ER0023
Errata
SmartFusion Devices

Microsemi.

Power Matters.™

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

© 2016 Microsemi Corporation. All rights reserved. Microsemi and the Microsemi logo are trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.

Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer's responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided “as is, where is” and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, California, and has approximately 4,800 employees globally. Learn more at www.microsemi.com.

55800023. 1.3 11/16

Contents

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 1.3

Errata item 25, While programing the SmartFusion device in IAP mode, with external clock source driving the PLL to clock the MSS, the device may hang at the end of IAP operation, page 17 is added in the revision 1.3 of this document.

2 Errata for SmartFusion Devices

2.1 Revisions Released per Device

The following table describes the silicon revisions released per device.

Table 1 • **Silicon Revisions Released per Device**

Silicon Devices	Revisions
A2F060M3E	A
A2F200M3F	A, B, C, D
A2F500M3G	A, B

2.2 Introduction

This errata sheet contains updated information about any known SmartFusion® device family specific issues and provides available fixes and solutions.

2.3 Summary of SmartFusion Devices Errata

The following table gives the summary of SmartFusion devices errata items.

Table 2 • **Summary of SmartFusion Devices Errata**

No.	Issues	Affected Devices/ Software/ Revisions	Fixed in Device/Software/ Revisions
1.	Block Transfer (Burst mode) in Slave SPI mode is not supported for the A2F200 device, page 5	A2F200 Silicon issue	No workaround
2.	WDOGTIMEOUTEVENT is asserted incorrectly if the Watchdog is configured “Reset” when timeout occurs, page 5	A2F200Silicon issue	No workaround
3.	WDOGLOAD register is loaded with the reset value after Watchdog timeout reset, page 6	A2F200Silicon issue	No workaround
4.	FAB_CLK, configured as FCLK/2 or FCLK/4, is out of phase with GLC when both GLC and FAB_CLK are configured to be used as FPGA fabric clocks, page 6	All SmartFusion devices Silicon issue	No workaround
5.	ACE reset delay reduced from 50 ms to 3 ms after the occurrence of Watchdog timeout to address potential lock-up on reset, page 6	A2F500 Rev A/B All date codes prior to 1101 Silicon issue	Software mitigation–Libero® software v9.1 SPB and later with MSS v2.4.105 Fixed in Rev A/B Date code1102 and higher
6.	Software incorrectly computes flag (OVER_FLAG/UNDER_FLAG) trip levels, page 7	All SmartFusion devices Software issue	Libero software v9.1 SP2 with MSS version 2.5.106 in conjunction with MSS_ACE_Driver version 2.3.105
7.	Libero software incorrectly enables SSE and PPE prematurely, resulting in incorrect analog samples in the first few sampling iterations, page 7	All SmartFusion devices Software issue	Libero software release v9.1 SP2 with MSS version 2.5.106

Table 2 • Summary of SmartFusion Devices Errata

No.	Issues	Affected Devices/ Software/ Revisions	Fixed in Device/Software/ Revisions
8.	Channels that need strobing, such as CM4 and TM4, show up under ADC0 or ADC1 (ACE Controller tab in software) instead of ADC2, page 8	A2F500 Silicon issue	Software mitigation—Libero software v9.1 and later release. MSS 2.4.105 and later
9.	If the Watchdog timer is reset during the forbidden window, the system boot incorrectly refreshes the Watchdog counter after a reset, leading to an infinite loop during the boot sequence, page 9	All SmartFusion devices Software issue	Libero software v9.0 SP1 with MSS version 2.2.101 and later release
10.	After power-up, the device may lock-up until a Watchdog reset happens, page 10	A2F200 Rev A/B/C Date code 1101 and prior Silicon issue	A2F200 Rev C/D Date code 1013 and higher
		A2F500 Rev A Date code 1027 and prior Silicon issue	A2F500 Rev B Date code 1028 and higher
11.	MSS_RESET_N stuck low upon Watchdog generated system reset; as a result, the device may lock-up, page 11	All SmartFusion devices Software issue	Libero software v9.1 SPB with MSS version 2.4.105 and later releases
12.	Endless reset loop when using external reset chip, page 11	All SmartFusion devices	Ensure that the debounce delay setting in the Reset Management configurator matches reset delay of the external reset chip
13.	In Libero software, MSS CCC configurator delays do not match actual delays on GLC and GLB clock networks, page 12	All SmartFusion devices Software issue	Libero software v9.1 SP2 and later releases with MSS v2.5.106
14.	No option in the MSS CCC configurator to route the 32 KHz low-power OSC source into the FPGA fabric, page 12	All SmartFusion devices Software issue	Firmware fix
15.	Device locks up when using the function GetSystemClock in CMSIS system initialization, page 13	A2F060 Software issue	Libero software v9.0 SP3 and later release with MSS version 2.2.101 and later release In conjunction with CMSIS_PAL version 2.2.100 and later release
16.	System boot incorrectly resets when no application is running on the Cortex-M3 processor, page 13	All SmartFusion devices Software issue	Libero software v9.1 SP2 and later release with MSS version 2.5.106 and later release
17.	Fabric I/Os state cannot be set through BSR using IAP block, page 13	All SmartFusion devices	Setting fabric I/O state is not supported through IAP
18.	I2C SCL (serial clock) frequency on board is lower than expected, page 14	All SmartFusion devices	Expected behavior—refer to the workaround
19.	Programming SmartFusion FlashROM and/or security using IAP flow is not supported, page 15	A2F200 A2F500	–

