九十七年度
低軌道衛星通訊基頻接收器軟體設計
研究委託研究計畫期中報告

中文計畫名稱：低軌道衛星通訊基頻接收器軟體設計研究

英文計畫名稱: Software Design Research for the Baseband Receiver of Low-orbit Satellite Communications

裝

定

線

申請機構：台灣大學

執行單位：電信工程學研究所

主　持　人：電信工程學研究所　陳光禎教授

共同主持人：電信工程學研究所　林茂昭教授

中　華　民　國　9　8　年　　月　　日

Content

Abstract

In order to build up own design capability for LEO remote-sensing satellite communication receiver capability including hardware and software implementation, we proceed the design of baseband receiver for Foremosa-2 front-end Mira board. We proceed from entire system specification, interface (ECL/LVDS, time code, ..), to achieve high-rate data acquisition from baseband receiver. Based on Formosa-2 satellite system requirements, we will deliver the baseband design of receiver in this project, from architecture, specifications, block diagram, implementation design, MATLAB/C implementation, to simulations (in floating-point). It is expected to establish Taiwan's own design capability in LEO satellite communication receiver.

1. Purpose of project

The purpose of this project is to design a satellite communication baseband receiver to meet the system requirement of Formosa-2 satellite system, to reach the purpose of self-control key technology. It is a low-orbital satellite with average height 891km. The communication system parameters include (D)QPSK with concatenated codes to support up to 120M bps over 120M Hz bandwidth. In this proposed project, we shall develop own technology to design LEO satellite baseband receiver and its simulations. The key to success of system integration like this project would be the confirmation of system specification and interface functions. Based on our extensive hand-on experience in satellite communications (PI has worked on INMARSAT, DVB-S, etc. different kinds of satellite communication systems), we will work closely with NSPO to inverse the need of this LEO satellite communication link requirements. Bases on this step, the baseband receiver shall include ECL/LVDS Interface, high rate acquisition and time code interface. The functions of baseband design include frame-sync, unscrambling and RS decoding, which shall be facilitated by block/module design concept. We can therefore complete the design of baseband receiver and evaluate real implementation.

2. Introduction to LEO satellite

2.1 Overview of Low earth orbit satellite

The first artificial satellite has launched on October 4 by Soviet Union in 1957 for 51 years ago. In recent years, low earth orbit (LEO) satellite communication system has become more and more popular. The characteristics of LEO satellite are listed in the following [1]:

- Orbit height: 800-1500 Km height
- Period to encircle the earth: 90~120 mins
- Motion period LEO satellite in sky: 15~20 mins
- Ground antenna is needed to track satellite when satellite is appearing in the sky

Compared to Medium Earth Orbit (MEO) satellite and Geostationary Orbit satellite (GEO), the advantage of LEO satellite are

- Less radiated power from satellite to ground station
- Low propagation delay (< 20 ms)

The disadvantage of LEO compared to MEO and GEO is:

- Higher Doppler shift (about hundreds of kHz)
- Low lifetime because of gravity between LEO and earth is larger than GEO and MEO.

Another characteristic of LEO channel is that with high elevation angle observed from ground base station, the channel between satellite and ground station can be model almost AWGN channel. Similarly, the channel can be model as multipath channel at low elevation angle. The boundary between low and high elevation angle is not obvious and it depends on the true situation.

In the following section, we briefly review the characteristics of our Formosa II satellite system which is also a LEO satellite communication system.

2.2 Formosa II LEO satellite

Formosa II is a Low earth orbit (LEO) satellite communication system. It is a second our own artificial satellite in Taiwan. Formosa II is launched at Vandenberg of America on May 21 in 2004 [2]. The major mission of Formosa II is telemetry (image transmission). We can see the appearance of Formosa II from Fig. 2.1 and Fig. 2.2.

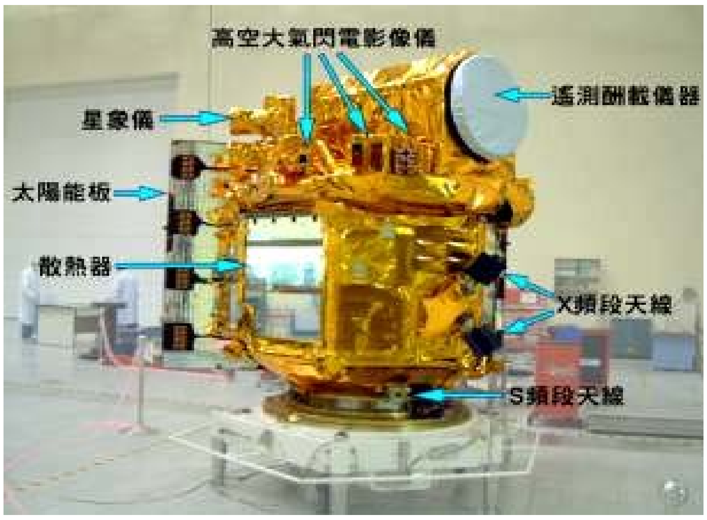

Fig 2.1. Appearance of Formosa II (1) [2]

Fig 2.2. Appearance of Formosa II (2) [2]

Although some parameter maybe list before, we still integrate some basic parameters in the Table. 1 for convenience and readable. There are some other basic properties of Formosa II lists as follows

Type	Science satellite
Orbit height	891 km
Height	2.4 m
Weight	760 kg
Period to encircle the earth	103 mins
Appear time above Taiwan	2 times. 1st about at am 9:40 2nd about at pm 9:40
Resolution for telemetry to the surface of earth	Black and white image: 2m Color image: 5m
Mission lifetime	5 years

Table. 1 Characteristic of Formosa II [2]

3. Receiver architecture

3.1 Overview

In this chapter, we will present the conventional LEO communication receiver with reference of Kongsberg MEOS capture system and other traditional basic design [6]. Some important components of receiver are introduced briefly in following sections. Although RF front end is not our main topic of this project, Section 3.2 still gives a basic idea about how RF front end works. Section 3.3 is our main focus in this project which concerns about the principle of baseband receiver design and more descriptions are given in this section. The concept of inner receiver and outer receiver is described in detail in [3]. Theoretical introduction will also cover this chapter.

3.2 RF front end

3.2.1 Architecture

There are three basic types in real life front-ends architecture. That are super-heterodyne, direct conversion and low IF architecture [4].

3.2.1.1 Super-heterodyne architecture

The feature of super-heterodyne architecture is that it down-convert received the RF signal to baseband signal with two times [4]. The advantage of this architecture is good selectivity for channel because it can produce high quality IF filter to reduce adjacent channel interference and free from DC offset. The disadvantage of super-heterodyne architecture may be the lager complexity than direct conversion architecture and less suitable for integration, due to the many chip connections to external lumped elements. The block diagram of super-heterodyne architecture is in Fig 3.1.

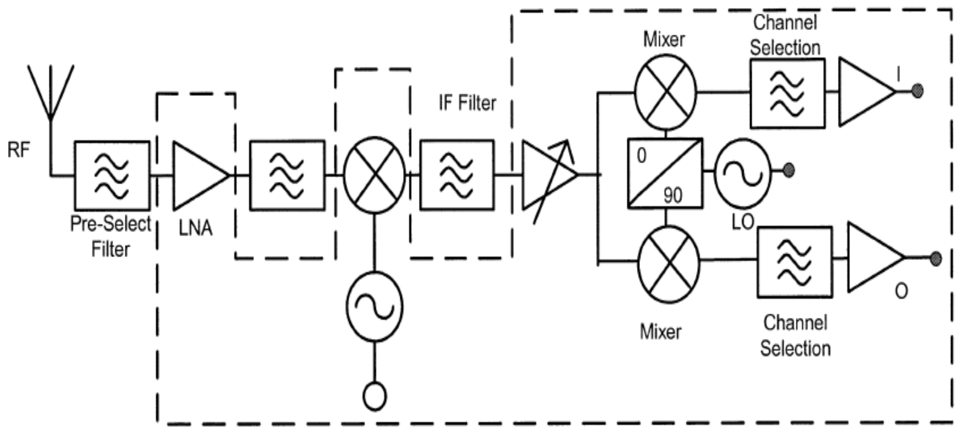

Fig 3.1.   Super-heterodyne architecture [5]

3.2.1.2 Direct conversion architecture

The feature of direct conversion architecture is that it down convert RF signal at one time [4]. The advantage of direct conversion architecture is that it greatly simplify the complexity by removing IF chain and single down-conversion section. Several hardware-driven tasks become software-driven, which make it easy for the way to software defined radio. The disadvantage of direct conversion architecture is that direct conversion receiver suffer from DC offsets, self mixing, LO leakage, flicker noise and IQ imbalance. At down-conversion section, its requirement is tougher than SHR case. The block diagram can be seen in Fig 3.2.

