

# A MIMO RECEIVER SOC FOR CDMA APPLICATIONS

Tongtong Chen   Zhengtao Yu   Yuantao Peng   Yanbin Zhang   Huaiyu Dai   Xun Liu

Department of Electrical and Computer Engineering  
North Carolina State University, Raleigh, NC 27695

## Abstract

In this paper, we present a Systems-on-Chip (SoC) design for the 3G Code Division Multiple Access (CDMA) receiver using the multiple-input multiple-output (MIMO) technique. Our chip integrates the entire digital signal processing part of the receiver. Furthermore, the proposed design can be reconfigured in real-time to handle different modulation schemes based on the signal-to-noise ratio, resulting in the highly efficient use of spectrum and energy. Designed using a 0.18 $\mu\text{m}$  standard cell library, our chip has a core area of 20 mm<sup>2</sup> and achieves a maximal throughput of 5 Mbps in simulation with 610 mW power dissipation.

## 1 Introduction

MIMO technology is a scheme that promises substantial increase of wireless channel capacity by deploying antenna arrays at both transmitters and receivers. Unfortunately, the design of MIMO SoCs is limited due to the high computational complexity of MIMO algorithms and stringent power constraints of MIMO applications. This paper presents a SoC implementation of the 3G CDMA receiver that utilizes the MIMO technology. The receiver targets at a MIMO channel with four transmit antennas and four receive antennas. Our chip has a very high level of system integration. It contains all digital processing components of the MIMO CDMA receiver and is able to perform signal equalization, maximum likelihood MIMO detection, and turbo decoding. The novelty of our design also stems from its reconfigurability. The current implementation can be reconfigured in real time into two decoding modes that handle 16QAM and QPSK modulations, respectively. More modulation schemes can be handled by replacing a ROM in our design with a flash memory and corresponding flash memory updating circuits.

Our SoC design strategy achieves a high-performance and low-power MIMO receiver. In particular, the integration of a large number of modules on a single chip not only provides a high computational capacity but also reduces the overhead due to the data transfer among modules. Our MIMO receiver is designed using a 0.18 $\mu\text{m}$  CMOS technology. Simulation results have shown that our receiver achieves a 5 Mbps transmission rate at each user's end. The chip core area and average power dissipation are 20mm<sup>2</sup> and 610 mW, respectively.

The rest of this paper has four sections. Section 2 reviews background information on MIMO designs. Section 3 describes the details of our MIMO receiver. Section 4 presents our design procedure, final chip layout, and simulation results. Section 5 summarizes this paper.

## 2 Background

### 2.1 CDMA MIMO communication

Figure 1 shows the encoding procedure in a CDMA MIMO channel with four transmit antennas. The raw data from several users are first encoded using convolutional codes. The coded bit streams are converted into symbols and spread using Walsh codes, which are different for various users. The symbols from multiple users are added together to form the transmitted symbols. Since four transmit antennas are used, the transmitted symbols are distributed to four transmission paths evenly. In each path, the symbols are combined with the pilot symbols first. The results are multiplied with scrambling codes to increase the overall signal randomness. Finally, the scrambled symbols are sent out through an antenna. Note that different transmission paths have different pilots and scrambling codes.

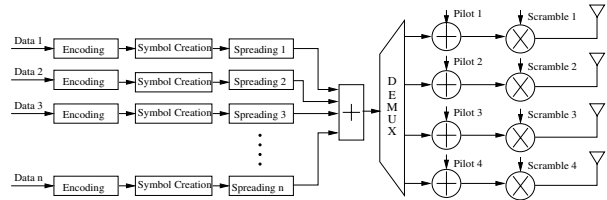


Figure 1. A CDMA transmitter diagram.

