# XC878

# AP08095

Power Saving Features
Achieving Low Power on Applications

## Application Note

V1.0, 2010-02

Microcontrollers

**Edition 2010-02**

**Published by**
**Infineon Technologies AG**
**81726 Munich, Germany**



**LEGAL DISCLAIMER**

THE INFORMATION GIVEN IN THIS APPLICATION NOTE IS GIVEN AS A HINT FOR THE IMPLEMENTATION OF THE INFINEON TECHNOLOGIES COMPONENT ONLY AND SHALL NOT BE REGARDED AS ANY DESCRIPTION OR WARRANTY OF A CERTAIN FUNCTIONALITY, CONDITION OR QUALITY OF THE INFINEON TECHNOLOGIES COMPONENT. THE RECIPIENT OF THIS APPLICATION NOTE MUST VERIFY ANY FUNCTION DESCRIBED HEREIN IN THE REAL APPLICATION. INFINEON TECHNOLOGIES HEREBY DISCLAIMS ANY AND ALL WARRANTIES AND LIABILITIES OF ANY KIND (INCLUDING WITHOUT LIMITATION WARRANTIES OF NON-INFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS OF ANY THIRD PARTY) WITH RESPECT TO ANY AND ALL INFORMATION GIVEN IN THIS APPLICATION NOTE.

**Information**

For further information on technology, delivery terms and conditions and prices, please contact the nearest Infineon Technologies Office (**www.infineon.com**).

**Warnings**

Due to technical requirements, components may contain dangerous substances. For information on the types in question, please contact the nearest Infineon Technologies Office.

Infineon Technologies components may be used in life-support devices or systems only with the express written approval of Infineon Technologies, if a failure of such components can reasonably be expected to cause the failure of that life-support device or system or to affect the safety or effectiveness of that device or system. Life support devices or systems are intended to be implanted in the human body or to support and/or maintain and sustain and/or protect human life. If they fail, it is reasonable to assume that the health of the user or other persons may be endangered.

**XC878**
**Revision History: V1.0 2010-02**

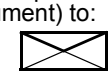
Previous Version(s):

| Page | Subjects (major changes since last revision) |
|---|---|
| – | |

**Trademarks**

TriCore® is a trademark of Infineon Technologies AG.

**We Listen to Your Comments**
Is there any information in this document that you feel is wrong, unclear or missing? Your feedback will help us to continuously improve the quality of this document. Please send your proposal (including a reference to this document) to:
**mcdocu.comments@infineon.com**



# Table of Contents

# 1 Overview

This application note is an extension to the previous note (AP08056) which describes the various power saving features in the XC800 architecture. In this note, users of the XC878 will be provided with information and guidelines on the application of the power saving features via a combination of techniques which include:

- Stopping the CPU clock (Idle Mode)
- Reducing clock speed (Slow-Down Mode)
- Power down of the entire system (Power-Down Mode)
- Stopping the clocks of individual system components (Peripheral Management)

In this document, users can find measurements of the current consumed by the XC878 device when operating in different power modes. There is also a breakdown of the current consumed by the peripherals at different clock speeds. With these measurements, users will have a better understanding of how the different power saving features affect the power consumption of the device and therefore apply them effectively. However, these results are meant as a reference only as the values may vary from device to device. The power consumption can also be affected by various factors such as the type of instructions used in the code, program flow, peripheral settings, measuring equipment, power supply, temperature and the frequency of the oscillator etc.

The measurements recorded in this document are the averages of the readings taken from three different XC878 devices.

Unless otherwise stated, all the readings were carried out under the following conditions:

- Power supply of 5 V
- Ambient temperature of approximately 25 $^{0}$C
- Port 2 (analog inputs) were externally grounded
- All the other port pins were configured as output and set to high
- Peripherals were not programmed to perform any operations

# 2 Stopping the CPU Clock (Idle Mode)

The microcontroller can reduce power consumption by stopping the CPU clock. This can be achieved by putting the device in idle mode. In this mode, the oscillator continues to run, but the CPU is stopped with its clock disabled. Peripherals whose input clocks are not disabled are still functional.

