MAX22512/MAX22522 IO-Link Device Transceiver with Integrated Arm Cortex-M0 and Analog Front-End User Guide

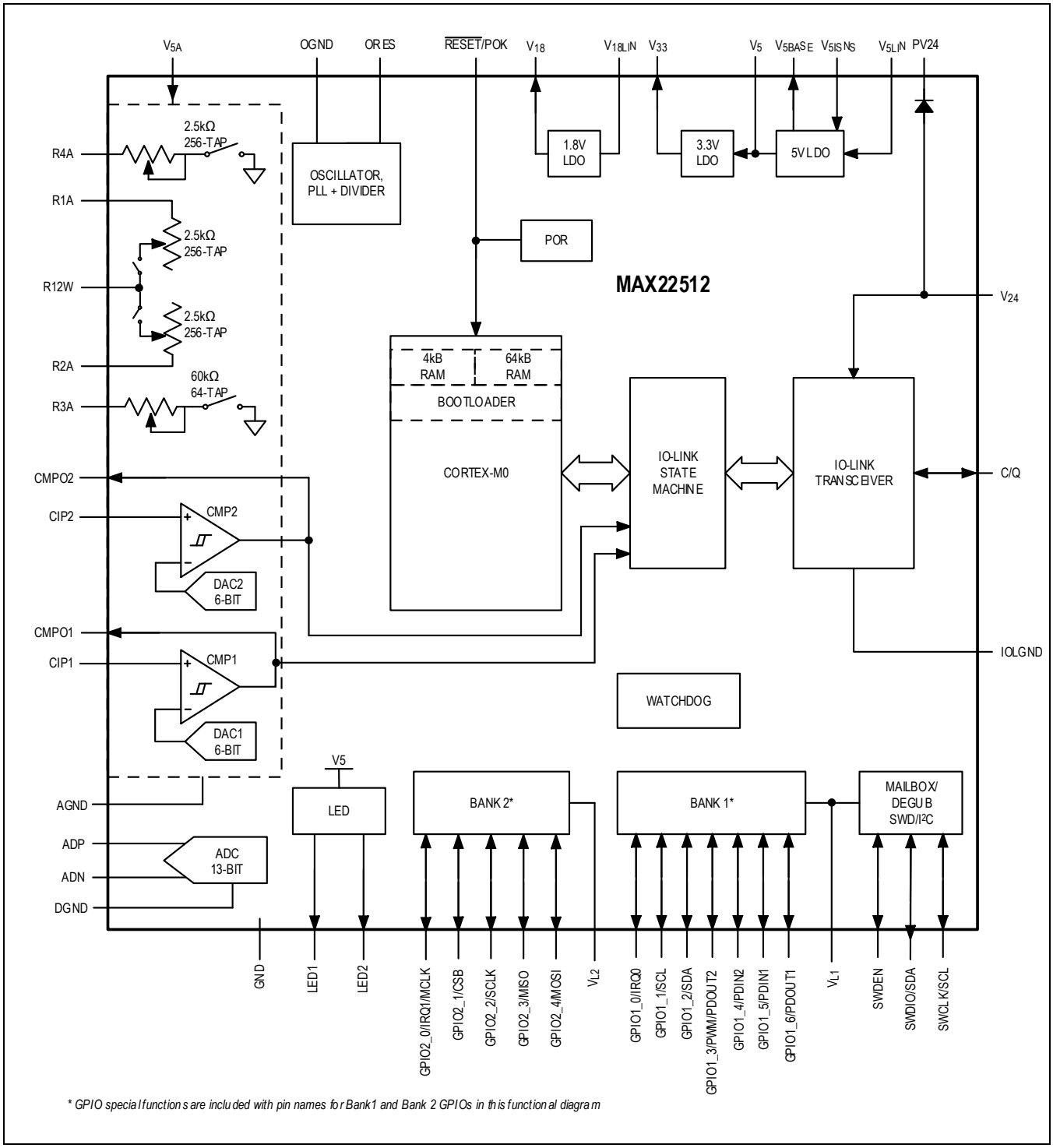
Rev. 0; 09/26

Abstract

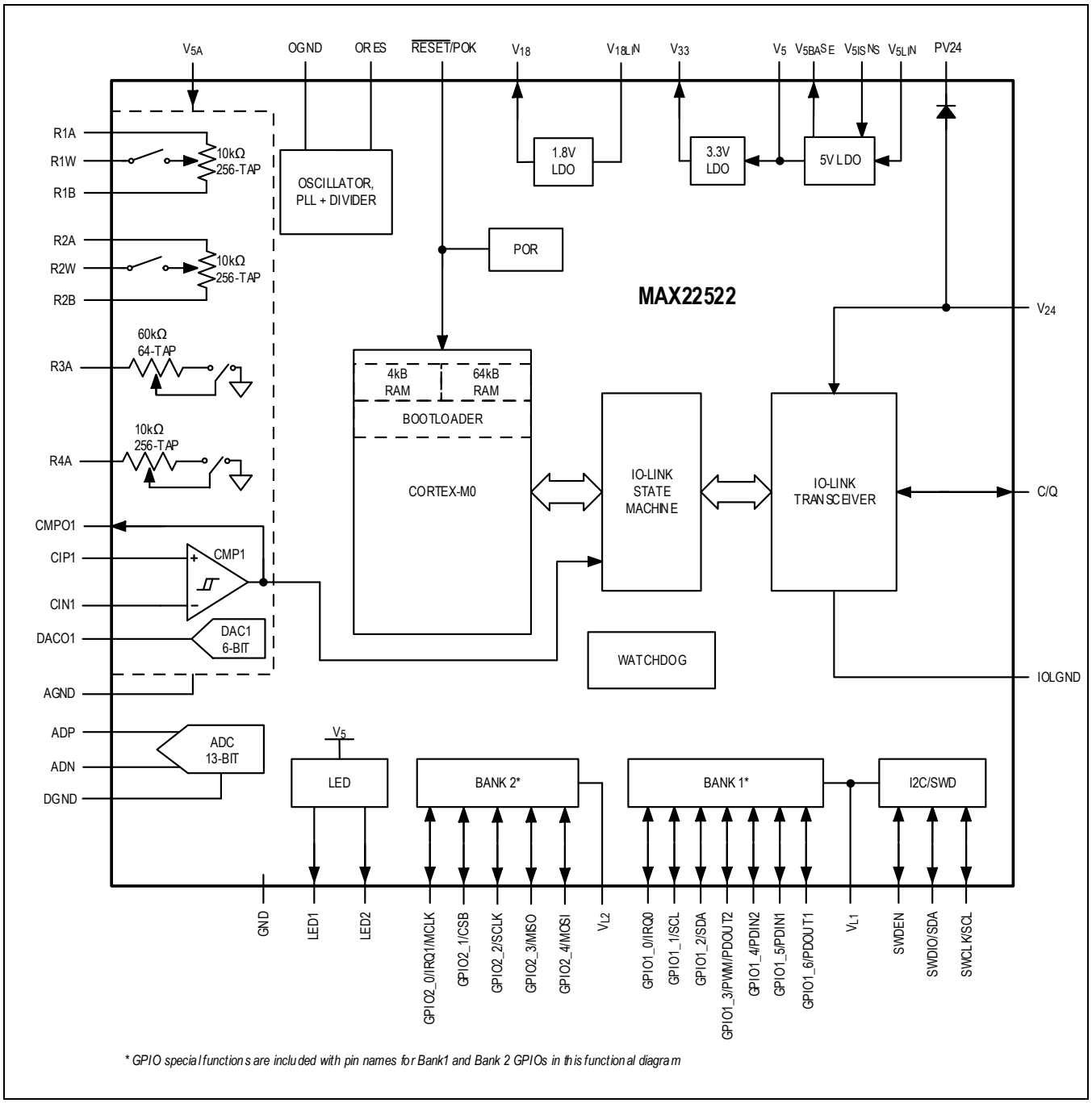
This user guide provides application developers information on how to use the memory and peripherals of the MAX22512/MAX22522 IO-Link device transceivers with an integrated Arm® Cortex®-M0 and analog front-end. The user guide covers detailed information for all registers and fields in the device. It also provides guidance for managing all the peripherals, clocks, power, and startup for the device.

TABLE OF CONTENTS

Functional Diagram



** GPIO special functions are included with pin names for Bank1 and Bank 2 GPIOs in this functional diagram*



Power-Up

When power is applied to the MAX22512/MAX22522, an internal check and verification process starts. Once completed, the integrated Arm Cortex-M0 is powered up and Tiny Bootloader execution begins, as shown in *Figure 1*.

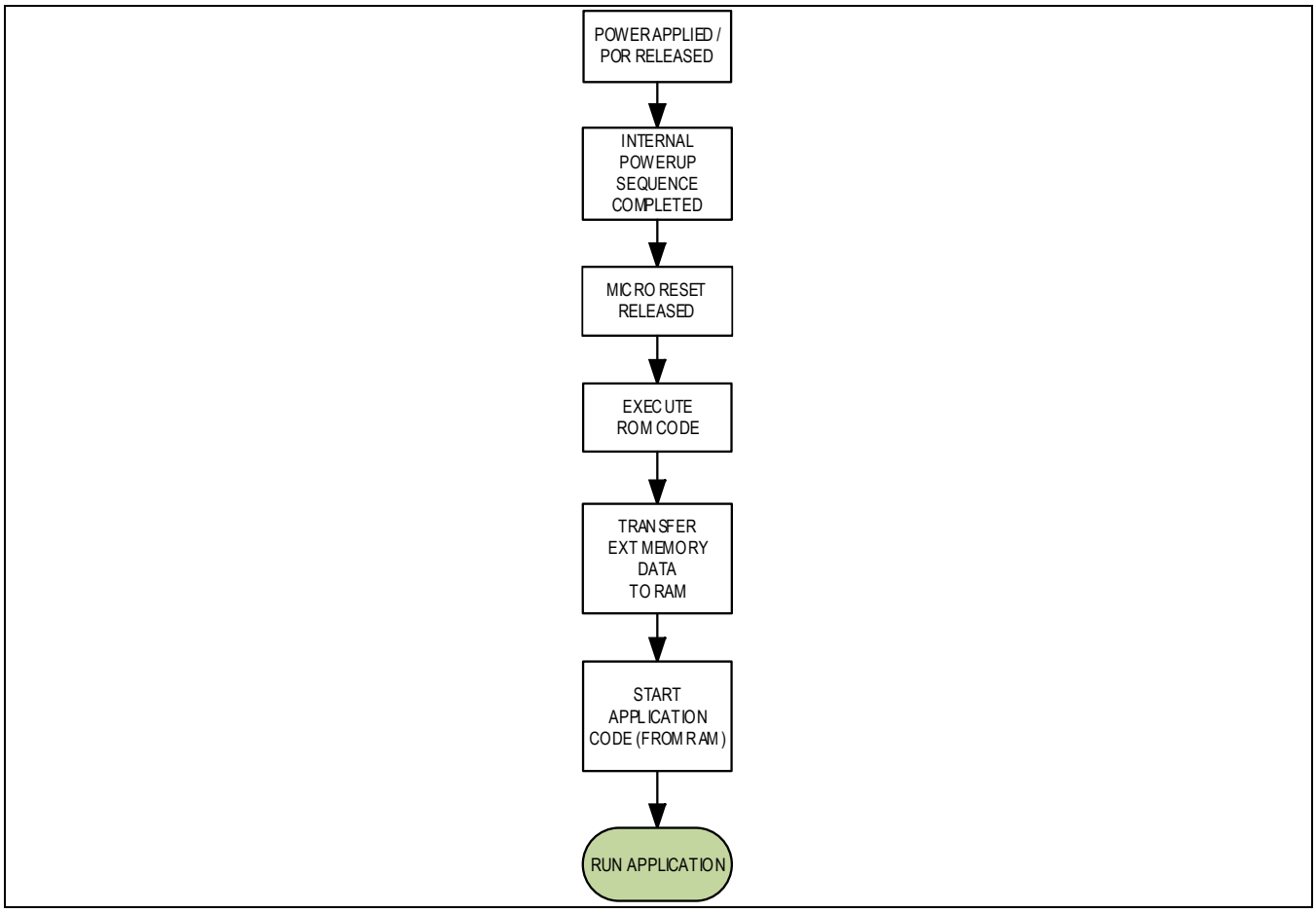


Figure 1. MAX22512/MAX22522 Startup Flowchart

Table 1 shows the memory mapping for the device.

Table 1. User Memory Map

USER MEMORY	START ADDRESS
ROM	0x0008 0000
RAM	0x0000 0000
4kB DATA RAM	0x3000 0000
PERIPHERALS	0x4000 0000 (see *Table 2*)
SYSTICK	0xE000 E010

Memory Map

The MAX22512/MAX22522 use application memory mapping to execute the application software stored in the external memory (accessed through I^2C or SPI). The 64kB RAM is relocated to address 0x00000000 to run the code (*Figure 2).* Refer to the Arm Cortex-M0 documentation for more information on the interrupt vector.

The available onboard RAM for the application software is 4kB data RAM + the amount of RAM not used by the application software. It can be calculated as:

Application software available RAM = 4kB + 64kB − (application software code size)

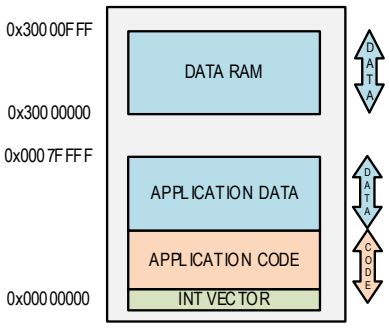


Figure 2. Memory Mapping

Boot Time

The total boot time for the 64kB is 100ms (typ.). See *Figure 3*.

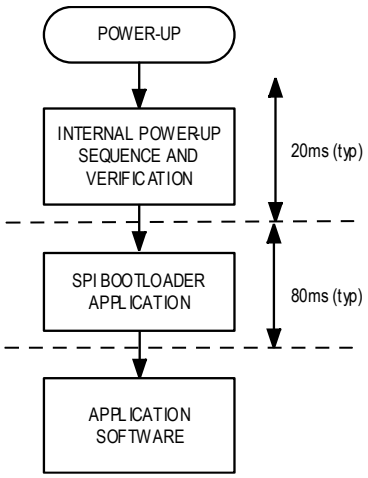


Figure 3. Boot Time

Peripheral Memory Mapping

Table 2 shows the peripheral address map. Peripherals are discussed in more detail in the subsequent sections.

Table 2. Peripheral Memory Map

PERIPHERAL	START ADDRESS	END ADDRESS
General-Purpose Input/Output (GPIO)	0x40000000	0x4000002C
I^2C Host	0x40010000	0x4001001C
System Watchdog (WDT)	0x40020000	0x40020018

Clock Control	0x40030000	0x40030090
Serial Peripheral Interface (SPI) Host	0x48009000	0x48009048
IO-LINK	0x40040000	0x400403FF
IO-LINK Buffers	0x40040400	0x400408FF
Analog-to-Digital Converter (ADC)	0x40050000	0x400500FF
Variable Resistors, Comparator, and DAC (DIGIPOT)	0x40060000	0x400600FF
Enhanced Timer	0x40080000	0x400800FF

General-Purpose Inputs/Outputs (GPIOs)

The MAX22512/MAX22522 integrate 12 GPIO pins, divided into two groups or banks: Bank 1 and Bank 2.

Bank 1 includes seven GPIOs: GPIO1_0 to GPIO1_6. Bank 1 GPIOs are powered by V_{L1} and operate from 2.5V to 5.5V. Bank 2 includes five GPIOs: GPIO2_0 to GPIO2_4. Bank 2 GPIOs are powered by V_{L2} and operate from 1.62V to 5.5V. All GPIOs are referenced to AGND.

All GPIOs can be configured as inputs or open-drain or push-pull outputs and feature pull-ups/pull-downs that can be enabled or disabled. Additionally, all GPIO pins can be programmed with an alternate function. *Table 3* shows the control registers for the GPIOs.

Table 3. GPIO Configuration Registers

GPIO CONTROL/ CONFIGURATION REGISTER	BANK 1 REGISTER ADDRESS	BANK 2 REGISTER ADDRESS	GPIO DIRECTION	FUNCTION
GPIOx_OUT	0x00	0x18	Output	Configures GPIOs as outputs.
GPIOx_IN	0x04	0x1C	Input	Reflects the logic state of GPIOs.
GPIOx_PUD_DIS	0x08	0x20	In/Out	Disables/enables pull-up on GPIOs.
GPIOx_PUD_SEL	0x0C	0x24	In/Out	Configures pull-up/pull-down on GPIOs.
GPIOx_ALT_FUNC	0x10	0x28	In/Out	Enables alternate functionality on GPIOs.
GPIOx_EN	0x14	0x2C	Out	Enables/disables GPIO as an output.

Figure 4 shows the GPIO pin logic. This logic applies to all GPIO pins in both Bank 1 and Bank 2.

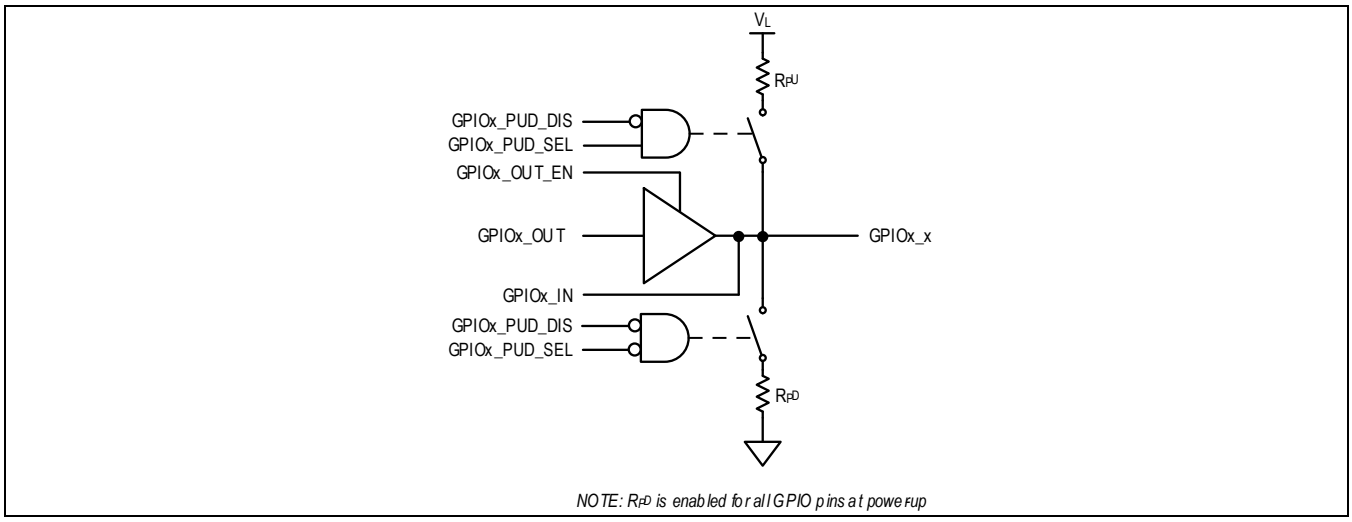


Figure 4. GPIO Logic Circuit

Alternate Functions

All GPIOs in Bank 1 and Bank 2 are configurable as standard I/Os. Additionally, each GPIO can be configured for a specialized or alternate functionality. Alternate functions are enabled by setting bits in the GPIO1_ALT_FUNC and/or GPIO2_ALT_FUNC registers.

Table 4 shows the alternate functions for each GPIO in Bank 1. *Table 5* shows the alternate functions for each GPIO in Bank 2.

Table 4. Bank 1 GPIO Alternate Functions

BANK 1 GPIO	ALTERNATE FUNCTION
GPIO1_0	IRQ0 Input
GPIO1_1	I²C Host Controller – SCL
GPIO1_2	I²C Host Controller – SDA
GPIO1_3*	PWM Output (*Higher Priority*)
	PDOUT2 (*Lower Priority*)
GPIO1_4	PDIN2
GPIO1_5	PDIN1
GPIO1_6	PDOUT1

**Note that the MAX22512/MAX22522 enable the higher priority function in the table of each GPIO when multiple functions are selected/enabled in the GPIO1_ALT_FUNC register.*

Table 5. Bank 2 GPIO Alternate Functions

BANK 2 GPIO	ALTERNATE FUNCTION
GPIO2_0*	MCLK Input (*Highest Priority*)
	SPI Host – CS1 Output *(Mid Priority)*
	IRQ1 Input (*Lowest Priority*)
GPIO2_1	SPI Host – CS0 Output
GPIO2_2	SPI Host – SCLK Output
GPIO2_3	SPI Host – SDI Output

GPIO2_4	SPI Host – SDO Input

**Note that the MAX22512/MAX22522 enable the higher priority function in the table of each GPIO when multiple functions are selected/enabled in the GPIO2_ALT_FUNC register.*

Note that the MAX22512/MAX22522 enable the higher priority function of each GPIO, as indicated in the *Table 5*, when multiple functions are selected for a single GPIO. For example, if both PWM and PDOUT2 are enabled for GPIO1_3, GPIO1_3 operates as a pulse-width modulation (PWM) output. Alternate functions automatically set the direction and output value of a given GPIO.

Some alternate functions depend on other configurations and settings in other registers. For example, the MCLK function on GPIO2_0 is enabled and configured using the clock control register. This functionality also disables any other functions for this GPIO. See the ***Error! Reference source not found.*** section for more information.

Configuring the GPIO Push-Pull or Open-Drain Output

All GPIOs can be configured as inputs or open-drain or push-pull outputs, and feature pull-ups/pull-downs that can be enabled or disabled.

To operate the GPIOs as push-pull outputs, set the GPIOx_EN bit and set the output high or low by setting or clearing the associated GPIOx_OUT bit.

To operate any GPIO as an open-drain output, enable the internal pull-up/pull-down resistor for that GPIO using the GPIOx_PUDEN and GPIOx_PUDSEL registers. Toggle the pin between a high-impedance state and a known state by setting the GPIOx_EN and GPIOx_OUT bits appropriately. See *Table 6*.

Table 6. GPIOx Configuration

GPIOx_EN	GPIOx_PUD_DIS	GPIOx_UPDN_SEL	GPIOx_OUT	MODE	LOGIC STATE
0	0	0	X	Input or Output, OD	Low*
0	0	1	X	Input or Output, OD	High*
1	0	0	0	Output, PP	Low
			1	Output, PP	High

**This pin is pulled low or high by the enabled internal pull-down or pull-up resistor. To drive this pin as an open-drain output in a set state (not high impedance), set the GPIOx_EN and GPIOx_OUT bits.*

I²C Host Controller (GPIO1_1, GPIO1_2)

The MAX22512/MAX22522 feature an integrated I²C host controller using GPIO1_1 (SCL) and GPIO1_2 (SDA). I²C communication is configured and operated by setting registers in the I2C_HOST block.

I²C Host Registers

Configure the GPIO1_1 and GPIO1_2 for the I²C host controller functionality by setting the GPIO2_I2C bit in the GPIO1_ALT_FUNC register. *Table 7* shows the I²C host controller configuration registers.

Table 7. I²C Host Configuration Registers

I²C HOST CONFIGURATION REGISTER	REGISTER ADDRESS	ACCESS TYPE*	FUNCTION
I2C_CLK_PRESCALE	0x00	RW	Program SCL frequency

I2C_CTRL	0x04	RW	Enable I²C engine and interrupt, reset I²C engine and FIFOs
I2C_RX_FIFO	0x08	RO	Read/receive (RX) FIFO data
I2C_STATUS	0x0C	RO	I²C status flags
I2C_TX_FIFO	0x10	RW	Write/transmit (TX) FIFO data
I2C_CMD	0x14	RW	START, STOP, etc. commands
I2C_RX_NUM_BYTES	0x18	RW	Number of bytes to be received during read
I2C_CLR_RX_FIFO	0x1C	RW	Clear read FIFO
I2C_READ_ADD_BYTES	0x20	RW	Number of address bytes transmitted during a read

**RW = Read/write, RO = Read only*

SCL Clock Prescaler

Set the I2C_CLK_PRESCALE register to program the SCL clock frequency for I²C communication. The value written to the I2C_CLK_PRESCALE register provides a divider for the system clock frequency (f_{HCLK}).

Calculate the I2C_CLK_PRESCALE value using the following equation:

$$\text{I2C\_CLK\_PRESCALE}[7:0] = \frac{f_{HCLK}}{5 \times f_{SCL}} - 1$$

where, f_{HCLK} is the system clock frequency and f_{SCL} is the desired I²C clock frequency.

For example, using a system clock of 36MHz, set I2C_CLK_PRESCALE to 0x05 to generate a 1MHz I²C SCL frequency. The I2C_CLK_PRESCALER register is 16 bits wide.

Basic I²C Write

Use the following steps to enable and configure the I²C host controller for a basic I²C write:

1. Configure the Bank 1 GPIOs by setting the GPIO1_I2C bit in the GPIO1_ALT_FUNC register.
2. Program the I²C clock (f_{SCL}) by setting the I2C_CLK_PRESCALE register bits.
3. Set the I²C enable bit in the I2C_CTRL register (I2C_EN = 1) to enable the I²C host controller.

   Note that configuration registers need only be programmed once, before the first I²C communication.
4. Write the address of the I²C device, register address, and data to be transmitted to the device to the transmit FIFO register (TX_FIFO). Ensure the address and data do not exceed the FIFO depth: 32-bytes ≥ device address + register address + data.
5. Set the start and/or stop bits in the I2C_CMD register to start the I²C transmission. The TX FIFO is automatically cleared after the I²C write transmission is complete.

Ensure that data written to the I2C_TX_FIFO register does not exceed the 32-byte transmit FIFO. Data written to the I2C_TX_FIFO after the FIFO is filled corrupts the entire FIFO. Set I2C_GLB_SYNC_RST bit to 1 in the I2C_CTRL register to reset the TX_FIFO, RX_FIFO, and I²C host controller engine.

Bits in the I2C_CMD register are automatically cleared after the transmit/receive cycle is completed and must be reset for each write cycle.

The following is a basic I²C write example. This example presumes a system clock frequency of 36MHz (f_{HCLK} = 36MHz), an I²C clock frequency of 1MHz (f_{SCL} = 1MHz), an I²C device address of 0x01, and a device register address of 0x45.

1. Write 0x05 → I2C_CLK_PRESCALE register to set f_{SCL} = 1MHz with f_{HCLK} = 36MHz.
2. Write 0xC0 → I2C_CTR register to enable the I²C host controller (I2C_EN = 1) and to enable I²C interrupts (I2C_IRQ_EN = 1).
3. Write device address, register address, and data to the TX FIFO:
    a. Write 0x02 → I2C_TX_FIFO register. This is the device address + R/$\overline{W}$ bit.
    b. Write 0x45 → I2C_TX_FIFO register. This is the register address to write to on the device.
    c. Write the data to be written to the device to the TX FIFO register. *Ensure the address and data do not exceed the 32-byte FIFO depth.*
4. Write 0xD0 → I2C_CMD register to execute the write function. This command clears the IRQ_FLAG_STATUS bit (I2C_INT_ACK_CMD = 1), initiates transmission (I2C_START = 1), and automatically generates the required STOP condition (I2C_STOP) at the end of the transmission. The TX FIFO is automatically cleared after the I²C write transmission is complete.

Basic I²C Read

Use the following steps to enable and configure the I²C host controller for a basic I²C read:

6. Configure the Bank 1 GPIOs by setting the GPIO1_I2C bit in the GPIO1_ALT_FUNC register.
7. Program the I²C clock (f_{SCL}) by setting the I2C_CLK_PRESCALE register bits.
8. Set the I²C enable bit in the I2C_CTRL register (I2C_EN = 1) to enable the I²C host controller.

Note that configuration registers need only be programmed once, before the first I²C communication.

9. Write the address of the I²C device and the register address to be read to the I²C TX FIFO.
10. Write the number of address bytes transmitted during the read cycle to the I2C_RX_ADD_BYTES register.
11. Write the number of bytes to be read to the I2C_RX_NUM_BYTES register. *Ensure the number of expected bytes does not exceed the depth of the 32-byte I2C_RX_FIFO. Setting I2C_RX_NUM_BYTES results in a corrupted FIFO during the read transaction.*
12. Set the start and/or stop bits in the I2C_CMD register to start the I²C transmission. The TX FIFO is automatically cleared after the I²C write transmission is complete.
13. Read data from the I2C_RX_FIFO register. The I2C_RX_FIFO does not automatically clear after it is read. Set the I2C_CLR_RX_FIFO bit in the I2C_CLR_RX_FIFO register to clear the FIFO.

Ensure that data expected for the I2C_RX_FIFO register does not exceed the 32-byte depth of the receive (RX) FIFO. Data received to the I2C_RX_FIFO after the FIFO is filled is ignored.

Bits in the I2C_CMD register are automatically cleared after the transmit/receive cycle is completed and must be reset for each write/read cycle.

The following information is a basic I^2C read example. It presumes a system clock frequency of 36MHz (f_{HCLK} = 36MHz), an I^2C clock frequency of 1MHz (f_{SCL} = 1MHz), an I^2C device address of 0x01, and a device register address of 0x45.

1. Write 0x05 → I2C_CLK_PRESCALE register to set f_{SCL} = 1MHz with f_{HCLK} = 36MHz.
2. Write 0xC0 → I2C_CTR register to enable the I^2C host controller (I2C_EN = 1) and to enable I^2C interrupts (I2C_IRQ_EN = 1).
3. Write device address and register address to the TX FIFO:
    a. Write 0x02 → I2C_TX_FIFO register. This is the device address + R/W bit.
    b. Write 0x45 → I2C_TX_FIFO register. This is the register address to write to on the device.
    c. Write 0x03 → I2C_TX_FIFO register. This is the device address + R/W bit.
4. Write 0x02 → RX_ADD_BYTES register. This is the number of address bytes transmitted during the read operation.
5. Write the number of bytes to be read to the RX_BYTES register. Ensure the expected bytes do not exceed the 32-byte depth of the RX FIFO.
6. Write 0xD0 → I2C_CMD register to execute the write function. This command clears the IRQ_FLAG_STATUS bit (I2C_INT_ACK_CMD = 1), initiates transmission (I2C_START = 1), and automatically generates the required STOP condition (I2C_STOP) at the end of the transmit/receive cycle. The TX FIFO is automatically cleared after the I^2C write transmission is complete.
7. Read data from the I2C_RX_FIFO.
8. Write 0x01 → I2C_CLR_RX_FIFO register to clear the RX FIFO.

I²C Host Controller Status Indicators

The I²C host controller has five read-only status bits in the I2C_STATUS register to monitor the I²C bus. See *Table 8*.

Table 8. I²C Host Controller Status Bits (I2C_STATUS)

BIT	BIT NAME	FUNCTION
7	I2C_RX_ACK_STATUS	Receive ACK Status Indicator. This bit is set when an ACK is received from the device.
6	I2C_BUSY_STATUS	I²C Engine Busy Indicator. This bit is set when the I²C host controller is busy (transmitting or receiving).
5	I2C_ARBL_STATUS	I²C Communication Arbitration Lost Indicator. This bit is set when an arbitration lost for the current transaction occurs. This bit is reset when a new I²C transaction begins.
1	I2C_TIP_STATUS	I²C Transfer in Progress Indicator. This bit is set when a read/write transaction is in progress and is automatically cleared at the completion of the last data transaction.
0	I2C_IRQ_STATUS	I²C Interrupt Status Indicator. This bit is set when an I²C interrupt occurs (example, transaction done or arbitration lost).

The interrupt status bit (I2C_IRQ_STATUS) indicates that either (1) the last I²C transaction is complete, or (2) an arbitration lost error occurred during the last I²C transaction. An arbitration lost signal is generated when either:

- The I²C host controller attempts to write a 1 on the bus during an I²C write but the bus is not released, or
- An unexpected STOP condition is detected on the I²C bus.

The I²C communication is halted immediately and the I2C_IRQ_STATUS bit and the I2C_ARBL_STATUS bit in the I2C_STATUS register are set when either of these conditions occur (*Table 9*). The I2C_ARBL_STATUS bit is cleared when a new I²C transaction is started, but the IRQ_FLAG_STS bit remains set until cleared. Set the I2C_INT_ACK_CMD bit in the I2C_CMD register to clear the IRQ_FLAG_STS bit.

Table 9. I²C Host Controller Communication Lost/Complete

I2C_ARBL_STATUS BIT	I2C_IRQ_STATUS BIT	NOTE
0	0	No I²C transaction occurred or last I²C transaction completed without error, and I2C_IRQ_STATUS bit is cleared.
0	1	Last I²C transaction completed without error.
1	1	Last I²C transaction ended with arbitration error.

SPI Host Controller (GPIO2_0 to GPIO2_4)

The MAX22512/MAX22522 GPIOs can be configured to operate as an SPI host. Optimized for software-minimal transactions, the SPI host can be used for programming external FLASH or otherwise transmitting large datagrams with limited software interaction. *Table 10* shows the setup and configuration registers used for SPI host communication.

Table 10. SPI Host Status and Configuration Register

REGISTER NAME	REGISTER ADDRESS	ACCESS TYPE*	DESCRIPTION
SPI1_STATUS	0x00	RO, WCA	SPI Status flags
SPI1_GEN_CONFIG	0x04	RW	SPI Configuration select
SPI1_SCLK_CONFIG	0x08	RW	SPI Clock and chip-select (CS) timing
SPI1_TRANSFER_CONFIG	0x10	RW	CS and SCLK configuration
SPI1_RD_BUFF_0	0x2C	RW	SPI Read buffer 0
SPI1_RD_BUFF_1	0x30	RW	SPI Read buffer 1
SPI1_RD_BUFF_2	0x34	RW	SPI Read buffer 2
SPI1_RD_BUFF_3	0x38	RW	SPI Read buffer 3
SPI1_WR_BUFF_0	0x3C	RW	SPI Write buffer 0
SPI1_WR_BUFF_1	0x40	RW	SPI Write buffer 1
SPI1_WR_BUFF_2	0x44	RW	SPI Write buffer 2
SPI1_WR_BUFF_3	0x48	RW	SPI Write buffer 3

**RO = Read only, WCA = Write clears all, RW = Read/write*

The SPI1_RD_BUFF and SPI1_WR_BUFF buffers are 16-byte FIFOs to store, transmit, and receive data for the SPI bus. The SDI data for an SPI transaction is taken from the write buffer, while the SDO data is stored in the read buffer.

The write buffers (SPI1_WR_BUFF_x) are not automatically cleared after a transaction. This is necessary for cyclic reading of same address in an SPI device.

Configuring the SPI for Communication

Set the SPI configuration and enable the interface by programming the SPI1_GEN_CONFIG register. Bits in the SPI1_GEN_CONFIG register are latched for one transaction and should be updated/rewritten before the next transaction begins.

Defining and Configuring a Datagram

Define a datagram for transmission by setting the SPI1_TRANSFER_CONFIG and SPI1_SCLK_CONFIG registers.

The SPI1_TRANSFER_CONFIG register describes the format of the datagram. Select the switching clock phase and polarity, CS polarity, and transfer mode in this register.

The SPI1_TRANSFER_CONFIG[22:16] and SPI1_TRANSFER_CONFIG[14:8] define the length of the first and second datagrams. Values for the datagram length can be calculated as (bits + 1). The longest datagram is 128 bits.

The SYNC_CONFIG register is used to define the configuration of the trigger for an SPI transaction. The SPI master can be configured to automatically send datagrams on hardware (HW) strobe signals, allowing the Arm Cortex-M0 to run the SPI while performing other time-sensitive or demanding tasks. In this configuration, the SPI interrupt is set when SPI communication is complete, notifying the microcontroller that all datagrams are transferred. Set the STROBE_MASK bits when using a HW strobe so that only correctly masked events trigger a transaction. Alternately, it is possible to trigger an SPI transaction with a software (SW) strobe. Set the SW_STROBE bit to immediately begin a transaction after writing to the SPI1_WR_BUFF_0 register.

Timers and System Watchdog

The MAX22512/MAX22522 feature a system watchdog timer (WDT), a standard Arm Cortex-M0 SysTick timer, and an additional enhanced timer used to count external events or generate a PWM/TOGGLE output signal (single-cycle or continuous) based on the 18MHz raw oscillator.

System Watchdog Timer (WDT)

The system watchdog (WDT) timer resets the Arm Cortex-M0 core when watchdog clear events occur too often or not often enough. Configure the WDT timer and functionality by programming the registers shown in *Table 11*.

Table 11. System Watchdog Configuration Registers

REGISTER NAME	REGISTER ADDRESS	ACCESS TYPE*	FUNCTION
WDT_MAX	0x00	RW	Set the maximum counter for the WDT time. The Arm Cortex-M0 is reset when the watchdog counter reaches this value. By default, only a maximum watchdog time is enabled.
WDT_ENABLE	0x04	RW	Enable the WDT.
WDT_LOCK	0x08	RW	Lock the system watchdog settings. Any further changes to the WDT_MIN, WDT_MAX, WDT_LOCK, and WDT_CONFIG registers are ignored.
WDT_CLEAR	0x0C	RW	Write 0x01 to this register to generate a clear event. The watchdog counter is reset when a 1 is written to the WDT_CLEAR bit in this register.
WDT_MIN	0x10	RW	Set and enable a minimum WDT time between clear events. When enabled, the Arm Cortex-M0 is reset when a watchdog clear event occurs before the watchdog counter reaches this value. This functionality is disabled by default.
WDT_STATUS	0x14	RW	Indicates when a watchdog event occurred in the Arm Cortex-M0 core.

