

## AX-RadioLAB Gen2 for AX5051 User Manual



**ON Semiconductor®**

[www.onsemi.com](http://www.onsemi.com)

---

### APPLICATION NOTE

#### Introduction

The AX-RadioLAB is a hands on configuration and firmware source code generation tool for the AX5051 transceiver IC.

It creates C-code firmware projects based on the AXRadio V2 API (described in DOCU/AXRadioV2API.pdf), which demonstrate the use of the AX5051 transceiver together with the AX8052F100 micro controller for packet transmission and reception in several modes. Packets can be transmitted periodically or upon pressing a button. On the receiver side the available options are:

- RX Continuous (RX is always on)
- Wake on Radio
- RX Synchronized with TX (The RX is on for the time slots in which packets are expected only)

Optionally received packets can be answered with an acknowledge packet in all modes.

AX-RadioLAB allows direct downloading of the created firmware to AXSEM DVK-2 main boards, where it can be run and tested immediately. A dedicated firmware project implements basic tests like putting the transmitter into CW mode or measuring bit error rates.

The first step in working with AX-RadioLAB is to create a new AX-RadioLAB project. Firmware projects implementing the TX and RX mode selected in the AX-RadioLAB GUI are created inside your project directory.

The AX-RadioLAB GUI makes it easy to configure physical parameters of the wireless link as well as the framing format and various possible functions of the AX5051 general purpose pins.

Optionally you can view and modify the generated firmware source code using the AXCode::Blocks IDE.

The firmware also makes a nice starting point for your own development. You can always use AX-RadioLAB to modify wireless parameters of your project, even after modifying the firmware.

### INSTALLATION

**Software requirements:** For AX-RadioLAB to work correctly you need to install the AX8052-IDE first. Install all sub-components of the AX8052-IDE (AXSDB, SDCC, driver; AXCode::Blocks).

Execute the setup.exe to start the installation of AX-RadioLAB. If AX-RadioLAB was already installed,

the installer automatically uninstalls previous versions of AX-RadioLAB. Run setup.exe again to install. At the end of the each process you will be asked to reboot your computer. This is not necessary.

RADIOLAB MAIN PANEL

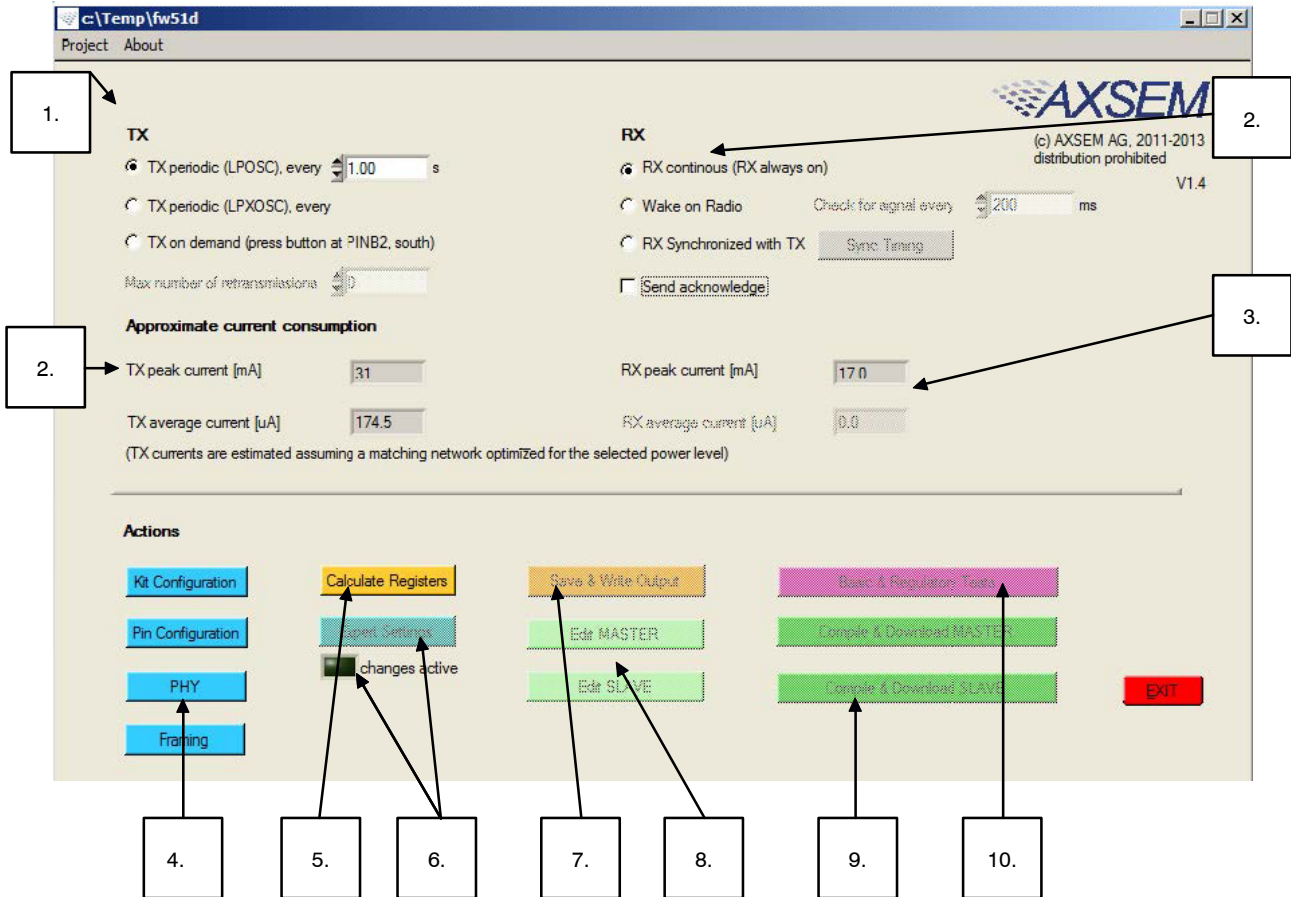


Figure 1. Radiolab Main Panel

Start a new project using the *Project* menu. You have to specify a project directory where AX-RadioLAB will create firmware projects called MASTER, SLAVE and TESTS. To open an existing project use the directory chooser dialog to descend into the project directory (where the *axradiolabstate.xml* file resides) and hit done. Note: do not open the *template\_firmware\_AX5051* project inside the AX-RadioLAB installation directory, as this is the template from which the firmware is copied upon creating a new project.

