AN4193
Application note

Low duty cycle operation with the SPIRIT1 transceiver

Introduction

The SPIRIT1 is a very low power RF transceiver, intended for RF wireless applications in the sub-1 GHz band. It is designed to operate both in the license-free ISM and SRD frequency bands at 169, 315, 433, 868, 915 and 920 MHz. This application note describes how to use the low duty cycle (LDC) mode in order to further optimize power consumption during reception and transmission operation.

Contents

1 Overview of low duty cycle mode

The SPIRIT1 includes hardware support to manage low duty cycle operation that can be used in two ways:

1. To reduce average power consumption during receive operations
2. To build a synchronized star network where both transmitter and receiver can sleep periodically to reduce average power consumption.

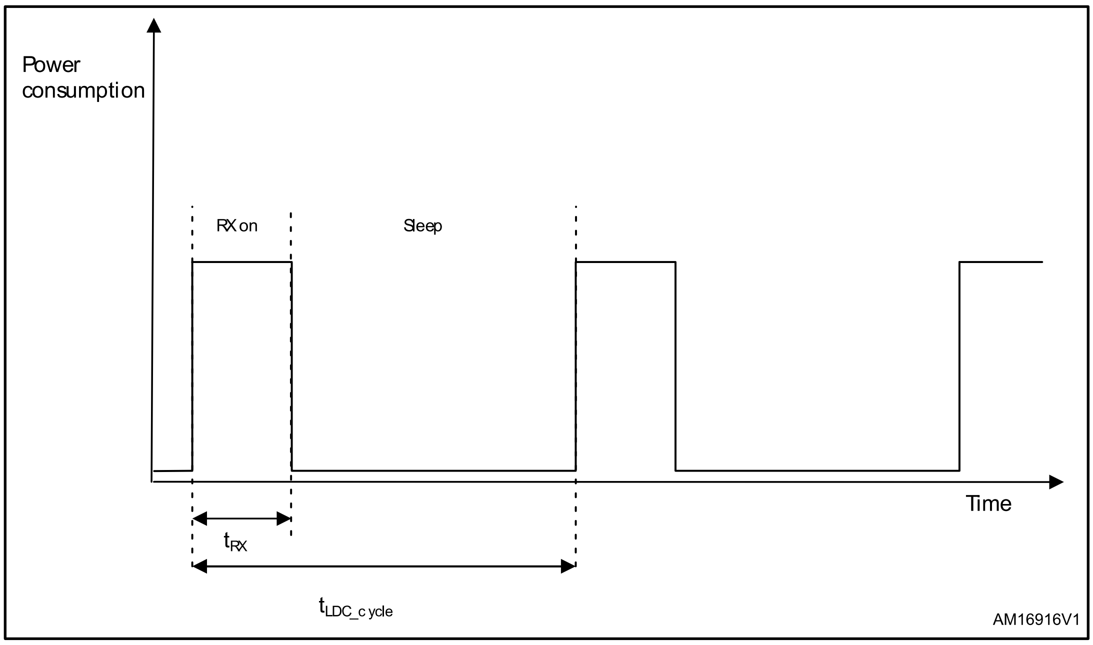
LDC mode is essentially controlled by two timers:

1. The LDC timer which defines the window where the duty cycle operation take place
2. The RX_TIMEOUT timer which defines the amount of time that the receiver is active.

1.1 LDC reception mode

In this mode, the SPIRIT1 can periodically switch on the receiver to check if a packet is being transmitted, and then go back to sleep in order to save power. This mode allows the reduction of average reception power consumption, while allowing the device to continue receiving packets under certain conditions.

Figure 1: LDC reception mode



In this mode, the user defines an operation cycle (t_{LDC_cycle} in *Figure 1: "LDC reception mode"*) whose max length is about 2 seconds with a step of 29 µs. In the t_{LDC_cycle} period, a period is also defined where the receiver is automatically turned on for packet reception (t_{RX} in *Figure 1: "LDC reception mode"*). In this mode, users can set the duration of the t_{LDC_cycle} and the t_{RX}, in order to reduce average power consumption based on the target application. The average current consumption, in mA for the SPIRIT1, is calculated using:

