



### Features

- Constraint length 7
- Nominal code rate  $\frac{1}{2}$
- Punctured rates  $\frac{2}{3}$ ,  $\frac{3}{4}$ ,  $\frac{5}{6}$ ,  $\frac{7}{8}$
- Simple handshake protocol for reliable interfacing
- Fully synchronous design
- Very high speed operation
- Comprehensive verification plan provided

### General Description

The SALxx300E consists of verilog IP for implementing the rate  $\frac{1}{2}$  constraint length 7 transparent convolutional code which is well suited to channels with predominantly Gaussian noise. The device supports the nominal rate  $\frac{1}{2}$  code as well as punctured rates  $\frac{2}{3}$ ,  $\frac{3}{4}$ ,  $\frac{5}{6}$ , and  $\frac{7}{8}$ .

The encoder is of the non-systematic non-recursive type, with user-defined connection polynomials (default:  $g_1 = 171$  and  $g_2 = 133$ ).

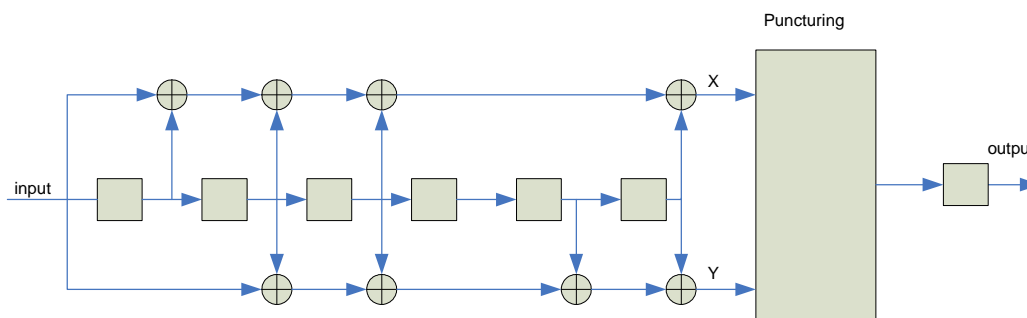


Figure 1: Encoder with default connections

## Theory of Operation

The SALxx300E is a 64-state nominal rate 1/2 (7,1/2) convolutional encoder. Its operation is very simple: a serial stream of data is shifted into the device, and an encoded serial data stream is shifted out of the device. Refer to figure 1. If the *punct* bit is not set, then puncturing is not enabled and the code is a rate 1/2 code, i.e., 2 bits are shifted out of the encoder for every bit shifted into the encoder. The bits are output in the order X(1), Y(1), X(2), Y(2), etc.

## Punctured output

If the *punct* bit is set, then some of the bits are removed from the nominal rate 1/2 output by a technique known as puncturing to achieve a higher code rate. Available code rates are 2/3, 3/4, 5/6, and 7/8. Although these rates make more efficient use of the available channel bandwidth, they suffer in their error correcting performance due to the removal of some redundant bits in the output. The puncture patterns are shown below:

Puncture pattern 1 = transmit 0 = don't transmit	Code Rate	Output Sequence
X: 1 0 Y: 1 1	2/3	X(1), Y(1), Y(2), ...
X: 1 0 1 Y: 1 1 0	3/4	X(1), Y(1), Y(2), X(3), ...
X: 1 0 1 0 1 Y: 1 1 0 1 0	5/6	X(1), Y(1), Y(2), X(3), Y(4), X(5), ...
X: 1 0 0 0 1 0 1 Y: 1 1 1 1 0 1 0	7/8	X(1), Y(1), Y(2), Y(3), Y(4), X(5), Y(6), X(7), ...

Table 1. Puncture patterns

## Signal Descriptions

The module pinout is shown in the figure below, and in table 1. The signals are conveniently organized into functional groups as follows:

### Clock and Reset

The design is fully synchronous with a single clock signal. The reset signal is synchronous and needs to be asserted for at least one full clock cycle to reset internal logic.

### Control signals

Two signals control flow of data into the device, *din\_rdyin\_n* and *din\_rdyout\_n*. The *din\_rdyin\_n* signal indicates that data into the device is valid. The *din\_rdyout\_n* signal indicates that the device is ready to receive data.

Two signals control flow of data out of the device, *dout\_rdyout\_n* and *dout\_rdyin\_n*. The *dout\_rdyout\_n* signal indicates that data out of the device is valid. The *dout\_rdyin\_n* signal indicates to the device that it's OK to shift data out of the device.

