WIRELESS M-BUS SOFTWARE IMPLEMENTATION

1. Introduction

This application note describes the Silicon Labs implementation of Wireless M-Bus using a Silicon Labs C8051 MCU and EZRadioPRO®. Wireless M-bus is a European Standard for meter-reading applications using the 868 MHz frequency band.

1.1. Stack Layers

Wireless M-Bus uses the 3-layer IEC model, which is a subset of the 7-layer OSI model (see Figure 1).
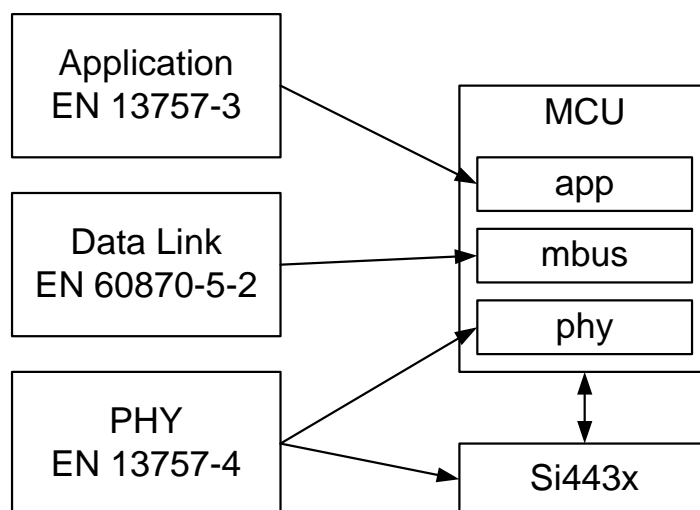

Figure 1. Stack Layers

The Physical (PHY) layer is defined in EN 13757-4. The physical layer defines how the bits are encoded and transmitted, the RF modem characteristics (chip rate, preamble, and synchronization word), and RF parameters (modulation, center frequency, and frequency deviation).

The PHY layer is implemented using a combination of hardware and firmware. The EZRadioPRO performs all of the RF and modem functions. The EZRadioPRO is used in FIFO mode with the packet handler. The MbusPhy.c module provides SPI interface, encoding/decoding, block read/write, and packet handling and manages the transceiver states.

The M-Bus Data link layer is implemented in the MbusLink.c module. The M-Bus Application Programming interface consists of public functions that may be called from the application layer in the main thread. The MbusLink module also implements the Data Link Layer. The Data link layer will format and copy data from the application TX buffer to the MbusPhy TX buffer, adding the required headers and CRCs.

The Application layer itself is not part of the M-bus firmware. The application layer defines how a wide variety of data is to be formatted for transmission. Most meters only need to transmit one or two types of data. Adding a large amount of code to accommodate any kind of data to the meter would add unnecessary code and cost to the meter. It might be feasible to implement a library or a header file with an exhaustive list of data types. However, most metering customers know exactly what kind of data they need to transmit and can refer to the standard for formatting details. A universal reader or sniffer might implement a complete set of application data types on the PC GUI. For these reasons, the application layer is implemented using example applications for a meter and reader.

AN451

1.2. Required Standards

1.2.1. EN 13757-4

EN 13757-4

Communication system for meters and remote reading of meters

Part 4: Wireless meter readout

Radio meter reading for operation in the 868 MHz to 870 MHz SRD band

1.2.2. EN 13757-3

Communication system for meters and remote reading of meters

Part 3: Dedicated application layer

1.2.3. IEC 60870-2-1:1992

Telecontrol equipment and systems

Part 5: Transmission protocols

Section 1:Link transmission procedure

1.2.4. IEC 60870-1-1:1990

Telecontrol equipment and systems

Part 5: Transmission protocols

Section 1: Transmission frame formats

1.3. Definitions

- **M-Bus**—M-Bus is a wired standard for meter reading in Europe.
- **Wireless M-Bus**—Wireless M-Bus for meter reading applications in Europe.
- **PHY**—Physical Layer defines how data bits and bytes are encoded and transmitted.
- **API**—Application Programmer interface.
- **LINK**—Data Link Layer defines how blocks and frames are transmitted.
- **CRC**—Cyclic Redundancy Check.
- **FSK**—Frequency Shift Keying.
- **Chip**—Smallest unit of transmitted data. One data bit is encoded as multiple chips.
- **Module**—A C code source .c file.

