

Serial To Parallel Tool

[**Online Help**](#)



Agilent Technologies

Notices

© Agilent Technologies, Inc. 2001-2007

No part of this manual may be reproduced in any form or by any means (including electronic storage and retrieval or translation into a foreign language) without prior agreement and written consent from Agilent Technologies, Inc. as governed by United States and international copyright laws.

Trademarks

Microsoft®, MS-DOS®, Windows®, Windows 2000®, and Windows XP® are U.S. registered trademarks of Microsoft Corporation.

Adobe®, Acrobat®, and the Acrobat Logo® are trademarks of Adobe Systems Incorporated.

Manual Part Number

Version 03.70.0000

Edition

December 1, 2007

Available in electronic format only

Agilent Technologies, Inc.
1900 Garden of the Gods Road
Colorado Springs, CO 80907 USA

Warranty

The material contained in this document is provided “as is,” and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Agilent disclaims all warranties, either express or implied, with regard to this manual and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Agilent shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Agilent and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

Technology Licenses

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license.

Restricted Rights Legend

If software is for use in the performance of a U.S. Government prime contract or sub-contract, Software is delivered and licensed as “Commercial computer software” as defined in DFAR 252.227-7014 (June 1995), or as a “commercial item” as defined in FAR 2.101(a) or as “Restricted computer software” as defined in FAR 52.227-19 (June 1987) or any equivalent

agency regulation or contract clause. Use, duplication or disclosure of Software is subject to Agilent Technologies’ standard commercial license terms, and non-DOD Departments and Agencies of the U.S. Government will receive no greater than Restricted Rights as defined in FAR 52.227-19(c)(1-2) (June 1987). U.S. Government users will receive no greater than Limited Rights as defined in FAR 52.227-14 (June 1987) or DFAR 252.227-7015 (b)(2) (November 1995), as applicable in any technical data.

Safety Notices

CAUTION

A **CAUTION** notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in damage to the product or loss of important data. Do not proceed beyond a **CAUTION** notice until the indicated conditions are fully understood and met.

WARNING

A **WARNING** notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in personal injury or death. Do not proceed beyond a **WARNING** notice until the indicated conditions are fully understood and met.

Using the Serial To Parallel Tool

The Serial To Parallel tool is used to convert a serial data stream into parallel data words. Parallel data word width is selectable up to 128 bits and can be displayed in a variety of number bases.

Advanced options let you: specify bit times (clock recovery) when there is no clock reference signal for the serial data stream, and identify frames and send only certain frame data bits to the output.

- **Serial to Parallel Conversion Overview** (see [page 7](#))
- **Probing Serial Data Streams** (see [page 9](#))
- **Setting Up the Logic Analyzer** (see [page 11](#))
- **Capturing Data (Triggering) on a Serial Pattern** (see [page 13](#))
- **Converting Serial Data to Parallel** (see [page 15](#))
 - Adding the Serial to Parallel Tool (see [page 16](#))
 - Selecting the Input Stream (see [page 19](#))
 - Specifying the Parallel Output (see [page 20](#))
 - Identifying Frames and Filtering Frame Data (see [page 21](#))
 - Using Clock Recovery (When There is No Clock) (see [page 23](#))
- **Analyzing Serial To Parallel Tool Output** (see [page 25](#))
- **Application Examples** (see [page 27](#))
 - RS-232C (see [page 28](#))
- **Serial To Parallel Tool Properties Dialog** (see [page 41](#))
- **Serial To Parallel Tool Control, COM Automation** (see [page 47](#))
- **Serial To Parallel Tool Setup, XML Format** (see [page 49](#))

Contents

Using the Serial To Parallel Tool 3

1 Serial to Parallel Conversion Overview

2 Probing Serial Data Streams

3 Setting Up the Logic Analyzer

4 Capturing Data (Triggering) on a Serial Pattern

5 Converting Serial Data to Parallel

Adding the Serial To Parallel Tool 16

Selecting the Input Stream 19

Specifying the Parallel Output 20

Identifying Frames and Filtering Frame Data 21

Using Clock Recovery (When There is No Clock) 23

6 Analyzing Serial To Parallel Tool Output

7 Application Examples

Converting RS-232C Serial Data to Parallel 28

Step 1: Probe the RS-232C serial data stream 28

Step 2: Set up the logic analyzer 29

Step 3: Capture the serial data 32

Step 4: Add the Serial To Parallel tool 32

Step 5: Select the input stream 33

Step 6: Specify the parallel output 33

Step 7: Identify frames 34

Step 8: Set up clock recovery 36

Step 9: View the Serial To Parallel tool output 36

8 Serial To Parallel Dialog

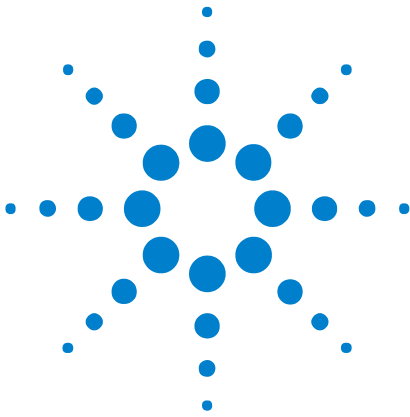
Frame Processing Dialog	43
Start of Frame Tab	43
Data Block Tab	44
End of Frame Tab	44
Clock Recovery Dialog	46

9 Serial To Parallel Tool Control, COM Automation

10 Serial To Parallel Tool Setup, XML Format

<ClockRecovery> Element	50
<DataBlock> Element	51
<EndOfFrame> Element	52
<Setup> Element	53
<StartOfFrame> Element	54

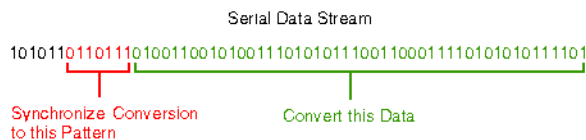
Index



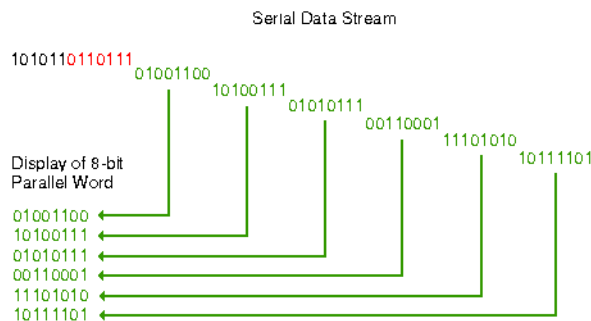
1 Serial to Parallel Conversion Overview

Serial to parallel conversion is used to convert long streams of serial data into parallel words which are easier to view and analyze.

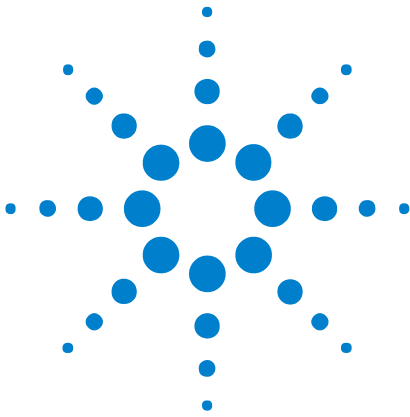
Another important element in the conversion of serial data is the ability to find specific bit patterns which identify the starting point of new data or the point where data changes. Identifying and using serial bit patterns to synchronize the start of data conversion lets you view and analyze only the desired data.



Once the data is converted, it is displayed in parallel words. For analysis purposes, it is important to be able to view words in variable lengths and in an order of least significant bit (LSB) first, or most significant bit (MSB) first.

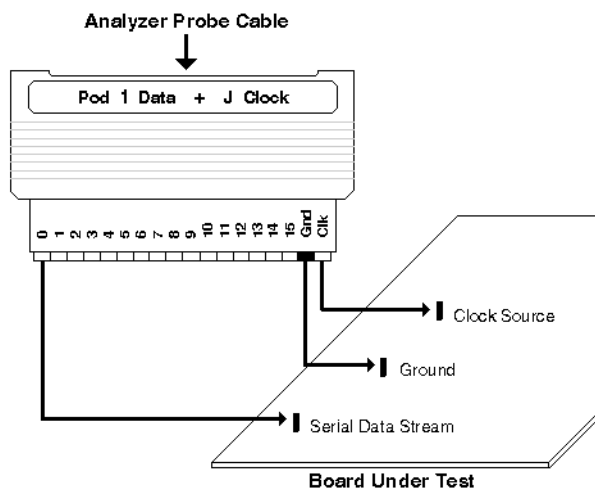


1 Serial to Parallel Conversion Overview



2 Probing Serial Data Streams

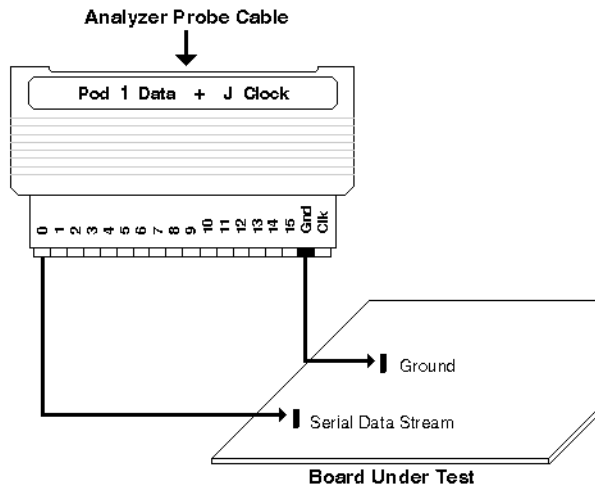
In most cases, only one logic analyzer pod is required for a serial to parallel conversion. In the device under test, a single logic analyzer data channel is connected to the serial data stream signal, and the clock input channel is connected to the reference clock signal.



