

---

**INSTALLATIONGUIDE**

---

**DSP/BIOS™ LINK****DM6446 Media Processor****LNK 110 USR****Version 1.65.00.03****JUL 12, 2010**

This page has been intentionally left blank.

---

## **IMPORTANT NOTICE**

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Mailing Address:  
Texas Instruments  
Post Office Box 655303  
Dallas, Texas 75265

Copyright ©. 2003, Texas Instruments Incorporated

This page has been intentionally left blank.

---

**TABLE OF CONTENTS**

---

<b>A.</b>	<b>INTRODUCTION.....</b>	<b>6</b>
1	Purpose.....	6
2	Text Conventions.....	6
3	Terms & Abbreviations .....	6
4	References .....	6
<b>B.</b>	<b>INSTALLATION .....</b>	<b>7</b>
5	Basic Installation.....	7
5.1	Installing Standalone DSP/BIOS™ and CGTools.....	7
6	Creating development workspace.....	9
6.1	Tools Directory Structure Layout (Default).....	9
6.2	Tools Directory Structure Layout (User-Defined).....	9
7	Setting up Windows Workstation .....	11
8	Configuring CCS.....	11
8.1	DM6446 EVM .....	11
<b>C.</b>	<b>WORKING ON TARGET PLATFORM .....</b>	<b>12</b>
9	DM6446 EVM .....	12
9.1	Configure the DSP/BIOS LINK for DM6446.....	12
9.2	Build the DSP/BIOS LINK.....	12
10	Running the sample applications.....	16
10.1	Copying files to target file system.....	16
10.2	Invoking the application .....	16
<b>D.</b>	<b>ADDITIONAL INFORMATION .....</b>	<b>19</b>

## A. INTRODUCTION

### 1 Purpose

DSP/BIOS™ LINK is foundation software for the inter-processor communication across the GPP-DSP boundary. It provides a generic API that abstracts the characteristics of the physical link connecting GPP and DSP from the applications. It eliminates the need for customers to develop such link from scratch and allows them to focus more on application development.

This document provides the users necessary information to install DSP/BIOS™ LINK on the development host.

This document corresponds to the product release Version 1.65.00.03 dated JUL 12, 2010.

### 2 TextConventions

	This bullet indicates important information. Please read such text carefully.
	This bullet indicates additional information.
[ arg1   arg2 ]	In context of the commands, contents enclosed in square brackets are the optional arguments to the command. Different values of these arguments are separated by " ".

### 3 Terms&Abbreviations

CCS	Code Composer Studio
IPC	Inter Processor Communication
GPP	General Purpose Processor e.g. ARM
DSP	Digital Signal Processor e.g. TMS320C5510
CGTools	Code Gen Tools, e.g. Compiler, Linker, Archiver

### 4 References

1.	Various	Documentation included with the DM6446 hardware.
----	---------	--

---

## **B. INSTALLATION**

### **5 BasicInstallation**

The DSP/BIOS™ LINK is made available as a tar.gz file. To install the product follow the steps below:

1. Unzip and untar the file dsplink\_<version>.tar.gz.

 *This document assumes the install path to be L:\dsplink if working on a Windows PC. This path will be used in remainder of this document. A user can install the dsplink in any convenient location and the dsplink scripts will adapt to that location automatically (without any changes)*

It is advisable to archive the released sources in a configuration management system. This will help in merging:

- The updates delivered in the newer releases of DSP/BIOS™ LINK.
- The changes to the product, if any, done by the users.

#### **5.1 Installing Standalone DSP/BIOS™ and CGTools**

The compilation of this release is dependent on following components:

1. CGTools (v 6.0.18)
2. DSP/BIOS (v 5.32.03 )
3. XDCTools (v 3.10.05)
4. Windows Embedded CE 6.0 (including Visual Studio 2005 + Platform Builder R2)
5. WinCE BSP for DM6446
6. Perl (ex: ActiveState Perl)

For compilation of DSP-side sources and applications for Davinci DM6446, the CGTools version 6.0.18 can be used. This release has been validated with DSP/BIOS™ version 5.32.03.

**This release will not work with BIOS version earlier than 5.32.01.**

Refer to the URL mentioned below for getting the distribution of DSP/BIOS™ and the associated installation instructions.

[https://www-a.ti.com/downloads/sds\\_support/targetcontent/bios/index.html](https://www-a.ti.com/downloads/sds_support/targetcontent/bios/index.html)

For compilation of GPP side sources and applications for DM6446, WINCE environment is used. This release requires Windows CE installation on the host computer (Visual Studio 2005 + Platform Builder R2) and a WinCE BSP for DM6446. This release has been validated using Mistral BSP.

