Low-Complexity VLSI Architecture for OTFS Transceiver Under Multipath Fading Channel

Ashish Ranjan Shadangi, *Graduate Student Member, IEEE*, Suvra Sekhar Das, *Member, IEEE*, and Indrajit Chakrabarti, *Member, IEEE*

***Abstract*— Orthogonal time frequency space (OTFS) modulation has established itself as a dependable protocol for high-speed vehicular communication. This pioneering technique operates within a novel 2-D delay-Doppler domain waveform. When compared with conventional modulation methods like orthogonal frequency-division multiplexing (OFDM), OTFS demonstrates superior performance enhancements in scenarios involving rapidly moving wireless channels. This article begins by initially unveiling the input–output association of the OTFS signal within the delay-time domain. A comprehensive comparison with the established OFDM waveform highlights the potential of OTFS for achieving a notably lower bit error rate (BER) under various conditions, which has been obtained by using the minimum mean square equalizer (MMSE) equalization technique. Finally, we have proposed a novel and low-complexity VLSI architecture for the OTFS transmitter and the receiver by using the lower–upper (LU) decomposition technique for the first time in the literature. We have compared the performance metrics of our proposed transmitter architecture with the existing work, where our design works 7.394% faster than others, utilizing 89.354% less in the number of lookup tables (LUTs) and 79.984% less in the number of flip-flops (FFs), which shows that our design is more optimized in latency and resource utilization. There is no architecture design of the OTFS receiver part in the existing literature to compare; we have shown the resource utilization of our proposed receiver architecture for the first time in the literature, followed by timing analysis and functionality testing of the proposed architecture.**

***Index Terms*— Bit error rate (BER), lower–upper (LU) decomposition, minimum mean square equalizer (MMSE), orthogonal frequency-division multiplexing (OFDM), orthogonal time frequency space (OTFS), VLSI.**

I. Introduction

THE requirement of high-quality service in fast-moving vehicular scenarios [1] including vehicle-to-vehicle (V2V) communications, unmanned aerial vehicle communications, and other fifth-generation (5G) applications is steadily rising day by day. Although orthogonal frequency-division multiplexing (OFDM) is a widely used transmission technology, it faces challenges in delivering reliable connections at speeds exceeding 300 km/h due to its susceptibility to intercarrier interference (ICI) arising from Doppler spread and phase noise. In contrast, orthogonal time frequency space (OTFS), as demonstrated in [2], has proven to be superior to OFDM in such a highly mobile environment, making it an essential component of 5G's operational scenarios.

To the best of authors' knowledge, OTFS modulation is a recently introduced 2-D modulation method that leverages the delay-Doppler domain for encoding information symbols, as detailed in [2]. It is noteworthy that OTFS incorporates both preprocessing and postprocessing steps into conventional multicarrier modulation schemes, resulting in enhanced bit error performance compared to traditional multicarrier techniques. Furthermore, channel variations have been observed to be more gradual in the delay-Doppler domain as compared to the time-varying multipath channel. This simplifies the equalizer design and allows for less frequent channel estimation in OTFS, consequently reducing the overhead associated with channel estimation in rapidly changing channels [3].

In the OTFS framework, data symbols are arranged in the delay-Doppler domain, as distributed in the time–frequency grid used in OFDM. Subsequently, a unitary transformation, known as the inverse symplectic finite Fourier transform (ISFFT), is applied to disperse the data across the time–frequency grid. Next, an OFDM modulation technique is employed [2]. For delay spread, a cyclic prefix is added in the case of OFDM modulation, which is referred to as CP-OTFS. On the other hand, when OTFS is combined with block OFDM and block CP, it is denoted as reduced CP-OTFS [4].

The OTFS signal, which exhibits time–frequency spread and is susceptible to intersymbol interference and ICI when transmitted through a linear time-varying channel, is subjected to advanced interference cancellation techniques. These receivers can be categorized into two types: 1) linear receivers [5] and 2) nonlinear receivers [6]. Nonlinear receivers offer lower error probabilities but come with higher computational complexity compared to their linear counterparts. Our focus is exclusively on linear receivers due to their practical feasibility.

In order to deal with high ICI due to Doppler spread and phase noise, 5G new radio (NR) has adopted a variant of a contemporary edition of OFDM, which is known as variable subcarrier bandwidth OFDM (VSB-OFDM) [7], [8].

Manuscript received 25 September 2023; revised 6 January 2024 and 5 March 2024; accepted 26 March 2024. Date of publication 22 April 2024; date of current version 28 June 2024. This work was supported in part by the Ministry of Electronics and Information Society (MeitY), Government of India, under the project "Next Generation Wireless Research and Standardization on 5G and Beyond." *(Corresponding author: Ashish Ranjan Shadangi.)*

Ashish Ranjan Shadangi and Suvra Sekhar Das are with the G. S. Sanyal School of Telecommunications (GSSST), IIT Kharagpur, Kharagpur 721302, India (e-mail: arshadangi@iitkgp.ac.in).

Indrajit Chakrabarti is with the Department of Electronics and Electrical Communication Engineering (E & ECE), IIT Kharagpur, Kharagpur 721302, India.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TVLSI.2024.3384114.

Digital Object Identifier 10.1109/TVLSI.2024.3384114

VSB-OFDM, which is characterized by different bandwidths for different subcarriers, serves to reduce ICI [9].

OTFS modulation was initially documented in [10], demonstrating its remarkable error performance, particularly in scenarios involving vehicle speeds as high as 500 km/h. Following this, numerous instances of work have emerged, each delving into various aspects of OTFS modulation [5], [11], [12], [13], [14], [15], [16]. Based on the multiplexing of MN information symbols across M delay bins and N Doppler bins, an analytical upper limit on the peak-to-average power ratio (PAPR) has been established. It is worth noting that this bound exhibits linear growth with the number of Doppler bins N and not with the number of delay bins M (or, equivalently, the number of subcarriers in the time–frequency domain). This contrasts with the conventional multicarrier waveform, where PAPR increases linearly with M. Consequently, OTFS systems with $N < M$ (typically the case) can potentially possess a lower PAPR compared to a multicarrier system with M subcarriers [17].

Very limited attention has been focused on the VLSI architecture of the OTFS transceiver on its hardware implementation in field-programmable gate array (FPGA), as observed in the current body of the literature. Dora et al. [18] have proposed their low-complexity OTFS transmitter architecture, where they estimated performance metrics of conventional architecture and their proposed optimized architecture implemented on 7vx485tffg1157 −1 FPGA device. However, we compared the performance metrics of our proposed transmitter architecture with [18] in the synthesis and implementation section. No architecture as of now is available in the existing literature on OTFS receivers. The primary obstacles within this field pertain to the substantial scale of matrix inversion required for computing the final mathematical equations in OTFS. Our proposed architecture has been developed to tackle these challenges. This work offers the advantage of efficiently utilizing FPGA board resources through the proposed architecture. This article introduces an architectural solution designed for an OTFS transceiver for the fading channel.

The novel features of this article are as follows.

1) In the transmitter module, the successful amalgamation of both the ISFFT and Heisenberg transform blocks has been achieved within a cohesive framework without compromising operational efficiency. This integration has facilitated the reduction of the complexity of two resource-intensive processes, namely, the M-point fast Fourier transform (FFT) and inverse FFT (IFFT) operations. As these operations exhibit self-complementarity, their combined effects result in mutual cancellation. The synergistic integration of ISFFT and Heisenberg transform has yielded a marked reduction in system complexity and resource utilization when compared to the approach outlined in [18].
2) In the receiver module, we focus on the implementation of the inversion of the matrix, which is involved in the linear inimum mean-square-error (LMMSE) equalizer because finding the matrix inversion using a particular FPGA device is a nontrivial and complex process. Our main objective is to investigate the hardware utilization of the implementation of the LMMSE receiver, which is not found in the literature to the best of authors' knowledge. Simultaneously, we have performed the functionality test and verified the timing analysis.
3) To validate the operational resilience of the proposed architecture and provide precise performance comparison results, measurement has been performed on both the transmitted and detected signals. The observed outcome reveals an exact correspondence between the transmitted and detected signals, highlighting the robust operational capabilities inherent in the proposed architecture.

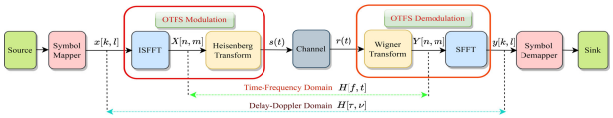


Fig. 1. OTFS modulation and demodulation schemes.

The structure of the remaining article is outlined as follows. In Section II, we delve into the models pertaining to the OTFS system. Section III elaborates on the matrix representation of the OTFS model. Section IV corresponds to the simulation results. Section V shows a low-complexity receiver. Section VI depicts the proposed architecture of the OTFS transmitter, and Section VII shows the proposed architecture of the OTFS receiver. Synthesis and implementation reports have been shown in Section VIII. Sections IX and X analyze timing and functionality tests, respectively. Finally, this article concludes in Section XI.

II. System Models

In the context being discussed, a single-input single-output (SISO) OTFS system is under consideration,[1] which involves the transmission and reception of uncoded quadrature amplitude modulation (QAM) symbols. This approach can be visualized as an augmentation of the conventional OFDM system through the incorporation of pre- and postprocessing modules. The modulation and demodulation procedures for OTFS are depicted in Fig. 1. Initially, at the transmitting end, the QAM symbols are organized within a 2-D matrix, with N columns in the Doppler domain and M rows in the delay domain. Subsequently, the signal undergoes a transformation into the time–frequency domain through the ISFFT. Following the Heisenberg transformation, the signal is converted back to the time domain and transmitted through the doubly-dispersive channel. At the receiving end, the received signal is subjected to processing using the Wigner transform and SFFT.

