



APPLICATION NOTE

Using newWaveX—Waveform Testing

Verifying a waveform fits within a specified envelope

Ashley Hulme

Application Note 1004/ Issue 2, January 2010

Introduction

At a recent 1641 presentation a question was asked about how it was possible to express a test in which a waveform was to be checked. The test was to verify that a waveform remained within a specified envelope. The waveform itself was not specified, only the limits (envelope) which it must not cross.

Basic solution

This can be achieved by defining the upper and lower bounds as waveforms. Either WaveformRamp or WaveformStep may be used. The initial example uses WaveformRamp as it is easier to use for an example with few points.

The input waveform is subtracted from the upper envelope, and if the waveform remains below this envelope the output will be zero or negative over the measurement period.

The input waveform is also subtracted from the lower envelope, and if the waveform remains above this envelope the output will be zero or positive over the measurement period.

If the waveform goes outside the envelope over the period of the measurement, an event will be raised which indicates the point at which the error occurred. This can be shown graphically using newWaveX (see figure 1).

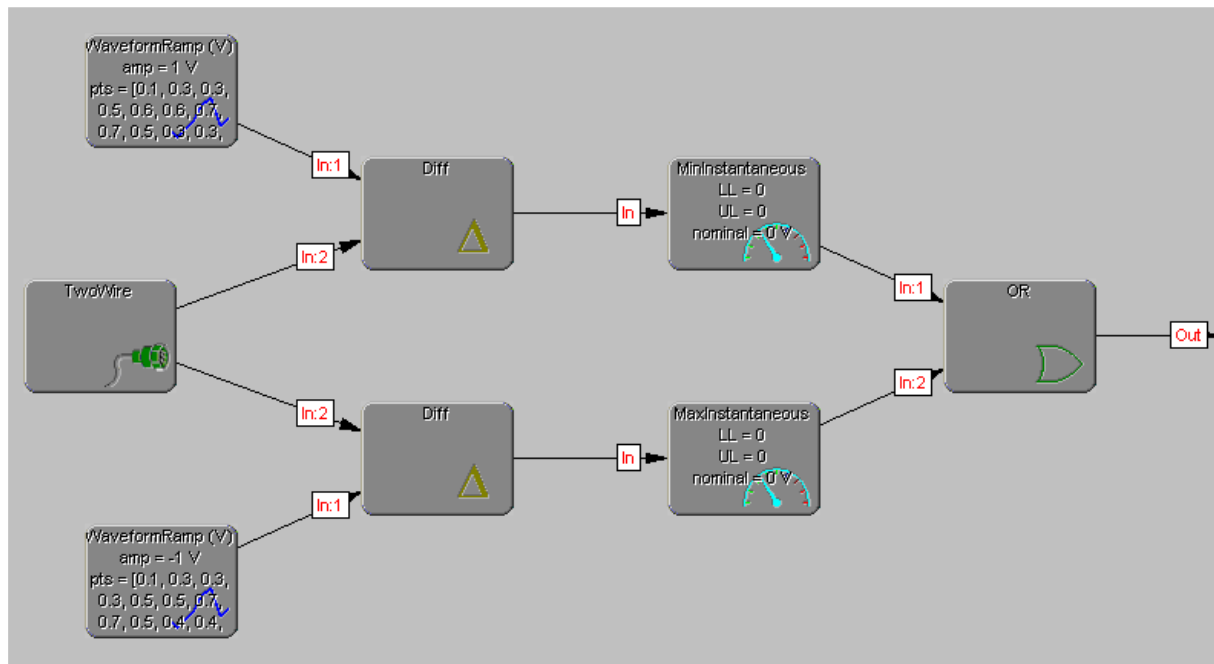


Figure 1—Signal graph of measurement

This can be presented in XML as follows:

```
<Signal name="Waveform-out-of-envelope-Detector" Out="Any_Event_is_Fail">
  <WaveformRamp name="UpperWaveformEnvelope" amplitude="1 V"
    points="[0.1, 0.3, 0.3, 0.5, 0.6, 0.6, 0.7, 0.7, 0.5, 0.3, 0.3, 0.1]" />
  <WaveformRamp name="LowerWaveformEnvelope" amplitude="-1 V"
    points="[0.1, 0.3, 0.3, 0.3, 0.5, 0.5, 0.7, 0.7, 0.5, 0.4, 0.4, 0.1]" />
  <TwoWire name="SignalInput" channelWidth="1" />
  <Diff name="CompareSignal_to_UpperEnvelope"
    In="UpperWaveformEnvelope SignalInput" />
  <Diff name="CompareSignal_to_LowerEnvelope"
    In="LowerWaveformEnvelope SignalInput" />
  <MinInstantaneous name="LTzero_is_Fail" nominal="0 V" condition="LT"
    In="CompareSignal_to_UpperEnvelope" />
  <MaxInstantaneous name="GTzero_is_Fail" nominal="0 V" condition="GT"
    In="CompareSignal_to_LowerEnvelope" />
</Signal>
```

```
<OrEvent name="Any_Event_is_Fail" In="LTzero_is_Fail GTzero_is_Fail" />
</Signal>
```

Figure 2 shows the upper and lower boundaries for the signal under test. This is represented in the XML by the two WaveformRamp BSCs "UpperWaveformEnvelope" and "LowerWaveformEnvelope":

```
<WaveformRamp name="UpperWaveformEnvelope" amplitude="1 V"
  points="[0.1, 0.3, 0.3, 0.5, 0.6, 0.6, 0.7, 0.7, 0.5, 0.3, 0.3, 0.1]" />
<WaveformRamp name="LowerWaveformEnvelope" amplitude="-1 V"
  points="[0.1, 0.3, 0.3, 0.3, 0.5, 0.5, 0.7, 0.7, 0.5, 0.4, 0.4, 0.1]" />
```