*Note: If the watchdog timer (WDT) is still active when the device goes into idle mode, it will generate an internal reset when an overflow occurs. It is therefore necessary to disable the WDT before entering idle mode.*

The CPU status is preserved in its entirety; the stack pointer, program counter, program status word, accumulator, and all other registers maintain their data during idle mode. The port pins hold the logical state they had at the time the idle mode was activated, unless they are modified by the peripherals. For example, UART, SPI data transfer in progress.

## 2.1 Entering and Exiting Idle Mode

Idle mode can be entered by setting the bit PCON.IDLE.

```
PCON |= 0x01;
```

**Table 1** shows the savings in current consumption between active and idle mode of the XC878 device.

**Table 1 Active and Idle mode current measurements**

| Device | CPU Clock (MHz) | Current (mA) | | |
|---|---|---|---|---|
| | | Active[1)] | Idle | Savings |
| XC878-16FF | 23.58 | 34.96 | 24.12 | 10.84 |

1) Program waits in a continuous loop. i.e. while(1);

Consider a program that is constantly waiting to service a Timer interrupt. The program can:

a) wait in an endless loop for the interrupt event to occur (active mode), or
b) wait for the interrupt event to occur while the CPU is disabled (idle mode) and then enable the CPU to service the interrupt routine when an interrupt occurs.

Method (b) will consume less power as indicated in **Table 1**.

The device in idle mode can exit to active mode in the event of any of these two conditions:

1. Hardware reset. The device resets and code execution starts from address $0_H$.
2. An interrupt has occured from an enabled interrupt source. The device will service the interrupt routine and resumes its operation from the next instruction after the instruction that has previously set the PCON.IDLE bit to 1.

# 3 Reducing Clock Speed (Slow-Down Mode)

The slow-down mode reduces the device's power consumption by scaling down the CPU clock and the peripheral clock with a programmable factor. The slow-down mode is activated by setting the bit PMCON0.SD after the programmable factor in the bitfield CMCON.CLKREL has been selected.

The slow-down mode is terminated by clearing bit PMCON0.SD, which is a protected bit. This means that it cannot be written directly when the protection scheme is activated. An example below shows a software example of a XC878 device entering slow-down mode with its system clock scaled down by a factor of 32.

```
SYSCON0 &= 0xFE;//Access standard SFR region (RMAP = 0)
SCU_PAGE = 0x01;      //Open SCU page 1
CMCON    &= 0xF0;     //Clear CLKREL
CMCON |= 0x06;        //Select clock divider = fsys/32
PASSWD = 0x98;        //Open access to writing of all protected bits
PMCON0 |= 0x08;       //Set SD to enter slow-down mode
PASSWD = 0xA8;        //Close access to writing of all protected bits
```

Similarly, to exit slow-down mode,

```
SYSCON0 &= 0xFE;//Access standard SFR region (RMAP = 0)
SCU_PAGE = 0x01;      //Open SCU page 1
PASSWD = 0x98;        //Open access to writing of all protected bits
PMCON0 &= 0xF7;       //Clear SD to exit slow-down mode
PASSWD = 0xA8;        //Close access to writing of all protected bits
```

## 3.1 Combining Slow-Down Mode with Idle Mode

The slow-down mode can be combined with the idle mode (slow-down-idle mode) by performing the following sequence:

1. Select the slow-down frequency in the bitfield CMCON.CLKREL.
2. Enter the slow-down mode by setting the bit PMCON0.SD.
3. Activate idle mode by setting the PCON.IDLE mode.

The slow-down-idle mode can be terminated by first exiting from idle mode, and by then clearing the bit PMCON0.SD. It will also be terminated by a hardware reset.

