The following document contains information on Cypress products. Although the document is marked with the name "Spansion" and "Fujitsu", the company that originally developed the specification, Cypress will continue to offer these products to new and existing customers.

Continuity of Specifications
There is no change to this document as a result of offering the device as a Cypress product. Any changes that have been made are the result of normal document improvements and are noted in the document history page, where supported. Future revisions will occur when appropriate, and changes will be noted in a document history page.

Continuity of Ordering Part Numbers
Cypress continues to support existing part numbers. To order these products, please use only the Ordering Part Numbers listed in this document.

For More Information
Please contact your local sales office for additional information about Cypress products and solutions.

About Cypress
Cypress (NASDAQ: CY) delivers high-performance, high-quality solutions at the heart of today's most advanced embedded systems, from automotive, industrial and networking platforms to highly interactive consumer and mobile devices. With a broad, differentiated product portfolio that includes NOR flash memories, F-RAM™ and SRAM, Traveo™ microcontrollers, the industry's only PSoC® programmable system-on-chip solutions, analog and PMIC Power Management ICs, CapSense® capacitive touch-sensing controllers, and Wireless BLE Bluetooth® Low-Energy and USB connectivity solutions, Cypress is committed to providing its customers worldwide with consistent innovation, best-in-class support and exceptional system value.

Colophon

The products described in this document are designed, developed and manufactured as contemplated for general use, including without limitation, ordinary industrial use, general office use, personal use, and household use, but are not designed, developed and manufactured as contemplated (1) for any use that includes fatal risks or dangers that, unless extremely high safety is secured, could have a serious effect to the public, and could lead directly to death, personal injury, severe physical damage or other loss (i.e., nuclear reaction control in nuclear facility, aircraft flight control, air traffic control, mass transport control, medical life support system, missile launch control in weapon system), or (2) for any use where chance of failure is intolerable (i.e., submersible repeater and artificial satellite). Please note that Spansion will not be liable to you and/or any third party for any claims or damages arising in connection with above-mentioned uses of the products. Any semiconductor devices have an inherent chance of failure. You must protect against injury, damage or loss from such failures by incorporating safety design measures into your facility and equipment such as redundancy, fire protection, and prevention of over-current levels and other abnormal operating conditions. If any products described in this document represent goods or technologies subject to certain restrictions on export under the Foreign Exchange and Foreign Trade Law of Japan, the US Export Administration Regulations or the applicable laws of any other country, the prior authorization by the respective government entity will be required for export of those products.

Trademarks and Notice

The contents of this document are subject to change without notice. This document may contain information on a Spansion product under development by Spansion. Spansion reserves the right to change or discontinue work on any product without notice. The information in this document is provided as is without warranty or guarantee of any kind as to its accuracy, completeness, operability, fitness for particular purpose, merchantability, non-infringement of third-party rights, or any other warranty, express, implied, or statutory. Spansion assumes no liability for any damages of any kind arising out of the use of the information in this document.

Fujitsu Semiconductor Europe
Application Note

MCU-AN-300125-E-V12

FCR4 FAMILY

32-BIT MICROCONTROLLER

MB9EF126

DEBUG REQUIREMENTS FOR TOOL CHAIN

APPLICATION NOTE

Revision History

Date	Issue
2010-11-25	MSt, 1.0 First release
2011-02-14	CEy, 1.1 Several changes
20132-05-02	MSt, 1.2 Removed confidential remark

This document contains 16 pages.

Warranty and Disclaimer

The use of the deliverables (e.g. software, application examples, target boards, evaluation boards, starter kits, schematics, engineering samples of IC's etc.) is subject to the conditions of Fujitsu Semiconductor Europe GmbH ("FSEU") as set out in (i) the terms of the License Agreement and/or the Sale and Purchase Agreement under which agreements the Product has been delivered, (ii) the technical descriptions and (iii) all accompanying written materials.

