Freescale Semiconductor
Application Note

Document Number: AN3603
Rev. 0, 03/2008

How to Use the MC33596

by: Stephane Lestringuez
Freescale RF Application Engineer
Microcontroller Solutions Group
Toulouse, France

This document provides some considerations to help you properly use the MC33596 RF Receiver. Everything true for the MC33596 receiver is true for the receiver part of the MC33696 transceiver.

The document is split into the following sections:

- Description of the communication between MC33596 and MCU; SPI considerations
- Using the data manager and recommended frame
- Strobe oscillator sizing
- Using both the data manager and the strobe oscillator in a practical example
- Configuration switching description
- RSSI acquisition modes
- Frequency addressing

Contents

freescale semiconductor

1 MC33596/MCU Interface

MC33596 communicates with the MCU by means of a bidirectional serial digital interface (SPI). This approach minimizes the connection between the two devices.

Some additional digital pins associated with further features are also implemented and listed.

1.1 Digital Pins

1.1.1 Minimum Configuration

The communication interface of MC33596 is operated by these five pins:

- SEB (input) — serial interface enable: When high, pins SCLK, MOSI, and MISO are set to high impedance state, and SPI is disabled. This allows individual selection in a multiple device system, where all devices are connected via the same bus. The rest of the circuit remains in the current state, enabling fast recovery times.
- SCLK (I/O) — serial clock: Synchronizes data flow through MOSI and MISO lines. The master and slave devices are able to exchange a byte of information during a sequence of eight clock cycles. Because SCLK is generated by the master device, this line is an input for a slave device.
- MOSI (I/O) — master output slave input: Transmits bytes when master, and receives bytes when slave, with the most significant bit first. When no data are output, SCLK and MOSI force a low level.
- MISO (I/O) — master input slave output: Transmits bytes when slave, with the most significant bit first.
- CONFB (input) — configuration mode selection: Selects the mode of the receiver. Receive mode selected when high level (receiver master) and configuration mode selected when low level (receiver slave).

1.1.2 Additional Pins

- STROBE (input) — strobe control: Allowing on/off sequencing of the receiver either by using the internal strobe oscillator (SOE = 1, external capacitor needed to clock the off time), or by using external management from the MCU (SOE = 0).
- RSSIC (input) — RSSI control: When high, the RSSI value is continuously updated, regardless of whether it is the analog form on the RSSIOUT pin or the digital form in the RSSI register. The management of this pin in pulsed mode allows sampling of the incoming signal.
- DATACLK (output) — clock toward microcontroller: Provide to the microcontroller a stable reference frequency (around 300 kHz) generated from crystal division.

1.2 Serial Peripheral Interface (SPI)

According to the selected mode through the level on CONFB, either the receiver or the MCU manages the data transfer (on SCLK and MOSI lines in writing mode):

- When master (CONFB = 0), the microcontroller sends or checks data in receiver registers through MOSI or MISO lines. This data is triggered on the falling edge of the clock provided by the MCU on the SCLK line.
- When master (CONFB = 1), the receiver sends received data to the MCU through the MOSI line. This data is triggered on the falling edge of the clock generated on the SCLK line in case the data manager is used.

The aim of this section is to explain the correct way to manage the link between receiver and MCU.

Additional external hardware, but also the definition of a strategy to cleverly manage the MCU wakeup, is presented. Cleverly here means to find the best balance between two competing requirements: optimizing consumption without missing any incoming information.

1.2.1 MC33596 in Configuration Mode (MCU Master)

In configuration mode, the MCU is master. It sends its own clock through the SCLK line, and writes or reads MC33596 register values.

1.2.1.1 Chronograms in Configuration Mode

Figure 1 and Figure 2 show chronograms associated with the configuration mode (CONFB = 0). In this mode the user can write or read MC33596 registers, depending on the first 8-bit stream (called the command byte) sent on the MOSI line by the MCU.

The command byte specifies the number of registers to access (N[1:0]), the address of the first register to access (A[4:0]), and the type of operation to perform (read or write). This last bit is thus associated with the presence of information on MOSI or MISO lines.

The clock is generated by the microcontroller (see datasheet for parametric value). Data must be valid on falling edges of SCLK to be well understood on the receiver side.

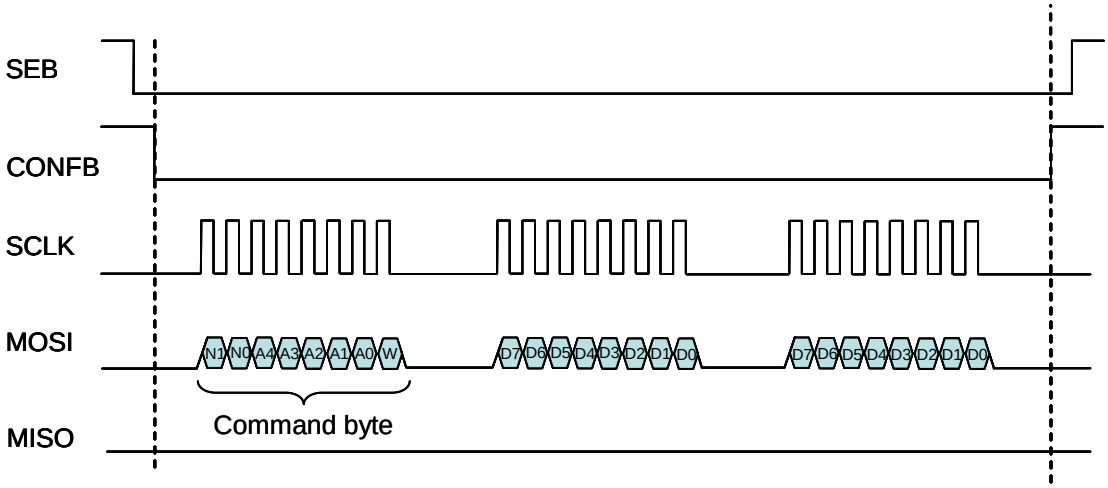


Figure 1. Write in Configuration Mode (N[1:0] = 01)

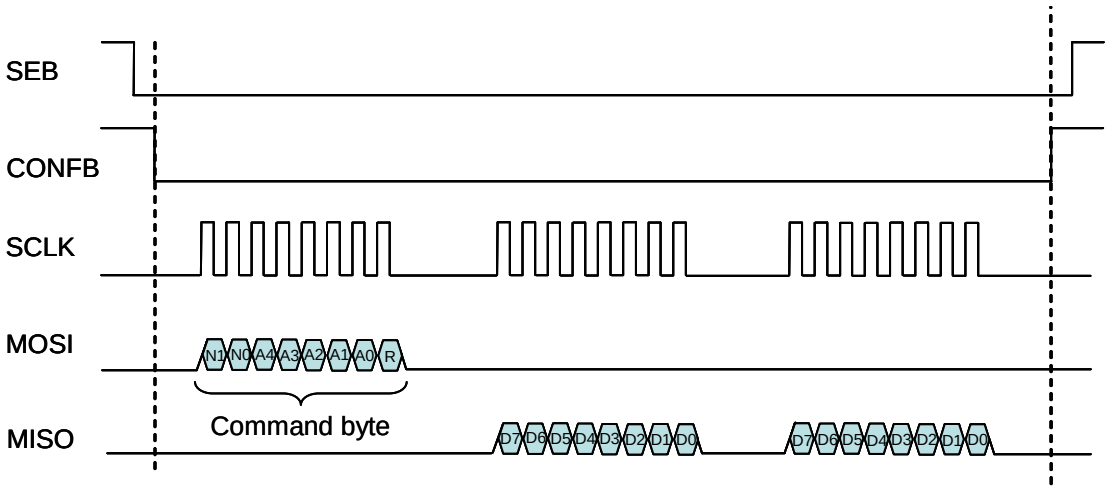


Figure 2. Read in Configuration Mode (N[1:0] = 01)

1.2.1.2 Enter Receiver in Configuration Mode

The behavior of the receiver is managed by a finite state machine fully described in the datasheet. At any time, a low level applied on the CONFB line forces the state machine to enter state 1 (the configuration mode), regardless of its current state. It is thus the microcontroller that selects the configuration mode to write or read receiver internal registers, to apply a new configuration to the system, or to check the current configuration. In this mode, the SPI is slave. Its corresponding inputs are high impedance.

1.2.1.3 Changing Register Configuration Dynamically

In some cases, it is necessary to change register configuration while RF data is being received. To change this register configuration, the CONFB line must be forced to low level. A problem can occur if the SCLK line is high when CONFB is forced to low, because there is a time when neither receiver nor MCU is master, and the SCLK line is in a high impedance state. In this state it has no control of its status (after the receiver has been set to slave and before the MCU becomes master).

Thus, an unwanted falling edge on the SCLK line can be generated either because the MCU forces a low level on it or because the SCLK voltage is discharged through the input capacitor path (if the time before the MCU becomes master is long enough). This unwanted falling edge could be totally unacceptable in MC33591 family products, as receivers could understand it as a normal clock sent by the MCU, and thus start to read or write registers, depending on the MOSI line value (read for low level and write for high level). This criticality is linked to the first position of the Write/Read order in the command byte.

In MC33596 the Write/Read order is located at the last position of the command byte, and consequently this unwanted falling edge should not produce any unwanted writing in the registers. But to completely avoid any erratic behavior, we recommend that in critical applications (automotive area), the level on SCLK be managed properly during this high impedance status of the line by adding a pullup resistor on the line or, by applying a short low level impulsion on CONFB line, to reset the state machine and consequently force a low level on the SCLK line before the MCU becomes master.