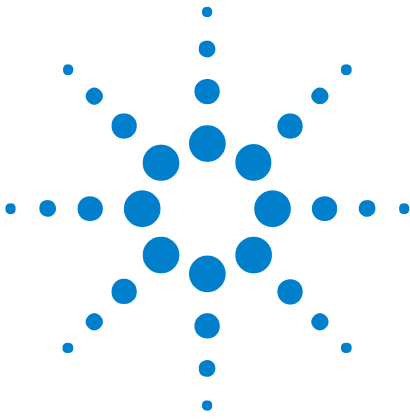
If multiple clock signals are required "to set up the state sampling clock" (in the online help), multiple logic analyzer pods are required. Each pod has one clock channel available.

If there is no reference clock for the serial data stream, a single logic analyzer data channel is connected to the device under test:

2 Probing Serial Data Streams



The pictures above show the flying-lead probe set, but there are many other options for probing a device under test. For more information on the available probing options, see "Probing the Device Under Test" (in the online help).



3 Setting Up the Logic Analyzer

The logic analyzer must be set up appropriately to capture data on the serial input stream. In general:

- The threshold voltage must be set appropriately (see "Setting the Logic Analyzer Threshold Voltage" (in the online help)).
- You must define a bus or signal name that contains the logic analyzer channel connected to the serial data signal (see "Defining Buses and Signals" (in the online help)).
- You must choose the appropriate logic analyzer sampling mode:
 - If there is a reference clock signal for the serial data stream (and you have connected it to the clock input channel on the logic analyzer probe), choose the **State - Synchronous Sampling** acquisition mode.
 - If there is no reference clock signal, choose the **Timing - Asynchronous Sampling** acquisition mode, and (for best results) set the **Sampling Period** so that 4 or more sample points are taken on each serial bit – this will allow clock recovery.

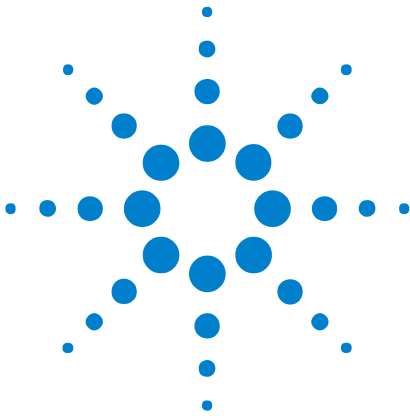
NOTE

The "*Transitional / Store Qualified Timing Mode*" (in the online help) sampling option (see "To select the timing sampling option" (in the online help)) is not supported by the Serial To Parallel tool.

For more information, see "Choosing the Sampling Mode" (in the online help).



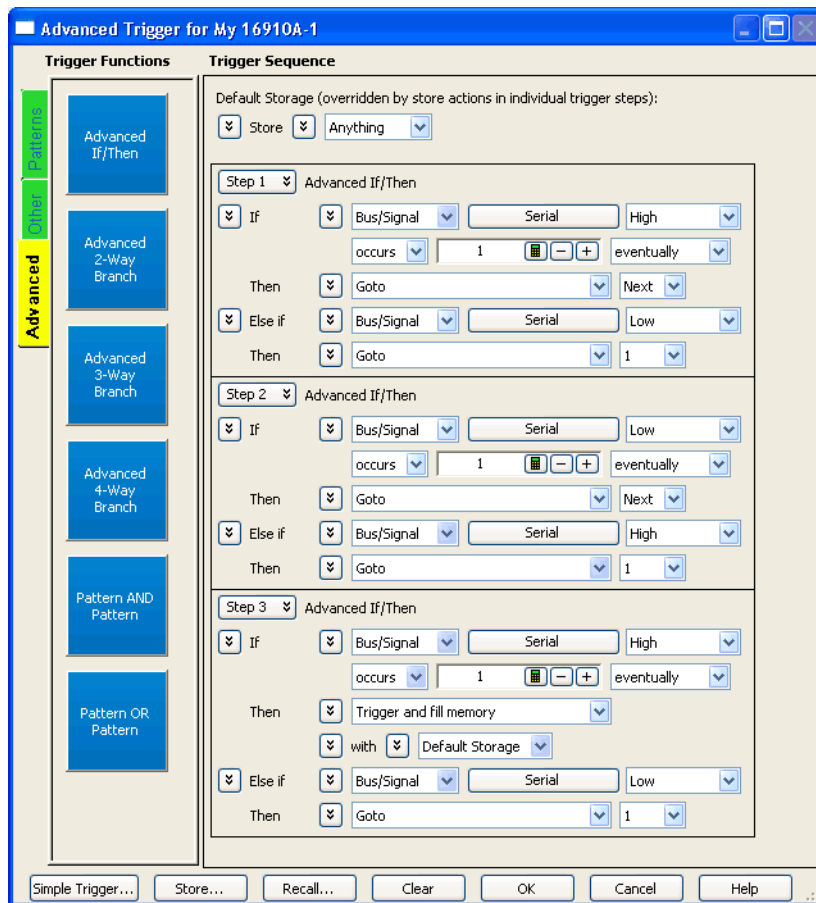
3 Setting Up the Logic Analyzer



4 Capturing Data (Triggering) on a Serial Pattern

In the State (Synchronous Sampling) Acquisition Mode

In the logic analyzer's state acquisition mode, you can trigger on a serial data pattern by using a multi-step trigger sequence where the steps look for consecutive bit values in the serial data stream. By using advanced two-way branch trigger functions in each step, you can go to the next step if the bit value you are looking for is found, or you can restart the sequence (at step 1) if the opposite bit value is found. For example, the following trigger specification triggers on the 3-bit serial pattern "101":



4 Capturing Data (Triggering) on a Serial Pattern

In the Timing (Asynchronous Sampling) Acquisition Mode

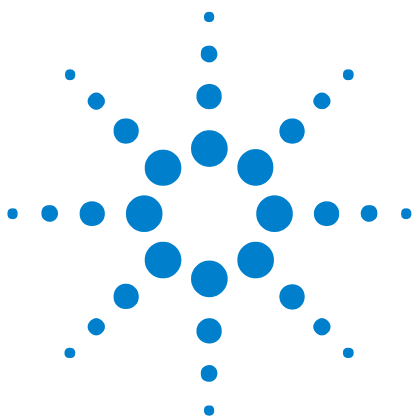
In the logic analyzer's timing acquisition mode, there are several ways to specify a trigger. If the start of the serial transmission can be controlled, the trigger can be set to the first occurrence of an edge in the serial data.

If the serial data is continuously transmitted, the easiest method is to trigger on any edge and let the Serial To Parallel tool find the first start pattern. With deep memory, triggering on a start of frame is usually not an issue since many frames of data are captured.

A second method is the use of the trigger macro *Pattern present/absent for > duration*. Some serial protocols such as the CAN bus have idle periods where the data line goes high for a certain minimum period of time.

A third method is to trigger on an external event that indicates the beginning of a frame of data.

- See Also**
- For more information on setting up advanced triggers, see "Specifying Advanced Triggers" (in the online help).



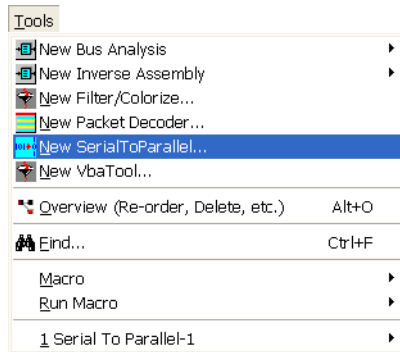
5 Converting Serial Data to Parallel

- Adding the Serial to Parallel Tool (see [page 16](#))
- Selecting the Input Stream (see [page 19](#))
- Specifying the Parallel Output (see [page 20](#))
- Identifying Frames and Filtering Frame Data (see [page 21](#))
- Using Clock Recovery (When There is No Clock) (see [page 23](#))

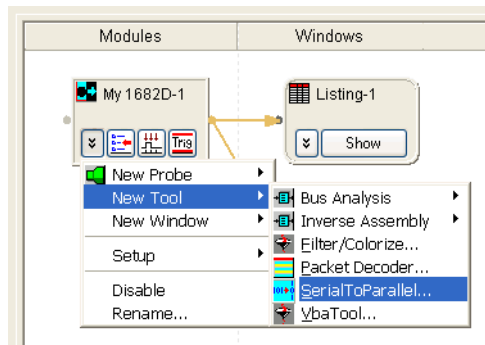


Adding the Serial To Parallel Tool

- 1 From the main menu, choose **Tools>New SerialToParallel...**

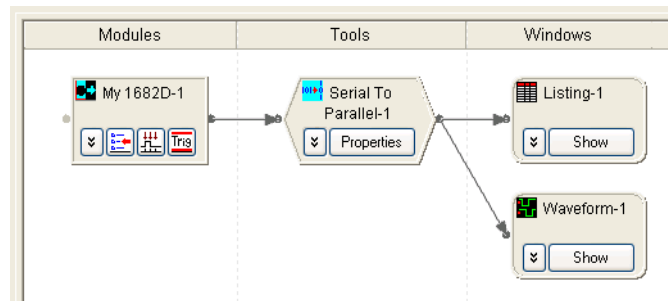


Or, in the Overview window, from a logic analyzer module's drop-down menu, choose **New Tool>SerialToParallel**.