SILICON LABS

2. M-Bus PHY Functional Description

2.1. Preamble Sequence

The Preamble sequence specified by the M-bus specification is an integer number alternating zeros and ones. A one is defined as the higher frequency, and a zero is defined as the lower frequency.

n x (01)

The Preamble options for the Si443x is an integer number of nibbles consisting of alternating ones and zeros.

n x (1010)

A preamble with an extra leading one would not be a problem, but, then, the synchronization word and payload would be misaligned by one bit.

The solution is to invert the entire packet by setting the *eninv* bit in the *Modulation Control 2* register (0x71). This will invert the preamble, sync word, and TX/RX data. As a consequence, the data should be inverted when writing the TX data or reading the RX data. Also, the synchronization word is inverted before writing to the Si443x Synchronization Word registers.

2.2. Synchronization Word

The synchronization word required by EN-13757-4 is either 18 chips for Mode S and Mode R or 10 chips for Mode T. The synchronization word for the Si443x is 1 to 4 bytes. However, since the synchronization word is always preceded by the preamble, the last six bits of the preamble can be considered part of the synchronization word; so, the first synchronization word is padded by three repetitions of a zero followed by a one. The synchronization word is complemented before writing to the Si443x registers.

Table 1. Synchronization Word for Mode S and Mode R

EN 13757-4	00	01110110	10010110	binary
	00	76	96	hex
pad with (01) x 3	01010100	01110110	10010110	binary
	54	76	96	hex
complement	10101011	10001001	01101001	binary
	AB	89	69	hex
	⬇	⬇	⬇	
	SYNCH	SYNCH	SYNCH	
	WORD	WORD	WORD	
	3	2	1	

Si443x Register

SILICON LABS

Table 2. Synchronization Word for Mode T Meter to Other

EN 13757-4	00	00111101	binary
	00	3D	hex
pad with (01) x 3	01010100	00111101	binary
	54	3D	hex
complement	10101011	11000010	binary
	AB	C2	hex
	↓	↓	

SYNCH WORD 3	SYNCH WORD 2
Si443x register	

2.3. Transmit Preamble Length

The minimum preamble is specified for four different operating modes. It is acceptable to have a preamble longer than specified. Subtracting six chips for the preamble gives the minimum number of chips for the Si443x preamble. The implementation adds two extra nibbles of preamble in all short preamble modes to improve preamble detection and interoperability. The preamble on Mode S with a long preamble is very long; so, the minimum preamble is used. The preamble length in nibbles is written to the Preamble Length (0x34) register. The preamble length register determines the preamble upon transmission only. The minimum specification and preamble length settings are summarized in Table 3.

Table 3. Transmit Preamble Length

	EN-13757-4 minimum		Si443x Preamble Setting		Sync Word	Total	extra
	n x (01)	chips	nibbles	chips	chips	chips	chips
Mode S short preamble	15	30	8	32	6	38	8
Mode S long preamble	279	558	138	552	6	558	0
Mode T (meter-other)	19	38	10	40	6	46	8
Mode R	39	78	20	80	6	86	8

The minimum preamble for reception is determined by the Preamble Detection Control register (0x35). Upon reception, the number of bits in the sync word must be subtracted from the specified minimum preamble to determine the usable preamble. The minimum settling time of the receiver is 16-chips if AFC is enabled or 8-chips if AFC is disabled. The receiver settling time is also subtracted from the usable preamble to determine the minimum setting for the Preamble Detection Control register.

