DS-CDMA Implementation with Iterative Multiple Access Interference Cancellation

Russell Dodd, Student Member, IEEE, Christian Schlegel, Fellow, IEEE, Vincent Gaudet, Senior Member, IEEE

Abstract—In this paper an implementation of iterative joint detection for multiple access interference using direct-sequence code-division multiple-access (DS-CDMA) is presented. Results for multiple field programmable gate array (FPGA) platforms and multiple technology nodes for synthesized application specific integrated circuits (ASIC) are presented. The joint detection is performed using a generalized version of interleave-division multiple-access (IDMA) known as partition spreading (PS) CDMA. Decoding is performed using iterative methods from turbo and sum-product decoding. The synthesized ASIC system demonstrates a maximum aggregate throughput of 197Mb/s for a fully loaded 50-user system, while the implemented FPGA 50-user system has a maximum aggregate throughput of 119Mb/s.

I. INTRODUCTION

One approach to creating flexible and spectrally efficient multiuser wireless communication systems is to use direct-sequence code-division multiple access (DS-CDMA) technology. Instead of dividing the available spectrum into time or frequency slots, the users are separated by unique pseudo-random signature sequences that spread their signal across a frequency range, such that each user’s signal appears as noise to all other users. The signals are decoded using the same unique pseudo-random sequences at the receiver in a correlation front-end.

The resulting detectors can create sufficient statistics of the transmitted signals by using minimum mean squared error (MMSE) based filters [1] or matched filters [2]. The second generation of DS-CDMA systems uses matched filters, implemented as rake receivers, but they demonstrate a capacity limit since they ignore the presence of multiple access interference (MAI). Another constraint in conventional DS-CDMA systems, linked to interference, is that in the uplink the requirement for power control is severe due to the near-far effect [3].

We are interested in systems that can increase the number of users beyond the limits of conventional DS-CDMA, yet still have feasible hardware realizations. The optimal MAI detector addresses the contributions of MAI, but is impractical for implementation since its complexity increases exponentially with the number of users [4]. Implementations of low-density-parity-check (LDPC) decoders [5] and turbo decoders [6] use soft information processing and their techniques can be applied in other detectors that operate on log likelihood values or soft information. There are similarities to suboptimal detectors proposed that demonstrate performance (spectral efficiency) better than the conventional detector, but with lower computational complexity than the optimal detector. These detectors come in many flavors. Some are: parallel interference cancellation (PIC) [7], [8], multistage PIC [9], successive interference cancellation (SIC) [10], expectation maximization [11] and space alternating generalized expectation (SAGE) [12], [13]. Another suboptimal detector, partition spread (PS)-CDMA, previously introduced in a theoretical context, is considered in this paper.

PS-CDMA can be viewed as a generalized approach to interleave division multiple access (IDMA) [14] and draws upon elements of PIC as well as LDPC sum-product and turbo-decoding [15], [16]. It is an attractive technology for implementation due to its simple construction and iterative nature. Additionally, the PS-CDMA system does not require the strict power controls of conventional DS-CDMA [16]. Further, it can be shown that PS-CDMA with iterative decoding can approach both the performance of optimal detection of CDMA, [17], and also the capacity of the CDMA channel if used together with spatial coupling [18].

Iterative algorithms, like PS-CDMA, present tradeoffs to be explored regarding data dependencies, complexity and throughput. Implementations of iterative processors allow the examination of specific applications and give the opportunity to tradeoff design size and cost for performance.

In PS-CDMA we take the view that a direct-sequence spread spectrum (DSSS) signal can be viewed as a repetition code with a rate inverse to the processing gain1. Furthermore, unlike most other iterative joint decoding algorithms [13], [22], with PS signaling the detection process naturally breaks into a “demodulation” process, and a “decoding” process. The demodulation process is iterative in nature and involves only the repetition code and serves to remove mutual interference, while the decoding process applies standard off-the-shelf error control codes which are applied externally to largely interference-free individual signal streams. In this paper we are only concerned with the iterative demodulation part, which not only comprises the novel process and which forms the iterative part of an iterative multiuser receiver, but is also the processing intensive part of the receiver.

