

EFM[®]32

... the world's most energy friendly microcontrollers

Energy Modes

AN0007 - Application Note

A decorative graphic on the right side of the page consisting of concentric circles. The outermost ring is light green and contains the number '0'. The next ring inward is a darker green and contains the number '1'. The third ring is blue and contains the number '2'. The fourth ring is a darker blue and contains the number '3'. The innermost circle is black and contains the number '4'. The circles are arranged in a way that they appear to be part of a larger, partially visible circular structure.

Introduction

This application note describes strategies to reduce current consumption as well as how to enter different energy modes.

Additionally the prime number calculation code used in current consumption benchmarking is included.

This application note includes:

- This PDF document
- Source files (zip)
 - Example C-code
 - Multiple IDE projects



ENERGY[®]
micro

www.energymicro.com

1 Energy Saving

1.1 General

In battery powered microcontroller applications, energy saving is essential. By reducing the current consumption, the mean time between battery charging / replacement can be significantly increased. Following these principles will drastically reduce the current consumption:

- Turn off Unused Modules / Peripherals
- Disable Clocks to Unused Modules / Peripherals
- Reduce Clock Frequency
- Lower the Operating Voltage

The EFM32G supports extensive usage of all these principles. This will be explained in this and the following chapters.

1.2 Turn off Unused Modules / Peripherals

At a given time in every microcontroller application, there are modules / peripherals which are not used. By turning these off, current will be saved.

If a module is used part-time (e.g. the ADC performs a temperature measurement each second), it will be more energy efficient to only enable the module when it is used and keep it turned off otherwise. This way, extra current is only consumed when necessary and idle current is kept at a minimum.

This also applies to the CPU itself. If the core is idle (e.g. waiting for data reception), it can be turned off and energy is saved. This is one of the main features of the energy modes of the EFM32G. Please see the Energy Modes chapter for details.

Remember to take startup/stop overhead into consideration.

1.3 Disable Clocks to Unused Modules / Peripherals

Even though a module / peripheral is disabled (e.g. TIMER0 is stopped), energy will still be consumed in that module if its clock is running. Therefore, it is important to turn off the clocks of all unused modules. The EFM32G fully supports this, please see the attached example and the Clock Management Unit (CMU) chapter in the EFM32G reference manual for details.

1.4 Reduce Clock Frequency

Current consumption scales with clock frequency. In general, a certain task or module should be operated at the lowest possible frequency. E.g. if a timer is to give an interrupt each ms, it is better to run it at a few kHz, rather than several MHz. This can easily be implemented using the prescaling functionality in the CMU.

Likewise, one approach regarding the CPU frequency is that it should be so low that the CPU is not idle (some margin should be added). However, in many cases it is by far better to complete the current tasks quickly and then enter a suitable energy mode until new tasks must be handled. The different energy modes of the EFM32G are optimized for this purpose, please see the Energy Modes chapter for a further description.

1.5 Lower the Operating Voltage

The power consumption of a digital circuit is proportional to the square of the supply voltage. For example, this means that reducing the supply voltage from 3.3V to 1.8V will theoretically lower the power consumption by more than two thirds.

The EFM32G supports running at a supply voltage as low as 1.8V.

2 Energy Modes

2.1 General

The different energy modes of the EFM32G differ in energy consumption, CPU activity, active peripherals and clocks. A brief presentation of each energy mode is given below.

2.2 Run Mode (Energy Mode 0)

This is the default mode. In this mode the CPU fetches and executes instructions from flash or RAM, and all peripherals may be enabled.

2.3 Sleep Mode (Energy Mode 1)

In Sleep Mode the clock to the CPU is disabled. All peripherals, as well as RAM and Flash are available. The EFM32G has extensive support for operation in this mode. By using the Peripheral Reflex System (PRS) and DMA, several operations can be performed autonomously. For example, the timer may repeatedly trigger an ADC conversion at a given instant. When the conversion is complete, the result is moved by the DMA to RAM. When a given number of conversions have been performed, the DMA may wake up the CPU using an interrupt.

Sleep Mode is entered by executing either the "Wait for Interrupt (WFI)" or the "Wait for Event (WFE)" instruction. Please see the attached example.

2.4 Deep Sleep Mode (Energy Mode 2)

In the Deep Sleep Mode no high frequency oscillators are running, i.e. only asynchronous or low frequency peripherals are available. This mode is extremely energy efficient, and may be used for a wide range of useful cases. A few examples:

- The LCD controller drives an LCD.
- The LEUART is receiving or transmitting a byte.
- The I2C is performing address match check.
- The RTC is counting and will wake up the CPU after a programmed number of ticks.
- An Analog Comparator (ACMP) is comparing a voltage to a programmed threshold.
- GPIO is checking for transitions on an IO line.

Deep Sleep Mode is entered by first setting the SLEEPDEEP bit in the System Control Register (SCR) and then executing either the "Wait for Interrupt (WFI)" or the "Wait for Event (WFE)" instruction. Please see the attached example.