Follow through the main panel step-by-step to configure your project:

1. Select the TX Operation Mode:

- TX Periodic (LPOSC):  
Periodic packet transmission driven by the MCU's low power RC oscillator.
- TX Periodic (LPXOSC):  
Periodic packet transmission driven by the MCU's low power 32 kHz tuning fork crystal oscillator
- TX on Demand:  
Packet transmission upon pressing a button.

2. Select the RX Operation Mode:

- RX Continuous (RX always on):  
Packet reception keeping the receiver continuously on. This mode is used for AC-powered systems and allows asynchronous packet reception with low latency.
- Wake on Radio:  
Packet reception with the receiver waking up periodically and going immediately back to sleep if no signal is detected. This mode is well suited e.g. for infrequently operated remote control applications and achieves very low (idle) current consumption. *Mind that preambles longer than the time between RX wake-ups are necessary.* The latter also determines the RX latency.
- RX Synchronized with TX:  
Periodic packet reception enabling the receiver only for the time slots in which packets are expected. This is appropriate whenever periodic data transmission is required. Timing is driven by the MCU's low

power 32 kHz tuning fork crystal oscillator and thus requires MCU activity. It is mandatory to use *TX periodic (LPXOSC)* on the transmitter side.

TX and RX modes are summarized in Table 1. *Wake on Radio* and *RX Synchronized with TX* modes are described in more detail in section Wake on Radio Mode (WOR) and RX Synchronized with TX Mode.

The MASTER and SLAVE firmware projects are created in your project directory (if not present yet) upon hitting the *Save & Write Output* button. Manual changes in the firmware projects are never overwritten by the GUI except for the configuration files in *your\_projectdir/AX\_Radio\_Lab\_output/*.

The *Send acknowledge* checkbox causes the slave module (“RX”) to answer successful packet reception with an acknowledge packet. The master module (“TX”) tries to receive the acknowledge packet. The smiley on the LCD indicates whether the acknowledge packet has been received. Failure is also signaled via LED2. Optionally the MASTER can retransmit a packet if acknowledge fails. This feature is enabled by setting *Max number of retransmissions* to a nonzero value. The acknowledge mode is the reason why the terminology MASTER and SLAVE (rather than TX and RX) has been chosen. The acknowledge feature can be selected in all TX and RX modes.

3. Estimated current consumption is displayed here. The peak current values are drawn when TX or RX are running. For duty cycled modes the estimated average currents are also displayed. In *Wake on Radio* mode the average RX current is the idle current, assuming a silent channel. In *RX Synchronized with TX* mode receiving a packet in at each wake-up is assumed. If *Send acknowledge* is selected the average RX current also includes transmission of the default acknowledge packet and the average TX current includes the corresponding receive current. Currents drawn by TCXO and MCU are not included. TX currents assume a matching network optimized for the selected power level. Mind that the default matching network on the AXSEM RF modules is not optimal for reduced power levels. This leads to reduced power amplifier efficiency and thus currents higher than estimated.
4. Buttons to open configuration panels for Kit Configuration, Pin Configuration, PHY layer and Framing. The PHY panel covers physical parameters of the wireless link such as carrier frequency, data rate and the modulation scheme. The Framing panel deals with packet delimiting, CRC etc.

5. *Calculate Registers* calculates register values to configure the AX5051 according to your Pin Configuration, PHY and Framing selections.
6. After calculating register values you can optionally tweak values using *Expert Settings*. Click values in the rightmost column of the *Expert Settings* panel to edit. Tweaked values are sticky. They keep overriding the defaults even if the defaults have changed (e.g. upon making changes in the PHY panel and hitting *Calculate Registers*). Tweaked values are highlighted in yellow (or green if they accidentally correspond to the default values). The *changes active* indicator in the main panel reminds you that tweaked values are being used. Double click a tweaked register (column 1, 2 or 3) in order to return to the default value. Hitting *Reset Changes* at the bottom of the *Expert Settings* panel or clicking the *changes active* indicator in the main panel drops all tweaks.
7. *Save & Write Output*:
  - Saves configuration and register values into *your\_projectdir\axradiolabstate.xml*, allowing you to reopen your project with AX-RadioLAB.
  - Outputs files *config.c*, *configmaster.[ch]* and *configslave.[ch]* to *your\_projectdir/AX\_Radio\_Lab\_output/*. These files are included by/linked to the MASTER, SLAVE and TESTS firmware projects.
  - The firmware projects MASTER and SLAVE are written into your project directory if they are not present yet.
8. The *Edit MASTER* and *Edit SLAVE* buttons open the corresponding firmware project from your project directory using the AXCode::Blocks IDE. This is handy for tweaking the firmware code, but touching the code is not necessary for first experiments with packet transmission. Instead you can download the demo firmware to the MCU directly from AX-RadioLAB (see next point). Button labels adapt to show which firmware projects are currently used.
9. The *Compile & Download MASTER* and *Compile & Download SLAVE* buttons compile and download the corresponding firmware onto the main board connected to the AXDBG USB debug adapter. Note that *Compile & Download* does not work if AXCode::Blocks is currently opened. In this case just use *Save & Write Output* and then compile and download the code from within AXCode::Blocks.
10. Opens the *Basic & Regulatory Tests* panel, allowing you to transmit CW, simple patterns or random data and to measure bit error rates.

