OTU2 I.4 FEC IP Core (IP-OTU2EFECI4Z) Data Sheet

Revision	0.08
Release Date	2014-03-29
Document number	TD0307

Table of Contents

1 Introduction

The Altera G.975 I.4, IP-OTU2EFECI4Z IP core implements the RS(1023,1007)/BCH(2047,1952) super FEC code from the appendix I.4 of the ITU-T G.975.1 standard, [1].

The main features are:

- Net electrical coding gain (NECG) of ~8.3 dB
- 7% overhead
- Latency of 58 µsec (includes both encoder + decoder latency)
- 64 bit data path width
- Error statistics monitoring:
    - Counts of corrected errors
    - Counts of uncorrectable component codes
    - Counts of corrected zeros relative to corrected ones

2 Architecture

Figure 1 illustrates the system architecture of the IP-OTU2EFECI4Z IP core.

The FEC encoder receives OTU2 data including frame alignment signal (FAS) and multi-frame alignment signal (MFAS) from the device core. The OTU2 data is encoded with redundant FEC data. The resulting FEC encoded OTU2 must subsequently be scrambled before it is transmitted across the OTN network

In the other direction the frame start of the OTU2 received from the OTN network must be recovered in order ro forward the descrambled OTU2 data to the FEC decoder. The redundant FEC data is decoded and identified errors are corrected before the data is forwarded to the device core.

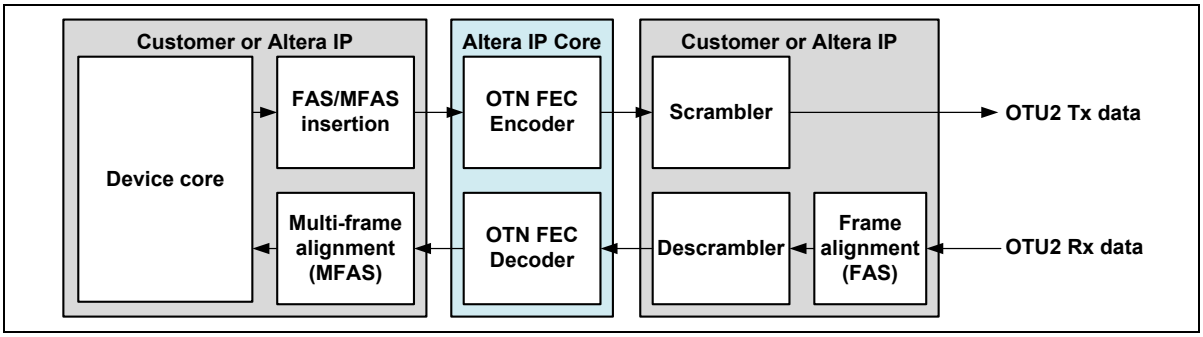

Figure 1: G.975 I.4 EFEC System Architecture

3 Performance and Resource Utilization

Stratix V devices use combinational adaptive look-up tables (ALUTs) and logic registers. Table 1 shows the typical performance and resource usage for the 10 Gbit/s G.975 I.4 EFEC on a Stratix V GX device as reported by the Quartus® II software. The resource figures will vary slightly depending on the Quartus version used, synthesis seed numbers etc. The latencies through the encoder and decoder do not vary from frame to frame, they are a fixed number of clock cycles.

Table 1: Performance - 10 Gbit/s G.975.1 I.4 EFEC on Stratix V GX

Block	ALUTs	MLABs	Logic Registers	Memory (M20Ks)	fMax (MHz)	Latency (Cycles)	Latency (µs @175 MHz)
Encoder	8,202	8	6,560	38	>200	~550	~3
Decoder	45,676	82	14,214	162	>300	~9700	~55

4 Functional Overview

4.1 Encoder

The encoder expects an OTU frame including the FEC field (which will be overwritten, and the input value is thus ignored) on its input, and generates an OTU frame with the parity information added on its output. The block uses two clocks, clk_i which is at least OTU2/64, and clk2_i which is twice that frequency. A rising edge on clk_i must coincide with a rising edge on clk2_i.

The data to be encoded shall be delivered on block_data_i one complete frame at a time, in 2040 consecutive cycles. During the first of these cycles block_start_i shall be asserted (high). If the clock for the block is faster than OTU2/64, the differences is handled by inserting idle cycles after each frame. All data delivered on block_data_i after the complete frame has been delivered and until block_start_i is asserted again is ignored.

The data format is MSB first, with the first bit on the line being the MSB of the word delivered to the encoder while block_start_i is asserted. The input format is shown in Figure 2.

