STR91xFA
Errata sheet

STR91xFA limitations and corrections

Silicon identification

STR91x microcontrollers have two major salestype groups, as follows:

- The initial group consists of STR91xFxxxxx devices, for example STR912FW44X6, with silicon revisions B, and D. Detailed technical information for these devices can be found in the STR91xF datasheet, STR91xF reference manual, and STR91xF errata sheets.
- The second group consists of STR91xFAxxxxx devices, for example STR912FAW44X6, with silicon revision G and H (production devices). Detailed technical information for these devices can be found in the STR91xFA datasheet and STR91xFA reference manual.

This errata sheet covers only the second group.

Table 1 summarizes the marking to assist identification. *Figure 1* through *Figure 4* show where the physical marking can be found on the devices.

Table 1. Device identification

Salestype group, external marking	Internal Silicon Revision	External Marking	Note
STR91xFAxxxxx	G	G	see *Figure 1*. Production devices. LQFP128 and LQFP80 packages.
		G	see *Figure 2*. Production devices. LFBGA144 packages.
	H	H	see *Figure 3*. Production devices. LQFP128 and LQFP80 packages.
		H	see *Figure 4*. Production devices. LFBGA144 packages.

Figure 1. Device marking for revision G LQFP80 and LQFP128 packages

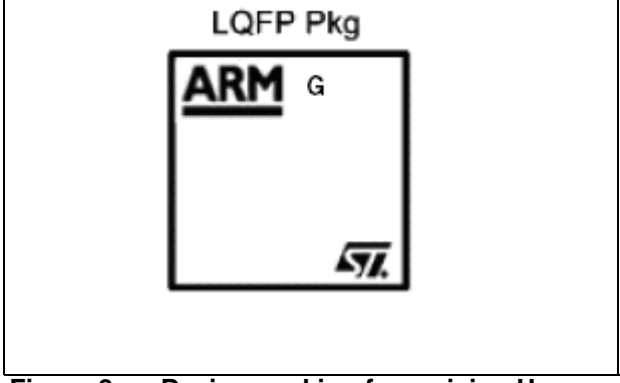


Figure 2. Device marking for revision G LFBGA144 packages

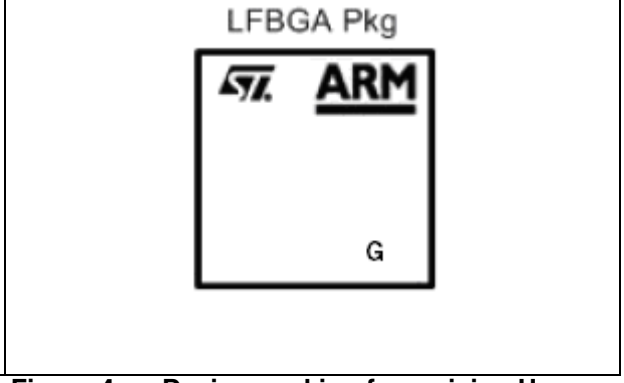


Figure 3. Device marking for revision H LQFP80 and LQFP128 packages

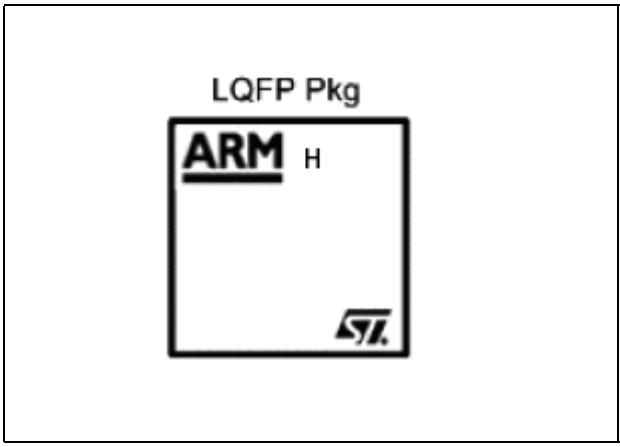
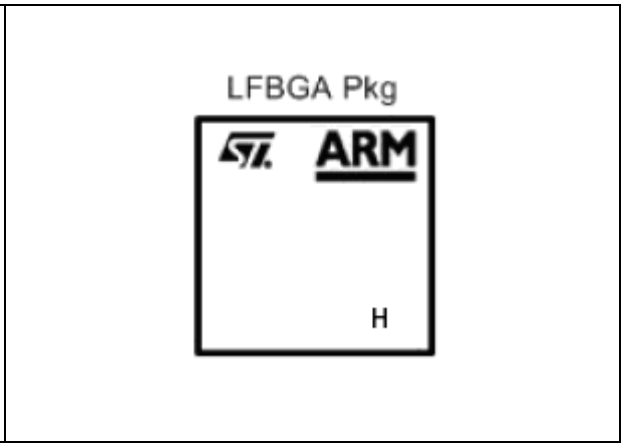