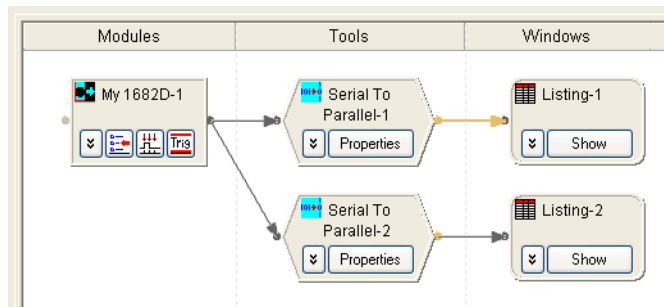
- 2 In the Serial To Parallel dialog (see [page 41](#)), set up the tool properties:
 - a Select the bus/signal and bit of the logic analyzer channel that is connected to the serial data input stream (see [Selecting the Input Stream](#) (see [page 19](#))).
 - b Specify the parallel output bus, name width, bit order, and the serial data sample that the conversion should start on (see [Specifying the Parallel Output](#) (see [page 20](#))).
 - c If you want to identify frames and filter frame data, check **Enable frame processing**, click **Define...**, and specify the Start Of Frame and End Of Frame patterns (see [Identifying Frames and Filtering Frame Data](#) (see [page 21](#))).
 - d If there is no reference clock for the serial input data stream, check **Enable clock recovery**, click **Define...**, and specify the bit time, the recovered data signal name, and the type of data encoding used (see [Using Clock Recovery \(When There is No Clock\)](#) (see [page 23](#))).

- Click **OK** to close the Serial To Parallel dialog and apply the conversion. In the Overview window, the tool connection will look something like:



Using Multiple Serial To Parallel Tools in Parallel

If your application requires you to convert and view multiple serial segments, or minor frames of data within a major frame, all from the same serial data stream, use multiple Serial To Parallel tools. In the Listing display window, simply insert the desired bus/signal names created in the multiple Serial To Parallel tools.



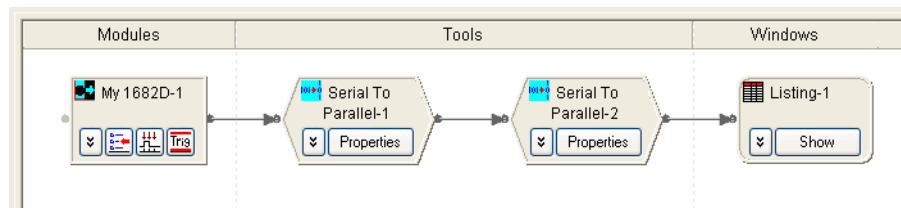
Examples of these types of applications are:

- To view both SEND and RECEIVE data.
- To view multiple serial lines simultaneously.
- Using the frame processing **Pass selected bits in data block** feature to view selected words within the same frame.

Using Multiple Serial To Parallel Tools in Series

You can configure multiple Serial To Parallel tools in series. An example of this would be in an RS-232C application where two computers are communicating with each other using a modem. Here, you would use one tool to strip off the start bit, parity bit, and stop bit, and use the second tool to search for frame start and end patterns.

5 Converting Serial Data to Parallel



Selecting the Input Stream

Before you can select the serial data input stream in the Serial To Parallel tool's properties dialog, you must define a bus or signal name that contains the logic analyzer channel connected to the serial data signal (see "Defining Buses and Signals" (in the online help)).

- 1 In the Serial To Parallel dialog (see [page 41](#))'s **Input Bus/Signal (Serial)** box, click the bus button and select the name of bus/signal that has the logic analyzer channel connected to the serial data input stream.
- 2 If you select a bus, a drop-down menu appears to let you select the bit that is connected to the serial data input stream.

Specifying the Parallel Output

After you have added the Serial to Parallel tool (see [page 16](#)) and selecting the input serial data stream (see [page 19](#)), you are ready to specify the parallel output bus data. In the Serial To Parallel dialog (see [page 41](#))'s **Output Bus (Parallel)** box:

- 1 Enter the **Name** of the output bus.
- 2 Enter the **Width** of the output bus.
- 3 Select the **Bit Order** (most significant bit or least significant bit first).
- 4 Enter the input stream data **Start Sample** at which serial to parallel data conversion should take place.

Identifying Frames and Filtering Frame Data

You can configure a serial to parallel conversion to start and stop at specified bit patterns in the serial data stream. You do this by defining a frame of data with a start point and an end point. The start point is a user-defined bit pattern and the end point can be either another bit pattern or a selected number of bits.

Once a frame is defined, you have the option to process and view the data in that frame. When you **Enable frame processing**, a new start signal name that shows the start point of the defined frame is inserted into the output. If an end pattern is defined, an end signal name is also inserted into the output.

Framing data is very useful if you want to synchronize the start of a serial to parallel conversion to a specific pattern of bits, and have the conversion end with a specific pattern of bits or after a data block of a specified length.

NOTE

If a defined frame is seen multiple times in a serial data stream, all instances are converted.

The following procedure begins from a basic serial analysis configuration. For an example of a serial to parallel conversion that includes frame processing, see [Converting RS-232C Serial Data to Parallel](#) (see [page 28](#)).

To define a frame of data

In the Serial To Parallel dialog (see [page 41](#))'s **Advanced Processing** box:

- 1 Check **Enable frame processing** and click **Define...**
- 2 In the Frame Processing dialog's Start of Frame tab (see [page 43](#)), specify the Start of Frame bus/signal name, pattern width, and pattern.

When the Start of Frame pattern is found, that pattern is output on the bus/signal name you specify.

Also, output with each parallel bus value is a single bit **SOF** signal value; its value is "1" for the first parallel output value after the Start of Frame and "0" otherwise (when there are multiple parallel words in a frame). This signal is useful for identifying the start of frame when only parallel output data is being displayed.

5 Converting Serial Data to Parallel

- 3 In the Frame Processing dialog's Data Block tab (see [page 44](#)):
 - a If you want to remove a "stuffed bit", check **Remove stuffed**, select whether a "0" or "0/1" is stuffed, and enter the number of "1's" that the stuffed bit follows.
 - b Choose whether to pass the entire data block or just selected bits in the data block.
 - c If you chose **Pass selected bits in data block**, enter the number of the first bit that should be passed; then, choose either **Bit** and enter the ending bit number, or choose **End of data block**.
- 4 In the Frame Processing dialog's End of Frame tab (see [page 44](#)):
 - a Choose whether the frame ends after a certain number of bits or on a pattern.
 - b If you chose **End frame on pattern**, specify the End of Frame signal name, pattern width, and pattern.
- 5 Click **OK** to close the Frame Processing dialog.

Using Clock Recovery (When There is No Clock)

Clock recovery is used when the incoming serial data has no reference clock. Because there is no clock, the input serial data is captured using the *timing* (asynchronous sampling) acquisition mode (using the internal sample period clock) and a bit time that you specify.

NOTE

For best results, the sample period of the timing analyzer should be set so that four or more sample points are taken on each serial bit.

If the clock recovery function is enabled, the incoming serial data is first sampled before the other serial analysis functions are performed. The recovered data is displayed under the signal name defined in the Clock Recovery dialog.

To set up clock recovery

In the Serial To Parallel dialog (see [page 41](#))'s **Advanced Processing** box:

- 1 Check **Enable clock recovery** and click **Define...**
- 2 In the Clock Recovery dialog (see [page 46](#)):
 - a Enter the recovered data **Signal Name**. The recovered data becomes the serial data source that is converted to parallel data.
 - b Enter the **Bit Time** of the input serial data stream. The bit time is equal to:

$1/\text{bit_rate}$

Or:

$1/\text{baud_rate}$

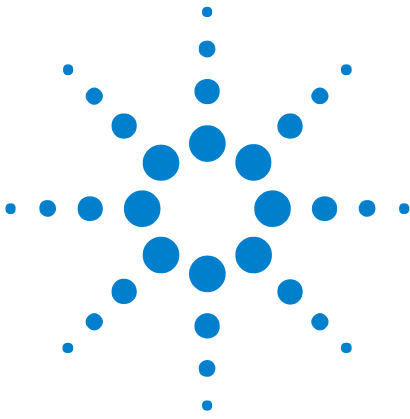
The serial data is sampled in the middle of each bit, and data edges are used to re-synchronize the sampling.

The bit time can be any number. It doesn't have to be a multiple of the timing analyzer sample period clock. An internal bit clock is maintained in software and is re-synchronized on an edge. The first sampling point after an edge is at half the bit time from the edge, and successive points are sampled at intervals of the bit time. The sampled points are taken from the timing analyzer points that are closest to the ideal sample times of the internal bit clock.

- c Select the type of **Data Encoding** that is used in the serial input data stream:
 - Normal, also known as NRZ (Non-Return to Zero), means high for a "1" and low for a "0".

5 Converting Serial Data to Parallel

- NRZI (Non-Return to Zero Inverted), also known as differential encoding, means the signal level changes when a "0" occurs, whereas a "1" does not cause a change. For example, a stream of 0's encoded in NRZI looks like a square wave. The signal changes on every bit.
 - d Select whether to wait for the first edge in the input data before starting clock recovery or to start clock recovery immediately.
- 3 Click **OK** to close the Clock Recovery dialog.