Table 2 • **Summary of SmartFusion Devices Errata**

No.	Issues	Affected Devices/ Software/ Revisions	Fixed in Device/Software/ Revisions
20.	Serial Wire Viewer (SWV) does not work at full speed, page 15	A2F200 Rev A Date code 948 and prior Silicon issue	A2F200 Rev B/C/D Date code 950 and higher
21.	Post Processing Engine (PPE) microcode is incorrectly generated when “Send filtered value to DMA” is enabled in ACE channel configuration, page 15	All SmartFusion Devices	Libero SoC release v10 SP2
22.	The ACE_convert_to_mA() function in ace_convert.c computes Amps instead of mA, page 16	All SmartFusion devices Firmware Issue	Modify ACE_convert_to_mA() function
23.	Signal driving fabric CCC/PLL glitchless mux (NGMUX) must be a free running clock signal, page 16	A2F500	–
24.	Cold Sparing feature for ABPS signals, page 17	All SmartFusion devices	–
25.	While programing the SmartFusion device in IAP mode, with external clock source driving the PLL to clock the MSS, the device may hang at the end of IAP operation, page 17	All SmartFusion devices	MSS IAP Driver version 2.3

Note: “–” indicates that the errata does not exist or the feature does not exist for that particular device and revision number.

Note: Contact *Microsemi Global support* if you have additional questions. To order a specific die, contact your local Microsemi sales office

3 Errata Descriptions and Solutions

3.1 Block Transfer (Burst mode) in Slave SPI mode is not supported for the A2F200 device

In Slave SPI mode, transferring the data in burst mode is not supported for A2F200 device. The following table summarizes the supported transfer modes for different protocols and modes of operation of the SPI controller. These limitations apply only to the A2F200 device.

Table 3 • **SPI Data Transfer Support**

	Master Mode			**Slave Mode**		
	Motorola	**NSC**	**TI**	**Motorola**	**NSC**	**TI**
Block Transfer (Burst mode)	Yes[1]	Yes[1]	Yes[1]	No	No	No
Frame Transfer	Yes	Yes	Yes	Yes	No	Yes

1. When performing block transfers using the A2F200 SmartFusion device, it is recommended that you use a general purpose I/O (GPIO) to control the target slave device's chip select input. This ensures reliable operations by preventing the chip select signal from becoming deasserted before the transfer completes, as a result of the SPI transmit FIFO becoming empty. This issue may occur when a high priority interrupt is serviced in some other part of the system while the SPI transfer is taking place. This potential problem only exists for block transfers on the A2F200 SmartFusion device.

Solution

No workaround for the options that are shown as not supported in the preceding table.

3.2 WDOGTIMEOUTEVENT is asserted incorrectly if the Watchdog is configured "Reset" when timeout occurs

The WDOGTIMEOUTEVENT is a bit in the microcontroller subsystem (MSS) status register (MSS_SR). When asserted, this bit indicates that the Watchdog has timed out. The Watchdog can only be configured to generate an Interrupt or a reset when timeout occurs.

The intent of this bit is that it should be asserted (and remain asserted through the system reset) if the Watchdog generates a Watchdog timeout reset (such as, not an Interrupt).

The issue is that the WDOGTIMEOUTEVENT bit in SYSREG is asserted when the Watchdog timer is configured to generate an Interrupt on timeout, rather than when configured to generate a reset.

Therefore, in the A2F200 device, if a Watchdog reset occurs, the WDOGTIMEOUTEVENT (in SYSREG) bit is not asserted. This means that when the MSS boots up after the Watchdog reset, the firmware has no way of knowing that the reset was due to a Watchdog timeout.

Solution

No workaround if the Watchdog is configured to generate reset on timeout.

3.3 WDOGLOAD register is loaded with the reset value after Watchdog timeout reset

The WDOGLOAD register is used to store the value which is loaded into the counter each time the Watchdog is refreshed or when the counter value reaches zero. The default Watchdog delay is 5 seconds. When a value other than the default is written to WDOGLOAD and a system reset occurs due to a Watchdog reset, WDOGLOAD is reloaded with the default delay instead of what you specified.

Solution

No workaround.

3.4 FAB_CLK, configured as FCLK/2 or FCLK/4, is out of phase with GLC when both GLC and FAB_CLK are configured to be used as FPGA fabric clocks

When both GLC and FAB_CLK are configured to be used by the FPGA fabric, for some configurations FAB_CLK would be out of phase with GLC. See the following Known Issue (KI) link for more details:

http://www.microsemi.com/soc/kb/article.aspx?id=KI8852

3.5 ACE reset delay reduced from 50 ms to 3 ms after the occurrence of Watchdog timeout to address potential lock-up on reset

The reset-to-application time was 50 ms in the event of system reset due to Watchdog reset trigger. This causes a potential lock-up on reset.

Solution

Update to Libero software v9.1 SPB or later release along with MSS version 2.4.105 or later release.

The system boot code was updated to reduce the reset-to-application time on a Watchdog reset from 50 ms to 3 ms. The process is as follows:

1. In the event of a system reset, the system boot code first checks the device to ascertain that the device is A2F500.
2. If the device is A2F500:

The system boot checks for an IDcode in the eNVM.

If the IDcode exists:

- Then it looks at the SYSREG status register to detect whether a Watchdog triggered the system reset.
  - If true, this causes the system boot code to speed up the boot process by releasing the ACE reset as soon as possible and then branching off to the application code.
  - If false, then it applies any optimized boot-up process.

If the IDcode does not exist:

- The boot-up process is used with a 50 ms delay. In this case there is the risk of lock-up on reset.

3. If the device is A2F200, then it applies any optimized boot-up process.

3.6 Software incorrectly computes flag (OVER_FLAG/UNDER_FLAG) trip levels

This issue is found in the MSS_ACE_Driver prior to version 2.3.105 and MSS version prior to 2.5.106.

