MAX32664 User Guide

UG6806; Rev 3; 8/20

Abstract

The MAX32664 user guide provides flow charts, timing diagrams, GPIOs/pin usage, I²C interface protocol, and annotated I²C traces between the host microcontroller and the MAX32664. Typical application uses the MAX32664 as a low-power microcontroller in a sensor hub configuration to provide processed data such as heart rate and SpO_2.

Table of Contents

List of Figures

List of Tables

Introduction

The MAX32664 is a pre-programmed microcontroller with firmware drivers and algorithms. Combined with the appropriate sensor devices, the MAX32664 acts as a sensor hub to provide processed data to a host device. This solution seamlessly enables customers to receive raw and/or calculated data from Maxim optical sensor solutions, while keeping overall system power consumption in check. The tiny form factor (1.6mm x 1.6mm 16-bump WLP) allows for integration into extremely small applications. The MAX32664 is integrated into Maxim's complete reference design solutions, which shortens the time to market.

The MAX32664 is the same hardware as the MAX32660 but with a pre-programmed bootloader that accepts in-application programming (IAP) of Maxim supplied algorithms and sensor drivers. The MAX32664 provides a fast-mode, I²C slave interface to a microcontroller host. A second I²C interface is dedicated to communicating with sensors.

The MAXM86146 is an additional sensor hub product that integrates two photodiodes, the MAX32664C sensor hub, and the bio-sensing analog front end (AFE) into one compact IC package.

For further details on memory, register mapping, system clocks, reset, power management, GPIOs/alternate functions, DMA controller, UART, RTC, timers, WDT, I²C, and SPI, see the MAX32660 User Guide.

For ordering information, mechanical and electrical characteristics, and the pinout for the MAX32664 family of devices, refer to the MAX32664 data sheet.

For information on the Arm® Cortex®-M4 with FPU core, refer to the Cortex-M4 with FPU Technical Reference Manual.

MAX32664 Variants

The MAX32664 is pre-programmed with bootloader software that accepts in-application programming of Maxim application code which consists of algorithms and the associated sensor driver. The MAX32664 is used as a sensor hub controller.

The algorithm/application code provides processed and/or raw data through the I^2C interface. Several variants of the MAX32664 exist based on the target application. These variants come pre-programmed with a bootloader that only accepts the matching encryption keys for the part (e.g., the MAX32664A bootloader is pre-programmed with the A encryption key, reference designs are programmed with Z keying, etc.). Designers should use the table below in order to select the correctly keyed part.

Table 1. MAX32664 Variants, Matching Algorithms, and Reference Designs

MAX32664 VARIANT	APPLICATION ALGORITHM/FIRMWARE	BOOTLOADER KEY	MAXIM REFERENCE DESIGN
MAX32664A	**MaximFast:** Maxim Integrated finger-based heart-rate and SpO_2 monitoring algorithm (100Hz sampling). The MaximFast algorithm is compatible with the sensor hub combination of the MAX32664A, MAX30101 AFE, and KX-122 accelerometer. It is recommended, but not mandatory, to use an accelerometer with the MaximFast algorithm. Do not enable the accelerometer if there is no accelerometer in your design. If the KX-122 accelerometer is not installed in the design and external accelerometer data is supplied, then the accelerometer should use the 100Hz sampling rate. **Automatic gain control (AGC)**: If the AGC is enabled, the LED currents and pulse width are automatically determined by the algorithm. If the AGC is not enabled, the LED currents and pulse width registers should be configured by the host software.	A	MAXREFDES220#

MAX32664 VARIANT	APPLICATION ALGORITHM/FIRMWARE	BOOTLOADER KEY	MAXIM REFERENCE DESIGN
MAX32664B	**Wearable heart-rate monitoring (WHRM) algorithm**: The WHRM algorithm is configured to use LED1 and photodiode (PD) 1, and it is compatible with the sensor hub combination of the MAX32664B, MAX86141 AFE, and KX-122 accelerometer. Using the KX-122 accelerometer or external accelerometer data with the WHRM algorithm is required to compensate motion artifacts. If the KX-122 accelerometer is not connected to the MAX32664, then the external accelerometer data should be supplied at the 25Hz sampling rate. The WHRM algorithm includes automatic exposure control (AEC) and skin control detection (SCD. If AEC is enabled, the LED current, pulse width, and sample rate are automatically determined by the algorithm. If AEC is disabled, the LED current, LED current range, pulse width, and ADC range registers are set to default and can be updated by the host software.	B	MAXREFDES101#

MAX32664 VARIANT	APPLICATION ALGORITHM/FIRMWARE	BOOTLOADER KEY	MAXIM REFERENCE DESIGN
MAX32664C MAX32664C MAXM86146	**Wearable heart-rate monitoring and wearable oxygen saturation (WHRM+WSpO$_2$)** algorithm version 3x.xx.x, where xx is 3 or greater: The wearable algorithm suite can monitor heart rate and SpO$_2$ simultaneously. It is configured to use LED1 (green), LED2 (IR), LED3 (red), and photodiode 1 and 2, and it is compatible with the sensor hub combination of the MAX32664C, MAX86141 (or MAXM86161/ MAXM86146) AFE, and KX-122 accelerometer. Using the KX-122 accelerometer or external accelerometer data with the WHRM algorithm is required to compensate for motion artifacts. If the KX-122 accelerometer is not connected to the MAX32664, then the external accelerometer data should be supplied at the 25Hz sampling rate. The wearable algorithm suite includes automatic exposure control (AEC) and skin control detection (SCD). If AEC is enabled, the LED currents, pulse width, and sample rate are automatically determined by the algorithm. The AEC algorithm adjusts averaging and sample rates for an effective rate of 25Hz. If AEC is not enabled, the rates are set to default and can be updated by the host software. Low power mode is enabled in the firmware. Normally, when the MAX32664 is idle, it switches to the Deep Sleep state to save power. An external interrupt like a sensor, the host MFIO, or RTC alarm forces the MAX32664 to wake up.	C C C	MAXREFDES103# MAXM86161EVSYS# MAXM86146EVSYS#

MAX32664 VARIANT	APPLICATION ALGORITHM/FIRMWARE	BOOTLOADER KEY	MAXIM REFERENCE DESIGN
MAX32664D	**Finger-based blood pressure trending (BPT), heart-rate, and SpO_2 monitoring algorithm** (100Hz Sampling). The algorithm is compatible with the sensor hub combination of the MAX32664D and MAX30101 AFE. No accelerometer is required for this algorithm. The BPT algorithm includes automatic gain control to adjust the LED currents. Prior to running the algorithm, a calibration procedure is required to determine blood pressure and SpO_2 calibration coefficients. **Automatic gain control (AGC).** If the AGC is enabled, the LED currents and pulse width are automatically determined by the algorithm. If the AGC is not enabled, the LED currents and pulse width registers should be configured by the host software.	D	MAXREFDES220#
MAX32664Z	The algorithms listed in this table have a corresponding algorithm/application Z-keyed .msbl file, which can be flashed to the MAX32664Z using in-application programming on the MAX32664Z.		MAXREFDES220# MAXREFDES101# MAXREFDES103#

For all the MAX32664 parts, the latest algorithm (.msbl file) with the corresponding bootloader key must be downloaded, and these parts must be programmed using the in-application programming feature of the bootloader.

Reference Designs with the MAX32664

Maxim provides multiple reference designs to its customers to enable quick and effective adoption of the MAX32664 and fastest time to market. For detailed schematics, refer to the user guide of each reference design.

MAXREFDES220#

The MAXREFDES220# reference design provides everything you need to quickly prototype your product to measure finger-based heart rate and blood oxygen saturation level (SpO_2).

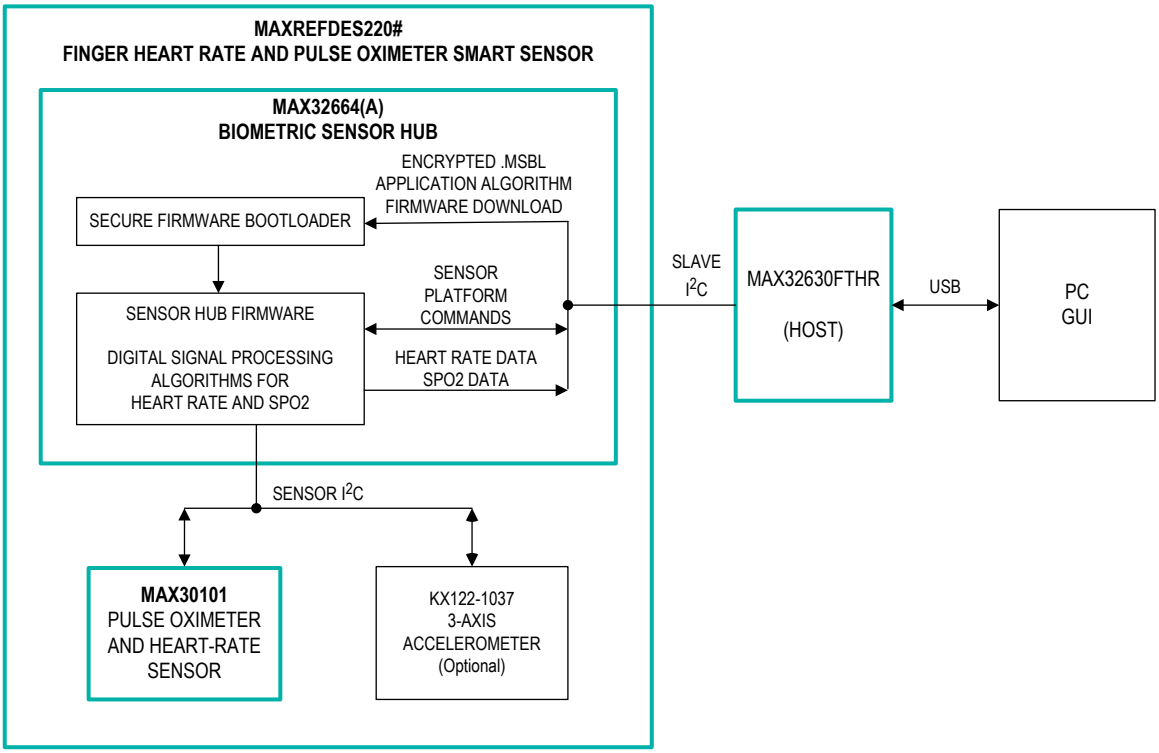


Figure 1. MAXREFDES220# block diagram.

The MAXREFDES220# solution, which includes the MAX30101 and the MAX32664, provides an integrated hardware and software solution for finger-based applications. The MAX32664 is used as a sensor hub to collect data from the MAX30101 analog front end (AFE). The reference design also includes a tri-axis accelerometer (KX-122) to detect motion artifacts. (Accelerometer support in the MAXREFDES220# is optional.)

The MAX32630FTHR is used as a sample host is included in MAXREFDES220# reference design.

MAXREFDES101#

The MAXREFDES101# is a unique evaluation and development platform in a wrist-worn wearable form factor that demonstrates the functions of a wide range of Maxim's products for health-sensing applications.

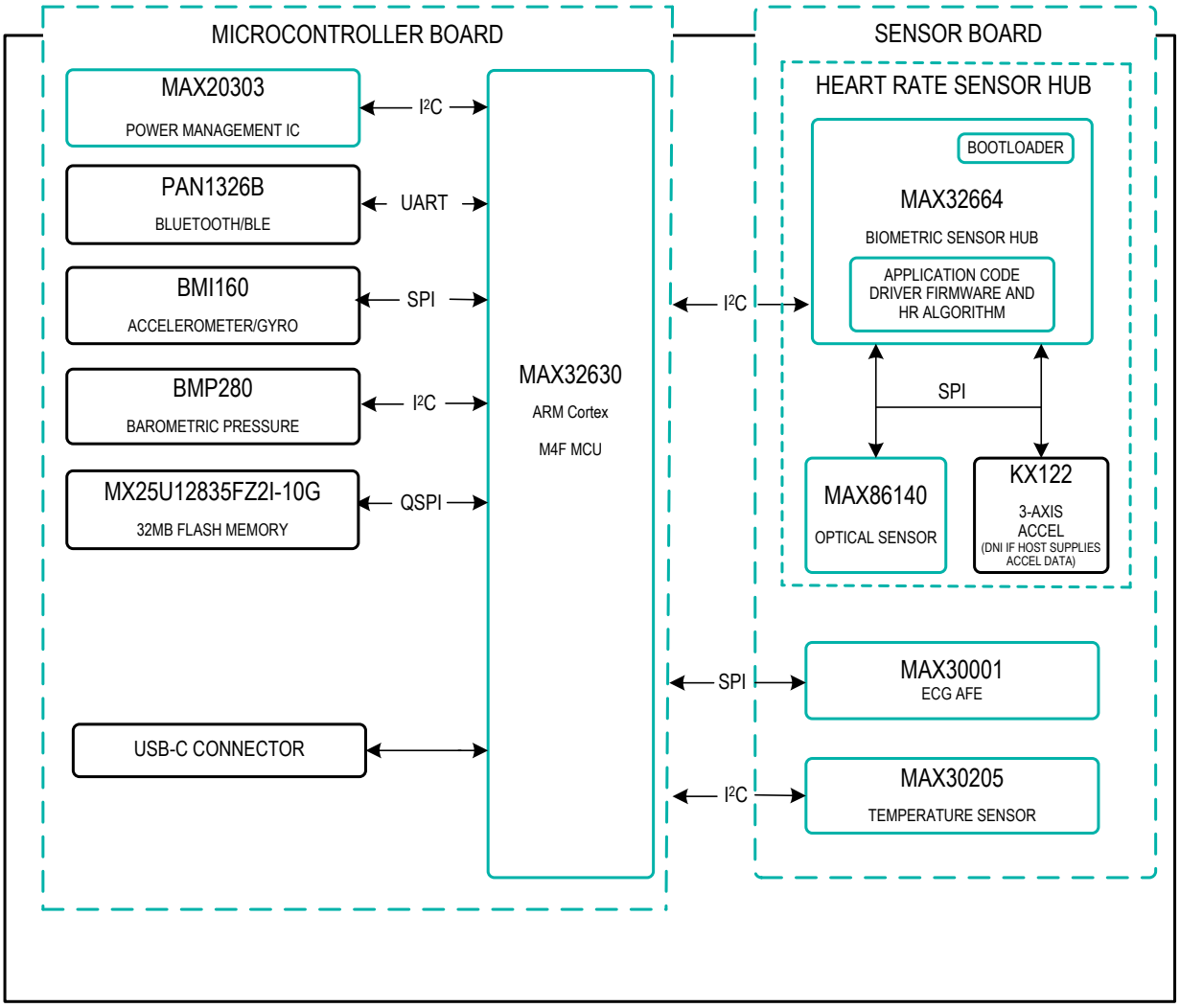


Figure 2. MAXREFDES101# block diagram.