1.2.2 MC33596 in Receive Mode (MC33596 Master)

This section applies only to the data manager mode (DME = 1). Otherwise, an incoming message is sent directly to the MOSI line, without any associated clock, decoding, or recognition tasks (ID, header), in such a manner that we do not use SPI termination in DME = 0 mode anymore.

1.2.2.1 Chronograms in Receive Mode

Figure 3 shows chronograms associated with the receive mode (CONFB = 1).

An example of Manchester-coded input data is shown. SPI signals generated with the receiver configured in data manager mode (DME = 1) are illustrated; a clock is generated on SCLK at the data rate, and decoded data (NRZ) is transmitted on the MOSI line, triggered with the falling edges of the SCLK signal.

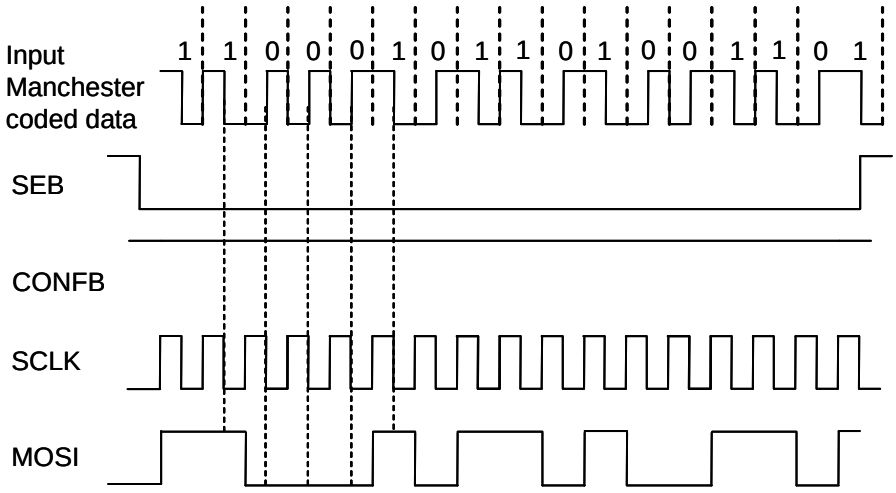


Figure 3. SPI in Receive Mode

1.2.2.2 MCU Wakeup Strategy

1.2.2.2.1 Using Two Configurations

When system consumption is a critical parameter, the user must take care in configuring the MC33596 receiver and associated MCU to fully optimize the power budget.

First, to not waste power consumption, the MCU shouldn't be awake when it is not necessary. In other words, when useful incoming data is not being received the MCU should be asleep. So we assume in this section that the MCU is asleep as long as no wanted RF signal is incoming.

Another way to reduce power consumption is to periodically check whether useful information is incoming on the receiver side. This is the reason the receiver is generally used in periodic on/off cycle (called strobe mode in MC33596). During the on time, the incoming signal is checked by means of recognition tasks, including in the data manager, and if no useful signal is detected, the receiver returns to sleep mode until the next on time.

As soon as useful data is detected, the MCU wakes and data is sent to it through the SPI link on the MOSI and SCLK lines.

The MCU wakeup has to be managed with care, so two register configurations will be defined: Conf1 and Conf2.

Conf1 is used only to wake the MCU by means of sending an interrupt request to it on the falling edge of the SCLK signal.

Basically, the user has to configure MC33596 ID and header contents (ID1 and HD1) to efficiently poll any useful incoming signal in strobe mode. After the ID1 has been detected, the receiver stays in run mode, and after HD1 is detected, the interrupt request is sent to the MCU by means of SCLK line activation.

The user has to find optimized values and lengths for ID and header contents according to the system used and incoming expected frame. The goal is to find a good trade-off between receiver consumption, system consumption (including false wakeup), and the response time of the system.

For these reasons, we recommend choosing a rather long ID1 to decrease the number of false occurrences of wakeup (which over time will increase system consumption). The associated recommendation is to keep HD1 (1 bit) as short as possible to reduce the MCU wakeup time after a correct ID has been detected. Indeed, data coming after ID1 has no importance for the MCU — the goal in Conf1 mode is just to wake the MCU as quickly as possible after ID is detected.

After the MCU is awake, the next step is to reprogram MC33596 registers (Conf2) to send useful data to the MCU.

When MC33596 is in Conf2 and the MCU is awake, NRZ decoded data is sent to the MCU on the MOSI line after the header (HD2) detection. When the MC33596 is in Conf2 mode, we advise the user to force the STROBE pin high. This avoids the scenario in which Conf2 mode is started with a possible off time but the MCU is waiting for useful data.

The different steps described in this receive mode, including use of the strobe oscillator and MCU wakeup, will be illustrated in section 1.2.3.

1.2.2.2.2 Pullup on CONFB Pin

As already mentioned, a high level must be applied on the CONFB line to enter MC33596 in receive mode.

When the MCU is in sleep mode, its ports are in input state, thus in high impedance.

Consequently a pullup resistor needs to be added on the CONFB line to allow receive mode when MCU is in sleep mode, and to avoid entering configuration mode accidentally.

The choice of the pullup resistor depends on MCU port characteristics in terms of VIL/VIH and current consumption capability.

1.2.3 SPI States When Using Power Consumption Optimization

Figure 4 shows the various SPI steps in reception when using a power consumption optimization scheme, in other words, using receiver strobe oscillator and MCU wakeup as soon as useful information is incoming.

The different steps mentioned in Section 1.2.2.2.1, “Using Two Configurations,” are illustrated: receive in strobe mode with Conf1, MCU wakeup by means of an interrupt, reconfiguration of the receiver with Conf2, and receive in continuous mode with Conf2.

Notice that in the following schematic, the only pullup resistor represented is on the CONFB line, which has been identified as the most critical one for the global operation. According to the context where the receiver will be implemented, users will be able to add pullup or pulldown resistors on all SPI paths, to fix a level on pins when both receiver and MCU pins are in a high impedance state.

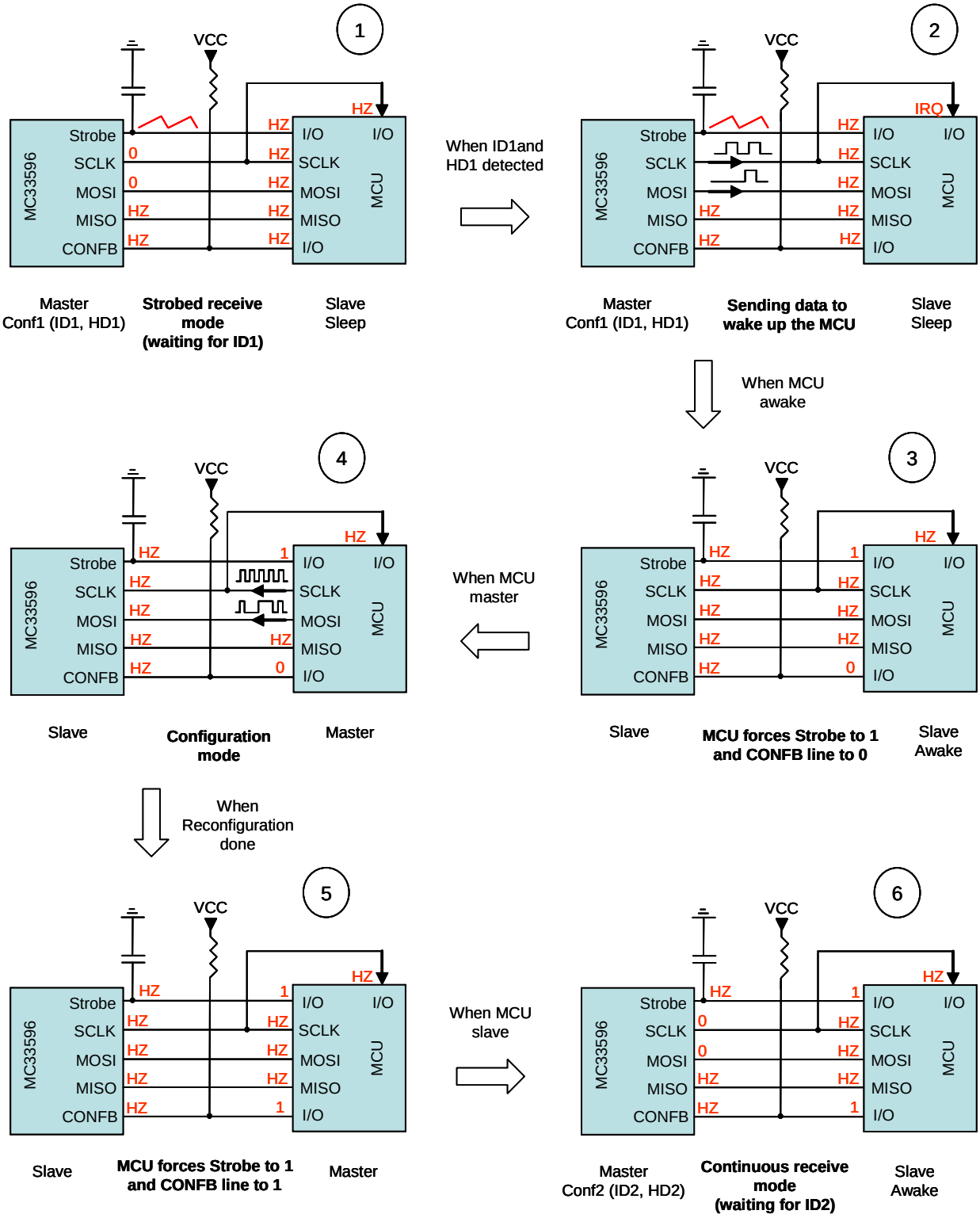


Figure 4. SPI Configurations with MCU Wakeup Steps

2 Data Manager

The goal of using the data manager is to reduce system power consumption through the following generic methods:

- Wake the receiver only when a predefined ID is recognized
- Convert a Manchester-coded signal to NRZ format, reducing microcontroller power consumption
- Provide clock at the data rate to the MCU (clock recovery)

If selected (DME = 1), this process is initiated when the receiver wakes up and detects a Manchester-coded signal at a selected data rate with a valid predefined ID detected. The receiver wakeup occurs when either off counter time is reached and SOE = 1 (state 11 of the state machine), or the STROBE pin is set to high level and SOE = 0 (state 21 of the state machine).