**Table 1. SUMMARY OF TX AND RX MODES AND CORRESPONDING FIRMWARE PROJECTS**

Mode, Purpose	Used Timer	Radio/MCU Activity (Note 1)
TX periodic (LPOSC), Periodic packet transmission	MCU libmfwtimer infrastructure with timer0 clocked by MCU LPOSC (640 Hz RC oscillator)	MCU periodically wakes up, powers up the TX and initiates packet transmission
TX periodic (LPXOSC), Periodic packet transmission (used with RX Synchronized with TX)	MCU libmfwtimer infrastructure with timer0 clocked by MCU LPXOSC (32 kHz tuning fork crystal oscillator)	MCU periodically wakes up, powers up TX and initiates packet transmission.
TX on demand, Packet transmission upon pressing a button or similar event.	MCU libmfwtimer infrastructure with timer0 clocked by MCU LPOSC (640 Hz RC oscillator) Only used for generating a timeout in ACK mode	MCU wakes up upon pressing a button, powers up the TX and initiates packet transmission.
RX Continuous, AC-powered systems (allows asynchronous packet reception)	MCU libmfwtimer infrastructure with timer0 clocked by MCU LPOSC (640 Hz RC oscillator) Only used for ACK timing and channel state polling.	RX is continuously on. Upon receiving a valid packet it generates an interrupt waking up the MCU.
Wake on Radio, Remote control (asynchronous packet reception at low idle currents at the expense of long preambles and increased latency)	MCU libmfwtimer infrastructure with timer0 clocked by MCU LPOSC (640 Hz RC oscillator) Only used for generating a timeout in ACK mode	MCU periodically wakes up, powers up the RX, checks for a preamble signal. If found, RX is kept running for some time to receive the following packet. If not found, RX is immediately powered down.
RX Synchronized with TX, Low power periodic data transmission	MCU libmfwtimer infrastructure with timer0 clocked by MCU LPXOSC (32 kHz tuning fork crystal oscillator)	MCU periodically wakes up, powers up the RX and waits for packet reception or timeout.

1. MCU activity apart from timekeeping wake-ups due to the running libmfwtimer infrastructure.

**Wake On Radio Mode (WOR)**

In *Wake on Radio* mode the receiver periodically wakes up and searches the channel for a signal. In absence of signal the receiver is switched off again immediately in order to save power. If a matching preamble is detected the receiver stays on and looks for a valid packet delimiter. A typical application of this mode would be an infrequently operated remote control. Make sure you understand the following points:

1. An important part of the process of powering-up the receiver and deciding whether a preamble signal is present takes a time inversely proportional to the data rate. To achieve minimum current consumption you should therefore use the maximum possible data rate (100 kbps or larger).
2. Frequency tracking (AFC) requires settling time, which adds to the receiver start-up time and therefore increases the average current consumption. Large data rates and precise frequency references render frequency tracking unnecessary. By default frequency tracking is disabled if  $2 * XTAL \text{ precision} * f_{\text{carrier}}$  is smaller than 10% of the FSK half-deviation.

3. The average receiver (idle) current is proportional to the wake up frequency. On the other hand in the remote control mentioned above, the wake up frequency determines how quickly the receiver will react.
4. To guarantee reception the transmit preamble has to be longer than the receiver wake-up period. AX-RadioLAB sets the preamble length to 120% of the wake up period.
5. Mind that the current estimate displayed in the main panel is the idle current in absence of a signal. Thus while the RX wakes up periodically, actual packet reception is the exception in this mode, occurring only e.g. if a remote control button was pressed. This is in contrast to the *Synchronized with TX* mode, in which a packet is expected each time the RX wakes up.

**RX Synchronized with TX Mode**

The purpose of *RX Synchronized with TX* mode is periodic reception of packets with minimal current consumption by enabling the RX only for the time slots in which packets are expected. Short preambles can be used on the TX side<sup>2</sup>. The following points explain how the *RX Synchronized with TX* mode works:

<sup>2</sup>This contrast to the asynchronous nature of *Wake on Radio* mode, where a preamble at least as long as the RX sleep interval is required.

- *RX Synchronized with TX* mode is used together with *TX periodic (LPXOSC)* on the TX side. Timing is driven by the MCU's low power 32 kHz tuning fork crystal oscillator which ensures an accurate packet frequency and thus allows for tight margins in the time slots in which the RX is enabled.
- After power-up LPXOSC is settled and the RX is switched continuously on until a packet is received or a timeout occurs. LED3 indicates the RX is running. When a packet is received the RX is put to sleep and only wakes up again for the time slot in which the next packet is expected.
- With each further packet reception the RX measures the effective time elapsed since the last packet and (via low pass filtering) corrects its own wake-up frequency accordingly. This corrects for crystal variations as well as temperature differences and slow gradients, and minimizes required margins in the RX-on time slots. (An increased margin is used for the packet reception right after synchronization, where the effective period has not been measured yet.)
- If no packet is received during an RX-on time slot, the RX margins are increased for the next time slot. If no packet is received during N subsequent time slots, the RX module switches back to synchronization mode, i.e. the RX is switched on continuously until a packet is received or a timeout occurs. (This synchronization mode is the same as for initial synchronization after power up. The only difference lies in the timeout interval: A timeout (in seconds) can be configured for initial synchronization after power-up (e.g. time to put batteries into the TX module). A second timeout (in packet periods) applies for re-synchronization after losing multiple packets in a row.
- If (re-)synchronization fails, (no packet is received until the timeout), the RX module is put to sleep for a configurable, typically long, period before trying to synchronize again. The rationale here is to limit RX power consumption in the case where the TX is off or out of reach.
- In *RX Synchronized with TX* the LCD displays the usual packet information. Pressing and holding button SW03 (south button on DVK-2 mainboards, connected to PINB2) during packet reception switches to the packet timing information display:
  - “T:” is the current, low pass filtered packet period deviation measured in LPXOSC periods. Thus it indicates by how much the interval between packets is (on average) longer than programmed value from the perspective of the RX. It is understood, that the position of the next time RX on time slot is corrected by this deviation. Thus if TX and RX clocks run at slightly different but constant speeds, the corresponding timing deviation is displayed here, but the RX on time slots are perfectly aligned to the actual packets.
  - “t:” indicates by how many LPXOSC cycles the last packet was late compared to the expected time, i.e. by how much it was misaligned to the RX on time slot. After acquisition of “T:” nonzero values of “t:” are due to timing fluctuations, which have fit into the configured RX margin.

The *Sync Timing* button opens the Sync Timing panel, where various timing parameters can be configured.