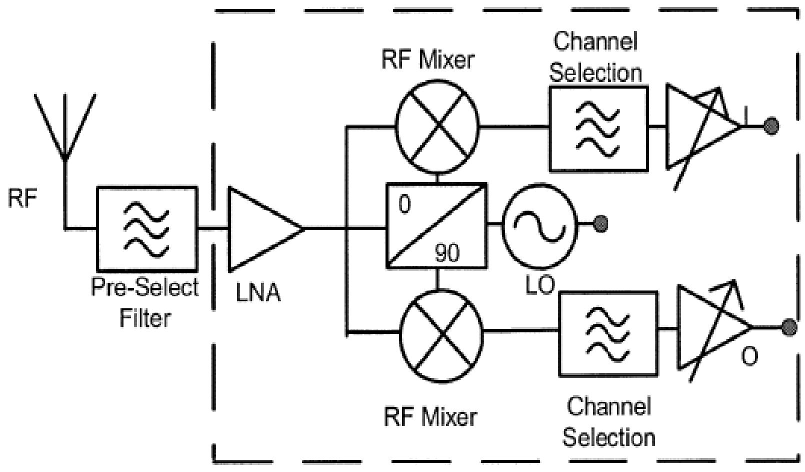

Fig 3.2. Direct conversion architecture [5]

3.2.1.3 Low IF architecture

The feature of low IF architecture is mostly like direct-conversion architecture except the down-convert signal is not DC signal but a low IF signal [4]. The complexity of low-IF architecture lies between the direct-conversion architecture and super- heterodyne architecture. This architecture reserve more advantage of direct conversion architecture but inject the image signal issue which is also existing in super- heterodyne architecture. The remedy is the same as super- heterodyne architecture that image-rejection filter is put in front of mixer. So the output of mixer may not be distorted by image signal. The block diagram is the same as Direct conversion architecture in Fig. 3.2. The only different portion is the output of mixer which is an intermediate frequency (IF) signal. In following section, we briefly introduce the function of some key components.

3.2.2 Pre-select filter [4]

In the Fig. 3.1 and Fig. 3.2, we can see that there is a pre-select filter after the transmitted band-pass signal is received from antenna. Pre-select filter is also a band-pass filter. It is an indispensable element to RF front end when we want to choose the desired frequency band signal. Much outer band noise can be filtered by this band-pass filter.

3.2.3 Low noise amplifier (LNA) [4]

When the band-pass signal passed through pre-select filter, the next stage is the low noise amplifier. It is another key function of RF front end circuit that amplitude of received signal can be enhanced and the noise of all the subsequent stages is reduced by the gain of the LNA. Thus, it is necessary for an LNA to increase the desired signal power with little noise and distortion so that the amplitude of this signal can be utilize in the later stages of the system.

3.2.4 Local Oscillator (LO)[4]

Local oscillator is the heart of all RF front end circuit. Without local oscillator, the high frequency signal can not be generated to down-convert the received RF signal. In general, the pure oscillator only supplies one high frequency signal with high precision. By cooperation with frequency synthesizer, LO can provide many different frequency carrier in order to perform multiple choices of channel selection.

3.2.5 Mixer [4]

In spite of which architecture is chosen, the mixer which performs the multiplication of two signals always exists in Fig. 3.1 and Fig. 3.2. The operation of mixer uses Prosthaphaeresis formula to perform signal down-conversion. In usual, a low pass filter with desired bandwidth is following the mixer in order to filter out the desired signal.

3.3 Baseband receiver

3.3.1 Overview

After RF front end processing, the received signal is down-converted to baseband. There are still many processes that are needed to proceed. Several compensations of non-ideal effect of channel are performed in baseband processing chain. According to [3], we can divide the baseband receiver into two categories. One is inner receiver whose function is to make channel effect at the output of inner receiver like the AWGN channel. Section 3.3.2 will introduce more detail about inner receiver. The other is outer receiver whose main function is to perform channel decoding and source decoding with respect to the signal from output of inner receiver. Section 3.3.3 will introduce the detail of outer receiver.

Kongsberg’s High Rate Demodulator (HRD) is our major reference for inner receiver introduction, and Front end processor (FEP) is our main reference for portion of outer receiver which does not include source decoder [6].

3.3.2 Inner receiver

In the Fig 3.3, the inner receiver structure is shown including AGC, RSSI, ADC, Carrier recovery, Timing recovery, Timing recovery, Equalizer and Channel estimation. All of the block will be explained in following section.

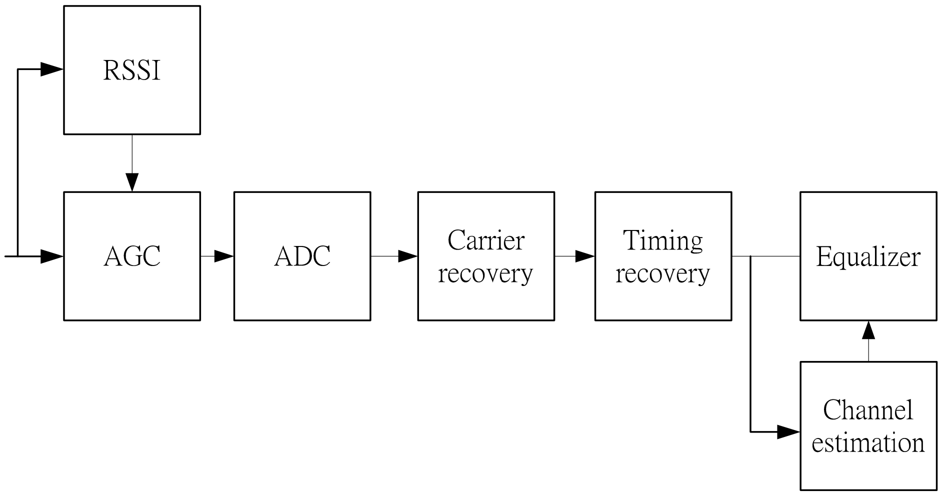

Fig 3.3. Inner receiver structure

3.3.2.1 Automatic gain control (AGC) [4],[6]

The reason why we need automatic control is that AGC can maintain the input dynamic range in certain level. What an important thing for AGC that it makes Analog to digital converter (ADC) design because the higher dynamic range is, the more difficult for ADC design. AGC can also help the received signal avoid fluctuation of amplitude which is good properties for phase-shift keying modulation.

3.3.2.2 Received signal strength indicator (RSSI) [4],[6]

The function of Received signal strength indicator (RSSI) is mainly to measure the total power in front of AGC. RSSI also provide an indication for AGC that AGC can depend on the indication to adjust the amplifier gain to satisfy the need in practice. When we detect the higher value of signal strength above some threshold, RSSI give information to AGC and it can adjust the amplifier gain smaller. In reverse, the lower value of signal strength is detected, AGC will adjust the amplifier gain larger.

3.3.2.3 Analog to Digital Converter (ADC) [7]

In recent years, with the progress of VLSI technique, the computation time of DSP processor or FPGA become more and more short. It is an important role for ADC to convert analog signal into digital signal and DSP processor can perform processing. Sampler and quantization is two key function of ADC. By determining sampling rate and quantization bits, analog signal could be transformed into more sampling values with determined bits to represent them by ADC.

3.3.2.4 **Matched filter** [7]

In communication theory, the real life communication channel is often band-limited and the signal will be distorted in time domain when passing through such band-limited channel. This situation will lead to inter-symbol interference (ISI)problem. Matched filter is a good way to overcome this problem. By putting pulse-shapping filter in transmitter side and matched filter in receiver side, we can reduce the ISI which caused by the effect of band-limited channel.

3.3.2.5 **Carrier synchronization** [3],[7]

Consider the received baseband signal x(n) as follows,

$$x(n) = e^{j(2\pi vn+\theta)} \sum_{k=1}^{L-1} h(k)a(n-k-\varepsilon) + w(n)$$

where v is the normalized carrier-frequency offset (CFO) and θ is the phase offset. h(k) means the channel taps corresponds to channel impulse response and L stands for total channel taps number. is the information signal which is usually represented by complex number if QPSK modulation is adopted. ε is the normalized timing offset with respect to modulation symbol time. w(n) is an AWGN channel that all RF front end thermal noise can be modeled by AWGN.

The goal of carrier synchronization is to recovery the carrier-frequency offset and phase offset. The phase offset is mainly caused by oscillator phase noise. The cause of CFO is owing to the relative speed between transmitter and receiver, and the mismatch of their local oscillators. Without this compensation, the system performance could degrade severely due to CFO and phase offset. Especially for LEO satellite, as result of high speed of motion (maybe several ten thousand km per hour), the Doppler shift is larger than any cellular communication system. Therefore, the CFO is the main issue to overcome in LEO satellite receiver design.

3.3.2.6 Timing synchronization [3],[7]

As seen in the upper equation, timing synchronization is try to estimate the timing offset ε and compensate the offset in order to make sure the signal is sampled at the best sampling time. That is to say, sample the received signal at the maximum eye open position. The reason why the timing synchronization is required is that the effects of propagation delay and sampling clock offset due to real life non-idealities make us have to do compensation for the timing error. Without this compensation, the input to the decision device could not be maximum signal to noise ratio (SNR) at that situation and performance will degrade.

3.3.2.7 Channel estimation [3],[7]

After synchronization procedure is over, the signal would enter into next stage and go to cancel out another impairment due to multipath channel propagation. The procedure should divide into two steps. The first step is trying to estimate the multipath channel L taps, that is h(k), k=0….L-1. The second step is to utilize the equalizer to compensate the effect which caused by each channel tap. The function of channel estimation is reproduction of multipath channel taps in some constant period which might be a frame period. In other words, there is an assumption of channel to be static in some constant period. Although this is not the truth in real life, this method is a good way in implementation issue.

