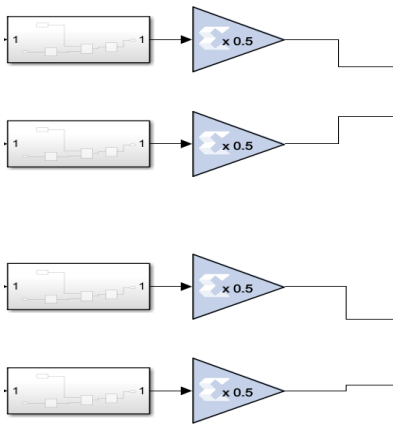

Figure 6. Xilinx I and Q Digital Modulator and Upconverter

As shown in the figure, the output of two Mult4 blocks is fed to a Xilinx AddSub block, which is designed to perform addition operations on the output data of 18-bit fixed-point representation and signed arithmetic type. All Mult4 and AddSub blocks in the I and Q digital modulators are configured for truncation and carry for quantization and overflow operations, respectively. The aforementioned Xilinx design not only combines the I and Q signals into a single channel but also upconverts them at a 25 MHz intermediate frequency with a sampling rate of 100 Mbps. In fact, the sampling period is equal to an integer number of clock cycles, so the maximum sampling rate that can be provided is equal to the board clock frequency.

The finite impulse response filter is one of the most common and fundamental building blocks for UFMC systems on FPGAs. Although its algorithm is extremely simple, the implementation options can be overwhelming and time-consuming for hardware engineers today, especially in filter-heavy systems such as Digital Radios. The FIR compiler reduces filter implementation time with a single click, while also allowing users to find tradeoffs between different hardware architectures for their FIR filter specification. The filter used to implement this system, along with its characteristics, is presented below:

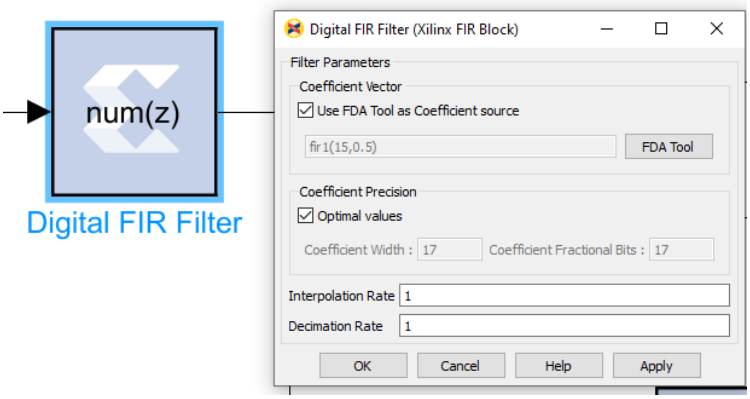

Figure 7. Xilinx FIR filter

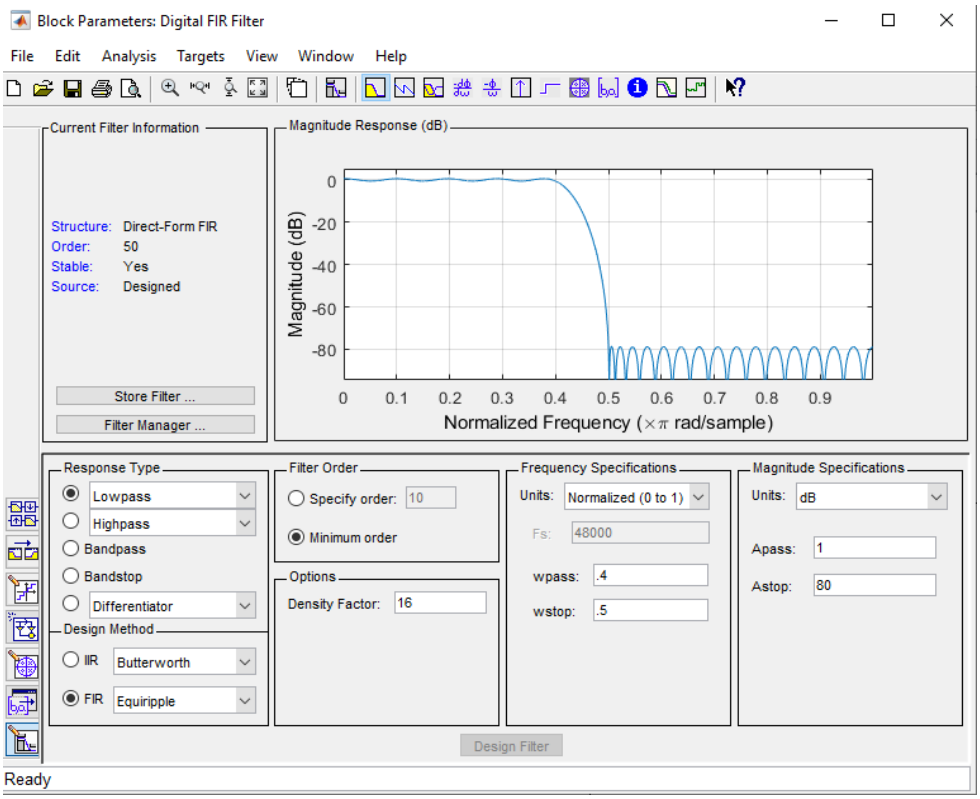

Figure 8. Xilinx FIR Filter Characteristics

As mentioned earlier, the UFMC system is based on the fact that the UFMC time-domain signal is a superposition of subband-filtered signals with a filter of order L and an IFFT length N. When designing the circuit, the choice of filter was an important factor. For modeling our system, we chose an elliptic filter and a second-order Chebyshev filter.