Please note that the deliverables are intended for and must only be used for reference in an evaluation laboratory environment.

The software deliverables are provided on an as-is basis without charge and are subject to alterations. It is the user's obligation to fully test the software in its environment and to ensure proper functionality, qualification and compliance with component specifications.

Regarding hardware deliverables, FSEU warrants that they will be free from defects in material and workmanship under use and service as specified in the accompanying written materials for a duration of 1 year from the date of receipt by the customer.

Should a hardware deliverable turn out to be defect, FSEU's entire liability and the customer's exclusive remedy shall be, at FSEU´s sole discretion, either return of the purchase price and the license fee, or replacement of the hardware deliverable or parts thereof, if the deliverable is returned to FSEU in original packing and without further defects resulting from the customer's use or the transport. However, this warranty is excluded if the defect has resulted from an accident not attributable to FSEU, or abuse or misapplication attributable to the customer or any other third party not relating to FSEU or to unauthorised decompiling and/or reverse engineering and/or disassembling.

FSEU does not warrant that the deliverables do not infringe any third party intellectual property right (IPR). In the event that the deliverables infringe a third party IPR it is the sole responsibility of the customer to obtain necessary licenses to continue the usage of the deliverable.

In the event the software deliverables include the use of open source components, the provisions of the governing open source license agreement shall apply with respect to such software deliverables.

To the maximum extent permitted by applicable law FSEU disclaims all other warranties, whether express or implied, in particular, but not limited to, warranties of merchantability and fitness for a particular purpose for which the deliverables are not designated.

To the maximum extent permitted by applicable law, FSEU's liability is restricted to intention and gross negligence. FSEU is not liable for consequential damages.

Should one of the above stipulations be or become invalid and/or unenforceable, the remaining stipulations shall stay in full effect.

The contents of this document are subject to change without a prior notice, thus contact FSEU about the latest one.

Contents

REVISION HISTORY **2**
WARRANTY AND DISCLAIMER **3**
CONTENTS **4**
1 INTRODUCTION **5**
2 DEBUG FEATURES & REQUIREMENTS **6**
2.1 CoreSight AHB access port 6
2.2 CoreSight APB access port 6
2.3 General overview about BootROM flow 6
2.4 BootROM waiting time for debugger 7
2.5 Establishing debug connection to MCU with empty Flash 8
2.6 Hardware Watchdog 8
2.6.1 Overview and general requirements 8
2.6.2 Stopping Watchdog during breakpoint 8
2.7 Debugger connection sequence 9
2.7.1 MCU unsecured 10
2.7.2 MCU secured 11
2.7.3 MCU state unknown, generic sequence 12
2.8 Power-on debugging 13
2.8.1 Required registers 14
2.9 Black list support 15
3 APPENDIX **16**
3.1 References 16
3.2 Tables 16
3.3 Figures 16

1 Introduction

For debugging the ARM CoreSight® interface is used in Fujitsu ARM Cortex R4 MCUs. Some additional features are added to enhance the debug possibilities and security. Following chapter will describe these features and the requirements for the debug tool chain in order to support these features.

2 Debug Features & Requirements

This chapter describes the additional debug features and requirements of FCR4 devices

2.1 CoreSight AHB access port

Besides the non-invasive trace functionality an AHB access port (AHB-AP) of the CoreSight system is connected to the bus fabric and allows for non-invasive debugging – or minimal-invasive since the AHB-AP bus master has lowest master priority.

For peripheral register access it is recommended to use the AHB access port. Some peripherals may evaluate the master ID of the access and e.g. will not change FIFO read pointers or clear flags on reading.

The debug tool chain should provide a way to select how the accesses (i.e. using AHB-AP or ARM core) are performed.

2.2 CoreSight APB access port

The CoreSight in Calypso also offers an APB access port for accessing the dedicated debug bus. The base addresses of the debug components can be found in the memory map or by evaluating the DAP ROM table. The Security Checker address space that is important for the debugger connection sequence starts at address 0xF000.

