TDA Progress Report 42-87

July–September 1986

A Single VLSI Chip for Computing Syndromes in the (255, 223) Reed–Solomon Decoder

I. S. Hsu, T. K. Truong, H. M. Shao, and L. J. Deutsch
Communications Systems Research Section

This article presents a description of a single VLSI chip for computing syndromes in the (255, 223) Reed–Solomon decoder. The architecture that leads to this single VLSI chip design makes use of the dual basis multiplication algorithm. The same architecture can be applied to design VLSI chips to compute various kinds of number theoretic transforms.

I. Introduction

Berlekamp (Ref. 1) has developed for JPL a bit-serial multiplication algorithm for the encoding of Reed–Solomon (RS) codes. This multiplication algorithm uses a dual basis over a Galois field. It is shown in Ref. 2 that by the use of this new dual basis multiplication algorithm, a (255, 223) Reed–Solomon encoder can be realized readily on a single VLSI chip with NMOS technology. Recently, based on the idea of Berlekamp, the authors (Ref. 3) developed a new dual basis multiplier which requires less transistors than normal basis (Ref. 4) or standard basis (Ref. 5) multipliers in VLSI implementation. This new dual basis multiplier can be used as a building block in the Reed–Solomon code decoder design (Ref. 6). It is demonstrated in this article how this dual basis multiplier is used as a basic cell to compute syndromes for a (255, 223) Reed–Solomon decoder (Ref. 7). The number of transistors needed in this design is about 6000. This is much less than it would be if the architecture used in the design of a 4-bit syndrome computing cell were adopted in this design.[1] With that type of architecture, the expected number of transistors needed for the implementation of the syndrome computing chip is about 16,000. The major improvements that lead to the substantial reduction of the number of transistors needed are twofold:

(1) The dual basis multiplication algorithm is used in the present design. In the previous design, the multiplier used is based on the normal basis multiplication algorithm developed by Massey and Omura (Ref. 8). This change results in a saving of about 200 transistors for each multiplier used in the syndrome computation. Totally, it is a reduction of 200 × 32 = 6400 transistors for the syndrome computing chip. The number 32 comes from the fact that there are 32 syndrome subcells included in the (255, 223) Reed–Solomon decoder syndrome computer design.

(2) In the previous design of the syndrome computing chip, the multipliers used are general-purpose multipliers. This means that both the multiplier and multiplicand are variable. This design did not take full advantage of the syndrome computation property. In

[1] H. M. Shao, "A VLSI Syndrome Computing Chip for Reed–Solomon Decoder," private communication, 1985.

computing the syndromes $S_1, S_2, \ldots, S_{32}$ of the received message, the multiplicand is always a fixed element. There is, hence, no need to use a general-purpose multiplier in syndrome computation. The general-purpose multiplier used in the previous design is replaced by a fixed multiplicand multiplier. One serial-to-parallel conversion unit is also saved by this modification. Since one serial-to-parallel conversion unit consists of two sets of 8-bit registers and several control gates, this saving means a reduction of about 100 transistors for each syndrome computing subcell. Alternatively, it is a saving of about 32 × 100 = 3200 transistors for the whole chip.

In conclusion, the number of transistors saved for the present design over the previous design is 6400 + 3200 = 9600 transistors. That is, almost two thirds of the number of transistors are saved by these modifications.

II. A VLSI Design for Computing Syndromes in a (255, 223) Reed–Solomon Decoder

In this section, a VLSI architecture developed in Ref. 7 is used to compute syndromes for a (255, 223) Reed–Solomon decoder over $GF(2^8)$. The dual basis multiplier developed in Ref. 3 is used to compute the inner product in the basic subcell of the syndrome chip. Figure 1 shows the overall diagram of the syndrome chip. In the figure, "Vdd" and "GND" are two power pins. "Phi-1" and "Phi-2" are the inputs for the two nonoverlapping clocks. "Start" is a signal that indicates the start of a received codeword. "Load" and "N-Load" are used to control the serial-to-parallel shifting of input data. These two signals are complements of each other. The shift operation is performed once for every eight bits. "Out" and "N-Out" are used to control the synchronization of input and output data. Again, these two are complementary signals. The function of these control signals will be explained in detail in the following section.

The operations that a syndrome computing chip performs are described below.

The syndromes of the received message for a (255, 223) Reed–Solomon code can be written as

$$S_k = \sum_{n=0}^{254} r_n \alpha^{nk}, \quad \text{for} \quad 1 \leqslant k \leqslant 32 \tag{1}$$

where $\{r_n\}$ is a received message. There are 255 symbols in a code word of a (255, 223) RS code. Hence, n ranges from 0 to 254 for computing the syndrome of each code word.