Following are the two causes-which lead to this issue:

- The ACE driver conversion functions ACE_convert_to_mV() and ACE_convert_from_mV() return incorrect values for ABPS inputs when an external VAREF other than 2.45 V is used. This results in the voltage reported by the application executing on the ARM Cortex-M3 being different from the actual voltage being sampled.
- The ACE configurator computes incorrect threshold values for the flags when an external VAREF other than 2.45 V is used. This is because the ACE configurator assumes that the ABPS inputs also use the external VAREF as the reference voltage for the prescaler.

Note: The ABPS is connected to the internal VAREF and not the EXTVAREF.

Solution

Update to Libero software v9.1 SP2 or later release along with MSS version 2.5.106 in conjunction with the MSS_ACE_Driver version 2.3.105.

3.7 Libero software incorrectly enables SSE and PPE prematurely, resulting in incorrect analog samples in the first few sampling iterations

As soon as the ACE init block is programmed, the SSE and PPE are being enabled, which happens before the PPE calibration data is merged and before VAREF has stabilized, as shown in the following snippet:

```
#ACE_INIT2 = 0x40020000

#  SSE_TS_CTRL:0x4

#    SSE_SRAM_ENABLE[1:1] = 0x1

#    TS_ENABLE[0:0] = 0x1  >>ENABLE bit is set

0x3

#  ANA_COMM_CTRL:0xc

#    ABPOWERON[3:3] = 0x1

#    ACB_RESETN[4:4] = 0x0

#    ADCRESET[1:1] = 0x1

#    PWRDWN[2:2] = 0x0

#    VREFSEL[0:0] = 0x0

0xa

#  PPE_CTRL:0x1404

#    PPE_EN[0:0] = 0x1

#    PPE_RRDIS0[14:14] = 0x0

#    PPE_RRDIS1[15:15] = 0x0

#    PPE_RRDIS2[16:16] = 0x0

0x1
```

As a result, incorrect analog samples may be obtained in the first (few) sampling iterations.

Solution

Update to Libero software v9.1 SP2 or later and MSS version 2.5.106 or later.

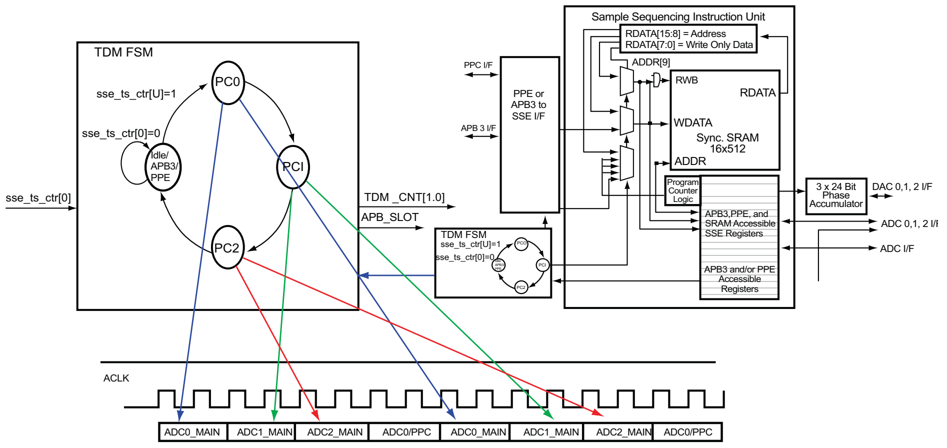
Note that you have the option to initiate the ACE sampling from the application code or have the system boot automatically enable the SSE.

You can initiate the ACE sampling from the application code by checking the option **Initiate ACE sampling from application code** in the **Advance Options** GUI in the ACE configurator. By default this option is disabled.

3.8 Channels that need strobing, such as CM4 and TM4, show up under ADC0 or ADC1 (ACE Controller tab in software) instead of ADC2

The TM4 and CM4 are pads that connect directly to the ADC2 (available only on the A2F500 device). Part of the ACE controller is a Time Division Multiplexing (TDM) FSM, which implements a mechanism to sequence through the different ADCs available on the SmartFusion devices. The FSM allows equal access through three separate program counters. program counter0 (PC0) is associated with ADC0. program counter1 (PC1) is associated withADC1 and program counter2 (PC2) is associated with ADC2, as shown in the following figure.

Figure 1 • **SSE Micro Architecture**



Solution

The strobe channels on ADC2 (current, temperature, differential voltage) are instead made available through the ADC Controller tab in software on the procedures that are targeted for ADC0 or ADC1 (instead of ADC2 where they are supposed to be available). This is a software workaround. Even though the ADC2 analog signal moved under the PC0 and/or PC1 time slot, physically those analog signals are still connected to ADC2 and are sampled on the ADC2. You should be aware that there will be a contention when sampling is performed on one ADC and then simultaneous samples are performed on the other procedure using the same ADC. The likely outcome of such a scenario is that the samples are dropped. Furthermore, the ACE controller issues a warning, as shown in the following figure.