The debug tool chain should provide a way to access the debug bus, e.g. if data shall be exchanged between user and application on a secured device via SCCFG_GPREG0~1.

2.3 General overview about BootROM flow

After any reset the BootROM code is executed. Access to the device via debug interface is completely blocked (with the exceptions mentioned further below) until the device security settings have been evaluated. After this evaluation the blocking may

- persist forever
- persist until the user application allows security key entering and a security key has been entered successfully
- persist until a security key has been entered successfully
- be removed

depending on the configured security markers in the Flash memory.

The following resources are accessible at any time with the debugger by using the APB access port (APB-AP) of Coresight:

- DAP ROM table
- SCCFG_STAT2 (Security Checker Status register 2)
- SCCFG_SECKEY0~3 (Security Checker Security Key register 0~3)
- SCCFG_GPREG0~1 (Security Checker General Purpose register 0~1)
- SCCFG_UNLCK (Security Checker Unlock register), writing the correct unlock key to this register is required before write accesses to other Security Checker registers are accepted

The point in time when BootROM has finished the evaluation is indicated by the status bit SCCFG_STAT2_DBGRDY.

Any debugger access to other resources than mentioned above must wait until SCCFG_STAT2_DBGRDY is ‘1’ and device is either unsecured or secured but security key

has been entered successfully. Note, this also means that security key can be entered at any point of time, e.g. while BootROM is still evaluating security settings.

At the end of the BootROM code an optional waiting time may be enabled with a certain marker in the Flash memory. Afterwards the BootROM will branch to the user application.

2.4 BootROM waiting time for debugger

There is a Boot Description Record (BDR) in the Flash memory. Besides other entries this BDR also contains a so called marker that determines if the BootROM will wait a certain time before branching to the user application.

By programming a “magic word” to this marker in the Flash memory the waiting time is skipped. If the marker value does not match this magic word – e.g. if Flash is empty – the waiting time is applied. Before it is applied it is checked if a debugger is connected at all otherwise the waiting time is also skipped.

Conditions that leave or skip the waiting loop are:

- Magic word in Flash marker is set.
- No debugger connection was recognized. The MCU assumes that a debugger is connected if at least 16 JTAG clock cycles at the TCLK pin have been recognized (status bit: SYSC_JTAGDETECT_DBGCON)
- Debugger sets bit SYSC_JTAGCNFG_DBGDONE to indicate that the debug configuration has finished (e.g. trace, breakpoint settings)
- Half of the default watchdog interval has elapsed (since reset). This ensures that for the user application at least the other half of the default watchdog interval is remaining for watchdog configuration

The minimum default watchdog interval after reset is dependent on the maximum possible RC clock frequency (nominal frequency + “plus-tolerance”).

If no other conditions mentioned above apply, the BootROM will branch to the user application after a certain amount of time. In this time frame the debugger must also complete all necessary configurations:

$$\text{Maximum configuration time after reset} = \frac{1}{2} * \frac{\text{Initial watchdog counter value}}{\text{Maximum RC frequency}}$$

Initial watchdog counter value: 0x01000000

Maximum RC frequency = Nominal frequency + Plus-Tolerance

= 12 MHz + 100% * 12 MHz =24 MHz

Maximum configuration time after reset = 350 ms

Most debugger accesses are blocked until a certain point in the BootROM execution flow that is indicated by the SCCFG_STAT2_DBGRDY status bit (refer chapter 2.3). The BootROM execution time up to this point must be subtracted from the calculated 350ms.

The BootROM execution duration up to this point is not specified but since it is very small compared to the 350ms calculated above it can be neglected for most cases.

2.5 Establishing debug connection to MCU with empty Flash

In case a connection to a MCU with empty Flash memory shall be done, debugger needs to ensure that the ARM core is halted after the connection has been successfully established.

