Fachhochschule Vorarlberg

Master in Mechatronics

Embedded Systems with Constrained Resources

Microcontroller - SGL

Author:
Stefan Stark

24th November 2016

FH Vorarlberg
University of Applied Sciences

Contents

List of Figures

1 SGL1: Power Saving

1.1 Get an overview of techniques used to reduce the power consumption of microcontroller.

1.1.1 Operating voltage

$$uC_{powerConsumtion} \sim U_{supply}^2$$

Reduce supply voltage -> lower limit for the maximum system clk -> increase time in ACTIVE mode.

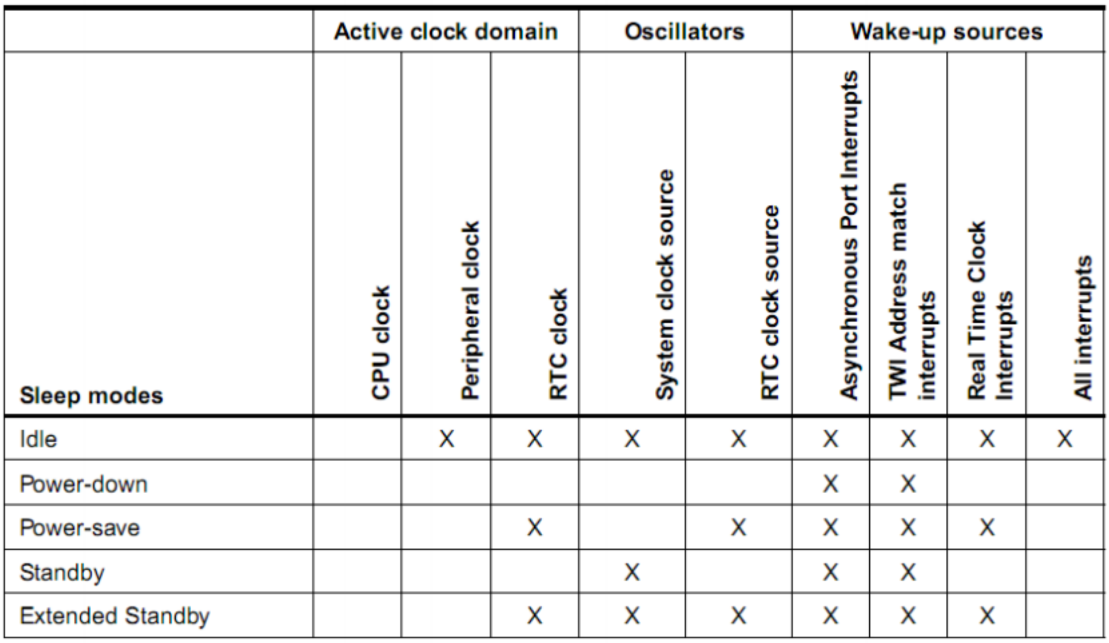
1.1.2 Active mode operation

in active mode

$$uC_{powerConsumtion} \sim f_{clk}$$

use lowest possible system clock frequency.

1.1.3 Sleep modes



	Active clock domain			Oscillators		Wake-up sources			
Sleep modes	CPU clock	Peripheral clock	RTC clock	System clock source	RTC clock source	Asynchronous Port Interrupts	TWI Address match interrupts	Real Time Clock Interrupts	All interrupts
Idle		X	X	X	X	X	X	X	X
Power-down						X	X		
Power-save			X		X	X	X	X	
Standby				X		X	X		
Extended Standby			X	X	X	X	X	X	

Figure 1.1: Available Sleepmodes for XMEGA

IDLE

- Most peripherals are still operating, only CPU core and non-volatile memories (Flash and EEPROM) are stopped.
- DMA & Event sys = active (AD conv. & transfers via USART is possible)
- Can be woken up by all interrupts

POWER-SAVE

- RealTimeClock is still running
- CPU and most other peripherals are stopped.
- RTC to wake the device up at timed intervals.
- Wake-up takes a bit longer than for IDLE (sys clk must be stabilize before operation).

POWER-DOWN

- deepest sleep mode
- unable to wake itself up
- relies on external input (asy. Pin interrupts or TWI) to wake the device up.