The GPP side and DSP side make system also requires Perl to be installed on your build system.

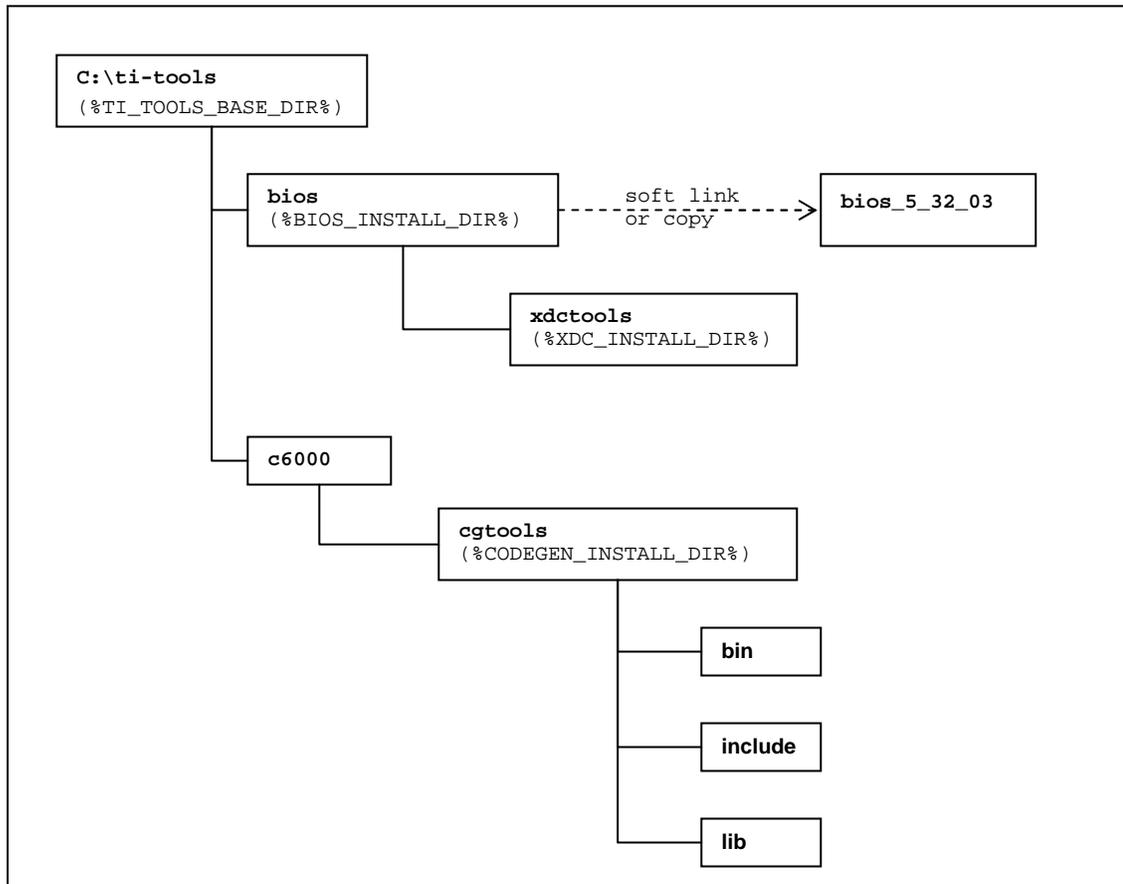
### 5.1.1 ToolsInstallationDirectory

Please refer to the next section for details on the directory layout of these various tools.

## 6 Creating development workspace

### 6.1 ToolsDirectoryStructureLayout(Default)

This document and the scripts included in the release assume the following directory structure on your development host. If your tools aren't laid out as shown, please see section 6.2 for more details.



**Figure 1: Expected directory structure on development host**

For the Windows development host, the build system shipped with DSP/BIOS™ LINK assumes that the standalone DSP/BIOS™ is installed in the 'ti-tools\' directory on the <ROOT-DRIVE> and CGTools is installed in the 'ti-tools\c6000\' directory on the <ROOT-DRIVE>. **If this is not true, see the Section 6.2 below or edit the distribution files.**

To support multiple installations of DSP/BIOS with a single DSP/BIOS™ LINK DSP-side distribution file, a standard c:\ti-tools\bios directory is used for the BIOS installation. This can be a soft link or copy to the actual DSP/BIOS installation directory.