Equation 1

$$RX_{consumption\_average} = \frac{t_{RX}}{t_{LDC\_cycle}} \cdot 9 + \frac{t_{LDC\_cycle} - t_{RX}}{t_{LDC\_cycle}} \cdot 0.00085 \text{ mA}$$

The following example shows how to set up the SPIRIT1 with a specific duty cycle. Assuming that the application is using a t_{LDC_cycle} of 100 ms and a t_{RX} of 10 ms, the average RX current consumption will be:

Equation 2

$$\frac{10}{100} \cdot 9 + \frac{100 - 10}{100} \cdot 0.00085 = 0.9 \text{ mA}$$

The registers to configure the SPIRIT1 timers are as described in *Table 1: "LDCR and RX_TIMEOUT prescaler and counter settings"*.

Table 1: LDCR and RX_TIMEOUT prescaler and counter settings

Register name	Register address	Register value	Comment
LDCR_PRESCALER	0x55	0x26	Assuming RCO clock of 34.7 kHz
LDCR_COUNTER	0x56	0x58	
RX_TIMEOUT_PRESCALER	0x53	0x2A	Assuming 26 MHz digital domain clock
RX_TIMEOUT_COUNTER	0x54	0x04	

1.1.1 RX timeout using sync detection

In this mode, the RX timeout is stopped when a valid SYNC word, which marks the beginning of the packet, is received and the receiver is expected to receive a complete packet. When the receiver is working in LDC mode, the transmitter should take this into account, retransmitting the packet for a period of time at least equal to tLDC_cycle, so that it is certain that the receiver has the opportunity to catch the SYNC word. In addition, the tRX needs to be selected according to the length of the packet to be received in order to ensure that a valid sync word is detected during the RX window. The tRX should be equal to the packet period plus the length of the preamble. The packet period can be defined as follows:

Equation 3

$$t_{packet\_period} = (l_{preamble} + l_{sync} + l_{packet}) \cdot datarate + t_{interpacket}$$

where datarate is the bit rate at which the information is sent and the other equation parameters are according to *Table 2*.

Table 2: Packet period key parameters

Name	Description
$l_{preamble}$	Length of the preamble in bits
l_{sync}	Length of the sync in bits
l_{packet}	Length of the packet in bits
$t_{interpacket}$	Delay between two consecutive transmissions; it is the cumulative time of interrupt response time, reload of the TX fifo, send TX command, ready to TX transition.

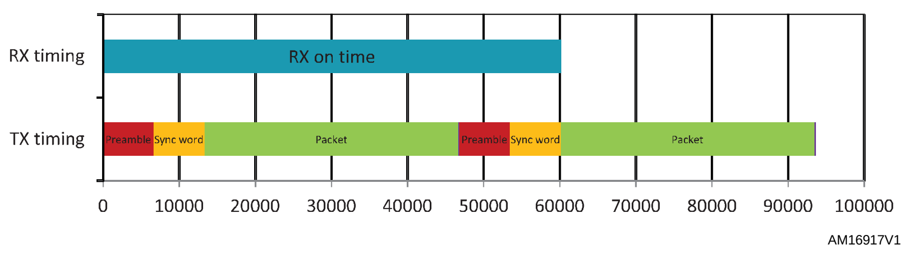
Finally, the parameter tRX can be set based on the following equation:

Equation 4

$$t_{RX} \geq t_{packet\_period} + (l_{preamble} + l_{sync}) \cdot datarate$$

This ensures that a node doing RX in LDC mode is able to receive a packet sent by a transmitter, which repeats the message for a certain period of time. *Figure 2* shows the minimum RX ON time to ensure a valid sync is captured.

Figure 2: RX on minimum time for RX timeout on sync



The transmitter needs to repeat the packet to be transmitted for at least tLDC_cycle. From this we can see how the consumption of the receiver is reduced at the expenses of increasing the consumption on the transmitter side and increasing the latency of response. This may be perfectly acceptable and desirable, for example, in applications where the receiver is battery-operated and the transmitter is powered by mains.

1.1.2 RX timeout using RSSI

In this mode, the receiver timeout is stopped upon detection of signal energy above a certain user defined threshold. This mode will further reduce average power consumption by decreasing the time when receiver must be on. On the other hand this mode will add some further burden on the micro which needs to check that a valid message is received within a user defined timeout. In this mode, the SPIRIT1 cycles between RX active and sleep as described in *Figure 1*; the only difference is that the minimum time tRX will be much smaller than that described in Section 1.1.1. The time to measure RSSI of the incoming signal will vary according to the RX filter bandwidth shown in *Table 3: "RSSi detecion time"*.