Figure 4. Device marking for revision H LFBGA144 packages



Contents

1 Product evolution

Table 2 summarizes the fix status.(● = fix)

Table 2. Product evolution summary

Section	Limitation	Revision G status	Revision H status
Section 2.1	*Flash memory status register bit 7*	workaround	workaround
Section 2.2	*Flash memory sector protection*	workaround	workaround
Section 2.3	*Flash memory remapping*	workaround	●
Section 2.4	*Flash sequential fetch bank crossing*	workaround	workaround
Section 2.5	*ETM (embedded trace module) configuration*	workaround	workaround
Section 2.6	*EMI bus limitations in 16-bit mode*	workaround	workaround
Section 2.7	*Sleep mode current (I_{SLEEP}) on V_{DDQ} pins*	workaround	●
Section 2.8	*Waking up from Sleep mode*	workaround	workaround
Section 2.9	*Sleep and Idle Mode Requirements*	workaround	workaround
Section 2.10	*Pull-up effect on GPIO ports*	workaround	workaround
Section 2.11	*High leakage on GPIO ports at 5.5 V*	workaround	workaround
Section 2.12	*ADC interrupt generation*	workaround	workaround
Section 2.13	*ADC conversion time and external trigger mode*	workaround	●
Section 2.14	*ADC scan and continuous modes*	N/A	workaround

2 Silicon limitations and fixes

2.1 Flash memory status register bit 7

Description of limitation

Status Register Bit 7 (ready bit) does not reflect the correct status when it is read immediately after the CPU issues a Flash memory Program or Erase command.

Workaround

This requires bit 18 (Instruction TCM order bit) in the Configuration Control Register of the ARM966E-S core to be set. This can be done by the following assembler code:

```
MOV R0, #0x40000
MCR P15,0x1,R0,C15,C1,0
```

When set, the write and read to the Flash Bank are performed in the order generated by the ARM966-ES core.This ensures that writes are committed to the Flash memory before any subsequent read.

This will not be fixed in future silicon revisions.

2.2 Flash memory sector protection

Description of limitation

At power up, the Flash Protection Level 1 register is reset to 0FFFh (all Flash sectors are protected). A warm reset does not reset the register.

Workaround

Firmware must change the values if desired. This will not be fixed in future silicon revisions.

2.3 Flash memory remapping

Description of limitation in revision G

There are two independent Flash memory banks (primary and secondary) that allow read-while-write capability from one bank to the other, enabling In-Application Programming (IAP) of the primary bank while executing code from the secondary bank. After IAP has upgraded firmware in the primary bank, the bootloader code in secondary bank should remap the banks by writing to FMI registers FMI_BBADR and FMI_NBBADR in order to have the primary bank start at address 0x0, which allows interrupt handlers to reside in the application code primary bank instead of interrupts being handled by bootloader code in the secondary bank.

There are two forms of reset in the STR91xFA. A Global Reset (from power-up or voltage drop-out) that clears all functions in the device, and a System Reset (from reset input pin, watchdog, or JTAG reset command) that clears all but a few configuration registers.

Silicon Limitation: The remapping operation is possible. However, when a System Reset occurs after the remapping, the application hangs because the FMI_BBADR and

FMI_NBBADR registers which define bank locations are not cleared by the System Reset, while the FMI_CR register which controls the Chip Select enable of the non boot bank are cleared by the System Reset.

Workaround using revision G

Do not remap the banks after IAP (keep secondary bank at address 0x0), and in the secondary bank redirect ARM interrupts to be handled in the primary bank.

This limitation is fixed in revision H devices by clearing all FMI registers (including bank base address and bank size) after a System Reset.

