TMS320VC5509A
Digital Signal Processor
Silicon Revision 1.0 and 1.1

Silicon Errata

Literature Number: SPRZ200F
June 2003–Revised September 2010

1 Introduction **5**
1.1 Device and Development-Support Tool Nomenclature 5
1.2 Revision Identification 6
2 Silicon Revision 1.0 and 1.1 Usage Notes and Known Design Exceptions to Functional Specifications **8**
2.1 Usage Notes for Silicon Revision 1.0 and 1.1 8
2.1.1 RTC: Seconds Alarm Functionality 8
2.1.2 Attempting to Idle CPU Domain with INTx Asserted. 8
2.2 Silicon Revision 1.0 and 1.1 Known Design Exceptions to Functional Specifications 9
Revision History **42**

List of Figures

1 PGE Example and Device Revision Codes 6
2 GHH Example and Device Revision Codes 6
3 Bad TCK Transition 38
4 Good TCK Transition 39

List of Tables

1 Determining Silicon Revision From Lot Trace Code 7
2 Silicon Revision 1.0 and 1.1 Advisory List 9

TMS320VC5509A Digital Signal Processor **Silicon Revision 1.0 and 1.1**

1 Introduction

This document describes the known exceptions to the functional specifications for the TMS320VC5509A devices (i.e.,144-pin LQFP, PGE suffix and 179-pin MicroStar BGA™, GHH suffix) . For more detailed information on this device, see the device-specific data manual:

- *TMS320VC5509A Fixed-Point Digital Signal Processor* data manual (Literature Number SPRS205)
- *TMS320C55x DSP CPU Programmer's Reference Supplement* (Literature Number SPRU652)

Throughout this document, unless otherwise specified, 5509A, C5509A, and VC5509A refers to the TMS320VC5509A device.

The advisory numbers in this document are not sequential. Some advisory numbers have been moved to the next revision and others have been removed and documented in the user's guide. When items are moved or deleted, the remaining numbers remain the same and are not resequenced.

1.1 Device and Development-Support Tool Nomenclature

To designate the stages in the product development cycle, TI assigns prefixes to the part numbers of all DSP devices and support tools. Each DSP commercial family member has one of three prefixes: TMX, TMP, or TMS (e.g.,TMS320VC5509A). Texas Instruments recommends two of three possible prefix designators for its support tools: TMDX and TMDS. These prefixes represent evolutionary stages of product development from engineering prototypes (TMX/TMDX) through fully qualified production devices/tools (TMS/TMDS).

Device development evolutionary flow:

TMX — Experimental device that is not necessarily representative of the final device's electrical specifications.

TMP — Final silicon die that conforms to the device's electrical specifications but has not completed quality and reliability verification.

TMS — Fully-qualified production device.

Support tool development evolutionary flow:

TMDX — Development-support product that has not yet completed Texas Instruments internal qualification testing.

TMDS — Fully qualified development-support product.

TMX and TMP devices and TMDX development-support tools are shipped against the following disclaimer:

"Developmental product is intended for internal evaluation purposes."

TMS devices and TMDS development-support tools have been characterized fully, and the quality and reliability of the device have been demonstrated fully. TI's standard warranty applies.

Predictions show that prototype devices (TMX or TMP) have a greater failure rate than the standard production devices. Texas Instruments recommends that these devices not be used in any production system because their expected end-use failure rate still is undefined. Only qualified production devices are to be used.

TI device nomenclature also includes a suffix with the device family name. This suffix indicates the package type (for example, ZCH), the temperature range (for example, "Blank" is the commercial temperature range), and the device speed range in megahertz (for example, "10" is the default 100 MHz device).

1.2 Revision Identification

Figure 1 and Figure 2 provides example(s) of the TMS320VC5509A device markings. The device revision can be determined by the symbols marked on the top of the package.

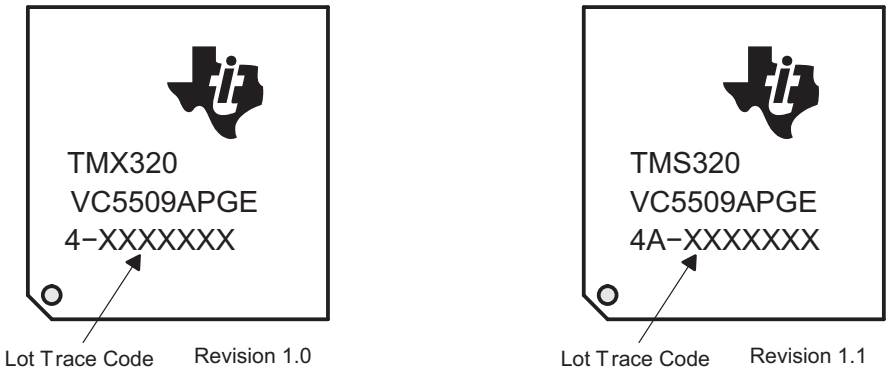
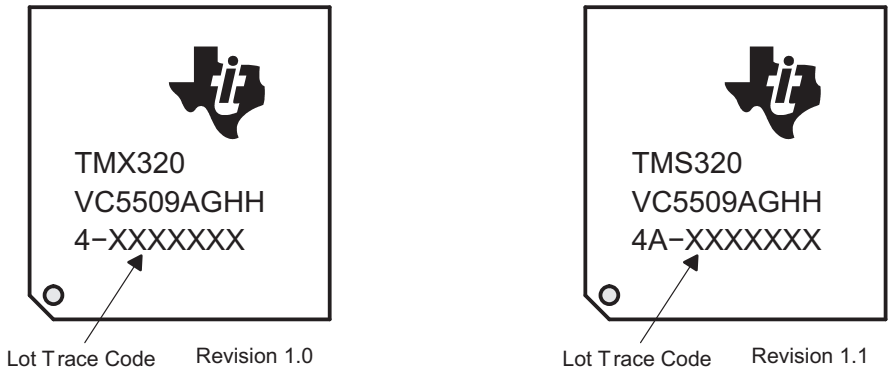