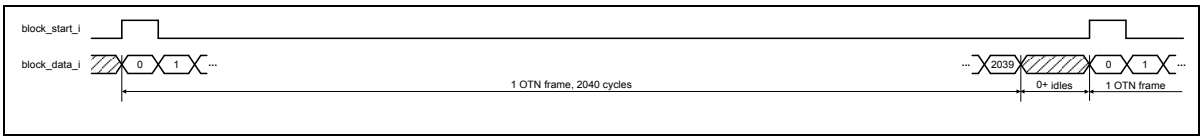

Figure 2: Encoder input format

The output of the encoder uses the same format, and is generated a fixed number of cycles after the input. This latency does not vary from frame to frame.

Table 2 lists the encoder input and output ports for connecting to the OTU2 G.975 I.4 EFEC IP core.

Table 2: Encoder I/O Port Listing

I/O port	Name	Port Width (Bits)	Description
in	clk_i	1	Clock port.
in	clk2_i	1	Used for double clocking logic & RAM. A rising clk_i must coincide with a rising clk2_i.
in	reset_i	1	Async reset. Deassert synchronized to clk_i.
in	reset2_i	1	Async reset. Deassert synchronized to clk2_i.
in	block_start_i	1	Pulsed with first word of block input.
in	block_data_i	64	Data words to be encoded. Input in 2040 consecutive cycles, i.e. a complete frame. If clk_i is faster than the nominal OTU2 clock (166.33MHz) the difference is absorbed by buffering up enough data from the next frame such that insertion of data from that frame can again happen in 2040 consecutive cycles
out	block_start_o	1	Pulsed with 1st word of block output.
out	block_data_o	64	Block output in 2040 consecutive cycles

4.2 Decoder

The decoder expects a complete I.4 OTU2 frame on its input, and from this it generates 4 OTU2 rows on its output. The output sequence mimics the way data is typically output by a GFEC decoder. E.g. 4 rows each with 3824 bytes of data and 256 bytes to discard. The block uses one clk_i which must be at least OTU2/64.

The data to be decoded shall be delivered on block_data_i one complete frame at a time, in 2040 consecutive cycles. If the clock for the block is faster than OTU2/64, the difference shall be handled by pausing before each frame to accumulate sufficient data such that the complete OTU2 frame can be delivered to the decoder without pauses. Any data presented on block_data_i after the complete frame has been input and until block_start_i is asserted again is ignored.

The data format is MSB first, with the first bit on the line being the MSB of the word delivered to the encoder while block_start_i is asserted. The input format is shown in Figure 3 and the output format is shown in Figure 4. Note that the figure does not imply a particular timing of the status outputs relative to block_start_o. All the status outputs for a block (e.g. an OTU2 frame) are however complete before the last data of the block is output.

