Realtime Parallel Software Implementation of a DS-CDMA Multiuser Detector

Luís Carlos Gonçalves*, Rui Escadas Martins†,
António Brito Ferrari†
*Instituto de Telecomunicações, 3810-193 Aveiro, Portugal
†Instituto de Engenharia Electrónica e Telemática de Aveiro (IEETA), Universidade de Aveiro, 3810-193 Aveiro, Portugal
Emails: lgoncalves@av.it.pt, rmm@ua.pt, ferrari@ua.pt

***Abstract*—In this article the complexity and runtime performance of two Multiuser Detectors for Direct Sequence-Code Division Multiple Access were evaluated in two different hardware platforms. The innovation and aim is to take advantage of present parallel hardware to bring Multiuser technology to present and future Base Stations in order to increase the capacity of the overall system, to reduce the transmission power by the mobile stations and to reduce base station hardware requirements, in Universal Mobile Telecommunications System. The detectors are based on the Frequency Shift Canceller concatenated with a Parallel Interference Canceller. This detector implies the inversion of multiple identical size small matrices and because of that it is very scalable contrary to other solutions/detectors that only permits a sequential implementation despite their lower complexity. Implementations for the Time Division-Code Division Multiple Access, in two software platforms one in OpenMP and the other in CUDA were done taking into account the carrier and doppler frequency offsets (offset different for each user). The result shows that this deployment aware real-time implementation of the Multiuser Detectors is possible with a Graphics Processor Unit being three times faster than required.**

***Index Terms*—Heterogeneous Computing, Real Time Implementation, High Performance Computing, Frequency Shift Canceller, Parallel Interference Canceller, Multiuser Detection**

I. Introduction

Third Generation Universal Mobile Telecommunications System-Time Division Duplex (UMTS-TDD) specs define three chip rates for transmission: 7.68MChips/s, 3.84MChips/s and 1.28MChips/s. The latter is the one used in the People's Republic of China since 2007 with 1.28Mchips/s and it is named Time Division-Synchronous Code Division Multiple Access (TD-SCDMA).

The hardware of the base stations are actually upgraded several times during a decade to meet the technology advances. Better Multiuser Detection (MUD) has the potential to increase the spectral efficiency and wireless network coverage in the uplink of base stations, as to increase the energy efficiency in mobile stations. It can also decrease the number of diversity antennas in the base station, thereby decreasing costs in hardware and also increasing energy efficiency. Depending on the age and provider of those base stations the upgrade can be done through a board connected to a backplane or connected through optic fiber to a standalone card or computer.

MUD algorithms could be deployed in mobile station receivers and base stations receivers from the UMTS-TDD standard in all chip rates: 1.28 MChips/s, 3.84 MChips/s and 7.68 MChips/s. In this work a possible implementation in base stations, in uplink is studied. MUD application to the uplink is transparent to the mobile stations with the specifications having no restrictions about using it.

At uplink the signal received at the base station has passed through different (transmission) channels. MUD is used in the receiver and acts over the sampled spread signal at baseband with the goal of cancelling the other user's signals (Multiuser Access Interference (MAI)) to recover the user of interest.

The use of MUD includes some single user detector functionality because it needs to deal with the channel distortion of the portion of the received signal related to the user of interest. The MUD detectors like the Minimum Mean Square Error Detector (MMSE) and the Frequency Shift Canceller (FSC) can be integrated in a RAKE [1] (composed structures) and can be concatenated with a Parallel Interference Canceller (PIC) or a Serial Interference Canceller (SIC) to improve even more its performance. The concatenation with a SIC is more appropriate for downlink because of the power differences of the users' signals components of the signal received in each mobile station and with a PIC for uplink because of similar receiving signals power of the different users in the base station. In [2]–[4] such composite structures with multiuser, single user and spatial processing using configurations including the FSC concatenated with a PIC were studied.

The order of complexity of the optimal multiuser detector (OMUD) [5], or maximum likelihood detector, is exponential and hence not physically realizable. Different algorithms to reduce the complexity and find solutions whose Bit Error Rate (BER) comes close to the optimum have been proposed.

The FSC belongs to the category of Frequency Shift Filters (FRESH) [6] which has structures that use the existing correlation between frequency bands of man-made signals.

The MMSE detector [7]–[9] implies the inversion of a large diagonal matrix typically with LsUxLsU size (Ls is the number of symbols in a slot and U the number of users). This is in an ideal case, as typically the upsampling

and channel length must also be taken into account. It is to be expected that much larger memory resources would be needed for the implementation of such algorithm. Also the MMSE Algorithm is not so scalable as the FSC given the latter has decoupled user processing and multiple small matrix inversions.

Despite Iterative Multiuser Detection being claimed as a less complex solution [10], it is an iterative procedure and so its implementation must be sequential (serial). Also, it might not converge to the right solution.

Genetic Algorithms-based multiuser detectors have been proposed by a number of authors ([11], [12] and citations within). These algorithms are not as scalable in terms of parallel computation because the software code used to recover concurrently DS-CDMA users diverges. To the best of our knowledge, no implementations of such algorithms that satisfy the strict timing requirements have been reported.

With the emergence of integrated parallel processor architectures and the availability of parallel extensions to programming languages [13], [14] (like OpenMP, and CUDA both extensions of C) many algorithms that were previously too complex can now be efficiently implemented in software. This paper reports on the implementation of DS-CDMA MUD in multicore processors and in parallel heterogeneous architectures with GPUs. There has been previous work reported on the use of GPUs for interference cancellation in CDMA communications [15]. The execution times reported are in the range of seconds to tens of seconds for 20,000 bits, whereas a real-time implementation requires a maximum execution time of 1.4ms for 1,408 bits.

As far as is the knowledge of the authors, this work is innovative in the sense that it is the first realtime implementation of DS-CDMA MUD in parallel architectures with Graphical Processor Units (GPUs), making possible their incorporation in base stations.

In [16], [17] can be found single elementar tasks than can be done by GPUs and in [18] with other Parallel Architectures like the Intel Xeon Phi. In [19], [20] can be found full algorithms implemented in GPUs.

In the past, simpler Multiuser Detectors (PICs and Sequential Detectors) were implemented in Field-Programmable Gate Arrays (FPGAs) [21]–[23] and GPUs [15]. And signal processing algorithms applied to Radars were explored in [24].

Previous work of the authors, concerning this subject was presented in [25].

In Section II the basics of DS-CDMA systems, and details of the MUD detectors implemented, are presented. In Section III the implementation done of the Multiusers Detectors, in OpenMP and CUDA are described. In Section IV the complexity and performance analysis are analyzed. And finally in Section V, the main conclusions are stated.