3.3.2.8 Equalizer [3],[7]

The function of equalizer takes over the unfinished work of compensation of Channel effect. It performs compensation by designing the tap-delay line filter with the inverse channel response or channel state information (CSI) as the coefficients of filter. After this operation, the signal is going to the decision device to identify which information is transmitted and recovery them into bit sequence. Therefore, the input signal of outer receiver is always bit sequence.

3.3.3 Outer receiver

In this section, the introduction of outer receiver is given. The block diagram is shown in Fig 3.4. Including Diff decoder, Viterbi decoder, PCM decoder, Frame synchronization, Cyclic Redundancy Check, R-S decoder and Source decoder. Each component of outer receiver will explain in detail in the following.

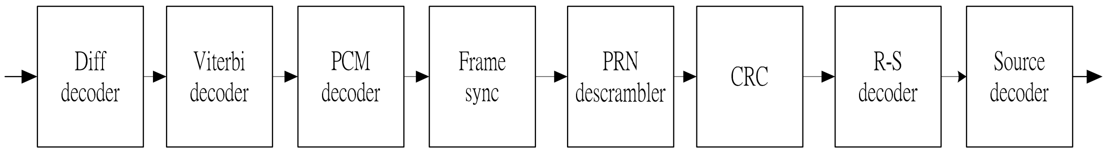

Fig 3.4. Outer receiver structure

3.3.3.1 Differential decoder [6],[7]

In digital communications, differential coding is a technique which used to give unambiguous signal reception when using some types of modulation. It makes the transmitted data depend not only on current bit (or symbol), but also from the previous one. This function block is optional in Kongsberg FEP. In particular, two bits are combined into a symbol and encode or decode in use of the previous symbol relationship with present symbol.

3.3.3.2 **Viterbi decoder** [7]

Viterbi dcoder use the famous Viterbi algorithm to decode any encoder which can be represented by finite state machine. A convolutional code encoder consists of several shift registers and XOR operations on the input and output from certain registers, as depicted in Fig. 3.5. A code generator is defined by its code rate and constraint length (defined as number of shift registers plus one). For simplicity, we consider code rate to be 1/2 only. As we have a code generator, then a series of encoded bits can be generated. In addition, the optimal choice of code generators for different constraint length and even code rate is provided in [7],[8].

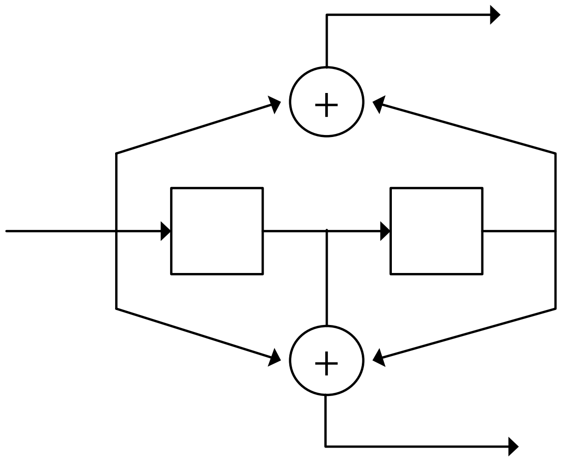

Fig. 3.5 Code rate 1/2 of CC Encoder with code generator

Convolution code is the well-known channel coding which can decode by this method. Although Viterbi algorithm has the more consumption of resource compared to other decoding method, it performs maximum likelihood (ML) decoding which have better performance than others The ML-based Viterbi algorithm is used for decoding. The decoder has a total of 2^(number of shift registers) states, see Fig. 3.6.

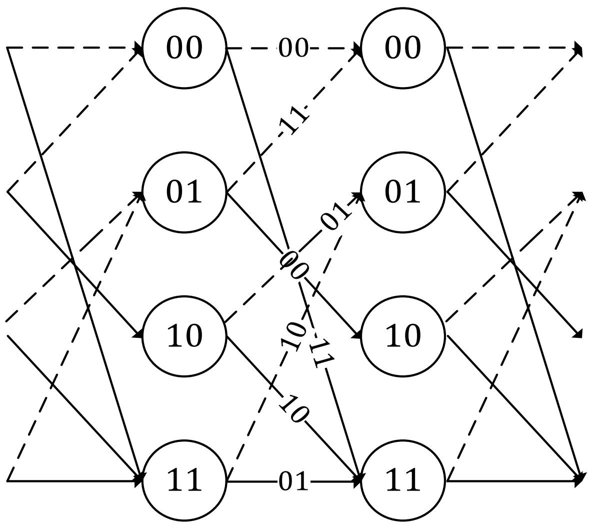

Fig 3.6. Decoder state diagram for Viterbi algorithm

On each state transition, it corresponds to a sequence of output values. Note that the output values are binary for hard decision, or with higher quantization for soft decision. As a codeword is received, then it is compared to each of the state transitions at each time interval. The hamming distance between the received codeword and transition outputs at each interval is recorded as a branch metric (BM). For each decoding path, it corresponds to a path metric (PM) which is the accumulation of BM along the path. In addition, at each time interval, PM of the paths which transit to a common state are added with current BMs respectively, compared, and the one with the smallest consequent PM is selected as the best. Note that at each time instant, only a path is stored for each state. Consequently, the path at the end with the smallest PM is determined as the decoded output. The more information can be find in [7],[8].

3.3.3.3 PCM decoder [6],[7]

Pulse code modulation decoder operates in one bit as an unit and it changes some other type line coding waveform into the only waveform of NRZ-L. The input of PCM decoder waveform can be NRZ-L, NRZ-M, NRZ-S and output waveform of PCM decoder is always NRZ-L. This function is also optional in Kongsberg FEP board. When we close this function, the input and output format should be the same. Another reason to use this function is to ensure the input format to frame synchronization identically.

3.3.3.4 Frame synchronization [3],[6],[9]

Since there are many sensors in satellite, each sensor would like transmitting the measurement signal. The more efficient way is to design a frame structure, and combine every sensor's measurement signal into a frame. Therefore, multiplexing technique is required in satellite communication system. How to de-multiplex is an important thing.

In order to extract information of each sensor, frame synchronization is necessary for our baseband receiver design. Wrong frame synchronization could make every individual sensor information incorrectly acquired by receiver and let all sequence of frame be scrambled. The algorithm of frame synchronization utilizes the pattern to recognize if the frame is correctly identified.

In Kongsberg FEP, the synchronization pattern is pre-determined according to Consultive Committee for Space Data System (CCSDS) standard [9]. It is shown as follows:

0x1acffc1d

Fig 3.7. Synchronization pattern in CCSDS standard [9]

We can see in Fig. 3.7 that a 32-bit synchronization pattern is represented by hexadecimal.

3.3.3.5 PRN Descrambler [6],[7],[9],[10]

Scrambler in transmitter is used to ensure that the data has sufficient bit transition density. It also can help timing recovery to capture the zero cross point and make the timing recovery circuit more simple. The function of descrambler is the same as scrambler. Just put input bit sequence to do exclusive-or with the sequence generated by pseudo random noise (PRN) generator. For example, the implementation of a generator polynomial

is shown in Fig 3.8.

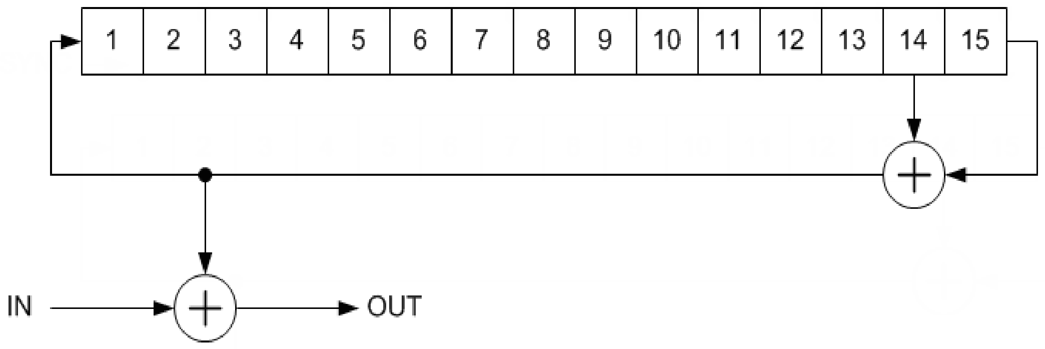

Fig 3.8. PRN descrambler generator

In Kongsberg FEP, the generator polynomial of descrambler is chose according to Consultive Committee for Space Data System (CCSDS) standard [6],[9]. It is shown as follows:

where means that there are nine registers in generator of PRN and every input bit performs exclusive-or with every output bit of PRN generator. The more detail is shown in Fig. 3.9. In Fig. 3.9, ASM means attached syn marker which is also named synchronization pattern..

