Design and Implementation of a Low Complexity Multiuser Detector for Hybrid CDMA Systems

Vittorio Rampa

Original scientific paper

Abstract— **In hybrid CDMA systems, multiuser detection (MUD) algorithms are adopted at the base station to reduce both multiple access and inter-symbol interference by exploiting space-time (ST) signal processing techniques. Linear ST-MUD algorithms solve a linear problem where the system matrix has a block-Toeplitz shape. While exact inversion techniques impose an intolerable computational load, reduced complexity algorithms may be efficiently employed even if they show sub-optimal behavior introducing performance degradation and near-far effects. The block-Fourier MUD algorithm is generally considered the most effective one. However, the block-Bareiss MUD algorithm, that has been recently reintroduced, shows also good performance and low computational complexity comparing favorably with the block-Fourier one. In this paper, both MUD algorithms will be compared, along with other well known ones, in terms of complexity, performance figures, hardware feasibility and implementation issues. Finally a short hardware description of the block-Bareiss and block-Fourier algorithms will be presented along with the FPGA (Field Programmable Gate Array) implementation of the block-Fourier using standard VHDL (VHSIC Hardware Description Language) design.**

Index Terms— **Low complexity multiuser detector, hybrid CDMA system, TD-SCDMA mobile radio system, CWTS, block-Bareiss algorithm, block-Fourier algorithm, FPGA implementation.**

I. INTRODUCTION

In 3G hybrid CDMA systems with antenna arrays at the receivers, space-time (S-T) multiuser detection (MUD) algorithms are adopted at the base station to reduce both inter-symbol (ISI) and multiple access (MAI) interference [1]. For each data-packet or block, in the up-link scenario, linear MUD solves a linear system where the system matrix has a specific block-Toeplitz structure.

Exact MUD computation exhibits a high computational load for large matrix size (i.e. high number of users, antennas and/or symbols per block), as it involves the inversion of a large correlation matrix. Exact inversion of the system matrix is not feasible due to large matrix size and real-time constraints. In fact, in real-time mobile radio systems, computational resources are limited and only reduced complexity algorithms can be employed. Nevertheless, when computational complexity is strongly reduced, sub-optimal algorithms might introduce performance degradation and other unwanted effects (e.g. near-far problems).

Two main MUD techniques are available in literature: block-based MUD [2],[3],[4] and one-shot MUD also known as sliding windows detector (SWD) [5]. Both families may be implemented using real-time hardware [4],[6],[7],[8],[9],[10]. In practice, algorithm selection is performed taking into account various aspects such as performance, complexity and implementation issues. However, when computational power requirement and system complexity are the key constraints, some performance degradation have to be tolerated.

According to these considerations, a new MUD detector scheme based on the block-Bareiss (BB) algorithm has been reintroduced [11],[12]. This algorithm, derived from the plain Bareiss factorization technique [13], combines good performance and low computational load with simple hardware implementation. For a specific hybrid CDMA radio system (i.e. China Wireless Telecommunication Standard or CWTS for short), the BB detector is compared with the reference direct inversion methods and with other well known block algorithms namely the block-Levinson algorithm (BL) [14] and the block-Fourier Transform (BFT) algorithm [4].

Finally a floating-point implementation of the BB processor is compared with the equivalent BFT one. With respect to the above mentioned algorithms, the BB one is well suited for hardware FPGA (Field Programmable Gate Array) implementation not only for its low computational load but also for its good performance. In addition, it does not suffer from near-far effects that plague very low complexity implementation such as low order BFT detectors [15]. It is also worth noticing that the BB detector is also suitable for hardware parallel implementation. However, for perfect power control radio systems, the BFT remains the most effective MUD algorithm in terms of complexity and implementation feasibility. Finally, a FPGA implementation of the BFT algorithm is presented and commented.

The paper is organized as follows: Section II describes the signal model used in hybrid CDMA systems and underlines the peculiarities due to the specific CWTS standard adopted. Section III introduces the Bareiss algorithm and the other reduced complexity algorithms selected for comparison. Both performance and computational complexity of the mentioned MUD algorithms are evaluated and compared in Sections IV and V, respectively. The proposed hardware implementation is described in Section VI while Section VII draws some conclusions.

Manuscript received September 04, 2004; revised May 09, 2005 and August 31, 2005. The paper was presented in part at the Conference on Software, Telecommunications and Computer Networks (SoftCOM) 2004.

Author is with IEIIT - Sezione di Milano – CNR, Via Ponzio 34/5, Milano, 20133(MI),Italy(e-mail:rampa@elet.polimi.it).

1845-6421/05/5010

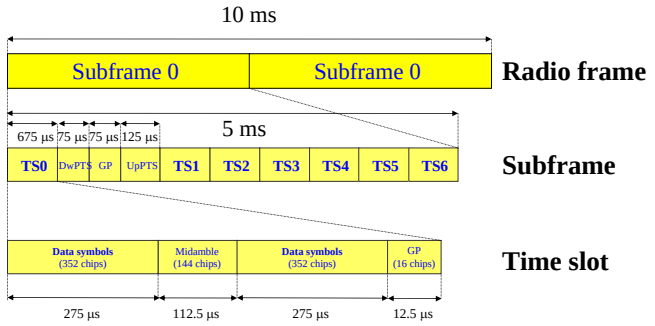


Fig. 1. Up-link burst structure for the CWTS system.

II. Hybrid CDMA signal model

In hybrid synchronous CDMA systems, such as the CWTS standard [16] whose frame structure is shown in Fig. 1, K users ($1 \leq K \leq 8$) are active in the same frequency band and in the same time slot being separated only by different spreading codes. Each user transmits a data burst consisting of $2N$ QPSK symbols ($N = 352/Q$, N symbols of duration $T_s = QT_c$) where Q is the spreading factor ($Q \in \{1, 2, 4, 8, 16\}$) and the chip rate F_c is defined as $F_c = 1/T_c = 1.28$ Mcps. In the following part of this paper, Q is assumed constant for all K users: $Q = 16$. The semi-burst period T_{sb} is defined as $T_{sb} = NQT_c = 352/F_c = 275\ \mu s$.

The column vector $\mathbf{h}_{k,m}$ represents the channel impulse response (CIR) for the link between the *k*th user and the *m*th antenna of the array. Each vector $\mathbf{h}_{k,m}$ is also assumed of length W ($W = 16$) when expressed in chip intervals T_C. Space-time matrix $\mathbf{H}_k = [\mathbf{h}_{k,1}...\mathbf{h}_{k,m}...\mathbf{h}_{k,M}]^T$ for the *k*th user has size $M \times W$ while the complete multi-user channel matrix $\mathbf{H} = [\mathbf{H}_1...\mathbf{H}_K]$ has size $M \times WK$. Each CIR is assumed known and does not vary during the time slot.