The simulation of the presented circuit using an elliptical filter and the filter characteristics are shown in Figures 9-12.

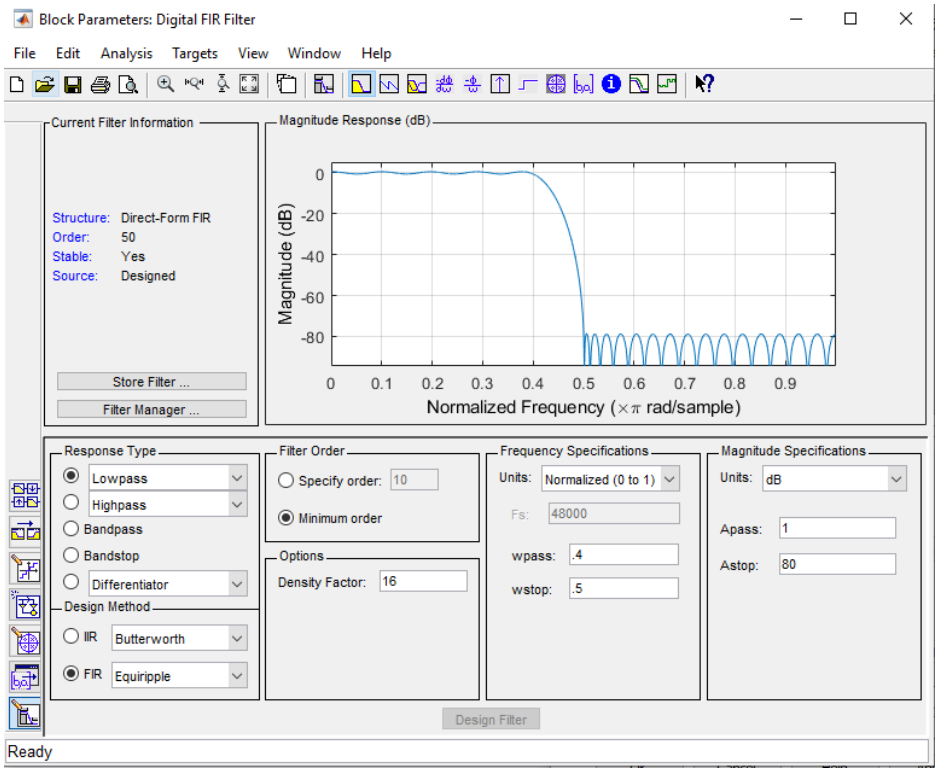

Figure 9. Elliptical FIR filter characteristics

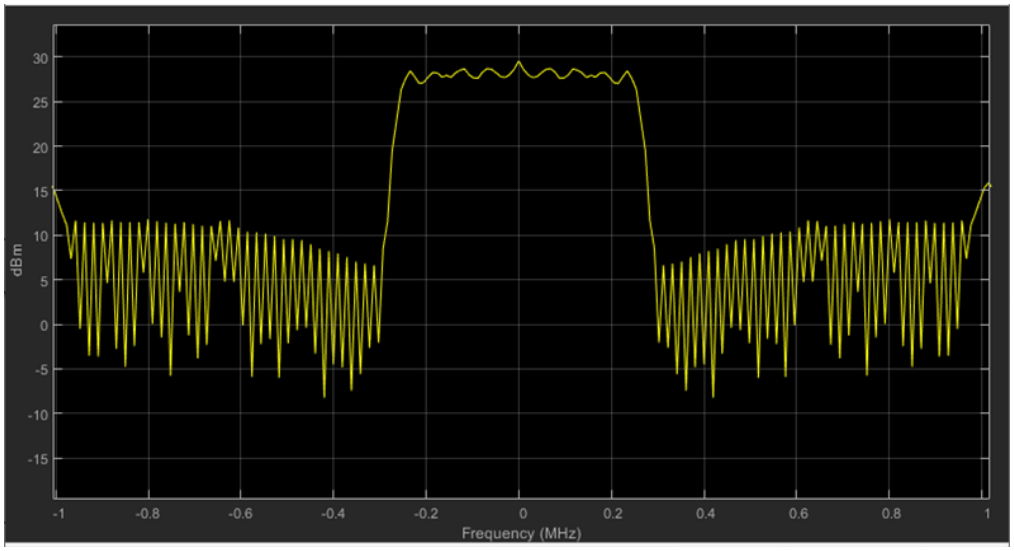

Figure 10. Spectrogram of the UFMC signal in the circuit for two sub-ranges at T=1 with an elliptical filter

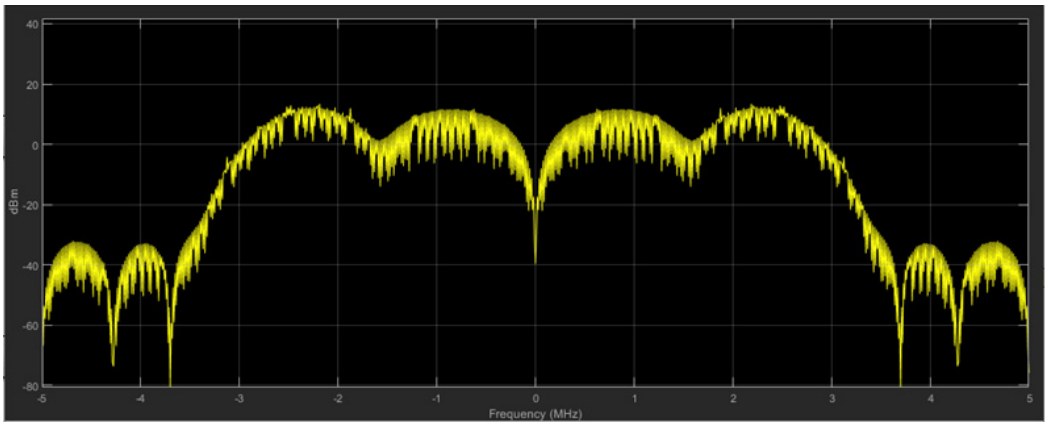

Figure 11. Spectrogram of the UFMC signal in the circuit for two sub-ranges at T=5 with an elliptical filter