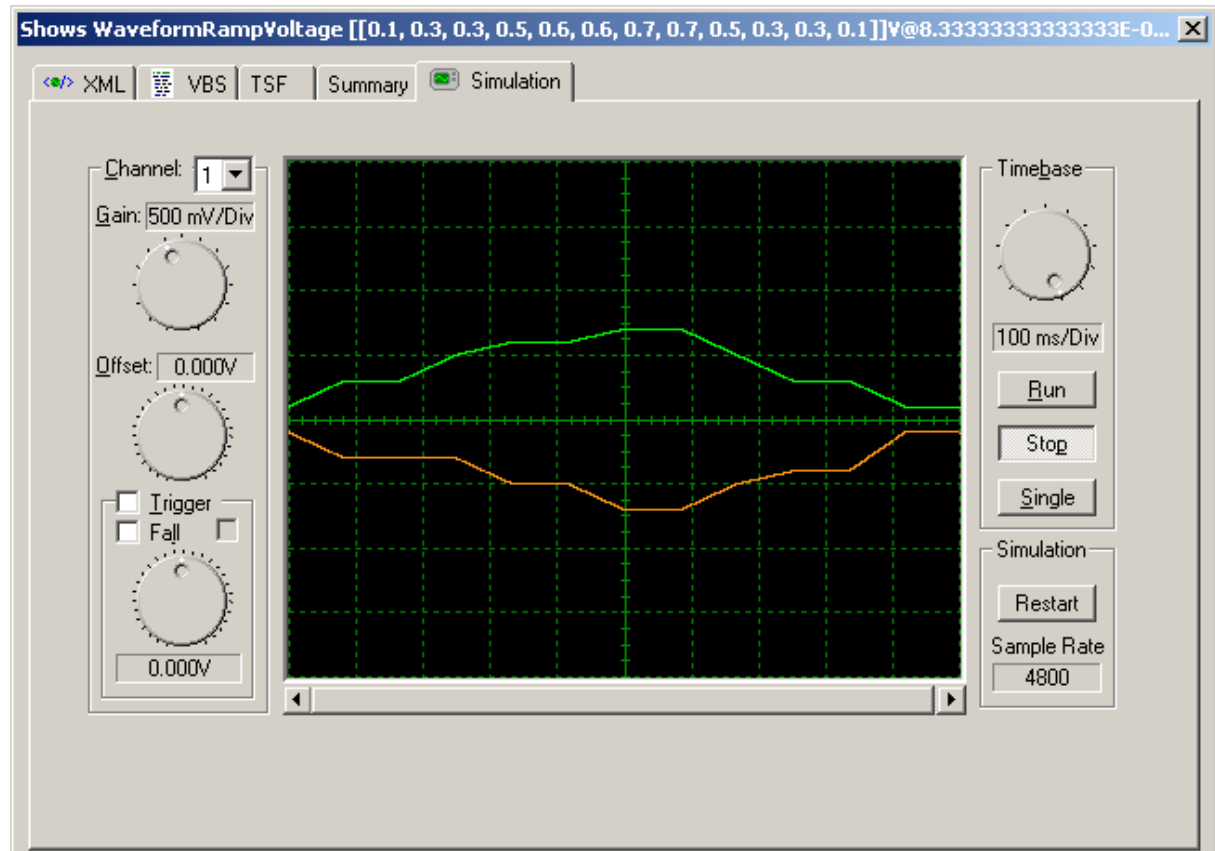


Figure 2—Test envelope for waveform

The next figure (figure 3) shows a waveform which is to be checked to see that it falls within the envelope. The envelope is also shown for comparison. It may be seen that the waveform exceeds the envelope at a point on the lower boundary. This waveform is input via the TwoWire connection BSC "SignalInput".

The input waveform is subtracted from the defined boundaries by the two Diff BSCs. The following MinInstantaneous (and MaxInstantaneous) BSCs are used to raise an event if the waveform crosses the boundary. This is shown in the following XML; first for the upper boundary, and then for the lower boundary:

```
<Diff name="CompareSignal_to_UpperEnvelope"
  In="UpperWaveformEnvelope SignalInput" />
<MinInstantaneous name="LTzero_is_Fail" nominal="0 V" condition="LT"
  In="CompareSignal_to_UpperEnvelope" />

<Diff name="CompareSignal_to_LowerEnvelope"
  In="LowerWaveformEnvelope SignalInput" />
<MaxInstantaneous name="GTzero_is_Fail" nominal="0 V" condition="GT"
  In="CompareSignal_to_LowerEnvelope" />
```