The time–frequency grid is discretized with resolutions of T and Δf, which define the units of time and subcarrier frequency spacing, respectively. This grid is composed of a collection of discrete time–frequency elements denoted as $\Phi = (kT, l\Delta f), k = 0, \ldots, N-1, l = 0, \ldots, M-1$. Here, T represents the duration of M QAM symbols and Δf represents

[1]Implementation of the multi-in multi-out (MIMO) realization of OTFS algorithm would be considerably involved, and thus, it is kept outside the purview of this work. We, therefore, intend to keep augmentation from SISO to MIMO implementation of OTFS as a future extension of our research.

the spacing between consecutive subcarriers. Subsequent to the ISFFT processing, the signal in the time–frequency domain can be represented as

$$X[n,m]=\frac{1}{\sqrt{NM}}\sum_{k=0}^{N-1}\sum_{l=0}^{M-1}x[k,l]e^{j2\pi\left(\frac{nk}{N}-\frac{ml}{M}\right)}. \tag{1}$$

In the given context, $x[k,l]$ represents the signal existing within the delay-Doppler domain with Doppler resolution as $f_r=(1/NT)$ and $d_r=(1/M\triangle f)$ as delay resolution. The framework of the delay-Doppler grid is defined as a specific collection characterized by a set of elements

$$\phi=\left(\frac{n}{NT},\ \frac{m}{M\triangle f}\right),\quad n=0,\ldots,N-1 \quad m=0,\ldots,M-1. \tag{2}$$

Following the reconfiguration of the matrix $X[n,m]$ into a sequence within the time domain, the Heisenberg transform (as detailed in [1] and [15]) is applied to generate the signal within the time domain:

$$s(t)=\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}X[n,\ m]g_{tx}(t-nT)e^{j2\pi m\triangle f(t-nT)} \tag{3}$$

where $g_{\mathrm{tx}}(t)$ represents the window function, and the signal $s(t)$ is propagated through the delay-Doppler channel. This propagation results in the received signal, which can be denoted as follows:

$$r(t)=\int\int h(\tau,\nu)s(t-\tau)e^{j2\pi\nu t}\mathrm{d}\tau\mathrm{d}\nu+w(t) \tag{4}$$

where $h(\tau,\nu)$ is the channel delay-Doppler response, $w(t)$ is the additive white Gaussian noise, and $h(\tau,\nu)$ is defined as

$$h(\tau,\ \upsilon)=\sum_{i=1}^{p}h_i\delta(\tau-\tau_i)\delta(\upsilon-\upsilon_i). \tag{5}$$

In this context, h_i, τ_i, and υ_i indicate the path gain, delay, and Doppler shift pertaining to the ith path, respectively.

Upon reception, the process involves applying a Wigner transform to initially convert the received signal from the time domain to the time–frequency domain, defined as follows:

$$Y[n,\ m]=\left[\int g_{rx}(t-\tau)r(t)e^{-j2\pi\nu(t-\tau)}dt\right]. \tag{6}$$

In this context, the parameters τ and ν are represented as $\tau=nT$ and $\nu=m\triangle f$, respectively, where $g_{\mathrm{rx}}(t)$ denotes the window function. Subsequently, the decoding of the signal is accomplished through the use of the SFFT method

$$y[k,\ l]=\frac{1}{\sqrt{NM}}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}Y[n,\ m]e^{-j2\pi\left(\frac{nk}{N}-\frac{ml}{M}\right)}. \tag{7}$$

Due to the influences of Doppler effects and channel delay spread, equalization is necessary to effectively retrieve data symbols from $y[k,l]$. We have utilized the minimum mean-squared equalization approach.

III. Matrix Representation

To facilitate the analysis of the OTFS system within fast-fading channels, we transform the given formulas into a framework involving vectors and matrices. We employ the Kronecker product symbol $\otimes$ and the function vec$(\cdot)$ to represent the Kronecker product and vectorization, respectively. Specifically, let $\mathbf{x}=\mathrm{vec}(\mathbf{D})$ symbolize the signal intended for transmission, where $\mathbf{D}\in\mathbb{C}^{M\times N}$ represents the 2-D data matrix. During the transmission process, the ISFFT operation can be expressed as follows:

$$\mathbf{X}=\mathbf{F}_M\mathbf{D}\mathbf{F}_N^H. \tag{8}$$

Here, $\mathbf{X}\in C^{M\times N}$ represents the signal intended for transmission in the time–frequency domain. The matrices $\mathbf{F}_M$ and $\mathbf{F}_N^H$ correspond to the FFT and IFFT matrices, respectively. Based on (3), the expression for the Heisenberg transform is derived as follows:

$$\mathbf{S}=\mathbf{G}_{tx}\mathbf{F}_M^H\mathbf{X}=\mathbf{G}_{tx}\mathbf{F}_M^H\left(\mathbf{F}_M\mathbf{D}\mathbf{F}_N^H\right)=\mathbf{G}_{tx}\mathbf{D}\mathbf{F}_N^H. \tag{9}$$

The matrix $\mathbf{S}$ is then converted into the time domain in vector form, as shown in the following equation, where $\mathbf{G}_{tx}$ is the identity matrix $(\mathbf{I}_M)$ of order M:

$$\mathbf{s}=\mathrm{vec}(\mathbf{S})=\left(\mathbf{F}_N^H\otimes\mathbf{G}_{tx}\right)\mathbf{x}. \tag{10}$$

After undergoing transmission through the delay-Doppler channel, the received signal can be formulated as follows:

$$\mathbf{r}=\mathbf{H}\mathbf{s}+\mathbf{w} \tag{11}$$

where $\mathbf{H}$ is defined as

$$\mathbf{H}=\sum_{p=1}^{P}h_p\mathbf{\Pi}^{l_p}\mathbf{\Delta}^{k_p} \tag{12}$$

where

$$\mathbf{\Pi}=\begin{bmatrix}0&\cdots&0&1\\1&\ddots&0&0\\\vdots&\ddots&\ddots&\vdots\\0&\cdots&1&0\end{bmatrix}_{MN\times MN} \tag{13}$$

$$\mathbf{\Delta}=\mathrm{diag}\left[z^0,z^1,\ldots,z^{MN-1}\right],\quad z=e^{j2\pi/MN}. \tag{14}$$

The received baseband signal subsequently passes through pulse shaping, Wigner transforms, and SFFT processing modules. By reorganizing the signal r into an $M\times N$ matrix denoted as $\mathbf{R}$, the received signal Y can be acquired as follows:

$$\mathbf{Y}=\mathbf{F}_M^H(\mathbf{F}_M\mathbf{G}_{rx}\mathbf{R})\mathbf{F}_N. \tag{15}$$

Finally, the signal is transformed back into a vector format as shown in the following equation:

$$\mathbf{y}=(\mathbf{F}_N\otimes\mathbf{G}_{rx})\mathbf{r}. \tag{16}$$

Here, $\mathbf{G}_{rx}$ represents the shaping pulse employed at the receiver. In the context of the OTFS system, we consider using standard rectangular pulse shaping. In this scenario, both the pulse shaping matrices $\mathbf{G}_{\mathrm{tx}}$ and $\mathbf{G}_{\mathrm{rx}}$ are equivalent to identity matrices. As a result, the relationship between the input and output of the OTFS signal can be articulated as follows:

$$\mathbf{y}=(\mathbf{F}_N\otimes\mathbf{I}_M)\mathbf{H}\left(\mathbf{F}_N^H\otimes\mathbf{I}_M\right)\mathbf{x}+(\mathbf{F}_N\otimes\mathbf{I}_M)\mathbf{w}. \tag{17}$$

TABLE I
SIMULATION PARAMETERS OF OTFS AND OFDM

Parameters for OTFS	Value
Carrier frequency (f_c)	6 GHz
Subcarrier Spacing($\Delta f = 1/T$)	30 KHz
Bandwidth (W)	10 MHz
No. of Subcarriers (M)	16
No. of OFDM Symbol (N)	16
Delay Resolution $d_r = 1/W$	100 ns
Doppler Resolution $f_r = 1/NT$	1875 Hz
Modulation Scheme	QPSK
Parameters for OFDM	**Value**
IFFT size	128
CP length	16
Modulation format	QPSK
No. of Subcarriers	16

Let $\sigma_{\mathbf{d}}{}^2$ be the transmission signal power, $\sigma_{\mathbf{n}}{}^2$ be the noise variance, and $\mathbf{H}_{\text{eff}}$ is given as

$$\mathbf{H}_{\text{eff}} = (\mathbf{F}_N \otimes \mathbf{I}_M)\mathbf{H}\big(\mathbf{F}_N^H \otimes \mathbf{I}_M\big). \tag{18}$$

The received signal undergoes equalization using conventional minimum mean square equalizer (MMSE). The receiver possesses accurate channel state information. The output of the MMSE equalizer, which considers the noise variance σ_n^2, can be represented as follows:

$$\hat{\mathbf{x}} = \mathbf{H}_{\text{eff}}^{\dagger}\left(\mathbf{H}_{\text{eff}}^{\dagger}\mathbf{H}_{\text{eff}} + \frac{\sigma_n^2}{\sigma_d^2}\mathbf{I}_{MN}\right)^{-1}\mathbf{y}. \tag{19}$$

When the signal $\mathbf{r}$ is subjected to processing through an LMMSE equalizer, then (19) can be simplified and written as follows [19]:

$$\hat{\mathbf{x}} = (\mathbf{HA})^{\dagger}\left[(\mathbf{HA})(\mathbf{HA})^{\dagger} + \frac{\sigma_n^2}{\sigma_d^2}\mathbf{I}\right]^{-1}\mathbf{r} \tag{20}$$

where $\mathbf{A}$ is equal to $(\mathbf{F}_N^H \otimes \mathbf{G}_{tx})$ and $\mathbf{F}_N^H$ is the $\mathbf{N}$-order normalized inverse discrete Fourier transform (IDFT). Thus, the above equation is simplified as