For each semi-burst, the discrete-time base-band MIMO (Multiple Input Multiple Output) signal model is indicated by the following equation:

$$\mathbf{y} = \mathbf{A}\mathbf{d} + \mathbf{n}. \tag{1}$$

The $M(NQ + W - 1) \times 1$ vector $\mathbf{y}$ represents the signal received by the array of M antennas ($1 \leq M \leq 8$) located at the base station while vector $\mathbf{d}$ of size $NK \times 1$ represents the transmitted data for all K users. Moreover, it is $E\left[\mathbf{d}\mathbf{d}^H\right] = \mathbf{I}$. Noise vector $\mathbf{n}$ takes care of both electronic noise and inter-cell interference; $\mathbf{n}$ is assumed spatially correlated and temporally uncorrelated with covariance matrix given by:

$$\mathcal{R}_n = E[\mathbf{n}\mathbf{n}^{\mathrm{H}}] = \mathbf{R}_n \otimes \mathbf{I}_{NQ+W-1}, \tag{2}$$

where $\otimes$ is the Kronecker's product, I_{NQ+W-1} is the identity matrix of size $NQ + W - 1$ and $\mathbf{R}_n$ is the spatial covariance matrix ($[\mathbf{R}_n]_{m,m} = \sigma_n^2$ for $m = 1, ..., M$) of size $M \times M$.

Since the CIRs are assumed constant in the data slot, the matrix $\mathbf{A}$ of size $M(NQ + W - 1) \times NK$ containing N shifted copies of blocks $\mathbf{B}$. Fig. 2 shows that every sub-matrix $\mathbf{B}$ of size $M(Q + W - 1) \times K$ is composed by K column vectors $\mathbf{b}_k$ ($k = 1, ..., K$); each of them is composed by the convolution $\mathbf{b}_{k,m} = \mathbf{c}_k * \mathbf{h}_{k,m}$ between the *k*th spreading codes

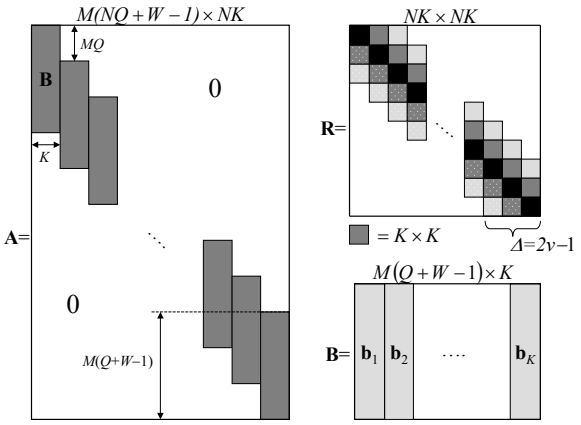


Fig. 2. Matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{R} = \mathbf{A}^{\mathrm{H}}\mathcal{R}_n^{-1}\mathbf{A}$.

$\mathbf{c}_k$ and the CIR vector $\mathbf{h}_{k,m}$ ordered for all M antennas: $\mathbf{b}_k = [\mathbf{b}_{k,1}^T...\mathbf{b}_{k,M}^T]^T$. The normalized term $v = \lceil (Q + W - 1)/Q \rceil$ is the delay spread expressed in symbol intervals.

In the following sections we will assume, without any loss of generality, a zero-forcing (ZF-MUD) detector scheme. The interested reader may refer to [1] for a detector that employs a different estimation technique (e.g. MMSE-MUD). According to Fig. 3, the received signal vector $\mathbf{y}$ is at first filtered by the whitening S-T matched filter producing the $NK \times 1$ output vector:

$$\mathbf{y}_{\mathrm{MF}} = \mathbf{A}^{\mathrm{H}}\mathcal{R}_n^{-1}\mathbf{y} = \mathbf{A}^{\mathrm{H}}\mathcal{R}_n^{-1/2}\mathcal{R}_n^{-\mathrm{H}/2}\mathbf{y}; \tag{3}$$

then linear ZF-MUD is performed by estimating the data symbols vector $\bar{\mathbf{d}}$ of size $NK \times 1$ according to:

$$\bar{\mathbf{d}} = \mathbf{R}^{-1}\mathbf{y}_{\mathrm{MF}} = \left(\mathbf{A}^{\mathrm{H}}\mathcal{R}_n^{-1}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathrm{H}}\ \mathcal{R}_n^{-1}\mathbf{y}, \tag{4}$$

where $\mathbf{R}^{-1}$ is the $NK \times NK$ decorrelating matrix obtained from matrix $\mathbf{R} = \mathbf{A}^{\mathrm{H}}\mathcal{R}_n^{-1}\mathbf{A}$. Finally, hard decided QPSK symbols $\hat{\mathbf{d}}$ are obtained by the threshold detector $\mathrm{dec}(\cdot)$:

$$\hat{\mathbf{d}} = \mathrm{dec}\left(\bar{\mathbf{d}}\right) = \mathrm{dec}\left(\mathbf{R}^{-1}\mathbf{y}_{\mathrm{MF}}\right). \tag{5}$$

Equation (3) may be rewritten as:

$$\mathbf{y}_{\mathrm{MF}} = \widetilde{\mathbf{A}}^{\mathrm{H}}\widetilde{\mathbf{y}} = \mathbf{R}\mathbf{d} + \mathbf{n}_{MF} \tag{6}$$

to indicate that the S-T matched filter output $\mathbf{y}_{\mathrm{MF}}$ may be also obtained by calculating the output vector $\widetilde{\mathbf{y}} = \mathcal{R}_n^{-\mathrm{H}/2}\mathbf{y}$ of the first spatial whitening filter (see Fig. 3) and then applying the matched filter $\widetilde{\mathbf{A}} = \mathcal{R}_n^{-\mathrm{H}/2}\mathbf{A}$; $\mathbf{n}_{MF}$ is the noise at the output of the S-T matched filter with $E\left[\mathbf{n}_{MF}\mathbf{n}_{MF}^H\right] = \mathbf{R}$. Please note that, in the case of MMSE-MUD, equation (5) becomes now [1]:

$$\hat{\mathbf{d}} = \mathrm{dec}\left(\bar{\mathbf{d}}\right) = \mathrm{dec}(\left(\mathbf{R} + \mathbf{I}\right)^{-1}\mathbf{y}_{\mathrm{MF}}). \tag{7}$$

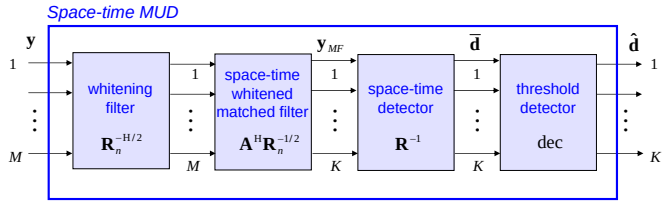


Fig. 3. Block diagram of the S-T ZF-MUD.

III. Efficient detector algorithms

Most of the complexity of the linear detector algorithms employed for MUD scheme proposed in the previous section arises from the inversion methods of the large correlation matrix $\mathbf{R}$ and from the matched filter computation [8]. As depicted in Fig. 2, matrix $\mathbf{R}$ is block-Toeplitz and block-banded. It is composed by $N \times N$ sub-matrices $\mathbf{R}_i$ $(i = 0, ..., N-1)$ having size $K \times K$:

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_0 & \mathbf{R}_{-1} & \dots & \mathbf{R}_{-(N-1)} \\ \mathbf{R}_1 & \mathbf{R}_0 & \dots & \dots \\ \dots & \dots & \dots & \mathbf{R}_{-1} \\ \mathbf{R}_{(N-1)} & \dots & \mathbf{R}_1 & \mathbf{R}_0 \end{bmatrix}, \tag{8}$$

where $\mathbf{R}_{-i} = \mathbf{R}_i^H$ $(i = 1, ..., N-1)$. Moreover, matrix $\mathbf{R}$ has only $2v-1$ non null block-diagonals: $\mathbf{R}_{-i} = \mathbf{R}_i^H = 0$, for $i = v, v+1, ..., N-1$.

