Application Report
SPRABL2E–January 2012–Revised October 2016

KeyStone I DDR3 Initialization

Thomas Johnson

ABSTRACT

The initialization of the DDR3 DRAM controller on KeyStone I DSPs is straightforward as long as the proper steps are followed. However, if some steps are omitted or if some sequence-sensitive steps are implemented in the wrong order, DDR3 operation will be unpredictable.

Detailed explanations of the registers and their functionality are contained in the *KeyStone Architecture DDR3 Memory Controller User's Guide* (SPRUGV8). Board layout guidance is provided in the *DDR3 Design Requirements for KeyStone Devices* (SPRABI1).

All DDR3 initialization routines must contain the basic register writes to configure the memory controller within the DSP as well as register writes that configure the mode registers within the attached DRAM devices. The data sheet for the DRAM implemented must be referenced to optimize these values.

DDR3 implementations use a fly-by routing topology, PCB track lengths for the fly-by signals (Address, Command, Control, and Clock) and data group signals (DQ, DQS, and DQM) must be available to properly initialize the leveling registers.

The spreadsheet mentioned in this application report can be downloaded from the following URL: http://www.ti.com/lit/zip/sprabl2.

Contents

List of Tables

1 New DDR3 Feature: Leveling

This section provides an introduction to the new DDR3 physical interface concept called *leveling*. A basic understanding of this technology will help programmers understand the required configuration steps.

The physical DDR3 interface on the KeyStone I DSPs is often called the DDR3 PHY. It includes the I/O buffers and all of the logic required to support the DDR3 interface technology. The DDR3 interface circuitry also includes registers and control logic to support the physical DDR3 interface as well as control for the DRAM devices. The terms PHY and Controller are used interchangeably in this document when referring to this circuitry.

The KeyStone I DDR3 controller supports three modes of leveling:

- Write leveling
- Read eye training
- Read gate training

These three specific leveling modes are also generally referred to as write and read leveling. This is discussed more in the next subsection. The following subsection explains that the leveling circuitry in the DDR3 controller must be properly configured for robust operation. The final subsection in this introduction describes the purpose of using an inverted DDR3 clock for certain memory implementations.

1.1 Write and Read Leveling

Write and read leveling are new controller features in the JEDEC DDR3 implementation. DDR3 operating frequencies are achieved by allowing the address, control, command, and clock nets to be routed in a fly-by arrangement. This allows for optimum signal integrity. However, this also results in a different delay from the controller to each DRAM. The write leveling circuitry adaptively estimates this delay and offsets the data group signals so that they arrive at the DRAM coincident with the fly-by signals. This adaptation occurs independently for each byte lane resulting in a unique and optimized offset for each byte lane. The different delays for the clock to each DRAM also causes different launch times for the data group signals during reads. This causes the need for read-eye and read-gate training (leveling).

Write leveling must be executed prior to DRAM writes so that the data and control signals arrive at the DRAM with valid timing relationships. Similarly, read-eye and read-gate training must be completed prior to DRAM reads so that the data will be received and sampled by the KeyStone I DSPs at the appropriate time.

1.2 Automatic Leveling Initialization

The leveling circuitry within the KeyStone I DDR3 controller converges all nine byte lanes simultaneously. This is possible because an initial value for each leveling mode is written into the DDR3 PHY byte lanes prior to convergence. There are two primary initialization values for each byte lane. The first is the WRLVL_INIT_RATIO (write leveling initialization ratio) value, which is based on the difference between the routed clock length and the routed data strobe length.

The second is the GATELVL_INIT_RATIO (gate leveling initialization ratio) value, which is approximately the round trip delay. This is equal to the routed clock length plus the routed data strobe length plus the read sampling delay. The routed clock length represents the routed length for all of the fly-by signals and the routed data strobe length represents the routed length for the data group signals. The fly-by signals are the address, command, control, and clock signals, and the data group signals are the data, data strobe, and data mask signals.

The DDR3 PHY Calc spreadsheet is provided to help users calculate the initial values and to translate them into the proper units. The inputs are the routed clock and data strobe lengths. The result values are the initial values in units of DLL taps, of which there are 256 per clock period. In addition, because the initial leveling algorithm adapts only in the positive direction, the initial values are offset 128 DLL steps in the negative direction. The spreadsheet can be downloaded from the following URL: http://www.ti.com/lit/zip/sprabl2.

1.3 Invert Clock Out

Some DDR3 board layouts have very short fly-by routes to the first DRAM and some have long routes to the first DRAM that form a loop. UDIMM modules always have a long loop. Implementations of individual DRAMs often have a fly-by routing delay to the first DRAM that is about equal to the data group delay to that DRAM. Because of the timing uncertainties in the DDR3 PHY circuitry, whenever the board routing delay for the clock is not longer than the data routing by at least one-quarter of a clock period, leveling may fail. To solve this problem, the DDR3 clock output can be inverted to add an apparent extension of one-half the clock period to the clock net and the other associated fly-by routes. This feature is enabled by setting the INVERT_CLKOUT bit in one of the PHY control registers.

2 DDR3 Controller Configuration

The initialization routines normally have five basic pieces. This is true whether the routine is implemented within a Code Composer Studio (CCS) GEL file or embedded software such as C-code. The five sections are:

- **Header section** - contains # *define* macros that represent cryptic addresses with useful names
- **KICK unlock and DDR3 PLL configuration** - this may reside elsewhere but it must be completed prior to initializing the DDR3 controller and DRAM
- **Leveling register configuration** - register writes to configure the write and read leveling circuitry
- **Basic controller and DRAM configuration** - register writes that configure the critical timing parameters and mode register parameters similar to those used for DDR2
- **Leveling execution** - register writes to invoke the write and read leveling processes

The DDR3 controller initialization process as defined in this document has been proven to be robust. It is meant to be executed once from start to finish. Portions of this sequence cannot be implemented in a loop. If a system design requires multiple DDR3 Controller initialization sequences, each initialization should follow a device reset.

2.1 Header Section

The header section contains '# define' macros that represent cryptic addresses with useful names. The register addresses can be obtained from the *KeyStone Architecture DDR3 Memory Controller User's Guide* (SPRUGV8) or the Data Manual for the KeyStone I DSPs. Register mapping varies from KeyStone I DSPs to DSPs. But they do fall into two groups: the Combined Fixed Ratio Register DSPs and the Expanded Fixed Ratio Register DSPs. Table 1 shows these two groups.