SILICON LABS

The probability of a false preamble depends on the setting of the Preamble Detection Control register. A short setting of 8-chips may result in a false preamble detected every few seconds. The recommended setting of 20-chips makes false preamble detection an unlikely event. The preamble lengths for Mode R and Mode SL are sufficiently long for the recommended setting to be used. There is very little benefit to making the preamble detect longer than 20 chips.

The AFC is disabled for Mode S with a short preamble and Mode T. This reduces the receiver settling time and permits a longer preamble detection setting. With AFC disabled, Mode T can use the recommended setting of 20 chips. A setting of 4 nibbles or 20 chips is used for Mode S with a short preamble. This makes the probability of a false preamble detection slightly higher for this mode.

Table 4. Preamble Detection

	EN-13757-4 minimum		Sync Word	usable preamble	RX Settling	Detect min	Si443x Preamble Detection Setting	
	n x (01)	chips	chips	chips	chips	chips	nibbles	chips
Mode S short preamble	15	30	6	24	8*	16	4	16
Mode S long preamble	279	558	6	552	16	536	5	20
Mode T (meter-other)	19	38	6	32	8*	24	5	20
Mode R	39	78	6	72	16	56	5	20
*Note: AFC disabled								

The receiver is configured to interoperate with a transmitter using the minimum specified preamble. This ensures the receiver will interoperate with any M-bus-compliant transmitter.

The Wireless M-Bus specification requires a very long preamble for Mode S1 of at least 558 chips. This will take about 17 ms just to transmit the preamble. The Si443x does not require such a long preamble and does not benefit from the long preamble. While the long preamble is noted as optional for Mode S2, there is no reason to use a long preamble with the Si443x. If one-way communication is desired, Mode T1 will provide a shorter preamble, higher data rate, and longer battery life. If two-way communication using Mode S2 is required, the short preamble is recommended.

Notice that the detection threshold for Mode S with a long preamble is longer than the number of preamble nibbles transmitted for Mode S with a short preamble. This means that the long preamble Mode S receiver will not detect a preamble from a short preamble Mode S transmitter. This is necessary if the long preamble Mode S receiver is to receive any benefit from the longer preamble.

Note that the short preamble Mode S receiver will detect the preamble and receive packets from both a short-preamble Mode S transmitter and a long-preamble Mode S transmitter; so, in general, the meter reader should use the short preamble Mode S receiver configuration.

2.4. Encoding/Decoding

The Wireless M-bus specification requires two different encoding methods. Manchester encoding is used for Mode S and Mode R. Manchester encoding is also used for the other-to-meter link in Mode T. The Mode T meter-to-other link uses 3 out of 6 encoding.

2.4.1. Manchester Encoded/Decoding

Manchester encoding is common historically in RF systems to provide robust clock recover and tracking using a simple and inexpensive modem. However, a modern high-performance radio like the Si443x does not need Manchester encoding. Manchester encoding is supported primarily for compatibility with existing standards, but the data rate for the Si443x is effectively doubled when not using Manchester encoding.

The Si443x supports Manchester encoding and decoding of the entire packet in hardware. Unfortunately, the synchronization word is not Manchester encoded. An invalid Manchester sequence was intentionally chosen for the synchronization word. This makes Manchester encoding incompatible with most existing radios, including the Si443x. As a consequence, the Manchester encoding and decoding must be performed by the MCU.

Each byte on unencoded data consists of eight data bits. Using Manchester encoding, each data bit is encoded into a two-chip symbol. Since the encoded data must be written to the radio FIFO eight chips at a time, one nibble of data is encoded and written to the FIFO at a time.

Table 5. Manchester Encoding

data	0x12		0x34		bytes
	0x1	0x2	0x3	0x4	nibbles
	0001	0010	0011	0100	binary
chip	10101001	10100110	10100101	10011010	binary
FIFO	0xA9	0xA6	0xA5	0x9A	hex

Each byte to be transmitted is passed one byte at a time to the encode byte function. The encode byte function will call the encode nibble function twice, first for the most significant nibble and then for the least significant nibble.

