



Errata: EP9301 - Silicon revision: E0

Reference EP9301 Data Sheet revision DS636PP5 dated March 2005.

Determining the Silicon Revision of the Integrated Circuit

On the front of the integrated circuit, directly under the part number, is an alpha-numeric line. Characters 5 and 6 in this line represent the silicon revision of the chip. For example, this line indicates that the chip is a "E0" revision chip:

EFWAE0AM0340

This Errata is applicable only to the E0 revision of the chip.

Please refer to AN273, "EP93xx Rev E0 Design Guidelines" for additional information.

AC'97

Description

Disabling audio transmit by clearing the TEN bit in one of the AC97TXCRx registers will not clear out any remaining bytes in the TX FIFO. If the number of bytes left in the FIFO is not equal to a whole sample or samples, this will throw off subsequent audio playback causing distortion or channel swapping.

Workaround

To stop audio playback, do the following:

- 1) Pause DMA
- 2) Poll the AC97SRx register until either TXUE or TXFE is set.
- 3) Clear the TEN bit.

This ensures that the TX FIFO is empty before the transmit channel is disabled.

Ethernet

Description 1

The Ethernet controller does not correctly receive frames that have a size of 64 bytes.

Workaround

In order to receive frames of 64 bytes, enable the RCRCA bit in RxCTL. This will prevent the Ethernet controller from discarding the 64-byte-long frames.

Description 2

When there is inadequate AHB bus bandwidth for data to be transferred from the Ethernet controller FIFO to the receive descriptor, the Ethernet FIFO will overflow and cause the Ethernet controller to fail to receive any more packets.

This problem will also occur if the processor is too busy to service incoming packets in a timely manner. By the time that new receive descriptors are available, the data in the FIFO will contain frames that are corrupted.

It is the job of the system designer to ensure that there is adequate bandwidth for the applications being run.

Workaround

This is a rare occurrence, however at a system level it is important to reserve adequate bandwidth for the Ethernet controller. This can be accomplished by some of the following:

- Reducing the bandwidth use of other bus masters in the system.
- Lowering Ethernet rate to half duplex or 10Mbit if higher bandwidth is not required.
- Insuring that the Ethernet controller receive descriptor processing is given a high enough priority to ensure that the controller never runs out of receive descriptors.

HDLC

Description

When the final byte of a received packet is read into the DMA controller's buffer, the software will be notified by an HDLC RFC interrupt. However, the DMA controller may not have written the currently buffered part of the packet to memory, so that the last one to fifteen bytes of a packet may not be accessible.

Workaround

To insure that the DMA channel empties the buffer, do the following (in the HDLC interrupt handler, for example):

- 1) Note the values in the MAXCNTx and REMAIN registers for the DMA channel. The difference is the number of bytes read from the UART/HDLC, which is the size of the HDLC packet. Call this number N. Note that the BC field of the UART1HDLCRXInfoBuf register should also be N.
- 2) Temporarily disable the UART DMA RX interface by clearing the RXDMAE bit in the UART1DMACtrl register.
- 3) Wait until the difference between the CURRENTx and BASEx registers in the DMA channel is equal to N + 1.

At this point, the rest of the packet is guaranteed to have been written to memory. Using this method will cause an extra byte to be read from the UART by the DMA channel and also written to memory. This last byte should be ignored.

SDRAM Controller

Description 1

Using the SDRAM controller in auto-precharge mode will produce system instability at external bus speeds greater than 50MHz.

Workaround

Do not turn on the auto-precharge feature of the SDRAM controller if the external bus speed will be greater than 50 MHz.

Description 2

When the SDRAM controller is configured for PRECHARGE ALL command, the actual sequence is not always issued to the SDRAM device(s).

Workaround

Do a read from each SDRAM bank so that a PRECHARGE command is issued to each bank of the SDRAM device. This will satisfy the required SDRAM initialization sequence.

Due to the effectiveness and simplicity of the software workaround, no silicon fix is planned.

Reset

Description 1

No SDCLK after system reset. This occurs when switching from the crystal clock to the PLL clock upon a system reset. When this condition occurs the processor will not start its boot sequence.

Workaround

Implement the recommended external reset circuit described in AN258, “EP93xx Power-up and Reset Luckup Workaround”, which can be found at <http://www.cirrus.com/en/pubs/appNote/AN258REV2.pdf>

A fix for this bug has been implemented for silicon revision E1.

Description 2

SDRAM controller gets stuck in a busy state after resetting from sync mode. When this condition occurs the processor will not complete its boot sequence.

Workaround

Implement the recommended external reset circuit described in AN258, “EP93xx Power-up and Reset Luckup Workaround”, which can be found at <http://www.cirrus.com/en/pubs/appNote/AN258REV2.pdf>

A fix for this bug has been implemented for silicon revision E1.

Description 3

Double reset after a software reset is issued. This condition occurs when the state of $\overline{CS1}$ is intended to be latched as a logic 0 during reset. However, because of a slow slew rate, they are latched as a logic 1 before they have a chance to reach their final value. When this happens, the Watchdog reset, which is based on a counter, is directly set and generates another reset.

Workaround

There is no work around at this time.

A fix for this bug has been implemented for silicon revision E1.

