APPLICATION NOTE

DUTY CYCLE AND POWER OPTIMIZATION

Version 1.0

Document	Type	Application Note
Document	Status	PRELIMINARY
Document	Version	Version 1.0
Product		duty cycle and power optimization

Table of Contents

1. Duty Cycle

1.1. General

Consider a battery powered device A, which should receive data from a second device B from time to time. To reduce power consumption, device A switches its receiver on for only a short while, checking if there is any RF activity, and returns to sleep if there is no data to receive. The ratio of the time t_{on} the receiver is powered on, to the time t_{off} the receiver is powered off is called duty cycle. If for example device A is powered off for 1 second, powered on for 1 millisecond, powered off for 1 second and so on, its duty cycle is 1:1000.

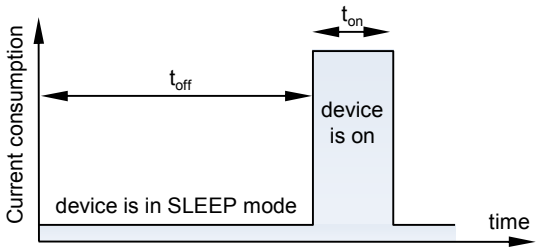

Figure 1 illustration of duty cycle

Hardware description

The hardware used in this document consists of a AX5051 evaluation board that is plugged into a suitable controller board, containing a PIC16f886 microcontroller from Microchip. The microcontroller is required to configure and control the AX5051 transmitter.

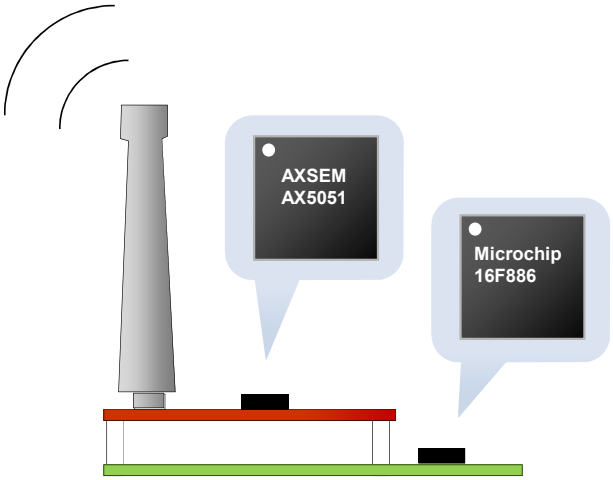

Figure 2 Hardware used for duty cycle measurements

The PIC16f886 has a feature called 'Ultra low power wake up' (ULPWU). With this feature, an external condensator is being charched when the system is running, and discharched when the system is in sleep mode. When the condensator voltage reaches a defined level, the processor is being woken up by an interrupt. This feature is very suitable for duty cycle applications.

Software description

The software demonstrates how duty cycle can be implemented in software. The software can be compiled for the transmitter or for the receiver. When defining CONST_TX, the source code will be compiled for the transmitter, if CONST_RX is defined, the source code will be compiled for the receiver. Make sure, only CONST_TX or CONST_RX is defined at one time. The transmitter sends a 1.2s long preamble every time a button is pressed. After the data is transmitted, it enters sleep mode and consumes only 650 nA. The receiver is in sleep mode too, but it wakes up every second. First the crystal oscillator needs to be switched on. This can take 1.7ms. After the crystal oscillator is running, the receiver is switched on and the program checks if the RSSI is above the defined threshold. If this is the case, 5 bytes of data are received, to check if there is a valid preamble present. If the transmitter is sending data at that time, the receiver will be switched on, until the data is received. This will take max. 1.2 seconds. After this the receiver is switched off, and the controller enters sleep mode again.

The software was compiled and tested with sdcc. To compile the code, please install the gputils (http://gputils.sourceforge.net), and modify the makefile according to your system settings. For downloading and debugging, the MPLAB IDE from Microchip was used together with the MPLAB ICD2 programmer/debugger.

Important

- Make sure, that when porting the software to another platform, calling delay(x) will keep the processor busy for x ms. The delay must be exact, otherwise some parts of the program may not work correct (e.g. starting the crystal oscillator).
- As mentioned above, CONST_RX and CONST_TX must be defined before compiling the code.