Figure 1. PGE Example and Device Revision Codes



NOTE: Early revisions may have a letter “D” in front of “VC5509A” in the GHH package.

Figure 2. GHH Example and Device Revision Codes

The device revision can be determined by the lot trace code marked on the top of the package. The locations for the lot trace codes for the PGE and the GHH packages are shown in Figure 1 and Figure 2, respectively. The location of other markings may vary per device.

Qualified devices in the PGE and GHH packages are marked with the letters “TMS” at the beginning of the device name, while nonqualified devices are marked with the letters “TMX” or “TMP” at the beginning of the device name. Table 1 lists the information associated with each silicon revision.

Table 1. Determining Silicon Revision From Lot Trace Code

Lot Trace Code	Silicon Revision
Blank (no second letter in prefix)	Indicates Original Silicon (1.0)
A (second letter in prefix is A)	Indicates Silicon Revision A (1.1)

2 Silicon Revision 1.0 and 1.1 Usage Notes and Known Design Exceptions to Functional Specifications

This section describes the usage notes and advisories that apply to silicon revision 1.0 and 1.1 of the TMS320VC5509A device.

2.1 Usage Notes for Silicon Revision 1.0 and 1.1

Usage notes highlight and describe particular situations where the device's behavior may not match presumed or documented behavior. This may include behaviors that affect device performance or functional correctness. These usage notes will be incorporated into future documentation updates for the device (such as the device-specific data sheet), and the behaviors they describe will not be altered in future silicon revisions.

2.1.1 RTC: Seconds Alarm Functionality

On the 5509A device, the Seconds Alarm Register (RTCSECA) cannot be used to generate an alarm every second, but the update-ended interrupt can.

The Real-Time Clock (RTC) executes an update cycle once per second to update the current time in the time/calendar registers:

- Seconds Register (RTCSEC)
- Minutes Register (RTCMIN)
- Hours Register (RTCHOUR)
- Day of the Week and Day Alarm Register (RTCDAYW)
- Day of the Month (Date) Register (RTCDAYM)
- Month Register (RTCMONTH)
- Year Register (RTCYEAR)

At the end of every update cycle, the RTC sets the update-ended interrupt flag (UF) in the Interrupt Flag Register (RTCINTFL). If the update-ended interrupt enable bit (UIE) in the Interrupt Enable Register (RTCINTEN) is set to 1, an interrupt request is sent to the CPU.

2.1.2 Attempting to Idle CPU Domain with INTx Asserted.

The C5509A DSP can idle CPU domain to save power consumption and the CPU can be re-activated by the external INTx interrupts. When the CPU is in idle state, the INTx becomes level-sensitive (active low). This means that a low state on the INTx pin will generate interrupts when the CPU is in idle mode. If the corresponding interrupt resource is enabled, it will cause the CPU to exit the idle state. Thus, attempting to idle the CPU domain with (an) INTx interrupt(s) enabled and the INTx pin(s) held low will be unsuccessful.