As far as the matched filter output $\mathbf{y}_{\mathrm{MF}}$ is concerned, its computational cost dominates using an antenna array ($M >> 1$); however its calculation has a high degree of parallelism that can be easily exploited during the hardware implementation (e.g. using a polyphase Q-decimated filter bank) [8].

Some MUD algorithms compute directly the decorrelating matrix $\mathbf{R}^{-1}$ and then calculate the solution (4) by matrix multiplication. On the contrary, other algorithms factorize matrix $\mathbf{R}$ and then solve the equation:

$$\mathbf{R}\bar{\mathbf{d}} = \mathbf{y}_{\mathrm{MF}}. \tag{9}$$

While the block-Levinson algorithm [14] belongs to the first class of techniques, the latter family includes the well known block-Fourier (BFT) [17] algorithm and all methods derived from the QR decomposition [18] of the matrix $\mathbf{R}$. To this second family belongs also the block-Bareiss algorithm that will be discussed later on.

All the methods that belong to the second family are similar to the plain Cholesky algorithm and differ only in the used factorization algorithm that takes care of the block-Toeplitz matrix $\mathbf{R}$ having in common the remaining processing steps. For instance, the Cholesky algorithm computes the Cholesky factor $\mathbf{U}$ of the matrix $\mathbf{R}$: $\mathbf{R} = \mathbf{U}^{\mathrm{H}}\mathbf{U}$. Then, both the lower triangular system:

$$\mathbf{U}^{\mathrm{H}}\mathbf{z} = \mathbf{y}_{\mathrm{MF}} \tag{10}$$

and the upper triangular one:

$$\mathbf{U}\bar{\mathbf{d}} = \mathbf{z}, \tag{11}$$

are computed sequentially by using the backward and forward (B/F) algorithm, respectively. The Cholesky factor $\mathbf{U}$ of a block-Toeplitz matrix $\mathbf{R}$ is block-banded but not block-Toeplitz. However, the Cholesky factor structure shown in [19] may be exploited to approximate the factorization and to speed-up the computation if $N >> v$. This approximation may be used together with the well known generalized Schur algorithm [17] to speed up the whole computation. It is worth mentioning that the B/F algorithm has a complexity in the order of $\mathcal{O}[N^2K^2]$.

A. Factorization algorithms

The exact computation of the Cholesky factor $\mathbf{U}$ for a arbitrary $NK \times NK$ matrix $\mathbf{R}$ requires a number of operations in the order of $\mathcal{O}[N^3K^3]$ that is prohibitive for a large number of symbols N and/or users K. For this reason, several fast inversion or factorization algorithms have been developed.

B. Block-Fourier algorithm

The BFT algorithm [4] is derived from the plain Fourier algorithm briefly recalled here. The Fourier technique solves the generic problem:

$$\mathbf{R}\mathbf{x} = \mathbf{y}, \tag{12}$$

where $\forall h, k = 0, ..., n-1$ it is $[\mathbf{R}]_{h,k} = r_{h,k}$, $r_{h,k}$ depends only on the difference $k-h$ and $r_{h,h} = r_0$. The solution is computed by transforming equation (12) into the frequency domain by exploiting the nth order DFT matrix operator $\mathbf{F}$ defined as:

$$[\mathbf{F}]_{h,k} = \exp\left\{-j2\pi(h-1)(k-1)/n\right\}, \tag{13}$$

where n is the problem size and $h, k = 1, ..., n$. If the matrix $\mathbf{R}$ is circulant, then it is well known [18] that $\mathbf{R}$ may be factorized as:

$$\mathbf{R} = \mathbf{F}^{-1}\mathbf{\Lambda}\mathbf{F}, \tag{14}$$

where the matrix $\mathbf{\Lambda}$ is the diagonal matrix containing the eigenvalues of $\mathbf{R}$. It may be noticed that if $\mathbf{A}$ is circulant then $\mathbf{R}$ is circulant, too.

The convolution problem (12) becomes now:

$$\mathbf{F}\mathbf{R}\mathbf{x} = \mathbf{\Lambda}\mathbf{F}\mathbf{x} = \mathbf{F}\mathbf{y}, \tag{15}$$

while the solution vector $\mathbf{x}$ is obtained by inverse DFT transformation $\mathbf{F}^{-1}$:

$$\mathbf{x} = \mathbf{F}^{-1}\mathbf{\Lambda}^{-1}\mathbf{F}\mathbf{y}. \tag{16}$$

The Fourier method is computationally effective because the diagonal matrix $\mathbf{\Lambda}$, that contains the eigenvalues of $\mathbf{R}$, can be efficiently computed by transforming only the first column $\mathbf{r}_1$ of the matrix $\mathbf{R}$ according to:

$$\boldsymbol{\lambda} = \mathbf{F}\mathbf{r}_1, \tag{17}$$

where column vector $\boldsymbol{\lambda}$ is obtained from eigenvalue matrix $\mathbf{\Lambda}$ by rearranging its diagonal elements by means of the diag operator:

$$\boldsymbol{\lambda} = \mathrm{diag}\,(\mathbf{\Lambda}) = \mathbf{\Lambda}\,\mathbf{1}_n \tag{18}$$

and $\mathbf{1}_n$ is the column vector of length n having all terms equal to one.

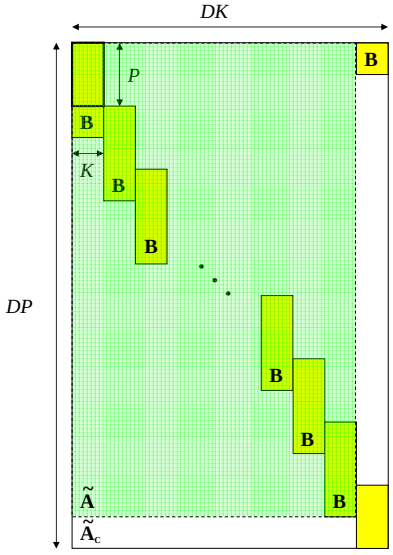


Fig. 4. Structure of circulant matrix $\widetilde{\mathbf{A}}_c$ used by the Block-Fourier algorithm. The matrix size is $DP \times DK$ where $D = N + v - 1$ and $P = MQ$. It is also indicated the original matrix $\widetilde{\mathbf{A}}$.

Unfortunately, the matrix $\mathbf{R}$ is not circulant, it is also block-Toeplitz and block-band but it has only $2v-1$ non null block-diagonals (Fig. 2). It may be easily made circulant with few modifications: the circulant block matrix $\widetilde{\mathbf{A}}_c$ is obtained from $\widetilde{\mathbf{A}}$ just adding and arranging columns of blocks as shown in Fig. 4. The number of added blocks depend on the value v: in our case, the matrix $\widetilde{\mathbf{A}}_c$ has size $DP \times DK$ where $D = N+v-1$ and $P = MQ$ while circulant matrix $\mathbf{R}_c = \widetilde{\mathbf{A}}_c^H \widetilde{\mathbf{A}}_c$, derived from $\widetilde{\mathbf{A}}_c$, has size $DK \times DK$.