By partitioning the original DSSS signal into M sections—the partitions—we obtain a system that can naturally be

1Other redundant error control codes can be substituted for the repetition code; however, as shown in [15], [16], repetition codes simultaneously achieve the best spectral efficiency due to the dynamics of the iterative demodulation process, and the lowest implementation complexity.
interpreted as the concatenation of a repetition code with a \((M\)-fold shortened) DS-CDMA pseudo-random spreading code \(s_{n,k}\). As novelty, we also add an interleaver between different partitions, as shown in Figure 1. This spreading of the partitions of a given symbol will create a correlated system with a sparse graphical representation, which is amenable to an iterative demodulation algorithm as shown below. Each repetition of a data symbol \(d_{q,k}\) is known as a partition and \(M\) partitions use up the same resources as a single spread symbol in conventional CDMA [19], [20]. Partitioning results in \(N/M\) chips/partition, and therefore the overall processing gain is preserved at \(M \cdot N/M = N\).

In a PS-CDMA detector \(K\) matched filters operate on the received chips to generate soft partition estimates for each user. These estimates are used in a similar fashion to sum-product decoding to attain extrinsic estimates, which are iteratively calculated to cancel out the multiple access interference. Cancellation is performed at the chip or signal scale, while interleaving and soft estimation are performed at the partition scale. As it turns out, involving only the repetition code in the cancelation of the mutual interference is sufficient to achieve optimal performance (see [16], [18]).

In [21] we presented preliminary field programmable gate array (FPGA) results for a 50-user PS-CDMA detector within a Virtex-IV device and in this paper we extend the results to application specific integrated circuits (ASIC) synthesized results from the 130nm and 90nm technology nodes. This paper is organized as follows: in Section-II, we present the DS-CDMA system model along with the PS-based elements within an additive white Gaussian noise (AWGN) channel. In Section-III, PS-CDMA proposed hardware architecture is shown. Synthesis results for FPGA, for systems with up to 50-users, and synthesized circuits in 130nm and 90nm technologies are shown in Section-IV. A comparison to other DS-CDMA systems using DSP and FPGA implementations is presented and in addition, comparisons to theory with respect to quantization effects are made. The conclusion is in Section-V.

II. SYSTEM MODEL

A. Transmitter

We consider a DS-CDMA system model using binary phase shift keying (BPSK) modulation. Figure 1 shows the transmitter block diagram for each user. At the transmitter encoded bits \(d_{q,k}\) of user \(k\), \(k \in \{1, \ldots, K\}\), \(q, q \in \{1, \ldots, L\}\) (\(d\) could be the result of high-rate LDPC encoding as this detector performs very well in conjunction with high rate codes [15]) are encoded by a repetition code \((M, 1)\) as part of the DSSS modulation operation. Typically \(M\) is chosen between 3–5 as larger values have limited additional effect on performance – see (14) below. The resulting coded bits \(b_{j,k}, j \in \{1, \ldots, N_m\}\), \(N_m = LM\), are interleaved by a user-distinct interleaver \(\Pi_k\) to obtain \(b'_{j,k}\). Since the system uses DS-CDMA, the interleaved coded bits are spread by a user-distinct spreading code \(s_{j,k}\). These spreading codes have a reduced length, or processing gain of \(N/M\), with respect to the baseline CDMA system which would use spreading sequences of length \(N\). This normalization is used to ensure fair power and spectral comparisons. Figure 2 shows a small example of the relationship of data symbols and partitions for a single user, \(L = 4, M = 3\).

Before transmission, each user has a power \(P_k\) assigned to it and its chips are normalized to the overall (uncoded) spreading gain of \(N\), i.e., \(c_k = 1/\sqrt{N}\). This gives the resulting individual transmissions at the coded, or partitioned, level as

\[
y_{j,k} = \sqrt{P_k} c_k b'_{j,k} s_{j,k}, \quad j \in \{1, \ldots, LN/M\}
\]

(1)

where \(s_{j,k} = [s_{j,k,1}, \ldots, s_{j,k,N/M}]\) is the reduced-length spreading sequence for user \(k\) and partition \(j\).

The system load \(\alpha = K/N\) is a key parameter for comparing performance of different systems. We show that for a given \(\alpha\) the absolute size of the system does not affect the bit error rate (BER) performance of the interference cancellation appreciably. The improvements of PS-CDMA over conventional CDMA can be seen with high system loads over and near \(\alpha = 1\). In fact, the traditional use of a correlation receiver achieves spectral efficiencies of only around \(\alpha = 0.1 – 0.2\), while the more advanced minimum mean-square error (MMSE) receiver can achieve values of \(\alpha \approx 0.5\) without significant degradation [20]. The iterative receiver examined here shows nearly no degradation at values exceeding \(\alpha = 1\).