Description 4

The internal RTC oscillator is susceptible to noise which can lead to extra clocks on the internal 32.768-kHz signal.

Workaround

Please refer to application note AN265, “EP93xx RTC Oscillator Circuit”, which can be found at <http://www.cirrus.com/en/pubs/appNote/AN265REV1.pdf>

No fix of this bug is planned for future silicon revisions.

Software Reset

Description

The EP93xx device may hang during the boot process after a software or watchdog reset has been issued.

Workaround

Implement the recommended external reset circuit described in AN258, which can be found at <http://www.cirrus.com/en/pubs/appNote/AN258REV2.pdf>

A fix for this bug has been implemented for silicon revision E1.

USB and DMA Arbitration

Description

The EP93xx device can hang during AHB arbitration if the USB controller and the DMA controller are used at the same time. This issue has been observed only when the EP93xx is configured to use 16-bit memory boot mode. It has not been observed when the device is configured for 32-bit memory boot mode.

Workaround

Either do not use the DMA controller or do not use the USB controller. Items potentially affected by not using the DMA controller are IDE, audio, SSP, UART, IrDA, and external DMA. Most of these items can be used in PIO mode with little performance impact. Refer to the User's Guide for details about DMA controller operation.

A fix for this bug has been implemented for silicon revision E1.

I²S Audio

Description

I²S slave mode operation does not function as expected when I2SonAC97 bit (DeviceCfg[6]) is set in the DeviceCfg register causing the ASYNC pin to be driven as LRCLK input.

Workaround

Use I2SonSSP (DeviceCfg[7]) mode instead of I2SonAC97 (DeviceCfg[6]).

A fix for this bug has been implemented for silicon revision E1.

Contacting Cirrus Logic Support

For all product questions and inquiries contact a Cirrus Logic Sales Representative.
To find one nearest you go to www.cirrus.com

IMPORTANT NOTICE

"Preliminary" product information describes products that are in production, but for which full characterization data is not yet available.

Cirrus Logic, Inc. and its subsidiaries ("Cirrus") believe that the information contained in this document is accurate and reliable. However, the information is subject to change without notice and is provided "AS IS" without warranty of any kind (express or implied). Customers are advised to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgment, including those pertaining to warranty, indemnification, and limitation of liability. No responsibility is assumed by Cirrus for the use of this information, including use of this information as the basis for manufacture or sale of any items, or for infringement of patents or other rights of third parties. This document is the property of Cirrus and by furnishing this information, Cirrus grants no license, express or implied under any patents, mask work rights, copyrights, trademarks, trade secrets or other intellectual property rights. Cirrus owns the copyrights associated with the information contained herein and gives consent for copies to be made of the information only for use within your organization with respect to Cirrus integrated circuits or other products of Cirrus. This consent does not extend to other copying such as copying for general distribution, advertising or promotional purposes, or for creating any work for resale.

CERTAIN APPLICATIONS USING SEMICONDUCTOR PRODUCTS MAY INVOLVE POTENTIAL RISKS OF DEATH, PERSONAL INJURY, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE ("CRITICAL APPLICATIONS"). CIRRUS PRODUCTS ARE NOT DESIGNED, AUTHORIZED OR WARRANTED FOR USE IN AIRCRAFT SYSTEMS, MILITARY APPLICATIONS, PRODUCTS SURGICALLY IMPLANTED INTO THE BODY, AUTOMOTIVE SAFETY OR SECURITY DEVICES, LIFE SUPPORT PRODUCTS OR OTHER CRITICAL APPLICATIONS. INCLUSION OF CIRRUS PRODUCTS IN SUCH APPLICATIONS IS UNDERSTOOD TO BE FULLY AT THE CUSTOMER'S RISK AND CIRRUS DISCLAIMS AND MAKES NO WARRANTY, EXPRESS, STATUTORY OR IMPLIED, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR PARTICULAR PURPOSE, WITH REGARD TO ANY CIRRUS PRODUCT THAT IS USED IN SUCH A MANNER. IF THE CUSTOMER OR CUSTOMER'S CUSTOMER USES OR PERMITS THE USE OF CIRRUS PRODUCTS IN CRITICAL APPLICATIONS, CUSTOMER AGREES, BY SUCH USE, TO FULLY INDEMNIFY CIRRUS, ITS OFFICERS, DIRECTORS, EMPLOYEES, DISTRIBUTORS AND OTHER AGENTS FROM ANY AND ALL LIABILITY, INCLUDING ATTORNEYS' FEES AND COSTS, THAT MAY RESULT FROM OR ARISE IN CONNECTION WITH THESE USES.

Cirrus Logic, Cirrus, the Cirrus Logic logo designs, and MaverickCrunch are trademarks of Cirrus Logic, Inc. All other brand and product names in this document may be trademarks or service marks of their respective owners.

Microsoft and Windows are registered trademarks of Microsoft Corporation.

Microwire is a trademark of National Semiconductor Corp. National Semiconductor is a registered trademark of National Semiconductor Corp.

Texas Instruments is a registered trademark of Texas Instruments, Inc.

Motorola is a registered trademark of Motorola, Inc.

LINUX is a registered trademark of Linus Torvalds.