## 3.2 Slow-Down Mode vs. Slow-Down-Idle Mode Current Measurements

This section aims to illustrate the effect of operating in the slow down and slow-down-idle modes in terms of the amount of current consumed by the device. The measurements are tabulated in **Table 2**.

Typically, the amount of current consumed by the device decreases with decreasing peripheral clock frequency for both modes of operation. The column marked "Difference" in **Table 2** lists the difference in the amount of current consumed between the two operating modes and can also be attributed to the amount of current consumed by the CPU at each peripheral clock frequency. The CPU consumes less power as the CPU clock is slowed down but the user can achieve further current savings by stopping the clock to the CPU. The amount of current consumed by the peripherals decreases significantly as the peripheral clock frequency reduces. This can be observed in **Table 4**.

**Figure 1** gives a graphical representation of the relationship between the two modes of operation. The graph highlights the benefits of operating in slow-down-idle mode over slow-down mode across all peripheral clock frequencies in reducing the power consumption. Therefore, whenever possible, the user should stop the CPU clock even when operating at a reduced CPU and peripheral clock speeds to reduce the device's power consumption.

**Table 2 Current comparison between slow-down mode and slow-down-idle mode**

| $f_{sys}$ = 141.52 MHz | | Current (mA) | | |
|---|---|---|---|---|
| **Peripheral Frequency (MHz)[1)]** | **CLKREL** | **Slow-Down (X)** | **Slow-Down-Idle (Y)** | **Difference (X-Y)** |
| 23.58[2)] | $0000_B$ | 34.96[3)] | 24.10[4)] | 10.86 |
| 11.79 | $0001_B$ | 21.78 | 14.56 | 7.22 |
| 5.89 | $0011_B$ | 14.84 | 9.55 | 5.29 |
| 2.94 | $0101_B$ | 11.29 | 7.01 | 4.28 |
| 1.47 | $0111_B$ | 9.57 | 5.76 | 3.81 |
| 0.73 | $1001_B$ | 8.71 | 5.14 | 3.57 |
| 0.36 | $1011_B$ | 8.27 | 4.83 | 3.44 |
| 0.18 | $1101_B$ | 8.07 | 4.67 | 3.40 |

1) Peripheral frequency = $f_{sys}$/CLKREL/Fixed divider of 2 (refer to user's manual on Clock System).
2) This is the value of PCLK in active mode (no reduction in PCLK).
3) At this PCLK frequency, the current consumed by the device is similar to that consumed when the device is operating in active mode.
4) At this PCLK frequency, the current consumed by the device is similar to that consumed when the device is operating in idle mode.