### Data signals

The data are clocked in on *din* and clocked out on *dout*.

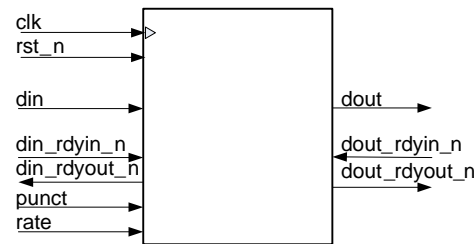


Figure 2: Component pinout

Pin	Sense	Width	Description
clk	in	1	Clock
rst_n	in	1	Synchronous reset
init	in	1	Soft reset signal
punct	in	1	When set, code is punctured
rate	in	2	Code rate: 00 = 2/3, 01 = 3/4, 10 = 5/6, 11 = 7/8
din	in	1	Serial data (message) in
din_rdyin_n	in	1	Indicates to the device that serial data in is valid
din_rdyout_n	out	1	Indicates the device is ready to receive data in
dout	out	1	Data out
dout_rdyin_n	in	1	Indicates to the device that it can shift data out
dout_rdyout_n	out	1	Indicates that data is available to be shifted out

Table 2. Component pinout

## Waveforms

### Input

The input functional timing is shown below. *Din\_rdyin\_n* is used as an input data enable, *din\_rdyout\_n* is used to indicate when the device is ready to receive data.

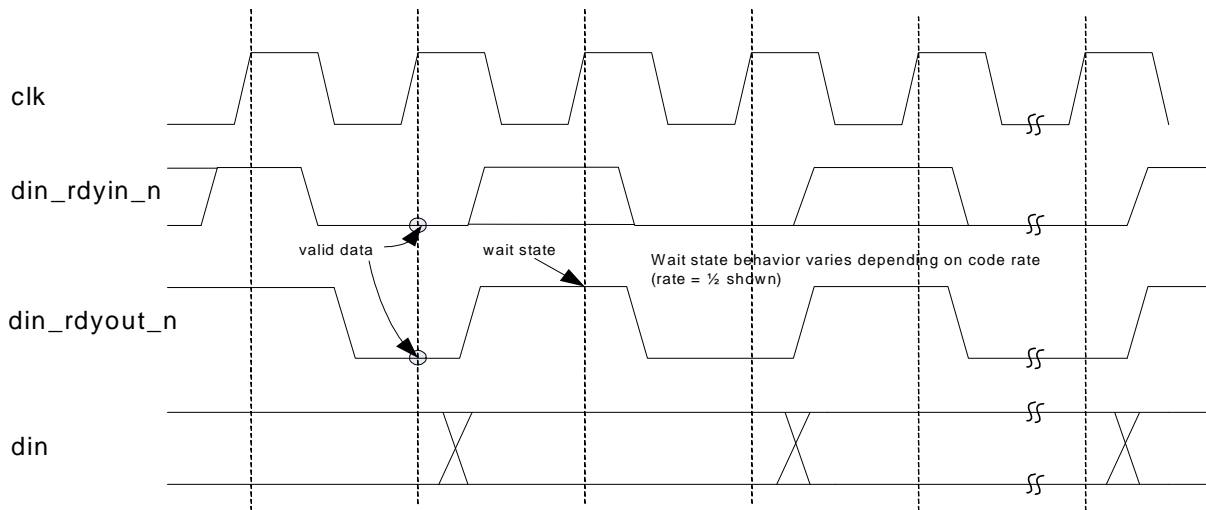


Figure 3: Input timing

### Output

The output functional timing is shown below. *Dout\_rdyout\_n* is used as an output data ready indication, *dout\_rdyin\_n* is used to indicate to the device that it's OK to shift data out.