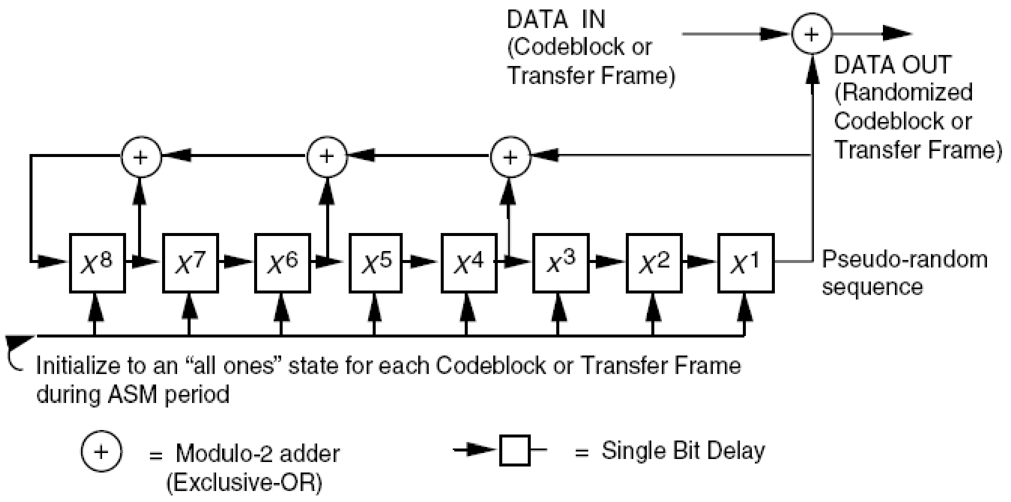

Fig. 3.9 PRN descrambler logic diagram

3.3.3.6 Cyclic Redundancy Checker (CRC) [6],[7],[8],[9]

The principle is that the CRC generator polynomial divide data field in finite field operation. The remainder of this operation is called CRC checksum. On the transmitter side, there is a calculated CRC checksum appended to data end of frame.

After pass through the channel, the CRC checker performs division operation again with frame including data and checksum. If the remainder is not zero, we infer that there must be something wrong with this frame that we can

In Kongsberg FEP, the generator polynomial of descrambler is chose according to Consultive Committee for Space Data System (CCSDS) standard [9]. It is shown as follows:

A 16-bit cyclic redundancy checksum is calculated by g(x) in finite field GF(2) operation. The CRC checker cannot correct errors.

3.3.3.7 Overview of Reed-Solomon decoder

The use of error-correction coding to eliminate the necessity of retransmission of the data is called forward error correction (FEC). The basic concept is to systematically add redundancy to messages at the encoder such that the decoder can successfully recover the messages from the received block possibly corrupted by channel noise. The use of FEC in optical networks was pioneered for submarine systems where detection and correction of errors was critical for transmission over very long haul networks. Out of many error correction codes, Reed–Solomon (RS) codes have been widely used in a variety of communication systems such as space communication link, digital subscriber loops, and wireless systems as well as in networking communications [10].

RS decoders can be used to protect digital data against errors occurred and reduce the signal to noise ratio in the transmission process. The RS decoder can be implemented using the modified Euclidean (ME) algorithm or Berlekamp–Massey (BM) algorithm to solve a key equation. For either algorithm, a finite-field (also called Galois-field) is a mathematical structure which plays a crucial role in the theory of RS codes and the finite-field arithmetic operations are the fundamental building blocks for the RS decoder [11], [12]. Knowledge of basic finite-field concepts and Reed–Solomon encoding algorithm was well introduced in [10]–[12]. If no erasure is taken into consideration, a syndrome-based RS decoder consists of three components. First part is a syndrome computation (SC) block. It generates a syndrome polynomial S(x) that is used in the key-equation solver (KES) block for solving a key equation $s(x)\sigma(x) = w(x) \bmod x^{2t}$. Either the ME algorithm or BM algorithm can be used to solve a key equation for an error-locator polynomial $\sigma(x)$ and an error-value polynomial w(x). Then in the third component, these two polynomials are used to find out the error locations and the corresponding error values according to the Chien search and Forney algorithm and corrects the errors as the received word is being read out of the decoder. In addition, a first-in first-out (FIFO) memory is used in order to buffer the received symbols according to the latency of these components. The very high-speed data transmission techniques that have developed for fiber optical networking systems have necessitated the implementation of high-speed RS decoder architecture to meet continual demand for ever higher data rates. With the advent of dense wavelength division multiplexing (DWDM), the capacity of optical transmission systems has rapidly increased over the past ten years. In particular, the RS(255, 239) code is now commonly used in high-speed (10 Gb/s and beyond) fiber optic systems such as DWDM systems due to its 8-B error-correcting capability. The throughput bottleneck in an RS decoder is in the KES block, but the other blocks of the decoder are simple to pipeline due to their feedforward structure. Thus, we propose a highly pipelined ME algorithm block that improves the clock frequency. A key advantage of RS decoders based upon the ME algorithm is their regular data-flow structure that can be easily pipelined and implemented in very large scale integration (VLSI) [19]–[23].

Several parallel architectures used for Viterbi decoding, CRC codes, and BCH codes [13]–[15] have been reported in order to obtain processing rates higher than the clock frequency. Such an approach is an effective method of speeding up more than 10-Gb/s data-processing rate, since the maximum clock frequency is generally bounded by the physical constraints of the circuit.

3.3.3.8 Syndrome Computation

Let C(x)and R(x) be the codeword polynomial and the received polynomial, respectively. The transmitted polynomial can be corrupted by channel noise during transmission. Therefore, the received polynomial can be described as $R(x) = C(x) + E(x) = R_{n-1}x^{n-1} + \ldots + R_1 x + R_0$, where E(x) is the error polynomial (where is the maximum number of errors that can be corrected in the RS code). The first step in the decoding algorithm is to calculate 2t syndromes, $S_i, 0 \le i \le 2t-1$, which are used to correct the correctable errors. If all 2t syndromes $S_i (0 \le i \le 2t-1)$ are zero, then the received polynomial R(x) is a valid codeword C(x) with no transmission errors.

The syndrome polynomial S(x) is defined as $S(x) = S_0 + S_1 x + \ldots + S_{2t-1}x^{2t-1} = \sum_{i=0}^{2t-1} S_i x^i$, with $S_i = \sum_{j=0}^{n-1} r_j \alpha^{ij}$,where α is a root of a primitive polynomial

$p(x) = x^8 + x^4 + x^3 + x^2 + 1$ and t=8, which is a primitive element in $GF(2^8)$. For RS(255 239) code, $\alpha^i (0 \le i \le 254)$ denotes the possible error locations. We considers the symbol values as being polynomial coefficients and determines if the series of symbols contained in a data block form a valid codeword for the particular RS code chosen. The partial syndrome is multiplied by at each cycle and accumulates with the received symbol.

3.3.3.9 Modified Euclidean algorithm

The ME algorithm is used to obtain the error locator polynomial $\sigma(x)$ and the error value polynomial $\sigma(x)$ by solving the $s(x)\sigma(x) = w(x)$ mod x^{2t} key equation . The algorithm is described as follows:

Input: $S(x), x^{2t}$

Initialization:

$$R_0(x) = x^{2t}, Q_0(x) = S(x), L_0(x) = 0, U_0(x) = 1;$$
$$\deg(R_0(x)) = 2t, \deg(Q_0(x)) = 2t - 1;$$
$$l_0 = \deg(R_0(x)) - \deg(Q_0(x));$$

Index ‘i ’ is initialized to 0;

```
Index 'Step ' is initialized to 1;

Start Algorithm:

    while (Step<2t+1)do

        begin

            $Step \leftarrow Step + 1;$
            $i \leftarrow i + 1;$
            $a_{i-1} \leftarrow leading\ coefficient\ of\ R_{i-1}(x)$
            $b_{i-1} \leftarrow leading\ coefficient\ of\ Q_{i-1}(x)$

If ( $\deg(R_i(x)) < t$ )

            begin

                $R_i(x) = R_i(x)$
                $Q_i(x) = Q_i(x)$
                $L_i(x) = L_i(x)$
                $U_i(x) = U_i(x)$

Skip the following statements & stop the algorithm

end

if( $l_{i-1} \geq 0$ )

          begin

            $R_i(x) = [b_{i-1}R_{i-1}(x)] - x^{|l_i - 1|}[a_{i-1}Q_{i-1}(x)]$
            $Q_i(x) = Q_{i-1}(x)$
            $L_i(x) = [b_{i-1}L_{i-1}(x)] - x^{|l_i - 1|}[a_{i-1}U_{i-1}(x)]$
            $U_i(x) = U_{i-1}(x)$
```

```
                end

            else

                begin

    $R_i(x) = [a_{i-1}Q_{i-1}(x)] - x^{|l_i-1|}[b_{i-1}R_{i-1}(x)]$
    $Q_i(x) = R_{i-1}(x)$
    $L_i(x) = [a_{i-1}U_{i-1}(x)] - x^{|l_i-1|}[b_{i-1}L_{i-1}(x)]$
    $U_i(x) = L_{i-1}(x)$

    end

    $l_{i-1} \leftarrow \deg(R_{i-1}(x)) - \deg(Q_{i-1}(x))$

    end

    Output: $\sigma(x), w(x)$
```

In the ith iteration, a_{i-1} and b_{i-1} are the leading coefficients of $R_{i-1}(x)$ and $Q_{i-1}(x)$, respectively. The algorithm stops when $\deg(R_i(x)) < t$, where $\deg(\bullet)$ denotes the degree of a polynomial.