**RW = Read/write, RO = Read only*

Enable the system WDT by setting the WDT_EN bit in the WDT_ENABLE register to 1. The WDT is enabled and counts-up when WDT_EN = 1. When WDT_EN is 0, the watchdog timer is disabled, the WDT counter is reset to 0, and the counter stops counting.

Set CLEAR = 1 in the WTD_CLEAR register to execute a clear event. If WDT_EN = 1 and CLEAR = 1, the watchdog timer restarts from 0.

The WDT_MAX register defines the maximum time frame allowed to pass between two consecutive WDT clear events. The maximum time allowed between clear events can be calculated as:

$$t_{WD\_MAX} = 30.52\mu s \times \text{WDT\_MAX value}$$

The WDT_MIN register defines the minimum time frame that must pass before a watchdog clear event is allowed to take place. When enabled, the minimum time required between clear events can be calculated as:

$$t_{WD\_MIN} = 30.52\mu s \times \text{WDT\_MIN value}$$

Ensure that WDT_MAX is larger than WDT_MIN.

The WDT_RESET bit in the WDT_STATUS register stores information about the last WDT timeout event. The WDT_RESET bit is set when the WDT counter reaches or exceeds the value in the WDT_MAX register, or if a clear event occurs before the counter exceeds the value in the WDT_MIN register. When WDT_RESET = 1, an interrupt is generated (WD_INT = 1), and a reset request is sent to the Arm Cortex-M0. Once the reset event is completed, the Arm Cortex-M0 can read the WDT_RESET bit to clarify the source of the last reset event.

When the WDT enable bit is set (WDT_EN = 1), the WDT_MAX and WDT_MIN registers are not locked until a reset event occurs, or the WDT is disabled (WDT_EN = 0). The WDT can be disabled if WDT_EN = 1 and the WDT lock bit in the WDT_LOCK register is clear (LOCK = 0). If the WDT is disabled (WDT_EN=0), it can always be enabled, regardless of the state of the LOCK bit.

If the WDT lock bit is clear (LOCK = 0), the WDT configuration registers (WDT_MIN, WDT_MAX, WDT_LOCK) can be updated by disabling the WDT (WDT_EN = 0).

If the WD timer lock bit is set (WDT_LOCK = 0x01), the WDT configuration registers cannot be written until the next reset event occurred. The WDT is locked when LOCK = 1, and only the WDT_EN bit can be toggled. See *Table 12*.

Table 12. Watchdog Timer Enable and Lock Functions

WDT_EN BIT	WDT_LOCK BIT	AVAILABLE ACTIONS
0	0	Update WDT configuration (WDT_MIN, WDT_MAX, WDT_ENABLE, WDT_LOCK). Lock WDT configuration (WDT_LOCK). Enable/disable WDT (WDT_ENABLE).
0	1	WDT can only be enabled (WDT_EN = 1).
1	0	WDT can only be disabled to stop the counter and reset the value to 0 (WDT_EN = 0).
1	1	The WDT configuration registers cannot be written until the next reset event occurred.

Figure 5 shows the suggested configuration and service routine for the WDT.

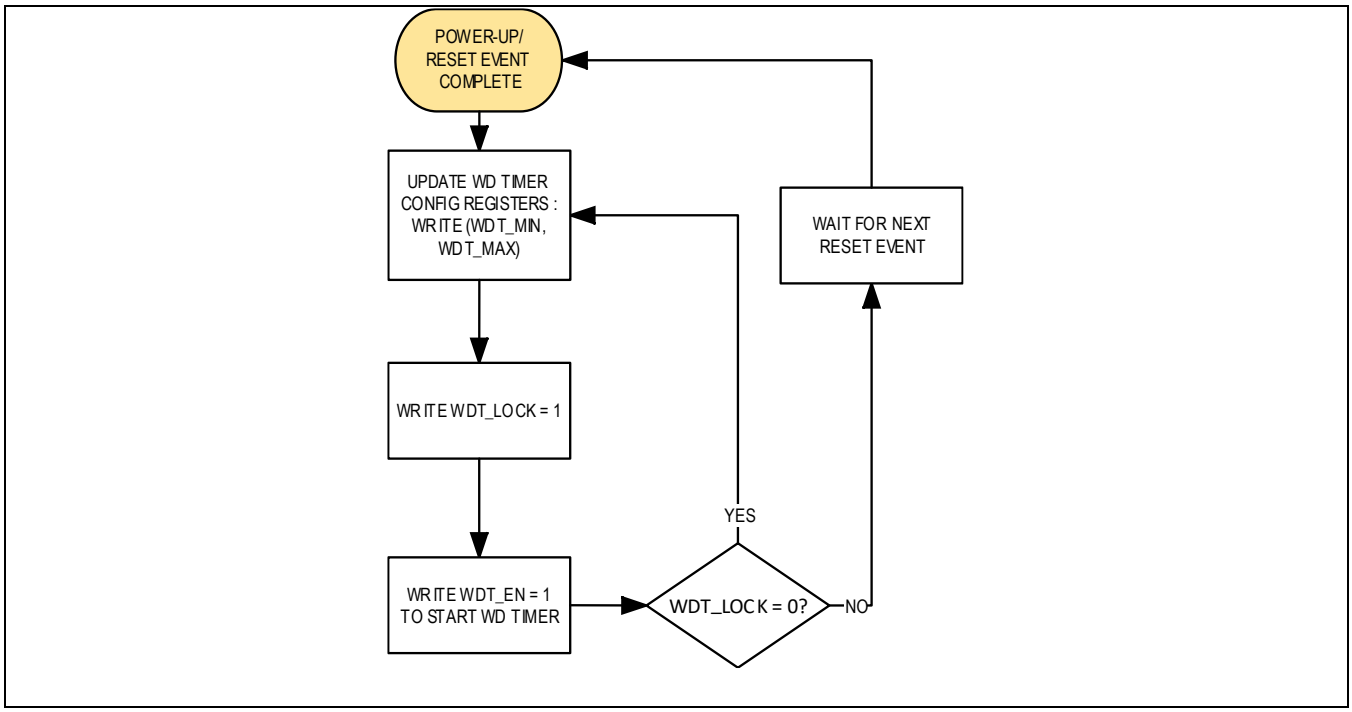


Figure 5. WD Timer Configuration Routine Flowchart

SysTick Timer

The SysTick timer features a 24-bit, down-counter with autoreload capability and a maskable system interrupt when the counter reaches 0. Refer to the Arm Cortex-M0 developer documentation on the Arm website for more information.

Enhanced Timer

The Enhanced Timer is a programmable 32-bit autoreload up-counter that can be used for many different functions including a generic timer for software interrupt generation, PWM signal generation, or a toggle output. GPIO1_3 is the output of the enhanced timer. Set GPIO1_PWM in the GPIO1_ALT_FUNC register to 1 to enable GPIO1_3 as the timer output.

The Enhanced Timer can be enabled and configured using the registers in *Table 13*.

Table 13. Enhanced Timer Status and Configuration Registers

REGISTER NAME	REGISTER ADDRESS	ACCESS TYPE*	DESCRIPTION
ENH_TIMER_COUNTER	0x00	RO	Enhanced Timer Counter Value
ENH_TIMER_LIMIT	0x04	RW	Enhanced Timer Counter Overflow Limit
ENH_TIMER_COMPARE	0x08	RW	Enhanced Timer Counter PWM Output Compare Value
ENH_TIMER_CONFIG	0x0C	RW	Enhanced Timer Configuration Register
ENH_TIMER_IRQ_CTRL	0x10	RW	Enhanced Timer Interrupt Enable Control Register
ENH_TIMER_STATUS	0x14	COW1	Enhanced Timer Status Register and Interrupt Flags
ENH_TIMER_EXT_COUNTER	0x18	RO	External Event Counter Value

ENH_TIMER_EXT_THRES	0x1C	RW	External Event Counter Threshold Value
ENH_TIMER_EXT_CONFIG	0x20	RW	External Event Counter Configuration Register
ENH_TIMER_OUT_CONFIG	0x24	RW	External Trigger Event Select Register

**RO = Read only, RW = Read/write, COW = Clear on write-one*

Enhanced Timer Configuration Registers

Configure the enhanced timer by setting bits in the ENH_TIMER_CONFIG register (*Table 14*). The enhanced timer can be configured as a PWM output (PWM_EN = 1 in the ENH_TIMER_CONFIG register) or as a toggle output (TOGGLE = 1 in the ENH_TIMER_CONFIG register).

As shown in *Table 14*, both the PWM and TOGGLE outputs can be triggered by an external event. Set the PWM_TRG_GPIO_EN bit to 1 to enable the enhanced timer output to be triggered by a GPIO input. Select the external trigger (GPIO) and the rising, falling, or both edges for the trigger using the ENH_TIMER_OUT_CONFIG register.

Do not set TOGGLE and PWM_EN at the same time. The enhanced timer output is disabled when this occurs.

Table 14. Enhanced Timer Output Configuration (TOGGLE and PWM)

ENH_TIMER_CONFIG REGISTER			ENH_TIMER_OUT_CONFIG REGISTER	ENHANCED TIMER COUNTER BEHAVIOR	TIMER OUTPUT FUNCTION
TOGGLE BIT	PWM_EN BIT	ENABLE_CYCLE BIT	PWM_TRG_GPIO_EN BIT		
0	0	0	0	Disabled	Not active
0	1	0	0	Single-Cycle Counting Enabled	Single-Cycle PWM Output
1	0	0	0		Single-Cycle TOGGLE Output
1	1	0	0		Not active
0	0	1	0	Cyclic Counting Enabled	Not active
0	1	1	0		Continuous PWM Output
1	0	1	0		Continuous TOGGLE Output
1	1	1	0		Not active
0	0	-	1	Single-Cycle Counter Triggered by an External Event	Not active
0	1	-	1		PWM output is trigged by the GPIO input signal selected by PWM_TRG_GPIO_SEL.
1	0	-	1		TOGGLE output is trigged by the GPIO input signal selected by PWM_TRG_GPIO_SEL.
1	1	-	1		Not active

PWM Output

When configured for PWM operation (PWM_EN = 1), the enhanced timer outputs a PWM signal with a duty cycle defined as the ratio between the value in the ENH_TIMER_COMPARE register and the value in the ENH_TIMER_LIMIT register (*Figure 6*).

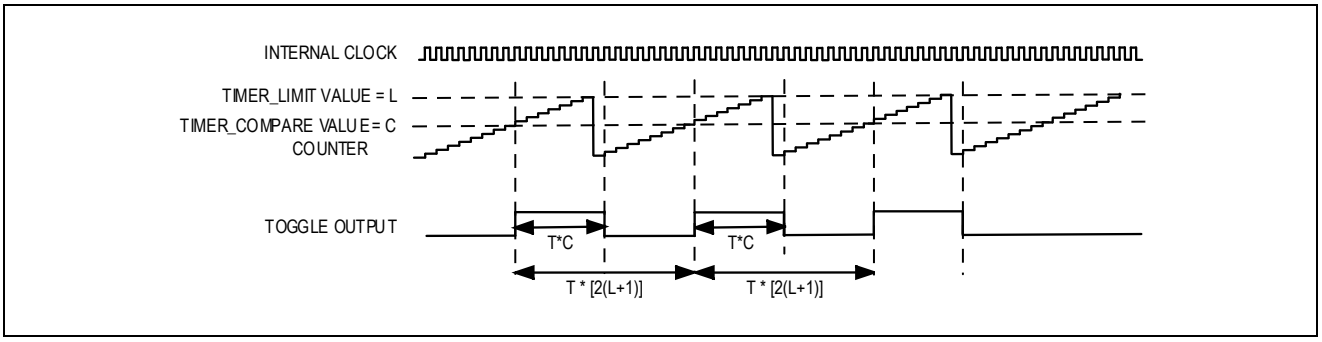


Figure 6. Enhanced Timer Output, PWM Mode, TIMER_LIMIT = L, TIMER_COMPARE = C, TIMER_CONFIG = 0x03

Calculate the frequency for a continuous PWM cycle using the following equation:

$$f_{PWM}(Hz) = \frac{1}{T \times (L + 1)}$$

where T is the period of the clock and L is the TIMER_LIMIT register value.

The duty cycle for the PWM output signal is:

$$D_{PWM}(\%) = \frac{C}{(L + 1)} \times 100$$

where C is the ENH_TIMER_COMPARE register value and L is the ENH_TIMER_LIMIT register value.

TOGGLE

Set the TOGGLE bit in the TIMER_CONFIG register (TOGGLE = 1) to configure the enhanced timer to output a single or multicycle toggle. In this mode, the timer output togglers (low-to-high or high-to-low) counter value (ENH_TIMER_COUNTER) equals the overflow limit defined by the ENH_TIMER_LIMIT register (*Figure 7*).

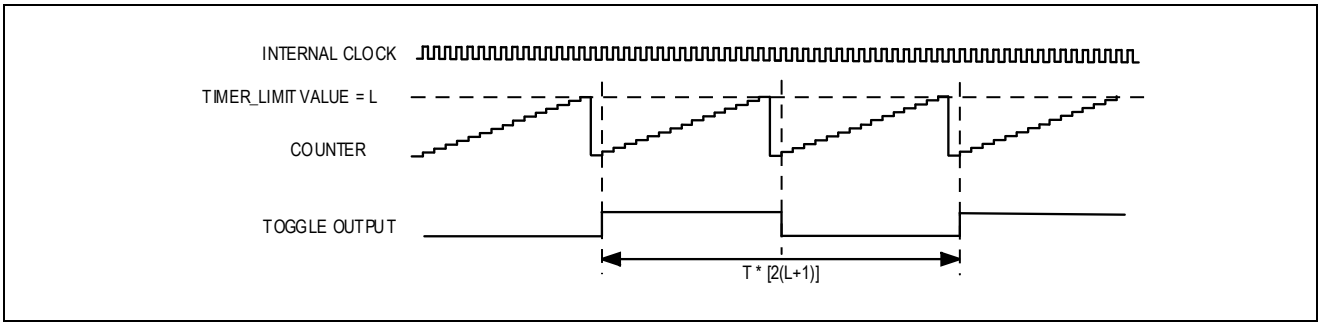


Figure 7. Enhanced Timer Output, TOGGLE Mode, TIMER_LIMIT = L, TIMER_CONFIG = 0x05

If ENABLE_CYCLE = 1, the counter is reset to 0 when ENH_TIMER_COUNTER = ENH_TIMER_LIMIT and begins to count up again. Calculate the frequency of a continuous TOGGLE output signal as:

$$f_{TOGGLE}(Hz) = \frac{1}{T \times 2(L + 1)}$$

where, T is the period of the internal clock (that is, for a 36MHz clock, T = 29.1ns) and L is the ENH_TIMER_LIMIT register value.

If ENABLE_CYCLE = 0 in the ENH_TIMER_CONFIG register, the counter is enabled for a single cycle only, and there is no restart after the overflow event occurs.

External Event Counter

The MAX22512/MAX22522 enhanced timer includes an external event counter. The external event counter is a 16-bit autoreload counter triggered by a GPIO. The enhanced timer receives external events from GPIOs through multiple input ports. Set the EXT_TRG_GPIO_SEL bits in the ENH_TIMER_EXT_CONFIG to define the external event input port selection (*Table 15*). Note that GPIO selection values are the same for EXT_TRG_GPIO_SEL and PWM_TRG_GPIO_SEL bits.

Table 15. PWM and External Event Counter GPIO Input Select

PWM_TRG_GPIO_SEL[3:0] EXT_TRG_GPIO_SEL[3:0]	GPIO INPUT PORT
0000	GPIO1_0
0001	GPIO1_1
0010	GPIO1_2
0011	GPIO1_3
0100	GPIO1_4
0101	GPIO1_5
0110	GPIO1_6
0111	GPIO2_1
1000	GPIO2_2
1001	GPIO2_3
1010	GPIO2_4

The external event counter is always enabled and is incremented on rising, falling, or both edges of selected GPIO inputs as programmed in the ENH_TIMER_EXT_CONFIG register. Input pulses must be at least two clock cycles long to be detected for the external event counter.

The counter limit value is 65,535 and it is not programmable. When the overflow limit value is reached, the EXT_OVR_FLAG bit in the ENH_TIMER_EXT_THRES register is set and the counter is reset to 0. Set the counter programmable threshold value in the ENH_TIMER_EXT_THRES register. When the counter value equals the threshold, the EXT_THRES_IRQ interrupt bit in the ENH_TIMER_STATUS register is set. Write a 1 to the EXT_THRES_IRQ bit to clear it.

Timer Status and Interrupts

The ENH_TIMER_STATUS register provides five different clear-on-write-one interrupt bits. *Table 16* shows the available enhanced timer interrupt bits in the ENH_TIMER_STATUS register. Enable these interrupts by setting the associated bits in the ENH_TIMER_IRQ_CTRL register.

Table 16. Enhanced Timer Interrupt Bits

TIMER_STATUS BITS	NAME	FUNCTION

[32:5]	—	Reserved
[4]	EXT_OVR_FLAG	This bit is set when TIME_EXT_COUNTER value = 65,535. The count is reset to 0x00000000 and is restarted after this interrupt occurs.
[3]	EXT_THRES_IRQ	This bit is set when TIMER_EXT_COUNTER value = TIMER_EXT_THRES value. The counter continues counting up after this interrupt occurs.
[2]	STOP_FLAG	This bit is set when the enhanced timer counter is stopped at 0 (example, after a single cycle when ENABLE_CYCLE = 0) or due to a configuration change.
[1]	COMP_MATCH_FLAG	This bit is set when the TIMER_COUNTER value = TIMER_COMPARE value.
[0]	OVER_FLAG	This bit is set when the TIMER_COUNTER value = TIMER_LIMIT value. The TIMER_COUNTER value is reset to 0x00000000 to restart (if ENABLE_CYLE = 1 in the TIMER_CONFIG register) after this interrupt occurs.

Clock Control

The MAX22512/MAX22522 feature extensive clock control and configuration (*Figure 8*). After a system reset, the device powers up using the 18MHz raw oscillator. The bootloader program from the ROM is executed at this frequency. After initialization, program the clock-control configuration as part of the application program.

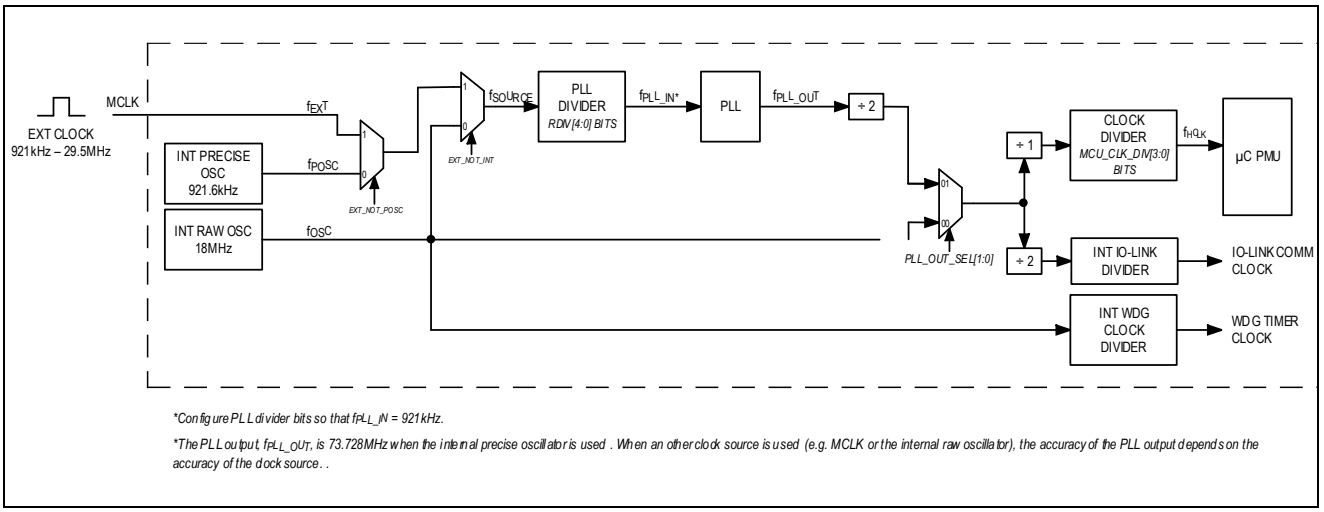


Figure 8. Clock Structure

Phase-Locked Loop (PLL)

The internal PLL is used as the clock source for the internal digital blocks. The PLL accepts a 921.6kHz input clock source (f_{PLL_IN}) and outputs a clock frequency (f_{PLL_OUT}) that is divided by two.

Once the power-up sequence is complete, the PLL can be latched to one of three selectable clocks: an external clock (f_{EXT}), the 18MHz raw oscillator, or the 921.6kHz internal precise oscillator. See *Table 17*.

Table 17. PLL Input Select

EXT_NOT_POSC	EXT_NOT_INT	SELECTED PLL INPUT	NOTES
X	0	f_{OSC}	18MHz raw oscillator. This is the default clock source and PLL source configuration.
0	1	f_{POSC}	921.6kHz precise internal oscillator
1	1	f_{EXT}	External clock input. Configure GPIO2_0 as MCLK input and connect external clock from 921kHz to 29.5MHz to GPIO2_0 as the clock source.

Set the PLL divider (from 1 to 32) by programming the RDIV[4:0] bits. Program the divider to set the PLL input clock, f_{PLL_IN}, to 921.6kHz (typ). If the PLL input clock frequency (fPLL_IN) does not equal 921.6kHz ±5%, the MAX22512/MAX22522 automatically switch to the 18MHz raw oscillator as the clock source.

Clock Configuration Examples

The clock and PLL must be configured as soon as the device is powered up. To configure the PLL output as the digital clock source, use the following steps:

1. Set the clock control enable bit in the CLK_CTRL_OPT register.
2. Configure the PLL input selection by setting the EXT_NOT_POSC and EXT_NOT_INT bits in the CLK_CTRL_SOURCE register.
3. Configure the PLL output as the digital clock source by setting the PLL_OUT_SEL bits in the CLK_CTRL_PLL_CFG register.
4. Set the clock input divider in the CLK_CTRL_PLL_CFG register.
5. Commit the clock configuration changes by setting the COMMIT bit in the CLK_CTRL_PLL_CFG register.
6. Poll the CLK_CTRL_PLL_CFG register until COMMIT = 0, to verify the clock configuration is complete.

Table 18 to *Table 21* show the register and bit settings for specific examples using the internal oscillators or an external clock source.

Table 18. PLL Output Sourced by 921.6kHz Internal Precision Oscillator

REGISTER	BIT(S)	SETTING
CLK_CTRL_OPT	CLK_CTRL_EN	1
CLK_CTRL_SOURCE	EXT_NOT_POSC	0
	EXT_NOT_INT	1
CLK_CTRL_PLL_CFG	PLL_OUT_SEL[1:0]	01
	RDIV[4:0]	00000
	COMMIT	1

Table 19. PLL Output Sourced by 18MHz Internal Raw Oscillator (*)

REGISTER	BIT(S)	SETTING
CLK_CTRL_OPT	CLK_CTRL_EN	1
CLK_CTRL_SOURCE	EXT_NOT_POSC	0
	EXT_NOT_INT	0
CLK_CTRL_PLL_CFG	PLL_OUT_SEL[1:0]	01

	RDIV[4:0]**	10011
	COMMIT	1

Clock accuracy for this configuration does not meet IO-Link requirements. This configuration is not recommended for IO-Link communication.

***The RDIV bits must be set to ensure a 921.6kHz clock is delivered to the PLL. RDIV is calculated as RDIV = (f_{SOURCE}/921kHz) - 1.*

Table 20. PLL Output Sourced by External Clock Source

REGISTER	BIT(S)	SETTING
CLK_CTRL_OPT	CLK_CTRL_EN	1
CLK_CTRL_SOURCE	EXT_NOT_POSC	1
	EXT_NOT_INT	1
CLK_CTRL_PLL_CFG	PLL_OUT_SEL[1:0]	01
	RDIV[4:0]*	See Note (*)
	COMMIT	1

**The RDIV bits must be set to ensure a 921.6kHz clock is delivered to the PLL. RDIV is calculated as RDIV = (f_{SOURCE}/921kHz) - 1.*

Table 21. PLL Bypassed, 18MHz Internal Oscillator (*)

REGISTER	BIT(S)	SETTING
CLK_CTRL_OPT	CLK_CTRL_EN	1
CLK_CTRL_SOURCE	EXT_NOT_POSC	Don’t care
	EXT_NOT_INT	Don’t care
CLK_CTRL_PLL_CFG	PLL_OUT_SEL[1:0]	00
	RDIV[4:0]*	Don’t care
	COMMIT	1

**Clock accuracy for this configuration does not meet IO-Link requirements. This configuration is not recommended for IO-Link communication.*

Clock Control Status Indicators

After power-up, configure the clock and verify it is stable and operating before starting the application. Ready indicators and fault bits are located in the CLK_CTRL_STATUS register. Bits in the CLK_CTRL_STATUS register are not latched and change as internal clock sources are setup and become stable.

Clock and PLL events are also recorded in the CLK_CTRL_LATCH_RE, and CLK_CTRL_LATCH_FE registers. Bits in these registers are latched when set and must be manually cleared.

Enable associated interrupts for events related to the CLK_CTRL_STATUS register (and CLK_CTRL_LATCH_xx registers) in the CLK_CTRL_INT_EN_RE and CLK_CTRL_INT_EN_FE registers. Interrupts enabled in the CLK_CTRL_INT_EN_RE register generate an interrupt when the associated bit in the CLK_CTRL_STATUS register and/or CLK_CTRL_LATCH_xx register is set. Interrupts enabled in the CLK_CTRL_INT_EN_FE register generate an interrupt when the associated bit in the CLK_CTRL_STATUS register is cleared.

Interrupt Controller

The MAX22512/MAX22522 implements a standard Arm Cortex-M0 interrupt controller. Each peripheral includes an associated interrupt enable that leads to an interrupt line. *Table 22* shows the interrupt mapping for the MAX22512/MAX22522 peripherals.

Table 22. Peripheral Interrupts

IRQ_LINE_x	PERIPHERAL	ATTACHED INTERRUPTS TO THE LINE
0	GPIO1 – IRQ0	External Interrupt
1	GPIO2 – IRQ1	External Interrupt
2	ADC	ADC_READY_INT, ADC_TRF_INT, ADC_CONV_INT
3	SPI Host	TRANSACT_DONE
4	I^2C Host	IRQ_FLAG_SET, AL_FLAG_STS
6	IO-LINK	WD_INT, PDOUT_DATRX_INT, CLR_EVENT_FLG_INT, WU_INT, MCMD_INT, DIR_PAGE1_INT, DL_MODE_INT, FRM_ERR_CNT_INT, IOL_ERR_CNT_INT, CQ_FAULT_INT, V24_ERR_INT, V24_WRN_INT, THM_WRN_INT, THM_SHD_INT, ISDU_ABRT_INT, ISDU_BAV_INT, CHKPDU_ERR_INT, NEW_ISDU_WR_START_INT, I ISDU_ABRT_INT, ISDU_IDLE_INT, ISDU_PCKT_INT, MC_BYTE_RCV_INT, CKS_SENT_INT
7	CLOCK CONTROL	POSC_RDY_FE, XTAL_RDY_RE, POSC_RDY_TMO, CLK_OK, CLKT_FLT, CLK_FLT_TMO, PLL_STUCK, PLL_LOCK_FE, PLL_ERR, CLK_FAIL, INT_FSM
8	ENHANCED TIMER	EXT_OVR_INT, EXT_THRESH_INT, STOP_INT, COMP_MATCH_INT, OVER_INT
9	DIGIPOT	COMPO1_INT, COMPO2_INT

When an interrupt event occurs, an internal interrupt circuit sets the associated interrupt bit (x_INT), which remains set until the bit is cleared or the Arm Cortex-M0 is reset. All interrupt bits are cleared by writing a 1 to the bit. Refer to the Arm Cortex-M0 developer documentation on the Arm website for more information on the interrupt line.

Most interrupts include an associated enable or mask bit that control when an interrupt event is reported to the Arm Cortex-M0 for interrupt handling. When enabled, an interrupt generates an interrupt signal on the associated IRQ_LINE to the Arm Cortex-M0. When the interrupt is not enabled, the associated interrupt bit is set, following an interrupt event, but no interrupt signal is sent to the IRQ_LINE. *Figure 9* shows the interrupt circuit for the enhanced timer peripheral, as an example.

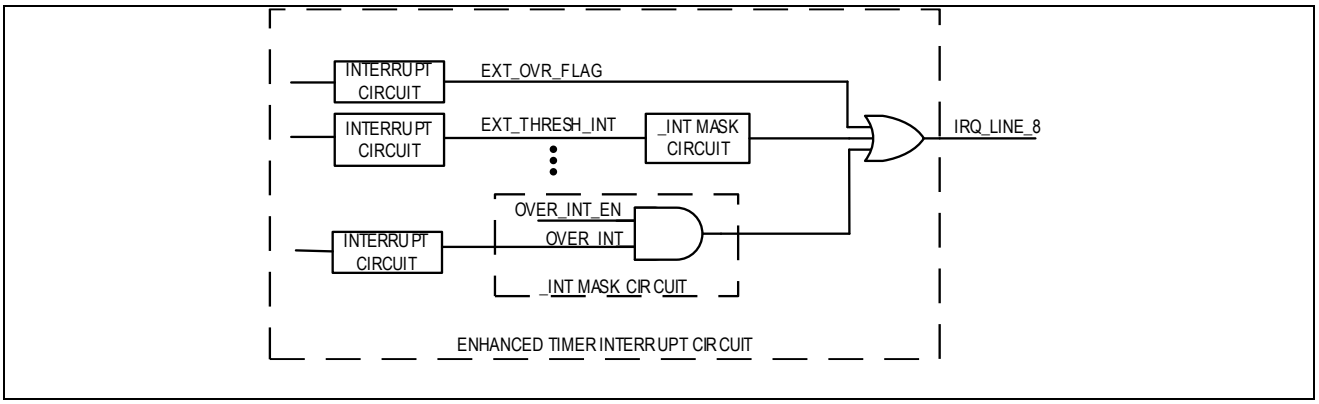


Figure 9. Enhanced Timer Interrupt Circuit

Each interrupt line (IRQ_LINE_x) is passed to the Arm Cortex-M0 through the interrupt handler (*Figure 10*).

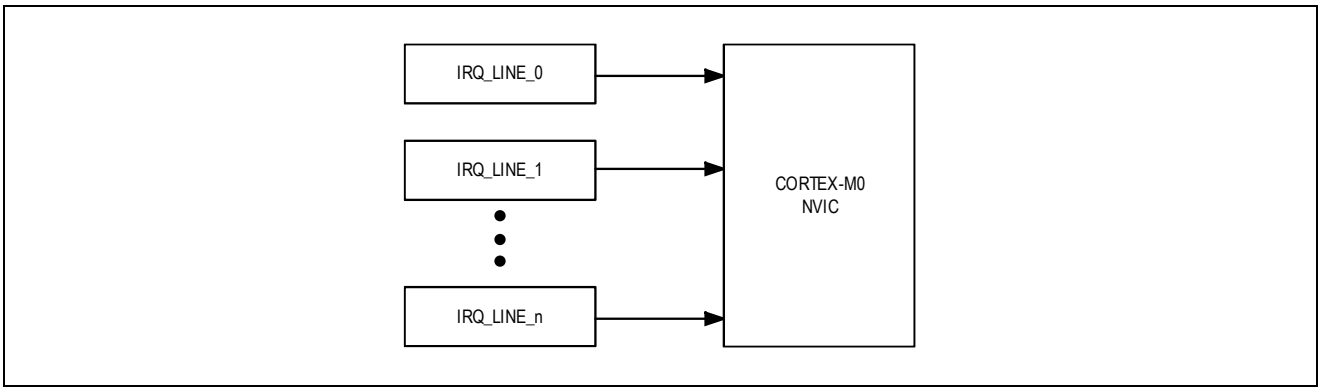


Figure 10. Peripheral Interrupt Lines are All Aent to Arm Cortex-M0 NVIC

See the ***Error! Reference source not found.*** section for more information about individual peripheral interrupts.

IO-Link Data Link Layer

The MAX22512/MAX22522 integrate an IO-Link device data link layer required for IO-Link communication. Internal state machines are capable of handling both cyclic and acyclic data transmission types, as outlined in the IO-Link standard version 1.1.4.

IO-Link Configuration and Registers

After the device has powered up and is stabilized, configure the MAX22512/MAX22522 for device operation by configuring the registers shown in *Table 23*.

Table 23. IO-Link and C/Q Configuration Registers

REGISTER BLOCK	START ADDRESS	OFFSET	REGISTER NAME	DESCRIPTION
IO-LINK ID REGISTER	0x4004 0000	0x00	IOL_REV_ID	Revision ID of the MAX22512/MAX22522
IO-LINK STATUS REGISTERS	0x4004 0008	0x00	IOL_STAT	IO-Link mode
		0x04	IOL_DEV_STAT1	Device Status 1
		0x08	IOL_ DEV_STAT2	Device Status 2: V_{24}, C/Q and Thermal warning

		0x0C	IOL_ISDU_STAT	ISDU buffer activity status
		0x10	IOL_ERR_CNT	IO-Link communication error counter
		0x14	IOL_FRM_ERR_CNT	IO-Link communication frame parity error counter
IO-LINK INTERRUPT REGISTERS	0x4004 0020	0x00	IOL_INT	IO-Link communication interrupts
		0x04	IOL_DEV_INT	IO-Link communication, V_{24}, and thermal interrupts
		0x08	IOL_ISDU_INT	ISDU interrupts
		0x0C	IOL_HEART_INT	IO-Link message send/receive interrupts
IO-LINK INTERRUPT ENABLE REGISTERS	0x4004 0038	0x00	IOL_INT_EN	IO-Link communication interrupt enable bits
		0x04	IOL_DEV_INT_EN	IO-Link communication, V_{24}, and thermal interrupt enable bits
		0x08	IOL_ISDU_INT_EN	ISDU interrupt enable bits
		0x0C	IOL_HEART_INT_EN	IO-Link message send/receive interrupt enable bits
IO-LINK COMMUNCATION CONFIGURATION REGISTERS	0x4004 0050	0x00	IOL_CFG	IO-Link COM rate, device delay, SIO mode, and configuration done
		0x04	IOL_WDG_TMR	IO-Link cycle watchdog counter threshold
		0x08	IOL_WDG_CLR	Clear IO-Link errors and configure the IO-Link watchdog clear process
		0x0C	IOL_MISC_CFG	IO-Link interpacket delay and/or reset the IO-Link registers
IO-LINK DIRECT PAGE 1 REGISTERS	0x4004 0068	0x00 – 0x3C	PAGE1_BYTE00 – PAGE1_BYTE0F	IO-Link direct page 1 parameters
IO-LINK EVENT REGISTERS	0x4004 00A8	0x00	IOL_WDG_EVENT	IO-Link watchdog event code
		0x04	IOL_STATUS_CODE_DEF	IO-Link device default event code
		0x08	IOL_STATUS_CODE	IO-Link device status code
		0x0C	IOL_EVENT_QUALIFIER	IO-Link event code qualifier
		0x10	IOL_EVENT_CODEMSB	IO-Link event code MS
		0x14	IOL_EVENT_CODELSB	IO-Link event code LSB
		0x18	IOL_EVENT_FLAG	IO-Link event when the IO-Link watchdog is triggered and IO-Link event flag bit
IO-LINK BUFFER CONTROL REGISTERS	0x4004 00D4	0x00	IOL_PDIN_LOCK	IO-Link PDIn buffer lock/unlock and process data valid/invalid status bit
		0x04	IOL_PDOUT_LOCK	IO-Link PDOut buffer lock/unlock
		0x08	IOL_IDSU_DATA_RDY	IO-Link ISDU buffer lock/unlock and ISDU data ready bit
		0x0C	IOL_ISDU_LEVEL	IO-Link ISDU FIFO fill level
IO-LINK LED CONTROL REGISTERS	0x4004 0140	0x00 – 0x0C	IOL_LED1CTRMSB, IOL_LED1CTRLSB,	LED1 and LED2 output sequence bits

			IOL_LED2CTRMSB, IOL_LED2CTRLSB	
IO-LINK TRANSCEIVER CONTROL REGISTERS	0x4004 015nno8	0x00	IOL_CQ_CTRL1	Configure and enable the C/Q driver
		0x04	IOL_CQ_CTRL2	Configure and enable the C/Q driver
		0x08	IOL_TX_CTRL	C/Q control inputs and GPIO to PDIn invert logic enable
		0x0C	IOL_RX_CTRL	C/Q receiver filter enable
		0x10	IOL_MISC_CTRL	C/Q 2mA pull-up/pull-down enable
CQ_CFG REGISTER	0x4007 0048	0x00	CQ_CFG_REG	Set this register to 0x04 to finalize the C/q configuration.

When the register map is programmed, set CONF_DONE = 1 in the IOL_CFG register to allow the device to proceed through the establish-communication sequence when a valid wake-up pulse is detected. The MAX22512/MAX22522 ignore incoming wake-up pulses and remains in SIO mode while the CONF_DONE = 0.

SIO Mode

The MAX22512/MAX22522 are configured to operate in SIO mode when powered up, and/or after a hardware or software reset. In SIO mode, the C/Q driver is controlled by the CQ_TXEN and CQ_TX bits in the IOL_TX_CTRL register. Note that, when CMP_TO_SIO_TX in the CMP_TO_PDIN_REG register is set, the CQ_TX bit is ignored and the logic of the C/Q driver is set by the CMP1 output (CMPO1). In SIO mode, C/Q is configurable for negative-positive-negative (NPN), positive-negative-positive (PNP), or push-pull operation, and features a programmable current limit and slew rate.