II. Detectors with the Frequency Shift Canceller

A baseband Direct Sequence-Spread Spectrum (DS-SS) signal, for one user and spreading sequences of spreading factor of Q_{max}, is represented at the receiver (base station) by

$$s_i^l(t) = \sum_k a^i_{kD+l}\, g_i^l(t - kT) \tag{1}$$

where $\{a^i_{kD+l}\}$ is the information symbol sequence, l is the index of the spreading sequence , i is the user index, $1/T$ is the symbol rate and $g_i^l(t)$ is the *signature waveform*. $g_i^l(t)$ is given by

$$g_i^l(t) = \sum_{q=0}^{Q_{\max}-1} \widetilde{c}_i^l p(t - qT_c) * h_i(t) \tag{2}$$

for the maximum spreading factor of the system (SF=Q_{max}), where $\{\widetilde{c}_i^l\}$ is the spreading sequence of index l, $p(t)$ the normalized elementary pulse[1], T_c is the chip period, $h_i(t)$ is a linear filter representing the impulsional response of the transmission channel and the symbol * represents the convolution operation.

Multiple signals $s_i^l(t)$, of multiple transmitting users and corresponding spreading sequences, are received and are superimposed in the time and frequency in the base station.

Table I
Simulation Parameters

Number of Users	16
Number of Antennas	2
Number of Taps per antenna	2
Spreading Factor	16
Chip Rate	1.28 MChips/s
Modulation	QPSK
Channel	GBSBEM [26], [27]
Mobiles Speed	50 Km/h
PathLoss Exponent	3.8
Maximum Delay Spread	4.0 μs
UpSampling Factor	8
Line of Sight Distance of Mobiles	600 m
Number of bits simulated	10 Million

A Data Symbol is a complex number and it can represent several bits encoded in phase and amplitude. The mapping of Bits in Symbols is named modulation *i.e.* Quadrature Phase Shift Keying (QPSK), 8-Phase Shift Keying (8-PSK), 16-Quadrature Amplitude Modulation (16-QAM) with different mapping each one.

For proper operation of the Multiuser Detector a discrete version of the signature sequence represented by $g_i^l(t)$ must be generated/replicated in the receiver.

The mobile stations are commanded in such way that the signals in the base station in one uplink slot are superimposed and due to non-ideal conditions not totally synchronized. The policy of a Multiuser Detector is to cancel the interfering users signals, named Multiple Access Interference (MAI) from the user of interest. Due to the effect of the Transmission Channels even with synchronism the user's signals are not orthogonal. Such detectors also have some single user functionality (*i.e.* RAKE) that reduces the impairment introduced by the Transmission Channel of the user of interest.

The signal to noise ratio can also be improved by having

[1]Impulse Responde of Raised Cosine

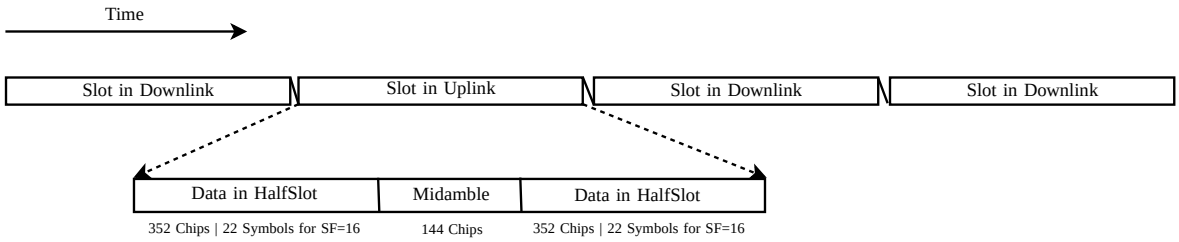

Figure 1. Transmission in Slots. Definition of HalfSlots. Information for 1.28MChips/s

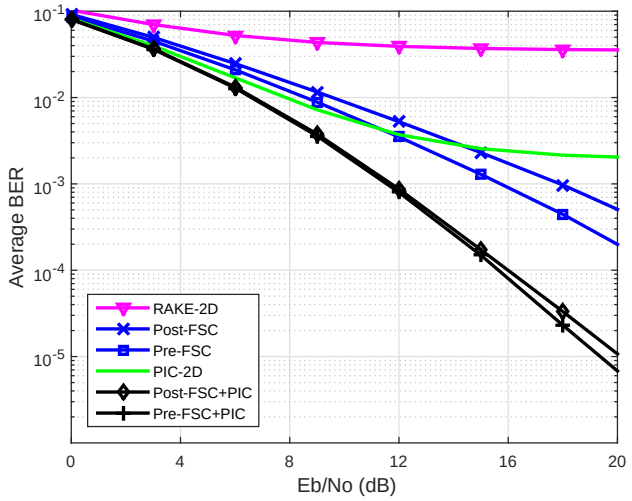

Figure 2. Performance (BER) for QPSK, SF=16 and 2 receiver diversity antennas.

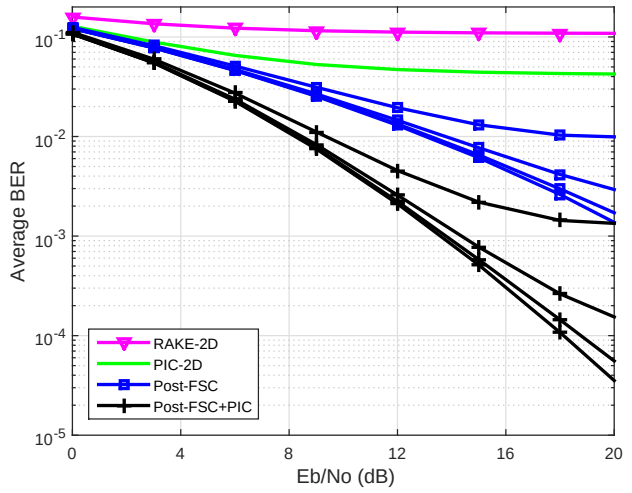

Figure 3. Performance (BER) for QPSK, SF=16 and 2 receiver diversity antennas with different upsamples in the receiver. Curves with decreasing performance (upwards) of Post-FSC and Post-FSC+PIC: non-degradation (or ideal) case, upsample of 16, 8 and 4. The curves of the RAKE and PIC are for non-degradation (or ideal) case.

several receiving antennas in the base station.