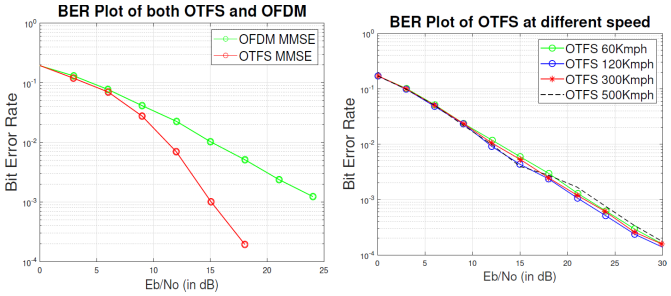
$$\hat{\mathbf{x}} = \mathbf{A}^{\dagger}\overbrace{\mathbf{H}^{\dagger}\left[\mathbf{H}\mathbf{H}^{\dagger} + \frac{\sigma_n^2}{\sigma_d^2}\mathbf{I}\right]^{-1}\mathbf{r}}^{\mathbf{H}_{\text{eq}}}. \tag{21}$$
$$\underbrace{\qquad\qquad\qquad\qquad}_{\mathbf{r}_{\text{ce}}=\mathbf{H}_{\text{eq}}\mathbf{r}}$$

Now, we can plot the bit error rate (BER) by comparing the original De-Do data ($\mathbf{x}$) and the equalized De-Do data ($\hat{\mathbf{x}}$).

IV. SIMULATION RESULTS

In this section, we conduct simulations for two modulation techniques: traditional OFDM and OTFS. Table I shows the details of the simulation parameters. We make the assumption that the receiver possesses accurate knowledge of the channel conditions; in other words, channel coding is not implemented.

Fig. 2(a) demonstrates that OTFS exhibits substantially improved BER performance compared to other modulation schemes, especially in fast-fading channels. It is notable that OTFS, specifically in LMMSE receiver, can effectively exploit diversity gain. For instance, when considering a BER of 10^{-3}, the OTFS-LMMSE receiver demonstrates an improvement of 8 dB in contrast to the OFDM-MMSE receiver. In light of these observations, we can deduce that OTFS surpasses OFDM in terms of performance.



(a) (b)

Fig. 2. (a) Performance comparison between OFDM and OTFS [20]. (b) BER plot of OTFS at different vehicle speeds [21].

Here, Fig. 2(b) depicts that OTFS works well for different high-speed scenarios for ideal channel estimation. Here, we can see that the BER plot has close proximity for almost all the speeds, which means that OTFS supports different orders of speeds. Compared to different speeds, we can see that the 120 km/h speed shows slightly better BER performance than all other speeds.

V. LOW-COMPLEXITY RECEIVER

Referring to (21), LMMSE equalization can be executed as a two-step process. In the initial stage, LMMSE channel equalization is performed to derive $\mathbf{r}_{\text{ce}} = \mathbf{H}_{\text{eq}}\mathbf{r}$, where $\mathbf{H}_{\text{eq}}$ represents the equalized channel matrix. The subsequent stage involves an OTFS-matched filter receiver, yielding the result $\hat{\mathbf{d}} = \mathbf{A}^{\dagger}\mathbf{r}_{\text{ce}}$, where $\mathbf{A}^{\dagger}$ signifies the conjugate transpose of matrix $\mathbf{A}$.

In the context of this study, it is demonstrated that the execution of $\hat{\mathbf{d}} = \mathbf{A}^{\dagger}\mathbf{r}_{\text{ce}}$ is straightforward and demands $(MN/2)\log_2(N)$ complex multiplications (CMs). However, the direct implementation of $\mathbf{r}_{\text{ce}} = \mathbf{H}_{\text{eq}}\mathbf{r}$ necessitates the inversion of the matrix $\mathbf{\Psi} = \mathbf{H}\mathbf{H}^{\dagger} + (\sigma_n^2/\sigma_d^2)\mathbf{I}$ and the multiplication of $\mathbf{H}^{\dagger}$, involving a computational complexity of $O(M^3N^3)$ CMs. To address this, Tiwari et al. [19] have reduced the computational complexity associated with $\mathbf{r}_{\text{ce}} = \mathbf{H}_{\text{eq}}\mathbf{r}$ by using lower–upper (LU) decomposition.

A. Structure of $\mathbf{\Psi} = [\mathbf{H}\mathbf{H}^{\dagger} + (\sigma_n^2/\sigma_d^2)\mathbf{I}]$

Using (19), $\mathbf{H}\mathbf{H}^{\dagger}$ can be expressed as

$$\mathbf{H}\mathbf{H}^{\dagger} = \sum_{p=1}^{P} h_p \mathbf{\Delta}^{k_p}\mathbf{\Pi}^{l_p}\sum_{s=1}^{P}\bar{h}_s\mathbf{\Delta}^{-k_s}\mathbf{\Pi}^{-l_s} \tag{22}$$

$$\mathbf{H}\mathbf{H}^{\dagger} = \sum_{\substack{p=1\\p=s}}^{P}|h_p|^2\mathbf{I} + \sum_{\substack{p=1\\p\neq s}}^{P}\sum_{s=1}^{P}h_p\bar{h}_s\mathbf{\Pi}^{l_p-l_s}\mathbf{\Delta}^{k_p-k_s}. \tag{23}$$

Using (12), $\mathbf{\Psi}$ becomes

$$\mathbf{\Psi} = \sum_{\substack{p=1\\p=s}}^{P}\left(|h_p|^2 + \frac{\sigma_n^2}{\sigma_d^2}\right)\mathbf{I} + \sum_{\substack{p=1\\p\neq s}}^{P}\sum_{s=1}^{P}h_p\bar{h}_s\mathbf{\Pi}^{l_p-l_s}\mathbf{\Delta}^{k_p-k_s}. \tag{24}$$

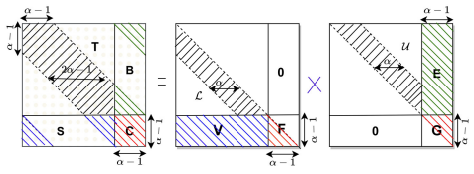


Fig. 3. Structure of $\mathbf{\Psi} = [\mathbf{H}\mathbf{H}^{\dagger} + (\sigma_n^2/\sigma_d^2)\mathbf{I}]$ matrix and its LU factorization.

Based on the aforementioned equations, it can be inferred that the utmost displacement of diagonal elements within matrix $\mathbf{\Delta}$ is approximately $\pm(\alpha - 1)$, where α is the channel delay length (where $\alpha = \lceil \tau_{\max} M \rceil$, $\tau_{\max}$ is the maximum delay spread). Furthermore, due to the cyclic pattern of this displacement, matrix $\mathbf{\Psi}$ demonstrates a quasi-banded structure, with a bandwidth of $(2\alpha - 1)$, as illustrated in Fig. 3. Given that, α is significantly smaller than MN, matrix $\mathbf{\Psi}$ also possesses sparsity characteristics in the context of a typical wireless channel. Considering the necessity of computing $\mathbf{\Psi}^{-1}$ for the practical realization of an LMMSE receiver, the authors put forth a method for achieving a computationally efficient LU decomposition of $\mathbf{\Psi}$.

B. Low-Complex LU Decomposition of $\mathbf{\Psi}$

In order to carry out the low-complexity LU decomposition of matrix Ψ, they suggest the following partitioning of Ψ, taking into account the values of $\theta = \alpha - 1$ and $\mathbf{Q} = \mathbf{MN} - \theta$:

$$\underbrace{\begin{bmatrix} \mathbf{T}_{Q\times Q} & \mathbf{B}_{Q\times\theta} \\ \mathbf{S}_{\theta\times Q} & \mathbf{C}_{\theta\times\theta} \end{bmatrix}}_{\mathbf{\Psi}} = \underbrace{\begin{bmatrix} \mathcal{L}_{Q\times Q} & \mathbf{0}_{Q\times\theta} \\ \mathbf{V}_{\theta\times Q} & \mathbf{F}_{\theta\times\theta} \end{bmatrix}}_{\mathbf{L}} \times \underbrace{\begin{bmatrix} \mathcal{U}_{Q\times Q} & \mathbf{E}_{Q\times\theta} \\ \mathbf{0}_{\theta\times Q} & \mathbf{G}_{\theta\times\theta} \end{bmatrix}}_{\mathbf{U}}. \quad (25)$$

Using the partition as shown in the above equation, the following equalities hold:

$$\mathbf{T} = \mathcal{L}\,\mathcal{U} \quad (26)$$

$$\mathbf{E} = \mathcal{L}^{-1}\,\mathbf{B} \quad (27)$$

$$\mathbf{V} = \mathbf{S}\,\mathcal{U}^{-1} \quad (28)$$

$$\mathbf{FG} = \mathbf{C} - \mathbf{VE}. \quad (29)$$

1) As the matrix T is a banded matrix, its LU decomposition can be calculated using the low-complexity algorithm detailed in [22].
2) The computation of $\mathcal{L}^{-1}\mathbf{B}$ can be achieved through a forward substitution algorithm tailored for lower triangular banded matrices, detailed in Algorithm 1, as stated in [19].
3) The calculation of (28) involves two sequential steps. Since $\mathcal{U}^{\dagger}$ forms a lower triangular banded matrix, the initial step entails the computation of $\mathbf{V}^{\dagger} = (\mathcal{U}^{\dagger})^{-1}\mathbf{S}^{\dagger}$ using Algorithm 1 as depicted in [19]. Subsequently, $\mathbf{V}$ can be readily obtained by computing the conjugate transpose of $\mathbf{V}^{\dagger}$.
4) A direct computation of (29) necessitates $O(\theta^2 MN)$ calculations. Given the lower triangular nature of matrix $\mathbf{F}$ and the upper triangular nature of matrix $\mathbf{G}$, both $\mathbf{F}$ and $\mathbf{G}$ can be computed through the $\mathbf{LU}$ decomposition of (29). The pivotal Gaussian elimination algorithm [23] can be applied to attain the $\mathbf{LU}$ decomposition of (31) without introducing significant complexity overhead. Notably, the diagonal elements of matrix $\mathcal{L}$ and $\mathbf{F}$ are all set to unity, implying that the diagonal elements of matrix $\mathbf{L}$ are also unity.