3.3.3.10 Chien Search and Forney Algorithm

The error locator polynomial and the error value polynomial $w(x)$ are fed into the Chien search block, which calculates the roots of the error locator polynomial. The Forney algorithm block works in parallel with the Chien search block to calculate the magnitude of the error symbol at each error location. Let the error locator polynomial of the degree over $GF(2^m)$ be defined by,

$\sigma(x) = \sigma_t x^t + \sigma_{t-1} x^{t-1} + \ldots + \sigma_0$, where the coefficients $\sigma_i \in GF(2^m)$ for $0 \le i \le t-1$. It is well known that Chien search algorithm [11] can be used to determine the roots of an error locator polynomial of degree in $GF(2^m)$, where is the maximum number of errors that can be corrected

in the RS code. Fig. 3 shows the Chien search block, the Forney algorithm and the error correction blocks, which generate the error value and then the corrected symbol. For division of the Galois-field, first of all, the inverse element of the divisor is derived, and it is then multiplied with the element of the dividend by the pipelined fully-parallel multiplier. A straightforward approach for computation of the inverse of a nonzero element in $GF(2^8)$ is to use a simple look-up table composed of 255 words of 8-bits, in which inverse of the field elements are stored. Consequently, it can be realized by means of a static ROM, which gives a path delay less than that of pipelined multiplier.

In the final step, each error value is simply added (XORing in binary) to the received symbol fetched from a FIFO to produce the corrected symbol. At the locations where there are no errors, the error values are zero and the received polynomial is not changed at those locations when added.

3.3.3.11 FIFO Memory Buffers and Control Logic

As each error value is calculated, the corresponding received symbol is fetched from a FIFO memory, which buffers the receivedsymbols during the decoding process. Each error value is simply added to the received symbol to produce a corrected symbol. At the locations where no errors have occurred, the error values are zero and there is no change in the received polynomial at those locations when added.

Since the received data coming into the RS decoder is continuous, elaborate sequences of controllers are required to generate control signals for each step of the decoding. The design of the controller is carried out by implementing local slave controllers for each component with special handshake protocols between two successive components through the master controller.

3.3.3.12 Reed-Solomon Hardware scheme

We have simulated Matlab or C programming of CCSDS standard RS code

(255,223) with error correcting capability t=16 and convolutional code (2,1,7) with constraint length 7. The hardware description has programmed in Verilog. The hardware implementation of RS decoder is illustrated in Fig 3.10 as follows.

5 states for the finite state machine are shown in fig.3, which are IDLE, WR0, SYN0, MEU0 and CHECK_END. The state transition control pin are summarized. At fig.2, the timing schedule for these 5 states illustrates the main control pin waveform. We directly explain the function of these main control pin. RSADA0 represent the input RS codeword from last stage. RSWRA0 activate high for the SRAM writing.

RSRDA0 activate high for the SRAM reading. The syn_chien_mux pin activates low for the data going to Syndrom calculation block and activates high for the data going to Chien search block. The syn_go, meu_go and chien_go activate high for Syndrom calculation block, Modified Euclidean block and Chien search &Forney block.

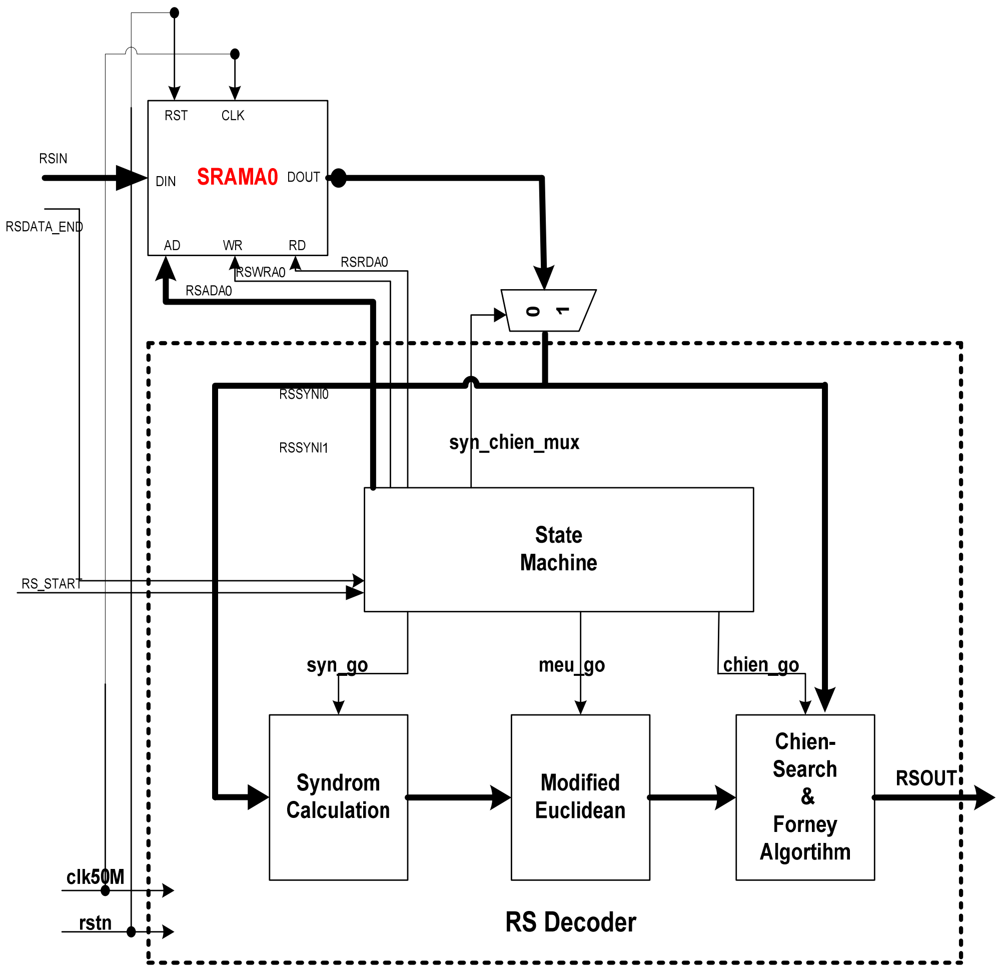

Fig. 3.10 Block diagram of R-S decoder scheme

Each SRAM in this design is required 64×32 bits to store receiving decoding bit from previous stage, but TriMatrix memory feature for Stratix II set the memory size and we choose 128×32. For buffering data, RSIN pin are 16bits per clock cycle. When writing each address, we write the MSB 16 bits for first clock and then LSB 16bit for next clock. Timing chart and finite state machine are illustrated at Fig. 3.11 and Fig. 3.12.