Now, the plain Fourier algorithm must be modified to handle the block-circulant matrix $\mathbf{R}_c$ containing D^2 blocks of size $K \times K$. Instead of the Fourier matrix (13), now it is defined the block-Fourier matrix transformation $\mathbf{F}_{(K)} = \mathbf{F} \otimes \mathbf{I}_K$ where $\otimes$ is the Kronecker's product, $\mathbf{I}_K$ is the $K \times K$ identity matrix, $\mathbf{F}_K$ and $\mathbf{F}$ represent the *K*th and the *D*th order DFT, respectively.

The equation equivalent to (14) is:

$$\mathbf{R}_c = \mathbf{F}_{(K)}^{-1} \mathbf{\Lambda}_{(K)} \mathbf{F}_{(K)}, \tag{19}$$

where $\mathbf{\Lambda}_{(K)}$ is the block-diagonal matrix composed of D non null blocks of size $K \times K$ along the main block-diagonal. It is computed (cf. Eq.17):

$$\mathrm{diag}_{(K)} \left(\mathbf{\Lambda}_{(K)} \right) = \mathbf{F}_{(K)} \mathbf{R}_{\mathbf{c}1}, \tag{20}$$

where $\mathbf{R}_{\mathbf{c}1}$ is the first block column of the matrix $\mathbf{R}_c$ and $\mathrm{diag}_{(K)}(\cdot)$ is the extension of the diag operator (18) to the $K \times K$ block size case.

Finally, the BFT algorithm computes the following matrix equation corresponding to (15):

$$\mathbf{F}_{(K)} \bar{\mathbf{d}} = \mathbf{\Lambda}_{(K)}^{-1} \mathbf{F}_{(K)} \mathbf{y}_{MF} \tag{21}$$

and then applying the inverse DFT operator $\mathbf{F}_{(K)}^{-1}$:

$$\bar{\mathbf{d}} = \mathbf{F}_{(K)}^{-1} \mathbf{\Lambda}_{(K)}^{-1} \mathbf{F}_{(K)} \mathbf{y}_{MF}. \tag{22}$$

The block elements of block-diagonal matrix $\mathbf{\Lambda}_{(K)}$ have no particular structure; therefore $\mathbf{\Lambda}_{(K)}^{-1}$ may be obtained with standard (or approximated) methods such as the Cholesky factorization. In addition, the matched filter output $\mathbf{y_{MF}}$ is obtained from equation (6). Only the first NK values of $\bar{\mathbf{d}}$ from the equation (22) are used to compute the vector $\hat{\mathbf{d}}$.

It is possible to speed up the block-Fourier algorithm by reducing the ideal length D of the FFT (i.e. using optimized radix-4 operators) with respect to the true value $N+v-1$ and by using the well-known overlap-and-save technique. It requires the use of $L = \lceil N/(D - \mathrm{prelap} - \mathrm{postlap}) \rceil$ data vector slices of reduced size D to cover N symbols [4].

C. Block-Levinson algorithm

The BL algorithm is derived from the plain Levinson algorithm that computes the direct problem $\mathbf{Rx} = \mathbf{y}$ by inverting the matrix $\mathbf{R}$ that is Hermitian, Toeplitz and positive defined. For a generic matrix $\mathbf{R}$ of size $n \times n$, this technique requires a number of operations in the order of $\mathcal{O}[n^2]$. The BL algorithm has been extended [14],[20] to the block-Toeplitz matrices $\mathbf{R}^{(i)}$ of size $NK \times NK$ by solving the following system at step $i+1$:

$$\mathbf{R}^{(i+1)} \bar{\mathbf{d}}^{(i+1)} = \mathbf{y}_{\mathrm{MF}}^{(i+1)} \tag{23}$$

starting from the solution at step i. The original problem (9) is solved through $N-1$ iterations. By defining matrix $\mathbf{R}^{(i)}$ as:

$$\mathbf{R}^{(i)} = \begin{bmatrix} \mathbf{R}_0 & \mathbf{R}_{-1} & \dots & \mathbf{R}_{-i} \\ \mathbf{R}_1 & \mathbf{R}_0 & \dots & \dots \\ \dots & \dots & \dots & \mathbf{R}_{-1} \\ \mathbf{R}_i & \dots & \mathbf{R}_1 & \mathbf{R}_0 \end{bmatrix}, \tag{24}$$

the system (23) may be rewritten as ($1 \le i \le N-1$):

$$\begin{bmatrix} \mathbf{R}^{(i)} & \mathbf{E}_i \left(\mathbf{G}^{(i)} \right)^T \\ \left(\mathbf{G}^{(i)} \right)^T \mathbf{E}_i & \mathbf{R}_0 \end{bmatrix} \begin{bmatrix} \bar{\mathbf{d}}^{(i)} \\ \boldsymbol{\mu} \end{bmatrix} = \\ = \begin{bmatrix} \mathbf{y}_{\mathrm{MF}}^{(i)} \\ \mathbf{y}_{\mathrm{MF}}^{(i,i+1)} \end{bmatrix}, \tag{25}$$

where $\bar{\mathbf{d}}^{(i)}$ and $\mathbf{y}_{\mathrm{MF}}^{(i)}$ are sub-vectors of length iK obtained from the vectors $\bar{\mathbf{d}}$ and $\mathbf{y}_{\mathrm{MF}}$, respectively. The vector $\mathbf{y}_{\mathrm{MF}}^{(i,i+1)}$, that is extracted from the matched filter output $\mathbf{y}_{\mathrm{MF}}$, has size $K \times 1$. Matrix $\mathbf{G}^{(i)}$ is defined as $\mathbf{G}^{(i)} = \left[\mathbf{R}_1^T \mathbf{R}_2^T \dots \mathbf{R}_i^T \right]^T$, $\mathbf{E}_i$ is the exchange block matrix of size $iK \times iK$ that inverts the order of the blocks of the previous matrix $\mathbf{G}^{(i)}$ and $\boldsymbol{\mu}$ is an auxiliary vector [14] of size $K \times 1$.

As indicated in the BFT algorithm, it is also possible to increase the algorithm processing speed by considering that, after few iterations, some internal parameters converge rapidly to their final value and may therefore considered constant. Reference [4] shows a detailed description about these optimizations.

D. Block-Bareiss algorithm

The plain Bareiss algorithm [13] employs an iterative technique, derived from the classical Schur algorithm, to solve a

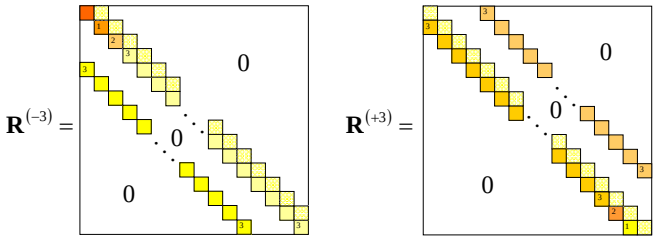


Fig. 5. Structure of matrices $\mathbf{R}^{(-i)}$ and $\mathbf{R}^{(+i)}$ used in the Bareiss algorithm for $i = 3$. The example refers to tri-diagonal block matrix $\mathbf{R}$ found in the CWTS standard with $v = 2$.

generic linear Toeplitz system $\mathbf{Rx} = \mathbf{y}$ by LU factorization of the positive definite system matrix $\mathbf{R}$ of size $n \times n$. The complexity of the algorithm is in the order of $\mathcal{O}[n^2]$. The basic idea is to transform the original system (12) in the following equivalent ones (for $1 \leq i \leq n-1$):

$$\mathbf{R}^{(\pm i)}\mathbf{x} = \mathbf{y}^{(\pm i)}, \tag{26}$$

where the matrices $\mathbf{R}^{(-i)}$ have zero elements along the i sub-diagonals below the main diagonal and matrices $\mathbf{R}^{(+i)}$ have zero elements along the i sub-diagonals above the main diagonal. As shown in Fig. 5 for $i = 3$, $\mathbf{R}^{(-3)}$ is an upper triangular matrix while $\mathbf{R}^{(+3)}$ is a lower triangular matrix. These transformations are chosen to annihilate diagonals rather than columns or rows. Each transformation requires a number of operations in the order of $\mathcal{O}[n]$.