The format of a slot in a TD-SCDMA frame is depicted in Figure 1. In [2]–[4], [25] are defined two configurations for the Multiuser detector one named Pre-FSC [25, Fig. 1] and the other Post-FSC [25, Fig. 2] either concatenated with a Hard-PIC [25, Fig. 3]. These schematics are reflected in Algorithm 1 and Algorithm 2 respectively. A good description of the FSC canceler can be found in [2].

In the case of a standalone implementation, the Frequency Shift Algorithm is well adapted to single user processing (different from joint detection in MMSE where all users must be detected at the same time) because the processing is decoupled from the other users even if it needs to know which users are active. The algorithm proposed in [2]–[4] does not need large matrix inversions (18x18 for 1.28 Mchips (China) and 3.84Mchips/s (Europe)).

The proposed implementation is for 1.28MChips/s but it is easily configurable for 3.84MChips/s. Despite the fact that in this work, the case of 16 users of spreading factor (SF) of 16 is treated, the detector supports the mix of other lower spreading factors. For example, one user of spreading factor 4 is treated as 4 users (16/SF=4) of spreading factor 16 [2]–[4], in both the FSC and the PIC. Also the detector allows a mix of QPSK, 8PSK and 16-QAM modulations.

This detector is valid for Beamforming and for Spatial Diversity if it is given the correspondent channel to do the processing. The frequency offset impairment between the carrier in the transmitter plus the doppler offset due to movement and the reference carrier in the receiver can be compensated at the end of the receiver chain because each user spread spectrum signal remains cyclostationary with that offset. It is considered that the doppler offset is equal for the paths in each user transmission channel. Because the midamble interval (Figure 1) and the Carrier Frequency Offset, the bits on each side of the slot must be recovered separately. In the case of joining the two sides, each user signal looses the cyclostationary.

Simulations of the BER versus the Energy of the Bit to Noise Ratio (Eb/N0) were performed with the parameters given by Table I, for the Pre-FSC and Post-FSC configurations. These parameters were chosen in order to reflect a medium size cell (600m line of sight, $4\mu s$ Maximum Delay Spread) in a typical high damping scenario (3.8 Path Loss Exponent). Figure 2 depicts the results. All the taps are aligned with the samples by default (non-degradation (or ideal) of performance case) and the upsample is the same in the whole simulation chain. While the Pre-FSC in standalone performs slight better than the Post-FSC, when concatenated with the PIC the performance is significantly better and both configurations, Pre-FSC+PIC and Post-FSC+PIC, have similar performances.

In real conditions the first tap, with greater amplitude, of each user's channel are not aligned[2] between the users, causing an increase in the BER. Figure 3 shows how the BER improves when upsampling factors of 4, 8 and 16 are

[2]The others taps are already not aligned.

Algorithm 1 Multiuser Detector Code Description of Pre-FSC+PIC Configuration

```
Load data (Burst, Channel)
Generate Signatures Waveforms in Discrete Fourier domain without Channel impairment
Generate Fast Fourier Transform (FFT) of Root Raised Cosine(RRC) and Raised Cosine
Start statistics, Start counting time
Filter with Root Raised Cosine the input Burst (one for each antenna), It is kept a discrete time domain and a discrete frequency domain copy, BURSTANTn (discrete time domain)
Generate the Noise Power Density at the input of the FSCs (from the estimate of the noise power, number of symbols per user in a half slot and the RRC filter)
Parallel begin nusers
    Generate Signatures Waveforms with Channel Impairment for the user correspondent to the thread and each antenna
    Barrier
    for nantennas do
        Frequency Shift Canceller
        Matching Filter to user channel at the antenna
        Accumulate
    end for
    Downsampling (correspondent operation in Discrete Fourier domain) by the upsampling factor
    Matching Filter to the Spreading Code
    Downsampling(correspondent operation in Discrete Fourier domain)by the Maximum Spreading Factor of the system
    Inverse Fast Fourier Transform
    Symbol Demodulation
    Reconstruction of the user signal with Channel impairment from the bits for each antenna, USERiANTn
    Barrier
    Sum of the all users reconstructed signals for each antenna and with channel impairment, SUMANTn (operation divided by the threads equally). Each thread sum, one subset of the samples, through the users at each antenna.
    Barrier
    for nantennas do
        Cancellation (BURSTANTn-SUMANTn+USERiANTn)
        Matching Filter (discrete time domain operation) to the antenna user channel
        Accumulate
    end for
    Downsampling by the upsampling factor
    Correlation (equivalent to Matching Filter) to the Spreading Code
    Downsampling by the Maximum Spreading Factor of the system
    Symbol Demodulation
Parallel end
Stop statistics, Stop counting time
```

Algorithm 2 Multiuser Detector Code Description of Post-FSC+PIC Configuration. Only shown the diferences from Algorithm 1

```
...
...
Parallel begin nusers
    Generate Signatures Waveforms with Channel Impairment (the channel includes Maximum Ratio Combining before FSC) correspondent to the thread (nusers signatures each thread)
    for nantennas do
        Matching Filter to user channel at the antenna
        Accumulate
    end for
    Frequency Shift Canceller
    ...
    ...
Parallel end
...
```

used for the Post-FSC and Post-FSC+PIC. Contrary to the simulations in Figure 2 the profile of each user transmission channel suffers a delay with an uniform distribution between zero (0) and 4 Chips ($3.125\mu s$) representing the imprecision of the time advance[3]. In this simulation chain the sampling factor of the Transmitter and the Channel was 128 (8x16) increasing the Channel time representation precision. Figure 3 shows that with an upsampling of 16 the BER curve is very close to the "ideal" case, represented by the bottom curve. It also shows that for Eb/No < 12 an upsampling of 8 virtually has no degradation in BER, *i.e.* in such conditions there is no advantage in using higher upsampling factors. The curves of the performance of the RAKE and the PIC are for the case when the taps are aligned with the samples by default (ideal). The curves for the Pre-FSC(+PIC) are not represented because they are similar, but not equal, to the Post-FSC(+PIC).

[3]The advance in time that each mobile station must provide in transmission in order that all users signals are synchronized (by leading tap) in the base station.

III. Implementation

Both the Pre-FSC+PIC and Post-FSC+PIC were implemented in serial code, in OpenMP and in CUDA (version 10.0) in a computer with a i9-9900K CPU (8 cores, with

AVX2) with a RTX2070 Graphic Processing Unit (GPU) and a computer with i7-8750H Processor (6 cores, AVX2) with a GTX1050Ti GPU. Both CPUs feature hyperthreading. Eclipse with *gcc* was used in both platforms. The CPUs and GPUs are connected through a PCIe3 bus in both platforms.