The Manchester-coded conversion performed on the receiver side avoids the complex task of decoding data with the MCU, regardless of whether the MCU is continuously running or periodically entering run/sleep mode. This saves system power because the state machine in the receiver has been optimized for power savings. Otherwise, the MCU would have consumed more power to perform decoding.

The frame-recognition tasks allow the MCU to wake only when a valid frame is received, thus saving additional system power consumption. But the MCU wakeup has to be managed with care to not lose useful information during receiver and MCU stabilization times.

2.1 Using the Data Manager

2.1.1 Data Manager Disabled

The MC33596 can be used without the data manager activated (DME = 0). This option is useful when one wants to receive any coding format other than Manchester. Indeed, in this case SPI is deactivated, and demodulated data (with original coding) is sent directly on the MOSI line in raw format, whatever the frame protocol received.

The MCU and receiver are generally continuously running in this case. System power consumption is not optimized and the MCU has to decode data itself in case of a non-NRZ incoming signal.

2.1.2 Data Manager Enabled

2.1.2.1 MCU Continuously Running

When system consumption is not a crucial parameter, for example when using standard alternating current power supply, or when the MCU has to manage additional and parallel tasks, then the MCU is generally used in continuous run mode. This avoids the task of wakeup by means of dedicated strategy.

In this situation, data and clock are sent on SPI as soon as the identifier and header have been properly detected. Of course, strobe mode can be also used to reduce average power consumption of the receiver.

2.1.2.2 MCU Wakeup by Receiver

As we have already seen, the use of the data manager allows the receiver to be periodically awake/asleep, and maintains its awake status only when a useful message is received. This mode, reached by using both strobe oscillator and data manager, allows optimization of receiver power consumption.

Of course, the same philosophy regarding the MCU can be applied — in other words, keep the MCU in sleep mode as long as no useful information is incoming to the receiver, and wake it as soon as a useful message has to be managed. The MCU run/sleep management, associated with use of the strobe oscillator and data manager on the receiver side, allows optimization of consumption for the whole system (receiver plus MCU).

The aim is to define a clever strategy for waking the MCU without missing any useful data.

No interrupt has been scheduled in the receiver to send to the MCU to inform it that useful data is incoming. One might think of waking the MCU on the first falling (or rising) edge of the SCLK line that is active as soon as the SPI is sending decoded data. But this last option would result in some lost data, at least during the MCU wakeup time.

Consequently, to send the integrity of the signal to the MCU, as already mentioned in Section 1.2.2.2, “MCU Wakeup Strategy, we choose two configurations for the receiver:

- Conf1 — used to wake up the MCU
- Conf2 — used to send data to the MCU

The key point here becomes to define well both the data manager and on/off sequences for each configuration. By doing this you will accomplish two things: optimize consumption, and not miss any useful message during MCU wakeup.

2.2 Recommended Frame with Data Manager

When using the data manager (DME = 1), the structure of the frame must be chosen carefully to optimize the RF link either on the transmit side or on the receive side.

Basically, MC33596 must catch an identifier (ID) before sending data to the MCU. This ID is useful in periodic sleep/run mode to keep the receiver awake as soon as it has detected the right ID, and thus stop the on/off cycle.

Based on this principle, the transmitter can send either discrete ID fields or continuous ones. Before being ready to receive Manchester-coded information (ID, header, or data), the receiver needs a preamble duration time.

This preamble contains some pulses to stabilize the receiver’s internal AGC and average filters for demodulation, and to initiate clock recovery.

The preamble duration depends on the choice of the modulation (refer to the MC33596 datasheet to estimate it). It can represent a non-negligible part of the total run time when the receiver is used in periodic sleep/run cycles (strobe mode). It is the reason that although MC33596 can handle a Manchester coding

violation (as it occurs with discrete ID fields), it is recommended to build the frame based on the following protocol format:

$$\text{Preamble} + \underbrace{\text{ID} + \text{ID} + ... + \text{ID}}_{\text{ID field (N * ID)}} + \text{HD} + \text{Data} + \text{EOM}$$

whether or not there is a header field in the transmitted frame.

In this recommended format, the preamble is necessary only at the beginning of the frame, thus saving power consumption. At the opposite, with discrete ID, a preamble would be necessary before each ID since a Manchester coding violation requires a new preamble duration to stabilize the receiver.

2.3 ID Length versus False Wakeup

When clock recovery is done (during preamble), the data manager is waiting for a valid identifier (ID). An ID is a word whose length and value are programmable and which is inserted into the useful transmitted data.

After the ID is detected, a header (mandatory in MC33596) is expected to identify the beginning of useful data to send to the MCU. Everything coming after the header is sent to the MCU until the EOM (end of message) detection. The EOM is simply a Manchester coding violation coming after useful data.

A timeout is initiated at the end of the ID reception to limit the wait between ID and header. Indeed, the receiver must be able to be automatically sent into sleep mode if an ID is detected and no header is coming. Normally it cannot happen if the frame has been well-dimensioned (if the ID is long enough to be discriminated from noise or other various signals), but in a power-consumption-saving approach, the user can reduce the ID field, and false wakeup events can then increase (consequently raising system power consumption).

In conclusion, it is difficult to define general rules for sizing an RF system, as both transmitted frame and receiver on/off cycles are closely linked to the application the RF system will cover.

For example, in automotive remote keyless entry applications (RKE), the transmitter sends a frame rarely per day. Its average power consumption is mainly determined by its standby current, and the length of the transmitted frame has little influence on its average consumption. The receiver, however, is waiting for a possible incoming signal all day long. Its average power consumption, and in particular the strobe sizing, is of great importance, because the receiver is connected to the battery of the car. As we will show in the section dedicated to strobe sizing, in these applications the user must take into consideration the impact that increasing the number of consecutive IDs (or the ID field length) will have on consumption constraints in the transmitter side.

Now consider a different example. In an automotive tire pressure monitoring system (TPMS), the transmitter sends lot of frames to the receiver to improve the security system by rapidly updating tire pressure information. Moreover, the TPMS system is located in the wheels, and the battery life is required to last up to ten years. Therefore transmitter consumption is crucial, and the user will have to increase the data rate and decrease the ID field length to save battery life. In these applications, the receiver will have to be awake more often and consequently will have its average consumption increased by the on time sizing effect.

3 Strobe Mode

When reducing system consumption is a primary objective, receivers are generally used in periodic run/sleep cycles (or on/off).

Indeed, on and off consumptions are very dissimilar, with a few mA in running mode, versus generally 1 µA or less for consumption during sleep. The consumption during sleep in an MC33596 includes internal oscillator (strobe) consumption, which allows it to have an independent counter from the MCU to define on/off sequences.

Ideally, it is required that the MCU and the receiver should sleep as long as no RF frame is received. To check this, the receiver has to be awake periodically in coordination with the expected frame. Periodic wakeup of the receiver can be performed by the MCU, but in this case the MCU cannot be in a deep sleep mode, as at least an internal timer must run continuously.

An alternative is to use the integrated low-frequency oscillator included in the receiver, which is linked to its state machine. If this internal timer is used (SOE = 1), off time is clocked by this strobe oscillator, whereas on time is directly clocked by the crystal oscillator, allowing more accurate control.

3.1 Consumption in Strobed Mode

If the receiver is operating during t_{On} and sleeping during t_{Off}, then a strobe ratio (SR) can be defined as follows:

$$SR = t_{Off}/t_{On}$$

The associated consumption diagram is shown in Figure 5:

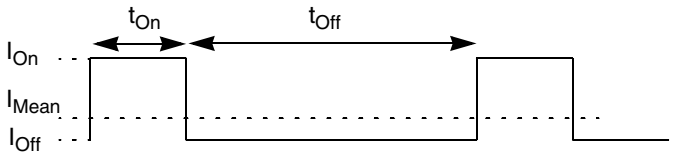


Figure 5. SR Definition, On and Off Periods

Based on this strobe ratio definition, the average consumption is defined by

$$I_{Mean} = (I_{On}*t_{On} + I_{Off}*t_{Off})/(t_{On} + t_{Off})$$

and with SR introduction, we keep in mind the average consumption under the following format

$$I_{Mean} = I_{On}/(1 + SR) + I_{Off}/(1 + 1/SR) \quad (1)$$

The equation for I_{Mean} shows that choosing a high SR is very helpful for reducing power consumption. However, the SR size must remain compatible with the incoming frame, because the most significant on/off timing constraint is a balance between properly catching the useful signal and not compromising the system response time too much.

Indeed, raising the SR will reduce average consumption but also increase the occurrences of missing a useful message, because the off time will increase in comparison with the on time.

Finally, the crucial point when using the strobe mode is to find a good compromise between reducing average consumption and decreasing the possibility of missing a useful incoming frame (of course a receiver almost always off won't consume any energy, but most of the time won't be able to catch any signal).

