(255, 239) Reed-Solomon Decoder

Features

- **Industry standard (255, 239) Reed-Solomon decoder**
- **Available for FPGA or ASIC implementation**
- **High speed design, low latency implementation reaches 2.0 Gbps data rate and low gate count implementation reaches 1.1 Gbps data rate in Virtex II, higher in ASIC**
- **Compact design, low gate count implementation uses 691 CLB slices and low latency implementation uses 1125 CLB slices in Virtex II, among the smallest on the market**
- **Can work continuously with no gap between code blocks**
- **Fully synchronous one clock design**
- **255 + 97 clock cycle latency in low latency implementation**
- **255 + 204 clock cycle latency in low gate count implementation**

Functional Description

The decoder has three functional blocks and

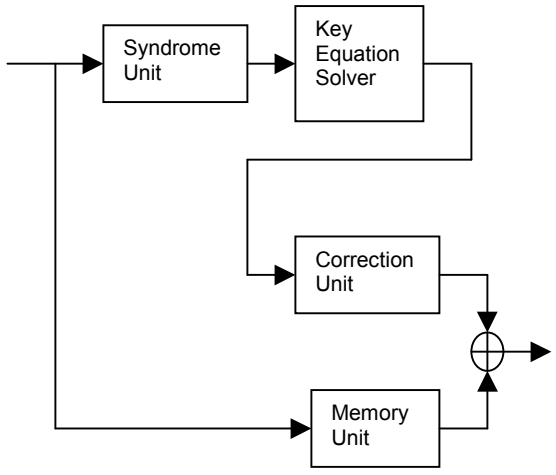

Figure 1. Block diagram of the Reed-Solomon decoder.

one memory block as shown in Figure 1. First the syndrome unit calculates the syndromes. Then the key equation solver solves the key equation for the error location polynomial. The correction unit calculates the error location and value and then adds the error sequence to the received code word to get the corrected code word. The memory unit is used to store the received code word while the decoder calculates the syndromes and solves the key equation. During the correction stage, the stored code word is read out from the memory and added to the error sequence to get the corrected code word.

Pin Out

Figure 2 is the schematic symbol of the Reed-Solomon decoder.

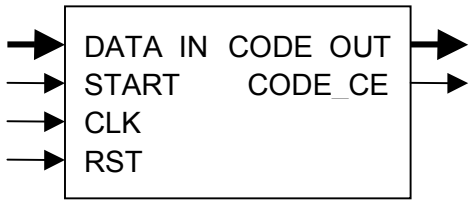

Figure 2. Schematic symbol of the decoder

RST

One bit input, the asynchronous reset. When RST is set high, all the internal flip-flops are asynchronously initialized. The core will stay in this state until RST is set low.

CLK

One bit input, the global clock. All sequential logic acts on the rising edge of CLK.

START

One bit input, the signal to start the decoding process. At the rising edge of CLK, if START is high while RST is low, the internal state machine will start the decoding process. For continuous operation, the START signal for the nest code block must line up with the last byte of the previous code block. Otherwise, the START signal for the next code block must be at least five

clocks after the last byte of the previous code block. The length of START should be one clock cycle.

DATA_IN
Eight bit input, the received code word. The first byte of the code word should be one clock after the START pulse. The decoder reads in one byte every clock. Each code word must have 255 bytes with 239 data bytes and 16 parity check bytes. If there is a second code word following the first one, the second code word must follow the first one either with no gap or with a six-clock cycle gap.

CODE_OUT
Eight bit output, the decoded code word. If there are eight or less than eight erroneous bytes, the output is the corrected code word. Otherwise, the output is unpredictable.

CODE_CE
One bit output, the clock enable for outputting the corrected code word. The length of CODE_CE is 255 clock cycles.

Timing Diagrams

The Reed-Solomon decoder is very easy to be integrated into a larger design. The following timing diagrams help to clarify some of the synchronization issues.

Figure 3 shows the timing diagram at the starting point of the decoding process, where D0 is the first byte of the received code word.

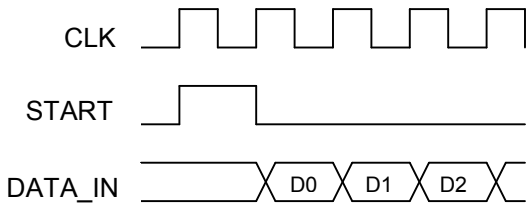

Figure 3. The timing diagram at the starting point of the decoding process

Figure 4 shows the timing diagram at the starting point of the output, where C0 is the first byte of the decoded code word.

Figure 5 shows the timing diagram of the continuous operation mode, where DA254 is the last byte of the first received code word and DB0 is the first byte of the following

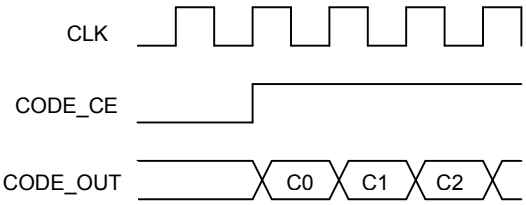

Figure 4. The timing diagram at the starting point of the output

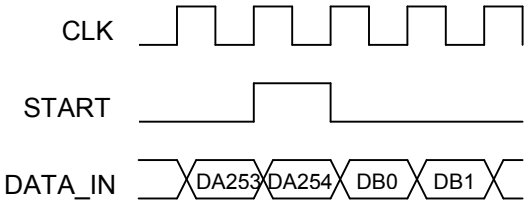

Figure 5. The timing diagram of the continuous operation mode

code word.

Deliverables

Deliverables include the encoder/decoder and the test bench. For Xilinx FPGA implementation, both source code and netlist are available. For ASIC implementation, only source code will be delivered. Source code can be in VHDL or Verilog.

Ordering Information

We have flexible licensing structures. Please use the following information to contact us:

Highland Communications Technologies
928 Concession Road, Fort Erie
Ontario, Canada L2A 6B8

Tel: 1-905-658-0989
Fax: 1-905-248-5188
Email: sales@highlandcomm.com

Web site:
http://www.highlandcomm.com/