

F²MC-16L/16LX FAMILIES

16-BIT MICROCONTROLLER

ALL SERIES

TRANSITION TO AND WAKE-UP FROM STANDBY MODES

APPLICATION NOTE

Revision History

Date	Issue
22. July 2002	V0.1, Hlo First draft
24. July 2002	V1.0, Hlo
15. Oct 2002	V2.0, HLo document reference changed from 390586 (LL) to 900058 (application); figure on signal flow added; statement on "qualified instructions" added; minor corrections
30. Oct 2002	V2.1, Hlo Term of "request" changed in figure; abbreviations added; minor corrections.

This document contains 15 pages.

Warranty and Disclaimer

To the maximum extent permitted by applicable law, Fujitsu Microelectronics Europe GmbH restricts its warranties and its liability for **all products delivered free of charge** (eg. software include or header files, application examples, target boards, evaluation boards, engineering samples of IC's etc.), its performance and any consequential damages, on the use of the Product in accordance with (i) the terms of the License Agreement and the Sale and Purchase Agreement under which agreements the Product has been delivered, (ii) the technical descriptions and (iii) all accompanying written materials. In addition, to the maximum extent permitted by applicable law, Fujitsu Microelectronics Europe GmbH disclaims all warranties and liabilities for the performance of the Product and any consequential damages in cases of unauthorised decompiling and/or reverse engineering and/or disassembling. **Note, all these products are intended and must only be used in an evaluation laboratory environment.**

1. Fujitsu Microelectronics Europe GmbH warrants that the Product will perform substantially in accordance with the accompanying written materials for a period of 90 days from the date of receipt by the customer. Concerning the hardware components of the Product, Fujitsu Microelectronics Europe GmbH warrants that the Product 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.
2. Should a Product turn out to be defect, Fujitsu Microelectronics Europe GmbH's entire liability and the customer's exclusive remedy shall be, at Fujitsu Microelectronics Europe GmbH's sole discretion, either return of the purchase price and the license fee, or replacement of the Product or parts thereof, if the Product is returned to Fujitsu Microelectronics Europe GmbH 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 Fujitsu Microelectronics Europe GmbH, or abuse or misapplication attributable to the customer or any other third party not relating to Fujitsu Microelectronics Europe GmbH.
3. To the maximum extent permitted by applicable law Fujitsu Microelectronics Europe GmbH disclaims all other warranties, whether expressed or implied, in particular, but not limited to, warranties of merchantability and fitness for a particular purpose for which the Product is not designated.
4. To the maximum extent permitted by applicable law, Fujitsu Microelectronics Europe GmbH's and its suppliers' liability is restricted to intention and gross negligence.

NO LIABILITY FOR CONSEQUENTIAL DAMAGES

To the maximum extent permitted by applicable law, in no event shall Fujitsu Microelectronics Europe GmbH and its suppliers be liable for any damages whatsoever (including but without limitation, consequential and/or indirect damages for personal injury, assets of substantial value, loss of profits, interruption of business operation, loss of information, or any other monetary or pecuniary loss) arising from the use of the Product.

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

Contents

REVISION HISTORY	2
WARRANTY AND DISCLAIMER	3
CONTENTS	4
0 INTRODUCTION	5
1 LOW POWER MODES	6
1.1 Overview	6
1.2 Wake-up from standby modes.....	7
1.2.1 Wake-up by interrupt request.....	7
1.2.2 Wake-up by wake-up request only.....	8
2 TRANSITION TO AND WAKE-UP FROM STANDBY MODE	9
2.1 Sequence for transition to standby mode and wake-up by single events	10
2.2 Sequence for transition to standby mode and wake-up by cyclic events.....	11
2.3 Sequence for transition to standby mode and wake-up by cyclic events with the need of resetting the watchdog counter.....	12
3 EXAMPLE OF RESETTING WATCHDOG COUNTER IN TIMER MODE	13
3.1 Used Resources.....	13
3.1.1 External Interrupts (EI).....	13
3.1.2 Real Time Clock (RTC).....	13
3.1.3 Watchdog Counter (WDT)	13
3.1.4 Timebase Timer (TBT).....	13
3.2 Standby/wake-up Flow	14
4 APPENDIX.....	15
4.1 Rules.....	15
4.2 Figures	15
4.3 Tables	15