This second-generation health sensor platform (a follow-on to the MAXREFDES100#) integrates a PPG AFE sensor (MAX86141), a biopotential AFE (MAX30001), a human body temperature sensor (MAX30205), a microcontroller (MAX32630), a power-management IC (MAX20303), and a 6-axis accelerometer/gyroscope. The complete platform includes a watch enclosure and a biometric sensor hub with an embedded application code for heart-rate algorithm and AFE drivers (MAX32664). Algorithm output and sensor data can be streamed through Bluetooth® to an Android® application or PC GUI for demonstration, evaluation, and customized development.

MAXREFDES103#

The MAXREFDES103# is a wrist-worn wearable form factor that demonstrates the high-sensitivity and algorithm processing functions for health-sensing applications. This health sensor band platform includes an enclosure and a biometric sensor hub with an embedded algorithm for heart rate and SpO_2 (MAX32664C) which processes PPG signals from the analog-front-end (AFE) sensor (MAX86141). Algorithm output and raw data can be streamed through Bluetooth® to an Android® app or PC GUI for demonstration, evaluation, and customized development.

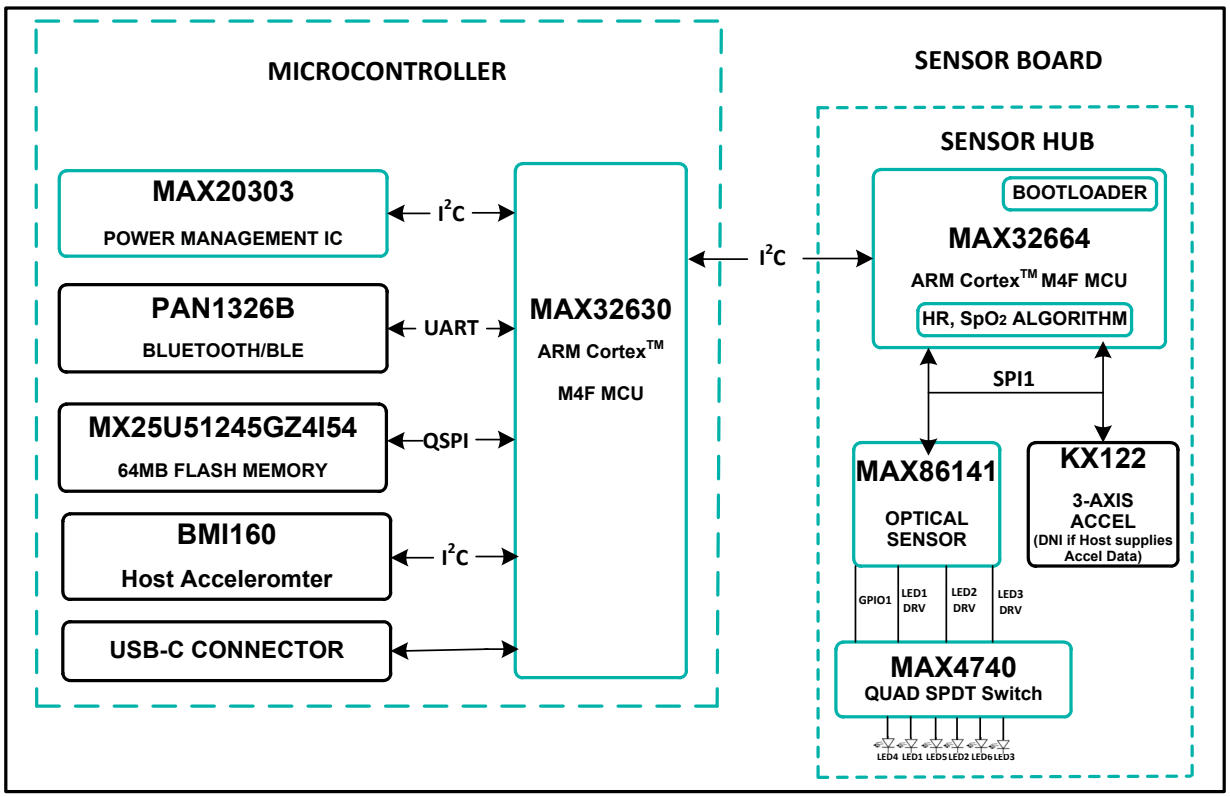


Figure 3. MAXREFDES103# block diagram.

Additional Sensor Hub Products

MAXM86161 Integrated Optical Module for In-Ear HR and SpO_2 Measurement

The MAXM86161 is an ultra-low-power, completely integrated, optical data-acquisition system ideally suited for in-ear products. On the transmitter side, the MAXM86161 has three programmable high-current LED drivers. On the receiver side, MAXM86161 consists of a high-efficiency PIN photo-diode and an optical readout channel. The optical readout has a low-noise signal conditioning analog front-end (AFE), including 19-bit ADC, an industry-lead ambient light cancellation (ALC) circuit, and a picket fence detect-and-replace algorithm. Due to the low power consumption, compact size, easy, flexible-to-use, and industry-lead ambient light rejection capability of the MAXM86161, the device is ideal for a wide variety of optical sensing applications such as heart-rate detection and pulse oximetry.

MAXM86161 MAX32664C application .msbl are versioned as 32.x.x. If using the MAXM86161EVSYS#, the MAX32664 must be flashed with the compatible .msbl application file that matches the Nordic binary.

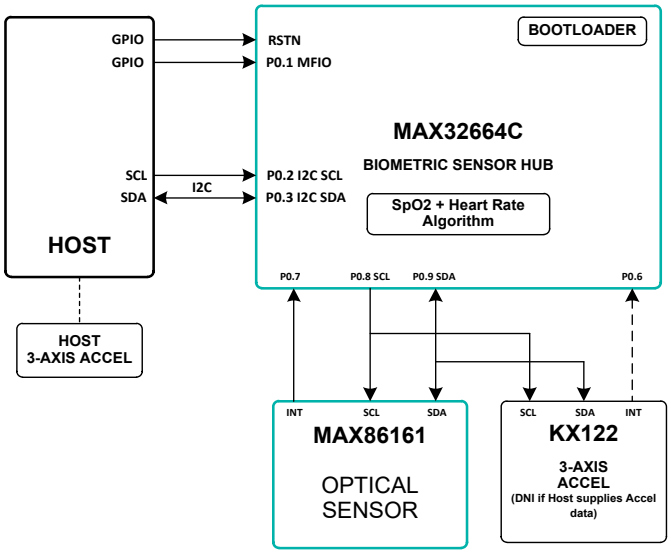


Figure 4. Block diagram for Host, MAX32664C Sensor Hub, and MAXM86161.

MAXM86146 Integrated Sensor Hub with AFE and Two Integrated Photodiodes for Wrist-Based HR and SpO_2 Measurements

The MAXM86146 is an ultra-low-power, completely integrated, optical data acquisition system specifically designed for battery-powered devices and wireless sensors. It combines Maxim's best in class optical bio-sensing analog front end (AFE) with the powerful Arm Cortex-M4 sensor hub microcontroller and two high sensitivity photo diodes, all in a compact 4.5mm x 4.1mm x 0.88mm, 38-pin OLGA package with commercial operating temperature range of 0ºC to +70ºC. The AFE has two, low-noise, optical readout channels. Both channels have independent 19-bit ADCs, industry leading ambient-light cancellation (ALC) circuit, and a picket fence detect-and-replace algorithm. The AFE includes three programmable high-current LED drivers and operates on a 1.8V main supply voltage and a 3.1V-5.5V LED driver supply voltage.

The sensor hub MCU within the MAXM86146 is factory programmed with the sensor hub bootloader; the application algorithm .msbl is not included in the factory programming. The latest

MAX32664C MAXM86146 application algorithm .msbl from the MAX32664 website must be flashed to the MAXM86146 MCU.

MAXM86146 MAX32664C application .msbl are versioned as 33.x.x. If using the MAXM86146EVSYS#, the MAX32664 must be flashed with the compatible .msbl application file that matches the Nordic binary.

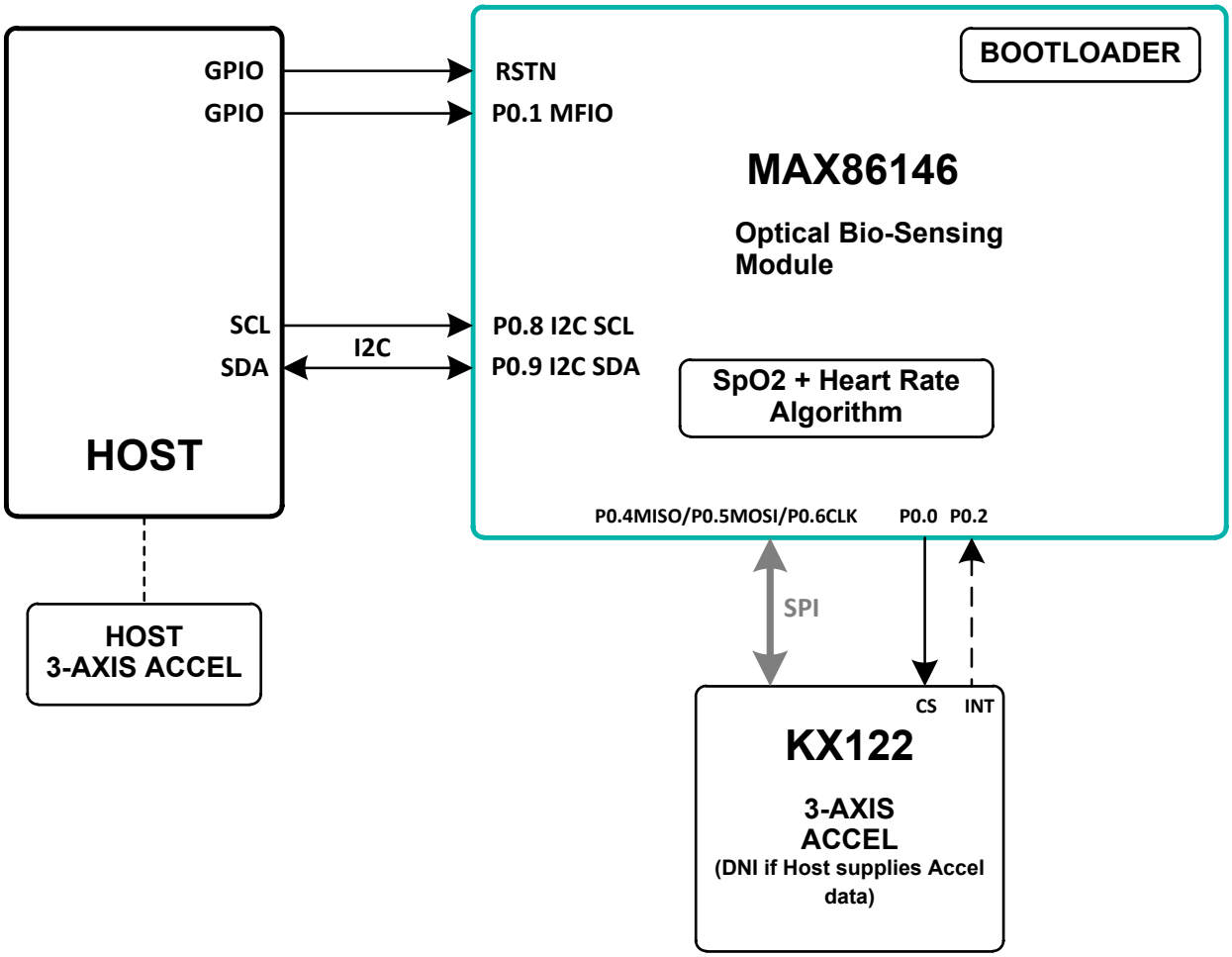


Figure 5. Block diagram for Host, MAXM86146 (Sensor Hub with Integrated AFE and Photodiodes).

MAX32664 GPIOs and RSTN Pin

To control and communicate with the MAX32664, the RSTN pin and GPIOs P0.1, P0.2, P0.3 of the MAX32664 are connected to the host as pictured in Figure 6.

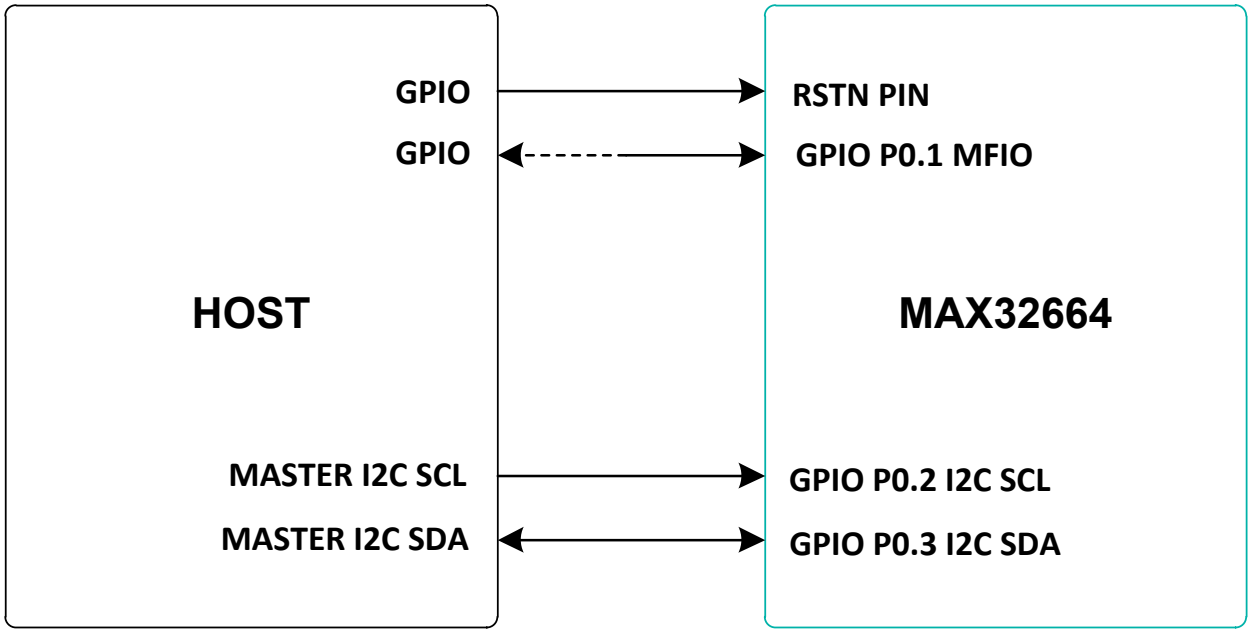


Figure 6. Pin connections between the host and the MAX32664.