The transmitted signals from all antennas are combined in the wireless channel. On the receiver side, the continuous signals are sampled to produce the received symbols. Mathematically, if  $x_{k,j}$  represents the transmitted symbol  $j$  at antenna  $k \in \{0, 1, 2, 3\}$  and  $N$  is the interference window width, the  $i$ th received symbol at receive antenna  $m \in \{0, 1, 2, 3\}$ , denoted as  $r_m(i)$ , can be described as

$$r_m(i) = \vec{C}_m \cdot \vec{x}(i) + n(i), \quad (1)$$

where  $n(i)$  is the additive white Gaussian noise,  $\vec{C}_m$  is the space-time channel vector with a dimension of  $4 \times N$ , and  $\vec{x}$  is a vector consisting of transmitted signals  $x_{k,j}$ ,  $k \in \{0, 1, 2, 3\}$ ,  $j \in \{i, i-1, \dots, i-N+1\}$ .

The receiver recovers the transmitted user data from the received symbols. In particular, since the channel vector  $\vec{C}_m$  is often stable, adaptive filters can be used to estimate transmitted symbols from the received ones. To eliminate the white Gaussian noise, maximum likelihood detection is often applied. To this end, Turbo decoding has been shown very effective in noise elimination [2, 6, 9].

### 2.2 Previous MIMO receiver design

Since the discovery of the MIMO technique, several attempts have been made to implement MIMO based systems. In particular, the Bell Labs layered space-time architecture is the first MIMO architectural design [3]. Early hardware prototypes of MIMO implementations are based on DSP microprocessors or FPGAs [5, 7, 8]. These early designs dissipate prohibitively high power and are therefore impractical for mobile applications. The ASIC implementation of a MIMO receiver has been developed by Bell Labs, Lucent Technologies, targeting at the high speed downlink packet access applications for 3G networks [4]. This chip can only handle QPSK modulation for  $4 \times 4$  MIMO configuration, i.e., four transmit and receive antennas, however. Airgo Networks Inc., has also announced its MIMO chip set, targeting at the next-generation broadband wireless LAN applications. To our knowledge, however, no single chip implementation of MIMO receiver has been reported in the literature that consists of both adaptive filtering and Turbo decoding.

### 3 SoC Design of MIMO receiver

Our SoC is based on the configuration of four transmitters and four receivers. It supports two modulation schemes: 16QAM and QPSK. Figure 2 shows the overall system architecture with five key components. The input buffer serves as the temporary storage for the received symbols. The equalization module applies the adaptive filtering technique to remove the inter-symbol interference and spatial correlation among antennas. It also despreads the equalized symbols. The flat channel estimation module uses the equalization results to estimate the channel matrix. The sphere decoder produces a posteriori probabilities (APP) of potential transmitted symbols. The APP results are then used by the turbo decoder to compute the user data. The following sections describe the detailed implementations of the equalization module, the sphere decoder, and the turbo decoder.

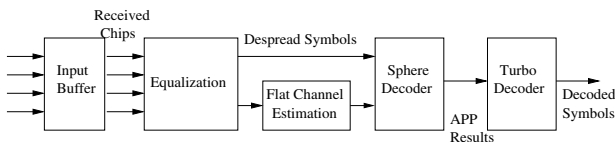


Figure 2. MIMO receiver diagram.

### 3.1 Equalization module

The equalization module consists of four equalizers, deriving the transmitted symbols from four transmit antennas, respectively. The difference among them is that different pilots and scrambling codes are used. Figure 3 shows the block diagram of one equalizer. It contains four finite impulse response (FIR) filters, one for each receive antenna. The filter lengths are set to 16 to handle inter-symbol interference that spans across 16 consecutive symbols. The filter outputs are added together. The summation result is despread to generate the equalized symbols.

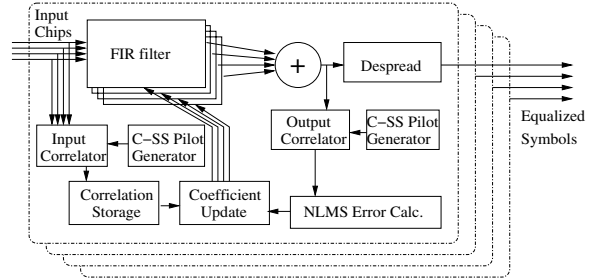


Figure 3. Equalizer diagram.