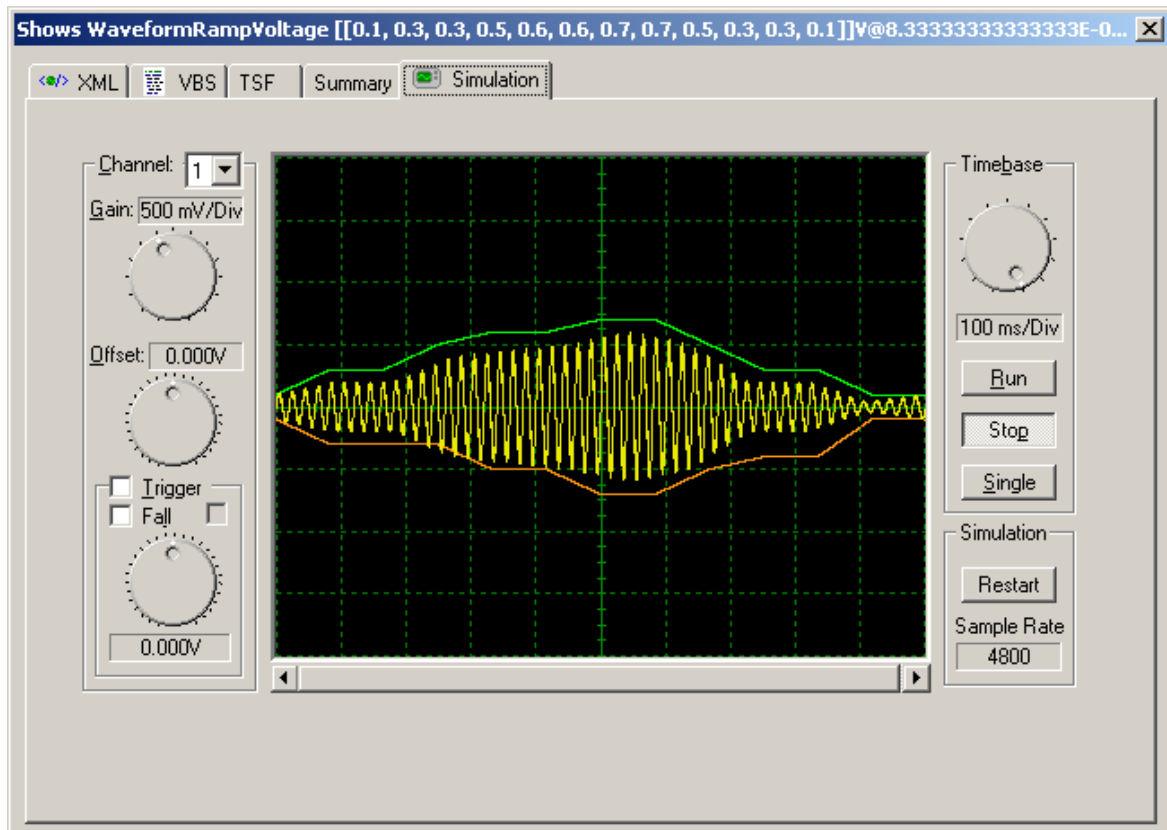


Figure 3—Test waveform

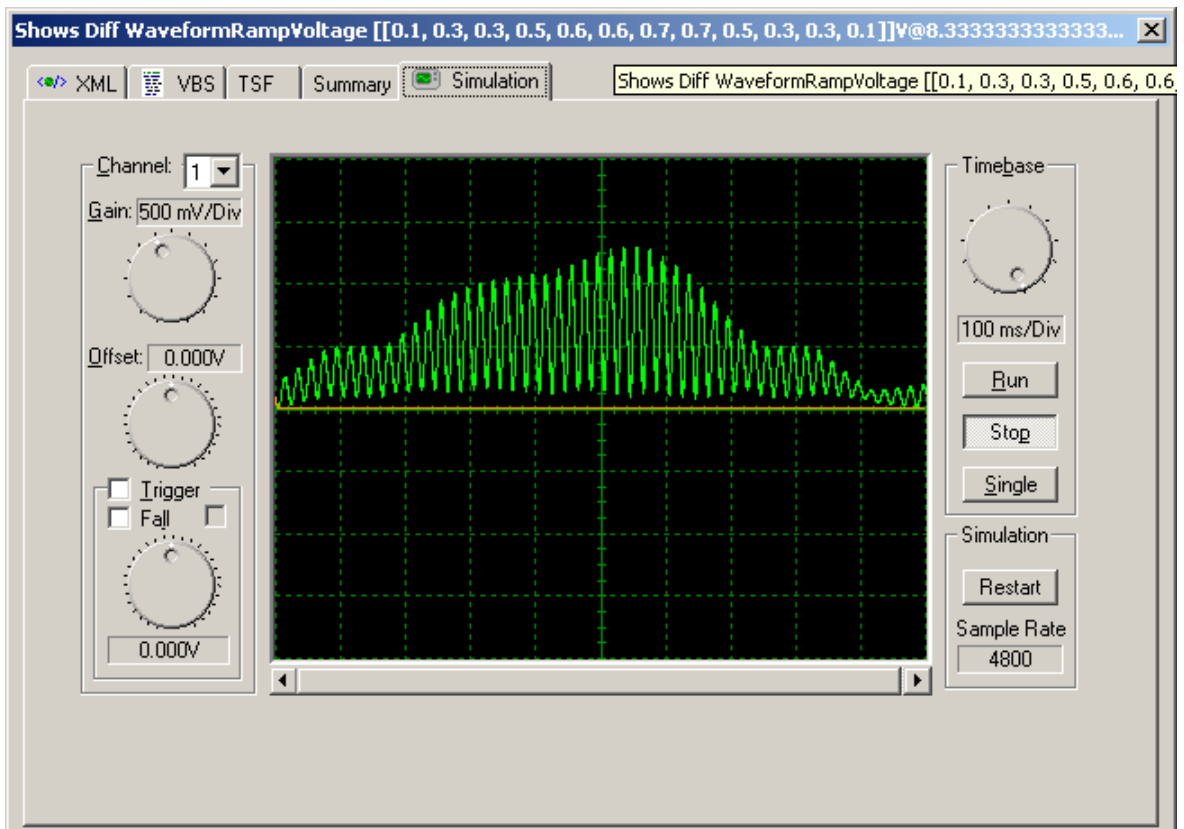


Figure 4—Upper Boundary Test

Figures 4 and 5 show the result of the comparison with the upper and lower boundaries respectively. In figure 5 it may be seen that the test fails because the waveform (after subtraction) goes positive. That is where the waveform goes below the lower boundary.