The RSTN pin is used in conjunction with the GPIO P0.1 MFIO pin to control whether the MAX32664 starts up in Application mode or Bootloader mode. While in application mode, the MFIO pin is configured to provide an interrupt signal to the host, or the host can use it to wake the MAX32664 when using the low-powered firmware.

The host acts an I2C master to communicate with the MAX32664. GPIO P0.2 is used as the SCL line and GPIO P0.3 is used as the SDA line.

Table 2. RSTN Pin and GPIOs Pins

MAX32664	DESCRIPTION	DIRECTION FROM THE MAX32664 SIDE
Pin RSTN	Reset_N	Input
GPIO P0.1	GPIO MFIO interrupt to host, wake from host, bootloader/application on power up. Interrupt to host is not used on the following: • MAX32664B WHRM v20.2.0+ • MAX32664C WHRM+WSpO$_2$ v30.2.4+, v32.1.2+, v33.6.0+	Input/Output Input only for the following: • MAX32664B WHRM v20.2.0+ • MAX32664C WHRM+WSpO$_2$ v30.2.4+, v32.1.2+, v33.6.0+
GPIO P0.2	I2C0 Host SCL	Input
GPIO P0.3	I2C0 Host SDA	Input/Output

To achieve a lower power profile, the following versions of the .msbl algorithm use a polling method instead of the MFIO pin as an interrupt to the host:

- MAX32664B WHRM v20.2.0+
- MAX32664C WHRM WHRM+WSpO$_2$ v30.2.4+, v32.1.2+, v33.6.0+

For these versions of the algorithm, the MAX32664B/C switches to “Deep Sleep” state to save power. The MAX32664B/C can be woken from deep sleep by the internal RTC, the connected sensor, or the MFIO pin. The host is required to wake up the MAX32664B/C prior to any I^{2}C communication by performing the following:

- Setting MFIO to low at least 250µsec before the beginning of an I^{2}C communication to make sure the MAX32664B/C is awake
- Keeping MFIO low until the end of the I^{2}C communication to make sure the MAX32664B/C does not switch to “Deep Sleep” state
- Setting MFIO to high after the end of I^{2}C communication to allow the MAX32664B/C to switch back to “Deep Sleep” state

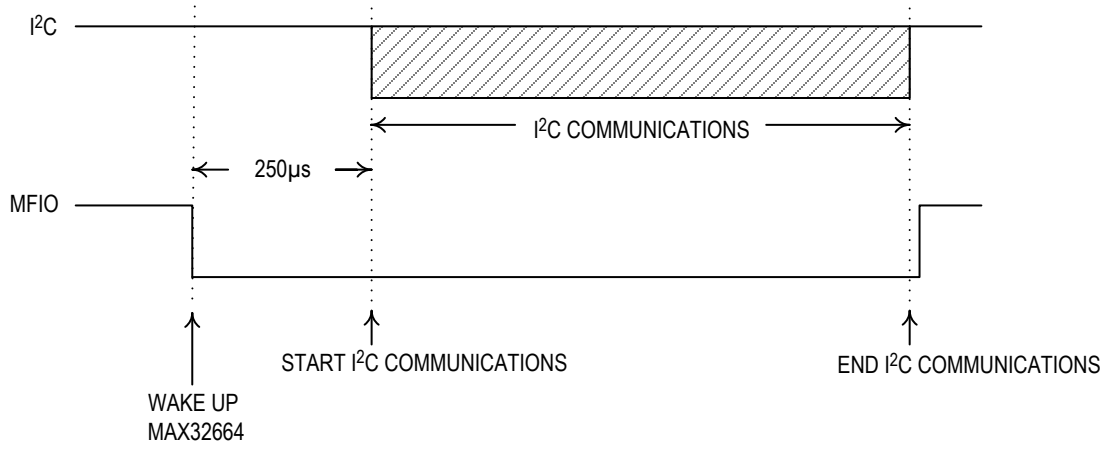


Figure 7. Host sets MFIO low to wake up the low-powered versions of the MAX32664.

For the WHRM (v20.2.0+) and the WHRM+WSpO$_2$ (v30.2.4+, v32.1.2+, v33.6.0+), the host is required to regularly poll the MAX32664B/C to read the measurement data. The host is required to regularly empty the measurement data in the MAX32664B/C FIFO at a periodic rate. The periodic rate depends on the rate that the MAX32664B/C samples report is generated. By reducing the samples report period, the FIFO does not need to be emptied as often.

The host can read samples in the output FIFO at a period (host reading FIFO period) five times the length of the samples report period to avoid FIFO overflow. In this example, an average of five samples is in the output FIFO.

By default, the samples report period (read samples report period, 0x11 0x02) is set to 40ms. In this case, it is recommended that the host read samples from the output FIFO every 200ms (host reading period). At these rates, on average there will be five samples in the output FIFO for the host to read.

Variations of the MAX32664 use additional GPIO pins in order to communicate and control sensor devices. For example, in the MAXREFDES220#, the additional GPIOs listed in Table 3 are used to control the sensors used.

Table 3. Additional MAX32664 GPIOs for the MAXREFDES220#

MAX32664	DESCRIPTION	DIRECTION FROM THE MAX32664 SIDE
GPIO P0.6	KX122 ACCEL Interrupt	Input
GPIO P0.7	MAX30101 Interrupt	Input
GPIO P0.8	MAX30101, KX122 I2C1_SCL	Output
GPIO P0.9	MAX30101, KX122 I2C1_SDA	Input/Output

Table 4. Additional MAX32664 GPIOs or the MAXREFDES101#

MAX32664	DESCRIPTION	DIRECTION FROM THE MAX32664 SIDE
GPIO P0.0	KX122 ACCEL Select	Output
GPIO P0.4	SPI MISO: MAX86141, KX122	Input
GPIO P0.5	SPI MOSI: MAX86141, KX122	Output
GPIO P0.6	SPI CLK: MAX86141, KX122	Output
GPIO P0.7	MAX86141 Select	Output
GPIO P0.8	MAX86141 Interrupt	Input
GPIO P0.9	KX122 Interrupt (N/A for polling versions 30.2.3+ for the MAX32664C and v20.2.x+ for the MAX32664B)	Input

MAX32664 Bootup and Application Mode

The MAX32664 is programmed to enter either bootloader mode or application mode at the start-up based on the state of the MFIO pin.

Variations of the MAX32664 part are pre-programmed with the different algorithms and application firmware. Table 11 details the applications firmware that are pre-programmed. It is strongly recommended that the application firmware be updated to the latest version.

MAX32664 Bootloader Mode

The MAX32664 enters bootloader mode based on the sequencing of the RSTN pin and the MFIO pin. The necessary sequence is as follows:

- Set the RSTN pin low for 10ms.
- While RSTN is low, set the MFIO pin to low (MFIO pin should be set low at least 1ms before RSTN pin is set high.)
- After the 10ms has elapsed, set the RSTN pin high.
- After an additional 50ms has elapsed, the MAX32664 is in bootloader mode.
- If the enter bootloader mode command, 0x01 0x00 0x08, is not received within the first approximately 780ms and there is a valid .msbl application that has been flashed to the MAX32664, then the mode changes to the application mode automatically.

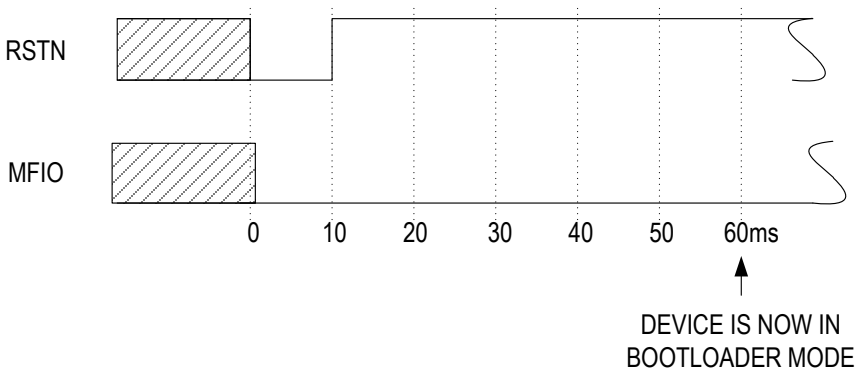


Figure 8. Entering bootloader mode using the RSTN pin and the MFIO GPIO pin.

MAX32664 Application Mode

The MAX32664 enters application mode based on the sequencing of the RSTN pin and the MFIO pin. The necessary sequence is as follows:

- Set the RSTN pin low for 10ms.
- While RSTN is low, set the MFIO pin to high.
- After the 10ms has elapsed, set the RSTN pin high. (MFIO pin should be set high at least 1ms before RSTN pin is set high.)
- After an additional 50ms has elapsed, the MAX32664 is in application mode and the application performs its initialization of the application software.
- Approximately 1.5 second after the RSTN is set to high, the application completes the initialization and the device is ready to accept I²C commands. (For MAX32664A and MAX32664D, the startup time is 1.0 second).

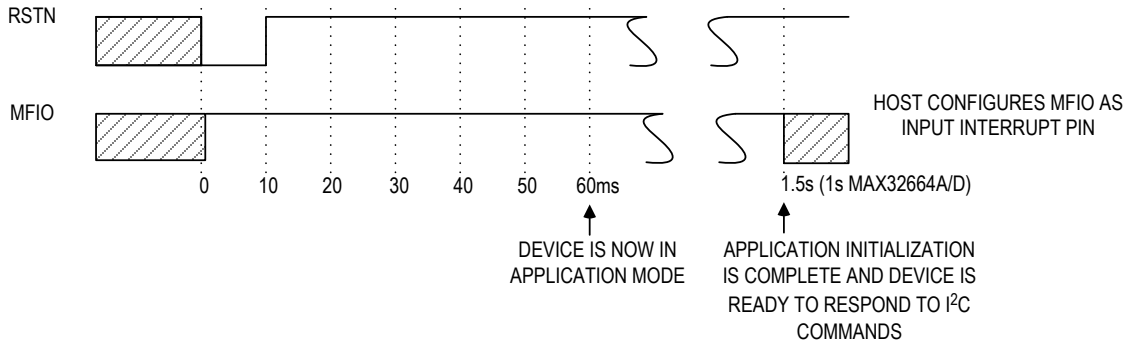


Figure 9. Entering application mode using the RSTN pin and MFIO pin.

Communications to the MAX32664 over I2C

The host communicates to the MAX32664 through the I2C bus. The MAX32664 uses 0xAA as the I2C 8-bit slave write address and 0xAB is used as the I2C 8-bit slave read address. The maximum I2C data rate supported is 3400Kbps.

Bit Transfer Process

The defined bit transfer process is described below. It is recommended that I2C GPIO 'bit-bang' software be implemented on the host if the host MCU I2C hardware/HAL is not compatible with sensor hub protocol.

Both SDA and SCL signals are open-drain circuits. Each has an external pullup resistor that ensures each circuit is high when idle. The I2C specification states that during data transfer, the SDA line can change state only when SCL is low, and that SDA is stable and able to be read when SCL is high. Typical I2C write/read transactions are shown in Figure 10.

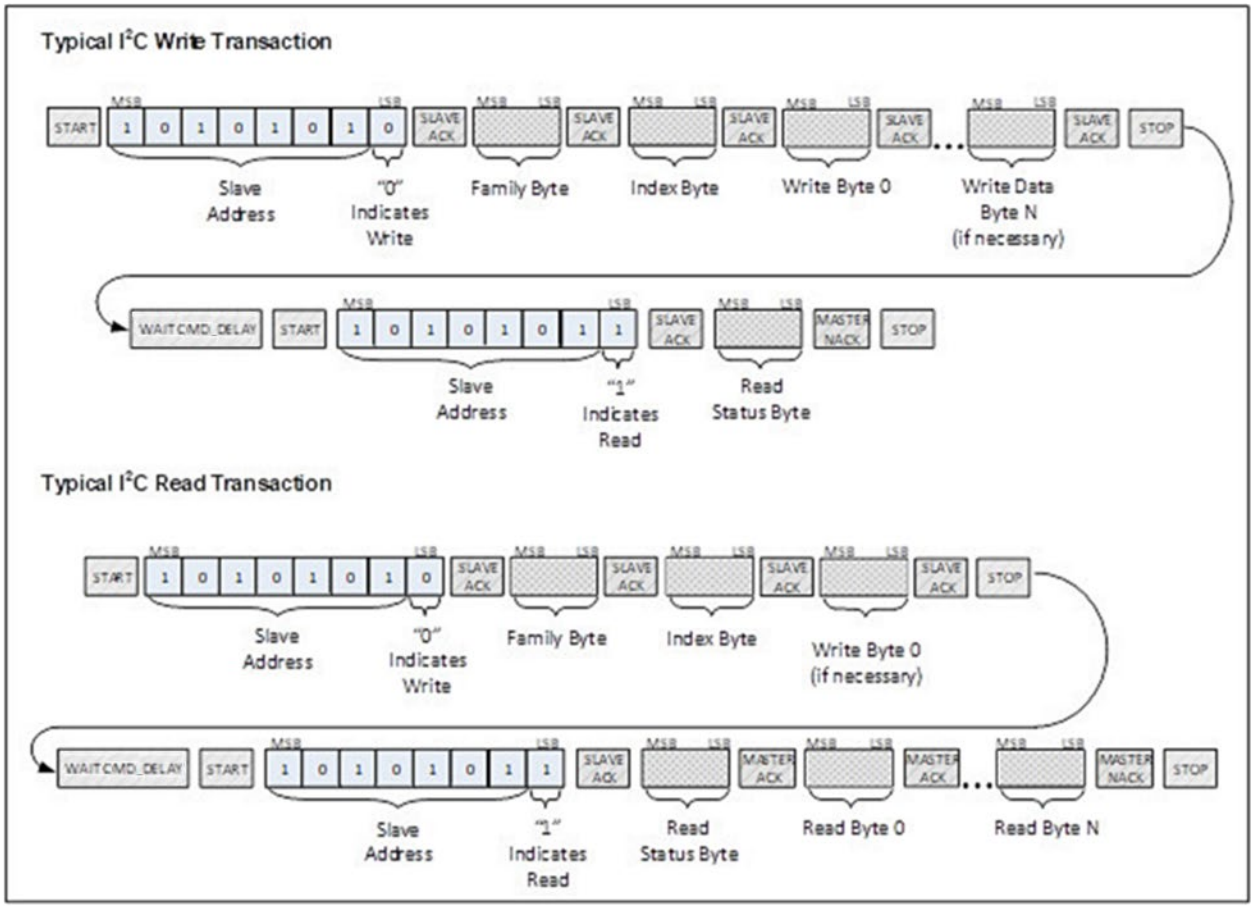


Figure 10. I2C Write/Read data transfer from host microcontroller.