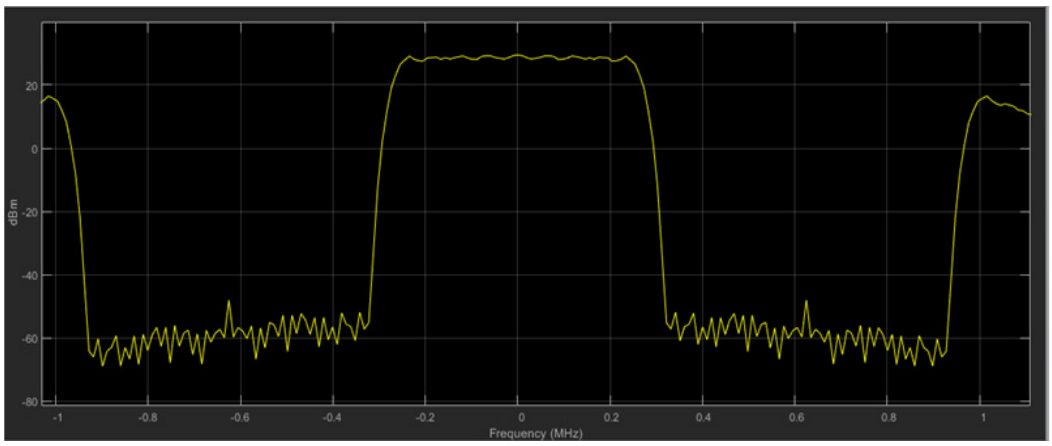

Figure 12. Spectrogram of the UFMC signal in the circuit for two sub-ranges at T=10 with an elliptical filter

The simulation of the presented circuit using a 2nd order Chebyshev filter and its characteristics are shown in Figures 13-16.

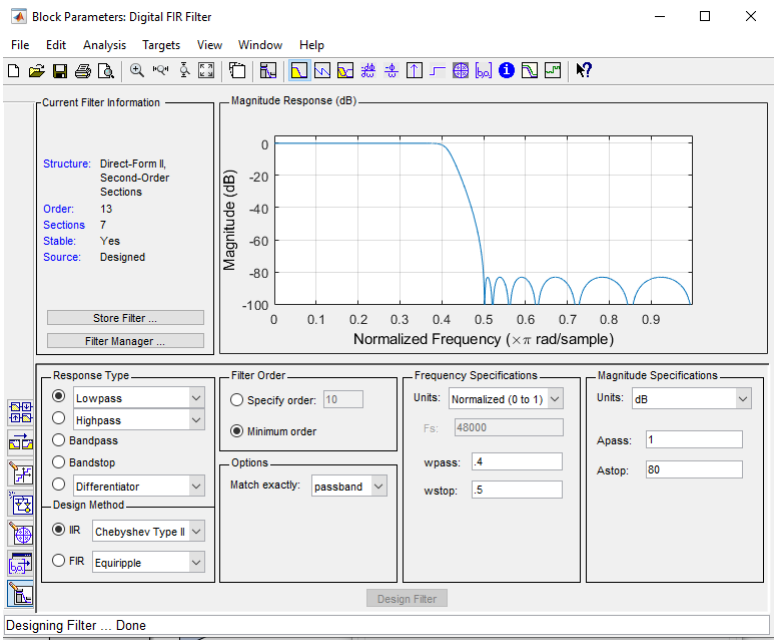

Figure 13. Characteristics of the 2nd order Chebyshev filter

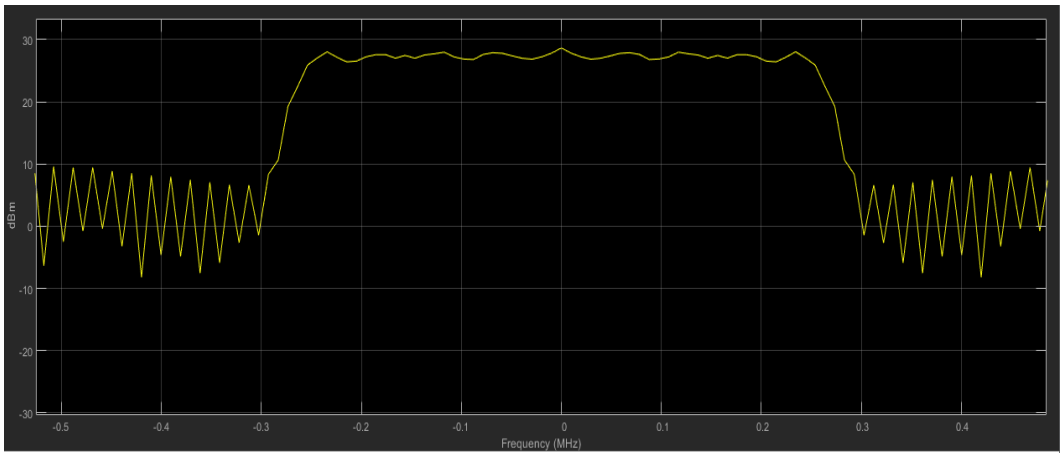

Figure 14. Spectrogram of the UFMC signal in the circuit for two sub-ranges at T=1 with a 2nd order Chebyshev filter