In order to develop a VLSI architecture to compute the S_k's, first let the expression in (1) be rewritten in recursive form as

$$S_k = (\ldots((0 + r_{254}) \cdot \alpha^k + r_{253}) \cdot \alpha^k \ldots) + r_0, \; 1 \leqslant k \leqslant 32 \tag{2}$$

This expression is the famous Horner's rule. A VLSI architecture is developed to compute (2) as shown in Fig. 2. The function of each cell depicted in Fig. 2 can be best described by the register transfer relation

$$R_B \leftarrow R_B + R_A \times R_C$$

where "←" denotes the operation "is replaced by," R_A, R_B, and R_C are contents of registers A, B, and C, respectively. The received messages are sent into the chip bit-by-bit serially. They are in the order $r_{254}, r_{253}, \ldots, r_0$ for each received code word. Each incoming bit is then sent to all the subcells simultaneously. Figure 3 shows the block diagram of the syndrome computing chip. This chip consists of 32 subcells. Each subcell is responsible for computing a specific syndrome, i.e., subcell S_1 computes syndrome S_1, subcell S_2 computes syndrome S_2, etc. These subcells are arranged in the form of a 4 by 8 matrix with each element in the matrix a syndrome computing subcell. By this organization, the syndrome computing unit can be integrated with the polynomial expansion unit which accepts data from the syndrome computing unit (Ref. 6) without any matching problem. Each subcell shown in Fig. 3 performs the operation described in (2). The block diagram of a subcell is depicted in Fig. 4. As shown in Fig. 4, a syndrome computing subcell is divided into three units: (1) serial-to-parallel conversion unit, (2) multiplication unit, and (3) accumulator register unit. The detailed operation of each unit is explained in the following.

A. Serial-to-Parallel Unit

This unit performs the serial-to-parallel operation. This unit is needed because, as mentioned above, the received messages are a stream of bits, one bit after the other, while the multiplication algorithm is designed to perform operations on the basis of symbols. That is, 8 bits are used to do the calculation simultaneously. The serial-to-parallel conversion is controlled by the signals "Load" and "N-Load." The signal "Load" is high for every 8 bits, which corresponds to the number of bits in a symbol in (255, 223) RS code. When signal "Load" is high, the serial-to-parallel operation is carried out. Otherwise, all the data are shifted forward bit-by-bit.

B. Multiplication Unit

This unit performs the multiplication of an input symbol by a fixed finite element. Since the multiplicand is fixed, it

can be built in the circuit as a stream of exclusive-OR gates (Ref. 3). For example, in subcell S_1, the fixed multiplicand is α^1; therefore, only the output data from the second register in the serial-to-parallel unit is needed as shown in Fig. 4. In subcell S_2, the fixed multiplicand is α^2, only the output data from the third register in the serial-to-parallel unit is required, etc.

C. Accumulator Register Unit

This unit is the temporary storage of the accumulated products for the operation described in Subsection A, above. At the 255th clock cycle for each input code word, the accumulated data in the registers are all shifted out bit-by-bit and added to the input message r_{254}. The result, which is the syndrome for this code word, is then shifted out bit-by-bit from each syndrome subcell. Syndromes $S_1, S_2, \ldots, S_{32}$ are shifted out in parallel. The data shift operations are controlled by signals "Out" and "N-Out." Initially, the control signal "N-Out" is high for 254 symbol times, i.e., it is high for message symbols $r_{254}, r_{253}, \ldots, r_1$. During this period, the received messages are allowed to come into the chip and perform the operation as described by Eq. (2). The temporary sum for each syndrome subcell is stored in their respective accumulator register. When the 255th received symbol r_0 comes in, the control signal "N-Out" is switched to low, and its complementary signal "N-Out" is changed to high. The result in the accumulator register in each syndrome subcell is then added to the incoming symbol r_0 and the results, which are the calculated syndromes, are shifted out of the chip in parallel. This is due to the fact that the next step in RS code decoding process is to perform polynomial multiplication and to multiply the syndrome polynomial with the erasure polynomial. The polynomial multiplication unit is designed to accept data in parallel.

The layout of this syndrome chip is complete and is shown in Fig. 5. The layout of the whole chip was checked by the layout rule checker against a set of layout rules offered by the manufacturer. Logic, circuit, and timing simulations were performed thoroughly so that the risk of design errors is reduced to the least possible. This chip was sent to MOSIS (Ref. 9) for fabrication. After the chips are returned, they will be tested and evaluated. The technology used to fabricate the chip is 3 μm NMOS. The operating frequency is estimated to be around 10 MHz. The power consumption of this chip is estimated to be 150 mW. The total number of transistors in this chip is approximately 6000, and the area of this chip is about $6330 \times 2600\ \mu m^2$.

References

1. Berlekamp, E. R., "Bit-Serial Reed-Solomon Encoders," *IEEE Trans. on Information Theory*, Vol. IT-28, No. 6, pp. 869–874, Nov. 1982.