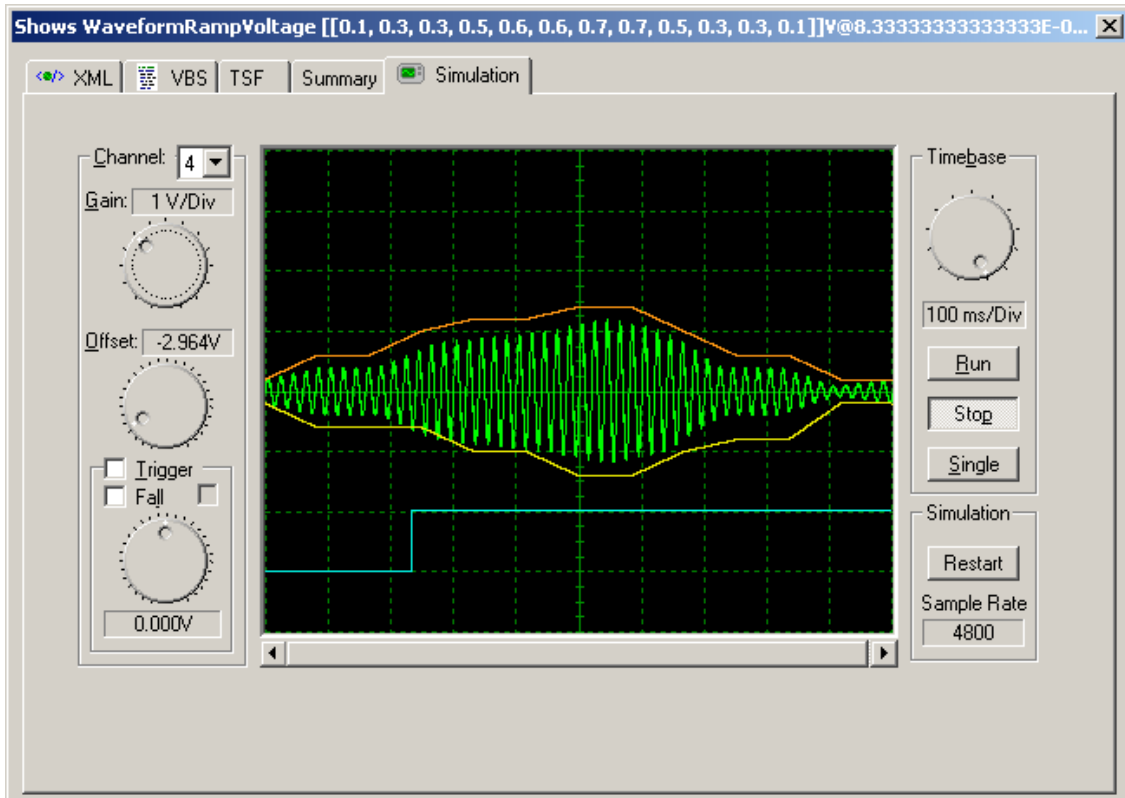


Figure 5—Lower Boundary Test

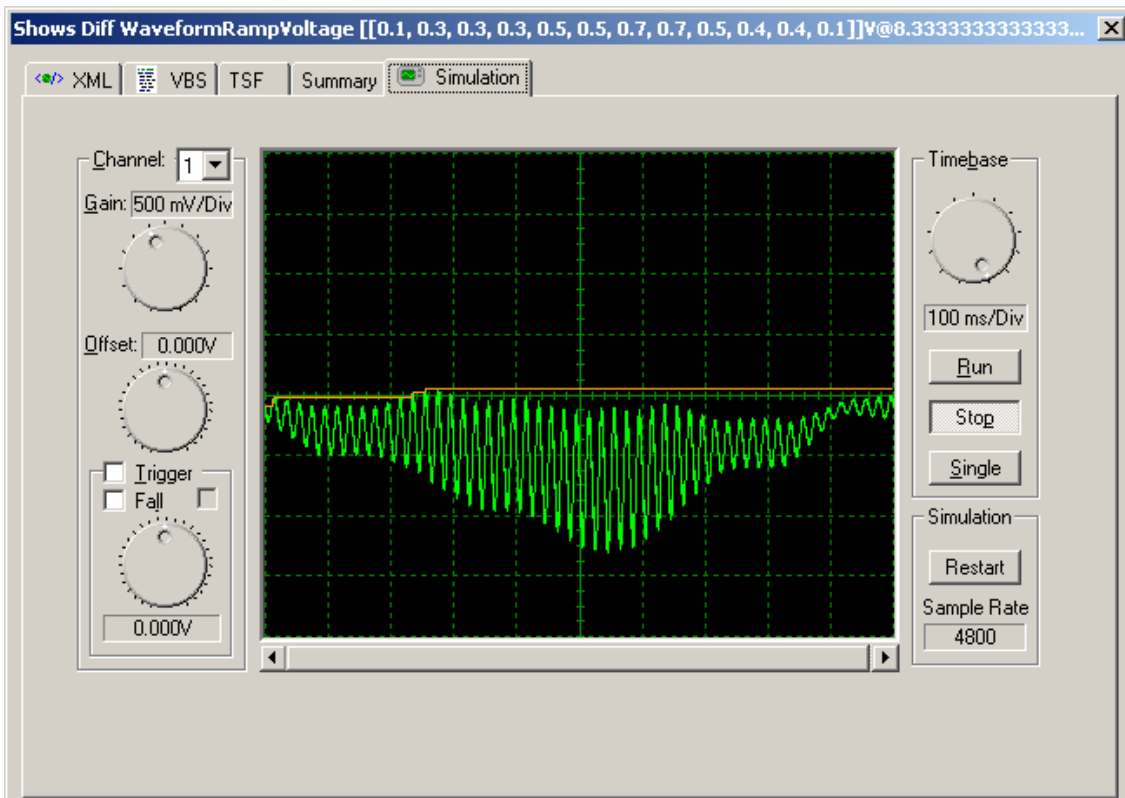


Figure 6—Test showing point of failure

The OrEvent BSC is used to detect whether either sensor BSC detects a failure. The OrEvent provides a combined event to indicate a failure.

```
<OrEvent name="Any_Event_is_Fail" In="LTzero_is_Fail GTzero_is_Fail" />
```

Figure 6 shows the failure event adjacent to the input waveform and the envelope boundaries. The event occurs at the first point that the waveform exceeds the permissible envelope which is the point of failure of the test.

It is not necessary to generate the event to complete the test. The sensor BSCs may be set with the appropriate Upper Limit (UL = 0 V) and Lower Limit (LL = 0 V) to indicate a failure. The event shows exactly where the first failure occurs in the waveform.

Using WaveformStep BSCs

Figures 7 and 8, together with the corresponding XML, show a similar test in which the envelope described with much more detailed WaveformStep boundaries. This is more in line with the way that instruments such as Arbitrary Waveform Generators operate.

The principle is the same as with the previous example.

