14th European Signal Processing Conference (EUSIPCO 2006), Florence, Italy, September 4-8, 2006, copyright by EURASIP

AN 8×8 RLS BASED MIMO DETECTION ASIC FOR BROADBAND MIMO-OFDM WIRELESS TRANSMISSIONS

Jingming Wang and Babak Daneshrad

Electrical Engineering Department
University of California, Los Angeles, CA 90095, USA
phone: +1 310-825-7792, email: babak@ee.ucla.edu

ABSTRACT

This paper presents the architecture and VLSI implementation of a highly flexible MIMO detection engine which supports a wide array of configurations ranging from 1×1 to 8×8 square, as well as all possible non-symmetric MIMO configurations. The chip is specifically designed to work with an underlying OFDM modulation scheme, and covers the range of 64 to 1024 subchannels. It implements an RLS based MIMO solution which provides a good balance between hardware complexity and overall system performance [1]. To further reduce the complexity frequency domain linear interpolation is also used [2].

The actual implementation is based upon the highly scalable inverse QR decomposition based systolic array architecture [3]. A single systolic array is time-multiplexed for all OFDM subchannels. This naturally overcomes the pipelining difficulty inherent in traditional single channel systolic arrays without doubling the array size. In conjunction with the array design, a unique input tagging scheme is incorporated to allow dynamic reconfiguration of the ASIC on a per packet basis, and also to reduce power consumption when only a sub-array is needed for the operation.

The final implementation of the MIMO detection engine can support configurations as large as 8×8 in 12.5 MHz of bandwidth, or a 4×4 or any smaller array at up to 25 MHz of bandwidth. The chip was fabricated using a 3.3V/1.8V 0.18μm CMOS technology. The resulting core layout measures 29.2mm^2 and clocks at a maximum frequency of 58MHz. In 2×2, 25 MHz configuration, it consumes 360mW, whereas in 12.5 MHz, 8×8 mode it consumes 830mW.

1. INTRODUCTION

The utilization of MIMO techniques in wireless transmission promises a tremendous increase in spectral efficiency [4]. A major challenge faced by a broadband MIMO system is the tremendous processing power required for MIMO detection. The next challenge is to find an architecture that is capable of dynamically restructuring for trading off between system performance and throughput. In this paper, we report on the architecture, design, fabrication, and measurement results of a highly flexible recursive least-squares based MIMO detection ASIC that addresses these challenges. The chip, referred to as the RELIC, is designed for OFDM based spatial multiplexing wireless communication systems and can support any MIMO antenna configuration up to 8×8, symmetric or non-symmetric, at bandwidths of 25MHz or less.

In Section II, we provide an overview of the MIMO channel model, the recursive least-squares (RLS) algorithm for MIMO processing, and its inverse QR-decomposition based systolic array (IQR-RLS) [3]. When the number of antennas is relatively large (> 4), the RLS algorithm provides a better performance versus cost tradeoff [1] compared to algorithms such as BLAST [5] and sphere decoder [6]. Section III provides details about the VLSI implementation of the RELIC ASIC. Targeted for OFDM systems, this engine features frequency-domain scalability, a time-multiplexed systolic array that avoids the doubling in size of a traditional triangular array for pipelining, and a unique tagging system that allows the array to be dynamically reconfigured on a packet-by-packet basis. This level of reconfigurability will help maximize power saving while allowing the system to quickly respond to environmental changes. The characteristics of the final implementation is presented in Section IV.

The notation used in this paper is as follows: bold capital letters stand for matrices (e.g. $\mathbf{H}$), bold lower-case letters for *column* vectors (e.g. $\mathbf{y}$), and italic lower-case letters for scalar variables (e.g. i). $(\cdot)^{\mathsf{T}}, (\cdot)^*$ represent simple transpose and complex conjugate transpose (Hermitian). All matrices are assumed to be invertible when applicable. N_t and N_r are number of transmit and receive antennas, respectively.