Let $\mathbf{Z}_i = (\delta_{w-j+i})_{w=0,\ldots,n-1;j=0,\ldots,n-1}$ be a shift matrix such that pre-multiplication by $\mathbf{Z}_{+i}$ shifts i rows up with zero fill and pre-multiplication by $\mathbf{Z}_{-i}$ shifts i rows down with zero fill. For the ith iteration ($i = 1, \ldots, n-1$), the corresponding transformations are defined as:

$$\begin{aligned}
\mathbf{R}^{(-i)} &= \mathbf{R}^{(-(i-1))} - m_{-i}\mathbf{Z}_{-i}\mathbf{R}^{(+(i-1))} \\
\mathbf{y}^{(-i)} &= \mathbf{y}^{(-(i-1))} - m_{-i}\mathbf{Z}_{-i}\mathbf{y}^{(+(i-1))} \\
\mathbf{R}^{(+i)} &= \mathbf{R}^{(+(i-1))} - m_{+i}\mathbf{Z}_{+i}\mathbf{R}^{(-(i))} \\
\mathbf{y}^{(+i)} &= \mathbf{y}^{(+(i-1))} - m_{+i}\mathbf{Z}_{+i}\mathbf{y}^{(+(i))},
\end{aligned} \tag{27}$$

where m_{-i} and m_{+i} are defined as:

$$m_{-i} = r_{i,0}^{(-(i-1))}/r_0 \tag{28}$$

and

$$m_{+i} = r_{0,i}^{(+(i-1))}/r_{n-1,n-1}^{(-i)}. \tag{29}$$

After $n-1$ iterations, $\mathbf{R}^{(-(n-1))}\mathbf{x} = \mathbf{y}^{(-(n-1))}$ and the system may be solved with backward substitution. This method is equivalent to the LU factorization of the matrix $\mathbf{R} = \mathbf{LU}$: $r_0\mathbf{L} = \left(\mathbf{R}^{(n-1)}\right)^{T2}$ and $\mathbf{U} = \mathbf{R}^{(-(n-1))}$ where $T2$ denotes matrix transposition above the main diagonal [21].

The Bareiss algorithm has essentially the same numerical properties as Gaussian elimination without pivoting: it is numerically stable if $\mathbf{R}$ is positive definite or diagonally dominant, but it is unstable in general [22].

The block version of the Bareiss algorithm still employs the same ideas behind equations (26) but now transformations try to annihilate block-diagonals of size $K \times K$ of the matrix (8) instead of single elements. After i iterations, the matrix $\mathbf{R}^{(-i)}$ has null blocks along the i sub-diagonals below the main diagonal and the matrix $\mathbf{R}^{(+i)}$ has null blocks along the i sub-diagonals above the main diagonal. The transformations of the ith iteration ($i = 1, \ldots, N-1$) are defined by the following equations:

$$\begin{aligned}
\mathbf{R}^{(-i)} &= \mathbf{R}^{(-(i-1))} - \mathbf{M}_{-i}\mathbf{Z}_{-i}\mathbf{R}^{(+(i-1))} \\
{\mathbf{y_{MF}}}^{(-i)} &= {\mathbf{y_{MF}}}^{(-(i-1))} - \mathbf{M}_{-i}\mathbf{Z}_{-i}{\mathbf{y_{MF}}}^{(+(i-1))} \\
\mathbf{R}^{(+i)} &= \mathbf{R}^{(+(i-1))} - \mathbf{M}_{+i}\mathbf{Z}_{+i}\mathbf{R}^{(-i)} \\
{\mathbf{y_{MF}}}^{(+i)} &= {\mathbf{y_{MF}}}^{(+(i-1))} - \mathbf{M}_{+i}\mathbf{Z}_{+i}{\mathbf{y_{MF}}}^{(-i)},
\end{aligned} \tag{30}$$

where:

$$\mathbf{Z}_i = (\delta_{w-j+i})_{w=0,\ldots,N-1;j=0,\ldots,N-1}, \tag{31}$$

$$\mathbf{M}_{-i} = \mathbf{R}_{i,0}^{(-(i-1))}\left(\mathbf{R}_0\right)^{-1} \tag{32}$$

and

$$\mathbf{M}_{+i} = \mathbf{R}_{0,i}^{(+(i-1))}\left(\mathbf{R}_{N-1,N-1}^{(-i)}\right)^{-1}. \tag{33}$$

After $N-1$ iterations, the system indicated by equation $\mathbf{R}^{(-(N-1))}\bar{\mathbf{d}} = {\mathbf{y_{MF}}}^{(-(N-1))}$ may be solved by a block backward substitution since $\mathbf{R}^{(-(N-1))}$ is an upper triangular block matrix.

E. Remarks

For the sake of completeness, the generalized Schur algorithm with the Cholesky approximation (ACD - Approximate Cholesky Decomposition) [19] can reduce the number of operations in the order of $\mathcal{O}[NK^3]$. For a complete description of this algorithm, the interested reader is referred to [17]. In addition, references [8] and [23] show a brief comparison of the ACD algorithm performance with respect to the SWD one in the TDD-UTRA and the CWTS scenario, respectively.

All these MUD algorithms may have different behavior depending on one or more parameters: FFT size (D), overlap-and-save parameters (prelap, postlap or L) greatly affect BFT algorithm while iteration step (i) is the main parameter for BL and BB detectors. These parameters greatly affect performance, computational complexity and architectural issues. Fine tuning of these parameters depends on the adopted mobile radio scenario and will be evaluated in Section IV.

IV. Simulation Results

Following the CWTS standard [16], the parameters employed in the system simulations are: $K = 8$, $N = 22$, $Q = 16$ and $W = 16$. The simulation scenario is characterized by $100,000$ bursts of up-link traffic data only with scrambling and spreading codes, without channel coding and with perfect power control scheme. The spatio-temporal channel matrix $\mathbf{H}$ adopted for the simulations is derived from the Typical Urban (TU) multipath propagation channel as introduced by the COST-207 group [24]. More sophisticated S-T channel models are available; however, this simpler model has been adopted for easy and direct comparison with respect to spatial only algorithms ($M = 1$).