Figure 6 shows I_{Mean} = f(SR) with typical MC33596 values, in other words I_{On} = 10.3 mA and I_{Off} = 24 µA (including strobe oscillator consumption), and the average consumption related to a quite high SR = 31.

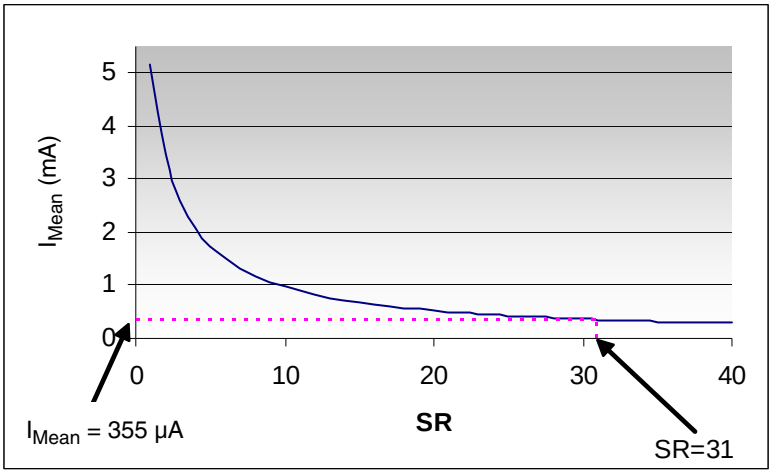


Figure 6. I_{Mean} = f(SR)

3.2 Strobe Sizing

Different methods are used to match the strobe ratio to the transmitted frame. One of them is for the transmitter side to send discrete multiple frames to guarantee that at least one frame will be received. This method has the drawback of increasing the reaction time of the system, because some incoming information can be missed.

The method we describe below, associated with the recommended frame defined in Section 2.2, "Recommended Frame with Data Manager," guarantees that no transmitted frames are lost. In the first step, no MCU wakeup is taken into account in on/off calculations. In the second step, the MCU wakeup and the reconfiguration by means of the MCU are included in the maximum off time definition in Section 3.3.2, "Taking MCU Wakeup into Account in Strobe Sizing."

Different strobe ratios are represented in Figure 7 for a predefined transmitted frame. The frame studied here is based on the recommended format, in other words several IDs continuously transmitted, making a continuous ID field. The primary goal for the receiver is to catch at least one ID during its on state.

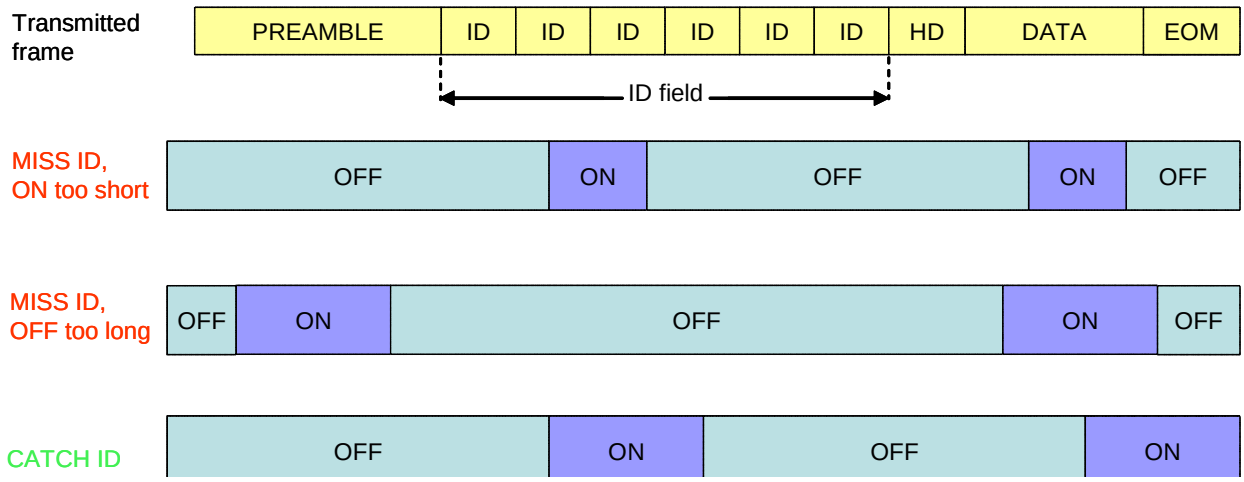


Figure 7. Strobe Ratio Configurations

During t_{On}, the receiver should be able to detect an ID. But because receiver and transmitter are not synchronized, an ID may already have been transmitted when on time begins. That is the reason why t_{On} should be sized to receive at least two IDs.

The off time must be sized carefully for the same reason. Consequently, its duration should not exceed the transmitted ID field length.

These constraints on the on and off times needed to properly receive an incoming frame must be refined as the receiver is not stabilized immediately after its wakeup. Consequently, t_{On} must also include the wakeup time of the receiver, t_{Wakeup}.

This wakeup time includes the crystal oscillator startup, the PLL lock time, and all analog parameters setup, in other words AGC and demodulator stabilization.

With the wakeup time, the minimum on time formula becomes:

$$t_{On} = 2 * t_{ID} + t_{Wakeup}(2)$$

as illustrated in Figure 8.

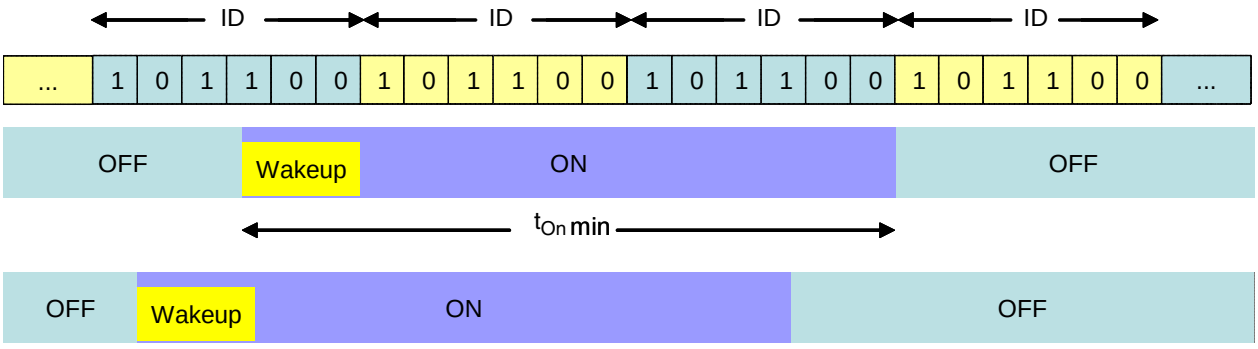


Figure 8. t_{On} Min Schematic

Notice that in case of specific structure for IDs, for example ID = b0000 or ID = b1111, it is possible to detect the ID in a shorter time as there is no identified beginning and end among ID fields in this case. This reduces the requested minimum t_{On} time:

$$t_{On} = t_{ID} + t_{Wakeup}$$

In the same manner, t_{Off} should be sized to allow the positioning of an on state during the transmission of the ID field. Moreover, no reception is possible during t_{Wakeup}. Here we present the limit condition to guarantee that an ID will be detected — if the first is missed, it will be possible to detect the last, as we see in Figure 9:

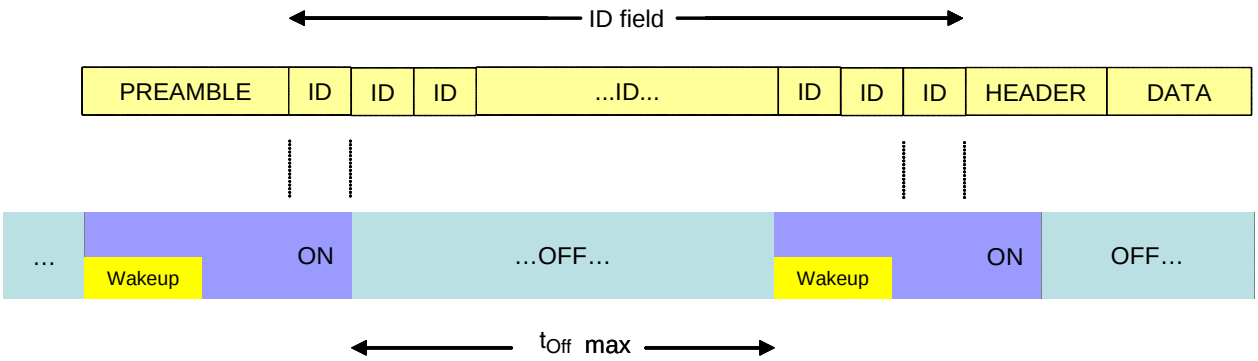


Figure 9. t_{Off} Max Schematic

Based on this, the maximum off time formula gives:

$$t_{Off} = t_{IDField} - t_{Wakeup} - 2 * t_{ID} \quad (3)$$

It should be noted that this relation is also valuable for the particular cases ID = b0000 or ID = b1111, as the two concerned IDs for t_{Off} max calculation are the first and the last in the ID field. Thus no more shift is possible.

Referring to equations (2) and (3), the general rules for increasing the strobe ratio t_{Off}/t_{On} (thus decreasing average consumption) are:

- Increase the number of consecutive IDs in the ID field, and/or
- Reduce the ID length.

In reducing the receiver consumption by raising the strobe ratio, you must find an appropriate balance for the following system factors:

- Not increasing the transmitter consumption too much if the ID field rises;
- Not decreasing the instances of false wake-up immunity in case of short ID lengths.