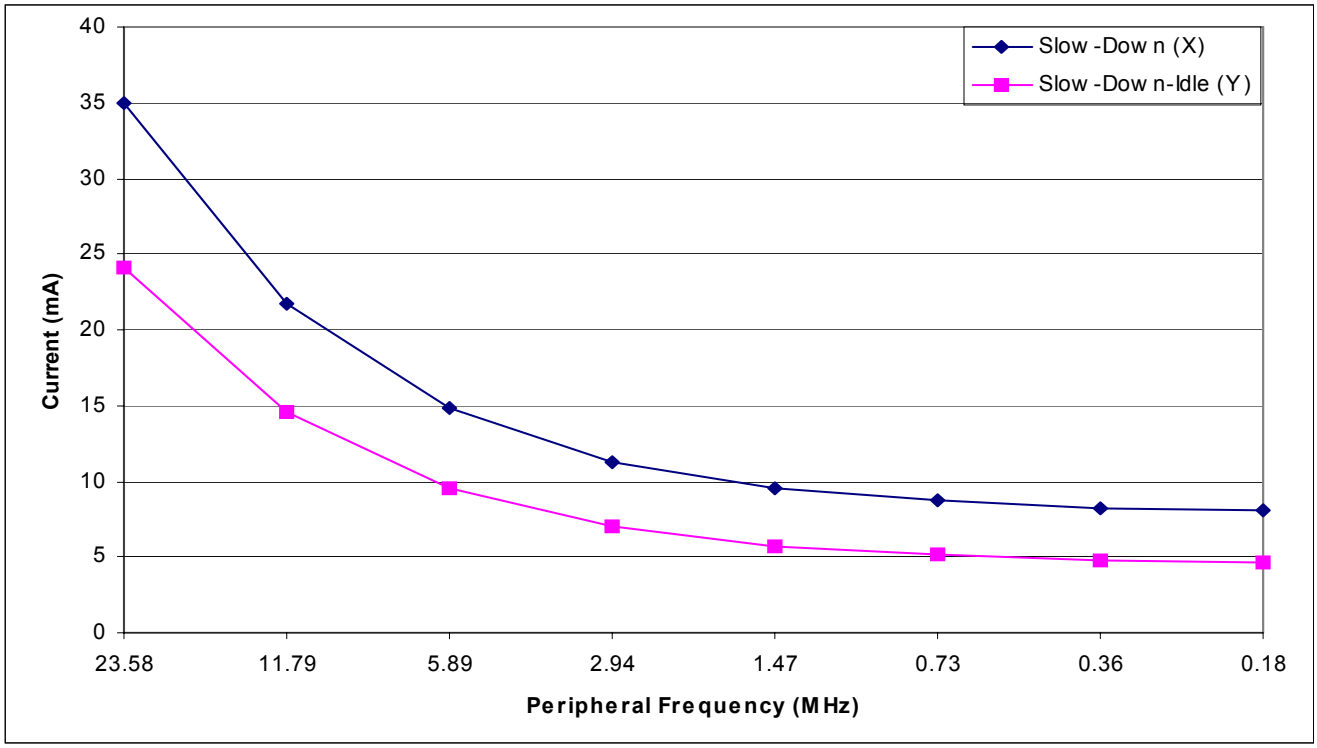



**Figure 1 Graphical representation of X and Y series in Table 2**

# 4 Power Down of the Entire System (Power-Down Mode)

Power-down of the entire system can be achieved by setting the device in power-down mode. In this mode, all functions of the microcontroller are stopped and only the contents of the FLASH, on-chip RAM, XRAM and the SFRs are maintained. The port pins hold the logical state they had when the power-down mode was activated. For the digital ports, the user should ensure that the ports are not left floating. Floating input port pins should be set as output and set to high. For the analog Port 2, the user should ground the port pins if they are not in use.

In power-down mode, the clock is turned off. Hence, it cannot be awakened by an interrupt or by the WDT. It is awakened only when it receives an external wake-up signal or reset signal.

## 4.1 Entering Power-Down Mode

Software requests power-down mode by setting the bit PMCON0.PD (protected bit) to 1. Two NOP instructions must be inserted after the bit PMCON0.PD is set to 1. The device will not enter power-down mode immediately after executing the instruction to set PMCON0.PD, therefore the two NOP instructions ensure that the first instruction that follows them is executed correctly after wake-up from power-down mode.

If the external wake-up from power-down is used, software must select the EXINT0 pin, the RXD pin, or either of these two pins as the wake-up source by selecting the WS bit of the PMCON0 register. Exit from power-down mode can be achieved when a falling edge trigger is detected at the selected source(s). Bit MODPISEL.URRIS is used to select one of the two RXD inputs and bit MODPISEL.EXINT0IS is used to select one of the two EXINT0 inputs. The wake-up source and wake-up type must be selected before the system enters power-down mode.