According to the WSSUS (Wide Sense Stationary Uncorrelated Scattering) model introduced previously, for the kth user, channel $\mathbf{H}_k$ consists of a single cluster of $N_p = 12$

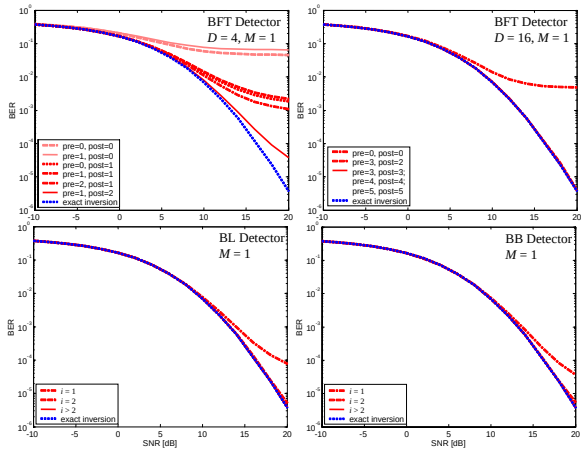


Fig. 6. Performances of MUD algorithms in terms of BER vs. SNR for different design parameters (D, prelap, postlap, i). Clockwise from top left plot: *i*) BFT-4; *ii*) BFT-16; *iii*) BB and *iv*) BL detectors. Coefficient values adopted for simulations are: $M = 1$, $K = 8$, $Q = 16$, $W = 16$ and $v = 2$.

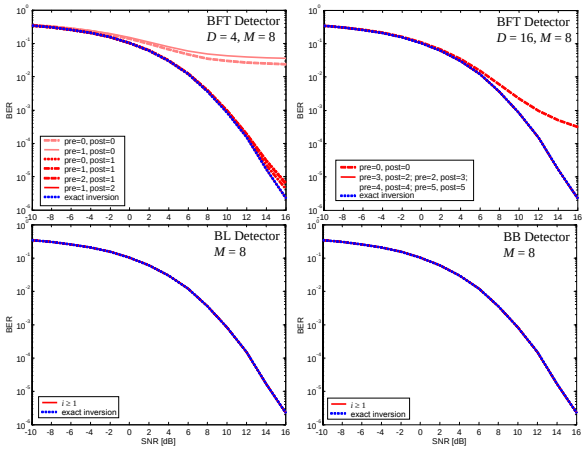


Fig. 7. Performances of MUD algorithms in terms of BER vs. SNR for different design parameters (D, prelap, postlap, i). Clockwise from top left plot: *i*) BFT-4; *ii*) BFT-16; *iii*) BB and *iv*) BL detectors. Coefficient values adopted for simulations are: $M = 8$, $K = 8$, $Q = 16$, $W = 16$ and $v = 2$.

uncorrelated paths, whose delays and mean powers are fixed and set according to the TU model. Each *p*th path angle is a random variable:

$$\theta_{k,p} = \mathcal{N}\left(\theta_k, \sigma_\theta^2\right) \tag{34}$$

with mean value $\theta_k = \mathcal{U}\left[-\pi/3, +\pi/3\right]$ and standard deviation $\sigma_\theta = \pi/36$. Amplitude variations due to fast fading are simulated by introducing the Rayleigh distribution. In addition, to speed-up the simulations, no mobile speed is considered since the maximum speed constraint (i.e., $|\mathbf{v}| \leq v_{\max} = 120$ Km/h) introduced by the CWTS standard does not cause significant channel decorrelation during the semiburst period T_{sb}. In fact, according to the Clarke model [25], the correlation of the CIR fading amplitudes is described by the Jakes' autocorrelation function $\rho_0(\tau) = J_0(2\pi v_m \tau/\lambda)$ where λ is the carrier wavelength ($\lambda = 15$ cm) and v_m the radial mobile speed. In our case, it is $\rho_0(T_{sb})|_{v_m = v_{\max}} = 0.96$ and therefore the channel may be considered constant during each time slot (see reference [26] for similar considerations about the TDD-UTRA scenario). Perfect knowledge of the channel is assumed while near-far effects are not present: $E\left[\|\mathbf{H}_k\|^2\right]$ is constant for all K users. For discussions about near-far effects of all algorithms presented here, reference [15] may be consulted. The base station receiver employs a single antenna ($M = 1$) or a linear array ($M = 8$) of equally spaced antennas at $\lambda/2$.

In Fig. 6 and 7, the performance of the approximate MUD and the exact inversion algorithms are compared in terms of BER for varying signal to noise ratio at the antenna array receiver $SNR = QE\left[\|\mathbf{H}_k\|^2\right]/2\sigma_n^2$ in the case of a single antenna ($M = 1$) and antenna array ($M = 8$), respectively. In these figures, the block-Fourier algorithms are indicated as BFT-4 or BFT-16 if $D = 4$ or $D = 16$ is adopted, respectively. The other algorithm parameters are indicated in the figures.

In the single antenna scenario, for two or more iterations ($i \geq 2$) both BB and BL algorithms have performance similar to the one corresponding to the exact system inversion method. On the contrary, the BFT algorithm strongly depends on the FFT size D and the prelap (pre) and postlap (post) values. In fact, for $D = 16$, pre > 0 and post > 0, the BFT detector performance is close the the optimum one, while if $D = 4$ any combination of pre and post coefficients cannot obtain figures near to those corresponding to the exact inversion case being combination pre $= 1$, post $= 2$ the most effective ($L = 22$).

For the linear antenna array scenario with $M = 8$ elements, performance is significantly better than the previous one due to the effect of the array spatial processing. All algorithms show good performance except the BFT one for the artifacts introduced by the overlap-and-save technique (e.g. $D = 16$, pre $= 0$, post $= 0$; $D = 4$, pre $= 0$, post $= 0$; $D = 4$, pre $= 1$, post $= 0$ and $D = 4$, pre $= 0$, post $= 1$). It is worth noticing that only one iteration is enough for BB and BL algorithms to converge to the exact solution.

V. Algorithm Computational Complexity

Tab. I shows the computational complexity of BB, BFT and BL algorithms in terms of complex multiplications ($\times 10^5$) for each semi-burst. The shown algorithms perform the matched filter in the time domain (e.g. BL and BB) or frequency domain (e.g. BFT). Matrix $\mathbf{R}$ is tri-diagonal ($v = 2$) block-Toeplitz, with a lot of null blocks. Several algorithm optimizations have been adopted to reduce both computational complexity and storage size of the algorithms shown in Section III. For instance, in the BB algorithm it is possible to: *i*) reduce the matrix multiplications of (30) to block multiplications, *ii*) rearrange the matrices of blocks $\mathbf{R}^{(+i)}$ and $\mathbf{R}^{(-i)}$ as one column of blocks of variable size for each modified diagonal, *iii*) reuse the already inverted blocks in the backward substitution phase [12].

Fig. 8 tries to summarize the information from Tab. I and Fig. 7. It is apparent that both Bareiss and Fourier (BFT-4) algorithms are well suited for low complexity detectors. On the contrary, the Levinson algorithm, even achieving good

TABLE I

COMPUTATIONAL COMPLEXITY OF DETECTOR ALGORITHMS IN TERMS OF COMPLEX MULTIPLICATIONS ($\times 10^5$) PER SEMI-BURST. BFT-4 AND BFT-16 INDICATE THE BFT ALGORITHM WITH $D = 4$ AND $D = 16$, RESPECTIVELY.