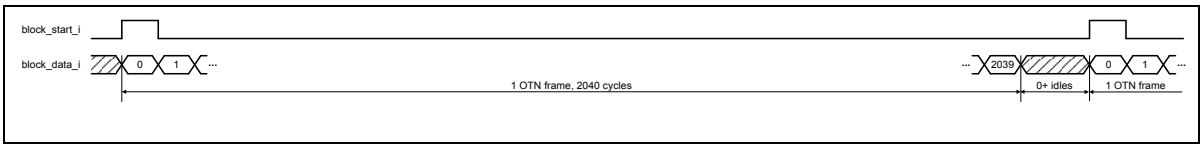

Figure 3: Decoder input format

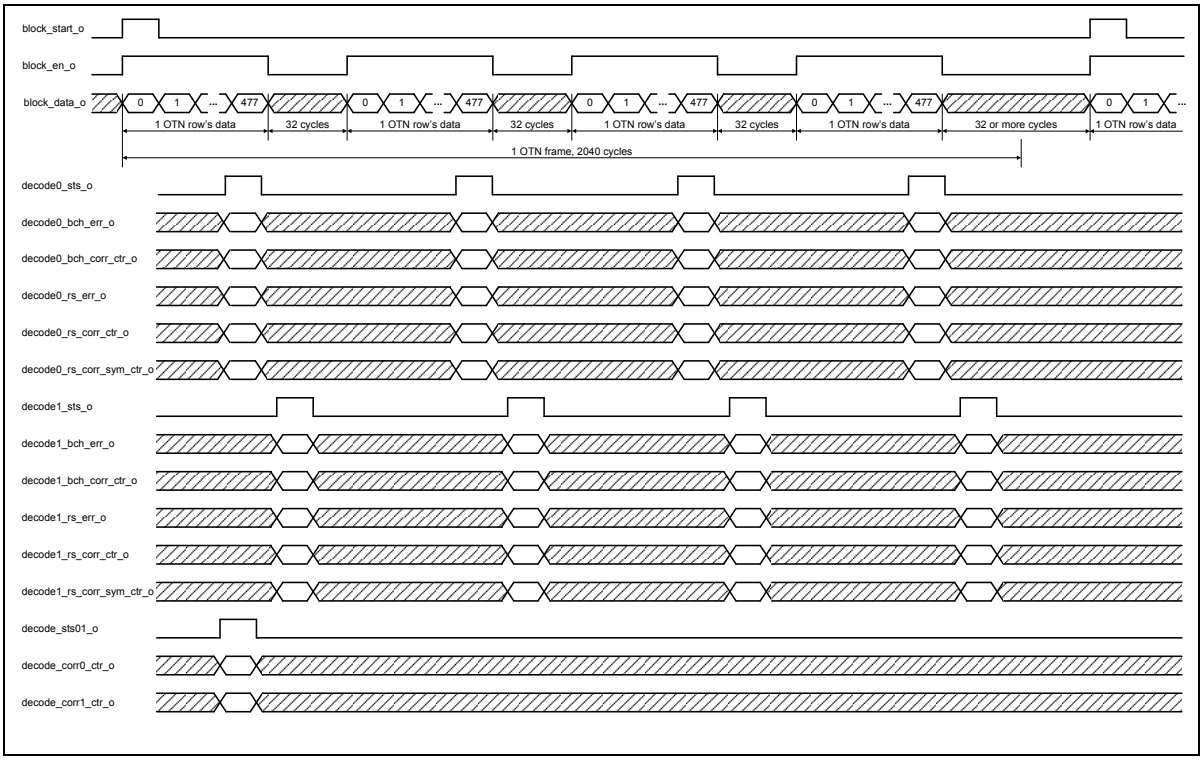

Figure 4: Decoder output format

The decoder output is delivered a fixed number of cycles after the input. This latency does not vary from frame to frame.

Table 4 lists the decoder input and output ports for connecting to the OTU2 G.975 I.4 EFEC IP core. The error statistics counters for corrected zeros and ones are enabled through generic STS01 as described in Table 3.

Table 3: Design parameters

Generic	Type	Description
STS01	integer	If non-zero, the decode_sts01_o, decode_corr0_ctr_o and decode_corr1_ctr_o outputs are active

Table 4: Decoder I/O Port Listing

I/O port	Name	Port Width (Bits)	Description
in	clk_i	1	Clock port.
in	reset_i	1	Async reset. Deassert synchronized to clk_i.
in	block_start_i	1	Pulsed with 1st word of received block input.
in	block_data_i	64	Received data words. Input in 2040 consecutive cycles, i.e. a complete frame. If clk_i is faster than the nominal OTU2 clock (166.33MHz) the difference is absorbed by buffering up enough data from the next frame such that insertion of data from that frame can again happen in 2040 consecutive cycles.
out	block_start_o	1	Pulsed with first word of corrected block output.
out	block_en_o	1	Set when block_data_o is valid.
out	block_data_o	64	Corrected payload bytes. Output in 4 times 478 consecutive cycles with 32 cycles between each run of 478 cycles (to emulate G.709 RS FEC output sequence).
Ports related to the first decoding iteration:			
in	bch0_corr_en_i	1	Enables BCH correction and normally set. Can be used to test correct RS operation when cleared.
in	rs0_corr_en_i	1	Enables RS correction and normally set. Can be used to test correct BCH operation when cleared The calculations (including statistics) are still carried out with one or both of the above cleared. Only the corrections are omitted.
out	decode0_sts_o	1	Pulsed when status presented. Pulsed once per row being output.
out	decode0_bch_err_o	7	Number of BCH codes, that couldn't be corrected.
out	decode0_bch_corr_ctr_o	10	Number of bits successfully corrected by inner BCH code.
out	decode0_rs_err_o	3	Number of RS codes, that couldn't be corrected.
out	decode0_rs_corr_ctr_o	9	Number of bits successfully corrected by outer RS code.
out	decode0_rs_corr_sym_ctr_o	6	Number of symbols (= dectets) successfully corrected by outer RS code.
Ports related to the second, final decoding iteration:			
in	bch1_corr_en_i	1	Enables BCH correction and normally set. Can be used to test correct RS operation when cleared.
in	rs1_corr_en_i	1	Enables RS correction and normally set. Can be used to test correct BCH operation when cleared The calculations (including statistics) are still carried out with one or both of the above cleared. Only the corrections are omitted.
out	decode1_sts_o	1	Pulsed when status presented. Pulsed once per row being output.
out	decode1_bch_err_o	7	Number of BCH codes, that couldn't be corrected.