0 Introduction

All Fujitsu's 16LX-Microcontrollers offer a wide range of low power modes. The standby modes are a subset of these. In the standby modes "timer mode" and "sleep mode" CPU operation is stopped. However, timers can be used to provide periodic wake-up. Also single events may wake-up the MCU.

This application note shows roughly the necessary steps for transition to standby mode and wake-up.

Used abbreviations:

CPU	Central Processing Unit	IRQ	Interrupt Request
C-Unit	Clock Unit	ISR	Interrupt Service Routine
EI	External Interrupt	MCU	Microcontroller Unit
HW	Hardware	MUX	Multiplexer
I	Interrupt Enable-Flag of PS	PLL	Phase Locked Loop
ICR	Interrupt Control Register	PS	Processor Status
IE	Interrupt Enable (Resource)	RTC	Real-Time Clock
IL	Interrupt Level (ICR)	SW	Software
ILM	Interrupt Level Mask (PS)	TBT	Timebase Timer
INT	Interrupt (CPU)	WDT	Watchdog Timer
IR	Interrupt Request (Resource)		

1 Low Power Modes

The current consumption of the MCU can be reduced by limiting execution speed and number of running resources on chip.

1.1 Overview

For details of the following modes, refer to the hardware manual. In general, the normal run modes provide higher performance but also higher current consumption. The standby modes have lowest current consumption. Hence, the following table is somehow sorted from top to bottom, from high performance to low performance, from normal current consumption to low current consumption.

Operation		Clock supply				watchdog
		PLL	CPU	Periphery	Main	
Normal Run modes	PLL run mode	factor 4 factor 3 factor 2 factor 1	active	active	active	active
	PLL intermittent operation mode	factor 4 factor 3 factor 2 factor 1				
Low Power Run modes	Main run mode	inactive				
	Main intermittent operation mode					
Low Power Standby modes	PLL Sleep mode	factor 4 factor 3 factor 2 factor 1	inactive	inactive	inactive	active or inactive ¹
	Main Sleep mode	inactive				
	PLL Timer mode ²	factor 4 factor 3 factor 2 factor 1				
	Main Timer mode	inactive				
	Stop					inactive

Table 1 Operation modes

In standby modes the CPU is not running. Only resources are supplied with clock. Those resources can be used to wake-up the MCU from standby mode. Depending on standby mode, different clock scheme is used.

Because even main oscillator is stopped, the stop mode has lowest current consumption. However, only external interrupts can wake-up from stop mode.

¹ behaviour (active, inactive, configurable) depends on device

² existence of this mode depends on device

1.2 Wake-up from standby modes

To wake up from a standby mode an event from a resource is necessary. The possible resources, which can wake-up an MCU, depend on standby-mode.

standby mode	wake-up source		
	external interrupts	time-base timer, RTC	other resources
Sleep modes	✓	✓	✓
Timer modes	✓	✓	-
Stop mode	✓	-	-

Table 2 Wake-up sources in standby modes

The wake-up request is similar to interrupt requests. The major difference to interrupts is that CPU operation is stopped. Unlike interrupt requests, wake-up requests are accepted regardless of the processor status in PS-register. Hence, wake-up is possible, even if interrupts are disabled or interrupt priority is not sufficient for interrupts.

Consequently, wake-up conditions are enabled by a subset of interrupt settings. Because of these conditions, interrupt configuration is a possible condition but not a necessary condition.

Rule 1: Standby wake-up conditions are a subset of interrupt conditions.