Table 3: RSSi detecion time

RXfilter min (kHz)	RXfilter max (kHz)	RSSI detection time (µS)	Comment
4.2	7.0	1800	7
7.0	14.0	950	6
14.0	28.0	550	5
28.0	56.1	346	4
56.1	112.3	280	3
112.3	224.7	175	2
224.7	450.9	90	1
450.9	800.1	34	0

Since in this mode the RX active time can be very short, it may be worthwhile to also take into account the time to move from the SLEEP state to the RX state, which is in the order of 120 µs. An equation that approximates the average current consumption in this mode is as follows:

Equation 5

$$RX_{consumption\_average} = \frac{120^{-6} + t_{RX}}{t_{LDC\_cycle}} \cdot 9 + \frac{t_{LDC\_cycle} - t_{RX} - 120^{-6}}{t_{LDC\_cycle}} \cdot 0.00085\ \text{mA}$$

The equation assumes that there is an average current consumption of 9 mA during the transition from SLEEP to RX. It is important to understand that this mode also requires some cooperation from the microcontroller driving the SPIRIT1. In particular, since the RX timeout is stopped upon detection of an RSSI value above a certain threshold, two cases can occur:

1. A valid packet is detected and the reception is completed and signaled to the micro.
2. No packet is detected and the SPIRIT1 can stay indefinitely in the RX state.

In order to avoid case 2, it is suggested that upon detection of the RSSI signal, the microcontroller starts a timer and then aborts reception if a valid packet is not received within a user-defined timeout.

1.1.3 Current consumption figures

A set of measurements have been taken using an LDC cycle of 100 ms and a variable duty cycle.The values have been taken using a Keysight N6782A source and measure unit with ‘Seamless Current Ranging' and are plotted in Figure 3. The actual values with comparison to predicted values are shown in *Table 4*.