Manchester encoding in software is not difficult. Starting from the most significant bit, a one is encoded as a "01" chip sequence. A zero is encoded as a "10" chip sequence. This can be easily accomplished using a loop and shifting two-bits for each symbol. However, it is faster to just use a simple 16 entry look-up table for each nibble. The encode Manchester nibble function encodes a nibble of data then writes it to the FIFO. The chips are inverted before writing to the FIFO to account for the inverted preamble requirements.

When receiving, each byte in the FIFO consists of eight chips and is decoded into one nibble of data. The read block function reads one byte at a time from the FIFO and calls the decode byte function. The chips are inverted after reading from the FIFO to account for the inverted preamble requirements. Each byte of Manchester encoded chips is decoded into a nibble of data. The decoded nibble is written to the RX buffer using the write nibble RX buffer function.

Notice that both encoded and decoding are performed one data nibble at a time on the fly. Encoding to a buffer would require an additional buffer twice the size of the unencoded data. Encoding and decoding is much faster than the fastest supported data rate (100 k chips per second). Since the Si443x supports multiple-byte reads and writes to the FIFO, there is a small overhead in using only single byte reads and writes. The overhead is about 10 µs for 100 encoded chips. The benefit is a RAM savings 512 bytes.

SILICON LABS

2.4.2. Three Out of Six Encoding Decoding

The Three-out-of-Six encoding method specified in EN-13757-4 is also implemented in firmware on the MCU. This encoding is used for the high-speed (100 k chips per second) Mode T from meter to other. Mode T provides the shortest transmission time and the longest battery life for a wireless meter.

Each byte of data to be transmitted is divided into two nibbles. The most significant nibble is encoded and transmitted first. Again, this is implemented using and encode byte function that calls the encode nibble function twice.

Each nibble of data is encoded into a six-chip symbol. The sequence of six-chip symbols must be written to the 8-chip FIFO. During encoding, two bytes of data are encoded as four nibbles. Each nibble is a 6-chip symbol. Four 6-chip symbols are aggregated as three bytes.

Table 6. Three Out of Six Encoding

data	0x12		0x34		bytes
	0x1	0x2	0x3	0x4	nibbles
chip	015	016	013	034	octal
	001101	001110	001011	011100	binary
FIFO	00110100	11100010		11011100	binary
	0x34	0xE2		0xDC	hex

In software, the three-out-of-six encoding is implemented using three nested functions. The encode byte function will call the encode nibble function twice. The encode nibble function uses a look-up table for the six-chip symbol and writes the symbol to the Shift Three out of Six function. This function implements a 16-chip shift register in software. The symbol is written to the least significant byte of the shift register. The register is shifted left twice. This is repeated three times. When a complete byte is present in the upper byte of the shift register, it is inverted and written to the FIFO.

Since each byte of data is encoded as one and a half encoded bytes, it is important to clear the shift register initially so that the first encoded byte is correct. If the packet length is an odd number, after encoding all bytes, there will still be one nibble left in the shift register. This is handled with the postamble as explained in the next section.

Decoding the three out of six encoded is the reverse procedure. When decoding, three encoded bytes are decoded into two data bytes. The software shift register is again used to aggregate bytes of decoded data. A 64-entry inverse look-up table is used for decoding. This uses fewer cycles but more code memory. Searching a 16-entry look-up table for the corresponding symbol takes considerably longer.

2.5. Postamble

The Wireless M-bus specification has specific requirements for the postamble or trailer. For all modes, the minimum is two chips, and the maximum is eight chips. Since the minimum atomic unit for the FIFO is one byte, an 8-chip trailer is used for Mode S and Mode R. The Mode T postamble is eight chips if the packet length is even or four chips if the packet length is odd. The four chip postamble for an odd packet length meets the requirements of having at least two alternating chips.