Algorithm 1 Computation of $\mathbf{Y} = \mathbf{\Gamma}^{-1}\mathbf{X}$

Input: a lower triangular matrix $\mathbf{\Gamma}_{Q\times Q}$ and $\mathbf{X}_{Q\times\theta}$
Output: $\mathbf{Y}_{Q\times\theta} = \mathbf{\Gamma}_{Q\times Q}^{-1}\mathbf{X}_{Q\times\theta}$

```
for s ← 1 to θ do
    Y(1, s) = X(1,s) / Γ(1,1)
    for k ← 2 to θ do
        Y(k, s) = (1/Γ(k,k)) (X(k, s) − Σ_{i=1}^{k−1} Γ(k, k − i) Y(k − i, s))
    end
    for k = θ + 1 : Q do
        Y(k, s) = (1/Γ(k,k)) (X(k, s) − Σ_{i=1}^{P−1} Γ(k, k − i) Y(k − i, s))
    end
end
```

Algorithm 2 Computation of $\mathbf{r}^{(1)} = \mathbf{L}^{-1}\mathbf{r}$

Input: a quasi banded lower triangular matrix $\mathbf{L}_{MN\times MN}$ and $\mathbf{r}_{MN\times 1}$
Output: $\mathbf{r}^{(1)}_{MN\times 1} = \mathbf{L}^{-1}_{MN\times MN}\mathbf{r}_{MN\times 1}$

```
r^(1)(1, 1) = r(1, 1)
for k ← 2 to α − 1 do
    r^(1)(k) = r(k) − Σ_{i=1}^{k−1} L(k, k − i) r^(1)(k − i, 1)
end
for k ← α to Q do
    r^(1)(k) = r(k) − Σ_{i=1}^{α−1} L(k, k − i) r^(1)(k − i, 1)
end
for k ← Q + 1 to MN do
    r^(1)(k) = r(k) − Σ_{i=1}^{k−1} L(k, k − i) r^(1)(k − i, 1)
end
```

C. Computation of $\boldsymbol{r}$

$$\mathbf{r}_{\text{ce}} = \mathbf{H}^{\dagger}\overbrace{\mathbf{U}^{-1}\underbrace{\mathbf{L}^{-1}\mathbf{r}}_{\mathbf{r}^{(1)}}}^{\mathbf{r}^{(2)}}. \quad (30)$$

Since matrix $\mathbf{L}$ is a quasi-banded lower triangular matrix, the computation of $\mathbf{r}^{(1)} = \mathbf{L}^{-1}\mathbf{r}$ can be efficiently achieved using a forward substitution approach, as elaborated in Algorithm 2, as shown in [19]. Subsequently, the calculation of $\mathbf{r}^{(2)} = \mathbf{U}^{-1}\mathbf{r}^{(1)}$ can be carried out employing Algorithm 3 following [19]:

$$\mathbf{r}_{\text{ce}} = \sum_{p=1}^{P} \bar{h}_p \mathbf{\Delta}^{-k_p} \underbrace{\mathbf{\Pi}^{-l_p}\mathbf{r}^{(2)}}_{\text{circular shift}}. \quad (31)$$

For the computation of $\mathbf{r}_{\text{ce}}$, the vector $\mathbf{r}^{(2)}$ is initially subjected to a circular shift by a delay of $-l_p$ and subsequently multiplied by $\bar{h}_s \text{diag}(\Delta^{-k_p})$ using elementwise multiplication for each path p. The resultant vectors from this process are eventually summed to yield $\mathbf{r}_{\text{ce}}$.

Algorithm 3 Computation of $\mathbf{r}^{(2)} = \mathbf{U}^{-1}\mathbf{r}^{(1)}$

Input: a quasi banded upper triangular matrix $\mathbf{U}_{MN\times MN}$ and $\mathbf{r}^{(1)}_{MN\times 1}$

Output: $\mathbf{r}^{(2)}_{MN\times 1} = \mathbf{U}^{-1}_{MN\times MN}\mathbf{r}^{(1)}_{MN\times 1}$

$\mathbf{r}^{(2)}(MN-1) = \frac{\mathbf{r}^{(1)}(MN-1)}{\mathbf{U}(MN-1,MN-1)}$

for $k \leftarrow MN-1$ **to** $MN-2(\alpha-1)$ **do**

  $\mathbf{r}^{(2)}(k,1) = \frac{1}{\mathbf{U}(k,k)}(\mathbf{r}^{(1)}(k) - \sum_{i=1}^{MN-k}\mathbf{U}(k,k+i)\mathbf{r}^{(2)}(k+i,1))$

end

for $k = MN-(2*(\alpha-1))-1$ **to** 1 **do**

  $\mathbf{r}^{(2)}(k,1) = \frac{1}{\mathbf{U}(k,k)}(\mathbf{r}^{(1)}(k,1) - \sum_{i=1}^{\alpha}\mathbf{U}(k,k+i)\mathbf{r}^{(2)}(k+i,1) - \sum_{r=MN-(\alpha-2)}^{MN}\mathbf{U}(k,r)\mathbf{r}^{(2)}(r,1))$

end

Instead of directly calculating $\hat{\mathbf{x}}$ as $\mathbf{A}^{\dagger}\mathbf{r}_{\text{ce}}$, a preliminary step involves reshaping $\mathbf{r}_{\text{ce}}$ into a matrix $\mathbf{R}$ of size $M \times N$, following which we perform the detection of the symbol as shown in the following equation:

$$\hat{\mathbf{x}} = \text{vec}\left\{\mathbf{R}\mathbf{W}_N^{\dagger}\right\}. \tag{32}$$

This operation can be implemented by using M number of N-point FFT operations.

Considering low-complexity LMMSE equalization technique, it takes $((\alpha(MN-2\alpha+2)^2)/2) + (((\alpha^2-\alpha)(3MN-5\alpha+4))/6) + 2(\theta^3+\theta^2(1-P)+PQ\theta + ((\alpha(\alpha-1)(4\alpha+1))/6) + Q\theta + (\alpha-2) + (\alpha+1)(2MN-2\alpha+1) + 1 + (\alpha+1)(6MN-6\alpha+5) + MNP(\beta+1) + (MN/2)\log_2 N$ CMs. On the other hand, direct implementation would involve $(MN/2)\log_2 N + (8/6)(MN)^3 + 2(MN)^2$ computations.

Considering the number of subcarriers M, which may be $2, 4, 8, \ldots,$ $N = 4$, and $\alpha = 3$, we compared the computational complexity of CMs performed with and without LU decomposition (direct), as shown in Fig. 4.

After verifying the OTFS transmitter steps and analyzing the complexity of the OTFS receiver, one can now design the VLSI architecture of the OTFS transceiver for a multipath fading channel pertaining to the SISO case. Here, we assume that the channel characteristics are known. In other words, one performs an ideal channel estimation at the receiver.

VI. Architecture Design for OTFS Transmitter

Fig. 5 shows the architectural design of the OTFS transmitter design based on the block diagram of the OTFS modulation and demodulation schemes, which is portrayed in Fig. 1. This architecture is designed by following the equation shown in (8)–(10).

The novelty of our proposed transmitter architecture is that we have clubbed the ISFFT and Heisenberg transform to get a simplified equation ($\mathbf{s} = \text{vec}(\mathbf{S}) = (\mathbf{F}_N^H \otimes \mathbf{G}_{tx})\mathbf{x}$), without affecting the performance.

As mentioned in (8), the ISFFT operation can be expressed as $\mathbf{X} = \mathbf{F}_M\mathbf{D}\mathbf{F}_N^H$. From (3), the expression for the Heisenberg transform is derived like, $\mathbf{S} = \mathbf{G}_{tx}\mathbf{F}_M^H\mathbf{X} = \mathbf{G}_{tx}\mathbf{F}_M^H(\mathbf{F}_M\mathbf{D}\mathbf{F}_N^H) = \mathbf{G}_{tx}\mathbf{D}\mathbf{F}_N^H$. The matrix $\mathbf{S}$ is then converted into the time domain in vector form like $\mathbf{s} = \text{vec}(\mathbf{S}) = (\mathbf{F}_N^H \otimes \mathbf{G}_{tx})\mathbf{x}$.

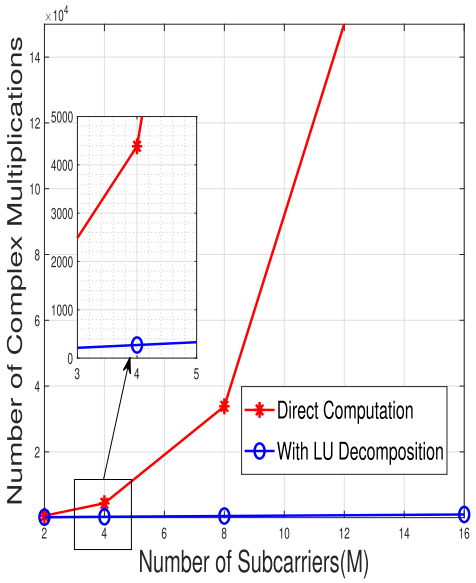


Fig. 4. Computation complexity comparison of the direct and LU decomposition.