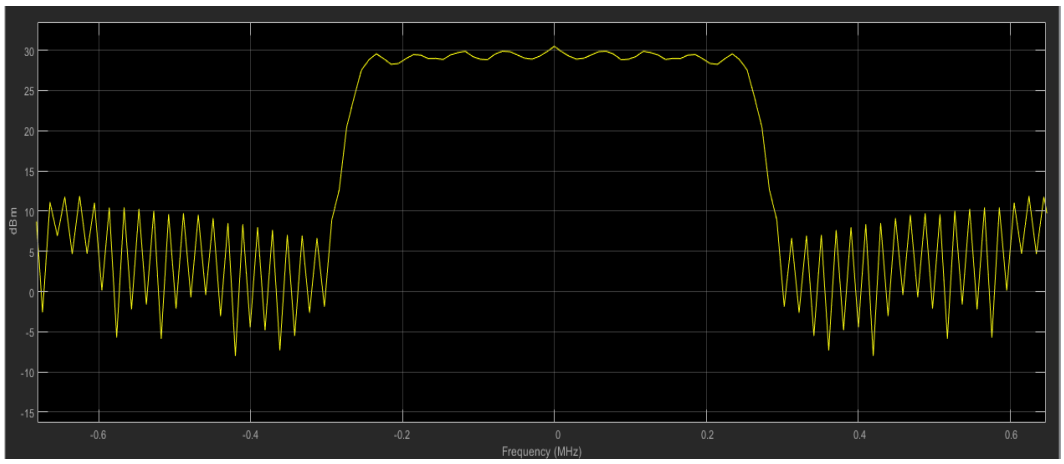

Figure 15. Spectrogram of the UFMC signal in the circuit for two sub-ranges at T=5 with a 2nd order Chebyshev filter

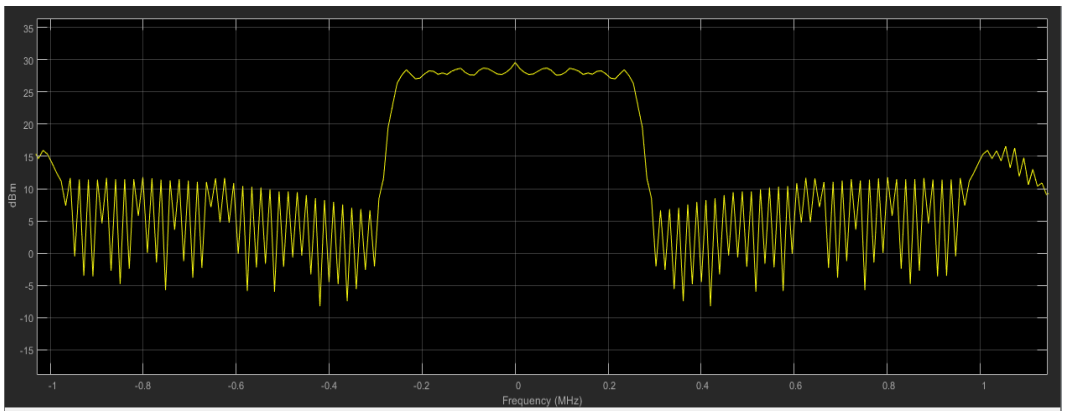

Figure 16. Spectrogram of the UFMC signal in the circuit for two sub-ranges at T=10 with a 2nd order Chebyshev filter