During the waiting time explained in chapter 2.4 the debugger must connect, disable the watchdog and halt the ARM core. After that e.g. the Flash programming kernel can be downloaded.

If the core is not halted in time the BootROM will branch to an “empty” Flash address and therefore fetches 0xFFFFFFFF as instruction code. Since 0xFFFFFFFF is an undefined instruction an exception handler is called. In the default exception handler implemented in the BootROM a software-triggered hardware reset will be executed.

2.6 Hardware Watchdog

2.6.1 Overview and general requirements

The FCR4 series supports an RC oscillator based Hardware Watchdog. This watchdog is enabled by default and its configuration needs to be locked before the default watchdog interval has elapsed, otherwise the MCU is reset.

The configuration options allow the user to completely disable the watchdog or to enable a debug mode. If the debug mode is enabled the watchdog timer will be halted while the core is in debug state.

After every reset the configuration can only be adjusted and locked once. After it has been locked no further writes are possible to the configuration registers and they will result in an error response.

All the points mentioned above must be regarded in the following situations:

- Flash is empty: Since no user application is configuring/clearing the watchdog, it should be disabled and the configuration must be locked afterwards. Then, e.g. the Flash programming kernel can be downloaded
- User application is programmed that configures the watchdog at a certain point in the application start-up code. User should enable debug mode during application development.
    - Debugger should not halt the core before the application has locked the watchdog configuration.
    - If it is necessary to debug the application start-up code up to the point where the watchdog is configured, debugger must configure the watchdog appropriately. If the application is not aware of this problem and gets to the point where it tries to write the locked watchdog registers a data abort exception will be raised. (Workaround: application may check if watchdog registers are already locked before it tries to write the configuration registers)

For above stated reasons the debugger user needs to have the possibility to set the required watchdog configuration via debugger. In case of script file support of a tool chain this could probably be done with a script.

2.6.2 Stopping Watchdog during breakpoint

The bit WDG_CFG_DEBUGEN defines if watchdog is stopped during breakpoint or not. This bit should be set during application development. If this bit is set debugger does not need to trigger watchdog during breakpoint.

As an option debugger could check register configuration given by user and notify if this bit is not set.

2.7 Debugger connection sequence

It is possible to close the debug port for security reasons. A secured debug interface requires the debugger user to send a valid security key in order to open the debug port.

Security settings are configured with markers in the Flash memory.

It is possible to identify if the debug port is secured by reading SCCFG_STAT2 register. This register can be read at any time via the debug bus (APB access port). Following information can be obtained by reading this register:

- Bit 0 (SECEN) identifies if the device is secured (‘1’) or not (‘0’).
- Bit 1 (SECLOCK) identifies if the device is locked (‘1’) or unlocked (‘0’).
- Bit 2 (SEC) identifies if the Security key can be entered (‘1’) or not (‘0’).
- Bit 8 (DBGRDY) identifies if the device is ready for debug access (‘1’) or not (‘0’). This bit indicates the end of the BootROM security evaluation. Certain resources are also accessible while DBGRDY is ‘0’.

After reset there is a certain time while debug port blocks most other accesses. Refer chapter 2.3 for more information.

The user and/or debugger must be aware of following three potential situations and behave accordingly when connecting to the target MCU:

- It is known that the target MCU is unsecured
- It is known that the target MCU is secured
- The security status of the target MCU is unknown (generic approach)

These 3 possibilities are covered in the following sub-chapters.

Furthermore, the debugger must be aware that most types of resets cause a re-evaluation of the security setting by the BootROM, and hence unlocking the device may also be necessary during a debug session if such a reset had occurred. Information about which reset types cause a re-evaluation of security can be found in sub chapter “Effect of reset sources” of System Controller chapter in the Hardware Manual.

2.7.1 MCU unsecured

Although target MCU is actually unsecured debugger must wait until BootROM has finished security evaluation before debugger is accessing any resources other than those that are always accessible.