B. Detector

All the users’ signals \(y_{j,k}\) aggregate in the AWGN channel, to give the received signal shown in (2), where \(n\) is the contribution of the additive white noise.

\[
r_j = \sum_{k=1}^{K} y_{j,k} + n
\]

(2)

Synchronicity is assumed here for simplicity, however, simulation results and the analysis in [16] show that this assumption...
is not critical for either performance or implementation complexity of the decoder itself.

The detector consists of $K$ parallel individual data receivers, shown in Figure 3 with the single user’s processor outlined with a dashed box. The receivers decode their respective transmission embedded in $r$ and then in parallel share their estimate of $y_{j,k}$ with each other to attain an interference reduced version of $r$. This means that all receivers share a common input $r$ and an estimated channel aggregator at the chip level $\sum y_{j,k,n}$. Here the estimate of each received signal $\hat{y}_{j,k,n}$ at the chip level is summed to approach an interference free estimate of the received signal at the $i$-th iteration of processing. The first iteration has no estimates available and skips the cancellation stage.

To attain an estimate of the user’s transmitted partitions we use a conventional matched filter to correlate the unique signature sequences $s_{j,k}$ with the incoming noisy waveform to give sufficient statistics of the user’s signal. We have assumed synchronicity within the system for this implementation, but this is not an essential prerequisite. Each received partition element after matched filtering with its individual spreading sequence is given as

$$\hat{\tilde{b}}_{j,k}^{i} = \sum_{n=1}^{N/M} \left( r_n - \tilde{y}_{k,n}^i \right) s_{j,k,n}$$

$$= \sum_{n=1}^{N/M} \left( \sum_{k=1}^{K} y_{j,k,n} + n - \sum_{k' \neq k}^{K} \tilde{y}_{n,k'}^{i-1} \right) s_{j,k,n}$$

$$= \sum_{n=1}^{N/M} \left( y_{j,k,n} + I_{j,k,n}^{i-1} + n \right) s_{j,k,n}$$

Equation 3 contains the contributing elements to the partition estimates as the detector iterates. With adequate signal-to-noise ratio (SNR) the chip samples $\tilde{y}_{j,k,n}$ approaches the value of $y_{j,k,n}$, meaning the interference $I_{j,k,n}^{i-1}$ goes to zero with iterations.

The log-likelihood ratio (LLR) of $d_{k}$ at iteration $i$ is computed as

$$\log \left( \frac{\Pr (d_{k} = 1)}{\Pr (d_{k} = 0)} \right) = \frac{2}{\sigma^2} \sum_{j=1}^{M} \hat{b}_{j,k}$$

where $\sigma^2$ is the variance of the interference and noise, which is shown in [22] to be Gaussian. Additionally to (4), log-likelihood ratios for each partitioned bit $\hat{b}_{j,k}$ can be computed individually as the sum of the LLRs of all related partitions, i.e., as

$$\log \left( \frac{\Pr (b_{j,k} = 1)}{\Pr (b_{j,k} = 0)} \right) = \frac{2}{\sigma^2} \sum_{m=1 \atop m \neq j}^{M} \hat{b}_{m,k}$$

From these LLR values, soft estimates of each partitioned bit can be computed as the expectation of (5), i.e., as

$$\tilde{b}_{j,k} = \tanh \left( \frac{1}{\sigma^2} \sum_{m=1 \atop m \neq j}^{M} \hat{b}_{m,k} \right)$$

The argument in (6) is akin to the LLR addition that occurs in the iterative decoder of a low-density parity-check (LDPC) code, which aggregates these LLRs at its variable nodes. These variable nodes, as in this system, represent repetition codes.

In order to perform (6) the partitions are deinterleaved so that the summation occurs on partitions with the same origin. Using the user-distinct interleaver we can deinterleave the estimated partitions $\hat{b}_{j,k}^{i}$ into their original natural order $\tilde{b}_{j,k}^{i}$.

If we view each partition as a separate estimate of the original data symbol $d_{q,k}$ we can sum the $M$ partitions and take the sign of the sum as the hard decision of the detector output. This sum can be reused in the extrinsic summation of Equation 6 by adding all the partitions of a symbol and then subtracting the self-information to obtain the extrinsic estimate. This is analogous to the sum node processing of an LDPC decoder, but instead of parallel wires entering the summation we have serial partitions.