2.2 Silicon Revision 1.0 and 1.1 Known Design Exceptions to Functional Specifications

Table 2. Silicon Revision 1.0 and 1.1 Advisory List

Title	Page
Advisory DL_2 — Software Modification of MPNMC Bit is Not Pipeline-Protected	10
Advisory DL_7 — RETI Instruction may Affect the XF State	11
Advisory DL_10 — First Word of Data on Consecutive DMA Transmissions Using McBSP is Lost	12
Advisory BL_3 — USB Bootloader Returns Incorrect DescriptorType Value When String Descriptors are Requested by the Host	13
Advisory BL_4 — USB Bootloader Returns Incorrect PID During Enumeration Phase	14
Advisory DMA_1 — Early Sync Event Stops Block Transfer	15
Advisory DMA_2 — DMA Does Not Support Burst Transfers From EMIF to EMIF	16
Advisory EMIF_8 — ARDY Pin Requires Strong Pullup Resistor	17
Advisory EMIF_9 — External Memory Write After Read Reversal	18
Advisory EMIF_10 — Block Write Immediately Following a Block Read May Cause Data Corruption	19
Advisory EMIF_11 — EMIF Asynchronous Access Hold = 0 is Not Valid for Strobe > 3	20
Advisory EMIF_12 — 8-Bit Asynchronous Writes on 5509A EMIF Not Supported	21
Advisory EMIF_13 — After Changing CE Control Registers and Disabling SDRAM Clock in Divide-by-8 and Divide-by-16 Modes, Asynchronous Access Followed by SDRAM Access Will Not Supply a Ready Signal to CPU	22
Advisory EMIF_14 — SETUP = 2 Configuration is not Valid for Asynchronous Memory	23
Advisory EHPI_5 — HPID Read Following a HPID Write While HRDY Low Corrupts the Read	24
Advisory EHPI_6 — HPIC/HPIA Access Following an Autoincremented HPID Write Causes Next HPID Address to Increment to the Incorrect Address	25
Advisory EHPI_7 — HRDY is Always Driven	26
Advisory RTC_3 — RTC Interrupts are Perceived by the User as Happening One Second Before	27
Advisory RTC_4 — Any Year Ending in 00 Will Appear as a Leap Year	28
Advisory RTC_5 — Midnight and Noon Transitions Do Not Function Correctly in 12h Mode	29
Advisory USB_2 — CPU Might Miss Back-to-Back USB Interrupts When CPU Speed is Less Than or Equal to 24 MHz	30
Advisory USB_5 — USB Input Cell Does Not Power Down When USB is Placed in IDLE	31
Advisory USB_6 — CPU Read/Write to USB Module may Return Incorrect Result if the USB Clock is Running Slower Than Recommended Speed (48 MHz)	32
Advisory I2C_3 — ARDY Interrupt is not Generated Properly in Non-Repeat Mode if STOP Bit is Set	33
Advisory I2C_5 — Repeated Start Mode Does Not Work	34
Advisory I2C_6 — Bus Busy Bit Does Not Reflect the State of the I^2C Bus When the I^2C is in Reset	35
Advisory I2C_8 — DMA Receive Synchronization Pulse Gets Generated Falsely	36
Advisory MCBSP_1 — McBSP May Not Generate a Receive Event to DMA When Data Gets Copied From RSR to DRR	37
Advisory EMU_1 — Emulation Prone to Failure Under Certain Situations	38
Advisory PM_1 — Repeated Interrupts During CPU in Idle	40
Advisory HWA_1 — Pixel Interpolation Hardware Accelerator	41

Advisory DL_2 *Software Modification of MPNMC Bit is Not Pipeline-Protected*

Revision(s) Affected	1.0 and 1.1
Details	Software modification of the MPNMC bit in status register 3 (ST3_55) is not pipeline-protected so changes to the device memory map may not become valid before the instructions that immediately follow the modification.
Assembler Notification	None
Workaround(s)	Insert six NOPs after the MPNMC modification.

Advisory DL_7 *RETI Instruction may Affect the XF State*

Revision(s) Affected 1.0 and 1.1

Details The XF pin state is saved on the stack as a part of the ST1 context saving during interrupts servicing. If the XF pin state is changed inside the ISR, upon execution of the RETI, the XF bit will be restored to the value prior to entering the ISR. If XF state is not changed inside the ISR, then there is no issue.

Assembler Notification None

Workaround(s) BIOS takes care of this problem with software workaround, which is transparent to the users. Non-BIOS users who are changing XF pin state in an ISR should also modify the ST1 value on the stack to maintain the correct XF pin state upon exiting the ISR.

Advisory DL_10 *First Word of Data on Consecutive DMA Transmissions Using McBSP is Lost*

Revision(s) Affected 1.0 and 1.1

Details When executing multiple DMA transfers consecutively using the same DMA Transmit Channel and McBSP, an extra DMA TX request generated by the McBSP at the end of the first transfer will not be serviced by the DMA until the next DMA transfer is initiated by the McBSP. At the next DMA transfer, this DMA TX request will be serviced as soon as the DMA TX channel is enabled.

This transmitted data will remain valid on the bus as long as the McBSP is disabled. However, once the McBSP is enabled, it sends out another DMA TX request, and the DMA transmits the second word. This results in the loss of the first word of data on consecutive DMA transmissions.

Assembler Notification None

Workaround(s) Only the systems where McBSP is turned off following each block of DMA transfer are affected. In such case, a dummy DMA transfer with the DMA synchronization event set to no sync event will flush out the pending TX request from the McBSP before programming the DMA to send the next block of data to the McBSP.

Advisory BL_3 *USB Bootloader Returns Incorrect DescriptorType Value When String Descriptors are Requested by the Host*

Revision(s) Affected 1.0 and 1.1

Details When the host requests for the string descriptor, the USB bootloader returns 0x00 for DescriptorType value instead of 0x03.

Assembler Notification None

Workaround(s) Ignore the DescriptorType returned by the DSP. String Descriptor is not necessary for successfully bootloading the device through the USB.