For example:

ADC0_MAIN procedure samples „³ TM4 (TM4 is physically connected on ADC2)

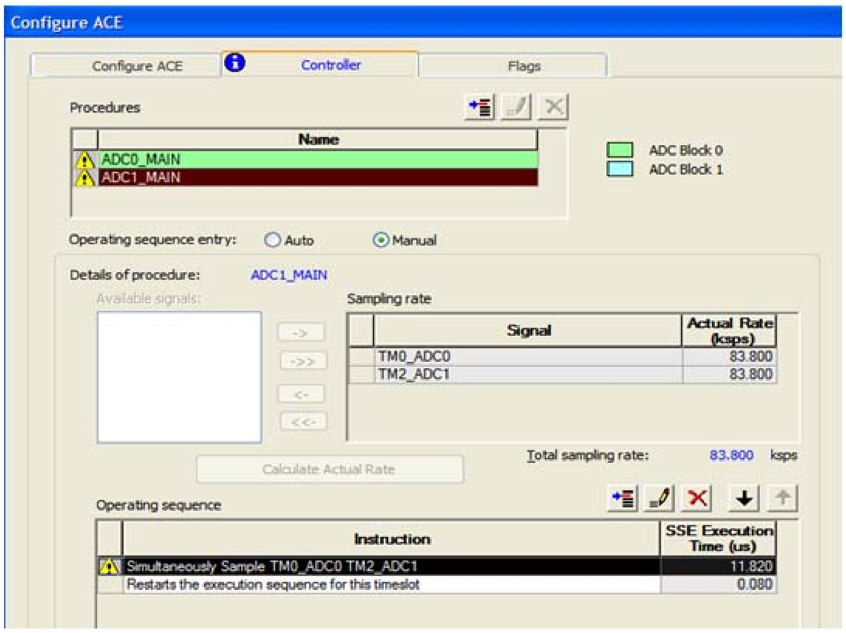
ADC2_MAIN procedure samples „³ Voltage input (or any other input) that is physically on ADC2

Therefore, there are two procedures setup to access the same ADC (ADC2) and both are (by default) active in your application at the same time.

In such a situation there is contention of the ADC resources. This leads to the ADC sampling data being dropped occasionally, as indicated by the warning message shown in the following figure. To fix this issue, you can either change the procedure specification, so you don’t use the same ADC in the multiple procedures or the firmware must enable/disable the procedures so that the conflicting procedures are not simultaneously active at the same time.

The ACE driver Use Guide has an API that you can use to enable/disable the procedures.

Figure 2 • **Sampling Simultaneous Operations**



Procedures

	Name
⚠	ADC0_MAIN
⚠	ADC1_MAIN

🟩 ADC Block 0
🟦 ADC Block 1

Operating sequence entry: ○ Auto ● Manual

Details of procedure: ADC1_MAIN

Available signals:

Sampling rate

	Signal	Actual Rate (ksps)
	TM0_ADC0	83.800
	TM2_ADC1	83.800

Total sampling rate: 83.800 ksps

Operating sequence

	Instruction	SSE Execution Time (us)
⚠	Simultaneously Sample TM0_ADC0 TM2_ADC1	11.820
	Restarts the execution sequence for this timeslot	0.080

Note: Sampling of TM0_ADC0 conflicts with simultaneous operation of procedure(s) for ADC0. See, the ACE Configuration user guide for more information.

3.9 If the Watchdog timer is reset during the forbidden window, the system boot incorrectly refreshes the Watchdog counter after a reset, leading to an infinite loop during the boot sequence

The main issue here is that the system boot is refreshing the Watchdog after the reset is issued. The Watchdog does not enter into the permitted window by the time the refresh is happening in the system boot code. The Watchdog stays in the forbidden window and keeps on resetting the system.

Background

The Watchdog can be configured to generate a reset or interrupt if counter timeout occurs. This is done by setting the MODE control bit in the WDOGCONTROL register.

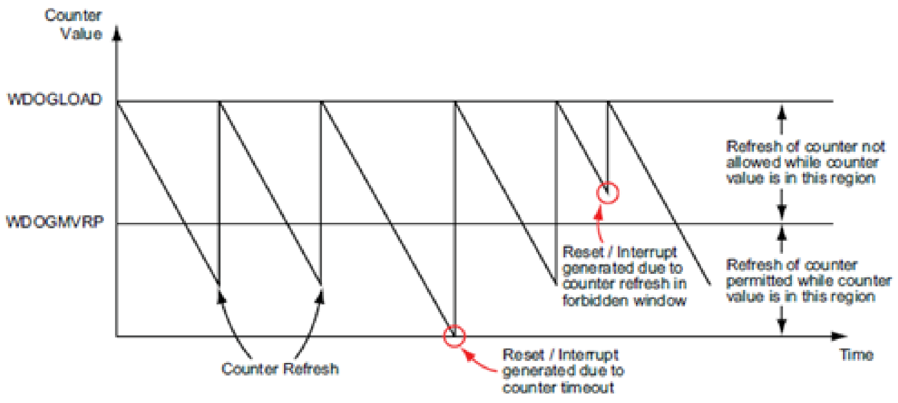
The Watchdog counter is refreshed by writing the value 0xAC15DE42 to the WDOGREFRESH register. This causes the counter to be loaded with the value in the WDOGLOAD register. An appropriate value must be written to the WDOGLOAD register before writing to the WDOGREFRESH register.

Forbidden and permitted windows in time regulate when refresh can occur. The size of these windows is controlled by the value programmed in the WDOGMVRP control register.

When the counter value is greater than the value in WDOGMVRP, the Watchdogrefresh is forbidden. If a refresh is executed in these circumstances, the refresh is successful but a reset or interrupt (depending on the operation mode selected) is also generated. This is illustrated in the following figure.

When the counter value falls below the level programmed in WDOGMVRP, the Watchdogrefresh is permitted. The REFRESHSTATUS status bit in the WDOGSTATUS register is set when in the permitted window.

Figure 3 • **Watchdog Timer Windowing Example**



Solution

Update to Libero software v9.0 SP1 or later release along with MSS version 2.2.101.

The workaround when using an MSS version prior to version 2.2.101 is to avoid using the forbidden window by setting the maximum value to 0xFFFFFFFF, which is also the reset value.