A complete numerical example is given in Section 4, “System Sizing in Numerical Examples,” to illustrate the strobe sizing and its impact on average consumption.

3.3 Wakeup Time Refinement

3.3.1 XCO Startup

As already mentioned, the wakeup time is the time necessary for the receiver to be stabilized when it goes through off to on status.

The wakeup time includes:

- RF stabilization: XCO startup and PLL lock time
- Analog stabilization: AGC and data slicer reference
- Digital stabilization: Clock recovery

Notice analog and digital stabilization are described as the preamble in the datasheet.

To properly fill the RXONOFF register (which will define the strobe ratio), we need to refine the wakeup time definition. Indeed, the on time defined in RXONOFF register begins after the crystal oscillator has started. Consequently, we split the wakeup time in:

$$t_{Wakeup} = t_{Wakeup1} + t_{Wakeup2}$$

where $t_{Wakeup1}$ is XCO startup only and $t_{Wakeup2}$ defines the remaining stabilization.

Equation (2) is slightly modified by taking into account this split as follow:

$$t_{On} = 2 * t_{ID} + t_{Wakeup2}$$

Equation (3) remaining

$$t_{Off} = t_{IDField} - t_{Wakeup} - 2 * t_{ID}$$

Figure 10 illustrates these last equations.

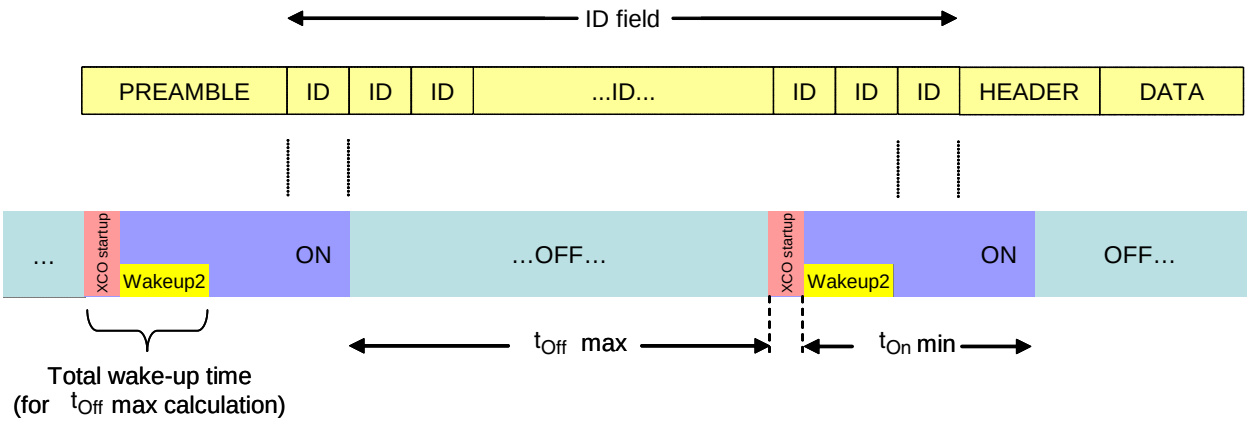


Figure 10. t_{On} Min and t_{Off} Max Representation without MCU Wakeup Consideration

3.3.2 Taking MCU Wakeup into Account in Strobe Sizing

When it is necessary to wake the MCU then t_{Off} max is reduced, as we have to take into account the MCU wakeup time and the receiver reconfiguration before receiving a useful signal (steps 2 and 3 in Figure 4).

The MCU timings represented in Figure 11 impact the average consumption, as the numerical example will show in Section 4, “System Sizing in Numerical Examples.”

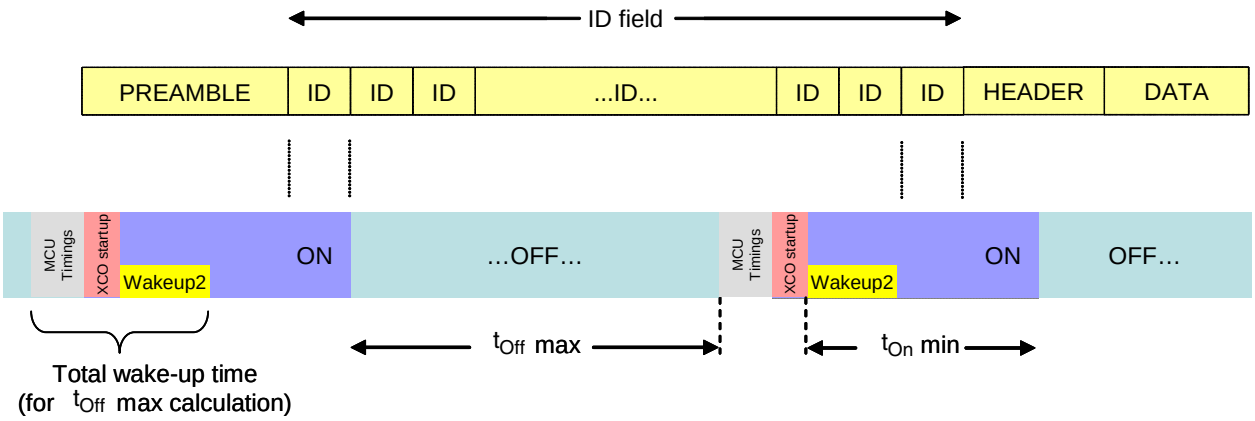


Figure 11. t_{On} Min and t_{Off} Max Representation with MCU Timings Included

And t_{Off} max formula becomes:

$$t_{Off} = t_{IDField} - t_{Wakeup} - 2 * t_{ID}$$

with:

$$t_{Wakeup} = t_{Wakeup1} + t_{Wakeup2} + t_{MCU}$$

and:

$t_{Wakeup1}$: XCO startup only

$t_{Wakeup2}$: PLL lock time + preamble

t_{MCU}: MCU wakeup + receiver reconfiguration time

In summary, the user will have to size the on and off times according to which configuration (continuous running of the MCU or MCU wakeup) is used.

3.4 External Capacitor Choice

The strobe oscillator clocks only the off time while the on time is clocked by the crystal oscillator.

The strobe oscillator is a relaxation oscillator in which an external capacitor is charged by an internal current source. When a threshold is reached, this capacitor is discharged through an internal resistive path, and the cycle restarts.

Nominal value for the strobe oscillator capacitor is 1 nF. This gives a strobe period of $t_{Strobe} = 10^6 * 1\ nF = 1\ ms$. Receiver off time is derived from this strobe period, and the user can change the capacitor value to closely match the desired off time with the actual off time that is feasible to reach, as the off time formula shows:

$$t_{Off} = N * t_{Strobe} + \min(t_{Strobe} / 2, t_{On})$$

with N coming from RXONOFF register. We will see an example of this capacitor sizing in Section 4, “System Sizing in Numerical Examples.”

4 System Sizing in Numerical Examples

This section provide practical examples of the recommended way to configure the MC33596 receiver according to the transmitted frame, especially the management of both MCU wakeup and the strobe oscillator.

Two examples are described, transmitted frames with and without header included, as the header is mandatory in MC33596 frame recognition.

4.1 Example 1: Transmitted Frame with Header

Let's consider a frame compatible with MC33591/2/3/4 receivers, in other words including an identifier coded on eight bits fixed length, and a header coded on four bits fixed length whose value is fixed as well and equal to 0110.

We take for example the followed transmitted frame to be received:

- ID_{TX} = B9 = 10111001
- HD_{TX} = 0110

With MC33596, ID length is programmable among two, four, five, and six bits. The header is mandatory and its length is also programmable among one, two, four, or six bits.

4.1.1 MCU Wakeup and MC33596 Reconfiguration

4.1.1.1 Conf1, MCU Wakeup

The first step is to allow the MCU to wake as soon as a useful incoming message is detected by the receiver. But to avoid false wakeup occurrences or at least to minimize them, the ID length has to be long enough. Let's take for example the maximum length, six bits.

After the ID has been detected, the MCU needs an interrupt request coming from the receiver to reprogram it with Conf2. As already mentioned in Section 1.2.2.2.1, "Using Two Configurations, "this interrupt is taken from the SCLK line, which becomes active as soon as the header is detected.

Consequently, to react as quickly as possible, the user should choose the shortest possible header in Conf1, in other words one bit long. This header is used only to initiate the SCLK activation. With the ID_{TX} and HD_{TX} chosen, it could give:

- ID_{RX1} = 101110
- HD_{RX1} = 0 (or 1 — it does not really matter here)

Thus the register value for Conf1 will be set with the following hexadecimal values:

- ID_{RX1} register = EE (6 bits + binary value)
- HD_{RX1} register = 00 (1 bit + binary value)

4.1.1.2 Conf2, Data Reception

After the MCU has received the interrupt request, it configures the receiver in Conf2 much faster than the data rate (maximum SCLK frequency 1 MHz) by means of SPI orders.

In this Conf2 mode the receiver remains awake, because useful incoming data has been initially detected. Thus we advise forcing the strobe pin to a high level up to EOM detection, as illustrated in Figure 4 (step 5).

ID recognition is not relevant information anymore, so ID_{RX2} could be as short as possible, which is two bits. To increase discrimination between useful data and transmitted ID/header fields, the header in Conf2 should be as long as possible, six bits length. With the ID_{TX} and HD_{TX} chosen, it could give:

- ID_{RX2} = 10
- HD_{RX2} = 010110 (the beginning of the header is taken in the last bits of the transmitted ID pattern)

Thus the register value for Conf2 will be set with the following hexadecimal values:

- ID_{RX2} register = 02 (2 bits + binary value)
- HD_{RX2} register = D6 (6 bits + binary value)

After the HD_{RX2} has been detected by the data manager, all following data is decoded and finally sent to the MCU (with a clock at data rate on the SCLK line) through the MOSI line, until EOM detection.