If the wake-up condition is already present when changing to standby mode, the MCU does not enter standby mode. In this case, it immediately continues with wake-up operation.

Rule 2: MCU will not enter standby mode, if a wake-up/interrupt request is present.

1.2.1 Wake-up by interrupt request

An interrupt is executed by adequate configuration of resource, interrupt controller and CPU:

module	Flag/Register	Value
Resource	Interrupt Request Flag	requested (IR = 1)
	Interrupt Enable Flag	enabled (IE = 1)
Interrupt Controller	Interrupt Level of resource	resource level (7 > ICR:IL ≥ 0)
CPU processor status	Interrupt Level Mask	level mask (PS:ILM > ICR:IL)
	Global Interrupt Enable	enabled (PS:I = 1)

Table 3 Conditions for accepting interrupt request

- The request flag of resource is set (e.g. "buffer full") and interrupts enabled.
- The level (IL-bits) in specific Interrupt Control Register (ICRxx) is smaller than 7.
- The level has to pass the level mask and enable flag in Processor Status Register (PS).

In addition, a valid interrupt vector has to be set in the program code. If all conditions are met, the Interrupt Service Routine (ISR) is entered.

If an interrupt request occurs during standby mode, the controller will wake-up first. After that, the interrupt service routine is executed.

Depending on MCU-series and additional conditions, the ISR is executed right after the instruction, which set the standby mode or it is executed one or more instructions later.

1.2.2 Wake-up by wake-up request only

Unlike interrupts, the wake-up conditions do not depend on dedicated CPU-register PS (Processor Status), because CPU-operation is stopped.

module	Flag/Register	Value
Resource	Interrupt Request Flag	requested (IR = 1)
	Interrupt enable Flag	enabled (IE = 1)
Interrupt Controller	Interrupt level of resource	enabled ($7 > ICR:IL \geq 0$)

Table 4 Conditions for accepting wake-up request

Wake-up is not priority-dependent. All resources, which are valid for that standby mode and which are activated, may wake-up the MCU, if their level in interrupt control register is less than seven.

Furthermore, there is no dependency on global interrupt enable flag (PS:I). Hence, wake-up is possible, even if interrupts are disabled.

Rule 3: Wake-up from standby mode is independent from CPU status PS (Interrupt Level Mask PS:ILM, Interrupt Enable flag PS:I)

After wake-up from standby mode, the interrupt request is still pending. As soon as all remaining conditions are met (e.g. by enabling interrupts), the ISR can be executed.