3.10 After power-up, the device may lock-up until a Watchdog reset happens

Upon deassertion of the RESET signal (VCC15_GOOD >1.3 V), the Cortex-M3 processor fetches data from eNVM during the boot-up sequence. Glitches on the VCC15_GOOD signal cause wrong instructions or data to be read by the Cortex-M3 processor during boot-up. The potential outcome of this behavior is as follows:

- Normal, no issue encountered
- Looks normal, that is, the firmware continues to function but wrong data is being fetched
- Cortex-M3 processor freezes until Watchdog timer resets the processor.
- Cortex-M3 processor halts.

The root cause of this problem is the fact that the Cortex-M3 reset (PORESETN) and the eNVM reset (RESETN) are not the same.

Solution

If you are still holding a device that is affected by this issue, the solutions are as follows:

Option#1

Update to the latest device silicon revision. See Table 2, page 2 for more details.

Option#2

Recommended if you do not want to update to a new silicon revision:

- Use the internal voltage regulator.
- Set PUPO = 0, AND PU_N = 1 or floating, AND TRSTB = 0.
- Use the external reset chip (system manager) to turn on PU_N.
  - Recommended Part #1 National Semi LM810, $0.25 @ 1Ku
  - Recommended Part #2 Dallas Semi, DS1819B, $0.72 @1Ku (also used on SmartFusion Development Kit and Evaluation Kit)

Pros:

- Most economical and easy solution

Cons:

- You cannot put SmartFusion device in power-down mode.

- Workaround: Using Part #3, National Semi LM3723, allows turning off the PU_N pin using an extra control.

Option#3

If you do not want to update to a new silicon revision:

- Use external 1.5 V voltage regulator
  - 1.5 V power-up after when 3.3 V reaches 2.7 V
  - 1.5 V power supply ramp rate must be less than 1 ms

Pros:

- No restriction on PUPO, PU_N, TRSTB signals

Cons:

- More expensive solution than the recommended solution
- Restrictive ramp rate for customer

3.11 MSS_RESET_N stuck low upon Watchdog generated system reset; as a result, the device may lock-up

You may encounter this behavior when there is a capacitor on board between the MSS_RESET_N and the GND instead of an on-board reset chip.

The system boot is not taking into account the programmable reset debouncing delay in the case of a Watchdog reset.

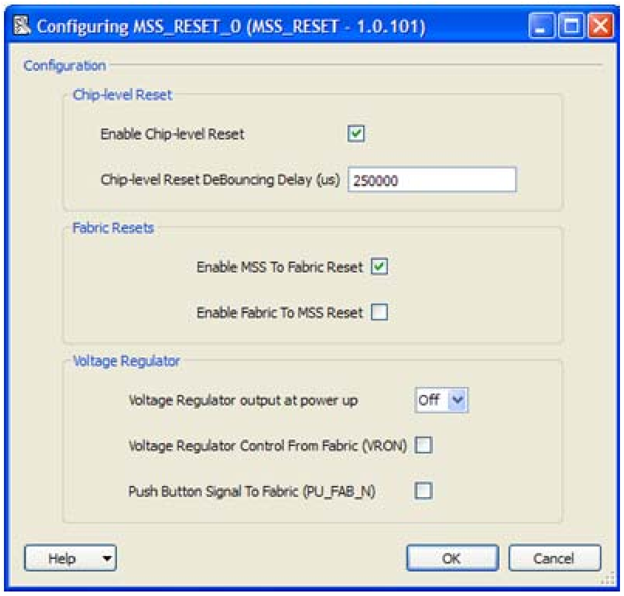
Solution

Update to Libero IDE software v9.1 SPB or later release along with MSS version 2.4.105 or later. Note that there is an RC time constant associated with the capacitor/resistor circuit on MSS_RESET_N line. If you are using capacitors on the MSS_RESET_N line, you are advised to compute the RC time constant on the line, then use that value as the minimum value to be specified as the **Chip-Level Reset Debouncing Delay** in the Reset Management configurator in the MSS, as shown in the following figure.

3.12 Endless reset loop when using external reset chip

This issue happens if the debouncing delay generated by the external reset chip does not match the chip level reset debouncing delay specified in the Reset Management configurator in the MSS, as shown in the following figure.

Figure 4 • **Reset Management Configurator**


Configuring MSS_RESET_0 (MSS_RESET - 1.0.101)

Configuration

Chip-level Reset

Enable Chip-level Reset ✓

Chip-level Reset DeBouncing Delay (us) 250000

Fabric Resets

Enable MSS To Fabric Reset ✓

Enable Fabric To MSS Reset ☐

Voltage Regulator

Voltage Regulator output at power up: Off

Voltage Regulator Control From Fabric (VRON) ☐

Push Button Signal To Fabric (PU_FAB_N) ☐

Solution

Ensure that the chip level reset debouncing delay specified in the Reset Management configurator in the MSS matches the external reset chip delay.

3.13 In Libero software, MSS CCC configurator delays do not match actual delays on GLC and GLB clock networks

In the Libero v9.1 SP1 and SP1A versions of the software, both GLB and GLC outputs of the MSS CCC have more clock network delay than what is reported in the MSS CCC configurator. Furthermore, the GLC output of MSS CCC has ambiguity in the start edge of the clock coming out of reset.

Solution

If you are using v9.1 SPB with MSS v2.4.105, the MSS configurator issues a warning tool-tip related to this issue. See the following link for more details.

http://www.microsemi.com/soc/kb/article.aspx?id=KI8846.

3.14 No option in the MSS CCC configurator to route the 32 KHz low-power OSC source into the FPGA fabric

The MSS CCC configurator does not provide an option to route the 32 KHz low-power OSC source into the FPGA fabric.