Table 1. KeyStone I DSPs Grouping

Group	KeyStone I Devices
Combined Fixed Ratio Register DSPs	TCI6602
	TCI6604
	TCI6608
	TCI6616
	TCI6618
	C6670
	C6671
	C6672
	C6674
	C6678
Expanded Fixed Ratio Register DSPs	TCI6612
	TCI6613
	TCI6614
	C6654
	C6655
	C6657

Example 1 shows the addresses as defined for the Combined Fixed Ratio Register DSPs while Example 2 shows the addresses as defined for the Expanded Fixed Ratio Register DSPs. Note that some of the registers are in the address region starting at 0x2100_0000 defined for the DDR3 memory controller and that many of the registers are located in the chip-level registers address region starting at 0x0262_0000. Chip-level registers are also referred to as boot configuration registers in other KeyStone I documentation.

Example 1 and Example 2 contain all of the register locations.

Example 1. Combined Fixed Ratio Register Sample Header

```
#define DDR3_BASE_ADDR     (0x21000000)
#define DDR_SDCFG          (*(unsigned int*)(DDR3_BASE_ADDR + 0x00000008))
#define DDR_SDRFC          (*(unsigned int*)(DDR3_BASE_ADDR + 0x00000010))
#define DDR_SDTIM1         (*(unsigned int*)(DDR3_BASE_ADDR + 0x00000018))
#define DDR_SDTIM2         (*(unsigned int*)(DDR3_BASE_ADDR + 0x00000020))
#define DDR_SDTIM3         (*(unsigned int*)(DDR3_BASE_ADDR + 0x00000028))
#define DDR_PMCTL          (*(unsigned int*)(DDR3_BASE_ADDR + 0x00000038))
#define RDWR_LVL_RMP_WIN   (*(unsigned int*)(DDR3_BASE_ADDR + 0x000000D4))
#define RDWR_LVL_RMP_CTRL  (*(unsigned int*)(DDR3_BASE_ADDR + 0x000000D8))
#define RDWR_LVL_CTRL      (*(unsigned int*)(DDR3_BASE_ADDR + 0x000000DC))
#define DDR_ZQCFG          (*(unsigned int*)(DDR3_BASE_ADDR + 0x000000C8))
#define DDR_PHYCTRL        (*(unsigned int*)(DDR3_BASE_ADDR + 0x000000E4))

#define DDR3PLLCTL0        (*(unsigned int*)(0x02620330))
#define DDR3PLLCTL1        (*(unsigned int*)(0x02620334))

#define DATA0_WRLVL_INIT_RATIO  (*(unsigned int*)(0x0262040C))
#define DATA1_WRLVL_INIT_RATIO  (*(unsigned int*)(0x02620410))
#define DATA2_WRLVL_INIT_RATIO  (*(unsigned int*)(0x02620414))
#define DATA3_WRLVL_INIT_RATIO  (*(unsigned int*)(0x02620418))
#define DATA4_WRLVL_INIT_RATIO  (*(unsigned int*)(0x0262041C))
#define DATA5_WRLVL_INIT_RATIO  (*(unsigned int*)(0x02620420))
#define DATA6_WRLVL_INIT_RATIO  (*(unsigned int*)(0x02620424))
#define DATA7_WRLVL_INIT_RATIO  (*(unsigned int*)(0x02620428))
#define DATA8_WRLVL_INIT_RATIO  (*(unsigned int*)(0x0262042C))

#define DATA0_GTLVL_INIT_RATIO  (*(unsigned int*)(0x0262043C))
#define DATA1_GTLVL_INIT_RATIO  (*(unsigned int*)(0x02620440))
#define DATA2_GTLVL_INIT_RATIO  (*(unsigned int*)(0x02620444))
#define DATA3_GTLVL_INIT_RATIO  (*(unsigned int*)(0x02620448))
#define DATA4_GTLVL_INIT_RATIO  (*(unsigned int*)(0x0262044C))
#define DATA5_GTLVL_INIT_RATIO  (*(unsigned int*)(0x02620450))
#define DATA6_GTLVL_INIT_RATIO  (*(unsigned int*)(0x02620454))
#define DATA7_GTLVL_INIT_RATIO  (*(unsigned int*)(0x02620458))
#define DATA8_GTLVL_INIT_RATIO  (*(unsigned int*)(0x0262045C))

#define DDR3_CONFIG_REG_0       (*(unsigned int*)(0x02620404))
#define DDR3_CONFIG_REG_1       (*(unsigned int*)(0x02620408))
#define DDR3_CONFIG_REG_12      (*(unsigned int*)(0x02620434))
#define DDR3_CONFIG_REG_23      (*(unsigned int*)(0x02620460))
#define DDR3_CONFIG_REG_24      (*(unsigned int*)(0x02620464))

#define KICK0                   (*(unsigned int*)(0x2620038))
#define KICK1                   (*(unsigned int*)(0x262003C))

#define KICK0_UNLOCK            0x83E70B13
#define KICK1_UNLOCK            0x95A4F1E0

#define KICK0_LOCK              0
#define KICK1_LOCK              0
```

