Use xTIMEcomposer and xSCOPE to trace data in real-time

IN THIS DOCUMENT

xTIMEcomposer and the xSCOPE library let you instrument your program with probes that collect application data in real-time. This data can be sent over an XTAG-2 debug adapter to xTIMEcomposer for real-time display or written to a file for offline analysis.

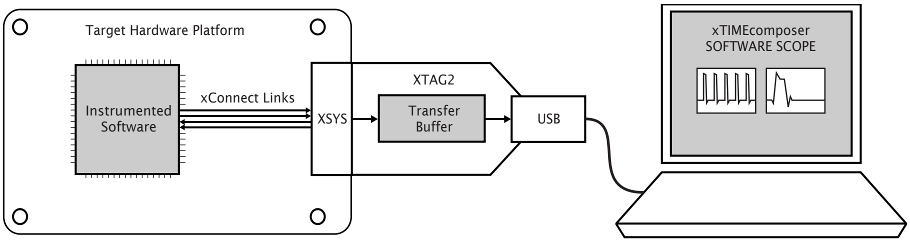

Figure 1: xSCOPE connectivity

If you are using a legacy FTDI or XTAG-1 debug adapter, or if the XSYS connector on your target hardware does not provide an xCONNECT Link, you can configure the probes to output trace data over your adapter's UART interface instead (see XM-000957-PC). Note that the UART interface is supported on a single tile only and offers significantly reduced performance.

1 XN File Configuration

To allow the tools to configure the xCONNECT link required for high speed data collection using xscope, the XN file for a board must be modified to expose the connection to the XTAG-2 device. The following information must be added to the links section of an XN file for a board to set up the link used by the target device to communicate with the XTAG-2 and the xscope channel.

```
<Link Encoding="2wire" Delays="4,4" Flags="XSCOPE">
  <LinkEndpoint NodeId="0" Link="XOLD"/>
  <LinkEndpoint RoutingId="0x8000" Chanend="1"/>
</Link>
```

Note that when the link is set to 2 wire, the minimum delay is set to 4 and the flags specify that this link is to be used for streaming debug. Setting the delay higher results in the output of packets used by xscope being less frequent. The RoutingId is also important as the value 0x8000 specifies to the tools that this is a special link used for xscope.

When used in a multi-tile system the NodeId of the package which is connected to the XSYS connector must be specified. The tools set up the links with the other tiles but they need to know which specific device has the external link to be connected to the XTAG-2.

2 Instrument a program

The example program in Figure 2 uses the xSCOPE instrumentation functions to trace the input levels to a microphone.

```
#include <xscope.h>

port micL;
port micR;

void xscope_user_init(void) {
  xscope_register(2,
    XSCOPE_CONTINUOUS, "Microphone Left",  XSCOPE_UINT, "mV",
    XSCOPE_CONTINUOUS, "Microphone Right", XSCOPE_UINT, "mV"
  );
}

int main() {
  while (1) {
    int sample;
    micL :> sample;
    xscope_uint(0, sample);
    micR :> sample;
    xscope_uint(1, sample);
  }
}
```

Figure 2: Program that traces input levels to a microphone

The constructor `xscope_user_init` registers two probes for tracing the left and right inputs to a microphone. The probes are defined as **continuous**, which means xTIMEcomposer can interpolate values between two subsequent measurements. The probes are defined to take values of type `unsigned int`.

In `main`, the program calls the probe function `xscope_uint` each time it samples data from the microphone. This function creates a trace record and sends it to the PC.

Figure 3 summarizes the different types of probes that can be configured. Only **continous** probes can be displayed real-time.

Probe Type	Data Type	Scope View	Example
XSCOPE_CONTINUOUS	XSCOPE_UINT XSCOPE_INT XSCOPE_FLOAT	Line graph. May be interpolated	Voltage levels of a motor controller
XSCOPE_DISCRETE	XSCOPE_INT	Horizontal lines	Buffer levels of audio CODEC
XSCOPE_STATEMACHINE	XSCOPE_UINT	State machine	Progression of protocol
XSCOPE_STARTSTOP	XSCOPE_NONE XSCOPE_UINT XSCOPE_INT XSCOPE_FLOAT	Start/stop bars	Recorded function entry and exit, with optional label value

Figure 3: Supported probe types

3 Configure and run a program with tracing enabled

Once you have instrumented your program, you must compile and link it with the xSCOPE library, and run it in either offline or real-time mode.

To link with the xSCOPE library and run xSCOPE, follow these steps:

1. Open the Makefile for your project.
2. Locate the `XCC_FLAGS_`*config* variable for your build configuration, for example `XCC_FLAGS_Release`.
3. Add the option `-fxscope`.
4. Create a Run Configuration for your target device (see XM-000963-PC).
5. Click the ** xSCOPE** tab and select **Offline Mode** to save data to a file for offline analysis, or **Real-Time Mode** to output the data to the real-time viewer.
   - In offline mode, xTIMEcomposer logs trace data until program termination and saves the traced data to the file `xscope.xmt`. To change, enter a filename in the **Output file** text box. To limit the size of the trace file, enter a number in the **Limit records to** text box.
   - In real-time mode, xTIMEcomposer opens the Scope view and displays an animated view of the traced data as the program executes.
6. Click **Run** to save and run the configuration.