Advisory BL_4 *USB Bootloader Returns Incorrect PID During Enumeration Phase*

Revision(s) Affected	1.0 and 1.1
Details	The correct PID is 0x9003; however, the bootloader reports 0x9001 to the host.
Assembler Notification	None
Workaround(s)	PID 0x9001 belongs to 5509. Both 5509 and 5509A USB bootloader work the same way; hence, the incorrect PID does not affect the functionality of the USB bootmode.

Advisory DMA_1 *Early Sync Event Stops Block Transfer*

Revision(s) Affected 1.0 and 1.1

Details When a DMA block transfer is initiated by a sync event, if the same sync event occurs before the last element of the block transfer has been completed, an event drop occurs and the channel becomes disabled.

Assembler Notification None

Workaround(s) Ensure that the duration between the sync events is long enough to allow the block transfer to complete. The DMA end-of-block interrupt can be used as an indicator.

Advisory DMA_2 *DMA Does Not Support Burst Transfers From EMIF to EMIF*

Revision(s) Affected	1.0 and 1.1
Details	The DMA controller does not support burst mode transfers with the EMIF as both the source and the destination port.
Assembler Notification	None
Workaround(s)	Do not use burst mode for EMIF-to-EMIF transfers.

Advisory EMIF_8 *ARDY Pin Requires Strong Pullup Resistor*

Revision(s) Affected 1.0 and 1.1

Details When the parallel bus is used to access external memory, a strong pullup resistor is required for the ARDY pin for the asynchronous memory interface.

Assembler Notification None

Workaround(s) Pull up ARDY with a 2.2-kΩ resistor.

Advisory EMIF_9 *External Memory Write After Read Reversal*

Revision(s) Affected 1.0 and 1.1

Details If an external memory write is followed immediately by an external memory read, the external memory read will occur first, followed by the write. See the example below.

Example:

```
MOV #1770h, *(100001h) ; External Memory Write
MOV *(#100000h), AR1   ; External Memory Read
```

Assembler Notification None

Workaround(s) Insert two NOPs between the memory write/read pair.

Example:

```
MOV #1770h, *(100001h) ; External Memory Write
NOP
NOP
MOV *(#100000h), AR1   ; External Memory Read
```

Advisory EMIF_10 *Block Write Immediately Following a Block Read May Cause Data Corruption*

Revision(s) Affected 1.0 and 1.1

Details When performing a block write immediately following a block read, data may get corrupted. See the example below.

Example:

```
Write 0x55 to addr1
Write 0xAA to addr2
Read addr1
Read addr2
```

When executed, the above code will follow this order:

```
Write 0x55 to addr1
Read addr1
Write 0xAA to addr2
Read addr2
```

Assembler Notification None

Workaround(s) Insert two NOPs between write and read. Since reads occur before writes in the pipeline, the read must be delayed after the write so that the read does not occur before the write.

Advisory EMIF_11 *EMIF Asynchronous Access Hold = 0 is Not Valid for Strobe > 3*

Revision(s) Affected 1.0 and 1.1

Details For asynchronous EMIF accesses, a hold time of 0 is not valid for strobe lengths greater than 3 cycles if the ARDY_OFF bit is not set. If the above configuration is used but the ARDY_OFF bit is clear, then the EMIF automatically gives a hold time of 1 cycle.

Assembler Notification None

Workaround(s) None

Advisory EMIF_12 *8-Bit Asynchronous Writes on 5509A EMIF Not Supported*

Revision(s) Affected	1.0 and 1.1
Details	8-bit asynchronous writes are not supported; however, 8-bit asynchronous reads are supported.
Assembler Notification	None
Workaround(s)	None

Advisory EMIF_13 *After Changing CE Control Registers and Disabling SDRAM Clock in Divide-by-8 and Divide-by-16 Modes, Asynchronous Access Followed by SDRAM Access Will Not Supply a Ready Signal to CPU*

Revision(s) Affected 1.0 and 1.1

Details If the SDRAM clock (EMIF.CLKMEM) is set to divide-by-8 and divide-by-16 of the CPU clock and if the user disables the SDRAM clock before accessing asynchronous memory, the EMIF will fail to supply the ready signal to the CPU under the following two conditions:

- Case 1:
  SDRAM access
  Switch off the SDRAM clock
  Change CE Space Control Register to Asynchronous Mode
  Perform an asynchronous access to the same CE space
- Case 2:
  SDRAM access
  Switch off the SDRAM clock
  Change CE Space Control Register to Asynchronous Mode
  Perform an asynchronous access to a different CE space

This failure of the ready signal will make the CPU wait indefinitely.

Assembler Notification None

Workaround(s) Switch the SDRAM clock to divide-by-1 before programming the CE Space Control Register to asynchronous memory.

Advisory EMIF_14 *SETUP = 2 Configuration is not Valid for Asynchronous Memory*

Revision(s) Affected 1.0 and 1.1

Details When using the EMIF in asynchronous memory mode, a read or write SETUP time setting of two clocks actually behaves like timing of one clock of setup time.

Assembler Notification None

Workaround(s) If a read setup time of two clocks is required for asynchronous memory, a value of three clock cycles must be used.

