DSP/BIOS™ LINK

DRA44x + DM6437

LNK 191 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. 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.

2 Text Conventions

O	This bullet indicates important information. 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. 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.	User Guide	DSP/BIOS™ LINK user guide
2.	InstallGuide_<OS>_Jacinto_DM6437_VLYNQ.pdf	Installation guide for relevant OS if present.
3.	Porting Guide	Porting guide for relevant OS if present.

5 Configuring CCS

5.1 DRA44x EVM

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

- 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 Flashing DM6437

VLYNQ module on DM6437 needs to be powered on so that ARM on DRA44x can access the DM6437 via VLYNQ. This can be done in two ways either using GEL file on DM6437 or flashing the NAND of DM6437 with initialization code. Here we will be describing the later.

O DM6437 has problems with NAND bootmode, so we will use I2C bootmode and secondary bootloader mechanism.

We will use the AIS script to flash the I2C and NAND of DM6437. Steps are as follows:

1. Compile secondary boot (customize if required)
2. Use genAIS utility to create AIS bootable image for secondary boot code from I2C/SPI
3. Compile application code.
4. Use genAIS utility to create binary AIS bootable image of application code from NAND.
5. Use NAND Writer utility to write application AIS image to NAND flash.
6. Use I2C Writer utility to write secondary boot AIS image to EEPROM.
7. Set BOOTMODE [3:0] = 1001b to select boot from I2C.
8. Issue POR reset to DM6437 device.

Following steps are described in the NANDSecondaryBoot.doc in the zip file from the link below:
http://focus.ti.com/dsp/docs/dspsupporttechdocsc.tsp?sectionId=3&tabId=409&familyId=1301&abstractName=spraag0d