### 6.2 ToolsDirectoryStructureLayout(User-Defined)

The directory structure specified in Figure 1 is expected by the build system of DSP/BIOS™ LINK. If you install the tools to a different directory, you can define the

following environment variables on your desktop (My Computer -> Right Click -> Properties -> Advanced -> Environment Variables -> System Variables):

ENVvariablename	Purpose	Example	Default
<b>TI_TOOLS_BASE_DIR</b>	Defines the based directory for the TI Tools. This should be defined only when your TI Tools are not under C:\ti_tools as the scripts and make file expect. Refer to Figure 1 for more information.	c:\my_tools\ti_tools	C:\ti-tools
<b>BIOS_INSTALL_DIR</b>	Defines the based directory where the bios is located. This should be defined only when bios is not located under C:\ti-tools\bios or \$(TI_TOOLS_BASE_DIR)\bios	c:\my_tools\bios_5_32_03	C:\ti-tools\bios
<b>XDC_INSTALL_DIR</b>	Defines the based directory of XDC tools. This should be defined only when the XDC tools are not located in xdc tools directory under your bios installation (i.e. \$(BIOS_INSTALL_DIR)\xdc tools).  <i>Note: XDCROOT is now obsolete and is no longer used by DSPLINK. Please use XDC_INSTALL_DIR</i>	C:\xdc tools	C:\ti-tools\bios\xdc tools
<b>CODEGEN_INSTALL_DIR</b>	Defines the based directory (not the bin directory) for your CG Tools installation. This should be defined only when the CG Tools are not located under C:\ti-tools\c6000\cg tools or \$(TI_TOOLS_BASE_DIR)\C6000\cg tools  <i>Note: CGTOOLS_INSTALL_DIR is now obsolete and is no longer used by DSPLINK. Please use CODEGEN_INSTALL_DIR</i>	C:\C6000_v3_10_05	C:\ti-tools\C6000\cg tools
<b>BASE_PERL</b>	Defines the based directory (not the bin directory) for your Perl installation	C:\my_tools\Perl	C:\Perl

Alternatively, you can also modify the make system and the scripts contained in the release package. You may need to copy the directories to create the structure expected for compiling sources. Refer to section on "Understanding the MAKE System" in the User Guide for details.

## 7 Setting up Windows Workstation

The dsplink tar ball can be extracted in any convenient location and the dsplink scripts will adapt to that location automatically (without any changes). For command line builds, one needs to run the dsplinkenv.bat for every new build window so that the DSPLINK path is set correctly.

**ASSUMPTION:** Dsplink build process for Wince assumes that an OS-design has been created and setup using Platform Builder. Please refer to your BSP guide or MSDN website on how to create an OS-Design.

## 8 Configuring CCS

### 8.1 DM6446EVM

To use CCS for debugging the DSP side application, you will need to configure CCS to use both ARM and DSP with the DM6446 EVM. The EVM must be configured to use the DSP self-boot mode.

- CCS can attach to only ARM in the beginning. It can attach to the DSP only after the ARM-side application releases it from reset through a call to `PROC_Start ()`.*

## C. WORKING ON TARGET PLATFORM

### 9 DM6446EVM

#### 9.1 Configure the DSP/BIOS LINK for DM6446

The build configuration command must be executed to configure DSPLink for the various parameters such as platform, GPP OS, build configuration etc.

```
DM6446 (WinCE)
dsplinkcfg.bat --platform=DAVINCI --nodsp=1 --dspcfg_0=DM6446GEMSHMEM --
dspos_0=DSPBIOS5XX --gppos=WINCE --comps=ponslm
```

- For details please refer user guide.

#### 9.2 Build the DSP/BIOS LINK

The DSP/BIOS Link can be built using two different methods – command line or Platform Builder IDE. The command line builds involves editing a few BSP related files. The IDE build process is much more streamlined and simply involves adding and building a pre-existing Subproject (provided with DSP/BIOS Link installation) to your OS-Design.

##### 9.2.1 Command Line Build

Following are the steps to build the DSP/BIOS Link using command line build system.

1. Set the environment variables as described in Section 5.1 **OR** Modify the distribution file to update the BASE\_BUILDOS (Base directory for the GPP OS) and BASE\_TOOLCHAIN (Base for toolchain)

```
notepad %DSPLINK%/make/WinCE/davinci_wince6.0.mk
```

2. Open your OS-design project for your DM6446 in Platform Builder. Build the BSP. Now go to **Build Menu -> Open Release Directory in build window**. This will start the DOS command window with WINCE environment set.