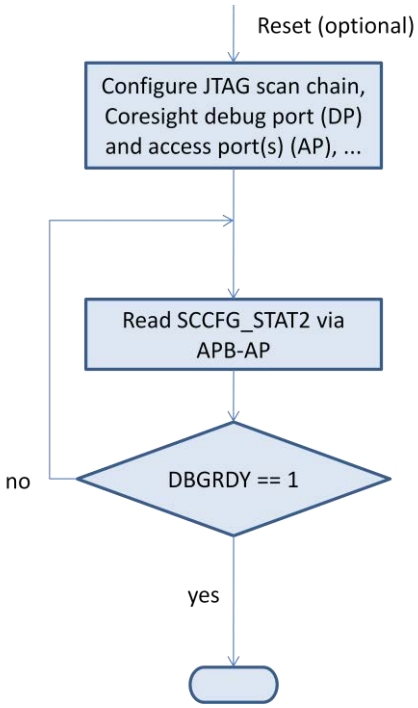

Figure 2-1: Connection sequence if MCU is unsecured

2.7.2 MCU secured

Security key can be entered at any time. The key comparison and its result can be obtained when SCCFG_STAT2_DBGRDY is '1'

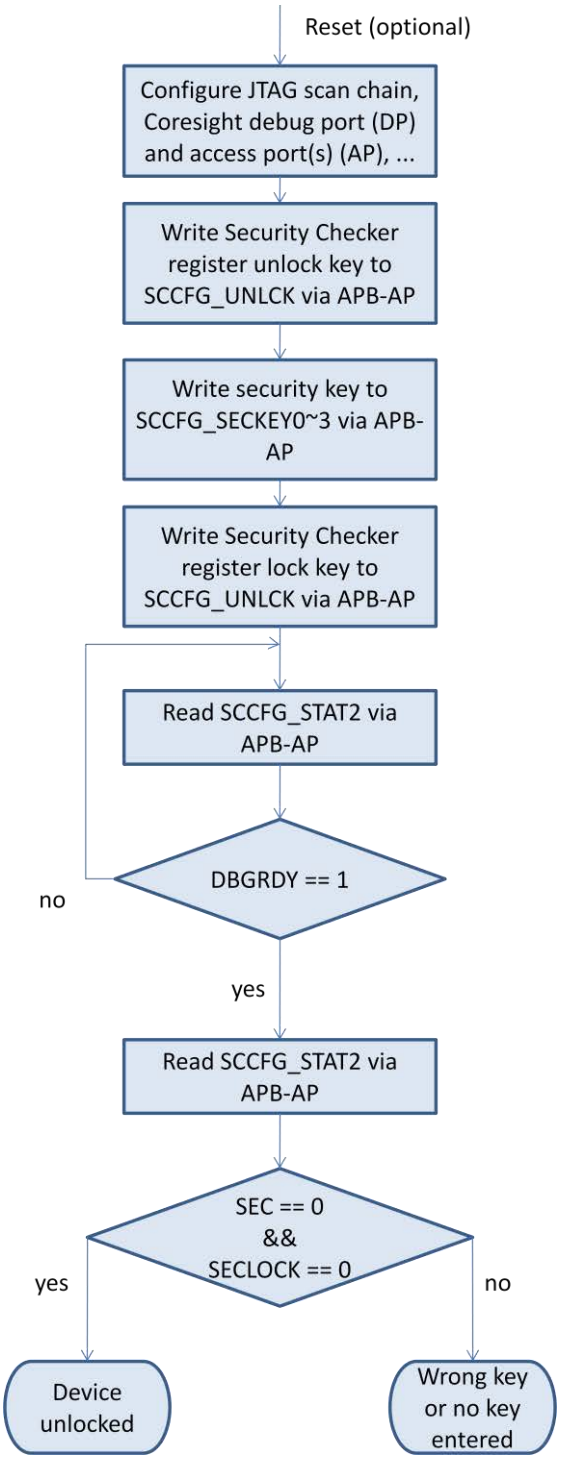

Figure 2-2: Connection sequence if MCU is secured

In case a wrong key has been entered no further key entry is possible without resetting the MCU.