2. THE ALGORITHM

2.1 MIMO Channel Model

Consider a MIMO system in which N_t different signals are transmitted and arrive at an array of N_r receivers via a rich-scattering, flat-fading environment. In matrix-vector form, the system can be viewed as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{v}, \tag{1}$$

where $\mathbf{x}$ is the $N_t \times 1$ transmit signal vector, $\mathbf{y}$ is the $N_r \times 1$ received vector. $\mathbf{H}$ is the $N_r \times N_t$ channel matrix and $\mathbf{v}$ is the additive noise at the receiver. In dispersive channels, OFDM modulation is used such that the entire channel is divided into N_c subchannels [7]. When N_c is sufficiently large, each subchannel can be regarded as flat-fading. Therefore, the MIMO detection over a single dispersive channel is converted into the detection over N_c narrowband flat-fading channels.

2.2 RLS Algorithm

At the receiver, the task is to recover $\mathbf{x}$ from $\mathbf{y}$. Instead of achieving this directly, the recursive least-squares algorithm recursively calculates $\mathbf{W}_i$, the combining weight matrix at time instant i, through the following iteration such that $\hat{\mathbf{x}}_i = \mathbf{W}_i \times \mathbf{y}_i$ is the exponentially weighted least-squares (LS) estimation of the training vector $\mathbf{x}_i$ [3]:

$$\mathbf{W}_i = \mathbf{W}_{i-1} + (\mathbf{x}_i - \mathbf{W}_{i-1}\mathbf{y}_i)\,\mathbf{y}_i^*\mathbf{P}_i, \tag{2}$$