We split (6) into two operations, the extrinsic estimate

$$b_{\text{ext},j,k}^{i} = \sum_{m=1 \atop m \neq j}^{M} \hat{b}_{m,k}^{i}$$

and the variance-scaled tanh soft-bit operator:

$$\tilde{b}_{j,k}^{i} = \tanh \left( \frac{1}{\sigma^2} b_{\text{ext},j,k}^{i} \right)$$

Performing Equations 7 and 8 sequentially gives the same result as Equation 6. In (8), ideally, the variance is calculated from a training sequence, but we show in Section IV that an estimate of the variance performs well enough.

In order to perform the partial interference cancellation of chips, we must recreate them identically to the user’s respective transmitted signals. Using, the user-distinct interleaver and coding sequence an estimate of the interference signal for user $k$ is generated as

$$\tilde{r}_{k}^{i} = \sum_{k' = 1 \atop k' \neq k}^{K} \sqrt{P_{k} c_{k} \hat{b}_{j,k}^{i} s_{j,k}}$$

From these LLR values, soft estimates of each partitioned bit can be computed as the expectation of (5), i.e., as

$$\tilde{b}_{j,k}^{i} = \tanh \left( \frac{1}{\sigma^2} \sum_{m=1 \atop m \neq j}^{M} \hat{b}_{m,k}^{i} \right)$$

The argument in (6) is akin to the LLR addition that occurs in the iterative decoder of a low-density parity-check (LDPC) code, which aggregates these LLRs at its variable nodes. These variable nodes, as in this system, represent repetition codes.

In order to perform (6) the partitions are deinterleaved so that the summation occurs on partitions with the same origin. Using the user-distinct interleaver we can deinterleave the estimated partitions $\hat{b}_{j,k}^{i}$ into their original natural order $\tilde{b}_{j,k}^{i}$.

If we view each partition as a separate estimate of the original data symbol $d_{q,k}$ we can sum the $M$ partitions and take the sign of the sum as the hard decision of the detector output. This sum can be reused in the extrinsic summation of Equation 6 by adding all the partitions of a symbol and then subtracting the self-information to obtain the extrinsic estimate. This is analogous to the sum node processing of an LDPC decoder, but instead of parallel wires entering the summation we have serial partitions.

We split (6) into two operations, the extrinsic estimate

$$b_{\text{ext},j,k}^{i} = \sum_{m=1 \atop m \neq j}^{M} \hat{b}_{m,k}^{i}$$

and the variance-scaled tanh soft-bit operator:

$$\tilde{b}_{j,k}^{i} = \tanh \left( \frac{1}{\sigma^2} b_{\text{ext},j,k}^{i} \right)$$

Performing Equations 7 and 8 sequentially gives the same result as Equation 6. In (8), ideally, the variance is calculated from a training sequence, but we show in Section IV that an estimate of the variance performs well enough.

In order to perform the partial interference cancellation of chips, we must recreate them identically to the user’s respective transmitted signals. Using, the user-distinct interleaver and coding sequence an estimate of the interference signal for user $k$ is generated as

$$\tilde{r}_{k}^{i} = \sum_{k' = 1 \atop k' \neq k}^{K} \sqrt{P_{k} c_{k} \hat{b}_{j,k}^{i} s_{j,k}}$$
and used in (3) for the next iteration.

All $K$ users’ chip estimates are aggregated and subtracted from the original received aggregate signal $r_n$. If the estimates are perfect, the only values left are the contribution of noise to that chip. However, the interference cancellation has removed the user’s own information, so that is added back in, which each user does within its respective receiver to attain a user specific version of the interference-reduced received signal $r_n - \hat{r}_{k,n}$. The system then iterates until MAI has been cancelled sufficiently to attain acceptable BER. Further details on the dynamics and performance of this specialized correlation-cancellation process can be found in [15]–[17].

The effect of processing CDMA signals in this iterative fashion via partitioning the original DSSS signal into a few components is a dramatic increase in the achievable spectral efficiency with respect to conventional CDMA, as well as with respect to the at least equally complex MMSE receiver. These performance results are discussed in Section IV.

III. SYSTEM ARCHITECTURE

In this section the design choices for the FPGA and ASIC implementation are presented. Designs are implemented in VHDL and C++. The C++ program is used to plan design parameters for BER performance and perform bit-true calculations. In addition, the program is used to perform quantization analysis and its effect on BER. For the FPGA study Xilinx ISE 9.2 is used for synthesis compilation and Synopsys D-2010 and Encounter EDI 9.1 are used for ASIC synthesis and layout respectively.

