Lab 01 Tips and General Information

Course : ET032G, Elektroteknik GR (A), Mikrodatorteknik at Mittuniversitetet, Sweden.

Extracted by : Muhammad Imran, Mid Sweden University, Sweden.

RTCCTRL – RTC Control register

Bit	7	6	5	4	3	2	1	0
+0x03	–	–	–	–	RTCSRC[2:0]			RTCEN
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

- **Bit 7:4 – Reserved**

These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written.

- **Bit 3:1 – RTCSRC[2:0]: RTC Clock Source**

These bits select the clock source for the real-time counter according to Table 7-4.

Table 7-4. RTC clock source selection.

RTCSRC[2:0]	Group Configuration	Description
000	ULP	1kHz from 32kHz internal ULP oscillator
001	TOSC	1.024kHz from 32.768kHz crystal oscillator on TOSC
010	RCOSC	1.024kHz from 32.768kHz internal oscillator
011	—	Reserved
100	—	Reserved
101	TOSC32	32.768kHz from 32.768kHz crystal oscillator on TOSC
110	RCOSC32	32.768kHz from 32.768kHz internal oscillator
111	EXTCLK	External clock from TOSC1

- **Bit 0 – RTCEN: RTC Clock Source Enable**

Setting the RTCEN bit enables the selected RTC clock source for the real-time counter.

When using real time counter, you can select the clock source by using RTCCTRL register. You need to set RTCSRC values, and enable RTCEN. (Some students forget to enable RTCEN and the didn't get count values for RTC)

Example: if I am using RCOSC clock with 1.024 Khz frequency, I have to write register value as

```
CLK.RTCCTRL = CLK_RTCSRC_RCOSC_gc; //   (iox128b1.h)
CLK.RTCCTRL=CLK_RTCEN_bm;          //* Clock Source Enable bit mask. */ (iox128b1.h)
```

Or alternatively

CLK.RTCCTRL = 0x03;

Remember to set `RTC.CTRL = 0xSome value; otherwise default is 00`

Some more TIPS.
You can use RTC.INTFLAGS and RTC.COMP for this task.
In RTC.INTFLAGS the COMPIF can be accessed as (RTC.INTFLAGS & 0x02)
And this flag would be generated when Comparator register (COMP) values equal to real time counter value (RTC.CNT).

INTFLAGS – Interrupt Flag register

Bit	7	6	5	4	3	2	1	0
+0x03	–	–	–	–	–	–	COMPIF	OVFIF
Read/Write	R	R	R	R	R	R	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

- **Bit 7:2 – Reserved**

These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written.

- **Bit 1 – COMPIF: Compare Match Interrupt Flag**

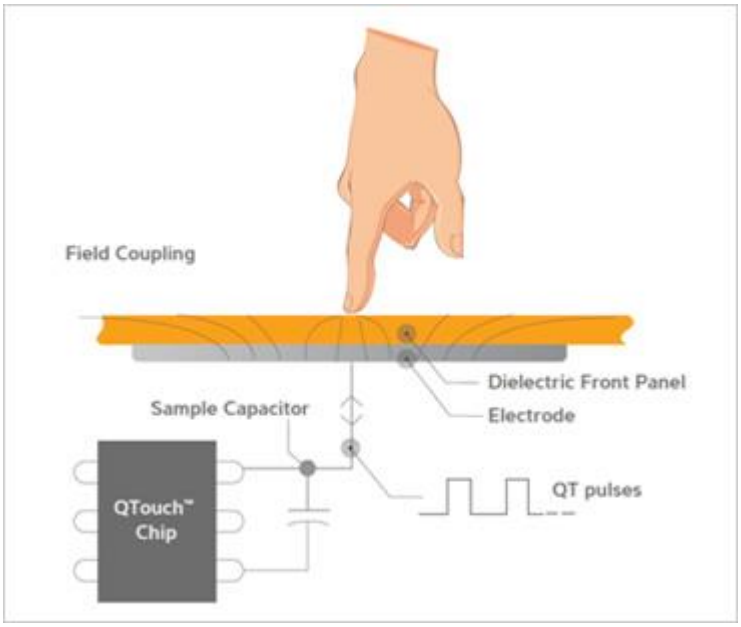
This flag is set on the next count after a compare match condition occurs. It is cleared automatically when the RTC compare match interrupt vector is executed. The flag can also be cleared by writing a one to its bit location.

Think of this register also
RTC.CTRL = 0x01; // Dividing RCT by 1

General information about the Xmega board

ATxmega128B1
Low power, high performance 8/16-bit AVR microcontroller featuring 128KB self-programming flash program memory, 8KB boot code section, 8KB SRAM, 2048-Byte EEPROM, 2-channel DMA controller, 4-channel event system, USB Full speed Device, 4x40 segment LCD controller and up to 32 MIPS throughput at 32MHz. The ATxmega B3 series features 100-pin packages.