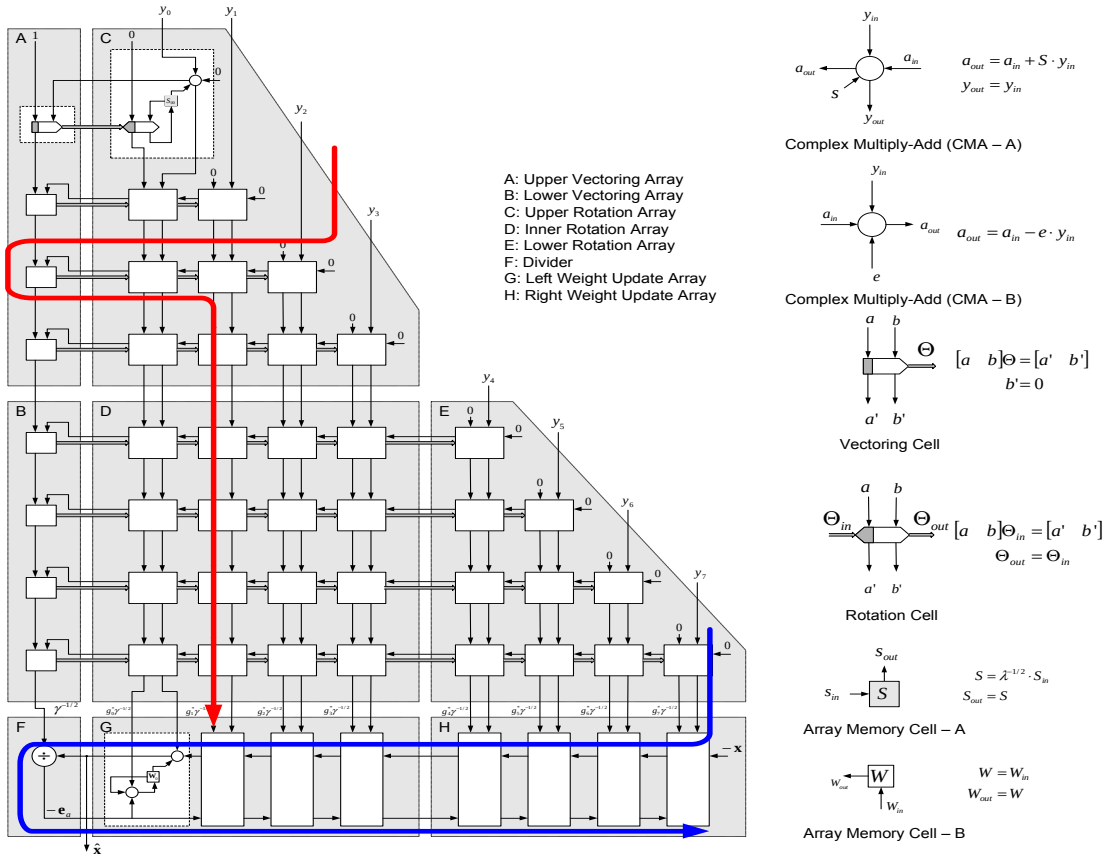

Figure 1: Architecture of RELIC in half band mode

$$\mathbf{P}_i = \lambda^{-1}\left[\mathbf{P}_{i-1} - \frac{\lambda^{-1}\mathbf{P}_{i-1}\mathbf{y}_i\mathbf{y}_i^*\mathbf{P}_{i-1}}{1+\lambda^{-1}\mathbf{y}_i^*\mathbf{P}_{i-1}\mathbf{y}_i}\right], \quad (3)$$

where $0 \ll \lambda < 1$ is the exponential forgetting factor and is usually chosen to be very close to 1, and $\mathbf{P}_i = \mathbf{\Phi}_i^{-1} = \left(\sum_{k=0}^{i}\lambda^{i-k}\mathbf{y}_k\mathbf{y}_k^*\right)^{-1}$ is the inverse of $\mathbf{\Phi}_i$, the weighted correlation matrix of $\mathbf{y}_i$, with initial condition $\mathbf{P}_{-1} = \pi_0\mathbf{I}$. The scalar π_0 is usually chosen to be a large positive number, equivalent to assuming that the initial received signals do not have cross-correlations among different antennas and the received signal power is very low. At the end of training, the RLS iteration is stopped. The final $\mathbf{W}_i$ is then used to combine with the received vectors $\mathbf{y}$ for the rest of the packet to detect the unknown transmitted vectors $\mathbf{x}$.

2.3 IQR-RLS

According to [3], RLS in the form of (2) and (3) is subject to instability under finite precision implementation. A major reason is that $\mathbf{\Phi}_i$, which is a positive definite Hermitian matrix, might be pushed out of its physical domain due to the accumulation of quantization noise. To address this problem, QR decomposition based algorithms have been developed [3, 8]. Three forms of QR-RLS algorithms exist, of which the inverse QR-decomposition based RLS (IQR-RLS) algorithm is preferred over the others for its numerical stability and lower implementation cost [3, 8]. The IQR-RLS algorithm can be formulated as:

$$\mathbf{\Theta}_i\begin{bmatrix} 1 & \mathbf{0}^* \\ \lambda^{-\frac{1}{2}}\mathbf{S}_{i-1}\mathbf{y}_i & \lambda^{-\frac{1}{2}}\mathbf{S}_{i-1}\end{bmatrix} = \begin{bmatrix} \gamma_i^{-\frac{1}{2}} & \mathbf{g}_i^*\gamma_i^{-\frac{1}{2}} \\ \mathbf{0} & \mathbf{S}_i\end{bmatrix}, \quad (4)$$

$$\mathbf{W}_i = \mathbf{W}_{i-1} + (\mathbf{x}_i - \mathbf{W}_{i-1}\mathbf{y}_i)\left[\mathbf{g}_i^*\gamma_i^{-\frac{1}{2}}\right]\left[\gamma_i^{-\frac{1}{2}}\right]^{-1}, \quad (5)$$

where $\mathbf{\Theta}_i$ is a unitary transformation that transforms the pre-array on the left-hand side of (4) to the post-array on the right-hand side. $\mathbf{S}_i$ is the inverse of $\mathbf{\Phi}_i^{1/2}$, the Cholesky factor of $\mathbf{\Phi}_i$, and is therefore a lower triangular matrix. γ_i and $\mathbf{g}_i$ are defined as

$$\gamma_i = (1+\lambda^{-1}\mathbf{y}_i^*\mathbf{P}_{i-1}\mathbf{y}_i)^{-1}, \quad (6)$$

$$\mathbf{g}_i = \mathbf{P}_i\mathbf{y}_i = \lambda^{-1}\gamma_i\mathbf{P}_{i-1}\mathbf{y}_i. \quad (7)$$

γ_i is also referred to as the conversion factor [3] since

$$\mathbf{e}_{p,i} = \mathbf{e}_{a,i}\gamma_i, \quad (8)$$

where $\mathbf{e}_{a,i} = \mathbf{x}_i - \mathbf{W}_{i-1}\mathbf{y}_i$ is *a priori* error and $\mathbf{e}_{p,i} = \mathbf{x}_i - \mathbf{W}_i\mathbf{y}_i$ is *a posteriori* estimation error, and $0 < \gamma_i < 1$.