4 Analyze data offline

Double-click a trace file in *Project Explorer* to open it in the **Scope view**, as shown in Figure 4.

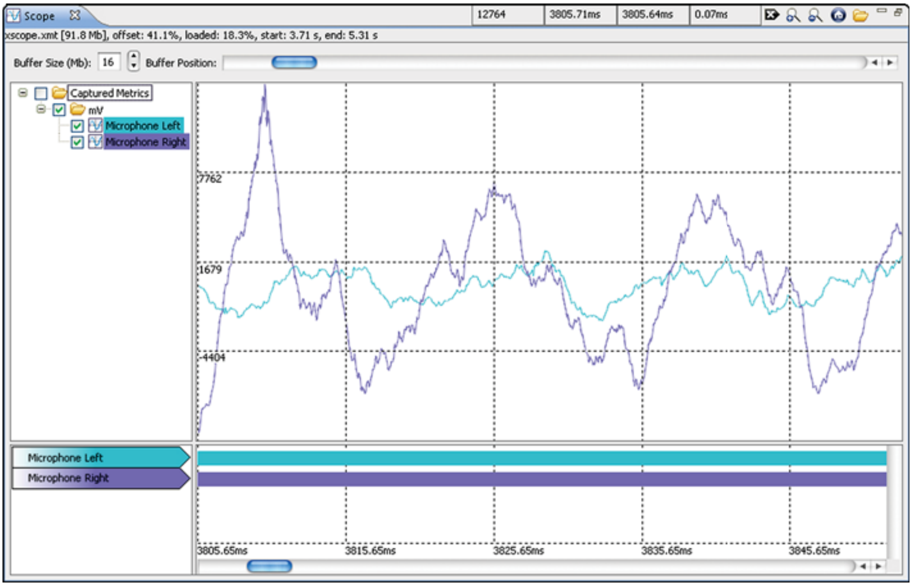

Figure 4: Offline Scope view

The top panel of the **Scope** view displays a graph of the data values for each selected probe: the *x*-axis represents time (as per the timeline in the bottom panel) and the *y*-axis represents the traced data values. The probes are grouped by their assigned units, and multiple probes with the same unit can be overlaid onto a single graph.

Moving the cursor over the scope data displays the current data (*y*-value) and time (*x*-value) in the first two of the four numeric boxes at the top of the window. Left-click on the view to display a marker as a red line - the associated time is displayed in the third numeric box. The fourth numeric box displays the difference between the marker time and the current cursor position.

If the cursor changes to a pointing finger, double-click to locate the statement in the source code responsible for generating the trace point.

The bottom panel of this view displays a timeline for each probe: vertical lines on a probe's timeline indicate times at which the probe created a record.

Drag the **Buffer Position** slider left or right to move through the timeline. To show more information in the window, increase the value in the **Buffer Size** field.

Use the **Scope** view toolbar at the top of the window to perform additional tasks:

To show data points for interpolated continuous signals, click the **Continuous**

points button.

To view all data points, click the **Zoom Fit** button.

Load a trace file that is not part of your project, click the **Open** button and browse to the file.

5 Analyze data in real-time

The *Scope* view can display trace data streamed from hardware in real-time. The left panel displays the signal information and controls and the right panel displays the screen view for the signals.

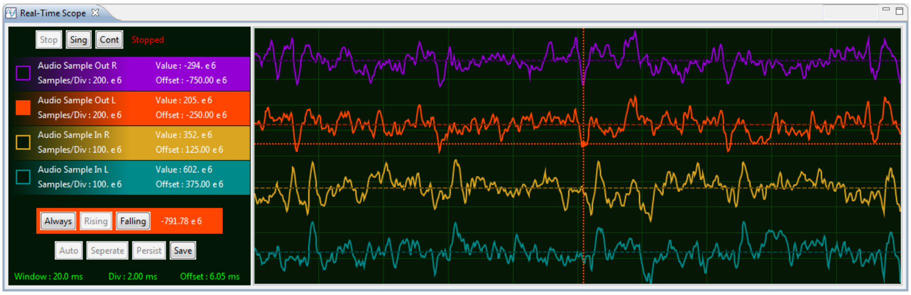

Figure 5: Real-Time Scope view

The left panel displays a list of the *continous* probes registered by the application (see §2). Each named probe is assigned a color that is used to draw events on the display, and which is used to identify the probe in the screen panel.