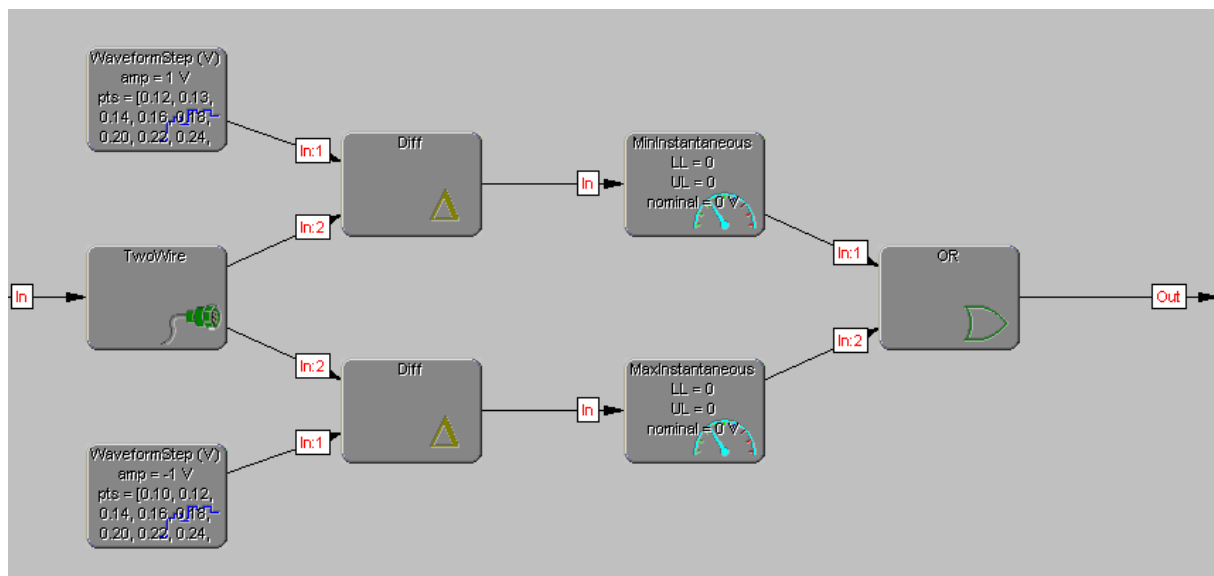


Figure 7—Signal graph for WaveformStep version

XML Description of signal:

```
<Signal name="Waveform-out-of-envelope-Detector" Out="Any_Event_is_Fail" >
  <WaveformStep name="UpperWaveformEnvelope" amplitude="1 V"
    samplingInterval="8.26446280991736E-03"
    points="[0.12, 0.13, 0.14, 0.16, 0.18, 0.20, 0.22, 0.24, 0.26, 0.28, 0.30, 0.30,
      0.30, 0.30, 0.30, 0.30, 0.30, 0.30, 0.30, 0.30, 0.30, 0.30, 0.32,
      0.34, 0.36, 0.38, 0.40, 0.42, 0.44, 0.46, 0.48, 0.50, 0.50, 0.51, 0.52,
      0.53, 0.54, 0.55, 0.56, 0.57, 0.58, 0.59, 0.60, 0.60, 0.60, 0.60, 0.60,
      0.60, 0.60, 0.60, 0.60, 0.60, 0.60, 0.60, 0.60, 0.61, 0.62, 0.63, 0.64,
      0.65, 0.66, 0.67, 0.68, 0.69, 0.70, 0.70, 0.70, 0.70, 0.70, 0.70, 0.70,
      0.70, 0.70, 0.70, 0.70, 0.70, 0.70, 0.68, 0.66, 0.64, 0.62, 0.60, 0.59,
      0.58, 0.57, 0.56, 0.55, 0.40, 0.40, 0.40, 0.40, 0.40, 0.40, 0.40, 0.40,
      0.40, 0.40, 0.40, 0.40, 0.38, 0.36, 0.34, 0.32, 0.30, 0.28, 0.26, 0.24,
      0.22, 0.20, 0.10, 0.10, 0.10, 0.10, 0.10, 0.10, 0.10, 0.10, 0.10, 0.10,
      0.10]" />
  <WaveformStep name="LowerWaveformEnvelope" amplitude="-1 V"
    samplingInterval="8.26446280991736E-03"
    points="[0.10, 0.12, 0.14, 0.16, 0.18, 0.20, 0.22, 0.24, 0.26, 0.28, 0.30, 0.30,
      0.30, 0.30, 0.30, 0.30, 0.30, 0.30, 0.30, 0.30, 0.30, 0.30, 0.30,
      0.30, 0.30, 0.30, 0.30, 0.30, 0.30, 0.30, 0.30, 0.30, 0.32, 0.34,
```

```

0.36, 0.38, 0.40, 0.42, 0.44, 0.46, 0.48, 0.50, 0.50, 0.50, 0.50, 0.50,
0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.52, 0.54, 0.56, 0.58,
0.60, 0.62, 0.64, 0.66, 0.68, 0.70, 0.70, 0.70, 0.70, 0.70, 0.70, 0.70,
0.70, 0.70, 0.70, 0.70, 0.70, 0.70, 0.69, 0.68, 0.67, 0.66, 0.65, 0.64,
0.63, 0.62, 0.61, 0.60, 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.50,
0.50, 0.50, 0.50, 0.50, 0.48, 0.46, 0.44, 0.42, 0.40, 0.38, 0.36, 0.34,
0.32, 0.30, 0.20, 0.20, 0.20, 0.20, 0.20, 0.20, 0.20, 0.20, 0.10, 0.10, 0.10, 0.10,
0.10]"/>
<TwoWire name="InputSignal" lo="Pin2" hi="Pin1" channelWidth="1" />
<Diff name="CompareSignal_to_UpperEnvelope"
  In="UpperWaveformEnvelope InputSignal" />
<Diff name="CompareSignal_to_LowerEnvelope"
  In="LowerWaveformEnvelope InputSignal" />
<MinInstantaneous name="LTzero_is_Fail" nominal="0 V" condition="LT"
  In="CompareSignal_to_UpperEnvelope" />
<MaxInstantaneous name="GTzero_is_Fail" nominal="0 V" condition="GT"
  In="CompareSignal_to_LowerEnvelope" />
<OrEvent name="Any_Event_is_Fail" In="LTzero_is_Fail GTzero_is_Fail" />
</Signal>

```

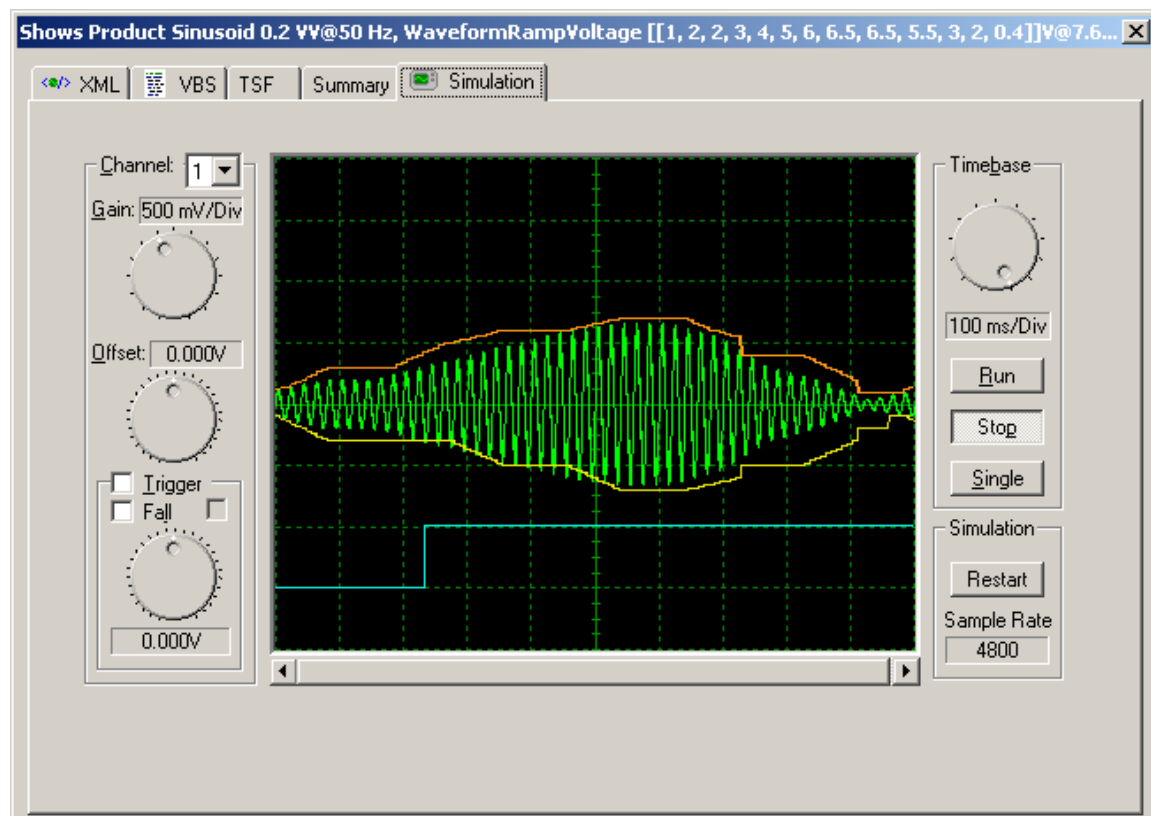


Figure 8—Waveform and permissible envelope showing point of failure

The XML for the WaveformStep BSCs show a much larger array of data defining the upper and lower waveforms which form the boundaries of the envelope. The example shown is a "fixed" envelope. The test could be converted to a TSF, in which the envelope arrays could be passed to the model as attributes.