Figure 4 shows the system architecture. All data paths use sign-magnitude representation and control signals are omitted. The elements within the dashed box make the individual data receiver, while the contents within the shaded area make the partition processing datapath. The datapaths each have a bit width or precision of P-bits or H-bits. All circuits within the shaded area use P-bits, otherwise circuits have a precision of H-bits. Hard decisions for BER calculation are generated with the partition processing datapath.

The transmitted signal is stored in memory, using $L \times N$ entries each with H-bits. This memory is shared among all receivers as the data is streamed to all $K$ receivers simultaneously. The H-bit adder, after the memory, is for interference cancellation and in the first iteration nothing is added (or subtracted).

A. Matched Filter

Chip arrival between all users is assumed to be synchronous, so the matched filter consists of an XOR gate, an accumulator (with reset), and a scaler. The scaler multiplies the partitions by $\sqrt{M/N}$. The XOR takes advantage of the sign-magnitude form by tying the output of the linear feedback shift register (LFSR) with the most significant bit (MSB) of the data. This avoids the need for multiple bit switching that two’s complement would require. The design of the matched filter is shown in Figure 5.

The LFSR is a 51 stage delay line, with stages 1 and 4 tapped. The value of each stage is programmed to a unique sequence for each user. The length and taps were chosen to supply a suitable pseudo-random number generator [23].

In the matched filter a transition between two processing domains occurs, the chip processing domain and the partition processing domain. The chip processing domain requires a greater precision in data representation of H-bits, while the partition processing domain requires a less precise representation of P-bits. This occurs due to the soft processing that occurs in the tanh-limit, whereas the cancellation requires a precision that can handle fractions of transmitted amplitudes. After the matched filter the data takes two parallel paths. The first path forms estimates of the data and calculates hard decisions. The second path is used to form an estimate of the variance in the signal.

B. Interleaver

The interleaver is based on the work of Takeshita [24]. It has a depth of $L \times M$ elements, and instead of storing the indices, they are calculated in order to reduce usage of memory resources. The design can operate in parallel, but we chose a serial implementation shown in Figure 6.

The permutation polynomial being calculated is

$$ (63x + 128x^2 + h) \mod (L \times M). \quad (10) $$

The value $x$ in Equation 10 is generated by a counter and only one multiplier is needed. The value $h$ is introduced to create unique interleavers. The interleavers implemented in the FPGA rely on available block random access memory (BRAM). The sequential design introduces latency since the entire frame of $L \times M$ partitions must be written in to memory before being read out. However, at the expense of another memory block and multiplexors the interleaver can be pipelined, since the writing index and reading index are generated separately.
C. Partition Estimation

In the previous section, Figure 3 contains the partition estimate module positioned between the deinterleaver $\Pi^{-1}$ and the interleaver $\Pi$. In the implementation, shown in Figure 4, the partition estimation is split around the interleaver. The extrinsic step, Equation 7, comes first followed by the tanh step, Equation 8, after the interleaver. The tanh step of the estimation operation is separated from the extrinsic step in order to give the system a chance to develop an estimate on the variance statistic used to weight the tanh operation.

The extrinsic step uses an accumulator and a delay element (flip-flops). A partition is delayed by $M$ cycles so that the $M$ partitions of a symbol can be accumulated. Then the partition is subtracted from the sum, to attain the extrinsic partition. This occurs for all $M$ partitions in the delay pipeline. By looking at the MSB of the accumulator's sum we can also get the hard decision for the iteration.

For the tanh step, we use a small LUT with P-bit inputs and H-bit outputs. We have also moved the scaling from the cancellation into the LUT to avoid an extra multiplier.

D. Variance Calculation

The variance is calculated assuming that the partitions of the matched filter give the correct hard decision, and is called the sliced variance $\hat{\sigma}^2$ [25].

$$\frac{1}{\hat{\sigma}^2} = \frac{2}{N_0} = \frac{LN}{\sum_1^{LN} \left( \hat{r}_k - \frac{1}{\sqrt{M}} \right)^2} \quad (11)$$

The sliced variance operation is performed by a series of shifts with a multiplier and an accumulator followed by a look-up-table (LUT). The LUT performs the $\frac{1}{2}$ function to weight the extrinsic partitions as given in Equation 6.

E. Cancellation

The final stage of the receiver creates an estimate of the transmitted signal. The P-bit result from the decision operation is sign-extended to an H-bit signal for the cancellation operation. Before cancellation, the chips are first scaled by $\frac{1}{\sqrt{N}}$ (performed in the tanh LUT) and then scrambled with the appropriate spreading sequence in order to align them with the transmitted values.