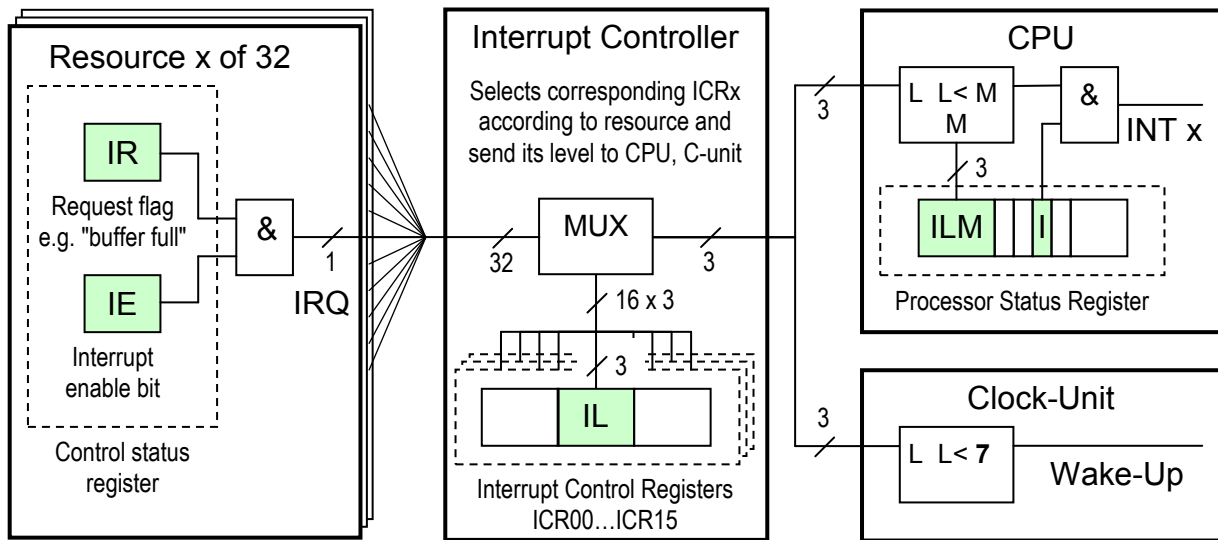


Figure 1 Simplified signal flow from resource to CPU and Clock-unit. Note that two hardware resources share one Interrupt Control Register

2 Transition to and wake-up from Standby Mode

Transition to standby mode and wake-up from standby mode requires setting only a view registers. However, there might be side effects on other parts of software. Following sequences do not apply to all applications but describe a general scenario.

The possible interrupt handling of events, which also wake-up the controller, should be separated from wake-up sequence.

Interrupts should be disabled during the whole sequence of...

- checking software conditions whether to change to standby mode,
- transition to standby mode,
- re-transition to standby
- and final release.

This avoids that interrupt service routines will interfere with pre-standby or post-standby operation.

Rule 4: Standby-transition and wake-up should be performed while interrupts are disabled.

The transition to standby mode is initiated by writing a certain bit. However, only qualified instructions have to be used (e.g. MOV io, A; SETB io, bp ...). Check chapter Low-Power Mode Operation in hardware manual.

Rule 5: Qualified instructions have to be used to activate the low-power mode.

If PLL is disabled in standby mode, it needs to start-up again, after wake-up. During PLL-stabilisation time (a few milliseconds), the application may perform any operations, which do not rely on CPU-speed or resource-speed (e.g. UART). In that time, the application could perform post-standby operations.

Rule 6: The PLL-stabilisation time should be considered as usable time after wake-up.

2.1 Sequence for transition to standby mode and wake-up by single events

A single event can be e.g. an external interrupt or an UART-RX-interrupt. On those events, the MCU might have to release standby mode. However, there might be some more conditions, which are not covered by the request of the resource (e.g. additional port pins or certain character in UART-buffer).

If those conditions are not met yet, the MCU is sent to standby mode again.

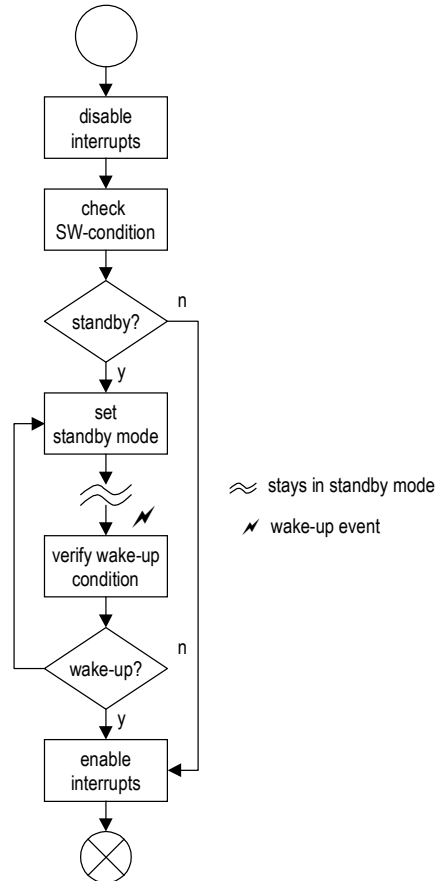


Figure 2 Standby transition and wake-up by by single event

2.2 Sequence for transition to standby mode and wake-up by cyclic events

This is same as with single wake-up event. In addition, a cyclic event (timer) can be used to regularly update data (e.g. a watch clock). In this case, the application might have to leave shown block in order to enable interrupts or to change the clock mode. However, if amount of code is small, the update operation can be called from the shown block.