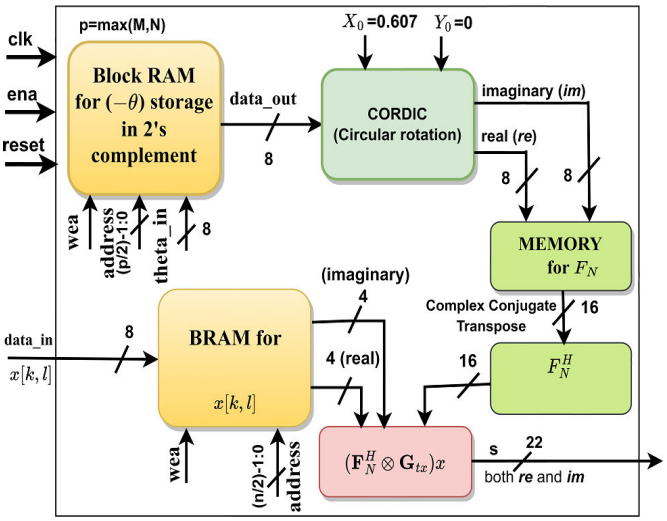


Fig. 5. Proposed OTFS transmitter architecture.

Hence, by clubbing both ISFFT and the Heisenberg transform, the two separate blocks combine as a simplified one, due to which the architecture of the transmitter becomes very simple and less complex. This integration has facilitated the reduction of the complexity of two resource-intensive processes, namely, the M-point FFT and IFFT operations. This indicates the novelty of the proposed OTFS transmitter. Based on this, the proposed architecture has been built.

Here, for the coding part, we have employed the HDL coder (Verilog code, Vivado 2019.2 version toolbox).

The detailed proposed architecture is explained as follows.

A. Original Signal Generation

The main building block for the transmitter is the constellation mapper to produce the transmitted signal; for this, we can use any digital modulation scheme like BPSK, QPSK, or any M-Ary modulation scheme. We considered the normalized

QPSK modulation scheme for our design, where each symbol consists of two bits. Once the QPSK-modulated signal is generated, it is stored in the block random access memory (BRAM), as shown in Fig. 5. The main inputs for our architecture are the clock signal of period 10 ns, enable, and reset. These inputs are common to all the blocks shown in Fig. 5. The clock time period is fixed based on timing analysis, where 10 ns meets all the timing constraints of our proposed architecture. As we know, BRAM stores the symbols serially in vector form; we stored the delay-Doppler symbols of size $M \times N$ in vector form. Notably, we have chosen $M = N = 4$ in this specific context. The symbols of the first row are stored at the addresses $0, 1, \ldots, 3$. Similarly, the symbols of the second are stored at the addresses from $4, 5, \ldots, 7$ and so on with the write enable pin (wea), as shown in Fig. 5.

B. Generation of $\mathbf{F}_N^H$

Our main focus is to design FFT matrix ($\mathbf{F}_N$), which later gives the $\mathbf{F}_N^H$, by conjugate transposing to $\mathbf{F}_N$. For finding $\mathbf{F}_N$, we have gone for our own customized coordinate rotation digital computer (CORDIC) architecture. To reduce resource utilization and power consumption to a considerable extent, we may use the custom CORDIC architecture instead of the CORDIC IP core of Vivado. We have considered the values of parameters like M, N, MN, and the number of pipeline stages for the CORDIC program. Here, we need to compute the values of $\cos(\theta)$ and $\sin(\theta)$ by using area- and speed-efficient CORDIC architecture. There are three types of CORDIC operations, which are known as CIRCULAR, HYPERBOLIC, and LINEAR. Again in each and every operation, there are two modes of operation, namely, rotation and vectoring modes, including circular rotation, circular vectoring, hyperbolic rotation, hyperbolic vectoring, linear rotation, and linear vectoring. By using a specific mode of operation, one can perform a particular operation. Here, we have considered the circular rotation for finding the trigonometric function such as $\sin(\theta)$ and $\cos(\theta)$. Here, we have considered 16 stages of pipelined CORDIC operation.

The CORDIC algorithms for this mode are derived from the general rotation transform

$$X_N = X_0 \cos\theta - Y_0 \sin\theta \tag{33}$$

$$Y_N = X_0 \sin\theta + Y_0 \cos\theta \tag{34}$$

$$X_N = \cos\theta(X_0 - Y_0 \tan\theta) \tag{35}$$

$$Y_N = \cos\theta(X_0 \tan\theta + Y_0) \tag{36}$$

$$X_{i+1} = X_i - \delta_i Y_i 2^{-i} \tag{37}$$

$$Y_{i+1} = \delta_i X_i 2^{-i} + Y_i \tag{38}$$

$$\epsilon_{i+1} = \epsilon_i - \delta_i \alpha_i \tag{39}$$

$$\delta_i = \text{Sign}\epsilon_i \tag{40}$$

where X_0 and Y_0 are the initial values, and X_N and Y_N are the final values. Here, $X_0 = \prod_{i=0}^{N-1} \cos\alpha_i = 0.607$ and $Y_0 = 0$. ϵ_0 is the user-given angle. Here, $\epsilon_0 = \theta$. α_i is the predefined angle and $\alpha_i = \tan^{-1} 2^{-i}$. This acts as a trivial rotator of vectors, resulting in occupying less area. $\delta_\mathbf{i}$ gives the sign of $\epsilon_\mathbf{i}$. Within the CORDIC methodology, the process of performing a plane rotation by an angle α_i involves breaking down the desired angle into multiple elementary angles. Subsequently, rotations are executed for each of these elementary angles, resulting in the overall rotation

$$\epsilon_0 = \sum_{i=0}^{N-1} \delta_i \alpha_i. \tag{41}$$

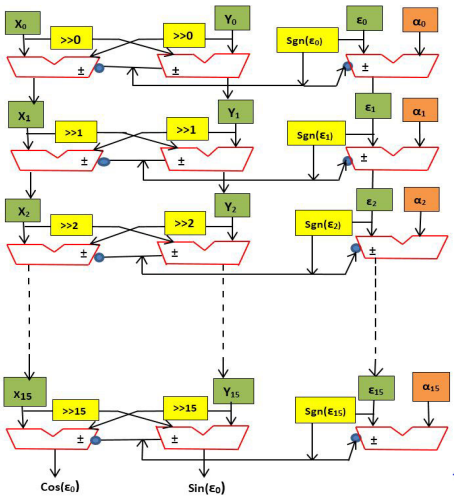


Fig. 6. Architecture of pipelined CORDIC.

In the context of Fig. 6, the variable ϵ_0 corresponds to θ. At the final stage of the process, the quantities $\cos(\epsilon_0)$ and $\sin(\epsilon_0)$ correspond to $\cos(\theta)$ and $\sin(\theta)$, respectively. A set of θ values are stored in the BRAM to initiate the procedure. These theta values are 8 bits in width, with a specific resolution pattern. The allocation includes 1 bit for the sign, the following bit for the integer part, and the remaining 6 bits for the fractional portion. Sequentially, these theta values are transmitted to the CORDIC stages, aligning with each positive edge of the clock cycle within the code. In our custom CORDIC, we are using the wired right shifter technique in place of the division of 2's power number in each stage of the custom CORDIC to reduce resource utilization and power consumption. Due to the utilization of a 16-stage pipelined CORDIC module, the initial output from the CORDIC block becomes available after 16 clock cycles. The outcome is saved as F_N, while the 2's complement of $\sin(\theta)$ is stored as F_N^H. Both $\cos\theta$ and $\sin\theta$ are calculated with 8 bits for each value; thus collectively, the width size for each component of F_N and F_N^H is 16 bits. Once these values are entirely accessible, subsequent calculations are executed.

C. Transmitted Signal Generation

Once the original signal and $\mathbf{F}_N^H$ are generated, we can go for final transmitted signal ($\mathbf{s} = \text{vec}(\mathbf{S}) = (\mathbf{F}_N^H \otimes \mathbf{G}_{\text{tx}})\mathbf{x}$) generation. $\mathbf{G}_{\text{tx}}$ is readily recognized as the $M \times M$ identity matrix. Here, $\otimes$ represents the Kronecker product, where the first Nth elements of $\mathbf{F}_N^H$ are multiplied with the M-set of element of $\mathbf{G}_{\text{tx}}$, N times till end of $\mathbf{G}_{\text{tx}}$. Similarly, the second set of Nth elements of $\mathbf{F}_N^H$ is multiplied with the M-set of element of $\mathbf{G}_{\text{tx}}$, N times till end of $\mathbf{G}_{\text{tx}}$, and this process continues till

all elements of $\mathbf{F}_N^H$ are covered. By this process, we will have $MN \times MN$ matrices, which are stored in some intermediate memory in the vector. Once the Kronecker product is done, we go for the final product with the original signal (x). Here, we are taking the first MN set of elements from the intermediate memory, do elementwise multiplication with "x," and finally add all to get a single time-domain symbol. Similarly, take the second MN set of elements from the intermediate memory, do elementwise multiplication with "x," and finally add all to get a single time-domain symbol. By this process, we will have MN number of time-domain symbols to transmit through the transmitting antenna.

After completing the abovementioned process of the OTFS transmitter, one can now take up the design of the architecture of the OTFS receiver.