In the case where a wake-up with reset is selected, while the program will not be able to resume operation from where it was stopped after the reset, it is still possible to save device status to memory. The example below shows how the power-down mode can be entered correctly with the port status stored to stack.

```
//...SP was initialized as 0x41
PORT_PAGE = 0x00;//Open port page 0 to access Px_DATA
_push_(P3_DATA);//Save port 3, current SP = 0x42.
                      //_push_() function is defined in intrins.h
STACKSP = SP;         //Save current stack pointer at an absolute memory
                      //location. e.g. unsigned char idata STACKSP _at_
                      //0xe0; iram location 0xe0 = 0x42
SCU_PAGE = 0x01;      //Open SCU page 1 to access PMCON0 and PASSWD
PASSWD = 0x98;        //Open access to writing of all protected bits
PMCON0 |= 0x17;       //Enable power-down with wake up source RXD or
                      //EXINT0 selected and select wake-up with reset
_nop_();              //2 NOPs necessary for proper execution of
_nop_();              //code upon wakeup. Function is defined in
                      //intrins.h.
```

*Note: Some IRAM locations will be corrupted after a reset. The stack and the constant variable location must avoid these locations. Please refer to the Errata Sheet for the detailed locations.*

*Note: Compiler C-Startup files may also clear IRAM locations ranging from 0 to the number defined in the symbol IDATALEN. It is therefore necessary to redefine the data assigned to this symbol to avoid destroying the stack and the constant variable data.*

## 4.2 Exiting Power-Down Mode

Power-down mode can be exited in two ways:

1. Hardware reset. The device resets and code execution starts from address $0_H$.
2. The EXINT0 pin, the RXD pin or either of these two pins detects a falling edge. The wake-up source(s) must be selected prior to entering power-down mode.

When a wake up source is selected, the device will wake-up from power-down mode upon detecting a falling edge trigger at that pin. In power-down mode, EXINT0 pin/RXD pin must be held at high level. Power-down mode is exited when EXINT0 pin/RXD pin goes low for at least 100ns.

The device can wake-up with or without reset depending on the setting of the PMCON0.WKSEL bit. If the WKSEL bit was set to 1 prior to entering power-down mode, the system will execute a reset sequence similar to the power-on reset sequence. Therefore, all port pins and SFRs are put into their reset state and will remain in this state until they are affected by program execution. The only exception is that hardware will set the wake-up indication bit PMCON0.WKRS to indicate that the device is woken up from power-down reset.

If bit WKSEL was cleared to 0 before entering power-down mode, a fast wake-up sequence is used. The port pins continue to hold their state which was valid during power-down mode until they are affected by program execution.

The example below shows how the previous example can restore the port 3 status after the power down mode wakeup with reset.

```
//... after initialization
SCU_PAGE = 0x01;      //Open SCU page 1 to access PMCON0
if (PMCON0 & 0x20)    //Check PMCON0.WKRS for wakeup indication
{
       SP = STACKSP; //Restore stack pointer, SP = 0x42
       PORT_PAGE = 0x00; //Open port page 0 to access Px_DATA
       _pop_(P3_DATA);       //Restore P3 from stack
}
//... continue...
```

Table 3 illustrates the timing for a device to wake-up from power-down mode to active mode for both wake-up with/without reset.

**Table 3 Wakeup timing from power-down mode**

| Reset Sequence | XC878-16FF | |
|---|---|---|
| | WKSEL = 0 | WKSEL = 1 |
| EVR is stable | 300µs (Typical) | |
| PLL is locked | 200µs (Maximum) | |
| BootROM execution | 0µs[1] | 250µs[2] (Typical) |
| Total wake-up time | 500µs | 750µs |

1) For wake-up without reset (WKSEL = 0), this refers to the instruction after the instruction that set the power down bit.
2) For wake-up with reset (WKSEL = 1), this refers to the execution of the first instruction in the user code.

In power-down mode, typically, the amount of current drawn by the device is approximately 10µA at 25°C.

# 5 Stopping the Clocks of Individual System Components (Peripheral Management)

Peripherals which are not required for a particular functionality can be disabled by programming the assigned register bits in PMCON1 and PMCON2 which would gate off clock inputs to the individual peripherals. This further reduces overall power consumption of the device.