The disadvantage with this method is that there are two parallel measurements; the MaxInstantaneous and MinInstantaneous BSCs represent the two sensors which are operating in parallel. It is possible to perform this measurement with a single sensor with upper and lower limits.

Measurement using normalized limits

This method requires more signal processing but is perhaps a more elegant solution. The waveform is measured between normalized limits using a signal graph similar to figure 9 which give limits of 0 & 1 provided that the upper limit is never the same as the lower limit.

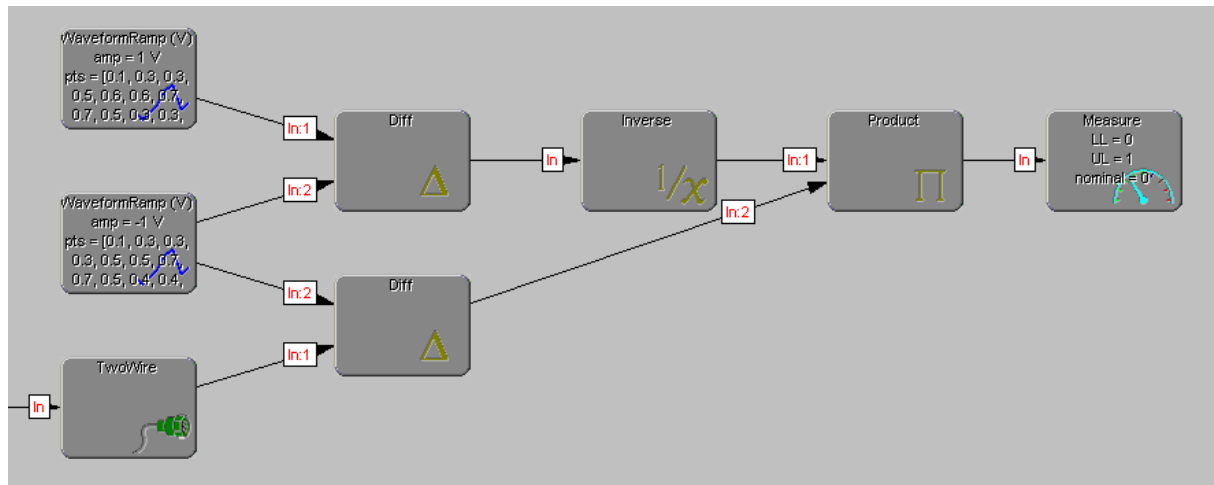


Figure 9—Signal graph of normalized measurement

Figure 10 shows the measurement task, with the unspecified waveform to be checked between the upper and lower boundaries.

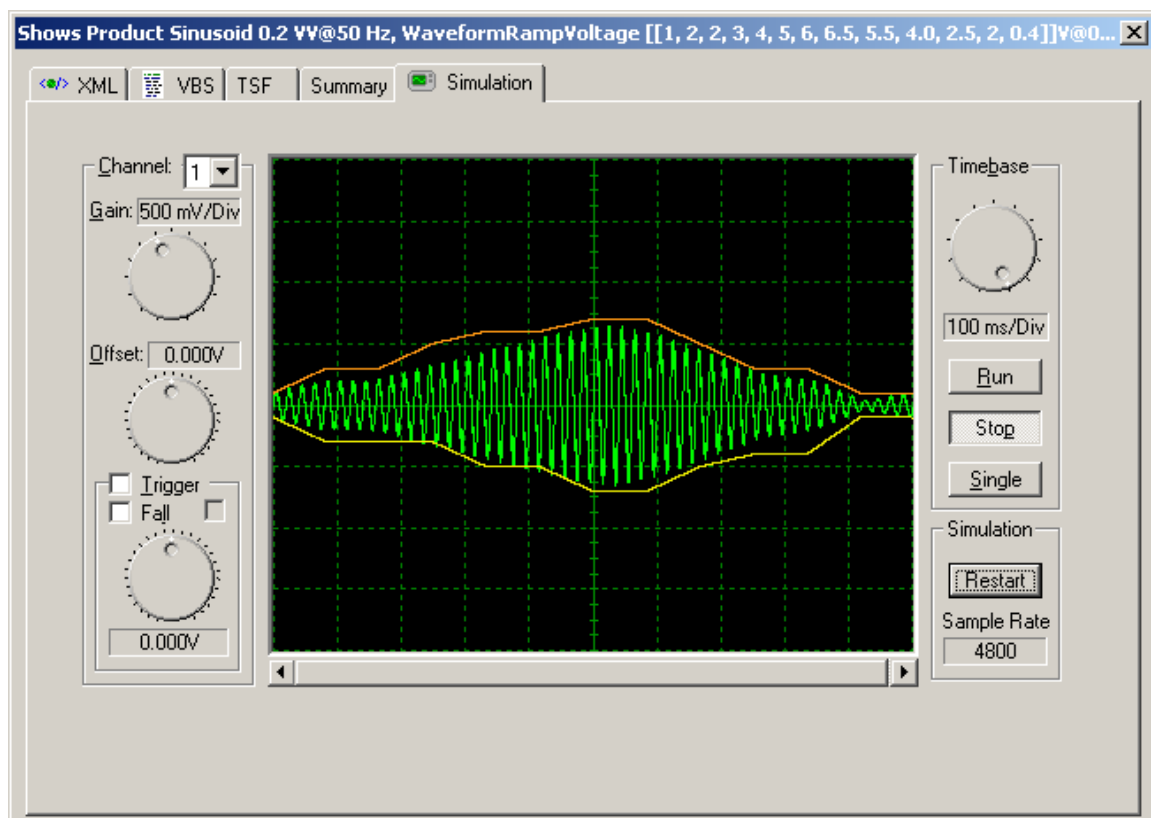


Figure 10—Measurement task; check waveform remains between boundaries

The lower boundary waveform is subtracted from the input which gives the waveform shown in figure 11 (green trace). The lower boundary waveform is subtracted from the upper boundary to give the upper boundary to the modified waveform (orange trace in figure 11).

