xSPY Getting Started

xSPY is a free DAVE plug-in used to visualize and analyze the tracing data generated on the microcontroller target by the debug log App: DBG002 App

xSPY is the PC host counterpart of DBG002.

Using xSPY the user can easily define a graphical interface to control and monitor the microcontroller from the PC. xSPY provides views to monitor generated debug log messages, control page to GET/SET variables, and oscilloscopes to visualize the variables.

DBG002 App adds minimal intrusive software trace capabilities to your application. The trace capabilities include generation of debug logging messages at user defined positions in the code, i.e. code instrumentation. In addition it provides assertions to check the behavior of the code. Using both greatly simplify the debugging of your application.

Even more detail information can be obtained using the DBG002 App. The DBG002 App provides the means to sample the contents of user selected variables at user defined positions in the code. The contents of the variables are streamed continuously to the PC for further analysis. The sampling frequency is controlled by the user.

DBG002 features include

- **printf** like debugging: debug or logging messages are sent to the host using a communication interface, i.e. UART.
- Message generation filter depending on message severity level
- **assertions**, using Nana library. The validity of the assertions are logged in the host using the communication interface
- Enable/disable the assertions in the code
- Ability to handle concurrently messages generated from different apps.
- Variable Get/Set functionality
- xSPY interface to sample and continuously stream the value of internal variables to the host

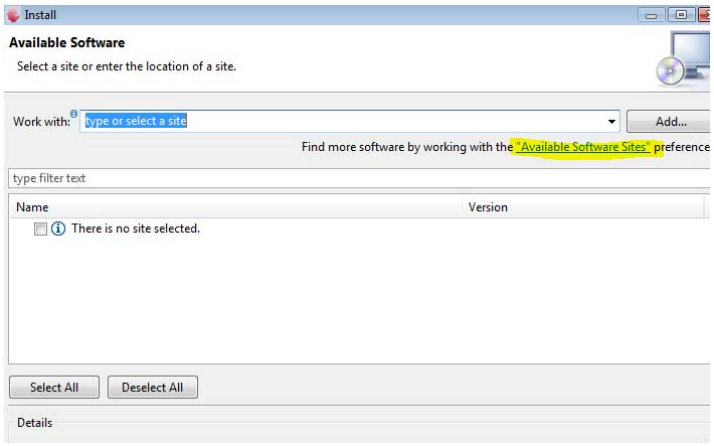
This tutorial shows you how to add the DBG002 App to an existing project, how to design the xSPY control page and basic usage of the xSPY plug-in.

The respective example projects can be downloaded from www.infineon.com/xspy

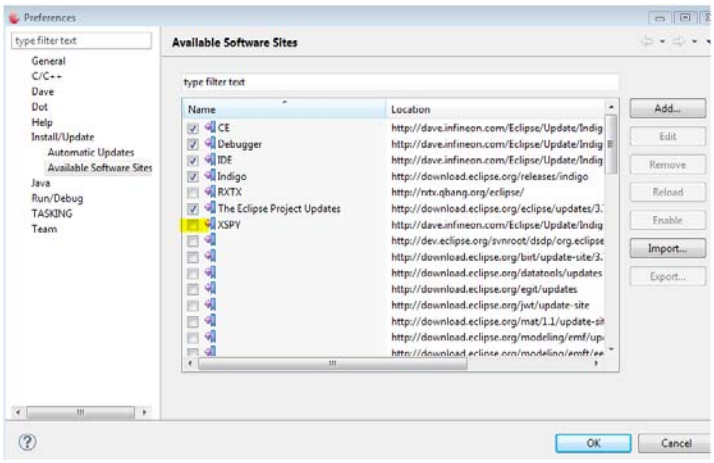
Contents

1 Installation

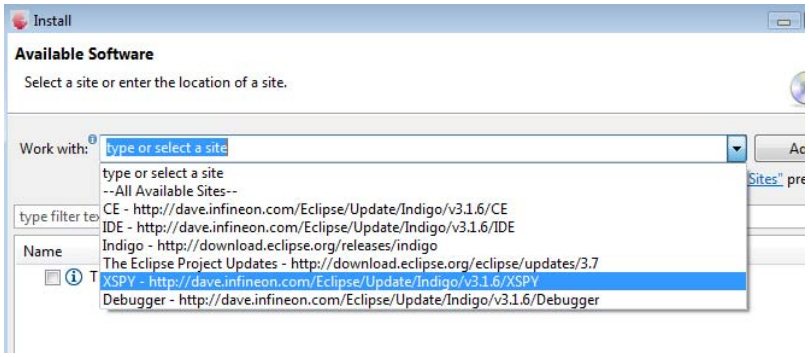
To install the xSPY plug-in, go to Help->Install New Software.



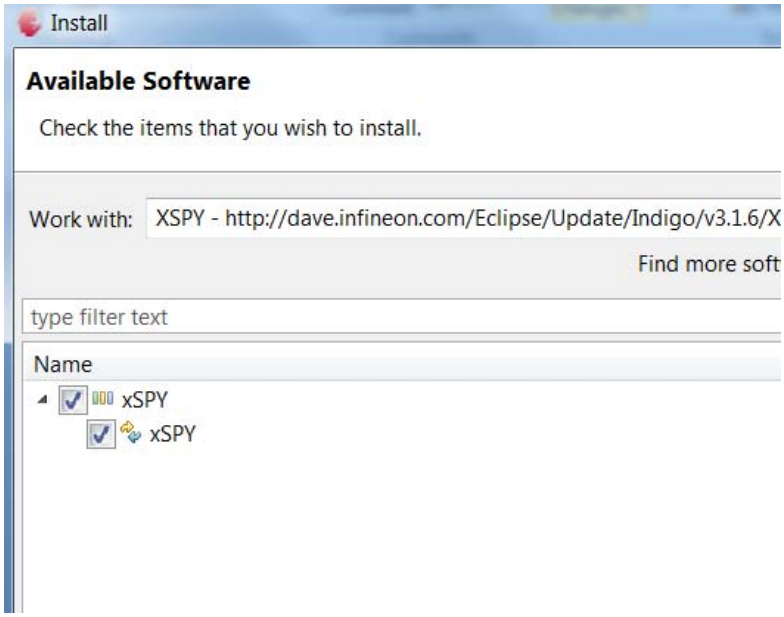
Click on “Available Software Sites“



Check “XSPY“ and click Ok



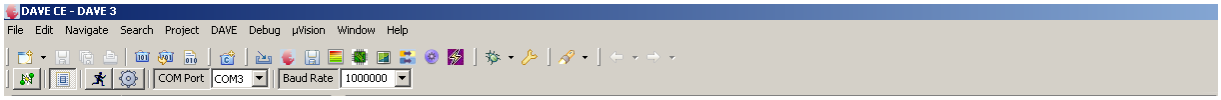
Select in “Work with:” the link to xSPY using the scroll list



Select xSPY by clicking on check box, press Next, accept the license agreement and after the installation is finished restart DAVE.