Table 7. Postamble Length

	Postamble Length (chips)				chip sequence
	min	max	Implementation		
Mode S	2	8	8		01010101
Mode T	2	8	4	(odd)	0101
			8	(even)	01010101
Mode R	2	8	8		01010101

2.6. Packet Handler

The packet handler on the Si443x can be used in a variable packet width mode or a fixed packet width mode. The variable packet width mode requires a packet length byte after the synchronization word and optional header bytes. Upon reception, the Radio will use the length byte to determine the end of a valid packet. On transmission, the radio will insert the length field after the header bytes.

The L field for the wireless M-bus protocol cannot be used for the Si443x length field. First, the L field is not the actual packet length. It is the number of link layer payload bytes not including the CRC bytes or encoding. Secondly, the L -field itself is encoded using either Manchester encoding or Three out of Six encoding for Mode T meter to other.

The implementation uses the packet handler in fixed packet width mode for both transmission and reception. Upon transmission, the PHY layer will read the L field in the transmit buffer and calculate the number of encoded bytes, including the postamble. The total number of encoded bytes to be transmitted is written to the Packet Length register (0x3E).

Upon reception, the first two encoded bytes are decoded, and the L-field is written to the receive buffer. The L-field is used to calculate the number of encoded bytes to be received. The number of encoded bytes to be received is then written to the Packet Length register (0x3E). The postamble is discarded.

The MCU must decode the L-field, calculate the number of encoded bytes, and write the value to the Packet Length register before the shortest possible packet length has been received. The shortest permissible L-field for the PHY layer is 9, giving 12 unencoded bytes. This gives 18 encoded bytes for Mode T. The first two bytes have already been decoded. Thus, the packet Length register must be updated in 16 byte times at 100 kcps or 1.28 milliseconds. This is no problem for an 8051 running at 20 MIPs.

The number of bytes to be received does not include the postamble, except for the four-chip postamble used for Mode T packets with an odd packet length. Thus, the receiver does not require a postamble, except for the Mode T odd length packets. This postamble is needed only to give an integer number of encoded bytes. The content of the postamble is ignored; so, if the postamble is not transmitted, four chips of noise will be received and ignored.

Since the total number of encoded bytes is limited to 255 (0xFF), the implementation limits the maximum L-field for the different modes.

SILICON LABS

Table 8. Packet Size Limits

	encoded		decoded		M-Bus	
	bytes		bytes		L-Field	
	dec	hex	dec	hex	dec	hex
Mode S	255	FF	127	7F	110	6E
Mode T (meter-other)	255	FF	169	A9	148	94
Mode R	255	FF	127	7F	110	6E

These limits are normally well above the typical usage case for a wireless meter. The packet length should be kept small to get the best possible battery life.

In addition, the user may specify the maximum L-field that should be received (USER_RX_MAX_L_FIELD). This determines the required size for the receive buffer (USER_RX_BUFFER_SIZE).

Supporting a maximum L-field of 255 would require a receive buffer of 290 bytes and a maximum of 581 Manchester encoded bytes. The packet handler would need to be disabled and the Packet Length register could not be used in that case. This is feasible, but it is more convenient to use the packet handler, if possible.

2.7. FIFO Usage

The Si4431 provides a 64 byte FIFO for transmit and receive. Since the number of encoded bytes is 255, an entire encoded packet may not fit within the 64-byte buffer.

2.7.1. Transmission

On transmission, the total number of encoded bytes is calculated. If the total number of encoded bytes, including the postamble, is less than 64 bytes, the entire packet is written to the FIFO and only the packet sent interrupt is enabled. Most short packets will be sent in one FIFO transfer.

If the number of encoded bytes is greater than 64, multiple FIFO transfers will be required to send the packet. The first 64 bytes are written to the FIFO. The Packet Sent and TX FIFO Almost Empty interrupts are enabled. The TX FIFO Almost Empty threshold is set to 16 bytes (25%). Upon each IRQ event, the status 2 register is read. The Packet Sent bit is checked first, and, if the packet has not been completely sent, the next 48 bytes of encoded data are written to the FIFO. This continues until all encoded bytes have been written and the Packet Sent interrupt occurs.