The read status byte is an indicator of the success or failure of the Write Transaction. The read status byte must be accessed after each write transaction to the device. This ensures that write transaction processing is understood and any errors in the device command handling can be corrected. The value of the read status byte is summarized in Table 5.

Table 5. Read Status Byte Value

STATUS BYTE VALUE	DESCRIPTION
0x00	SUCCESS. The write transaction was successful.
0x01	ERR_UNAVAIL_CMD. Illegal Family Byte and/or Index Byte was used. Verify that the Family Byte, Index Byte are valid for the host command sent. Verify that the latest .msbl is flashed.
0x02	ERR_UNAVAIL_FUNC. This function is not implemented. Verify that the Index Byte and Write Byte(s) are valid for the host command sent. Verify that the latest .msbl is flashed.
0x03	ERR_DATA_FORMAT. Incorrect number of bytes sent for the requested Family Byte. Verify that the correct number of bytes are sent for the host command. Verify that the latest .msbl is flashed.
0x04	ERR_INPUT_VALUE. Illegal configuration value was attempted to be set. Verify that the Index Byte is correct for Family Byte 0x44. Verify that the report period is not 0 for host command 0x10 0x02. Verify that the Write byte for host command 0x10 0x03 is in the valid range specified. Verify that the latest .msbl is flashed.
0x05	Application mode: ERR_INVALID_MODE. Not used in application mode.
	Bootloader mode: ERR_ BTLDR_TRY_AGAIN. Device is busy. Insert delay and resend the host command.
0x80	ERR_BTLDR_GENERAL. General error while receiving/flashing a page during the bootloader sequence. Not used.
0x81	ERR_BTLDR_CHECKSUM. Bootloader checksum error while decrypting/checking page data. Verify that the keyed .msbl file is compatible with MAX32664A/B/C/D.
0x82	ERR_BTLDR_AUTH. Bootloader authorization error. Verify that the keyed .msbl file is compatible with MAX32664A/B/C/D.
0x83	ERR_BTLDR_INVALID_APP. Bootloader detected that the application is not valid.
0xFE	ERR_TRY_AGAIN. Device is busy, try again. Increase the delay before the command and increase the CMD_DELAY.
0xFF	ERR_UNKNOWN. Unknown Error. Verify that the communications to the AFE/KX-122 are correct by reading the PART_ID/WHO_AM_I register. For MAX32664B/C, the MAX32664 is in deep sleep unless the host sets the MFIO pin low 250µs before and during the I^2C communications.

I2C Write

The process for an I2C write data transfer is as follows:

1. The bus master indicates a data transfer to the device with a START condition.
2. The master transmits one byte with the 7-bit slave address (most significant 7 bits of the 8-bit address) and a single write bit set to zero. The eight bits to be transferred as a slave address for the MAX32664 is 0xAA for a write transaction.
3. During the next SCL clock following the write bit, the master releases SDA. During this clock period, the device responds with an ACK by pulling SDA low.
4. The master senses the ACK condition and begins to transfer the Family Byte. The master drives data on the SDA circuit for each of the eight bits of the Family byte, and then floats SDA during the ninth bit to allow the device to reply with the ACK indication.
5. The master senses the ACK condition and begins to transfer the Index Byte. The master drives data on the SDA circuit for each of the eight bits of the Index byte, and then floats SDA during the ninth bit to allow the device to reply with the ACK indication.
6. The master senses the ACK condition and begins to transfer the Write Data Byte 0. The master drives data on the SDA circuit for each of the eight bits of the Write Data Byte 0, and then floats SDA during the ninth bit to allow the device to reply with the ACK indication.
7. The master senses the ACK condition and can begin to transfer another Write Data Byte if required. The master drives data on the SDA circuit for each of the eight bits of the Write Data Byte, and then floats SDA during the ninth bit to allow the device to reply with the ACK indication. If another Write Data Byte is not required, the master indicates the transfer is complete by generating a STOP condition. A STOP condition is generated when the master pulls SDA from a low to high while SCL is high.
8. The master waits for a period of CMD_DELAY (2ms is the default) for the device to have its data ready.
9. The master indicates a data transfer to a slave with a START condition.
10. The master transmits one byte with the7-bit slave address and a single write bit set to one. This is an indication from the master of its intent to read the device from the previously written location defined by the Family Byte and the Index Byte. The master then floats SDA and allows the device to drive SDA to send the Status Byte. The Status Byte reveals the success of the previous write sequence. After the Status Byte is read, the master drives SDA low to signal the end of data to the device.
11. The master indicates the transfer is complete by generating a STOP condition.
12. After the completion of the write data transfer, the Status Byte must be analyzed to determine if the write sequence was successful and the device has received the intended command.

I²C Read

The process for an I²C read data transfer is as follows:

1. The bus master indicates a data transfer to the device with a START condition.
2. The master transmits one byte with the 7-bit slave address and a single write bit set to zero. The eight bits to be transferred as a slave address for the MAX32664 is 0xAA for a write transaction. This write transaction precedes the actual read transaction to indicate to the device what section is to be read.
3. During the next SCL clock following the write bit, the master releases SDA. During this clock period, the device responds with an ACK by pulling SDA low.
4. The master senses the ACK condition and begins to transfer the Family Byte. The master drives data on the SDA circuit for each of the eight bits of the Family byte, and then floats SDA during the ninth bit to allow the device to reply with the ACK indication.
5. The master senses the ACK condition and begins to transfer the Index Byte. The master drives data on the SDA circuit for each of the eight bits of the Index byte, and then floats SDA during the ninth bit to allow the device to reply with the ACK indication.
6. The master senses the ACK condition and begins to transfer the Write Data Byte if necessary for the read instruction. The master drives data on the SDA circuit for each of the eight bits of the Write Data byte, and then floats SDA during the ninth bit to allow the device to reply with the ACK indication.
7. The master indicates the transfer is complete by generating a STOP condition.
8. The master waits for a period of CMD_DELAY (2ms is the default) for the device to have its data ready.
9. The master indicates a data transfer to a slave with a START condition.
10. The master transmits one byte with the 7-bit slave address and a single write bit set to one. This is an indication from the master of its intent to read the device from the previously written location defined by the Family Byte and the Index Byte. The master then floats SDA and allows the device to drive SDA to send the Status Byte. The Status Byte reveals the success of the previous write sequence. After the Status Byte is read, the master drives SDA low to acknowledge the byte.
11. The master floats SDA and allows the device to drive SDA to send Read Data Byte 0. After Read Data Byte 0 is read, the master drives SDA low to acknowledge the byte.
12. The master floats SDA and allows the device to drive SDA to send the Read Data Byte N. After Read Data Byte N is read, the master drives SDA low to acknowledge the Read Data Byte N. This process continues until the device has provided all the data that the master expects based upon the Family Byte and Index Byte definition.
13. The master indicates the transfer is complete by generating a STOP condition.

MAX32664 I2C Message Protocol Definition

Table 6 defines the I2C message protocol for the MAX32664.

Table 6. MAX32664 I2C Message Protocol Definitions

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Read Sensor Hub Status	Read sensor hub status (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x00	0x00	-	**Err0[0]**: 0 = No Error; 1 = Sensor Communication Problem **Err1[0]**: Not used **Err2[0]**: Not used **DataRdyInt[3]**: 0 = FIFO below threshold; 1 = FIFO filled to threshold or above. **FifoOutOvrInt[4]**: 0 = No FIFO overflow; 1 = Sensor Hub Output FIFO overflowed, data lost. **FifoInOvrInt[5]**: 0 = No FIFO overflow; 1 = Sensor Hub Input FIFO overflowed, data lost. **HostAccelUfInt[6]**: 0 = No underflow; 1 = Host data to input FIFO is slow and the input FIFO has underflowed. See Table 7 for the bit field table.
Device Mode	Select the device operating mode. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x01	0x00	**0x00**: Exit bootloader mode, enter application mode. **0x01:** Shutdown the MAX32664B/C. Restart by power cycling or pulsing RSTN. **0x02**: Reset. **0x08**: Enter bootloader mode.	-
Device Mode	Read the device operating mode. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x02	0x00	-	**0x00**: Application operating mode. **0x02:** Reset. **0x08**: Bootloader operating mode.

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Set Output Mode	Set the output format of the sensor hub. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x10	0x00	**0x00**: Pause (no data) **0x01**: Sensor Data **0x02:** Algorithm Data **0x03:** Sensor Data and Algorithm Data **0x04**: Pause (no data) **0x05**: Sample Counter byte, Sensor Data **0x06:** Sample Counter byte, Algorithm Data **0x07:** Sample Counter byte, Sensor Data and Algorithm Data	-
Set Output Mode	Set the threshold for the FIFO interrupt bit/pin. The MFIO pin is used as the interrupt and the host should configure this pin as an input interrupt pin. The status bit DataRdyInt is set when this threshold is reached. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x10	0x01	**0x01 to 0xFF:** Sensor Hub Interrupt Threshold for FIFO.	-
Set Output Mode	Set the samples report period (e.g., a value of 25 means a samples report is generated once every 25 samples). (MAX32664C)	0x10	0x02	**0x01 to 0xFF:** LSB is 40ms. N, where a samples report is generated once every N samples.	-
Set Output Mode	Change I^2C address of the MAX32664. (MAX32664B, MAX32664C)	0x10	0x03	**0x0**2 **to 0xFF:** New I^2C address (8-bit I^2C write address)	
Set Output Mode	Set the sensor hub counter. (MAX32664B, MAX32664C)	0x10	0x04	**0x00 to 0xFF:** Counter	

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Read Output Mode	Read the output format of the sensor hub. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x11	0x00	-	**0x00**: Pause (no data) **0x01**: Sensor Data **0x02:** Algorithm Data **0x03:** Sensor Data and Algorithm Data **0x04**: Pause (no data) **0x05**: Sample Counter byte, Sensor Data **0x06:** Sample Counter byte, Algorithm Data **0x07:** Sample Counter byte, Sensor Data, and Algorithm Data
Read Output Mode	Read the threshold for the FIFO interrupt bit/pin. The MFIO pin is used as the interrupt and the host should configure this pin as an input interrupt pin. The status bit DataRdyInt is set when this threshold is reached. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x11	0x01	-	**0x01 to 0xFF:** Sensor Hub Interrupt Threshold for FIFO.
Read Output Mode	Read the samples reporting period (e.g., a value of 25 means a report is generated once every 1s. The default of 1 is one report is generated once per sample or every 40ms). (MAX32664C)	0x11	0x02	-	**0x01 (default) to 0xFF:** LSB is 40ms. N, where a samples report is generated once every N samples.
Read Output Mode	Read the I^2C address of the MAX32664. (MAX32664B, MAX32664C)	0x11	0x03		**0x00 to 0xFF:** I^2C address
Read Output Mode	Read the sensor hub counter. (MAX32664B, MAX32664C)	0x11	0x04		**0x00 to 0xFF:** Counter

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Read Output FIFO	Get the number of samples available in the FIFO. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x12	0x00	-	Number of samples available in the FIFO.
Read Output FIFO	Read data stored in output FIFO. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x12	0x01	-	See Table 8, Output FIFO Format Definitions. The internal FIFO read pointer increments once the sample size bytes have been read.
Read Input FIFO for External Sensors [1]	Read the sensor sample size. (MAX32664A, MAX32664B, MAX32664C)	0x13	0x00	**0x04**: Accelerometer	**0x06:** Bytes per sample for the external accelerometer. Three 16-bit 2's complement with LSB = 0.001g. See Table 9 for an example.
Read Input FIFO for External Sensors	Read the input FIFO size for the maximum number of samples that the input FIFO can hold (16-bit). (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x13	0x01	-	MSB, LSB
Read Input FIFO for External Sensors	Read the sensor FIFO size for the maximum number of samples that the sensor FIFO can hold (16-bit). (MAX32664A, MAX32664B, MAX32664C)	0x13	0x02	**0x04**: Accelerometer	MSB, LSB
Read Input FIFO for External Sensors	Read the number of samples currently in the input FIFO (16-bit). (MAX32664A, MAX32664B, MAX32664C)	0x13	0x03	**0x04**: Accelerometer	MSB, LSB

[1] Systems that have an externally supplied accelerometer.

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Read Input FIFO for External Sensors	Read the number of samples currently in the sensor FIFO (16-bit). (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x13	0x04	-	MSB, LSB
Write Input FIFO for External Sensors	Write data to the input FIFO. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x14	0x00	Sample one value, …, Sample N values See Table 9 for an example.	-
Write Register	Write a value to a writable MAX86140/ MAX86141/ MAXM86161 register. (MAX32664B, MAX32664C)	0x40	0x00	Register address, Register value	-
Write Register	Write a value to a writable MAX30205 register. (MAX32664B)	0x40	0x01	Register address, Register value	-
Write Register	Write a value to a writable MAX30001 register. (MAX32664B)	0x40	0x02	Register address, Register value	-
Write Register	Write a value to a writable MAX30101/ MAX30102 register. (MAX32664A, MAX32664D)	0x40	0x03	Register address, Register value	-
Write Register	Write a value to a writable accelerometer sensor register. (MAX32664A, MAX32664B, MAX32664C)	0x40	0x04	Register address, Register value	-
Read Register	Read the value of a MAX86140/ MAX86141/ MAXM86161 register. (MAX32664B, MAX32664C)	0x41	0x00	Register Address	Register value
Read Register	Read the value of a MAX30205 register. (MAX32664B)	0x41	0x01	Register Address	Register value

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Read Register	Read the value of a MAX30001 register. (MAX32664B)	0x41	0x02	Register Address	Register value
Read Register	Read the value of a MAX30101/ MAX30102 register. (MAX32664A, MAX32664D)	0x41	0x03	Register Address	Register value
Read Register	Read the value of an accelerometer sensor register. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x41	0x04	Register Address	Register value
Get Attributes of the AFE	Retrieve the attributes of the MAX86140/ MAX86141/ MAXM86146/ MAXM86161 AFE. (MAX32664B, MAX32664C)	0x42	0x00	-	Number of bytes in a word for this sensor, Number of registers available for this sensor.
Get Attributes of the AFE	Retrieve the attributes of the MAX30205 AFE. (MAX32664B)	0x42	0x01	-	Number of bytes in a word for this sensor, Number of registers available for this sensor.
Get Attributes of the AFE	Retrieve the attributes of the MAX30001 AFE. (MAX32664B)	0x42	0x02	-	Number of bytes in a word for this sensor, Number of registers available for this sensor.
Get Attributes of the AFE	Retrieve the attributes of the MAX30101/ MAX30102 AFE. (MAX32664A, MAX32664D)	0x42	0x03	-	Number of bytes in a word for this sensor, Number of registers available for this sensor.
Get Attributes of the AFE	Retrieve the attributes of the accelerometer sensor AFE. (MAX32664A, MAX32664B, (MAX32664C, MAX32664D)	0x42	0x04	-	Number of bytes in a word for this sensor, Number of registers available for this sensor.
Dump Registers	Read all the MAX86140/ MAX86141/ MAXM86161 registers. (MAX32664B, MAX32664C)	0x43	0x00	-	Register address 0, register value 0, register address 1, register value 1, …, register address n, register value n