After restarting it is convenient, for later usage, to Open the xSPY Design Mode perspective. This can be done clicking on the right upper corner on Open Perspective -> Other ->xSPY DM.

Finally the added toolbar can be docked to the most convenient position.



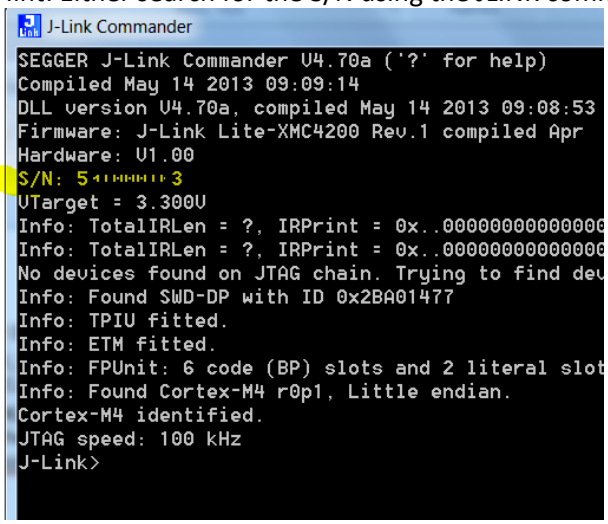
2 Prerequisite: Update Segger J-link ARM Driver

Latest Segger J-link ARM Driver (v4.70 or above) needs to be installed from the below link:

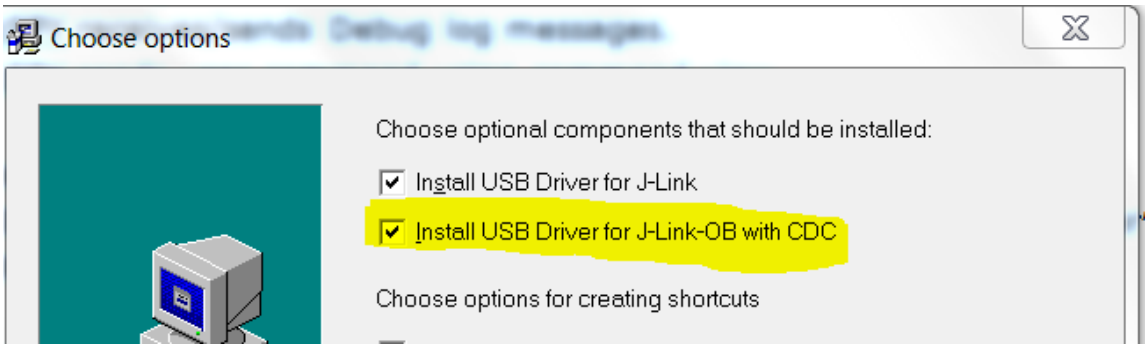
http://www.segger.com/jlink-software.html

Steps:

- Download the latest driver selecting
  Hint: Either search for the S/N using the JLINK commander once the board is attached to your PC



  or simple select the option "*I do not have a serial number because I own an eval board with J-Link on-board. How can I download J-Link software for it?*"
- Unplug the board
- Start the installation. Select the following option during installation

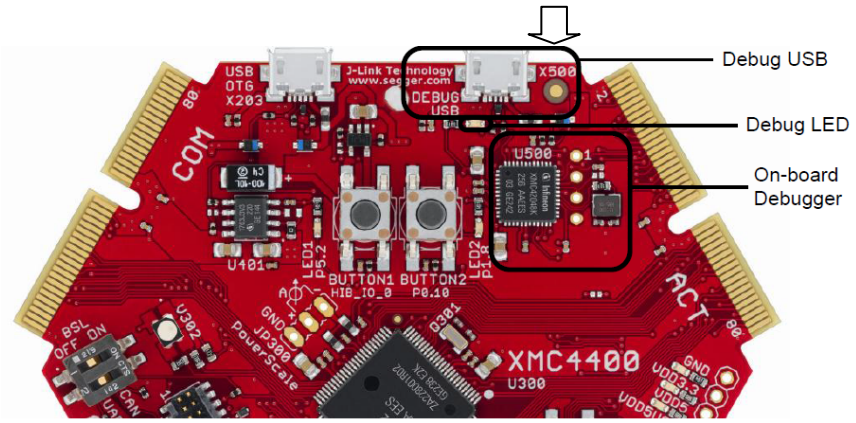


Note: This installs the required driver for on chip debug & VCOM

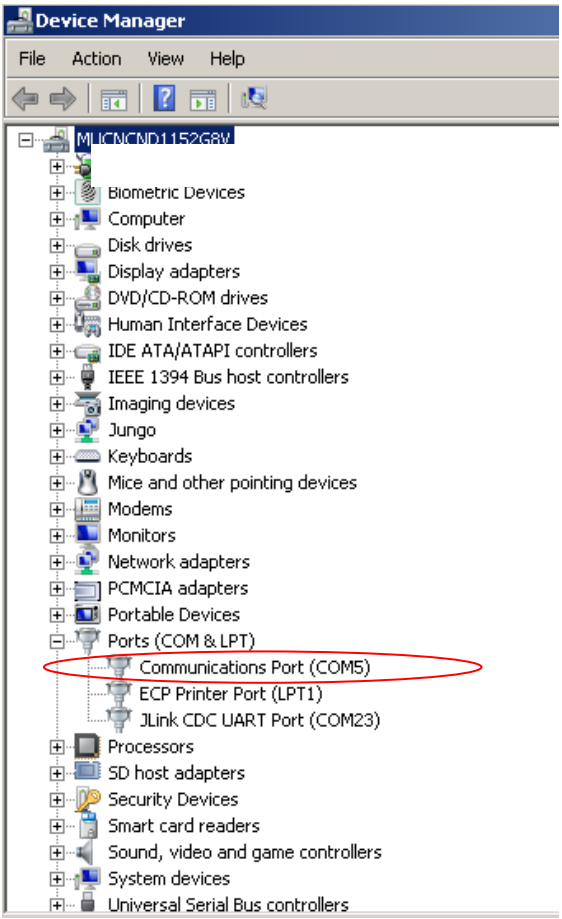
- Complete the Installation

Instructions for connecting to target board:

1. Connect USB Cable from Laptop/PC to below “Debug USB” Connector



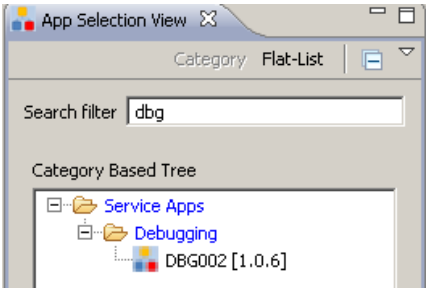
2. Check the serial port assigned (needed later to establish connection)



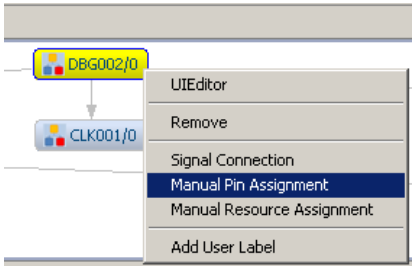
3. Segger Firmware update for board is necessary for VCOM communication. This will be prompted during debug/download of Target ELF/Hex file to target in DAVE3.