Wake-Up and Establish COM

The wake-up sequence is the prerequisite for placing an IO-Link device in preoperate and operate modes. A wake-up condition is detected when the IO-Link master shorts the C/Q line to the opposite logic polarity for 80µs (typ). Enable the C/Q driver by setting the CQ_EN bit in the CQ_CTRL1 register and set CQ_TXEN high to enable the MAX22512/MAX22522 to detect a wake-up request from the IO-Link master when in SIO mode.

The MAX22512/MAX22522 feature an integrated IO-Link establish communication (EST_COM) sequencer to autonomously perform the IO-Link establish communication sequence when a valid wake-up pulse is detected (*Figure 11*). Once the part is powered and the clocks are stable, configure the C/Q in PNP, NPN, or push-pull mode (IOL_CQ_CTRL1 and IOL_CQ_CTRL2 registers), and set the IO-Link device communication rate to COM1, COM2, or COM3 by setting the COMx bits in the IOL_CFG register. Set the C/Q slew rate by setting the CQ_SLEW[1:0] bits in the IOL_CQ_CTRL1 register. ADI recommends setting CQ_SLEW[1:0] = 01 for COM3 communication. After the IO-Link configuration registers are programmed, set the CQ_CFG bits in the CQ_CFG_REG register to 0x04 to finalize the configuration and prepare for the wake-up pulse and EST_COM sequence.

The DL_MODE bits in the IOL_STAT register indicate the current state of the DL-mode handler during the power-up and establish communication sequence. When the DL-mode changes state, the DL_MODE_INT bit in the IOL_INT register is set.

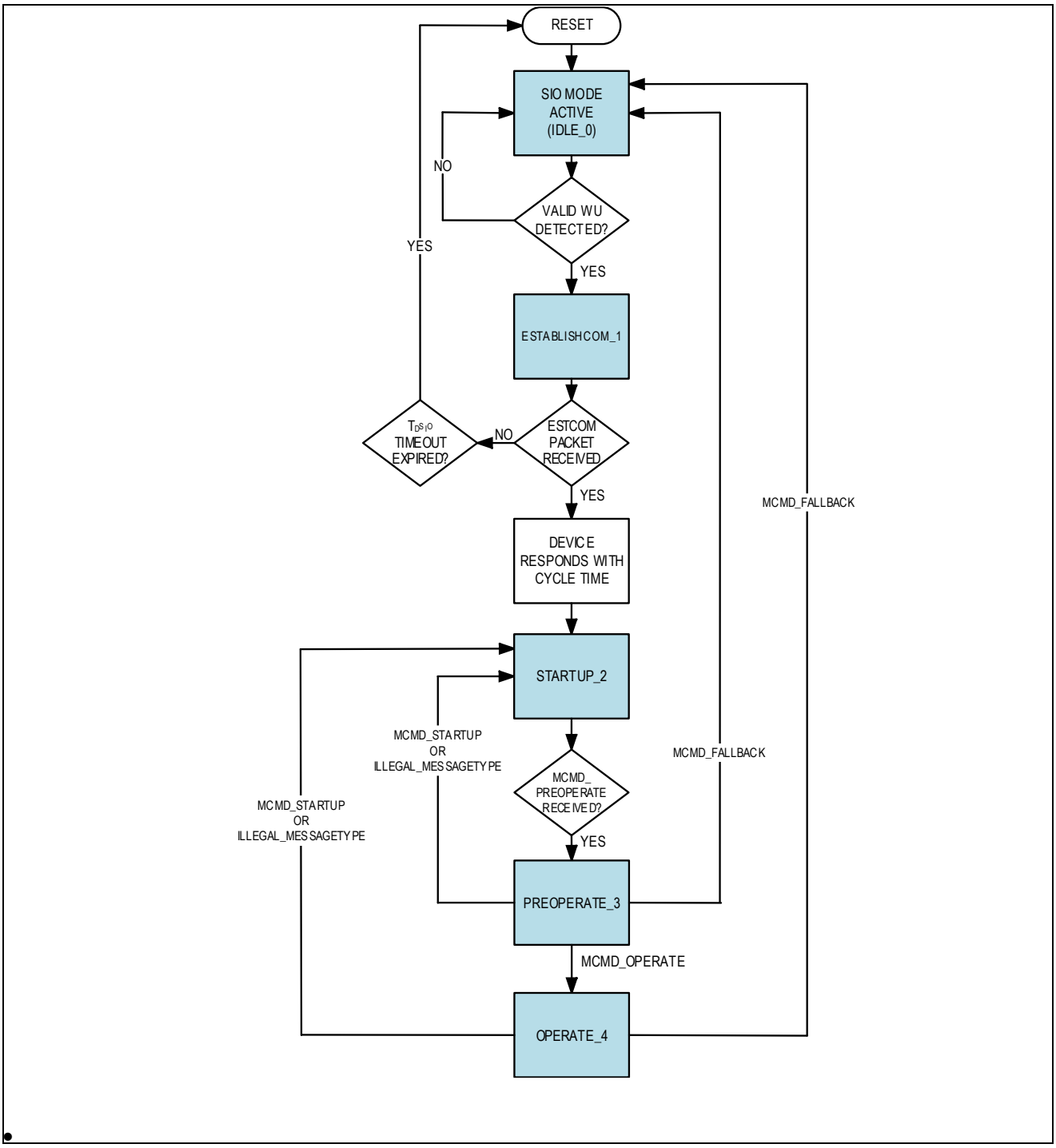


Figure 11. DL-Mode Handler Flowchart

Fallback Procedure

The MAX22512/MAX22522 return to SIO mode within 420ms (typ) after receiving a fallback command from an IO-Link master.

It is also possible to reset the MAX22512/MAX22522 to SIO mode. Set the SIO_FORCE bit in the IOL_CFG register to force the MAX22512/MAX22522 to return to SIO mode. There is no delay when switching to SIO mode once the SIO_FORCE bit is set.

Process Data Transfers

Process Data Output (PDOut)

The MAX22512/MAX22522 feature a process data out (PDOut) buffer architecture that supports up to 32-byte IO-Link PDOut data with buffering for reliable data transfer. The MAX22512/MAX22522 autonomously handle the tasks supporting process data output from the IO-Link master to the PDOut buffer.

The PDOut buffer is mapped from 0x40040480 to 0x400404FC. Data read from the PDOut buffer is always the latest received data. The PDOUT_DATRX_INT interrupt bit in the IOL_INT register is set when the MAX22512/MAX22522 receive a new PDOut message.

The MAX22512/MAX22522 can only receive the number of PDOut bytes defined in the MSEQ_CAPABILITY and PROCESS_DATA_OUT bytes in the IOL_PAGE1_BYTE02 and IOL_PAGE1_BYTE06 registers. If the IO-Link master transmits the wrong number of PDOut bits, the M-sequence is ignored, and the MAX22512/MAX22522 return to the IO-Link STARTUP state.

Access the PDOut buffer by setting PDOUT_LOCK = 1 in the IOL_PDOUT_LOCK register. When this bit is set, the PDOut last message received is locked. The MAX22512/MAX22522 continue to monitor the C/Q line and can receive new PDOut messages while the current message is being processed. The PDOUT_LOCK bit is automatically cleared once the locked PDOut message is available for read.

Figure 12 shows an example of a PDOut buffer read.

```
void IOL_GetPDout(uint8_t *pdout)
{
  for (int i = 0; i < PDout.length; i++)
    {
        IOLINK_PDOUT_LOCK = 0x02;                       // lock PDout Buffer
        while ((IOLINK_PDOUT_LOCK & 0x02) != 0) {}
        for (int i = 0; i < PDout.length; i++) { pdout[i] = IOLINK_PDOUT_BUF(i); }
        IOLINK_PDOUT_LOCK = 0x01;                       // unlock PDout Buffer
    }
}
```

Figure 12. PDOut Buffer Read Example Code

At the end of the read operation, set PDOUT_UNLOCK = 1 in the IOL_PDOUT_LOCK register and repeat this procedure to read in the next PDOut message.

Process Data Input (PDIn)

A process data in (PDIn) buffer that supports up to 32-byte IO-Link PDIn data handles all real-time tasks related to process data input from the PDIn buffer to the IO-Link master.

The MAX22512/MAX22522 transmit the number of PDIn bytes declared in the MSEQ_CAPABILITY and PROCESS_DATA_IN bytes in the IOL_PAGE1_BYTE03 and IOL_PAGE1_BYTE05 registers. The PDIn buffer is mapped from 0x40040400 to 0x4004047C.

Access the PDIn buffer by setting PDIN_LOCK = 1 in the IOL_PDIN_LOCK register. This bit is used to lock the PDIn buffer and is automatically cleared after the lock is performed. To ensure the PDIn buffer is write-accessible, wait until the PDIN_LOCK bit is cleared before writing to the buffer. Once the write to the buffer is complete, set

PDIN_UNLOCK = 1 in the IOL_PDIN_LOCK register. When PDIN_UNLOCK = 1, the locked PDIn buffer is unlocked. The PDIN_UNLOCK bit is automatically cleared after unlock is performed.

Figure 13 shows the example code to execute the PDIN_LOCK and write to the PDIn buffer data.

```
void IOL_SetPDin(uint8_t *pdin)
{
    volatile uint8_t pdin_lock = IOLINK_PDIN_LOCK;
    pdin_lock |= 0x02;                                    // Lock the PDin Buffer
    IOLINK_PDIN_LOCK = pdin_lock;

    volatile uint8_t lock_read = 0x02;
    while ((lock_read & 0x02) != 0)
    { lock_read = IOLINK_PDIN_LOCK;}

  for (int i = 0; i < PDin.length; i++)
    { IOLINK_PDIN_BUF(i) = pdin[i];}                      // Write PDin Data

    pdin_lock |= 0x01;
    IOLINK_PDIN_LOCK = pdin_lock;
}
```

Figure 13. PDIn Buffer Write Example Code

Set the PD_STATUS bit in the IOL_PDIN_LOCK register after data is written to the PDIn buffer data. Once received, the MAX22512/MAX22522 automatically transmit data in the PDIn buffer in the next device M-sequence. The PDValid bit in CKS byte of the M-packet is set and 0x00 is transmitted as PDIn data when PD_STATUS = 0.

Process Data Using GPIOs

Process data exchanges allow regular cyclical transmission between a device and an IO-Link master. The MAX22512/MAX22522 feature the ability to configure GPIO pins to aid in this transfer as PDIn logic inputs (GPIO1_4 and GPIO1_5) or as PDOut logic outputs (GPIO1_3 and GPIO1_6).

PDIN1, PDIN2 (GPIO1_4, GPIO1_5)

PDIn data, transmitted from the MAX22512/MAX22522 to the IO-Link master can be generated using the GPIO1_4 and GPIO1_5 pins. Set the GPIO1_PDIN1 and GPIO1_PDIN2 bits in the GPIO1_ALT_FUNC register to enable this functionality.

When GPIO1_PDIN1 = 1 and GPIO1_PDIN2 = 0, the status of the GPIO1_5 pin is inserted into bit 0 of the first byte (BYTE 0) of the PDIn data sent to the IO-Link master. When GPIO1_PDIN1 = 0 and GPIO1_PDIN2 = 1, the status of GPIO1_4 is inserted into bit 0 of BYTE 0. Alternatively, setting both GPIO1_PDIN1 = 1 and GPIO1_PDIN2 = 1 inserts the status of the GPIO1_4 and GPIO1_5 pins as bit 1 and bit 0, respectively, of BYTE 0. See *Table 24*.

Only BYTE 0 of the PDIn data contains the GPIO1_4 and/or GPIO1_5 status. Additional bytes in the PDIn communication use only data from the PDIn buffers.

The status of the GPIO pins configured for PDIn functionality are sampled shortly before the falling edge of the start bit when the PDIn byte is sent. Changes after the status is sampled are ignored until the next PDIn byte is generated.

Table 24. PDIn Using GPIOs

	GPIO1 _PDIN1	GPIO1_PDIN2	PDIn[7:2]	PDIn[1]	PDIn[0]
BYTE 0	0	0	FROM BUFFER		
	0	0	0	0	GPIO1_4
	1	0	0	0	GPIO1_5

	1	0	0	GPIO1_4	GPIO1_5
BYTE 1	X	X	FROM BUFFER		
BYTE 2	X	X	FROM BUFFER		
		⋮			
BYTE n	X	X	FROM BUFFER		

PDOUT1, PDOUT2 (GPIO1_3, GPIO1_6)
PDOut data, received from the IO-Link master, can be used to set the status of some GPIOs. Set the GPIO1_PDOUT1 and GPIO1_PDOUT2 bits in the GPIO1_ALT FUNC register to enable this functionality.

When GPIO1_PDOUT1 = 1 and GPIO1_PDOUT2 = 0, the status of the GPIO1_6 pin reflects the status of bit 0 of the PDOut data received from the IO-Link master. When GPIO1_PDOUT1 = 0 and GPIO1_PDOUT2 = 1, the status of GPIO1_3 reflects the status of bit 0 in the received PDOut data. Similarly, setting both GPIO1_PDOUT1 = 1 and GPIO1_PDOUT2 = 1 enables this functionality on both GPIO1_3 and GPIO1_6 to reflect the status of bit 1 and bit 0 of the received PDOut data. See *Table 25*.

When more than one PDOut byte is transmitted by the IO-Link master, the status of the enabled GPIO(s) is overwritten for every byte transmitted.

All PDOut data bits are stored in the PDOUT buffer, regardless of the state of the GPIO1_PDOUTx enable bits.

Table 25. PDOut Using GPIOs

GPIO1_PDOUT1	GPIO1_PDOUT2	PDOut[7:2]	PDOut[1]	PDOut[0]
0	0	TO BUFFER		
0	1	X	X	GPIO1_3
1	0	X	X	GPIO1_6
1	1	X	GPIO1_3	GPIO1_6

X = Don't care

ISDU Transmission

The MAX22512/MAX22522 handle the real-time tasks to support ISDU data transfer in both IN (that is, from the IO-Link device to the IO-Link master) and OUT (that is, from the IO-Link master to the device) directions. The MAX22512/MAX22522 integrate a 256-bytes ISDU buffer for both IN and OUT directions. The ISDU buffer is mapped from 0x40040500 to 0x400408FC.

When the IO-Link master sends an ISDU to the device, the MAX22512/MAX22522 decode the length of the ISDU structure and verifies the CRC. If the ISDU data is valid, the ISDU_PCKT_INT interrupt bit in the ISDU_INT register is set. If the received data is invalid (that is, the CRC is incorrect), the MAX22512/MAX22522 ignore the ISDU request.

Access the ISDU buffer by writing ISDU_LOCK = 1 in the ISDU_DATA_RDY register.

When the ISDU read/write is complete, set ISDU_LOCK = 0 in the ISDU_DATA_RDY register, as shown in *Figure 14*.

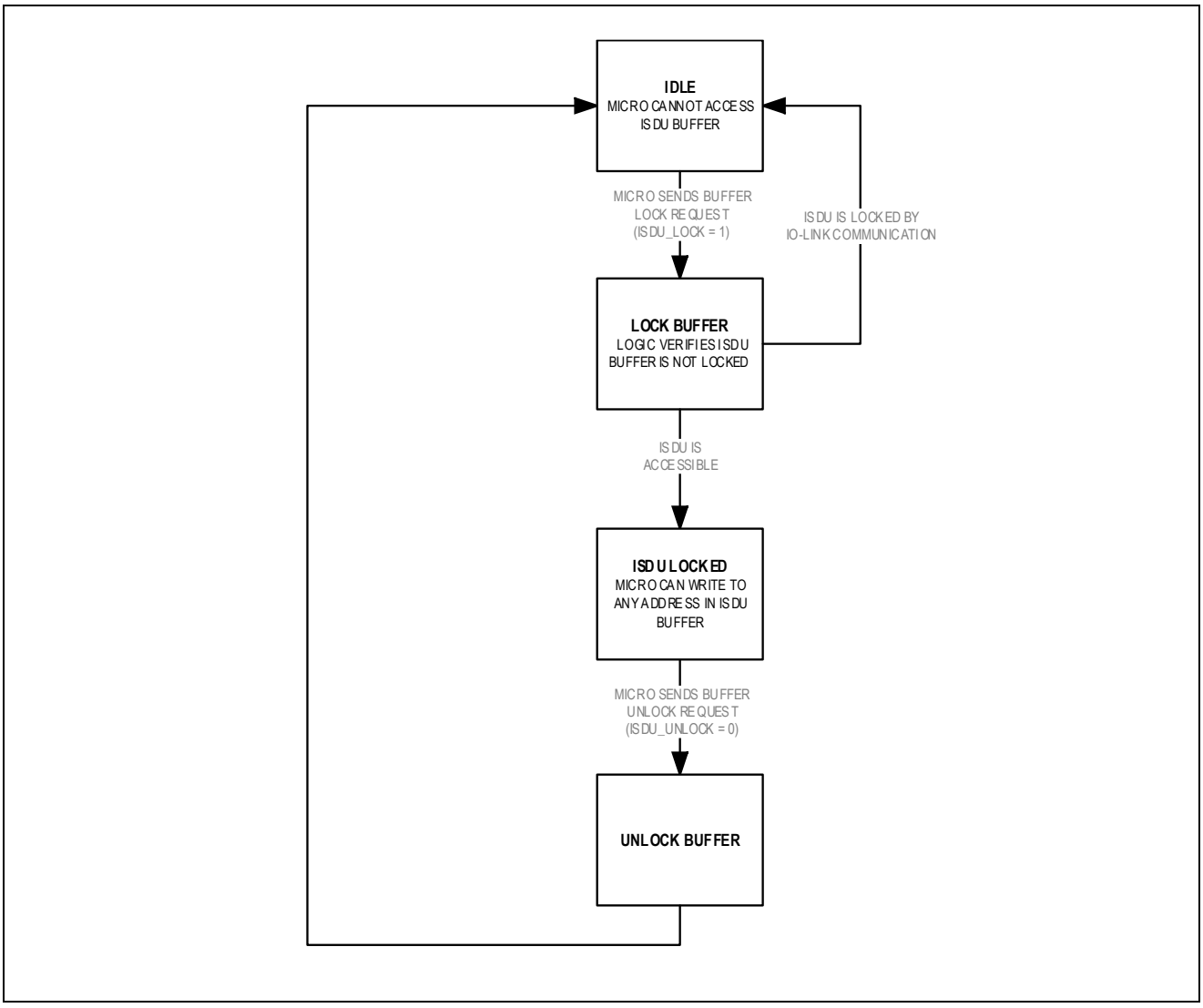


Figure 14. ISDU Read/Write Access

The MAX22512/MAX22522 automatically respond to any further master requests with a “busy” indication until the microcontroller loads the response into the ISDU buffer. Once done, the MAX22512/MAX22522 set the ISDUIN_DATA_RDY bit in the ISDU_DATA_RDY register. Data in the ISDU FIFO is then transferred over the C/Q line to the waiting IO-Link master.

The MAX22512/MAX22522 detect that the ISDU data transmission is complete when the IO-Link master sends an ISDU idle command, an ISDU abort command, or a new ISDU request. Once transmission is complete, the MAX22512/MAX22522 set the ISDU_IDLE_INT bit in the ISDU_INT register and the ISDUINFIFO register data is cleared.

Figure 15 shows an example code to access the ISDU buffers.

```
int16_t IOL_GetISDUreq(uint8_t *buf)
{
    isdu_size = 0;


    // get size by reading first byte
```

```
IOLINK_ISDUDATARDY = 0x02;                // lock ISDU buffer

isdu_size = (IOLINK_ISDU_BUF(0) & 0x0f);
if(isdu_size == 1) isdu_size = IOLINK_ISDU_BUF(1);

for (int i = 0; i < isdu_size; i++)
{
    buf[i] = IOLINK_ISDU_BUF(i);
}

if ((IOLINK_ISDUDATARDY & 0x02) != 0x02)  // if the bit was cleared during read, then the master sent another
                                          // request -> ABORT
{
    return -1;
}
IOLINK_ISDUDATARDY = 0x00;         // Bit was obviously not cleared by the master -> release the buffer now
return isdu_size;
}
```

Figure 15. ISDU Buffer Access Example Code

ISDU Idle

The MAX22512/MAX22522 automatically detect and responds to an ISDU idle command from the IO-Link master without extra information or data from the device microcontroller.

Events

IO-Link device events are transmitted using acyclic transfers. The MAX22512/MAX22522 use four bytes to generate an event structure: the bytes in the STATUS_CODE, EVENT_QUALIFIER, EVENT_CODE1MSB, and EVENT_CODE1LSB registers.

Event signaling begins when the MAX22512/MAX22522 write the event information to the STATUS_CODE, EVENT_QUALIFIER, EVENT_CODE1MSB, and EVENT_CODEL1LSB registers, and sets the EVENT_FLAG_BIT bit in the EVENT_FLAG register. Setting the EVENT_FLAG_BIT bit corresponds to the EventFlag bit in the IO-Link checksum/status (CKS) packet. Note that the MAX22512/MAX22522 support one event only.

Once the MAX22512/MAX22522 set the EventFlag bit, data in the STATUS_CODE, EVENT_QUALIFIER, EVENT_CODE1MSB, and EVENT_CODE1LSB registers are locked until the current event is processed by the IO-Link master. The IO-Link master writes to the STATUS_CODE register to indicate the end of event processing. The EVENT_FLAG_BIT is automatically cleared and the CLR_EVENT_FLAG_INT bit in the IOL_INT register is set when event processing is completed.

LEDs: Status and Diagnostic Indicators

The MAX22512/MAX22522 integrate two logic outputs for controlling LEDs (LED1 and LED2). These pins can be used as indicators of active single-drop digital communication interface (SDCI) communication and are controlled by setting bits in the LED1CTRMSB, LED1CTRLSB, LED2CTRMSB, and LED2CTRLSB registers.

Set the 16-bit LED control sequence for each LED by setting the bits in the corresponding LEDxCTRMSB and LEDxCTRLSB registers. The LED status (ON or OFF) is set according to the configuration in these registers. In sequential order, a bit in the registers (MSB to LSB) is sampled every 63ms and the 16-bit pattern is repeated every 1.008 seconds. *Table 26* shows an example of an LED control sequence.

Timers for the LED1 and LED2 channels are independent and are reset when the control registers are written.

Write to both the LEDxCTRMSB and LEDxCTRLSB registers when setting the LED bits.

Table 26. LED1 Program Example

	LED1CTRMSB								LED1CTRLSB							
REGISTER BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BIT LOGIC	0	1	1	0	1	0	1	0	0	1	1	0	1	0	1	0
LED1 STATUS	OFF	ON	ON	OFF	ON	OFF	ON	OFF	OFF	ON	ON	OFF	ON	OFF	ON	OFF

High-Speed Comparators

The MAX22512 features two high-speed, rail-to-rail, 5V tolerant comparators: CMP1 and CMP2. The negative input terminals of the comparators (CIN1 and CIN2) are connected internally to the DAC1 and DAC2 outputs, respectively (not externally available). Set the comparator input thresholds by programming the DAC1 and DAC2 registers. DAC1, DAC2, as well as the CMP1 and CMP2 comparators are powered by the V_{5A} supply input and are referenced to AGND.

Set the CMP1_EN bit in the CMP1 register to enable the comparator 1 output, CMPO1. Set the CMP2_EN bit in the CMP2 register to enable the comparator 2 output, CMPO2.

The MAX22522 features a single high-speed, rail-to-rail, 5V tolerant comparator, CMP1. Enable and configure the comparator by writing to the bits in the CMP1 register. Set the comparator input thresholds by programming the DAC1 register and connecting the DAC1 output (DACO1) to CIN1 or CIP1. The DAC1 and comparator are powered by the V_{5A} supply input and are referenced to AGND.

Set the CMPx_EN bit in the CMP1 register to enable the comparator output, CMPO1.

Table 27 shows the configuration registers for the comparator and DACs.

Table 27. Comparator and DAC Configuration Registers

REGISTER NAME	REGISTER ADDRESS	ACCESS TYPE*	FUNCTION
CMP1	0x00	RW	CMP1 enable and configuration
CMP2**	0x04	RW	CMP2 enable and configuration (MAX22512 only)
DAC1	0x08	RW	DAC1 enable and value
DAC2**	0x0C	RW	DAC2 enable and configuration (MAX22512 only)
CMP_INT_SET	0x20	RW	CMP debounce and edge selection
CMP_INT	0x24	RW	CMP interrupt enable and indicator bits
CMP_TO_PDIN_REG	0x2C	RW	CMP to SIO functionality and CMP to PDIN configuration

**RW = Read/write, RO = Read only*

***MAX22512 only*

Selecting the Comparator Input Range

Comparators are 5V tolerant and are designed to allow either rail-to-rail functionality, or a reduced offset near a selected rail. For applications that do not require rail-to-rail compatibility, it is possible to reduce the input range and reduce offset that occurs as the input approaches either the high or low rail.

Set the CMPx_IN_LOW and CMPx_IN_HIGH bits in the CMPx register to set the input range. Note that either the CMPx_IN_LOW bit or the CMPx_IN_HIGH bit must be set for normal operation. See *Table 28*.

Table 28. Comparator Input Range Select Bits

CMPx_IN_LOW BIT	CMPx_IN_HIGH BIT	CMP INPUT RANGE
0	0	Reserved. Do not use this configuration.
0	1	Offset is reduced as the input signal approaches the high-side rail (V_{5A} voltage).
1	0	Offset is reduces as the input signal approaches the low-side rail (AGND).
1	1	Rail-to-rail input range

Comparator Filters

The comparator features a programmable propagation delay and a selectable filter. Set the CMPx_SLOW_EN bit in the associated CMP register to increase the propagation delay from 75ns (typ) to 700ns (typ). Additionally, set the CMPx_FILT_EN bit in the associated CMP register to enable/disable a 1µs (typ) filter on the comparator output. See *Table 29*.

Table 29. Comparator Propagation Delay and Filter Select Bits

CMPx_FILT_EN BIT	CMPx_SLOW_EN BIT	CMPO OUTPUT
0	0	Fast comparator configuration. 75ns (typ) propagation delay, no filter enabled.
0	1	Slow comparator configuration. 465ns (typ) propagation delay, no filter enabled.
1	0	Fast comparator configuration with 700ns (typ) filter enabled.
1	1	Slow comparator configuration with 1µs (typ) filter enabled.

Setting the Comparator Threshold with the DAC

The MAX22512 features two DACs: DAC1 and DAC2, that are internally connected to the inverting terminal of the CMP1 and CMP2 comparators, respectively. Set the DACx_EN bit in the associated DAC register to enable the DAC output and set the comparator threshold value using the equation below.

The MAX22522 features a 6-bit DAC1 output, DACO1, that can be connected to the CMP1 input (CIN1 or CIP1) to set the comparator threshold.

Set the DACx_EN bit in the associated DACx register to enable the DAC output (DACOx). Set the DAC output voltage, by setting the DAC_VALUE bits in the DAC register as:

$$\text{DAC\_VALUE} = \frac{V_{COMP\_TH}}{5V} \times 64$$

where, VCOMP_TH is the desired comparator threshold in volts.

Analog-to-Digital Converter (ADC)

The MAX22512/MAX22522 feature a 13-bit (12-bit + plus sign) SAR ADC to sample analog signals from the ADP and ADN pins directly, or from the GPIO1_3 – GPIO1_6 pins using a low-offset buffer, an internal reference, or internal thermal sensor (*Figure 16*). The ADC features an internal 1.25V (typ) reference.

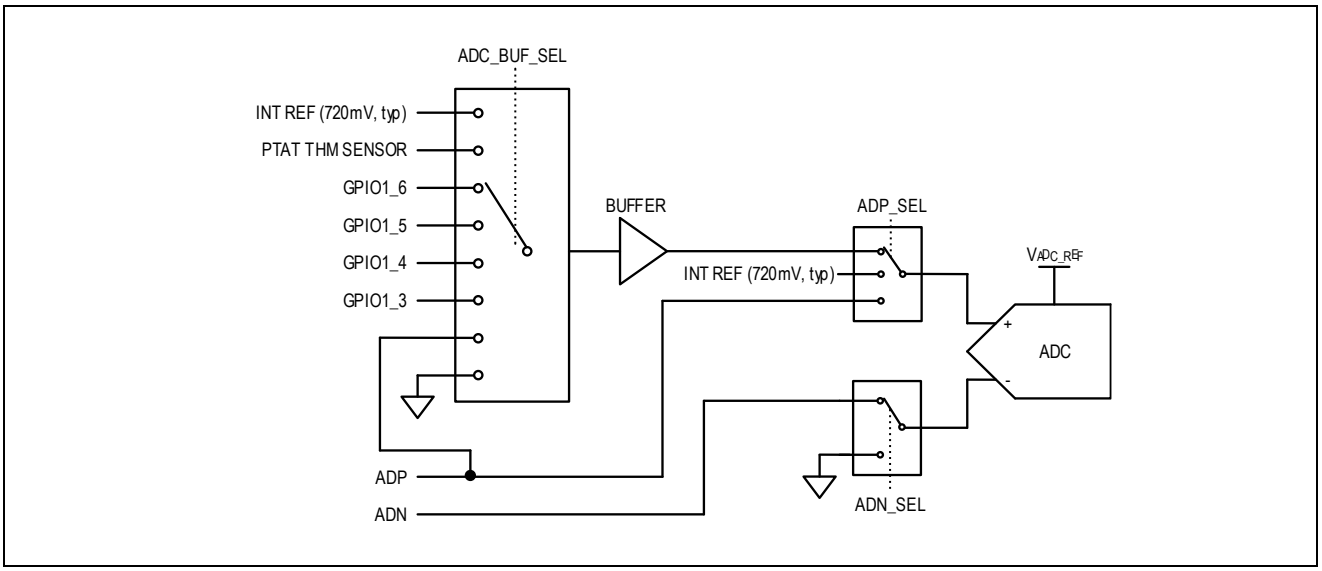


Figure 16. ADC Block Diagram

Table 30 shows the configuration and conversion registers for the ADC.

Table 30. ADC Configuration Registers

REGISTER NAME	REGISTER ADDRESS	ACCESS TYPE*	FUNCTION
ADC_CONV	0x00	RW	ADC input configuration, conversion mode configuration, conversion START bit, and conversion BUSY bit
ADC_DATA	0x04	RO	ADC conversion result including value, sign, and conversion clip indicator
ADC_CTRL	0x08	RO, RW	ADC ready and initialization
ADC_INT	0x14	RW	ADC interrupt enable/disable and interrupt indicator bits
ADC_START_DELAY	0x18	RW	ADC conversion start delay
ADC_AVERAGE	0x20	RW	ADC averaging mode (sample delay and averaging count)

*RW = Read/write, RO = Read only

Configuring and Measuring with the ADC (ADC_CONV, ADC_DATA, ADC_CTRL)

Selecting the ADC Inputs (ADP/AND or Buffer Input)

The ADC can directly sample signals on the ADP and ADN pins, or from the GPIO1_ inputs, an internal reference, or an internal thermal sensor using an internal buffer. Select the ADC inputs by programming the ADC_BUF_SEL bits in the ADC_CONV register.

Ensure the voltages on the ADC inputs do not exceed the absolute maximum ratings for each pin. ADP and ADN analog inputs can range from 0V to 1.8V (typ). However, signal inputs at the multiplexer have a 5V tolerance. Program the ADP_SEL, ADN_SEL and ADC_BUF_SEL bits to configure the inputs to the ADC.

ADC Averaging Mode

It is possible to configure the ADC to convert multiple samples over a programed time period following an ADC trigger event. Set the number of samples for conversion by programming the ADC_AVG_NUM bits in the ADC_AVERAGE register. By default, ADC_AVG_NUM = 000, which is one sample per conversion. To enable averaging mode, set ADC_AVG_NUM to a value > 000. See the ADC_AVERAGE register in the register map for more information.

The ADC averaging mode sample time ($t_{ADC\_AVG\_DLY}$) can be programmed by setting the ADC_AVG_DLY bits in the ADC_AVERAGE register.

Selecting the ADC Conversion Trigger

Set the ADC_CONV_MODE bits in the ADC_CONV register to select the trigger for an ADC conversion. The ADC is disabled by default (ADC_CONV_MODE = 000).

Software Trigger (ADC_CONV_MODE = 001)

Set ADC_CONV_START = 001 to generate an ADC conversion through software. In this mode, set ADC_CONV_START = 1 to generate an ADC conversion at any time. ADC_CONV_START is automatically cleared when the conversion is complete.

GPIO Event Trigger (ADC_CONV_MODE = 010, 011, or 100)

While only the GPIO1_x pins can be used as ADC inputs, an ADC conversion can be triggered on the rising, falling, or rising and falling edges of any of the GPIO1_x or GPIO2_x pins configured as inputs. Set the ADC_TRG_GPIO_SEL bits in the ADC_CONV register to select the GPIO that can trigger an ADC conversion. Set the ADC_CONV_MODE bits to select a trigger on the rising edge (010), falling edge (011), or on both rising and falling edges (100) for the selected GPIO.

Free Running Conversion (ADC_CONV_MODE = 101)

Set the ADC_CONV_MODE bits to 101 to configure the ADC in free running mode. In this mode, the ADC performs continuous successive conversions.

The conversion interrupt is triggered (ADC_CONV_INT = 1) at the completion of each conversion. To ensure continuity during high-speed conversions, disable or mask this interrupt by setting ADC_CONV_INT_EN = 0.

HEART Signal Trigger (ADC_CONV_MODE = 110)

Set ADC_CONV_MODE bits to 110 to enable an ADC conversion whenever a HEART signal is received. A HEART signal is triggered when an M-sequence control byte is received.

ADC Conversion Delays

Up to two delays can be configured for the ADC: a conversion start delay between the selected ADC trigger ($t_{ADC\_START\_DLY}$) and the start of the conversion, and the sample time when using ADC averaging mode ($t_{ADC\_AVG\_DLY}$). These delays apply for all conversion modes. Set the ADC_START_DLY bits in the ADC_START_DELAY register to program the $t_{ADC\_START\_DLY}$ time after an ADC trigger event occurs. Set the ADC_AVG_DLY bits in the ADC_AVERAGE register to program the $t_{ADC\_AVG\_DLY}$ sample time.

Figure 17 shows a sample timing diagram for an ADC conversion. In this example, four samples are averaged (ADC_AVERAGE_NUM = 010). The ADC sampling begins immediately after the ADC trigger event, after which, the ADC converts the programmed number of samples using the programmed sample time and hold delay.

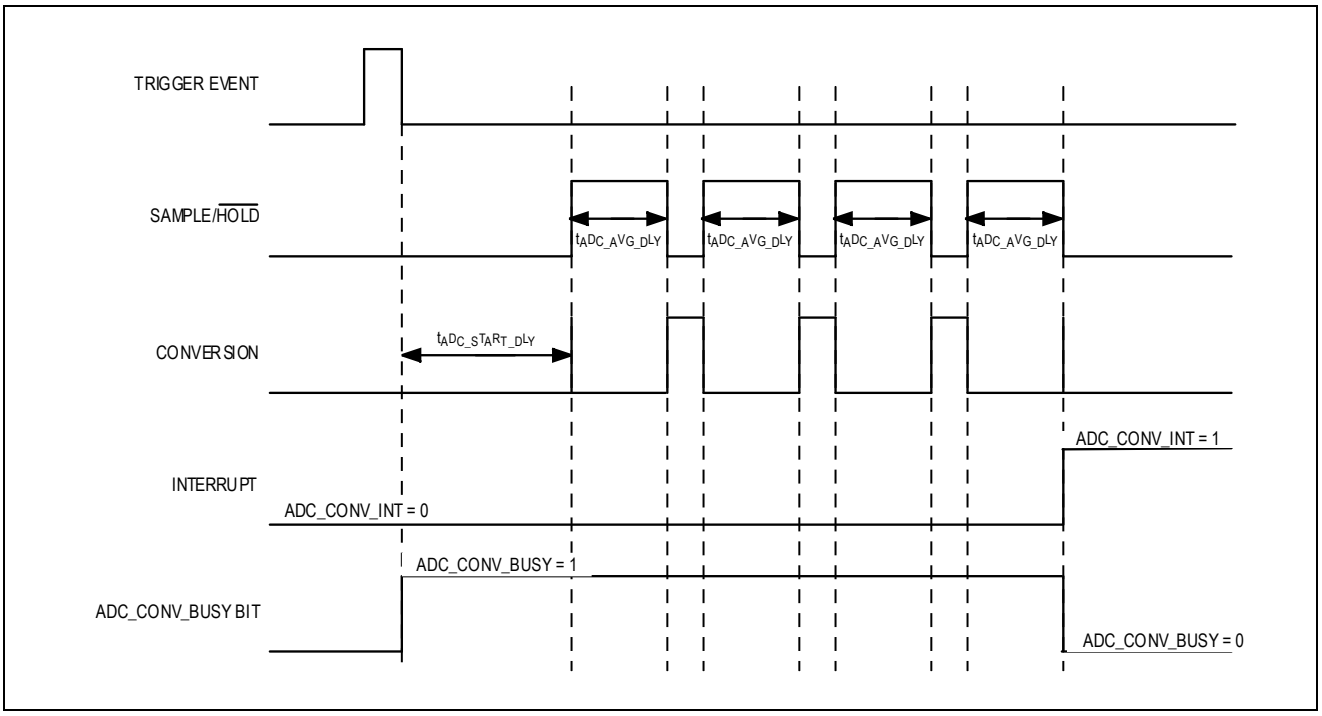


Figure 17. Event-Triggered ADC Conversion with Programmed Startup Delay and Average Sample Time

Figure 18 shows a sample timing diagram when the ADC is configured to run in free mode, average four samples, and includes a startup delay after the trigger. Note that in this diagram, the interrupt signal is triggered after the previous conversion and starts high during this conversion. During the conversion, the interrupt is read and reset, and is triggered again at the end of the conversion.