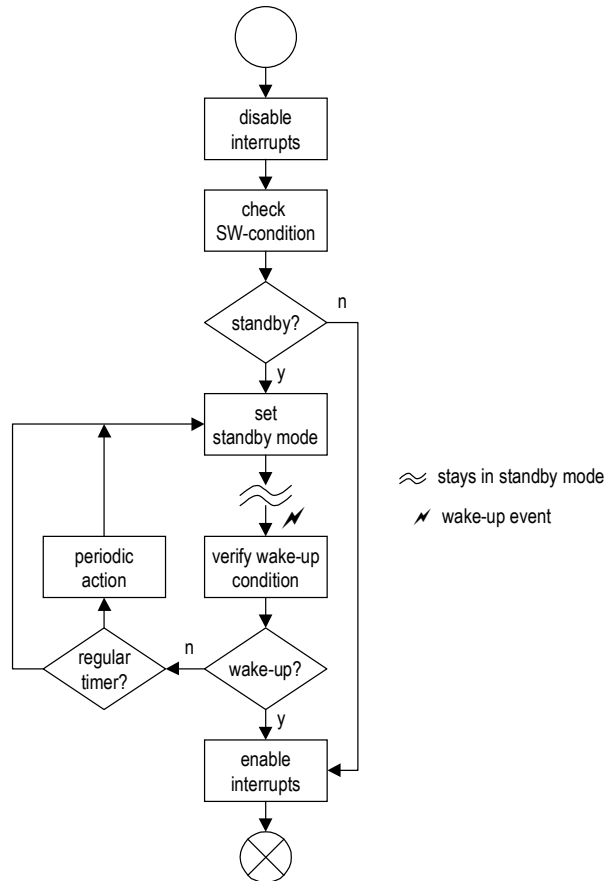


Figure 3 Standby transition and wake-up by by single event and by cyclic timer event

2.3 Sequence for transition to standby mode and wake-up by cyclic events with the need of resetting the watchdog counter.

This is same as wake-up by single plus cyclic event. However, it is extended by the operation needed for resetting the watchdog counter.

Furthermore, the SW-wake-up condition is more complex, because it has to distinguish wake-up events for resetting watchdog, wake-up events for cyclic action and final wake-up of application.

The cyclic wake-up time (e.g. timer) has to be set to a period, which covers both the cyclic update of application data and the watchdog timer period.

The combination and shifted occurrence of those events must be considered.