2.7.3 MCU state unknown, generic sequence

Following flow chart shows the generic connection sequence

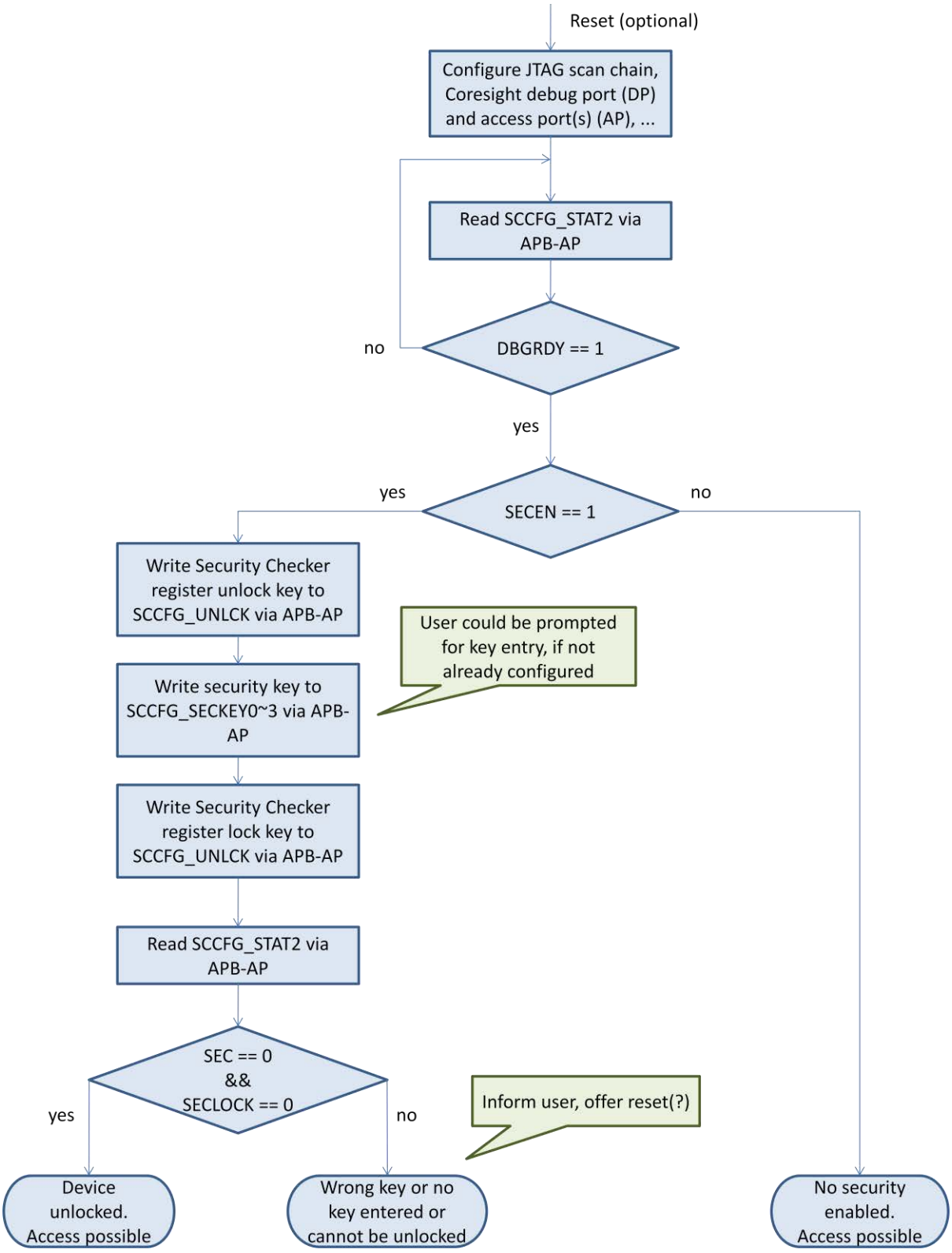

Figure 2-3: Generic connection sequence

2.8 Power-on debugging

It is possible to do power-on debugging with the FCR4 series. Since the debug configuration is lost after a power on reset it is normally not possible to .e.g. setup the trace component for tracing the first user instructions. This can be achieved by using the waiting time mechanism that is described in chapter 2.4.

First time configuration of the CoreSight system can be done as follows:

After the hidden part of the BootROM is executed, the BDR is evaluated. In case the debugger waiting time is enabled and a debugger is connected, the