I/O port	Name	Port Width (Bits)	Description
out	decode1_bch_corr_ctr_o	10	Number of bits successfully corrected by inner BCH code.
out	decode1_rs_err_o	3	Number of RS codes, that couldn't be corrected. If != 0, then the block is err'd.
out	decode1_rs_corr_ctr_o	9	Number of bits successfully corrected by outer RS code.
out	decode1_rs_corr_sym_ctr_o	6	Number of symbols (= dectets) successfully corrected by outer RS code.
Counting of corrected zeros and ones			
in	sts_scr_en_i	1	Normally set. When set, scrambling is accounted for when calculating decode_corr0_ctr_o, decode_corr1_ctr_o. Ignored unless generic STS01 is non-zero.
out	decode_sts01_o	1	Pulsed once per frame. Only pulsed if generic STS01 is non-zero.
out	decode_corr0_ctr_o	10	Number of bits corrected 0 -> 1 in the first 952 bytes of the block. Valid when decode_sts01 set.
out	decode_corr1_ctr_o	10	Number of bits corrected 1 -> 0 in the first 952 bytes of the block. Valid when decode_sts01 set.

5 Synthesis constraints

5.1 Encoder

The encoder top level name is: `fecenc_otn_i4` and the corresponding ZIP file is named `fecenc_otn_i4.zip`.

The `fecenc_otn_i4.zip` contains a complete set of encrypted RTL files for the encoder as well as the license file needed for Quartus decryption.

The following sections describe the settings needed to be used in a high level synthesis flow when the encoder is to be used in a larger design.

5.1.1 Parameters

The toplevel entity for the encoder has 2 generics (parameters in Quartus terms): `RS_CFG` and `BCH_CFG`. These should not be modified and must be left at their default values.

5.1.2 QSF settings

The following qsf settings are required in a higher chip level synthesis when the decoder is instantiated

```
set_global_assignment -name "REMOVE_DUPLICATE_LOGIC"           "ON"
set_global_assignment -name "REMOVE_DUPLICATE_REGISTERS"       "OFF"
set_global_assignment -name "AUTO_ROM_RECOGNITION"             "ON";
set_global_assignment -name "AUTO_RAM_RECOGNITION"             "OFF";
set_global_assignment -name "AUTO_SHIFT_REGISTER_RECOGNITION" "OFF";
```

If this conflicts with chip level preferences then these setting must be applied to the entity level instead.

5.1.3 SDC settings

The encoder entity has two clocks: `clk_i` and `clk2_i`, and these must be driven from the same PLL.

All remaining inputs and outputs to the encoder are synchronous with respect to `clk_i` and they are all single cycle timed.

5.2 Decoder

The decoder top level name is: `fecdec_otn_i4_x2` and the corresponding ZIP file is named `fecdec_otn_i4_x2.zip`.

The `fecdec_otn_i4_x2.zip` contains a complete set of encrypted RTL files for the decoder as well as the license file needed for Quartus decryption.

The following sections describe the settings needed to be used in a high level synthesis flow when the decoder is to be used in a larger design.

5.2.1 Parameters

The toplevel entity consists of 3 generics (parameters in Quartus terms): `RS_B0`, `USECLK2` and `STS01`. `RS_B0` and `USECLK2` should not be modified and must be left at their default values. `STS01` can be set as described previously in Table 3.

5.2.2 QSF settings

The following qsf settings are required in a higher chip level synthesis when the decoder is instantiated

```
set_global_assignment -name "REMOVE_DUPLICATE_LOGIC"           "ON"
set_global_assignment -name "REMOVE_DUPLICATE_REGISTERS"       "OFF"
set_global_assignment -name "AUTO_ROM_RECOGNITION"             "ON";
set_global_assignment -name "AUTO_RAM_RECOGNITION"             "OFF";
set_global_assignment -name "AUTO_SHIFT_REGISTER_RECOGNITION" "OFF";
```

If this conflicts with chip level preferences then these setting must be applied to the entity level instead.

5.2.3 SDC settings

The decoder entity has two clocks: `clk_i` and `clk2_i`, but the `clk2_i` is not used and hence can be left unconnected. The `clk_i` must be driven from a PLL.

All remaining inputs and outputs to the decoder are synchronous with respect to `clk_i` and they are all single cycle timed.

There are a few signals which can be considered semi-static and hence get relaxed timing constraints; these are:

```
set_false_path -from bch0_corr_en_i
set_false_path -from rs0_corr_en_i
set_false_path -from bch1_corr_en_i
set_false_path -from rs1_corr_en_i
```

These false paths can be included in higher level chip synthesis with proper instance path names pre-pended to the names.

6 References

[1] ITU-T Recommendation G.975.1, *Forward error correction for high bit-rate DWDM submarine systems*, ITU-T, Geneva, February 2004.

7 Revision history

Date	Revision	Description of changes
2013-07-11	0.01	• Initial version.
2013-07-12	0.02	• Comment resolution.
2013-07-19	0.03	• Corrected 0/1 statistics added.
2013-08-09	0.04	• Resource utilization updated.
2013-08-14	0.05	• Update system architecture.
2013-08-14	0.06	• Section 4 expanded (with waveforms etc.). • Section 5 added.
2013-08-15	0.07	• Delivery method changed to ZIP files with encrypted RTL files
2014-03-29	0.08	• Update PN.