Table 4 provides the relationship between the amount of current that can be saved when each/all of the peripheral module(s) are disabled and the peripheral clock frequency ($f_{PCLK}$). It can be observed that the power consumed by the peripherals reduces with decreasing peripheral clock speed, thus more current can be saved with the disabling of a peripheral.

Software example to disable all peripherals:

```
SCU_PAGE = 0x01;              //Open SCU page 1 to access PMCON1
PMCON1 |= 0x7F;               //Disable ADC, CCU, SSC, T2CCU, MDU, CAN and
                              //CORDIC
SCU_PAGE = 0x03;              //Open SCU page 3 to access PMCON2
PMCON2 |= 0x03;               //Disable T21 and UART1
```

**Table 4 Current consumption by the peripherals of XC878 device at different peripheral clock (PCLK) frequencies**

| | Current (mA) | | | | |
|---|---|---|---|---|---|
| **Peripheral Frequency (MHz)** | **23.58** | **11.79** | **2.94** | **0.73** | **0.18** |
| **CLKREL** | **0000B** | **0001B** | **$0101_B$** | **$1001_B$** | **$1101_B$** |
| ADC | 1.04 | 0.51 | 0.13 | 0.03 | 0[1)] |
| CCU | 1.55 | 0.76 | 0.20 | 0.04 | 0.01 |
| SSC | 1.01 | 0.51 | 0.12 | 0.03 | 0[1)] |
| T2/T2CCU | 1.11 | 0.54 | 0.14 | 0.03 | 0[1)] |
| T21 | 0.25 | 0.12 | 0.03 | 0[1)] | 0[1)] |
| CAN | 6.52 | 3.25 | 0.81 | 0.20 | 0.04 |
| CORDIC | 1.00 | 0.49 | 0.12 | 0.03 | 0[1)] |
| MDU | 0.40 | 0.20 | 0.05 | 0.01 | 0[1)] |
| UART1 | 0.41 | 0.17 | 0.05 | 0.01 | 0[1)] |
| All | 12.66 | 6.43 | 1.59 | 0.39 | 0.09 |

1) <10µA

# 6 Reset Current

This section describes the current consumption by the devices in reset state, i.e. when the reset pin is held low. All the port pins are left unconnected except the reset pin which is grounded.

**Table 5 Current drawn in reset state**

| Device | Current (mA) |
|---|---|
| XC878-16FF | 3.269 |

# 7 Power Saving Checklist

- Ground the unused analog pins (Port 2).
- Input port pins should not be left floating. Port pins should be pulled to a defined level (as input or output) instead of being left floating. Output pins that are pulled-up should be set to high or have the pull-up disabled.
- The analog part of the ADC module can be disabled by resetting the bit GLOBCTR.ANON. This feature causes the generation of $f_{ADCI}$ to be stopped and allows a reduction in power consumption when no conversion is needed.
- Disable peripherals that are not used.
- Use idle mode instead of polling loops (see **Chapter 2.1**).
- Reduce the operating frequency of the system, CPU or peripherals via the selection of power saving modes and clock modes when the maximum bandwidth is not required. The user must also consider the response time of the system to interrupt events. When the system frequency is reduced, the interrupt latency will increase accordingly thus making it unsuitable for applications that require quick response time.

# 8 Summary

- Idle mode, slow-down mode and power-down mode are the three main power saving modes offered in the XC878 device.
- Idle mode reduces the device's power consumption by shutting off the CPU. This is implemented by stopping the clock to the CPU.
- Slow-down mode reduces the device's power consumption by reducing the system frequency that is supplied to the CPU and the peripherals when the maximum bandwidth is not required.
- A combination of idle and slow-down mode can help to reduce power consumption further.
- Power-down mode reduces the device's power consumption by stopping the clock to the CPU and the peripherals. Only the contents of the FLASH, on-chip RAM, XRAM and the SFRs are maintained.
- Unused peripherals can be disabled by gating off the clock input to save power.

w w w . i n f i n e o n . c o m

Published by Infineon Technologies AG