Advisory EHPI_5 *HPID Read Following a HPID Write While HRDY Low Corrupts the Read*

Revision(s) Affected

1.0 and 1.1

Details

Whether in muxed or non-muxed mode, if a data read overlaps with a HRDY low from a previous data write, the EHPI address used for the read access will be the same as the one preceding the data write.

The only case not affected by this bus is the non-autoincremented write−>read from the same address. In the case of autoincremented write followed by autoincremented read, normally, the address will be incremented after the write, but if the read access is initiated while the HRDY is low, it will not be incremented due to the problem. If one more autoincremented read is performed after this, the EHPI address used will be twice the incremented version.

Assembler Notification

N/A

Workaround(s)

- Muxed Mode: Following a sequence of EHPI write(s), perform a HPIA update before initiating an EHPI read access.
- Non-Muxed Mode: Following a sequence of EHPI write(s), add a dummy HPID read before initiating the actual EHPI read access.

Advisory EHPI_6 *HPIC/HPIA Access Following an Autoincremented HPID Write Causes Next HPID Address to Increment to the Incorrect Address*

Revision(s) Affected 1.0 and 1.1

Details If any HPIA/HPIC access is pipelined with a previous autoincremented write, the succeeding autoincremented HPID access (whether write or read) will use the same address as the previous write. In other words, the address will not be incremented. Only Muxed Mode is affected.

Assembler Notification None

Workaround(s) Muxed Mode: Following a sequence of EHPI write(s), perform an HPIA update before initializing an EHPI read/write access.

Advisory EHPI_7 *HRDY is Always Driven*

Revision(s) Affected 1.0 and 1.1

Details HRDY is always driven to the same value as its internal state. This is only a problem when a single system has devices that have ready signals that are used in conjunction with the DSP's HRDY signal. This could include a single system with multiple DSPs using their HRDY signals in conjunction to signal a master device of their ready status.

Workaround(s) Logically AND all ready signals with the DSP's HRDY signal to generate a valid ready status.

Advisory RTC_3 *RTC Interrupts are Perceived by the User as Happening One Second Before*

Revision(s) Affected 1.0 and 1.1

Details When the user reads the Real Time Clock time register, these register are read one second after the RTC's internal timer counter register. The RTC interrupts are triggered by the internal counter register, thus it seems to the user that the interrupt was triggered one second earlier. For example, an alarm set to every minute alarm generates an interrupt at xx:xx:59 instead of xx:xx:00.

Assembler Notification None

Workaround(s) Take into account the one second difference when using the alarm interrupt.

Advisory RTC_4 *Any Year Ending in 00 Will Appear as a Leap Year*

Revision(s) Affected 1.0 and 1.1

Details Since the year can be varied from 00−99 only, any year ending with 00 will always appear as a Leap Year, which is not always the case. For example, 2100 ends in 00 and is not a Leap Year.

Assembler Notification None

Workaround(s) None

Advisory RTC_5 *Midnight and Noon Transitions Do Not Function Correctly in 12h Mode*

Revision(s) Affected 1.0 and 1.1

Details The normal transition from Midnight and Noon should be the following:
11:59am → 12:00pm → 12:59pm → 1:00pm
11:59pm → 12:00am → 12:59am → 1:00am

However, if the RTC is used in the 12h time format, the transitions around Noon and Midnight are as below:
11:59am → 12:00am → 12:59am → 1:00pm
11:59pm → 12:00pm → 12:59pm → 1:00am

Assembler Notification None

Workaround(s) The problem can be worked around using the 24h mode.

Advisory USB_2 *CPU Might Miss Back-to-Back USB Interrupts When CPU Speed is Less Than or Equal to 24 MHz*

Revision(s) Affected 1.0 and 1.1

Details When the CPU operates with a clock less than or equal to half the USB module clock, back-to-back USB interrupts might be missed by the CPU. Back-to-back interrupts occur when multiple endpoints are active simultaneously or when SOF or SETUP events occur with one endpoint. The USB module needs to operate at 48 MHz, so the CPU needs to operate at a clock speed greater than 24 MHz.

Assembler Notification None

Workaround(s) Recommended CPU operating frequency is 48 MHz or higher if the USB module is running.

Advisory USB_5 *USB Input Cell Does Not Power Down When USB is Placed in IDLE*

Revision(s) Affected	1.0 and 1.1
Details	USB input cells are always powered unless the oscillator is disabled.
Assembler Notification	None
Workaround(s)	None

Advisory USB_6 *CPU Read/Write to USB Module may Return Incorrect Result if the USB Clock is Running Slower Than Recommended Speed (48 MHz)*

Revision(s) Affected 1.0 and 1.1

Details If the CPU speed is x12 or higher than the USB module clock speed, then the USB RAM and register read/write will return incorrect result. This is not an issue during normal USB operation where the USB module clock is 48 MHz. But at power up, the USB DPLL is in bypass div2 mode; hence, the USB module clock is CLKIN/2. As most of the applications program the DSP PLL first and then all other modules (including USB), this can be a problem if the (CPU clock):(USB module clock) ratio > 12:1.

Assembler Notification None

Workaround(s) Program the USB PLL first to speed up the USB module clock to 48 MHz before programming the DSP PLL.