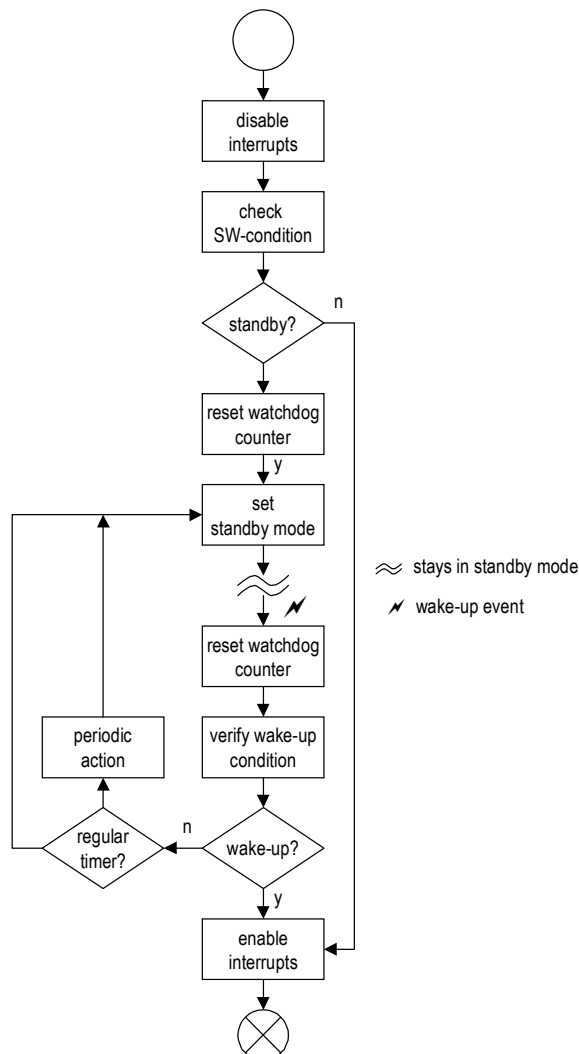


Figure 4 Standby transition and wake-up by by single event and by cyclic timer event and the need to reset watchdog

3 Example of Resetting Watchdog Counter in Timer Mode

This example illustrates the scenario that two timers and watchdog counter have to be coordinated. Therefore, it covers several possible cases. Please note that this combination of resources is not available on all MCU-series.

3.1 Used Resources

3.1.1 External Interrupts (EI)

External interrupts are used wake-up on changing external signals as keypad, switches and others. In this example, the configuration of external interrupts of running application differs from wake-up configuration.

3.1.2 Real Time Clock (RTC)

RTC is used here to provide wake-up events for Seconds and Minutes. Apart from wake-up, RTC also triggers the appropriate interrupt service routines in order to update an display.

3.1.3 Watchdog Counter (WDT)

Watchdog counter is used to observe the application. If application does not regularly reset the WDT, it expires and resets the entire MCU. This enables to restart malfunctioning applications (e.g. deadlock).

In this example, we assume a WDT, which does not stop in standby mode. In order to has to be clear WDT, the standby mode has to be released cyclically.

3.1.4 Timebase Timer (TBT)

Since RTC does not wake-up MCU as often as needed to reset the watchdog counter, TBT is used to wake-up MCU several times in a Second.