6 Analyzing Serial To Parallel Tool Output

After data has been converted from serial to parallel and displayed in the Listing window (or any of the other display windows), you can analyze the output parallel bus data as you would any other data bus.

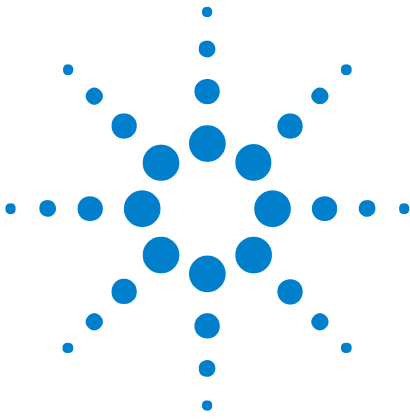
Finding Frames and Placing Markers

A useful feature for doing serial analysis in the display windows is **Find** (see "Searching the Captured Data" (in the online help)). For example, you can use Find to search for the same bit pattern as the *Start Of Frame*, and you can use **Find Previous** or **Find Next** to locate adjacent frames.

You can also use **Find** to search for multiple occurrences of the Start Of Frame pattern and place markers on each occurrence (see "To specify "found" marker placement" (in the online help)).



6 Analyzing Serial To Parallel Tool Output



7 Application Examples

- Converting RS-232C Serial Data to Parallel (see [page 28](#))



Converting RS-232C Serial Data to Parallel

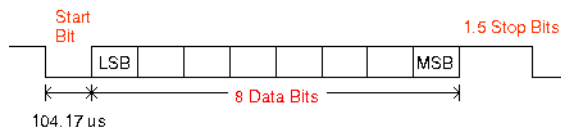
The following example analyzes serial data from an RS-232C output port for the purpose of verifying a *Print Screen* function. The intent of this example is to give you a conceptual view of using the Serial To Parallel tool for similar measurements.

RS-232C Characteristics

The characteristics for the RS-232C line in this example are as follows:

- Bit Rate = 9600 Baud (9600 bits/second).
- Voltage = -6 Volts to +6 Volts, inverted data.
- Data Format = 1 start bit, 8 data bits, and 1.5 stop bits. Parity off.

The timing for RS-232C data before it is inverted is shown below.



To convert RS-232C serial data to parallel

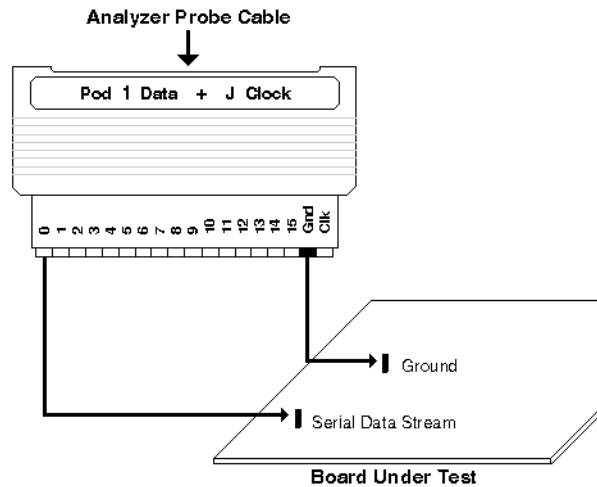
- 1 Probe the RS-232 serial data stream (see [page 28](#))
- 2 Set up the logic analyzer (see [page 29](#))
- 3 Capture the serial data (see [page 32](#))
- 4 Add the Serial To Parallel tool (see [page 32](#))
- 5 Select the input stream (see [page 33](#))
- 6 Specify the parallel output (see [page 33](#))
- 7 Identify frames (see [page 34](#))
- 8 Set up clock recovery (see [page 36](#))
- 9 View the Serial To Parallel tool output (see [page 36](#))

Step 1: Probe the RS-232C serial data stream

NOTE

The following connection scheme is conceptual and should be changed according to your specific application.

You can probe the RS-232C serial data stream using a single logic analyzer channel:



The picture above show the flying-lead probe set, but there are many other options for probing a device under test. For more information on the available probing options, see "Probing the Device Under Test" (in the online help).

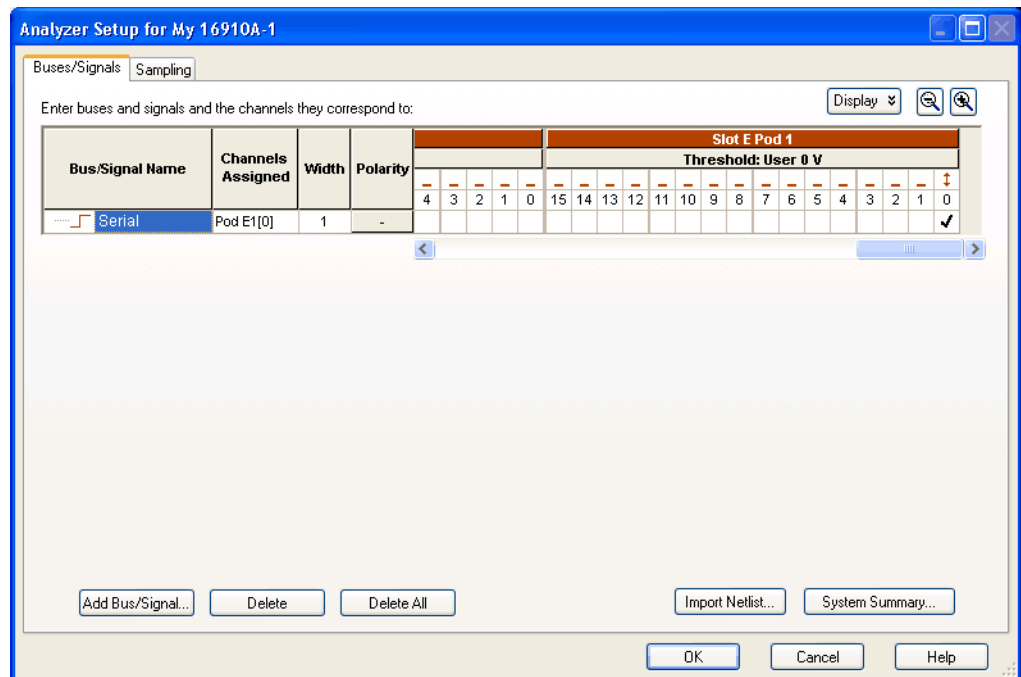
Next • Step 2: Set up the logic analyzer (see [page 29](#))

Step 2: Set up the logic analyzer

1 Set the threshold voltage.

With a voltage swing from -6 Volts to +6 Volts, the logic analyzer threshold voltage should be set to 0 V. For more information, see "Setting the Logic Analyzer Threshold Voltage" (in the online help).

2 Define a bus or signal name that contains the logic analyzer channel connected to the serial data signal.



NOTE

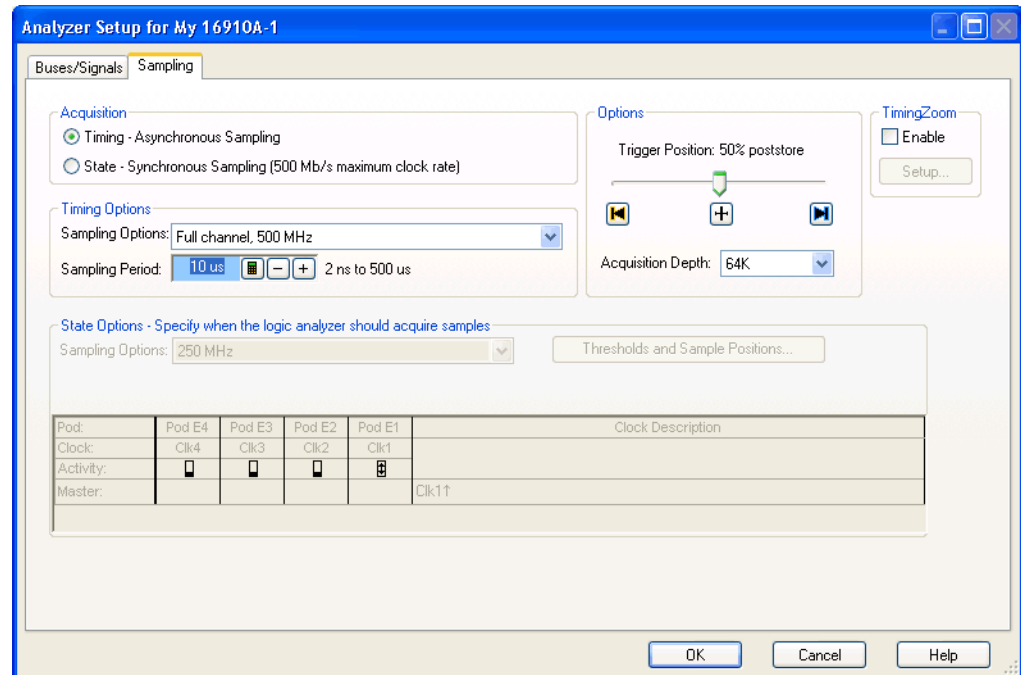
Because the serial data is inverted, choose negative polarity.

For more information, see "Defining Buses and Signals" (in the online help).

3 Set up the logic analyzer sampling mode.

Because there is no reference clock signal, choose the **Timing - Asynchronous Sampling** acquisition mode.

With a bit time of 104.17 us, a **Sampling Period** of 10 us will give about 10 sample points for each serial bit and allow for clock recovery.