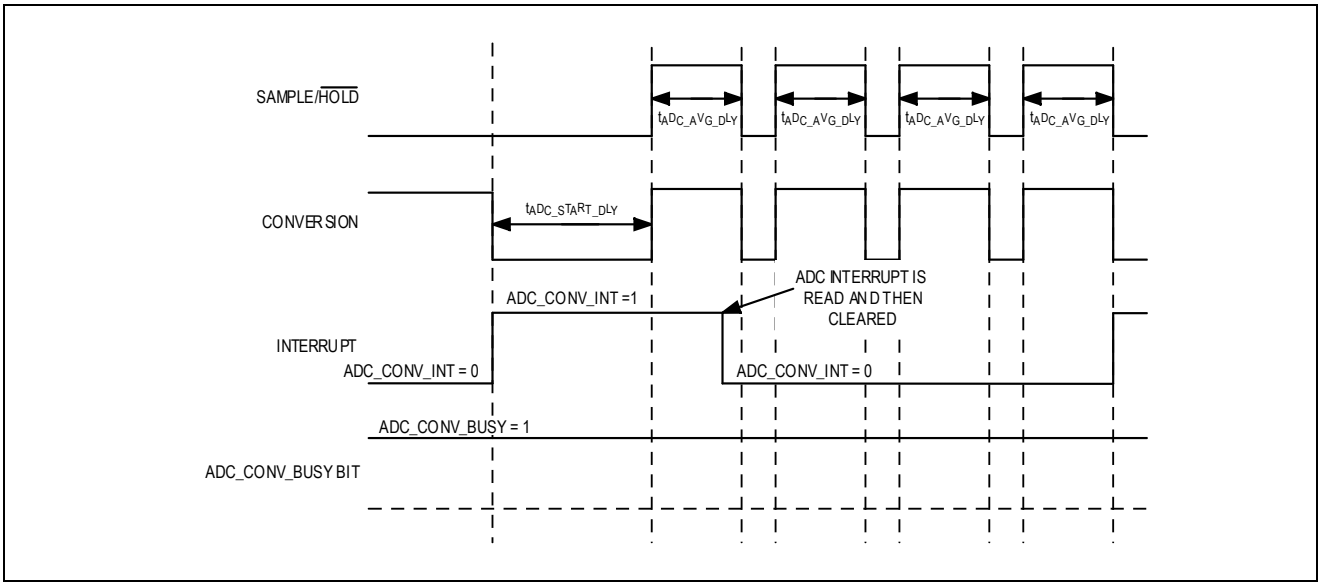


Figure 18. Free Running ADC Conversion Mode with Programmed Startup Delay and Averaging Mode

ADC Calibration

The MAX22512/MAX22522 is calibrated during test and calibration values are preprogrammed into the device at that point. No additional calibration should be required.

Set the ADC_INIT bits in the ADC_CTRL register to 11 to initialize the device for ADC operation.

ADC Interrupts (ADC_INT)

Two interrupts are available to optimize ADC performance: ADC_READY_INT and ADC_CONV_INT.

The ADC_READY_INT bit is set when the ADC has completed its startup routine and is ready for operation. Set the ADC_READY_INT_EN bit in the ADC_INT register to generate an interrupt when this bit is set. Write a 1 to the ADC_READY_INT bit to clear it.

Alternatively, poll the ADC_READY bit to ensure ADC_READY = 1 before attempting to begin a conversion.

The ADC_CONV_INT bit is set when an ADC conversion is completed. Set the ADC_CONV_INT_EN bit in the ADC_INT register to generate an interrupt when this bit is set. Write a 1 to the ADC_CONV_INT bit to clear it.

Variable Resistors

The MAX22512/MAX22522 feature four variable resistors for onboard trimming. *Table 31* shows the configuration and conversion registers for the digipots.

Table 31. ADC Configuration Registers

REGISTER NAME	REGISTER ADDRESS	ACCESS TYPE*	FUNCTION
R1	0x10	RW	R1 enable and wiper position
R2	0x14	RW	R2 enable and wiper position
R3	0x18	RW	R3 wiper position
R4	0x1C	RW	R4 enable and wiper position

**RW = Read/write, RO = Read only*

High-Speed Variable Resistors (R1, R2)

MAX22512: R1A, R2A, R12W

The MAX22512 features two low-capacitance variable resistors, R1 and R2, connected to a common wiper, R12W, which can be used in either potentiometer or rheostat modes (*Figure 19*). R1 and R2 have an end-to-end impedance of 2.5kΩ (typ) and operate up to 5V. Ensure that the current into the R1 and/or R2 does not exceed 2mA.

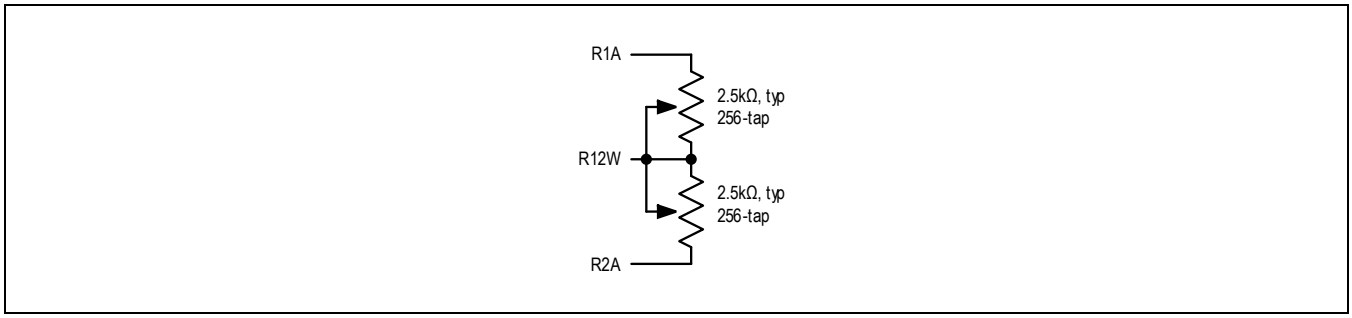


Figure 19. MAX22512 Digital Potentiometers (R1, R2)

The 8-bit data in the R1_POS and R2_POS bits in the R1 and R2 registers, respectively, is decoded to select one of 256 possible settings for each variable resistor. Enable R1 and R2 by setting the R1_EN and R2_EN bits, respectively.

Calculate the output resistance between R12W and R1 or R2 as:

$$R_{to\_R12W} = \left(2.5k\Omega - \frac{Rx\_POS\ value}{256} \times 2.5k\Omega\right) + (2 \times R_W)$$

where, R_W is the 85Ω (typ) wiper resistance.

MAX22522: R1A, R1W, R1B, R2A, R2W, R2B

The MAX22522 features two low-capacitance variable resistors, R1 and R2, which can be used in either potentiometer or rheostat modes. Both R1 and R2 have an end-to-end impedance of 10kΩ (typ) and operate up to 5V. Ensure the current into either R1 and/or R2 does not exceed 2mA.

Enable R1 and R2 by setting the R1_EN and R2_EN bits, respectively. The 8-bit data in the R1_POS and R2_POS bits in the R1 and R2 registers, respectively, is decoded to select one of 256 possible settings for each variable resistor.

Calculate the output resistance between the wiper (R1W, R2W) and the associated RxB output as:

$$R_{xW} = \left(10k\Omega - \frac{Rx\_POS\ value}{256} \times 10k\Omega\right) + (R_W)$$

where, R_W is the 85Ω (typ) wiper resistance.

Variable Resistors (R3, R4)

The MAX22512/MAX22522 feature two low-speed digital potentiometers (R3 and R4). The R3_POS and R4_POS bits in the R3 and R4 registers are decoded to one of 64 and 256 (respectively) settings to set the resistance to ground. Both resistors are capable of operating with voltages up to 5V.

R3 has an end-to-end impedance of 60.4kΩ (typ) to AGND. Ensure the current into R3 does not exceed 2mA. Enable the R3 variable resistor by setting the R3_POS bits in the R3 register to any value greater than 0x00. Calculate the resistance at R3 as:

$$R_3 = 60.4k\Omega - \left(\frac{R3\_POS\ value}{63} \times 60.4k\Omega\right)$$

The R3 variable resistor can also be enabled or disabled with the CMPO1 output by setting the CMPO1_TO_R3_EN and CMPO1_TO_R3_DIS_SEL bits in the R3 register. See *Table 32*.

Table 32. R3 Control with CMP1 Comparator Output

CMPO1_TO_R3_DIS_SEL BIT	CMPO1_TO_R3_EN BIT	CMPO1	R3
X	0	Low	R3 is not enabled/disabled by CMPO1 output.
		High	
0	1	Low	Disabled
		High	Enabled
1	1	Low	Enabled
		High	Disabled

X = Don't care

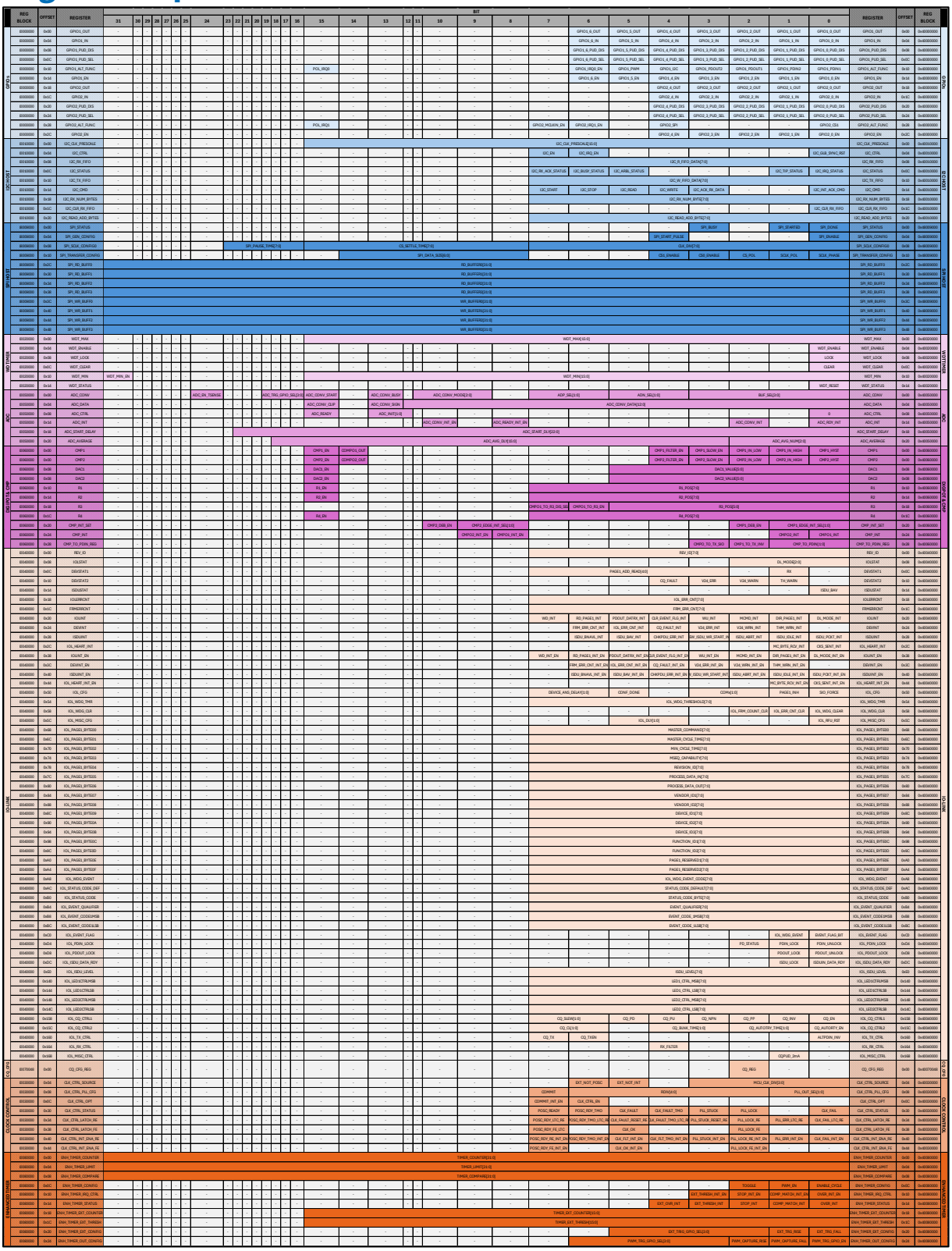
For the MAX22512, R4 has an end-to-end impedance of 2.5kΩ (typ) to AGND. Ensure the current into R4 does not exceed 2mA. Set the R4_EN bit in and program the R4_POS bits in the R4 register to move the wiper from the R4 pin towards ground. Calculate the resistance at R4 as:

$$R_{4\_MAX22512} = 2.5k\Omega - \left(\frac{R4\_POS\ value}{255} \times 2.5k\Omega\right)$$

For the MAX22522, R4 has an end-to-end impedance of 10kΩ (typ) to AGND. Ensure the current into R4 does not exceed 2mA. Set the R4_EN bit in and program the R4_POS bits in the R4 register to move the wiper from the R4 pin towards ground. Calculate the resistance at R4 as:

$$R_{4\_MAX22522} = 10k\Omega - \left(\frac{R4\_POS\ value}{255} \times 10k\Omega\right)$$

Register Map Overview



Register Map

GPIO REGISTERS

ADDRESS	NAME	MSB							LSB
GPIO REGISTERS									
0x40000000	GPIO1_OUT[7:0]	–	GPIO1_OUT[6:0]						
0x40000004	GPIO1_IN[7:0]	–	GPIO1_IN[6:0]						
0x40000008	GPIO1_PUD_DIS[7:0]	–	GPIO1_PUD_DIS[6:0]						
0x4000000C	GPIO1_PUD_SEL[7:0]	–	GPIO1_PUD_SEL[6:0]						
0x40000010	GPIO1_ALT_FUNC[15:8]	POL_IRQ0	–	–	–	–	–	–	–
	GPIO1_ALT_FUNC[7:0]	–	GPIO1_IRQ0_EN	GPIO1_PWM	GPIO1_I2C	GPIO1_PDOUT2	GPIO1_PDOUT1	GPIO1_PDIN2	GPIO1_PDIN1
0x40000014	GPIO1_EN[7:0]	–	GPIO1_OUT_EN[6:0]						
0x40000018	GPIO2_OUT[7:0]	–	–	–	GPIO2_OUT[4:0]				
0x4000001C	GPIO2_IN[7:0]	–	–	–	GPIO2_IN[4:0]				
0x40000020	GPIO2_PUD_DIS[7:0]	–	–	–	GPIO2_PUD_DIS[4:0]				
0x40000024	GPIO2_PUD_SEL[7:0]	–	–	–	GPIO2_PUD_SEL[4:0]				
0x40000028	GPIO2_ALT_FUNC[15:8]	POL_IRQ1	–	–	–	–	–	–	–
	GPIO2_ALT_FUNC[7:0]	GPIO2_MCLKIN_EN	GPIO2_IRQ1	–	GPIO2_SPI	–	–	–	GPIO2_CS1
0x4000002C	GPIO2_EN[7:0]	–	–	–	GPIO2_OUT_EN[4:0]				

Register Details

GPIO1_OUT (0x40000000)

Set the logic state of the Bank 1 GPIO pins (GPIO1_x) configured as outputs.

BIT	7	6	5	4	3	2	1	0
Field	–	GPIO1_OUT[6:0]						
Reset	–	0x00						
Access Type	–	Write, Read						

BITFIELD	BITS	DESCRIPTION	DECODE
GPIO1_OUT	6:0	**GPIO1_x Output Logic State** Set these bits to program the logic of a GPIO1_x pin when that GPIO is configured as an output (GPIO1_x_EN = 1). These bits are ignored when GPIO1_x_EN = 0. GPIO1_x bits are assigned as follows: **BIT 6** / **BIT 5** / **BIT 4** / **BIT 3** / **BIT 2** / **BIT 1** / **BIT 0** GPIO1_6 / GPIO1_5 / GPIO1_4 / GPIO1_3 / GPIO1_2 / GPIO1_1 / GPIO1_0	0 = GPIO1_x is low 1 = GPIO1_x is high

GPIO1_IN (0x40000004)

Logic state of the Bank 1 GPIO pins (GPIO1_x).

BIT	7	6	5	4	3	2	1	0
Field	–	GPIO1_IN[6:0]						
Reset	–							
Access Type	–	Read Only						

BITFIELD	BITS	DESCRIPTION	DECODE
GPIO1_IN	6:0	**GPIO1_x Input Logic State** These bits reflect the logic state of the GPIO1_x pins. GPIO1_x bits are assigned as follows: **BIT 6** / **BIT 5** / **BIT 4** / **BIT 3** / **BIT 2** / **BIT 1** / **BIT 0** GPIO1_6 / GPIO1_5 / GPIO1_4 / GPIO1_3 / GPIO1_2 / GPIO1_1 / GPIO1_0	0 = GPIO1_x is low 1 = GPIO1_x is high

GPIO1_PUD_DIS (0x40000008)

Enable/disable the internal pull-up or pull-down for each Bank 1 GPIO pin (GPIO1_x).

BIT	7	6	5	4	3	2	1	0
Field	–	GPIO1_PUD_DIS[6:0]						

Reset	–	0x00
Access Type	–	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
GPIO1_PUD_DIS	6:0	**GPIO1_x Internal Pull-up/pull-down disable/enable.** Set the GPIO1_PUD_DIS bit to enable the internal pull-up or pull-down for a given GPIO1_x pin. Set the GPIO1_PUD_SEL bit to select the pull-up or pull-down for the GPIO1_x pin.	0 = Pull-up/-down is enabled 1 = Pull-up/-down is disabled

GPIO1_PUD_SEL (0x4000000C)

Connect the internal pull-up or pull-down for each Bank 1 GPIO pin (GPIO1_x).

BIT	7	6	5	4	3	2	1	0
Field	–	GPIO1_PUD_SEL[6:0]						
Reset	–	0x00						
Access Type	–	Write, Read						

BITFIELD	BITS	DESCRIPTION	DECODE
GPIO1_PUD_SEL	6:0	**GPIO1_x Internal pull-up or pull-down select.** When GPIO1_PUD_DIS bit is 0, set this bit to 1 to connect the internal pull-up on the associated GPIO1_x pin. Set this bit to 0 to connect the internal pull-down.	0 = Internal pull-down connected 1 = Internal pull-up connected

GPIO1_ALT_FUNC (0x40000010)

Enable alternate functions for each Bank 1 GPIO (GPIO1_x).

BIT	15	14	13	12	11	10	9	8
Field	POL_IRQ0	–	–	–	–	–	–	–
Reset	0x0	–	–	–	–	–	–	–
Access Type	Write, Read	–	–	–	–	–	–	–
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	–	GPIO1_IRQ0_EN	GPIO1_PWM	GPIO1_I2C	GPIO1_PDOUT2	GPIO1_PDOUT1	GPIO1_PDIN2	GPIO1_PDIN1
Reset	–	0b0	0x0	0x0	0x0	0x0	0x0	0x0

Access Type	–	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
POL_IRQ0	15	**IRQ0 Polarity Select** Set this bit to define the polarity of GPIO1_0 when it is configured as an IRQ0 input. See the Cortex-M0 developer documentation on the Arm website for more information.	0 = IRQ0 is inverted 1 = IRQ0 is not inverted
GPIO1_IRQ0_EN	6	**GPIO1 IRQ0 Alternate Function Enable** Set this bit to configure GPIO1_0 as the IRQ0 interrupt. Use the POL_IRQ0 bit to define the polarity when the IRQ0 input is enabled. *Other configuration bits for GPIO1_0 (example, GPIO1_0_OUT, GPIO1_0_EN) are ignored when this bit is set.*	0 = GPIO1_0 is not configured as the IRQ0 input 1 = GPIO1_0 is configured as the IRQ0 input
GPIO1_PWM	5	**GPIO1 PWM Alternate Function Enable** Set this bit to configure GPIO1_3 as a PWM output. *Other configuration bits for GPIO1_3 (example, GPIO1_3_OUT, GPIO1_3_EN) are ignored when this bit is set.* See the Enhanced Timer section for more information.	0 = GPIO1_3 is not configured as a PWM output 1 = GPIO1_3 is configured as a PWM output
GPIO1_I2C	4	**GPIO1 I²C Alternate Function Enable** Set this bit to enable the I²C host controller core. I²C signals are as follows: SCL = GPIO1_1 SDA = GPIO1_2 Internal pull-ups for GPIO1_1 and GPIO1_2 (GPIO1_x_PUD_DIS and GPIO1_x_PUD_SEL registers) can be enabled when the I²C core is enabled. *Other configuration bits for GPIO1_1 and GPIO1_2 (example, GPIO1_x_OUT, GPIO1_x_EN) are ignored when this bit is set.* See the I²C Host Controller (GPIO1_1, GPIO1_2) for more information.	0 = I²C core is disabled 1 = I²C core is enabled
GPIO1_PDOUT2	3	**GPIO1 PDOUT2 Alternate Function Enable** Set this bit to configure GPIO1_3 as a PDOUT output. When GPIO1_PDOUT1 = 0 and GPIO1_PDOUT2 = 1, GPIO1_3 is configured as an output and is driven according to the LSB of each received PDOUT byte. When both GPIO1_PDOUT1 = 1 and GPIO1_PDOUT2 = 1, GPIO1_6 = PDOUT bit 0 (LSB) and GPIO1_3 = PDOUT bit 1. *Other configuration bits for GPIO1_3 (example, GPIO1_3_OUT, GPIO1_3_EN) are ignored when this bit is set.* See the PDOUT1, PDOUT2 (GPIO1_3, GPIO1_6) section for more information.	0 = GPIO1_3 is not configured as PDOUT2 data bit 1 = GPIO1_3 is configured as PDOUT2 data bit
GPIO1_PDOUT1	2	**GPIO1 PDOUT1 Alternate Function Enable** Set this bit to configure GPIO1_6 as a PDOUT output. When GPIO1_PDOUT1 = 1, GPIO1_6 is	0 = GPIO is not configured as PDOUT1 data bit 1 = GPIO1_6 is configured bit 0 of the PDOUT byte received from the IO-Link master

BITFIELD	BITS	DESCRIPTION	DECODE
		configured as an output and is driven accroding to the LSB of each received PDOUT byte. When both GPIO1_PDOUT1 = 1 and GPIO1_PDOUT2 = 1, GPIO1_6 = PDOUT bit 0 and GPIO1_3 = PDOUT bit 1. *Other configuration bits for GPIO1_6 (example, GPIO1_6_OUT, GPIO1_6_EN) are ignored when this bit is set.* See the PDOUT1, PDOUT2 (GPIO1_3, GPIO1_6) section for more information.	
GPIO1_PDIN2	1	**GPIO1 PDIN2 Alternate Function Enable** Set this bit to configure GPIO1_4 as a PDIN input logic bit. When GPIO1_PDIN2 = 1, GPIO1_4 is configured as an input and sets the LSB of the first PDIN byte sent to the IO-Link master. When both GPIO1_PDIN1 = 1 and GPIO1_PDIN2 = 1, GPIO1_4 sets bit 1, and GPIO1_5 sets bit 0, of the first PDIN byte sent to the IO-Link master. *Other configuration bits for GPIO1_4 (example, GPIO1_4_OUT, GPIO1_4_EN) are ignored when this bit is set.* See the PDIN1, PDIN2 (GPIO1_4, GPIO1_5) section for more information.	0 = GPIO1_4 is not configured as PDIN2 1 = GPIO1_4 is configured as an input and its sets the LSB of the PDIN bytes sent to the IO-Link master
GPIO1_PDIN1	0	**GPIO1 PDIN1 Alternate Function Enable** Set this bit to configure GPIO1_5 as a PDIN input. When GPIO1_PDIN1 = 1, GPIO1_5 is configured as an input and sets the LSB of the PDIN bytes sent to the IO-Link master. When both GPIO1_PDIN1 = 1 and GPIO1_PDIN2 = 1, GPIO1_4 sets bit 1 of the PDIN and GPIO1_5 sets bit 0] of the PDIN byte sent to the IO-Link master. *Other configuration bits for GPIO1_5 (example, GPIO1_5_OUT, GPIO1_5_EN) are ignored when this bit is set.* See the PDIN1, PDIN2 (GPIO1_4, GPIO1_5) section for more information.	0 = GPIO is not configured as PDIN1 1 = GPIO1_5 is configured as an input and its sets the LSB of the PDIN bytes sent to the IO-Link master

GPIO1_EN (0x40000014)

Enable Bank 1 GPIO pins (GPIO1_x) as outputs.

BIT	7	6	5	4	3	2	1	0
Field	–	GPIO1_OUT_EN[6:0]						
Reset	–	0x00						
Access Type	–	Write, Read						

BITFIELD	BITS	DESCRIPTION	DECODE
GPIO1_OUT_EN	6:0	**GPIO1_x Output Enable** Set these bits to configure a GPIO1_x pin as an output. GPIO1_x is a push-pull output when GPIO1_x_EN = 1. Set the logic state of the output by setting the	0 = GPIO1_x is an input 1 = GPIO1_x is a push-pull output

BITFIELD	BITS	DESCRIPTION	DECODE
		GPIO1_x_OUT bit in the GPIO1_OUT register. GPIO1_x bits are assigned as follows: BIT 6: GPIO1_6 | BIT 5: GPIO1_5 | BIT 4: GPIO1_4 | BIT 3: GPIO1_3 | BIT 2: GPIO1_2 | BIT 1: GPIO1_1 | BIT 0: GPIO1_0	

GPIO2_OUT (0x40000018)

Set the logic state of the Bank 2 GPIO pins (GPIO2_x) configured as outputs.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	GPIO2_OUT[4:0]				
Reset	–	–	–	0x00				
Access Type	–	–	–	Write, Read				

BITFIELD	BITS	DESCRIPTION	DECODE
GPIO2_OUT	4:0	**GPIO2_x Output Logic State** Set these bits to program the logic of a GPIO2_x pin when that GPIO is configured as an output (GPIO2_x_EN = 1). These bits are ignored when GPIO2_x_EN = 0. GPIO2_x bits are assigned as follows: BIT 4: GPIO2_4 | BIT 3: GPIO2_3 | BIT 2: GPIO2_2 | BIT 1: GPIO2_1 | BIT 0: GPIO2_0	0 = GPIO2_x is low 1 = GPIO2_x is high

GPIO2_IN (0x4000001C)

Logic state of the Bank 2 GPIO pins (GPIO2_x)

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	GPIO2_IN[4:0]				
Reset	–	–	–					
Access Type	–	–	–	Read Only				

BITFIELD	BITS	DESCRIPTION	DECODE
GPIO2_IN	4:0	**GPIO2_x Input Logic State** These bits reflect the logic state of the GPIO2_x pins. GPIO2_x bits are assigned as follows:	0 = GPIO2_x is low 1 = GPIO2_x is high

BITFIELD	BITS	DESCRIPTION	DECODE
		BIT 4 / BIT 3 / BIT 2 / BIT 1 / BIT 0 GPIO2_4 / GPIO2_3 / GPIO2_2 / GPIO2_1 / GPIO2_0	

GPIO2_PUD_DIS (0x40000020)

Enable/disable the internal pull-up or pull-down for each Bank 2 GPIO pin (GPIO2_x).

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	GPIO2_PUD_DIS[4:0]				
Reset	–	–	–	0x00				
Access Type	–	–	–	Write, Read				

BITFIELD	BITS	DESCRIPTION	DECODE
GPIO2_PUD_DIS	4:0	**GPIO2_x Internal Pull-up/-down Disable/Enable** Set the GPIO2_PUD_DIS bit to enable the internal pull-up or pull-down for a given GPIO2_x pin. Set the GPIO2_PUD_SEL bit to select the pull-up or pull-down for the GPIO2_x pin.	0 = Pull-up/-down is enabled 1 = Pull-up/-down is disabled

GPIO2_PUD_SEL (0x40000024)

Connect the internal pull-up or pull-down for each Bank 2 GPIO pin (GPIO2_x).

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	GPIO2_PUD_SEL[4:0]				
Reset	–	–	–	0x00				
Access Type	–	–	–	Write, Read				

BITFIELD	BITS	DESCRIPTION	DECODE
GPIO2_PUD_SEL	4:0	**GPIO2_x Internal Pull-up or Pull-down Select** When GPIO2_PUD_DIS bit is 0, set this bit to 1 to connect the internal pull-up on the associated GPIO2_x pin. Set this bit to 0 to connect the internal pull-down.	0 = Internal pull-down connected 1 = Internal pull-up connected

GPIO2_ALT_FUNC (0x40000028)

Enable alternate functions for each Bank 2 GPIO (GPIO2_x).

BIT	15	14	13	12	11	10	9	8
Field	POL_IRQ1	–	–	–	–	–	–	–
Reset	0x0	–	–	–	–	–	–	–
Access Type	Write, Read	–	–	–	–	–	–	–
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	GPIO2_MCLKIN_EN	GPIO2_IRQ1	–	GPIO2_SPI	–	–	–	GPIO2_CS1
Reset	0x0	0x0	–	0x0	–	–	–	0x0
Access Type	Read Only	Write, Read	–	Write, Read	–	–	–	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
POL_IRQ1	15	**IRQ1 Polarity Select** Set this bit to define the polarity of GPIO2_0 when it is configured as an IRQ1 input. See the Cortex-M0 developer documentation on the Arm website for more information.	0 = IRQ1 is inverted 1 = IRQ1 is not inverted
GPIO2_MCLKIN_EN	7	**GPIO2 MCLK Input Alternate Function Enable** Set this bit to enable GPIO2_0 as an MCLK input. Set the EXT_NOT_POSC bit in the CLK_CTRL_SOURCE register and apply an external clock to GPIO2_0 when this bit is set. *Other configuration bits for GPIO2_0 (example, GPIO2_0_OUT, GPIO2_0_EN) are ignored when this bit is set.* See the Clock Control section and CLK_CTRL registers for more information.	0 = GPIO2_0 is not configured as an MCLK input 1 = GPIO2_0 is configured as an MCLK input
GPIO2_IRQ1	6	**GPIO2 IRQ1 Alternate Function Enable** Set this bit to configure GPIO2_0 as the IRQ1 interrupt. Use the POL_IRQ1 bit to define the polarity when the IRQ1 input is enabled. *Other configuration bits for GPIO2_0 (e.g. GPIO2_0_OUT, GPIO2_0_EN) are ignored when this bit is set.*	0 = GPIO2_0 is not configured as the IRQ1 input 1 = GPIO2_0 is configured as the IRQ1 input
GPIO2_SPI	4	**GPIO2 SPI Alternate Function Enable** Set this bit to enable the SPI host controller core. When the SPI core is enabled, the SPI signals are as follows: CS0 = GPIO2_1 SCLK = GPIO2_2 SDO = GPIO2_4 SDI = GPIO2_3 Internal pull-ups for GPIO2_1 - GPIO2_4 (GPIO2_x_PUD_DIS and GPIO2_x_PUD_SEL registers) can be enabled when the SPI host controller core is enabled. *Other configuration bits for GPIO2_1 - GPIO2_4 (example, GPIO2_x_OUT, GPIO2_x_EN) are ignored when this bit is set.*	0 = SPI host core is disabled 1 = SPI core is enabled

BITFIELD	BITS	DESCRIPTION	DECODE
		See the SPI Host Controller (GPIO2_1, GPIO2_4) section for more information.	
GPIO2_CS1	0	**GPIO2 CS1 Alternate Function Enable** Set this bit to configure GPIO2_0 as a chip-select output (CS1) when the SPI host controller core is enabled. *Other configuration bits for GPIO2_0 (example, GPIO2_x_OUT, GPIO2_x_EN) are ignored when this bit is set.* See the SPI Host Controller (GPIO2_1, GPIO2_4) section for more information.	0 = GPIO2_0 is not configured as CS1 output 1 = GPIO2_0 is configured as CS1 output

GPIO2_EN (0x4000002C)

Enable Bank 2 GPIO pins (GPIO2_x) as outputs.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	GPIO2_OUT_EN[4:0]				
Reset	–	–	–	0x00				
Access Type	–	–	–	Write, Read				

BITFIELD	BITS	DESCRIPTION	DECODE
GPIO2_OUT_EN	4:0	**GPIO2_x Output Enable** Set these bits to a GPIO2_x pin as an output. GPIO2_x is a push-pull output when GPIO2_x_EN = 1. Set the logic state of the output by setting the GPIO2_x_OUT bit in the GPIO2_OUT register. GPIO2_x bits are assigned as follows: **BIT 4** / **BIT 3** / **BIT 2** / **BIT 1** / **BIT 0** GPIO2_4 / GPIO2_3 / GPIO2_2 / GPIO2_1 / GPIO2_0	0 = GPIO2_x is an input 1 = GPIO2_x is a push-pull output

I2C HOST REGISTERS

ADDRESS	NAME	MSB							LSB
I2C HOST REGISTERS									
0x40010000	I2C_CLK_PRESCALE[15:8]	I2C_SCL_PRESCALE[15:8]							
	I2C_CLK_PRESCALE[7:0]	I2C_SCL_PRESCALE[7:0]							
0x40010004	I2C_CTRL[7:0]	I2C_EN	I2C_IRQ_EN	–	–	–	–	–	I2C_GLB_SYNC_RST

ADDRESS	NAME	MSB							LSB
0x40010008	I2C_RX_FIFO[7:0]	I2C_R_FIFO_DATA[7:0]							
0x4001000C	I2C_STATUS[7:0]	I2C_RX_ACK_STATUS	I2C_BUSY_STATUS	I2C_ARBL_STATUS	_	_	_	I2C_TIP_STATUS	I2C_IRQ_STATUS
0x40010010	I2C_TX_FIFO[7:0]	I2C_W_FIFO_DATA[7:0]							
0x40010014	I2C_CMD[7:0]	I2C_START	I2C_STOP	I2C_READ	I2C_WRITE	I2C_ACK_RX_DATA	_	_	I2C_INT_ACK_CMD
0x40010018	I2C_RX_NUM_BYTES[7:0]	I2C_RX_NUM_BYTE[7:0]							
0x4001001C	I2C_CLR_RX_FIFO[7:0]	_	_	_	_	_	_	_	I2C_CLR_RX_FIFO
0x40010020	I2C_READ_ADD_BYTES[7:0]	I2C_RD_ADD_BYTE[7:0]							

Register Details

I2C_CLK_PRESCALE (0x40010000)

Program the I²C clock (SCL) frequency.

BIT	15	14	13	12	11	10	9	8
Field	I2C_SCL_PRESCALE[15:8]							
Reset								
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	I2C_SCL_PRESCALE[7:0]							
Reset								
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION	DECODE
I2C_SCL_PRESCALE	15:0	**I²C Clock (SCL) Prescale Value** Program this register to set the I²C SCL frequency. See the I²C Host Controller (GPIO1_1, GPIO1_2) section for more information.	I2C_SCL_PRESCALE[7:0] = [f_{HCLK} / (5 x f_{SCL})] -1 where f_{HCLK} is the system clock frequency and f_{SCL} is the desired I²C clock frequency.

I2C_CTRL (0x40010004)

Enable the I²C host controller, interrupts, and reset the I²C engine and FIFOs.

BIT	7	6	5	4	3	2	1	0
Field	I2C_EN	I2C_IRQ_EN	–	–	–	–	–	I2C_GLB_SYNC_RST
Reset			–	–	–	–	–	
Access Type	Write, Read	Write, Read	–	–	–	–	–	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
I2C_EN	7	**I²C Host Controller Enable** Set this bit to enable the I²C host engine.	0 = I²C host controller disabled 1 = I²C host controller enabled
I2C_IRQ_EN	6	**I²C Interrupt Enable** Set this bit to enable the I²C interrupt. See the Interrupt Controller section for more information.	0 = I²C interrupts are disabled. 1 = I²C interrupts are enabled.
I2C_GLB_SYNC_RST	0	**I²C Host Controller and FIFO Synchronous Reset** Set this bit to reset the I²C engine and both the I²C TX and RX FIFOs. This bit is not automatically cleared. Write a 0 to this bit to clear it.	0 = No reset 1 = I²C engine and I²C TX and RX FIFOs are reset

I2C_RX_FIFO (0x40010008)

Read/Receive (RX) FIFO

BIT	7	6	5	4	3	2	1	0
Field	I2C_R_FIFO_DATA[7:0]							
Reset								
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
I2C_R_FIFO_DATA	7:0	**32-Byte I²C Read (RX) FIFO Data** When an I²C read transmission is complete, data is transferred to the 32-byte RX FIFO and can be read from the I2C_RX_FIFO register. This FIFO is not automatically cleared after a read. Write a 1 to the I2C_CLR_RX_FIFO bit to clear this FIFO. See the I²C Host Controller (GPIO1_1, GPIO1_2) section for more information.