The initialization code is below:

```
#include "stdio.h"
volatile unsigned int * bootAddr  = (unsigned int *) 0x1C40008 ;
volatile unsigned int * bootCmplt = (unsigned int *) 0x1C4000C ;
#define PSC_BASE          0x01C41000
#define PSC_EPCPR          *( unsigned int* )( 0x01C41070 )
#define PSC_PTCMD          *( unsigned int* )( 0x01C41120 )
#define PSC_PTSTAT         *( unsigned int* )( 0x01C41128 )
#define PSC_MDSTAT_BASE     ( 0x01C41800 )
#define PSC_MDCTL_BASE      ( 0x01C41A00 )
```

```
#define BOOTCFG          *( unsigned int* )( 0x01C40014 )
#define PINMUX0          *( unsigned int* )( 0x01C40000 )
#define PINMUX1          *( unsigned int* )( 0x01C40004 )
#define VDD3P3V_PWDN       *( unsigned int* )( 0x01C40048 )
#define PCISLVCNTRL       *( unsigned int* )( 0x01C1A180 )
#define PCICFGDONE        *( unsigned int* )( 0x01C1A3AC )

typedef void (*PC) () ;
/* ------------------------------------------------------------------- *
 *                                                                     *
 *  main( )                                                            *
 *                                                                     *
 * ------------------------------------------------------------------- */
void main( void )
{
   int i ;
   PC pcReg ;

   if ( ( BOOTCFG & 0x00020000 ) == 0 )
   {
      PINMUX0 = 0x00148001;
          /*| ( 0 << 30 )   // CI10SEL   - No CI[1:0]
            | ( 0 << 28 )   // CI32SEL   - No CI[3:2]
            | ( 0 << 26 )   // CI54SEL   - No CI[5:4]
            | ( 0 << 25 )   // CI76SEL   - No CI[7:6]
            | ( 0 << 24 )   // CFLDSEL   - No C_FIELD
            | ( 0 << 23 )   // CWENSEL   - No C_WEN
            | ( 0 << 22 )   // HDVSEL    - No CCDC HD and VD
            | ( 1 << 20 )   // CCDCSEL   - CCDC PCLK, YI[7:0] enabled
            | ( 4 << 16 )   // AEAW      - EMIFA full address mode
            | ( 1 << 15 )   // VPBECLK   - VPBECLK enabled
            | ( 0 << 12 )   // RGBSEL    - No digital outputs
            | ( 0 << 10 )   // CS3SEL    - LCD_OE/EM_CS3 disabled
            | ( 0 <<  8 )   // CS4SEL    - CS4/VSYNC disabled
            | ( 0 <<  6 )   // CS5SEL    - CS5/HSYNC disabled
            | ( 0 <<  4 )   // VENCSEL   - Video encoder outputs disabled
            | ( 1 <<  0 );*/// AEM       - 8-bit EMIFA, 8-bit CCDC in
   }

   /*
```

```
 *  PINMUX settings for PCI operation
 */
if ( ( BOOTCFG & 0x00020000 ) != 0 )
{
   PINMUX0 = 0x00148000;
        /*| ( 0 << 30 )   // CI10SEL   - No CI[1:0]
          | ( 0 << 28 )   // CI32SEL   - No CI[3:2]
          | ( 0 << 26 )   // CI54SEL   - No CI[5:4]
          | ( 0 << 25 )   // CI76SEL   - No CI[7:6]
          | ( 0 << 24 )   // CFLDSEL   - No C_FIELD
          | ( 0 << 23 )   // CWENSEL   - No C_WEN
          | ( 0 << 22 )   // HDVSEL    - No CCDC HD and VD
          | ( 1 << 20 )   // CCDCSEL   - CCDC PCLK, YI[7:0] enabled
          | ( 4 << 16 )   // AEAW      - EMIFA full address mode
          | ( 1 << 15 )   // VPBECLK   - VPBECLK enabled
         | (  0 << 12 )   // RGBSEL    - No digital outputs
          | ( 0 << 10 )   // CS3SEL    - LCD_OE/EM_CS3 disabled
          | ( 0 <<  8 )   // CS4SEL    - CS4/VSYNC disabled
          | ( 0 <<  6 )   // CS5SEL    - CS5/HSYNC disabled
          | ( 0 <<  4 )   // VENCSEL   - Video encoder outputs disabled
          | ( 0 <<  0 );*/// AEM       - N/A
}

PINMUX1 = 0x01618530;
        /*| ( 1 << 24 )   // SPBK1     - McBSP1 enabled
          | ( 1 << 22 )   // SPBK0     - McBSP0 enabled
          | ( 2 << 20 )   // TIM1BK    - UART1 enabled, Timer1 disabled
          | ( 1 << 16 )   // TIM0BK    - Timer0 enabled
          | ( 2 << 14 )   // CKOBK     - CLKOUT disabled, PWM2 enabled
          | ( 0 << 12 )   // PWM1BK    - PWM1 disabled, GIO84 enabled
          | ( 1 << 10 )   // UR0FCBK   - UART0 HW flow control enabled
          | ( 1 <<  8 )   // UR0DBK    - UART0 data enabled
          | ( 3 <<  4 )   // HOSTBK    - VLYNQ + MII + MDIO Mode
          | ( 0 <<  0 );*/// PCIEN     - PCI disabled

VDD3P3V_PWDN = 0x00000000;  // Everything on

for ( i = 0 ; i <= 9 ; i++ )
   *( unsigned int* )( PSC_MDCTL_BASE + 4 * i ) |= 0x0003;
for ( i = 11 ; i <= 28 ; i++ )
```

```
        *( unsigned int* )( PSC_MDCTL_BASE + 4 * i ) |= 0x0003;
    i = 39;
        *( unsigned int* )( PSC_MDCTL_BASE + 4 * i ) |= 0x0003;

    PSC_PTCMD = 0x0001;
    while( ( PSC_PTSTAT & 0x0001 ) );

    if ( ( BOOTCFG & 0x00020000 ) != 0 )
    {
        PCICFGDONE |= 1;
        PCISLVCNTRL |= 1;
    }

    while (1) {
        if (*bootCmplt & 0x1) {
            *bootCmplt = 0 ;
            break ;
        }
    }

    pcReg = (PC) (*bootAddr) ;
    (*pcReg) () ;
}
```

Create a project and compile the above file and follow the steps as specified in the NAND flashing sections.

6.2 EDMA resource requirements for DM6437 over VLNQ

By default, DSPLINK uses the following resources of DRA44x EDMA to achieve DMA between host and DM6437 connected over VLNQ interface.

EDMA Channel 0.

Shadow region 0

EDMA Param Entry 0

Interrupt Set/Pending bits: Bit0

Event number: 0 (I.e. Bit0 of these registers: ISER, ICR, IPR, EESR, ESR)

TC value 0

- Application writers can change the EDMA channel ID by modifying the arg5 of LINKCFG_dspObject in the platform configuration file.

All other resources used by this channel are defaults/user programmed values (like queue etc)