2.7.2. Reception

On reception, initially only the Sync Word interrupt is enabled. After receiving the sync word, the sync word interrupt is disabled and the FIFO Almost Full interrupt is enabled. The FIFO almost full threshold is initially set to 2 bytes. The first FIFO Almost Full interrupt is used to know when the two length bytes have been received. Once the length has been received, the length is decoded and the number of encoded bytes is calculated. The RX FIFO almost Full threshold is then set to 48 bytes. The RX FIFO almost full and Valid Packet interrupts are enabled. Upon the next IRQ event, the status 1 register is read. First, the Valid Packet bit is checked, and then the FIFO Almost Full bit is checked. If only the RX FIFO Almost Full bit is set, the next 48 bytes are read from the FIFO. If the valid packet bit is set, the remainder of the packet is read from the FIFO. The MCU keeps track of how many bytes have been read and stops reading after the last byte.

3. Data Link Layer

The data link layer module implements a 13757-4:2005 compliant link layer. The data link layer (LINK) provides an interface between the physical layer (PHY) and the application layer (AL).

The Data Link Layer performs the following functions:

- Provides functions that transfer data between PHY and AL
- Generates CRCs for outgoing messages
- Detects CRC errors in incoming messages
- Provides physical addressing
- Acknowledges transfers for bidirectional communication modes
- Frames data bits
- Detects framing errors in incoming messages

3.1. Link Layer Frame Format

The Wireless M-Bus frame format used in EN 13757-4:2005 is derived from the FT3 (Frame Type 3) frame format from IEC60870-5-2. The frame consists of one or more blocks of data. Each block includes a 16-bit CRC field. The first bock is a fixed length block of 12 bytes that includes the L-field, C-field, M-field, and A-Field.

3.1.1. L-Field

The L-field is the length of the Link layer data payload. This does not include the L-field itself or any of the CRC bytes. It does include the L-field, C-field, M-field, and A-Field. These are part of the PHY payload.

Because the number of encoded bytes is limited to 255 bytes, the maximum supported value for the M-field is 110 bytes for Manchester encoded data and 148 bytes for Mode T Three-Out-of-Six encoded data.

The Link layer is responsible for calculating the L-field on transmission. The link layer will use the L-field on reception.

Note the L-field does not indicate the PHY payload length or the number of encoded bytes. Upon transmission, the PHY will calculate the PHY payload length and the number of encoded bytes. Upon reception, the PHY will decode the L-field and calculate the number of bytes to decode.

3.1.2. C-Field

The C-field is the frame control field. This field identifies the frame type and is used for the link data exchange service primitives. The C-field indicates the frame type - SEND, CONFIRM, REQUEST, or RESPOND. In the case of SEND and REQUEST frames, the C-field indicates whether a CONFIRM or RESPOND is expected.

When using the basic Link TX function, any value of C can be used. When using the Link Service Primitives, the C field is populated automatically according to EN 13757-4:2005.

3.1.3. M-Field

The M-field is the manufacturers code. Manufacturers can request a three letter code from the following web address: http://www.dlms.com/flag/INDEX.HTM

Each character of the three letter code is encoded as five bits. The 5-bit code may be obtained by taking the ASCII code and subtracting 0x40 ("A"). The three 5-bit codes are concatenated to make 15-bits. The most significant bit is zero.

3.1.4. A-Field

The address field is a unique 6-byte address for each device. The unique address should be assigned by the manufacturer. It is the responsibility of each manufacturer to ensure that each device has a unique 6-byte address.

The address for Send and Request frames is the self-address of the meter or other device. The confirm and response data frames are sent using the address of the originating device.

SILICON LABS

3.1.5. CI-Field

The CI-field is the application header and specifics the type of data in the application data payload. While EN13757-4:2005 specifies a limited number of values, the Link Service Primitives will permit any value to be used.

3.1.6. CRC

The CRC is specified in EN13757-4:2005.