I2C_STATUS (0x4001000C)

I²C status flags

BIT	7	6	5	4	3	2	1	0

Field	I2C_RX_ACK_STATUS	I2C_BUSY_STATUS	I2C_ARBL_STATUS	–	–	–	I2C_TIP_STATUS	I2C_IRQ_STATUS
Reset				–	–	–		
Access Type	Read Only	Read Only	Read Only	–	–	–	Read Only	Read Only

BITFIELD	BITS	DESCRIPTION	DECODE
I2C_RX_ACK_STATUS	7	**I²C RX Acknowledge Status** This bit is set when an ACK is received in the last I²C communication.	0 **=** RX acknowledge received 1 = No acknowledge received
I2C_BUSY_STATUS	6	**I²C Busy Status Indicator** This bit is set following the START pulse when the I²C host controller is busy (transmitting or receiving).	0 = I²C bus is not busy 1 = I²C bus is busy
I2C_ARBL_STATUS	5	**Arbitration Lost Status Indicator** This bit it is set when an arbitration lost error occurred for the last I²C transaction. This bit is reset to 0 when a new I²C transaction is started. See the I²C Host Controller (GPIO1_1, GPIO1_2) section for more information.	0 = The current I²C transation has not produced an abritration lost condition. 1 = The current I²C transation has produced an abritration lost condition.
I2C_TIP_STATUS	1	**I²C Transfer in Progress Status Indicator** This bit is set when an I²C read/write command is received and transaction is in progress. This bit is automatically cleared at the completion of the last data transaction.	0 = No data transfer in process 1 = Data transfer in process. I²C controller is busy.
I2C_IRQ_STATUS	0	**I²C Interrupt Status Indicator** This bit is set when an I²C interrupt occurs (example, transaction done or arbitration lost). See the I²C Host Controller (GPIO1_1, GPIO1_2) section for more information. Set the INT_ACK_CMD bit in the I2C_CMD register (INT_ACK_CMD = 1) to clear this bit.	0 = No interrupt occurred 1 = Interrupt occurred

I2C_TX_FIFO (0x40010010)

Write/Transmit (TX) FIFO

BIT	7	6	5	4	3	2	1	0
Field	I2C_W_FIFO_DATA[7:0]							
Reset								
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
I2C_W_FIFO_DATA	7:0	**32-Byte I²C Transmit (TX) FIFO Data** Write data to be transmitted in an I²C transaction to this register. Send a write command to begin transmission. See the I²C Host section for more information.

I2C_CMD (0x40010014)

Program the START, STOP, and clear I²C interrupt commands.

BIT	7	6	5	4	3	2	1	0
Field	I2C_START	I2C_STOP	I2C_READ	I2C_WRITE	I2C_ACK_RX_DATA	–	–	I2C_INT_ACK_CMD
Reset						–	–	
Access Type	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	–	–	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
I2C_START	7	**I²C START (S) Command Bit** Set this bit to generate a start command bit for I²C communication.	0 = I²C communication is not started 1 = I²C start (S) bit is sent
I2C_STOP	6	**I²C STOP (P) Command Bit** Set this bit to generate an automatic stop condition during I²C communication.	0 = I²C communication stop condition is not automatically generated. 1 = I²C communication stop condition is automatically generated.
I2C_READ	5	**I²C Read Command Enable** Set this bit to enable an I²C read.	0 = I²C read command not enabled 1 = Enable I²C communication for read
I2C_WRITE	4	**I²C Write Command Enable** Set this bit to enable an I²C write.	0 = I²C write command not enabled 1 = Enable I²C communication for write
I2C_ACK_RX_DATA	3	**I²C Manual Not-Acknowledge (NACK) for Received Data** A NACK is automatically generated at the end of burst read, but transmission can be ended earlier by setting this bit. Set this bit to 1 after a byte is transmitted to generate a NACK for a byte received from the I²C device. This bit is automatically cleared at the end of the transfer.	0 = No action. NACK is automatically generated at the end of a burst read. 1 = Generate NACK at end of next received byte.
I2C_INT_ACK_CMD	0	**I²C Interrupt Acknowledge and Clear Command** Set this bit to clear the IRQ_FLAG_STATUS bit in the I2C_STATUS register. This bit is automatically cleared the next time that IRQ_FLAG_STATUS = 1.	0 = No action 1 = IRQ_FLAG_STATUS bit is cleared

I2C_RX_NUM_BYTES (0x40010018)

Enter the number of bytes to be received from an I²C read.

BIT	7	6	5	4	3	2	1	0
Field	I2C_RX_NUM_BYTE[7:0]							
Reset								
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
I2C_RX_NUM_BYTE	7:0	**I²C Receive Data Size** Program this register to reflect the number of bytes to be received during the next read operation. Ensure this value does not exceed the 32-byte RX FIFO length.

I2C_CLR_RX_FIFO (0x4001001C)

Clear the read/receiver (RX) FIFO.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	–	I2C_CLR_RX_FIFO
Reset	–	–	–	–	–	–	–	
Access Type	–	–	–	–	–	–	–	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
I2C_CLR_RX_FIFO	0	**Clear I²C RX FIFO** Set this bit to clear the I²C RX FIFO after a read operation is completed.	0 = RX FIFO is not cleared from last read 1 = Clear RX FIFO

I2C_READ_ADD_BYTES (0x40010020)

Set the number of address bytes transmitted during an I²C read transaction.

BIT	7	6	5	4	3	2	1	0
Field	I2C_RD_ADD_BYTE[7:0]							
Reset								
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
I2C_RD_ADD_BYTE	7:0	**I²C Address Bytes Count** Program this register to reflect the number of bytes transmitted as address bytes during a read operation.

CQ_CFG REGISTER

ADDRESS	NAME	MSB							LSB
CQ_CFG REGISTER									
0x40070048	CQ_CFG_REG[31:24]	CQ_CFG[31:24]							
	CQ_CFG_REG[23:16]	CQ_CFG[23:16]							

ADDRESS	NAME	MSB							LSB
	CQ_CFG_REG[15:8]	CQ_CFG[15:8]							
	CQ_CFG_REG[7:0]	CQ_CFG[7:0]							

Register Details

CQ_CFG_REG (0x40070048)

BIT	31	30	29	28	27	26	25	24
Field	CQ_CFG[31:24]							
Reset								
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	CQ_CFG[23:16]							
Reset								
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	CQ_CFG[15:8]							
Reset								
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	CQ_CFG[7:0]							
Reset								
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION	DECODE
CQ_CFG	31:0	**Finalize the CQ Configuration Register** After the IO-Link configuraiton bits are programmed, set this register to 0x04 to finalize the C/Q configuration	0x04 = Finalize the C/Q configuration All other settings are not allowed

SPI HOST REGISTERS

ADDRESS	NAME	MSB							LSB
SPI1 HOST REGISTERS									
0x48009000	SPI1_STATUS[7:0]	–	–	–	–	SPI1_BUSY	–	SPI1_STARTED	SPI1_DONE
0x48009004	SPI1_GEN_CONFIG[7:0]	–	–	–	SPI1_START_PULSE	–	–	–	SPI1_ENABLE
0x48009008	SPI1_SCLK_CONFIG0[23:16]	SPI_PAUSE_TIME[7:0]							
	SPI1_SCLK_CONFIG0[15:8]	CS_SETTLE_TIM[7:0]							
	SPI1_SCLK_CONFIG0[7:0]	CLK_DIV[7:0]							
0x48009010	SPI1_TRANSFER_CONFIG[23:16]	–	–	–	–	–	–	–	–
	SPI1_TRANSFER_CONFIG[15:8]	–	SPI_DATA_SIZE[6:0]						
	SPI1_TRANSFER_CONFIG[7:0]	–	–	–	CS1_ENABLE	CS0_ENABLE	CS_POL	SCLK_POL	SCLK_PHASE
0x4800902C	SP1_RD_BUFF0[31:24]	RD_BUFFER0[31:24]							
	SP1_RD_BUFF0[23:16]	RD_BUFFER0[23:16]							
	SP1_RD_BUFF0[15:8]	RD_BUFFER0[15:8]							
	SP1_RD_BUFF0[7:0]	RD_BUFFER0[7:0]							
0x48009030	SP1_RD_BUFF1[31:24]	RD_BUFFER1[31:24]							
	SP1_RD_BUFF1[23:16]	RD_BUFFER1[23:16]							
	SP1_RD_BUFF1[15:8]	RD_BUFFER1[15:8]							
	SP1_RD_BUFF1[7:0]	RD_BUFFER1[7:0]							
0x48009034	SP1_RD_BUFF2[31:24]	RD_BUFFER2[31:24]							
	SP1_RD_BUFF2[23:16]	RD_BUFFER2[23:16]							
	SP1_RD_BUFF2[15:8]	RD_BUFFER2[15:8]							
	SP1_RD_BUFF2[7:0]	RD_BUFFER2[7:0]							
0x48009038	SP1_RD_BUFF3[31:24]	RD_BUFFER3[31:24]							
	SP1_RD_BUFF3[23:16]	RD_BUFFER3[23:16]							
	SP1_RD_BUFF3[15:8]	RD_BUFFER3[15:8]							
	SP1_RD_BUFF3[7:0]	RD_BUFFER3[7:0]							

ADDRESS	NAME	MSB							LSB
0x4800903C	SP1_WR_BUFF0[31:24]	WR_BUFFER0[31:24]							
	SP1_WR_BUFF0[23:16]	WR_BUFFER0[23:16]							
	SP1_WR_BUFF0[15:8]	WR_BUFFER0[15:8]							
	SP1_WR_BUFF0[7:0]	WR_BUFFER0[7:0]							
0x48009040	SP1_WR_BUFF1[31:24]	WR_BUFFER1[31:24]							
	SP1_WR_BUFF1[23:16]	WR_BUFFER1[23:16]							
	SP1_WR_BUFF1[15:8]	WR_BUFFER1[15:8]							
	SP1_WR_BUFF1[7:0]	WR_BUFFER1[7:0]							
0x48009044	SP1_WR_BUFF2[31:24]	WR_BUFFER2[31:24]							
	SP1_WR_BUFF2[23:16]	WR_BUFFER2[23:16]							
	SP1_WR_BUFF2[15:8]	WR_BUFFER2[15:8]							
	SP1_WR_BUFF2[7:0]	WR_BUFFER2[7:0]							
0x48009048	SP1_WR_BUFF3[31:24]	WR_BUFFER3[31:24]							
	SP1_WR_BUFF3[23:16]	WR_BUFFER3[23:16]							
	SP1_WR_BUFF3[15:8]	WR_BUFFER3[15:8]							
	SP1_WR_BUFF3[7:0]	WR_BUFFER3[7:0]							

Register Details

SPI1_STATUS (0x48009000)

SPI host controller BUSY, START, and DONE status indicators.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	SPI1_BUSY	–	SPI1_STARTED	SPI1_DONE
Reset	–	–	–	–	0b0	–	0b0	
Access Type	–	–	–	–	Read Only	–	Read Only	Read Only

BITFIELD	BITS	DESCRIPTION	DECODE
SPI1_BUSY	3	**SPI BUSY1 Status Indicator Bit** This bit is set when the SPI is busy.	0 = SPI interface is not busy 1 = SPI interface is busy

BITFIELD	BITS	DESCRIPTION	DECODE
SPI1_STARTED	1	**SPI Transaction Started Status Indicator Bit** This bit is set when an SPI transaction has started, but is not yet complete. This bit is cleared when the current SPI transaction is complete.	0 = SPI transaction is currently in process or no SPI transaction has occurred yet. 1 = SPI transaction has started.
SPI1_DONE	0	**SPI Transaction Completed Status Bit** This bit is set when an SPI communication is complete and no SPI transaction is currently active. This bit is automatically cleared when a new SPI transaction is started.	0 = SPI transaction is currently in process or no SPI transaction has occurred yet. 1 = An SPI transaction has been completed and no SPI transaction is active.

SPI1_GEN_CONFIG (0x48009004)

Enable the SPI host controller and select the SPI configuration.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	SPI1_START_PULSE	–	–	–	SPI1_ENABLE
Reset	–	–	–	0b0	–	–	–	0b0
Access Type	–	–	–	Write, Read	–	–	–	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
SPI1_START_PULSE	4	**SPI Automatic Start Pulse Enable Bit** Set this bit to 1 to enable the SPI host to automatically generate the START pulse at tbe beginning of the next SPI transaction.	0 = Automatic START pulse not enabled. 1 = Automatic START pulse in SPI communication enabled.
SPI1_ENABLE	0	**SPI Enable Bit** Set this bit to enable the SPI host controller core.	0 = SPI interface is not enabled. 1 = SPI interface is enabled.

SPI1_SCLK_CONFIG0 (0x48009008)

Set the SPI clock (SCLK) frequency.

BIT	23	22	21	20	19	18	17	16
Field	SPI_PAUSE_TIME[7:0]							
Reset	0x00							
Access Type	Write, Read							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	CS_SETTLE_TIM[7:0]							
Reset	0x0a							

Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	CLK_DIV[7:0]							
Reset	0x0a							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION	DECODE
SPI_PAUSE_TIME	23:16	SPI Communication Pause Time Set these bits to program the pause time between two SPI cycles, and between when CS asserts and the SCLK begins. t_{SPI_PAUSE} = SPI_PAUSE_TIME value x 16 clock cycles	
CS_SETTLE_TIM	15:8	SPI Chip Select Settle Time. Program this register for the desired settle time between when CS asserts and SCLK begins switching.	SETTLE TIME = (CS_SETTLE_TIM + 1) * T_System + 0.5*T_SPI
CLK_DIV	7:0	**SPI Clock Configuration** Program this register to set the SPI clock frequency.	where f_{HCLK} is the sytsem clock frequency and f_{SCLK} is the desired SPI clock frequency. f_{SCLK_MAX} = f_{HCLK} / 4.

SPI1_TRANSFER_CONFIG (0x48009010)

Configure the SPI engine.

BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	SPI_DATA_SIZE[6:0]						
Reset	–	0b101000						
Access Type	–	Write, Read						
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

Field	–	–	–	CS1_ENABLE	CS0_ENABLE	CS_POL	SCLK_POL	SCLK_PHASE
Reset	–	–	–	0b0	0b0	0b1	0b1	0b1
Access Type	–	–	–	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
SPI_DATA_SIZE	14:8	**SPI Data Size** Program these bits to set the total number of bits to transfer in a single transaction.	Calculate the number of bits as: Bits = (Number of bytes x 8) - 1
CS1_ENABLE	4	**SPI Chip Select 1 Enable Bit** Set this bit to use GPIO2_0 as a CS1 output. The SPI engine controls CS1. **NOTE:** GPIO2_0 must be configured as CS1.	0 = CS1 is disabled. 1 = CS1 is enabled.
CS0_ENABLE	3	**SPI Chip Select 0 Enable Bit** Set this bit to use GPIO2_1 as a CS0 output. The SPI engine controls CS0.	0 = CS0 is always inactive. 1 = CS0 is enabled.
CS_POL	2	**SPI Chip Select Polarity** Set this bit to configure the polarity for the selected chip-select (CS) output(s).	0 = CS output(s) are active-low. 1 = CS output(s) are active-high.
SCLK_POL	1	**SPI SCLK Polarity Select** Set this bit to set the SCLK polarity during the idle state.	0 = SCLK is logic low in the idle state. 1 = SCLK is logic high in the idle state.
SCLK_PHASE	0	**SPI SCLK Phase** Set this bit to set the SCLK phased used to sample and/or shift the data.	SCLK_POL = 0 and SCLK_PHASE = 0: data is sampled on rising edge and shifted out on the falling edge of the clock. SCLK_POL = 0 and SCLK_PHASE = 1: data is sampled on falling edge and shifted out on the rising edge of the clock. SCLK_POL = 1 and SCLK_PHASE = 0: data is sampled on falling edge and shifted out on the falling edge of the clock. SCLK_POL = 1 and SCLK_PHASE = 1: data is sampled on rising edge and shifted out on the falling edge of the clock.

SP1_RD_BUFF0 (0x4800902C)

SPI host controller read buffer 0

BIT	31	30	29	28	27	26	25	24
Field	RD_BUFFER0[31:24]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16

Field	RD_BUFFER0[23:16]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	RD_BUFFER0[15:8]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	RD_BUFFER0[7:0]							
Reset	0x00000000							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
RD_BUFFER0	31:0	**SPI Read Buffer 0** See the SPI Host Controller (GPIO2_0 - GPIO2_4) section for more information.

SP1_RD_BUFF1 (0x48009030)

SPI host controller read buffer 1

BIT	31	30	29	28	27	26	25	24
Field	RD_BUFFER1[31:24]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	RD_BUFFER1[23:16]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**

Field	RD_BUFFER1[15:8]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	RD_BUFFER1[7:0]							
Reset	0x00000000							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
RD_BUFFER1	31:0	**SPI Read Buffer 1** See the SPI Host Controller (GPIO2_0 - GPIO2_4) section for more information.

SP1_RD_BUFF2 (0x48009034)

SPI host controller read buffer 2

BIT	31	30	29	28	27	26	25	24
Field	RD_BUFFER2[31:24]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	RD_BUFFER2[23:16]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	RD_BUFFER2[15:8]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0

Field	RD_BUFFER2[7:0]
Reset	0x00000000
Access Type	Write, Read

BITFIELD	BITS	DESCRIPTION
RD_BUFFER2	31:0	**SPI Read Buffer 2** See the SPI Host Controller (GPIO2_0 - GPIO2_4) section for more information.

SP1_RD_BUFF3 (0x48009038)

SPI host controller read buffer 3

BIT	31	30	29	28	27	26	25	24
Field	RD_BUFFER3[31:24]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	RD_BUFFER3[23:16]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	RD_BUFFER3[15:8]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	RD_BUFFER3[7:0]							
Reset	0x00000000							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
RD_BUFFER3	31:0	**SPI Read Buffer 3** See the SPI Host Controller (GPIO2_0 - GPIO2_4) section for more information.

SP1_WR_BUFF0 (0x4800903C)

SPI host controller write buffer 0

BIT	31	30	29	28	27	26	25	24
Field	WR_BUFFER0[31:24]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	WR_BUFFER0[23:16]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	WR_BUFFER0[15:8]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	WR_BUFFER0[7:0]							
Reset	0x00000000							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
WR_BUFFER0	31:0	**SPI Write Buffer 0** See the SPI Host Controller (GPIO2_0 - GPIO2_4) section for more information.

SP1_WR_BUFF1 (0x48009040)

SPI host controller write buffer 1

BIT	31	30	29	28	27	26	25	24
Field	WR_BUFFER1[31:24]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	WR_BUFFER1[23:16]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	WR_BUFFER1[15:8]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	WR_BUFFER1[7:0]							
Reset	0x00000000							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
WR_BUFFER1	31:0	**SPI Write Buffer 1** See the SPI Host Controller (GPIO2_0 - GPIO2_4) section for more information.

SP1_WR_BUFF2 (0x48009044)

SPI host controller write buffer 2

BIT	31	30	29	28	27	26	25	24
Field	WR_BUFFER2[31:24]							
Reset	0x00000000							
Access Type	Write, Read							

BIT	23	22	21	20	19	18	17	16
Field	WR_BUFFER2[23:16]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	WR_BUFFER2[15:8]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	WR_BUFFER2[7:0]							
Reset	0x00000000							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
WR_BUFFER2	31:0	**SPI Write Buffer 2** See the SPI Host Controller (GPIO2_0 - GPIO2_4) section for more information.

SP1_WR_BUFF3 (0x48009048)

SPI host controller write buffer 3

BIT	31	30	29	28	27	26	25	24
Field	WR_BUFFER3[31:24]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	WR_BUFFER3[23:16]							
Reset	0x00000000							
Access Type	Write, Read							

BIT	15	14	13	12	11	10	9	8
Field	WR_BUFFER3[15:8]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	WR_BUFFER3[7:0]							
Reset	0x00000000							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
WR_BUFFER3	31:0	**SPI Write Buffer 3** See the SPI Host Controller (GPIO2_0 - GPIO2_4) section for more information.

SYSTEM WATCHDOG TIMER (WDT)

ADDRESS	NAME	MSB							LSB
WDT									
0x40020000	WDT_MAX[15:8]	WDT_MAX[15:8]							
	WDT_MAX[7:0]	WDT_MAX[7:0]							
0x40020004	WDT_ENABLE[7:0]	–	–	–	–	–	–	–	WDT_EN
0x40020008	WDT_LOCK[7:0]	–	–	–	–	–	–	–	LOCK
0x4002000C	WDT_CLEAR[7:0]	–	–	–	–	–	–	–	WDT_CLR
0x40020010	WDT_MIN[31:24]	WDT_MIN_EN	–	–	–	–	–	–	–
	WDT_MIN[23:16]	–	–	–	–	–	–	–	–
	WDT_MIN[15:8]	WDT_MIN[15:8]							
	WDT_MIN[7:0]	WDT_MIN[7:0]							
0x40020014	WDT_STATUS[7:0]	–	–	–	–	–	–	–	WDT_RESET

Register Details

WDT_MAX (0x40020000)

Program the maximum system watchdog timer (WDT) delay.

BIT	15	14	13	12	11	10	9	8
Field	WDT_MAX[15:8]							
Reset	0x0000							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	WDT_MAX[7:0]							
Reset	0x0000							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION	DECODE
WDT_MAX	15:0	**WDT Timer Maximum Value** Program this register to set the maximum delay between watchdog events. When enabled (WDT_EN = 1 in the WDT_ENABLE register), a reset is generated when the watchdog timer reaches this value. See the System Watchdog Timer (WDT) section for more information.	WDT_MAX[16:0] = t_{WDT_MAX} / 30.52 where t_{WDT_MAX} is the desired maximum watchdog time in µs.

WDT_ENABLE (0x40020004)

Enable the WDT.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	–	WDT_EN
Reset	–	–	–	–	–	–	–	0b0
Access Type	–	–	–	–	–	–	–	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
WDT_EN	0	**WDT Enable Bit** Set this bit to enable the system watchdog counter. The system watchdog counter begins to count up when WDT_EN = 1.	0 = System watchdog is not enabled. 1 = System watchdog is enabled and counting.

WDT_LOCK (0x40020008)

Lock the WDT registers/settings.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	–	LOCK
Reset	–	–	–	–	–	–	–	0b0
Access Type	–	–	–	–	–	–	–	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
LOCK	0	**WDT Lock Bit** Set this bit to lock the watchdog registers once the watchdog is configured. Once set, any attempts to write to the WDT_MAX, WDT_MIN or WDT_LOCK registers is ignored.	0 = WDT_MAX, WDT_MIN or WDT_LOCK registers are not locked 1 = WDT_MAX, WDT_MIN or WDT_LOCK registers are locked

WDT_CLEAR (0x4002000C)

Write to this register to generate a WDT clear event.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	–	WDT_CLR
Reset	–	–	–	–	–	–	–	0b0
Access Type	–	–	–	–	–	–	–	Write Only

BITFIELD	BITS	DESCRIPTION	DECODE
WDT_CLR	0	**WDT Clear** Write a 1 to this register to clear, or reset, the system watchdog timer. The watchdog counter is reset to 0x00 when this bit is set.	0 = Watchdog counter is not reset 1 = Generate a watchdog clear event. Watchdog counter is reset.

WDT_MIN (0x40020010)

Enable and program a minimum WDT delay.

BIT	31	30	29	28	27	26	25	24
Field	WDT_MIN_EN	–	–	–	–	–	–	–
Reset	0b0	–	–	–	–	–	–	–
Access Type	Write, Read	–	–	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**

Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	WDT_MIN[15:8]							
Reset	0x0000							
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	WDT_MIN[7:0]							
Reset	0x0000							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION	DECODE
WDT_MIN_EN	31	**WDT Minimum Timer Enable Bit** Set this bit to enable a minimum delay between two watchdog clear events (t_{WDT_MIN}). Set the WDT_MIN bits to set the minimum watchdog timer delay.	0 = Watchdog minimum timer is disabled 1 = Watchdog minimum timer is enabled
WDT_MIN	15:0	**WDT Minimum Value** Program these bits set the minimum delay before a watchdog event can occur. When enabled (WDT_EN = 1 in the WDT_ENABLE register and WDT_MIN_EN = 1 in the WDT_IN register), a reset is generated when the watchdog timer reaches this value.	WDT_MIN[16:0] = t_{WDT_MIN} / 30.52 where t_{WDT_MIN} is the desired minimum watchdog time in µs.

WDT_STATUS (0x40020014)

WDT Interrupt indicating that a watchdog reset event has occured in the Cortex-M0.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	–	WDT_RESET
Reset	–	–	–	–	–	–	–	0b0
Access Type	–	–	–	–	–	–	–	Write 1 to Clear, Read

BITFIELD	BITS	DESCRIPTION	DECODE
WDT_RESET	0	**WDT Reset Event Indicator** This bit is set when a system reset, caused by the watchdog timer, occurred since the last power-up. Write a 1 to this bit to clear it.	0 = No watchdog-initiated reset has occurred since the last power-up 1 = A watchdog-initiated system reset has occurred since the last power-up

ADC REGISTERS

ADDRESS	NAME	MSB							LSB
ADC REGISTERS									
0x40050000	ADC_CONV[31:24]	–	–	–	–	–	–	–	ADC_EN_TSENSE
	ADC_CONV[23:16]	–	–	–	–	ADC_TRG_GPIO_SEL[3:0]			
	ADC_CONV[15:8]	ADC_CONV_START	–	ADC_CONV_BUSY	–	ADC_CONV_MODE[2:0]			–
	ADC_CONV[7:0]	ADP_SEL[1:0]		ADN_SEL[1:0]		ADC_BUF_SEL[3:0]			
0x40050004	ADC_DATA[15:8]	ADC_CONV_CLIP	–	ADC_CONV_SIGN	ADC_CONV_DATA[12:8]				
	ADC_DATA[7:0]	ADC_CONV_DATA[7:0]							
0x40050008	ADC_CTRL[15:8]	ADC_READY	–	ADC_INIT[1:0]		–	–	–	–
	ADC_CTRL[7:0]	–	–	–	–	–	–	–	–
0x40050014	ADC_INT[15:8]	–	–	–	–	–	ADC_CONV_INT_EN	–	ADC_READY_INT_EN
	ADC_INT[7:0]	–	–	–	–	–	ADC_CONV_INT	–	ADC_READY_INT
0x40050018	ADC_START_DELAY[23:16]	–	ADC_START_DLY[22:16]						
	ADC_START_DELAY[15:8]	ADC_START_DLY[15:8]							
	ADC_START_DELAY[7:0]	ADC_START_DLY[7:0]							
0x40050020	ADC_AVERAGE[23:16]	–	–	–	–	–	ADC_AVG_DLY[15:13]		
	ADC_AVERAGE[15:8]	ADC_AVG_DLY[12:5]							
	ADC_AVERAGE[7:0]	ADC_AVG_DLY[4:0]					ADC_AVG_NUM[2:0]		

Register Details

ADC_CONV (0x40050000)

Configure the ADC input and trigger.

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	ADC_EN_TSENSE
Reset	–	–	–	–	–	–	–	
Access Type	–	–	–	–	–	–	–	Write, Read
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	ADC_TRG_GPIO_SEL[3:0]			
Reset	–	–	–	–				
Access Type	–	–	–	–	Write, Read			
BIT	15	14	13	12	11	10	9	8
Field	ADC_CONV_START	–	ADC_CONV_BUSY	–	ADC_CONV_MODE[2:0]			–
Reset		–		–				–
Access Type	Write, Read	–	Read Only	–	Write, Read			–
BIT	7	6	5	4	3	2	1	0
Field	ADP_SEL[1:0]		ADN_SEL[1:0]		ADC_BUF_SEL[3:0]			
Reset								
Access Type	Write, Read		Write, Read		Write, Read			

BITFIELD	BITS	DESCRIPTION	DECODE
ADC_EN_TSENSE	24	**ADC PTAT Sensor Enable (TSENSE)** Set this bit to enable the internal PTAT sensor. Calculate the PTAT temperature as: T_{MEAS}(ºC) = (ADC conversion value - 590mV) / (2mV/ºC) See the Analog-to-Digital Converter (ADC) section for more information.	0 = PTAT sensor is disabled 1 = PTAT sensor is enabled
ADC_TRG_GPIO_SEL	19:16	**ADC GPIO Trigger Select** Set these bits to select the GPIO input signal that triggers an ADC conversion when a GPIO is used to trigger an ADC conversion (ADC_CONV_MODE[2:0] = 010, 011, or 100).	0x00 - 0x06: Select GPIO1_0 to GPIO1_6 0x07 - 0x0A: Select GPIO2_1 to GPIO2_4

BITFIELD	BITS	DESCRIPTION	DECODE
ADC_CONV_START	15	**ADC Conversion Start Bit** When configured for software mode (ADC_CONV_MODE = 001), set this bit to start an ADC conversion. This bit is automatically cleared when the ADC conversion is complete.	0 = No ADC conversion started 1 = Start ADC conversion
ADC_CONV_BUSY	13	**ADC Busy Indicator Bit** This bit is set during an ADC conversion. This bit is automatically cleared when the ADC conversion is complete.	0 = ADC conversion is not active 1 = ADC conversion is active
ADC_CONV_MODE	11:9	**ADC Conversion Mode Select** Set these bits to select the ADC conversion mode. See the Analog-to-Digital (ADC) section for more information.	000 = Off (switch off or remain off) 001 = Software 010 = Event, Negative Edge 011 = Event, Positive Edge 100 = Event, Both Edges 101 = Free 110 = Heart 111 = Not Allowed
ADP_SEL	7:6	**ADC Positive Input Select** Set these bits to select the positive input souce for the ADC.	00, 01 = ADP pin 10 = ADC buffer output. See also the ADC_BUF_SEL bits. 11 = Internal 720mV (typ) reference
ADN_SEL	5:4	**ADC Negative Input Select** Set these bits to select the negative input souce for the ADC.	00, 01 = ADN pin 10 = AGND pin 11 = GND pin
ADC_BUF_SEL	3:0	**ADC Buffer Input Select** Set these bits to select the input of the ADC buffer.	0000 = AGND pin 0001 = ADP pin 0010 to 0011 = Reserved. Do not use. 0100 = GPIO1_3 pin 0101 = GPIO1_4 pin 0110 = GPIO1_5 pin 0111 = GPIO1_6 pin 1000 to 1101: Reserved. Do not use. 1110 = Internal thermal sensor (PTAT voltage). Ensure that ADC_EN_TSENSE = 1. 1111 = Internal 720mV (typ) voltage

ADC_DATA (0x40050004)

ADC clip bit, sign bit, and ADC conversion value

BIT	15	14	13	12	11	10	9	8
Field	ADC_CONV_CLIP	–	ADC_CONV_SIGN	ADC_CONV_DATA[12:8]				
Reset		–						
Access Type	Read Only	–	Read Only	Read Only				
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

Field	ADC_CONV_DATA[7:0]
Reset	
Access Type	Read Only

BITFIELD	BITS	DESCRIPTION	DECODE
ADC_CONV_CLIP	15	**ADC Conversion Clip Indicator Bit** This bit is set when the measured ADC input exceeds the input range. When using ADC averaging mode, this bit is set when 1 or more samples is clipped during the conversion.	0 = ADC conversion value is not clipped 1 = ADC conversion value is clipped (single conversion), or 1 or more samples included during averaging was clipped (average conversion)
ADC_CONV_SIGN	13	**ADC Conversion Data Sign Bit** This bit indicates the polarity of the ADC conversion value.	0 = ADC_CONV_DATA value is positive 1 = ADC_CONV_DATA value is negative
ADC_CONV_DATA	12:0	**ADC Conversion Value** This register contains the most recent ADC conversion. Read also the ADC_CONV_SIGN bit for the polarity of the ADC conversion result.	Read the ADC conversion value and sign.

ADC_CTRL (0x40050008)

Initialize the ADC.

BIT	15	14	13	12	11	10	9	8
Field	ADC_READY	–	ADC_INIT[1:0]		–	–	–	–
Reset		–			–	–	–	–
Access Type	Read Only	–	Write, Read		–	–	–	–
BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–

BITFIELD	BITS	DESCRIPTION	DECODE
ADC_READY	15	**ADC Ready Indicator Bit** This bit is set when the ADC is powered up and has stabilized. The ADC is ready for operation when ADC_READY = 1.	0 = ADC is not ready 1 = ADC is ready
ADC_INIT	13:12	**Initialize ADC Command** Write 11 to these bits to initialize the ADC for conversion.	11 = ADC is initialized All other configurations are not allowed

ADC_INT (0x40050014)

Enable the ADC interrtupts.

BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	ADC_CONV_INT_EN	–	ADC_READY_INT_EN
Reset	–	–	–	–	–		–	
Access Type	–	–	–	–	–	Write, Read	–	Write, Read
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	–	–	–	–	–	ADC_CONV_INT	–	ADC_READY_INT
Reset	–	–	–	–	–		–	
Access Type	–	–	–	–	–	Write 1 to Clear, Read	–	Write 1 to Clear, Read

BITFIELD	BITS	DESCRIPTION	DECODE
ADC_CONV_INT_EN	10	**ADC Conversion Complete Interrupt Enable** Set this bit to generate an interrupt when the ADC completes a conversion.	0 = No interrupt is generated when an ADC conversion is complete. 1 = Interrupt is generated when an ADC conversion is complete.
ADC_READY_INT_EN	8	**ADC Ready Indicator Interrupt Enable** Set this bit to generate an interrupt when the ADC is ready (ADC_READY changes from 0 to 1).	0 = ADC_READY bit is not set 1 = ADC_READY bit is set
ADC_CONV_INT	2	**ADC Conversion Complete Interrupt** This bit is set when an ADC conversion is completed. Write a 1 to this bit to clear it.	0 = No conversion completed since this bit was last cleared. 1 = ADC conversion completed
ADC_READY_INT	0	**ADC Ready Indicator Interrupt** This bit is set when ADC_READY changes from 0 to 1. Write a 1 to this bit to clear it.	0 = ADC_READY = 0 1 = ADC_READY has changed from 0 to 1

ADC_START_DELAY (0x40050018)

Set the ADC delay from when the trigger is received to when the conversion begins.

BIT	23	22	21	20	19	18	17	16
Field	–	ADC_START_DLY[22:16]						
Reset	–							

Access Type	–	Write, Read						
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	ADC_START_DLY[15:8]							
Reset								
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	ADC_START_DLY[7:0]							
Reset								
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION	DECODE
ADC_START_DLY	22:0	**ADC Converstion Start Delay** Set these bits to configure the delay from the ADC trigger event before conversion begins. Ensure that these bits are not changed during an ADC conversion. Change these bits only when ADC_CONV_MODE = 0 and ADC_CONV_BUSY = 0.	ADC_START_DLY value = $t_{ADC\_START\_DELAY}$ (µs) * f_{HCLK}. where $t_{ADC\_START\_DELAY}$ is the desired start delay and f_{HCLK} is the system clock frequency.

ADC_AVERAGE (0x40050020)

Program the delay between samples, and the number of samples to be read, when ADC averaging is enabled.

BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	ADC_AVG_DLY[15:13]		
Reset	–	–	–	–	–			
Access Type	–	–	–	–	–	Write, Read		
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	ADC_AVG_DLY[12:5]							
Reset								
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	ADC_AVG_DLY[4:0]					ADC_AVG_NUM[2:0]		
Reset								
Access Type	Write, Read					Write, Read		

BITFIELD	BITS	DESCRIPTION	DECODE
ADC_AVG_DLY	18:3	**ADC Averaging Mode Sample Delay** Set thes bits to set the delay between samples when using averaging mode for the ADC. Ensure that these bits are not changed during an ADC conversion. Change these bits only when ADC_CONV_MODE = 0 and ADC_CONV_BUSY = 0.	Set the delay time as ADC_AVG_DLY value = (t_{SAMPLE} (µs) * f_{HCLK}) - 7 where t_{SAMPLE} is the ADC sample time and f_{HCLK} is the system clock frequency.
ADC_AVG_NUM	2:0	**ADC Averaging Mode Sample Count** Number of consecutive conversions taken for an ADC measurement. The final ADC value is the average of these measurement. If enabled, the ADC conversion complete interrupt is asserted at the end of the last measurement.	000 = 1 samples 001 = 2 samples 010 = 4 samples 011 = 8 samples 100 = 16 samples 101, 110, 111 = 32 samples

DIGIPOT/VARIABLE RESISTOR REGISTERS

ADDRESS	NAME	MSB							LSB
DIGIPOT/VARIABLE RESISTOR REGISTERS									
0x40060000	CMP1[15:8]	CMP1_EN	CMPO1_OUT	–	–	–	–	–	–
	CMP1[7:0]	–	–	–	CMP1_FILT_EN	CMP1_SLOW_EN	CMP1_IN_LOW	CMP1_IN_HIGH	CMP1_HYST
0x40060004	CMP2[15:8]	CMP2_EN	CMPO2_OUT	–	–	–	–	–	–
	CMP2[7:0]	–	–	–	CMP2_FILT_EN	CMP2_SLOW_EN	CMP2_IN_LOW	CMP2_IN_HIGH	CMP2_HYST
0x40060008	DAC1[15:8]	DAC1_EN	–	–	–	–	–	–	–
	DAC1[7:0]	–	–	DAC1_VALUE[5:0]					
0x4006000C	DAC2[7:0]	–	–	DAC2_VALUE[5:0]					
0x40060010	R1[15:8]	R1_EN	–	–	–	–	–	–	–
	R1[7:0]	R1_POS[7:0]							
0x40060014	R2[15:8]	R2_EN	–	–	–	–	–	–	–
	R2[7:0]	R2_POS[7:0]							

ADDRESS	NAME	MSB							LSB
0x40060018	R3[7:0]	CMPO1_TO_R3_DIS_SEL	CMPO1_TO_R3_EN	R3_POS[5:0]					
0x4006001C	R4[15:8]	R4_EN	–	–	–	–	–	–	–
	R4[7:0]	R4_POS[7:0]							
0x40060020	CMP_INT_SET[15:8]	–	–	–	–	–	–	–	–
	CMP_INT_SET[7:0]	–	–	–	–	–	CMPO1_DEB_EN	CMPO1_EDGE_INT_SEL[1:0]	
0x40060024	CMP_INT[15:8]	–	–	–	–	–	–	–	CMPO1_INT_EN
	CMP_INT[7:0]	–	–	–	–	–	–	–	CMPO1_INT
0x40060028	CMP_TO_PDIN_REG[7:0]	–	–	–	–	CMPO1_TO_TX_SIO	CMPO1_TO_TX_INV	CMP_TO_PDIN[1:0]	

Register Details

CMP1 (0x40060000)

Configure the CMP1comparator.