Advisory I2C_3 *ARDY Interrupt is not Generated Properly in Non-Repeat Mode if STOP Bit is Set*

Revision(s) Affected 1.0 and 1.1

Details In non-repeat mode, if the STP bit of ICMDR is set, the master sends the STOP condition and does not assert ARDY interrupt after sending data. If the STP bit is set, the I^2C sends the STOP condition and clears the ARDY bit.

Assembler Notification None

Workaround(s) If the ARDY interrupt is desired after sending data, start the data transfer without setting the STP bit. If the STOP bit is not set beforehand, the master will not send the STOP condition and asserts the ARDY interrupt after sending the data. Set the STP bit when the last ARDY interrupt arrives (all data sent out).

Advisory I2C_5 *Repeated Start Mode Does Not Work*

Revision(s) Affected	1.0 and 1.1
Details	Repeated Start Mode does not work on the I^2C peripheral.
Assembler Notification	None
Workaround(s)	None

Advisory I2C_6 *Bus Busy Bit Does Not Reflect the State of the I^2C Bus When the I^2C is in Reset*

Revision(s) Affected

1.0 and 1.1

Details

The Bus Busy bit (BB) indicates the status of the I^2C bus. The Bus Busy bit is set to '0' when the bus is free and set to '1' when the bus is busy. The I^2C peripheral cannot detect the state of the I^2C bus when it is in reset (IRS bit is set to '0'); therefore, the Bus Busy bit will keep the state it was at when the peripheral was placed in reset. The Bus Busy bit will stay in that state until the I^2C peripheral is taken out of reset (IRS bit set to '1') and a START condition is detected on the I^2C bus. When the device is powered up, the Bus Busy bit will stay stuck at the default value of '0' until the IRS bit is set to '1' and the I^2C peripheral detects a START condition.

Systems using a multi-master configuration can be affected by this issue.

Assembler Notification

None

Workaround(s)

Wait a certain period after taking the I^2C peripheral out of reset (setting the IRS bit to '1') before starting the first data transfer. The period should be set equal to or larger than the total time it takes for the longest data transfer in the application. By waiting this amount of time, it can be ensured that any previous transfers finished. After this point, BB will correctly reflect the state of the I^2C bus.

Advisory I2C_8 *DMA Receive Synchronization Pulse Gets Generated Falsely*

Revision(s) Affected 1.0 and 1.1

Details When receiving an I^2C data stream in master mode (i.e., a read is performed), and the DMA is started, a DMA synchronization event is triggered upon enabling the DMA channel if a byte is present in the DRR (even if it has already been read). This leads to the first byte read being a duplicate of the previous byte that was already read from the DRR.

Assembler Notification None

Workaround(s) Set DMA transfers from DRR to read one more byte than necessary and discard the first byte.

Advisory MCBSP_1 *McBSP May Not Generate a Receive Event to DMA When Data Gets Copied From RSR to DRR*

Revision(s) Affected 1.0 and 1.1

Details When there is heavy peripheral activity, and the DRR is read, a new receive interrupt might not be generated to the DMA when data in the RSR is copied to the DRR. When this condition occurs, the McBSP overwrites the DRR before the DMA had an opportunity to read its value.

This problem arises when the DRR read occurs at the “exact moment” the REVT needs to be generated. The DRR servicing gets delayed if there are other heavy DMA channels or CPU activities on the peripheral bus.

Assembler Notification None

Workaround(s) Optimize the peripheral bus access by the CPU and the DMA by carefully scheduling the DMA and CPU activities so the DMA channel servicing the DRR is not stalled to the point where new data is about the move in the DRR.

Advisory EMU_1 *Emulation Prone to Failure Under Certain Situations*

Revision(s) Affected 1.0 and 1.1

Details

Under certain conditions, the emulation hardware may corrupt the emulation control state machine or may cause it to lose synchronization with the emulator software. When emulation commands fail as a result of the problem, Code Composer Studio™ Integrated Development Environment (IDE) may be unable to start or it may report errors when interacting with the TMS320C55x™ DSP (for example, when halting the CPU, reaching a breakpoint, etc.).

This phenomenon is observed when an erroneous clock edge is generated from the TCK signal inside the C55x™ DSP. This can be caused by several factors, acting independently or cumulatively:

- TCK transition times (as measured between 2.5 V and 0.6 V) in excess of 3 ns.
- Operating the C55x DSP in a socket, which can aggravate noise or glitches on the TCK input.
- Poor signal integrity on the TCK line from reflections or other layout issues.

A TCK edge that can cause this problem might look similar to the one shown in Figure 3. A TCK edge that does not cause the problem will look similar to the one shown in Figure 4. The key difference between the two figures is that Figure 4 has a clean and sharp transition whereas Figure 3 has a “knee” in the transition zone. Problematic TCK signals may not have a knee that is as pronounced as the one in Figure 3. Due to the TCK signal amplification inside the chip, any perturbation of the signal can create erroneous clock edges.