3. CHIP ARCHITECTURE

3.1 IQR-RLS Systolic Array

The overall IQR-RLS systolic array is shown in Figure 1, along with its basic building elements. It is constructed by concatenating sub-arrays that implement:

1. Matrix-vector multiplication: $\lambda^{-1/2}\mathbf{S}_{i-1}\mathbf{y}_i$;
2. Unitary transformation/Givens rotation: $\lambda^{-1/2}\mathbf{S}_{i-1} \rightarrow \mathbf{S}_i$;
3. Matrix-vector multiplication: $-\mathbf{e}_{a,i} = -\mathbf{x}_i + \mathbf{W}_{i-1}\mathbf{y}_i$;
4. Vector-vector multiplication: $\mathbf{W}_i = \mathbf{W}_{i-1} - \left(-\mathbf{e}_{a,i}[\mathbf{g}_i^*\gamma_i^{-1/2}][\gamma_i^{-1/2}]^{-1}\right)$.

The operation of the array is divided into two parts, the training portion during which the recursion of equations (4) and (5) are performed, and the detection portion where $\hat{\mathbf{x}} = \mathbf{W}\mathbf{y}$ is calculated. The operation of the array during the training is based on the signal flow illustrated in Figure 1. The core is the triangular memory array (C/D/E) that stores $\lambda^{-1/2}\mathbf{S}$ before and after each Givens rotation. First, the received vector $\mathbf{y}_i$ enters C/D/E from the top to get multiplied with the pre-array element $\lambda^{-1/2}\mathbf{S}_{i-1}$. The resulting product vector $\lambda^{-1/2}\mathbf{S}_{i-1}\mathbf{y}_i$ is sent to the vectoring array (A/B) on the left. Meanwhile, the "1" entry in the pre-array of (4) enters A/B from the top. In the top cell of A (vectoring cell in Figure 1), a Givens rotation is applied between 1 and the first element of $\lambda^{-1/2}\mathbf{S}_{i-1}\mathbf{y}_i$ so that the latter is transformed to zero. The rotation result of 1 is propagated downwards to null out the second element of $\lambda^{-1/2}\mathbf{S}_{i-1}\mathbf{y}_i$ in the next vectoring cell. This continues until the $\lambda^{-1/2}\mathbf{S}_{i-1}\mathbf{y}_i$ entry in the pre-array is entirely transformed to $\mathbf{0}$ in the post-array. As the nulling process continues, the Givens rotation direction parameters determined in the vectoring array A/B are propagated rightwards into C/D/E to meet with the row vector $\mathbf{0}^*$ in the pre-array that enters C/D/E from the top. When that happens, the column $[\mathbf{0}^* \;\; \lambda^{-1/2}\mathbf{S}_{i-1}]^\mathsf{T}$ in the pre-array is updated to $[\mathbf{g}_i^*\gamma_i^{-1/2} \;\; \mathbf{S}_i]^\mathsf{T}$ in the post-array. $\mathbf{S}_i$ is saved back to C/D/E and $\mathbf{g}_i^*\gamma_i^{-1/2}$ exits C/D/E from the bottom.

At the bottom of the triangular array is a rectangular memory array (G/H) that stores the weight matrix $\mathbf{W}$ before and after being updated. The received signal vector $\mathbf{y}_i$ is passed down from the triangular array on the top and is multiplied with $\mathbf{W}_{i-1}$. Then $\mathbf{W}_{i-1}\mathbf{y}_i$ is added to the training vector $-\mathbf{x}_i$ that enters from the right side of array H. The result, $-\mathbf{e}_{a,i}$, then gets divided by $\gamma_i^{-1/2}$ that is the final result of 1 in the pre-array after all Givens rotation.