2.4 Flash sequential fetch bank crossing

Description of limitation

When the two banks (Bank0 and Bank1) are configured at contiguous addresses in the STR91xFA address map and if a code sequence is stored and then fetched by the CPU across this boundary, the CPU may freeze or generate an exception.

Workarounds

1. Do not use the last 8 x 32-bit words of a Flash bank for storing code and use it instead to store data constants (this can be configured by the linker to specify the last zone).
2. If the application firmware is larger than one bank in size, ensure that before reaching the last 8 x 32-bit words of one bank, application code must jump (using LDR, or a branch) to an address forced by the linker to a location in the other bank.

This will not be fixed in future silicon revisions.

2.5 ETM (embedded trace module) configuration

Description of limitation

By default, the ETM9 interface in the STR91xFA core is an 8-bit medium size model as defined by ARM Ltd. When an emulator tool boots up and performs an automatic configuration on the ETM (ETM sniffers), the Configuration register always sends back the 8-bit model status.

However, in order to reduce the number of I/O pins required for debugging, the ETM trace data port is implemented as a 4-bit port. When polled, the 8-bit status provided by the Configuration register is incorrect.

Workaround

When booting up an emulator, do not select the automatic configuration option. Instead, configure the ETM to be a 4-bit port manually.

This limitation will not be fixed in future silicon revisions.

2.6 EMI bus limitations in 16-bit mode

2.6.1 Address boundary limitation in 16-bit asynchronous or synchronous modes

Description of limitation

When accessing the external memory in 16-bit mode, the address must always be aligned to half word (16 bit) or word (32 bit) boundary. The memory address must consequently be an even address, and reading or writing from/to an odd address location may result in incorrect data.

This limitation applies to data memory space only, as code fetch is always aligned to word boundary.

Workaround

All data must be stored at even address location.

2.6.2 Write signal configuration in 16-bit synchronous mode

Description of limitation (applies only to BGA144 devices)

There are two set of write control signals for 16-bit write bus cycle:

1. EMI_WRLn, EMI_WRHn
2. EMI_WEn, EMI_UBn, EMI_LBn

The 2nd set of signals is activated by setting bit 2 in the SCU_GPIOEMI register. When configured in 16-bit synchronous mode, the EMI bus works only with the 2nd set of write signals. The 1st set of signal results in incorrect write signal timings.

Workaround

When using EMI in 16-bit synchronous mode, choose a memory device which accepts Write Enable (EMI_WEn), EMI_UBn and EMI_LBn signals.

If these signals are not available on the memory, it is recommended to operate in asynchronous mode.

2.7 Sleep mode current (I_{SLEEP}) on V_{DDQ} pins

Description of limitation

When the STR91xFA enters Sleep mode, the current drawn from the CPU core voltage (V_{DD}) and from the I/O supply voltage (V_{DDQ}) should drop to a very low value.

I_{SLEEP} current on V_{DD} pins correctly drops to as low as 50 µA at 25 °C.

However I_{SLEEP} current on the V_{DDQ} pins drops to around 500 µA at 25 °C while it should be less than 10 µA.

Workaround using revision G

A limited workaround may be implemented to save around 120 µA by using firmware to put the USB transceiver into Suspend mode (bit2, LP_MODE in the USB_CNTR register). This allows reducing Sleep mode current on V_{DDQ} pins to around 380 µA.

Sleep mode current consumption on V_{DDQ} pins is reduced to less than 10 µA on revision H silicon devices.

2.8 Waking up from Sleep mode

Description of limitation

After the CPU enters Sleep mode, it can be woken up by:

1. External interrupt
2. RTC/USB interrupt
3. External reset

When an oscillator chip is used as the clock source for the STR91xFA, the CPU wakes up from Sleep mode following any of the above three input events. If a crystal is used as the clock source, the crystal is disabled in Sleep mode to save power consumption. When a wake-up event occurs, the crystal will not recover fast enough and the CPU hangs.