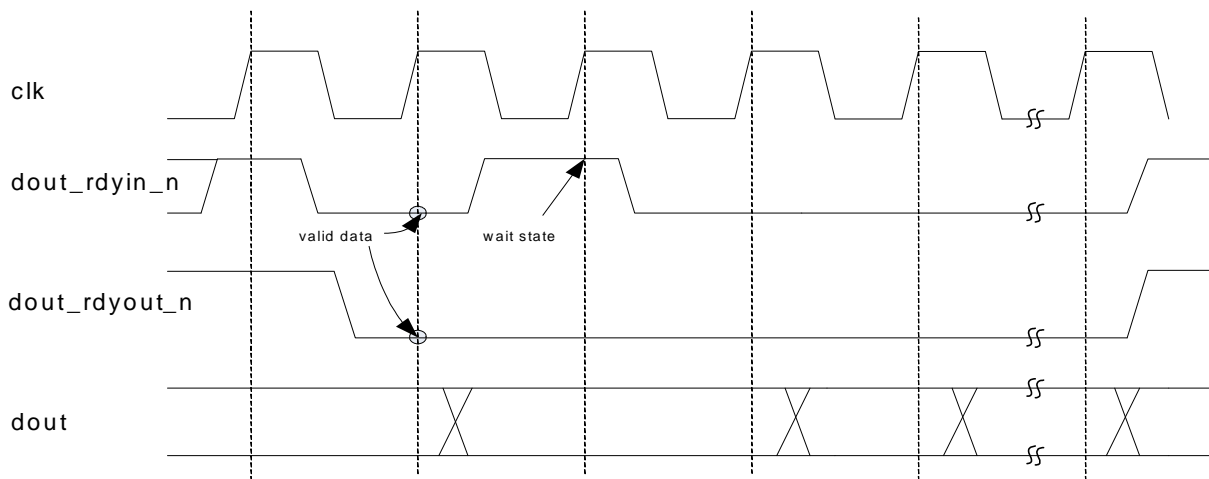


Figure 4. Output timing

## Module Verification

The SALxx300E has been subjected to extensive verification to ensure the highest quality product possible. A comprehensive test plan was implemented which included the following:

- High-quality random data source
- High-quality random noise source
- Extensive flow-control simulations
- Verification of operation against known data sequences