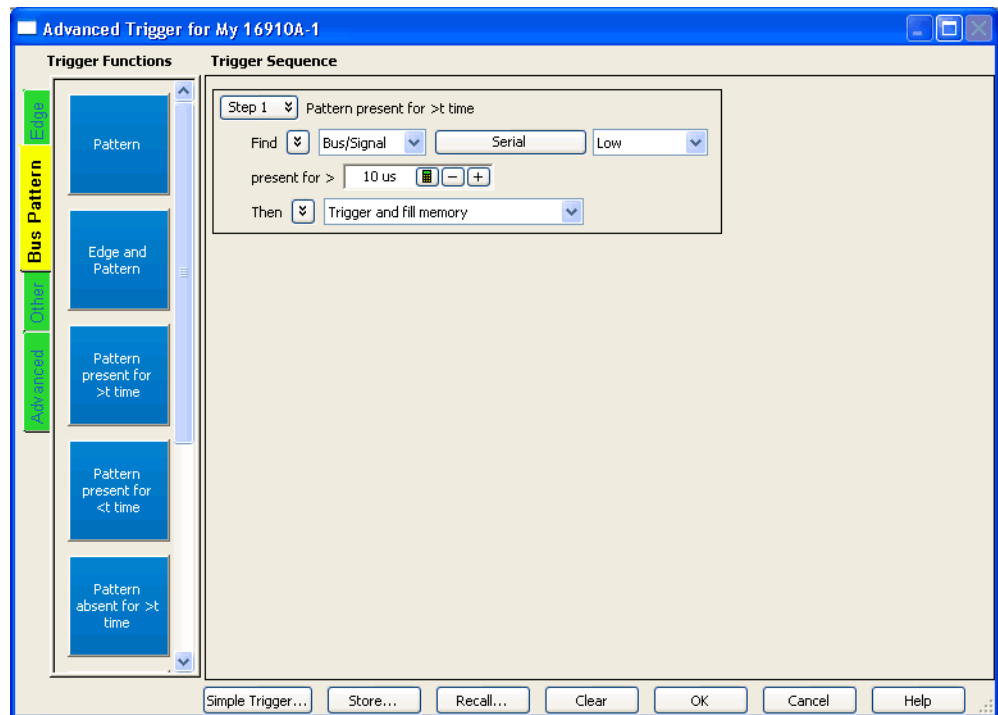


For more information, see "Choosing the Sampling Mode" (in the online help).

- Next** • Step 3: Capture the serial data (see [page 32](#))

Step 3: Capture the serial data

- 1 Set up a trigger to capture data when "0" is present for greater than 10 us (the beginning of the start bit):

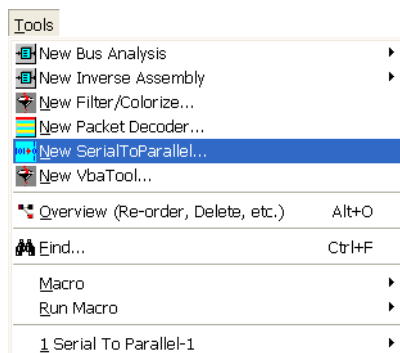


- 2 Run the logic analyzer to capture data from the serial input data stream.

Next • Step 4: Add the Serial To Parallel tool (see [page 32](#))

Step 4: Add the Serial To Parallel tool

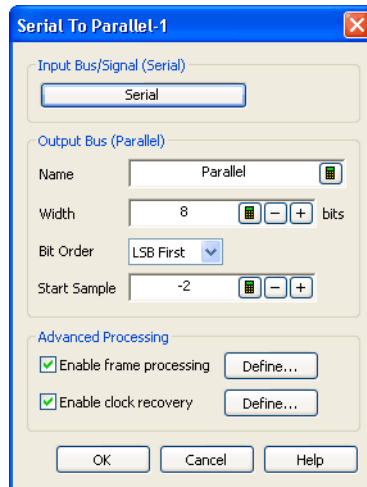
From the main menu, choose **Tools>New SerialToParallel...**



Next • Step 5: Select the input stream (see [page 33](#))

Step 5: Select the input stream

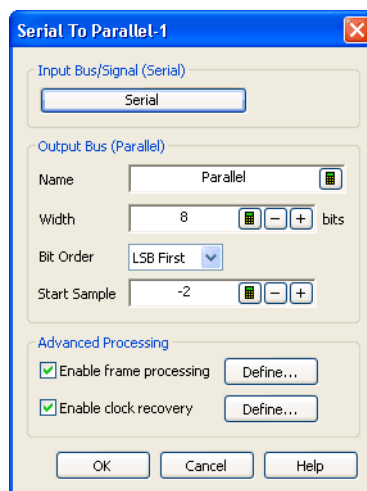
Select the signal name of the logic analyzer channel that is connected to the serial data input stream.



Next • Step 6: Specify the parallel output (see [page 33](#))

Step 6: Specify the parallel output

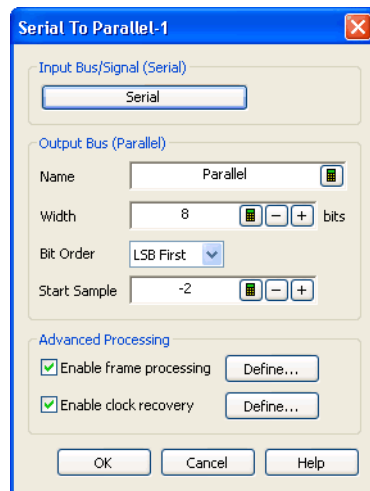
Specify the parallel output bus name, width, **LSB First** bit order, and "-2" for the serial data sample that the conversion should start on (this should be at least one sample before the beginning of the start bit).



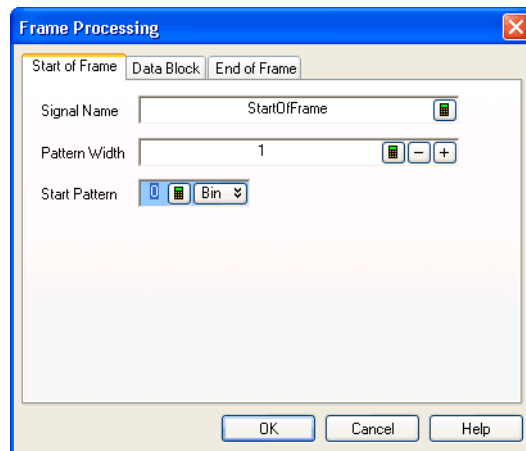
Next • Step 7: Identify frames (see page 34)

Step 7: Identify frames

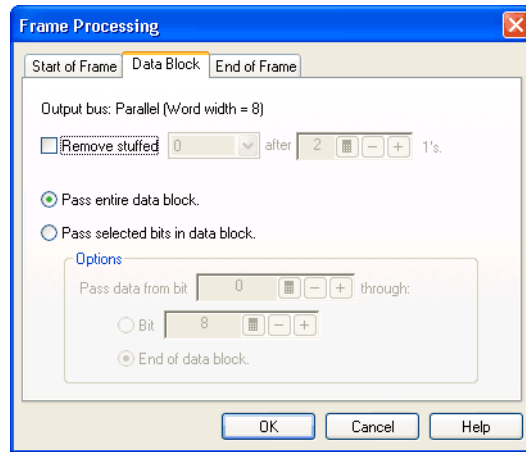
- 1 Check **Enable frame processing** and click **Define...**



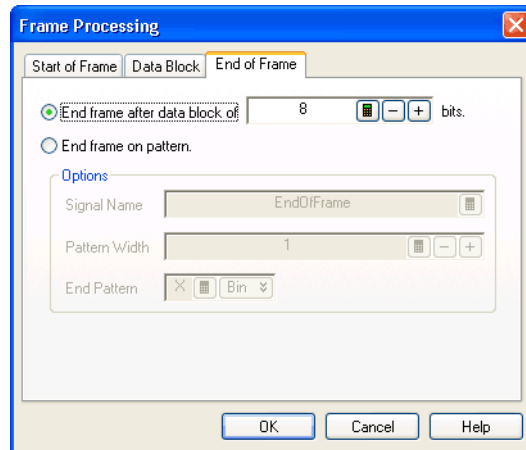
- 2 In the Start of Frame tab, specify the Start Of Frame signal name and one low bit as the start pattern.



- 3 In the Data Block tab, uncheck **Remove stuffed** and select **Pass entire data block**.



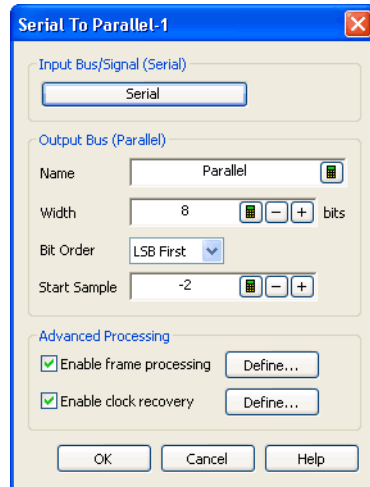
- 4 In the End of Frame tab, specify that the frame ends after a data block of 8 bits.



