



Fraunhofer Institut
Integrierte Schaltungen

***CorePool* FHG_RS_DEC**

Preliminary Databook

- subject to change without notice -

Version: v0.9

Date: 06.08.02

Document History

Table 1: Document History

Version	Date	Responsible	Description
v0.9	07.01.00	Spt	Generic datasheet

Contact

CorePool
Fraunhofer Institute Integrated Circuits

Am Wolfsmantel 33
91058 Erlangen
Germany

Phone: +49 (0) 9131 776 777
Fax: +49 (0) 9131 776 499
Email: info@corepool.com
Internet: <http://www.corepool.com>

Table of Contents

Document History	2
Contact	2
Purpose	7
Features	7
Design Kit	7
Requirements	7
References	8
Block Diagram	8
Signal Description	9
General Information	9
Parameter Description	10
Operating speed	11
Notes	12
Timing Diagramms	13
Waveform of external read and write access of the RAMs	14
Description of the decoding algorithm	16
Application notes	18
Example 1	18
Example 2	19

List of Tables and Figures

Block Diagram	8
Reed-Solomon decoder with two RAMs	12
Initialization	13
Input synchronization	13
Output synchronization	14
Write and read cycles of the two RAMs	14
Output waveform	15
Reed-Solomon decoder basic structure	16

Purpose

The FHG_RS_DEC is a parametrizable and synthesizable Reed-Solomon decoder. Its parameter set enables the designer to implement Reed-Solomon decoders for any codeword length and any error correcting capability, and thus provides versatility for the design of corresponding applications. The Reed-Solomon decoder uses (N, K) Reed-Solomon codes for block error correction, being the number of correctable symbol errors within one block: $t = \frac{N-K}{2}$.

Features

- Fully parametrizable Reed-Solomon decoder
- Fully synthesizable
- Codeword length and error correcting capability parameterized
- Generator polynomial parameterized
- Registered inputs and outputs
- Up to 3.94 MSymbols/sec decoding rate (code (255, 223))
- Silicon proven in AMS 0.6 μ technology
symbol length = 8, codeword length = 255,
number of information symbols = 223, error correction capability = 16

Design Kit

- Technology Independent Implementation as Synopsys Design Ware Components
- VHDL/Verilog Simulation Models
- Test Suite
- Synthesis and Testsynthesis Scripts
- Example Design available
- Design Support, Netlist Synthesis Service and Consulting available

Requirements

Simulation

- VHDL IEEE 1076 Simulator (Synopsys VSS, Modeltech VSIM, Vantage, others)
- Verilog IEEE 1364 Simulator, netlist only
(Cadence VerilogXL, Modeltech VSIM, others)