3. Set the environment

```
<dsplink_path>\etc\host\scripts\msdos\dsplinkenv.bat
```

4. Run the configuration command.

```
dsplinkcfg.bat --platform=DAVINCI --nodsp=1 --
dspcfg_0=DM6446GEMSHMEM --dspos_0=DSPBIOS5XX --gppos=WINCE --
comps=ponslm
```

5. Build the GPP sources and samples. One needs to build only one of the configurations – release or debug. Debug builds generate pdb files that can be used in debugging using platform builder; however debug builds affect performance. Release builds are optimized for speed and hence provide good performance but no debugging capability.

The make system for GPP side copies the dlls and exes to the %\_TARGETPLATROOT%\target\%\_TGTCPU%\%WINCEDEBUG%. Since it uses the %WINCEDEBUG% variable, there is no relation between the build configuration of your OS-design and dsplink. Thus you can have a release configuration for your OS-Design (BSP+Kernel) but still build and build a debug configuration for DSPLink.

```
cd %DSPLINK%/gpp/src
make -s release
make -s debug
or
make -s

cd $(DSPLINK)/gpp/src/samples
make -s release
make -s debug
or
make -s
```

6. Build the DSP sources and samples

```
cd $(DSPLINK)/dsp/src
make -s release
make -s debug

cd $(DSPLINK)/dsp/src/samples
make -s release
make -s debug
```

7. Edit the platform.bib file for your BSP

```
notepad %_TARGETPLATROOT%\FILES\platform.bib
```

and add the following line at any convenient location. Make sure it is not inside any other IF flags.

```
dsplinkk.dll      $(_FLATRELEASEDIR)\dsplinkk.dll      NK SHK
```

8. Edit the platform.reg file for your BSP

```
notepad %_TARGETPLATROOT%\FILES\platform.reg
```

and add the following line at any convenient location. Make sure it is not inside any other IF flags.

```
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\DSPLINK]
    "Prefix"="DBL"
    "Dll"="dsplinkk.dll"
    "Index"=dword:1
```

9. Now go back to your Platform Builder window and select **"Build Current BSP and Subprojects"** from the **Build -> Advanced Build Commands** menu. The GPP dsplink dlls will now be included in your NK.bin image.

10. Follow the BSP instructions to connect to target, load the NK.bin image and boot-up. The GPP side dsplinkk.dll will be loaded automatically at boot-up

time.

## 9.2.2 Building Platform Builder IDE

**Important Note:** It is possible that pre-built dsplink binaries are provided by the BSP. Make sure that the inclusion of these binaries is disabled.

Following are the steps to build the DSP/BIOS Link using Platform builder build system.

1. Set the environment variables as described in Section 5.1 **OR** Modify the distribution file to update the BASE\_BUILDOS (Base directory for the GPP OS) and BASE\_TOOLCHAIN (Base for toolchain). This is an important step to successfully build both GPP and DSP side sources.

```
notepad %DSPLINK%/make/WinCE/davinci_wince6.0.mk
notepad %DSPLINK%/make/DspBios/c64xxp_5.xx_windows.mk
```

2. Open your OS-design project for your DM6446 in Platform Builder.
3. Add the provided DspLink Sub-project to your OS-Design solution.

```
Go to Menu Project-> Add existing project
Browse to the %DSPLINK%\etc\host\projects\WinCE\DSPLink.
Select DSPLink.pbxml.
This will add the subproject to your solution.
```

3. Add the Catalog Sysgen variable for DspLink

```
copy %DSPLINK%\etc\host\projects\WinCE\CATALOG\dsplink.PbcXml
%_WINCEROOT%\PLATFORM\

```

4. Add the Environment variable to configure the dsplink for your platform. The value of this variable is same as the list of arguments that are given to dsplinkcfg.bat

```
In the Solution Explorer view, right click on your project and
select properties. For the current configuration, go to
Configuration Properties->Environment. Add a new env. variable
CFG_DSPLINK_STR and set the value equal to the arguments that
need to be given to the dsplinkcfg.bat.
```

For DM6446, the value would be:

```
--platform=DAVINCI --nodsp=1 --dspcfg_0=DM6446GEMSHMEM --
dspos_0=DSPBIOS5XX --gpos=WINCE --comps=ponslm
```

5. Select "Build Current BSP and Subprojects" from the Build -> Advanced Build Commands" menu. The GPP dsplink dlls will now be included in your NK.bin image.
6. Follow the BSP instructions to connect to target, load the NK.bin image and boot-up. The GPP side dsp linkk.dll will be loaded automatically at boot-up time.