Example 2. Expanded Fixed Ratio Register Sample Header

```
#define DDR3_BASE_ADDR     (0x21000000)
#define DDR_SDCFG          (*(unsigned int*)(DDR3_BASE_ADDR + 0x00000008))
#define DDR_SDRFC          (*(unsigned int*)(DDR3_BASE_ADDR + 0x00000010))
#define DDR_SDTIM1         (*(unsigned int*)(DDR3_BASE_ADDR + 0x00000018))
#define DDR_SDTIM2         (*(unsigned int*)(DDR3_BASE_ADDR + 0x00000020))
#define DDR_SDTIM3         (*(unsigned int*)(DDR3_BASE_ADDR + 0x00000028))
#define DDR_PMCTL          (*(unsigned int*)(DDR3_BASE_ADDR + 0x00000038))
#define RDWR_LVL_RMP_WIN   (*(unsigned int*)(DDR3_BASE_ADDR + 0x000000D4))
#define RDWR_LVL_RMP_CTRL  (*(unsigned int*)(DDR3_BASE_ADDR + 0x000000D8))
#define RDWR_LVL_CTRL      (*(unsigned int*)(DDR3_BASE_ADDR + 0x000000DC))
#define DDR_ZQCFG          (*(unsigned int*)(DDR3_BASE_ADDR + 0x000000C8))
#define DDR_PHYCTRL        (*(unsigned int*)(DDR3_BASE_ADDR + 0x000000E4))

#define DDR3PLLCTL0        (*(unsigned int*)(0x02620330))
#define DDR3PLLCTL1        (*(unsigned int*)(0x02620334))

#define DATA0_WRLVL_INIT_RATIO     (*(unsigned int*)(0x0262040C))
#define DATA1_WRLVL_INIT_RATIO     (*(unsigned int*)(0x02620410))
#define DATA2_WRLVL_INIT_RATIO     (*(unsigned int*)(0x02620414))
#define DATA3_WRLVL_INIT_RATIO     (*(unsigned int*)(0x02620418))
#define DATA4_WRLVL_INIT_RATIO     (*(unsigned int*)(0x0262041C))
#define DATA5_WRLVL_INIT_RATIO     (*(unsigned int*)(0x02620420))
#define DATA6_WRLVL_INIT_RATIO     (*(unsigned int*)(0x02620424))
#define DATA7_WRLVL_INIT_RATIO     (*(unsigned int*)(0x02620428))
#define DATA8_WRLVL_INIT_RATIO     (*(unsigned int*)(0x0262042C))

#define DATA0_GTLVL_INIT_RATIO     (*(unsigned int*)(0x0262043C))
#define DATA1_GTLVL_INIT_RATIO     (*(unsigned int*)(0x02620440))
#define DATA2_GTLVL_INIT_RATIO     (*(unsigned int*)(0x02620444))
#define DATA3_GTLVL_INIT_RATIO     (*(unsigned int*)(0x02620448))
#define DATA4_GTLVL_INIT_RATIO     (*(unsigned int*)(0x0262044C))
#define DATA5_GTLVL_INIT_RATIO     (*(unsigned int*)(0x02620450))
#define DATA6_GTLVL_INIT_RATIO     (*(unsigned int*)(0x02620454))
#define DATA7_GTLVL_INIT_RATIO     (*(unsigned int*)(0x02620458))
#define DATA8_GTLVL_INIT_RATIO     (*(unsigned int*)(0x0262045C))

#define DDR3_CONFIG_REG_0          (*(unsigned int*)(0x02620404))
#define DDR3_CONFIG_REG_12         (*(unsigned int*)(0x02620434))
#define DDR3_CONFIG_REG_23         (*(unsigned int*)(0x02620460))
#define DDR3_CONFIG_REG_24         (*(unsigned int*)(0x02620464))
#define DDR3_CONFIG_REG_52         (*(unsigned int*)(0x026204D4))
#define DDR3_CONFIG_REG_53         (*(unsigned int*)(0x026204D8))
#define DDR3_CONFIG_REG_54         (*(unsigned int*)(0x026204DC))
#define DDR3_CONFIG_REG_55         (*(unsigned int*)(0x026204E0))
#define DDR3_CONFIG_REG_56         (*(unsigned int*)(0x026204E4))
#define DDR3_CONFIG_REG_57         (*(unsigned int*)(0x026204E8))
#define DDR3_CONFIG_REG_58         (*(unsigned int*)(0x026204EC))
#define DDR3_CONFIG_REG_59         (*(unsigned int*)(0x026204F0))
#define DDR3_CONFIG_REG_60         (*(unsigned int*)(0x026204F4))

#define KICK0                      (*(unsigned int*)(0x2620038))
#define KICK1                      (*(unsigned int*)(0x262003C))

#define KICK0_UNLOCK               0x83E70B13
#define KICK1_UNLOCK               0x95A4F1E0

#define KICK0_LOCK                 0
#define KICK1_LOCK                 0
```

2.2 KICK Unlock and DDR3 PLL Configuration

The KICK unlock and DDR3 PLL configuration may reside elsewhere in the initialization code but they must be completed prior to initialize the DDR3 controller and DRAM. Prior to writing to the DDR3 Controller configuration registers, the KICK registers need to be unlocked. These may then be locked again after configuration is complete. For more information, see the *KeyStone Architecture DSP Bootloader User's Guide* (SPRUGY5) or the device-specific data manual. Software examples provided within the multicore software development kit (MCSDK) have a separate subroutine for DDR3 PLL configuration. This is the recommended implementation solution.

Early versions of the KeyStone I DSPs contain PLL initialization errata that must be resolved through software during the PLL configuration. Information about PLL configuration can be found in the *KeyStone Architecture Phase-Locked Loop (PLL) User's Guide* (SPRUGV2) and the respective KeyStone I Data Manual. Information about the PLL errata can be found in the silicon errata document for the specific KeyStone I DSP.

The DDR3 PLL within the KeyStone I DSP receives the reference clock provided at the DDR3CLK input and multiplies it up to the rate desired for the DDR3 interface. Note that the clock rate to the DDR3 memory is half of the data rate (DDR3-1333 operates with a 666.67-MHz clock).

Example 3 unlocks the KICK registers and programs the DDR3 PLL to generate a 666.67-MHz clock (for DDR3-1333 operation) from an input clock of 66.667 MHz. For a detailed explanation of the sequence of steps and the necessary delays, see the *KeyStone Architecture Phase-Locked Loop (PLL) User 's Guide* (SPRUGV2). Note that the DDR3 clock rate affects timing parameters throughout this application report

Example 3. Programming DDR3 PLL

```
#define PLL2_PLLD 0   // Must be less than 64
#define PLL2_PLLM 19  // Must be less than 4096

DDR3PLLCTL1 |= 0x00000040;  // Set ENSAT bit = 1
DDR3PLLCTL0 |= 0x00800000;  // Set BYPASS bit = 1

// Clear and program PLLD field
DDR3PLLCTL0 &= ~(0x0000003F);
DDR3PLLCTL0 |= (PLL2_PLLD & 0x0000003F);

// Clear and program PLLM field
DDR3PLLCTL0 &= ~(0x0007FFC0);
DDR3PLLCTL0 |= ((PLL2_PLLM << 6) & 0x0007FFC0 );

// Clear and program BWADJ field
PLL2_BWADJ = ((PLL2_PLLM + 1) >> 1) - 1;
DDR3PLLCTL0 &= ~(0xFF000000);
DDR3PLLCTL1 &= ~(0x0000000F);
DDR3PLLCTL0 |= ((PLL2_BWADJ << 24) & 0xFF000000);
DDR3PLLCTL1 |= ((PLL2_BWADJ >> 8) & 0x0000000F);

DDR3PLLCTL1 |= 0x00002000;    // Set RESET bit = 1
for(i=0;i<10000;i++); // Wait at least 5us for reset complete
DDR3PLLCTL1 &= ~(0x00002000); // Clear RESET bit
for(i=0;i<70000;i++); // Wait at least 50us for PLL lock
DDR3PLLCTL0 &= ~(0x00800000); // Clear BYPASS bit = 0
```