STANDBY

- POWER-DOWN with sys clk still running

EXTENDED STANDBY

- POWER-SAVE with sys clk still tuning

1.1.4 Clock Prescaling

- Situations (involve waiting in ACTIVE or IDLE mode for something that takes a fixed amount of time, e.g. serial communication) where it is better to reduce the clock rate.
- Using clock prescaling, which can be changed without causing glitches in the clock signal.

1.1.5 Clock Source Switching

- Generate a 16Mhz sys clk by use of the PLL with the 2MHz RC oscillator as reference rather than the 32MHz RC oscillator with prescaling to 16MHz.
- Use faster clk to go into sleep mode and wake up to reduce wake-up delay

1.1.6 Wake-Up Delays

- Sys clk must stabilize before the CPU starts to operate
- -> Delay (depending on the selected clk)
- 6 cycles (internal RC oscillator or external clk) in addition to the RC oscillator start-up time
- 1000 cycles for ceramic resonators
- 16000 cycles for quartz crystals
- In addition, 13 cycle minimum delay before an ISR start executing after wake-up
- During startup - power consumption = power consumption during IDLE
- wake up as seldom as possible

1.1.7 Power Reduction Registers

- stop internal modules and peripherals to avoid draw pore in ACTIVE mode in IDLE sleep.
- Setting the respective bits in PowerReductionRegisters
- Disable modules and peripherals via their respective control registers before setting their PRR bit.

1.1.8 RTC Clock Source

- Minimize power consumption by clocking the RTC at 1kHz with an external crystal in low power mode

1.1.9 State of Digital I/O Pins

- All digital I/O -> by default floating
- sporadic internal switching and leakage
- disable digital input buffer (if not input)

1.1.10 Virtual Port Registers

- minimize time spent in ACTIVE mode
- virtual port registers (allows single cycle access with I/O memory)

1.1.11 General Purpose I/O Registers

- minimize time spent in ACTIVE mode
- use GPIO registers for storage of variables

1.1.12 Watchdog

- timer with a separate clock source
- use internal 32kHz ULP prescaled to 1kHz
- disable the Watchdog

1.1.13 Brown Out Detector

- BOD ensures that device is not operating at a too low voltage
- During sleep -> device is not operation
- BOD can be configured separately
- enable BOD only in ACTIVE and IDLE mode
- BOD may be run in sampled mode (no voltage dips)

1.1.14 JTAG interface and On-Chip Debugging

- JTAG IF used for programming and OnChipDebugging
- disabled if it is not needed
- OCD disabled in fuses
- JTAG IF disabled in fuses and in software (disable in sw -> device can be reprogrammed - JTAG IF is re-enabled upon RESET)
- PDI IF can be used for programming and debugging

1.1.15 Flash and EEPROM Power Reduction Modes

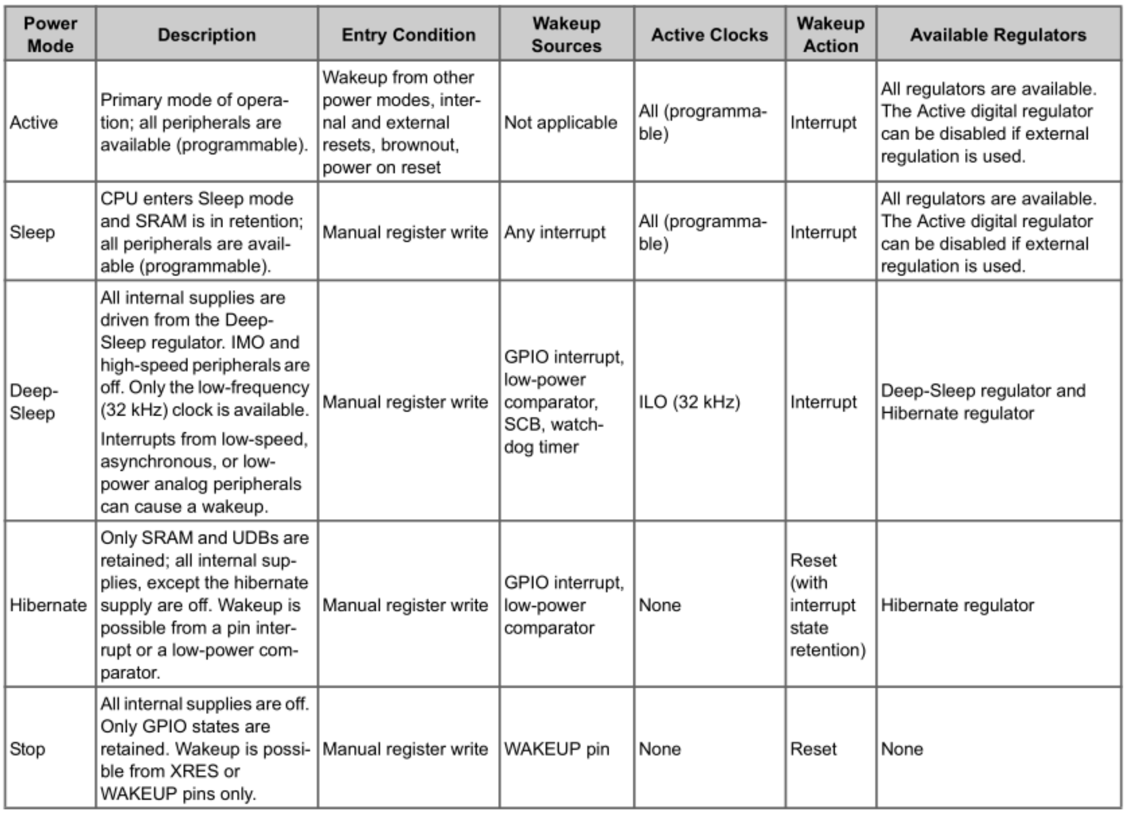
- power reduction modes for the EEEPROM and Flash
- EEPROM and unused section of Flash are powered down in ACTIVE mode