VII. Architecture Design for OTFS Receiver

Prior to the design, we have to calculate the values of the parameters l_p, k_p, and h_p.

A. Calculation of l_p, k_p, and h_p

As we have considered the extended vehicular A (EVA) model from ETSI TS 136 104 V14.3.0 (2017-04), page number 186. To assess the reduction in complexity facilitated by the proposed receiver, we analyze the OTFS system using the following parameters: $\Delta f = 30$ kHz, $f_c = 6$ GHz, and a vehicular speed of 500 km/h. The investigation includes vehicular channel models specified by 3GPP [24], denoted as EVA with parameters $P = 9$ and $\tau_{\max} = 2.51$ μs.

l_p Calculation:

$l_p = \text{round}(\tau/d_r) = \text{round}(\tau/(1/M\Delta f)) = \text{round}(\tau * M\Delta f))$

h_p Calculation:

If, $\text{pow}_{\text{liner}} = 10^{(\text{pow}_{\text{dB}}/10)}$ W,

then, $h_p = (\text{randn}(\text{length}(\tau), 1) + 1i * \text{randn}(\text{length}(\tau), 1)).$ $* \text{sqrt}(\text{pow}_{\text{liner}}/2)$

k_p Calculation:

Let $f_d = (v_{\max} * f_c/c)*\cos(\theta_p)$. Here, θ_p is uniformly distributed over $[-\pi\ \pi]$, $f_c = 6$ GHz, $c =$ light speed in m/s, and $v_{\max} =$ Maximum vehicle speed. Then,

$$k_p = \text{round}(f_d/f_r) = \text{round}(f_d * (N/\Delta f)).$$

If we consider the transmitter and receiver in the static scenario, then $v_{\max} = 0$ and $f_d = 0$; hence, k_p is made of all zeros.

The above calculations are generalized one. Thus, one can calculate the necessary values of l_p, k_p, and h_p based on the value of M, N, Δf, and $v_{\max}$.

B. OTFS Receiver Architecture

Fig. 7 shows the proposed receiver architecture. We have followed the equations given in (20)–(32) to derive this receiver architecture. As we have already assumed complete knowledge of the channel characteristics, we can calculate the value of h_p, k_p, and l_p at first by considering the EVA model from ETSI TS 136 104 V14.3.0 (2017-04), page number 186.

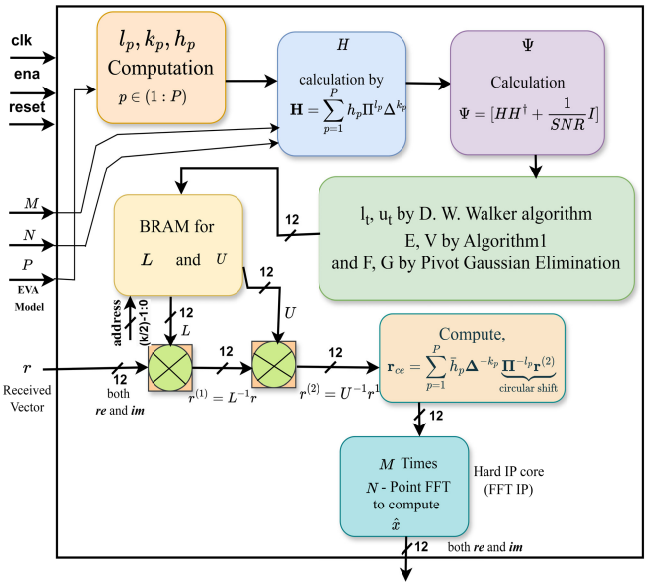


Fig. 7. Proposed OTFS receiver architecture.

Subsequently, we can find out the Ψ according to the equation mentioned in (25). The primary inputs for our architectural framework include the clock signal with a fixed period of 20 ns, alongside signals for enable, reset, M, N, P, and the received signal vector r. These input parameters are universally applicable across all blocks delineated in Fig. 7. The prescribed clock time period of 20 ns has been determined through meticulous timing analysis, ensuring its compliance with all specified timing constraints inherent to our proposed architecture.

1) Formation of H Matrix: Upon computation of h_p, k_p, and l_p derived based on the parameter P, the subsequent step involves the computation of the matrix H. The formulation of the H matrix is contingent upon the summation of the product of the complex attenuation (h_p) of P paths, the circulant delay matrix (Π), and the diagonal Doppler matrix (Δ). This operation, the creation of H, requires a total of $P + 2$ clock cycles. The additional two clock cycles are essential for retrieving the h_p, k_p, and l_p values from memory. This fetching or memory access operation occurs concurrently. Through this process, the channel matrix (H) is prepared with dimensions of $MN \times MN$. Each element within the H matrix is represented by 24 bits, where the initial 12 bits denote the real part, and the subsequent 12 bits denote the imaginary part of the complex value.

2) Formation of Ψ Matrix: The construction of the Ψ matrix is contingent upon the precomputed H matrix and the signal-to-noise ratio (SNR) value. Leveraging the available H matrix, the last 12 bits of each number undergo a 2's complement transformation. The initial MN set of numbers from H is selected, and these values are stored at the $MN(i)$th position within the $\mathbf{H}^{\dagger}$ memory, where "i" ranges from 0 to $MN - 1$. Subsequently, the second MN set of numbers from H is chosen and stored at the $MN(i + 1)$th position in $\mathbf{H}^{\dagger}$ memory, with "i" iterating from 0 to $MN - 1$, and this process repeats until all elements of H are accounted for.

To generate the complete Ψ matrix, the reciprocal of the SNR value is added to the first element of $\mathbf{HH}^{\dagger}$ for the initial MN element set, the second element of the second MN element set in $\mathbf{HH}^{\dagger}$ memory, and so forth.

Finally, we can do the LU decomposition of Ψ as mentioned in Section V-B and store the values of L and U in the BRAM by using the address of the required depth. All the values stored in the BRAM are of width 24 bits, of which 12 bits are for the real part and 12 bits for the imaginary part. All the stored numbers are in fixed point notation with 12 bits (MSB 1 bit for the sign, 4 bits for the integer part, and 7 bits for the fractional part).

At this juncture, our preparations include the matrices L and U. It is imperative to verify the nonsingularity of both L and U since we are incorporating their inverses, denoted as L^{-1} and U^{-1}, in the computations for $r^{(1)}$ and $r^{(2)}$, respectively. Given that $(1/\text{SNR})$ is a positive number, and Ψ is a positive definite matrix, the invertibility of Ψ is assured. Drawing from this, we affirm the nonsingularity of both L and U. In each clock cycle, we are retrieving one value of $r^{(1)}$ and performing the operation $r^{(1)} = L^{-1}r$ as per [19, Algorithm 2]. Then, in a similar manner, we take the values of L from BRAM and perform the operation $r^{(2)} = U^{-1}r^{1}$ following [19, Algorithm 3]. The multipliers shown in the architecture are power- and area-efficient approximate multipliers following [25].

Once the value of $r^{(2)}$ becomes available at the output of the second multiplier, we found out the r_{ce} as given in (31). Here, we have used the low-latency memory-based hard core IP for FFT [26] calculation (M numbers of N-point FFT) to find out $\hat{x}$. However, one can use parallel-processing memory-based FFT processors for high-throughput applications [27]. Next, we go for synthesis and implementation of the proposed architecture and present the utilization and timing analysis reports, and finally, we perform the functionality testing of the proposed receiver architecture.

VIII. Synthesis and Implementation

Once the Verilog code for the OTFS transmitter and the receiver has been formulated within a fading channel scenario for the SISO configuration, the next step is the synthesis and implementation using the Vivado tool. The target platform for this endeavor is the ZC706 BASE Board, which features the XC7Z045-2FFG900 device from the Zynq-7000 family. Specifically, we utilized the ffg900 package and set the speed grade to -2 as per the ZC706 evaluation kit specifications. The synthesis and implementation phases were executed successfully. For comprehensive insights, Tables II–IV provide an in-depth breakdown of the Synthesis and Implementation report of the transmitter and receiver, respectively.

We have implemented both algorithms in Verilog and showed the implemented results/resource utilization report and the flowchart of the implementation process of the proposed architectures.

In accordance with the introduction, our synthesis report is juxtaposed with the findings presented in [18]. The referenced work delineates an architecture exclusively designed for the transmitter component, specifically targeting a delay-Doppler grid with dimensions of 8×8. This study employs the 7vx485tffg1157-1 FPGA device to elucidate resource utilization and power consumption aspects.

TABLE II

Synthesized and Implementation Report of Transmitter, Considering (XC7Z045-2FFG900) FPGA Device

Setup and Hold	Value
Worst Negative Slack (WSN)	+1.264 ns
Total Negative Slack (TNS)	0.00 ns
Total Number of Endpoints	200
Utilization	**Value**
Slice LUTs	46/218600 (0.021%)
FF	80/437200 (0.018%)
Block RAM Tile	1/545 (0.183%)
IO	2/362 (0.552%)
On-Chip Power	**Value**
Dynamic Power	0.624Watt
Static Power	0.204Watt

TABLE III

Resource Utilization Comparison Table of Transmitter, Considering (7vx485tffg1157-1) FPGA Device

Parameters	The architecture shown in [18]	Our proposed architecture	% of improvemeent
Frequency (MHz)	100	100	-
Time Period (ns)	10	10	-
Latency (ms)	1.42	1.315	7.394
Max. frequency (MHz)	139.64	146.38	4.604
Min. time period (ns)	7.16	6.8315	4.587
Throughput (Tbps)	196.67	210.978	6.781
LUTs	75,026	7987	89.354
FFs	26,924	5389	79.984
BRAM	-	1	-
Power (Watt)	1.273	0.958	24.744

To facilitate a comprehensive comparison, our assessment encompasses a delay-Doppler grid of dimensions 8×8, mirroring the configuration utilized in [18]. The evaluation is conducted on the 7vx485tffg1157-1 FPGA device, aligning with the hardware context of the referenced work. The results, presented in Table III, distinctly illustrate the percentage improvement realized by our proposed architecture in terms of resource utilization and power consumption relative to the design outlined in [18].