2.3 Leveling Register Configuration

The chip-level registers that provide initialization values to the leveling circuitry need to be programmed next. These values will be used after the basic controller and DRAM configuration but they must be written before that part of the initialization sequence. Values need to be written to the registers DDR3_CONFIG_REG_0 and DDR3_CONFIG_REG_12. Depending on the leveling process chosen, they may need to be written to the fixed leveling registers. Combined Fixed Ratio Register DSPs may require a write to DDR3_CONFIG_REG_23, whereas, the Expanded Fixed Ratio Register DSPs may require writes to DDR3_CONFIG_REG_52 through DDR3_CONFIG_REG_60. This topic is explained in Section 3. In addition, initial values must be written to the WRLVL_INIT_RATIO and GATELVL_INIT_RATIO registers for all of the data lanes in use when automatic leveling is used.

The INVERT_CLKOUT bit discussed previously is one of these configuration values. This feature is enabled by setting the INVERT_CLKOUT bit in the DDR3_CONFIG_REG_12 register. Note that when we set INVERT_CLKOUT to 1, the CTRL_SLAVE_RATIO field in DDR3_CONFIG_REG_0 should be programmed to 0x100. If the INVERT_CLKOUT bit remains at 0, the default value of the CTRL_SLAVE_RATIO field must be programmed to 0x80. Example 4 sets the INVERT_CLKOUT bit and programs the CTRL_SLAVE_RATIO field properly.

Example 4. INVERT_CLKOUT Programming

```
DDR3_CONFIG_REG_0  &= ~(0x007FE000);  // clear ctrl_slave_ratio field
DDR3_CONFIG_REG_0  |= 0x00200000;     // set ctrl_slave_ratio to 0x100
DDR3_CONFIG_REG_12 |= 0x08000000;     // Set invert_clkout = 1
```

The DLL_LOCK_DIFF field is also programmed at this time to 0xF because it is the only other programmable value in the DDR3_CONFIG_REG_0 register. The DLL_LOCK_DIFF field must always be written with this value.

Example 5. DLL_LOCK_DIFF Programming

```
DDR3_CONFIG_REG_0 |= 0xF;            // set dll_lock_diff to 15
```

Additional writes may be added at this point depending on the leveling mode chosen. The values for those registers are explained in Section 3. Specifically, Combined Fixed Ratio Register DSPs may have an additional write to DDR3_CONFIG_REG_23 at this point in the sequence. Alternately, Expanded Fixed Ratio Register DSPs in the TCI6612/3/4 family may have additional writes to DDR3_CONFIG_REG_52 through DDR3_CONFIG_REG_60 at this point in the sequence. Expanded Fixed Ratio Register DSPs in the C665x family may have additional writes to only to DDR3_CONFIG_REG_52 through DDR3_CONFIG_REG_55 and DDR3_CONFIG_REG_60 at this point in the sequence. For more information, see Section 3.

Expanded Fixed Ratio Register DSPs require the following assignment at this point in the initialization sequence. This sets the DQ to DQS offset properly during write cycles. This assignment is not required for Combined Fixed Ratio Register DSPs.

Example 6. DATA_REG_PHY_DQ_OFFSET Programming

```
DDR3_CONFIG_REG_1 = 0x01000000;
```

The next group of chip-level registers that needs to be written are the initial values for the leveling process. The DDR3 PHY Calc spreadsheet is provided to help calculate the initial values and to translate them into the proper units. The inputs are the routed clock and data strobe lengths. The result values are the initial values in units of DLL taps. Follow the instructions in the DDR3 PHY Calc spreadsheet to generate the set of initial values for the board implementation. The initial values below are those calculated from the C6678 EVM design. Note that devices in the C665x family will not have entries for DATA4 through DATA7.

Example 7. Leveling Initialization Values

```
DATA0_WRLVL_INIT_RATIO = 0x99;
DATA1_WRLVL_INIT_RATIO = 0x99;
DATA2_WRLVL_INIT_RATIO = 0x99;
DATA3_WRLVL_INIT_RATIO = 0x8D;
DATA4_WRLVL_INIT_RATIO = 0x75;
DATA5_WRLVL_INIT_RATIO = 0x77;
DATA6_WRLVL_INIT_RATIO = 0x62;
DATA7_WRLVL_INIT_RATIO = 0x5E;
DATA8_WRLVL_INIT_RATIO = 0x80;

DATA0_GTLVL_INIT_RATIO = 0xDF;
DATA1_GTLVL_INIT_RATIO = 0xDF;
DATA2_GTLVL_INIT_RATIO = 0xC2;
DATA3_GTLVL_INIT_RATIO = 0xCE;
DATA4_GTLVL_INIT_RATIO = 0xAE;
DATA5_GTLVL_INIT_RATIO = 0xAC;
DATA6_GTLVL_INIT_RATIO = 0xA4;
DATA7_GTLVL_INIT_RATIO = 0xA7;
DATA8_GTLVL_INIT_RATIO  = 0xBE;
```

Finally, the PHY_RESET is pulsed (0 → 1 → 0) to latch these leveling configuration values into the PHY logic.

Example 8. Pulsing the PHY_RESET

```
DDR_PHYCTRL &= ~(0x00008000);
DDR_PHYCTRL |=  (0x00008000);
DDR_PHYCTRL  &= ~(0x00008000);
```

2.4 Basic Controller and DRAM Configuration

The basic controller and DRAM configuration writes occur next. These register writes configure the controller critical timing parameters and the DRAM mode register parameters. The *KeyStone Architecture DDR3 Memory Controller User's Guide* (SPRUGV8) must be used to create these configuration values. The data sheet timing from the DRAM will also need to be referenced to extract critical timing parameters. Formulas and look-up tables (LUTs) in the *DDR3 Controller User's Guide* provide the translation from data sheet parameters to the bit-field values. A DDR3 Register Calc spreadsheet is also available to streamline this process. It can be downloaded from the following URL: http://www.ti.com/lit/zip/sprabl2.

The programmed SDRAM timing values below are based on implementation of a 64-bit memory rank composed of four 16-bit SDRAMs. The parts used in this example are 2-Gb DRAMs from Samsung (K4B2G1646C). The DDR3 clock rate is 666.667 MHz. These are the DRAMs and the topology implemented on the C6678 EVM.