The CRC Polynomial is:

$X16 + x13 + x12 + x11 + x10 + x8 + x6 + x5 + x2 + 1$

Note that the M-Bus CRC is calculated over each 16-byte block. The result is that each 16 bytes of data require 18 bytes to be transmitted,

3.2. Additional Information

For additional information about the Link Layer Implementation, see "AN452: Wireless M-Bus Stack Programmers Guide".

4. Power Management

Figure 2 shows the power management timeline for a meter example using the Mode T1.
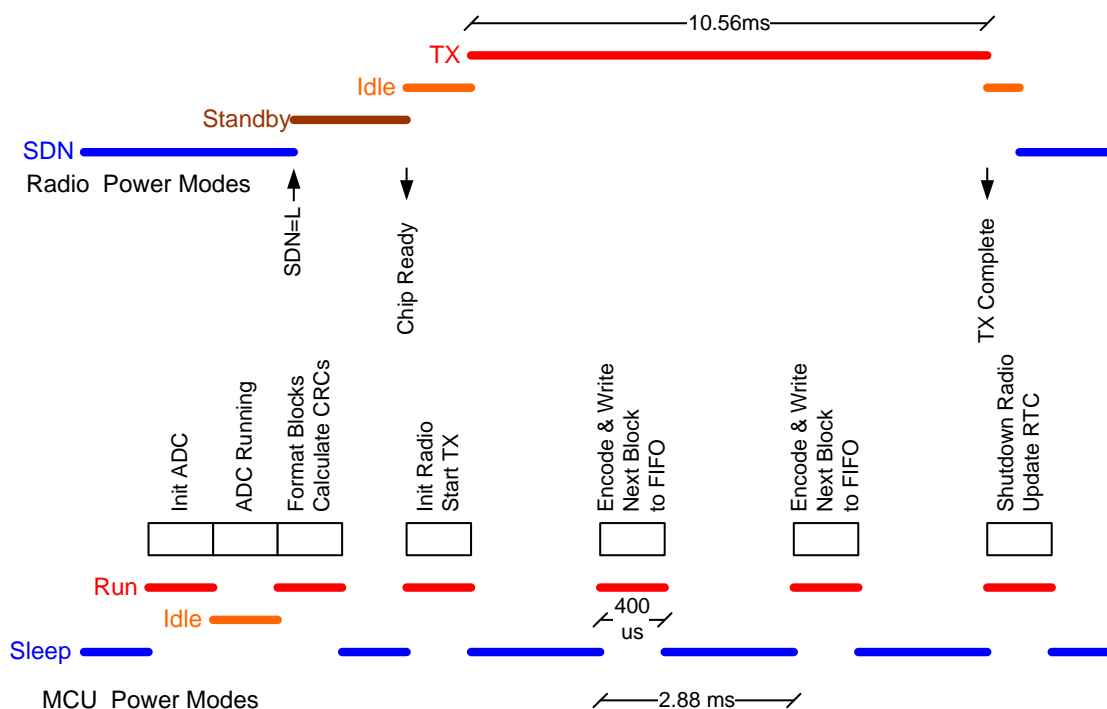
Figure 2. Transmit Only Meter Power Management

The MCU should be in Sleep mode whenever possible to conserve energy. In this example, the MCU is sleeping when the RTC is running, when waiting on the radio crystal start-up, and when transmitting from the FIFO. The MCU will wake-up from the EZRadioPRO IRQ signal connected to a Port Match wake-up.

When transmitting messages longer than one block, the MCU must wake up to fill the FIFO (based on the FIFO almost empty interrupt) and then go back to sleep.

The MCU should be in Idle mode running from the low power oscillator or burst-mode oscillator when reading from the ADC. The ADC requires a SAR clock.

When not in use, the EZRadioPRO should be in Shutdown mode with the SDN pin driven high. This requires a hardwired connection to the MCU. The EZ Radio Pro registers are not preserved in shutdown mode; so, the EZRadioPro is initialized on each RTC interval. Initializing the Radio takes less than 100 μs and conserves 400 nA. This results in a 10 μJ energy savings, based on a 10 second interval.