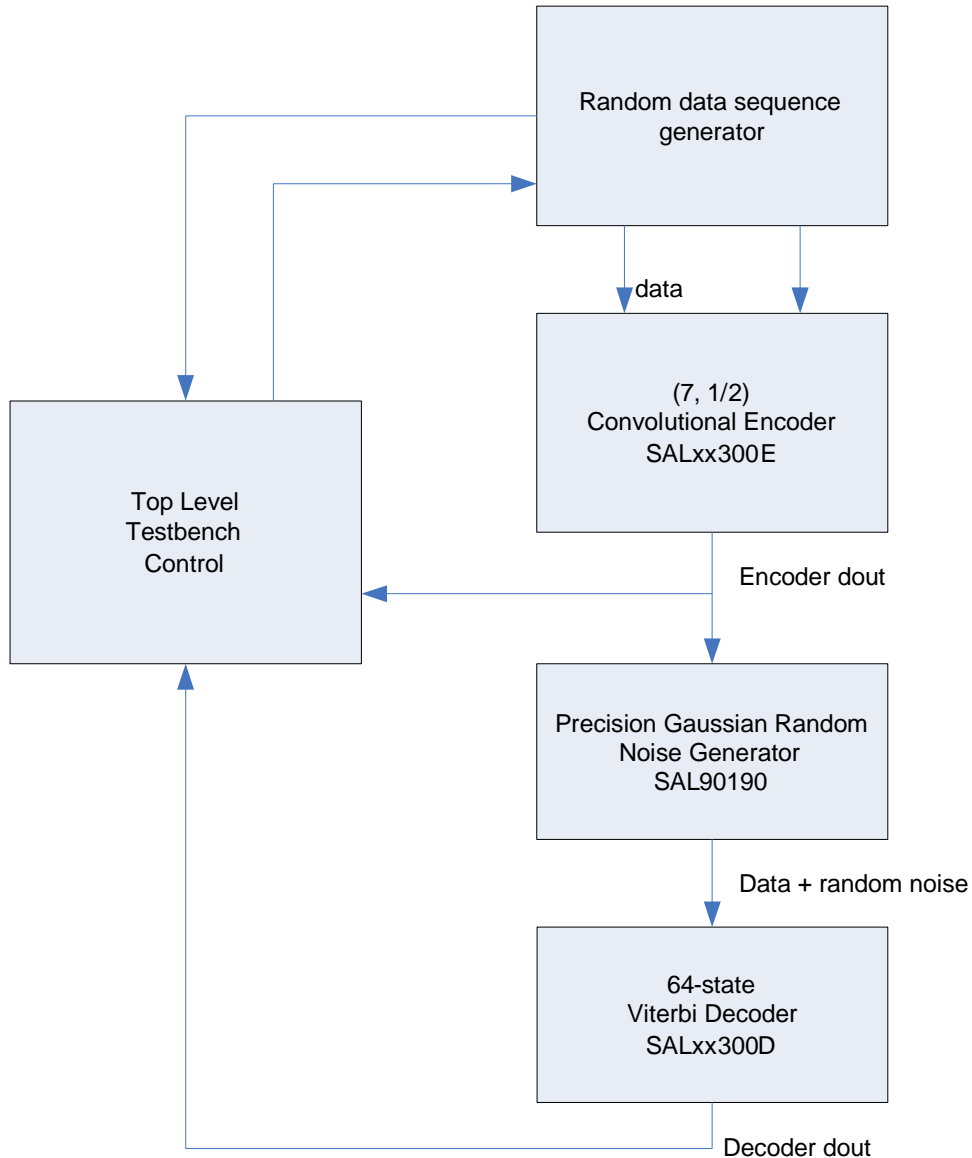
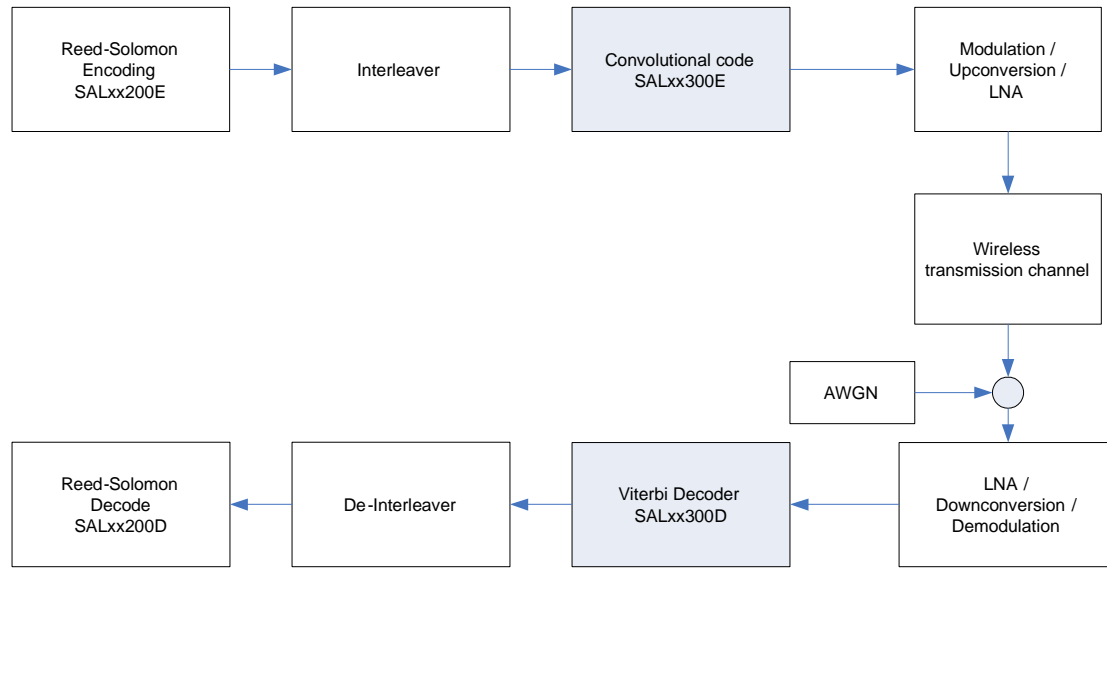


Figure 5: Testbench Block Diagram

## Application: Wireless Internet System

The (7, 1/2) convolutional code forms an integral part of a wireless Internet telemetry system.



## Ordering Information

Salamander Error Correction currently has 2 convolutional encoder IP modules available:

**SALxx300E** (7, 1/2) convolutional encoder

**SALxx320E** (9, 1/2) convolutional encoder

## About Salamander:

**Salamander Error Correction** develops and sells error correction modules of the highest quality worldwide.

Salamander Error Correction is a division of Komodo Industries, Inc.

Salamander Error Correction:  
5330 Carroll Canyon Rd  
San Diego, CA 92121  
(858) 373-2112  
fax: (858) 373-1224  
[www.salamander-ecc.com](http://www.salamander-ecc.com)  
[sales@salamander-ecc.com](mailto:sales@salamander-ecc.com)