We adopt the correlation-based scheme in [4] to generate the coefficients for the FIR filters. Specifically, two correlator modules are designed. The input correlator contains four submodules, each of which performs the correlation of one distinct input symbol sequence and a special pilot sequence, called C-SS pilot. The C-SS pilot is the conjugate of the spread and scrambled pilot. The output correlator correlates the summation of four FIR outputs and the C-SS pilot. The result derived by the output correlator is compared with the expected amplitude. The difference, denoted as the normalized least mean square (NLMS) error, is then used in conjunction with the outputs from the input correlator to update the coefficients of the FIR filters.

### 3.2 Sphere decoder

The equalization module is not effective in reducing the white Gaussian background noise in the communication channel. As a result, its output symbols  $\vec{y}$  unlikely match any constellations. If the entire effect of the channel and equalization procedure is modeled as a flat-channel matrix  $\hat{H}$ , the transmitted symbol vector  $\vec{s}^*$  should satisfy  $\vec{y} = \hat{H}\vec{s}^*$ . In practice, a flat channel matrix estimation  $H$  is used to approximate  $\hat{H}$ . The transmitted symbol vector  $\vec{s} = \{s_0, s_1, s_2, s_3\}$  that minimizes the cost function  $J(\vec{s}) = \|H\vec{s} - \vec{y}\|^2$  can be used as the estimation for  $\vec{s}^*$ .

Our MIMO receiver targets at the configuration of 4 transmit antennas and 16 QAM modulation. Consequently, there exist  $16^4 = 65,536$  different symbol vectors, making it impractical to derive the  $\vec{s}$  that minimizes  $J(\vec{s})$  by calculating the cost functions of all possible symbol vectors. We implement a technique called sphere decoding to simplify

the search for the minimal  $J(\vec{s})$ . The basic principle of sphere decoding is that the symbol vector  $\vec{s}$  which results in a small  $J(\vec{s}) = \|H\vec{s} - \vec{y}\|^2$  is likely to have a small  $\|\vec{s} - H^{-1}\vec{y}\|^2$ . Consequently, we only calculate  $J(s)$  for a limited number of  $\vec{s}$  around the location  $H^{-1}\vec{y}$ .

Our sphere decoding procedure can be explained by Figure 4. The black dots represent the symbol vectors. The unconstrained estimation  $s' = H^{-1}\vec{y}$  is first calculated. Using  $s'$  as the center, a circle with a gradually increasing radius is created. A search path can be formed by placing the black dots, i.e., symbol vectors, in the order that they are crossed by the circle. The number of symbols on each path is selected based on the receiver throughput requirement. Apparently, different  $s'$  may have different search paths. We partition the symbol vector space into several regions. All  $s'$  in a single region share the same path. A memory module is designed as a look-up table to store all paths so that, whenever an  $s'$  is derived, the corresponding search path can be applied.

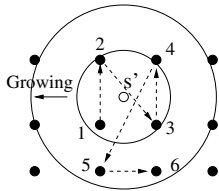


Figure 4. Sphere decoder algorithm illustration.

The block diagram of the sphere decoder is shown in Figure 5. It can be divided into two major parts. The first part includes the decomposition of channel matrix and calculation of the unconstrained transmitted symbol. The second part performs a search procedure that derives the minimum cost function. In particular, a matrix computation block computes the symbol cost function and a matrix enumeration block determines the next candidate symbol along the search path. Both matrix computation and enumeration blocks contain two identical datapaths that operate in parallel for high throughput. All search paths are stored in the path look-up memory (PLM). In our design, the PLM is a ROM that contains paths created for QPSK and 16QAM modulations. Other modulation schemes can be handled by changing the PLM contents or converting the ROM into a programmable memory. A book-keeping module records the cost function values of all symbols visited during the search. It also computes the log-likelihood ratios (LLRs) for each equalizer output  $\vec{y}$  as follows.

$$\ln\left(\frac{P(b_j = 0|\vec{y})}{P(b_j = 1|\vec{y})}\right) \approx \min_{b_j(\vec{s})=1, \forall \vec{s} \in SP} J(\vec{s}) - \min_{b_j(\vec{s})=0, \forall \vec{s} \in SP} J(\vec{s}), \quad (2)$$

where  $b_j$  is the bit  $j$  of the transmitted symbol and  $SP$  is the search path. The transmitted symbols can then be estimated using the LLRs.