Solution

Currently there is no plan to support this in the MSS CCC GUI. As a workaround, if you configure the PLL to put the XTAL_OSC on GLC via the GUI (to establish connectivity in your design); you can then use the following commands in the firmware to modify the PLL settings to source LPOSC to GLC:

```
 /* from mss_rtc.c */
        #define CTRL_STAT_XTAL_EN   0x01u


/* enable the low-power 32k Hz crystal */
RTC->CTRL_STAT_REG |= CTRL_STAT_XTAL_EN;


 /* Switch GLC to low-power crystal */
/* table 8-19 of MSS UG */


 #define RXCSEL 0x20000
#define DYNCSEL 0x40000
SYSREG->MSS_CCC_MUX_CR |= RXCSEL | DYNCSEL;
```

For more details, see the link: http://www.microsemi.com/soc/kb/article.aspx?id=SL5564

3.15 Device locks up when using the function GetSystemClock in CMSIS system initialization

The GetSystemClock function that is in the `system_a2fxxxm3.c` file refers to two address locations (shown below) that are hard-coded and not valid for A2F060 device. The device goes into NMI fault if this function is used (such as in UART initialization).

```
#define SYSBOOT_1_3_FCLK_ADDR   (uint32_t *)0x6008162C
#define SYSBOOT_2_x_FCLK_ADDR   (uint32_t *)0x60081EAC
```

Solution

The FCLK location in eNVM is different for the A2F200/A2F500 compared to A2F060. This fix takes this into account by first reading the device ID from eNVM to find out which eNVM location should be read to get the FCLK value.

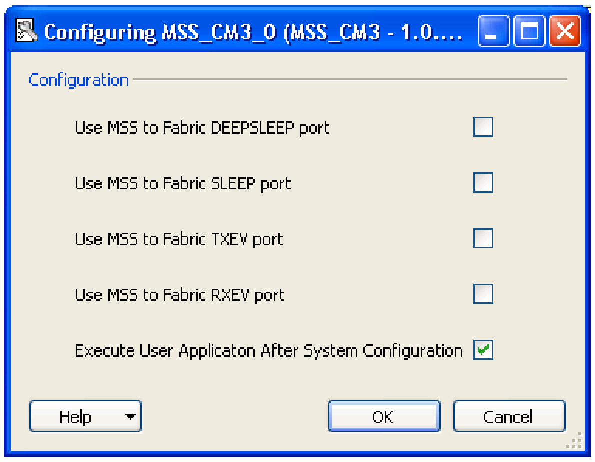
3.16 System boot incorrectly resets when no application is running on the Cortex-M3 processor

When there is no application design, the device periodically resets after system configuration, even though the Watchdog was disabled. The system boot should not go to the application if there is no application specified.

Solution

A new option, **Execute User Application After System Configuration,** has been added into the Cortex-M3 configurator as per the following figure, that gives the user the flexibility of deciding the behavior after the system configuration. The option is ON by default, which indicates that your application is executed after system configuration. If you do not have any application to run, the check box options should be cleared.

Figure 5 • **System Behavior Option after System Configuration**



Configuration

Use MSS to Fabric DEEPSLEEP port ☐

Use MSS to Fabric SLEEP port ☐

Use MSS to Fabric TXEV port ☐

Use MSS to Fabric RXEV port ☐

Execute User Applicaton After System Configuration ✓

3.17 Fabric I/Os state cannot be set through BSR using IAP block

When using the IAP to load BSR to set the state of the I/Os, the I/O state is not set as expected. The IAP cannot access the BSR. The fabric I/Os are always in High-Z state during IAP. As such, the following actions are not possible:

- Hold the fabric I/O at last known state
- Set the fabric I/O to a particular state via BSR

Setting the fabric I/O state is not supported through IAP.

3.18 I2C SCL (serial clock) frequency on board is lower than expected

The serial clock (SCL) is the clock that comes out of the I2C block along with the data (SDA). The SCL rate is defined as the PCLK frequency divided by a factor of 8, 60, 120, 160, 192, 224, 256, or 960. This factor is set by setting the CR0, CR1, and CR2 bits in the CTL register, as given in the following table.

Table 4 • **Serial Clock Frequency Settings**

CR2	CR1	CR0	SCL Frequency
0	0	0	PCLK frequency/256
0	0	1	PCLK frequency/224
0	1	0	PCLK frequency/192
0	1	1	PCLK frequency/160
1	0	0	PCLK frequency/960
1	0	1	PCLK frequency/120
1	1	0	PCLK frequency/60
1	1	1	PCLK frequency/8

Note: BCLK is synchronized to PCLK and hence must be PCLKFREQ/2 or less

The issue is that the SCL clock measured on the silicon/board does not match the expected values (given the PCLK and divider values).

Workaround

This is not a silicon issue. This behavior is due to the nature of the composite RC time constant of the I2C bus, which cannot be avoided in a real system as it depends on the components on the board other than the FPGA. The R is the composite pull-up resistor (determined by all pull-ups internal and external to the FPGA) and C is the composite capacitance on the bus (determined by the total number of I2C devices hanging off the same bus, that offer capacitive loading).

The transition from 0 to 1 edge of the I2C clock and data is slowed down exponentially with a rise time that is a direct function of the RC time constant. The lower the RC time constant, the higher the I2C clock frequency at which error-free operation can take place.

The maximum I2C clock frequency is limited by this RC parameter and needs to be characterized by the customer on the board or I2C bus system.

See the article for more information: http://www.microsemi.com/soc/kb/article.aspx?id=KI8850

3.19 Programming SmartFusion FlashROM and/or security using IAP flow is not supported

Programming the FROM/security on the A2F200 and security on A2F500 is not supported using the IAP block. See the following table for a summary of all silicon features that can be programmed through the IAP flow.