Finally, the weight matrix is updated by multiplying the output of the divider $-\mathbf{e}_{a,i}\gamma_i^{1/2}$ with triangular array output $\mathbf{g}_i^*\gamma_i^{-1/2}$, and adding it to $\mathbf{W}_{i-1}$, same as in (5). Note that a traditional implementation of the weight update array would suggest an 8×8-sized array. However, in practice, it is possible to fold it into a 1×8 array through time-multiplexing.

After RLS training, we enter the decoding operation. Here, all array operations are halted except for calculating $-\mathbf{e}_a$ and passing $\mathbf{y}$ through the triangular array. $\mathbf{x}$ is set to $\mathbf{0}$ such that the output at $-\mathbf{e}_a$ becomes the estimate of the transmitted symbol $\hat{\mathbf{x}} = \mathbf{W}\mathbf{y}$. The array update can resume any time if further RLS training is necessary.

3.2 Scalability

There are two aspects of scalability for a MIMO-OFDM system. The first is the capability to dynamically restructure the RELIC for all antenna configurations, symmetric or asymmetric, such as 2×2, 3×5, etc., up to 8×8. It is preferred that there is no limit on which antenna can be chosen in a particular setup. Mathematically, this is guaranteed by the IQR-RLS equations in (4) and (5) by setting the elements of $\mathbf{y}$ corresponding to the unused antenna to zero. Architecturally, however, this requires careful design of the array control logic. In addition, the cells of the array should be independent of their relative location in the array so as to minimize the number of different types of cells for easy dynamic restructuring. These requirements can be achieved by the input tagging scheme to be introduced later.

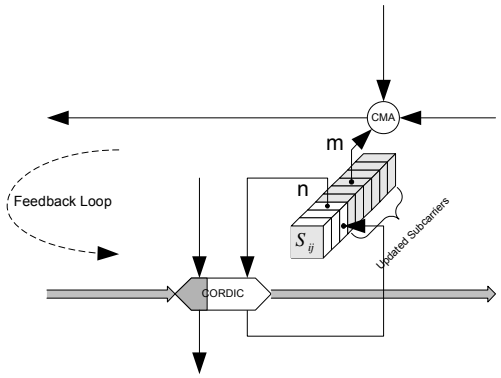

Figure 2: Memory access for different OFDM subchannels

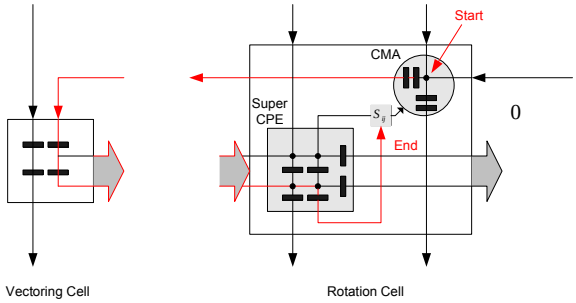

Figure 3: Internal delay in vectoring cell and rotation cell

The second aspect of scalability is the expandability in frequency domain to higher bandwidth and more OFDM subchannels. Given our design target of up to 1024 OFDM subchannels in 25MHz of bandwidth, two different modes – half band and full band, are introduced to make the design more affordable and versatile. Each memory cell is accommodated with 256 memory units to support up to 512 subchannels in 12.5MHz of bandwidth with factor-2 linear interpolation [2]. In the half band mode, the sub-arrays are inter-connected as they are in Figure 1, and up to 8×8 MIMO processing is supported. In full band mode, the inner array D is bypassed. Upper rotation array C is directly connected to G, and E is connected to B, so that they operate like two independent half band arrays covering 12.5MHz each. Hence, a single chip can support up to 4×4 in full band (25MHz) mode. For 8×8, 25MHz spectrum, two chips working in half band mode side-by-side are needed. Support for higher bandwidth is also possible in a similar way.