Nvidia GPU architecture has evolved through several generations, featuring increased GFlops ratings and faster and more unified CPU-GPU memory models. The RTX2070, based in the Turing architecture features 36 Streaming Multiprocessors (SMXs) each with 64 streaming processor (SP) cores. The 4 MByte Outer Cache is common to all the SMXs. The GTX1050Ti is based on the Pascal architecture. It has 6 SMXs, each with 128 SP cores, and a 1MByte outer cache.

The serial code was used to measure the complexity in Millions of Single Precision Floating Point Operations with PAPI[4] and to take the reference runtime in both machines. The serial code runtimes were taken at maximum/Turbo 5GHz Clock of the i9-9900K and at maximum/Turbo 4.1GHz Clock of the i7-8750H.

The comparison between the single precision and the double precision versions showed that single precision had less than 1% relative error in the symbols (before quantization) recovered by the FSC and the PIC. Hence, single precision was used for the computations.

The target time to recover both Half Slots (HSs) is about 1.4ms considering one single carrier with half of the slots used for the uplink. In order to take full advantage of the parallel processing power of the GPUs, instead of processing 1 HS at a time a set of 36 half slots were acquired to be processed simultaneously. This maximizes the use of the SMXs present in each architecture while keeping the extra delay introduced within bounds that do not affect the quality of the communications, corresponding to a maximum latency of 25.7ms[5] which added to the processing time must be below 150ms, the maximum acceptable delay in one way call path. The latency of acquiring the HSs for the processing will be shorter in the case of a basestation with multiple carriers or/and multiple scrambling codes where the 36 HSs can be obtained faster. The serial code was evaluated for 1 HS, the runtime for 36 HSs being 36x higher.

OpenMP was used to parallelize the code in both platforms. Algorithm 1 presents the description of the Multiuser Detector Code for the Pre-FSC+PIC configuration and Algorithm 2 shows the differences from Algorithm 1 for the Post-FSC+PIC configuration. The implementation is the same for the i9 and the i7. Because the processing for recovering each user in the detector with FSC is (almost) decoupled of the processing to recover the other users, 16 threads, one for each user, were created. The PIC has an identical structure to the Pre-FSC but the equivalent operations between the two are made in the time domain instead. As can be seen inside the parallel section of the code of both algorithms there are barrier instructions to synchronize the data between threads. The first barrier, present only in Algorithm 1, guarantees that the Channel Impaired Signature Waveforms of each user (thread) needed by all threads are all generated when needed. The second barrier, in both Algorithms, guarantees that the reconstruction of each spread signal (in each thread) from the bits detected by the detector with FSC is completed when needed. This operation is part of the PIC detector. The third and last barrier, also in the PIC code, guarantees the generation of the sum of all reconstructed users when needed.

The algorithms were also implemented using CUDA. The implementations made for the RTX2070 and GTX1050Ti were the same with no differences in the grid and block sizes because the number of SMXs in the RTX2070 is a multiple of the number in the GTX1050Ti. The implementation was made with more than 20 kernels for each implementation (two Pre-FSC+PIC implementations and two Post-FSC+PIC implementations). Each kernel is called once, and does the processing for the 16 users, 36 HSs at once. There is only one copy from host (CPU) to device (GPU) with the initial data and another from device to host with the final bits (data of all 36 HSs). Between host to device kernel calls the data remains all the time in the GPU external memory. Synchronization between the kernels (like the barriers in OpenMP) was not needed. Since there is no CPU code between kernels, consecutive kernels in the same CUDA stream are serialized. The block sizes for the kernels were dimensioned in order to use the maximum number of SMX registers with the active warps[6]. Generally, a maximum block size of 128 (1024 possible) was used, mostly for simple wide equal operations kernels like the combining or accumulate. A reduction of more than 10% in global execution time was achieved with this kind of optimizations. With more complex kernels (like the FSC) and high degree of parallelism the amount of L1 cache used by each thread is very small and has little impact on the performance. For the case of 1HS it was necessary to reduce the block size and increase the number of blocks in order to have a better distribution of the processing through all the SMXs. It was not necessary to use atomic operations. Also there is little code divergence[7] accross the kernels. There was no runtime advantage to use nested parallelism either in CUDA or in OpenMP. In the case of the Post-FSC+PIC implementations, given that the Downsamplings and Matched Filter had the same grid and block sizes, they were included in the FSC kernel with almost no increase in the runtime relative to the FSC when processed alone, due to the reduction of time in passing the data to the next kernel.

In the Pre-FSC configuration the number of signatures generated (in each HS), which are discrete frequency signals, is equal to the number of users times the number of antennas (see Algorithm 1) and in the Post-FSC configuration the number of signatures generated is equal to the number of users squared (see Algorithm 2). Hence, in the latter the number of signature waveforms does not increase

[4] http://icl.cs.utk.edu/papi/

[5] 14 slots allocated to uplink in 2 frames and 4 slots of a third frame for a total of 25.7ms, totalizing 36 HSs.

[6] A warp is set of 32 synchronous threads that are executed at same time with the same code.

[7] Divergence means, in synchronous threads running the same code, some need to follow a different code path and the others wait that that path converges with the own *e.g.* an *if else*

with the number of antennas, but they are more demanding to compute because their number is higher. The Post-FSC was implemented in two configurations, one in which the impaired signature waveforms are generated directly in the discrete frequency domain[8] (with transcendental functions, sine and cosine) from previously ones generated offline without the impairment (in the frequency domain but through FFTs in the time domain) and the other in which the signature waveforms are generated from scratch in the time domain and then they are converted to the discrete frequency domain by the FFTs. The former is less complex but the latter permits to do the cancellation with the users signals affected with carrier plus doppler frequency offsets making it more suitable in practice. Affecting the signature waveforms with the carrier plus doppler frequency offsets must be done in the discrete time domain because in the discrete frequency domain the offset is a fraction of a sample. In the time domain the phase of (signature waveforms) samples must be affected by a linear increasing or decreasing angle.

In both implementations (OpenMP and CUDA) the code for the inversion of the matrices was taken from CLAPACK[9]. The code was inlined and cleaned of redundancy to make a single block of code embedded in the FSC function or kernel. The functions used from CLAPACK were *cpptrf* and *cpptri*. The first does the Cholesky decomposition in single precision and the latter finds the invert of the matrix having the result of the decomposition. Only the lower part of the matrices are stored. Other approaches of CUDA implementations of matrix inversions with Cholesky decomposition can be seen in [28], [29]. This solution has better performance than the native CUDA matrix inversion functions because when using them it is necessary to split the FSC in several kernels originating more accesses to the video main memory to pass the data between the kernels thereby increasing the latency.