3 Adding DBG002 App to a project

1. Create a new DAVE CE project and add DBG002 to your project. In the App Selection View, click on the DBG002 App.

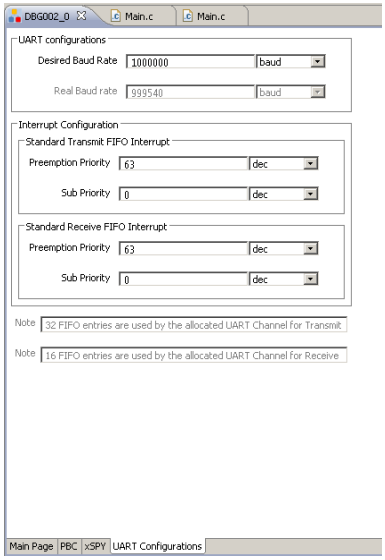


2. Configure DBG002 App
   a. Right click on the DBG002 App. Select the pins used in your board for UART communication



	UART Tx	UART Rx
XMC4500 Hexagon Kit	P1.5	P1.4
XMC4400 Hexagon Kit	P1.7	P1.5
XMC4200 Hexagon Kit	P1.5	P1.4
XMC1x Bootkits	P1.2	P1.3

   b. Click on the DBG002app and configure the desired baud rate, i.e. 1MBit/s, under the UART Configurations tab. The interrupt priority for transmit and reception IRQs should be set so that they do not affect the functionality of your application. By default they are set to the lowest level.



   c. Default values for the rest of the parameters will work fine for basic usage.

3. Declare variables available for tracing using the Variable Table.
   For demonstration purposes we split the available variables in two tables. This can be useful if you want to sample variables at two different positions in your code and/or different sampling frequency.

```
#define NO_XSPY_APPS 2
#define NUMBER_VARIABLES_XSPY1 4
DBG002_VariableDescriptionType xSpyTable1[NUMBER_VARIABLES_XSPY1] = {
                XSPY_ADD_FLOAT("Sin Waveform", sinWav),
                XSPY_ADD_FLOAT("Cos Waveform", cosWav),
                XSPY_ADD_UINT32("Sin frequency", sinFreq),
                XSPY_ADD_UINT32("Cos frequency", cosFreq),
};

#define NUMBER_VARIABLES_XSPY2 6
DBG002_VariableDescriptionType xSpyTable2[NUMBER_VARIABLES_XSPY2] = {
                XSPY_ADD_FLOAT("Square Waveform", squareWav),
                XSPY_ADD_FLOAT("Triangle Waveform", triangleWav),
                XSPY_ADD_FLOAT("Sawtooth Waveform", sawtoothWav),
                XSPY_ADD_UINT32("Square frequency", squareFreq),
                XSPY_ADD_UINT32("Triangle frequency", triangleFreq),
                XSPY_ADD_UINT32("Sawtooth frequency", sawtoothFreq)
};

DBG002_XSpyDataType xSpyApps[NO_XSPY_APPS];
```

Hint: A scale factor can be applied to the variables if desired using XSPY_ADD_SCALED_xxxx

4. Initialize the xSPY instances in the body of the main function.

```
DBG002_XspyInit(&xSpyApps[0], xSpyTable1, NUMBER_VARIABLES_XSPY1, DBG002_GID_XSPY1);
DBG002_XspyInit(&xSpyApps[1], xSpyTable2, NUMBER_VARIABLES_XSPY2, DBG002_GID_XSPY2);
```

5. Add the data logging call at the point of interest.
   To sample the data for streaming the DBG002 API provides the function
   *DBG002_XspyStreamData(DBG002_XSpyDataType *);*

   This function can be invoked either in a while loop in the main function, in a dedicated timer interrupt handler or a dedicated App interrupt handler. In our example we add the function at the end of the system timer ISR.

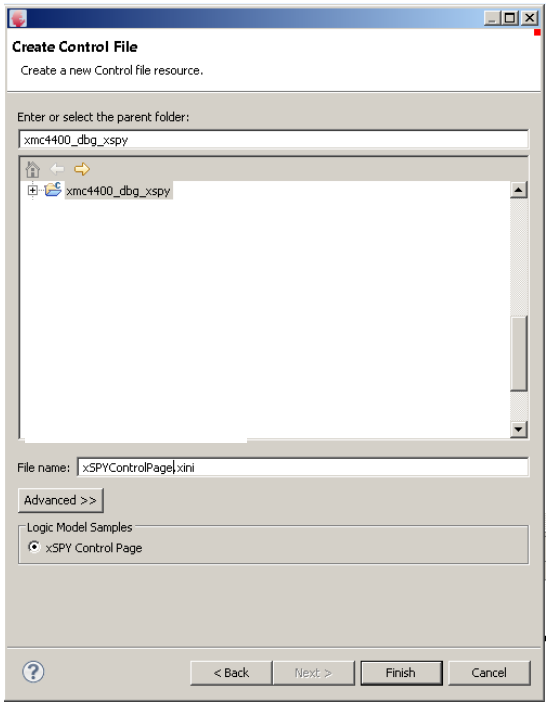
```
void SysTick_Handler(void)
{

        …

        DBG002_XspyStreamData(&xSpyApp[0]);

}
```

6. Click on Generate Code, compile and download the code to the target with the debugger.
7. Start the microcontroller.

4 Designing a control page

8. Add a Control Page to your project.
   Change back to DAVE CE perspective

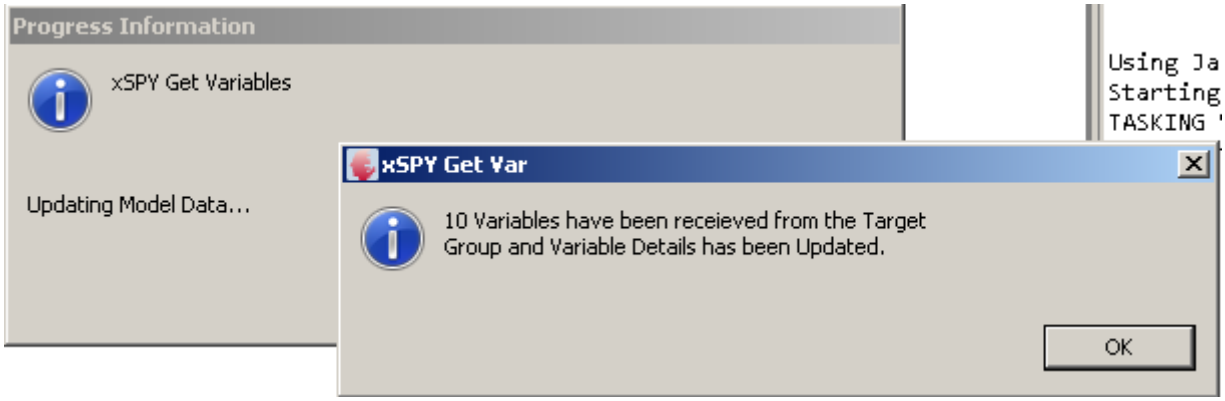
Left click on your project, click on New->Other-> xSPY Control Page.
Set filename and finally click Finish.