SYSC_JTAGDETECT_DBGCON bit is set to "1". In this case the hardware initializes the SYSC_JTAGCNFG_DBGDONE bit to "0". After the debugger has finished the configuration of all necessary CoreSight components, the debugger has to set the SYSC_JTAGCNFG_DBGDONE bit to "1". Now the boot resumes with public boot ROM part. If a timeout occurs before the DBGDONE bit is set the boot process resumes also. The check if a debugger is connected is done by counting 16 TCK clocks. In case no debugger is connected or a timeout occurs the boot process resumes. The CoreSight System uses TRST for reset, so the CoreSight configuration is still valid after system resets other than power on reset.

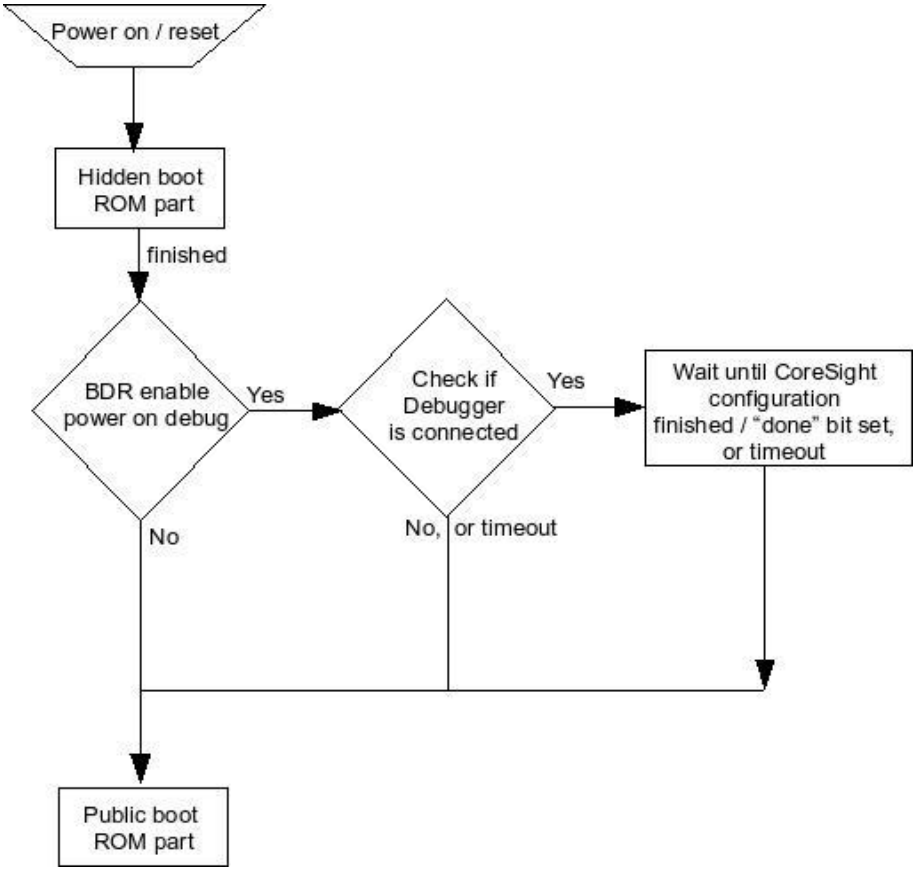

Figure 2-4: Power on debug flow chart (BootROM)