BIT	15	14	13	12	11	10	9	8
Field	CMP1_EN	CMPO1_OUT	–	–	–	–	–	–
Reset			–	–	–	–	–	–
Access Type	Write, Read	Read Only	–	–	–	–	–	–

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	CMP1_FILT_EN	CMP1_SLOW_EN	CMP1_IN_LOW	CMP1_IN_HIGH	CMP1_HYST
Reset	–	–	–					
Access Type	–	–	–	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
CMP1_EN	15	**CMP1 Comparator Enable** Set this bit to enable the CMP1 comparator. The	0 = CMP1 is disabled 1 = CMP1 is enabled

BITFIELD	BITS	DESCRIPTION	DECODE
		comparator output, CMPO1, is not high impedance when disabled.	
CMPO1_OUT	14	**Comparator Output Logic State** This bit reflects the logic state of the CMP1 comparator output (CMPO1). This bit is set when CMPO1 is high.	0 = CMPO1 is low 1 = CMPO1 is high
CMP1_FILT_EN	4	**Comparator Output (CMPO1, CMPO) Filter Enable** Set this bit to enable the 700ns (typ) filter on the CMP1 output (CMPO1). See the High Speed Comparators section for more information.	0 = 700ns (typ) filter disabled 1 = 700ns (typ) filter enabled
CMP1_SLOW_EN	3	**CMP1, CMP Slow Comparator Configuration Enable** Set this bit to increase the propagation delay to 465ns (typ). When CMP1_SLOW_EN = 0, CMP1 has a 75ns (typ) propagation delay. See the High Speed Comparators section for more information.	0 = 465ns (typ) propagation delay disabled 1 = 465ns (typ) propagation delay enabled
CMP1_IN_LOW	2	**CMP1, CMP Low-Side Offset-Reduced Enable** Set this bit to reduce the offset on CMP1 when the input signal approaches the low-side rail (AGND). See the High Speed Comparators section for more information.	0 = Low-side offset is not reduced 1 = Low-side offset is reduced (CMP1_IN_HIGH = 0) or rail-to-rail operation (CMP1_IN_HIGH = 1
CMP1_IN_HIGH	1	**CMP1, CMP High-Side Offset-Reduced Enable** Set this bit to reduce the offset on CMP1 when the input signal approaches the high-side rail (V_{5A} voltage). See the High Speed Comparators section for more information.	0 = High-side offset is not reduced 1 = High-side offset is reduced (CMP1_IN_LOW = 0) or rail-to-rail operation (CMP1_IN_LOW = 1)
CMP1_HYST	0	**CMP1, CMP Hysteresis Enable** Set this bit to enable the 37mV (typ) hysteresis on the CMP1 comparator.	0 = Hysteresis disabled 1 = Hysteresis enabled

CMP2 (0x40060004)

Configure the CMP2 comparator. (MAX22512 Only)

** Writes to this register on the MAX22522 are ignored.*

BIT	15	14	13	12	11	10	9	8
Field	CMP2_EN	CMPO2_OUT	–	–	–	–	–	–
Reset			–	–	–	–	–	–
Access Type	Write, Read	Read Only	–	–	–	–	–	–

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	CMP2_FILT_EN	CMP2_SLOW_EN	CMP2_IN_LOW	CMP2_IN_HIGH	CMP2_HYST
Reset	–	–	–					
Access Type	–	–	–	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
CMP2_EN	15	**CMP2 Comparator Enable (MAX22512 only)** Set this bit to enable the CMP2 comparator. The comparator output, CMPO2, is not high impedance when disabled.	0 = CMP2 is disabled 1 = CMP2 is enabled
CMPO2_OUT	14	**CMP2 Comparator Output Logic State (MAX22512 only)** This bit reflects the logic state of the CMP2 comparator output (CMPO2). This bit is set when CMPO2 is high.	0 = CMPO2 is low 1 = CMPO2 is high
CMP2_FILT_EN	4	**CMPO2 Output Filter Enable (MAX22512 only)** Set this bit to enable the 700ns (typ) filter on the CMP2 output (CMPO2). See the High Speed Comparators section for more information.	0 = 700ns (typ) filter disabled 1 = 700ns (typ) filter enabled
CMP2_SLOW_EN	3	**CMP2 Slow Comparator Configuration Enable (MAX22512 only)** Set this bit to increase the propagation delay to 465ns (typ). When CMP2_SLOW_EN = 0, CMP2 has a 75ns (typ) propagation delay. See the High Speed Comparators section for more information.	0 = 465ns (typ) propagation delay disabled 1 = 465ns (typ) propagation delay enabled
CMP2_IN_LOW	2	**CMP2 Low-Side Offset Reduced Enable (MAX22512 only)** Set this bit to reduce the offset on CMP2 when the input signal approaches the low-side rail (AGND). See the High Speed Comparators section for more information.	0 = Low-side offset is not reduced 1 = Low-side offset is reduced (CMP2_IN_HIGH = 0) or rail-to-rail operation (CMP2_IN_HIGH = 1)
CMP2_IN_HIGH	1	**CMP2 High-Side Offset Reduced Enable (MAX22512 only)** Set this bit to reduce the offset on CMP2 when the input signal approaches the high-side rail (V_{5A} voltage). See the High Speed Comparators section for more information.	0 = High-side offset is not reduced 1 = High-side offset is reduced (CMP2_IN_LOW = 0) or rail-to-rail operation (CMP2_IN_LOW = 1)
CMP2_HYST	0	**CMP2 Hysteresis Enable (MAX22512 only)** Set this bit to enable the 37mV (typ) hysteresis on the CMP2 comparator.	0 = Hysteresis disabled 1 = Hysteresis enabled

DAC1 (0x40060008)

Set the DAC1 output voltage.

BIT	15	14	13	12	11	10	9	8
Field	DAC1_EN	–	–	–	–	–	–	–
Reset		–	–	–	–	–	–	–
Access Type	Write, Read	–	–	–	–	–	–	–
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	–	–	DAC1_VALUE[5:0]					
Reset	–	–						
Access Type	–	–	Write, Read					

BITFIELD	BITS	DESCRIPTION	DECODE
DAC1_EN	15	**DAC1 Enable** Set this bit to enable the DAC1 output. The DAC1 output is internally connected to the inverting input terminal of the CMP1 comparator to generate the CMP1 switching threshold voltage. Set the DAC output voltage by programming the DAC_VALUE bits.	0 = DAC1 is disabled 1 = DAC1 is enabled
DAC1_VALUE	5:0	**DAC1 Output Value** Set these bits to program the output voltage of DAC1. The output of DAC1 is internally connected to the inverting terminal of CMP1. Program the DAC1 output voltage to set the switching threshold for CMP1.	Calculate the DACO1 value as: DAC_VALUE value = (V_{DACO1} x 64)/ 5V where V_{DACO1} is the DAC1 output voltage in volts.

DAC2 (0x4006000C)

Set the CMP2 comparator input reference voltage.
** Writes to this register on the MAX22522 are ignored.*

BIT	7	6	5	4	3	2	1	0
Field	–	–	DAC2_VALUE[5:0]					
Reset	–	–						
Access Type	–	–	Write, Read					

BITFIELD	BITS	DESCRIPTION	DECODE
DAC2_VALUE	5:0	**CMP2 DAC Output Value (MAX22512 only)** Set these bits to program the output voltage of DAC2. The output of DAC2 is internally connected to the inverting terminal of CMP2. Program the DAC2 output voltage to set the switching threshold for CMP2.	Calculate the DAC2 value as: DAC2_VALUE value = (V_{DAC2} x 64)/ 5V where V_{DAC2} is the DAC2 output voltage in volts.

R1 (0x40060010)

Enable and set the R1 variable resistor value.

BIT	15	14	13	12	11	10	9	8
Field	R1_EN	–	–	–	–	–	–	–
Reset		–	–	–	–	–	–	–
Access Type	Write, Read	–	–	–	–	–	–	–
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	R1_POS[7:0]							
Reset								
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION	DECODE
R1_EN	15	**R1 Variable Resistor Enable** Set this bit to enable the R1 variable resistor. Set the resistance by programming the R1_POS bits.	0 = R1 is disabled 1 = R1 is enabled
R1_POS	7:0	**R1 Variable Resistor Value** **MAX22512:** Set these bits to program the R12W to R1A resistance. Calculate this resistance as: $R_{12W\ to\ R1A}$ (kΩ) = [2.5kΩ - (R1_POS / 256) x 2.5kΩ] + (2 x R_W) where R_W is the wiper switch resistance. **MAX22522:** Set these bits to program the R1W to R1A resistance. Calculate R1A as: R_{1A} (kΩ) = [10kΩ - (R1_POS / 256) x 10kΩ] + R_W where R_W is the wiper switch resistance.	0x00 = R_{1A}kΩ (typ) 0xFF = 0Ω (typ) For all other values, use the provided equation

R2 (0x40060014)

Enable and set the R2 variable resistor value.

BIT	15	14	13	12	11	10	9	8
Field	R2_EN	–	–	–	–	–	–	–
Reset		–	–	–	–	–	–	–
Access Type	Write, Read	–	–	–	–	–	–	–
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

Field	R2_POS[7:0]
Reset	
Access Type	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
R2_EN	15	**R2 Variable Resistor Enable** Set this bit to enable the R2 variable resistor. Set the resistance by programming the R2_POS bits.	0 = R2 is disabled 1 = R2 is enabled
R2_POS	7:0	**R2 Variable Resistor Value** **MAX22512:** Set these bits to program the R12W to R2A resistance. Calculate this resistance as: $R_{12W\ to\ R2A}$ (kΩ) = [2.5kΩ - (R2_POS / 256) x 2.5kΩ] + (2 x R_W) where R_W is the wiper switch resistance. **MAX22522:** Set these bits to program the R2W to R2A resistance. Calculate R2A as: R_{2A} (kΩ) = [10kΩ - (R1_POS / 256) x 10kΩ] + R_W where R_W is the wiper switch resistance.	0x00 = R_{2A}kΩ (typ) 0xFF = 0Ω (typ) For all other values, use the provided equation

R3 (0x40060018)

Set the R3 variable resistor value.

BIT	7	6	5	4	3	2	1	0
Field	CMPO1_TO_R3_DIS_SEL	CMPO1_TO_R3_EN	R3_POS[5:0]					
Reset								
Access Type	Write, Read	Write, Read	Write, Read					

BITFIELD	BITS	DESCRIPTION	DECODE
CMPO1_TO_R3_DIS_SEL	7	**Comparator Output-to-R3 Disable Select Bit** The R3 variable resistor can be enabled and disabled using the comparator output, CMPO1. Set the CMPO1_TO_R3_EN bit to 1 to enable this function. Set this bit to select the logic state of the CMPO1 output that will disable R3.	0 = R3 is disabled when CMPO1 is high (CMPO1_TO_R3_EN = 1) 1 = R3 is disabled when CMPO is low (CMPO1_TO_R3_EN = 1)
CMPO1_TO_R3_EN	6	**Comparator Output-to-R3 Select Enable Bit** The R3 variable resistor can be enabled and disabled using the comparator output, CMPO1. Set the this bit to 1 to enable this function. Set the CMPO1_TO_R3_DIS_SEL to select the CMPO1 logic state that disables R3.	0 = R3 is not enabled/disabled by CMPO1. 1 = R3 is disabled when CMPO1 changes to the selected logic state (see the CMPO1_TO_R3_DIS_SEL bit).
R3_POS	5:0	**R3 Variable Resistor Value** Set these bits to enable R3 and to program the resistance from R3 to ground (AGND).	0x00 = Open 0x01 = 60.4kΩ (typ)

BITFIELD	BITS	DESCRIPTION	DECODE
		Calculate this resistance as: R_{3A} (kΩ) = 60.4kΩ - [(R3_POS / 63) x 60.4kΩ]	0x3F = 0Ω (typ) For all other values, use the provided equation

R4 (0x4006001C)

Enable and set the R4 variable resistor value.

BIT	15	14	13	12	11	10	9	8
Field	R4_EN	–	–	–	–	–	–	–
Reset		–	–	–	–	–	–	–
Access Type	Write, Read	–	–	–	–	–	–	–
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	R4_POS[7:0]							
Reset								
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION	DECODE
R4_EN	15	**R4 Variable Resistor Enable** Set this bit to enable the R4 variable resistor. Set the resistance by programming the R4_POS bits.	0 = R4 is disabled 1 = R4 is enabled
R4_POS	7:0	**R4 Variable Resistor Value** **MAX22512:** Set these bits to program the resistance from R4A to ground (AGND). Calculate this resistance as: $R_{12W\ to\ R1A}$ (kΩ) = [2.5kΩ - (R1_POS / 256) x 2.5kΩ] where R_W is the wiper switch resistance. **MAX22522:** Set these bits to program the resistance from R4A to ground (AGND). Calculate this resistance as: $R_{12W\ to\ R1A}$ (kΩ) = [10kΩ - (R1_POS / 256) x 10kΩ] where R_W is the wiper switch resistance.	0x00 = R_4 kΩ (typ) 0xFF = 0Ω (typ) For all other values, use the provided equation

CMP_INT_SET (0x40060020)

Set and configure comparator interrupts.

BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–

Access Type	–	–	–	–	–	–	–	–
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	–	–	–	–	–	CMPO1_DEB_EN	CMPO1_EDGE_INT_SEL[1:0]	
Reset	–	–	–	–	–			
Access Type	–	–	–	–	–	Write, Read	Write, Read	

BITFIELD	BITS	DESCRIPTION	DECODE
CMPO1_DEB_EN	2	**CMPO1, CMPO Interrupt Debounce Enable Bit** Set this bit to enable the 100µs (typ) debounce at the CMPO1 output when CMP1 is configured as an interrupt output (CMPO1_INT_EN = 1).	= CMPO1 interrupt debounce is disabled 1 = CMPO1 interrupt debounce is enabled
CMPO1_EDGE_INT_SEL	1:0	**CMPO1, CMPO Output Interrupt Trigger Select** Set these bits to select the interrupt trigger (rising, falling, or rising and falling) for CMPO1 when the CMP1 comparator input (CIP1) is configured as an interrupt input (CMPO1_INT_EN = 1).	00 = Interrupt generated at rising edge of CMPO1 01 = Interrupt generated at falling edge CMPO1 10 = Interrupt generated at rising and falling edges of CMPO1 11 = Reserved. Do not use.

CMP_INT (0x40060024)

Set and configure comparator interrupts.

BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	CMPO1_INT_EN
Reset	–	–	–	–	–	–	–	
Access Type	–	–	–	–	–	–	–	Write, Read
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	–	–	–	–	–	–	–	CMPO1_INT
Reset	–	–	–	–	–	–	–	
Access Type	–	–	–	–	–	–	–	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
CMPO1_INT_EN	8	**CMPO1 Output Interrupt Enable** Set this bit to enable an interrupt when CMPO1 rises and/or falls. Set the CMPO1_EDGE_INT bits to select the interrupt edge trigger.	0 = CMPO1 output interrupt is disabled 1 = CMPO1 output interrupt is enabled

BITFIELD	BITS	DESCRIPTION	DECODE
CMPO1_INT	0	**CMPO1 Output Interrupt Bit** This bit reflects the logic state of the CMPO1 output interrupt. Set the CMPO1_EDGE_INT_SEL bits to select the trigger for this interrupt.	0 = No interrupt generated on CMPO1 1 = Interrupt generated on CMPO1

CMP_TO_PDIN_REG (0x40060028)

Configure comparator data as PDIn Data inputs.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	CMPO1_TO_TX_SIO	CMPO1_TO_TX_INV	CMP_TO_PDIN[1:0]	
Reset	–	–	–	–				
Access Type	–	–	–	–	Write, Read	Write, Read	Write, Read	

BITFIELD	BITS	DESCRIPTION	DECODE
CMPO1_TO_TX_SIO	3	**CMPO1 to TX Control Bit (SIO Mode)** Set this bit to use the CMPO1 output as the TX input of the C/Q line when configured for SIO mode. The CQ_TX bit in the IOL_TX_CTRL register is ignored when this bit is set.	0 = C/Q is not driven by the CMPO1 output. 1 = C/Q is driven by the CMPO1 output.
CMPO1_TO_TX_INV	2	**CMPO1 to TX Invert Bit** Set this bit to invert the logic state of TX, compared to CMPO1 when this functionality is enabled.	0 = TX logic state is same as CMPO1 1 = TX logic state is opposite CMPO1
CMP_TO_PDIN	1:0	**Comparator Output(s) to PDIn Enable** Set these bits to use the output of the selected comparator to set the last bit (bit 0) of the PDIn data transmitted to the IO-Link master. If both comparators are enabled, CMPO1 sets bit [0] and CMPO2 sets bit [1] of PDIn data. See the Process Data Input section for more information.	00 = PDIn from compators is disabled. 01 = PDIn matches the CMPO1 logic state 10 = PDIn matches is the CMPO2 logic state 11 = CMPO1 sets bit 0 and CMPO2 sets bit 1 of PDIn data

IO-LINK REGISTERS

ADDRESS	NAME	MSB							LSB
IO-LINK ID REGISTER									
0x40040000	IOL_REV_ID[7:0]	RevID[7:0]							
IO-LINK STATUS REGISTERS									

ADDRESS	NAME	MSB							LSB
0x400 40008	IOL_STAT[7:0]	–	–	–	–	–	DL_MODE[2:0]		
0x400 4000C	IOL_DEV_STAT1[7:0]	PAGE1_ADD_READ[4:0]					–	RX	–
0x400 40010	IOL_DEV_STAT2[7:0]	–	–	–	CQ_FAULT	V24_ERR	V24_WRN	TH_WARN	–
0x400 40014	IOL_ISDU_STAT[7:0]	–	–	–	–	–	–	–	ISDU_BAV
0x400 40018	IOL_ERR_CNT[7:0]	IOL_ERR_CNT[7:0]							
0x400 4001C	IOL_FRM_ERR_CNT[7:0]	FRM_ERR_CNT[7:0]							
IO-LINK INTERRUPT REGISTERS									
0x400 40020	IOL_INT[7:0]	WD_INT	RD_PAGE1_INT	PDOUT_DATRX_INT	CLR_EVENT_FLG_INT	WU_INT	MCMD_INT	DIR_PAGE1_INT	DL_MODE_INT
0x400 40024	IOL_DEV_INT[7:0]	–	FRM_ERR_CNT_INT	IOL_ERR_CNT_INT	CQ_FAULT_INT	V24_ERR_INT	V24_WRN_INT	THM_WRN_INT	–
0x400 40028	IOL_ISDU_INT[7:0]	–	ISDU_BNAVL_INT	ISDU_BAV_INT	CHKPDU_ERR_INT	NEW_ISDU_WR_START_INT	ISDU_ABRT_INT	ISDU_IDLE_INT	ISDU_PCKT_INT
0x400 4002C	IOL_HEART_INT[7:0]	–	–	–	–	–	–	MC_BYTE_RCV_INT	CKS_SENT_INT
IO-LINK INTERRUPT MASK REGISTERS									
0x400 40038	IOL_INT_EN[7:0]	WD_INT_EN	RD_PAGE1_INT_EN	PDOUT_DATRX_INT_EN	CLR_EVENT_FLG_INT_EN	WU_INT_EN	MCMD_INT_EN	DIR_PAGE1_INT_EN	DL_MODE_INT_EN
0x400 4003C	IOL_DEV_INT_EN[7:0]	–	FRM_ERR_CNT_INT_EN	IOL_ERR_CNT_INT_EN	CQ_FAULT_INT_EN	V24_ERR_INT_EN	V24_WRN_INT_EN	THM_WRN_INT_EN	THM_SHD_INT_EN
0x400 40040	IOL_ISDU_INT_EN[7:0]	–	ISDU_BNAVL_INT_EN	ISDU_BAV_INT_EN	CHKPDU_ERR_INT_EN	NEW_ISDU_WR_START_INT_EN	ISDU_ABRT_INT_EN	ISDU_IDLE_INT_EN	ISDU_PCKT_INT_EN
0x400 40044	IOL_HEART_INT_EN[7:0]	–	–	–	–	–	–	MC_BYTE_RCV_INT_EN	CKS_SENT_INT_EN
IO-LINK CONFIGURATION REGISTERS									
0x400 40050	IOL_CFG[7:0]	DEVICE_ANS_DELAY[1:0]		CONF_DONE	–	COMx[1:0]		PAGE1_INH	SIO_FORCE
0x400 40054	IOL_WDG_TMR[7:0]	IOL_WDG_THRESHOLD[7:0]							

ADDRESS	NAME	MSB							LSB
0x40040058	IOL_WDG_CLR[7:0]	–	–	–	–	–	IOL_FRM_COUNT_CLR	IOL_ERR_CNT_CLR	IOL_WDG_CLEAR
0x4004005C	IOL_MISC_CFG[7:0]	–	–	IOL_DLY[1:0]		–	–	–	IOL_RFU_RST
IO-LINK DIRECT PAGE 1 REGISTERS									
0x40040068	IOL_PAGE1_BYTE00[7:0]	MASTER_COMMAND[7:0]							
0x4004006C	IOL_PAGE1_BYTE01[7:0]	MASTER_CYCLE_TIME[7:0]							
0x40040070	IOL_PAGE1_BYTE02[7:0]	MIN_CYCLE_TIME[7:0]							
0x40040074	IOL_PAGE1_BYTE03[7:0]	MSEQ_CAPABILITY[7:0]							
0x40040078	IOL_PAGE1_BYTE04[7:0]	REVISION_ID[7:0]							
0x4004007C	IOL_PAGE1_BYTE05[7:0]	PROCESS_DATA_IN[7:0]							
0x40040080	IOL_PAGE1_BYTE06[7:0]	PROCESS_DATA_OUT[7:0]							
0x40040084	IOL_PAGE1_BYTE07[7:0]	VENDOR_ID1[7:0]							
0x40040088	IOL_PAGE1_BYTE08[7:0]	VENDOR_ID2[7:0]							
0x4004008C	IOL_PAGE1_BYTE09[7:0]	DEVICE_ID1[7:0]							
0x40040090	IOL_PAGE1_BYTE0A[7:0]	DEVICE_ID2[7:0]							
0x40040094	IOL_PAGE1_BYTE0B[7:0]	DEVICE_ID3[7:0]							
0x40040098	IOL_PAGE1_BYTE0C[7:0]	FUNCTION_ID1[7:0]							
0x4004009C	IOL_PAGE1_BYTE0D[7:0]	FUNCTION_ID2[7:0]							
0x400400A0	IOL_PAGE1_BYTE0E[7:0]	PAGE1_RESERVED1[7:0]							
0x400400A4	IOL_PAGE1_BYTE0F[7:0]	PAGE1_RESERVED2[7:0]							
IO-LINK EVENT REGISTERS									

ADDRESS	NAME	MSB							LSB
0x400400A8	IOL_WDG_EVENT[7:0]	IOL_WDG_EVENT_CODE[7:0]							
0x400400AC	IOL_STATUS_CODE_DEF[7:0]	STATUS_CODE_DEFAULT[7:0]							
0x400400B0	IOL_STATUS_CODE[7:0]	STATUS_CODE_BYTE[7:0]							
0x400400B4	IOL_EVENT_QUALIFIER[7:0]	EVENT_QUALIFIER[7:0]							
0x400400B8	IOL_EVENT_CODEMSB[7:0]	EVENT_CODE_1MSB[7:0]							
0x400400BC	IOL_EVENT_CODELSB[7:0]	EVENT_CODE_1LSB[7:0]							
0x400400C0	IOL_EVENT_FLAG[7:0]	–	–	–	–	–	–	IOL_WDG_EVENT	EVENT_FLAG_BIT
IO-LINK BUFFER CONTROL REGISTERS									
0x400400D4	IOL_PDIN_LOCK[7:0]	–	–	–	–	–	PD_STATUS	PDIN_LOCK	PDIN_UNLOCK
0x400400D8	IOL_PDOUT_LOCK[7:0]	–	–	–	–	–	–	PDOUT_LOCK	PDOUT_UNLOCK
0x400400DC	IOL_ISDU_DATA_RDY[7:0]	–	–	–	–	–	–	ISDU_LOCK	ISDUIN_DATA_RDY
0x400400E0	IOL_ISDU_LEVEL[7:0]	ISDU_LEVEL[7:0]							
IO-LINK LED CONTROL REGISTERS									
0x40040140	IOL_LED1CTRLMSB[7:0]	LED1_CTRL_MSB[7:0]							
0x40040144	IOL_LED1CTRLSB[7:0]	LED1_CTRL_LSB[7:0]							
0x40040148	IOL_LED2CTRLMSB[7:0]	LED2_CTRL_MSB[7:0]							
0x4004014C	IOL_LED2CTRLSB[7:0]	LED2_CTRL_LSB[7:0]							
IO-LINK TRANSCEIVER CONTROL REGISTERS									
0x40040158	IOL_CQ_CTRL1[7:0]	CQ_SLEW[1:0]		CQ_PD	CQ_PU	CQ_NPN	CQ_PP	CQ_INV	CQ_EN

ADDRESS	NAME	MSB							LSB
0x400 4015C	IOL_CQ_CTRL2[7:0]	CQ_CL[1:0]		–	CQ_BLNK_TIME[1:0]		CQ_AUTOTRY_TIME[1:0]		CQ_AUTORTY_EN
0x400 40160	IOL_TX_CTRL[7:0]	CQ_TX	CQ_TXEN	–	–	–	–	–	ALTPDIN_INV
0x400 40164	IOL_RX_CTRL[7:0]	–	–	–	RX_FILTER	–	–	–	–
0x400 40168	IOL_MISC_CTRL[7:0]	–	–	–	–	–	–	CQPUD_2mA	–

Register Details

IOL_REV_ID (0x40040000)

The REV ID of the device.

BIT	7	6	5	4	3	2	1	0
Field	RevID[7:0]							
Reset	0x00							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
RevID	7:0	**IO-Link Device Revision ID** This register reflects the IO-Link device revision (RevID) of the IC.

IOL_STAT (0x40040008)

IO-Link mode status Indicator

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	DL_MODE[2:0]		
Reset	–	–	–	–	–	0b000		
Access Type	–	–	–	–	–	Read Only		

BITFIELD	BITS	DESCRIPTION	DECODE
DL_MODE	2:0	**DL-Mode Handler Finite State Machine Status** These bits indicate the DL-Mode of the device during the IO-Link EST_COM sequence.	000 = Idle_0 001 = EstablishCom_1 010 = Startup_2 011 = PreOperate_3

BITFIELD	BITS	DESCRIPTION	DECODE
			100 = Operate_4 All other configurations = Not allowed

IOL_DEV_STAT1 (0x4004000C)

Device status following an IO-Link master Page 1 Direct Parameter read and the C/Q status.

BIT	7	6	5	4	3	2	1	0
Field	PAGE1_ADD_READ[4:0]					–	RX	–
Reset	0x00000					–	0b0	–
Access Type	Read Only					–	Read Only	–

BITFIELD	BITS	DESCRIPTION	DECODE
PAGE1_ADD_READ	7:3	**Direct Page 1 Master Read Access Value** These bits are set when a successful Page 1 Parameter read is completed by the IO-Link master.	The value is the address of the Page 1 Parameter accessed.
RX	1	**RX Output Level Status Bit** By default, the logic state of RX is the inverse of C/Q. To match the logic state of RX to C/Q, set the CQ_INV bit in the IOL_CQ_CTRL1 register.	0 = RX is low 1 = RX is high

IOL_DEV_STAT2 (0x40040010)

V_{24}, C/Q, and thermal status indicators

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	CQ_FAULT	V24_ERR	V24_WRN	TH_WARN	–
Reset	–	–	–	0x0	0x0	0x0	0b0	–
Access Type	–	–	–	Read Only	Read Only	Read Only	Read Only	–

BITFIELD	BITS	DESCRIPTION	DECODE
CQ_FAULT	4	**C/Q Fault Status Bit** This bit is set when an overcurrent or overtemperature fault occurs on C/Q. This bit is automatically cleared when the fault is removed.	0 = No fault on C/Q 1 = Fault on C/Q
V24_ERR	3	**V_{24} Undervoltage Error Status Bit** This bit is set when the V_{24} voltage falls below the 6.1V (min) error threshold. This bit is automatically cleared when V_{24} rises above the threshold.	0 = V_{24} is above 6.9V (max) 1 = V_{24} is below 6.1V (min)
V24_WRN	2	**V_{24} Warning Voltage Status Bit** This bit is set when the V_{24} voltage falls below the	0 = V_{24} is above 16.9V (typ) 1 = V_{24} is below 16.5V (typ)

BITFIELD	BITS	DESCRIPTION	DECODE
		16.5V (typ) warning threshold. This bit is automatically cleared when V_{24} rises above the threshold.	
TH_WARN	1	**Thermal Warning Status Bit** This bit is set when the die temperature rises above the 135°C (typ) warning threshold. This bit is automatically cleared when the die temperature falls below the 121°C (typ).	0 = Die tempertaure is below 121°C (typ) 1 = Die tempertaure is above 135°C (typ)

IOL_ISDU_STAT (0x40040014)

ISDU buffer activity status

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	–	ISDU_BAV
Reset	–	–	–	–	–	–	–	0b0
Access Type	–	–	–	–	–	–	–	Read Only

BITFIELD	BITS	DESCRIPTION	DECODE
ISDU_BAV	0	**ISDU Buffer Activity Status Bit** This bit is set when the ISDU buffer is accessible to the microcontroller. This bit is 0 when the ISDU buffer is busy.	0 = The ISDU buffer is currently being accessed by the SPI master 1 = The ISDU buffer is accessable by the SPI master

IOL_ERR_CNT (0x40040018)

IO-Link communication error counter

BIT	7	6	5	4	3	2	1	0
Field	IOL_ERR_CNT[7:0]							
Reset	0x00							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
IOL_ERR_CNT	7:0	**IO-Link Communication Error Counter** This register values increments every time an error occurs in the IO-Link communication. Possible causes of communication errors include a wrong check type octect (CKT) and/or a wrong number of bytes from the IO-Link master. The IC can record up to 255 errors in this register before it is cleared. Reset this counter to 0 by setting IOL_CNT_CLR = 1 in the IOL_WDG_CLEAR register.

IOL_FRM_ERR_CNT (0x4004001C)

IO-Link communication frame parity error counter

BIT	7	6	5	4	3	2	1	0
Field	FRM_ERR_CNT[7:0]							
Reset	0x00							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
FRM_ERR_CNT	7:0	**IO-Link Communication Frame Parity Error** This register value increments every time a frame parity error is detected in communications from the IO-Link master. The IC can record up to 255 errors in this register before it is cleared. Reset this counter to 0 by setting at IOL_FRM_CNT_CLR = 1 in the IOL_WDG_CLEAR register.

IOL_INT (0x40040020)

IO-Link communication interrupts

BIT	7	6	5	4	3	2	1	0
Field	WD_INT	RD_PAGE1_INT	PDOUT_DATRX_INT	CLR_EVENT_FLG_INT	WU_INT	MCMD_INT	DIR_PAGE1_INT	DL_MODE_INT
Reset	0b0	0b0	0b0	0b0	0b0	0b0	0b0	0b0
Access Type	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read

BITFIELD	BITS	DESCRIPTION	DECODE
WD_INT	7	**IO-Link Watchdog Cycle Counter Interrupt** The IC monitors the IO-Link cycle watchdog timeout and sets the WD_INT bit when the timeout expires. Write a 1 to this bit to clear it.	0 = IO-Link cycle watchdog counter has not expired since this bit was last cleared. 1 = IO-Link cycle watchdog counter expired.
RD_PAGE1_INT	6	**Page1 Read Interrupt** This bit is set when the IO-Link master has completed a Direct Parameters Page 1 read. Write a 1 to this bit to clear it.	0 = IO-Link master has not read Page1 since this bit was last cleared. 1 = IO-Link master has read Page1.
PDOUT_DATRX_INT	5	**PDOut Data Received Interrupt** This bit is set when new PDOut data is received from the IO-Link master. Write a 1 to this bit to clear it.	0 = No new PDOut data has been received since this bit was last cleared. 1 = New PDOut data has been received.

BITFIELD	BITS	DESCRIPTION	DECODE
CLR_EVENT_FLG_INT	4	**Master Clears Event Flag Interrupt** This bit is set when the IO-Link master writes to the StatusCode[7:0] bits in the IOL_STATUS_CODE register to complete the event processing and clear the event flag. Write a 1 to this bit to clear it.	0 = No Event process has been completed since this bit was last cleared. 1 = Event process has been completed.
WU_INT	3	**Valid Wake-Up Detected Interrupt** This bit is set when a valid wake-up pulse is received. Note that a valid wake-up pulse is detected only in Idle_0. Write a 1 to this bit to clear it.	0 = A valid wake-up pulse has not been detected since this bit was last cleared. 1 = A valid wake-up pulse has been detected.
MCMD_INT	2	**Master Command Received Interrupt** This bit is set every time a valid IO-Link master command (MC) is received, or the master writes to Direct Page 1 byte 0x00 (IOL_PAGE1_BYTE00 register). Write a 1 to this bit to clear it.	0 = No valid Master Command has been received since this bit was last cleared. 1 = A valid Master Command has been received.
DIR_PAGE1_INT	1	**Direct Page 1 Content Changed by Master Interrupt** This bit is set when the IO-Link master changes the following bytes in any of the following registers: IOL_PAGE1_BYTE00, IOL_PAGE1_BYTE01, IOL_PAGE1_BYTE04, IOL_PAGE1_BYTE07, IOL_PAGE1_BYTE08, IOL_PAGE1_BYTE09, IOL_PAGE1_BYTE0A, IOL_PAGE1_BYTE0B, IOL_PAGE1_BYTE0F. Write a 1 to this bit to clear it.	0 = Direct Page 1 content has not been changed by the IO-Link master since this bit was last cleared. 1 = Direct Page 1 content has been changed by the IO-Link master.
DL_MODE_INT	0	**DL-Mode Interrupt** This bit is set when the DL-Mode status is changed in the IOL_STAT register. Write a 1 to this bit to clear it.	0 = DL-Mode status has changed since this bit was last cleared. 1 = DL-Mode status has not changed.

IOL_DEV_INT (0x40040024)

IO-Link communctaion, V_{24}, and thermal interrupts

BIT	7	6	5	4	3	2	1	0
Field	–	FRM_ERR_CNT_INT	IOL_ERR_CNT_INT	CQ_FAULT_INT	V24_ERR_INT	V24_WRN_INT	THM_WRN_INT	–
Reset	–	0b0	0b0	0b0	0b0	0b0	0b0	–
Access Type	–	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	–

BITFIELD	BITS	DESCRIPTION	DECODE
FRM_ERR_CNT_INT	6	**IO-Link Frame Error Interrupt** This bit is set when a parity or frame error has occurred during IO-link communication. The IOL_FRM_ERR_CNT register value increments when an IO-Link frame error occurs. Write a 1 to this bit to clear it.	0 = IO-Link frame error has not occurred since this bit was last cleared. 1 = IO-Link frame error has occurred.
IOL_ERR_CNT_INT	5	**IO-Link Communication Error Interrupt** This bit is set when an IO-Link communication error occurs. Possible causes of IO-Link communication errors include an incorrect CKT, an incorrect number of bytes, and any issue that might cause the IC to not send the CKS packet back to the master. The IOL_ERR_CNT register value also increments every time an error occurs in the IO-Link communication. Write a 1 to this bit to clear it.	0 = IO-Link communication error has not occurred since this bit was last cleared. 1 = IO-Link communication error detected
CQ_FAULT_INT	4	**C/Q Fault Interrupt** This bit is set when an overcurrent or overtemperature fault condition occurs on the C/Q driver. Write a 1 to this bit to clear it.	0 = C/Q fault has not been detected since this bit was last cleared. 1 = C/Q fault has been detected.
V24_ERR_INT	3	**V_{24} Undervoltage Error Threshold (UVLO) Interrupt** This bit is set when the V_{24} voltage falls below the error detection threshold. Write a 1 to this bit to clear it.	0 = The V_{24} voltage has not fallen below the 6.1V (min) threshold since this bit was last cleared. 1 = The V_{24} voltage has fallen below the 6.1V (min) threshold.
V24_WRN_INT	2	**V_{24} Warning Threshold Interrupt** This bit is set when the V_{24} voltage falls below the 16.5V (typ) warning threshold. Write a 1 to this bit to clear it.	0 = The V_{24} voltage has not fallen below the 16.5V (typ) warning threshold since this bit was last cleared. 1 = The V_{24} voltage has fallen below the 16.5V (typ) warning threshold.
THM_WRN_INT	1	**Thermal Warning Interrupt** This bit is set when the die temperature rises above the 135°C (typ) thermal warning threshold. Write a 1 to this bit to clear it.	0 = The IC junction temperature has not exceeded the 135°C (typ) thermal warning threshold since this bit was last cleared. 1 = The IC junction temperature has exceeded the 135°C (typ) thermal warning threshold.

