



OTX DSP C54x SDK Debugging Guide

Doc. No. 3112-1-SAA-1007-1

Rev. 1.0-P2

March 28, 2001

Copyright

Copyright (C) Odin TeleSystems Inc., 2000-2001. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior written consent of Odin TeleSystems Inc., 800 E. Campbell Road, Suite 334, Richardson, Texas, 75081-1873, U. S. A.

Trademarks

Odin TeleSystems, the Odin Logo and OTX are trademarks of Odin TeleSystems Inc., which may be registered in some jurisdictions. Other trademarks are the property of their respective companies.

Changes

The material in this document is for information only and is subject to change without notice. While reasonable efforts have been made in the preparation of this document to assure its accuracy, Odin TeleSystems Inc., assumes no liability resulting from errors or omissions in this document, or from the use of the information contained herein.

Odin TeleSystems Inc. reserves the right to make changes in the product design without reservation and notification to its users.

Warranties

THE PRODUCT AND ITS DOCUMENTATION ARE PROVIDED "AS IS" AND WITHOUT WARRANTY OF ANY KIND. ODIN TELESYSTEMS EXPRESSLY DISCLAIMS ALL THE WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR PARTICULAR PURPOSE. ODIN TELESYSTEMS DOES NOT WARRANT THAT THE FUNCTIONALITY OF THE PRODUCT WILL MEET ANY REQUIREMENTS, OR THAT THE OPERATIONS OF THE PRODUCT WILL BE UNINTERRUPTED OR ERROR-FREE, OR THAT DEFECTS WILL BE CORRECTED. FURTHERMORE, ODIN TELESYSTEMS DOES NOT WARRANT OR MAKE ANY REPRESENTATIONS REGARDING THE USE OF THE PRODUCT OR ITS DOCUMENTATION IN TERMS OF THEIR CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE. NO ORAL OR WRITTEN INFORMATION OR ADVICE GIVEN BY ODIN TELESYSTEMS OR ODIN TELESYSTEMS' AUTHORIZED REPRESENTATIVE SHALL CREATE A WARRANTY. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO THE ABOVE EXCLUSION MAY NOT APPLY.

UNDER NO CIRCUMSTANCE SHALL ODIN TELESYSTEMS INC., ITS OFFICERS, EMPLOYEES, OR AGENTS BE LIABLE FOR ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES (INCLUDING DAMAGES FOR LOSS OF BUSINESS, PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THE PRODUCT AND ITS DOCUMENTATION, EVEN IF ODIN TELESYSTEMS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT WILL ODIN TELESYSTEMS' LIABILITY FOR ANY REASON EXCEED THE ACTUAL PRICE PAID FOR THE PRODUCT AND ITS DOCUMENTATION. SOME JURISDICTIONS DO NOT ALLOW THE LIMITATION OR EXCLUSION OF LIABILITY FOR INCIDENTAL AND CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY.



Odin TeleSystems Inc.
<http://www.OdinTS.com>

This document is published by:
Odin TeleSystems Inc.
800 East Campbell Road, Suite 334
Richardson, Texas 75081-1873
U. S. A.

Printed in U. S. A.



1. Abstract

This document describes how to debug DSP SDK (Software Development Kit) applications in the OTX (Odin Telecom FrameworX) Hardware driver. It describes how to connect and configure a DSP debug session. It assumes that the reader already has general knowledge about the overall structure of the OTX Hardware Driver (as described in the Programmer’s Guide for OTX Hardware API, Doc. No. 1412-1-SAA-1006-1) and the overall structure of the DSP SDK (as described in Programmer's Guide for OTX DSP C54x Software Development Kit, Doc. No. 1412-1-SAA-1007-1).

2. Table of Content

1.	Abstract.....	3
2.	Table of Content.....	3
3.	Debugging without a Debugger.....	4
4.	Debugging with a Debugger.....	4
4.1	Required Software and Hardware for Code Composer.....	4
4.2	Configuring a Debug Session with Code Composer.....	5
4.3	Running a Debug Session with Code Composer.....	6



3. Debugging without a Debugger

Debugging without a Debugger might sound like a contradiction in terms. However, it is referring to a debug method which can be applied without any additional hardware or software, other than the OTX Nic board, the OTX Hardware driver, and the OTX DSP SDK.

All of the DSP SDK applications define a variable called `g_debugDataS`. This variable is a struct containing field which can be assigned values at different places in the code. Below is an example of the debug struct used in the `DspDemo1` example program:

```
typedef struct Demo1NotifyGetDebugDataS {  
    unsigned short    nData0;  
    unsigned short    nData1;  
    unsigned short    nData2;  
    unsigned short    sqBuf[300];  
} Demo1NotifyGetDebugDataS;
```

The DSP code sets the different fields in the struct and passes the struct the host application (by calling `OtxDspSetHostEvent()` with the `DEMO1_NOTIFY_DEBUG_DATA` IO Control code.) The DSP code can either send the Debug struct as a response to a host initiated DSK IO Control command (as in the `DspDemo1.c` application) or the DSP code can send the Debug struct as an unsolicited event (e.g. when the DSP code detects a certain error condition, or periodically when a DSP timer expires).

4. Debugging with a Debugger

If the debug method described in Chapter 3: "Debugging without a Debugger" is inefficient, there is another, more extensive, way to debug. The method involves acquiring of a third-party debugger (Code Composer), an Emulator board, and an OTX Emulator adapter board.

4.1 Required Software and Hardware for Code Composer

The OTX DSP SDK applications has been debugged using the following software and hardware.