The PIC16f886 only has a hardware stack. Because of that the number of function calls is limited to only a few levels. Because of this fact, the whole state machine is implemented in the main loop.

1.2. Duty cycle with AX5051

All results in this document are obtained by using the AX5051 as a RF receiver, which is being switched on every second. At first the RSSI is checked against a predefined threshold to determine if there is any RF activity present. If not, the AX5051 is switched off immediately by the microcontroller. Otherwise 5 bytes have to be received to determine if a valid preamble

is present. To wake up the receiver, the transmitting unit needs to transmit a preamble which is longer than the time t_{off}.

There are three stages used: Sleep, crystal startup and receive:

Sleep	Microcontroller and receiver are switched off for ~966ms. To wake up the microcontroller from sleep, a low power oscillator is required. The power consumption of the oscillator and the microcontroller in sleep mode is assumed to be 1uA. The current consumption of the AX5051 in power down mode is measured to be 0.5uA.
Crystal startup	When the AX5051 comes out of power down mode, it takes some time until the crystal oscillator settles on its nominal frequency of 16MHz. The time until the crystal is fully settled, was measured to be 1.7ms, but it should be possible to reduce the start up time even more. The current drawn by the controller was measured while it was running at 8MHz, which is way too much (it could be set to sleep during this time). But we decided not to change this value, to be on the safe side.
Receive	After the crystal has settled, the receiver is being switched on. At first the RSSI is calculated to see if it is above the defined threshold. If not, sleep mode is entered immediately. Otherwise 5 bytes are received, to check if a valid preamble is present. All diagrams in this document assume that no valid preamble is present (which is the normal case).

To provide a general impression, we measured both the currents drawn by the AX5051 and the microcontroller. Note that the microcontroller's current consumption is device specific and must be evaluated individually.