IOL_ISDU_INT (0x40040028)

ISDU interrupts

BIT	7	6	5	4	3	2	1	0
Field	–	ISDU_BNAVL_INT	ISDU_BAV_INT	CHKPDU_ERR_INT	NEW_ISDU_WR_START_INT	ISDU_ABRT_INT	ISDU_IDLE_INT	ISDU_PCKT_INT
Reset	–	0b0	0b0	0b0	0b0	0b0	0b0	0b0

Access Type	–	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read

BITFIELD	BITS	DESCRIPTION	DECODE
ISDU_BNAVL_INT	6	**ISDU Bus Not Available Interrupt** This bit is set when the IO-Link master attempts to access the ISDU while the microcontroller has locked access to the ISDU buffer. Write a 1 to this bit to clear it.	0 = IO-Link master has not attempted to access the locked ISDU buffer since this bit was last cleared. 1 = IO-Link master has attempted to access the locked ISDU buffer.
ISDU_BAV_INT	5	**ISDU Buffer Available Interrupt** This bit is set when the ISDU buffer is available to be accessed by SPI. Write a 1 to this bit to clear it.	0 = ISDU buffer is not available to the microcontroller interface since this bit was last cleared. 1 = ISDU buffer is available to the microcontroller.
CHKPDU_ERR_INT	4	**CHKPDU Error Interrupt** This bit is set when the CHKPDU packet at the end of an ISDU out transfer is incorrect. Write a 1 to this bit to clear it.	0 = Correct CHKPDU has been received. 1 = Incorrect CHKPDU received since this bit was last cleared.
NEW_ISDU_WR_START_INT	3	**New ISDU Out Cycle Write Start Interrupt** This bit is set when the IO-Link master controller packet 0x70 is received. This packet triggers the beginning of a new ISDU transfer from the IO-Link master. Write a 1 to this bit to clear it.	0 = No new ISDU out cycle started. 1 = New ISDU out cycle started since this bit was last cleared.
ISDU_ABRT_INT	2	**ISDU ABORT Command Received Interrupt** This bit is set when an ISDU ABORT command (IO-Link master command 0xFF) is received from the IO-Link master. Write a 1 to this bit to clear it.	0 = No ISDU ABORT command has been received. 1 = ISDU ABORT command has been received since this bit was last cleared.
ISDU_IDLE_INT	1	**ISDU IDLE Command Received Interrupt** This bit is set when a first ISDU IDLE master command is received after a completed ISDU request. ISDU IDLE command is 0b1111 xxxx, excluding the cases where xxxx=0000 and xxxx=1111. Write a 1 to this bit to clear it.	0 = No ISDU IDLE command has been received 1 = ISDU IDLE command has been received since this bit was last cleared.
ISDU_PCKT_INT	0	**ISDU Packet Received Interrupt** This bit is set when a complete ISDU packet is received from the IO-Link master, regardless of whether the CHKPDU is correct or incorrect. Before reading out the ISDU data, complete the following steps: 1) Clear this bit by writing a 1 to it. 2) Wait for the ISDU buffer to be available, by polling the ISDU_BAV status bit or by waiting for the interrupt ISDU_BAV_INT bit to be set.	0 = No ISDU packet has been received. 1 = Completed ISDU packet received since this bit was last cleared.

BITFIELD	BITS	DESCRIPTION	DECODE
		Write a 1 to this bit to clear it.	

IOL_HEART_INT (0x4004002C)

IO-Link message receive and send interrupts

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	MC_BYTE_RCV_INT	CKS_SENT_INT
Reset	–	–	–	–	–	–	0b0	0b0
Access Type	–	–	–	–	–	–	Write, Read	Write 1 to Clear, Read

BITFIELD	BITS	DESCRIPTION	DECODE
MC_BYTE_RCV_INT	1	**M-Sequence Control Byte Received Interrupt** This bit is set after device receives an M-sequence control byte from the IO-Link master. Write a 1 to this bit to clear it.	0 = M-sequence control byte has not been received. 1 = M-sequence control byte has been received since this bit was last cleared.
CKS_SENT_INT	0	**CKS Byte Sent Interrupt** This bit is set after the IC transmits the CKS byte to the IO-Link master. Write a 1 to this bit to clear it.	0 = No CKS byte has been transmitted. 1 = A CKS byte has been transmitted to the IO-Link master since this bit was last cleared.

IOL_INT_EN (0x40040038)

Enable IO-Link communication interrupts.

BIT	7	6	5	4	3	2	1	0
Field	WD_INT_EN	RD_PAGE1_INT_EN	PDOUT_DATRX_INT_EN	CLR_EVENT_FLG_INT_EN	WU_INT_EN	MCMD_INT_EN	DIR_PAGE1_INT_EN	DL_MODE_INT_EN
Reset	0b0	0b0	0b0	0b0	0b0	0b0	0b0	0b0
Access Type	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
WD_INT_EN	7	**IO-Link Watchdog Interrupt Enable** Set this bit to generate an interrupt when the WD_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked

BITFIELD	BITS	DESCRIPTION	DECODE
RD_PAGE1_INT_EN	6	**Page1 Read Interrupt Enable** Set this bit to generate an interrupt when the RD_PAGE1_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
PDOUT_DATRX_INT_EN	5	**PDOut Data Received Interrupt Enable** Set this bit to generate an interrupt when the PDOUT_DATRX_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
CLR_EVENT_FLG_INT_EN	4	**Master Clears Event Flag Interrupt Enable** Set this bit to generate an interrupt when the CLR_EVENT_FLG_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
WU_INT_EN	3	**Valid Wake-Up Detected Interrupt Enable** Set this bit to generate an interrupt when the WU_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
MCMD_INT_EN	2	**Master Command Received Interrupt Enable** Set this bit to generate an interrupt when the MCMD_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
DIR_PAGE1_INT_EN	1	**Direct Page 1 Content Changed by Master Interrupt Enable** Set this bit to generate an interrupt when the DIR_PAGE1_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
DL_MODE_INT_EN	0	**DL-Mode Interrupt Enable** Set this bit to generate an interrupt when the DL_MODE_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked

IOL_DEV_INT_EN (0x4004003C)

Enable IO-Link communctaion, V_{24}, and thermal interrupts.

BIT	7	6	5	4	3	2	1	0
Field	–	FRM_ERR_CNT_INT_EN	IOL_ERR_CNT_INT_EN	CQ_FAULT_INT_EN	V24_ERR_INT_EN	V24_WRN_INT_EN	THM_WRN_INT_EN	THM_SHD_INT_EN
Reset	–	0b0	0b0	0x0	0x0	0x0	0b0	0b0
Access Type	–	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
FRM_ERR_CNT_INT_EN	6	**IO-Link/UART Frame Error Interrupt Enable** Set this bit to generate an interrupt when the FRM_ERR_CNT_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
IOL_ERR_CNT_INT_EN	5	**IO-Link Communication Error Interrupt Enable** Set this bit to generate an interrupt when the IOL_ERR_CNT_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked

BITFIELD	BITS	DESCRIPTION	DECODE
CQ_FAULT_INT_EN	4	**C/Q Fault Interrupt Enable** Set this bit to generate an interrupt when the CQ_FAULT_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
V24_ERR_INT_EN	3	**V_{24} Undervoltage Error Threshold (UVLO) Interrupt Enable** Set this bit to generate an interrupt when the V24_ERR_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
V24_WRN_INT_EN	2	**V_{24} Warning Threshold Interrupt Enable** Set this bit to generate an interrupt when the V24_WRN_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
THM_WRN_INT_EN	1	**Thermal Warning Interrupt Enable** Set this bit to generate an interrupt when the THM_WRN_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
THM_SHD_INT_EN	0	**Thermal Shutdown Interrupt Enable** Set this bit to generate an interrupt when the THM_SHD_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked

IOL_ISDU_INT_EN (0x40040040)

Enable ISDU interrupts.

BIT	7	6	5	4	3	2	1	0
Field	–	ISDU_BNAVL_INT_EN	ISDU_BAV_INT_EN	CHKPDU_ERR_INT_EN	NEW_ISDU_WR_START_INT_EN	ISDU_ABRT_INT_EN	ISDU_IDLE_INT_EN	ISDU_PCKT_INT_EN
Reset	–	0b0	0b0	0b0	0b0	0b0	0b0	0b0
Access Type	–	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
ISDU_BNAVL_INT_EN	6	**ISDU Bus Not Available Interrupt Enable** Set this bit to generate an interrupt when the ISDU_BNAVL_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
ISDU_BAV_INT_EN	5	**ISDU Buffer Available Interrupt Enable** Set this bit to generate an interrupt when the ISDU_BAV_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
CHKPDU_ERR_INT_EN	4	**CHKPDU Error Interrupt Enable** Set this bit to generate an interrupt when the CHKPDU_ERR_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
NEW_ISDU_WR_START_INT_EN	3	**New ISDU Out Cycle Write Start Interrupt Enable** Set this bit to generate an interrupt when the NEW_ISDU_WR_START_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked

BITFIELD	BITS	DESCRIPTION	DECODE
ISDU_ABRT_INT_EN	2	**ISDU ABORT Command Received Interrupt Enable** Set this bit to generate an interrupt when the ISDU_ABRT_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
ISDU_IDLE_INT_EN	1	**ISDU IDLE Command Received Interrupt Enable** Set this bit to generate an interrupt when the ISDU_IDLE_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
ISDU_PCKT_INT_EN	0	**ISDU Packet Received Interrupt Enable** Set this bit to generate an interrupt when the ISDU_PCKT_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked

IOL_HEART_INT_EN (0x40040044)

Enable IO-Link message receive and send interrupts.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	MC_BYTE_RCV_INT_EN	CKS_SENT_INT_EN
Reset	–	–	–	–	–	–	0b0	0b0
Access Type	–	–	–	–	–	–	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
MC_BYTE_RCV_INT_EN	1	**M-Sequence Control Byte Received Interrupt Enable** Set this bit to generate an interrupt when the MC_BYTE_RCV_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked
CKS_SENT_INT_EN	0	**CKS Byte Sent Interrupt Enable** Set this bit to generate an interrupt when the CKS_SENT_INT bit is set.	0 = Interrupt is masked 1 = Interrupt is not masked

IOL_CFG (0x40040050)

Set the IO-Link communication and SIO mode.

BIT	7	6	5	4	3	2	1	0
Field	DEVICE_ANS_DELAY[1:0]		CONF_DONE	–	COMx[1:0]		PAGE1_INH	SIO_FORCE
Reset	0b00		0b0	–	0b00		0b0	0b0
Access Type	Write, Read		Write, Read	–	Write, Read		Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
DEVICE_ANS_DELAY	7:6	**Device Answer Delay** Set these bits to program the device answer delay (t_A). This delay is the time between the end of the master message received and the beginning of the response from the device to the IO-Link master. This delay is set in T_{BIT} intervals.	00 = 2 T_{BIT} 01 = 4 T_{BIT} 10 = 6 T_{BIT} 11 = 8 T_{BIT}
CONF_DONE	5	**IO-Link Register Map Configuration Complete** Set this bit when the IO-Link configuration is completed after power-up. **NOTE:** The C/Q transceiver does not exit from SIO mode when this bit is 0.	0 = IO-Link configuration is not complete 1 = IO-Link configuration is complete
COMx	3:2	**IO-Link Communication Speed** Set these bits to select the COM rate for IO-Link communication between the IC and the IO-Link master.	00, 11 = COM3 (230.4kbps) 01 = COM2 (38.4kbps) 10 = COM1 (4.8kbps)
PAGE1_INH	1	**PAGE 1 Inhibit for Master Writes** Set this bit to ignore IO-Link master writes to the: RevisionID (IOL_PAGE1_BYTE04 register), VendorID1 (IOL_PAGE1_BYTE07 register), VendorID2 (IOL_PAGE1_BYTE08 register), DeviceID1 (IOL_PAGE1_BYTE09 register), DeviceID2 (IOL_PAGE1_BYTE0A register), and DeviceID3 (IOL_PAGE1_BYTE0B register) Page 1 parameters. Writes to these registers are allowed when this bit is 0. The DIR_PAGE1_INT is triggered regardless of this bit.	0 = IO-Link Master can overwrite the listed Page 1 registers. 1 = IO-Link master cannot overwrite listed Page 1 registers. Writes to these registers are ignored.
SIO_FORCE	0	**Force SIO Mode** Set this bit to reset the DL-Mode Handler state machine and force the device into SIO mode.	0 = Do not force the device into SIO mode 1 = Force the device to SIO mode

IOL_WDG_TMR (0x40040054)

Set the IO-Link watchdog (WDG) timer threshold.

BIT	7	6	5	4	3	2	1	0
Field	IOL_WDG_THRESHOLD[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
IOL_WDG_THRESHOLD	7:0	**IO-Link Cycle Watchdog Threshold** Program this register to set the limit for the number of successful IO-Link cycles after which the device goes automatically into SIO mode and WD_INT is set. Periodically clear the IO-Link watchdog counter. The method to clear the register depends on the IOL_WDG_CLEAR bit setting in the IOL_WDG_CLR register. The register value should be changed only in SIO mode. The watchdog is not active in SIO and EST COM modes.

BITFIELD	BITS	DESCRIPTION
		Write 0x00 to this register to disable the IO-Link watchdog cycle counter.

IOL_WDG_CLR (0x40040058)

Clear IO-Link error registers and configure the IO-Link WDG clear process.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	IOL_FRM_COUNT_CLR	IOL_ERR_CNT_CLR	IOL_WDG_CLEAR
Reset	–	–	–	–	–	0b0	0b0	0b0
Access Type	–	–	–	–	–	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
IOL_FRM_COUNT_CLR	2	**Framer Error Counter Clear** Set this bit to 1 to reset the IOL_FRM_ERR_CNT register. This bit is automatically cleared after the register is reset.	0 = IOL_FRM_ERR_CNT register is not reset 1 = IOL_FRM_ERR_CNT register is reset
IOL_ERR_CNT_CLR	1	**IO-Link Error Counter Clear** Set this bit to 1 to reset the IOL_ERR_CNT register. This bit is automatically cleared after the register is reset.	0 = IOL_ERR_CNT register is not reset 1 = IOL_ERR_CNT register is reset
IOL_WDG_CLEAR	0	**IO-Link Cycle Watchdog Counter Clear Configuration** Set this bit to configure the clear process for the IO-Link cycle watchdog counter.	0 = The IO-Link watchdog counter value is reset after a microcontroller write to the IOL_WDG_TMR register. 1 = The IO-Link watchdog counter value is reset after loading PDIn data to the PDIn buffer.

IOL_MISC_CFG (0x4004005C)

Set IO-Link interpacket delay and/or reset the IO-Link registers and logic.

BIT	7	6	5	4	3	2	1	0
Field	–	–	IOL_DLY[1:0]		–	–	–	IOL_RFU_RST
Reset	–	–	0b00		–	–	–	
Access Type	–	–	Write, Read		–	–	–	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
IOL_DLY	5:4	**IO-Link Interpacket Delay Select** Set these bits to select the interpacket delay during IO-Link communication. This delay is set in T_{BIT} intervals.	00 = Delay is 0 T_{BIT} 01 = Delay is 1 T_{BIT} 10 = Delay is 2 T_{BIT} 11 = Delay is 3 T_{BIT}

BITFIELD	BITS	DESCRIPTION	DECODE
IOL_RFU_RST	0	**IO-Link and Buffer Logic Reset** Set this bit to reset the IO-Link registers and logic. This bit is automatically cleared after the reset is complete. If IO-Link communication is established before the reset event, a new wakeup is required for IO-Link communication after the reset.	0 = IO-Link registers and logic are not reset. 1 = IO-Link registers and logic are reset.

IOL_PAGE1_BYTE00 (0x40040068)

IO-Link Direct Page 1 BYTE00 (Master Command received)

BIT	7	6	5	4	3	2	1	0
Field	MASTER_COMMAND[7:0]							
Reset	0x00							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
MASTER_COMMAND	7:0	**Last Master Command Received** This register contains the last IO-Link master command to the device to switch operating states. *IO-Link master access: Write, Read* *Microcontroller access: Read only*

IOL_PAGE1_BYTE01 (0x4004006C)

IO-Link Direct Page 1 BYTE01 (Master cycle time)

BIT	7	6	5	4	3	2	1	0
Field	MASTER_CYCLE_TIME[7:0]							
Reset	0x00							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
MASTER_CYCLE_TIME	7:0	**IO-Link Master Cycle Duration** This register contains the actual cycle duration used by the IO-Link master to address the device. *IO-Link master access: Write, Read* *Microcontroller access: Read only*

IOL_PAGE1_BYTE02 (0x40040070)

IO-Link Direct Page 1 BYTE02 (Minimum cycle time)

BIT	7	6	5	4	3	2	1	0
Field	MIN_CYCLE_TIME[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
MIN_CYCLE_TIME	7:0	**Device Minimum Cycle Time** Program this register to set the minimum cycle time. The minimum cycle time is a performance feature of the device, and depends on its technology and implementation. *IO-Link master access: Write, Read* *Microcontroller access: Read only*

IOL_PAGE1_BYTE03 (0x40040074)

IO-Link Direct Page 1 BYTE03 (M-Sequence capability)

BIT	7	6	5	4	3	2	1	0
Field	MSEQ_CAPABILITY[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
MSEQ_CAPABILITY	7:0	**M-Sequency Capability** Program this register to set the M-sequence capability for the device. *IO-Link master access: Read only* *Microcontroller access: Write, Read*

IOL_PAGE1_BYTE04 (0x40040078)

IO-Link Direct Page 1 BYTE04 (Revision ID)

BIT	7	6	5	4	3	2	1	0
Field	REVISION_ID[7:0]							
Reset	0x03							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
REVISION_ID	7:0	**Device Protocol Revision** Program this register with the revision of the IO-Link protocol implemented in the IO-Link device. *IO-Link master access: Write, Read* *Microcontroller access: Write, Read*

IOL_PAGE1_BYTE05 (0x4004007C)

IO-Link Direct Page 1 BYTE05 (PDIn setting)

BIT	7	6	5	4	3	2	1	0
Field	PROCESS_DATA_IN[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
PROCESS_DATA_IN	7:0	**Process Data In (PDIn) Configuration** Program this register with the type and length of PDIn data (process data from the device to the IO-Link master) as outlined in the IO-Link specification. *IO-Link master access: Read only* *Microcontroller access: Write, Read*

IOL_PAGE1_BYTE06 (0x40040080)

IO-Link Direct Page 1 BYTE06 (PDOut setting)

BIT	7	6	5	4	3	2	1	0
Field	PROCESS_DATA_OUT[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
PROCESS_DATA_OUT	7:0	**Process Data Out (PDOut) Configuration** Program this register with the type and length of PDOut data (process data from the IO-Link master to the device) as outlined in the IO-Link specification. *IO-Link master access: Read only* *Microcontroller access: Write, Read*

IOL_PAGE1_BYTE07 (0x40040084)

IO-Link Direct Page 1 BYTE07 (Vendor ID 1)

BIT	7	6	5	4	3	2	1	0
Field	VENDOR_ID1[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
VENDOR_ID1	7:0	**Unique Vendor Identification (MSB)** Each IO-Link device vendor has a unique vendor identification number. This vendor identification must be communicated to the IO-Link master when establishing communication. Program this register with the MSB of the unique vendor identification of the IO-Link device. *IO-Link master access: Write, Read* *Microcontroller access: Write, Read*

IOL_PAGE1_BYTE08 (0x40040088)

IO-Link Direct Page 1 BYTE08 (Vendor ID 2)

BIT	7	6	5	4	3	2	1	0
Field	VENDOR_ID2[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
VENDOR_ID2	7:0	**Unique Vendor Identification (LSB)** Each IO-Link device vendor has a unique vendor identification number. This vendor identification must be communicated to the IO-Link master when establishing communication. Program this register with the LSB of the unique vendor identification of the IO-Link device. *IO-Link master access: Write, Read* *Microcontroller access: Write, Read*

IOL_PAGE1_BYTE09 (0x4004008C)

IO-Link Direct Page 1 BYTE09 (Device ID 1)

BIT	7	6	5	4	3	2	1	0
Field	DEVICE_ID1[7:0]							

Reset	0x00
Access Type	Write, Read

BITFIELD	BITS	DESCRIPTION
DEVICE_ID1	7:0	**Unique Vendor-Allocated Device Indentification (Octet 2, MSB)** Each IO-Link device has a unique device idenfication number allocated by the device vendor. Program this register with the MSB (Octet 2) of the unique device idenfication number of the IO-Link device. *IO-Link master access: Write, Read* *Microcontroller access: Write, Read*

IOL_PAGE1_BYTE0A (0x40040090)

IO-Link Direct Page 1 BYTE0A (Device ID 2)

BIT	7	6	5	4	3	2	1	0
Field	DEVICE_ID2[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
DEVICE_ID2	7:0	**Unique Vendor-Allocated Device Indentification (Octet 1)** Each IO-Link device has a unique device idenfication number allocated by the device vendor. Program this register with the Octet 1 of the unique device idenfication number of the IO-Link device. *IO-Link master access: Write, Read* *Microcontroller access: Write, Read*

IOL_PAGE1_BYTE0B (0x40040094)

IO-Link Direct Page 1 BYTE0B (Device ID 3)

BIT	7	6	5	4	3	2	1	0
Field	DEVICE_ID3[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
DEVICE_ID3	7:0	**Unique Vendor-Allocated Device Indentification (Octet 0, LSB)** Each IO-Link device has a unique device idenfication number allocated by the device vendor. Program this register with the LSB (Octet 0) of the unique device idenfication number of the IO-Link device. *IO-Link master access: Write, Read* *Microcontroller access: Write, Read*

IOL_PAGE1_BYTE0C (0x40040098)

IO-Link Direct Page 1 BYTE0C (Function ID 1)

BIT	7	6	5	4	3	2	1	0
Field	FUNCTION_ID1[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
FUNCTION_ID1	7:0	**Function ID 1 MSB (Reserved)** *IO-Link master access: Read only* *Microcontroller access: Write, Read*

IOL_PAGE1_BYTE0D (0x4004009C)

IO-Link Direct Page 1 BYTE0D (Function ID 2)

BIT	7	6	5	4	3	2	1	0
Field	FUNCTION_ID2[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
FUNCTION_ID2	7:0	**Function ID 2 LSB (Reserved)** *IO-Link master access: Read only* *Microcontroller access: Write, Read*

IOL_PAGE1_BYTE0E (0x400400A0)

IO-Link Direct Page 1 BYTE0E (Page 1, reserved)

BIT	7	6	5	4	3	2	1	0

Field	PAGE1_RESERVED1[7:0]
Reset	0x00
Access Type	Write, Read

BITFIELD	BITS	DESCRIPTION
PAGE1_RESERVED1	7:0	**Reserved** *IO-Link master access: Read only* *Microcontroller access: Write, Read*

IOL_PAGE1_BYTE0F (0x400400A4)

IO-Link Direct Page 1 BYTE0F (Page 1, reserved)

BIT	7	6	5	4	3	2	1	0
Field	PAGE1_RESERVED2[7:0]							
Reset	0x00							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
PAGE1_RESERVED2	7:0	**Reserved** *IO-Link master access: Write, Read* *Microcontroller access: Read only*

IOL_WDG_EVENT (0x400400A8)

Set the IO-Link watchdog event code.

BIT	7	6	5	4	3	2	1	0
Field	IOL_WDG_EVENT_CODE[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
IOL_WDG_EVENT_CODE	7:0	**IO-Link Watchdog Event Code** It is possible to send a designated data octet to the IOL_STATUS_CODE register when the IO-Link cycle watchdog timer expires. Program the watchdog event code to this register. Data in the IOL_STATUS_CODE register is replaced with the IOL_WDG_EVENT register value when the watchdog timer expires and IOL_WDG_EVENT = 1.

IOL_STATUS_CODE_DEF (0x400400AC)

Set the IO-Link device status code default message.

BIT	7	6	5	4	3	2	1	0
Field	STATUS_CODE_DEFAULT[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
STATUS_CODE_DEFAULT	7:0	**IO-Link Device Status Code Default Message** Write the default IO-Link device status code to this register. When the IO-Link master issues a command to read from Device Event memory address 0x00, and EVENT_FLAG_BIT = 0 in the IOL_EVENT_FLAG register, the contents of this register are transmitted to the IO-Link master. When the IO-Link master issues a command to read from Device Event memory address 0x00, and EVENT_FLAG_BIT = 1 in the IOL_EVENT_FLAG register, the contents of the IOL_STATUS_CODE register are transmitted to the IO-Link master.

IOL_STATUS_CODE (0x400400B0)

Set the IO-Link device status code message.

BIT	7	6	5	4	3	2	1	0
Field	STATUS_CODE_BYTE[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
STATUS_CODE_BYTE	7:0	**IO-Link Device Status Code** Write the IO-Link device status code to this register. When the IO-Link master issues a command to read from Device Event memory address 0x00, and EVENT_FLAG_BIT = 1 in the IOL_EVENT_FLAG register, the contents of the IOL_STATUS_CODE register are transmitted to the IO-Link master.

IOL_EVENT_QUALIFIER (0x400400B4)

Write the EventQualifier code to this register for IO-Link master event transmissions.

BIT	7	6	5	4	3	2	1	0
Field	EVENT_QUALIFIER[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
EVENT_QUALIFIER	7:0	**IO-Link Event Qualifier** Diagnosis information for an IO-Link device is transmitted to the IO-Link master through Events. Events comprise EventQualifiers and EventCodes. Write the EventQualifier code to this register for IO-Link master event transmissions.

IOL_EVENT_CODEMSB (0x400400B8)

Write the MSB of the EventCode to this register for IO-Link master event transmissions.

BIT	7	6	5	4	3	2	1	0
Field	EVENT_CODE_1MSB[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
EVENT_CODE_1MSB	7:0	**Event Code (MSB)** Diagnosis information for an IO-Link device is transmitted to the IO-Link master through Events. Events comprise EventQualifiers and EventCodes. Write the MSB of the EventCode to this register for IO-Link master event transmissions.

IOL_EVENT_CODELSB (0x400400BC)

Write the LSB of the EventCode to this register for IO-Link master event transmissions.

BIT	7	6	5	4	3	2	1	0
Field	EVENT_CODE_1LSB[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
EVENT_CODE_1LSB	7:0	**Event Code (LSB)** Diagnosis information for an IO-Link device is transmitted to the IO-Link master through Events. Events comprise EventQualifiers and EventCodes. Write the LSB of the EventCode to this register for IO-Link master event transmissions.

IOL_EVENT_FLAG (0x400400C0)

Configure the IO-Link watchdog event when the timer expires and/or set the IO-Link event flag bit.

BIT	7	6	5	4	3	2	1	0

Field	–	–	–	–	–	–	IOL_WDG_EVENT	EVENT_FLAG_BIT
Reset	–	–	–	–	–	–	0b0	0b0
Access Type	–	–	–	–	–	–	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
IOL_WDG_EVENT	1	**IO-Link Watchdog Event Flag** It is possible to send a designated data octect to the IOL_STATUS_CODE register when the IO-Link cycle watchdog timer expires. Set this bit to transmit data from the IOL_WDG_EVENT register when the IO-Link cycle watchdog timer expires.	0 = Contents of the IOL_STATUS_CODE register are not overwritten and the EVENT_FLAG_BIT is 0 when the IO-Link cycle watchdog timer expires. 1 = Contents of the IOL_STATUS_CODE register are overwritten with the contents of the IOL_WDG_EVENT register and the EVENT_FLAG_BIT is 1 when the IO-Link cycle watchdog timer expires.
EVENT_FLAG_BIT	0	**IO-Link Event Flag Bit** The checksum/status octet (CKS) is part of the reply message from an IO-Link device to the IO-Link master. Set the EVENT_FLAG_BIT bit to indicate to the IO-Link master when an event occurred. This bit is cleared when the device receives a write access request to IOL_STATUS_CODE register from the IO-Link master.	0 = No IO-Link event has occurred 1 = IO-Link event has occurred

IOL_PDIN_LOCK (0x400400D4)

Lock and unlock the PDIn buffer and set the process data valid/invalid status bit.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	PD_STATUS	PDIN_LOCK	PDIN_UNLOCK
Reset	–	–	–	–	–	0b0	0b0	0b0
Access Type	–	–	–	–	–	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
PD_STATUS	2	**Process Data Valid/Invalid Bit** The PD_STATUS bit in the CKS packet indicates whether the device can provide process data or not. Write a 1 to this bit if the PDIn data is not valid. Clear this bit as soon as there is valid process data to send to the IO-Link master.	0 = PDIn process data is valid 1 = PDIn process data is not valid
PDIN_LOCK	1	**PDIn Buffer Lock Bit** Set this bit to allow the microcontroller to write to the PDIn buffer. See the Process Data Input (PDIn) section for more information. This bit is automatically cleared.	Write a 1 to lock the PDIn buffer.
PDIN_UNLOCK	0	**PDIn Buffer Unlock Bit** Set this bit to unlock a locked PDIn buffer. The PDIn buffer must be locked to write to it, but must	Write 1 to unlock a locked PDIn buffer.

BITFIELD	BITS	DESCRIPTION	DECODE
		be unlocked to send process data to the IO-Link master. See the Process Data Input (PDIn) section for more information. This bit is automatically cleared.	

IOL_PDOUT_LOCK (0x400400D8)

Lock and unlock the PDOut buffer.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	PDOUT_LOCK	PDOUT_UNLOCK
Reset	–	–	–	–	–	–	0b0	0b0
Access Type	–	–	–	–	–	–	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
PDOUT_LOCK	1	**PDOut Buffer Lock Bit** Set this bit to lock the PDOut buffer. See the Process Data Output (PDOut) section for more information. This bit is automatically cleared.	Write 1 to lock the PDOut buffer.
PDOUT_UNLOCK	0	**PDOut Buffer Unlock Bit** Write 1 to this bit to unlock a locked PDOut buffer. See the Process Data Output (PDOut) section for more information. This bit is automatically cleared.	Write 1 to unlock a locked PDOut buffer.

IOL_ISDU_DATA_RDY (0x400400DC)

Lock and unlock the ISDU buffer and set the ISDU data ready bit.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	ISDU_LOCK	ISDUIN_DATA_RDY
Reset	–	–	–	–	–	–	0b0	0b0
Access Type	–	–	–	–	–	–	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
ISDU_LOCK	1	**ISDU Buffer Lock Bit** Set this bit to lock the ISDU buffer with newest data. The ISDU buffer is not accessable when ISDU_LOCK = 0.	0 = ISDU buffer is not locked 1 = ISDU buffer is locked

BITFIELD	BITS	DESCRIPTION	DECODE
		Write a 1 to this bit after processing ISDU data to unlock the ISDU buffer. If the IO-Link master attempts to access the ISDU buffer while it is locked (ISDU_LOCK = 1), this bit is cleared and ISDU_BNAVL_INT is set. See the ISDU Transmission section for more information.	
ISDUIN_DATA_RDY	0	**ISDU FIFO Data Ready Bit** Set this bit after writing data to the ISDU input FIFO to arm the IO-Link transfer. Ensure that ISDU_PCKT_INT = 0 in the IOL_ISDUI_NT register and NEW_ISDU_WR_START_IN must be 0. This bit is automatically cleared when the IO-Link master sends the ISDU-ABORT or ISDU_IDLE commands.	0 = Any request from the IO-Link master for an ISDU read is responded with an ISDU BUSY signal. 1 = Bytes loaded into the ISDU input FIFO will be transmitted to the IO-Link master.

IOL_ISDU_LEVEL (0x400400E0)

ISDU FIFO fill level

BIT	7	6	5	4	3	2	1	0
Field	ISDU_LEVEL[7:0]							
Reset	0x00							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
ISDU_LEVEL	7:0	**ISDU FIFO Fill Level** Data written to this register is the expected fill level of the ISDU FIFO after an ISDU out transfer initiated by the IO-Link master is complete. Calculate the fill level using the value of the ISDU header (that is. the first byte, or first two bytes) of the ISDU FIFO.

IOL_LED1CTRLMSB (0x40040140)

Program the LED1 output sequence by setting the LED1CTRMSB and LED1CTRLSB registers.

BIT	7	6	5	4	3	2	1	0
Field	LED1_CTRL_MSB[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
LED1_CTRL_MSB	7:0	**LED1 Control Sequence MSB** The LED1 output is low for each '1' in this register. LED1 is high impedance for each '0' in this register. Each bit is sampled with a 63ms period clock.

IOL_LED1CTRLSB (0x40040144)

Program the LED1 output sequence by setting the LED1CTRMSB and LED1CTRLSB registers.

BIT	7	6	5	4	3	2	1	0
Field	LED1_CTRL_LSB[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
LED1_CTRL_LSB	7:0	**LED1 Control Sequence LSB** The LED1 output is low for each '1' in this register. LED1 is high impedance for each '0' in this register. Each bit is sampled with a 63ms period clock.

IOL_LED2CTRLMSB (0x40040148)

Program the LED2 output sequence by setting the LED2CTRMSB and LED2CTRLSB registers.

BIT	7	6	5	4	3	2	1	0
Field	LED2_CTRL_MSB[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
LED2_CTRL_MSB	7:0	**LED2 Control Sequence MSB** The LED2 output is low for each '1' in this register. LED2 is high impedance for each '0' in this register. Each bit is sampled with a 63ms period clock.

IOL_LED2CTRLSB (0x4004014C)

Program the LED2 output sequence by setting the LED2CTRMSB and LED2CTRLSB registers.

BIT	7	6	5	4	3	2	1	0
Field	LED2_CTRL_LSB[7:0]							

Reset	0x00
Access Type	Write, Read

BITFIELD	BITS	DESCRIPTION
LED2_CTRL_LSB	7:0	**LED2 Control Sequence LSB** The LED2 output is low for each '1' in this register. LED2 is high impedance for each '0' in this register. Each bit is sampled with a 63ms period clock.

IOL_CQ_CTRL1 (0x40040158)

Configure and enable the C/Q driver.

BIT	7	6	5	4	3	2	1	0
Field	CQ_SLEW[1:0]		CQ_PD	CQ_PU	CQ_NPN	CQ_PP	CQ_INV	CQ_EN
Reset	0b00		0b0	0b0	0b0	0b0	0b0	0b0
Access Type	Write, Read		Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
CQ_SLEW	7:6	**C/Q Driver Slew Rate Select** Set the CQ_SLEW[1:0] bits to program the typical rising and falling slew rates on the C/Q driver.	00 = 0.23µs rising (max), 0.27µs falling (max) 01 = 0.48µs (max), 0.52µs falling (max) 10 = 0.75µs (max), 0.79µs falling (max) 11 = 6.3µs (max), 5.0µs falling (max)
CQ_PD	5	**C/Q Pull-Down Enable** Set this bit to enable the internal pull-down on C/Q. Set CQPUD_2mA = 0, CQ_PD = 1 to enable the internal weak pull-down on C/Q. CQPUD_2mA = 1, CQ_PD = 1 to enable the internal 2mA (typ) pull-down on C/Q.	0 = Internal pull-down disabled 1 = Internal pull-down enabled
CQ_PU	4	**C/Q Weak Pull-Up Enable** Set this bit to enable the internal pull-up on C/Q. Set CQPUD_2mA = 0, CQ_PU = 1 to enable the internal weak pull-up on C/Q. CQPUD_2mA = 1, CQ_PU = 1 to enable the internal 2mA (typ) pull-up on C/Q.	0 = Internal pull-up disabled 1 = Internal pull-up enabled
CQ_NPN	3	**C/Q Driver NPN Mode Select** Set this bit to configure the C/Q driver in NPN mode when CQ_PP = 0. **NOTE:** This bit is ignored when CQ_PP = 1 or when C/Q is configured in IO-Link mode. C/Q is forced to push-pull mode in IO-Link mode.	0 = C/Q is configured in PNP (CQ_PP = 0, CQ_NPN = 0) or PP mode (CQ_PP = 1) 1 = C/Q is configured in NPN or PP mode (CQ_PP = 1)
CQ_PP	2	**C/Q Driver Push-Pull Mode Select** Set this bit to configure the C/Q driver in push-pull mode. **NOTE:** This bit is ignored when C/Q is configured in IO-Link mode. C/Q is forced to push-pull mode in IO-Link mode.	0 = C/Q is configured in PNP (CQ_NPN = 0) or NPN mode (CQ_NPN = 1) 1 = C/Q is configured in PP mode

BITFIELD	BITS	DESCRIPTION	DECODE
CQ_INV	1	**C/Q Receiver/Driver Logic Invert** By default, C/Q is the logic inverse of the CQ_TX bit or the comparator output (CMPO) when CMP_TO_TX_SIO = 1. Set this bit to invert the C/Q signal so that the C/Q logic matches the CQ_TX bit or CMPO.	0 = C/Q logic is inverted compared to the CQ_TX bit or CMPO output when CMPO_TO_TX_SIO = 1. 1 = C/Q logic is the same as the CQ_TX bit or CMPO output when CMPO_TO_TX_SIO = 1.
CQ_EN	0	**C/Q Driver Enable**	0 = C/Q driver is disabled 1 = C/Q driver is enabled

IOL_CQ_CTRL2 (0x4004015C)

Configure and enable the C/Q driver.