Table 2 shows the relevant timing parameters extracted from the DRAM data sheet along with their calculated bit field values.

Table 2. SDRAM Timing Register Values (Continued)

Controller Register	Data Sheet Timing Parameter	Data Sheet Value	Calculated Value (Hexadecimal)
SDTIM1	T_RP	13.5 ns	0x8
	T_RCD	13.5 ns	0x8
	T_WR	15 ns	0x9
	T_RAS(min)	36 ns	0x17
	T_RC	49.5 ns	0x20
	T_RRD (use T_FAW since 8 banks)	45 ns	0x7
	T_WTR	7.5 ns	0x4
SDTIM2	T_XP	6 ns	0x3
	T_XS	170 ns	0x71
	T_XSRD	512 tCK	0x1FF
	T_RTP	7.5 ns	0x4
	T_CKE	5.625 ns	0x3
SDTIM3	T_CKESR	7.125 ns	0x4
	T_ZQCS	64 tCK	0x3F
	T_RFC (min)	160 ns	0x6A

There are several other parameters that must be configured that reside in the SDCFG register. These include impedance settings, which must match the board and SDRAM design and SDRAM size parameters. Some of these affect the operation of the DDR3 memory interface of the DSP and some of them affect the values programmed into the SDRAM Mode Registers. To determine the proper bit field values for this register from the LUTs provided, see the *KeyStone Architecture DDR3 Memory Controller User's Guide* (SPRUGV8).

Table 3 provides an overview of these parameters and shows the choices implemented in the C6678 EVM.

Table 3. SDRAM Configuration Register Values

Controller Register	Interface Parameter Field	Option Chosen	Value From LUT (Hexadecimal)
SDCFG	IBANK_POS	8 banks	0x0
	DDR_TERM	RZQ/6 Ω	0x3
	DYN_ODT	Disable	0x0
	SDRAM_DRIVE	RZQ/7 Ω	0x1
	CWL	CWL = 7 clocks	0x2
	NM	64-bit bus width	0x0
	CL	CAS = 9 clocks	0xA
	ROWSIZE	13 row bits	0x4
	IBANK	8 banks	0x3
	EBANK	DCE0#	0x0
	PAGESIZE	1024-word page	0x2

Each of the register writes is discussed individually in the recommended order.

The DDR_SDRFC register controls the behavior of refresh. To achieve proper initialization timing, the refresh interval must be 31.25 μs rather than the normal 7.8 μs interval to create the necessary initial CKE low period of 500 μs. The 0x5162 portion of this register sets the refresh period to 31.25 μs with the current clock rate. The MSB of this register may be set or cleared at this time to enable or disable SDRAM configuration. It will be cleared at a later time to enable configuration when desired.

Example 9. Refresh Control Register Programming #1

```
DDR_SDRFC    = 0x00005162;    // enable configuration
```

The DDR_SDTIM[3:1] registers contain most of the critical DRAM and controller timing parameters as shown in Table 2. To determine the values to write into these registers, see the *KeyStone Architecture DDR3 Memory Controller User's Guide* (SPRUGV8) and the SDRAM device-specific data sheet. These values are based on the timing parameters extracted from the DRAM data sheet. The sequences in Example 10 calculate and assign the DDR_SDTIM[3:1] values.

Example 10. SDRAM Timing Register Computation and Programming

```
TEMP = 0;
TEMP |= 0x8 << 25;     // T_RP bit field 28:25
TEMP |= 0x8 << 21;     // T_RCD bit field 24:21
TEMP |= 0x9 << 17;     // T_WR bit field 20:17
TEMP |= 0x17 << 12;    // T_RAS bit field 16:12
TEMP |= 0x20 << 6;     // T_RC bit field 11:6
TEMP |= 0x7 << 3;      // T_RRD bit field 5:3
TEMP |= 0x4;           // T_WTR bit field 2:0
DDR_SDTIM1 = TEMP;


TEMP = 0;
TEMP |= 0x3 << 28;     // T_XP bit field 30:28
TEMP |= 0x71 << 16;    // T_XSNR bit field 24:16
TEMP |= 0x1ff << 6;    // T_XSRD bit field 15:6
TEMP |= 0x4 << 3;      // T_RTP bit field 5:3
TEMP |= 0x3;           // T_CKE bit field 2:0
DDR_SDTIM2 = TEMP;

TEMP = 0;
TEMP |= 0x5 << 28;     // T_PDLL_UL bit field 31:28 (fixed value)
TEMP |= 0x5 << 24;     // T_CSTA bit field 27:24 (fixed value)
TEMP |= 0x4 << 21;     // T_CKESR bit field 23:21
TEMP |= 0x3f << 15;    // T_ZQCS bit field 20:15
TEMP |= 0x6A << 4;     // T_RFC bit field 12:4
TEMP |= 0xf;           // T_RAS_MAX bit field 3:0 (fixed value)
DDR_SDTIM3 = TEMP;
```

Alternately, the DDR_SDTIM[3:1] values can be calculated manually from guidance in the *KeyStone Architecture DDR3 Memory Controller User's Guide* (SPRUGV8) or the DDR3 Register Calc spreadsheet and written with the calculated values. The equations above will yield the values shown below. However, the equations are more useful if any of the individual fields are expected to change while tuning the board configuration.

Example 11. SDRAM Timing Register Programming

```
DDR_SDTIM1   = 0x1113783C;
DDR_SDTIM2   = 0x30717FE3;
DDR_SDTIM3   = 0x559F86AF;
```

DDR_PHYCTRL, also known as DDR_PHY_CTRL_1 or DDR_PHYC, programs termination modes for the output buffers and the read latency in the lower five bits. The values supported in the upper bits are given and cannot be changed on customer designs. The read latency is somewhat based on board layout. A conservative value for it is CL+3 because the controller can operate with up to four clocks of round trip delay. The value of 0xF chosen below adds extra margin but also impacts read-to-write turn-around performance. The value 0xC can be used in this field because CL=9 was programmed previously.

Example 12. PHY Control Register Programming

```
DDR_PHYCTRL  = 0x0010010F;
```

The DDR_ZQCFG register configures the adaptive impedance calibration capability. This is a new capability in DDR3. Based on a 240-Ω resistor attached to each DRAM, the DRAM can re-calibrate its output impedance periodically as its temperature changes. This example causes the ZQ calibration to occur once every 100 ms. (This interval is under study.) For more details, see the *KeyStone Architecture DDR3 Memory Controller User's Guide* (SPRUGV8).