Software:

- Windows 95 or later operating system
- Code Composer version 4.02 by Texas Instruments
- Emulator Board driver, `ge54xwd32.dll`, Version 1.0.0.2, by DSP Research
- OTX Hardware Driver, and OTX DSP SDK



Hardware:

- OTX Nic boards (Vidar-5x4-ASM and Vidar-5x16-PCI)
- OTX Hermod Active Emulator Adapter board (HMA-1057-1)
- Tiger Emulator board for C54x by DSP Research (ISA Bus)

4.2 Configuring a Debug Session with Code Composer

Before a Code Composer debug session can be started Code Composer needs to be configured. Code Composer needs the following information:

- Emulator board driver
- Number of DSPs on the target

This information is entered using the CC_SETUP.EXE (supplied with Code Composer) utility.

To configure Code Composer, please see the following configuration flow:

1. Install the Emulator board (by DSP Research) in an empty ISA slot.
2. Boot the PC in Windows 95.
3. Install the Code Composer software
4. Copy the Emulator Board driver (ge54xwd32.dll; supplied by DSP Research) to the %WINDIR%\ti\drivers directory.
5. Start CC_SETUP.EXE.
6. Drag the ge54xwd32 object from “Available Board Types” window pane and drop it on the “My System” object in the “System Configuration” window pane.
7. Right-click on the ge54xwd32 object in “System Configuration” window pane. Choose “Properties”
8. Select the “Board Name & Data File” tab.
9. Enter a more descriptive board name in the “Board Name” field (e.g. Vidar-5x4-ASM or Vidar-5x16-PCI).
10. Select the “Board Properties” tab.
11. Enter the IO address to be used for the Emulator board. This address must match the settings of the Dip switch S1 on the Emulator board. E.g. 0x220, which means that all four positions of S1 should be in the OFF position.
12. Select the “Processor Configuration” tab.



13. Click on “TMS320C54xx” in the list of “Available Processors”.
14. Enter a name for the processor in the “Processor Name” field (e.g. DSP_0 for the first DSP). Click on “Add Single”. Repeat this step for every processor on the OTX Nic or ASM board (i.e. 4 processors for Vidar-5x4-ASM and 16 processors for Vidar-5x16-PCI).
15. Close the Board Properties dialog by clicking “Finish”.
16. At this point it is recommend to save this configuration in a Code Composer Setup file (*.ccs). To to this, choose File/Export from the menu, enter a name for the configuration file (e.g. Vidar-5x4-ASM.ccs), and click Save.

Code Composer is now configured. Now the hardware needs to be connected. To do this, please see the following flow:

1. Set the dial on the OTX Hermod Emulator Adapter to the correct JTAG chain (which varies depending on the target OTX Nic or ASM board). Vidar-5x4-ASM uses JTAG chain 3 and Vidar-5x16-PCI uses JTAG chain 7.
2. Attach the OTX Hermod Emulator Adapter to BJ3 connector on the OTX Nic.
3. Attach the Emulator board cable (ribbon cable) to the Emulator connector (J4) on the Hermod Emulator Adapter. The connector is keyed and will only fit in one direction.
4. Attach the other end of the Emulator board cable to the Emulator board (in the Code Composer PC).

The hardware and software debug configuration is now complete.

4.3 Running a Debug Session with Code Composer

The following flow describes how to start a debugging session with Code Composer using the DspDemo1 demo program (using a Thor-2-PCI and Vidar-5x4-ASM).

1. Make sure that host application PC (where the Thor-2-PCI/Vidar-5x4-ASM is installed) is properly connected to the Code Composer PC (via the Emulator cable).
2. Compile the DspDemo1.out program with Symbolic Debug information (the -g command line switch to CL500.EXE)
3. Copy the DspDemo.out file to the directory where DspDemo1.exe is located (e.g. Demos\DspSdk\Pci\DspDemo1\W32\Release).
4. Start DspDemo1.exe on the OTX Target PC.

DspDemo1 /D



The /D command line switch is important to make the program stop right after it has made the OtxDspRunProgram() API call.

5. Start Code Composer on the Code Composer PC.
6. Choose “File/Load Program” from the Code Composer (Parallel Debug Manager) menu. Open the DspDemo1.out file from the location where it was compiled (e.g. Dsp\C54x\DspDemo1\DspDemo1.out). Code Composer should now load this DSP program to all four DSPs on the Vidar-5x4-ASM.
7. Click on the “Run” button in the Code Composer (Parallel Debug Manager) window. All four “heart-beat” LEDs on Vidar-5x4-ASM should now start blinking.
8. Once all DSPs are started from Code Composer, press any key in the DspDemo1.exe program (on the host application PC). DspDemo1 will now call the OtxDspRestartProgram() API function. This is an important step since it resynchronizes the DSP code (started by Code Composer) with the OTX Hardware driver. OtxDspRestartProgram() resets the HPI communication buffers, and send the OTXDSP_SDK_IO_START_PROGRAM and OTXDSP_SDK_IO_INIT_TIMER DSP IO Control codes to the DSP.
9. At this point it is safe to open a specific DSP window from the Code Composer (Parallel Debug Manager) window (e.g. Vidar-5x4-ASM/DSP0). The DSP code can be halted and breakpoints and watches can be set.

When the Code Composer debugging session is to be terminated it is important to follow a couple of rules:

- Make sure that all the DSP are running (not halted) when the host application is exited.
- If the host application is to be restarted without Code Composer (without the /D switch) make sure that Code Composer (Parallel Debug Manager) is closed before restarting the host application.

Doc. No. 3112-1-SAA-1007-1

For more information on this product, please contact:

Odin TeleSystems Inc.
800 East Campbell Road, Suite 334
Richardson, Texas 75081-1873
U. S. A.

Tel: +1-972-664-0100
Fax: +1-972-664-0855
Email: Info@OdinTS.com
URL: <http://www.OdinTS.com/>

Copyright (C) Odin TeleSystems Inc., 2000-2001