



---

**eSi-7601 Reed Solomon**

---

---

# 1 Contents

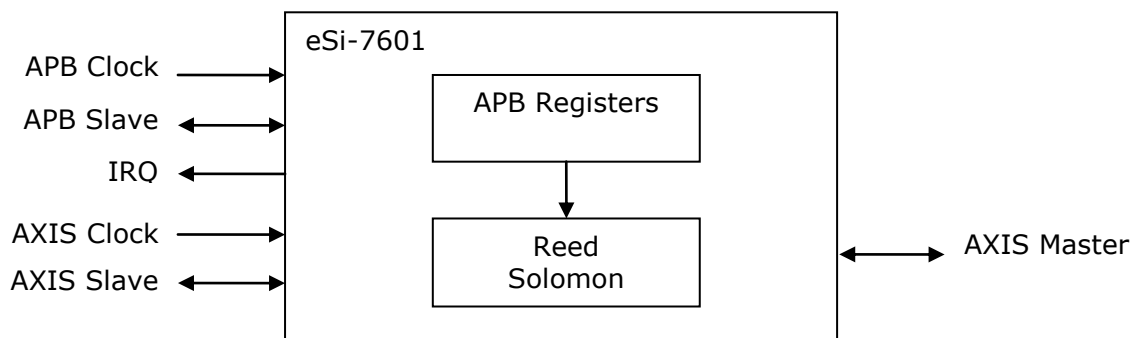
---

1	Contents	2
2	Overview	3
3	Hardware Interface	4
4	Software Interface	6
4.1	Register Map	6
4.2	Interrupts	7
5	Reed Solomon Operation	8
5.1	Interfacing	8
5.2	Cycle count	9
5.3	Gatecount	9

## 2 Overview

The eSi-7601 core provides the control and data plane interfaces to a medium throughput Reed Solomon Decoder. It supports the following features:

- Programmable generator polynomial start  $B_0$ .
- Programmable codeword length  $N$ .
- Programmable number of correctable errors  $T$ .
- Supports decoding with up to  $2T$  erasures.
- Programmable number of packets to decode before generating an interrupt.
- Continuous decode option
- Based on  $M=8$ , symbols.
- AMBA 3 APB slave control interface.
- AXI4-Stream data I/O interface



**Figure 1: eSi-7601**

The control logic handles the sequencing and control for decoding multiple codewords. This reduces the number of interrupts to the processor and significantly reduces the software overhead that would otherwise be spent on controlling the dataflow.

The default hardware generated for this Reed Solomon has extra resources to provide the full range of erasure support. If erasures are not required then the hardware usage can be significantly reduced.

The APB interface provides simple programmability to setup the decoder to support a number of different standards.

### 3 Hardware Interface

<b>Module Name</b>	esi_rs_dec
<b>HDL</b>	Verilog 2001
<b>Technology</b>	Generic
<b>Source Files</b>	esi_rs_dec.v, rs_dec_apb.v, rs_syndrome.v, rs_lamda.v, rs_dec.v, rs_fifo.v, sp_ram.v, gf_mult_include.v, rs_dec_include.v, gf_exp_include.v, gf_inv_include.v, axis_adapter.v

Parameter	Range	Default	Description
apb_address_width	3-16	16	APB address bus width
apb_data_width	16/32	16	APB data bus width (same as BITS)
t_width	1-4	4	T register width
n_width	2-8	8	N register width
b0_width	1-2	1	B0 register width
tmax	1-8	8	tmax is the maximum number of correctable errors. $2^{*t_{max}}$ is the maximum number of correctable erasures
id_width	1-8	1	Channel ID width

**Table 1: Parameters**

APB Port	Direction	Width	Description
pclk	Input	1	Clock
presetn	Input	1	Reset, active-low
paddr	Input	8	Address
psel	Input	1	Slave select
penable	Input	1	Enable
pwrite	Input	1	Write
pwdata	Input	BITS	Write data
pclk_cactive	Output	1	Clock active
pready	Output	1	Ready
prdata	Output	BITS	Read data
pslverr	Output	1	Slave error
interrupt_n	Output	1	Interrupt request, active-low