Synthesis

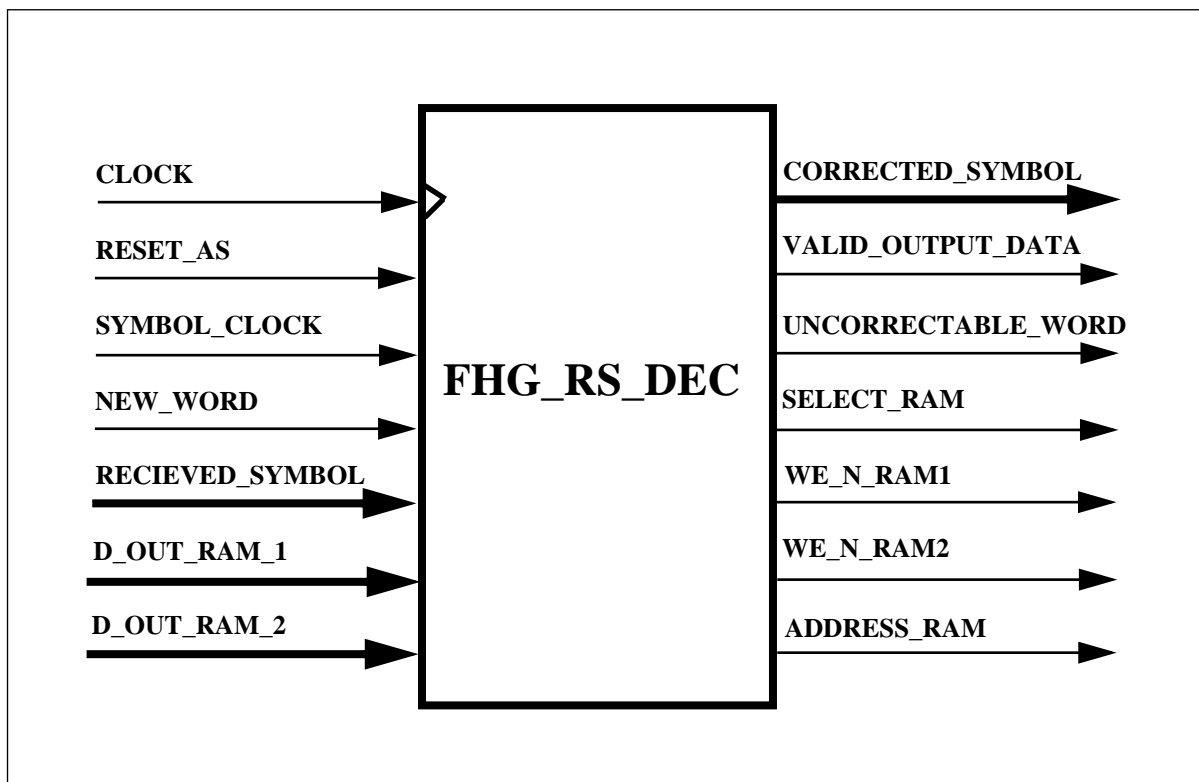
- Synopsys Design Compiler

References

For more detailed information about Reed-Solomon decoder refer to standard literature about the Reed-Solomon decoding procedure and Reed-Solomon decoders (e.g. Richard E. Blahut: "Theory and Practice of Error Control Codes", Addison-Wesley, 1983).

Block Diagram

Figure 1: Block Diagram



Signal Description

Table 2: Signal Description

Pin	Direction	Bit-width	Description
CLOCK	IN	1	System clock
RESET_AS	IN	1	Low activ asynchronous reset
SYMBOL_CLOCK	IN	1	Symbol clock signal driving input and output data
NEW_WORD	IN	1	Signal for detecting that there is a new block at the input
RECEIVED_SYMBOL	IN	M	Input data (received word) and RAMs input
D_OUT_RAM_1	IN	M	Stored received symbol data from RAM_1
D_OUT_RAM_2	IN	M	Stored received symbol data from RAM_2
CORRECTED_SYMBOL	OUT	M	Output data (decoded word)
VALID_OUTPUT_DATA	OUT	1	Signal for indicating that there is a valid data at the output, set to “1” each time a corrected symbol is available at the output
UNCORRECTABLE_WORD	OUT	1	Signal for indicating that there have been more than T errors during transmission of one word (T is the number of symbols the code is able to correct), set to ‘1’ during the output time of such word
SELECT_RAM	OUT	1	Indicates which RAM is being read and written ‘0’ → RAM_1 ‘1’ → RAM_2
WE_N_RAM1	OUT	1	Low active write enable for RAM1
WE_N_RAM2	OUT	1	Low active write enable for RAM2

General Information

Signal **NEW_WORD** must be set to “1” during at least half a symbol clock cycle and at most one symbol clock cycle.

Parameter Description

Tabelle 3: Parameter Description

Name	Type	Default Value	Value Range	Description
M	Integer	8	>1	Number of bits of the Galois field elements, and therefore, of the symbols. The number of symbols in one block will be $N = 2^M - 1$
PRIMITIVE_POL	Integer	285	>0	Primitive polynomial from which the $GF(2^M)$ is generated. It must be given in integer form. Example: Primitive polynomial in polynomial form: $P(x) = x^8 + x^4 + x^3 + x^2 + 1$. P(x) in binary form: P(x) = 100011101. P(x) in integer form: PRIMITIVE_POL = 285
K	Integer	223	$K = 2^M - 2T - 1$	Number of information symbols, that is, number of symbols in one block before it is encoded. Then, $N - K = 2T$ will be the number of parity symbols, being T , the number of symbols the code is able to correct
J0	Integer	0	$\{0 \dots 2^M - 2\}$	Constant introduced in the generator polynomial of the code. Its value must be in $\{0 \dots N-1\}$
K0	Integer	1	$\{1 \dots 2^M - 2\}$	Constant introduced in the generator polynomial of the code. Its value must be in $\{1 \dots N-1\}$, and N and K_0 must be mutually prime
AW	Integer	8	$\geq \text{ld}(K)$	Bitwidth of both RAMs. Its value must be so that $2^{AW} \geq K$

Operating speed

The relation between operating speed and symbol rate depends on the value of the error correcting capability, and the number of symbols in the information word and in the codeword. There are two different cases depending on the value of the error correcting capability of the code: When the error correction capability is bigger than five symbols ($t > 5$), the following conditions are required:

$$T_{SYMBOL-CLOCK} \geq 5T_{CLOCK} \quad \text{OR}$$

$$T_{SYMBOL-CLOCK} \geq \frac{[(4 \times (2t - 1)) + 5] + [n + 3] + [(t \times t) + 7] + 1}{n} T_{CLOCK}$$