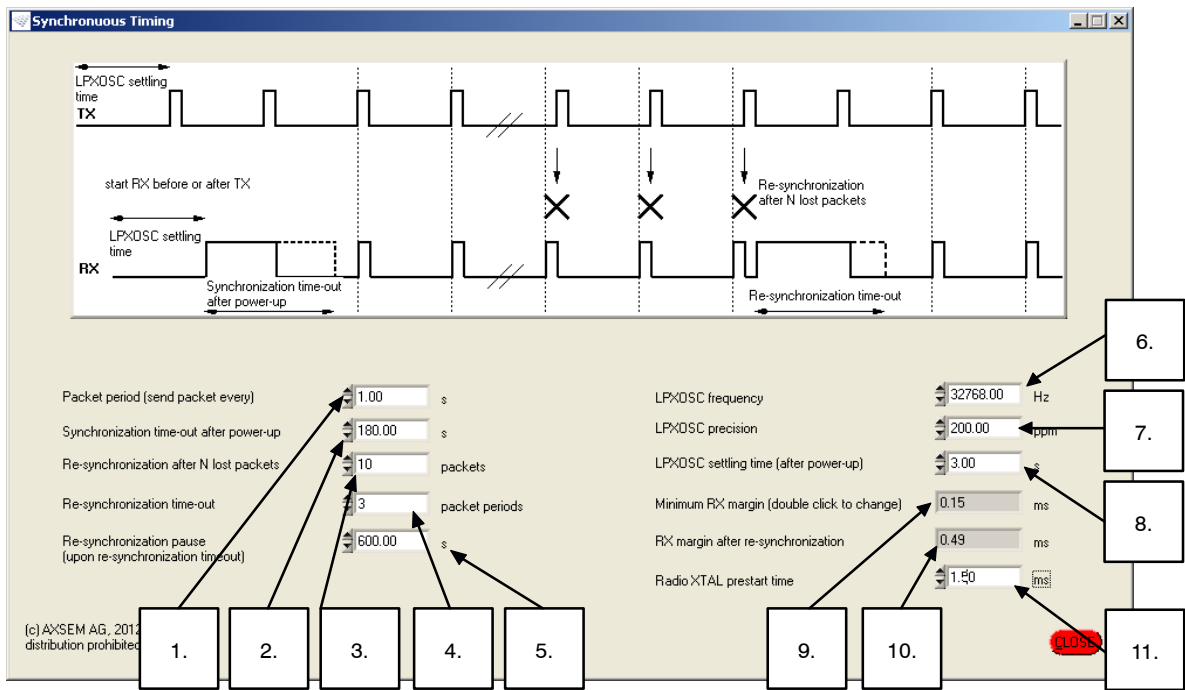


Figure 2. Synchronous Timing

1. **Packet Period (Send Packet Every):**  
This is simply a duplicate of the *TX periodic (every)* field in the main panel for convenience.
2. **Synchronization Time-Out after Power-Up:**  
Time the RX tries to receive the first packet, before temporarily giving up and switching to re-sync pause mode. E.g. this is the time you have to put batteries into the TX module after powering up the RX.
3. **Re-Synchronization after N Lost Packets:**  
Determines in how many subsequent RX-on time slots packet reception can fail before the RX switches to back to synchronization mode.
4. **Re-Synchronization Time-Out:**  
Time the RX tries to receive a packet for (re-) synchronization before temporarily giving up and switching to re-sync pause mode.
5. **Re-Synchronization Pause (Upon Re-Synchronization Time-Out):**  
Time for which the RX sleeps after failing to re-synchronize before trying again.
6. **LPXOSC Frequency:**  
Frequency of the low power tuning fork crystal oscillator.
7. **LPXOSC Precision:**  
Precision of the low power tuning fork crystal oscillator.
8. **LPXOSC Settling Time (after Power-Up):**  
TX as well as RX modules wait for this time before transmitting/trying to receive packets. This is to prevent synchronization and period measurement with an insufficiently settled clock source.
9. **Minimum RX Margin:**  
RX margin determines how long the RX is running, before the expected packet start. The same margin is also used as a timeout: The RX is powered down if no frame delimiter is detected by the time it was expected plus RX margin. (RX running means synthesizer running and analog baseband settled, but no AGC, AFC and bit synchronization, since there is no signal yet). The minimum value specified here has to accommodate timing fluctuations in normal operation. A default value is computed from *packet period* and *LPXOSC precision*. Double click the field to change it manually. If packet reception fails in a given RX-on time slot, the firmware automatically increases the margin for the next time slot.
10. **RX Margin after Re-Synchronization:**  
An increased RX margin which is used for the RX-on time slot right after (re-) synchronization. This is necessary, since (re-) synchronization only aligns the phases of periodic TX and RX. The RX needs a further packet to measure and adjust the wake-up period. Therefore the margin of the first RX-on time slot has to be large enough for the full, uncorrected timing deviation.
11. **Radio XTAL Prestart Time:**  
Determines how long the XTAL reference oscillator of the radio chip is settled before the RX synthesizer is started.

## KIT CONFIGURATION PANEL

The screenshot shows a 'Kit Configuration' window with three columns: Master, Slave, and Basic Tests. Each column has a dropdown menu for 'Kit Type' and checkboxes for various output options. The 'Master' column has 'DVK-2b' selected and 'Output on LCD display' and 'Output on debug link' checked. The 'Slave' and 'Basic Tests' columns have 'DVK-2b' selected and all checkboxes unchecked. At the bottom, there is a checkbox for 'Individual Settings for Master / Slave / Basic Tests' which is unchecked, and a red 'CLOSE' button.

	Master	Slave	Basic Tests
<b>Kit Type</b>	DVK-2b	DVK-2b	DVK-2b
<b>Output on LCD display</b>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>Output via UART0</b>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>Output on debug link</b>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>Extra debug messages on debug link</b> (Radio state changes)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>Use MCU sleep instead of Stdby</b>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> Individual Settings for Master / Slave / Basic Tests			

(c) AXSEM AG, 2011-2013  
distribution prohibited

**CLOSE**

Figure 3. Kit Configuration

In the Kit Configuration panel you can select the type of debug kit and the display options being used. The settings can be set either individually for Master/Slave/Basic tests or the same for all.