2. Hsu, I. S., Reed, I. S., Truong, T. K., Wang, K., Yeh, C. S., and Deutsch, L. J., "The VLSI Implementation of a Reed-Solomon Encoder Using Berlekamp's Bit-Serial Multiplier Algorithm," *IEEE Trans. on Computers*, Vol. C-33, No. 10, pp. 906–911, Oct. 1984.

3. Hsu, I. S., Truong, T. K., Shao, H. M., Deutsch, L. J., and Reed, I. S., "A Comparison of VLSI Architecture of Finite Field Multipliers Using Dual, Normal or Standard Bases," presented at Fourth International Workshop on VLSI in Communications, Quebec, Canada, June 1986.

4. Wang, C. C., Truong, T. K., Shao, H. M., Deutsch, L. J., Omura, J. K., and Reed, I. S., "VLSI Architecture for Computing Multiplication and Inverses in $GF(2^m)$," *IEEE Trans. on Computers*, Vol. C-34, No. 8, pp. 52–64, Aug. 1985.

5. Scott, P. A., Tarvares, S. E., and Peppard, L. E., "A Fast Multiplier for $GF(2^m)$," submitted to *IEEE Trans. on Computers*.

6. Hsu, I. S., Shao, H. M., and Deutsch, L. J., "A Design Plan for a Single Chip Reed-Solomon Decoder," to appear in *TDA Progress Report*, Jet Propulsion Laboratory, Pasadena, CA.

7. Shao, H. M., Truong, T. K., Deutsch, L. J., Yuen, J. H., and Reed, I. S., "A VLSI Design of a Pipeline Reed-Solomon Decoder," *IEEE Trans. on Computers*, Vol. C-34, No. 5, pp. 393–403, May 1985.

8. Hsu, I. S., Deutsch, L. J., Truong, T. K., and Shao, H. M., "A VLSI Single Chip 8-Bit Finite Field Multiplier," *TDA Progress Report 42-83*, pp. 45–50, Jet Propulsion Laboratory, Pasadena, CA, Nov. 15, 1985.

9. The MOSIS Project, *The MOSIS System (What it is and how to use it)*, Publication ISI/TM-84-128, Information Science Institute, University of Southern California, Marina Del Rey, CA, 1980.

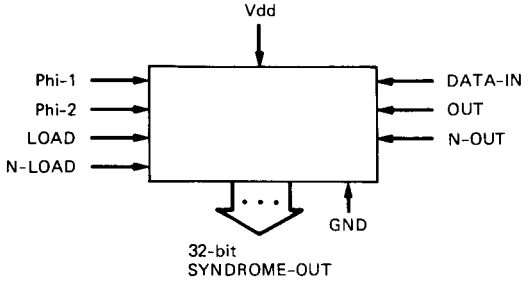
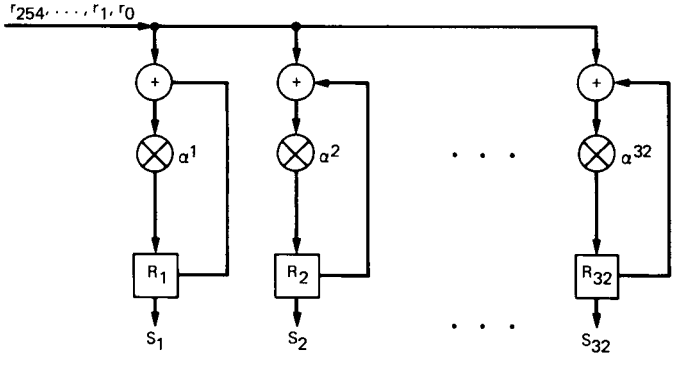
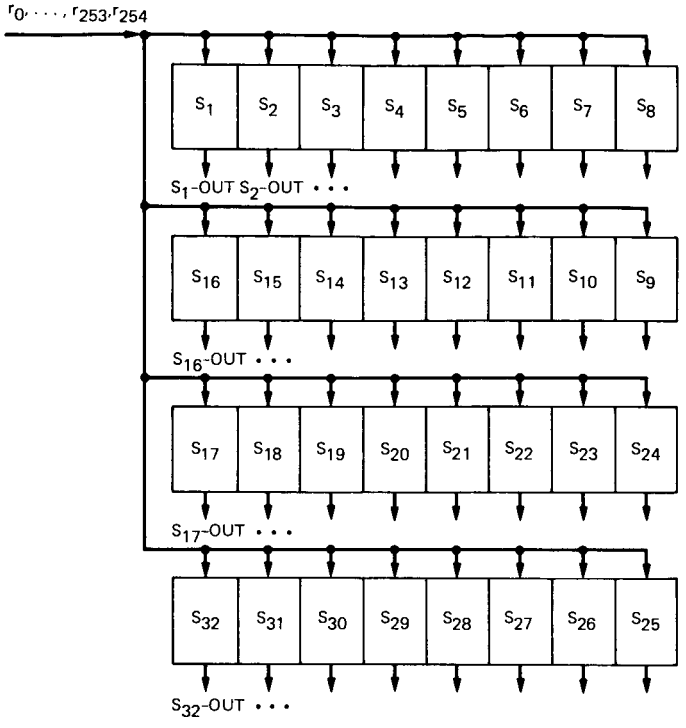