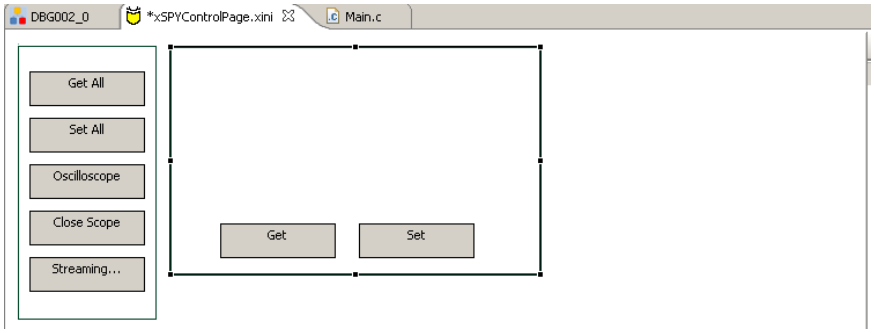
The Eclipse perspective is changed to xSPY Design Mode.

9. Select the COM Port of the JLINK CDC UART and the desired Baud Rate. Click on Connect

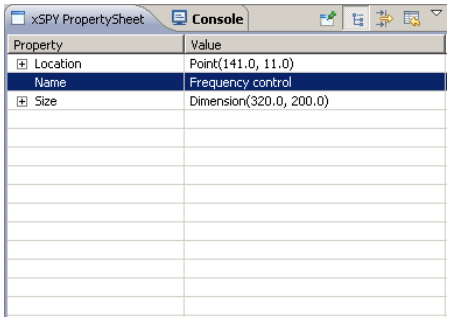
10. If the connection is successful the declared variables in the Variable Table will be available to UI designer.



11. From the Control Palette, drag and drop two Control Frames.

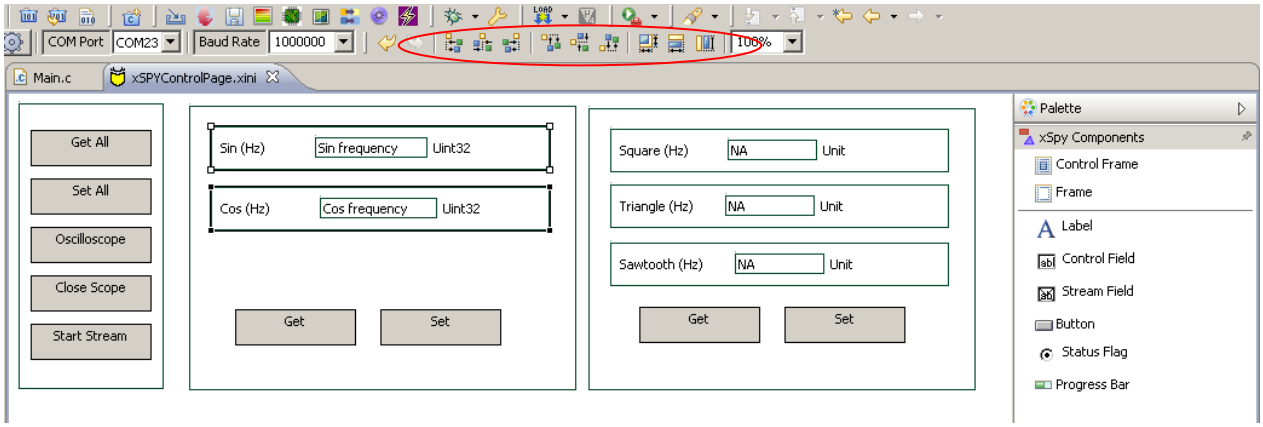


12. Set title to the Control Frame configuring the properties of the control using the xSPY PropertySheet editor on the left.



13. Add Control Fields to the Control Frame.

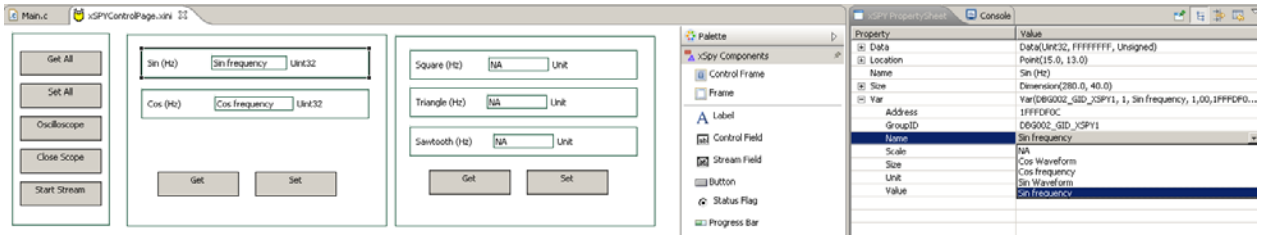
    The control fields allow SET/GET traced variables. In the xSPY PropertySheet editor enter the label of the control fields.



    Hint: Make use of the align buttons in the toolbar.

14. Assign variables to the Control Fields

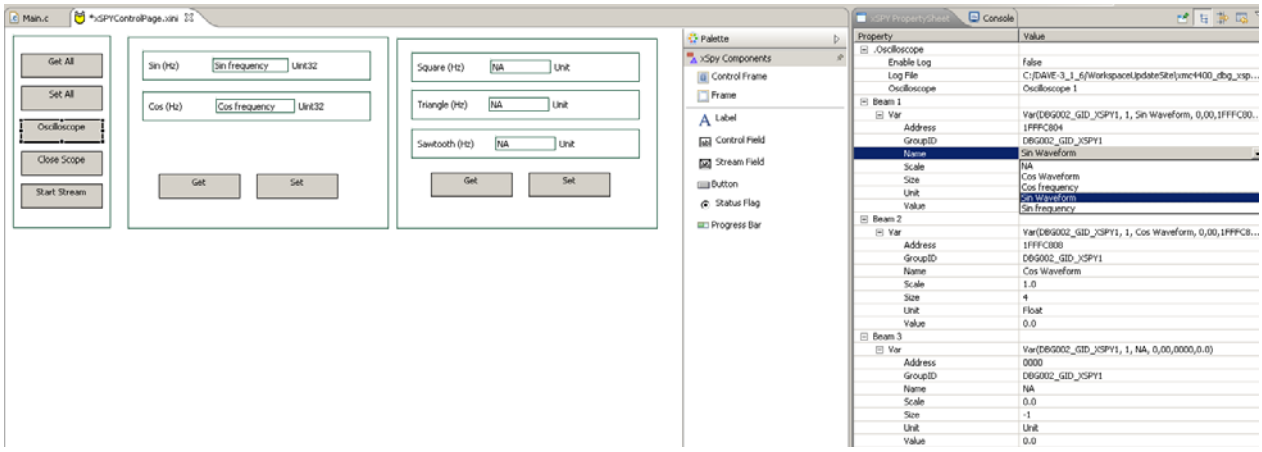
    In the xSPY PropertySheet editor, click on Var, select the Group ID and Name of the variable from the combo boxes.