4.1.2 Strobe Sizing

The strobe oscillator is used for Conf1 only, before any useful incoming data is detected. t_{On} and t_{Off} times have to be carefully dimensioned to guarantee both acceptable average consumption and that no relevant information is missed.

These timings and corresponding registers are obviously closely linked to the expected incoming signal characteristics. Let's take for example a carrier centered at 433.92 MHz, modulated in OOK at data rate 9600 bit/s, and an ID field that includes 80 transmitted IDs with the recommended format (80 * 8 bits in a continuous ID field).

4.1.2.1 Calculation of t_{On}

Remember that t_{On} duration is given by equation (2) as follows: $t_{On} = 2 * t_{IDRX1} + t_{Wakeup2}$.

A bit lasts around 104µs with the data rate chosen, and ID_{RX1} is coded on 6 bits, thus $t_{IDRX1} = 6 * 104 = 625$ µs.

As explained in Section 3, "Strobe Mode," the on time is clocked by the crystal oscillator. Thus to be set, XCO startup must be done. So wakeup time necessary for the minimum on time calculation must include:

- RF stabilization: PLL lock time at 100 µs max
- Analog stabilization: AGC needs 200 µs min to stabilize and data slicer reference needs 3 bits to stabilize
- Digital stabilization: clock recovery needs 1 bit to stabilize

Consequently:

$$t_{On} = 2 * 0.625 + 0.1 + 0.2 + 3 * 0.104 + 0.104 = 1.966 \text{ ms}$$

We notice that the higher the data rate, the higher the impact of stabilization time on the on time, because of the constant stabilization times required for PLL and AGC.

In this case, the wakeup time used in the on time calculation represents about 36% of the total time. Therefore, a significant amount of the total time is used simply for stabilizing the receiver before receiving.

The on time is linked to the RON register by the followed formula (see datasheet):

$$t_{On} = RON[3:0] * 512 * t_{digclk}$$

with t_{digclk} = 1/604.767 = 1.65 µs for a carrier centered at 433.92 MHz.

Here RON = 2.33, and we round up to the nearest greater integer value, in other words RON[3:0] = 3.

Finally, with the actual RON value, t_{On} = 3 * 512 * 1.65 µs = 2.53 ms.

4.1.2.2 Calculation of t_{Off}

As described in Section 3.3.2, “Taking MCU Wakeup into Account in Strobe Sizing,” when MCU wakeup management is needed, the user has to take into account the MCU wakeup, as well as the receiver reconfiguration, in the maximum t_{Off} time calculation. That is the case in this example.

Remember the maximum off time formula: $t_{Off} = t_{IDField} - t_{Wakeup} - 2 * t_{ID}$.

And $t_{IDField}$ = 80 * 8 * 0.104 = 66.7 ms, as the transmitter sends frames based on 80 (continuous) IDs constituted of eight bits each and clocked at 9600 bit/s.

The timings needed to perform MCU wakeup and receiver reconfiguration obviously depend on the MCU, and also on the number of registers to modify.

For example, let’s take an MCU wakeup in 2 ms and operations in configuration mode done at maximum speed, which is 1 MHz.

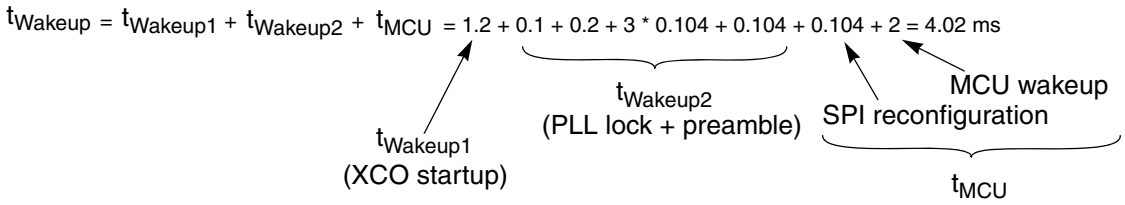
Two registers (ID and HD registers) have to be changed to go from Conf1 to Conf2. But users generally choose to completely rewrite the configuration to secure the system. Notice this is almost hidden time, as the SCLK clock is much faster than bit time (1 MHz compared to a data rate of a few kHz).

For example, changing two consecutive registers takes around 3 * 8 * 1 µs = 24 µs (see Figure 1), and reconfiguration of all registers takes about five times longer. Let’s take one bit duration (if some of them are not rewritten).

Consequently, MCU timings take 2 ms + 1 bit duration, and the off time calculation becomes:

$$t_{Off} = t_{IDField} - t_{Wakeup} - 2 * 0.625$$

with:



Notice that the MCU wakeup chosen represents 50% of the total wakeup time needed for the maximum off time calculation.

Thus, the maximum off time is t_{Off} = 66.7 – 4.02 – 1.25 = 61.43 ms in this example.

To cover the worst case, and in particular a process shift, one needs to take into account the strobe oscillator accuracy — the MC33596 datasheet gives a possible shift of 15.8% maximum. Consequently, maximum t_{Off} is weighted as:

$$t_{Off\_max} = 61.43 / 1.158 = 53.05 \text{ ms}$$

And real off time is linked to the strobe capacitor by means of the following formula:

$$t_{Off} = N * t_{Strobe} + \min (t_{Strobe} / 2, t_{On})$$

with N closely linked to the ROFF register, as shown in Table 21 of the MC33596 datasheet. From the maximum off time previously calculated and with the associated formula above, one has to properly choose both ROFF[2:0] value and the strobe capacitor.

Let’s take the maximum value for ROFF, which is 63. t_{Strobe} must be lower than 53.05 / 63 = 0.842. Thus, with an 820 pF strobe capacitor, t_{Off} becomes 63 * 0.82 + min (0.82/2, 2.53) = 52.07 ms, which is less than the t_{Off_max} allowed (53.05 ms).

Referring to Figure 6, SR = 52.07 / 2.53 = 20.6 giving an average current of I_{Mean} around 510 µA using this ROFF value and strobe capacitor. These choices optimize average consumption but can seem not completely safe, as dispersion on the capacitor value has not been taken into account, and also the final t_{Off} value is not really below the maximum allowed by the system.

There is an advantage to choosing a lower ROFF, 32 being the next lower value, and increasing the strobe capacitor value as follows. t_{Strobe} must be lower than 53.05 / 32 = 1.658. Thus, with a 1.5 nF strobe capacitor, t_{Off} becomes 32 * 1.5 + min (1.5 / 2, 2.53) = 48.75 ms, which is more comfortable regarding the maximum off time allowed — 53.05 ms. In this case SR = 48.75 / 2.53 = 19.27, giving an average current of I_{Mean} around 541 µA with this ROFF value and strobe capacitor. The slight over-consumption versus the previous choice is compensated by a more reliable sizing of the system.

4.2 Examples Derived from Example 1

4.2.1 Transmitted Frame without Header

We use the example here of a transmitted protocol compatible with the MC33591/2/3/4 receiver, with an identifier coded on eight bits fixed length and without a header.

- ID_{TX} = B9 = 10111001

Conf1 remains the same as in example 1.

Conf2 becomes, ID_{RX2} = 10 and HD_{RX2} = 111001 (because it took in the ID_{TX} field).

Remember that in this case the MCU will not necessarily receive only fully useful data, because even if H_{DRX2} is taken from an ID content, this ID is not necessarily the last one in the ID field. This last uncertainty regarding the source of data sent to the MCU is completely linked to the incoming frame without a header.

4.2.2 How to Decrease Average Consumption

Based on t_{On} and t_{Off} formulas, we can deduce three direct ways to reduce average consumption, without changing incoming frame characteristics in terms of modulation type or data rate:

- Choose an MCU with a faster wakeup
- Reduce the number of bits in received ID content
- Increase number of IDs in the transmitted ID field

Let's evaluate the impact of each of these on the average consumption.

4.2.2.1 Faster MCU Wakeup

First of all, we choose a faster MCU wakeup of 500 µs (versus 2 ms for the initial one). This reduction of the MCU wakeup can be achieved by using, for example, an internal unstable multivibrator instead of an external crystal.

t_{On} remains the same as in example 1.

Only t_{Off} is impacted through a t_{Wakeup} calculation that becomes:

$t_{Wakeup} = t_{Wakeup1} + t_{Wakeup2} + t_{MCU} = t_{Wakeup1} + t_{Wakeup2} + 0.5$ = 2.52 ms instead of the previous 4.02 ms.

Consequently, maximum off time becomes t_{Off} = 66.7 – 2.52 – 1.25 = 62.93 ms, and with uncertainty on the strobe oscillator, t_{Off} = 62.93/1.158 = 54.34 ms instead of the previous 53.05 ms.

Referring to the off time calculation described in Section 4.1, "Example 1: Transmitted Frame with Header," and especially to the choice of ROFF and strobe capacitor values, one could take advantage of this faster wakeup of the MCU by choosing the first configuration (ROFF = 63 and strobe capacitor 820 pF) and thus lower average consumption. Indeed, the margin becomes larger regarding actual off time versus the maximum allowed (maximum 54.34 ms versus real one 52.07 ms).

4.2.2.2 Reduce Number of Bits in ID

Let's choose an ID_{RX1} coded on four bits instead of the previous six bits.

t_{On} becomes t_{On} = 2 * (4 * 0.104) + 0.1 + 0.2 + 3 * 0.104 + 0.104 = 1.55 ms instead of the previous 1.97 ms.