As a result of the faster edge transition, there is increased ringing in Figure 4. As long as the ringing does not cross logic input thresholds (0.6 V for falling edges, and 2.5 V for rising edges), this ringing is acceptable.

When examining a TCK signal for this issue, either in board simulation or on an actual board, it is very important to probe the TCK line as close to the DSP input pin as possible. In simulation, it should not be difficult to probe right at the DSP input. For most physical boards, this means using the via for the TCK pad on the back side of the board. Similarly, ground for the probe should come from one of the nearby ground pad vias to minimize EMI noise picked up by the probe.

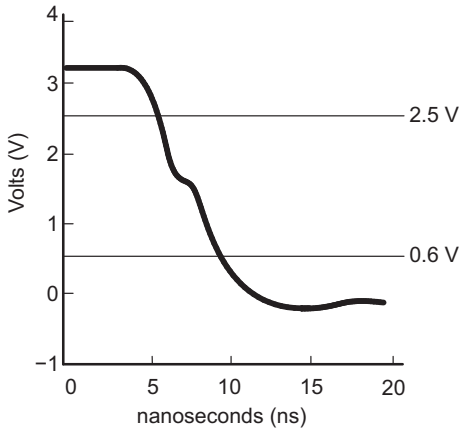


Figure 3. Bad TCK Transition

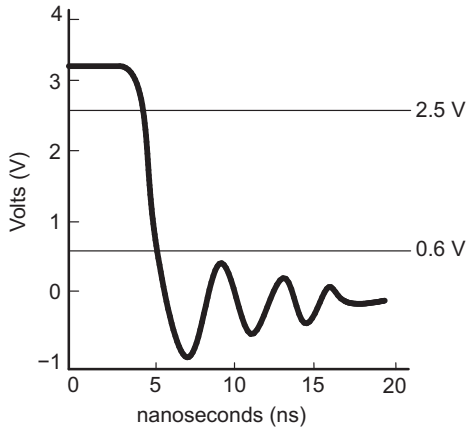


Figure 4. Good TCK Transition

Workaround(s)

As the problem may be caused by one or more of the above factors, one or more of the steps outlined below may be necessary to fix it:

- Avoid using a socket
- Ensure the board design achieves rise times and fall times of less than 3 ns with clean monotonic edges for the TCK signal.
- For designs where TCK is supplied by the emulation pod, use a C55x Emulation Adapter Board, part number DSP8102U. To order a C55x Emulation Adapter Board, please contact the TI Product Information Center (PIC).

Advisory PM_1 *Repeated Interrupts During CPU in Idle*

Revision(s) Affected 1.0 and 1.1

Details Any external interrupt staying low for an extended period should generate only one interrupt. The interrupt signal should normally be required to go high, then low again before additional interrupts would be generated. However, on the 5509A, when an external interrupt signal goes low and stays low while the CPU domain is in idle state, multiple interrupts will be generated. If the corresponding interrupt source is enabled, the CPU exits the idle state and the interrupt service routine will be executed and the interrupts will causes the CPU to exit the idle state.

Assembler Notification None

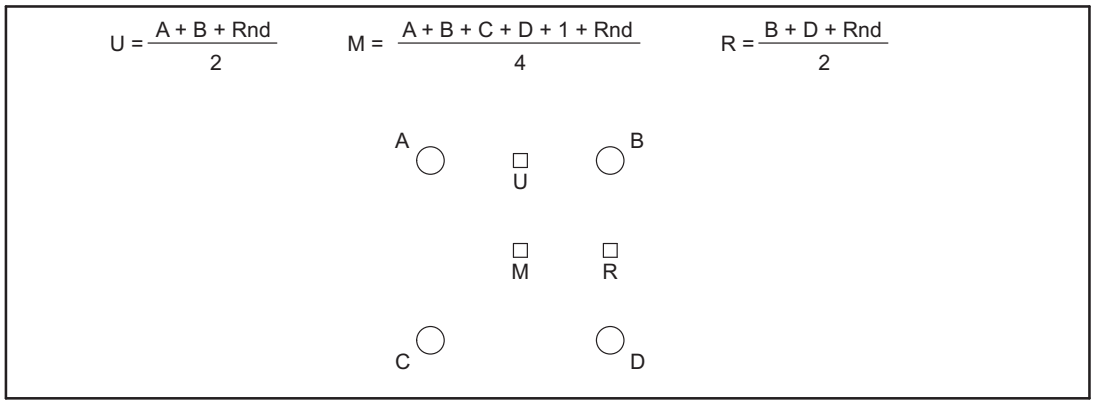
Workaround(s) To avoid multiple interrupts, limit the low pulse durations of external interrupts. When CPU is in idle but CLKGEN is active, the low pulse duration = ID2. When both CPU and CLKGEN are in idle, the low pulse duration = ID1 + ID2. Please see the *TMS320VC5509A Fixed-Point Digital Signal Processor* data manual (Literature Number SPRS205), *Wake-Up From IDLE* section for the details of ID1 and ID2.