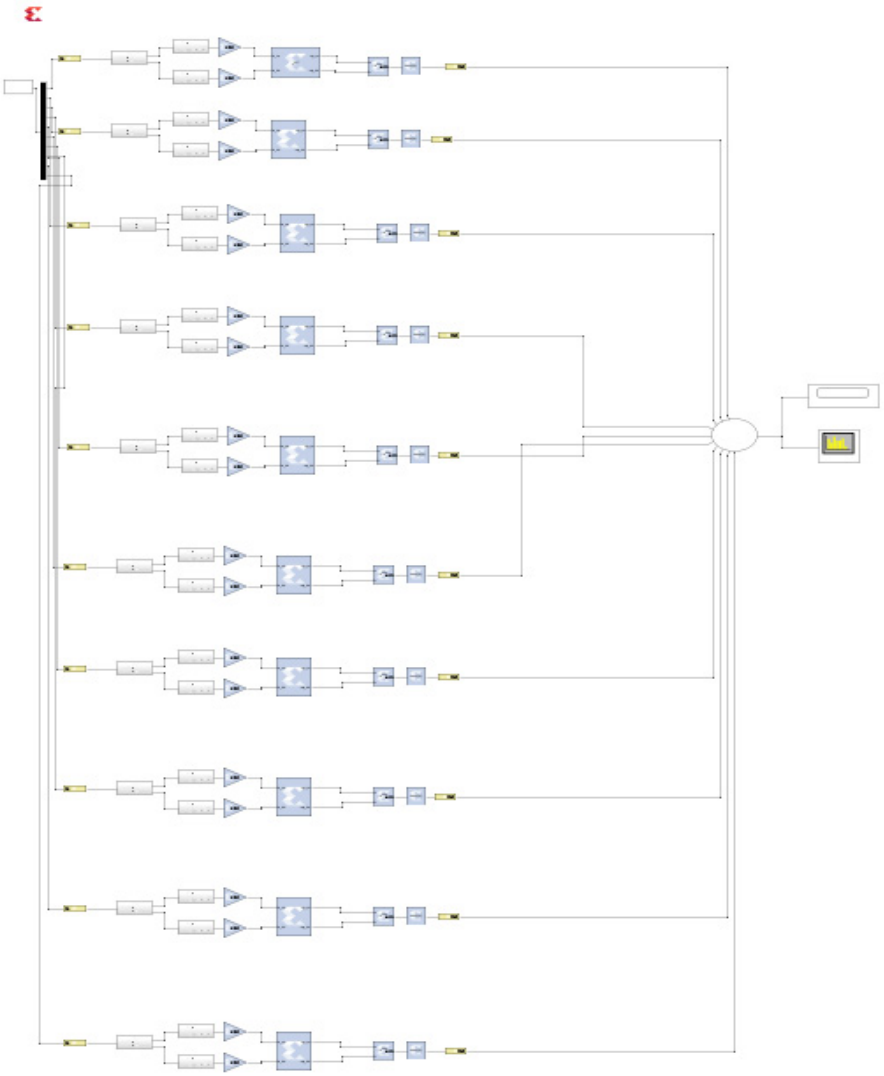

Figure 17. Block diagram of the Xilinx UFMC transmitter for ten sub-bands

Table 1. Filter characteristics in a ten-subband scheme **Filter №1**		**Filter № 2**		**Filter № 3**	**Filter № 4**	**Filter № 5**
Low-pass filter		Bandpass filter		Bandpass filter	Bandpass filter	Bandpass filter
PP	0.05	PZ1	0.045	0.095	0.145	0.195
		PP1	0.05	0.1	0.15	0.2
PZ	0.055	PP2	0.1	0.105	0.155	0.205
		PZ2	0.105	0.155	0.205	0.255
Filter № 6		**Filter № 7**		**Filter № 8**	**Filter № 9**	**Filter № 10**
Bandpass filter		Bandpass filter		Bandpass filter	Bandpass filter	Bandpass filter
0.245		0.295		0.345	0.395	0.445
0.25		0.3		0.35	0.4	0.45
0.255		0.305		0.355	0.405	0.455
0.305		0.355		0.405	0.455	0.505

The simulation of the presented circuit using an elliptic filter is shown in Figures 18-20.

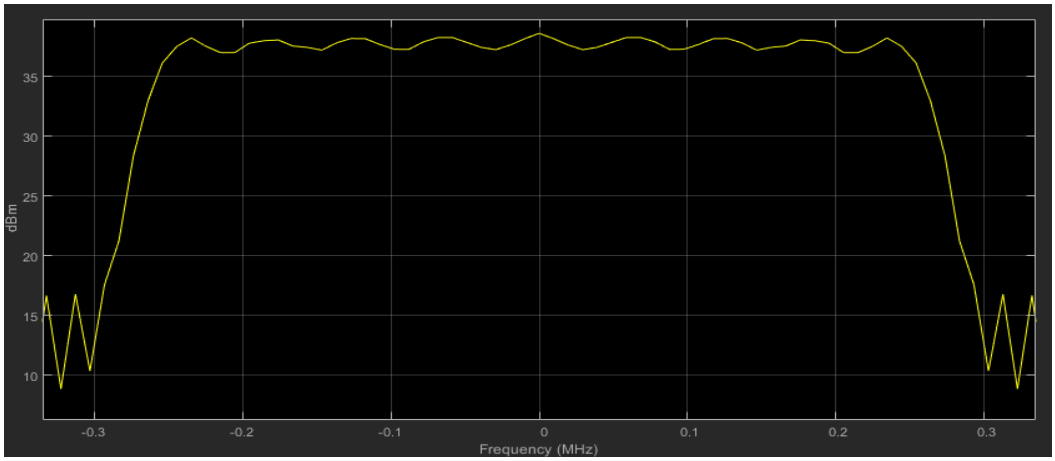

Figure 18. Spectrogram of the UFMC signal in the circuit for ten sub-ranges at T=1 with an elliptical filter.

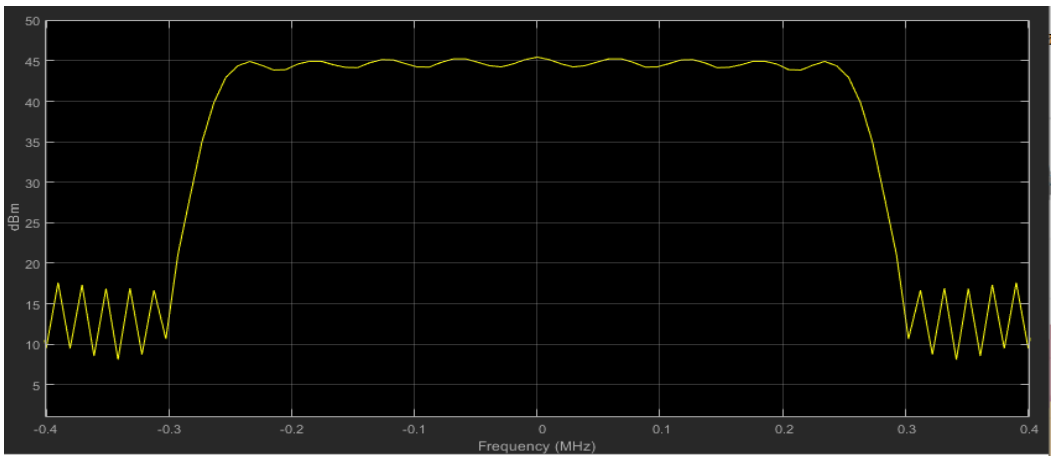

Figure 19. Spectrogram of the UFMC signal in the circuit for ten sub-ranges at T=5 with an elliptical filter.