In the CPUs there is some runtime warm-up between consecutive runs. For the serial code (with 1HS) the data and code fits in the L3 cache. In the case of the GPUs there is practically no warm-up as the data is everytime moved from CPU to GPU. There is warm-up related to transfer of the compiled kernels code in the first run and not in the subsequent runs.

IV. Complexity Results and Performance Discussion

Tables II and III show the complexity in Millions Floating Point Instructions, runtime (Wall time) for serial code for the i9 and i7 and runtime for CUDA for the RTX2070 and GTX1050Ti. The execution time of the serial implementations largely exceeds the 0.7ms (half of 1.4ms) target for 1 HS for all the different algorithms and configurations, with the execution times for the Post-FSC+PIC more or almost than double the times for the Pre-FSC+PIC.

Figure 4 shows the results of the profiling of the simultaneous CPU cores usage (CPU time which is different from Wall time) in the i7, with OpenMP, given by Intel's VTune profiler. The profile shows that the average CPU usage is about 7.5, which is a value typical for applications with an average amount of parallelism. This compares with a theoretical maximum of 12 that corresponds to the 6 cores running 2 simultaneous threads each. Figure 4 shows that during a significant part of the overall processing time less than 4 threads are executing. This is due to the fact that 16 threads are launched with the processor supporting a maximum of 12 threads running concurrently. However execution times feature a large variance, incompatible with the real-time requirements of the application.

When the GPUs are used the speedup achieved over the correspondent serial implementation figures also in Tables II and III both for 36 HSs and for a single one. When processing 1 HS at a time, the RTX 2070 and the GTX1050Ti do not achieve real time for the detectors that are implementation aware (Time Domain Sig. Wave. generation). Comparing the speedups achieved by the GPUs for 1HS they are of the same order of magnitude in both GPUs. Despite the RTX2070 be a much powerful GPU, the resources required to process 1HS is a small set of those resources and because of that the GTX1050Ti rivalizes with it in processing power. The full potential of the RTX2070 is revealed with the processing of simultaneous 36 HS. The joint processing of 36 HS roughly doubles the speedup in the case of the GTX1050Ti, while the speedup increase for the RTX 2070 is much higher, reaching 104.2 times for the Post-FSC+PIC with Time Domain Signatures Generation and 2 antennas.

In CUDA, the runtime includes the data transfer time between the CPU and GPU and back needed in a real-time implementation as well as the small CPU runtime. The arrays from the FSC kernel are stored in the GPU main memory. The amount of available cache memory (principally L2) correspondent to this main memory is important because the FSC kernel has many no coalescent accesses. The shortest execution times were achieved with a single thread program invoking the kernels. In all implementations, each time the FSC is called, 32 x 16 x nHS matrix inversions are made (nHS - number of HalfSlots), for 1.28MChips/s, for 16 users and 1 antenna. This number of matrix inversions is multiplied by the number of antennas for the Pre-FSC+PIC configuration whereas it remains constant in the configuration Post-FSC+PIC.

In CUDA with 36 HSs processing, the Post-FSC+PIC with Time Domain Signatures Generation, despite being more complex than correspondent Pre-FSC+PIC for 2 antennas, has better runtimes because it has only half of the threads (and half of the matrix inversions) launched by the FSC kernel and so less L2 cache constraints. That does not happen for 36HSs, 1 antenna where the Pre-FSC+PIC has better runtimes.

In the case of Post-FSC+PIC with Time Domain Signatures Generation (for any number of antennas) more than half of the operations are from FFTs and they are done with a much optimized library *cuFFT* from Nvidia. As a consequence, one of the highest speedups relative to the serial implementation and highest performance in GFlops

[8]The FSC needs the signature waveforms in the frequency domain
[9]http://www.netlib.org/clapack/

		MFPI	i9 (Serial)		RTX2070 CUDA	
1 ant, 1HS	Pre+PIC, Time Domain Sig Gen	52.7	7.4ms	1x	1.22ms	6.1x
	Pre+PIC, Transcendentals	47.31	6.6ms	1x	1.2ms	5.5x
	Post+PIC, Time Domain Sig Gen	138.1	23.2ms	1x	1.3ms	17.8x
	Post+PIC, Transcendentals	58.1	11.9ms	1x	0.69ms	17.2x
1 ant, 36HS	Pre+PIC, Time Domain Sig Gen	1897.2	0.2664s	1x	5.1ms	52.2x
	Pre+PIC,Transcendentals	1703.2	0.2376s	1x	4.9ms	48.5x
	Post+PIC, Time Domain Sig Gen	4971.6	0.8352s	1x	7.8ms	107.1x
	Post+PIC,Transcendentals	2091.6	0.4284s	1x	5.4ms	79.3x
2 ant, 1HS	Pre+PIC, Time Domain Sig Gen	93.38	12.7ms	1x	1.24ms	10.2x
	Pre+PIC, Trancendentals	82.61	11.0ms	1x	1.23ms	8.9x
	Post+PIC, Time Domain Sig Gen	144.0	24.6ms	1x	1.4ms	17.6x
	Post+PIC,Trancendentals	73.1	20.0ms	1x	0.72ms	27.8x
2 ant, 36HS	Pre+PIC, Time Domain Sig Gen	3361.7	0.4572s	1x	9.4ms	48.6x
	Pre+PIC, Transcendentals	2964.0	0.396s	1x	10.6ms	37.4x
	Post+PIC, Time Domain Sig Gen	5184	0.8856s	1x	8.5ms	104.2x
	Post+PIC, Transcendentals	2631.6	0.720s	1x	6.0ms	120.0x

Table II

PERFORMANCE DATA. RTX2070. UPSAMPLE EQUAL TO 8 AND THE NUMBER OF TAPS EQUAL TO 2. IT IS ASSUMED THAT THE COMPLEXITY FOR 36HS IS 36X OF THE 1HS. IT IS ASSUMED ALSO THAT THE TIME FOR SERIAL FOR 36HS IS 36X OF THE 1HS. HS - HALF SLOTS, MFPI - MILLION FLOATING POINT INSTRUCTIONS. . THE I9 IS A 8 CORE HYPERTHREADING CPU A MAXIMUM/TURBO 5GHZ CLOCK AND A 16MBYTES L3 CACHE