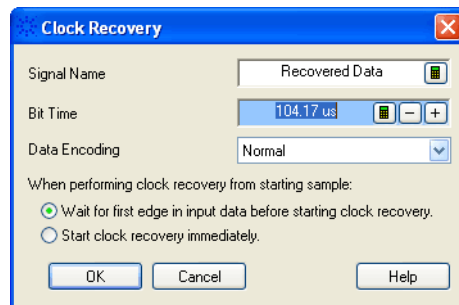
- Next** • Step 8: Set up clock recovery (see [page 36](#))

Step 8: Set up clock recovery

- 1 Because there is no reference clock for the serial input data stream, check **Enable clock recovery** and click **Define...**



- 2 In the Clock Recovery dialog, and specify the recovered data signal name, a bit time of 104.17 us (1/bit_rate, or 1/9600 bits/second), and a **Normal** data encoding.

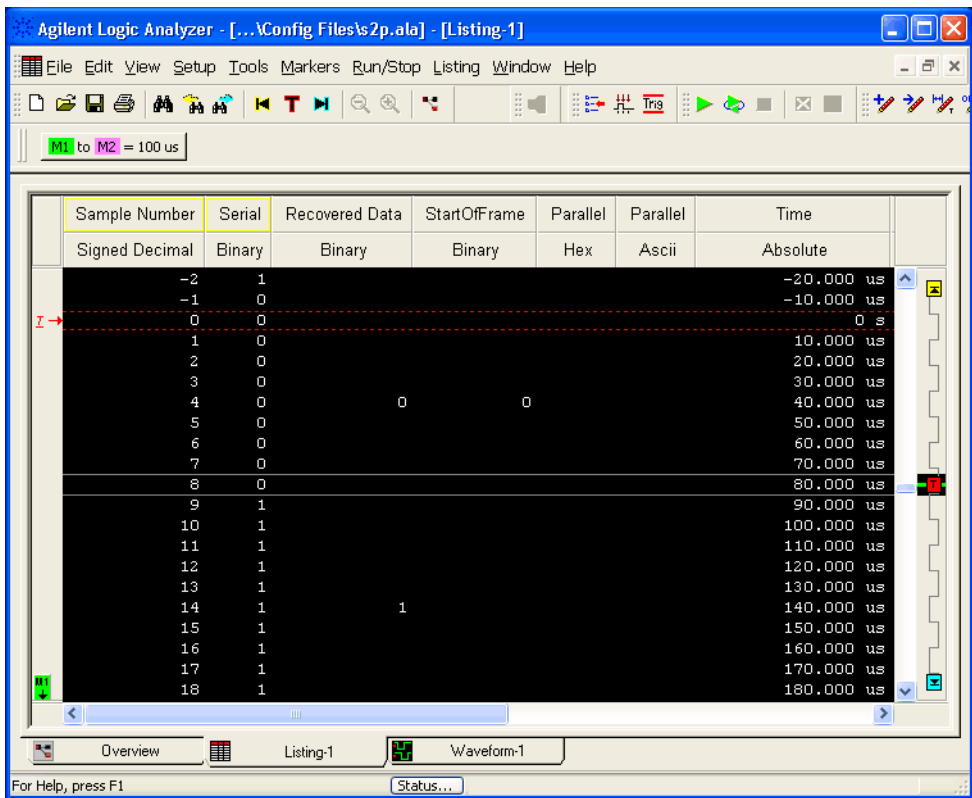


Next • Step 9: View the Serial To Parallel tool output (see [page 36](#))

Step 9: View the Serial To Parallel tool output

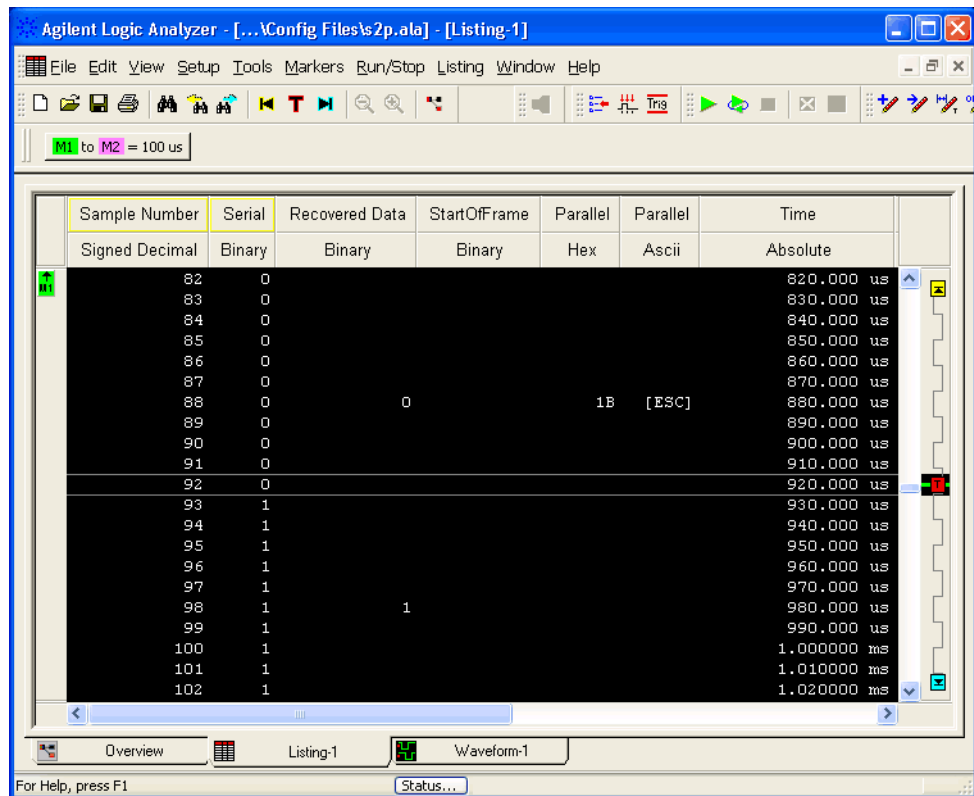
Viewing the Results In the Listing display window, the values in the recovered data signal column shows the results of the clock recovery. This is the actual serial data that the conversion tool operates on.

The start of frame signal column shows when the SOF is identified.




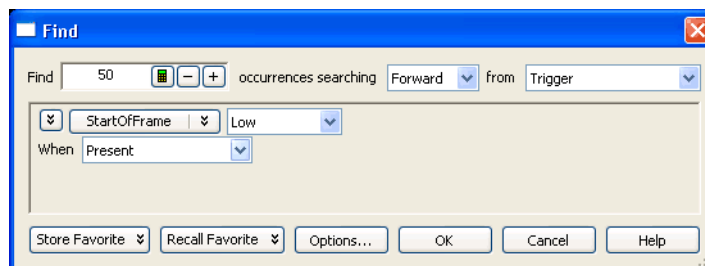
The parallel output bus column shows the results of tool conversion. You can add another parallel output bus column and display the conversion results in a different number base if desired.

7 Application Examples



Placing Markers on Converted Patterns

You can choose **Edit>Find...** from the main menu (or click the  icon in the standard toolbar) to search for occurrences of things like the start of frame.

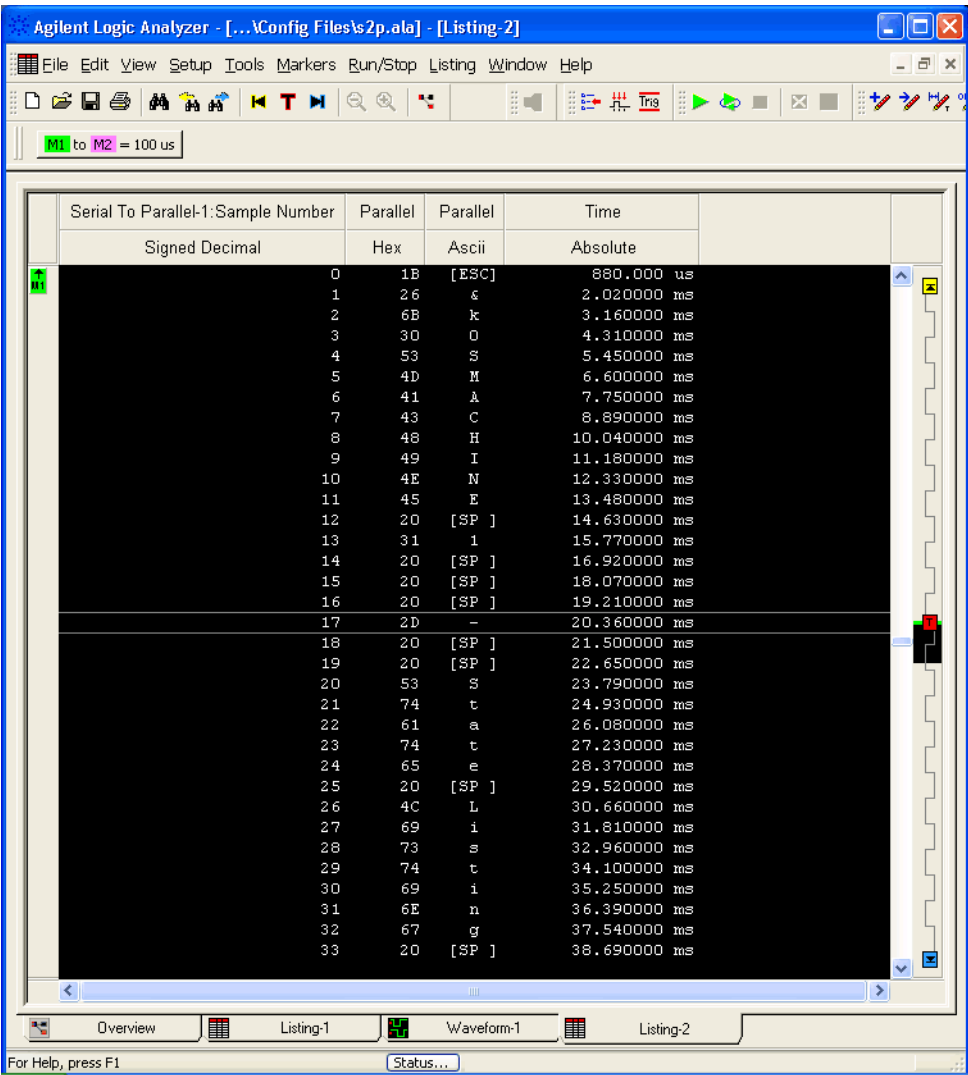


You can click **Options...** to specify where the Found marker should be placed or to place new markers on up to 1,024 occurrences that are found.