The percentage improvement mentioned in Table III is achieved due to the successful integration of both the ISFFT and Heisenberg transform blocks into a unified structure without compromising performance. This integration has led to a significant reduction in complexity and resource utilization, resulting in the total power consumption of the proposed transmitter architecture being reduced by around 25%.

The floor plan of the implemented design is shown in Fig. 8. This is generated after the postimplementation of our design.

IX. Timing Analysis

After successfully synthesizing the proposed architecture, we went for the implementation, where we took a clock period of 20 ns to implement the designed HDL code with an input and output delay of 2 ns. The timing constraints are perfectly met, as shown in Fig. 9. The detailed timing summary report is shown in Table V.

Here, we checked the timing for the max delay paths and the min delay paths of the design. Max delay paths represent

TABLE IV
SYNTHESIZED AND IMPLEMENTATION REPORT OF RECEIVER, CONSIDERING (XC7Z045-2FFG900) FPGA DEVICE

Setup and Hold	Value
Worst Negative Slack (WSN)	+2.364 ns
Total Negative Slack (TNS)	0.00 ns
Total Number of Endpoints	10097
Utilization	Value
Slice LUTs	7195/218600 (3.29%)
Slice Registers	1656/437200 (0.37%)
Block RAM Tile	2/545 (0.36%)
BUFGCTRL	1/32 (3.12%)
DSPs	732/900 (81.33%)
F7 Muxes	144/109300 (0.13%)
F8 Muxes	72/54650 (0.13%)
Bonded IOB	73/362 (20.16%)
On-Chip Power	Value
Dynamic Power	4.56Watt
Static Power	0.204Watt

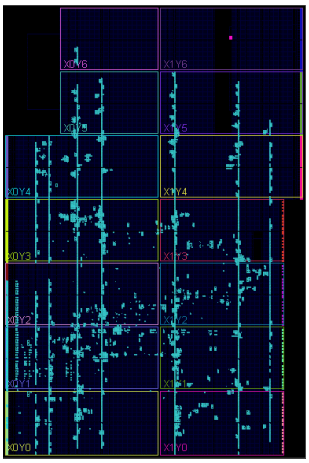


Fig. 8. Floor plan of the implemented design.

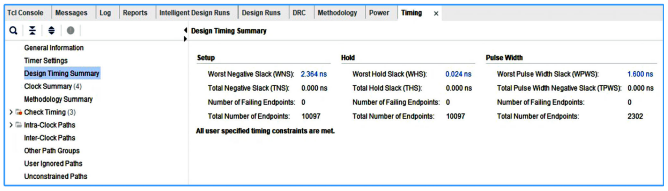


Fig. 9. Timing constraints.

the longest paths in our design from one flip-flop (register) to another. In other words, they are the paths that take the maximum time to propagate a signal from the source flip-flop to the destination flip-flop. Max delay paths are critical for ensuring that the clock frequency at which our design can operate is met. Similarly, min delay paths, on the other hand, represent the shortest paths in our design. In the case of max delay paths, the timing summary report table shows that the slack (required time–arrival time) for max delay paths is 9.160 ns, which is a positive number. The data path delay is 3.955 ns; out of this time, the delay due to logic block is 2.654 ns (67.108%), and the delay due to routing is 1.301 ns (32.898%). For our design, the clock path skew is −4.850 ns, represented by the formula destination clock delay (DCD) − source clock delay (SCD) + clock pessimism removal (CPR)

TABLE V
TABLE OF TIMING ANALYSIS REPORT

Max Delay Paths	Value
Slack (MET)	9.160ns
Requirement	20.000ns
Data Path Delay	3.955ns
Output Delay	2.000ns
Clock Path Skew	-4.850ns
Destination Clock Delay (DCD)	0.000ns
Source Clock Delay (SCD)	4.850ns
Clock Pessimism Removal (CPR)	0.000ns
Min Delay Paths	Value
Slack (MET)	0.159ns
Requirement	0.000ns
Data Path Delay	0.246ns
Clock Path Skew	0.000ns
Destination Clock Delay (DCD)	2.485ns
Source Clock Delay (SCD)	1.916ns
Clock Pessimism Removal (CPR)	0.570ns

(see Table V). One point is to observe that though the clock path skew is negative, and it will not affect our design because we have slack value to be much more than the skew value. The min delay paths are also shown in Table V, where we can observe that the slack in the case of min delay paths is 0.159 ns (positive number). The data path delay takes around 0.246 ns, out of which the delay due to logic block is 0.171 ns (69.631%), and the delay due to routing is 0.075 ns (30.369%). For our design, the clock path skew is 0 ns, represented by the formula DCD − SCD + CPR. The values of DCD, SCD, and CPR are shown in Table V. All the outputs are complex numbers of 24 bits; out of those, 12 bits from MSB are real parts, and 12 bits from LSB are imaginary parts of the resultant complex number. All the numbers we have taken in signed fixed point notation with one bit from MSB are for sign, the next 4 bits for the integer, and the last 7 bits for the fractional part. Maximum data path delay is 3.955 ns, so the minimum operating clock frequency of the FPGA board is 0.2528 GHz.

X. FUNCTIONALITY TESTING

Here, we do the functionality test of the proposed architecture. We show the timing diagram of both the transmitter and receiver and compare their output. We compared the output of both the transmitter and receiver and observed the performance. To validate our output, we compared the HDL-coded output with the MATLAB platform output. Fig. 10 shows the original constellation points and the transmitter outputs. Fig. 11 shows the timing diagram of the receiver outputs. Figs. 10 and 11 show that the original constellation and the detected symbols are almost identical, indicating that the transmitted symbols are correctly detected at the receiver. We can see that the detected symbols are not precisely the transmitted original symbols but are almost equal. This mismatch can be corrected by increasing the width of the fractional part of the number. The maximum error observed between transmitted symbols and detected symbols is within the range of ±0.07 (equivalent to 9.33%). This level of deviation is deemed acceptable for the intended application, as the symbols are detected with the correct sign bit. The significance of the proper sign bit lies in its indication of accurate detection of transmitted symbols. This

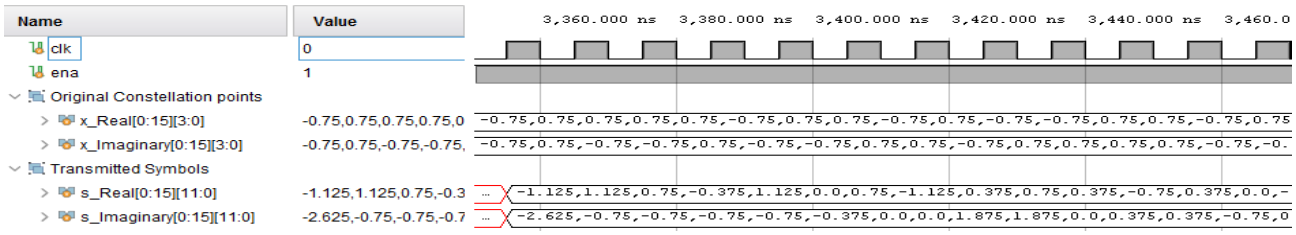


Fig. 10. OTFS transmitter outputs.

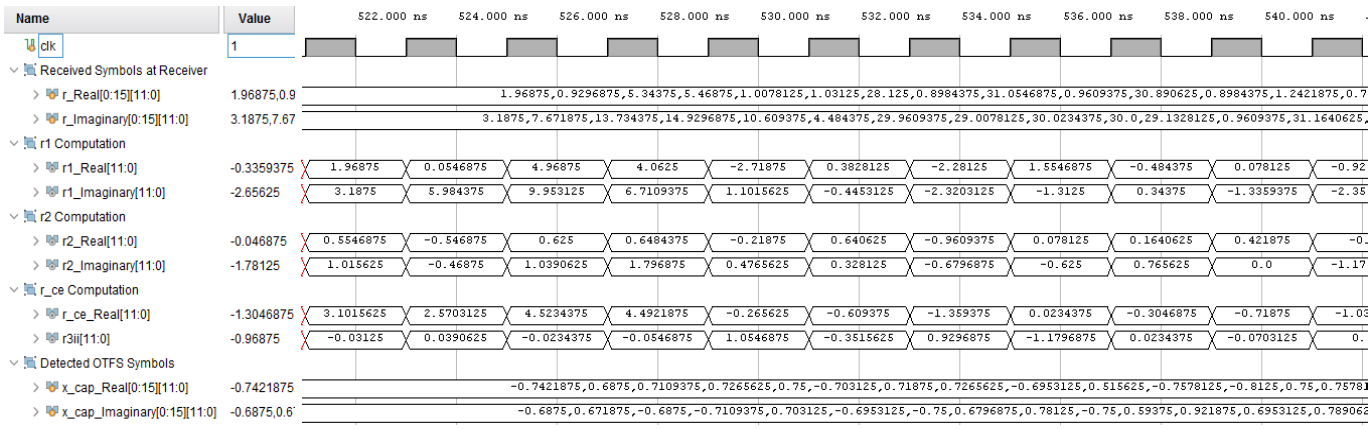


Fig. 11. OTFS receiver output.

is particularly relevant due to the utilization of QPSK for baseband modulation and demodulation, where each quadrant contains precisely one constellation point. Consequently, symbol detection with the correct signs is deemed sufficient for accurate symbol identification. Practically, the size of the delay-Doppler grid size is 512×128, which is of the size 2^{16}. However, as this article is at its preliminary stage, we take the delay-Doppler grid size to be 4×4 so that one frame contains 16 symbols, which is shown in the OTFS transmitter and receiver output. In the proposed design, we also checked the intermediate output as well, i.e., the output of every single step at the receiver [outputs of Algorithms 1–3 and outputs of (31) and (32)]. The number of symbols per frame can be increased by adapting the optimized architecture design of the OTFS transmitter and receiver for the multipath fading channel. Considering the optimized transceiver, the resource utilization and power consumption will be reduced drastically, and we can send a frame with more and more symbols.