		MFPI	i7 (Serial)		GTX1050Ti CUDA	
1 ant, 1HS	Pre+PIC, Time Domain Sig Gen	52.7	9.55ms	1x	1.12ms	8.5x
	Pre+PIC, Transcendentals	47.31	8.4ms	1x	1.11ms	7.6x
	Post+PIC, Time Domain Sig Gen	138.1	28.9ms	1x	1.67ms	17.3x
	Post+PIC, Transcendentals	58.1	15.1ms	1x	1.35ms	11.2x
1 ant, 36HS	Pre+PIC, Time Domain Sig Gen	1897.2	0.3438s	1x	20.4ms	16.9x
	Pre+PIC,Transcendentals	1703.2	0.3024s	1x	19.9ms	15.2x
	Post+PIC, Time Domain Sig Gen	4971.6	1.0404s	1x	30.6ms	34.0x
	Post+PIC,Transcendentals	2091.6	0.5436s	1x	21.6ms	25.2x
2 ant, 1HS	Pre+PIC, Time Domain Sig Gen	93.38	16.3ms	1x	1.69ms	9.6x
	Pre+PIC, Trancendentals	82.61	14.1ms	1x	1.66ms	8.5x
	Post+PIC, Time Domain Sig Gen	144.0	30.4ms	1x	1.75ms	17.4x
	Post+PIC,Trancendentals	73.1	25.0ms	1x	1.43ms	17.5x
2 ant, 36HS	Pre+PIC, Time Domain Sig Gen	3361.7	0.5868s	1x	38.7ms	15.2x
	Pre+PIC, Transcendentals	2964.0	0.5076s	1x	37.5ms	13.5x
	Post+PIC, Time Domain Sig Gen	5184	1.0944s	1x	32.9ms	33.3x
	Post+PIC, Transcendentals	2631.6	0.900s	1x	24.2ms	37.2x

Table III

PERFORMANCE DATA. GTX1050TI. UPSAMPLE EQUAL TO 8 AND THE NUMBER OF TAPS EQUAL TO 2. IT IS ASSUMED THAT THE COMPLEXITY FOR 36HS IS 36X OF THE 1HS. IT IS ASSUMED ALSO THAT THE TIME FOR SERIAL FOR 36HS IS 36X OF THE 1HS. HS - HALF SLOTS, MFPI - MILLION FLOATING POINT INSTRUCTIONS. . THE I7 IS A 6 CORE HYPERTHREADING CPU WITH A MAXIMUM/TURBO 4.1GHZ CLOCK AND A 9MBYTES L3 CACHE

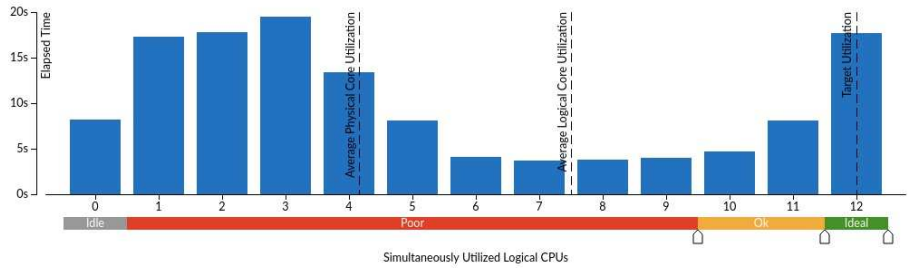

Figure 4. Bar Diagram given by Intel's Vtune profiler for Post-FSC+PIC, Time Domain Sig Gen, 2 antennas, 1HS with OpenMP on the i7 and 10000 continuous iterations. CPU with six Physical Cores and twelve Logical Cores.

Kernels	GTX1050Ti	RTX2070
FSC (Op. Ratio 66.6%)	84.7%	86.1%
FFTs Signatures(Op. Ratio 11.8%)	3.2%	3.3%
PIC Cancellation+Max-Rat-Comb (Op. Ratio 3.2%)	3.8%	2.4%
PIC Signals Reconstruction from Pre-FSC bits(Op. Ratio 2.2%)	2.5%	2.1%

Table IV

PERCENTAGE OF THE RUNTIME OF THE SEVERAL KERNELS IN PRE-FSC+PIC (WITH TIME DOMAIN SIGNATURES GENERATION), 2 ANT, 2 TAPS, UPS=8

Kernels	GTX1050Ti	RTX2070
FSC (Op. Ratio 75.1%)	86.1%	89.3%
PIC Cancellation+Max-Rat-Comb (Op. Ratio 3.6%)	3.9%	2.3%
PIC Signals Reconstruction from Pre-FSC bits (Op. Ratio 2.5%)	2.6%	1.9%

Table V

PERCENTAGE OF THE RUNTIME OF THE SEVERAL KERNELS IN PRE-FSC+PIC (WITH TRANSCENDENTALS), 2 ANT, 2 TAPS, UPS=8

are achieved by the Post-FSC+PIC with Time Domain Signatures Generation.

Because the number of taps of the transmission channel concatenated with Combining, that is seen by the FSC block in the Post-FSC is larger (compared with the Pre-FSC that only sees the taps of the channel) and the number of transcendental operations (in Post-FSC+PIC with transcendentals) are proportional to that number, the Post-FSC+PIC (with transcendentals) has less performance (GFlops) with the sequential program in the i9 and i7. Those operations,

Kernels	GTX1050Ti	RTX2070
FSC+DownS+MatchedF+DownS (Op. Ratio 21.7%)	57.2%	58.9%
FFTs Signatures(Op. Ratio 61.4%)	23.0%	24.1%
PIC Cancellation+Max-Rat-Comb (Op. Ratio 2.1%)	4.9%	3.3%

Table VI
PERCENTAGE OF THE RUNTIME OF THE SEVERAL KERNELS IN POST-FSC+PIC (WITH TIME DOMAIN SIGNATURES GENERATION), 2 ANT, 2 TAPS, UPS=8

Kernels	GTX1050Ti	RTX2070
FSC+DownS+MatchedF+DownS (Op. Ratio 42.7%)	69.6%	73.4%
Signature Waveforms Generation (Op. Ratio 28.8%)	10.3%	8.2%
PIC Cancellation+Max-Rat-Comb (Op. Ratio 4.1%)	5.9%	4.0%
PIC Signals Reconstruction from Post-FSC bits (Op. Ratio 2.9%)	4.0%	3.4%

Table VII
PERCENTAGE OF THE RUNTIME OF THE SEVERAL KERNELS IN POST-FSC+PIC (WITH TRANSCENDENTALS), 2 ANT, 2 TAPS, UPS=8

beyond being slower ones, are not (auto)vectorized in Intel CPUs because the vectorization hardware does not support them. That does not happen in CUDA where the latency is hidden with the switching of *Warps*. That explains the great speedup of the Post-FSC+PIC with transcendentals, with CUDA with 36 HSs.