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Dump Registers	Read all the MAX30205 registers. (MAX32664B)	0x43	0x01	-	Register address 0, register value 0, register address 1, register value 1, …, register address n, register value n
Dump Registers	Read all the MAX30001 registers. (MAX32664B)	0x43	0x02	-	Register address 0, register value 0, register address 1, register value 1, …, register address n, register value n
Dump Registers	Read all the MAX30101/ MAX30102 registers. (MAX32664A, MAX32664D)	0x43	0x03	-	Register address 0, register value 0, register address 1, register value 1, …, register address n, register value n
Dump Registers	Read all the accelerometer sensor registers. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x43	0x04	-	Register address 0, register value 0, register address 1, register value 1, …, register address n, register value n
Sensor Mode Enable	Enable the MAX86140/ MAX86141/ MAXM86146/ MAXM86161 sensor. CMD_DELAY = 250ms (MAX32664B, MAX32664C)	0x44	0x00	**0x00**: Disable **0x01**: Enable	-
Sensor Mode Enable	Enable the MAX30205 sensor. CMD_DELAY = 20ms (MAX32664B)	0x44	0x01	**0x00**: Disable **0x01**: Enable	-
Sensor Mode Enable	Enable the MAX30001 sensor. CMD_DELAY = 20ms (MAX32664B)	0x44	0x02	**0x00**: Disable **0x01**: Enable	-
Sensor Mode Enable	Enable the MAX30101/ MAX30102 sensor. CMD_DELAY = 40ms (MAX32664A, (MAX32664D)	0x44	0x03	**0x00**: Disable **0x01**: Enable	-

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Sensor Mode Enable	Enable the accelerometer sensor. CMD_DELAY = 20ms (MAX32664A, MAX32664B, MAX32664C)	0x44	0x04	**0x00, 0x00:** Disable sensor hub accelerometer **0x00, 0x01:** Disable external host accelerometer **0x01, 0x00:** Enable sensor hub accelerometer **0x01, 0x01:** Enable external host accelerometer	-
Sensor Mode Enable	Single command to enable multiple sensors. CMD_DELAY = 20ms For the MAX30101/ MAX30102 sensor, the CMD_DELAY = 40ms Use the total CMD_DELAY of all the sensors that are enabled. Exceptions: 1. If any sensor in the list is already enabled, it turns off and enables again. 2. If enabling one of the sensors in the list fails, the sensor hub disables all the sensors in the command list. 3. All sensors in this command list must be valid available hardware, otherwise, the sensor hub disables all the sensors listed in this command. (MAX32664B, MAX32664C)	0x44	0xFF	**N, SI, SM, SE, … SI, SM, SE:** Enable multiple sensors, where: • N is the number of sensors • SI is the sensor index • SM is the sensor mode • SE is 1 if the sensor is an external host or 0 if the sensor is connected to the sensor hub Sensor indices are defined as: **0x00:** MAX86140/ MAX86141/ MAXM86146/ MAXM86161 **0x01:** MAX30205 **0x02:** MAX30001 **0x03:** MAX30101/MAX30102 **0x04:** Accelerometer Sensor modes are defined in the first byte of Write Bytes field of the Sensor Mode Enable commands, 0x44 0x00 to 0x44 0x04	-

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Sensor Mode Read	Read the MAX86140/ MAX86141/ MAXM86146/ MAXM86161 sensor mode. (MAX32664B, MAX32664C)	0x45	0x00	-	**0x00**: Disabled **0x01**: Enabled
Sensor Mode Read	Read the MAX30205 sensor mode. (MAX32664B)	0x45	0x01	-	**0x00**: Disabled **0x01**: Enabled
Sensor Mode Read	Read the MAX30001 sensor mode. (MAX32664B)	0x45	0x02	-	**0x00**: Disabled **0x01**: Enabled
Sensor Mode Read	Read the MAX30101/ MAX30102 sensor mode. (MAX32664A, (MAX32664D)	0x45	0x03	-	**0x00**: Disabled **0x01**: Enabled
Sensor Mode Read	Read the external accelerometer sensor mode. (MAX32664A, MAX32664B, MAX32664C)	0x45	0x04	-	**0x00**, **0x00:** Sensor hub accelerometer disabled **0x00**, **0x01:** External host accelerometer disabled **0x01**, **0x00:** Sensor hub accelerometer enabled **0x01**, **0x01:** External host accelerometer enabled

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Sensor Configuration	Write the sensor configuration for the MAX86140/ MAX86141/ MAXM86146/ MAXM86161. The LSb0 of the Write Byte is the firmware_default bit. The LSb1 of the Write Byte is the dac_calib bit. CMD_DELAY = 220ms if the first Write Byte is 0x02 or 0x03. (MAX32664C with MAXM86161)	0x46	0x00	**First Byte** **0x00:** Do not use firmware default register settings, and do not run DAC calibration when the algorithm/sensor is enabled. The sensor hub does not overwrite the user settings when the algorithm/sensor is enabled. If the user does not disable AEC, then the sample rate, pulse interval, and LED current are managed by the algorithm. AEC disable is a separate command. Ignore the second byte ppg_cfg1 value. **0x01:** Use firmware default register settings and disable DAC calibration. As soon as the algorithm runs, it uses the firmware defaults. Do not run DAC calibration. Ignore the second byte ppg_cfg1 value.	-

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
	(Continued)			**0x02:** Immediately run DAC calibration but do not use the firmware default. Instead use the second byte ppg_cfg1 value and wait for the user settings to directly write user-defined register values to the MAX8614x. This mode does not use the firmware default. The algorithm does not run the calibration again, because it was run when the command was received. Only this mode immediately runs calibration and uses the second byte ppg_cfg1 value. **0x03 (Default):** Use the firmware default register settings and run DAC calibration when the algorithm/sensor is enabled. Ignore the second byte ppg_cfg1 value and use the firmware default ppg_cfg1. **Second Byte:** The ppg_cfg1 value.	

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Sensor Configuration	Enable/disable wake up on motion. (MAX32664C)	0x46	0x00	First Byte: **0x00:** Disable (1st byte) **0x01:** Enable Second Byte: **0x01 to 0xFE:** Wake up filter period (seconds). Motion must be present during this period time before a wake-up is generated. **0xFF:** Disable (2nd byte) Third Byte: **0x01 to 0x80** LSB = 0.0625g (1/16g. For example 0x08 is 0.5g.	

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Sensor Configuration	Read the sensor configuration for the MAX86140/ MAX86141/ MAXM86146/ MAXM86161. The LSb0 of the Write Byte is the firmware default bit. The LSb1 of the Write Byte is the dac_calib bit. (MAX32664C with MAXM86161)	0x47	0x00	-	**First Byte** **0x00:** Do not use the firmware default register settings, and do not run DAC calibration when the algorithm/sensor is enabled. The sensor hub does not overwrite the user settings when the algorithm/sensor is enabled. If the user does not disable AEC, then the sample rate, pulse interval, and LED current are managed by algorithm. AEC disable is a separate command. Ignore the ppg_cfg1 value. **0x01:** Use the firmware default register settings and disable DAC calibration. As soon as the algorithm is run, it uses the firmware defaults. Do not run DAC calibration. Ignore the ppg_cfg1 value. **0x02:** Immediately run DAC calibration using the ppg_cfg1 value, and wait for the user settings to directly write the user-defined register values to the MAX8614x. This mode does not use the firmware default. The algorithm does not run calibration again, because it was run when the command was received. Only this mode immediately runs calibration. Use the ppg_cfg1 value. **0x03 (Default):** Use the firmware default register settings and run DAC calibration when the algorithm/sensor is enabled. Ignore the ppg_cfg1 value and use the firmware default ppg_cfg1. **Second Byte:** The ppg_cfg value. **Third Byte:** The DAC calibration register value.

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Algorithm Configuration	Automatic Gain Control (AGC) algorithm: Set the target percentage of the full-scale ADC range that the automatic gain control (AGC) algorithm uses. (MAX32664A, MAX32664D)	0x50	0x00	0x00, 0 to 100 percent	-
Algorithm Configuration	AGC algorithm: Set the step size toward the target for the AGC algorithm. (MAX32664A, MAX32664D)	0x50	0x00	0x01, 0 to 100 percent	-
Algorithm Configuration	AGC algorithm: Set the sensitivity for the AGC algorithm. (MAX32664A, MAX32664D)	0x50	0x00	0x02, 0 to 100 percent	-
Algorithm Configuration	AGC algorithm: Set the number of samples to average for the AGC algorithm. (MAX32664A)	0x50	0x00	0x03, Number of samples to average (range is 0 to 255).	-
Algorithm Configuration	Blood Pressure Trending (BPT) algorithm: Set if the user is on blood pressure medication. (MAX32664D)	0x50	0x04	**0x00, 0x00**: Not using blood pressure (BP) medication **0x00, 0x01**: Using BP medication	-
Algorithm Configuration	BPT algorithm: Write the three samples of the systolic BP byte values needed by the calibration procedure. (MAX32664D)	0x50	0x04	0x01, systolic value 1, systolic value 2, systolic value 3	-
Algorithm Configuration	BPT algorithm: Write the three samples of the diastolic BP byte values needed by the calibration procedure. (MAX32664D)	0x50	0x04	0x02, diastolic value 1, diastolic value 2, diastolic value 3	-

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Algorithm Configuration	BPT algorithm: Write the calibration data for this user. (Use the data from the 0x51 0x04 0x03 command). CMD_DELAY = 30ms. (MAX32664D)	0x50	0x04	0x03, 824 bytes of calibration data	-
Algorithm Configuration	BPT algorithm: Configure whether the user is not resting or resting. (MAX32664D)	0x50	0x04	**0x05, 0x00**: Resting **0x05, 0x01**: Not resting	-
Algorithm Configuration	BPT algorithm: Set the SpO_2 coefficients A, B, C. (MAX32664D)	0x50	0x04	0x0B, 4 bytes signed integer A, 4 bytes signed integer B, 4 bytes signed integer C (32-bit integers which are the coefficients times 100,000) The MAXREFDES220# without the cover glass uses the following coefficients as the default values: A = 159584 B = -3465966 C = 11268987	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+$WSpO_2$): Set the SpO_2 coefficients A, B, and C. (MAX32664C)	0x50	0x07	0x00, 4 bytes signed integer A, 4 bytes signed integer B, 4 bytes signed integer C The MAXREFDES103# uses the following coefficients as the default values: A = 0 (0x00000000) B = -25.224999 (0xFFD7FBDD) C = 112.317421 (0x00AB61FE)	

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the motion detection period in seconds. The algorithm considers the state to be motionless if the motion is below the threshold for this duration of time. (MAX32664C)	0x50	0x07	0x01, MSB of period, LSB of period (16-bit unsigned integer, seconds)	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the motion threshold for the WSpO$_2$ algorithm. (MAX32664C)	0x50	0x07	0x02, 4 bytes (32-bit signed integers which are the motion threshold in milli-Gs times 100,000. For example, 0x1C9C380 is 0.3G)	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the WSpO$_2$ AGC timeout (seconds). (MAX32664C)	0x50	0x07	0x03, WSpO$_2$ AGC timeout (8-bit unsigned)	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the SpO$_2$ algorithm timeout (seconds). (MAX32664C)	0x50	0x07	0x04, SpO$_2$ algorithm timeout (8-bit unsigned)	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the initial heart-rate setting. (MAX32664C)	0x50	0x07	0x05, Initial heart-rate setting (8-bit unsigned)	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the user's height. (MAX32664C)	0x50	0x07	0x06, Height (16-bit unsigned integer which is the height in centimeters times 256)	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the user's weight. (MAX32664C)	0x50	0x07	0x07, Weight (16-bit unsigned integer which is the weight in kilograms times 256)	-

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the user's age. (MAX32664C)	0x50	0x07	0x08, Age in years (8-bit unsigned)	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the user's gender. (MAX32664C)	0x50	0x07	0x09, 0x00: Male 0x09, 0x01: Female	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the algorithm run mode. (MAX32664C)	0x50	0x07	0x0A, 0x00: Continuous HRM, continuous SpO$_2$ 0x0A, 0x01: Continuous HRM, one-shot SpO$_2$ 0x0A, 0x02: Continuous HRM 0x0A, 0x03: Sampled HRM 0x0A, 0x04: Sampled HRM, one-shot SpO$_2$ 0x0A, 0x05: Activity tracking only 0x0A,0x06: SpO$_2$ calibration	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Enable the AEC algorithm. (MAX32664C)	0x50	0x07	0x0B, 0x00: Disable 0x0B, 0x01: Enable	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Enable the SCD algorithm. (MAX32664C)	0x50	0x07	0x0C, 0x00: Disable 0x0C, 0x01: Enable	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the target PD current period (period to update the target PD current with the AEC formula). (MAX32664C)	0x50	0x07	0x0D, Target PD current period (16-bit unsigned integer)	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the motion magnitude threshold. (MAX32664C)	0x50	0x07	0x0E, Motion magnitude threshold (16-bit unsigned integer, 0.001g. For example, 0x0032 is 0.05g)	-

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the minimum PD current setting. (MAX32664C)	0x50	0x07	0x0F, Minimum PD current setting (16-bit unsigned integer, 0.1mA) This is the target PD current you would like AEC algorithm to maintain initially. It does not correspond to any register. Once you set what PD current you need, the algorithmcalculates what LED current should be.	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the initial PD current setting. (MAX32664C)	0x50	0x07	0x10, Initial PD current setting (16-bit unsigned integer, 0.1mA)	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the target PD current setting. (MAX32664C)	0x50	0x07	0x11, Target PD current setting (16-bit unsigned integer, 0.1mA)	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Enable the auto target PD current calculation. (MAX32664C)	0x50	0x07	0x12, 0x00: Value of target PD current is used (AGC functionality) 0x12, 0x01: Target PD current is calculated automatically	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the minimum integration time. (MAX32664C)	0x50	0x07	0x13, 0x00: 14.8µs (default) 0x13, 0x01: 29.4µs 0x13, 0x02: 58.7µs 0x13, 0x03: 117.3µs	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the minimum frequency sampling. (MAX32664C)	0x50	0x07	0x14, 0x00: 25sps, averaging = 1 (default) 0x14, 0x01: 50sps, averaging = 2 0x14, 0x02: 100sps, averaging = 4 0x14, 0x03: 200sps, averaging = 8 0x14, 0x04: 400sps, averaging = 16	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the maximum integration time. (MAX32664C)	0x50	0x07	0x15, 0x00: 14.8µs (default) 0x15, 0x01: 29.4µs 0x15, 0x02: 58.7µs 0x15, 0x03: 117.3µs	-