*Available Options:*

- Kit Types:  
Currently DVK-2b only.
- Output on LCD Display:  
Packet numbers, bit and packet error rates, RSSI and other information is display on the LCD display of the DVK-2b.
- Output on UART0:  
Instead of sending the information to the LCD display it is send to UART0.
- The Usage of LCD and COM0 are Mutually Exclusive.
- Output on Debug Link:  
In addition to one of the displays, the information is also send to the debug link.
- Extra Debug Messages on Debug Link:  
This option produces extra information about the status change of the radio state on the debug link.
- *MCU Sleep Instead of Stdby* option decides if the MCU uses sleep mode instead Stdby mode.

# AND9357/D

## PIN CONFIGURATION PANEL

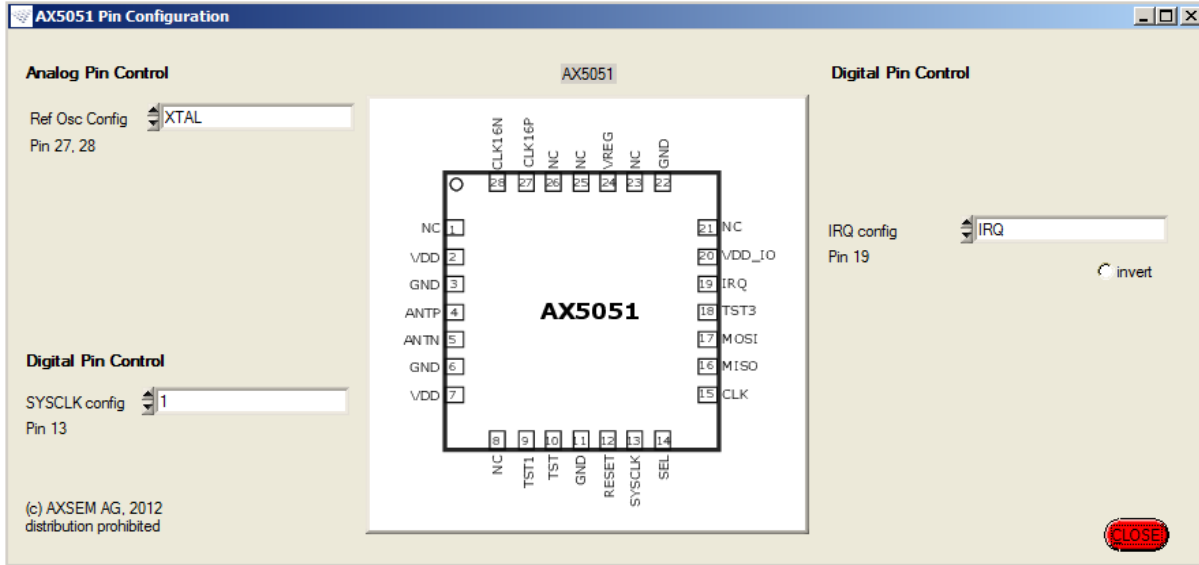


Figure 4. AX5051 Pin Configuration

- Ref Osc Config: select XTAL or TCXO
- IRQ Config: Output IRQ. No other setting is sensible if the AX5051 module is connected to the AXSEM main board.
- SYSCLK Config: Output 0, 1 or Reference Oscillator Clock.