**Table 2: APB I/O Ports**

AXI4S Port	Direction	Width	Description
<code>aclk</code>	Input	1	Clock
<code>aresetn</code>	Input	1	Reset, active-low
<code>s_tdata</code>	Input	<code>data_width</code>	Slave data in
<code>s_tid</code>	Input	<code>id_width</code>	Slave id
<code>s_tlast</code>	Input	1	Slave last
<code>s_tuser</code>	Input	1	Slave user (erasure indication)
<code>s_tvalid</code>	Input	1	Slave valid
<code>s_tready</code>	Output	1	Slave ready
<code>m_tdata</code>	Output	<code>data_width</code>	Master data
<code>m_tid</code>	Output	<code>id_width</code>	Master id
<code>m_tlast</code>	Output	1	Master last
<code>m_tuser</code>	Output	1	Master user (decode fail indication)
<code>m_tvalid</code>	Output	1	Master valid
<code>m_tready</code>	Input	1	Master ready
<code>aclk_cactive</code>	Output	1	Clock active

**Table 3: AXI4-Stream I/O Ports**

For complete details of the APB and AXI4-Stream signals, please refer to the AMBA Protocol specifications available at <http://www.arm.com/products/solutions/AMBAHomePage.html>

The APB clock `pclk` and AXI clock `aclk` must be synchronous, but do not have to be the same frequency. This enables the APB interface to be run from a slower clock than the AXI interfaces.

The `aresetn` and `presetn` are used internally as active low synchronous resets. These reset signals may be asserted asynchronous to the clocks but deassertion must be synchronous after the rising edge of their respective clock.

The core may also be used without an APB interface by instantiating the file `rs_dec.v` as the top level and providing signals `trigger`, `codeword_length`, `t` and `b0`.

## 4 Software Interface

### 4.1 Register Map

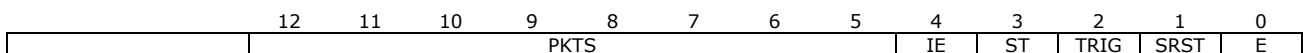
The software register map is given below

Register	Address offset	Access	Description
control	0x00	R/W	Control register
status	0x04	R/W	Status register
T	0x08	R/W	Correctable errors register
N	0x0C	R/W	Codeword length register
B0	0x10	R/W	Generator start register

**Table 4: Register Map**

#### 4.1.1 Control Register

The control register contains a selection of flags that control the operation of the module.



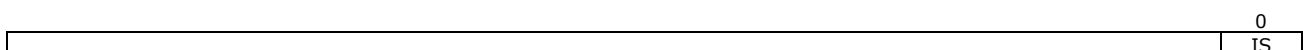
**Figure 2: Format of the control register**

Register	Values	Description
E	0 - Disabled 1 - Enabled	Enables the Reed Solomon Decoder. When disabled data will not be processed.
SRST	0 - Normal operation 1 - Synchronous reset	Synchronous reset of Reed Solomon Decoder and AXI4 streaming interface
TRIG	1 - Trigger	Write 1 to trigger decoding of PKTS packets. This field is self clearing. Reed Solomon decoding stops after PKTS packets are processed.
ST	0 - Packet based 1 - Continuous stream	Writing 0 to this register allow packet based control over the Reed Solomon decoder using the TRIG and PKTS field. Writing 1 causes the Reed Solomon decoder to continuously decode packets as soon as they become available.
IE	0 - Disable interrupts 1 - Enable interrupts	Writing 1 will generate an interrupt on interrupt_n once PKTS packets have been processed.
PKTS	1-255	Number of packets to decode before stopping

**Table 5: Fields of the control register**

#### 4.1.2 Status Register

The status register contains the interrupt bit. To clear a bit in the status register, write a 1 to it. Writing a 0 will leave it unchanged.



**Figure 3: Format of the status register**

Register	Values	Description
IS	0 - Interrupt not set 1 - Interrupt set	Interrupt status. This bit is set to 1 once PKTS packets have been processed. It is set independent of the IE field to allow software polling instead of hardware interrupt generation.

**Table 6: Fields of the status register**

### 4.1.3 T Register

The T register holds the maximum number of correctable errors in a codeword. A Reed Solomon codeword comprises a variable length message followed by  $2T$  parity symbols. A codeword is correctable if the number of errors,  $E$ , does not exceed  $T$ . When supporting up to  $e$  erasures then the following formula applies to determine if a codeword is correctable;  $2e + E \leq 2T$ . This register should only be written to when the Reed Solomon core is idle, for example after a packet based interrupt.

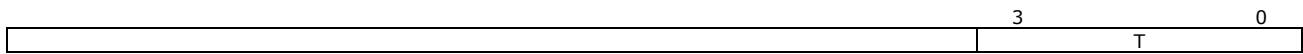


Figure 4: Format of the T register when  $t\_width=4$

### 4.1.4 N Register

The N register holds the codeword length. The message length is determined from  $N-2T$ . This register should only be written to when the Reed Solomon core is idle, for example after a packet based interrupt.

