HANDSET DETECTOR ARCHITECTURES FOR DS-CDMA WIRELESS SYSTEMS

Frank Livingston, Vikram Chandrasekhar, Mani Vaya, Joseph R. Cavallaro
Rice University, Dept. of ECE, 6100 S. Main St., Houston, TX 77005
fchl, cvikram, vaya, cavallar@rice.edu

ABSTRACT

This paper investigates detector architectures for wireless handsets employing DS-CDMA. The code-matched filter (MF) and minimum output energy (MOE) detectors are analyzed with respect to fixed-point arithmetic behavior. Architectures employing fixed-point arithmetic are then proposed for these detectors. The maximum throughput of these architectures and the associated costs in terms of area usage and power consumption are evaluated. Results of the fixed-point analysis indicate that the MOE detector is more susceptible to quantization than the MF detector. Results of implementation indicate that the superior performance of the MOE detector is achieved at a considerably higher cost in terms of area usage and power consumption. Finally, comparison of hardware implementation with software-based DSP implementation indicates that software approaches result in considerably lower throughputs.

1. INTRODUCTION

Direct Sequence Code Division Multiple Access (DS-CDMA) is widely accepted as a bandwidth efficient protocol for multi-user wireless communications [1]. DS-CDMA systems, however, inherently suffer from multi-access interference (MAI). As a result, systems employing DS-CDMA require advanced receiver structures to achieve acceptable levels of performance.

It is often assumed that the problem of designing a receiver for the forward link in a DS-CDMA system is similar to, but simpler than, the problem of designing a receiver for the reverse link, with design simplifications due to synchronous user signals with equal transmit powers. Such is not the case, however, because of the limited resources and knowledge available to a mobile handset. The low power consumption and compact size requirements of handsets limit the processing power available for advanced receivers. Higher data rates specified by 3rd generation standards – 2 Mbps (local area) and 384 Kbps (wide area) – limit the time within which each bit must be processed. Finally, handsets often lack knowledge of the spreading codes and preambles assigned to other users in the system, necessitating the use of ‘blind’ algorithms.

Two detector algorithms that can reasonably be employed in DS-CDMA handsets given the limited resources, processing time, and knowledge of other users in the system are the MF and MOE detectors. We analyze the MF and MOE detectors with respect to fixed-point arithmetic behavior under the assumption

that floating-point hardware is too expensive for a handset in terms of area and power. The detector architectures are implemented on a Xilinx FPGA and evaluated with respect to area requirements, power consumption, and achieved throughput. For comparison purposes, the detectors are also implemented in software on a TI ‘C6x DSP and analyzed with respect to achieved throughput.

2. BACKGROUND

A K -user, synchronous DS-CDMA system employing BPSK modulation is assumed for the purposes of modeling the forward link data received at the handset. During each symbol interval, the base station sends a symbol $b_k \in \{-1, +1\}$ to the k^{th} user. This symbol is spread by the user’s length N_c signature sequence $\mathbf{s}_k$, where $\|\mathbf{s}_k\|^2 = 1$. At the handset, the received continuous time waveform is discretized using a chip-matched filter. Assuming symbol-level timing has been acquired (e.g. using a channel estimator), this chip-matched filter output can be expressed as

$$\mathbf{y}[i] = A \sum_{k=1}^K b_k[i] \mathbf{s}_k + \boldsymbol{\sigma}\mathbf{m}[i] \quad (1)$$

where $\mathbf{y}[i]$ is the chip-matched filter output for the i^{th} symbol interval, A is the received amplitude, and $\boldsymbol{\sigma}\mathbf{m}$ represents additive white Gaussian (AWGN) noise with variance σ^2 [2].

2.1. MF Detector

The MF detector [2] matches to the signature waveform of the user of interest. Assuming user 1 is the user of interest, this detector computes a hard decision for the i^{th} symbol as

$$b_{1,mf}^{est}[i] = \text{sgn}(\mathbf{y}[i]^T \mathbf{s}_1). \quad (2)$$

This detector is computationally efficient and optimal in single-user AWGN channels. However, it is unable to combat MAI.