1.1.16

1.1.17 Writing to EEPROM

- > 1 Byte, use of the EEPROM page buffer
- takes just as long to write one byte as it takes to write an entire page toe EEPROM

1.2 Identify the differences between the power-down modes available on the PSoC4.



Power Mode	Description	Entry Condition	Wakeup Sources	Active Clocks	Wakeup Action	Available Regulators
Active	Primary mode of operation; all peripherals are available (programmable).	Wakeup from other power modes, internal and external resets, brownout, power on reset	Not applicable	All (programmable)	Interrupt	All regulators are available. The Active digital regulator can be disabled if external regulation is used.
Sleep	CPU enters Sleep mode and SRAM is in retention; all peripherals are available (programmable).	Manual register write	Any interrupt	All (programmable)	Interrupt	All regulators are available. The Active digital regulator can be disabled if external regulation is used.
Deep-Sleep	All internal supplies are driven from the Deep-Sleep regulator. IMO and high-speed peripherals are off. Only the low-frequency (32 kHz) clock is available. Interrupts from low-speed, asynchronous, or low-power analog peripherals can cause a wakeup.	Manual register write	GPIO interrupt, low-power comparator, SCB, watchdog timer	ILO (32 kHz)	Interrupt	Deep-Sleep regulator and Hibernate regulator
Hibernate	Only SRAM and UDBs are retained; all internal supplies, except the hibernate supply are off. Wakeup is possible from a pin interrupt or a low-power comparator.	Manual register write	GPIO interrupt, low-power comparator	None	Reset (with interrupt state retention)	Hibernate regulator
Stop	All internal supplies are off. Only GPIO states are retained. Wakeup is possible from XRES or WAKEUP pins only.	Manual register write	WAKEUP pin	None	Reset	None

Figure 1.2: PSoC 4 - Power Modes

1.3 Understand how the watchdog module can be used to wake up the PSoC4.

The watchdog counter can send interrupt requests to the CPU in Active power mode and to the WakeUp Interrupt Controller (WIC) in Sleep and Deep-Sleep power modes.

The interrupt request from the WDT is directly sent to the WIC, which will then wake up the CPU. The CPU acknowledges the interrupt request and executes the ISR. The interrupt must be cleared after entering the ISR in firmware.