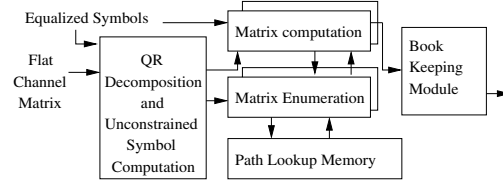


Figure 5. Sphere decoder diagram.

### 3.3 Turbo decoder

Our Turbo decoder conducts Turbo decoding using the LLRs from the sphere decoder and further reduces the communication bit error rate. Its block diagram is shown in Figure 6. It contains a datapath loop since Turbo decoding algorithm performs probability propagation iteratively in the Trellis graph defined by the encoding procedure. In particular, it first derives the branch transition metric  $\gamma$  based on the data from the sphere decoder. It then calculates the forward recursion metric  $\alpha$  and the backward recursion metric  $\beta$ . The values of  $\alpha$ ,  $\beta$ , and  $\gamma$  are updated iteratively and used to derive the decoding results. Although Turbo decoding algorithm consists two decoding processes that operate interchangeably, we implement only one decoder and use it in a time-multiplexing manner to reduce the chip area. We pipeline the datapath for high throughput. Furthermore, we apply the sliding window scheme [1] that limits the length of probability propagation to 16 Trellis columns for memory reduction. In the sliding window scheme, the state probabilities or branch transition metrics in three consecutive windows are calculated simultaneously using the  $\alpha$ ,  $\beta$ ,  $\gamma$ , and pre- $\beta$  blocks. Total 11 rounds of iterations are performed.

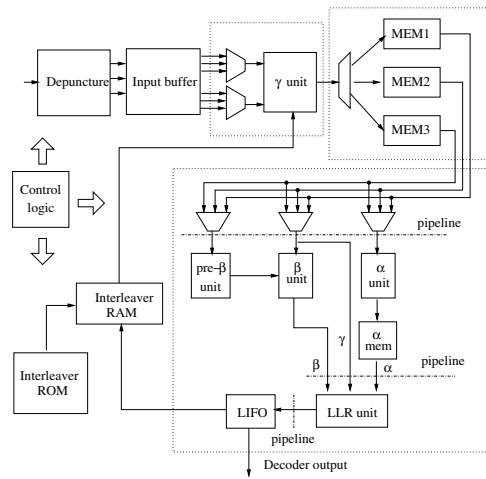


Figure 6. Turbo decoder.

## 4 Chip Design Results

### 4.1 Design flow

Our MIMO receiver is designed using a hierarchical ASIC design flow. Specifically, a MATLAB implemen-

tation of the target MIMO system is created first. In addition to the hardware verification purpose, the derivation of the MATLAB code provides an efficient way to optimize the MIMO receiver at the algorithm level. Once the MATLAB code is finalized, the MIMO receiver is designed in Verilog hardware description language. The datapath and controller modules are synthesized using *design\_compiler* from *Synopsys*. The memory modules are generated using the *Artisan* memory compiler. The layout of our SoC design is created using automatic placement and routing tool *soc\_encounter* from *Cadence*. The functional verification of the MIMO receiver is performed by simulating the synthesized Verilog code and comparing the results in a bit-by-bit fashion with those of the MATLAB simulation.

## 4.2 Chip performance and layout summary

We have simulated our MIMO receiver to demonstrate its performance. Specifically, the transmitted binary data are generated randomly. The transmitted symbols are derived based on the MIMO transmitter design described in Section 2.1. We model the wireless channel as a frequency selective channel with inter-symbol interference. The transfer function of the channel is written as  $H(\omega) = H_0 \sum_{n=1}^{16} \alpha^{n-1} e^{-j\omega n} + N$ , where  $\alpha$  is a constant smaller than 1 and  $N$  denotes the white Gaussian noise.

Figure 7 shows the BERs of our MIMO receiver under two modulation schemes, since our MIMO receiver can be reconfigured to handle 16QAM and QPSK. For each scheme, we performed simulations for three different channels with  $\alpha$  equal to 0.3, 0.4 and 0.5 respectively. For each value of  $\alpha$ , we changed the signal to noise ratio (SNR) by varying the amplitude of the white Gaussian noise  $N$ .