Advisory HWA_1 *Pixel Interpolation Hardware Accelerator*

Revision(s) Affected 1.0 and 1.1

Details The pixel interpolation computation is wrong by one pixel unit in “Decoder” mode, when “Rounding Mode” is set to zero and when half-pixel interpolation is performed in the middle of four full-resolution pixels (i.e., in the middle of two rows of pixels).

In “Decoder” mode, the Pixel Interpolator data path is configured to deliver two interpolated pixels per cycle. Each section of the data path implements the computations shown below, normalized according to video decoding standards:



U and R results are interpolated from lines and columns, respectively. M is computed from two following full-resolution pixels lines. Two U, R, or M results are computed during each cycle by the data path, according to the instruction being executed by the accelerator for the M results set (the results are denoted as M0 and M1). When “Rounding Mode” (Rnd in the above picture) is set to 1, the faulty data path section is implemented, M0 = (A+B+C+D+2)/4, which is the correct result; but when “Rounding Mode” is set to 0, the data path section implements M0=(A+B+C+D)/4. This is not correct and generates a bit exactness issue. The M1 result is always correct, regardless of the “Rounding Mode” state.

Assembler Notification None

Workaround(s) Do not use the PI data path in the “Decoder” software when computing M type points and when software-rounding is needed. Instead, use a routine consisting of regular C55x instruction combinations. This routine must be called from the point where the one using PI HWA instructions is and should take the same parameters and data organization. Hence, the new routine should perform following steps:

- Unpack the pixels in the CPU
- Diagonal interpolation

The routine below describes the M points computation (diagonal interpolation) for a full block:

Presetting:
DR1: is set with the block size (8 or 16 pixels)
AR2: is pointing to the first element in the block
Code example:
Begin:

```
DR2 = DR1 + #2                          ; DR2 = blk_size+2
AR4 = AR2                               ; AR4 -> block[0][0]
AR5 = AR2
DR3 = DR1 - #1                          ; DR3 = block_size - 1
BRC0 = DR3                              ; BRC0 = block_size - 1
AR5 = AR5 + DR2                         ; AR5 -> block[1][0]
DR0 = #2                                ; DR0 = 2
AC3 = DR0 - @rounding_control           ; AC3 = 2-rounding_control
|| DR1 = DR1 >> #1                      ; DR1 = block_size/2
@rnd_temp = AC3                         ; rnd_temp = 2-rounding_control
|| DR1 = DR1 - #1                       ; DR1 = block_size/2 - 1
BRC1 = DR1                              ; BRC1 = block_size/2 - 1
XAR3 = XDP
AR3 = AR3 + #rnd_temp                   ; AR3 -> rnd_temp

|| localrepeat {                       ; repeat blk_size times

AC0 = (*AR4+ <<  #16) + (*AR5+ << #16)       ; AC0 = b[i][j]+b[i+1][j]
AC0 = AC0 + (*AR3 << #16)               ; AC0 = AC0 + rnd

|| localrepeat {
AC1 = (*AR4- << #16) + (*AR5+ << #16)        ; AC0 = b[i][j+1]+b[i+1][j+1]
AC0 = AC0 + AC1                         ; AC0 = sum(b[i][j]) + rnd

*(AR4+DR0) = HI(AC0 << #(-2))           ; b[i][j] = (sum(b[i][j])+rnd)/4
|| AC1 = AC1 + (*AR3 <<  #16)                ; AC1 = AC1 + rnd

AC0 = (*AR4- << #16) + (*AR5+ <<  #16)       ; AC0 = b[i][j+2]+b[i+1][j+2]
AC1 = AC1 + AC0                              ; AC1 = sum(b[i][j]) + rnd
*(AR4+DR0) = HI(AC1 <<  #(-2))          ; b[i][j+1] = (sum(b[i][j])+rnd)/4
|| AC0 = AC0 + (*AR3 << #16)                ; AC0 = AC0 + rnd
}

mar (*AR4+) || mar (*AR5+)

}
```

Revision History

This silicon errata revision history highlights the technical changes made to the SPRZ200E revision to make it an SPRZ308F revision.

Scope: Applicable updates relating to the TMS320VC5509A devices have been incorporated.

Revision History

SEE	ADDITIONS/MODIFICATIONS/DELETIONS
Section 2 Silicon Revision 1.0 and 1.1 Usage Notes and Known Design Exceptions to Functional Specifications	Added note • Section 2.1.2, *Attempting to Idle CPU Domain with INTx Asserted.*
Section 2.2 Silicon Revision 1.0 and 1.1 Known Design Exceptions to Functional Specifications	Updated/Changed advisory • Advisory PM_1, *Repeated Interrupts During CPU in Idle*

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

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 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. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

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.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Communications and Telecom	www.ti.com/communications
DSP	dsp.ti.com	Computers and Peripherals	www.ti.com/computers
Clocks and Timers	www.ti.com/clocks	Consumer Electronics	www.ti.com/consumer-apps
Interface	interface.ti.com	Energy	www.ti.com/energy
Logic	logic.ti.com	Industrial	www.ti.com/industrial
Power Mgmt	power.ti.com	Medical	www.ti.com/medical
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Space, Avionics & Defense	www.ti.com/space-avionics-defense
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video and Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless-apps

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