Error correcting capability less than or equal to five symbols ($t \leq 5$):

$$T_{SYMBOL-CLOCK} \geq 5T_{CLOCK} \quad \text{OR}$$

$$T_{SYMBOL-CLOCK} \geq \frac{[(4 \times (2t - 1)) + 5] + [n + 3] + [(t + 5) \times t] + 1}{n} T_{CLOCK}$$

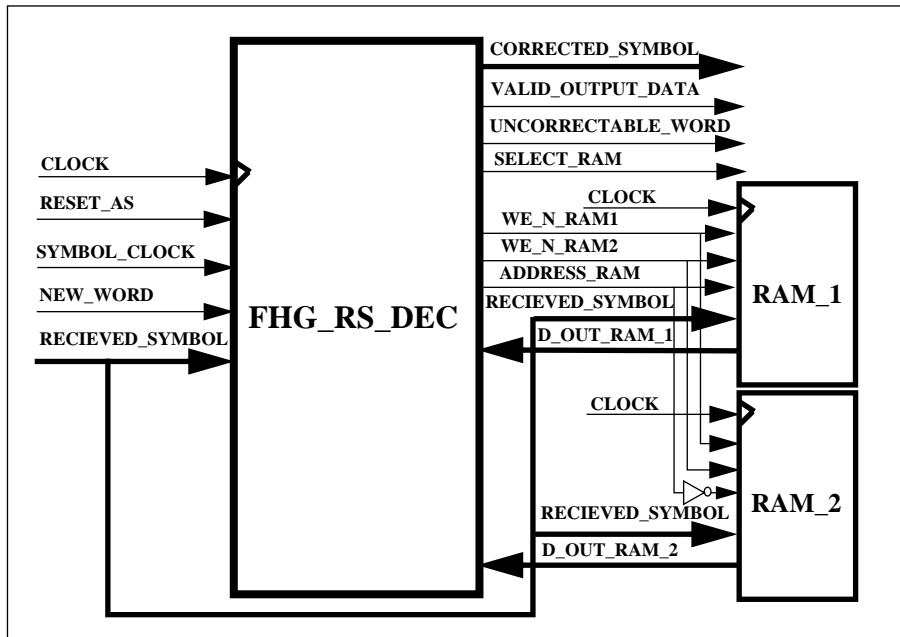
where n is the number of symbols in one codeword,
 k is the number of information symbols in one word,
 t is the number of errors the Reed-Solomon decoder is able to correct,
 T_{SYMBOL_CLOCK} is the period of the symbol clock, the symbol rate,
 T_{CLOCK} is the period of the system clock, that is, the operation speed.

Notes

The Reed-Solomon decoder and its environment are a fully synchronous design. Therefore, all design blocks (Reed-Solomon decoder and RAMs) have to be driven by the same CLOCK signal. It is assumed that the bits in one code symbol are applied in parallel to the input of the Reed-Solomon decoder. If this is not the case, a serial-to-parallel conversion of the received data is required.

The number of errors the Reed-Solomon decoder is able to correct must be at least 1 ($2t = N-K \geq 1$). The memories for the storage of the received word have to be implemented as external RAMs with corresponding memory sizes and datawidths (K and M). Both RAMs have to be synchronous Double-Port-RAMs (one read and one write port), with a clock input, low-active write-enable input, an address input, and a low-active chip-enable input, i.e. if the write enable and chip enable signals are low, the input data are stored with the next rising edge of the clock.