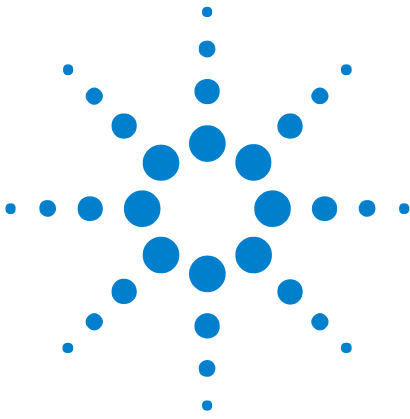


Compressing the Displayed Data

Finally, you can remove all data columns except the parallel bus output to see a compressed view of the captured and converted data.

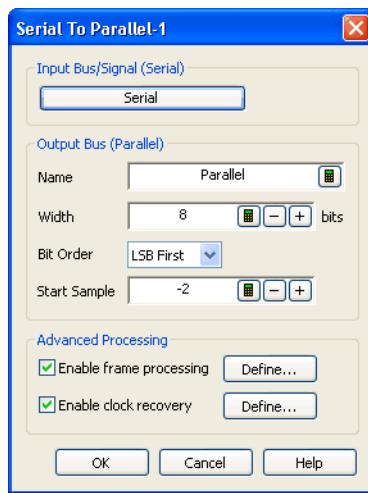


7 Application Examples



8 Serial To Parallel Dialog

The Serial To Parallel dialog lets you specify all the properties for the serial to parallel data conversion.



Input Bus/Signal (Serial)	Selects the bus and bit of the input serial signal that is to be converted into parallel bus data.
Output Bus (Parallel)	Contains options for decoding the input serial data signal to parallel bus data.
Name	Specifies the bus/signal name of the output parallel bus data.
Width	Specifies the width of the output parallel bus.
Bit Order	Specifies whether the first bit is the most significant bit (MSB) or the least significant bit (LSB).
Start Sample	Specifies the captured data sample at which the serial to parallel decoding should start.



Enable frame processing	Lets you enable or disable frame processing. When this box is checked, you can click Define... to open the Frame Processing dialog (see page 43) and specify patterns that identify Start Of Frame (SOF), frame data, and End Of Frame (EOF).
Enable clock recovery	Lets you enable or disable clock recovery. When this box is checked, you can click Define... to open the Clock Recovery dialog (see page 46) and specify bit times and data encoding.

- See Also**
- Using the Serial To Parallel Tool (see [page 3](#))
 - Converting Serial Data to Parallel (see [page 15](#))

Frame Processing Dialog

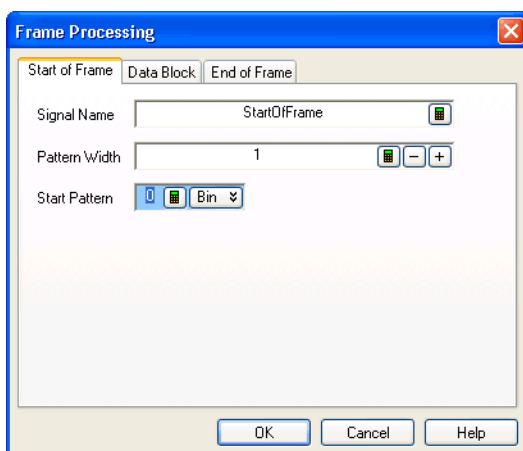
The Frame Processing dialog has three tabs:

- Start of Frame (see [page 43](#)) – for specifying the SOF pattern.
- Data Block (see [page 44](#)) – for specifying which bits of the data block to pass.
- End of Frame (see [page 44](#)) – for specifying the EOF pattern.

See Also • Identifying Frames and Filtering Frame Data (see [page 21](#))

Start of Frame Tab

The Frame Processing dialog's Start of Frame tab lets you specify the SOF pattern.

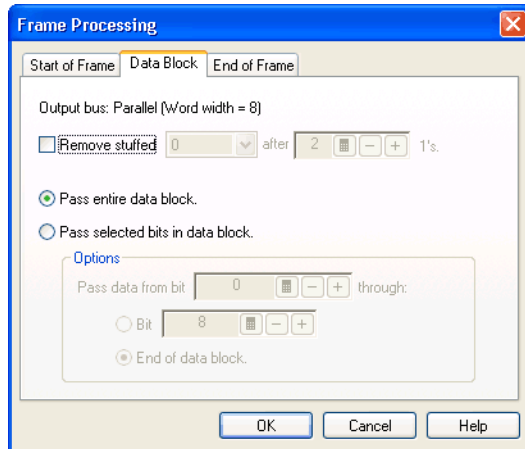


Signal Name	Specifies the bus/signal name of the bits that identify the Start of Frame. Also, output with each parallel bus value is a single bit SOF signal value; its value is "1" for the first parallel output value after the Start of Frame and "0" otherwise.
Pattern Width	Specifies the number of bits in the Start of Frame pattern.
Start Pattern	Specifies the Start of Frame pattern.

See Also • Identifying Frames and Filtering Frame Data (see [page 21](#))

Data Block Tab

The Frame Processing dialog's Data Block tab lets you specify which data block bits to pass on to the output. The size of the data block is defined by the End of Frame.

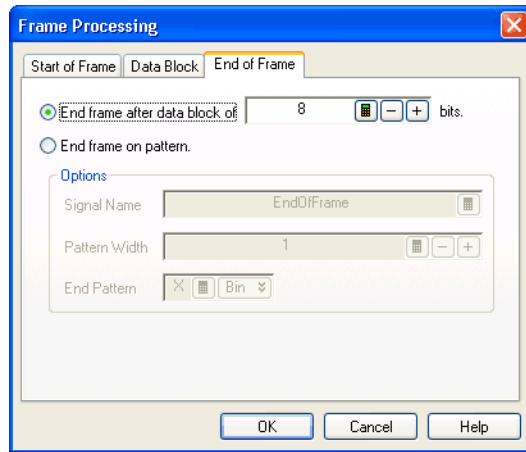


Removed stuffed	Lets you remove a stuffed bit after the the specified number of 1's.
Pass entire data block.	Sends the complete data block to the output.
Pass selected bits in data block.	Sends selected bits from the data block. You can specify the starting bit to send as a number, and you can specify the ending bit as a number or as the end of the data block.

See Also • Identifying Frames and Filtering Frame Data (see [page 21](#))

End of Frame Tab

The Frame Processing dialog's End of Frame tab lets you specify the EOF pattern.

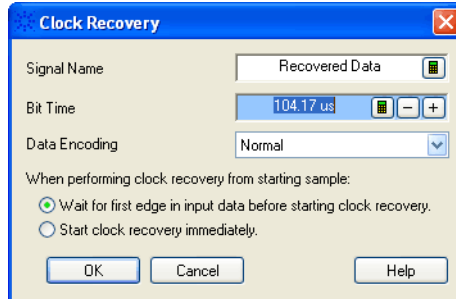


End frame after data block of N bits.	Specifies that the frame ends after a certain number of data block bits.
End frame on pattern.	Specifies that the frame ends on the specified pattern.
Signal Name	Specifies the bus/signal name of the bits that identify the End of Frame.
Pattern Width	Specifies the number of bits in the End of Frame pattern.
End Pattern	Specifies the End of Frame pattern.

See Also • Identifying Frames and Filtering Frame Data (see [page 21](#))

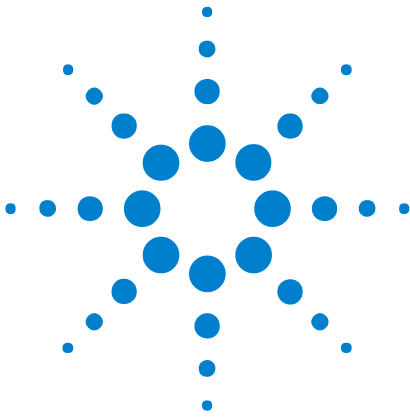
Clock Recovery Dialog

The Clock Recovery dialog lets you specify the bit time when there is no input clock reference for the incoming serial data.



Signal Name	Names the recovered data signal using the specified Bit Time and Data Encoding.
Bit Time	Specifies the bit time of the input serial data signal.
Data Encoding	Specifies the type of encoding used in the input serial data signal: <ul style="list-style-type: none"> • Normal or NRZ (Non-Return to Zero) — when the input signal level is low, it means a "0"; when the level is high, it means a "1". • NRZI (Non-Return to Zero Inverted) — when the input signal level changes, it means a "0"; when the level stays the same, it means a "1".
When performing clock recovery from starting sample	Lets you choose whether to wait for the first edge in the input data before starting clock recovery or to start clock recovery immediately.