Workaround

Workaround solutions include:

1. Use the 32 kHz RTC clock as the clock source for Sleep mode:
   a) Select the RTC clock as the CPU clock source prior entering Sleep mode.
   b) The CPU wakes up following any of the 3 wake up events and waits for the crystal to start oscillation. A crystal start up time is about 1.5 ms typical.
   c) After the crystal wakes up, the CPU waits for a t_{WAIT} time before the first code is fetched from Flash memory. The software can then change the CPU clock source back to the OSC or PLL clock. The duration of t_{WAIT} depends on the crystal frequency. t_{WAIT} equals 50 µs at 25 MHz and 312 µs at 4 MHz.
2. Instead of a crystal, use an oscillator as STR91xFA clock source.

This limitation will not be fixed in future silicon revisions

2.9 Sleep and Idle Mode Requirements

2.9.1 Code execution after setting the Sleep or Idle mode bit

Description of limitation

Once the Idle or Sleep mode are entered by writing the PWR_MODE[2:0] bits in the SCU_PWRMNG register, it takes about $12f_{OSC}$ cycles for the device to stop the execution.

In addition, if a wake-up event or an interrupt (external or internal coming from peripherals) occurs during this period while entering Idle, the internal low power state machine is frozen and the STR91xFA hangs. In this case, only a reset event can wake up the device.

Workarounds

In order to avoid executing any valid instructions after setting the Idle or Sleep bit and before entering the mode, it is mandatory to execute a certain number of dummy instructions after setting the SCU_PWRMNG register. The number of dummy instructions to be executed is given the following formula:

$$No\_dummy\_instr = (f_{CPUCLK}/f_{OSC}) \times 12$$

where f_{CPUCLK} is the CPU core clock frequency and f_{OSC} is the oscillator frequency.

The worst scenario is obtained when the core works from the PLL maximum frequency (96 MHz) with an 4 MHz crystal or oscillator connected to the X1_CPU input. In this case 288 dummy instructions are needed.

Note: If (f_{CPUCLK}/f_{OSC}) is less than 1, the number of dummy instruction is always 3.

Random external/internal wake-up events or interrupts may freeze the STR91xFA when occurring during the execution of these dummy instructions. In this case, only a reset event can wake up the CPU.

This limitation will not be fixed in future silicon revisions

2.9.2 Time required to enter Sleep mode

Description of limitation

After the mode bit is set in the SCU_PWRMNG register, the power management unit requires a period of time (t_{SLEEP}) to switch off all CPU and peripheral clocks safely before entering Sleep mode. A very slow peripheral clock results in a long switch off time.

The t_{SLEEP} time required to enter Sleep mode depends on the oscillator frequency, on the slowest peripheral clock frequency, and on the CPU clock frequency. If a wake-up event occurs during t_{SLEEP}, it is ignored and the STR91xFA does not exit from Sleep mode. t_{SLEEP} is given by the following formula:

$$t_{SLEEP} = 17 \times t\_OSC + 14 \times t\_Slowest\_IP\_CLK + 6 \times t\_CPUCLK$$

where t_OSC is the oscillator frequency, t_Slowest_IP_CLK the slowest peripheral clock frequency, and t_CPUCLK the CPU clock frequency.

Example

- CPU running on RTC clock before entering Sleep mode (f_{CPUCLK} = 32 kHz) (see *Section 2.8*).
- t_OSC = 40 ns (f_{OSC} = 25 MHz)
- t_CPUCLK = t_RTC = 31,250 ns (f_{CPUCLK} = 32 kHz)
- t_Slowest_IP_CLK = 2*31,250 ns assuming all clock dividers are set to 1 (default state) except for APB clock divided which is set to 2, the slowest peripheral clock frequency is then $f_{CPUCLK}/2$.

Then, the value of t_{SLEEP} is:

$$t_{SLEEP} = 17 \times 40 + 14 \times 2 \times 31,250 + 6 \times 31,250 \sim 1.06ms$$

Workarounds

To prevent random external wake-up events from occurring while the device is entering Sleep mode (during t_{SLEEP}), the maximum time required by the application to enter Sleep mode must be taken into account.

This limitation will not be fixed in future silicon revisions.

2.10 Pull-up effect on GPIO ports

Description of limitation

There are no internal or programmable pull-up resistors on I/O ports 0 to 9. By default they are in high-impedance input mode. When the voltage on the pads, V_{PAD}, is higher than the input low-level voltage, V_{IL}, a pull-up effect on the I/O pins is observed in the worst case conditions of manufacturing process variations and operating temperature. The maximum pull-up value is equivalent to a 24 µA current. As a consequence, a 100 kΩ external pull-down resistor is not sufficient to guarantee that the pin is pulled-down to 0.