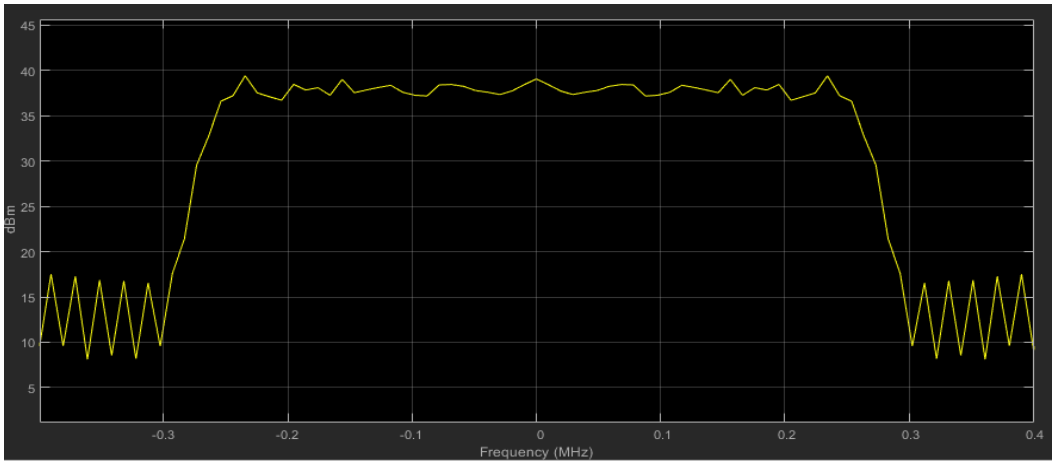

Figure 20. Spectrogram of the UFMC signal in the circuit for ten sub-ranges at T=10 with an elliptical filter

The simulation of the presented circuit using a 2nd order Chebyshev filter is shown in Figures 21-23.

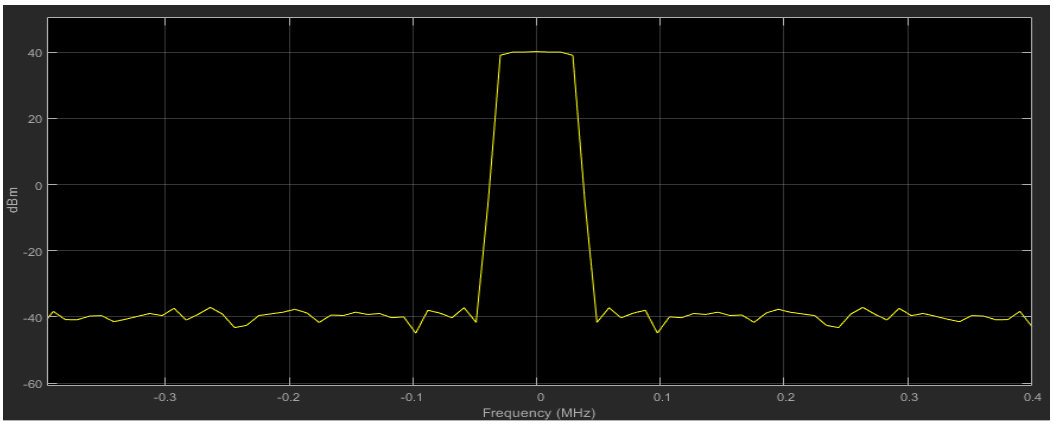

Figure 21. Spectrogram of the UFMC signal in the circuit for two sub-ranges at T=1 with a 2nd order Chebyshev filter

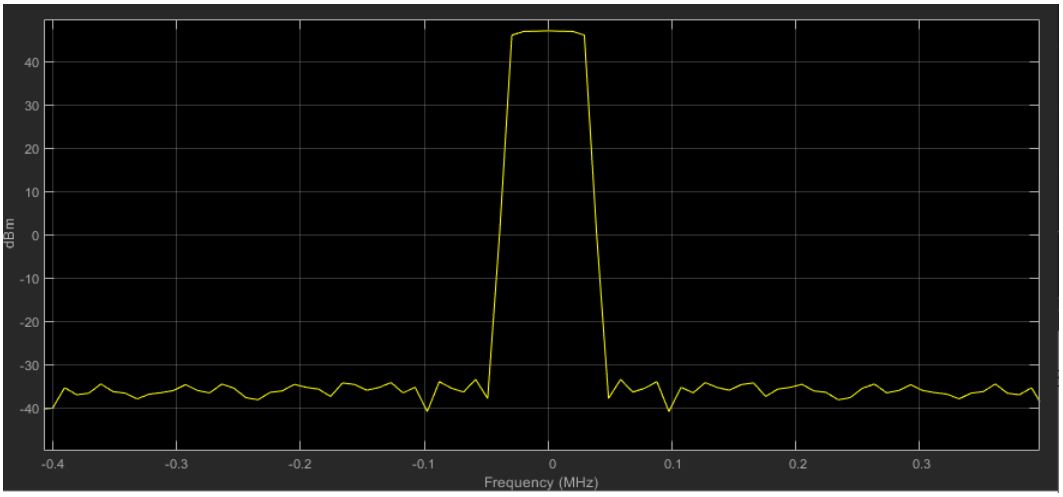

Figure 22. Spectrogram of the UFMC signal in the circuit for two sub-ranges at T=5 with a 2nd order Chebyshev filter