This makes RON = 1.55/(512 * 1.65 µs) = 1.83, and we choose the next higher integer value: RON[3:0] = 2.

Consequently, t_{On} = 2 * 512 * 1.65 µs = 1.69 ms instead of the previous 2.53 ms.

The maximum off time calculation becomes:

$t_{Off} = t_{IDField} - t_{Wakeup} - 2 * (4 * 0.104) = 66.7 - 4.02 - 0.832 = 61.85$ ms, and with uncertainty on the strobe oscillator, $t_{Off} = 61.85/1.158 = 53.41$ ms.

Maximum off time being slightly increased, we choose here to keep the previous configuration, which has less power consumption: 820 pF for strobe capacitor with ROFF = 63. It produced 510 µA average consumption.

Here SR becomes 52.07 / 1.69 = 30.81, giving an average current around 358 µA, which is much better than the original value but with less robustness regarding false wakeup (four bits in the ID instead of six).

4.2.2.3 Increase ID Field Length

The last basic way to reduce receive power consumption consists of increasing ID field length, thus reporting the consumption in the transmitter side (in RKE applications for instance). Let's take an ID field based on one hundred consecutive IDs instead of the previous 80 (instantaneous consumption of the transmitter increased by 25%).

t_{On} remains unchanged from example 1.

Maximum off time becomes $t_{Off} = t_{IDField} - t_{Wakeup} - 2 * 0.625 = 100 * 8 * 0.104 - 4.02 - 1.25 = 77.93$ ms.

And sizing ROFF register and strobe capacitor to meet an off time around 75 ms gives SR = 75/2.53 = 29.64, and an average consumption around 370 µA (compared with more than 500 µA with the original configuration).

Finally, with an additional 25% consumption in the transmitter side (100 compared to 80), the savings in the receiver side is between 25% and 30%. As already discussed, this last technique can be used in cases where there is no crucial power consumption in transmit, for instance in RKE systems.

5 Configuration Switching

This feature makes it possible to load two different configurations in two different register banks. This allows saving MCU consumption when the user wants to periodically check two kinds of incoming signals, because SPI access is no longer needed to reconfigure registers.

For example, two different frequencies, data rate and/or modulation type, can be preliminarily loaded to alternatively poll RKE and TPMS incoming signals with the following parameters:

- RKE, OOK, 4800 bit/s, 433.92 MHz
- TPMS, FSK, deviation +/– 35 kHz, 9600 bit/s, 433.92 MHz

Two sets of registers are grouped in two banks, BANK A and BANK B. Two bits, BANKA and BANKB, are available to define the receiver state as in the following truth table:

Table 1. BANKA and BANKB Bits Truth Table

BANKA	BANKB	Actions
X	0	BANK A is active
0	1	BANK B is active
1	1	Both BANK A and BANK B active one after the other

An additional bit, named BANKS, gives the current active bank (BANKS = 1 for BANKA active, BANKS = 0 for BANKB active). This bit is a read-only bit.

5.1 Ways to Switch Configurations

The switching of the two banks can be performed in several ways:

- Direct switch control — Set the strobe pin to high level, associated with SOE = 0. In this switching mode, the active bank is the one defined by the BANKA and BANKB values (BANKA = 1 and BANKB = 1 not allowed in this mode) through SPI access during configuration mode. The defined bank is active after leaving configuration mode, in other words after CONFB line is set to high level.
- Strobe pin switch control — Strobe pin managed by the MCU (with SOE = 1). In case of only one bank active (the two first rows in Table 1), the receiver will be alternatively switched between the active bank and the off status according to the strobe pin level.
- Strobe oscillator switch control — Strobe pin connected to external capacitor (with SOE = 1). In case of only one bank active (the two first rows in Table 1), the receiver will be alternately switched between the active bank and the off status according to the on/off state defined by the strobe oscillator timings.

5.2 Sequences When Both BANK A and BANK B Are Activated (BANKA = BANKB = 1)

This case represents the common case when both banks are used. The purpose is to switch alternatively and automatically (without reconfiguration of the registers) between the two banks.

5.2.1 Strobe Pin Switch Control (SOE = 0)

In this case, the MCU manages the on/off periods by means of the strobe pin level.

The receiver is switched on with alternately one or the other bank.

t_{On} and t_{Off} are forced by the MCU.

The BANKS bit is changed after each on duration.

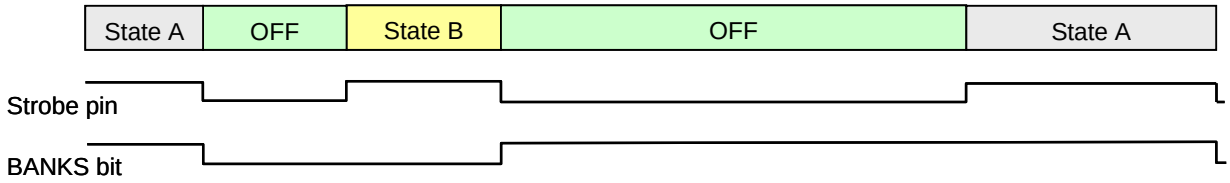


Figure 12. Receiver State When the Two Banks Are Activated and Switched by Means of MCU through the Strobe Pin Activation (SOE = 0)

5.2.2 Strobe Oscillator Switch Control (SOE = 1)

In this case, the strobe oscillator manages the on/off periods by means of an external capacitor for time constant definition.

State A and State B on periods are set by RON_A and RON_B respectively.

State off duration is set by ROFF_A exclusively (ROFF_B has no effect).

The BANKS bit is toggled at the end of each State A or State B.

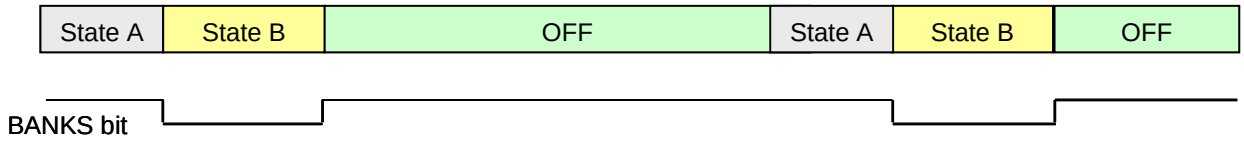


Figure 13. Receiver State When the Two Banks Are Activated and Switched by Means of Strobe Oscillator (SOE = 1)

If a message is being received during State A or State B, the current state remains active up to the end of message (EOM).

6 Received Signal Strength Indicator (RSSI)

An RSSI feature has been implemented in MC33596, allowing polling of the strength of the incoming signal. The RSSI feature is enabled by setting the RSSIE bit to 1.

6.1 Principle: Analog and Digital Forms

The incoming signal is measured at two different locations of the receiver lineup according to the range of the input power:

- At the IF filter output, by means of a progressive compression logarithmic amplifier for input power range from sensitivity level up to around –60 dBm.
- At the LNA output, by means of LNA AGC control voltage for input power range from around –60 dBm up to around –30 dBm.

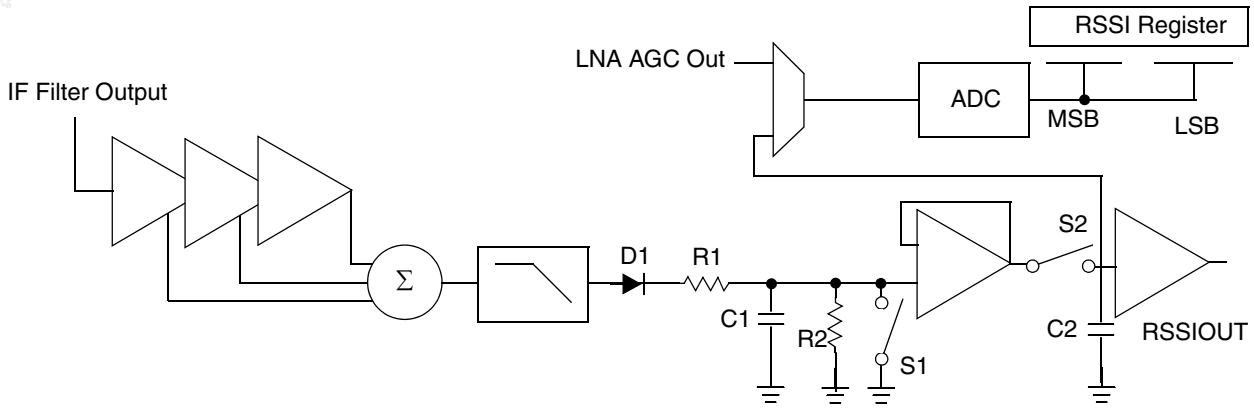


Figure 14. RSSI Simplified Block Diagram

The information provided by the logarithmic amplifier is available both on the RSSIOUT pin in analog form, but also in the RSSI register (at the 4 LSB) in digital form.

Higher level range (from –60 dBm) is available only in digital form in RSSI register (at the 4 MSB).

Figure 15 shows RSSI information given in analog form.