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set the maximum frequency sampling. (MAX32664C)	0x50	0x07	0x16, 0x00: 25sps, averaging = 1 (default) 0x16, 0x01: 50sps, averaging = 2 0x16, 0x02: 100sps, averaging = 4 0x16, 0x03: 200sps, averaging = 8 0x16, 0x04: 400sps, averaging = 16	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set which Slots and PDs are used for input 1 and 2 of the WHRM (Heart Rate) algorithm. Default: 0x0001 MAXM86146 Default: 0x0073 (MAX32664C)	0x50	0x07	0x17, 0xWX, 0xYZ WX is input 1 of the WHRM algorithm. W = 0 for Slot 1 W = 1 for Slot 2 W = 2 for Slot 3 W = 3 for Slot 4 W = 4 for Slot 5 W = 5 for Slot 6 W = 7 for Slot not used X = 0 for PD1 X = 1 for PD2 X = 3 for PD not used. YZ is input 2 of the WHRM algorithm. Y = 0 for Slot 1 Y = 1 for Slot 2 Y = 2 for Slot 3 Y = 3 for Slot 4 Y = 4 for Slot 5 Y = 5 for Slot 6 Y = 7 for Slot not used Z = 0 for PD1 Z = 1 for PD2 Z = 3 for PD not used.	-

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set which LEDs and PDs are used for IR and red LEDs for the input to the WSpO$_2$ algorithm. Default: 0x1020 MAXM86146 Default: 0x2111 (MAX32664C)	0x50	0x07	0x18, 0xWX, 0xYZ WX is the LED/PD used for IR for the WSpO$_2$ algorithm. W = 0 for Slot 1 W = 1 for Slot 2 W = 2 for Slot 3 W = 3 for Slot 4 W = 4 for Slot 5 W = 5 for Slot 6 W = 7 for Slot not used X = 0 for PD1 X = 1 for PD2 X = 3 for PD not used. YZ is the LED/PD used for red for the WSpO$_2$ algorithm and WHRM algorithm. Y = 0 for Slot 1 Y = 1 for Slot 2 Y = 2 for Slot 3 Y = 3 for Slot 4 Y = 4 for Slot 5 Y = 5 for Slot 6 Y = 7 for Slot not used Z = 0 for PD1 Z = 1 for PD2 Z = 3 for PD not used	-
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Set which slots are used for the firing of which LED. Default: 0x12 0x30 0x00 (slot 1:LED1, slot 2: LED2, slot 3: LED3, slot 4-6: Not used) MAXM86146 Default: 0x123456 (MAX32664C)	0x50	0x07	0x19, UV, WX, YZ U is Slot 1 V is Slot 2 W is Slot 3 X is Slot 4 Y is Slot 5 Z is Slot 6 U, V, W, X, Y, Z are defined as: 0: No LED firing 1: LED1 firing 2: LED2 firing 3: LED3 firing 4: LED4 firing 5: LED5 firing 6: LED6 firing 7: LED1 and LED2 firing 8: LED1 and LED3 firing 9: LED2 and LED3 firing	-

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Algorithm Configuration Read	Automatic Gain Control (AGC) algorithm: Read the target percentage of the full-scale ADC range that the AGC algorithm is using. (MAX32664A, MAX32664D))	0x51	0x00	0x00	0 to 100 Percent
Algorithm Configuration Read	AGC algorithm: Read step size toward the target. (MAX32664A, MAX32664D)	0x51	0x00	0x01	0 to 100 Percent
Algorithm Configuration Read	AGC algorithm: Read the sensitivity for the AGC algorithm. (MAX32664A, MAX32664D)	0x51	0x00	0x02	0 to 100 Percent
Algorithm Configuration Read	AGC algorithm: Read the number of samples to average for the AGC algorithm. (MAX32664A, MAX32664D)	0x51	0x00	0x03	Number of samples to average (range is 0 to 255)
Algorithm Configuration Read	BPT algorithm: Read the calibration data results from the calibration procedure. Host can use this for saving the user calibration data when switching users or for writing user calibration data after a reset. (MAX32664D)	0x51	0x04	0x03	824 bytes of calibration data
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO_2): Read the SpO_2 coefficients A, B, and C. (MAX32664C)	0x51	0x07	0x00	4 bytes signed integer A, 4 bytes signed integer B, 4 bytes signed integer C
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO_2): Read the motion detection period in seconds. (MAX32664C)	0x51	0x07	0x01	MSB of period, LSB of period (16-bit unsigned integer)

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Algorithm Configuration	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the motion threshold for the WSpO$_2$ algorithm. (MAX32664C)	0x51	0x07	0x02	4 bytes (32-bit signed integers which are the motion threshold times 100,000)
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the WSpO$_2$ AGC timeout (seconds). (MAX32664C)	0x51	0x07	0x03	WSpO$_2$ AGC timeout (8-bit unsigned)
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the SpO$_2$ algorithm timeout (seconds). (MAX32664C)	0x51	0x07	0x04	SpO$_2$ algorithm timeout (8-bit unsigned)
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the initial heart-rate setting. (MAX32664C)	0x51	0x07	0x05	Initial heart-rate setting (8-bit unsigned)
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the user's height. (MAX32664C)	0x51	0x07	0x06	Height (16-bit unsigned integer which is the height in centimeter times 256)
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the user's weight. (MAX32664C)	0x51	0x07	0x07	Weight (16-bit unsigned integer which is the weight in kilograms times 256)
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the user's age. (MAX32664C)	0x51	0x07	0x08	Age in years (8-bit unsigned)
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the user's gender. (MAX32664C)	0x51	0x07	0x09	0x00: Male 0x01: Female

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the algorithm run mode. (MAX32664C)	0x51	0x07	0x0A	0x00: Continuous HRM, continuous SpO$_2$ 0x01: Continuous HRM, one-shot SpO$_2$ 0x02: Continuous HRM 0x03: Sampled HRM 0x04: Sampled HRM, one-shot SpO$_2$ 0x05: Activity tracking only 0x06: SpO$_2$ calibration
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the AEC algorithm enable. (MAX32664C)	0x51	0x07	0x0B	0x00: Disabled 0x01: Enabled
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the SCD algorithm enable. (MAX32664C)	0x51	0x07	0x0C	0x00: Disabled 0x01: Enabled
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the target PD current period (period to update the target PD current with the AEC formula). (MAX32664C)	0x51	0x07	0x0D	Target PD current period (16-bit unsigned integer)
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the motion magnitude threshold. (MAX32664C)	0x51	0x07	0x0E	Motion magnitude threshold (16-bit unsigned integer, 0.001g)
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the minimum PD current setting. (MAX32664C)	0x51	0x07	0x0F	Minimum PD current setting (16-bit unsigned integer, 0.1mA)
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the initial PD current setting. (MAX32664C)	0x51	0x07	0x10	Initial PD current setting (16-bit unsigned integer, 0.1mA)

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the target PD current setting. (MAX32664C)	0x51	0x07	0x11	Target PD current setting (16-bit unsigned integer, 0.1mA)
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the auto target PD current calculation enable (MAX32664C)	0x51	0x07	0x12	0x00: Value of target PD current is used (AGC functionality) 0x01: Target PD current is calculated automatically
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the minimum integration time setting. (MAX32664C)	0x51	0x07	0x13	0x00: 14.8µs (default) 0x01: 29.4µs 0x02: 58.7µs 0x03: 117.3µs
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read minimum frequency sampling setting. (MAX32664C)	0x51	0x07	0x14	0x00: 25sps, averaging = 1 (default) 0x01: 50sps, averaging = 2 0x02: 100sps, averaging = 4 0x03: 200sps, averaging = 8 0x04: 400sps, averaging = 16
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the maximum integration time setting. (MAX32664C)	0x51	0x07	0x15	0x00: 14.8µs (default) 0x01: 29.4µs 0x02: 58.7µs 0x03: 117.3µs
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the maximum frequency sampling setting. (MAX32664C)	0x51	0x07	0x16	0x00: 25sps, averaging = 1 (default) 0x01: 50sps, averaging = 2 0x02: 100sps, averaging = 4 0x03: 200sps, averaging = 8 0x04: 400sps, averaging = 16

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read the slots and PD configuration. (MAX32664C)	0x51	0x07	0x17	0xWX, 0xYZ WX is the LED/PD used for IR for the WSpO$_2$ algorithm. W = 0 for Slot 1 W = 1 for Slot 2 W = 2 for Slot 3 W = 3 for Slot 4 W = 4 for Slot 5 W = 5 for Slot 6 W = 7 for Slot not used X = 0 for PD1 X = 1 for PD2 X = 3 for PD not used. YZ is the LED/PD used for red for the WSpO$_2$ algorithm. WHRM algorithm. Y = 0 for Slot 1 Y = 1 for Slot 2 Y = 2 for Slot 3 Y = 3 for Slot 4 Y = 4 for Slot 5 Y = 5 for Slot 6 Y = 7 for Slot not used Z = 0 for PD1 Z = 1 for PD2 Z = 3 for PD not used

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read which LEDs and PDs are used for IR and red LEDs for the input to the WSpO$_2$ algorithm. (MAX32664C)	0x51	0x07	0x18	0xWX, 0xYZ WX is the LED/PD used for IR for the WSpO$_2$ algorithm. W = 0 for Slot 1 W = 1 for Slot 2 W = 2 for Slot 3 W = 3 for Slot 4 W = 4 for Slot 5 W = 5 for Slot 6 W = 7 for Slot not used X = 0 for PD1 X = 1 for PD2 X = 3 for PD not used. YZ is the LED/PD used for red for the WSpO$_2$ algorithm. WHRM algorithm. Y = 0 for Slot 1 Y = 1 for Slot 2 Y = 2 for Slot 3 Y = 3 for Slot 4 Y = 4 for Slot 5 Y = 5 for Slot 6 Y = 7 for Slot not used Z = 0 for PD1 Z = 1 for PD2 Z = 3 for PD not used
Algorithm Configuration Read	Wearable Algorithm Suite (WHRM+WSpO$_2$): Read which slots are used for the firing of which LED. (MAX32664C)	0x51	0x07	0x19	0xUV, 0xWX, 0xYZ U is Slot 1 V is Slot 2 W is Slot 3 X is Slot 4 Y is Slot 5 Z is Slot 6 U, V, W, X, Y, Z are defined as: 0: No LED firing 1: LED1 firing 2: LED2 firing 3: LED3 firing 4: LED4 firing 5: LED5 firing 6: LED6 firing 7: LED1 and LED2 firing 8: LED1 and LED3 firing 9: LED2 and LED3 firing
Algorithm Mode Enable	AGC: Enable the AGC algorithm. CMD_DELAY = 20ms (MAX32664A)	0x52	0x00	0x00: Disable 0x01: Enable	-

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Algorithm Mode Enable	AEC: Enable the AEC algorithm.	0x52	0x01	0x00: Disable 0x01: Enable	-
Algorithm Mode Enable	MaximFast: Enable the WHRM, MaximFast algorithm. CMD_DELAY = 40ms (MAX32664A, MAX32664B)	0x52	0x02	0x00: Disable 0x01: Enable Mode 1 Report 0x02: Enable Mode 2, Extended Report	-
Algorithm Mode Enable	Electrocardiogram (ECG): Enable the ECG algorithm.	0x52	0x03	0x00: Disable 0x01: Enable	-
Algorithm Mode Enable	Blood Pressure Trending (BPT): Enable the BPT algorithm. CMD_DELAY = 20ms (MAX32664D)	0x52	0x04	0x00: Disable 0x01: Enable Calibration Mode 0x02: Enable Estimation Mode	-
Algorithm Mode Enable	Wearable Algorithm Suite (WHRM+WSpO$_2$): Enable the algorithm. (MAX32664C)	0x52	0x07	0x00: Disable (CMD_DELAY = 120ms) 0x01: Enable Mode 1 (CMD_DELAY = 320ms) 0x02: Enable Mode 2 (CMD_DELAY = 320ms)	
Bootloader Flash	Set the initialization vector (IV) bytes. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x80	0x00	Use bytes 0x28 to 0x32 from the .msbl file as the IV bytes.	-
Bootloader Flash	Set the authentication bytes. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x80	0x01	Use bytes 0x34 to 0x43 from the .msbl file.	-
Bootloader Flash	Set the number of pages. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x80	0x02	0x00, Number of pages located at byte 0x44 from the .msbl file.	-
Bootloader Flash	Erase the application flash memory. CMD_DELAY = 1400ms. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x80	0x03	-	-

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Bootloader Flash	Send the page values. CMD_DELAY = 680ms. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x80	0x04	The first page is specified by byte 0x4C from the .msbl file. The total bytes for each message protocol are the page size plus 16 bytes of CRC.	-
Bootloader Information	Get bootloader version. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x81	0x00	-	Major version byte, Minor version byte, Revision byte
Bootloader Information	Get the page size in bytes. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0x81	0x01	-	Upper byte of page size, Lower byte of page size
Sensor Hub AFE Authentication Read	Sensor hub authentication AFE sequence: (Wellness App library, MAX32664C)	0xB2	-	-	Twelve bytes of authentication data
Sensor Hub AFE Initialization Vector Read	Get the sensor hub authentication AFE initialization vector (IV) bytes. (MAX32664C)	0xB3	-	-	Twelve bytes of initialization vector data
Sensor Hub AFE Authentication Public Key Write	Set the sensor hub authentication AFE public key: Wellness app library uses this to authenticate the Maxim AFE. (Wellness App library, MAX32664C)	0xB4	-	Public key: twelve bytes	
Sensor Hub authentication AFE Public Key Read	Get the sensor hub authentication AFE public key: Wellness app library uses this to authenticate the Maxim AFE. (Wellness App library, MAX32664C)	0xB5	-	-	Twelve bytes of the public key

HOST COMMAND					MAX32664
FAMILY NAME	**DESCRIPTION**	**FAMILY BYTE**	**INDEX BYTE**	**WRITE BYTES**	**RESPONSE BYTES**
Identity	Read the MCU type. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0xFF	0x00	-	0x00: MAX32625 0x01: MAX32660/MAX32664
Identity	Read the sensor hub version. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0xFF	0x03	-	Major version byte, Minor version byte, Revision byte
Identity	Read the algorithm: version. Deprecated - no longer supported. (MAX32664A, MAX32664B, MAX32664C, MAX32664D)	0xFF	0x07	-	Major version byte, Minor version byte, Revision byte. Deprecated

Table 7 defines the bit fields of the sensor hub status byte.