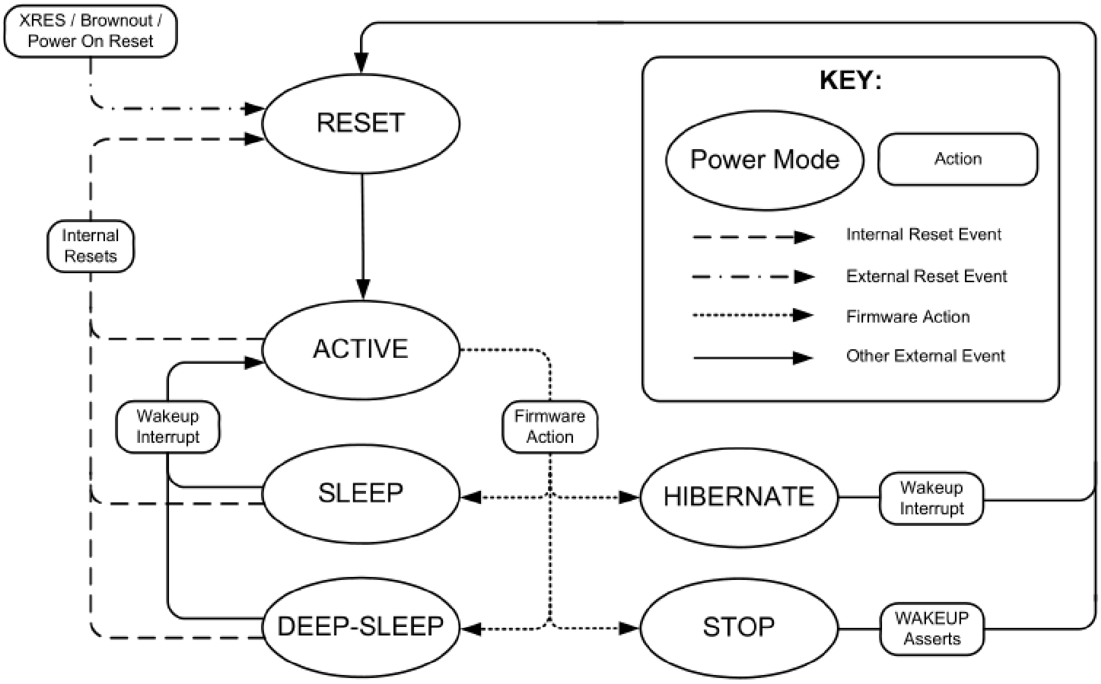


Figure 1.3: PSoC 4 - Power Mode Transitions State Diagram

2 SGL2: PWM

Understand the following:

2.1 The Structure of a capture-compare-PWM-timer based on the PSoC4

page 176

The capture/compare (CC) condition is generated by the TCPWM when the counter is running and one of the following conditions occur:

- The counter value equals the compare value.
- A capture event occurs - When a capture event occurs, the TCPWM_CNT_COUNTER register value is copied to the capture register and the capture register value is copied to the buffer capture register

2.2 The capture mode of operation

page 182

- Mode can be used for period and pulse width measurement.
- During a capture event (hw or software), the current counter register value is copied to the capture register and the capture register is copied to the buffer capture register.

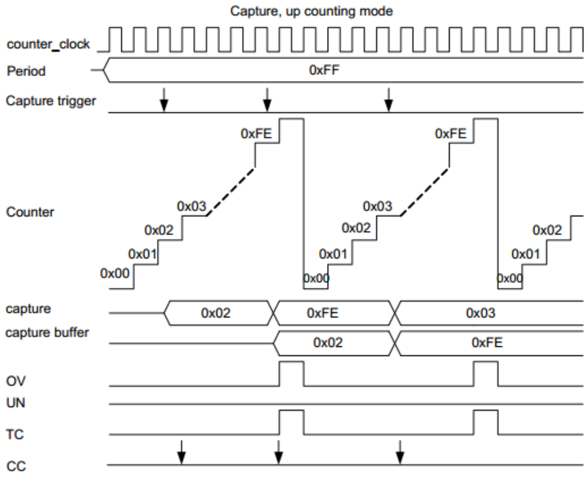


Figure 2.1: Timing Diagram of Counter in Capture Mode, Up Counting Mode

2.3 The quadrature decoder mode of operation

page 185

- Mode can be used to determine speed and position of a rotatory device.
- pos edge on phiA incr. counter when phiB is 0 and decrements the counter when phiB is 1
- Counter is init. with period value on an index/reload event
- TC is generated when counter is initialized -> can generate an Interrupt
- when count reg. reaches 0xFFFF -> count reg. value copied to the capture register and count reg. is init. with period value