2.5 Stop Mode (Energy Mode 3)

Stop Mode differ from Deep Sleep Mode in that no oscillator (except the ULFRCO to the watch dog) is running.

Modules available in Stop Mode are:

- I2C Address Check
- Watchdog
- Asynchronous Pin Interrupt

- Analog Comparator (ACMP)
- Voltage Comparator (VCMP)
- GPIO asynchronous interrupts

Stop Mode is entered the same way as Deep Sleep Mode, except that also the low frequency oscillators are turned off. The oscillators must be disabled "manually" by software prior to entering Stop Mode.

2.6 Shut Off Mode (Energy Mode 4)

In Shut Off Mode the EFM32G is completely shut down and the current consumption is as low as 20nA. The only way to exit this energy mode is to assert the reset line or cycle the power.

Shut Off Mode is entered by writing 0x3 and then 0x2 four times to the EM4CTRL field in the EMU_CTRL register of the Energy Management Unit (EMU).

3 Clock and Oscillator Control

3.1 General

As previously mentioned, the current consumption is highly dependent on clock frequency. Selecting correct oscillator and frequency is therefore a very important aspect in low energy applications. The following sections will discuss different modes and settings for clocks and oscillators.

3.2 Enabling Oscillators / Setting Clock Source

Oscillators are enabled and disabled through the CMU_OSCENCMD register in the CMU. Each oscillator has one enable and one disable field in this register (e.g. LFXOEN and LFXODIS). The CMU_STATUS register holds two flags for each oscillator; enabled and ready (e.g. LFXOENS and LFXORDY). Each enabled-flag is set when the corresponding oscillator is turned on, whereas the ready-flag is set when the oscillator is ready to be used as a clock source.

Note

Until the ready-flag is set, the oscillator output may be incorrect, both with respect to frequency and duty-cycle. If an oscillator is used before this flag is set, behavior may become unpredictable / undefined.

Out of reset, the High Frequency RC Oscillator (HFRCO) is set as source for the Core and high-speed peripherals (HFCLK domain). For the Low Energy (LE) clock domains (A and B), no oscillator is enabled or selected.

To change the source of a clock, please do the following:

1. Enable the desired oscillator by setting the corresponding bit in the CMU_OSCENCMD register in the CMU.
2. Wait until the ready-flag of the oscillator in the CMU_STATUS register is set.
3. Change clock source to the new oscillator. For the HFCLK, this is done in the CMU_CMD register. For the LE domains, use the CMU_LFCLKSEL register.

3.3 HFRCO Band Setting

The extreme frequency tuning range of the HFRCO is a major advantage, and should be used to minimize the energy consumption of any application. The following frequencies may be set [MHz]: 1 - 7 - 11 - 14 - 21 - 28. Frequency band is selected using the BAND field in the CMU_HFRCOCTRL register.

3.4 Enabling or Disabling a Clock to a Module / Peripheral

A module's clock may be enabled or disabled using the corresponding bit in the appropriate CLKEN register in the CMU. The following registers are used for this operation:

- CMU_HFCORECLKEN0: High-speed and core modules (DMA, EBI, AES and interface to LE peripherals).
- CMU_HFPERCLKEN0: Used for the regular peripherals (e.g. TIMERS, ADC, USART, ...).
- CMU_LFACLKEN0 / CMU_LFBCLKEN0: Used for the LE peripherals (e.g. RTC, LEUART, LETIMER, ...).

Please see the EFM32 Reference Manual for details on which register and field to use for a specific module or peripheral.

3.5 Clock Prescaling

The clocks of the EFM32G may be prescaled (divided by a given factor) in the CMU. The core clock prescaling is set in the CMU_HFCORECLKDIV register, and is common to the CPU as well as the other core modules (e.g. AES, DMA, EBI). The prescaling factor for the regular peripherals is also common and set in the CMU_HFPERCLKDIV register. However, for the LE peripherals, prescaling is set individually. E.g. RTC prescaling may be set 128, whereas the LEUART1 clock is not prescaled (division factor is 1). LE peripheral prescaling is set in CMU_LFAPRESC0 and CMU_LFBPRESC0.

4 Energymodes Software Example

4.1 General

The attached example illustrates how to enter different energy modes, how to enable different oscillators, enable / disable clocks and set up prescaling. To illustrate the importance of only enabling needed clocks and oscillators, the software starts off by turning on "everything". Then, more and more energy is saved by disabling clocks and oscillators, and by entering energy modes.

Please see the attached source code for reference.

4.2 Functions

The following functions are called in the main program. After each change of settings, a few seconds of waiting is inserted to make the current consumption visible.