The modified waveform is then normalized by multiplying it with the inverse of the modified upper boundary. This gives the waveform shown in figure 12 (green trace). This waveform may now be checked using a Measure BSC with upper and lower limits of zero and one respectively. If the normalized waveform goes outside of those limits then the test will have failed. The yellow and orange traces in figure 12 show the maximum excursions of the waveform. It may be seen that the

lower (orange) waveform goes negative indicating that the test has failed, because the original waveform exceeded the lower boundary of the envelope at that point.

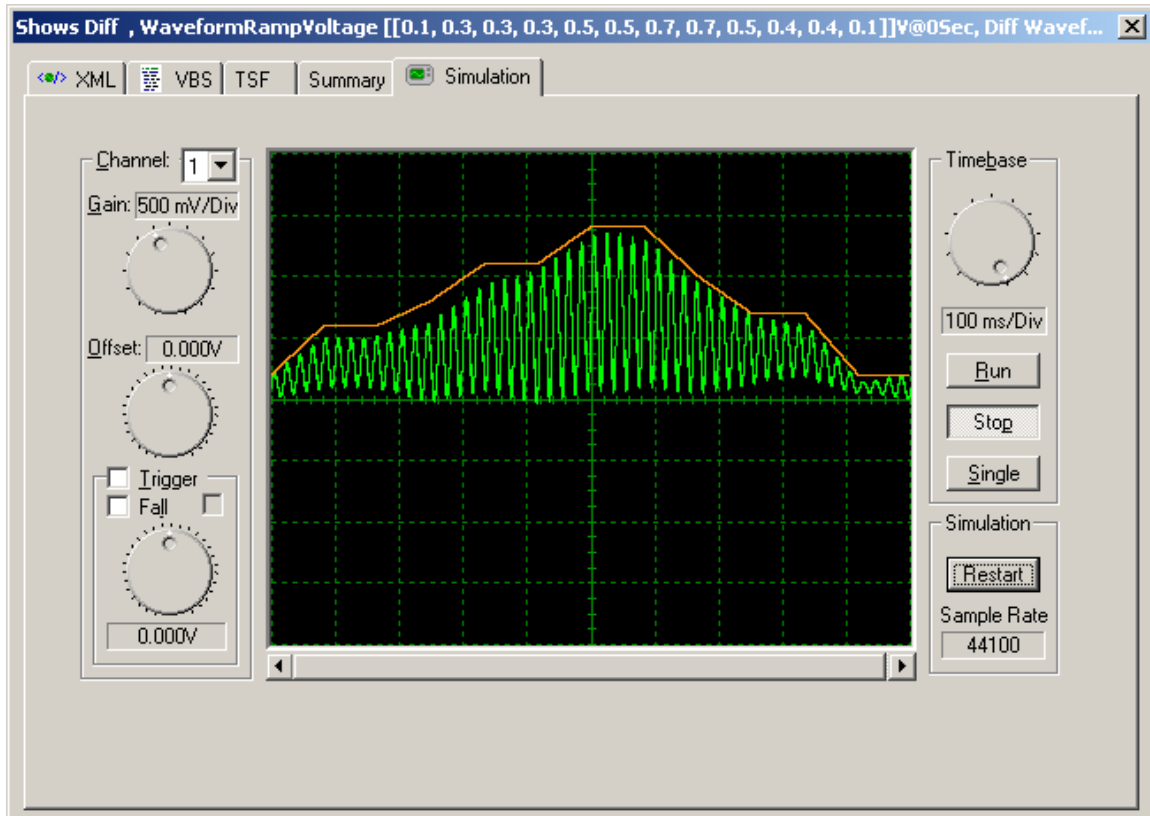


Figure 11—Output from the two "Diff" BSCs

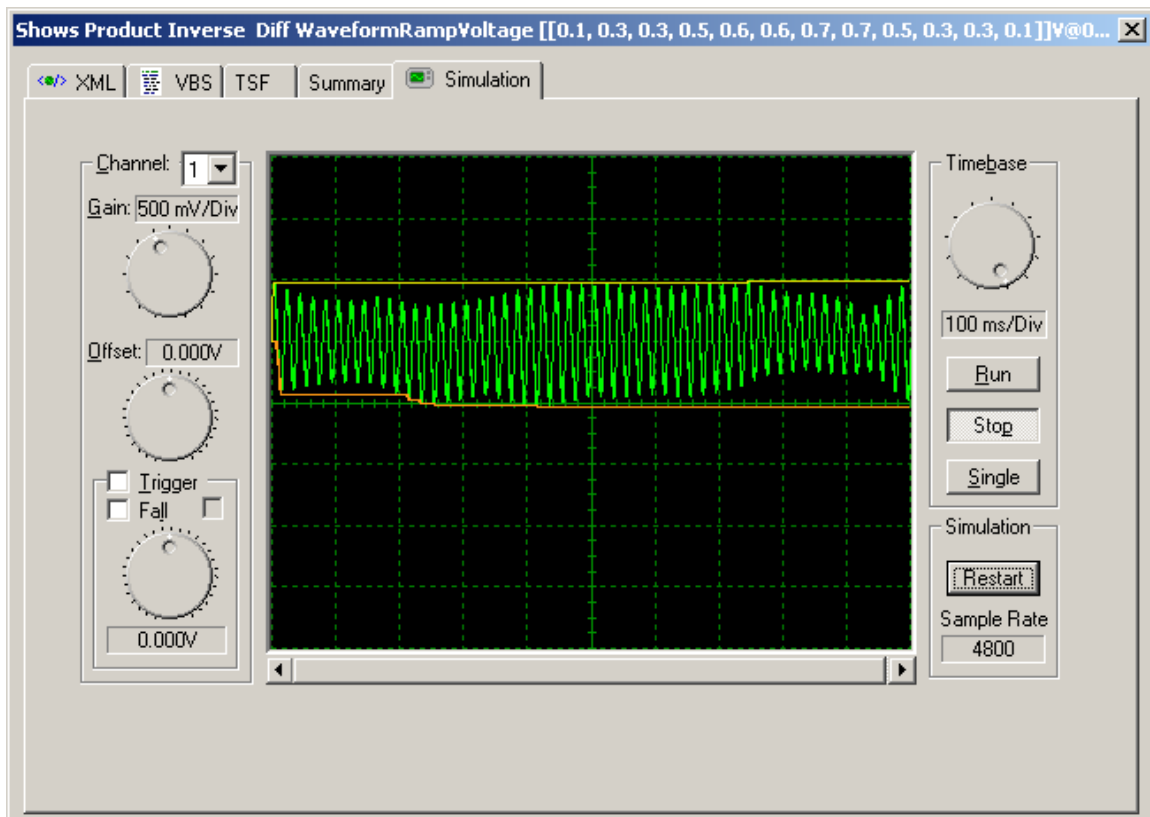


Figure 12—Normalized measurement

This test technique may be represented as a TSF. The example here is not identical to the signal graph in figure 9 as it uses WaveformStep BSCs. However, the principles are the same and the TSF may be modified by a user to represent the exact test required.