Fig. 1. Overall diagram of the (255, 223) Reed–Solomon code syndrome computing chip



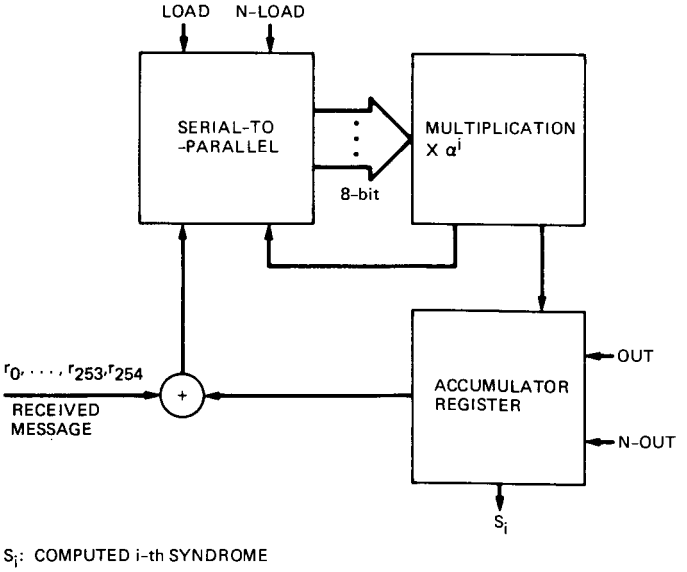
R_i: i-th 8-bit REGISTER

Fig. 2. VLSI architecture for syndrome computing chip of the (255, 223) Reed–Solomon code decoder



S_i-OUT: THE COMPUTED i-th SYNDROME OUTPUT

Fig. 3. Geometric position of each syndrome subcell in the syndrome computing chip



S_i: COMPUTED i-th SYNDROME

Fig. 4. Block diagram of the *i*th syndrome computing subcell for computing syndrome S_i

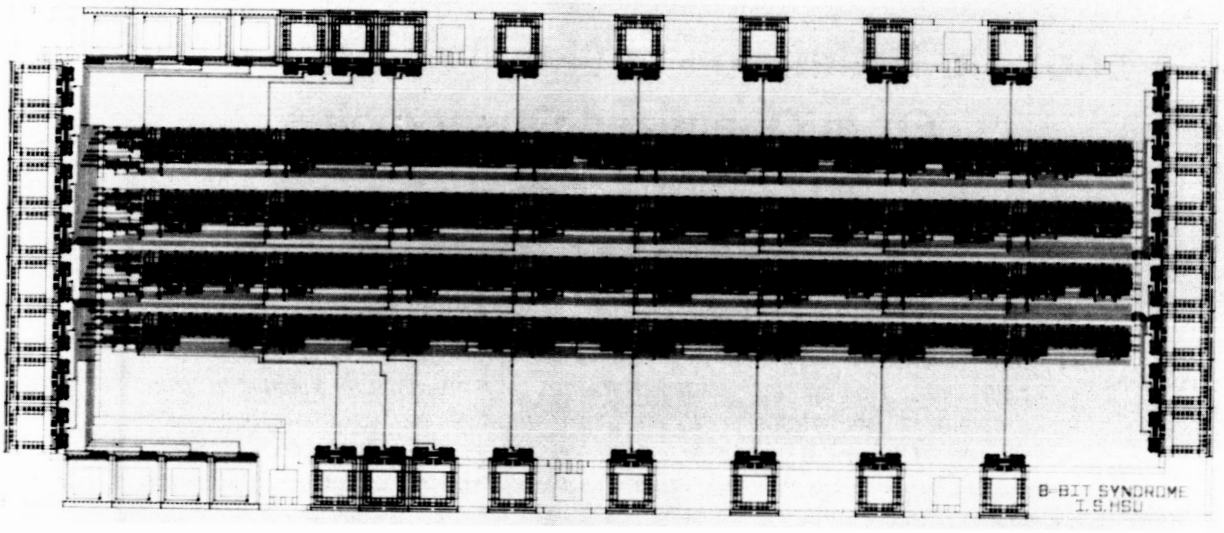


Fig. 5. Layout of the syndrome computing chip for the (255, 223) Reed–Solomon code decoder