3.3 Pipelining

A critical problem in designing a fully utilized systolic array is pipelining. Traditionally, there is only one single input channel. Due to the feedback loop in the signal path in the rows of the systolic array (see Figure 1), the content of each memory element in the array must wait for the signal wavefront to come back from the vectoring array to get updated before being available for the next matrix multiplication. Therefore, there is a wait period between each subsequent read and write access. This results in an under-utilized array. Moreover, since the loop delay varies with the location of the cell, so does the wait time, making the design of a location-independent cell very difficult. In [9], this problem is solved by doubling the size of the array, which not

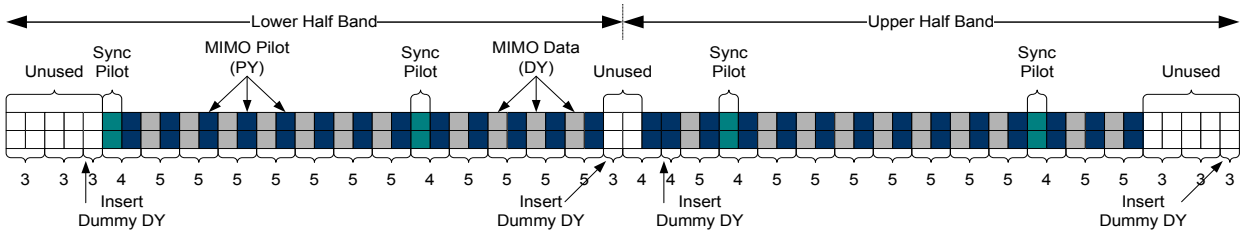

Figure 4: BLK_ID for the training block of a 64-subchannel UCLA METEOR packet[2]

only increases the implementation cost but also exposes the algorithm to more sources of quantization noise.

In a MIMO-OFDM system, this is alleviated naturally by the multi-channel nature of OFDM. In this case, the MIMO detection must be applied to a number of subchannels. If a single array is time-multiplexed for all subchannels, each array memory cell stores the results for all of them. The inherent wait period for a specific subchannel can be used for the processing of the other subchannels. As long as the time-multiplexing is controlled such that the write/update access on each subchannel happens between the two reads, i.e. the read-write order is maintained, there is no wait time for the datapath. Hence the datapath of the array is 100% utilized.

For instance, in Figure 2, suppose the inputs to the complex multiply-accumulate (CMA) unit are coming in from the m-th subchannel to the n-th, where $m < n$. If the current input is the m-th subchannel of the t-th OFDM block, then all the memory content for the k-th subchannel ($k \leq m$) must have been updated from the $(t-1)$-th OFDM block. However, the memory for the n-th subchannel with $n > m$ can continue the updating process of the $(t-1)$-th OFDM block without interfering with the correct CMA operation of the t-th block.

Though mitigated, limitations still exist for multi-channel systems. For the memory that is being read for CMA (subchannel m) and the one that is being updated (subchannel n), it is required that $m \leq n$, i.e. the memory cell for a specific subchannel cannot be read for the matrix multiplication of OFDM block t before being updated from block $t-1$. $m = n$ is valid if the memory update happens before read access for the CMA operation. This relationship between m and n translates to a minimum time interval for the same operation in two consecutive OFDM blocks. This in return correlates the minimum length of an OFDM block to the group delay of each array cell and the maximum size of the effective array.

Take Figure 3 for example. For the (i, j)-th rotation cell, the minimum delay is that between *start* and *end*, which is

$$2(j+1)+2+j+1 = 3j+5, \quad j = 0 \cdots N_r - 1. \qquad (9)$$

Therefore, the maximum of this delay is $3N_r + 2$, which is 14 for $N_r = 4$ and 26 for $N_r = 8$. In the design of RELIC, the input pair rate is 6.25MBaud because of factor-2 linear interpolation. For 64-subchannel OFDM with cyclic prefix length of 16, there are $(64+16)/25 \times 6.25 = 20$ pairs for each half band input. Therefore from $3N_r + 2 \leq 20$, we can calculate the maximum number of receive antennas supported for 64-subchannel MIMO-OFDM to be 6. For 128 subchannels and more, the systolic array can fully support all the antenna configurations up to 8×8. Note that these limits are specific to the delay scheme shown in Figure 3. For a different implementation, e.g. another clock frequency or CORDIC architecture, they could be completely different.