Example 13. Dynamic Impedance Configuration Register Programming

```
DDR_ZQCFG    = 0x70073214;
```

PMCTL is the power management control register. Writing a 0 to it disables power management. For more information on the supported features, see the *KeyStone Architecture DDR3 Memory Controller User's Guide* (SPRUGV8). Programming for the power management control register, latency configuration register, class of service registers, interrupt enable registers, and the ECC control registers can be done here or at a later time if these features are needed. Programming these registers is beyond the scope of this application report.

Example 14. Power Management Register Programming

```
DDR_PMCTL    = 0x0;
```

The additional write to DDR_SDRFC has the MSB cleared, which enables DRAM configuration. It still has the refresh interval programmed to the longer number needed during DRAM initialization.

Example 15. Refresh Control Register Programming #2

```
DDR_SDRFC    = 0x00005162;    // enable configuration
```

A write to DDR_SDCFG completes the controller configuration and also causes the mode registers in the DRAM memories to be programmed. The hardware configuration of the DRAM devices occurs immediately because the MSB in DDR_SDRFC is cleared.

The DDR_SDCFG register contains other DRAM and controller configuration parameters as shown in Table 2. To determine the values to write into these registers, see the *KeyStone Architecture DDR3 Memory Controller User's Guide* (SPRUGV8) and the SDRAM device-specific data sheet. Example 16 calculates and assigns the DDR_SDCFG value.

Example 16. SDRAM Configuration Register Computation and Programming

```
TEMP = 0;
TEMP |= 0x3 << 29;     // SDRAM_TYPE bit field 31:29 (fixed value)
TEMP |= 0x0 << 27;     // IBANK_POS bit field 28:27
TEMP |= 0x3 << 24;     // DDR_TERM bit field 26:24
TEMP |= 0x0 << 21;     // DYN_ODT bit field 22:21
TEMP |= 0x1 << 18;     // SDRAM_DRIVE bit field 19:18
TEMP |= 0x2 << 16;     // CWL bit field 17:16
TEMP |= 0x0 << 14;     // NM bit field 15:14
TEMP |= 0xA << 10;     // CL bit field 13:10
TEMP |= 0x4 << 7;      // ROWSIZE bit field 9:7
TEMP |= 0x3 << 4;      // IBANK bit field 6:4
TEMP |= 0x0 << 3;      // EBANK bit field 3:3
TEMP |= 0x2;           // PAGESIZE bit field 2:0
DDR_SDCFG = TEMP;
```

Alternately, the DDR_SDCFG value can be calculated manually from guidance in the *KeyStone Architecture DDR3 Memory Controller User's Guide* (SPRUGV8) or the DDR3 Register Calc spreadsheet and written with the calculated values. The equations in Example 16 yield the value shown below in Example 17. However, the equations above are more useful if any of the individual fields are expected to change while tuning the board configuration.

Example 17. SDRAM Configuration Register Programming

```
DDR_SDCFG = 0x63062A32; // last config write - DRAM init occurs
for(i=0;i<1000;i++);    //Wait 600 us for HW init to complete
```

A delay loop will be needed at this point to force the GEL/software initialization sequence to pause while the hardware initialization completes. The 600-µs minimum delay is required to allow the SDRAM devices to be completely initialized. After hardware initialization completes, the refresh interval can be reprogrammed to the standard operating rate.

Example 18. Refresh Control Register Programming #3

```
DDR_SDRFC = 0x00001450;       //Refresh rate = (7.8*666MHz]
```

3 Leveling Execution

Leveling execution is triggered by the last group of register writes. These writes initialize the write and read leveling circuitry to prepare the DDR byte lanes in the PHY for data transfer.

There are three different DDR PHY write and read leveling operating modes supported in the KeyStone I DSPs.

- **Fixed leveling** - a single set of delay values are used for all nine byte lanes (also known as *ratio forced* leveling)
- **Partial automatic leveling** - automatic write leveling and read gate training with a fixed read eye sample point that may be used with incremental leveling for the read gate
- **Full automatic leveling** - automatic write leveling and read gate training with an adaptive read eye sample point that is achieved with read eye incremental leveling

The last two modes can work with incremental read leveling. Incremental leveling allows the circuitry within the DDR PHY to adapt to delay changes in the DSP, PCB traces and DRAM to optimize the read sampling.

The leveling circuitry measures the delays in the read and write paths. These delays are different from device to device and they also vary when the temperature and voltage change. The device to device variation creates the need for the automatic leveling at system initialization. The read delay variations due to temperature and voltage changes may be compensated for by incremental leveling. Incremental leveling can occur periodically so that the leveling parameters track during full memory operation. This allows the memory interface to operate at higher speeds for longer periods of time without fault. Incremental leveling is highly recommended for long-term DDR3 operation stability at the higher operating rates.

Initial silicon errata limit the leveling modes available on some devices. Table 4 shows the leveling modes available for different version of KeyStone I DSPs.

Table 4. Leveling Modes Supported

KeyStone I Device	Rev.	Fixed Leveling	Partial Automatic Leveling	Partial Automatic and Incremental Leveling	Full Automatic and Incremental Leveling
TCI6616	PG 1.0	X			
TCI6602, TCI6604, TCI6608, C6671, C6672, C6674, and C6678	PG 1.0	X	X	X	
TCI6618 and C6670	PG 1.0 and PG 2.0	X	X	X	X
TCI6602, TCI6604, TCI6608, C6671, C6672, C6674, and C6678	PG 2.0				
TCI6612, TCI6613 and TCI6614	PG 1.x				
C6655 and C6657	PG 1.0				
C6654	PG 1.0				

3.1 Fixed Leveling

The original DDR3 PHY design had a weakness that allowed the read data eye leveling to fail to converge under some conditions. Because all of the leveling modes were combined, that prevented automatic leveling from being functional. Fixed leveling is the only solution for DDR3 operation on PG1.0 TCI6616 DSPs. For other Combined Fixed Ratio Register DSPs, this solution is not recommended.