Table 7. Sensor Hub Status Byte

BIT	7	6	5	4	3	2	1	0
Field	Reserved	HostAccelUfInt	FifoInOverInt	FifoOutOvrInt	DataRdyInt	Err2	Err1	Err0

Table 8 provides the sequence of commands for writing external (host connected) accelerometer data to the input FIFO for the MAX32664A. The KX-122 connected to the MAX32664 is not used. The MAX32664B and MAX32664C implementations are similar and require a couple of commands to be added to the setup sequence.

Table 8. Sequence of Commands to Write External Accelerometer Data to the Input FIFO

HOST COMMAND	COMMAND DESCRIPTION	READ MAX32664 RESPONSE	RESPONSE DESCRIPTION
0xAA 0x10 0x00 0x03†	Set output mode to sensor and algorithm data.	0xAB 0x00	No error.
0xAA 0x10 0x01 0x0F*‡	Set the threshold for the FIFO to 0x0F.	0xAB 0x00	No error.
0xAA 0x44 0x03 0x01*	Enable the MAX30101 sensor. (MAX32664A)	0xAB 0x00	No error.
0xAA 0x44 0x04 0x01 0x01*	Enable the input FIFO for host supplied accelerometer data.	0xAB 0x00	No error.
0xAA 0x52 0x02 0x01*	Enable MaximFast algorithm mode 1. (MAX32664A)	0xAB 0x00	No error.

HOST COMMAND	COMMAND DESCRIPTION	READ MAX32664 RESPONSE	RESPONSE DESCRIPTION
0xAA 0x13 0x00 0x04†	Read the sensor sample size for the accelerometer. (optional)	0xAB 0x00 0x06	No error. 6 bytes is the sample size.
0xAA 0x14 0x00 Sample 1 value to Sample N value*‡	Write data to the input FIFO. 6 bytes per accelerometer sample.	0xAB 0x00	No error.
0xAA 0x00 0x00*	Read the sensor hub status.	0xAB 0x00 0x08	No error. DataRdyInt bit is set
0xAA 0x12 0x00*	Get the number of samples in the FIFO.	0xAB 0x00 0x0F	No error. 0x0F samples are in the FIFO.
0xAA 0x12 0x01*	Read the data stored in the FIFO.	0xAB 0x00 0x03 0x6A 0x43 0x03 0x04 0x92 0x00 0x00 0x00 0x00 0x2E 0x15 0xFC 0xD8 0x00 0x04 0x02 0x3e 0x02 0x76 0x63 0x03 0xE4 0x03, data for fourteen other samples	No error. IR counts = 223811, Red counts = 19778, LED3 = 0, LED4 = 11797, X accelerometer = -0.808, Y accelerometer = 0.004, Z accelerometer = 0.574, Heart Rate = 63.0, Confidence = 99, SpO_2 = 99.6, MaximFast State Machine Status = 3, data for fourteen other samples.

**Mandatory*
†Recommended
‡Required for the MAX32664B and MAX32664C setup sequence

MAX32664 I2C Annotated Application Mode Example and Output FIFO Format

Refer to the following documents for example I2C sequences that the host microcontroller can use to configure the MAX32664 for data streaming. The output FIFO format for the sensors and algorithms are described in these documents.

- User Guide 7087: *Measuring Heart Rate and SpO_2 Using the MAX32664A*
- User Guide 6922: *Measuring Heart Rate Using MAX32664B*
- User Guide 6924: *Measuring SpO_2 and Heart Rate Using MAX32664C*
- User Guide 6921: *Measuring Blood Pressure, Heart Rate, and SpO_2 Using MAX32664D*

I2C Commands to Flash the Application Algorithm/Firmware

The MAX32664 is pre-programmed with bootloader firmware which accepts in-application programming of the Maxim supplied application algorithm/firmware file (.msbl).

To program the MAX32664 .msbl, the host microprocessor can implement the software to flash the .msbl file or the MAX32630FTHR can be used as a programmer. To use the MAX32630FTHR as a programmer, the following four MAX32630FTHR pins should be connected to the MAX32664

pins: P3.4 to SLAVE_SDA, P3.5 to SLAVE_SCL, P5.4 to MFIO, P5.6 to RSTN. The programming instructions and software needed are available in the HR, SpO_2 software download package on the MAXREFDES220 site.

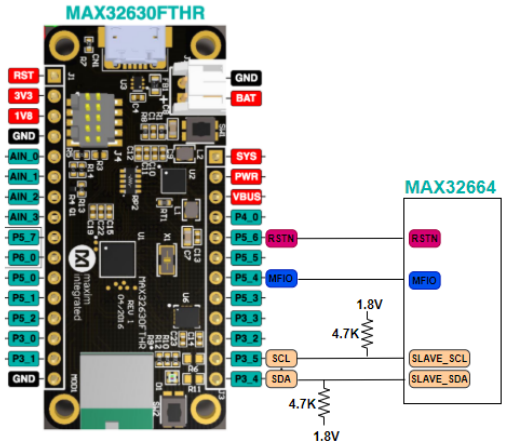


Figure 11. Using the MAX32630FTHR to flash the application .msbl to the MAX32664.

Sample host code to flash the .msbl can be found in the bootloader download package at the MAX32660 website. Another example is located at this Mbed website: Host_Software_MAX32664GWEC_SpO_2 (the host code that interfaces with the MAX32664 on the mbed site is dated. For the latest sample host code to interface with the MAX32664, use the compatible version of sample host code available in the download package on the MAX32664 website). The source code to the python .msbl download file is in the download package at the MAXREFDES220 website. The following constants in the sample .msbl code should be updated to reflect the latest CMD_DELAY definitions:

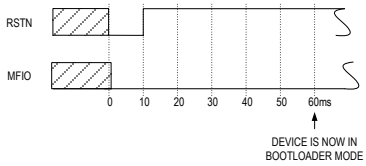
```
#define SS_BOOTLOADER_ERASE_DELAY 1400
#define PAGE_WRITE_DELAY_MS 680
```

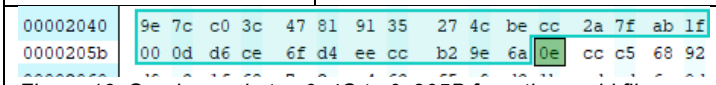
Table 9 is a capture of the I^2C commands that are necessary to flash the application algorithm/firmware to the MAX32664.

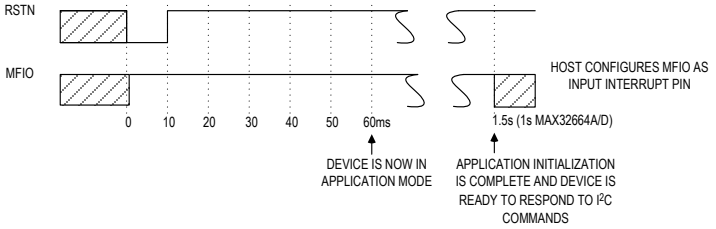
IMPORTANT: *Do not enable the accelerometer if your board does not have the accelerometer.*

This example was captured with the MAX32630FTHR acting as the host microcontroller. The MAX32664 uses the 8-bit slave address of 0xAA. The example encrypted algorithm file used was the MAX32660_SmartSensor_OS24_MaximFast_1.8.2a.msbl (26 pages, 8196 bytes for the page size). Each page sent includes 16 CRC bytes for that page, so there are 8208 bytes per page sent in the payload of the message. The number of pages is located at address 0x44 in the .msbl file. Values for the number of pages, initialization vector, and authorization bytes, might be different for the latest .msbl, but the locations of these values in the .msbl file remain the same. There are additional bytes in the .msbl past the last page; these are the file checksum bytes. Since the bootloader uses the commands listed below and it does not accept files, the file checksum bytes are not used by the bootloader.

Table 9. Annotated I2C Trace for Flashing the Application

HOST COMMAND	COMMAND DESCRIPTION	READ MAX32664 RESPONSE	RESPONSE DESCRIPTION
Sequence the MAX32664 to enter bootloader mode. *  *Figure 12. Sequence to enter bootloader mode.*			
0xAA 0x01 0x00 0x08*	Set mode to 0x08 for bootloader mode.	0xAB 0x00	No error.
0xAA 0x02 0x00	Read mode.	0xAB 0x00 0x08	No error. Mode is bootloader.
0xAA 0xFF 0x00+	Get ID and MCU type.	0xAB 0x00 0x01	No error. MCU is MAX32660/MAX32664.
0xAA 0x81 0x00	Read bootloader firmware version.	0xAB 0x00 0x03 0x00 0x00	No error. Version is 3.0.0.
0xAA 0x81 0x01	Read bootloader page size.	0xAB 0x00 0x20 0x00	No error. Page size is 8192.
0xAA 0x80 0x02 0x00 0x1A*	Bootloader flash. Set the “number of pages” to 31 based on the value at byte 0x44 from the application .msbl file.	0xAB 0x00	No error.
00000044 02 ed 27 af **1a** 00 00 20 04 00 00 00 c2 31 90 2c *Figure 13. Page number byte 0x44 from the .msbl file.*			
0xAA 0x80 0x00 0x1A 0xDB 0xE5 0x0D 0x90 0x79 0xE6 0xC6 0x13 0x87 0xB9*	Bootloader flash. Set the initialization vector bytes to the 0x28 to 0x32 values from the .msbl file.	0xAB 0x00	No error.
00000000 6d 73 62 6c 00 00 00 00 4d 41 58 33 32 36 36 30 00000010 00 00 00 00 00 00 00 00 41 45 53 2d 32 35 36 00 00000020 00 00 00 00 00 00 00 00 **1a db e5 0d 90 79 e6 c6** 00000032 **13 87** **b9** 00 2b f5 ad cd 2e 47 d2 83 23 88 37 63 00000040 02 ed 27 af 1a 00 00 20 04 00 00 00 c2 31 90 2c 00000050 e4 c8 37 e9 18 92 ad 3b 64 e7 0a ed eb 40 c1 66 00000060 e2 23 4f 71 d4 6b 98 e3 a7 f9 85 80 7a 4e 17 e7 *Figure 14. Initialization vector bytes 0x28 to 0x32 from the .msbl file.*			
0xAA 0x80 0x01 0x2B 0xF5 0xAD 0xCD 0x2E 0x47 0xD2 0x83 0x23 0x88 0x37 0x62 0x02 0xED 0x27 0xAF*	Bootloader flash. Set the authentication bytes to the 0x34 to 0x43 values from the .msbl file.	0xAB 0x00	No error.
00000030 13 87 b9 00 **2b f5 ad cd 2e 47 d2 83 23 88 37 63** 00000043 **02 ed 27 af** 1a 00 00 20 04 00 00 00 c2 31 90 2c *Figure 15. Authentication bytes 0x34 to 0x43 from the .msbl file.*			
0xAA 0x80 0x03*	Bootloader flash. Erase application.	0xAB 0x00	No error.
0xAA 0x80 0x04 0xC2 0x31 0x90 ... 0x9E 0x6A 0x0E*	Bootloader flash. Send page bytes 0x4C to 0x205B from the .msbl file.	0xAB 0x00	No error.
00000040 02 ed 27 af 1a 00 00 20 04 00 00 00 **c2 31 90 2c** 00000050 **e4 c8 37 e9 18 92 ad 3b 64 e7 0a ed eb 40 c1 66** 0000006f **e2 23 4f 71 d4 6b 98 e3 a7 f9 85 80 7a 4e 17** **e7**			

HOST COMMAND	COMMAND DESCRIPTION	READ MAX32664 RESPONSE	RESPONSE DESCRIPTION
 *Figure 16. Send page bytes 0x4C to 0x205B from the .msbl file.*			
0xAA 0x80 0x04 0xCC 0xC5 0x68 ... 0xF7 0xD6 0x4C*	Bootloader flash. Send page bytes 0x205C to 0x406B from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0x2E 0xA6 0x13 ... 0x84 0xF7 0xCF*	Bootloader flash. Send page bytes 0x406C to 0x607B from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0xD7 0x1F 0x7F ... 0x55 0xAB 0xB8*	Bootloader flash. Send page bytes 0x607C to 0x808B from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0xC4 0x63 0x2B ... 0x48 0xCD 0x52*	Bootloader flash. Send page bytes 0x808C to 0xA09B from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0x89 0x33 0x22 ... 0x31 0xAD 0x19*	Bootloader flash. Send page bytes 0xA09C to 0xC0AB from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0x8B 0x97 0x18 ... 0xF3 0xCF 0x90*	Bootloader flash. Send page bytes 0xC0AC to 0xE0BB from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0xD0 0x78 0x38 ... 0x1F 0x7F 0x92*	Bootloader flash. Send page bytes 0xE0BC to 0x100CB from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0xB1 0xE9 0x8F ... 0xF4 0x23 0xD8*	Bootloader flash. Send page bytes 0x100CC to 0x120DB from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0xF8 0xC6 0x83 ... 0xF4 0x24 0xE2*	Bootloader flash. Send page bytes 0x120DC to 0x140EB from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0x1F 0x4F 0x5C ... 0xCC 0x2E 0xCD*	Bootloader flash. Send page bytes 0x140EC to 0x160FB from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0x40 0x1F 0x03 ... 0x26 0xEB 0xB9*	Bootloader flash. Send page bytes 0x160FC to 0x1810B from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0x2F 0xD9 0xB2 ... 0xEE 0x2A 0x8F*	Bootloader flash. Send page bytes 0x1810C to 0x1A11B from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0x51 0x32 0x47 ... 0x41 0xE6 0x47*	Bootloader flash. Send page bytes 0x1A11C to 0x1C12B from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0x22 0xA6 0x06 ... 0x2A 0xCB 0x44*	Bootloader flash. Send page bytes 0x1C12C to 0x1E13B from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0x68 0x9E 0x1E ... 0x53 0x89 0xE8*	Bootloader flash. Send page bytes 0x1E13C to 0x2014B from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0x5F 0x1A 0x6A ... 0x14 0xA1 0x85*	Bootloader flash. Send page bytes 0x2014C to 0x2215B from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0xE8 0xDE 0xC9 ... 0x81 0xD8 0x00*	Bootloader flash. Send page bytes 0x2215C to 0x2416B from the .msbl file.	0xAB 0x00	No error.