For the Post-FSC+PIC and Pre-FSC+PIC configurations (with two antennas, Time Domain Signatures Generation (deployment aware) and upsample of 8) the processing of the 36 HSs takes less than 25.7ms in the RTX2070, satisfying the time specifications of UMTS-TDD. In the same conditions the GTX1050Ti does not achieve the goal to process 36 HSs in less that 25.7ms making, in this situation, a embedded solution with a single GPU not viable.

The speedups achieved in the GPUs are mainly limited by the available memory bandwidth. The necessary external GPU memory for the FSC kernel to run for 36HS, 16 users, 18x18 size matrix inversions (redundant bands), is about 32.4MBytes of temporary memory (for the PostFSC) far more than the 4 MBytes of the Level 2 cache of the RTX 2070. The higher memory bandwidth available in the RTX2070 jointly with greater number of cores is the main reason of achieving higher speedups than the GTX1050Ti. On the other hand, performance being limited by the memory bandwidth and not by the amount of parallel execution engines, implies that increasing the number of HSs processed in parallel brings no performance advantage.

Tables IV, V, VI and VII give the percentage of runtime of the most time consuming kernels reported by the profiler together with the percentage of floating point operations in each kernel for the RTX2070 and the GTX1050Ti. It can be noticed in Table VI how *cuFFT* library (see the FFTs Kernel) is optimized, certainly with low level programming, by the high percentage of operations corresponding to lower percentage of the total runtime.

It was found that the RTX2070 with the Post-FSC+PIC with Signatures Generation from time domain, 2 antennas, 2 taps, 36HS, upsample of 8 in continuous run consumes about 149W (with a stable temperature of 73°C) on a maximum of 175W. In these conditions, the RTX2070 is running at 609.9GFlops. The GTX1050Ti with the same settings runs at 157.6GFlops. For achieving realtime processing, with the same settings, is needed at least 201.7GFlops.

V. CONCLUSIONS

The complexity and runtime of a deployment aware Multiuser Detector for uplink that takes into account the carrier plus doppler frequency offsets was evaluated to investigate the possibility of its deployment in UMTS-TDD Base Stations. It was shown that when the detectors are implemented on i9 or i7 platforms the execution times achieved largely exceed the timing deadlines of UMTS-TDD. On the other hand, heterogeneous CPU+GPU architectures deliver a real-time solution, in particular when the amount of parallelism of the GPU architectures is fully taken into profit by processing several UMTS-TDD Half Slots in parallel. The results presented show that execution times with the RTX2070 satisfy the time constraints for base stations with either 1 or 2 antennas for any implementation of the two detection algorithms. The GTX1050Ti is also a solution for base stations with a single antenna if the Pre-FSC+PIC detector, with Time Domain Signatures Generation, is used. A significantly increased performance can be expected with increased memory access bandwidth.

The high processing scalability of the proposed Multiuser Detector, due to the existence of many small equal size matrices to be inverted, must be highlighted making it a good candidate for deployment. That permits to increase the capacity of the system, to reduce the power emitted by the mobile stations or to use less hardware in the base station. This solution can be provided by the base stations manufacturers to the operator with several versions featuring increased upsampling rates.

The detectors were validated in a simulation chain, that gave the Bit Error versus the Energy of bit over noise spectral density (Eb/No), showing similar performances between them and showing the degradation of performance with the sampling rate in one of them. The BER curves show a detection quality similar to the best achieved by other algorithms reported in the literature of which no real-time implementation is known.

Future work in the case of commercial use, includes the implementation of these algorithms on FPGA[10] [30] to achieve better energy efficiency [31].

ACKNOWLEDGMENTS

The core of the work presented here was done while the first author was a post-doctoral researcher within IEETA/University of Aveiro with a grant from Fundação para a Ciência e Tecnologia (FCT), Portugal supported by POPH/FSE. Thanks are due to Prof Atilio Gameiro and to Instituto de Telecomunicações of Aveiro for providing the computers with the GPUs, to Prof Paulo Dias and IrisLab both from University of Aveiro, to Massive Lab from INESC TEC and to Eng Paulo Ribeiro from Vodafone Portugal.

REFERENCES

[1] J. G. Proakis and M. Salehi, *Digital Communications*. McGraw-Hill, Inc, Fifth ed., 2007.

[10] With new FPGAs, programmed in OpenCL, that have floating point units in each cell.

[2] L. Gonçalves and A. Gameiro, "Multi-Sensor Frequency Domain Multiple Access Interference Canceller for DS-CDMA Systems," *European Transactions on Telecommunications, John Wiley & Sons, Ltd*, vol. 18, pp. 263–273, April 2007. http://doi.wiley.com/10.1002/ett.1146.

[3] L. Gonçalves and A. Gameiro, "Erratum: Multi-Sensor Frequency Domain Multiple Access Interference Canceller for DS-CDMA Systems," *European Transactions on Telecommunications, John Wiley & Sons, Ltd*, vol. 19, p. 495, June 2008. http://doi.wiley.com/10.1002/ett.1297.

[4] L. Gonçalves, *Detecção Multiutilizador no Domínio da Frequência para Sistemas DS-CDMA*. Ph. D. Thesis, Departamento de Electrónica, Telecomunicações e Informática, Universidade de Aveiro, Aveiro, Portugal, 2009. http://www.luisgo.pro/docu/Phd_Luis_Goncalves.pdf http://ria.ua.pt/bitstream/10773/2225/3/2010000093.pdf.

[5] S. Verdú, "Minimum Probability of Error for Asynchronous Gaussian Multiple-Access Channels," *IEEE Transactions of Information Theory*, January 1986. http://ieeexplore.ieee.org/document/1057121/?arnumber=1057121.

[6] W. A. Gardner, "Cyclic Wiener Filtering: Theory and Method," *IEEE Transactions on Communications*, vol. 41, January 1993. http://ieeexplore.ieee.org/document/212375/?arnumber=212375.

[7] A. Klein and P. W. Baier, "Linear Unbiased Data Estimation in Mobile Radio Systems Applying CDMA," *IEEE Journal of Selected Areas in Communications*, September 1993. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=233218.

[8] A. Host-Madsen and K.-S. Cho, "MMSE/PIC multiuser detection for DS/CDMA systems with inter- and intra-cell interference," *IEEE Transactions on Communications*, vol. 47, pp. 291–299, February 1999. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=752135.