The MCU has several internal power-domains which can be enabled / disabled separately to have a flexible low power mode for different use cases. The debug and trace components are located in power domain PD2 and PD3. Within the System Controller there is a Fake Power Down bit. Setting this bit will keep PD2 and PD3 active even if application is disabling these power domains.

2.8.1 Required registers

The MCU includes a SYSC_JTAGDETECT register, which is used for debugger connection status

Bit 0, DBGCON bit, signals that a connection is recognised; this bit is set / cleared by MCU.

Bit value	Description
0	Debugger unconnected (initial value after reset)
1	Debugger connected

Table 2-1: DBGCON bit description

The JTAG configuration register (SYSC_JTAGCNFG) has a configuration bit to show that CoreSight configuration is completed. This register is initialized to '0', when debugger is connected. This bit is set by the debugger after the debug configuration is done. This register bit is read by BootROM to check the status of debug configuration.

The DBGDONE bit signals if CoreSight configuration is done.

Bit value	Description
0	Debugger configuration ongoing
1	Debugger not connected or configuration done (initial value after reset)

Table 2-2: DBGDONE bit description

2.9 Black list support

The tool chain needs to support a black / or white list for memory accesses.

The FCR4 MCU will apply an abort when accessing reserved areas. For some legacy peripheral memory areas (peripheral bus 0, 1 and 3) a special unit, called Bus Error Collection Unit (BECU) is responsible for issuing an NMI and collecting information about the access if a connected peripheral reports an access violation.

For this reason it is mandatory that the tool chain can apply a black list for given regions e.g. memory window, watch window to ensure that only accesses to allowed memory areas are executed by the tool chain.

Following features are required:

Feature	Required	Details
Minimum region size	Byte (8-bit)	Also within a peripheral there are reserved areas
Write/ Read access	Selectable	Should be possible to define also if read / write / or both should not be done
Access width	Selectable 8-bit, 16-bit, 32-bit, 64-bit and all combination	Should be possible to define which access width is not allowed to use. As some of the registers also report violations when accessing with wrong access width. Also multiple access width should be able to set.

Table 2-3: Required Black list features

3 Appendix

3.1 References

Related documents for further information are listed in the table below:

Ref. #	Document file name	Description
1	FCR4-Cluster-MN707-00001-0v??-E.pdf	FCR4 Cluster Series Hardware Manual
2	MB9EF126-MN707-00001-0v??-E.pdf	FCR4 Cluster Series MB9EF126 - CALYPSO Datasheet
3	DDI0363D_cortexr4_r1p3_trm.pdf	ARM Cortex-R4 and Cortex-R4F Technical Reference Manual
4	DDI0406B_arm_architecture_reference_manual_errata_markup_4_0.pdf	ARM® Architecture Reference Manual ARM®v7-A and ARM®v7-R edition
5	IHI0031A_ARM_debug_interface_v5.pdf	ARM® Debug Interface v5 Architecture Specification
6	IHI0029B__CoreSight_ArchSpec.pdf	CoreSight™v1.0 Architecture Specification
7	CoreSightComponentsTechnicalReferenceManual.pdf	CoreSight™ Components Technical Reference Manual

3.2 Tables

Table 2-1: DBGCON bit description 14
Table 2-2: DBGDONE bit description 14
Table 2-3: Required Black list features 15

3.3 Figures

Figure 2-1: Connection sequence if MCU is unsecured 10
Figure 2-2: Connection sequence if MCU is secured 11
Figure 2-3: Generic connection sequence 12
Figure 2-4: Power on debug flow chart (BootROM) 13

-- END --