Table 5 • **Supported Feature Programming through IAP**

	IAP programming			
Device	**Fabric**	**FROM**	**eNVM**	**Security**
A2F200	Yes	**No**	Yes	**No**
A2F500	Yes	Yes	Yes	**No**
A2F600	Yes	Yes	Yes	Yes

Note: “Yes" indicates that the feature is supported and "**No**" indicates that the feature is not supported.

3.20 Serial Wire Viewer (SWV) does not work at full speed

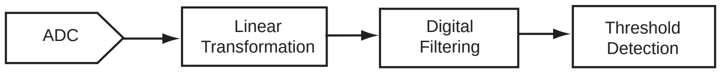
The A2F200 SWV runs at 98 KHz. The SWO clock signal is derived from and is a ratio of the trace clock. For the A2F200 device, it is equal to CPU/1,024 = 98 KHz.

The Serial Wire Viewer works at full speed for the A2F500 and A2F060 devices.

3.21 Post Processing Engine (PPE) microcode is incorrectly generated when “Send filtered value to DMA” is enabled in ACE channel configuration

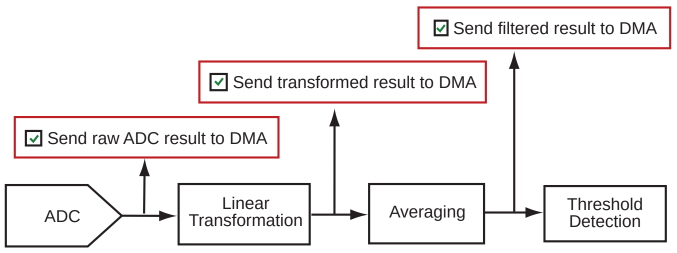
The ACE data path for an analog input signal is shown in the following figure.

Figure 6 • **ACE Data Path for Analog Input Value**



At any point in the data path, a SmartFusion PDMA peripheral can be used to directly access the data (DMA) from the ACE and write it to some designated memory location for later processing.Using the service configurators, you can choose to tap various points in this data path (as shown in the following figure). The **Send filtered result to DMA** check box passes the data to the DMA block after digital filtering.

Figure 7 • **ACE Data Path Available Tapping Points**


Send filtered result to DMA
Send transformed result to DMA
Send raw ADC result to DMA
ADC
Linear Transformation
Averaging
Threshold Detection

In Libero SoC v10 SP1, the PPE microcode is incorrectly generated when **Send filtered value to DMA** is enabled. Prior to Libero SoC v10 SP1 the microcode was generated correctly.

Solution

Update to Libero SoC software v10 SP2 or later.

3.22 The ACE_convert_to_mA() function in ace_convert.c computes Amps instead of mA

This occurs because the voltage is in mV and the resistance is in mOhms. The function calculates the current in Amps instead of mA. In order to get mA, you need to multiply the voltage by 1,000 before dividing by the comparator gain of 50. Therefore, multiply by 20 before dividing by mOhms, as shown in the following equation.

Solution

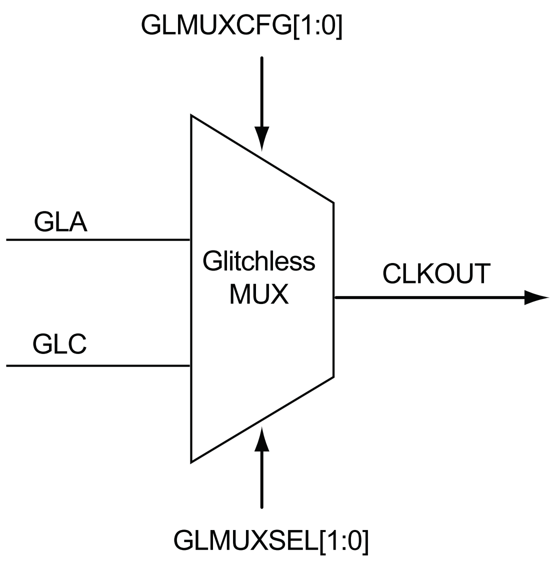
The final calculation must be as shown in the following equation.

$$\text{Current(mA)} = (\text{Voltage (mV)} \times 20\mu\text{s}) / ((\text{Resistance})(\text{mOhms}))$$

3.23 Signal driving fabric CCC/PLL glitchless mux (NGMUX) must be a free running clock signal

The NGMUX is a 2:1 multiplexer that switches glitch-free between two different clock sources and outputs the new clock to the global network, as shown in the following figure. Before switching the NGMUX, the clock source being switched to must be stable to avoid clock glitches propagation through the NGMUX. If not, the NGMUX can propagate those glitches.

Figure 8 • **Glitchless Multiplexer**



The signal assigned to the GCAx pins (Example: GCA0/IO36NDB1V0, GCA1/IO36PDB1V0, and others) as a CLKBUF or CLKBUF_LVPECL or CLKBUF_LVDS goes through the glitchless mux. For the glitchless mux to operate correctly, the signal must be a free-running clock signal. See the "**Glitchless MUX**" section in the *SmartFusion Microcontroller Subsystem User Guide* for more information. Verify that the signal driving this instance is a continuous clock signal.

On A2F500 designs, which have instances using the fabric CCC/PLL GLA output, the designer software issues the following warning message during the Place and Route stage.