2.2. MOE Detector

The MOE detector [3] processes the received chip-matched filter output such that the mean-squared error between the processed output and the symbol of the user of interest is minimized. Again assuming user 1 is the user of interest, this can be expressed as:

$$\mathbf{c}_1^* = \arg \min_{\mathbf{c}_1 \in \mathbb{C}^{N_c}} E[(b_1 - \mathbf{y}[i]^T \mathbf{c}_1)^2] \quad (3)$$

where $\mathbf{c}_1^*$ is the filter with which the MOE detector processes the chip-matched filter output. It can be shown that $\mathbf{c}_1^* = \mathbf{s}_1 + \mathbf{x}_1$, where $\mathbf{x}_1$ is orthogonal to $\mathbf{s}_1$. The MOE detector calculates $\mathbf{x}_1$ adaptively over a number of symbols using stochastic gradient-based techniques.

The MOE detector computes a soft decision $Z[i]$ for the i^{th} symbol based on the received chip-matched filter output $\mathbf{y}[i]$, the spreading code $\mathbf{s}_1$, and the most recent iterate of the adaptive vector $\mathbf{x}_1[i-1]$. It begins this computation by forming the matched-filter outputs given by

$$Z_{mf}[i] = \mathbf{y}[i]^T \mathbf{s}_1 \quad (4)$$

$$X_{mf}[i] = \mathbf{x}_1[i-1]^T \mathbf{s}_1. \quad (5)$$

The soft decision is then computed as

$$Z[i] = Z_{mf}[i] + \mathbf{y}[i]^T \mathbf{x}_1[i-1]. \quad (6)$$

After computing the soft decision, the adaptive vector is updated using

$$\mathbf{x}_1[i] = \mathbf{x}_1[i-1] - \mu Z[i](\mathbf{y}[i] - Z_{mf}[i]\mathbf{s}_1) - X_{mf}[i]\mathbf{s}_1. \quad (7)$$

The rate of convergence of $\mathbf{x}_1[i]$ to $\mathbf{x}_1$ is governed by the parameter μ shown in the second term of Equation (7). Numerical inaccuracies in a practical implementation of the MOE detector can cause $\mathbf{x}_1[i]$ and $\mathbf{s}_1$ to lose orthogonality. The third term in Equation (7) is included in the update to maintain orthogonality between $\mathbf{x}_1[i]$ and $\mathbf{s}_1$. After the adaptive vector update, the detector computes a hard decision for the i^{th} symbol as

$$b_{1,moe}^{est} = \text{sgn}(Z[i]). \quad (8)$$

Unlike the MF detector, the MOE detector is able to mitigate MAI. The superior performance of the MOE detector in multi-user channels comes at the expense of added complexity.

3. QUANTIZATION ANALYSIS

The dynamic range and precision requirements of the variables used in the MF and MOE detectors must be known before these detectors can be implemented using fixed-point arithmetic. This section presents a fixed-point word length analysis of the MF and MOE detectors.

To begin the quantization analysis, a dynamic analysis tool [4] was used to estimate the dynamic range and precision requirements of the detectors. This tool operates in two steps. In the first step, the tool collects dynamic range statistics for variables of interest from a floating-point C version of an algorithm under test. In the second step, the tool analyzes the collected data using an interactive Matlab program. Dynamic range statistics for the detectors were collected while using synthesized receiver data as detector input. The input data was synthesized using length 31 Gold codes, $K = 15$, and a signal-to-noise ratio (SNR) of 0 dB. The data was scaled to have amplitude values between -1 and $+1$ prior to being input to the detectors. The MOE detector

was executed with a convergence parameter μ of $1/64$, which was within theoretical limits for the given SNR and MAI. Estimated word lengths for the detector variables based on the output of the quantization tool are presented in Table 1.

Simulations were performed for the detectors using custom formats as per the formats in Table 1 and using a uniform word length for all variables. Using custom formats would be suitable for implementation of the detectors in an ASIC while uniform word lengths would be suitable for either an ASIC or a DSP implementation. The simulations were performed to provide a conservative estimate of the required word length: the results of computations were truncated instead of rounded, and intermediate results in iterative calculations were not maintained with higher precision than the input data. The simulations were performed using Matlab/C with SystemC providing fixed-point arithmetic support through C++ classes.

Two simulations were performed for both custom formats and uniform word lengths. In the first set of simulations, the MAI level was fixed by setting $K = 15$, and the SNR was varied. In the second set, the SNR was fixed at 0 dB, and the MAI varied. In both sets of simulations, the bit error rate (BER) calculated for a floating-point version of each detector was used to determine the best performance that could be expected of a fixed-point version of the detector. For the uniform formats, BERs were computed for several different word lengths. As before, length 31 Gold codes were used for data synthesis and the MOE detector was executed with $\mu = 1/64$.