PHY PANEL

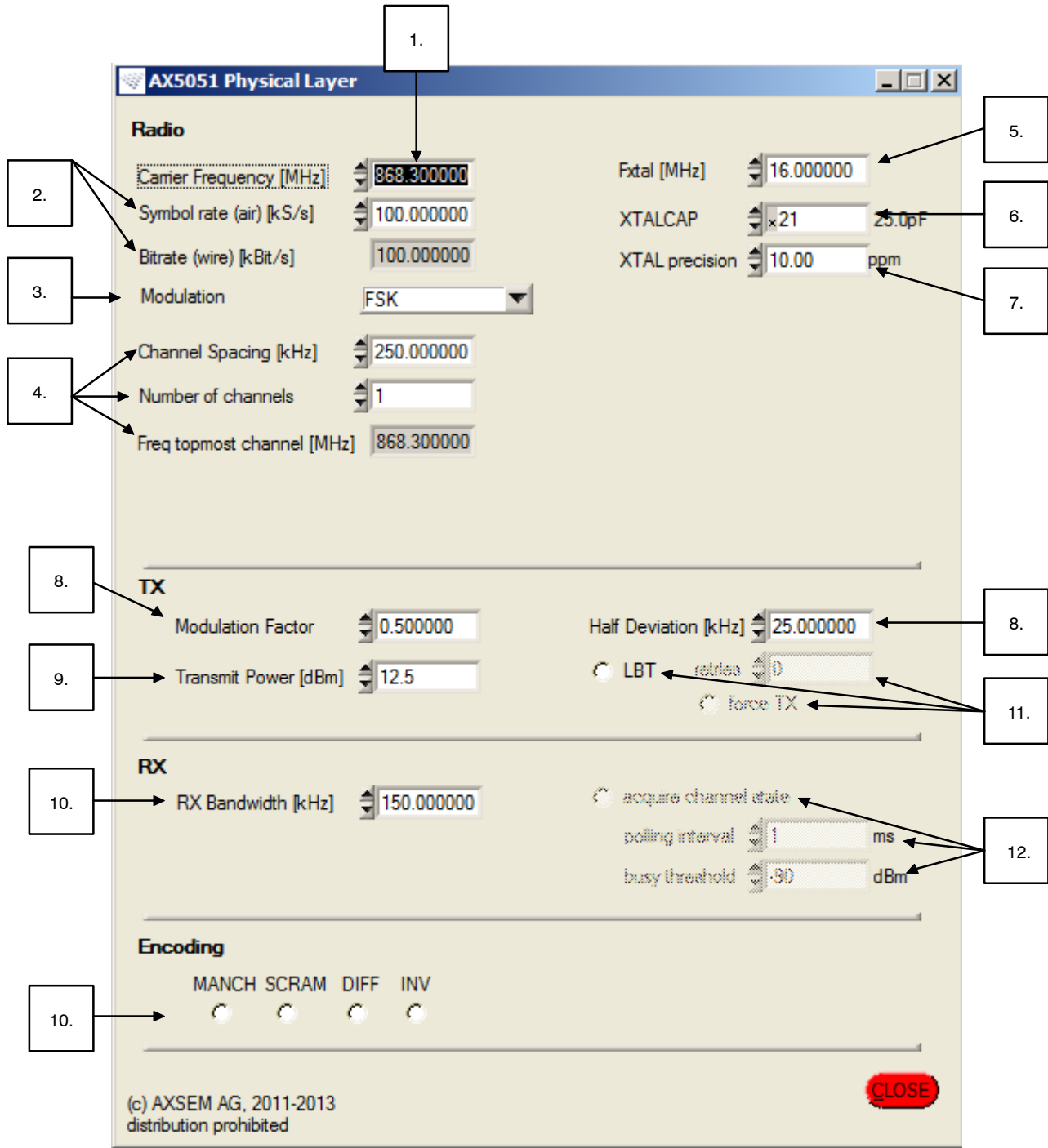


Figure 5. AX5051 Physical Layer

1. Select Carrier Frequency. If more than one channel is configured (see below) the frequency selected here is that of channel number 0.
2. Symbol rate & Bitrate specify the data rate. The values are identical except when using Manchester encoding (bitrate = 0.5 \* symbol rate).
3. Modulation Scheme.

4. Width of your channel. Red values signal a conflict with the TX and/or RX bandwidth. A number of channels can be specified. The firmware determines and stores the VCO range for each channel at startup. The carrier frequency of the topmost channel is displayed for convenience.

5. Frequency of the reference oscillator (XTAL/TCXO).
6. XTALCAP adjusts the on-chip XTAL load capacitance. Set to 0 when using a TCXO.
7. Precision of the reference oscillator (XTAL/TCXO)
8. Modulation factor (often denoted as h).  
FSK: half deviation =  $0.5 * \text{modulation factor} * \text{Symbol rate}$
9. Transmit power in dBm.
10. Bandwidth of the RX channel filter.
11. Listen Before Talk (LBT) option: The TX measures the channel power before sending a packet in order to assess whether the channel is free. If the channel is found to be busy the TX can retry a number of times. If the channel continues to be busy the TX can either give up or transmit anyway (force TX). The interval for re-assessing the channel state as well as the busy threshold can be configured in fields shared with the “acquire channel state” feature (see next point). It is not necessary to enable “acquire channel state” for using LBT.
12. Acquire channel state: In RX continuous mode the RX can periodically measure the channel energy and assess whether the channel is busy. The firmware displays the measured RSSI level on the LCD. A busy channel is indicated by an asterisks. Note that the LCD is updated every 500 ms only even if the measurement is performed more often.
13. Encoding: manchester, scrambler, differential and inversion.  
Note on reference oscillator precision, frequency tracking and bandwidth:
  - Without frequency tracking the worst case frequency deviation between RX and TX is  $\Delta f = 2 * XTAL \text{ precision} * f_{carrier}$ . (The sum of the worst case frequency deviations of TX and RX).
  - The frequency tracker must be allowed to track this deviation.

## FRAMING PANEL

Figure 6. AX5051 Framing

**Framing Mode**

Options are HDLC, Raw Pattern Match and PN9 Compatibility. In Raw Pattern Match mode an arbitrary SYNC WORD of up to 32 bit marks the beginning of a frame. For legacy systems the PN9 Compatibility mode implements PN9 data whitening in software, based on the AX5051's Raw Pattern Match mode.

**Preamble**

- Recommended default settings are chosen according to the selected modulation, encoding and precision of the reference clock.
- Specify the length of the preamble in bits, as well as one byte character which will be repeated length/8 times.
- The preamble character is always sent MSB first, irrespective of the “send MSB first” option in the DATA chunk. See comment on that option below.
- If length is not an integer multiple of 8, the fractional byte is sent last, e.g. for length = 33 bits the preamble character is sent 4 times followed by once the MSB of the preamble character.
- In Wake on Radio mode an additional WOR preamble length (in ms) can be specified. This chunk is transmitted in addition to the number of bits specified above. The length of this chunk should not be shorter than the WOR period.
- Occasionally it may be necessary to conclude the preamble with a few special bits. This can be achieved using the fields below “Append additional encoded bits”.

**SYNC WORD**

- In Raw Preamble Mode you can choose an arbitrary sync word of up to 32 bit as a start of frame delimiter. However it is recommended to use a sync word with well peaking auto correlation function.
- The SYNC WORD is always sent MSB first, irrespective of the “send MSB first” option in the DATA chunk. See comment on that option below.
- DC content, running DC content and the number of transitions are computed in order to indicate the quality of the selected SYNC WORD. Large DC or running DC components can impede correct frequency tracking, whereas a low number of transitions slows down time tracking.
- Selecting the “enable SFD callback” option causes the AXRadioAPI to notify the application level firmware each time a start of frame delimiter (a SYNC WORD or a HDLC flag) has been received.

**LEN & MAC**

- “MAC header length” specifies the length of the packet header, which can contain a length byte, a sequence number for acknowledge and addresses.
- In Raw Pattern Match mode a length byte can be used to communicate the length of the current packet to the RX. Alternatively (uncheck “enable len byte”) the RX can be configured to expect packets of a fixed length.
- “Position” determines the position of the length byte inside the packet header. Zero means the length byte follows immediately after the SYNC word.

- “Significant bits” tells the RX how many bits of the length byte are used to encode the packet length. This is useful if your protocol uses a shared byte to encode the packet length plus something else.
- “len offset”: The actual packet length (excluding SYNC WORD & CRC) assumed by the RX is len + len offset. E.g. set len offset to zero if in your protocol the length byte counts the number of bytes including the length byte. Set len offset = 1 if your length byte counts the number of bytes excluding the length byte itself.
- If you check “enable len byte” in HDLC mode, the TX will send a length byte but it will be ignored by the RX.
- Specifying a maximum packet length can limit the impact of bit errors in the length byte.
- If the acknowledge feature is used an ACK sequence number field can be configured. This is not mandatory for acknowledge to work. However, without this field the SLAVE cannot detect duplicate packets occurring if packet RX has worked but the ACK packet got lost, causing the MASTER to retransmit the packet.
- Specify a device addresses of up to 4 bytes and a corresponding address mask in order to restrict reception to packets containing a matching address.
- “address position” specifies the starting position of the address inside the packet. Zero corresponds to the byte immediately after the SYNC WORD/HDLC flag.
- Optionally the sender’s address can be added to the header. Note that address works on the destination address field (containing the slave’s address in normal MASTER to SLAVE communication and the master’s address for packets sent from SLAVE to MASTER, e.g. ACK packets in the demo firmware.)