- **enableAllClocks()** This function turns on every oscillator on the EFM32G. The HFCLK source is set to HFXO, which is the fastest oscillator. In addition, clocks to all core modules and regular peripherals are enabled.
- **disableAllClocks()** The HFCLK source is set back to HFRCO. All unused oscillators are turned off, and clocks to unused modules / peripherals are disabled.
- **downScaleCoreClock()** The core clock frequency is reduced by selecting the 7 MHz frequency band. In addition, the core clock is prescaled with factor 4, i.e. the core frequency is 1.75 MHz.
- **enterEM1()** The clock to TIMER0 is enabled, the clock to the regular peripherals is prescaled by 512 (maximum) and TIMER0 is set to issue an interrupt when it wraps. Then Sleep Mode is entered until the timer wakes up the device.
- **enterEM2()** The clock to the interface of the LE peripherals is enabled. Then the RTC is set up to issue an interrupt after a few seconds. Then Deep Sleep Mode is entered.
- **EM4_enter()** Finally EM4 is entered. In this mode everything is completely shut down and the only way to exit this mode is by asserting the reset line or cycling the power.

5 Prime Number Software Example

5.1 General

The prime number software example is used for current consumption benchmarking. It sets up the EFM32 to forever execute a prime calculation algorithm from flash.

5.2 Setup

The code makes the EFM32 run from the High Frequency Crystal Oscillator (HFXO) and disables all other oscillators. Then the Suppressed Conditional Branch Target Prefetch (SCBTP) option in the MSC module is enabled before the prime calculation algorithm is started.

Please see the attached source code for reference.

5.3 Current Consumption

During code execution the EFM32 consumes about 5.7mA. Dividing by 32 (i.e. the HFXO frequency) yields a current consumption slightly less than 180uA/MHz which is a clear evidence that the EFM32 is in fact the worlds most energy friendly microcontroller!

6 Revision History

6.1 Revision 1.03

2012-04-20

Adapted software projects to new peripheral library naming and CMSIS_V3.

6.2 Revision 1.02

2012-03-13

Fixed makefile-error for CodeSourcery projects.

6.3 Revision 1.01

2010-11-16

Changed example folder structure, removed build and src folders.

Added chip-init function.

6.4 Revision 1.00

2010-09-20

Initial revision.

A Disclaimer and Trademarks

A.1 Disclaimer

Energy Micro AS intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Energy Micro products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Energy Micro reserves the right to make changes without further notice and limitation to product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Energy Micro shall have no liability for the consequences of use of the information supplied herein. This document does not imply or express copyright licenses granted hereunder to design or fabricate any integrated circuits. The products must not be used within any Life Support System without the specific written consent of Energy Micro. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Energy Micro products are generally not intended for military applications. Energy Micro products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons.

A.2 Trademark Information

Energy Micro, EFM32, EFR, logo and combinations thereof, and others are the registered trademarks or trademarks of Energy Micro AS. ARM, CORTEX, THUMB are the registered trademarks of ARM Limited. Other terms and product names may be trademarks of others.

B Contact Information

B.1 Energy Micro Corporate Headquarters

Postal Address	Visitor Address	Technical Support
Energy Micro AS P.O. Box 4633 Nydalen N-0405 Oslo NORWAY	Energy Micro AS Sandakerveien 118 N-0484 Oslo NORWAY	support.energymicro.com Phone: +47 40 10 03 01

www.energymicro.com

Phone: +47 23 00 98 00

Fax: + 47 23 00 98 01

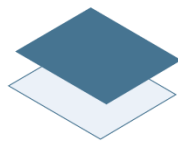
B.2 Global Contacts

Visit **www.energymicro.com** for information on global distributors and representatives or contact **sales@energymicro.com** for additional information.

Americas	Europe, Middle East and Africa	Asia and Pacific
www.energymicro.com/americas	www.energymicro.com/emea	www.energymicro.com/asia

Table of Contents

1. Energy Saving	2
1.1. General	2
1.2. Turn off Unused Modules / Peripherals	2
1.3. Disable Clocks to Unused Modules / Peripherals	2
1.4. Reduce Clock Frequency	2
1.5. Lower the Operating Voltage	3
2. Energy Modes	4
2.1. General	4
2.2. Run Mode (Energy Mode 0)	4
2.3. Sleep Mode (Energy Mode 1)	4
2.4. Deep Sleep Mode (Energy Mode 2)	4
2.5. Stop Mode (Energy Mode 3)	4
2.6. Shut Off Mode (Energy Mode 4)	5
3. Clock and Oscillator Control	6
3.1. General	6
3.2. Enabling Oscillators / Setting Clock Source	6
3.3. HFRCO Band Setting	6
3.4. Enabling or Disabling a Clock to a Module / Peripheral	6
3.5. Clock Prescaling	7
4. Energymodes Software Example	8
4.1. General	8
4.2. Functions	8
5. Prime Number Software Example	9
5.1. General	9
5.2. Setup	9
5.3. Current Consumption	9
6. Revision History	10
6.1. Revision 1.03	10
6.2. Revision 1.02	10
6.3. Revision 1.01	10
6.4. Revision 1.00	10
A. Disclaimer and Trademarks	11
A.1. Disclaimer	11
A.2. Trademark Information	11
B. Contact Information	12
B.1. Energy Micro Corporate Headquarters	12
B.2. Global Contacts	12



ENERGY[®]
micro

*Energy Micro AS
Sandakerveien 118
P.O. Box 4633 Nydalen
N-0405 Oslo
Norway*

www.energymicro.com