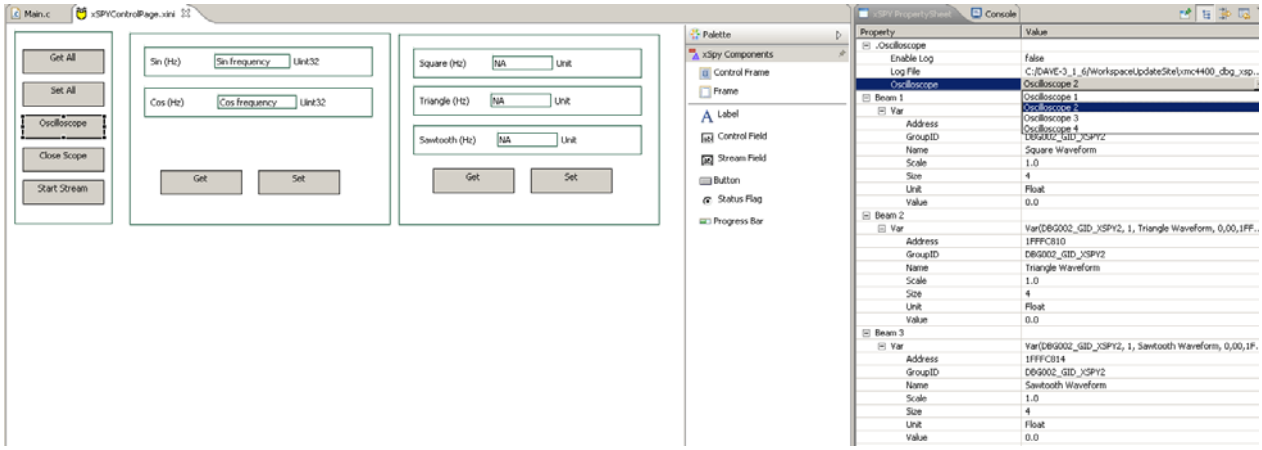


15. Configure the oscilloscopes.

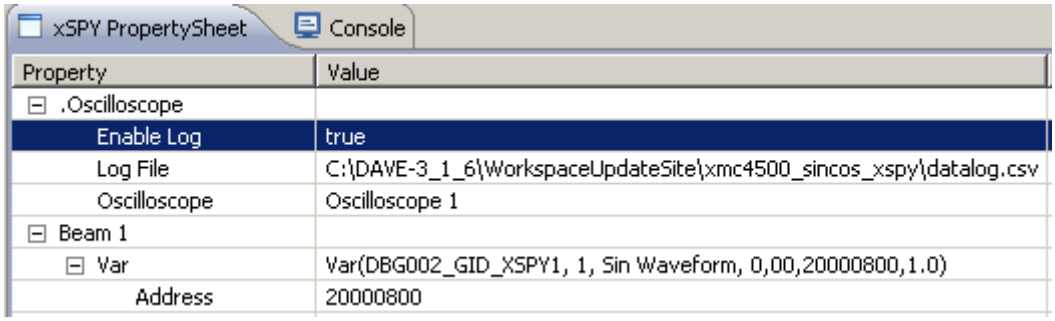
    Click on Oscilloscope button. Select the variables to be scoped in the xSPY PropertySheet editor. Select a variable for every beam (channel) being used.

    Up to 4 oscilloscopes can be configured. In this tutorial we make use of two.





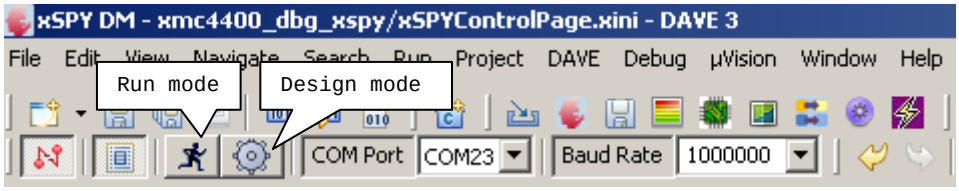
16. Set if desire the name of the logging file that will capture all the samples obtained from the target and enable the logging



17. Press Save All or Ctrl-S and your ready to go and try out your UI.

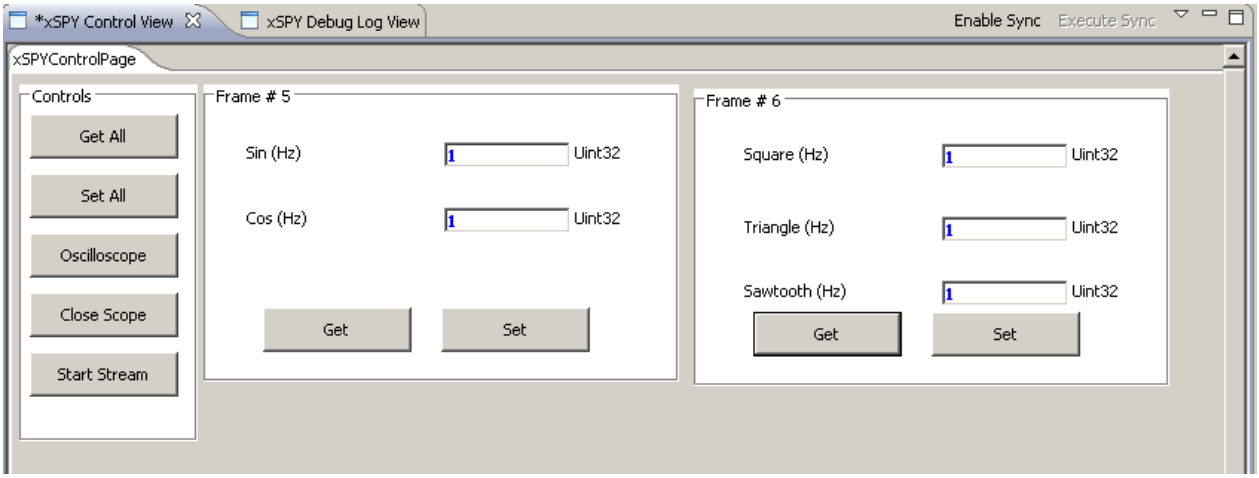
5 Switch to run mode

18. Switch to xSPY Run Mode perspective using the toolbar



19. The perspective will change to the xSPY Run mode

    Click on the GET button. You will get the value of the actual frequency of the different waveforms.