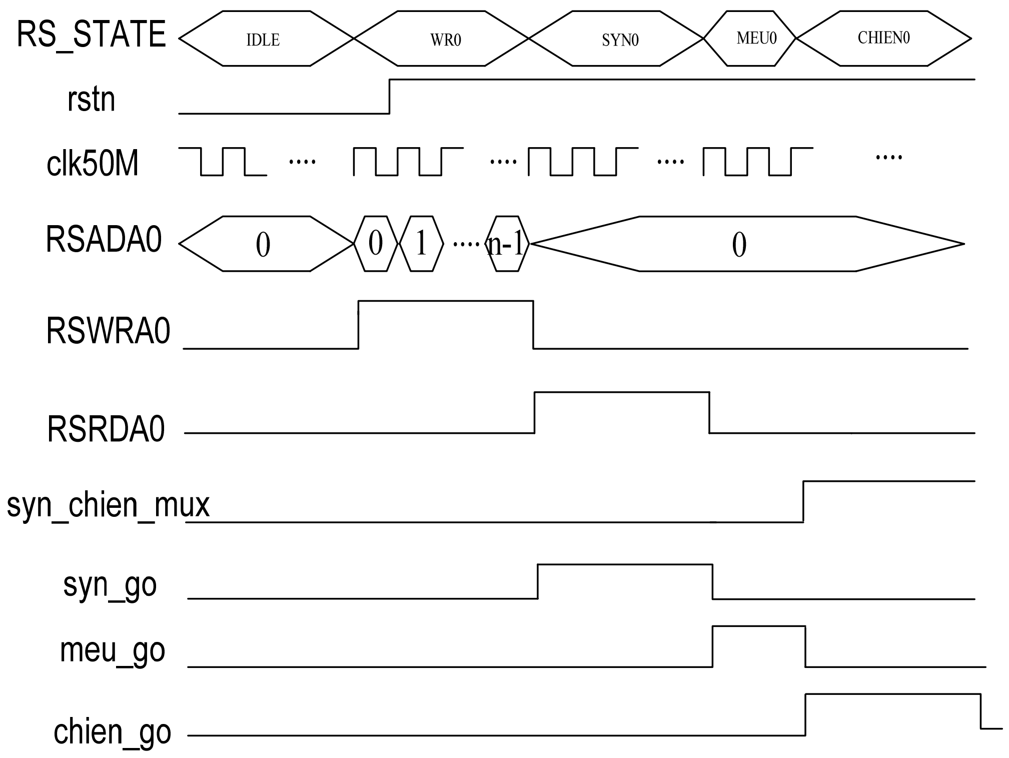

Fig. 3-11 Timing chart of RS decoder scheme

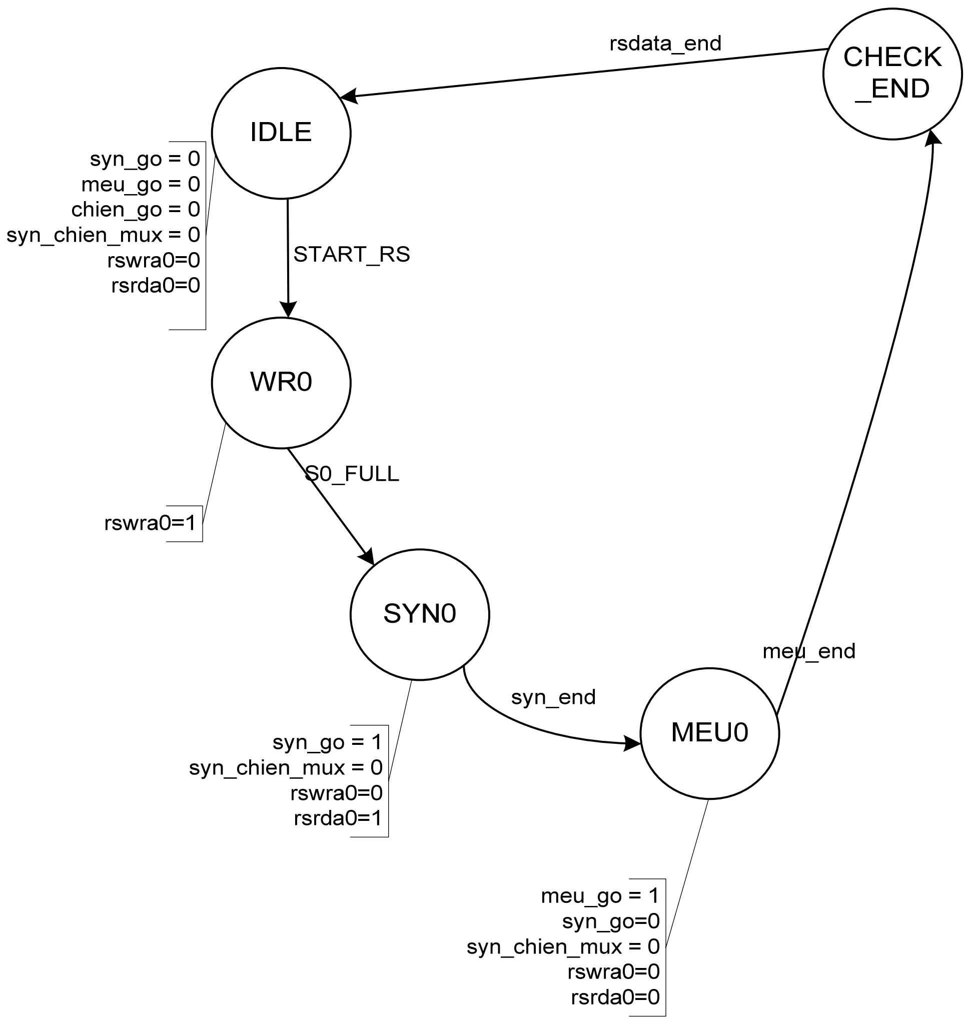

Fig. 3.12 Finite state machine of RS decoder scheme

The parallel structure for syndrome block is illustrated at Fig. 3.13. We set 16 bits for each clock input and process $\left\lfloor \frac{255}{16} \right\rfloor$ clock cycles to obtain syndrome of a block RS codeword. The parallel syndrome generator and Chien search unit is illustrated and depict to process 16 bits per clock cycle, only $\left\lfloor \frac{255}{16} \right\rfloor$ clock cycle are sufficient.