Fixed leveling (also known as ratio forced leveling) is only a partial solution. In this mode, the DRAM interface operates with automatic leveling circuitry disabled. KeyStone I PG 1.0 Combined Fixed Ratio Register DSPs have only a single set of fixed leveling registers for all nine byte lanes. Therefore, fixed leveling must be used with a mid-point value for each of the leveling parameters and that will be sub-optimal for most byte lanes. The error between these values and the corresponding delays at the first and last DRAM memories consumes part of the data sample window or *eye*. This results in a reduced speed of operation. For this reason, we recommend limiting fixed leveling to DDR3-800 with a 400-MHz DDR3 clock.

The values used are the midpoints of the estimated values from the DDR3 PHY Calc spreadsheet. These values are then written into the DDR3_CONFIG_23 and DDR3_CONFIG_24 registers. Example 19 completes this programming. These lines would be placed at the end of the leveling register configuration discussed in Section 2.3, and the byte lane initialization values would not be needed.

Example 19. Fixed Level Programming

```
#define RD_DQS_SLAVE_RATIO 0x34
#define WR_DQS_SLAVE_RATIO 0xBF
#define WR_DATA_SLAVE_RATIO 0xFF
#define FIFO_WE_SLAVE_RATIO 0x13D

DDR3_CONFIG_REG_23 = RD_DQS_SLAVE_RATIO;
DDR3_CONFIG_REG_23 |= (WR_DQS_SLAVE_RATIO << 10);
DDR3_CONFIG_REG_23 |= (WR_DATA_SLAVE_RATIO << 20);
DDR3_CONFIG_REG_24 = FIFO_WE_SLAVE_RATIO;
```

The Expanded Fixed Ratio Register DSPs can have unique fixed ratio values programmed for each byte lane. This solution is currently not being supported because it is not compatible with automatic leveling.

NOTE: Incremental leveling is not supported in this mode.

3.2 Partial Automatic Leveling

An enhancement was added to the previous DDR3 PHY implementation to fix the read eye sample point problem. This implementation allows the read data eye sample point to be programmed to a fixed value while the other portions of the leveling circuitry could converge as originally intended. This is the hybrid solution referred to as partial automatic leveling.

Because the write leveling and the read gate training can now converge automatically independent of the read eye sample point, the read gate timing can also support incremental adjustments to compensate for timing changes due to voltage and temperature changes. Incremental leveling for the write leveling and read eye sample point are not supported in this version of the DDR3 PHY.

3.2.1 Executing Partial Automatic Leveling

Partial automatic leveling must have all of the byte lane initialization values programmed in Section 2.3 prior to the controller and DRAM initialization. For Combined Fixed Ratio Register DSPs, partial automatic leveling needs the following write to DDR3_CONFIG_REG_23 during the leveling register configuration steps as shown in Example 20.

Example 20. Combined Fixed Ratio Register DSPs Partial Automatic Leveling Programming

```
DDR3_CONFIG_REG_23 |= 0x00000200;
```

For Expanded Fixed Ratio Register DSPs, partial automatic leveling needs the following write to DDR3_CONFIG_REG_52 through DDR3_CONFIG_REG_60 during the leveling register configuration steps as shown in Example 21. Devices in the C665x family do not use DDR3_CONFIG_REG_56 through DDR3_CONFIG_REG_59.

Example 21. Expanded Fixed Ratio Register DSPs Partial Automatic Leveling Programming

```
DDR3_CONFIG_REG_52 |= 0x00000200;
DDR3_CONFIG_REG_53 |= 0x00000200;
DDR3_CONFIG_REG_54 |= 0x00000200;
DDR3_CONFIG_REG_55 |= 0x00000200;
DDR3_CONFIG_REG_56 |= 0x00000200;
DDR3_CONFIG_REG_57 |= 0x00000200;
DDR3_CONFIG_REG_58 |= 0x00000200;
DDR3_CONFIG_REG_59 |= 0x00000200;
DDR3_CONFIG_REG_60 |= 0x00000200;
```

After the controller and DRAM initialization is complete, automatic leveling can be triggered. The remaining register writes simply enable and then start the initialization process. The RDWR_LVL_EN field is the MSB of RDWR_LVL_RMP_CTRL and the RDWR_LVL_FULL_START field is the MSB of RDWR_LVL_CTRL. Note that the other bit fields in these registers are all cleared to 0. These can be programmed at a later time to enable incremental leveling.

The writes in Example 22 trigger automatic convergence of the write leveling and the read gate training while leaving the default fixed value for the read eye sample point as indicated by the prior write to DDR3_CONFIG_REG_23 for Combined Fixed Ratio Register DSPs and DDR3_CONFIG_REG_52 through DDR3_CONFIG_REG_60 for Expanded Fixed Ratio Register DSPs. Note that the 3-ms stall after enabling leveling is required for proper operation.

Example 22. Partial Automatic Leveling Enable

```
RDWR_LVL_RMP_CTRL = 0x80000000;
RDWR_LVL_CTRL = 0x80000000;
for(i=0;i<1000;i++);          //Wait 3ms for leveling to complete
```

If there is concern that the leveling is not converging correctly, the timeout register bits can be read at this time to verify that the leveling completed as expected. The leveling timeout indications are bits 4, 5, and 6 in the DDR3 memory controller status register at address 0x21000004. If any of these bits are set to a 1, leveling has failed. This normally means there is either a hardware problem or the initial leveling values are incorrect. There is also an IFRDY bit in this register that will be set if leveling completes successfully.

3.2.2 Enabling Incremental Leveling with Partial Automatic Leveling

Incremental leveling after partial automatic leveling is optional. Separate control is available to enable or disable incremental adjustments to each of the leveling modes: write leveling, read eye training and read gate training. Incremental leveling after partial automatic leveling can be enabled only for read gate training. For a detailed explanation of these fields, see the *KeyStone Architecture DDR3 Memory Controller User's Guide* (SPRUGV8).

The following writes in Example 23 will enable incremental gate training. It will perform these incremental adjustments on a 30-μs interval during a SmartReflex voltage change and a 10-ms interval during normal operation. The ramp window is also set to 10 ms. (These intervals are under study.)

Example 23. Incremental Leveling After Partial Automatic Leveling

```
RDWR_LVL_RMP_WIN = 0x00000502;
RDWR_LVL_RMP_CTRL = 0x80000300;
RDWR_LVL_CTRL = 0xFF000900;
```

3.3 Full Automatic Leveling

The latest KeyStone I DSPs contain additional enhancement to allow all three leveling modes to be used. The write leveling, read data eye training, and read gate training are all triggered for initial convergence. However, the read eye sample point may still be invalid. Incremental leveling will then force the read eye sample point to a good starting value and then it will robustly optimize the read eye sample point after multiple successive iterations. Simultaneously, read gate timing will also robustly track voltage-driven and temperature-driven timing changes during the incremental leveling iterations.