Workaround

Use at least a 47 kΩ external pull-down resistor. This parameter will be specified in the next datasheet revision and mention of the limitation removed from the errata sheet.

2.11 High leakage on GPIO ports at 5.5 V

Description of Limitation

A high leakage current is observed when applying 5.5 V on 5 V-tolerant I/O pins. This leakage does not dependent on V_{DDQ}, and increases with the temperature.

This high leakage is distributed between the different pins when using a 5.5 V supply voltage. If the application hardware has 10 pins connected to 5.5 V, the leakage current is not multiplied by 10, but distributed between the 10 pins.

This limitation will not be fixed in future silicon revisions.

Workaround

Limit the voltage applied to the 5 V-tolerant I/O pins to 5V to limit the leakage current.

2.12 ADC interrupt generation

Description of limitation

The ADC generates an end of conversion (EVC) or an analog watchdog (AWD) interrupt when enabled. Before returning from serving the interrupt, the ISR typically clears the interrupt by setting the corresponding EVC or AWD flag bit in the ADC_CR register to ‘0’.

The ADC clock is used to clear the interrupt flags. The time taken to clear the flags is longer when the ADC runs on a slow clock. The CPU may return from ISR before the interrupt flag has been cleared. Since the Interrupt Controller input is level sensitive, the CPU sees immediately if another interrupt is pending.

Workaround

Instead of clearing the ADC interrupt flag at the end of the ISR, clear the flag when ISR is entered.

This limitation will not be fixed in future silicon revisions.

2.13 ADC conversion time and external trigger mode

Description of limitation

When the ADC unit is in single conversion mode, the time to complete the conversion varies between 36 and 48 ADC clock periods after the conversion is initiated by an external ADC trigger, a timer trigger, or a firmware command. This limits the maximum ADC conversion rate to 500,000 samples per second for a single channel using an external trigger. This situation also introduces a "jitter" of as many as 12 ADC clocks from one completed conversion to the next.

When the ADC unit is in continuous and scan conversion modes, the time to complete the first conversion varies between 36 and 48 ADC clocks like single conversion mode, but subsequent conversions complete every 16 ADC clocks producing a maximum ADC conversion rate of 1,500,000 samples per second for a single channel.

Workaround on revision G

In single conversion mode, there is no workaround to exceed 500 Ksps conversion rate with external trigger or timer trigger, and no workaround to reduce jitter from one sample to the next.

In continuous and scan conversion modes there is no workaround to reduce delay of the first conversion.

To eliminate jitter, a new conversion mode started by Fast trigger has been added in silicon revision H (refer to STR91xFA datasheet and reference manual). It allows completing each conversion in 16 ADC clocks after it has been triggered by external ADC trigger or internal timer event. A minimum conversion rate of 1.2 Msps can be achieved on a single channel. In single and scan mode, the time to complete the first conversion is reduced to the range of 20 to 32 clocks (down from 36 to 48 clocks) after a trigger event allowing continuous conversion. In this case, the maximum rate on subsequent continuous conversions remains 1.5 Msps on a single channel.

2.14 ADC scan and continuous modes

Description of limitation only in revision H

ADC scan and continuous mode cannot be used together on revision H silicon. The first conversion is performed, but the next channel is not selected, so the end of conversion never occurs.

Workaround

Instead of using scan mode with continuous mode, the application has to select scan mode in single mode. Each conversion can be started by an internal trigger (PWM) for the required number of channels.

The conversion time is deterministic in fast trigger mode. This is a new feature available in revision H silicon (see STR91xFA datasheet and reference manual).

It can be computed using the following formula:

$f_{TRIGGER} < 1/(nb\_channels * 16/f_{ADC})$.

3 Revision history

Table 3. Document revision history

Date	Revision	Changes
10-May-2007	1	Initial release
18-Dec-2007	2	Added *Section 2.4: Flash sequential fetch bank crossing*. Added *Section 2.9: Sleep and Idle Mode Requirements*. Added *Section 2.10: Pull-up effect on GPIO ports*. Added*Section 2.11: High leakage on GPIO ports at 5.5 V*. Added *Section 2.14: ADC scan and continuous modes*.

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST’s terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST’S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER’S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com