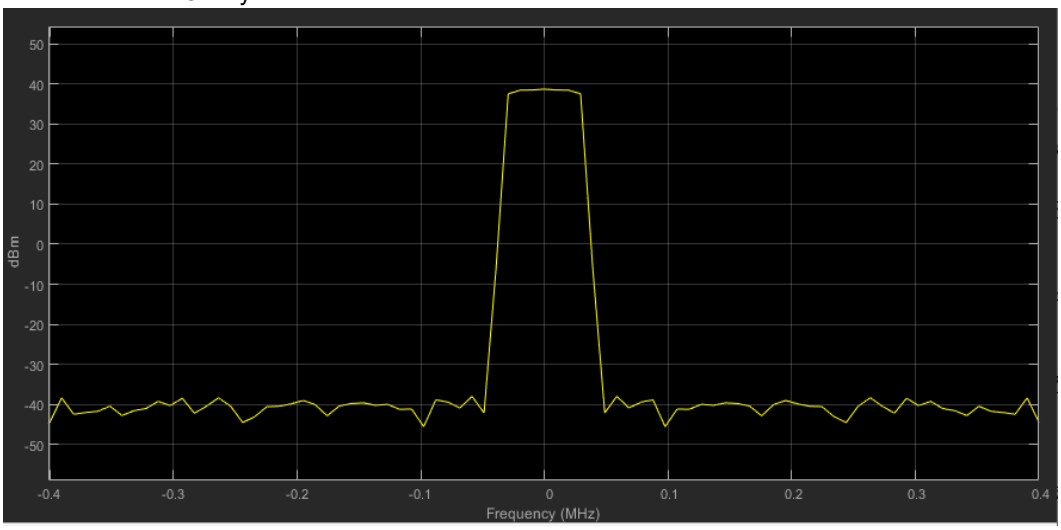

Figure 23. Spectrogram of the UFMC signal in the circuit for two sub-ranges at T=10 with a 2nd order Chebyshev filter

Figure 24 shows a comparison of the normalized spectral power density of OFDM signals and a UFMC signal. The number of sub-bands is 10, N = 512.

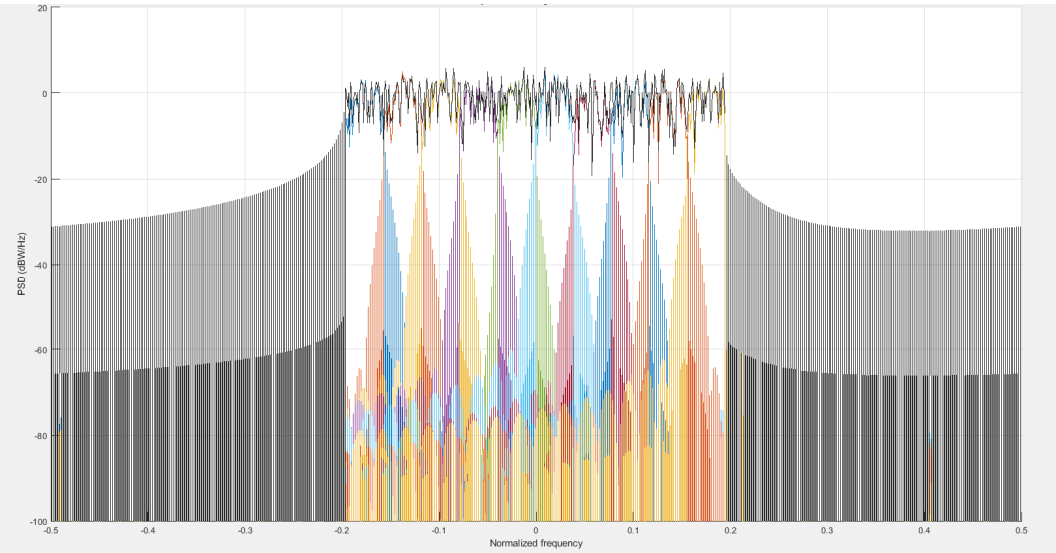

Figure 24. Comparison of the normalized spectral power density of OFDM signals and UFMC signals

The side lobe level drops by 35 dB.

Conclusion

In this article, the transmit section of a UFMC system with two and ten subbands was constructed using Simulink and Xilinx modules. This topic is currently the subject of intense research, as this technology may be used in next-generation 5G communications. An elliptic filter and a second-order Chebyshev filter were used for filtering. Spectrograms and oscillograms at various simulation times showed that the elliptic filter performed better than the second-order Chebyshev filter. When considering a system with ten subbands, spectrograms and oscillograms at various simulation times showed that the elliptic filter also performed better than the second-order Chebyshev filter. This system can be further implemented on FPGAs, first in cosimulation mode and then with direct FPGA programming. Therefore, the practical significance of using this UFMC system's transmitting component can be considered justified, as demonstrated by a MATLAB experiment.

In conclusion, among all alternative signal forms to OFDM, UFMC was considered the best choice for short-burst transmission and was successfully implemented in coordinated multipoint uplink communications. Generally, UFMC can be viewed as an intermediate method between OFDM and FBMC, combining the simplicity of OFDM with the noise immunity of FBMC. The filtering operation in UFMC is performed on a group of consecutive subcarriers with improved spectral localization, significantly reducing the filter length. Furthermore, quadrature amplitude modulation (QAM) at the transmitter and FFT-based processing at the receiver make UFMC compatible with multiple-input multiple-output (MIMO) methods in OFDM. In UFMC systems, the key challenge is the design of the filter for shaping the subbands. Unlike techniques such as coding and windowing in OFDM, waveform shaping filters with improved stopband attenuation can be used to improve sidelobe suppression between resource blocks and hence to minimize intersymbol interference.