Figure 2: Reed-Solomon decoder with two RAMs



Behavioral Timing Diagramms

Figure 3: Initialization

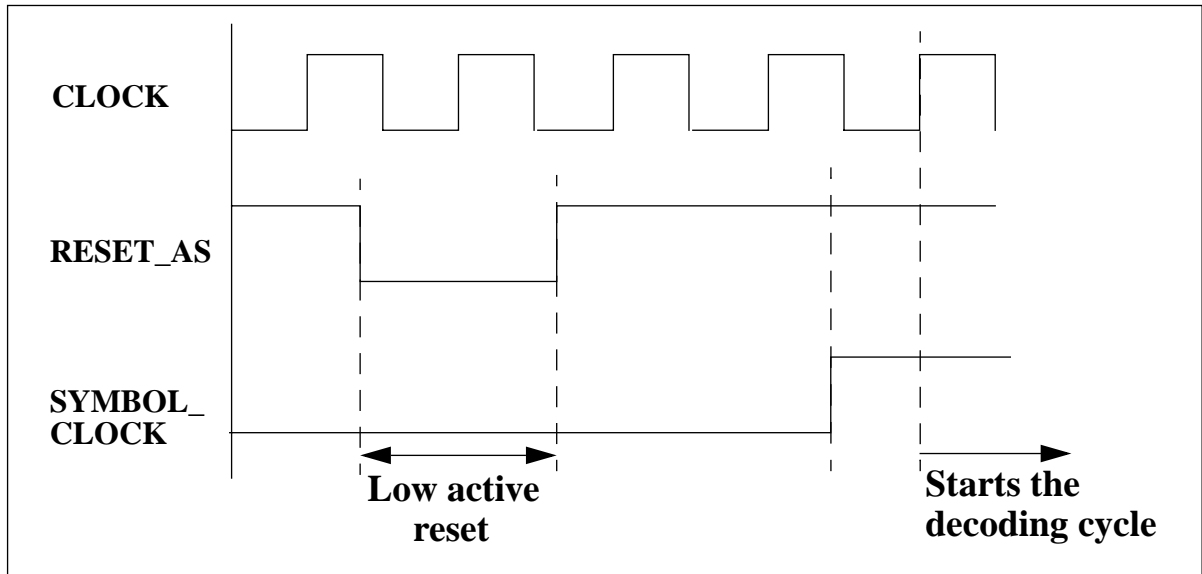


Figure 4: Input synchronization

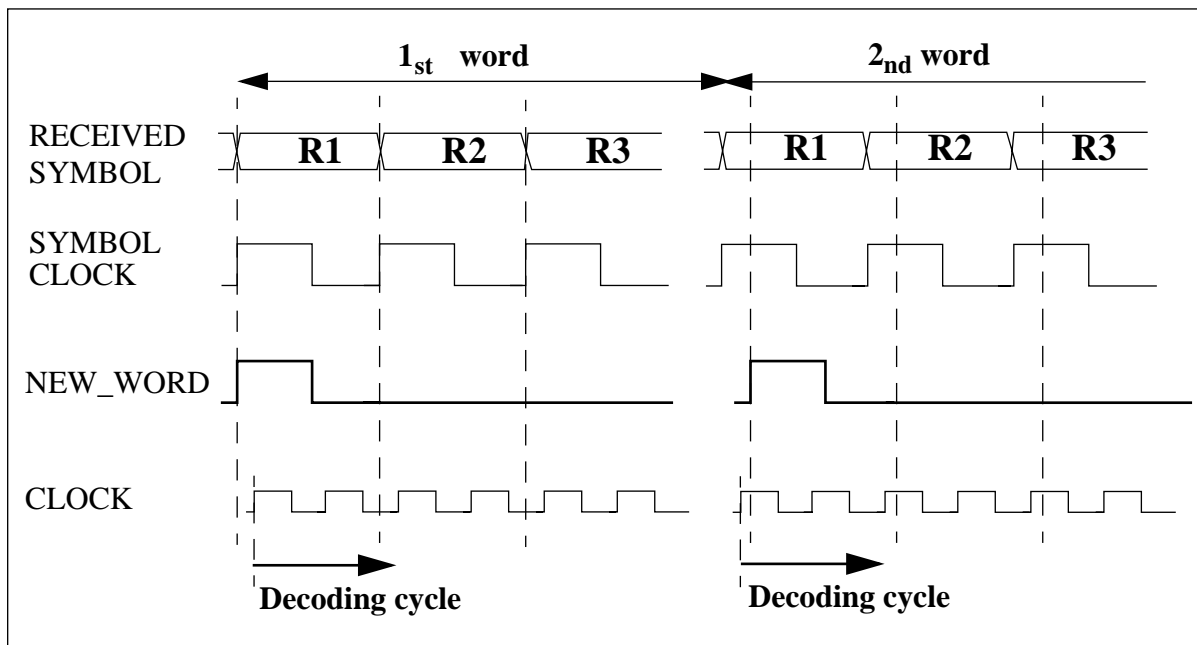
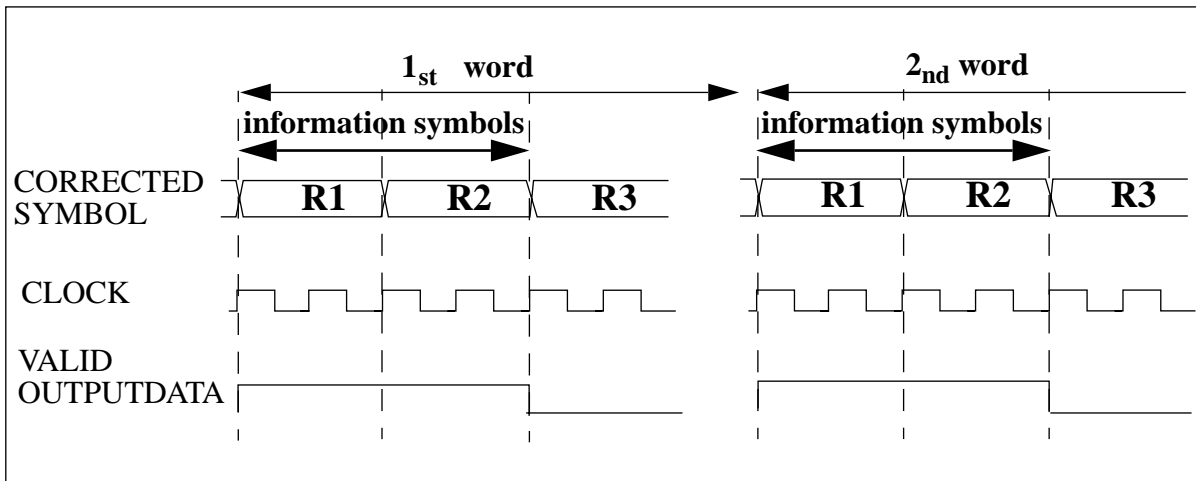