The EZRadioPRO crystal takes about 16 ms for a POR. This is long enough to calculate the CRC for about eight blocks. The MCU will go back to sleep if it completes all CRCs before the crystal has stabilized. If encryption is required, it too can be started while waiting on the crystal oscillator.

The MCU should run at 20 MHz using the low-power oscillator for most tasks. Tasks that require a precise timeout must use the precision oscillator and idle mode instead of sleep mode. The RTC provides enough resolution for most tasks. The power management timeline for the T2 meter example application is shown in Figure 3.
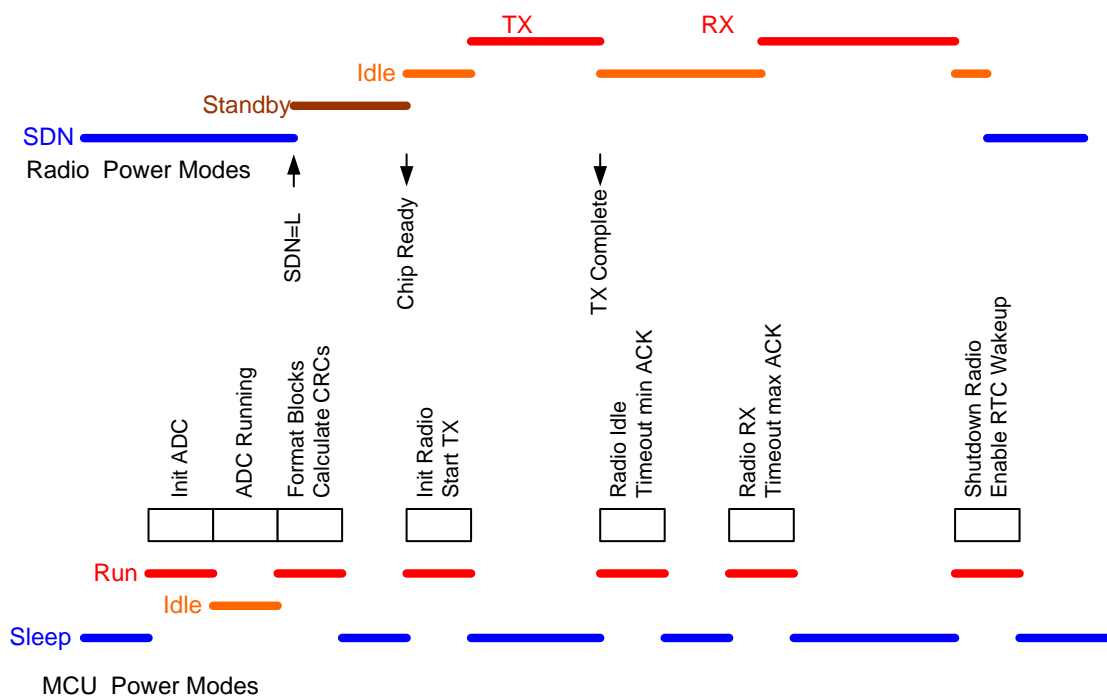
SILICON LABS

Figure 3. Meter Transceiver Power Management

The transceiver implementation should be optimized for the normal case when the meter wakes up and there is no reader present. The minimum/maximum ACK timeouts are sufficiently long so that it is possible to use the C8051F930 RTC and put the MCU into sleep mode.

Build options are provided for mains or USB powered readers that do not need to use sleep mode. Idle mode will be used instead of sleep so that the USB and UART may interrupt the MCU.

CONTACT INFORMATION

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
Tel: 1+(512) 416-8500
Fax: 1+(512) 416-9669
Toll Free: 1+(877) 444-3032

Please visit the Silicon Labs Technical Support web page:
https://www.silabs.com/support/pages/contacttechnicalsupport.aspx
and register to submit a technical support request.