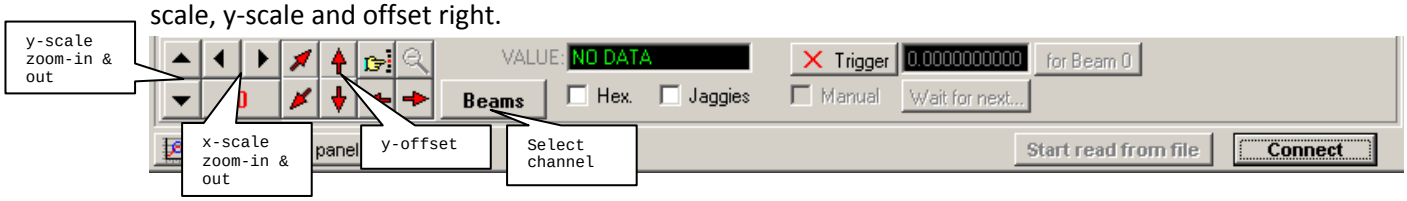


20. Click on Oscilloscope.

    The two oscilloscope windows should pop up.

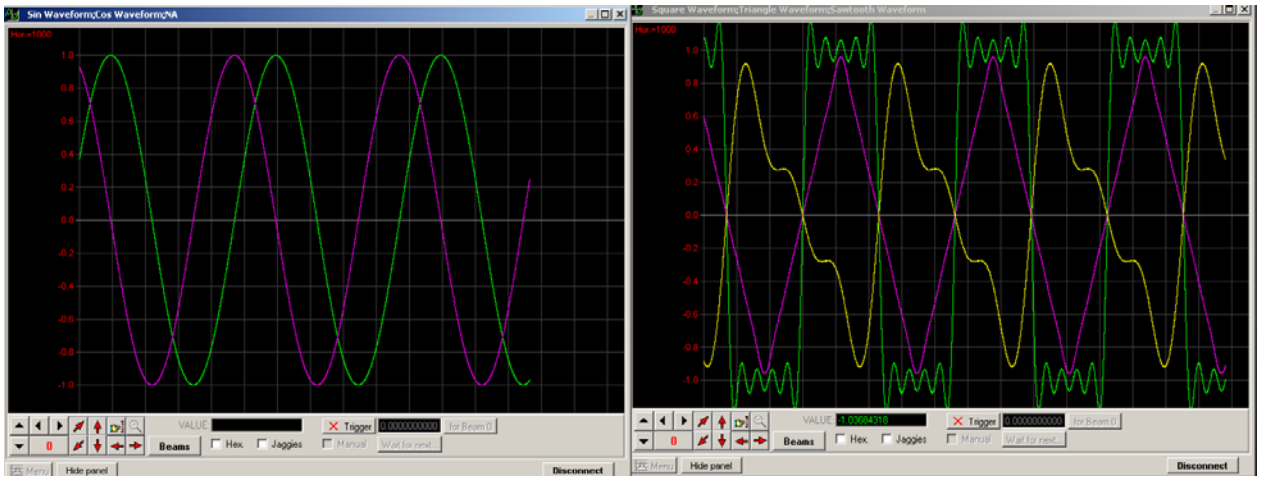
21. Click on Start Stream

    In the oscilloscopes, the waveforms are visible, although some tweaking is needed to get time scale, y-scale and offset right.



22. Finally you get something similar to below figure.

    Try changing the frequency of the signals.



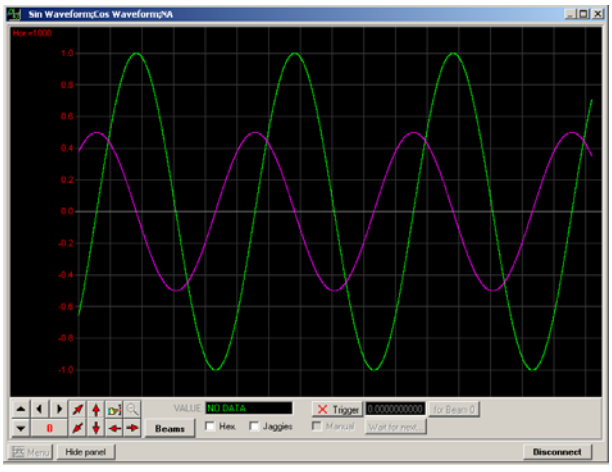
23. Change the scale factor of the signals

    In occasions the value of the variable received from the target need to be scaled. This information is received from the target if the user makes use of the XSPY_ADD_SCALED macros for adding variables to the Variable Table. Otherwise the default scale factor, 1.0, is used. See xSPY PropertySheet editor.

    We will change the scale factor of the cosine wave form to 0.5

xSPY PropertySheet | Console

Property	Value
.Oscilloscope	
Enable Log	False
Log File	C:\DAVE-3_1_6\WorkspaceUpdateSite\xmc4400_dbg_xsp..
Oscilloscope	Oscilloscope 1
Beam 1	
Var	Var(DBG002_GID_XSPY1, 1, Sin Waveform, 0,00,1FFFC80.
Address	1FFFC804
GroupID	DBG002_GID_XSPY1
Name	Sin Waveform
Scale	1.0
Size	4
Unit	Float
Value	0.0
Beam 2	
Var	Var(DBG002_GID_XSPY1, 1, Cos Waveform, 0,00,1FFFC8..
Address	1FFFC808
GroupID	DBG002_GID_XSPY1
Name	Cos Waveform
Scale	0.5
Size	4
Unit	Float
Value	0.0
Beam 3	
Var	Var(DBG002_GID_XSPY1, 1, NA, 0,00,0000,0.0)