## 10 Runningthesampleapplications

Seven sample applications are provided with DSPLINK for the DM6446 platform. All the sample applications are described in detail in the user guide. This section describes the way to execute the sample applications.

The specific instructions have been shown below for few of the samples. However, similar instructions can be used for the other applications also.

The steps for execution of the samples are given below for execution with WinCE running on the GPP.

### 10.1 Copyingfilestotargetfilesystem

The generated binaries on the GPP side and DSP side and the data files must be copied to the Flat Release directory.

The make system for the GPP side of DspLink copies the DLLs and EXEs to the target folder for your platform (%\_TARGETPLATROOT%\target\%\_TGTCPU%\%WINCE\_DEBUG%). When the run-time image is built using Platform Builder, these DLLs and EXEs are copied from the above folder to the Flat Release Directory for your OS-Design (%\_FLATRELEASEDIR%). This is true for both command-line builds as well as the IDE Builds.

For the DSP-side binaries, the user needs to manually copy the binaries to the Flat Release Directory.

For Release Build

```
copy
%DSPLINK%\dsp\export\BIN\DspBios\DAVINCI\DM6446GEM_0\RELEASE\*.out
ut %_FLATRELEASEDIR%
```

For Debug Build

```
copy
%DSPLINK%\dsp\export\BIN\DspBios\DAVINCI\DM6446GEM_0\DEBUG\*.out
%_FLATRELEASEDIR%
```

## 10.2 Invokingtheapplication

### 10.2.1 Messagesample

```
# Open the target control window from the Platform Builder Menu - Target
-> Target Control Window
# run message sample

Windows CE> s messagegpp /Release/message.out <number of iterations>
<processor identifier>
```

*The sample can be executed for infinite iterations by specifying the number of iterations as 0.*

*The DSP Processor Id is optional. By default it is zero.*

e.g.

```
Windows CE> s messagegpp /Release/message.out 10000
```

### 10.2.2 Readwritesample

```
# Open the target control window from the Platform Builder Menu - Target
-> Target Control Window
```

```
# run readwrite sample
```

```
Windows CE> s readwritegpp /Release/readwrite.out <DSP address> <buffer
size> <iterations> <processor identifier>
```

*The sample can be executed for infinite iterations by specifying the number of iterations as 0.*

*The DSP Processor Id is optional. By default it is zero.*

e.g.

**For DM6446**

```
Windows CE> s readwritegpp /Release/readwrite.out 2414804992 1024 10000
```

```
Windows CE> s readwritegpp /Release/readwrite.out 293601280 1024 10000
```

```
Windows CE> s readwritegpp /Release/readwrite.out 300957696 1024 10000
```

### 10.2.3 MPCSXFERsample

```
# Open the target control window from the Platform Builder Menu -
Target -> Target Control Window
```

```
# run mpcsxfer sample
```

```
Windows CE> s mpcsxfergpp /Release/mpcsxfer.out <buffer size>
<iterations> <processor identifier>
```

*The sample can be executed for infinite iterations by specifying the number of iterations as 0*

*The DSP Processor Id is optional. By default it is zero.*

e.g.

```
Windows CE> s mpcsxfergpp /Release/mpcsxfer.out 1024 10000
```

### 10.2.4 MP\_LISTsample

```
# Open the target control window from the Platform Builder Menu -
Target -> Target Control Window
```

```
# run mp_list sample
```

```
Windows CE> s mplistgpp /Release/mplist.out <iterations> <number of
elements> <processor identifier>
```

*The DSP Processor Id is optional. By default it is zero.*

e.g.

```
Windows CE> s mplistgpp /Release/mplist.out 1000 20
```

---

### 10.2.5 MESSAGE\_MULTIsample

```
# Open the target control window from the Platform Builder Menu -  
Target -> Target Control Window  
# run message_multi sample
```

```
Windows CE> s messagemultigpp /Release/messagemulti.out <number of  
transfers> <Application instance number 1 -> MAX_APPS> <processor  
identifier>
```

*The DSP Processor Id is optional. By default it is zero.*

e.g.

```
Windows CE> s messagemultigpp /Release/messagemulti.out 10000 1
```

**D. ADDITIONAL INFORMATION**

None.