3.24 Cold Sparing feature for ABPS signals

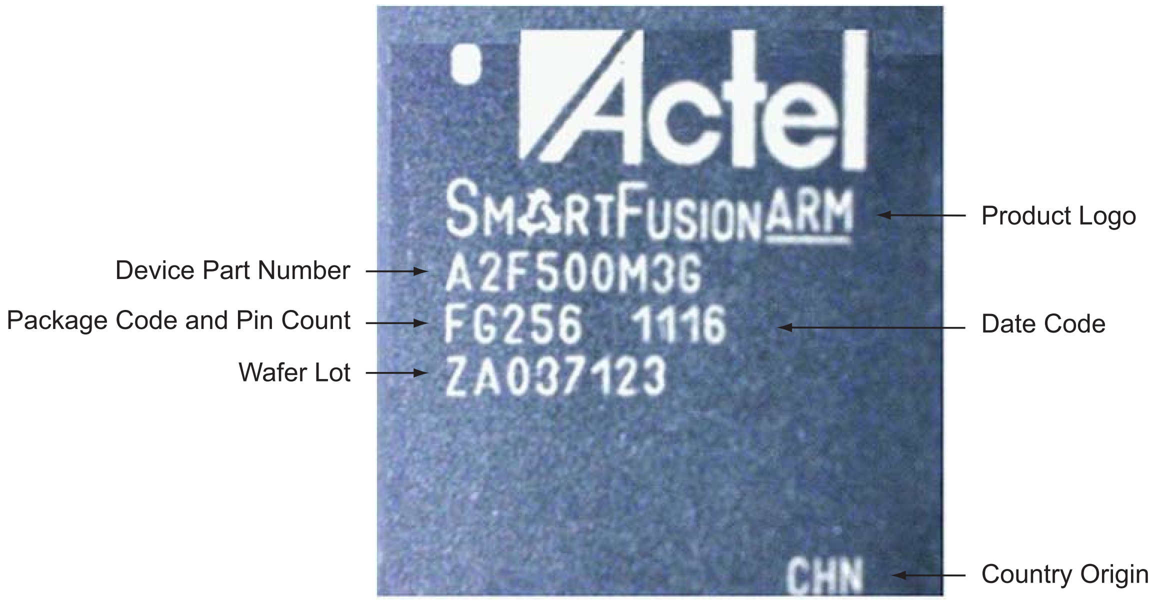
The ABPS inputs of SmartFusion support cold sparing feature. However, if the ABPS inputs are powered with voltages greater than 5 V when the device is not powered, the ESD circuitry on the pin can potentially be damaged and can cause a short and permanent silicon damage. If voltages greater than 5 V are applied to the ABPS inputs when the device is not powered, to prevent possible damage to the device, Microsemi recommends adding a >1K Ohm series resistor to limit the current flowing into the ESD circuitry.

3.24.1 Device Package Markings

The following figure shows a sample of the part marking methodology. Optionally, the following designators are located, as applicable, on the bottom left corner of the device:

- Product temperature grade (example: “I”)
- Speed grade (example: “–1”)
- Low power designator (example: “L”)

Figure 9 • **SmartFusion Part Marking Example**



3.25 While programing the SmartFusion device in IAP mode, with external clock source driving the PLL to clock the MSS, the device may hang at the end of IAP operation

While programing the SmartFusion device in IAP mode, the device may hang at the end of IAP operation. This symptom occurs only if external clock source is used to drive the PLL that is used to clock the MSS. Standard programing (non-IAP) mode does not use the PLL and is not affected by this issue.

When switching between programming and operational modes, in-rush current causes the PLL to lose the lock signal as well as the clock. In some instances the clock never recovers. Since the Cortex M3 in the SmartFusion device depends on the clock source for proper operation, it no longer can function when this condition occurs.

Workaround

To mitigate this issue, a software workaround has been identified which involves performing a system reset, at the end of IAP cycle, instead of the normal ISC_DISABLE instruction. Doing so resets the device and the Cortex-M3 processor.

This workaround has been thoroughly tested and will be available with the MSS IAP Driver version 2.3. This release has a variable called "`enable_system_reset_on_exit`" defined in the `dpuser.c` file. The default value of this variable is TRUE to perform the reset (specified above) at the end of the IAP operation. You may choose to set it to FALSE if you want the device to go through the normal process of exiting programming mode (that is, avoiding the system reset). This may be desired if the internal oscillator is used to drive the MSS clock as the device hang behavior is not observed in this use model.

Solution

The above workaround (that is, performing a system reset) has been thoroughly tested and will be available with MSS IAP Driver version 2.3.

With MSS IAP Driver version 2.3 release, user need not change the code manually. For MSS IAP Driver releases prior to version 2.3, users can manually change the code by calling `NVIC_SystemReset ( )` function in the `dp_exit ( )` function as shown in the following code snippet:

```
void dp_exit(void)
{
#ifdef NVM_SUPPORT
if (
((device_family & SFS_BIT) == SFS_BIT) &&
(dat_support_status | (NVM0_DAT_SUPPORT_BIT | NVM1_DAT_SUPPORT_BIT |
NVM2_DAT_SUPPORT_BIT | NVM3_DAT_SUPPORT_BIT)) &&
(
(Action_code == DP_PROGRAM_ACTION_CODE) ||
(Action_code == DP_VERIFY_ACTION_CODE) ||
(Action_code == DP_PROGRAM_NVM_ACTION_CODE) ||
(Action_code == DP_VERIFY_NVM_ACTION_CODE) ||
(Action_code == DP_PROGRAM_PRIVATE_CLIENTS_ACTION_CODE) ||
(Action_code == DP_VERIFY_PRIVATE_CLIENTS_ACTION_CODE) ||
(
((device_family & SFS_BIT) == SFS_BIT) &&
((Action_code == DP_PROGRAM_ARRAY_ACTION_CODE) ||(Action_code ==
DP_VERIFY_ARRAY_ACTION_CODE))
)
)
)
{
      dp_exit_access_nvm();
}
#endif
      NVIC_SystemReset();
       while( 1 )
        {
```

```
                    ;
                }
            return;
        }
```

Note: Before compiling the code, add the below header file in the `dpalg.c file`:
`#include "a2fxxxm3.h"`