BIT	7	6	5	4	3	2	1	0
Field	CQ_CL[1:0]		–	CQ_BLNK_TIME[1:0]		CQ_AUTOTRY_TIME[1:0]		CQ_AUTORTY_EN
Reset	0b00		–	0b00		0b00		0b0
Access Type	Write, Read		–	Write, Read		Write, Read		Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
CQ_CL	7:6	**C/Q Driver Current Limit Setting** Set these bits to select the active current limit level for the C/Q driver.	00 = 60mA (typ) 01 = 120mA (typ) 10 = 240mA (typ) 11 = 287mA (typ)
CQ_BLNK_TIME	4:3	**C/Q Driver Current Limit Blanking Time** Set these bits to program the blanking time for the C/Q driver. The C/Q driver is disabled when the load current exceeds the C/Q driver current threshold for longer than the programmed blanking time.	µ00 = 128µs (typ) 01 = 500us (typ) 10 = 1ms (typ) 11 = 5ms (typ)
CQ_AUTOTRY_TIME	2:1	**C/Q Driver Autoretry Driver Off Time** Set these bits to select the fixed off-time for the C/Q driver after a fault has been generated when autoretry functionality is enabled (CQ_AUTORTY_EN = 1). The driver is automatically reenabled after the fixed off-delay.	00 = 50ms (typ) 01 = 100ms (typ) 10 = 200ms (typ) 11 = 500ms (typ)
CQ_AUTORTY_EN	0	**C/Q Driver Autoretry Enable** Set this bit to enable autoretry functionality for the C/Q driver. When a fault is signaled on the C/Q driver, and CQ_AUTORTY_EN = 1, the driver is disabled for the programmed fixed off time (CQ_AUTOTRY_TIME[1:0]) and then automatically reenabled.	0 = Autoretry is not enabled 1 = Autoretry is enabled

IOL_TX_CTRL (0x40040160)

Set the C/Q TX and TXEN inputs and enable the GPIO to PDIN invert logic, if needed.

BIT	7	6	5	4	3	2	1	0
Field	CQ_TX	CQ_TXEN	–	–	–	–	–	ALTPDIN_INV
Reset	0b0		–	–	–	–	–	
Access Type	Write, Read	Write, Read	–	–	–	–	–	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
CQ_TX	7	**C/Q Driver Logic Control Bit** Set the CQ_TX and CQ_TXEN bits to control the C/Q driver when enabled.	0 = C/Q is high (CQ_TXEN = 1, CQ_EN = 1) 1 = C/Q is low (CQ_TXEN = 1, CQ_EN = 1)
CQ_TXEN	6	**C/Q Driver Enable Control Bit** Set the CQ_TX and CQ_TXEN bits to control the C/Q driver when enabled.	0 = C/Q driver is disabled (CQ_EN = X, CQ_TX = X) 1 = C/Q driver is enabled (CQ_EN = 1)
ALTPDIN_INV	0	**GPIO-to-PDIn Logic Invert Enable** Set this bit to invert the logic state of the PDIn bit logic value when a comparator output or GPIO is configured as a PDIn bit. See the PDIN1, PDIN2 (GPIO1_4, GPIO1_5) section and/or the Process Data Input Using the Comparator (CMPO) section for more information.	0 = PDIn bit(s) follow the logic state of the selected input(s). 1 = PDIn bit(s) are opposite the logic state of the selected input(s).

IOL_RX_CTRL (0x40040164)

Enable the receiver filter.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	RX_FILTER	–	–	–	–
Reset	–	–	–	0b0	–	–	–	–
Access Type	–	–	–	Write, Read	–	–	–	–

BITFIELD	BITS	DESCRIPTION	DECODE
RX_FILTER	4	**C/Q Receiver Glitch Filter Enable** Set this bit to enable the glitch filter on the C/Q receiver. When RX_FILTER = 1, pulses less than 1µs (typ) on C/Q are ignored.	0 = C/Q receiver filter is disabled 1 = C/Q receiver filter is enabled

IOL_MISC_CTRL (0x40040168)

Enable the 2mA pull-up/pull-down on C/Q.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	CQPUD_2mA	–
Reset	–	–	–	–	–	–	0b0	–

Access Type	–	–	–	–	–	–	Write, Read	–

BITFIELD	BITS	DESCRIPTION	DECODE
CQPUD_2mA	1	**C/Q 2mA Pull-Up/Pull-Down Enable** Set this bit to enable the 2mA pull-up/pull-down on C/Q. Enable the 2mA pull-up by setting CQ_PU = 1 in the IOL_CQ_CTRL1 register and CQPUD_2mA = 1. Enable the 2mA pull-down by setting CQ_PD = 1 in the IOL_CQ_CTRL1 register and CQPUD_2mA = 1. The internal weak pull-up/pull-down is enabled when CQ_PU = 1 and/or CQ_PD = 1 in the IOL_CQ_CTRL1 register and CQPUD_2mA = 0.	0 = 2mA load is disabled 1 = 2mA load is enabled

CLOCK CONTROL REGISTERS

ADDRESS	NAME	MSB							LSB
CLOCK CONTROL									
0x40030004	CLK_CTRL_SOURCE[7:0]	–	EXT_NOT_POSC	EXT_NOT_INT	–	MCU_CLK_DIV[3:0]			
0x40030008	CLK_CTRL_PLL_CFG[7:0]	COMMIT	RDIV[4:0]					PLL_OUT_SEL[1:0]	
0x4003000C	CLK_CTRL_OPT[7:0]	COMMIT_INT_EN	CLK_CTRL_EN	–	–	–	OTP_CLK_EN	–	–
0x40030030	CLK_CTRL_STATUS[7:0]	POSC_READY	POSC_RDY_TMO	CLK_FAULT	CLK_FAULT_TMO	PLL_STUCK	PLL_LOCK	–	CLK_FAIL
0x40030034	CLK_CTRL_LATCH_RE[7:0]	POSC_RDY_LTC_RE	POSC_RDY_TMO_LTC_RE	CLK_FAULT_RESET_RE	CLK_FAULT_TMO_LTC_RE	PLL_STUCK_RESET_RE	PLL_LOCK_RE	PLL_ERR_LTC_RE	CLK_FAIL_LTC_RE
0x40030038	CLK_CTRL_LATCH_FE[7:0]	POSC_RDY_FE_LTC	–	CLK_OK	–	–	PLL_LOCK_FE	–	–
0x40030040	CLK_CTRL_INT_ENA_RE[7:0]	POSC_RDY_RE_INT_EN	POSC_RDY_TMO_INT_EN	CLK_FLT_INT_EN	CLK_FLT_TMO_INT_EN	PLL_STUCK_INT_EN	PLL_LOCK_RE_INT_EN	PLL_ERR_INT_EN	CLK_FAIL_INT_EN
0x40030044	CLK_CTRL_INT_ENA_FE[7:0]	POSC_RDY_FE_INT_EN	–	CLK_OK_INT_EN	–	–	PLL_LOCK_FE_INT_EN	–	–

Register Details

CLK_CTRL_SOURCE (0x40030004)

Select the clock source and set the HCLK system clock divider.

BIT	7	6	5	4	3	2	1	0
Field	–	EXT_NOT_POSC	EXT_NOT_INT	–	MCU_CLK_DIV[3:0]			
Reset	–	0b0	0b0	–				
Access Type	–	Write, Read	Write, Read	–	Write, Read			

BITFIELD	BITS	DESCRIPTION	DECODE
EXT_NOT_POSC	6	**Clock Source Select** Set this bit to configure the device to use an external clock (MCLK) or the internal 921.6kHz (typ) precise oscillator clock. When EXT_NOT_POSC = 1, configure GPIO2_0 as MCLK input and apply an external clock to GPIO2_0	0 = Use the internal 921.6kHz precise oscillator clock. 1 = Use the MCLK external clock.
EXT_NOT_INT	5	**PLL Source Select** Set this bit to select the input clock for the PLL.	0 = Use the internal 18MHz raw oscillator. 1 = Use the MCLK external clock, or 921.6kHz precise oscillator, as set by the EXT_NOT_POSC bit.
MCU_CLK_DIV	3:0	**HCLK Clock Divider Select** Program these bits to select the clock divider for the Cortex-M0 clock (HCLK). **Note:** Increasing this divider value slows the peripheral clocks.	0x0 = Full clock speed (Divide by 1) 0x1 = Divide by 2 0x2 = Divide by 3 ... 0xE = Divide by 15 0xF = Divide by 16

CLK_CTRL_PLL_CFG (0x40030008)

Configure the PLL and commit the clock configuration.

BIT	7	6	5	4	3	2	1	0
Field	COMMIT	RDIV[4:0]					PLL_OUT_SEL[1:0]	
Reset	0b0	0b00000					0b00	
Access Type	Write, Read	Write, Read					Write, Read	

BITFIELD	BITS	DESCRIPTION	DECODE
COMMIT	7	**Clock Changes Commit** Set this bit to commit changes to the clock control registers. Changes to the clock control registers are ignored until this bit is set to 1. This bit is automatically cleared after the registers are updated.	0 = Changes to the clock control registers are ignored. 1 = Changes to the clock control registers are committed.
RDIV	6:2	**PLL Input Clock Divider** Set these bits to configure the input divider that generates the 921.6kHz ±5% clock input for the PLL.	0x00 = Divided by 1 0x01 = Divided by 2 ... 0x1E = Divided by 16 0x1F = Divided by 32

BITFIELD	BITS	DESCRIPTION	DECODE
		NOTE: An error is generated when the PLL input clock falls outside of the allowed range.	
PLL_OUT_SEL	1:0	**PLL Output Source Select** Set these bits to select either the 18MHz raw oscillator clock or the PLL output clock. The system clock is the selected clock divided by 2. **Note:** The 18MHz raw oscillator is not recommended for IO-Link communication.	00 = Clock source is 18MHz raw oscillator. 01 = Clock source is PLL output. 10, 11 = Rserved. Do not use.

CLK_CTRL_OPT (0x4003000C)

Set the system clock interrupt, enable, and OTP clock enable.

BIT	7	6	5	4	3	2	1	0
Field	COMMIT_INT_EN	CLK_CTRL_EN	–	–	–	OTP_CLK_EN	–	–
Reset	0b0	0b0	–	–	–	0b0	–	–
Access Type	Write, Read	Write, Read	–	–	–	Write, Read	–	–

BITFIELD	BITS	DESCRIPTION	DECODE
COMMIT_INT_EN	7	**Clock Changes Committed Interrupt Enable** Set this bit to enable an interrupt after the clock changes are committed (COMMIT bit is cleared).	0 = Interrupt is not generated when COMMIT sequence is complete 1 = Interrupt is generated when COMMIT sequence is complete
CLK_CTRL_EN	6	**Clock Control Enable Bit** This bit enables the internal clock control circuitry. For normal operation, set this bit (CLK_CTRL_EN = 1) and configure the PLL as described in the Clock Control section. This bit is reset to 0 at power-up. The clock control circuitry is disabled and the digital clock source is the 18MHz raw oscillator when CLK_CTRL_EN = 0.	0 = Clock control circuitry is not enabled. The system clock source is the 18MHz raw oscillator. 1 = Clock control circuitry is enabled. Configure the PLL and clock as described in the Clock Control section.
OTP_CLK_EN	2	**OTP Clock Enable** Set this bit to enable the OTP clock. To reduce power, set OTP_CLK_EN = 0 when the OTP is not being accessed.	0 = OTP clock is disabled. 1 = OTP clock is enabled.

CLK_CTRL_STATUS (0x40030030)

Various clock ready and fault indicators

BIT	7	6	5	4	3	2	1	0
Field	POSC_READY	POSC_RDY_TMO	CLK_FAULT	CLK_FAULT_TMO	PLL_STUCK	PLL_LOCK	–	CLK_FAIL
Reset	0b1	0b0	0b1	0b0	0b1	0b0	–	0b0

Access Type	Read Only	Read Only	Read Only	Read Only	Read Only	Read Only	–	Read Only

BITFIELD	BITS	DESCRIPTION	DECODE
POSC_READY	7	**921.6kHz Precise Oscillator Ready Indicator** This bit is set when the internal 921.6kHz precise osillator is enabled and stable.	0 = 921.6kHz precise oscillator is not ready. 1 = 921.6kHz precise oscillator is ready.
POSC_RDY_TMO	6	**921.6kHz Precise Oscillator Ready Timeout Indicator** This bit is set when the 8ms (typ) precise-oscillator-ready-timeout has expired.	0 = 921.6kHz precise oscillator timeout is not expired. 1 = 921.6kHz precise oscillator 8ms timeout has expired.
CLK_FAULT	5	**Clock Fault Indicator** This bit is set when a fault occurs on the 921.6kHz precise oscillator or MCLK external clock signal, as selected in the CLK_CTRL_SOURCE register. A fault occurs when the clock frequency is incorrect or missing. Poll this bit to verify PLL input signal, f_{SOURCE}, before activating PLL. When the system clock is derived from an external clock (MCLK) or the 921.6kHz precise oscillator, any clock fault immediately generates a system reset. Write a 1 to this bit to clear it.	0 = No clock fault was detected 1 = Clock fault was detected
CLK_FAULT_TMO	4	**PLL Clock Input Fault Indicator** This bit is set when the 80µs PLL clock input detection timeout has expired.	0 = Timeout is not expired 1 = Timeout is expired
PLL_STUCK	3	**PLL Stuck Fault Indicator** This bit is set when the PLL is stuck. The PLL is automatically reset when the PLL_STUCK = 1. Set COMMIT = 1 to restart the PLL and clear this bit.	0 = PLL is not stuck 1 = PLL is stuck
PLL_LOCK	2	**PLL Locked Indicator** This bit is set when the PLL is locked.	0 = PLL is not locked 1 = PLL is locked
CLK_FAIL	0	**Clock Faul Fault Indicator** This bit is set when any of the following bits are set: CLK_FAULT CLK_FAULT_TMO PLL_STUCK PLL_LOCK POSC_RDY_TMO	

CLK_CTRL_LATCH_RE (0x40030034)

BIT	7	6	5	4	3	2	1	0
Field	POSC_RDY_LTC_RE	POSC_RDY_TMO_LTC_RE	CLK_FAULT_RESET_RE	CLK_FAULT_TMO_LTC_RE	PLL_STUCK_RESET_RE	PLL_LOCK_RE	PLL_ERR_LTC_RE	CLK_FAIL_LTC_RE

Reset	0b0	0b0	0b0	0b0	0b0	0b0	0b0	0b0
Access Type	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read

BITFIELD	BITS	DESCRIPTION	DECODE
POSC_RDY_LTC_RE	7	**921.6kHz Precise Oscillator Ready Indicator** This bit is set when the 921.6kHz precise oscillator (f_{POSC}) is stable and ready. This bit is latched when set. Write a 1 to this bit to clear it.	0 = 921.6kHz precise oscillator is not ready. 1 = 921.6kHz precise oscillator is ready or was ready since the bit was last cleared.
POSC_RDY_TMO_LTC_RE	6	**921.6kHz Precise Oscillator 2ms Timeout Expired Indicator** This bit is set when the 921.6kHz precise oscillator (f_{POSC}) 8ms (typ) timeout has expired. This bit is latched when set. Write a 1 to this bit to clear it.	0 = f_{POSC} 2ms timeout is not expired. 1 = f_{POSC} 2ms timeout is expired or has expired since the bit was last cleared.
CLK_FAULT_RESET_RE	5	**Clock Fault to System Reset Indicator** This bit is set when a clock fault has caused a system reset since the last power-up. This bit is latched when set. Write a 1 to this bit to clear it.	0 = Clock fault has not caused a system reset since the last power-up. 1 = Clock fault has caused a system reset since the last power-up.
CLK_FAULT_TMO_LTC_RE	4	**PLL Clock Input 80µs Timeout Expired Indicator** This bit is set when the PLL clock input 80µs (typ) timeout has expired. This bit is latched when set. Write a 1 to this bit to clear it.	0 = 80µs timeout is not expired. 1 = 80µs timeout is expired or has expired since the bit was last cleared.
PLL_STUCK_RESET_RE	3	**PLL Stuck Fault to System Reset Indicator** This bit is set when a PLL stuck fault has caused a system reset since last power-up. This bit is latched when set. Write a 1 to this bit to clear it.	0 = PLL stuck fault has not caused a system reset since the last power-up. 1 = PLL stuck fault has caused a system reset since the last power-up.
PLL_LOCK_RE	2	**PLL Locked Indicator** This bit is set when the PLL is locked (PLL_LOCK is set). This bit is latched when set. Write a 1 to this bit to clear it.	0 = PLL_LOCK is set. PLL is locked/latched. 1 = PLL_LOCK is not set. PLL is stuck or has been stuck since the bit was last cleared.
PLL_ERR_LTC_RE	1	**PLL Error Indicator** This bit is set when a PLL error has occurred. This bit is latched when set. Write a 1 to this bit to clear it.	0 = PLL error has not occurred. 1 = PLL error has occurred or an error has occurred since the bit was last cleared.

BITFIELD	BITS	DESCRIPTION	DECODE
CLK_FAIL_LTC_RE	0	**Clock Control Fail Indicator** This bit is set when the internal clock control state machine has failed. This bit is latched when set. Write a 1 to this bit to clear it.	0 = Clock control state machine has not failed. 1 = Clock control state machine has occurred since the bit was last cleared.

CLK_CTRL_LATCH_FE (0x40030038)

BIT	7	6	5	4	3	2	1	0
Field	POSC_RDY_FE_LTC	–	CLK_OK	–	–	PLL_LOCK_FE	–	–
Reset	0b0	–	0b0	–	–	0b0	–	–
Access Type	Write 1 to Clear, Read	–	Write 1 to Clear, Read	–	–	Write 1 to Clear, Read	–	–

BITFIELD	BITS	DESCRIPTION	DECODE
POSC_RDY_FE_LTC	7	**921.6kHz Precise Oscillator Not Ready Indicator** This bit is set when the f_{POSC} oscillator is not ready. This bit is latched when set. Write a 1 to this bit to clear it.	0 = f_{POSC} is ready. 1 = f_{POSC} is not ready or has been not ready since the bit was last cleared.
CLK_OK	5	**Clock Fault to System Reset Indicator** This bit is set when the PLL input clock is ready and stable. This bit is latched when set. Write a 1 to this bit to clear it.	0 = Clock is not ready and stable. 1 = Clock is ready and stable.
PLL_LOCK_FE	2	**PLL Not Locked Indicator** This bit is set when the PLL is not locked/latched (PLL_LOCK = 0). This bit is latched when set. Write a 1 to this bit to clear it.	0 = PLL_LOCK is set. PLL is locked/latched. 1 = PLL_LOCK is cleared. PLL is not stuck or has been not stuck since the bit was last cleared.

CLK_CTRL_INT_ENA_RE (0x40030040)

BIT	7	6	5	4	3	2	1	0
Field	POSC_RDY_RE_INT_EN	POSC_RDY_TMO_INT_EN	CLK_FLT_INT_EN	CLK_FLT_TMO_INT_EN	PLL_STUCK_INT_EN	PLL_LOCK_RE_INT_EN	PLL_ERR_INT_EN	CLK_FAIL_INT_EN
Reset	0b0	0b0	0b0	0b0	0b0	0b0	0b0	0b0

Access Type	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
POSC_RDY_RE_INT_EN	7	**921kHz Precise Oscillator Ready Interrupt Enable (Rising Edge)** Write a 1 to this bit to trigger an interrupt when the POSC_RDY bit is set.	0 = No interrupt when POSC_RDY is set 1 = Interrupt is triggered when POSC_RDY is set
POSC_RDY_TMO_INT_EN	6	**921kHz Precise Oscillator 8ms Timeout Interrupt Enable (Rising Edge)** Write a 1 to this bit to trigger an interrupt when the POSC_RDY_TMO bit is set.	0 = No interrupt when POSC_RDY is set. 1 = Interrupt is triggered when for POSC_RDY is set.
CLK_FLT_INT_EN	5	**Clock Fault Indicator Interrupt Enable** Write 1 to trigger an interrupt when the CLK_FAULT bit is set.	0 = No interrupt when CLK_FAULT is set. 1 = Interrupt is triggered when CLK_FAULT is set.
CLK_FLT_TMO_INT_EN	4	**Clock Fault Indicator Interrupt Enable** Write 1 to this bit to trigger an interrupt when CLK_FAULT_TMO is set.	0 = No interrupt when CLK_FAULT_TMO is set. 1 = Interrupt is triggered when CLK_FAULT_TMO is set.
PLL_STUCK_INT_EN	3	**PLL Stuck Fault Indicator Interrupt Enable** Write 1 to this bit to trigger an interrupt when PLL_STUCK is set.	0 = No interrupt when PLL_STUCK is set. 1 = Interrupt is triggered when PLL_STUCK is set.
PLL_LOCK_RE_INT_EN	2	**PLL Locked Indicator Interrupt Enable (Rising Edge)** Write 1 to this bit to trigger an interrupt when PLL_LOCK is set.	0 = No interrupt when PLL_LOCK is set. 1 = Interrupt is triggered when PLL_LOCK is set.
PLL_ERR_INT_EN	1	**PLL Error Indicator Interrupt Enable** Write 1 to this bit to trigger an interrupt when PLL_ERR_LTC is set.	0 = No interrupt when PLL_ERR_LTC is set. 1 = Interrupt is triggered when PLL_ERR_LTC is set.
CLK_FAIL_INT_EN	0	**Clock Faul Fault Indicator Interrupt Enable** Write 1 to this bit to trigger an interrupt when CLK_FAIL is set.	0 = No interrupt when CLK_FAIL is set. 1 = Interrupt is triggered when CLK_FAIL is set.

CLK_CTRL_INT_ENA_FE (0x40030044)

BIT	7	6	5	4	3	2	1	0
Field	POSC_RDY_FE_INT_EN	–	CLK_OK_INT_EN	–	–	PLL_LOCK_FE_INT_EN	–	–
Reset	0b0	–	0b0	–	–	0b0	–	–
Access Type	Write, Read	–	Write, Read	–	–	Write, Read	–	–

BITFIELD	BITS	DESCRIPTION	DECODE
POSC_RDY_FE_INT_EN	7	**921kHz Precise Oscillator Ready Interrupt Enable (Falling Edge)** Write a 1 to this bit to trigger an interrupt when the POSC_RDY bit is cleared.	0 = No interrupt when POSC_RDY is cleared. 1 = Interrupt is triggered when POSC_RDY is cleared.
CLK_OK_INT_EN	5	**Clock OK Indicator Interrupt Enable** Write 1 to trigger an interrupt when the CLK_OK bit is set.	0 = No interrupt when CLK_OK is set. 1 = Interrupt is triggered when CLK_OK is set.
PLL_LOCK_FE_INT_EN	2	**PLL Not Locked Indicator Interrupt Enable** Write 1 to this bit to trigger an interrupt when PLL_LOCK is cleared (PLL_LOCL_FE = 1).	0 = No interrupt when PLL_LOCK is cleared. 1 = Interrupt is triggered when PLL_LOCK is cleared.

ENHANCED TIMER REGISTERS

ADDRESS	NAME	MSB							LSB
EHHANCED TIMER									
0x40080000	ENH_TIMER_COUNTER[31:24]	TIMER_COUNTER[31:24]							
	ENH_TIMER_COUNTER[23:16]	TIMER_COUNTER[23:16]							
	ENH_TIMER_COUNTER[15:8]	TIMER_COUNTER[15:8]							
	ENH_TIMER_COUNTER[7:0]	TIMER_COUNTER[7:0]							
0x40080004	ENH_TIMER_LIMIT[31:24]	TIMER_LIMIT[31:24]							
	ENH_TIMER_LIMIT[23:16]	TIMER_LIMIT[23:16]							
	ENH_TIMER_LIMIT[15:8]	TIMER_LIMIT[15:8]							
	ENH_TIMER_LIMIT[7:0]	TIMER_LIMIT[7:0]							
0x40080008	ENH_TIMER_COMPARE[31:24]	TIMER_COMPARE[31:24]							
	ENH_TIMER_COMPARE[23:16]	TIMER_COMPARE[23:16]							
	ENH_TIMER_COMPARE[15:8]	TIMER_COMPARE[15:8]							
	ENH_TIMER_COMPARE[7:0]	TIMER_COMPARE[7:0]							

ADDRESS	NAME	MSB							LSB
0x40080 00C	ENH_TIMER_CONFIG[7:0]	–	–	–	–	–	TOGGLE	PWM_EN	ENABLE_CYCLE
0x40080 010	ENH_TIMER_IRQ_CTRL[7:0]	–	–	–	–	EXT_THRESH_INT_EN	STOP_INT_EN	COMP_MATCH_INT_EN	OVER_INT_EN
0x40080 014	ENH_TIMER_STATUS[7:0]	–	–	–	EXT_OVR_INT	EXT_THRESH_INT	STOP_INT	COMP_MATCH_INT	OVER_INT
0x40080 018	ENH_TIMER_EXT_COUNTER[15:8]	TIMER_EXT_COUNTER[15:8]							
	ENH_TIMER_EXT_COUNTER[7:0]	TIMER_EXT_COUNTER[7:0]							
0x40080 01C	ENH_TIMER_EXT_THRES[15:8]	TIMER_EXT_THRES[15:8]							
	ENH_TIMER_EXT_THRES[7:0]	TIMER_EXT_THRES[7:0]							
0x40080 020	ENH_TIMER_EXT_CONFIG[7:0]	–	–	EXT_TRG_GPIO_SEL[3:0]				EXT_TRG_RISE	EXT_TRG_FALL
0x40080 024	ENH_TIMER_OUT_CONFIG[7:0]	–	PWM_TRG_GPIO_SEL[3:0]				PWM_CAPTURE_RISE	PWM_CAPTURE_FALL	PWM_TRG_GPIO_EN

Register Details

ENH_TIMER_COUNTER (0x40080000)

Enhanced timer counter value

BIT	31	30	29	28	27	26	25	24
Field	TIMER_COUNTER[31:24]							
Reset								
Access Type	Read Only							
BIT	23	22	21	20	19	18	17	16
Field	TIMER_COUNTER[23:16]							
Reset								
Access Type	Read Only							

BIT	15	14	13	12	11	10	9	8
Field	TIMER_COUNTER[15:8]							
Reset								
Access Type	Read Only							
BIT	7	6	5	4	3	2	1	0
Field	TIMER_COUNTER[7:0]							
Reset								
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
TIMER_COUNTER	31:0	**Enhanced Timer Counter Value** This register contains the value of the enhanced timer counter.

ENH_TIMER_LIMIT (0x40080004)

Set the enhanced timer counter overflow limit.

BIT	31	30	29	28	27	26	25	24
Field	TIMER_LIMIT[31:24]							
Reset								
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	TIMER_LIMIT[23:16]							
Reset								
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	TIMER_LIMIT[15:8]							
Reset								
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	TIMER_LIMIT[7:0]							
Reset								
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
TIMER_LIMIT	31:0	**Enhanced Timer Counter Overflow Limit** Program the enhanced timer counter overflow limit in this register. When the ENH_TIMER_COUNTER register value reaches the ENH_TIMER_LIMIT, the TIMER_COUNTER value is reset to 0 on the following clock cycle. See the Enhanced Timer section for more information.

ENH_TIMER_COMPARE (0x40080008)

Set the enhanced timer compare value.

BIT	31	30	29	28	27	26	25	24
Field	TIMER_COMPARE[31:24]							
Reset								
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	TIMER_COMPARE[23:16]							
Reset								
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	TIMER_COMPARE[15:8]							
Reset								
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	TIMER_COMPARE[7:0]							
Reset								

Access Type	Write, Read

BITFIELD	BITS	DESCRIPTION
TIMER_COMPARE	31:0	**Enhanced Timer Counter Compare Value** Program the enhanced timer compare limit in this register. The ratio between the ENH_TIMER_COMPARE register and the ENH_TIMER_LIMIT register defines the PWM signal duty cycle. See the Enhanced Timer section for more information.

ENH_TIMER_CONFIG (0x4008000C)

Configure the enhanced timer.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	TOGGLE	PWM_EN	ENABLE_CYCLE
Reset	–	–	–	–	–			
Access Type	–	–	–	–	–	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
TOGGLE	2	**Enhanced Timer Output Toggle Enable Bit** Set this bit to enable the enhanced timer to toggle GPIO1_3 (configured as a PWM output) when the TIMER_COUNTER reaches the value set in the TIMER_LIMIT register. See the Enhanced Timer section for more information.	0 = TOGGLE output signal is not generated 1 = The output signal TOGGLES only one time when basic timer counter overflow event occurs (ENABLE_CYCLE = 0).
PWM_EN	1	**Enhanced Timer Output PWM Enable Bit** Set this bit to enable a PWM output signal on GPIO1_3 generated by the enhanced timer. The PWM duty cycle and frequency are set by the TIMER_COMPARE and TIMER_LIMT registers. See the Enhanced Timer section for more information.	0 = PWM signal is not generated 1 = A single pulse PWM signal is generated. If ENABLE_CYCLE = 0, the counter starts, reaches overflow limit value, and then stops at 0. If ENABLE_CYCLE = 1, the PWM signal is generated and pulses continuously until ENABLE_CYCLE bit is cleared.
ENABLE_CYCLE	0	**Enhanced Timer Multicycle Enable Bit** Set ENABLE_CYCLE = 1 to restart the enhanced timer counter from 0 following an overflow event. This bit is write-protected by PWM_TRG_GPIO_EN. If PWM_TRG_GPIO_EN = 1, the ENABLE_CYCLE bit value is 0. See the Enhanced Timer section for more information.	0 = The basic timer counter is enabled for a single-cycle if TOGGLE = 1 or PWM_EN =1. If TOGGLE and PWM_EN are both 0 or 1, the basic counter is disabled. 1 = The multicycle timer counter is enabled and restarts from 0 after any overflow event (ENH_TIMER_COUNTER = ENH_TIMER_LIMIT) occurs.

ENH_TIMER_IRQ_CTRL (0x40080010)

Enable the enhanced timer and external counter interrupts.

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	EXT_THRESH_INT_EN	STOP_INT_EN	COMP_MATCH_INT_EN	OVER_INT_EN

Reset	–	–	–	–				
Access Type	–	–	–	–	Write, Read	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
EXT_THRESH_INT_EN	3	**Enhanced Timer External Event Counter Overflow Interrupt Enable Bit** Set this bit to enable an interrupt event when the external event counter reaches the programmed threshold value. See the Enhanced Timer section for more information. Write a 1 to this bit to clear it.	0 = Interrupt is not generated 1 = Interrupt is generated
STOP_INT_EN	2	**Timer Counter Stop Interrupt Enable Bit** Set this bit to enable an interrupt event when the enhanced timer stops. See the Enhanced Timer section for more information. Write a 1 to this bit to clear it.	0 = Interrupt is not generated 1 = Interrupt is generated
COMP_MATCH_INT_EN	1	**Compare Match Event Interrupt Enable Bit** Set this bit to enable an interrupt when the enhanced timer counter reaches compare value. See the Enhanced Timer section for more information. Write a 1 to this bit to clear it.	0 = Interrupt is not generated 1 = Interrupt is generated
OVER_INT_EN	0	**Overflow Event Interrupt** Set this bit to enable an interrupt when an overflow event occurs (e.g. timer counter reaches limit value). See the Enhanced Timer section for more information. Write a 1 to this bit to clear it.	0 = Interrupt is not generated 1 = Interrupt is generated

ENH_TIMER_STATUS (0x40080014)

Enhanced timer status and interrupt flags

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	EXT_OVR_INT	EXT_THRESH_INT	STOP_INT	COMP_MATCH_INT	OVER_INT
Reset	–	–	–					
Access Type	–	–	–	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read

BITFIELD	BITS	DESCRIPTION	DECODE
EXT_OVR_INT	4	**External Event Counter Overflow Interrupt** This bit is set when the external event counter reaches the 16-bit overflow value (65,535). Write a 1 to this bit to clear it. This interrupt bit is also cleared when the ENH_EXT_THRES_IRQ bit is written.	0 = An external counter overflow event has not occurred since the last interrupt was cleared. 1 = An external counter overflow event has occurred since the last interrupt was cleared.
EXT_THRESH_INT	3	**External Event Counter Threshold Interrupt** This bit is set when the external event counter reaches the threshold value. Write a 1 to this bit to clear it.	0 = An external counter threshold event has not occurred since the last interrupt was cleared. 1 = An external counter threshold event has occurred since the last interrupt was cleared.
STOP_INT	2	**Enhanced Timer Counter Stop Interrupt** This bit is set when the enhanced timer is stopped. This happens when the timer counter value is 0 and ENABLE_CYCLE = 0. Write a 1 to this bit to clear it.	0 = The timer has not stopped running since the interrupt was last cleared. 1 = The timer has stopped running since the interrupt was last cleared.
COMP_MATCH_INT	1	**Enhanced Timer Compare Match Event Interrupt** This bit is set when the ENH_TIMER_COUNTER value reaches the programmed ENH_TIMER_COMPARE value. See the Enhanced Timer section for more information. Write a 1 to this bit to clear it.	0 = Enhanced timer compare match event has not occurred since the interrupt was last cleared. 1= Enhanced timer compare match event has occurred since the interrupt was last cleared.
OVER_INT	0	**Enhanced Timer Overflow Event Interrupt** This bit is set when the ENH_TIMER_COUNTER value reaches the ENH_TIMER_LIMIT value and restarts. Write a 1 to this bit to clear it.	0 = Enhanced timer overflow event has not occured since the interrupt was last cleared. 1 = Enhanced timer overflow event has occured since the interrupt was last cleared.

ENH_TIMER_EXT_COUNTER (0x40080018)

External counter value

BIT	15	14	13	12	11	10	9	8
Field	TIMER_EXT_COUNTER[15:8]							
Reset								
Access Type	Read Only							
BIT	7	6	5	4	3	2	1	0
Field	TIMER_EXT_COUNTER[7:0]							
Reset								
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
TIMER_EXT_COUNTER	15:0	**External Event Counter Value** The ENH_TIMER_EXT_COUNTER register value is incremented on rising/falling edges of an input signal as set in the ENH_TIMER_EXT_CONFIG register. The external event counter overflow limit is 65,535 and it is not programmable. See the External Event Counter section for more information.

ENH_TIMER_EXT_THRES (0x4008001C)

Set the external event counter threshold value.

BIT	15	14	13	12	11	10	9	8
Field	TIMER_EXT_THRES[15:8]							
Reset								
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	TIMER_EXT_THRES[7:0]							
Reset								
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
TIMER_EXT_THRES	15:0	**External Event Counter Interrupt Threshold** Program this register to set the external event counter threshold. When the external event counter value (ENH_TIMER_EXT_COUNTER) reaches this value, the EXT_THRESH_INT bit is set and an interrupt is generated.

ENH_TIMER_EXT_CONFIG (0x40080020)

Configure the external event counter.

BIT	7	6	5	4	3	2	1	0
Field	–	–	EXT_TRG_GPIO_SEL[3:0]				EXT_TRG_RISE	EXT_TRG_FALL
Reset	–	–						
Access Type	–	–	Write, Read				Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
EXT_TRG_GPIO_SEL	5:2	**GPIO Input Select for External Event Counter** Set these bits to select the GPIO input signal that triggers an increase in the external event counter. See the Enhanced Timer section for more information.	0x00 - 0x06: Select GPIO1_0 to GPIO1_6 0x07 - 0x0A: Select GPIO2_1 to GPIO2_4
EXT_TRG_RISE	1	**External Event Rising Edge Trigger Select** Set this bit to trigger the external event counter on the rising edge of the input signal.	0 = The external event counter is not triggered on the rising edge of the input signal. 1 = The external event counter is triggered on the rising edge of the input signal.
EXT_TRG_FALL	0	**External Event Falling Edge Trigger Select** Set this bit to trigger the external event counter on the falling edge of the input signal.	0 = The external event counter is not triggered on the falling edge of the input signal. 1 = The external event counter is triggered on the falling edge of the input signal.

ENH_TIMER_OUT_CONFIG (0x40080024)

Configure the PWM and external event counter.

BIT	7	6	5	4	3	2	1	0
Field	–	PWM_TRG_GPIO_SEL[3:0]				PWM_CAPTURE_RISE	PWM_CAPTURE_FALL	PWM_TRG_GPIO_EN
Reset	–							
Access Type	–	Write, Read				Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
PWM_TRG_GPIO_SEL	6:3	**GPIO Input Select for PWM Output** Set these bits to select the GPIO input signal that triggers PWM output generation on GPIO1_3 when GPIO1_3 is configured as a PWM output. See the Enhanced Timer section for more information.	0x00 - 0x06: Select GPIO1_0 to GPIO1_6 0x07 - 0x0A: Select GPIO2_1 to GPIO2_4
PWM_CAPTURE_RISE	2	**PWM Output Rising Edge Trigger Select** Set this bit to trigger the PWM output on the rising edge of the input signal when PWM_TRG_GPIO_EN = 1. See the Enhanced Timer section for more information.	0 = PWM output is not triggered on the rising edge of the selected input signal. 1 = PWM output is triggered on the rising edge of the selected input signal.
PWM_CAPTURE_FALL	1	**PWM Output Falling Edge Trigger Select** Set this bit to trigger the PWM output on the falling edge of the input signal when PWM_TRG_GPIO_EN = 1. See the Enhanced Timer section for more information.	0 = PWM output is not triggered on the falling edge of the selected input signal. 1 = PWM output is triggered on the falling edge of the selected input signal.
PWM_TRG_GPIO_EN	0	**Enhanced Timer PWM Output Enable Bit** Set this bit to enable the PWM output to trigger when the selected input signal rises or falls. When set, this bit overrides the ENABLE_CYCLE bit in the	0 = PWM/TOGGLE output is triggered by the enhanced timer counter. 1 = PWM/TOGGLE output is triggered by the selected GPIO input signal.

BITFIELD	BITS	DESCRIPTION	DECODE
		ENH_TIMER_CONFIG register. See the Enhanced Timer section for more information.	