### DATA

- The DATA field allows you to specify a static sample payload which will be used by the demo firmware. Enter each byte as a two digit hex number (without leading 0x). Bytes may be separated by spaces.
- Optionally the firmware can insert a 16 bit counter into the data field at a specified position. This allows the RX to count the number of lost packets. The two bytes reserved for this counter are displayed as “00 00” in the data field.
- Checking the “send MSB first” button causes each byte in the DATA field to be sent MSB first. Otherwise each byte is sent LSB first. Note, however, that preamble character and SYNC WORD are always sent MSB first. (Technically, the RF chip will be configured to operate in MSB first or LSB first mode according to the “MSB first button”, which applies to every byte. However, in LSB first mode the AX–RadioLAB GUI reverses the entered preamble and SYNC WORD bytes in order to keep the representation in the GUI MSB first. The reason for this is, that the byte and nibble order of a SYNC WORD written left to right but read LSB first can be confusing.)
- “CRC SKIP FIRST” causes the first byte of the packet to be excluded from CRC calculation.

### CRC

- Select CRC algorithm and initial value.
- The CRC is automatically calculated by the RF chip or the MCU firmware and appended to the packet.
- Packets with incorrect CRC are silently dropped by the MCU firmware.

## BASIC AND REGULATORY TESTS PANEL

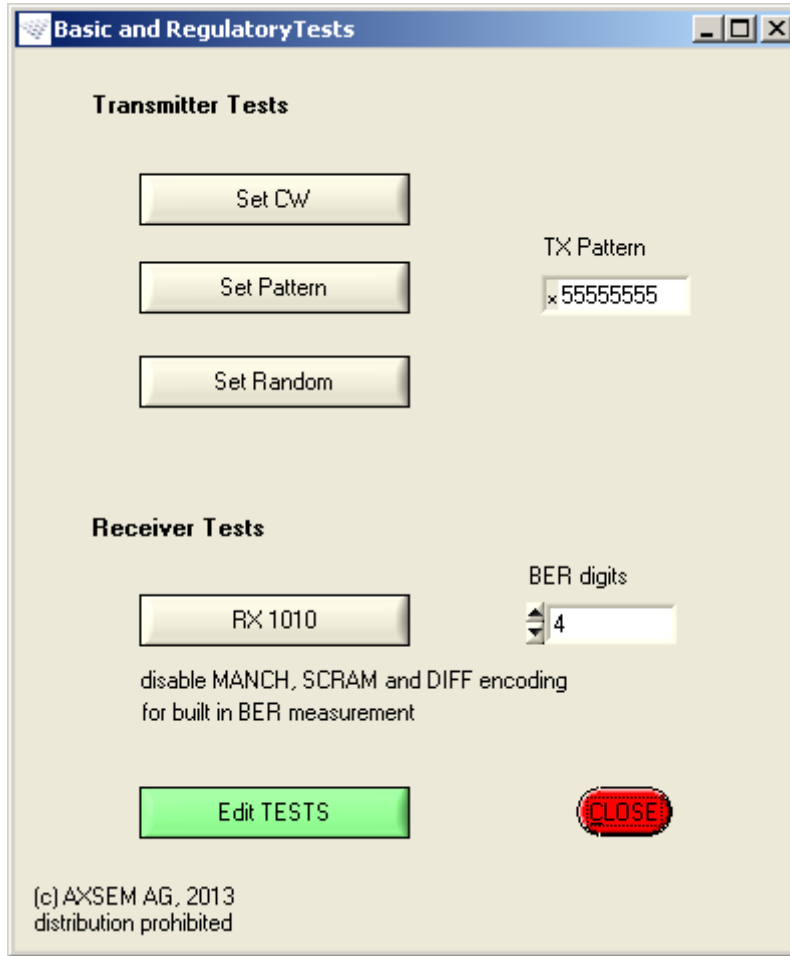


Figure 7. Basic and Regulatory Tests

- “Set CW” puts the RF chip to TX CW mode.
- “Set Pattern” transmits the specified 8 byte pattern forever. The byte specified by the two leftmost digits is sent first. Each byte is sent LSB first unless the “send MSB first” button in the framing panel is checked. Manchester, scrambler and differential encoder are bypassed (inversion encoding applies if selected!).
- “Set Random” transmits random data forever. (This is realized using the scrambler.)
- “RX 1010” performs measurement of bit error rate (BER). Typically connect the RX to a signal generator sending an infinite 1010 stream, i.e. in FSK mode the generator should toggle between  $\{+1, -1\}$  \* half deviation.
- Edit TESTS opens the firmware implementing the above test modes in the AXCode::Blocks IDE.

COMMENTS ON C-CODE FIRMWARE

**File Structure**

The MASTER and SLAVE and TESTS firmware projects implement the corresponding modes using the AXRADIO V2 API as a “driver” for the AX5051 RF chip.

**Table 2. THE AXRADIO V2 API CONSISTS OFF THE FOLLOWING FILES**

File	Description
COMMON/axradio.h	AXRadio API declaration
COMMON/easyax5051.h	AXRadio AX5051 private header
COMMON/easyax5051.h	AXRadio AX5051 main code
AX_Radio_Lab_output/config.c	Parameters generated by AX-RadioLAB

It is documented in DOCU/AXRadioV2API.pdf

**Table 3. THE APPLICATION LEVEL CODE CONSISTS OF THE FOLLOWING FILES**

File	Description
MASTER/main.c	MASTER application level main code
SLAVE/main.c	SLAVE application level main code
TESTS/main.c	TESTS application level main code
COMMON/misc.[ch]	Application level helper files
SLAVE/display.c	Application level helper file
SLAVE/lposc.c	helper file
AX_Radio_Lab_output/configslave.[ch]	Application level parameters generated by AX-RadioLAB
AX_Radio_Lab_output/configmaster.[ch]	Application level parameters generated by AX-RadioLAB

**Define Statements for Tweaking the Firmware**

`#define MCU_SLEEP` (MASTER/main.c, SLAVE/main.c) commenting this out puts the micro controller into standby, rather than sleep mode when idle.