The custom format simulation results indicate that the for-

Common MF/MOE Variables		
Variable	Word Length	# Integer Bits
$\mathbf{y}[i]$	19	1
$\mathbf{s}_1$	5	0
$\mathbf{y}[i]^T \mathbf{s}_1$	20	1
MOE Variables		
Variable	Word Length	# Integer Bits
μ	7	0
$\mathbf{x}_1[i]^T \mathbf{s}_1$	18	0
$Z[i]$	19	1
$\mathbf{x}_1[i]$	18	0

Table 1. Estimated Detector Word Lengths

malts presented in Table 1 give near floating-point performance. The uniform word length results presented in Figure 1-Figure 4 indicate fixed-point versions of the detectors employing uniform word lengths approach the limiting performance of floating-point versions for a word length as small as 8 bits for the MF and 16 bits for the MOE detector. The MOE detector thus outperforms the MF detector, but requires twice the word length of the MF detector to achieve this result. Further, it is apparent from Figure 1-Figure 4 that the MOE detector is far more sensitive to quantization than the MF detector.

Figure 1. BER vs. MAI for MF

Figure 2. BER vs. SNR for MF

4. ARCHITECTURE

4.1. MF Detector Architecture

The MF detector can be implemented by having an accumulator-based unit compute the decision $b_{l,mf}^{est}[i] = \text{sgn}(\mathbf{y}[i]^T \mathbf{s}_l)$, where $\mathbf{y}[i]$ is the received chip-serial sequence. Since the elements of the signature vector are numerically equal ($\mathbf{s}_l \in \{-1, +1\}$), the matched filtering block can be implemented as a simple accumulator using a two-input multiplexer that selects between its input $\mathbf{y}[i]$ and its two's complement. Let $f_{c,mf}$ be the clocking rate of the architecture. Then, the MF detector achieves a data rate of $R_{mf} = f_{c,mf} / N_c$ and a chipping rate of $C_{mf} = f_{c,mf}$.

4.2. MOE Detector Architecture

We describe a fully pipelined chip-serial architecture for the MOE detector employing fixed-point arithmetic. The chip-serial architecture offers the advantages of reduced hardware complexity as compared to a fully chip-parallel implementation.

For the i^{th} iteration, the chip inputs $\mathbf{y}[i]$ are assumed to arrive in a chip-serial fashion. Every sequence of N_c chips of incoming serial data $\mathbf{y}[i+1]$ is buffered and used for the computation of the adaptive vector $\mathbf{x}_l[i]$ during the $(i+1)^{\text{th}}$ iteration.

Figure 3. BER vs. MAI for MOE

Figure 4. BER vs. SNR for MOE

Certain optimizations can be applied in order to simplify the resulting architecture (ref. Equation (7)). The value of μ is chosen to be a power of two so that the product $\mu Z[i]$ can be implemented as a shifter rather than a relatively expensive multiplier. Since the signature $\mathbf{s}_l$ is antipodal, the inner products $(\mathbf{y}[i]^T \mathbf{s}_l, \mathbf{x}_l[i]^T \mathbf{s}_l, \mathbf{y}[i]^T \mathbf{x}_l[i-1])$ can be separated into an accumulator block and a multiply accumulate block.

The pipelined implementation of the above update equation is given below in Table 2. In the diagram, the letter 'E' denotes execution of an operation while 'F' denotes the completion of an operation. For illustration purpose, we assume a spreading sequence of length seven. We note that the incoming chip data $\mathbf{y}[i+1]$ (during the $(i+1)^{\text{th}}$ iteration), arrives in bursts of seven chips each. This is done in order to prevent a potential conflict when there is more outstanding data in the buffer than the rate at which the adaptation takes place.

Let $f_{c,moe}$ be the clocking rate of the architecture. The pipelined chip-serial implementation achieves a throughput (detection rate) of $R_{moe} = f_{c,moe} / (N_c + 5)$ and a maximum chip arrival rate of $C_{moe} = N_c f_{c,moe} / (N_c + 5) = N_c R_{moe}$. In Table 2, seven chips of data arrive between the clock cycles 1 and 12, so the effective chipping rate is $C_{moe} = 0.583 f_{c,moe}$.