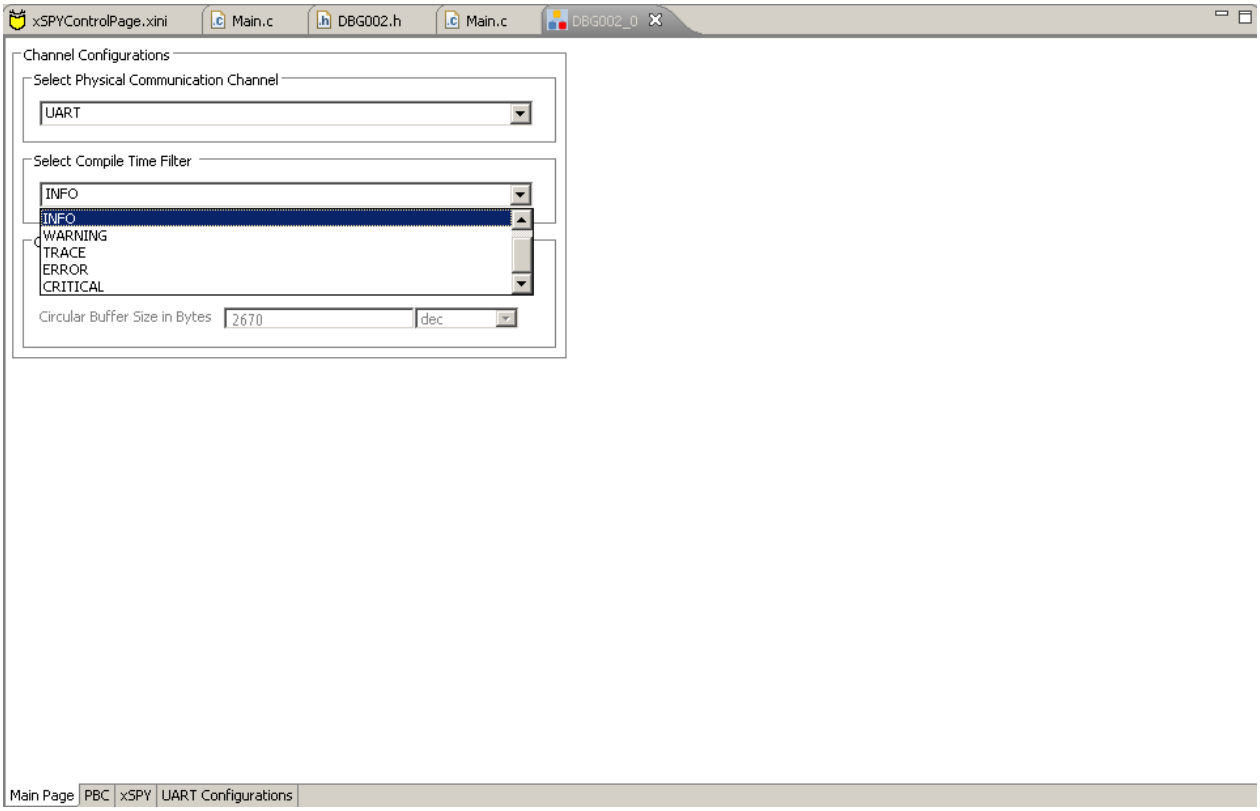
6 Sending debug messages

If you look into the code of the System Timer ISR, you will see that every second a debug message log is generated with severity INFO.

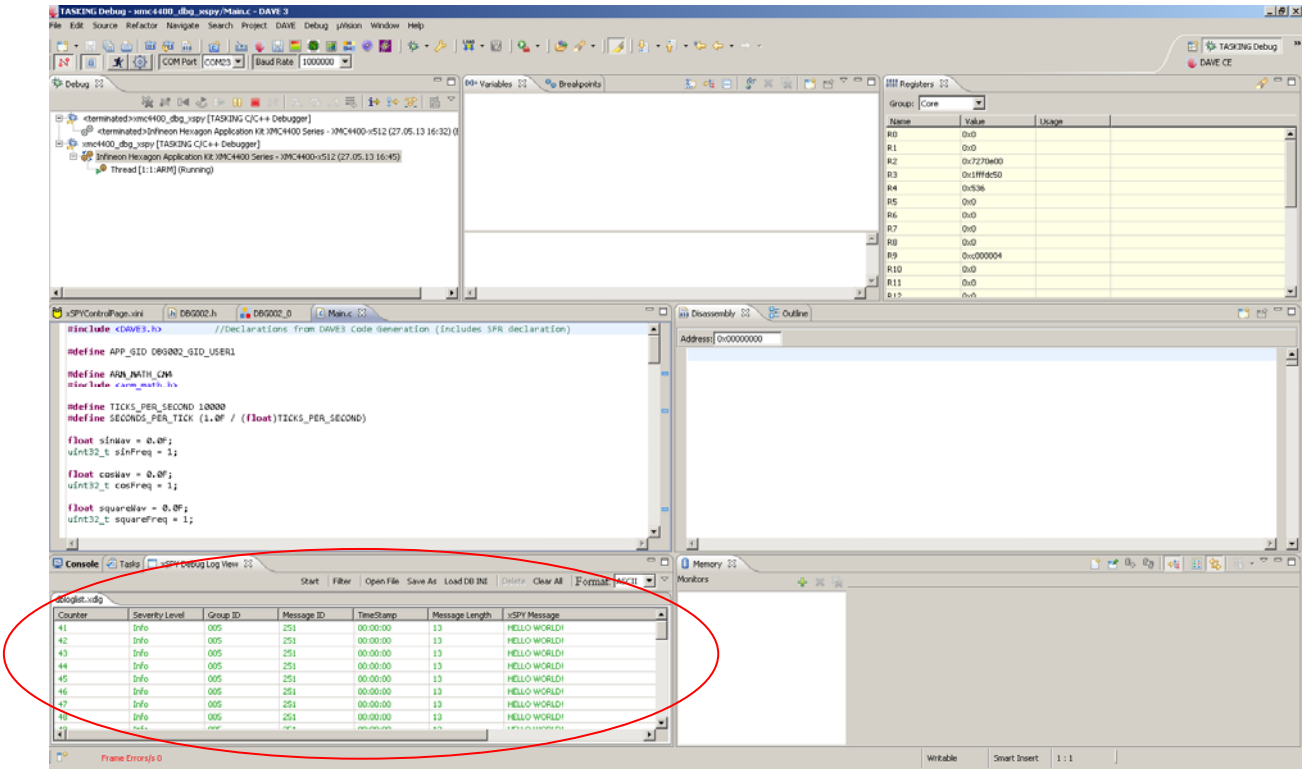
```
#define APP_GID DBG002_GID_USER1
…

void SysTick_Handler(void)
{
    …

    static uint32_t ticks = 0;

    ticks++;

    if (ticks == TICKS_PER_SECOND) {
        DBG002_I(sinFreq > 0);

        DBG002_INFO(DBG002_GID_USER1, DBG002_MESSAGEID_LITERAL, sizeof("HELLO WORLD!"), "HELLO
```

23. Change Compile Time debug message filter

    Go to the DAVE CE perspective. Select S/W App Connectivity View. Double click on the DBG002 App. Change the Compile Time Filter to INFO in the Main Page tab. Regenerate the code, compile and download again. Launch the debugger and resume the target



24. In the debugger perspective, click on Window->Show View->Other->xSPY Debug Log View.



Hint: The view can be attached anywhere.