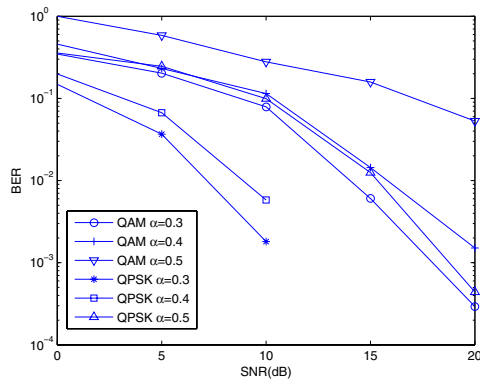


Figure 7. Bit error rate of the MIMO receiver.

The layout of the MIMO receiver is shown in Figure 8. It is designed using a 0.18  $\mu\text{m}$  standard cell library with 6 metal layers. It contains 0.27 million logic gates and 23 memory modules. The chip core area is  $20\text{mm}^2$ . With a power supply of 1.8 volts, the chip can operate at 80 MHz and deliver a throughput of 5 Mbps per user, capable of supporting a 75 Mbps wireless channel with 15 users. The estimated power dissipation is 610 mW.

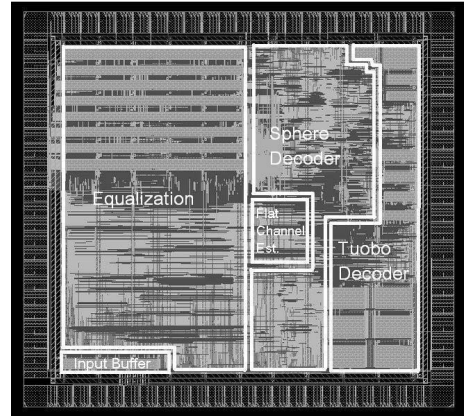


Figure 8. Full chip layout.

## 5 Conclusion

This paper presents a SoC implementation of a  $4 \times 4$  MIMO receiver that targets at CDMA applications. The receiver design achieves a very high level of integration by combining the adaptive filtering based equalization, sphere decoding, and Turbo decoding on a single chip. It can be reconfigured to handle 16QAM and QPSK modulations. Timing analysis results have shown that our receiver can provide a data transfer capacity of 5 Mbps per user. It achieves a BER as low as  $2 \times 10^{-3}$  at a SNR of 10 db.

## References

- [1] S. Benedetto et al. Soft-output decoding algorithms for continuous decoding of parallel concatenated convolutional codes. In *IEEE ICC'96*, pages 140–152, June 1996.
- [2] M. Bickerstaff et al. A 24 Mb/s radix-4 LogMAP turbo decoder for 3GPP-HSDPA mobile wireless. In *Inter. Solid-State Circuits Conference*, Mar. 2003.
- [3] G. J. Foschini. Layered space-time architecture for wireless communication in a fading environment when using multi-element antennas. *Bell Lab Tech. J.*, 1(2):41–59, Aug. 1996.
- [4] D. Garrett et al. A 28.8 Mb/s  $4 \times 4$  MIMO 3G CDMA receiver for frequency selective channels. *J. of Solid-State Circuits*, 40(1):320–330, Jan. 2005.
- [5] T. Kaiser et al. Prototyping for MIMO systems – an overview. In *Eusipco 2004*, Sept. 2004.
- [6] G. Maserà et al. VLSI architectures for turbo codes. *IEEE Trans. VLSI Systems*, 7(3), Sept. 1999.
- [7] P. Murphy et al. A hardware testbed for the implementation and evaluation of MIMO algorithms. In *Inter. Conf. on Mobile and Wireless Communication Networks*, Oct. 2003.
- [8] A. van Zelst and T. C. W. Schenk. Implementation of a MIMO OFDM-based wireless LAN system. *IEEE Trans. Signal Processing*, 52(2), Feb. 2001.
- [9] Z. Wang. High performance, low complexity VLSI design of turbo decoders. *PhD Thesis, Univeristy of Minnesota*, Sept. 2000.