[9] M. Vollmer, M. Haardt, and J. Götze, "Comparative Study of Joint-Detection Techniques for TD-CDMA Based Mobile Radio Systems," *IEEE Journal on Selected Areas in Communications*, vol. 19, August 2001. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=942509.

[10] E.-L. Kuan and L. Hanzo, "Burst-by-Burst Adaptive Multiuser Detection CDMA: A Framework for Existing and Future Wireless Standards," *Proceedings of the IEEE*, vol. 91, pp. 278–302, February 2003. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1182063.

[11] A. Zahedi and H. Bakhshi, "Multiuser Detection Based on Adaptive LMS and Modified Genetic Algorithm in DS-CDMA Communication Systems," *Wireless Personal Communications, Springer*, vol. 73, pp. 931–947, 2013. https://doi.org/10.1007/s11277-013-1224-7.

[12] A. Zahedi, S. Rajamand, S. Jafari, and M. Rajati, "A Novel Multiuser Detector Based on Restricted Search Space and Depth-First Tree Search Method in DS/CDMA Communication Systems," *Wireless Personal Communications, Springer*, vol. 82, pp. 1531–1545, 2015. https://doi.org/10.1007/s11277-015-2297-2.

[13] M. D. McCool, "Scalable Programming Models for Massively Multicore Processors," *Proceedings of IEEE*, vol. 96, pp. 816–831, May 2008. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4490125.

[14] O. Schenk, M. Christen, and H. Burkhart, "Algorithmic performance studies on graphics processing units," *Journal of Parallel and Distributed Computing*, vol. 68, pp. 1360–1369, October 2008. https://doi.org/10.1016/j.jpdc.2008.05.008.

[15] F. Arguello, D. Heras, and M. Boo, "GPU detectors for interference cancellation in chaos-based CDMA communications," *Electronic Letters*, vol. 46, pp. 727–729, 13th May 2010. http://ieeexplore.ieee.org/document/5466370/?arnumber=5466370.

[16] X. Chen, L. Ren, Y. Wang, and H. Yang, "GPU-Accelerated Sparse LU Factorization for Circuit Simulation with Performance Modeling," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, pp. 786–795, March 2015. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=6774937.

[17] G. Jo, J. Nah, J. Lee, J. Kim, and J. Lee, "Accelerating LINPACK with MPI-OpenCL on Clusters of Multi-GPU Nodes," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, pp. 1814–1825, July 2015. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=6846313.

[18] M. Lu, Y. Liang, H. P. Huynh, Z. Ong, B. He, and R. S. M. Goh, "MrPhi: An Optimized MapReduce Framework on Intel Xeon Phi Coprocessors," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, pp. 3066–3078, November 2015. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=6939728.

[19] C. G. Kim and Y. S. Choi, "Exploiting Multi -and Many-core Parallelism for Accelerating Image Compression," in *2011 Fifth FTRA International Conference on Multimedia and Ubiquitous Engineering*, (Crete, Greece), 28-30 June 2011. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5992185.

[20] H. Su, C. Zhang, J. Chai, M. Wen, N. Wu, and J. Ren, "A High-Efficient Software Parallel CAVCL Encoder Based on GPU," in *2011 34th International Conference on Telecommunications and Signal Processing (TSP)*, (Budapest, Hungary), 18-20 Aug 2011. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6043672.

[21] S. F. Swanchara, *An FPGA–Based Multiuser Receiver Employing Parallel Interference Cancellation*. Master of Science in Electrical Engineering, Faculty of the Virginia Polytechnic Institute and State University, Blacksburg, Virginia, July 22, 1998.

[22] A. O. Dahmane and L. Mejri, "FPGA Implementation of Block Parallel DF-MPIC Detectors for DS-CDMA Systems in Frequency-Nonselective Channels," *Journal of Electrical and Computer Engineering, Hindawi Limited*, 2008. http://dx.doi.org/10.1155/2008/435756.

[23] Z. Quan, J. Liu, and Y. Zakharov, "FPGA Implementation of DCD Based CDMA Multiuser Detector," in *2007 15th International Conference on Digital Signal Processing*, (Cardiff, UK), 1-4 July 2007. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4288583.

[24] R. S. Perdana, B. Sitohang, and A. B. Suksmono, "A survey of graphics processing unit (GPU) utilization for radar signal and data processing system," in *2017 6th International Conference on Electrical Engineering and Informatics (ICEEI)*, pp. 1–6, Nov 2017. http://ieeexplore.ieee.org/document/8312430/?arnumber=8312430.

[25] L. C. Gonçalves, R. E. Martins, and A. B. Ferrari, "Software Parallel Implementation of a DS-CDMA Multiuser Detector," in *The 23rd International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2015)*, (Split-Bol (Island of Brac), Croatia), 16-18 September 2015. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=7314124.

[26] J. C. Liberti and T. S. Rappaport, "A Geometrically based Model for Line-of-Sight Multipath Radio Channels," in *IEEE 46^{th} Vehicular Technology Conference (VTC'96)*, (Atlanta, USA), 28 April - 1 May 1996. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=501430.

[27] J. C. Liberti and T. S. Rappaport, *Smart Antennas for Wireless Communications: IS-95 and Third Generation CDMA Applications*. Prentice Hall, 1999.

[28] J. Kurzak, H. Anzt, M. Gates, and J. Dongarra, "Implementation and Tuning of Batched Cholesky Factorization and Solve for NVIDIA GPUs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, pp. 2036 – 2048, July 2016. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=7275187.

[29] A. Haidar, A. Abdelfattah, M. Zounon, S. Tomov, and J. Dongarra, "A Guide for Achieving High Performance with Very Small Matrices on GPU: A Case Study of Batched LU and Cholesky Factorizations," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, pp. 973–984, May 2018. http://ieeexplore.ieee.org/document/8214236/?arnumber=8214236.

[30] A. J. Maier and B. F. Cockburn, "Optimization of Low-Density Parity Check decoder performance for OpenCL designs synthesized to FPGAs," *Journal of Parallel and Distributed Computing*, vol. 107, pp. 134–145, September 2017. http://dx.doi.org/10.1016/j.jpdc.2017.04.001.

[31] K. Nagasu, K. Sano, F. Kono, and N. Nakasato, "FPGA-based tsunami simulation: Performance comparison with GPUs, and roofline model for scalability analysis," *Journal of Parallel and Distributed Computing*, vol. 106, pp. 153–169, August 2017. https://doi.org/10.1016/j.jpdc.2016.12.015.