Parameters	Algorithm	Computational complexity	
		$M = 1$	$M = 8$
$L = 4$, pre $= 5$, post $= 5$	BFT-16	0.27	1.19
$L = 3$, pre $= 4$, post $= 4$		0.22	1.01
$L = 2$, pre $= 2$, post $= 3$		0.18	0.83
$L = 22$, pre $= 2$, post $= 1$	BFT-4	0.25	1.19
$L = 11$, pre $= 1$, post $= 1$		0.14	0.68
$L = 6$, pre $= 0$, post $= 0$		0.09	0.45
$i = 21$	BB	1.43	1.97
$i = 5$		0.41	0.94
$i = 3$		0.28	0.82
$i = 1$		0.15	0.69
$i = 21$	BL	3.68	4.21
$i = 5$		1.06	1.59
$i = 3$		0.91	1.45
$i = 1$		0.81	1.34

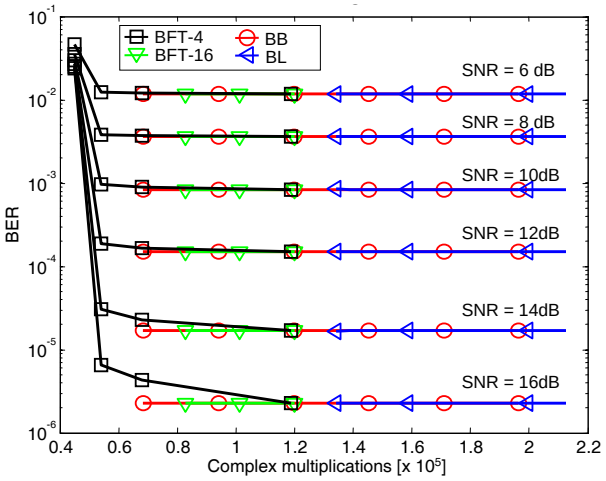


Fig. 8. Performances vs. computational complexity of MUD algorithms at different SNR conditions. Each dot of a curve indicates the approximation degree adopted for the algorithm at that specific SNR. From left to right it is: $L = 6, 11, 22$ for BFT-4 ($D = 4$), $L = 2, 3, 4$ for BFT-16 ($D = 16$), $i = 1, 5, 9, 13, 17, 21$ for BB and BL. Simulation values are the same employed in Fig. 7.

performance, has high computational complexity and it is not appropriate for a simple hardware implementation.

VI. HARDWARE IMPLEMENTATION

Fig. 9 and 10 show the block diagrams of the BB and BFT-4 algorithm implementation. The BFT-4 algorithm may be efficiently designed exploiting pipelined structures and parallel data units (see Fig. 10). However, the BB detector may be parallelized [21] and pipelined, too. Several MUD implementations for hybrid CDMA systems are available; however, while the BFT-4 FPGA implementation is straightforward, the other ones requires more sophisticated hardware architecture such as a FPGA systolic array [23],[9] or an array of DSP/processors such as the RCF (Reconfigurable Computer Fabric) processor [10].

To avoid numerical problems, floating-point arithmetics have been extensively used having care not to exceed the size

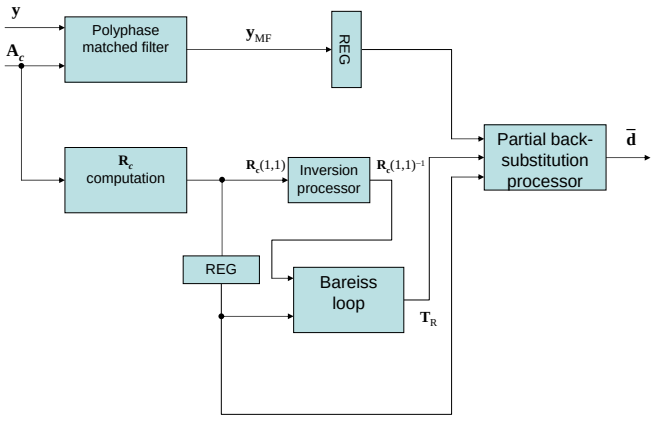


Fig. 9. Block diagram of the BB algorithm ($i = 1$). The REG elements are register blocks.

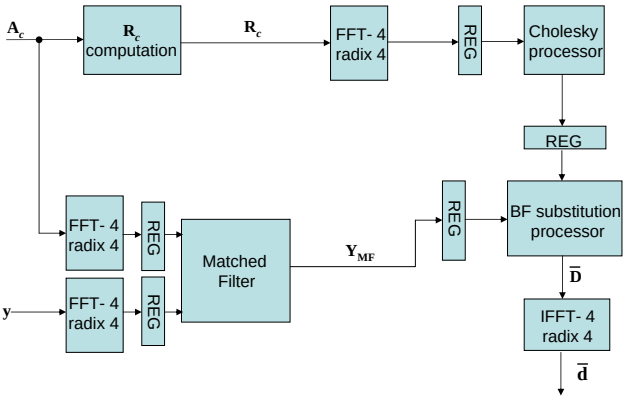


Fig. 10. Block diagram of the BFT algorithm ($D = 4$, pre $=$ post $= 1$). The REG elements are register blocks.

of the FPGA built-in multipliers. The floating-point numerical format is indicated as (N_t, N_e) where N_e is the exponent and $N_t - N_e$ is the mantissa size. Both detector implementations have been simulated adopting the same scenario parameters of Section IV and their performances shown in Fig. 11 according to the selected floating-point format. The FPGA implementation of the detectors dictates for the smallest data size: in both cases, the $(10, 5)$ format is enough to have figures very close to the ones corresponding to the full precision simulations of Fig. 7 and still to be under the size of the FPGA built-in multipliers. The bad performances of the $(10, 4)$ format are due to overflow/underflow problems.

From the implementation point of view, comparing *only* the design complexity of both BFT and BB processors, it is apparent that the BFT-4 detector is the most suited for hardware implementation due to its straightforward design. In fact, in spite of its performance, the design of the BFT-4 processor is very simple. The BFT-4 main elements are the radix-4 FFT/IFFTs that may be easily implemented without multipliers while the Cholesky computation element and the matched filter may be designed with fast pipelined multiply-and-accumulate (MAC) blocks. Assuming a latency of one time slot (i.e. 675 μs) and estimating about 180 μs for ancillary computations (e.g. channel estimation, whitening filtering,

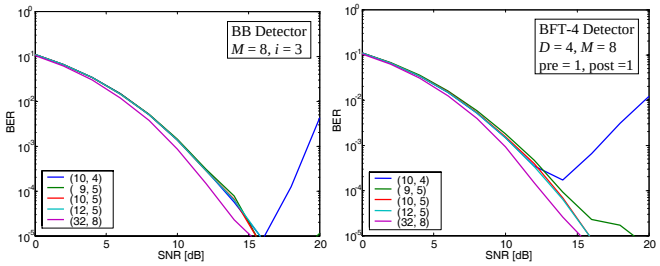


Fig. 11. Performance changes due to the floating-point numerical format (N_t, N_e) adopted for the algorithm implementations. From left to right: *i*) BB ($i = 3$) and *ii*) BFT-4 ($D = 4$, pre $=$ post $= 1$) performances vs. numeric formats. The simulation results correspond to the block diagrams of Fig. 9 and 10.

Fig. 12. Floorplan of the FPGA implementation of the BFT-4 detector using a Virtex-II XC2V6000 device (-6FF1517). No I/O pads have been plotted.

midamble interference cancellation), the delay introduced by the detector must be less than 495 μs.

The complete detector has been described using VHDL and designed using Mentor Graphics© and Xilinx© tools. The target FPGA is the Virtex-II XC2V6000 device from Xilinx©. In this design, about 60% of the configurable cells (CLB - Complex Logic Block) and 90% of internal RAM blocks has been used. The achieved clock frequency is 21.8 MHz [12] that meet the time constraints. The floorplan is depicted in Fig. 12.