Full automatic leveling must have all of the byte lane initialization values programmed in the leveling register configuration section prior to the controller and DRAM initialization. There must be no writes to any fixed ratio registers when full automatic leveling is used.

Once the controller and DRAM initialization is complete, the remaining register writes simply enable and then start the initialization process. The RDWR_LVL_EN field is the MSB of RDWR_LVL_RMP_CTRL and the RDWR_LVL_FULL_START field is the MSB of RDWR_LVL_CTRL. Note that the other bit fields in these registers are all cleared to 0. These will be programmed at a later time to enable incremental leveling. Note that the 3-ms stall after enabling leveling is required for proper operation.

Example 24. Full Automatic Leveling Enable

```
RDWR_LVL_RMP_CTRL = 0x80000000;
RDWR_LVL_CTRL = 0x80000000;
for(i=0;i<1000;i++);          //Wait 3ms for leveling to complete
```

If there is concern that the leveling is not converging, the timeout register bits can be read at this time to verify that the leveling completed as expected. The leveling timeout indications are bits 4, 5, and 6 in the DDR3 Memory Controller Status Register at address 0x21000004. If any of these bits are set to a 1, leveling has failed. This normally means there is either a hardware problem or the initial leveling values are incorrect. There is also an IFRDY bit in this register that will be set if leveling completes successfully.

Incremental leveling of at least the read eye sample point must be executed at least 64 times after full automatic leveling to converge it to an initial optimum value. Separate control is available to enable or disable incremental adjustments to each of the leveling modes - write leveling, read eye training and read gate training. Incremental leveling after a full automatic leveling may be enabled for read eye and read gate training. For a detailed explanation of these fields, see the *KeyStone Architecture DDR3 Memory Controller User's Guide* (SPRUGV8).

The writes in Example 25 will enable the read eye and read gate training modes of incremental leveling. It will perform these incremental adjustments on a 30-μs interval during a SmartReflex voltage change and a 10-ms interval during normal operation. The ramp window is also set to 10 ms. (These intervals are under study.)

Example 25. Incremental Leveling After Full Automatic Leveling

```
RDWR_LVL_RMP_WIN = 0x00000502;
RDWR_LVL_RMP_CTRL = 0x80030300;
RDWR_LVL_CTRL = 0x7F090900;
```

After minimum of 64 times, incremental leveling is disabled by the following writes as shown in Example 26.

Example 26. Incremental Leveling After Full Automatic Leveling

```
RDWR_LVL_RMP_CTRL = 0x0;
RDWR_LVL_CTRL = 0x0;
```

4 Dual Rank Support

The procedure above is sufficient for DDR3 topologies that only contain a single rank of memory. This was the only topology supported by KeyStone I devices since they did not support dual-rank UDIMMs. All dual-rank UDIMMs available previously required the DDR3 Controller to support address mirroring. The DDR3 Controller on KeyStone I devices does not support address mirroring. This has changed recently with the volume availability of twin-die SDRAM memory devices. These are now cost-effective for use with KeyStone I devices. The procedure described in the previous sections will support DDR3 memory topologies using twin-die devices with the changes listed below.

The DDR3 PHY logic in the KeyStone I devices supports use of a single set of leveling values for memory transactions (write and reads) on both rank 0 and rank 1. This is enabled by setting the use_rank0_delays bit in the DDR3_CONFIG_REG_12 register during the Leveling Register Configuration portion of the initialization procedure discussed previously in Section 2.3. Therefore, Example 4 from that section would now be replaced with the contents of Example 27.

Example 27. INVERT_CLKOUT and USE_RANK0_DELAYS Programming

```
DDR3_CONFIG_REG_0 &= ~(0x007FE000); // clear ctrl_slave_ratio field
DDR3_CONFIG_REG_0 |= 0x00200000;    // set ctrl_slave_ratio to 0x100
DDR3_CONFIG_REG_12 |= 0x08000000;   // Set invert_clkout = 1
DDR3_CONFIG_REG_12 |= 0x01000000;   // Set use_rank0_delays = 1
```

This setting allows the PHY configuration and leveling to be conducted only on rank 0 (the SDRAM dies attached to CE0). It also prepares the PHY to use the resulting leveling values for accesses to both rank 0 and rank 1.

DDR_ZQCFG will also need to be changed. The MSB must be set to enable impedance calibration on each rank of SDRAMs. This is discussed back in Section 2.4. Therefore, Example 13 from that section would now be replaced with the contents of Example 28.

Example 28. Dynamic Impedance Configuration Register Programming

```
DDR_ZQCFG = 0xF0073214;
```

The following DDR_SDCFG write to set the EBANK bit is needed to complete the SDRAM configuration for rank 1 (the SDRAM dies attached to CE1). Writing to DDR_SDCFG will result in an SDRAM initialization sequence which updates the Mode Registers for both ranks. This will be a second write to DDR_SDCFG after the SDRAM initialization and leveling has already been completed for rank 0. Therefore, it is added at the very end of the procedure after all of the leveling steps in Section 3 have been completed.

Example 29. SDRAM Configuration Register Programming

```
DDR_SDCFG = 0x63062A3A; // Set ebank = 1
for(i=0;i<1000;i++); //Wait 600us for HW init to complete
```

5 Lock KICK Registers

At the end of the DDR3 controller configuration, the KICK registers may be locked. This can be done immediately after the DDR3 controller configuration or later after other memory regions protected by this mechanism are completely configured.

Example 30. Lock Kick Registers

```
KICK0 = KICK0_LOCK;             //lock kicker registers
KICK1 = KICK1_LOCK;
```

Note that the need to lock the KICK registers is application-dependent. If the application will allow multiple cores to access the bootcfg registers, there may be race conditions (see the device-specific Errata). One workaround for this problem is not locking the KICK registers at the end of this process.

6 References

- *KeyStone Architecture DDR3 Memory Controller User's Guide* (SPRUGV8)
- *DDR3 Design Requirements for KeyStone Devices* (SPRABI1)
- *KeyStone Architecture DSP Bootloader User's Guide* (SPRUGY5)
- *KeyStone Architecture Phase-Locked Loop (PLL) User's Guide* (SPRUGV2)

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from D Revision (January 2015) to E Revision	Page
• New information was added to Section 4.	16

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

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

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license 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 significant portions of TI 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. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

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

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

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