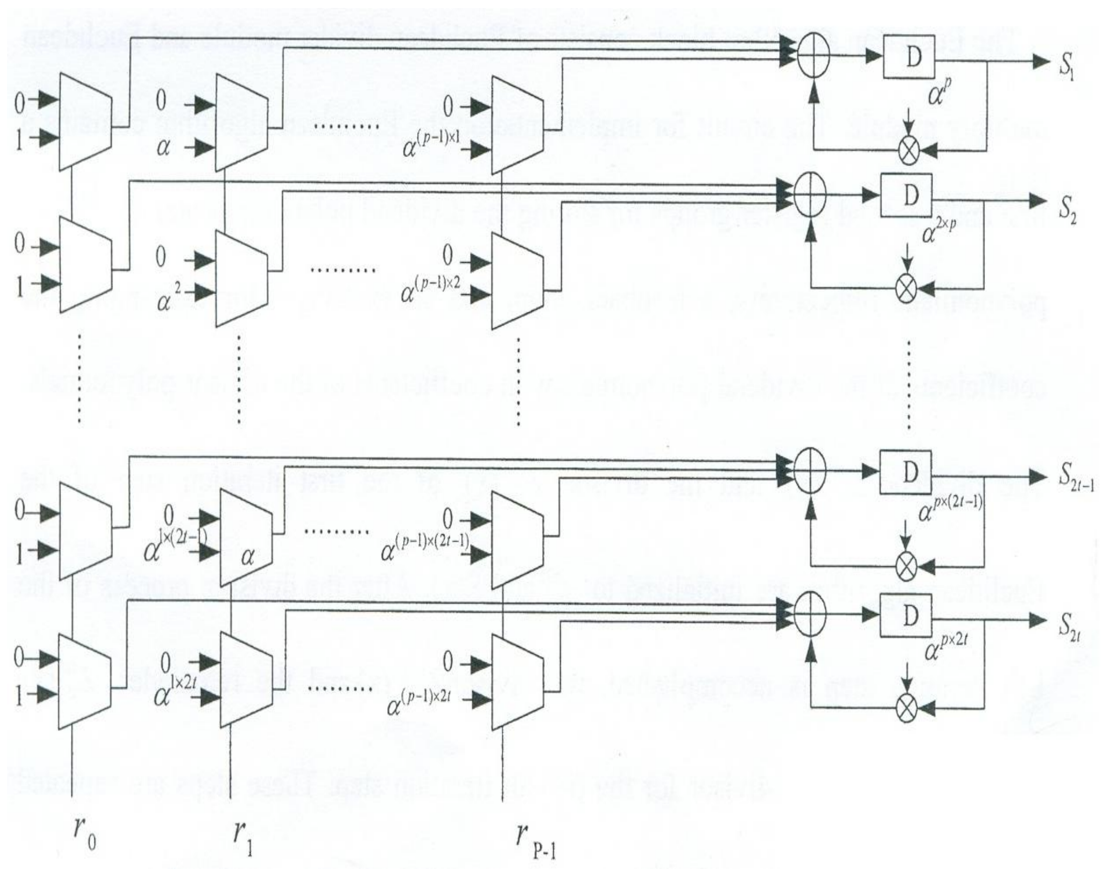

Fig. 3.13 Parallel syndrome generator with parallel factor p=16

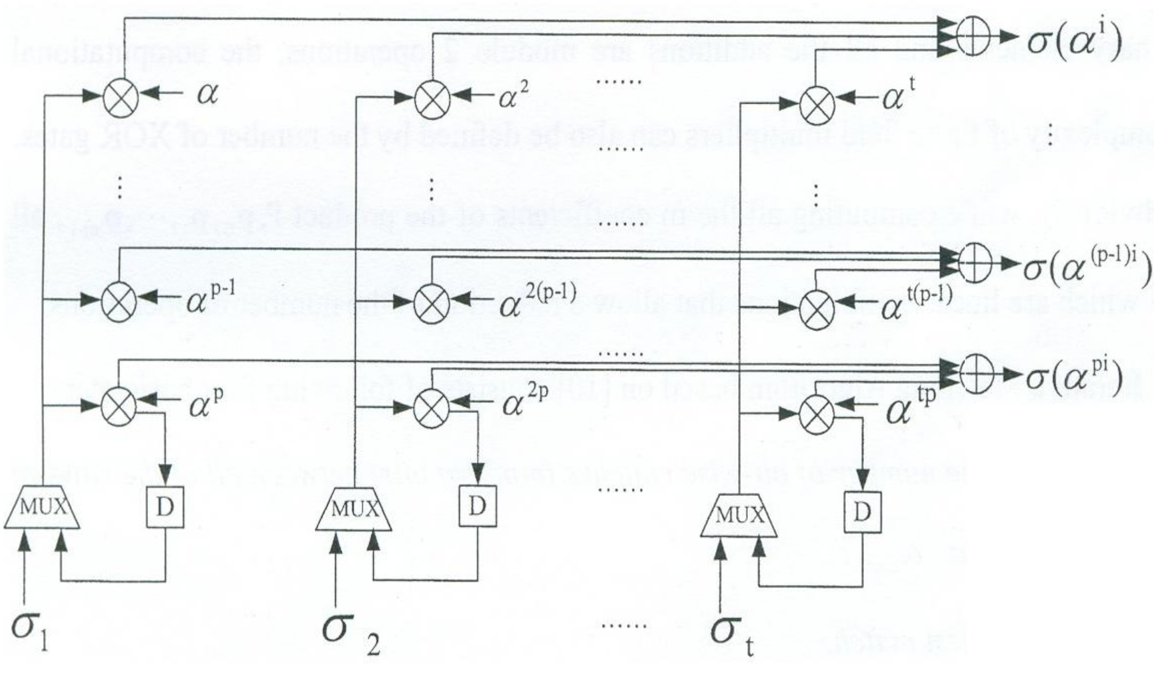

Fig.3.14 Parallel Chien search with parallel factor p=16

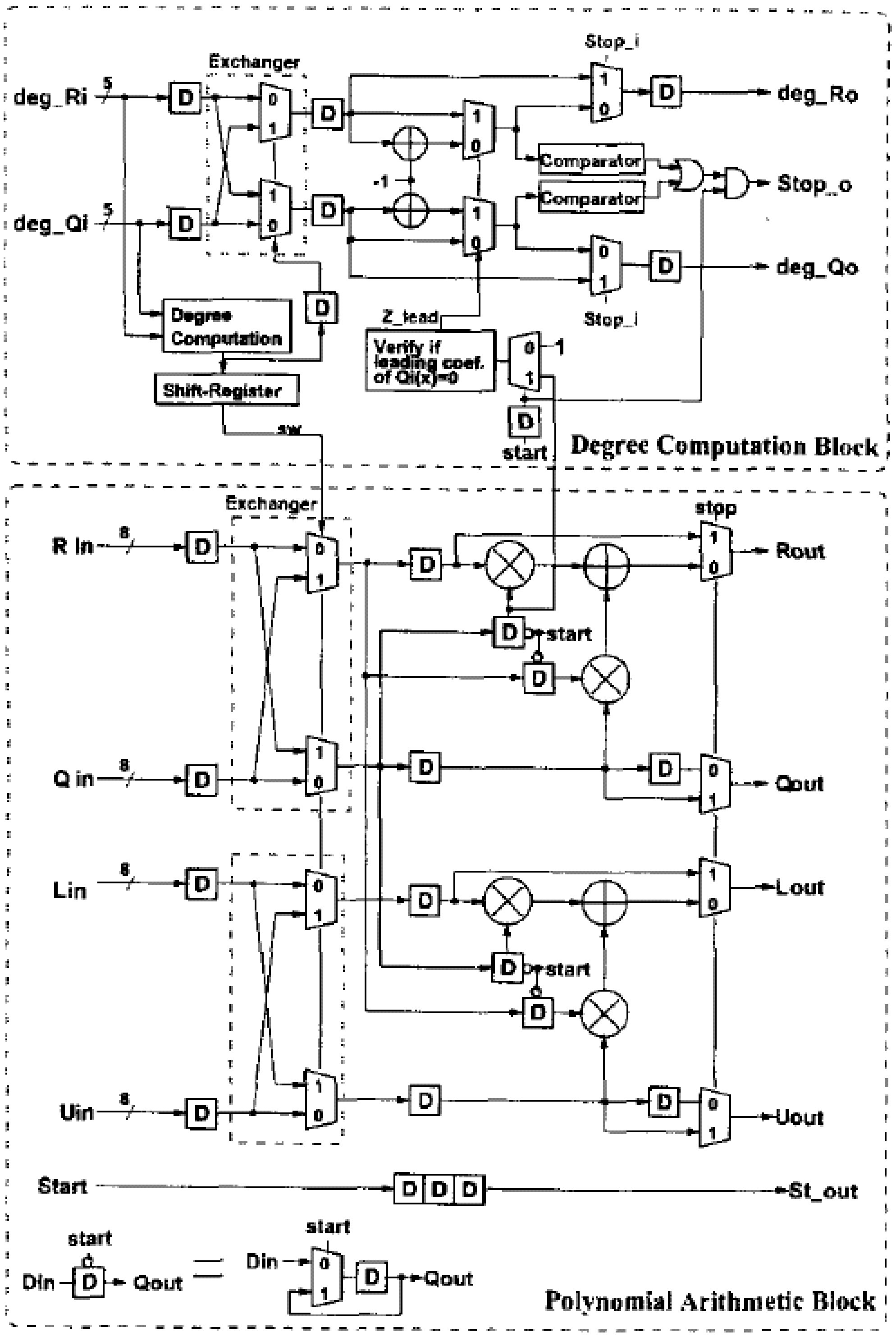

Fig. 3.15 Modified Euclidean algorithm processing

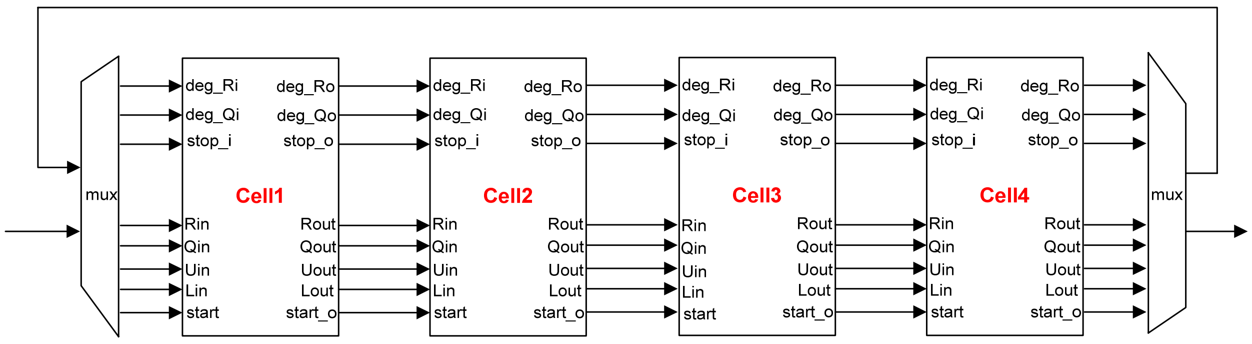

Fig.3.16 Modified Euclidean algorithm block

Fig. 3.15 shows a single cell in fig. 3.16. At fig. 3.15, the pipeline design for three registers would be able to efficiently improve the shortest critical path, which solves the problem of overall clock frequency. However, the pavement for this cell require three clocks, that is the total latency for modified Euclidean block are 2t*3 clock cycle.

Most complicated block of RS decoder is modified Euclidean algorithm block. We require 2t iterations to compute the key equation, so the data that processing from cell1 to cell4 require to feedback from the end of cell4 and feed into mux selector until 8th feedback is meet. We would follow the above hardware architecture to implement RS decoder. We compute the ideal overall throughput for this design as follows: Input buffering data require 16 cycles, syndrome unit require 16 cycle, key equation solving unit require 2t*3=96 cycle and the last stage is Chien search and Forney algorithm require 32 cycle. The overall clock cycles are 160cycles if we operate at 50MHz.

The throughput is ideally 223*8bit/160cycle*50*10^8=557.5Mbps. Although we probably suffer from any kinds of retiming solution to meet the clock frequency, the latency would be more than this ideal calculation. This design may seems to meet our target 150Mbps

The I/O pins are summarized in Table 2 as follows:

.Signal	**I/O**	**Description**
CLK50Mhz	I	Input clock 50 MHz from Clock Gen
rstn	I	Reset (*active low*) for clock generation
RS_START	I	RS decoder starting
RSDATA_END	I	No data to RS decoder, So, RS decoder ending
RSSYNI0[N-1:0]	I	Input date from SRAMA0
RSADA0[M-1:0]	O	Address for SRAMA0
RSWRA0	O	Write enable for SRAMA0
RSRDA0	O	Read enable for SRAMA0
ERRADD1[7:0]	O	Error address for 1st Error
ERRADD2[7:0]	O	Error address for 2st Error
ERRADD3[7:0]	O	Error address for 3st Error

ERRADD4[7:0]	O	Error address for 4st Error
ERRADD5[7:0]	O	Error address for 5st Error
ERRADD6[7:0]	O	Error address for 6st Error
ERRADD7[7:0]	O	Error address for 7st Error
ERRADD8[7:0]	O	Error address for 8st Error
ERRADD9[7:0]	O	Error address for 9st Error
ERRADD10[7:0]	O	Error address for 10st Error
ERRADD11[7:0]	O	Error address for 11st Error
ERRADD12[7:0]	O	Error address for 12st Error
ERRADD13[7:0]	O	Error address for 13st Error
ERRADD14[7:0]	O	Error address for 14st Error
ERRADD15[7:0]	O	Error address for 15st Error
ERRADD16[7:0]	O	Error address for 16st Error
SERR[3:0]	O	Corrected error amount

Table 2.   I/O pins of R-S decoder summary

3.3.3.11 Source decoder [7]

When the signal has passed through inner receiver and outer receiver, the remaining work is to do source decoder. In information theory, in order to reduce transmitting information, source coding can compress data in lossless or loss way. Source coding is also a data compression technique. The work of source decoder is to decompress the data and the raw data after decompression could send to upper layer for further application.