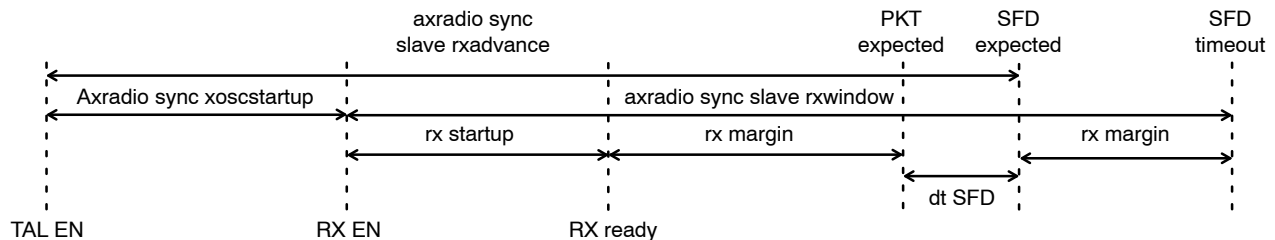
`#define USE_DBGLINK` (MASTER/main.c, SLAVE/main.c) use the debugger link to output status information and received packets if the main-board is connected to the debugger.

**SYNC Packet RX Timing**


In the AXRadio SYNC RX mode timing is controlled by the parameters `axradio_sync_xoscstartup`, `axradio_sync_slave_rxadvance`, `axradio_sync_slave_rxwindow` and `axradio_sync_slave_rxtimeout`. All times are measured in periods of `libmf_wtimer0`, which is clocked by the low power crystal oscillator. As shown at the top of *Figure 1* the radio XTAL oscillator of the SLAVE is enabled `axradio_sync_slave_rxadvance` before the frame delimiter (SFD) of the next packet is expected. After settling the XTAL for `axradio_sync_xoscstartup` the RX is enabled. The RX then settles and looks for an SFD, time outing after `axradio_sync_slave_rxwindow` if no SFD is received. If an SFD is received the RX stays on until successfully receiving a packet or until a timeout occurs after `axradio_sync_slave_rxtimeout`. (Not shown in the figure.)

AX-RadioLAB computes `axradio_sync_slave_rxadvance` and `axradio_sync_slave_rxwindow` based on the `rx_margin` which can be configured in the SYNC Timing panel as shown at the bottom of *Figure 1*. The goal is to have the RX ready `rx_margin` before the expected start of the packet. (Ready means that the synthesizer and the analog baseband are settled. AGC settling and bit synchronization can be done only if preamble signal is present.) The SFD timeout is set `rx_margin` after the expected SFD. Three instances of `axradio_sync_slave_rxadvance` and `axradio_sync_slave_rxwindow` are computed. The first instance contains the increased `rx_margin` to be used for the first packet after (re-)synchronization when the effective packet period measurement is available yet. The second instance contains the `rx_margin` used in normal operation (“Minimum RX margin” in the SYNC Timing panel). The third instance has `rx_margin` scaled by a factor of 4 and is used after a failed packet reception.

NOTE: A processing delay occurs in the RX, i.e. the detection of the SFD is not signalled by the RX the very moment it is on air. On the other the RX should be enabled `rx_margin` before the packet is on air. Therefore `dt_SFD` is not simply the length of preamble plus SFD but also contains the processing delay of roughly 19 bit.



**Figure 8. SYNC Firmware Packet Reception Timing**

ON Semiconductor and  are trademarks of Semiconductor Components Industries, LLC dba ON Semiconductor or its subsidiaries in the United States and/or other countries. ON Semiconductor owns the rights to a number of patents, trademarks, copyrights, trade secrets, and other intellectual property. A listing of ON Semiconductor's product/patent coverage may be accessed at [www.onsemi.com/site/pdf/Patent-Marking.pdf](http://www.onsemi.com/site/pdf/Patent-Marking.pdf). ON Semiconductor reserves the right to make changes without further notice to any products herein. ON Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does ON Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation special, consequential or incidental damages. Buyer is responsible for its products and applications using ON Semiconductor products, including compliance with all laws, regulations and safety requirements or standards, regardless of any support or applications information provided by ON Semiconductor. "Typical" parameters which may be provided in ON Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. ON Semiconductor does not convey any license under its patent rights nor the rights of others. ON Semiconductor products are not designed, intended, or authorized for use as a critical component in life support systems or any FDA Class 3 medical devices or medical devices with a same or similar classification in a foreign jurisdiction or any devices intended for implantation in the human body. Should Buyer purchase or use ON Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold ON Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that ON Semiconductor was negligent regarding the design or manufacture of the part. ON Semiconductor is an Equal Opportunity/Affirmative Action Employer. This literature is subject to all applicable copyright laws and is not for resale in any manner.

**PUBLICATION ORDERING INFORMATION**

**LITERATURE FULFILLMENT:**

Literature Distribution Center for ON Semiconductor  
19521 E. 32nd Pkwy, Aurora, Colorado 80011 USA  
**Phone:** 303-675-2175 or 800-344-3860 Toll Free USA/Canada  
**Fax:** 303-675-2176 or 800-344-3867 Toll Free USA/Canada  
**Email:** [orderlit@onsemi.com](mailto:orderlit@onsemi.com)

**N. American Technical Support:** 800-282-9855 Toll Free  
USA/Canada  
**Europe, Middle East and Africa Technical Support:**  
Phone: 421 33 790 2910  
**Japan Customer Focus Center**  
Phone: 81-3-5817-1050

**ON Semiconductor Website:** [www.onsemi.com](http://www.onsemi.com)  
**Order Literature:** <http://www.onsemi.com/orderlit>  
For additional information, please contact your local  
Sales Representative