TSF to measure a normalized waveform:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- generated with newWaveX v2.8.1.3 (http://www.eads-ts.com) -->
<tsf:TSFLibrary name="MyTSFLibrary" uuid="MyGUID" version="MyVersion"
  xmlns="STDBSC" xmlns:tsf="STDTSF"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <tsf:TSF name="Waveform_out-of-envelope_Detector"
    uuid="{AF44B930-15D4-4456-8989-395DB70ADF56}">
    <tsf:interface>
      <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        elementFormDefault="qualified">
        <xs:element name="Waveform_out-of-envelope_Detector">
          <xs:annotation>
            <xs:documentation>
              <!-- 'Waveform_out-of-envelope_Detector' verifies that an incoming waveform
                fits within a specified envelope. -->
            </xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:complexContent>
              <xs:extension base="SignalFunctionType">
                <xs:attribute name="input" type="BSTR">
                  <xs:annotation>
                    <xs:documentation>
                      <!-- 'input' is the name of the connector pin to which the
                        incoming signal is connected -->
                    </xs:documentation>
                  </xs:annotation>
                </xs:attribute>
                <xs:attribute name="reference" type="BSTR">
                  <xs:annotation>
                    <xs:documentation>
                      <!-- 'reference' is the name of the connector pin to which the
                        incoming signal is referenced (usually the ground pin). -->
                    </xs:documentation>
                  </xs:annotation>
                </xs:attribute>
                <xs:attribute name="UpperBoundaryAmplitude" type="Physical">
                  <xs:annotation>
                    <xs:documentation>
                      <!-- 'UpperBoundaryAmplitude' is the nominal amplitude of the upper
                        boundary of the envelope ' -->
                    </xs:documentation>
                  </xs:annotation>
                </xs:attribute>
                <xs:attribute name="UpperBoundaryArray" type="SAFEARRAY(VARIANT)">
                  <xs:annotation>
                    <xs:documentation>
                      <!-- 'UpperBoundaryArray' is an array containing the points which describe
                        the shape of the upper boundary of the envelope. Each value in the
                        array is relative to the amplitude 'UpperBoundaryAmplitude' -->
                    </xs:documentation>
                  </xs:annotation>
                </xs:attribute>
                <xs:attribute name="LowerBoundaryAmplitude" type="Physical">
                  <xs:annotation>
                    <xs:documentation>
                      <!-- 'LowerBoundaryAmplitude' is the nominal amplitude of the lower
                        boundary of the envelope ' -->
                    </xs:documentation>
                  </xs:annotation>
                </xs:attribute>
              </xs:extension>
            </xs:complexContent>
          </xs:complexType>
        </xs:element>
      </xs:schema>
    </tsf:interface>
  </tsf:TSF>
</tsf:TSFLibrary>
```

```

</xs:annotation>
</xs:attribute>
<xs:attribute name="LowerBoundaryArray" type="SAFEARRAY(VARIANT)">
  <xs:annotation>
    <xs:documentation>
      <!-- 'LowerBoundaryArray'is an array containing the points which describe
      the shape of the lower boundary of the envelope. Each value in the
      array is relative to the amplitude 'LowerBoundaryAmplitude' -->
    </xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="measurement_gate" type="Time">
  <xs:annotation>
    <xs:documentation>
      <!-- 'measurement_gate' is the total time over which the signal is measured
      (equivalent to the "length" of the envelope) -->
    </xs:documentation>
  </xs:annotation>
</xs:attribute>
</xs:extension>
</xs:complexType>
</xs:element>
</xs:schema>
</tsf:interface>
<tsf:model>
  <Signal Out="Normalized_Waveform_Measurement">
    <TwoWire name="Signal_Input" lo="(reference)" hi="(input)" channelWidth="1" />
    <WaveformStep name="UpperWaveformEnvelope"
      amplitude="(UpperBoundaryAmplitude)" period="(measurement_gate)"
      samplingInterval="0s" points="[UpperBoundaryArray]" />
    <WaveformStep name="LowerWaveformEnvelope"
      amplitude="(LowerBoundaryAmplitude)" period="(measurement_gate)"
      samplingInterval="0s" points="[LowerBoundaryArray]" />
    <Diff name="Signal_Difference" In="Signal_Input LowerWaveformEnvelope" />
    <Diff name="Boundary_Difference"
      In="UpperWaveformEnvelope LowerWaveformEnvelope" />
    <Inverse name="Boundary_Inversion" In="Boundary_Difference" />
    <Product name="Normalized_Signal"
      In="Boundary_Inversion Signal_Difference" />
    <Measure name="Normalized_Waveform_Measurement"
      gateTime="(measurement_gate)" UL="1" LL="0" In="Normalized_Signal" />
  </Signal>
</tsf:model>
</tsf:TSF>
</tsf:TSFLibrary>

```

This TSF has an interface (see table 1) that accepts the envelope definition and the input signal. The output of the TSF is a Pass/Fail status.

Table 1—TSF Interface

Description	Name	Type	Default	Range
Signal Under Test	Signal_Input	SignalFunction		
Nominal amplitude of upper boundary	UpperBoundaryAmplitude	Physical		
Array of points defining upper boundary (relative values)	UpperBoundaryArray	array of real		
Nominal amplitude of lower boundary	LowerBoundaryAmplitude	Physical		
Array of points defining lower boundary (relative values)	LowerBoundaryArray	array of real		
time over which the signal is measured	measurement_gate	Time		

The number of points in the arrays will determine the accuracy of the envelope. Each point is a real value which specifies the instantaneous value relative to the nominal amplitude. For example, a nominal amplitude of -1 V will give an actual value of -1.3 V for an array value of 1.3 and an actual value of -0.8 V for an array value of 0.8. The measurement gate is the time over which the waveform is being measured, this is normally the same as the length of the envelope (in time).