# DSP/BIOS™ LINK

# OMAP3530 EVM

# LNK 188 USR

# Version 1.65

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

This page has been intentionally left blank.

## TABLE OF CONTENTS

# A. PLATFORM GUIDE

## 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 about usage of DSP/BIOS™ LINK on the OMAP 3530 platform.

This document corresponds to the product release Version 1.65.

## 2 Text Conventions

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

## 3 Terms & Abbreviations

| | |
|---|---|
| CCS | Code Composer Studio |
| IPC | Inter Processor Communication |
| GPP | General Purpose 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 OMAP3530 hardware. |
| 2. | User Guide | DSP/BIOS™ LINK user guide |
| 3. | InstallGuide_<OS>_OMAP3530.doc | Installation guide for relevant OS if present. |
| 4. | Porting Guide | Porting guide for relevant OS if present. |

# 5 Configuring CCS

## 5.1 OMAP3530

To use CCS for debugging the DSP side application, you will need to configure CCS to use both ARM and DSP with the OMAP3530.

- 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 ()`.

# 6 Platform specific information

## 6.1 Working on target platform

### 6.1.1 Configuring Kernel Parameters

DSP/BIOS™ LINK requires a few specific arguments to be passed to the Linux kernel during boot up. 2MB of memory is used by DSP/BIOS™ LINK for communication between GPP and DSP, and for DSP external memory. This must be reserved by specifying 2MB less as available for the Linux kernel for its usage.

### 6.1.2 Using LPM

When cache is enabled, a MMU fault is seen when one application is run followed by a run of another application. This issue is not seen when one application is run followed by a run of the same application. To overcome this issue the LPM module is used to power cycle the EVM between runs.

### 6.1.3 Configuring Dynamic entries

Following are the assumptions for Dynamic entries:-

- The hardware provides a shared memory area, to which both the GPP and the DSP have access.
- The following page sizes are supported:
  Super section: 16M bytes
  Section : 1M bytes
  Large page : 64K bytes
  Small page : 4K bytes

Following are the constraints:-

Any memory overlap cases will treat as error case.
For optimization of TLB entries please make sure

- Requested entries should be continuous.
- Requested entries should be aligned with 16M or 1M or 64K or 4K.

API Usage:-

If DSP access an area whose mapping is not present in the TLB, then DSP's MMU generates an interrupt of PAGE fault type. This interrupt is generated on ARM (HOST). This happens only in case when table walking logic is disabled. Otherwise, if it is enabled, MMU checks with tables present in the memory.

To map some area into the DSP's address space, user can call PROC_control API with respective commands from GPI side. Below shows a usage scenario of mapping an area into DSP's address space.

Add the MMU entry:

```
MemMapInfo * mapInfo ;

mapInfo.dspAddr = DSP_ADDR1 ;
mapInfo.size    = 0x80000   ;

status = PROC_control (ID_PROCESSOR,
                       PROC_CTRL_CMD_MMU_ADD_ENTRY,
                       &mapInfo);
```

Here PROC_CTRL_CMD_MMU_ADD_ENTRY is an enumerated type, directing GPP side logic to add user given entries to the DSP's TLB. Now, User may want to write some information on the area, to be given to DSP. For this user has to map the area into GPP user/kernel address space. For this below example is useful:

```
MemMapInfo * mapInfo ;

mapInfo.dspAddr = DSP_ADDR1 ;
mapInfo.size    = 0x80000   ;

status = PROC_control (ID_PROCESSOR,
                       PROC_CTRL_CMD_MAP_DSPMEM,
                       &mapInfo);
```

Once the user has done with his/her protocol, he/she may want to unmap the area from GPP and DSP address space as well. Below code how to achieve this:

Delete the MMU entry:

```
MemUnmapInfo * mapInfo ;

mapInfo.dspAddr = DSP_ADDR1 ;
mapInfo.size    = 0x80000   ;

status = PROC_control (ID_PROCESSOR,
                       PROC_CTRL_CMD_UNMAP_DSPMEM,
                       &mapInfo);
status = PROC_control (ID_PROCESSOR,
                       PROC_CTRL_CMD_MMU_DEL_ENTRY,
                       &mapInfo);
```

- Given size should be same at add and delete the dynamic entry.
- Can not remove a static entry dynamically.
- Given memory area to create a dynamic entry should be physically contiguous.