The *Scope* view is based around a traditional oscilloscope, and data is *captured* around a *trigger*, and then *displayed*. The capture mode, display mode, trigger and timebase are all controlled in the left panel. The right panel has 10 horizontal and vertical divisions, and the scales are all shown as units per division.

Numeric controls can all be modified by using the mouse: click the left button to increase the value or the right button to decrease the value. The scroll wheel can be used if your platform supports it (Mac OS/X, Linux, and some but not all versions of Windows).

5.1 Capture control

There are three capture-modes: *continous*, *single capture* or *stopped*. The default mode on start-up is for the system to capture and display continuously. The label associated with the capture controls shows the current state of the xSCOPE system.

Figure 6: Capture controls

Stop Display

Stops the screen panel from triggering and capturing, no more updates will be applied to the screen whilst this mode is set. The mode can be used to inspect the captured data. The mouse can be used to change signal and time base scales and offsets as described below to inspect the signals in detail. When stopped, you can zoom in on the time base and view the signal in more detail: the displayed signals are subsampled when the timebase is large, and zooming in on the timebase will reveal all data.

Single Capture

Selest single shot mode to capture one screen of data and return to the stopped state. If a trigger is enabled (see Figure 8) the system will wait for this trigger condition to be met before updating the screen and returning to the stopped state.

Continuous Capture

Select free running mode to update the screen as frequently as possible. If triggers are enabled, the screen will update only when the trigger is met.

5.2 Signal Control

The signal controls are available for each registered probe on the coloured label displayed in the left panel (see Figure 7)

Figure 7: Signal Controls

Enable / Disable Signal

Toggle the visibility of the signal by double clicking on the name.

Signal Samples/Div

Change the *Samples per Division* of this probe with the mouse buttons; this affects the vertical scale of the signal.

Signal Screen Offset

Change the vertical *Offset* of this probe with the mouse buttons; this affects the vertical position of the signal.

Signal Trigger

The signal can be used as a trigger (see Figure 8) by clicking in trigger box to the left of the probe label. Only one signal can be used for triggering.

5.3 Trigger Control

A trigger can be used to restrict the system so that data is only captured when a condition is met. By default all triggers are disabled, causing data to be captured unconditionally. To enable triggering, a trigger must be selected by clicking on the box to the left of the probe label.

When triggering is enabled, a cross appears on the screen showing the trigger level (relative to the signal on which the trigger is selected) and the trigger offset on the timebase. The center of the cross is the time and value where the trigger happens/happened; to the left of this are the signals that lead up to the trigger; to the right are the signals after the trigger.

The trigger level and offset can be set directly by clicking in the right-hand pane. Changes only take effect if the scope is not stopped, and either running continuously, or set for a single trigger.

Figure 8: Trigger Controls

Always

Disables the trigger and captures data unconditionally.

Rising

Trigger on a rising edge of the signal. This is the default mode when selecting a signal to be used for triggering.

Falling

Trigger on a falling edge of the signal.

The value label associated with the enabled trigger shows the current trigger value set for the signal. This can be changed by using the mouse buttons.

5.4 Timebase Control

The timebase controls are used to set the time range for the signal capture window, allowing you to scale and shift the horizontal axis.

Figure 9: Timebase Controls

Time Window

The current size to the time window. Scales all signals in time and affects the time per division.

Time per Division

The time units per division. Scales all signals in time and affects the time window.

Time Window Offset

The position of the trigger in the time window. Shifts all signals left and right. Note that the trigger may not be visible, and could be to the left or right of the time window. The signals can be shifted right only a limited value.

5.5 Screen Control

Several commands are available that operate on all signals.

Figure 10: Screen Controls

Auto Range Signals

Automatically arranges all current signals to fit on the screen. The signals are measured for a short while, and each signal scaled and offset to fit the screen. All signals are displayed across each other.

Separate Signals

Similar to Auto Range, but all signals are scaled to fit in a small part of the screen. All signals are offset so that they are visible separately.

Persistant Display

Disabled.

Save Data

Saves the current scope view to a PNG file in a user-defined location.

6 Trace using the UART interface

If you are using a legacy FTDI or XTAG-1 debug adapter, or if the XSYS connector on your target hardware does not provide an xCONNECT Link, you can output data over the UART interface provided by your adapter.

To use the UART interface, you must provide the xSCOPE library with a 1-bit UART TX port that has been initialized with the pin connected to the UART-TX pin on your debug adapter. An example initialization is shown below.

```
#include <platform.h>
#include <xscope.h>

port uart_tx = PORT_UART_TX;

void xscope_user_init(void) {
  xscope_register(2,
    XSCOPE_CONTINUOUS, "Microphone Left",  XSCOPE_UINT, "mV",
    XSCOPE_CONTINUOUS, "Microphone Right", XSCOPE_UINT, "mV"
  );
  xscope_config_uart(uart_tx);
}
```

⚠ Because the UART interface uses a port instead of an xCONNECT Link, the probe functions can be called on a single tile only.