REFERENCES

1. A. E. Mikenin, G. A. Prokurat, A. V. Ermakova, N. M. Buzueva and A. A. Sergeev, "Noise Modeling in Voltage-Controlled Oscillators in SPICE-Based Programs," *2025 Systems of Signal Synchronization, Generating and Processing in Telecommunications (SYNCHROINFO)*, Tyumen, Russian Federation, 2025, pp. 1-5, doi: 10.1109/SYNCHROINFO65403.2025.11079323.
2. A. V. Ermakova, S. F. Gorgadze, “Synchronization of multivalued linear recurrent sequences based on the generalized fast Fourier transform,” *Electrosvyaz*. 2025. No. 4, pp. 74-86. DOI 10.34832/ELSV.2025.66.4.009
3. I. V. Vorozhishchev, G. S. Bochechka, “Study of the stability of multi-frequency transmission technology with universal filtering UFMC to frequency shifts in the channel,” *T-Comm*. 2017. Vol. 11, No. 6, pp. 25-28.
4. G. Bochechka, V. Tikhvinskiy, I. Vorozhishchev et al., “Comparative analysis of UFMC technology in 5G networks,” *2017 International Siberian Conference on Control and Communications, SIBCON 2017* - Proceedings, Astana, 2017. P. 7998465. DOI 10.1109/SIBCON.2017.7998465.
5. S. D. Vu, A. V. Ermakova and S. F. Gorgadze, "Fast Spectral Transformations in the Truncated Walsh-Hadamard Basic and Synchronization of M-like Sequences," *2024 Systems of Signal Synchronization, Generating and Processing in Telecommunications (SYNCHROINFO)*, Vyborg, Russian Federation, 2024, pp. 1-6, doi: 10.1109/SYNCHROINFO61835.2024.10617540.
6. A. Jamoos, M. Hussein, “Estimation of UFMC Time-Varying Fading Channel Using Adaptive Filters,” *Proceedings - 2018 International Conference on Promising Electronic Technologies, ICPET 2018*, Deir El-Balah, Deir El-Balah, 2018, pp. 43-48. DOI 10.1109/ICPET.2018.00014.
7. S. F. Gorgadze, A. V. Ermakova, A. Yu. Kudryashova, “Group signals based on symmetric orthogonal matrices and processing of multipath signals,” *T-Comm*. 2025. Vol. 19, No. 10, pp. 21-34. DOI 10.36724/2072-8735-2025-19-10-21-34.
8. Sh. D. Wu, A. V. Ermakova, S. F. Gorgadze, “Fast spectral transforms in the truncated Walsh-Hadamard basis and synchronization of m-like sequences,” *Systems of synchronization, formation and processing of signals*. 2024. Vol. 15, No. 5, pp. 32-39.
9. S. D. Vu, A. V. Ermakova, S. F. Gorgadze, “Fast Spectral Transformations in the Truncated Walsh-Hadamard Basic and Synchronization of M-like Sequences,” *Systems of Signal Synchronization, Generating and Processing in Telecommunications*. 2024. Vol. 7, No. 1, pp. 623-628. DOI 10.1109/SYNCHROINFO61835.2024.10617540.
10. A. V. Ermakova and S. F. Gorgadze, "Method for Transforming Matrix Circulants of Multiposition Linear Recurrence Sequences into Matrices of Vilenkin-Crestenson Functions," *2025 Systems of Signals Generating and Processing in the Field of on Board Communications*, Moscow, Russian Federation, 2025, pp. 1-7, doi: 10.1109/IEEECONF64229.2025.10947700.
11. G. A. Hussain, L. Audah, “UFMC and f-OFDM: Contender waveforms of 5G wireless communication system,” *International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)* : 7th, Yogyakarta, 2020, pp. 74-77. DOI 10.11591/eecsi.v7.2040.
12. S. F. Gorgadze, Sh. D. Wu, A. V. Ermakova, “Synchronization of m-sequences based on the fast Hadamard transform,” *Radio Engineering and Electronics*. 2024. Vol. 69, No. 2, pp. 122-136. DOI 10.31857/S0033849424020031.
13. B. Singh, M. R. Tripathy, R. Asthana, “BER Reduction of UFMC for 1024-QAM,” *2021 IEEE International Conference on RFID Technology and Applications, RFID-TA 2021*, Virtual, Delhi, 2021, pp. 293-296. DOI 10.1109/RFID-TA53372.2021.9617386.
14. S. F. Gorgadze, A. V. Ermakova, A. Yu. Kudryashova, “Multiple access based on circular matrices of multiposition linear recurrent sequences,” *T-Comm*. 2025. Vol. 19, No. 3, pp. 37-53. DOI 10.36724/2072-8735-2025-19-3-37-53.
15. V. D. Chintala, A. Sundru, “A Joint Time-Domain Channel Estimation with Hybrid PAPR Reduction Scheme in UFMC Systems,” *Journal of Circuits, Systems, and Computers*. 2022. Vol. 31, No. 07. DOI 10.1142/s0218126622501298.
16. R. R, P. V, B. M A, “Analysis of Optimum Technique for PAPR Reduction in UFMC System,” *International Journal for Research in Applied Science and Engineering Technology*. 2023. Vol. 11, No. 8, pp. 632-638. DOI 10.22214/ijraset.2023.37683.
17. E. A. Tuli, R. Akter, Ja. M. Lee, D. S. Kim, “Whale optimization-based PTS scheme for PAPR reduction in UFMC systems,” *IET Communications*. 2024. Vol. 18, No. 2, pp. 187-195. DOI 10.1049/cmu2.12708.
18. S. F. Gorgadze, Sh. D. Wu, A. V. Ermakova, “Synchronization of Gold sequences based on fast transform in a truncated basis of Walsh-Hadamard functions,” *Radio Engineering and Electronics*. 2024. Vol. 69, No. 2, pp. 137-145. DOI 10.31857/S0033849424020045.
19. I. Khelouani, K. Zerhouni, F. Elbahhar et al., “UFMC Waveform and Multiple-Access Techniques for 5G RadCom,” *Electronics*. 2021. Vol. 10, No. 7. P. 849. DOI 10.3390/electronics10070849.
20. A. V. Ermakova, “Using non-orthogonal subcarriers based on m-sequence segments to generate group signals in mobile communication systems,” *DSPA: Application Issues of Digital Signal Processing*. 2024. Vol. 14, No. 3, pp. 23-29.