## 6.2 DSP Memory Management Unit (MMU) Configuration

The DSP on the OMAP3530 has a MMU for accessing L3 interconnect address space (addresses above 0x11000000). DSP/BIOS™ Link automatically configures and enables the MMU. The translation from virtual to physical is a straight thru (i.e. physical address == virtual address). With the MMU enabled, many invalid memory

accesses (e.g. garbage pointer) can be detected. Also, the GPP memory is protected from DSP corruption.

### 6.2.1 Configuration

The memory regions that are added to the MMU configuration are determined by the `LINKCFG_MemEntry` array supplied by the application. All entries in this array are added into the DSP's MMU configuration. If the DSP accesses memory via the L3 interconnect and that memory is not in the configuration in the MMU, a MMU fault occurs and the GPP is notified (see section 6.2.3 for more details). Refer the `LINKCFG_memTable_00` array in `${DSPLINK}/config/all/CFG_OMAP3530.c` for the default configuration.

DSP/BIOS™ Link configures the MMU by locking entries into the Translation Look-aside Buffers (TLBs). There are 32 TLB entries, but only 31 can be locked. Each TLB entry can be used to map a 4KB, 64KB, 1MB or 16MB contiguous memory region. DSP/BIOS™ Link does not currently generate a separate translation table.

During initialization, DSP/BIOS™ Link scans the `LINKCFG_MemEntry` array and adds TLB entries. If the LINKCFG_MemEntry array has entries that have adjacent memory region, DSP/BIOS™ Link treats the adjacent regions as one larger region. Note this is possible because the translation is 1 to 1 (e.g. virtual address 0x87E00000 -> physical address 0x87F00000). Combining regions reduces the number of TLB entries used.

Since each TLB entry can only manage a 4KB, 64KB, 1MB, or 16MB memory region, the entries in the `LINKCFG_MemEntry` array must be a multiple of 4KB or entries with adjacent memory must add up to a multiple of 4KB. For example, the following is allowed because the two regions defined by the two entries are contiguous and the total size is 1MB (0xFFF80 + 0x80) which is a multiple of 4KB.

```
#define  RSTENTRYID         0u
#define  RESETCTRLADDR      0x87E00000u
#define  RESETCTRLSIZE      0x80u

#define  CODEENTRYID        1u
#define  CODEMEMORYADDR     (RESETCTRLADDR + RESETCTRLSIZE)
#define  CODEMEMORYSIZE     0xFFF80u
    {
        RSTENTRYID,     /* ENTRY       : Entry number */
        "RESETCTRL",    /* NAME        : Name of the memory region */
        RESETCTRLADDR,  /* ADDRPHYS    : Physical address */
        RESETCTRLADDR,  /* ADDRDSPVIRT : DSP virtual address */
        (Uint32) -1u,   /* ADDRGPPVIRT : GPP virtual address */
        RESETCTRLSIZE,  /* SIZE        : Size of the memory region */
        TRUE,           /* SHARED      : Shared access memory? */
        FALSE,          /* SYNCD       : Synchornized? */
    },
    {
        CODEENTRYID,    /* ENTRY       : Entry number */
        "DDR2",         /* NAME        : Name of the memory region */
        CODEMEMORYADDR, /* ADDRPHYS    : Physical address */
        CODEMEMORYADDR, /* ADDRDSPVIRT : DSP virtual address */
        (Uint32) -1u,   /* ADDRGPPVIRT : GPP virtual address */
        CODEMEMORYSIZE, /* SIZE        : Size of the memory region */
        TRUE,           /* SHARED      : Shared access memory? */
        FALSE,          /* SYNCD       : Synchornized? */
    },
```