A large combinational component of the receiver is the common aggregator that sums all K-users estimates in order to cancel out interference. In this implementation a full-adder tree is used, which gives a $\log_2(K)$ delay in each individual receiver to keep estimates aligned. The delay is implemented with a chain of $\log_2(K)$ registers. The cycle-time cost of the adder tree could be improved by using specially design multioperand adders [26] or using carry-save addition.

F. Hardware

The FPGA test bed is comprised of a personal computer communicating to a FPGA through ethernet and a DSP daughterboard. The synthesis results are based on outputs from Synopsys (D-2010) and Encounter (EDI 9.1).

On two different Lyrtech development boards (C67X and QC6416) the Virtex FPGA (XC2V8000 and XC4VLX160) is completely devoted to the implementation of the PS-CDMA detector, while the Texas Instruments digital signal processor (DSP) (TMSC32C7 and TMS320C6416) serves as a convenient gateway to the input and output pins of the FPGA through an ethernet connection to the PC. A small amount of routing and processing on the FPGA is used to provide hard decisions to the DSP and computer for calculation of BER. The $K$ transmitters and AWGN channel are implemented in software in the PC and their resulting cumulative transmission is sent over the ethernet connection to the DSP and subsequently to the FPGA.
The testbed and all simulation results are based on baseband processing. The chips are never scaled to radio frequencies. The synchronous AWGN channel is defined by \( \sigma = N(0,1) \) [23] with \( \sigma \) implemented as shown in Equation 12.

\[
\sigma = \sqrt{\frac{P_k}{R \cdot 10^{-\frac{SNR}{10}}}} \tag{12}
\]

where \( P_k \) is the power attributed to user \( k \), \( R \) is the rate of the code in \( d \) (\( R = 1 \) for BPSK) and SNR has units of dB.

IV. IMPLEMENTATION RESULTS

The implementation results for the FPGA testbed and ASIC synthesis are summarized in this section. Logical equivalence between C++ simulations, VHDL simulations and the FPGA implementation was confirmed using ModelSim and a custom program that interfaces with a test circuit to calculate errors seen in the detector.

Figure 7 shows the system results. The implementation was found to have acceptable performance using \( P=8 \) and \( H=11 \) on their respective datapaths. Although not implemented, higher loaded systems can be operated using an exponential power distribution among the users as shown in Equation 13, [16], with

\[
P[k] = e^{ak/N}; \quad a = 0.5 \tag{13}
\]

The systems under test use a partition gain of \( M = 4 \) with \( L = 512 \) symbols transmitted per packet. Each user has the same power, which is the worst-case scenario for this particular receiver. The overall system size does not affect the BER performance appreciably as long as the system load remains constant. However, as system load is increased, a higher SNR is required to achieve the same BER. For more details see [15]–[17]. In [27] it is shown that the error rate for partitioned signaling with large blocks, i.e., \( L \) goes to infinity, achieves an error rate given by

\[
P_0 = Q(\gamma_{\infty})
\]

\[
\gamma_{\infty}^2 = \frac{K}{N} \left[ 1 - \tanh \left( \frac{M-1}{M} \gamma_{\infty}^2 + \sqrt{\frac{M-1}{M} \gamma_{\infty}^2} \right) \right] + \sigma^2 \right)^{-1} \tag{14}
\]

Equation 14 is plotted in Figure 7 along with our experimental results. The implementation results are based on only 50 frames and a finite block length of \( L = 512 \), which accounts for the minor degradation of the performance with respect to the case of infinite block size. The decoders were allowed to iterate for many iterations (<100) in order to determine best BER performance possible.

A. An FPGA Implementation

In Table I the FPGA synthesis test results are given for two different implementations on FPGAs. The test results for a 37-user system on a Virtex-II XCV28000 and a 50-user system on a Virtex-IV XC4VXL160 are described. Both FPGAs reside on Lyrtech development boards. Slice flip-flops are not highly utilized, since most slices contain logic that is not pipelined. The maximum aggregate throughput is calculated by Equation 15 with \( L = 512 \), \( N = 32 \), \( I = 1 \), and \( M = 4 \).