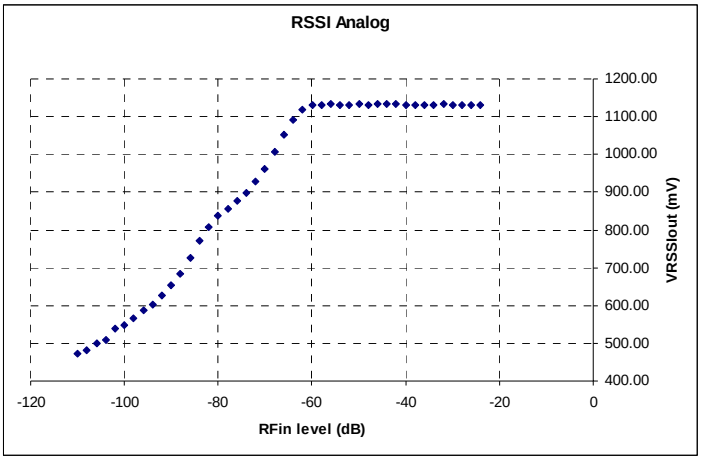


Figure 15. RSSIOUT Profile versus Input Power Strength

The digital form is coded on four bits, providing around 2 dB/bit or 3 dB/bit resolutions according to the range of the incoming strength (MSB or LSB). The user can take advantage of using the analog form to increase this resolution by associating RSSIOUT voltage to an external ADC that codes analog information with more than four bits.

6.2 Acquisition Modes, Continuous and Pulsed

Two acquisition modes of the RSSI are available in MC33596 — the continuous mode and the pulsed mode (or sample).

6.2.1 Continuous Mode

After the RSSI is enabled (RSSIE = 1), maintaining the RSSIC pin at high level allows for continuous measurement of the RSSI.

Referring to the RSSI circuit block diagram (Figure 14), the quasi peak detector (D1, R1, C1) is reset by closing S1. Then, after a 9 µs reset time, the RSSIOUT voltage reaches the follower input in a 20 µs rising time (R1C1), corresponding to the peak detector time constant. S2 is closed as soon as RSSIC is set high, and RSSIOUT voltage follows the peak value with a decay time constant of 5 ms (R2C1). Once RSSIC has been returned to low level, the RSSIOUT voltage drop with 300 µV/ms decay rate (C2 discharge).

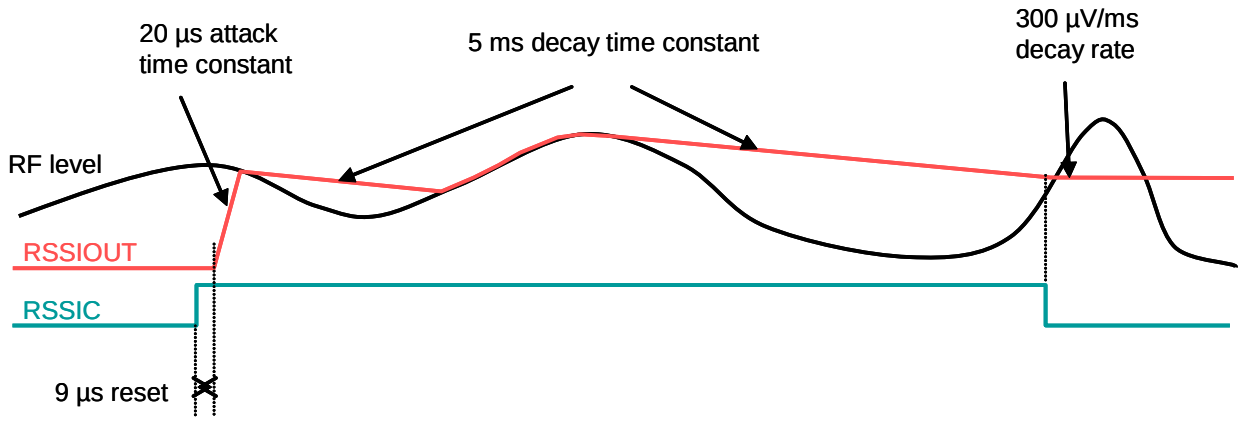


Figure 16. RSSI Behavior in Continuous Mode

Notice variations have been deliberately enlarged to highlight different timings. In real cases, RF level does not vary as much as represented in Figure 16, and RSSIOUT remains almost constant during the actual RF transmitted frame.

To store the digital form of the RSSI, users must schedule in their applications the management of the RSSIC pin by means of the MCU, even in this continuous mode (as it is done for sample mode). Indeed, the RSSI value is stored in the associated register only on the falling edges of the RSSIC signal. SPI access in read mode will then be necessary to catch the RSSI value in digital form in such a manner that it is impossible to have both RF reception and digital RSSI update at the same time.

This continuous mode of analog form acquisition can be very helpful, for instance, when the user has to optimize an input matching network. Indeed, sensitivity is not easy to catch, because the user needs to vary both input RF level and the matching network in order to evaluate the best case. It can be much easier to fix the input level first, and then to tune the matching network after the maximum analog form of the RSSI is reached. We therefore recommend to use this method to find the optimized matching network for each application.

6.2.2 Sample Mode

The sample mode consists of pulsing the RSSIC. Timings remain as described in the continuous mode. Consequently, RSSIOUT is reset at the beginning of each acquisition period (corresponding to high level on RSSIC), and sampled (with a decay time of 300 µV/ms) after these acquisition periods, as illustrated in Figure 17.

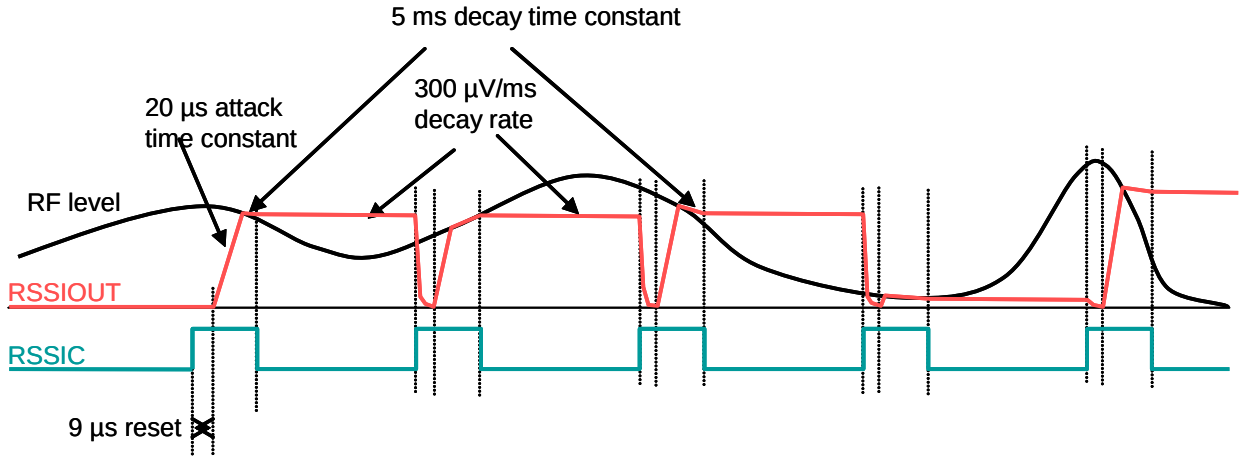


Figure 17. RSSI Behavior in Sample Mode

The very fast reset time allows using this sample mode measurement to help discriminate among tires (localization) in TPMS applications, by developing smart envelope detection techniques.

7 Frequency Addressing

Two ways for addressing receiver frequencies are implemented in MC33596 according to FRM bit values as follow:

- FRM = 0, Friendly access mode
- FRM = 1, Direct access mode

In both access modes the only register to fill is the F register (located at addresses $04 and $05, coded on 12 bits).

7.1 Friendly Access Mode

F register is supposed to directly program receiver frequency, but in reality this frequency will be approximated from the real one.

Indeed, the local oscillator (LO) frequency is derived from the IF frequency by $f_{lo} = f_{RX} + f_{IF}$. But in this mode f_{IF} is fixed at 1.5 MHz (hard-coded), whereas in reality the value is derived from crystal frequency f_{ref} according to the VCO range chosen through the following formulas:

- $f_{IF} = f_{ref}/9 * 1.5/2$ if CF0 = 0
- $f_{IF} = f_{ref}/12 * 1.5/2$ if CF0 = 1

Consequently, the approximation on the IF frequency will result in the same approximation on the final receiver carrier frequency.

For example, with the crystals recommended in the datasheet, in other words 17.5814 MHz for 315 MHz and 24.19066 MHz for 433.92 MHz, actual values for f_{IF} are 1.465 MHz and 1.512 MHz respectively, instead of 1.5 MHz hard-coded. Finally, the RX carrier frequency will be shifted from the desired one by

35 kHz at 315 MHz and 12 kHz at 433.92 MHz. Nevertheless these shifts should not result in significant loss in terms of sensitivity.

7.2 Direct Access Mode

In this mode, the user directly enters the local oscillator frequency, and the receiver carrier frequency is computed from the values of this frequency and the real IF frequency.

Considering the examples above, if the user wants to program the receiver frequency at 315 MHz or 433.92 MHz, he will have to take into account the IF frequency, in other words 1.465 MHz and 1.512 MHz respectively, and program values for F register of 316.465 MHz and 435.432 MHz.

The F register is linked to the desired frequency with the formula: F register = $(2 * f_{lo} / f_{ref} - 35) * 2048$.

Using the examples given, we get:

- F register = 2047.95 approximated to 2048 for 315 MHz
- F register = 2048.02 approximated to 2048 for 433.92 MHz

With the approximation due to binary conversion, for receiver carriers we get 315.0000833 MHz and 433.9199638 MHz respectively, thus 83.3 Hz and 36.25 Hz shifts from transmitted frequencies.

Notice this direct access mode is more robust in terms of accuracy and is thus recommended.

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN3603
Rev. 0
03/2008

freescale™
semiconductor