HOST COMMAND	COMMAND DESCRIPTION	READ MAX32664 RESPONSE	RESPONSE DESCRIPTION
0xAA 0x80 0x04 0x0E 0xD2 0x16 ... 0x8D 0x69 0xEE*	Bootloader flash. Send page bytes 0x2416C to 0x2617B from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0x2F 0x4B 0x38 ... 0x02 0xA7 0xDC*	Bootloader flash. Send page bytes 0x2617C to 0x2818B from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0xA5 0xFE 0xFD ... 0xE3 0x38 0x89*	Bootloader flash. Send page bytes 0x2818C to 0x2A19B from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0x52 0x88 0x9A ... 0xF0 0xC5 0x9D*	Bootloader flash. Send page bytes 0x2A19C to 0x2C1AB from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0xA3 0xA6 0x92 ... 0xA0 0x4D 0xBE*	Bootloader flash. Send page bytes 0x2C1AC to 0x2E1BB from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0x47 0x09 0x75 ... 0x24 0xBD 0x3D*	Bootloader flash. Send page bytes 0x2E1BC to 0x301CB from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0x44 0xEC 0xE6 ... 0xBC 0xC9 0x5E*	Bootloader flash. Send page bytes 0x301CC to 0x321DB from the .msbl file.	0xAB 0x00	No error.
0xAA 0x80 0x04 0xD3 0x58 0x34 ... 0x62 0x00 0x37*	Bootloader flash. Send page bytes 0x321DC to 0x341EB from the .msbl file.	0xAB 0x00	No error.
Sequence the MAX32664 to enter application mode. *  *Figure 17. Sequence to enter application mode.*			
Alternately, the MAX32664 can be commanded to application mode.+			
0xAA 0x01 0x00 0x00+	Set mode to 0x00 for bootloader mode.	0xAB 0x00	No error.
0xAA 0x02 0x00+	Read mode.	0xAB 0x00 0x00	No errors. Mode is application.
0xAA 0xFF 0x00	Get ID and MCU type.	0xAB 0x00 0x01	No error. MCU is MAX32660/MAX32664
0xAA 0xFF 0x03	Get Sensor Hub version.	0xAB 0x00 0x01 0x08 0x02	No error. Version is 1.8.2.
0xAA 0x42 0x03+	Get the MAX30101 AFE register attributes.	0xAB 0x00 0x01 0x24	No error. Attributes are 1 byte, 0x24 registers available.
0xAA 0x43 0x03	Read all the MAX30101 registers.	0xAB 0x00 0x00 0x00 0x01 0x00 0x02 0x40…	No error. Reg 0x00=0, reg 0x01=0, reg0x02=0x40, … Returns the Read Status Byte and 36 pairs of numbers.

HOST COMMAND	COMMAND DESCRIPTION	READ MAX32664 RESPONSE	RESPONSE DESCRIPTION
0xAA 0x41 0x03 0x07	Read the MAX30101 register 7.	0xAB 0x00 0x00	No error. Register 0x07 is 0.

**Mandatory*

+Recommended

It is recommended to program the latest version of the MAX32664 sensor hub application algorithm/firmware .msbl file into the MAX32664 chip. Check the version that is programmed into the chip by using the command “Identity, Read sensor hub version.” The latest sensor hub algorithm/firmware is available for download for the MAX32664, MAXREFDES220#, and MAXREFDES101# from the Maxim website.

In-Application Programming of the MAX32664

The MAX32664 allows for in-application programming of the application algorithm/firmware.

In-application programming allows for the programming of the sensor hub application firmware during manufacturing and for allowing over-the-air (OTA) updates of the application firmware in the product. **Figure 18** is a flowchart of the in-application programming.

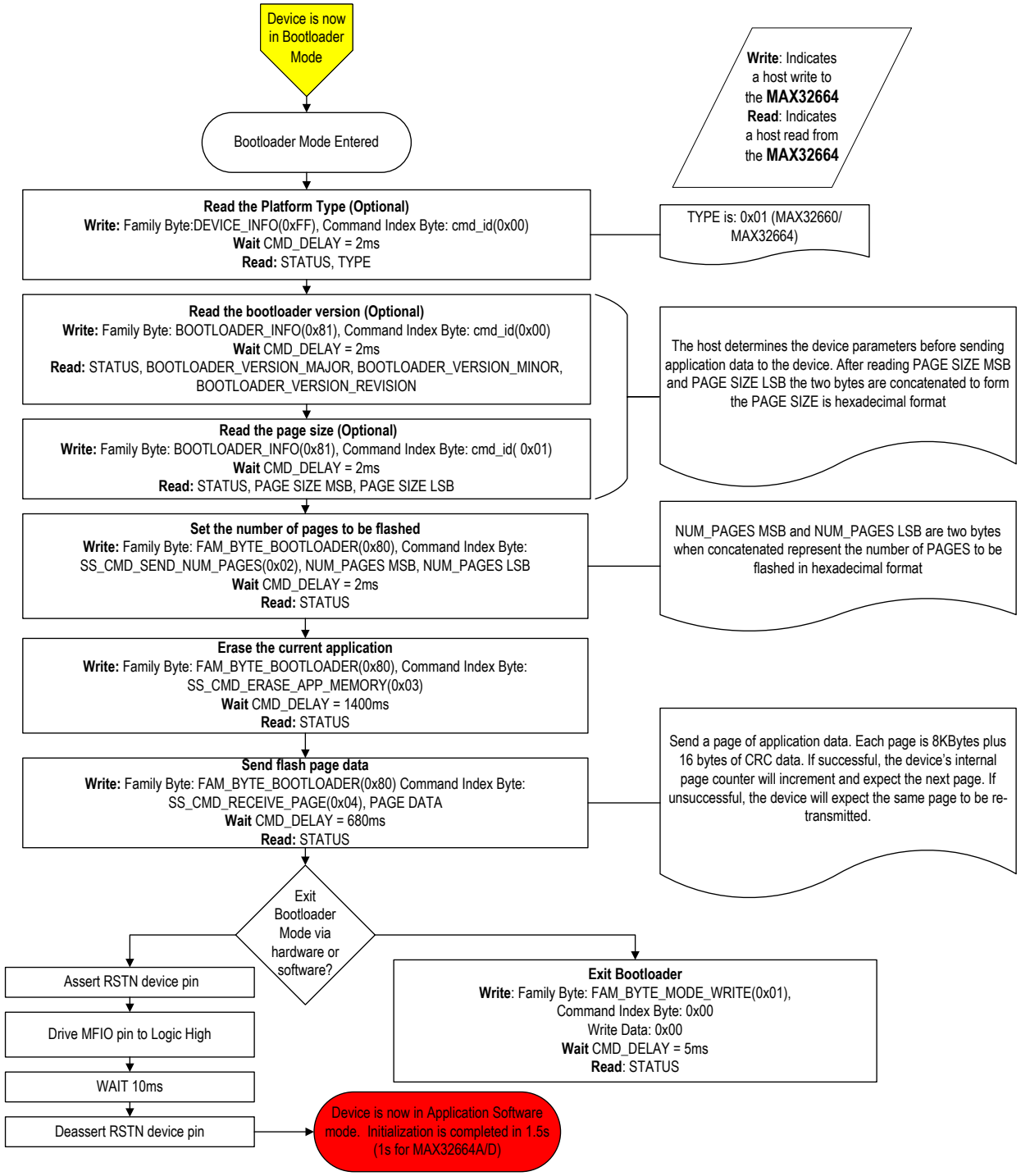


Figure 18. MAX32664 in-application programming flowchart.

MAX32664 APIs and Methods for Reset, Sleep, Status, Heartbeat

Table 10 summarizes the commands and methods to place the MAX32664 into reset or sleep, to interrogate its status, or to generate the “heartbeat” (a periodic signal generated by the software to indicate normal operation).

Table 10. MAX32664 I²C Message Protocol Definitions

COMMAND NAME	HOST COMMAND TO MAX32664	DESCRIPTION
MAX32664 Soft Reset	0xAA 0x01 0x00 0x02	Puts MAX32664 into reset.
MAX30101/MAX30102 AFE Soft Reset by Write Register to AFE	0xAA 0x40 0x03 0x09 0x40	Write 0x40 to MAX30101/ MAX30102 register 0x09 to issue a soft reset to the MAX30101. The AFE must be enabled using the enable command.
MAX32664A/B/C/D Shutdown	0xAA 0x01 0x00 0x01	Place the MAX32664 into shutdown (MAX32660 "Backup" mode with RAM disabled). Restart by power cycling or pulsing RSTN.
MAX32664 Sleep between Interrupts		V20.2.0+, v30.2.4+, v32.1.2+, v33.6.0+ use sleep/deep-sleep for low-powered mode between polling periods.
MAX86140/MAX86141/ MAXM86146/MAXM86161 AFE Shutdown. Use Write Reg to AFE	0xAA 0x40 0x00 0x0D 0x02	Write 0x06 to MAX86140/1, MAXM86146/61 register 0x0D (System Control) to put the MAX86140/1, MAXM86146/61 into shutdown (SHDN) mode. The AFE must be enabled using the enable command when using the read, write AFE register command)
MAX30101/MAX30102 AFE Sleep, Use Write Reg to AFE	0xAA 0x40 0x03 0x09 0x80	Write 0x80 to MAX30101/ MAX30102 register 0x09 (Mode Configuration) to put the MAX30101/MAX30102 into shutdown mode. The AFE must be enabled using the enable command when using the read, write AFE register command)
KX122 Standby. Use Write Reg to Sensor	0xAA 0x40 0x04 0x18 0x00	Write 0x00 to KX122 register 0x018 (CNTL1) to put the KX122 into "Standby" mode. The KX122 must be enabled using the enable command when using the read, write KX122 register command)
MAX32664 Hard Reset	Use MFIO and RSTN pins according to Figure 5 and Figure 6.	
WDT in MAX32664 Bootloader Mode		Not implemented.

WDT in MAX32664 .msbl Application mode		Not implemented.
Bootloader or Application Status	0XAA 0x02 0x00	Send the read mode command. Response is 0xAB 0x00 0x08 if in bootloader mode or 0xAB 0x00 0x00 if in application mode.
Heartbeat (signal to signify that the sensor hub firmware is not stuckl) for Application Mode		Not implemented.

Default Application .msbl Versions Pre-Programmed on the MAX32664A/B/C/D

The MAX32664A/B/C/D are pre-programmed with the bootloader and the application .msbl application/sensor hub version listed in Table 11. The pre-programmed application .msbl versions are not updated by Maxim. The pre-programmed parts may not be programmed with the latest version of the .msbl application. It is recommended that the sensor hub be updated with the latest application .msbl available on the Maxim Integrated website in order to be compatible with the latest sensor hub documentation.

Table 11. MAX32664A/B/C/D/MAXM86146 Pre-Programmed .msbl Version

MAXIM PART	PRE-PROGRAMMED .msbl APPLICATION/SENSOR HUB VERSION
MAX32664A	Version 1.9.1 (deprecated)
MAX32664B	Version 20.1.2 (deprecated)
MAX32664C	Version 30.2.2 (deprecated)
MAX32664D	Version 40.2.2 (deprecated)
MAXM86146	Application not pre-programmed.

MAX32664 Processing Capabilities

The MAX32664 IC hardware is the same as the MAX32660.

1. MIPS: Arm Cortex-M4 with FPU: 1.27 Dhrystone MIPS/MHz
2. RAM: 96kB SRAM
3. Flash: 256kB Flash Memory
4. CPU Frequency: 96MHz

References

MAX32664 website: MAX32664 user guides; C-keyed .msbl for MAX32664A, MAX32664D; sample host code: MAX32664 Design Resources Website.

Application Note 7148, protocol definition between sample host (MAX32630) and PC UART/BLE: Interface Guide for MAX32664 Sensor Hub-Based Reference Design Platforms

Frequently Asked Questions: Maxim Support Center

MAXREFDES101# hardware, software files: MAXREFDES101#: Health Sensor Platform 2.0

MAXREFDES103# hardware, software files: MAXREFDES103#: Wrist-Based SpO_2, HR, and HRV Health Sensor Platform

MAXREFDES220# hardware, software files: MAXREFDES220#: Finger Heart Rate and Pulse Oximeter Smart Sensor with Digital Signal Processing

Trademarks

Revision History

REVISION NUMBER	REVISION DATE	DESCRIPTION	PAGES CHANGED
0	01/19	Initial release	—
1	06/19	Added the MAX32664B/C/D application firmware descriptions, Table 4 for additional GPIOs for the MAXREFDES101#, and 0x11 to the commands. Changed 0x14 0x04 to 0x14 0x00. Updated the 0x44 0x04 command and Sequence of Commands for the external host. Added PD1, PD2 to 0x51/0 0x05 0x09, and additional modes to 0x52 0x02 and 0x52 0x04. Labeled commands in Table 6 and Table 8 as MAX32664A/B/C/D. Added mode 2 for MaximFast in Table 8. Updated Table 9. Added Table 12, Table 13, and Table 14 for annotated I^2C traces for MAX32664C and MAX32664D. Corrected the timing for CMD_DELAY and the application response time to I^2C commands. Added descriptions for the Wearable Algorithm Suite (WHRM+WSpO_2) and updated MFIO for WHRM+WSpO_2. Added sensor configuration, MAXM86161 calibration, Table 17 and mode 2 for the WHRM+WSpO_2 algorithm. Updated host command 0x51/0 0x07 0x17/8 for LED/PD configuration and the TRY_AGAIN values in Table 5.	1–72
2	11/19	Updated Table 1 by removing content for older MAX32664C WHRM and SpO_2 algorithms. Updated section MAX32664 GPIOs and RSTN Pin and Table 2 by adding content that the MAX32664B WHRM v20.2.x+ uses a polling method for the MFIO pin. Updated section MAX32664 Bootloader Mode. Updated Table 6 by adding shutdown command to Device Mode (0x01 0x00 0x01), removing the WHRM command (0x50) and WSpO_2 command (0x51), adding commands to change and read the I2C address (0x10/11 0x03), set and read the sensor hub counter (0x10/11 0x04), and a single command to enable multiple sensors (0x44 0xFF), reversing the systolic command (0x50 0x04 0x01) and the diastolic command (0x50 0x04 0x02), updating 0x50/51 0x07 0x017 for which slots and PDs are used, updating 0x50/51 0x07 0x018 for which LEDs and PDs are used for IR and red LEDs, and adding 0x50/51 0x07 0x18 for which slots are used for the firing of each LED. Removed Table 8 and Table 10 through Table 14. Updated section MAX32664 I2C Annotated Application Mode Example and Output FIFO Format. Updated Table 10 by adding the Shutdown command.	7, 12–13, 19–21, 26, 32–33, 38–39, 46–47, 57, 63

3	8/20	Updated the *Introduction, MAX32664 Variants, MAX32664 GPIOs and RSTN Pin, MAX32664 Bootup and Application Mode, MAX32664 Application Mode, Bit Transfer Process, I²C Write, I²C Read, I²C Commands to Flash the Application Algorithm/Firmware,* and *Default Application .msbl Versions Pre-Programmed on the MAX32664A/B/C/D* sections; updated Tables 1–2, 4–6, 8, and 10–11; replaced Figures 1–2, 6, 9–10, and 18; added Figures 3–5, and renumbered subsequent figures; added *MAXREFDES103#, Additional Sensor Hub Products*, *MAX32664 Processing Capabilities, References*, and *Trademarks* sections	4–5, 10–11, 14–21, 23, 52–53, 59–60, 62