Real time tests are performed with a DSP generated 80MHz clock frequency and new throughputs of 43.5Mb/s and 58.8Mb/s respectively can be calculated as shown in Equation 15 by setting \( F = 80MHz \). \( F \) is the system frequency and \( I \) is the number of iterations. The highest possible throughput is attained when \( I = 1 \), but if we consider various \( I \), for example \( I = 5 \), the (useable) throughput for the 50-user system becomes 18.8Mb/s. In addition, the system latency is determined by the number of iterations needed to reach the target BER. If we assume \( BER = 1e^{-4} \), low system loads typically have \( I < 5 \) and for high system loads \( I < 20 \). Latency can be calculated by the denominator in Equation 15 in units of cycles. Therefore a 50-user FPGA implementation requiring 5 iterations would have a latency of 108,544 cycles, which at 80MHz is 1.3ms.

\[
T_{put} = \frac{K \cdot L}{L \cdot N + \sum_{i=0}^{i>0} \frac{L \cdot M + L \cdot N}{L \cdot N + \sum_{i=0}^{i>0}}} \cdot F \tag{15}
\]

The detector prepared for the FPGA has specific parameters that limit the number of users implementable. Each user requires two large memory blocks for serial interleaving. Each block contains the number of repeated partitions at the implemented bit-precision, \( LNP \)-bits, so for the results shown memory required per user ranges from 16,384-bits to 22,528-bits. The FPGA block RAM (BRAM) can be setup for several different data widths (from available primitives), but if the input data width is less than the BRAM width then BRAM bits are left unused. The implementation was chosen to handle various data precisions up to 16-bits. Therefore, when \( P = 8 \) or 11, 8 to 5 bits are left unused respectively. In the case of \( P = 8 \) the memory blocks are 50% empty and present a significant under utilization of the BRAM. The logic breakdown per user is shown in Table II. A logic breakdown per module is shown in Table III.

Table II shows the memory requirements for a single user. 1 BRAM per interleaver, and another for testing the input sequence to generate BER curves (testing module). On a per-user basis, available memory is utilized at comparable rates to slices for operations. In Table III the control module is a state-machine and contributes significant overhead to the
Fig. 7. Various PS-CDMA systems with different loads. The systems use $M = 4$ partitions, $L = 512$ symbols. BER for two different precisions are for the partition processing $H=11$ and $P=8$ or $11$ over 10 frames.

### TABLE II

**SINGLE USER LOGIC UTILIZATION ON A VIRTEX-IV**

<table>
<thead>
<tr>
<th>FPGA Element</th>
<th>Number Occupied</th>
<th>Virtex-IV Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slice Flip Flops</td>
<td>691</td>
<td>0.511 %</td>
</tr>
<tr>
<td>4-input LUT</td>
<td>1,259</td>
<td>0.931 %</td>
</tr>
<tr>
<td>Occupied Slices</td>
<td>894</td>
<td>1.322 %</td>
</tr>
<tr>
<td>BRAMs</td>
<td>3</td>
<td>1.041 %</td>
</tr>
</tbody>
</table>

### TABLE III

**SINGLE USER MODULE LOGIC UTILIZATION CONTRIBUTION**

<table>
<thead>
<tr>
<th>PSCDMA Module</th>
<th>Slices</th>
<th>4-input LUTs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control</td>
<td>18 %</td>
<td>25 %</td>
</tr>
<tr>
<td>Testing</td>
<td>26 %</td>
<td>24 %</td>
</tr>
<tr>
<td>Variance Estimation</td>
<td>28 %</td>
<td>26 %</td>
</tr>
<tr>
<td>Processing</td>
<td>28 %</td>
<td>25 %</td>
</tr>
</tbody>
</table>

The FPGA results show that significant logic reductions can still be made within the control logic and cancellation operations. This indicates that future implementations may attain more users and therefore a higher aggregate throughput. Another limitation to the design is the serial nature of the interleavers and their impact on memory usage and throughput. Each interleaver and de-interleaver adds $LM$ cycles to each iteration, which lowers the throughput at a given number of iterations for successful cancellation given by Equation 15.

### B. An ASIC Implementation Study

The ASIC synthesis results from Synopsys Design Compiler are presented in Table IV for a single receiver core in the 90nm CMOS TSMC digital library (tcbnghp) technology node and
130nm (cmos8rf) technology node.

Table IV contains the estimates of the ASIC device based on logic only, it contains no routing or placement overhead. The power estimates are based on the capacitances available in the technology library and do not account for extracted parasitics on routing. In the ASIC implementation the longest delay path is in the variance module on a large fanout signal. From Equation 11 the summation in the denominator occurs over the entire frame and requires a relatively large data path for the accumulator to avoid overflow. The feedback in the accumulator is on the critical path. In future implementations the entire variance block should be optimized separately for timing driven analysis, since its output is ready long before it is needed.