Table 1: Tags and Cell Operations for Different Input Pairs

BLK_ID	Type	Comments
0	Load	Load user defined λ and π_0
1	Reset	Reset systolic array to default state
2	Idle	Idle time between two packets
3	Idle	Idle time between OFDM blocks
4	Training	DY is not a valid symbol
5		DY is a valid symbol
6	Data	DY is not a valid symbol
7		DY is a valid symbol

3.4 Input Tagging

The inputs to the RELIC consist of post-FFT samples with certain packet structure. As there are different types of subchannels in the frequency domain, such as data, synchronization pilots, MIMO pilots, and in the time domain, sequences for different purposes including coarse and fine synchronization, channel training, data, etc., using a global control would make configuring for different antenna selections or packet structures very complicated. Instead, a tagging scheme has been designed to label the inputs to each of the systolic array cells from every direction. Each cell processes the inputs according to its associated tag, not its pre-calculated position in the array. This way, it is possible to minimize the number of different array cell types and restructure them dynamically for different packet structures (i.e. IEEE 802.11n, WiMax, etc.) and different antenna configurations. On the other hand, this also allows each cell to minimize its power consumption by turning itself off when not being used.

In the RELIC, signals from different antennas are assembled into MIMO vector pairs {DY,PY} in a pre-processing interface (not shown in Figure 1), where DY represents the subchannel whose weight matrix is linearly interpolated over its two adjacent pilot subchannels [2] and PY is the received vector on the pilot subchannel next to DY. For each pair, a three-bit tag BLK_ID[2:0] is associated to indicate its part in the overall packet structure. A list of all the different states for BLK_ID is shown in Table 1.

The first three states of BLK_ID are for status when no packet is coming in. During these states, the user can load the desired $\lambda (= 0.99$ by default) or $\pi_0 (= 1$ by default) by setting BLK_ID=0 if values other than the defaults are desired. When BLK_ID=1, a soft reset signal is sent in and the systolic array cells are set to the default state gradually as the soft reset is piped through the entire array. Other than these two states, BLK_ID is set to 2 for idle state between packets.

The other five states of BLK_ID represent status within a packet. BLK_ID=3 is for the idle period between consecutive OFDM blocks. Such pairs appear only at the beginning and end of a half band OFDM block, representing unused