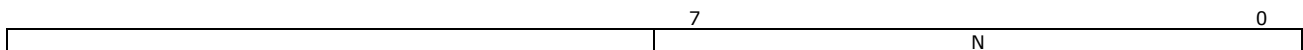


Figure 5: Format of the N register when  $n\_width=8$

### 4.1.5 B0 Register

The generator polynomial  $g(x)$  has  $2T$  roots given by successive powers of  $\alpha$  starting with power  $B0$ . The most common values of  $B0$  are 0 and 1. This Reed Solomon decoder uses a fixed value of  $\alpha = 2$ .

$$g(x) = \prod_{i=0}^{2T-1} x - \alpha^{(B0+i)}$$

This register should only be written to when the Reed Solomon core is idle, for example after a packet based interrupt.

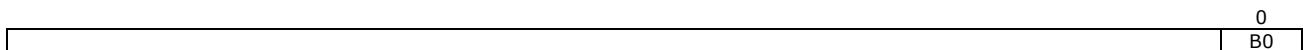


Figure 6: Format of the B0 register when  $b0\_width=1$

## 4.2 Interrupts

The `interrupt_n` signal will be raised when `PKTS` packets have been processed and the `IE` flag in the control register is set to 1. The interrupt status bit `IS` is set when `PKTS` packets have been processed. The `IS` bit is set independent of the `IE` flag setting. The `IS` bit and `interrupt_n` signal are both cleared by writing a 1 to the status register. The receive interrupt operation and `IS` flag are also active in streaming mode  $ST=1$ , but are not intended to be used in streaming mode.

## 5 Reed Solomon Operation

This Reed Solomon uses a mixed serial/parallel hardware implementation to minimise gate count and memory requirements. It can be used for medium throughput applications and requires a maximum clock frequency approximately 3x the decoded bit rate for maximum length codewords.

### 5.1 Interfacing

The core uses a full key size GF multiplier for the syndrome decoding followed by shared GF multipliers for the remaining decoding operations. The symbols may be applied on `s_tdata` at the maximum rate of one per clock cycle.

The slave AXI4-Stream input interface and master AXI4-Stream output interface obey a handshaking protocol using the `x_tvalid` and `x_tready` signals. The following description is taken from the AXI4-Stream specification v1.0. The `x_tvalid` and `x_tready` handshake determines when information is passed across the interface. A two-way flow control mechanism enables both the master and slave to control the rate at which the data and control information is transmitted across the interface. For a transfer to occur both the `x_tvalid` and `x_tready` signals must be asserted. Either `x_tvalid` or `x_tready` can be asserted first or both can be asserted in the same `aclk` cycle. A master is not permitted to wait until `m_tready` is asserted before asserting `m_tvalid`. Once `x_tvalid` is asserted it must remain asserted until the handshake occurs. A slave is permitted to wait for `s_tvalid` to be asserted before asserting the corresponding `s_tready`. If a slave asserts `s_tready`, it is permitted to deassert `s_tready` before `s_tvalid` is asserted.

The final symbol of a codeword must have `s_tlast` asserted on the slave handshake. The codeword length `N` programmed through APB and the occurrence of `s_tlast` should be consistent or undefined operation may result.

Once a full codeword is read in on the slave interface then the `s_tready` signal will be deasserted by the Reed Solomon decoder and no further data may be applied until the master interface has transferred all the decoded data.

An erasure may be signalled by asserting `s_tuser` on the same slave handshake as the erased symbol.

The stream identifier `s_tid` and `m_tid` are unused internally, but may be used to identify a stream for subsequent processing. `m_tid` will reflect the value on `s_tid` during the last codeword symbol handshake.

The decoded data provided on the master interface will indicate the last symbol of the message by asserting `m_tlast`. The master interface supports backpressure from a downstream module through the `m_tready` input as described above. The signal `m_tuser` indicates the decoder failed to correct all the errors in the codeword. Because this signal is unreliable when used with erasures it is currently not implemented but may be connected up for future use. The decoder fail signal also requires decoding of the whole codeword rather than stopping after the message.

The Reed Solomon uses one single port 8x256 bit synchronous SRAM to store the incoming codeword for correction.

The AXI interfacing is summarised in the waveform plot below. The input data is presented one symbol per clock cycle using the ready/valid handshake. Corrected symbols are presented without any active backpressure from the testbench to simplify the explanation. This plot was generated for `N=9`, `B0=0`, `T=1` and a 100MHz `aclk`, 50MHz `pclk`. It shows the