See Also • Using Clock Recovery (When There is No Clock) (see [page 23](#))



9 Serial To Parallel Tool Control, COM Automation

The *Agilent Logic Analyzer* application includes the COM Automation Server. This software lets you write programs that control the *Agilent Logic Analyzer* application from remote computers on the Local Area Network (LAN).

In a COM automation program, you can configure a tool by:

- Loading a configuration file (which configures the complete logic analyzer setup).
- Using the "Tool" (in the online help) object's "DoCommands" (in the online help) method with an XML-format string parameter (see Serial To Parallel Tool Setup, XML Format (see [page 49](#))).

You can get information about a tool's configuration using the Tool object's "QueryCommand" (in the online help) method. Queries supported by the Serial To Parallel tool are listed below.

For more information about logic analyzer COM automation and tool objects in general, see "COM Automation" (in the online help).

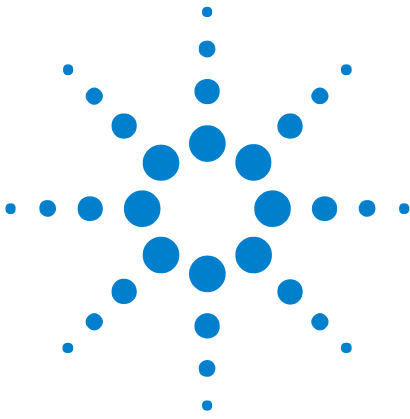
XML-Based Queries Supported

The Serial To Parallel tool supports the following XML-based queries (made with the "Tool" (in the online help) object's "QueryCommand" (in the online help) method).

Query	Description
GetAllSetup	Returns the current setup, using the full tag set, used for writing generic configuration files (see the XML format <Setup> element (see page 53)).

- See Also**
- "COM Automation" (in the online help)
 - Serial To Parallel Tool Setup, XML Format (see [page 49](#))





10 Serial To Parallel Tool Setup, XML Format

When you save logic analyzer configurations to XML format files, setup information for the Serial To Parallel tool is included.

This XML format setup information is also used when writing COM automation programs to control the logic analyzer from a remote computer.

XML elements for the Serial To Parallel tool have the following hierarchy:

```
<Setup> (see page 53)  
  <ClockRecovery> (see page 50)  
  <StartOfFrame> (see page 54)  
  <DataBlock> (see page 51)  
  <EndOfFrame> (see page 52)
```

- See Also**
- "XML Format" (in the online help)
 - Serial To Parallel Tool Control, COM Automation (see [page 47](#))



<ClockRecovery> Element

The <ClockRecovery> element specifies the clock recovery options.

Attributes

Name	Description
BitTime	'number "time_unit" (in the online help)'
Encoding	'NRZ' (Non-Return to Zero — high for a "1" and low for a "0") or 'NRZI' (Non-Return to Zero Inverted — signal level change means a "0", no signal level change means "1")
FindFirstEdge	'F' (false) or 'T' (true)
SignalName	'string'

Parents This element can have the following parents: <Setup> (see [page 53](#)).

Example

```
<ClockRecovery SignalName='Recovered Data' BitTime='31.25 ns'
  Encoding='NRZ' FindFirstEdge='T' />
```

<DataBlock> Element

The <DataBlock> element specifies the frame processing data block filtering options.

Attributes

Name	Description
BeginDataBlockBit	'number'
EndDataBlockBit	'number'
PassBits	'PassEntireDataBlock' or 'PassSelectedBitsInDataBlock'
PassThrough	'PassThroughSelectedBit' or 'PassThroughEndOfDataBlock'
RemoveStuffedBits	'F' (false) or 'T' (true)
StuffedBitValue	'0' (zero is the only valid value)
StuffingNumBits	'number' (2 through 999)

Parents This element can have the following parents: <Setup> (see [page 53](#)).

Example

```
<DataBlock RemoveStuffedBits='F' PassBits='PassSelectedBitsInDataBlock'
  BeginDataBlockBit='0' PassThrough='PassThroughSelectedBit'
  EndDataBlockBit='512' />
```

<EndOfFrame> Element

The <EndOfFrame> element specifies the frame processing End Of Frame options.

Attributes

Name	Description
Base	'Binary', 'Hex', or 'Octal'
BusName	'string'
DataBlockSize	'number'
DontCareMask	'number'
Option	'AfterDataBlock' or 'OnPattern'
Value	'number'
Width	'number'

Parents This element can have the following parents: <Setup> (see [page 53](#)).

Example `<EndOfFrame Option='AfterDataBlock' DataBlockSize='1025' />`

<Setup> Element

The <Setup> element contains setup information for the Serial To Parallel tool.

Attributes

Name	Description
BitOrder	'LSB' or 'MSB'
ClockRecoveryEnabled	'F' (false) or 'T' (true)
FrameProcessingEnabled	'F' (false) or 'T' (true)
InputBus	'string'
InputBusChannel	'number'
InputBusModule	'string'
OutputBusName	'string'
OutputBusWidth	'number'
StartingSampleNumber	'number'

Children This element can have the following children: <ClockRecovery> (see [page 50](#)), <StartOfFrame> (see [page 54](#)), <DataBlock> (see [page 51](#)), <EndOfFrame> (see [page 52](#)).

Parents This element can have the following parents: "<Tool>" (in the online help).

When used in COM automation, this element is returned by the "QueryCommand method" (in the online help)'s GetAllSetup query. You can also use this element string as an XMLCommand with the "DoCommands method" (in the online help) to configure the Serial To Parallel tool.

Example

```
<Setup InputBusModule='My 1682D-1' InputBus='My Bus 1'
  InputBusChannel='0' OutputBusName='Parallel' OutputBusWidth='8'
  StartingSampleNumber='0' BitOrder='MSB' FrameProcessingEnabled='T'
  ClockRecoveryEnabled='T'>
  <ClockRecovery SignalName='Recovered Data' BitTime='31.25 ns'
    Encoding='NRZ' FindFirstEdge='T' />
  <StartOfFrame BusName='StartOfFrame' Base='Binary' Width='8'
    Value='170' DontCareMask='0' />
  <DataBlock RemoveStuffedBits='F'
    PassBits='PassSelectedBitsInDataBlock' BeginDataBlockBit='0'
    PassThrough='PassThroughSelectedBit' EndDataBlockBit='512' />
  <EndOfFrame Option='AfterDataBlock' DataBlockSize='1025' />
</Setup>
```

<StartOfFrame> Element

The <StartOfFrame> element specifies the frame processing Start Of Frame options.

Attributes

Name	Description
Base	'Binary', 'Hex', or 'Octal'
BusName	'string'
DontCareMask	'number'
Value	'number'
Width	'number'

Parents This element can have the following parents: <Setup> (see [page 53](#)).

Example

```
<StartOfFrame BusName='StartOfFrame' Base='Binary' Width='8'
  Value='160' DontCareMask='15' />
```

Index

A

application examples, Serial To Parallel tool, [27](#)

B

baud rate, [23](#)
bit clock, [23](#)
bit time, [23](#)
bus width, parallel, [20](#)

C

clock recovery, [23](#)
Clock Recovery dialog, [46](#)
ClockRecovery, XML element, [50](#)
COM automation, Serial To Parallel tool, [47](#)

D

Data Block tab in Frame Processing dialog, [44](#)
data encoding method, [23](#)
DataBlock, XML element, [51](#)

E

encoding method, serial data, [23](#)
End of Frame tab in Frame Processing dialog, [44](#)
EndOfFrame, XML element, [52](#)
example, frame processing, [21](#)
example, markers to find frames, [25](#)
example, Serial To Parallel tool overview, [7](#)
example, triggering on serial pattern, [13](#)

F

Frame Processing dialog, [43](#)
frame, synchronizing to, [21](#)
framing options, [21](#)

I

input signal name, [19](#)
input, serial signal, [20](#)

M

markers to find frames, [25](#)

N

notices, [2](#)
NRZ, [23](#)
NRZI, [23](#)

O

overview, [7](#)

P

parallel bus width, [20](#)
pattern, triggering on serial, [13](#)
patterns, markers to find, [25](#)
patterns, SOF and EOF, [21](#)
probing for serial to parallel tool, [9](#)

R

recovery, clock, [23](#)
RS-232C example, [28](#)
RS-232C example, adding the Serial To Parallel tool, [32](#)
RS-232C example, clock recovery, [36](#)
RS-232C example, frame identification, [34](#)
RS-232C example, input stream, [33](#)
RS-232C example, logic analyzer setup, [29](#)
RS-232C example, parallel output, [33](#)
RS-232C example, probing, [28](#)
RS-232C example, Serial To Parallel tool output, [36](#)
RS-232C example, triggering, [32](#)

S

serial data analysis, [3](#)
serial data, converting to parallel output, [15](#)
serial input signal, [20](#)
serial pattern, triggering on, [13](#)
Serial To Parallel dialog, [41](#)
Serial To Parallel tool, [3](#)
Serial To Parallel tool, adding, [16](#)
Serial To Parallel tool, logic analyzer setup for, [11](#)
Setup, XML element, [53](#)
SOF signal, [21](#), [43](#)
Start of Frame tab in Frame Processing dialog, [43](#)
StartOfFrame, XML element, [54](#)
synchronize to frame, [21](#)

T

trademarks, [2](#)
triggering on serial pattern, [13](#)

W

width, parallel bus, [20](#)

X

XML format, Serial To Parallel tool, [49](#)