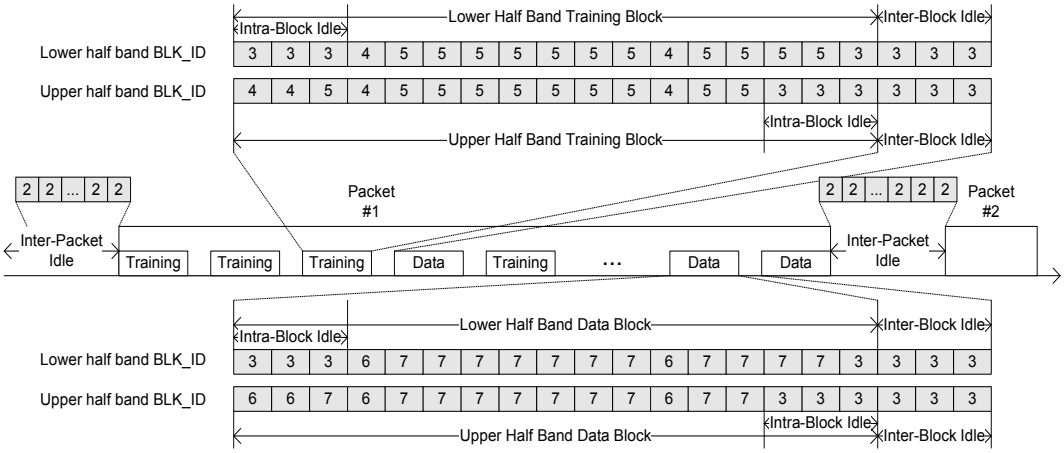

Figure 5: Typical flow of BLK_ID for multiple UCLA METEOR packets

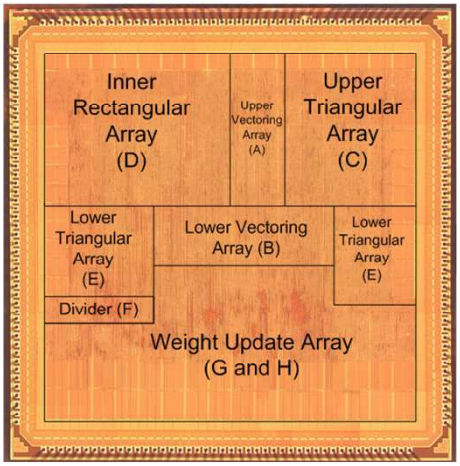

Figure 6: RELIC photomicrograph

subchannels and guard interval that required no RLS update. In the time domain, a packet consists of training and data sequences represented by BLK_ID=10x and BLK_ID=11x, respectively. The last digit of BLK_ID is used to identify whether the DY in the current pair is a valid subchannel that requires MIMO detection (through interpolation) or just a dummy filler that doesn't require any operation.

As an example, the pairing and BLK_ID association for the UCLA METEOR packet structure [2] is shown in Figure 4. As a companion, the flow of BLK_ID in the time domain for a) upper and lower half bands; b) training and data blocks, for the same packet structure is illustrated in Figure 5.

4. SUMMARY

The die of the chip, fabricated using TSMC 0.18μm CMOS technology, is shown in Figure 6. The characteristics of the chip are provided in Table 2. It is capable of supporting 64 to 1024-subchannel OFDM and up to 8×8 antenna configuration with no limit on the constellation size of any subchannel. Therefore, it offers a highly flexible MIMO processing engine for a wide range of broadband spatial multiplexing multiple antenna OFDM systems.

Table 2: RELIC characteristics

Process	TSMC 0.18μm CMOS, 6-level metal, 3.3V/1.8V
Die Size	39.4mm^2 (core: 29.2mm^2)
Gate Count	2.3M (SRAM: 819Kb)
Packaging	181-lead Pin Grid Array (PGA)
Clock	50MHz (max: 58MHz)
Power	360mW (2×2, full band, @58MHz) 830mW (8×8, half band, @58MHz)

REFERENCES

[1] J. Wang and B. Daneshrad, "A comparative study of MIMO detection algorithms for wideband spatial multiplexing systems," in *Proc. IEEE WCNC'05*, New Orleans, LA, USA, 2005, pp. 408–413.

[2] ——, "Performance of linear interpolation-based MIMO detection for MIMO-OFDM systems," in *Proc. IEEE WCNC'04*, Atlanta, GA, USA, 2004, pp. 981–986.

[3] S. Haykin, *Adaptive Filter Theory*, 3rd ed. Prentice-Hall, 1996.

[4] G. J. Foschini and M. J. Gans, "On limits of wireless communications in a fading environment when using multiple antennas," *Wireless Personal Communications*, vol. 6, pp. 311–335, March 1998.

[5] G. J. Foschini, "Layered space-time architecture for wireless communications in a fading environment using multi-element antennas," *Bell Labs Tech. J.*, vol. 1, no. 2, pp. 41–59, Autumn 1996.

[6] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closed point search in lattices," *IEEE Trans. Info. Theory*, vol. 48, pp. 2201–2214, August 2002.

[7] J. A. Bingham, "Multicarrier modulation for data transmission: An idea whose time has come," *IEEE Commun. Mag.*, vol. 28, no. 5, pp. 5–14, May 1990.

[8] B. Yang and J. F. Böhme, "Rotation-based RLS algorithms: unified derivations, numerical properties, and parallel implementations," *IEEE Trans. Acous. Speech Signal Process.*, vol. 40, pp. 1151–1167, May 1992.

[9] M. Moonen and E. Deprettere, "A fully pipelined RLS-based array for channel equalization," *J. VLSI Sig. Proc.*, vol. 14, no. 1, pp. 67–74, Oct. 1996.