Instead of mapping interleaver memories to SRAM or even DRAM, the memory is mapped to registers consisting of positive edge triggered D-flip-flops. This is done to avoid the implementation of a memory controller, to keep the memory local to the user and to avoid the complicated integration of memory blocks in the design flow. This also serves as an upper-limit to the area cost function of the system. The ASIC design complexity could be reduced by changing the interleaver address generator architecture. The interleaver permutation polynomial has an explicit multiplication $x^2$ as shown in Equation 10. This module occurs twice in each receiver and can lead to significant logic overhead in the 50-user ASIC implementation. In the FPGA this multiplication could be mapped to a hardware multiplier, which can free up slices, but in a ASIC device a multiplication is a computationally expensive operation and should be avoided if possible. To avoid the multiplication the address generator architecture can be replaced with modules similar to those used in turbo decoding [28], [29] for WiMAX and 3GPP standards. These parallel address generators are also better suited to a future parallel architecture study.

The single receiver core described in all tables has parameters $H = 12$, $P = 8$, $N = 32$, $M = 4$, and $L = 512$. The Synopsys synthesized module breakdown is shown in Table V. The area is presented as a percentage of the receiver core area and as a percentage of the core area, but without the inclusion of the memory blocks. This is done to avoid the utilization skew of the memory cost and present a comparison among blocks that perform calculations. The variance calculation is the largest of all the blocks roughly taking up 1/4 of the combinational design space. In the overall design the combined memory of the interleavers occupies 85% of the design space, so transforming the interleaver memories to DRAM or SRAM instead of registers is certainly worth the effort for future versions. This also indicates that transforming the core to parallel processing is worth further investigation.

The layout of a 50-user system built in Encounter EDI 9.1 is presented in Figure 8. The netlist standard cell area has a total logic area of 9,824,942 $\mu m^2$, ignoring the overhead of the clock tree insertion. Each individual receiver block contributes almost 2% to the total design and the K-operand H-bit adder has an area of 18,108 $\mu m^2$, which is approximately 9% of a receiver core with respect to area. The layout in Figure 8 is a partitioned design, where each receiver core is analyzed, placed and routed separately before being added later to the entire system. Extra space was added for routing the cores together and the final layout with standard input output pads is 4.52mm by 4.52mm.

Fig. 8. 50-user PS-CDMA receiver layout

C. Other DS-CDMA Implementations

In Table VI are additional implementations presented alongside this work. We try to compare any values reported, but most designs are unique in their goals and methods.

In [9] an FPGA implementation for frequency selective (FS) channels with DS-CDMA is presented. The authors utilize block parallel decision feedback multistage parallel interference cancellation (BP-DF-MPIC) detectors implemented on a Xilinx XC2VP100 FPGA. Their focus is improving the BER performance with minimal increase in logic resources. We
quote their block parallel $BP = 3$ results. Their FPGA design has an average of 2318 slices per user, whereas our design has 1351 slices per user, but our implementation utilizes large amounts of BRAMs.

In [30] a real-time DSP implementation of DS-CDMA multiuser receiver is presented. The authors implement partial parallel interference cancellation (PPIC) on a DSP and include radio frequency design elements as well as modules for dealing with a noncoherent system.

In [31] a real-time DSP RAKE DS-CDMA receiver is presented. They implement a fully loaded system with processing gain of 8 and 8 users. In [32] a FPGA implementation of DS-CDMA is presented. We quote the values from their base station receiver design. Their design has programmable bit-rates for each user, with the maximum being 2Mb/s per user. Their area cost is presented as a logic element, which is comparable to a slice.

In [33] a real-time DSP implementation for interference cancellation using MMSE is presented. Their results are quoted at an Eb/No of 4dB with a high system load 5/7. They also assume a synchronous detector and can attain a data rate of 186kb/s.

In comparison this work has the largest number of users and throughput for an implemented synchronous system. We also report static power estimates based on a Synopsys analysis of the 50-user system in the 90nm technology node. If $I = 1$, $F = 268.1$MHz and all transmitted bits are assumed decoded we can calculate a low estimate for the energy per information bit of 3.52nJ/bit. This estimate is very aggressive and does not account for parasites, glitches or process variations.

### V. Summary

This paper presented the implementation of a joint detection iterative interference cancellation architecture. The base unit of the architecture is comprised of a few simple components, which allows large system implementations, while still giving good performance. The control module, variance module and the cancellation modules were identified as logic blocks that could be reduced in their logic requirement, while the serial nature of the interleavers was identified as a bottleneck for the throughput and the major contributor to implementation area.

### REFERENCES


digital circuits.