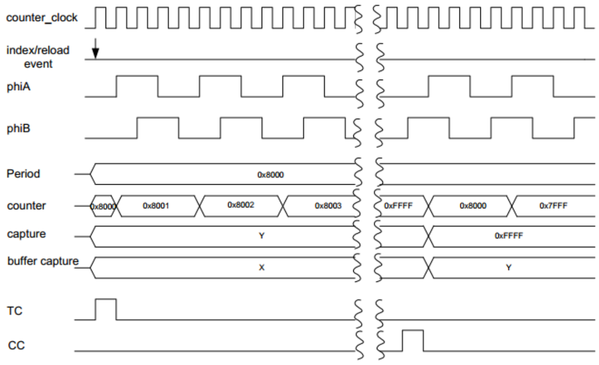


Figure 2.2: Timing Diagram for Quadrature Mode, X1 Encoding

- x2 counts twice times faster than x1
- x4 counts four times faster than x1

2.4 The PWM mode of operation

page 187

- Also called digital comparator mode.
- output = PWM signal,
  - period depends on period register value
  - duty cycle depends on the compare and period register values

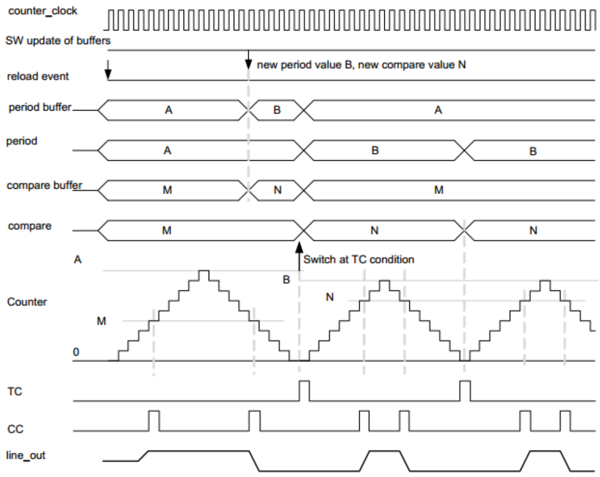


Figure 2.3: Timing Diagram for Center Aligned PWM

2.5 The PWM with dead time mode of operation

page 191

- dead time is used to delay the transitions of line_out and line_out_compl
- Allows to generate two non-overlapping pwm pulses

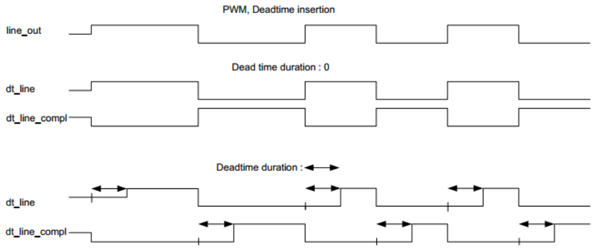


Figure 2.4: Timing Diagram for PWM, with and without Dead Time

2.6 The PWM pseudo-random mode of operation.

page 193

- use the linear feedbackshift register.
- counter register is pseudo-random sequence
- Can be used to modulate between two different compare values using a trigger input signal to control the modulation

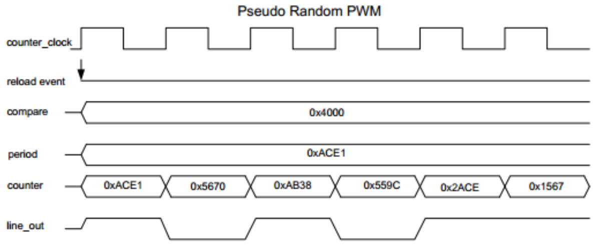


Figure 2.5: Timing Diagram for Pseudo-Random PWM

- line out = 1 when lower 15-bit of counter register is smaller than 16 bit compare register

3 SGL3: Memory Test

3.1 What are the differences between checksums and CRC (cyclic redundancy check) polynomials to verify the integrity of a non-volatile memory?

- CRC uses polynomial division instead of easy addition, that's why it is more effective by recognizing random errors
- Both methods are not save against intended data modifications (manipulation)
- Better methods are cryptographic algorithms like Hash-functions (electronic signature)

3.2 Which are the basic algorithms used for testing volatile memory?

- Classical Test Algorithms (Zero-one, Checkerboard, Walking 1/0)
  - simple, fast but have poor falut coverage (Zero-one, Checkerboard)
  - **OR** good fault coverage but complex and slow (Walking, GALPAT...)
  - Locate / Locate some
- March-based Test Algorithms (MATS, MATS+)
  - finite sequence of March Elements, a March element is specified by an adress order number of reads and writes.
  - Detect / Detect some

3.3 What are their pros and cons in an embedded system power-up sequence?