Szenario 1: 250kbit/s

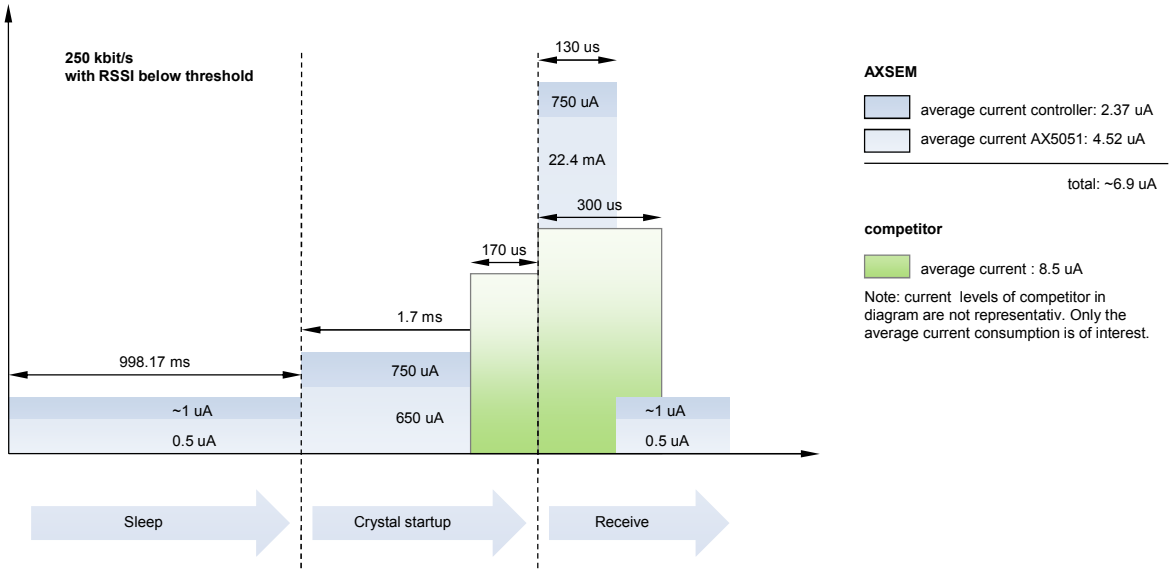

Figure 3 Current consumption with 250kbit/s, RSSI below threshold

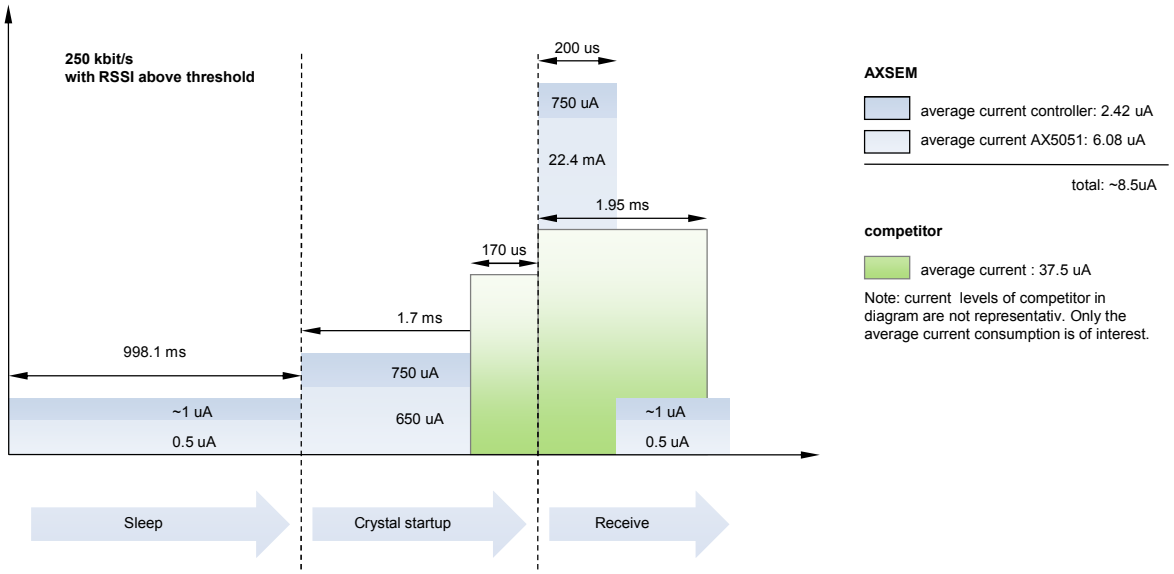

Figure 4 Current consumption with 250kbit/s, RSSI above threshold

Szenario 2: 10kbit/s

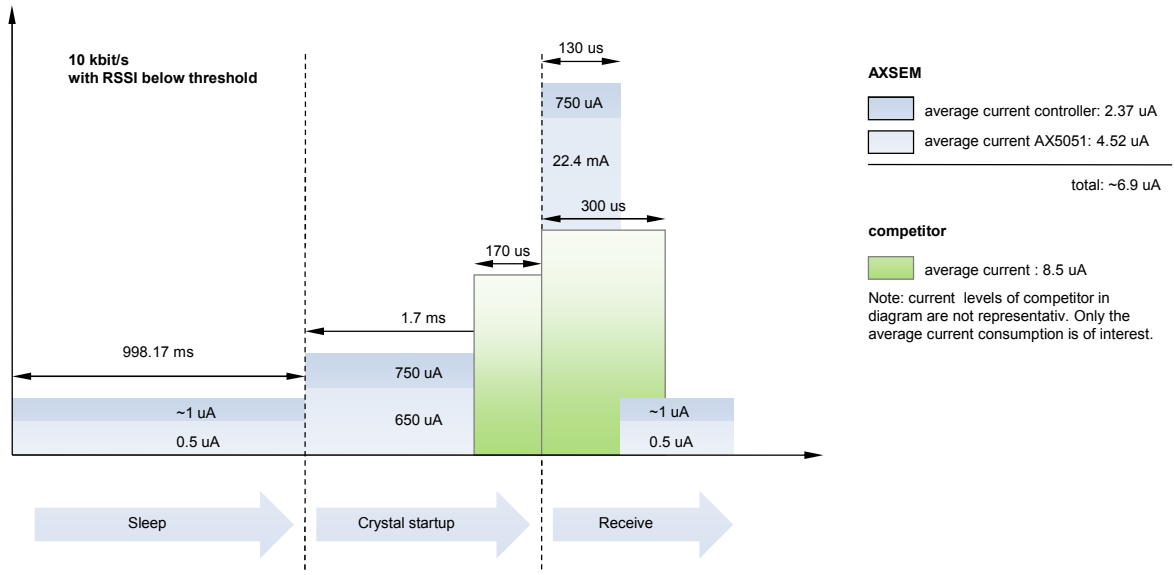

Figure 5 Current consumption with 10kbit/s, RSSI below threshold

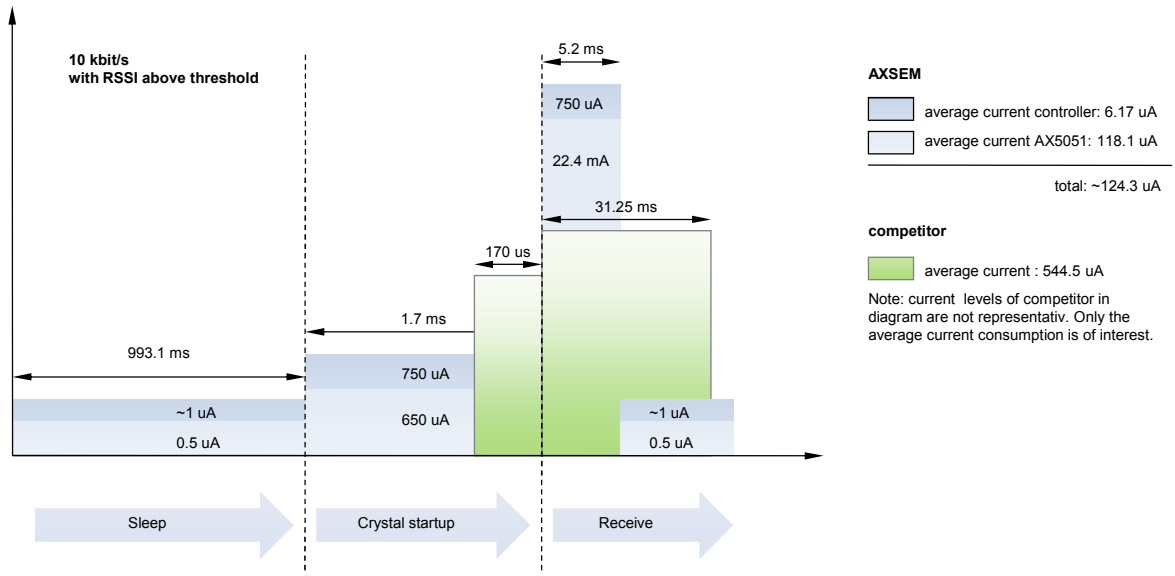

Figure 6 Current consumption with 10kbit/s, RSSI above threshold

Conclusion

The current used during receive mode has main impact on the average current consumption (this is especially true for low bitrates). It is therefore very important to keep the time when the receiver is switched on, as short as possible. Even if the AX5051 does not have built in WOR (Wake On Radio) capabilities, the overall current consumption (including the microprocessors current) is low compared to competitors. This is achieved by using the great flexibility provided by the AX5051 that makes it possible to reduce the time, the receiver is switched on, to a minimum. With the flexibility of the AX5051, one is free to optimize for own requirements, and is not bound to a specific packet format.

2. Appendix

2.1. References

AX5051 downloads

Datasheet
http://www.axsem.com/web/documents/reguserspace/ax5051/AX5051_DS_V1_5.pdf

Programming manual
http://www.axsem.com/web/documents/reguserspace/ax5051/AX5051_PM_V1_3.pdf

AX5051 HARDWARE

Find local distributors for AX5051 transceiver at www.axsem.com
Order development kit AX5051-DVK at www.axsem.com

PIC16F886 MICROCONTROLLER

Datasheet for the microcontroller used
http://ww1.microchip.com/downloads/en/DeviceDoc/41291E.pdf

2.2. Contact Information

For further information please visit our website or feel free to contact AXSEM AG

AXSEM AG
Oskar-Bider-Strasse 1
CH-8600 Dübendorf
SWITZERLAND

Phone +41 (44) 882 17 07
Fax +41 (44) 882 17 09
Email sales@axsem.com
www.axsem.com