XI. Conclusion

This work involves deriving the input–output relationship of the OTFS signal within a delay-Doppler channel, using the delay-time response of the channel as a foundation. The expression for the received OTFS signal, represented in a matrix format, has been established. A detailed investigation into BER performance, encompassing OTFS and OFDM modulation schemes, has been conducted. Given OTFS's inherent capability of harnessing the full diversity, the outcomes indicate its superiority over OFDM across diverse channel scenarios. Here, we have proposed a novel and low-complexity VLSI architecture for the OTFS transmitter and the receiver by using the LU decomposition technique. We have carried out a thorough comparison of the performance metrics of our implemented transmitter with those of the existing work. The results unequivocally demonstrate that our design operates 7.394% faster than the alternative while simultaneously exhibiting a reduction of 89.354% in the number of lookup tables (LUTs) and 79.984% in the number of flip-flops (FFs). These findings underscore the superior optimization of our design, both in terms of latency and resource utilization. We have ensured the accuracy of the Verilog code output by cross-referencing it with MATLAB simulations using identical parameter values, yielding complete agreement between the two by performing the functionality test of architecture and timing analysis. With the acceptable performance of the proposed transceiver as established above, it is deemed to be applicable to wireless mobile communication.

References

[1] L. Liang, H. Peng, G. Y. Li, and X. Shen, "Vehicular communications: A physical layer perspective," *IEEE Trans. Veh. Technol.*, vol. 66, no. 12, pp. 10647–10659, Dec. 2017.

[2] R. Hadani et al., "Orthogonal time frequency space modulation," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, San Francisco, CA, USA, Mar. 2017, pp. 1–6.

[3] H. Qu, G. Liu, L. Zhang, S. Wen, and M. A. Imran, "Low-complexity symbol detection and interference cancellation for OTFS system," *IEEE Trans. Commun.*, vol. 69, no. 3, pp. 1524–1537, Mar. 2021.

[4] C. Shen, J. Yuan, and H. Lin, "Error performance of rectangular pulse-shaped OTFS with practical receivers," *IEEE Wireless Commun. Lett.*, vol. 11, no. 12, pp. 2690–2694, Dec. 2022.

[5] A. Farhang, A. RezazadehReyhani, L. E. Doyle, and B. Farhang-Boroujeny, "Low complexity modem structure for OFDM-based orthogonal time frequency space modulation," *IEEE Wireless Commun. Lett.*, vol. 7, no. 3, pp. 344–347, Jun. 2018.

[6] R. Hadani and A. Monk, "OTFS: A new generation of modulation addressing the challenges of 5G," 2018, *arXiv:1802.02623*.

[7] P. Raviteja, E. Viterbo, and Y. Hong, "OTFS performance on static multipath channels," *IEEE Wireless Commun. Lett.*, vol. 8, no. 3, pp. 745–748, Jun. 2019.

[8] P. Raviteja, K. T. Phan, Y. Hong, and E. Viterbo, "Interference cancellation and iterative detection for orthogonal time frequency space modulation," *IEEE Trans. Wireless Commun.*, vol. 17, no. 10, pp. 6501–6515, Oct. 2018.

[9] S. S. Das, E. De Carvalho, and R. Prasad, "Variable sub-carrier bandwidths in OFDM systems," in *Proc. IEEE 65th Veh. Technol. Conf. VTC-Spring*, Apr. 2007, pp. 1866–1870.

[10] F. Wiffen, L. Sayer, M. Z. Bocus, A. Doufexi, and A. Nix, "Comparison of OTFS and OFDM in ray launched sub-6 GHz and mmWave line-of-sight mobility channels," in *Proc. IEEE 29th Annu. Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC)*, Sep. 2018, pp. 73–79.

[11] A. RezazadehReyhani, A. Farhang, M. Ji, R. R. Chen, and B. Farhang-Boroujeny, "Analysis of discrete-time MIMO OFDM-based orthogonal time frequency space modulation," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–6.

[12] P. Raviteja, Y. Hong, E. Viterbo, and E. Biglieri, "Practical pulse-shaping waveforms for reduced-cyclic-prefix OTFS," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 957–961, Jan. 2018.

[13] T. Zemen, M. Hofer, D. Löschenbrand, and C. Pacher, "Iterative detection for orthogonal precoding in doubly selective channels," in *Proc. IEEE 29th Annu. Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC)*, Sep. 2018, pp. 1–7.

[14] V. Khammammetti and S. K. Mohammed, "OTFS-based multiple-access in high Doppler and delay spread wireless channels," *IEEE Wireless Commun. Lett.*, vol. 8, no. 2, pp. 528–531, Apr. 2019.

[15] K. R. Murali and A. Chockalingam, "On OTFS modulation for high-Doppler fading channels," in *Proc. Inf. Theory Appl. Workshop (ITA)*, Feb. 2018, pp. 1–10.

[16] P. Raviteja, K. T. Phan, Q. Jin, Y. Hong, and E. Viterbo, "Low-complexity iterative detection for orthogonal time frequency space modulation," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2018, pp. 1–6.

[17] P. Wei, Y. Xiao, W. Feng, N. Ge, and M. Xiao, "Charactering the peak-to-average power ratio of OTFS signals: A large system analysis," *IEEE Trans. Wireless Commun.*, vol. 21, no. 6, pp. 3705–3720, Jun. 2022.

[18] S. K. Dora, H. B. Mishra, and M. Sahoo, "Low complexity implementation of OTFS transmitter using fully parallel and pipelined hardware architecture," *J. Signal Process. Syst.*, vol. 95, no. 8, pp. 955–964, Aug. 2023.

[19] S. Tiwari, S. S. Das, and V. Rangamgari, "Low complexity LMMSE receiver for OTFS," *IEEE Commun. Lett.*, vol. 23, no. 12, pp. 2205–2209, Dec. 2019.

[20] S. Tiwari and S. S. Das, "Circularly pulse-shaped orthogonal time frequency space modulation," *Electron. Lett.*, vol. 56, no. 3, pp. 157–160, Feb. 2020.

[21] S. Kumara and S. Deshwala, "A review on orthogonal time frequency space modulation," in *Proc. WCNC Workshop Comput. Netw. & Commun.*, 2021, pp. 1–14.

[22] D. W. Walker, T. Aldcroft, A. Cisneros, G. C. Fox, and W. Furmanski, "LU decomposition of banded matrices and the solution of linear systems on hypercubes," in *Proc. 3rd Conf. Hypercube Concurrent Comput. Appl.*, 1988, pp. 1635–1655.

[23] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th ed. Baltimore, MD, USA: Johns Hopkins Univ. Press, 2013.

[24] *Guidelines for the Evaluation of Radio Interface Technologies for IMT Advanced*, document Rep. ITU 2135-1, 2009.

[25] S. Venkatachalam and S.-B. Ko, "Design of power and area efficient approximate multipliers," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 5, pp. 1782–1786, May 2017.

[26] S. Liu and D. Liu, "A high-flexible low-latency memory-based FFT processor for 4G, WLAN, and future 5G," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 3, pp. 511–523, Mar. 2019.

[27] Y. Guo, Z. Wang, Q. Hong, H. Luo, X. Qiu, and L. Liang, "A 60-mode high-throughput parallel-processing FFT processor for 5G/4G applications," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 31, no. 2, pp. 219–232, Feb. 2023.

Ashish Ranjan Shadangi (Graduate Student Member, IEEE) received the B.Tech. degree in electronics and telecommunication engineering from the Biju Patnaik University of Technology (BPUT), Rourkela, Odisha, India, in 2015, and the M.Tech. degree in electronics and communication engineering from Odisha University of Technology and Research (OUTR), Bhubaneswar (formerly known as CET, Bhubaneswar), India, in 2019. He is currently working toward the Ph.D. degree at the G. S. Sanyal School of Telecommunications, IIT Kharagpur, Kharagpur, India.

His current research interests include VLSI architecture design for wireless communications.

Suvra Sekhar Das (Member, IEEE) received the B.Eng. degree in electronics and communication engineering from the Birla Institute of Technology, Ranchi, India, in 2000, and the Ph.D. degree from Aalborg University, Aalborg, Denmark, in 2007.

He was a Senior Scientist with the Innovation Laboratory, Tata Consultancy Services, Kolkata, India, from 2000 to 2008. He is currently a Professor with the G. S. Sanyal School of Telecommunications, IIT Kharagpur, Kharagpur, India. His current research interests include the design of waveforms, radio access technology, and radio access networks for quality of service (QoS) traffic.

Indrajit Chakrabarti (Member, IEEE) received the B.E. and M.E. degrees in electronics and telecommunication engineering from Jadavpur University, Kolkata, India, in 1987 and 1990, respectively, and the Ph.D. degree from IIT Kharagpur, Kharagpur, India, in 1997.

He is currently a Professor with the Department of Electronics and Electrical Communication Engineering, IIT Kharagpur. He has authored more than 140 articles in peer-reviewed journals and conferences. His research interests include VLSI architectures for image and video processing, digital signal processing, error control coding, and wireless communication.