Figure 5: Output synchronization



Waveform of external read and write access of the RAMs

Figure 6: Write and read cycles of the two RAMs

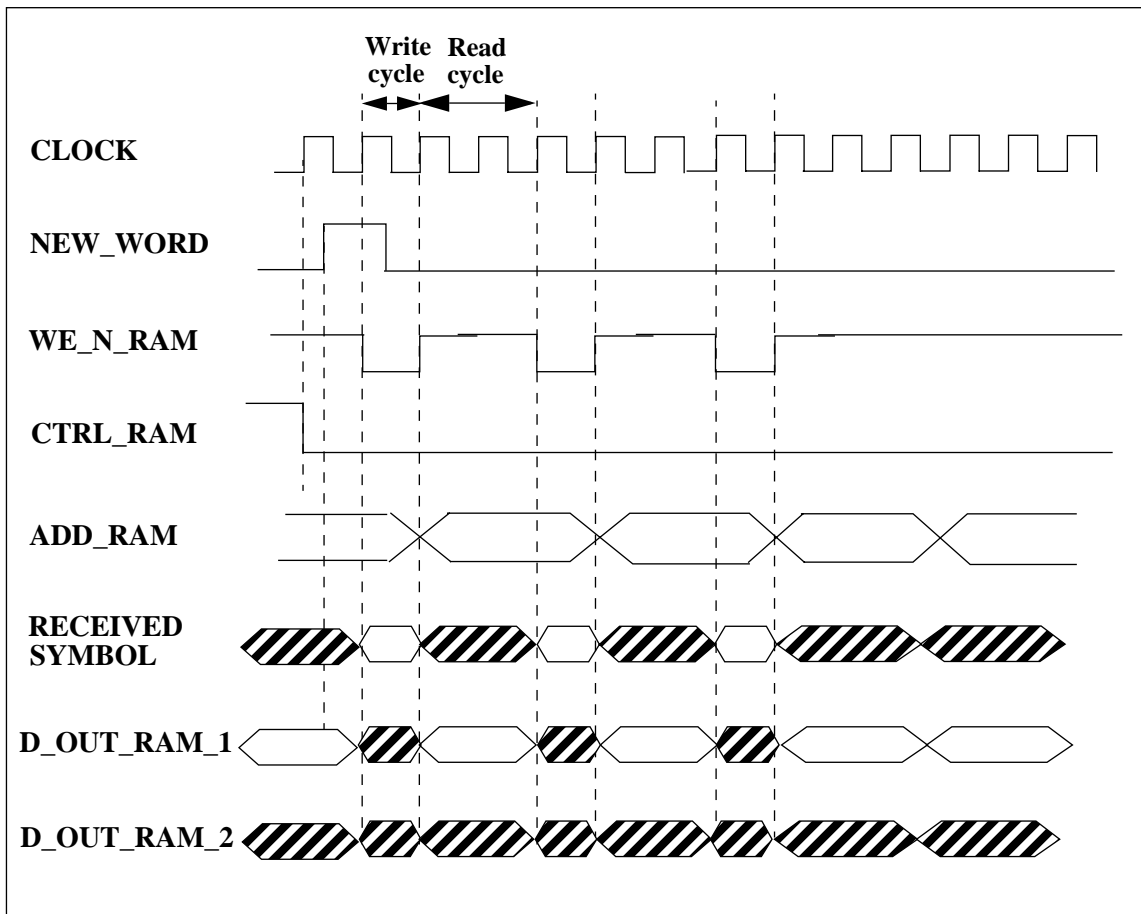
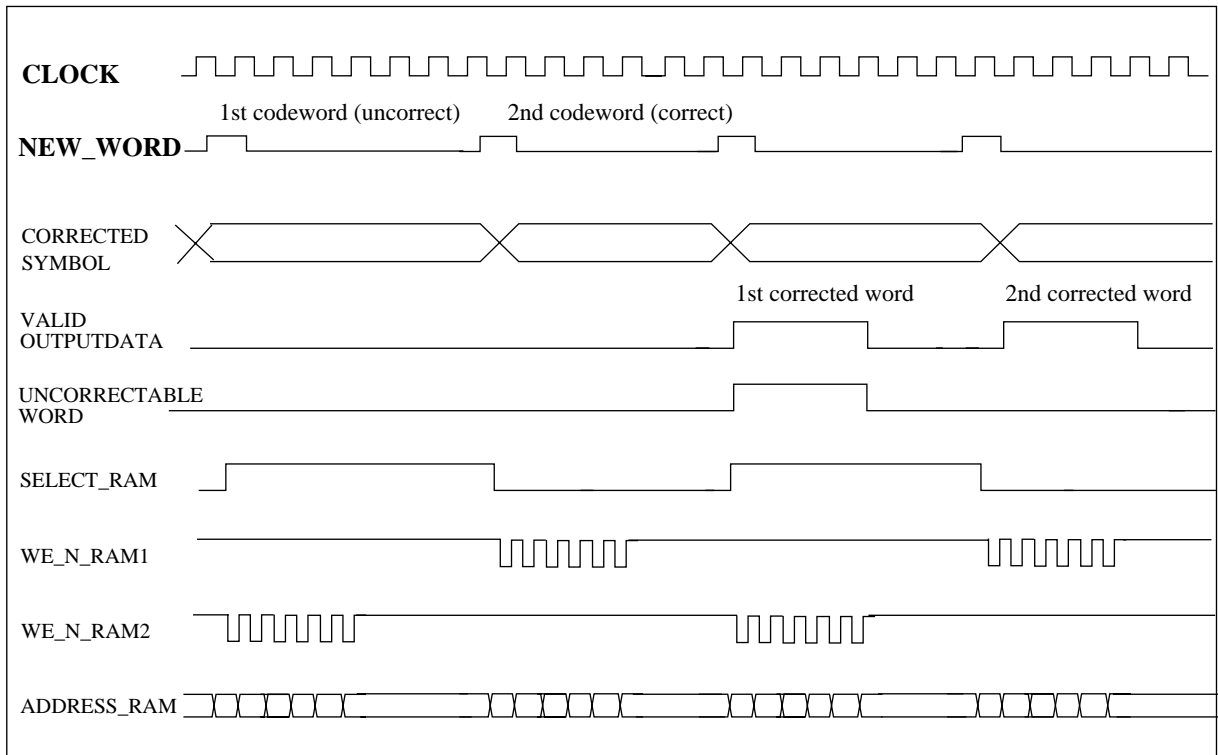
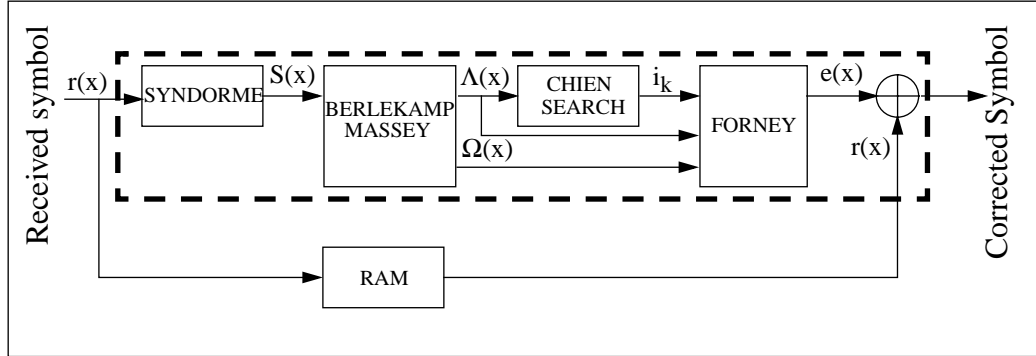