VII. Conclusions

Performance and complexity evaluation of CWTS multiuser detectors indicate that both BB and BFT algorithms may be efficiently adopted. When hardware implementation efficiency is mandatory, the BFT-4 algorithm is the best solution for its straightforward design. However, the BB detector may be also used to solve performance problems of the Fourier detector at a comparable complexity.

Acknowledgments

The author would like to thank C. Madè and C. Meregalli for the Simulink© software simulator of the CWTS system and L. Costa and E. Borello for FPGA simulation and implementation of the BFT-4 detector. The author would like also to show appreciation to M. Nicoli and A. Bifano for helpful discussions and to the Reviewers for their useful remarks and suggestions.

References

[1] S. Verdú: *Multi-user Detection*, Cambridge University Press, 1998.
[2] Y. Pigeonnat: *Joint detection for UMTS: complexity and alternative solutions*, Proc. IEEE Proc. of VTC 1999, Vol. 1, pp. 546-550, 1999.
[3] H.R. Karimi, N.W. Anderson: *A novel and efficient solution to block-based joint-detection using approximate Cholesky factorization*, Proc. IEEE Proc. of PIMRC 1998, Vol. 3, pp. 1340-1345, 1998.
[4] M. Vollmer, M. Haardt, J. Götze: *Comparative Study of Joint-Detection Techniques for TD-CDMA based Mobile Radio Systems*, Sel. Areas in Communications, Vol. 19, N. 8, pp. 1461-1475, Aug. 2001.
[5] M.J. Juntti, B. Aazhang: *Finite memory length linear multiuser detection for asynchronous CDMA communications*, IEEE Trans. on. Comm. Vol. 54, N. 5, pp. 611-622, May 1997.
[6] J. Mayer, J. Schlee, T. Weber: *Realtime Feasibility of Joint Detection CDMA*, Proc. of 2^{nd} European Personal Mobile Communications Conference, pp. 245-252, Bonn, Germany, Sept. 1997.
[7] T. Weber, J. Schlee, S. Bahrenburg, P.W. Baier, J. Mayer, C. Euscher: *A Hardware Demonstrator for TD-CDMA*, IEEE Trans. on Vehi. Tech., Vol. 51, N. 5, pp. 877-892, Sept. 2002.
[8] M. Beretta, A. Colamonico, M. Nicoli, V. Rampa, U. Spagnolini: *Space-Time multiuser detectors for TDD-UTRA: design and optimization*, IEEE Proc. VTC 2001, Vol. 1, pp. 78-82, 2001.
[9] D. Noguet: *A reconfigurable systolic architecture for UMTS/TDD joint detection real time computation*, IEEE Proc. of ISSSTAT, pp. 957-961, Sidney, Australia, Sept. 2004.
[10] E. Martinez: *Introduction to TD-SCDMA on the MRC6011 RCF Device*, Freescale Semiconductor Application Note AN2684, Rev. 1, 11/2004.
[11] A. Bifano, V. Rampa: *Low Complexity Multiuser Detectors for TD-SCDMA Systems: Design and Implementation*, Proc. of SoftCOM 2004, Split, pp. 444-448, Oct. 2004.
[12] E. Borello, L. Costa: *Algoritmi e Architetture efficienti per la rivelazione multi-utente nel sistema CWTS*, Thesis as Dottore in Ingegneria (in italian), Dipartimento di Elettronica e Informazione, Politecnico di Milano, 2004.
[13] E.H. Bareiss: *Numerical solution of linear equations with Toeplitz and vector Toeplitz matrices*, Numer. Math. Vol. 13, pp. 404-424, 1969.
[14] R. A. Wiggins, E. A. Robinson: *Recursive solutions to the multichannel filtering problem*, J. Geophys. Res., Vol. 70, pp. 1885–1891, 1965.
[15] A. Bifano, V. Rampa: *Multiuser Detector for hybrid SCDMA Systems based on the Bareiss algorithm*, IEEE Proc. of ICASSP 2005, Philadelphia, Vol. III, pp. 909-912, Mar. 2005.
[16] China Wireless Telecommunication Standard (CWTS) Working Group 1 (WG1): *Physical channels and mapping of transport channels onto physical channels*, TS C102 v3.3.0 (2000-09).
[17] M. Vollmer, M. Haardt, J. Götze: *Schur algorithms for joint-detection in TD-CDMA based mobile radio systems*, Ann. Telecommun. Vol. 54, N. 7-8, pp. 365-378, 1999.
[18] G.H. Golub, C.F. Van Loan: *Matrix computation*, The John Hopkins University Press, 1991.
[19] J. Rissanen: *Algorithms for Triangular Decomposition of Block Hankel and Toeplitz Matrices with Application to Factoring Positive Matrix Polynomials*, Math. Computations, Vol. 27, pp. 147-154, Jan 1973.
[20] A.E. Yagle: *Multichannel Coupled Split Algorithms for Non-Hermitian Block-Toeplitz Matrices*, IEEE Trans. on Signal Processing, Vol. 41, N. 1, 505-508, Jan. 1993.

[21] R.P. Brent: *Parallel algorithms for Toeplitz systems*, Numerical Linear Algebra, Digital Signal Processing, Proc. NATO ASI, Leuven, Belgium, August 1988, edited by G. H. Golub and P. Van Dooren, NATO ASI Series F: Computer and Systems Sciences, Vol. 70, Springer-Verlag, 1991.

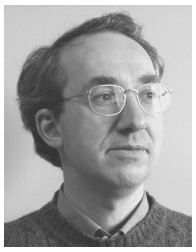
[22] A.W. Bojanczyk, R.P. Brent, F.R. de Hoog and D.R. Sweet: *On the stability of the Bareiss and related Toeplitz factorization algorithms*, SIAM J. Matrix Analysis and Applications, Vol. 10, 225-244, 1995.

[23] A. Bifano, A. Colamonico, M. Nicoli, V. Rampa, U. Spagnolini: *Sliding windows multiuser detectors for TD-SCDMA systems*, Proc. of European Conference on Wireless Technology - ECWT '02, Session E3, September 2002.

[24] COST-207: *Digital land mobile radio communications*, Final Report, Luxemburg, Office for Official Publications of the European Communities, 1989.

[25] G.L. Stuber: *Principles of Mobile Communications*, Kluwer Academic, 1996.

[26] M. Nicoli, M. Sternad, U. Spagnolini, A. Ahlen, *Reduced-rank channel estimation and tracking in time-slotted CDMA systems*, Proc. IEEE International Conference on Communications (ICC '02), vol. 1, pp. 533-537, April-May 2002.



Vittorio Rampa was born in Genoa, Italy, in 1957. He received the Laurea (with Honors) as Dottore in Ingegneria Elettronica in 1984 from the Politecnico di Milano. In 1986 he joined the CSTS-CNR (Center for Space Communications) of the Italian National Research Council (CNR) now IEIIT-CNR (Institute of Electronics, Computer and Telecommunication Engineering) where he is Senior Researcher. He was a Visiting Scholar at Center for Integrated Systems, Stanford University during 1987-1988. Since 1999 he is also Contract Professor at Politecnico di Milano with research in telecommunication architectures. Presently, his research interests include signal processing for telecommunication and radar systems and hardware/software reconfigurable architectures for mobile communication systems.