3.2 Standby/wake-up Flow

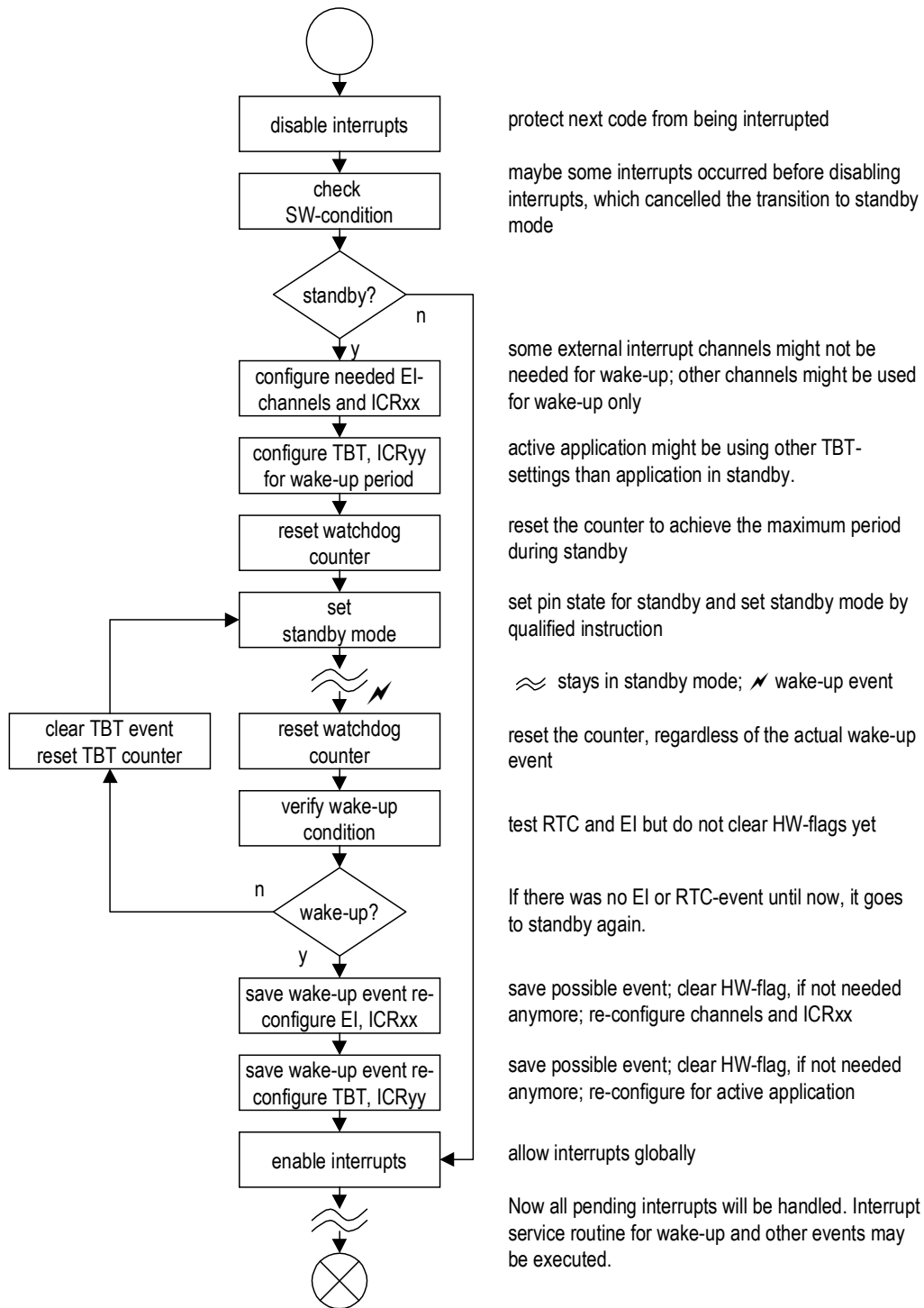


Figure 5 Example of transition to standby and wake-up by by single event, by cyclic timer event and the need to wake-up by extra timer for watchdog reset

4 Appendix

4.1 Rules

The List of rules gives an overview of the more or less strict requirements when using the standby modes. It can be used as a kind of checklist. For details go to the given page. Note that numbering can change with next version of this document.

Rule 1: Standby wake-up conditions are a subset of interrupt conditions.....	7
Rule 2: MCU will not enter standby mode, if a wake-up/interrupt request is present.	7
Rule 3: Wake-up from standby mode is independent from CPU status PS (Interrupt Level Mask PS:ILM, Interrupt Enable flag PS:I).....	8
Rule 4: Standby-transition and wake-up should be performed while interrupts are disabled.....	9
Rule 5: Qualified instructions have to be used to activate the low-power mode.	9
Rule 6: The PLL-stabilisation time should be considered as usable time after wake-up.....	9

4.2 Figures

Figure 1 Simplified signal flow from resource to CPU and Clock-unit. Note that two hardware resources share one Interrupt Control Register	8
Figure 2 Standby transition and wake-up by by single event.....	10
Figure 3 Standby transition and wake-up by by single event and by cyclic timer event	11
Figure 4 Standby transition and wake-up by by single event and by cyclic timer event and the need to reset watchdog	12
Figure 5 Example of transition to standby and wake-up by by single event, by cyclic timer event and the need to wake-up by extra timer for watchdog reset.....	14

4.3 Tables

Table 1 Operation modes.....	6
Table 2 Wake-up sources in standby modes.....	7
Table 3 Conditions for accepting interrupt request	7
Table 4 Conditions for accepting wake-up request.....	8