QTouch®
The QTouch® devices are charging a sense electrode of unknown capacitance to a known potential. The electrode is typically a copper area on a printed circuit board. The resulting charge is transferred into a measurement circuit. By measuring the charge after one or more charge-and-transfer cycles, the capacitance of the sense plate can be determined. Placing a finger on the touch surface introduces external capacitance that affects the flow of charge at that point. This registers as a touch. QTouch® microcontrollers can also be set up to detect the proximity of a finger, rather than absolute touch.



Signal processing in the decision logic makes QTouch® robust and reliable. False triggering due to electrostatic spikes or momentary unintentional touch or proximity is eliminated.

http://www.atmel.com/products/touchsolutions/bsw/qtouch.aspx

programming
The ATxmega128B1 can be programmed through the USB interface. This can be accomplished using the USB boot loader that is preprogrammed in the device.
The boot loader is evoked by shorting **pin 6 on J1 to GND** before applying power to the board. A 100 mil jumper can be used. Programming is performed through the FLIP plug-in in AVR Studio (which can also be started as a standalone application).
NOTE If any external programming tool is used on the ATxmega128B1, the boot loader might be erased, and it will not be possible to program the device through the USB interface. In this case, the boot loader has to be restored (available on the Atmel website) with an external programming tool.

The Atmel ATxmega128B1 that comes with the XMEGA-B1 Xplained kit is preprogrammed.
The device also features a USB boot loader (see the application note, "AVR1916: USB DFU Boot Loader for ATxmega") for the self-programming of the microcontroller. The boot loader can be started by shorting pin 6 of header J1 to GND while applying power to the board. The boot loader can be used with either FLIP or the batchISP command line tool (in the FLIP package).
http://www.atmel.com/Images/doc8397.pdf

The kit is powered via the USB connector, which offers two options to power it: connect the kit to a PC with a USB cable or to a 5V USB power supply (AC/DC adapter).

Power

The 5V supply voltage is regulated down to 3.3V with an onboard LDO regulator, which provides power to the entire board. The ATxmega128B1 is powered by 3.3V, but if operation at a lower voltage (1.8V min.) is desired, some modifications to the board are needed. This includes replacing the regulator with one that delivers the desired voltage and rerouting the power to the device (see schematic for explanation).

Connectors

The Atmel XMEGA-B1 Xplained board also has a USB 2.0 mini B connector.
The XMEGA-B1 Xplained board has four 10-pin, 100mil headers. Two of the headers provide a fixed communication interface (J1 and J4). One header provides analog functionality (J2), and the remaining header (J3) provides general purpose digital I/O.

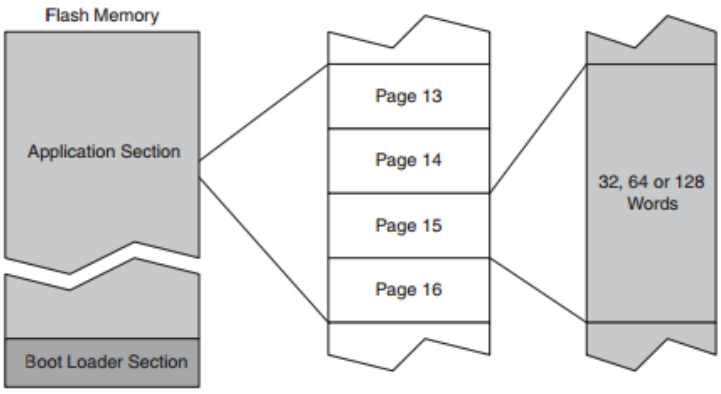
Analog input

The Atmel XMEGA-B1 Xplained offers two sensors: a temperature sensor and a light sensor. In single-ended mode, it can also measure two analog inputs, one from the on-board potentiometer and one from a source that is external to the board.

FLASH memory

The Flash memory is divided into two sections, one Application section and one Boot Loader section. The Application section contains the main code for the application, while the Boot Loader section contains the code for the actual Self-programming. The SPM instruction can only be executed from the Boot Loader section. (Note: The Boot Loader section can also be used for ordinary application code.)
The Flash memory is divided into pages containing 32, 64, or 128 words each. The usage of pages is explained later. The entire memory span, both Application and Boot Loader sections, is divided into pages. For instance, a device with 8 KB of Flash and page size of 32 words (64 bytes) will therefore have a total of 128 pages. The memory organization is shown in Figure 1.

Figure 1. Memory Organization



http://www.atmel.com/images/doc1644.pdf