Figure 7: Output waveform



Description of the decoding algorithm

Figure 8: Reed-Solomon decoder basic structure



There are five steps implied in the decoding process. The first step is the “syndrome” calculation. The syndrome is a polynomial that will help, in the next step, to calculate the “error-locator polynomial”. This polynomial is calculated by means of the Berlekamp-Massey algorithm. It has the characteristic that its roots are the positions, in the receive word, where there have been errors. The following steps, the Chien search and the Forney algorithm, calculate the positions of the errors (by calculating the roots of the error locator polynomial), and the magnitude of such errors, respectively. Then, the last step, is to subtract the error pattern from the received word, in order to get the correct codeword.

If we see the words as polynomials, the codeword, the error word and the received word are, respectively:

$$\begin{aligned} c(x) &= c_0 + c_1x + c_2x^2 + \dots + c_{n-1}x^{n-1} \\ e(x) &= e_0 + e_1x + e_2x^2 + \dots + e_{n-1}x^{n-1} \\ r(x) &= c(x) + e(x) = r_0 + r_1x + r_2x^2 + \dots + r_{n-1}x^{n-1} \end{aligned}$$

It is necessary to know the generator polynomial of the code for calculating the syndrome. This polynomial is:

$$g(x) = (x + \alpha^{k_0j_0})(x + \alpha^{k_0(j_0+1)}) \dots (x + \alpha^{k_0(j_0+2t-1)})$$

which has as roots $\alpha^{k_0j_0}, \alpha^{k_0(j_0+1)}, \dots, \alpha^{k_0(j_0+2t-1)}$, being α^i , elements of the finite field, $GF(2^m)$. The syndrome polynomial will be

$$S_j = r(\alpha^{(j_0+j-1)k_0})$$

The syndrome polynomial is used for the Berlekamp-Massey algorithm, in order to calculate the “error-locator” polynomial, $\Lambda(x)$, in this way:

$$S_j = - \sum_{i=1}^v \Lambda_i S_{j-i} \quad j = v+1, \dots, 2v$$

This polynomial is of the form:

$$\Lambda(x) = \prod_{k=1}^v (1 - x\alpha^{i_k}) = \Lambda_v x^v + \Lambda_{v-1} x^{v-1} + \dots + \Lambda_1 x + 1$$

Where the indices i_k correspond to the time indices of the v errors. That is, if we calculate the roots of this polynomial, we will know the positions of where the errors occurred. This is done by the third step: The “Chien Search”.

Once we know where are the errors, the only thing to do is to calculate the value of these errors. This is performed by the Forney algorithm, which uses this expression for doing it:

$$e_{i_m} = -\alpha^{i_m(1-j_0)} \frac{\Omega(\alpha^{-i_m})}{\Lambda'(\alpha^{-i_m})}$$

Where $\Omega(x)$ the “error-evaluator” polynomial, which is calculated during the Berlekamp-Massey algorithm, at the same time, and in the same way that the “error-locator” polynomial.

At this point, it is only one thing left before the last step is performed. Because of the existence of the parameter k_0 , the actual positions of the errors have been cyclically permuted, they must be permuted again to their correct location:

$$i_k = (i_q l_0)_n$$

Where l_0 has the value so that: $(l_0 k_0)_n = 1$.

The last step in the decoding process is to correct the error pattern. It can be accomplished by subtracting the error pattern e_j from the received word to obtain the corrected codeword. As in Galois field $GF(2^m)$ subtraction is the same operation as addition, this can be accomplished with modulo-2 addition.

Application notes

The following example gives evidence of how parameters of the Reed-Solomon decoder have to be chosen for a special application.

Example 1

Code: $(N, K) = (255, 223)$

Symbol length: $M = 8$ ($N = 2^M - 1 = 2^8 - 1 = 255$)

Primitive polynomial:

In polynomial form: $P(x) = x^8 + x^4 + x^3 + x^2 + 1$.

In binary form: $P(x) = 100011101$.

In integer form; **PRIMITIVE_POL = 285**.

Number of information symbols: $K = 223$. $g(x) = \prod_{i=0}^{2r-1} (x + \alpha^{k_0(j_0+i)})$.

Parameter J_0 : any value between 0 and $N-1$.

Parameter K_0 : any value between 1 and $N-1$, so that N and K_0 are mutually prime.