The MF and MOE detector architectures were implemented on a Virtex XCV800 FPGA and a TI 'C6x DSP using a length 31 Gold spreading code. The Virtex power estimator was used to estimate the power consumption of the FPGA for both architec-

Cycle Number	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\mathbf{y}[i]$	y_6						y_0	y_1	y_2	y_3	y_4	y_5	y_6			
$Z_{mf}[i]$	E	E	F						E	E	E	E	E	E	F	
$Z_{mf}[i]\mathbf{s}_1$				F	F	F	F	F	F	F						
$X_{mf}[i]$	E	E	F						E	E	E	E	E	E	F	
$\mathbf{y}[i] - Z_{mf}[i]\mathbf{s}_1$					F	F	F	F	F	F	F					
$\mu Z[i](\mathbf{y}[i] - Z_{mf}[i]\mathbf{s}_1)$						F	F	F	F	F	F	F				
$\mathbf{x}[i-1] - X_{mf}[i]\mathbf{s}_1$						F	F	F	F	F	F	F				
$\mathbf{x}[i]$	x_6						x_0	x_1	x_2	x_3	x_4	x_5	x_6			
$\mathbf{y}[i+1]^T \mathbf{x}[i]$	E	E	F						E	E	E	E	E	E	F	
$Z[i]$				F												F

Table 2. Pipeline Diagram

Detector	Fixed-Point Precision (bits)	4 Input LUT	CLB Slices	Gate Count (Gates)	Chip Rate (Mcps)	Throughput (Mbps)	Power Estimate (mW)	DSP Throughput (Mbps)	DSP Power Estimate (mW)
MF	16	78	42	745	99.71	3.22	310	1.61	810
	10	48	26	481	94.06	3.03	269	-	
	8	36	21	381	100.38	3.24	259	-	
MOE	16	1430	763	51269	45.3	1.46	1221	0.135	530
	14	1138	616	47495	47.72	1.54	1069	-	
	12	744	416	42641	48.61	1.57	788	-	

Table 3. Detector Implementation Results

tures. Uniform word lengths for each detector were chosen based on the quantization analysis reported in the previous section.

Results of the implementation are reported in Table 3. We make two key observations with respect to these results. Firstly, the MF detector benefits from its simplicity by requiring just 1.45% of the area of the MOE detector (fixed point word-length of 16). Secondly, it achieves a maximum throughput that is twice that of the MOE detector, while its power consumption is a quarter of the MOE detector. This indicates that the higher BER performance of the MOE detector is achieved at a significantly higher cost in terms of area usage and power consumption.

We also compare the performance of the detectors given a software-based DSP (fixed data path) implementation versus a custom implementation on an FPGA. The ratio of the DSP throughput to the FPGA throughput is 1/2 for the MF detector and 1/11 for the MOE detector, indicating that superior data-rates are achievable in an ASIC.

5. CONCLUSIONS

We have presented a performance-cost analysis of the MF and MOE detectors. Fixed-point analysis of the detectors indicate that the MF and MOE detectors can operate for word lengths as small as 8 and 16 bits, respectively. We infer that the MOE detector is more sensitive to quantization than the MF detector. Results of implementation show that the MF detector can be realized in an area that is 1.45% of the MOE detector. Further,

the MF detector achieves a throughput of twice the MOE detector for a quarter of the power consumption. The MOE detector thus costs significantly more for its superior performance. Finally, a DSP software-based solution for the detectors results in markedly lower throughputs compared to a hardware-centric approach.

ACKNOWLEDGEMENTS

This work was supported in part by Nokia Corporation, Texas Instruments Inc., the Texas Advanced Technology Program under grant 1999-003604-080, and by NSF under grant ANI-9979465.

6. REFERENCES

- [1] U. Madhow, "Blind Adaptive Interference Suppression for Direct-Sequence CDMA," *Proc. of the IEEE*, vol. 86, no. 10, pp. 2049-2069, October 1998.
- [2] S. Verdú, *Multiuser Detection*, Cambridge, UK: Cambridge University Press, 1998.
- [3] M. Honig, U. Madhow, and S. Verdú, "Blind Adaptive Multiuser Detection," *IEEE Trans. Info. Theory*, vol. 41, no. 4, pp. 944-960, July 1995.
- [4] D. Linerbarger, F.A. Abi Zeid and A.R. Shrivastava, "Dynamic Range Tool", Signal Processing Lab. Engineering and Computer Science Department, University of Dallas, Texas, December 2000.