While the following is not allowed because the RESETCTRL entry's size is not a multiple of 4KB and its address is not adjacent to a region that would make its size a multiple of 4KB. This would result in a runtime configuration error (see section 6.2.2 for more details).

```
#define  RSTENTRYID          0u
#define  RESETCTRLADDR       0x87E00000u
#define  RESETCTRLSIZE       0x80u

#define  CODEENTRYID         1u
#define  CODEMEMORYADDR      (RESETCTRLADDR + RESETCTRLSIZE + 0x20u)
#define  CODEMEMORYSIZE      0xFFF80u
    {
        RSTENTRYID,      /* ENTRY       : Entry number */
        "RESETCTRL",     /* NAME        : Name of the memory region */
        RESETCTRLADDR,   /* ADDRPHYS    : Physical address */
        RESETCTRLADDR,   /* ADDRDSPVIRT : DSP virtual address */
        (Uint32) -1u,    /* ADDRGPPVIRT : GPP virtual address */
        RESETCTRLSIZE,   /* SIZE        : Size of the memory region */
        TRUE,            /* SHARED      : Shared access memory? */
        FALSE,           /* SYNCD       : Synchornized? */
    },
    {
        CODEENTRYID,     /* ENTRY       : Entry number */
        "DDR2",          /* NAME        : Name of the memory region */
        CODEMEMORYADDR,  /* ADDRPHYS    : Physical address */
        CODEMEMORYADDR,  /* ADDRDSPVIRT : DSP virtual address */
        (Uint32) -1u,    /* ADDRGPPVIRT : GPP virtual address */
        CODEMEMORYSIZE,  /* SIZE        : Size of the memory region */
        TRUE,            /* SHARED      : Shared access memory? */
        FALSE,           /* SYNCD       : Synchornized? */
    },
```

There are configurations that can use up all 31 of the TLB entries. For example, specifying a region of size 0xFF000 (1MB - 4KB), would result in 30 TLB entries (15 64KB entries and 15 4KB entries). While a 1MB region would result in one 1MB TLB entry. If more than 31 TLB entries are needed, a runtime configuration error is reported (see section 6.2.2 for more details). The memory region configuration must be changed to allow at most 31 TLB entries.

### 6.2.2 Configuration Errors

Two types of configuration errors may arise in regards to the MMU.

1. LINKCFG_MemEntry configuration requires too many TLB entries
2. A LINKCFG_MemEntry entry's size is not a multiple of 4KB (or if the entry's memory is adjacent to another entry's memory, the combined size is not a multiple of 4KB).

Both would result in a runtime configuration error (DSP_ECONFIG). Additional information is displayed to stdio also (e.g. "Configuration error: Exceeded maximum number [31] of Translation Look-aside Buffers.").

### 6.2.3 MMU Faults

With the DSP's MMU enabled, if the DSP is accessing an L3 interconnect address space (addresses above 0x11000000) and it is not in the MMU TLB, a MMU fault occurs and an interrupt is sent to the ARM (IVA2_MMU_IRQ). DSP/BIOS™ Link

processes the interrupt and prints out a message to stdio. This message includes the invalid address. For example:

```
"DSP MMU Error Fault! MMU_IRQSTATUS = [1]. Virtual DSP addr reference that generated the interrupt = [0x20004000].\n"
```

Once the fault occurs, DSP/BIOS™ Link disables the generation of more MMU fault interrupts. The management of the interrupt is in `${DSPLINK}/gpp/src/ldrv/HAL/OMAP/3530/hal_mmu.c` and can be changed if a different action is needed.

### 6.2.4 MMU Faults when cache is enabled

When cache is enabled, a MMU fault is seen when one application is run followed by a run of another application. This issue is not seen when one application is run followed by a run of the same application. To overcome this issue a power cycle of the EVM between runs is needed.

## 6.3 DSP/BIOS™ CLK Manager Configuration

DSP/BIOS™ on the OMAP3530 uses Timer 5 by default. DSP/BIOS™ Link on the GPP must help in the management of this timer. Currently this management on the GPP is hard-coded to GPTimer5 (done in `${DSPLINK}/gpp/src/ldrv/HAL/OMAP/3530/hal_prcm.c`), thus the DSP/BIOS™ CLK Manager Timer Selection must use Timer 5. Below is an example of the required DSP/BIOS™ CLK Manager configuration (which again is the default).

```
bios.CLK.TIMERSELECT = "Timer 5";
```