Parameter AW : $2^{AW} \geq K$, $K = 223$, then $AW = 8$.

Error correcting capability: $t = \frac{N-K}{2} = 16$.

Operating speed:

$T_{\text{SYMBOL_CLOCK}} \geq 5 T_{\text{CLOCK}}$ or $T_{\text{SYMBOL_CLOCK}} \geq 8.3 T_{\text{CLOCK}}$ or
 $T_{\text{SYMBOL_CLOCK}} \geq 2.5 T_{\text{CLOCK}}$.

The most restrictive condition is the second one, then, the operating speed has to be: $f_{\text{CLOCK}} \geq 8.3 f_{\text{SYMBOL_CLOCK}}$.

The following example shows how the decoding procedure is performed step by step.

Example 2

Code: $(7, 3)$, $\mathbf{n} = 7$, $\mathbf{k} = 3$.

GF(2^3): $\mathbf{m} = 3$, $p(x) = x^3 + x + 1$ (11 in integer form)

$\mathbf{j}_0 = 5$, $\mathbf{k}_0 = 4$.

The generator polynomial is:

$$g(x) = (x + \alpha^{20})(x + \alpha^{24})(x + \alpha^{28})(x + \alpha^{32}) = x^4 + x^3 + \alpha^4 x^2 + \alpha^3 x + \alpha^6$$

With an information word like this:

$$i(x) = \alpha^2 x^2 + \alpha^5 x + \alpha^6$$

the corresponding systematic codeword is:

$$c(x) = \alpha^2 x^6 + \alpha^5 x^5 + \alpha^6 x^4 + \alpha^6 x^3 + \alpha^4 x^2 + x + \alpha^2.$$

If the error pattern is $e(x) = \alpha^5 x^5 + \alpha^6 x^4$, the received word will be

$$r(x) = \alpha^2 x^6 + \alpha^6 x^3 + \alpha^4 x^2 + x + \alpha^2$$

Calculating the j -th syndrome symbol as

$$S_j = r_{n-1} \alpha^{(n-1)(j_0+j-1)} + \dots + r_2 \alpha^{2(j_0+j-1)} + r_1 \alpha^{(j_0+j-1)} + r_0$$

we obtain the following syndrome coefficients:

$$\mathbf{S}_1 = r(\alpha^6) = E_6 = \alpha^6; \mathbf{S}_2 = r(\alpha^3) = E_3 = \alpha^3; \mathbf{S}_3 = r(\alpha^0) = E_0 = \alpha;$$

$$\mathbf{S}_4 = r(\alpha^4) = E_4 = \alpha^2$$

After the Berlekamp-Massey algorithm we obtain the error-locator polynomial: $\Lambda(x) = \alpha x^2 + x + 1$ and the error-evaluator polynomial: $\Omega(x) = \alpha^4 x + \alpha^6$

Applying the Chien-Search, we find that $\Lambda(\alpha^5) = 0$ and $\Lambda(\alpha) = 0$. Then, the assumed error locations are α^2 and α^6 . Deriving $\Lambda(x)$, we have $\Lambda'(x) = 1$, and evaluating $\Omega(x)$ in the assumed positions of the error: $\Omega(\alpha^6) = \alpha$ and $\Omega(\alpha^2) = 1$ and then:

$$-f_1 = \alpha^{6(1-5)} \alpha = \alpha^5, \text{ is the error magnitude in position 6 } (\alpha^6)$$

$$-f_2 = \alpha^{2(1-5)} = \alpha^6, \text{ is the error magnitude in position 2 } (\alpha^2)$$

The vector $f(x)$ is then

$$f(x) = \alpha^5 x^6 + 0 + 0 + 0 + \alpha^6 x^2 + 0 + 0$$

But this is not the real error pattern, but the cyclically permuted sequence of $e(x)$. We can calculate the real error positions as follows. We must find the integer l_0 that fulfils the equation $(k_0 l_0)_n = 1$, and then, using Eq. (3.27) we will have the correct error pattern. With $k_0 = 4$ the value of l_0 is $l_0 = 9$;

$$i_1 = (6 \cdot 9)_n, \text{ then the real error location here is } i_1 = 5$$

$$i_2 = (2 \cdot 9)_n, \text{ then, } i_2 = 4.$$

Then, the error pattern is

$$e(x) = 0 + \alpha^5 x^5 + \alpha^6 x^4 + 0 + 0 + 0 + 0$$

Adding this vector to the received word and getting the first part of the result, we obtain the information word

$$i(x) = \alpha^2 x^2 + \alpha^5 x + \alpha^6$$