Figure 3: Average RX current consumption vs. duty cycle

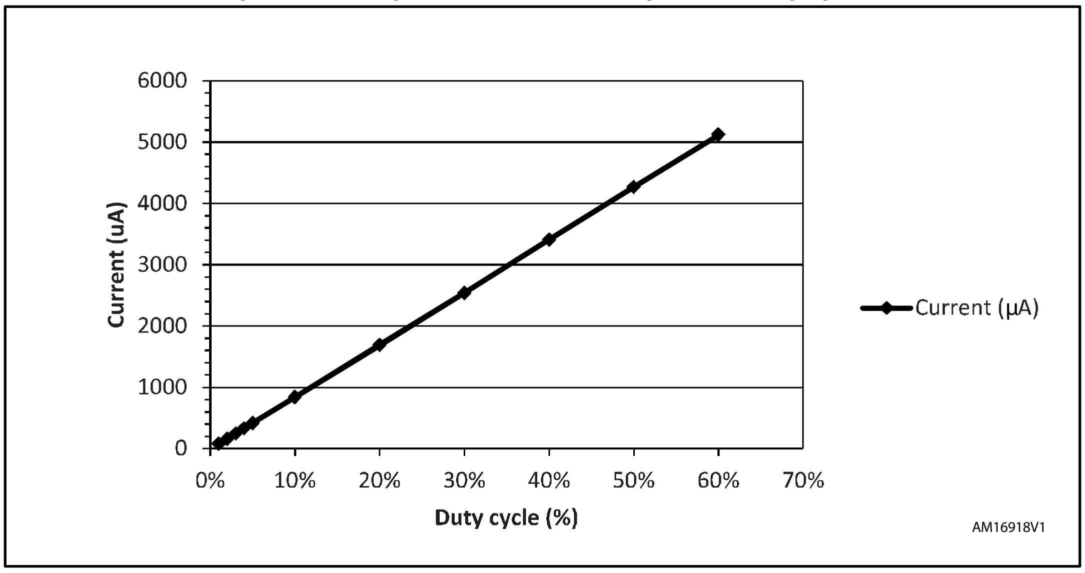
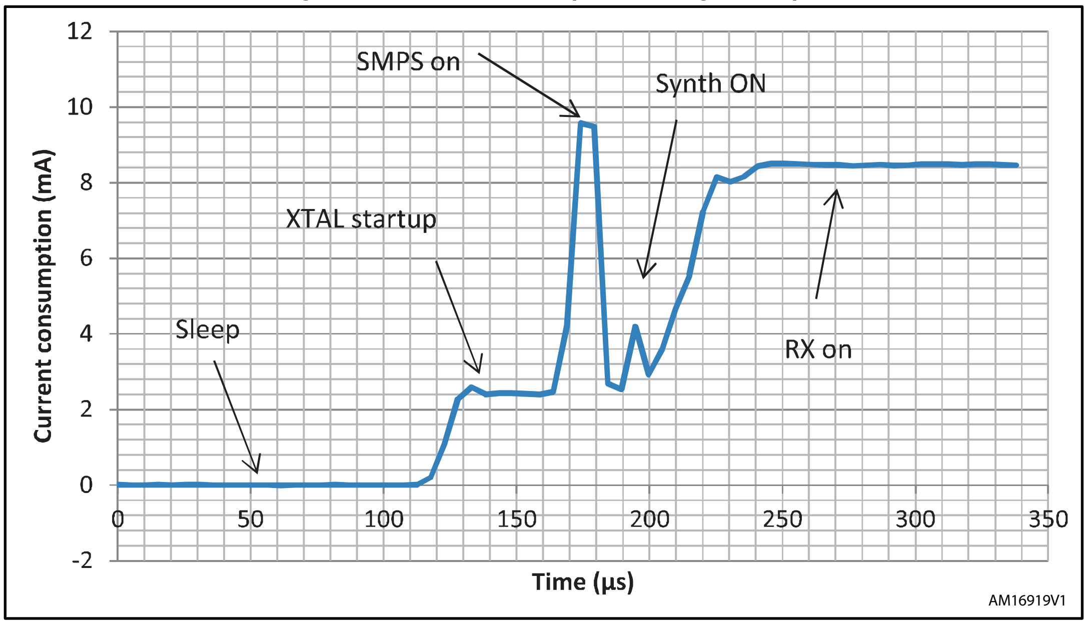


Table 4: Current consumption data with different duty cycles

Duty cycle (%)	Current (µA)	Predicted value (µA)
1%	80.0372	90.84
2%	162.9491	171.83
3%	249.9657	257.32
4%	332.9219	342.82
5%	419.8115	428.31
10%	841.9774	855.77
20%	1690.3268	1710.68
30%	2541.7257	2565.60
40%	3410.6563	3420.51
50%	4267.8439	4275.43
60%	5120.7135	5130.34

Figure 4: "Current consumption during wakeup" shows the dynamic power consumption during the transition from sleep to RX.

Figure 4: Current consumption during wakeup



2 Programming tips

This section provides some indication of how to program the SPIRIT1 to perform RX in LDC mode.

Assuming that the transceiver has been properly initialized, the following steps are required to activate the LDC mode in reception using the SPIRIT C driver:

1. SpiritTimerSetWakeUpTimerMs (LDC_CYCLE); // Specify the LDC cycle time in ms
2. SpiritTimerSetRxTimeoutMs (RX_TIMEOUT); // Specify RX timeout in ms
3. SpiritTimerLdcrMode(S_ENABLE); // Enable LDC mode
4. SpiritCmdStrobeRx(SPIRIT_CMD_RX); // Start RX

The same commands can be performed as a set of registers access as follows (assuming an LDC cycle of 100 ms and a RX timeout of 1 ms and a crystal of 50 MHz):

1. WriteRegister(0x55,0x26), WriteRegister(0x56,0x58) # Set LDC cycle time
2. WriteRegister(0x53,0x09), WriteRegister(0x54,0x02) # Set Rx timeout
3. WriteRegister(0x50, ReadRegister(0x50) | 1) # Set bit 0 of register 50 to 1 to activate LDC mode
4. SendStrobe (0x61)

3 Reference

Reference

1. SPIRIT1 datasheet

4 Revision history

Table 5: Document revision history

Date	Revision	Changes
07-Jan-2013	1	Initial release.
27-Feb-2015	2	Updated: Section 1.1.3