4. Front End Processer and replacement procedure:

4.3Front End Processer (IO21000 board)

The Front End Processer is implemented by the KONGSBERG corporation product IO21000 board. The IO21000 board is a software defined board based on fully FPGA. This board is showed in Fig. 4.1. Based on the entity of the IO21000 board, we observe some components of the FPGA board:

I. INTEL FW80321M600: CPU

II. xilinx XC2C256: Power saving

III. xilinx XCF32P VOG48: Flash memory (PROM)

IV. 72V36104: BiFIFO Buffer

V. 72T18125 (TeraSync (up to 9 Mb)): FIFO Buffer

VI. JS28F256: Numonyx StrataFlash Embedded Memory

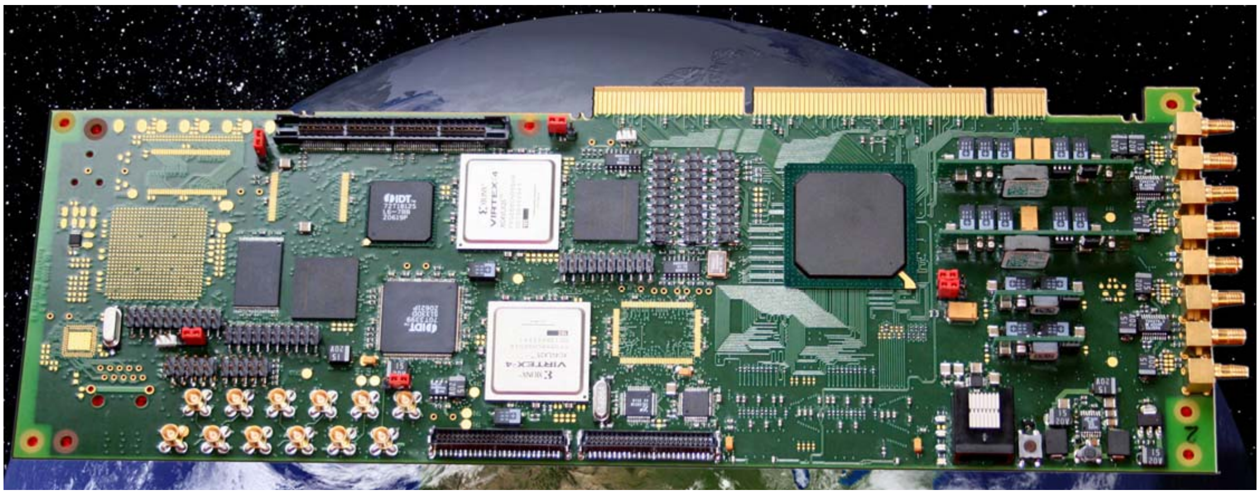

Fig. 4.1 Kongsberg IO 21000 board [6]

4.4 Interface of IO21000 board

The Interfaces are consists of two parts of I/O:

(i) The PCI interface: Input the data from demodulator (High rate Demodulator (HDR))

(ii) The SMA connector: Input data x 2, Output data x 2, and I/O clock x2 for two channels (i.e., total SMA x 8).

The PCI interface is the input for ingesting the data from demodulator. The Input and output of SMA are designed for (a) testing purpose (b) external data capture.

5. Verification:

Since the IO21000 board is a customized fully programmable FPGA board, if we want to copy this FPGA board, then we need full information of the board such as the source code or blueprint because we cannot obtain the information from the hardware only. Thus, if we do not have such Information, we can only “replace” it but not “copy” it. It means we may have different circuit structure, but have consistent function and I/O data of the IO21000 board.

In order to do that, the following procedure can be used for the verification:

I. Propose (or modify) the algorithm of each function of the IO21000 board.

II. Verify each function locally

III. Integrate all functions

IV. Verify the I/O signals by comparing to the IO21000 board. If the signals are correct, it implies our proposed algorithms are correct and hence we are done. However, if not, go to step I.

For replacing the IO21000, we should first propose the whole functions of the IO21000, including differential decoder, Viterbi decoder, PCM decoder, frame synchronization, time and quality status appending, PRN descrambler, CRC checker, and RS decoder. Then we verify each function locally. When each function is correct, we can integrate all of those functions into our developing board. And then we can make sure whether the replacement is success by comparing the I/O signals, from the IO21000 and our developing board. However, there are some difficulties of replacing IO21000 board we should note: (1) GUI and (2) PCI data input. Since this IO21000 provides the GUI for parameters setting and operational status information, we should consider the GUI for our replacement which increases the difficulty. The other difficulty is the PCI interface. Since the demodulated data is ingested form PCI, we should carefully identify the input pins such that the I/O pins are consistent with the IO21000.

6. Deliverables

From the deliverable of our proposal during this project, we will not complete the whole verification for replacing IO21000 board, but we will deliver three functions of the IO21000 board which include:

(i) Frame synchronization

(ii) PRN descrambler

(iii) RS decoder

where (i) and (ii) will be verified by software simulation (providing MATLAB code), and (iii) will be verified by hardware (FPGA) during this project. It implies we only verify the deliverable function locally according to our proposed verification procedure. However, those deliverables can be extended for future study by the proposed verification procedure until the replacement is success so that we can develop the self-control key technology for front end processer.

預定進度甘特圖(Gantt Chart)：以為進度控制及檢討之依據。

月次 工作項目	第一月	第二月	第三月	第四月	第五月	第六月	**第七月**	**第八月**	**第九月**	**第十月**	**第十一月**	**第十二月**	**備註**
System Spec. definition,	X	X	X	X									
Channel modeling	X	X	X										
Interface Identification			X	X	X	X	**X**	**X**					
RS Decoder				X	X	X	**X**	**X**					
Synchronization & de-scrambling				X	X	X	**X**	**X**	**X**				
MATLAB Codes/FPGA Codes					X	X	**X**	**X**	**X**	**X**			
Code integration &Simulations								**X**	**X**	**X**	**X**		
Final Report												**X**	
預定進度(累積數)	10%	20%	28%	35%	45%	50%	60%	70%	75%	85%	90%	100%	

Reference

[1] Elbert, Bruce R, Introduction to satellite communication, Artech House, Boston, 2008

[2] NSPO , http://www.nspo.org.tw/2008c/

[3] H, Myer, Digital communication receivers, 1998.

[4] B, Razavi, RF Microelectronics, Prentice Hall, 1998

[5] A. Loke and F. Ali, "Direct conversion radio for digital mobile phones—Design issues, status and trends," *IEEE Trans. Microwave Theory Tech.*, vol. 50, pp. 2422-2435, Nov. 2002.

[6] Kongsberg Spacetec AS, online source, http://www.spacetec.no/

[7] J. G. Proakis, Digital communications, 4th edition, McGraw-Hill, 2001

[8] S. Lin and D. J. Costello, Jr., Error Control Coding: Fundamentals and Applications, Englewood Cliffs, NJ: Prentice-Hall, 1983.

[9] CCSDS 131.0-B-1, blue book, "TM SYNCHRONIZATION AND CHANNEL CODING ", September, 2003.

[10] S. B. Wicker and V. K. Bhargava, *Reed–Solomon Codes and Their Applications*. Piscataway, NJ: IEEE Press, 1994.

[11] S. Lin, *Error Control Coding: Fundamentals and Applications*. Englewood Cliffs, NJ:

Prentice-Hall, 1983.

[12] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*. Englewood Cliffs, NJ: Prentice-Hall, 1995.

[13] G. Fettweis and H. Meyr, “High-speed parallel Viterbi decoding: Algorithm and VLSI-architecture,” *IEEE Commun. Mag.*, pp. 46–55, May 1991.

[14] T. B. Pei and C. Zukowski, “High-speed parallel CRC circuits in VLSI,” *IEEE Trans. Commun.*, vol. 40, pp. 653–657, Apr. 1992.

[15] T. K. Matsushima, T. Matsushima, and S. Hirasawa, “Parallel encoder and decoder architecture for cyclic codes,” *IEICE Trans. Fundamentals*, vol. E79-A, no. 9, pp. 1313–1323, Sept. 1996.

[16] ITU-T Recommendation G.975, “Forward Error Correction for Submarine Systems,” ITU, Geneva, Switzerland, Oct. 2000.

[17] R. T. Chien, “Cyclic decoding procedures for Bose–Chaudhuri–Hocquenghem codes,” *IEEE Trans. Inform. Theory*, vol. IT-10, pp. 357–363,1964.

[18] P. Tong, “A 40-MHz encoder-decoder chip generated by a Reed–Solomon code compiler,” in *Proc. EEE 1990 Custom Integrated Circuits Conf.*, May 1990, pp. 13.5.1–13.5.4.

[19] H. M. Shao, T. K. Truong, L. J. Deutsch, J. H. Yuen, and I. S. Reed, “A VLSI design

of a pipeline Reed–Solomon decoder," *IEEE Trans. Comput.*, vol. C-34, pp. 393–403, May 1985.

[20] H. M. Shao and I. S. Reed, "On the VLSI design of a pipeline Reed–Solomon decoder using systolic arrays," *IEEE Trans. Comput.*, vol. 37, pp. 1273–1280, Oct. 1988.

[21] W. Wilhelm, "A new scalable VLSI architecture for Reed–Solomon decoders," *IEEE J. Solid-State Circuits*, vol. 34, pp. 388–396, Mar. 1999.

[22] H. Lee, M. L. Yu, and L. Song, "VLSI design of Reed–Solomon decoder architectures," *IEEE Int. Symp. Circuits Syst.*, vol. 5, pp. 705–708, May 2000.

[23] H. Lee, "A VLSI design of a high-speed Reed–Solomon decoder," in *Proc. Int. ASIC/SOC Conf.*, Sept. 2001, pp. 316–320.

[24] S. R. Whitaker, J. A. Canaris, and K. B. Cameron, "Reed–Solomon VLSI Codec for advanced television," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 1, pp. 230–236, June 1991.

[25] S. Kwon and H. Shin, "An area-efficient VLSI architecture of a Reed–Solomon decoder/encoder for digital VCRs," *IEEE Trans.Consumer Electron.*, vol. 43, pp. 1019–1027, Nov. 1997.