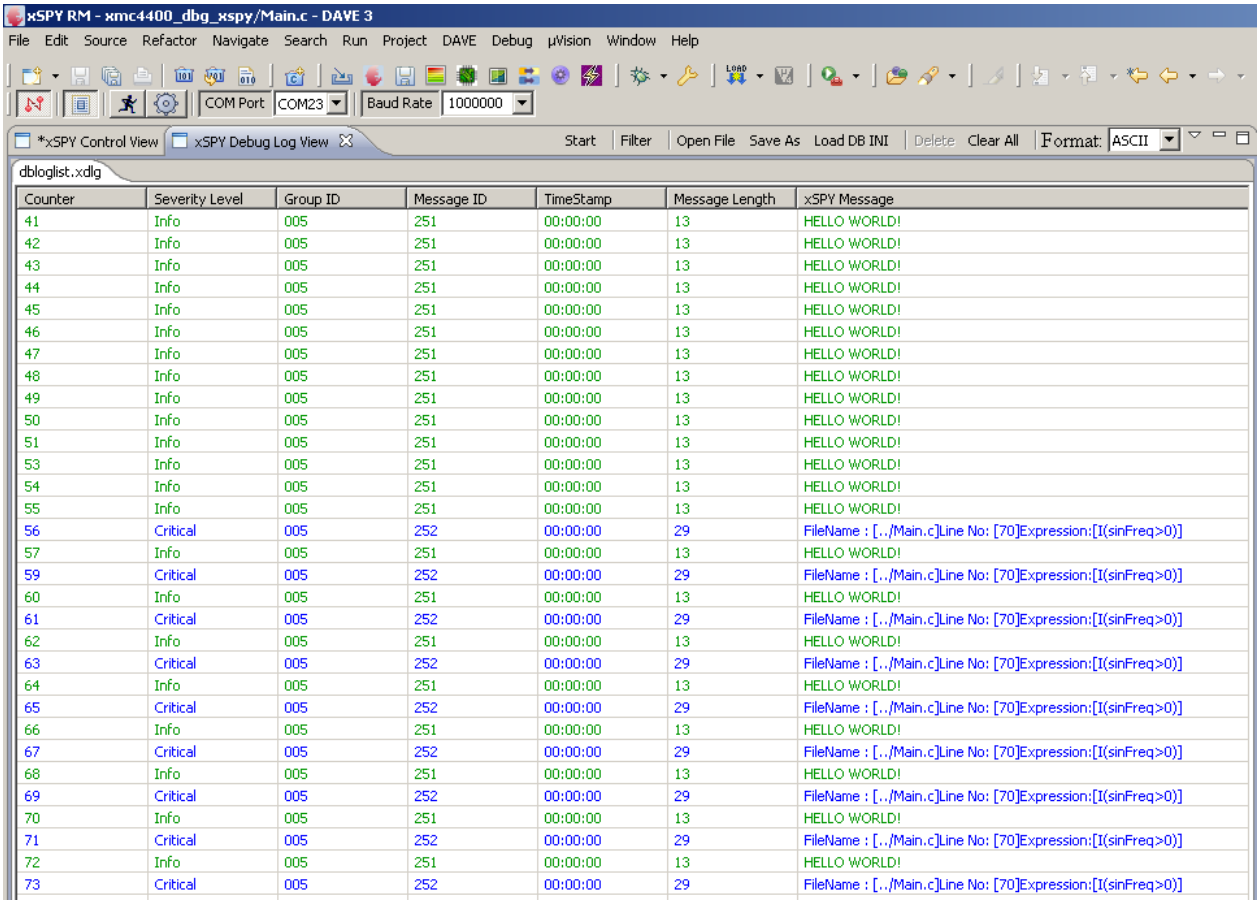
25. Connect to the device using the xSPY toolbar.

    In the xSPY Debug Log View, click on start. You will see a message being sent from the target every second.

26. Check validity of assertions

    Every second also it is checked that the set frequency is greater than zero.

    Change the perspective to xSPY Design Mode. Click on Switch to Run Mode. Change the frequency of the sine waveform to zero. You will see that in the xSPY Debug Log view that a critical error is generated indicating that the assertion is proven false.



Counter	Severity Level	Group ID	Message ID	TimeStamp	Message Length	xSPY Message
41	Info	005	251	00:00:00	13	HELLO WORLD!
42	Info	005	251	00:00:00	13	HELLO WORLD!
43	Info	005	251	00:00:00	13	HELLO WORLD!
44	Info	005	251	00:00:00	13	HELLO WORLD!
45	Info	005	251	00:00:00	13	HELLO WORLD!
46	Info	005	251	00:00:00	13	HELLO WORLD!
47	Info	005	251	00:00:00	13	HELLO WORLD!
48	Info	005	251	00:00:00	13	HELLO WORLD!
49	Info	005	251	00:00:00	13	HELLO WORLD!
50	Info	005	251	00:00:00	13	HELLO WORLD!
51	Info	005	251	00:00:00	13	HELLO WORLD!
53	Info	005	251	00:00:00	13	HELLO WORLD!
54	Info	005	251	00:00:00	13	HELLO WORLD!
55	Info	005	251	00:00:00	13	HELLO WORLD!
56	Critical	005	252	00:00:00	29	FileName : [../Main.c]Line No: [70]Expression:[I(sinFreq>0)]
57	Info	005	251	00:00:00	13	HELLO WORLD!
59	Critical	005	252	00:00:00	29	FileName : [../Main.c]Line No: [70]Expression:[I(sinFreq>0)]
60	Info	005	251	00:00:00	13	HELLO WORLD!
61	Critical	005	252	00:00:00	29	FileName : [../Main.c]Line No: [70]Expression:[I(sinFreq>0)]
62	Info	005	251	00:00:00	13	HELLO WORLD!
63	Critical	005	252	00:00:00	29	FileName : [../Main.c]Line No: [70]Expression:[I(sinFreq>0)]
64	Info	005	251	00:00:00	13	HELLO WORLD!
65	Critical	005	252	00:00:00	29	FileName : [../Main.c]Line No: [70]Expression:[I(sinFreq>0)]
66	Info	005	251	00:00:00	13	HELLO WORLD!
67	Critical	005	252	00:00:00	29	FileName : [../Main.c]Line No: [70]Expression:[I(sinFreq>0)]
68	Info	005	251	00:00:00	13	HELLO WORLD!
69	Critical	005	252	00:00:00	29	FileName : [../Main.c]Line No: [70]Expression:[I(sinFreq>0)]
70	Info	005	251	00:00:00	13	HELLO WORLD!
71	Critical	005	252	00:00:00	29	FileName : [../Main.c]Line No: [70]Expression:[I(sinFreq>0)]
72	Info	005	251	00:00:00	13	HELLO WORLD!
73	Critical	005	252	00:00:00	29	FileName : [../Main.c]Line No: [70]Expression:[I(sinFreq>0)]

7 Troubleshooting Guide

Situations	Possible Causes
Not able to connect to target	1. Is the target running? 2. Verify SEGGER driver v4.70 or above is installed. 3. Verify the On Board Debugger Chip firmware is updated 4. Verify no higher priority interrupt ISR is stalling the CPU
No stream data received from target	1. Verify no higher priority interrupt ISR is stalling the CPU 2. Verify DBG002_XspyStreamData() is called regularly
Oscilloscope not popping up	1. Verify the streaming is stopped. 2. Verify the user has enough rights to access the path of the logging file specified in the xSPY PropertySheet for the oscilloscope control.
Variable table empty	1. Is the variable table correctly defined?
No debug logging messages in xSPY Debug Log view	1. Check the compile time message filter settings of the DBG002
Status bar Frame Errors/s in run mode bigger than zero steadily	1. It is an indication of buffer overrun. Please reduce the sampling or debug log messages generation rate