Inventek Systems
Embedding Connectivity Everywhere

ISM14585-L35-P8

BLE 5.0 SiP

Preliminary Data Sheet

5.0 BLE + Cortex M0 + PMU + PA + 8Mb Flash

Table of Contents

1 PART NUMBER DETAIL DESCRIPTION

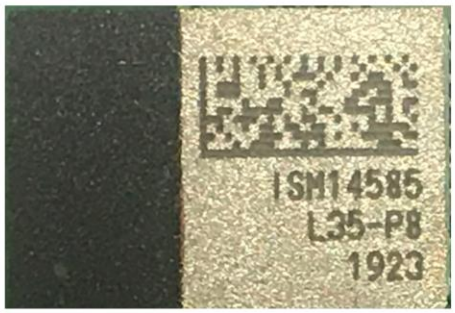
1.1 Ordering Information

Device	Description	Standard Ordering Number
ISM14585 Module Supports both **Internal Antenna or w.fl External Antenna options**.	Certified BLE 5.0 + Cortex M0 Module with integrated PMU, PA, 8Mb of Flash and supports both a certified internal antenna or certified w.fl external antenna	**ISM14585-L35-P8**
ISM14585-EVB **Internal Antenna** Evaluation Board	BLE 5.0 + Cortex M0 Module with integrated PMU, PA, 8Mb of Flash and internal antenna EVB (Evaluation Board)	**ISM14585-L35-P8-EVB**
ISM14585-EVB-W **w.fl External Antenna** Evaluation Board	BLE 5.0 + Cortex M0 Module with integrated PMU, PA, 8Mb of Flash and w.fl external antenna EVB (Evaluation Board)	**ISM14585-L35-P8-EVB-W**
B24P-W **w.fl Certified External Antenna**	Certified w.fl External Antenna to accompany the ISM14585-L35-P8-EVB-W External Antenna EVB option	**B24P-W**

1.2 Ordering Information Configuration

Module Standard Ordering Number:

ISM14585-L35-P8



LGA 35, 6.0mm x 8.6mm x 1.2mm

Evaluation Board Standard Ordering Numbers:

ISM14585-L35-P8-EVB: Internal Antenna EVB Option

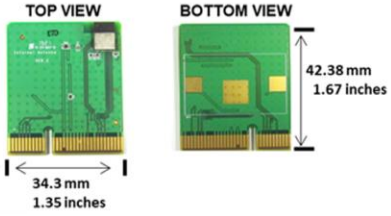
ISM14585-L35-P8-EVB-W: External w.fl Antenna EVB Option

- ISM14585-L35-P8-EVB-W requires the Inventek **B24P-W** Certified w.fl External Antenna

Evaluation Board Ordering Number:

ISM14585-EVB-X

Antenna:
- Blank = Internal Antenna
- W = w.fl External Antenna



B24P-W Certified w.fl External Antenna to accompany the ISM14585-L35-P8-EVB-W w.fl External Antenna Evaluation Board option

- Please refer to the ISM14585-L35-P8-EVB User's Manual for additional information on the ISM14585 Evaluation Board configuration options.

- For additional module configuration options please contact Inventek Systems
  - Audio Unit configuration
  - 4Mb or 16Mb integrated Flash
  - Other

2 OVERVIEW

The Inventek Systems **ISM14585-L35** SiP (System in Package), is one of the smallest, lowest power and most integrated Bluetooth® 5.0 solutions available. The **ISM14585-L35** platform is the first BLE module from Inventek's ***FLEXiBLE*** product family. The **ISM14585-L35** SiP is an embedded wireless Bluetooth Low Energy (BLE) IoT radio, based on the Dialog Semiconductor DA14585 radio SoC (System on Chip). The **ISM14585-L35** offers designers all the benefits of the industry-leading DA14580 technology but with even greater flexibility to create more advanced applications from the smallest footprints and power budgets.

The **ISM14585-L35** is ideal for applications such as remote controls, beacons, connected sensors and innovative medical devices. Among the new features, the **ISM14585-L35** supports Data Packet Length Extension, Link Layer Privacy v1.2, Secure Connections, Bluetooth low energy and Efficient connectable Advertising.

The **ISM14585-L35** integrated Audio Unit (AU) is equipped with a Pulse-Density Modulation (PDM) interface that can be connected to up to 2 input devices (e.g. MEMS microphones) or output devices, a PulseCode modulation (PCM) controller which provides an up to 192 kHz synchronous interface to external audio devices, ISDN circuits and serial data interfaces (I2S) and a 24-bit Sample Rate Converting unit (SRC) used to convert the sampling rate of audio samples between the various interfaces. PDM and PCM functionality can be mapped to any GPIO through the user programmable pin logic. An integrated DMA controller handles all data transfers between the AU and the RAM providing the CPU with the freedom to cope with other tasks. Please Note that the AU feature requires a version of the ISM14585 module without the integrated 8Mb of Flash. The Microphone feature requires I2S or PDM via the SPI interface.

The **ISM14585-L35** SiP provide smarter, more flexible, and even lower power BLE connectivity with an integrated 32bit CortexTM-M0 (16MHz), processor, an integrated PMU (Power Management Unit), an integrated PA (Power Amplifier), an integrated 128kB of ROM, 64KB OTP, 1MB SPI Flash as well as a foot print compatible options for an additional integrated 4Mb, 8Mb, or 16Mb of Flash.

The **ISM14585-L35** SiP provides a number of features and standard peripheral interfaces (see “Features” below), enabling a seamless connection to an embedded design. The **ISM14585-L35** is a very versatile SoC and is ideal for adding Bluetooth low energy to products like remote controls, proximity tags, beacons, connected medical devices and smart home nodes. The **ISM14585-L35** supports all Bluetooth developments up to and including Bluetooth 5.0.

The **ISM14585-L35** also includes 96KB of RAM, the **ISM14585-L35** has double the memory for User applications of its predecessor to take full advantage of the standard’s features. The **ISM14585-L35** also includes an integrated microphone interface for voice

support at no additional cost. The wide supply voltage range (0.9 –3.6 V) covering a larger choice of energy sources also enables full design flexibility.

The **ISM14585-L35** is easy to design-in and supports standalone as well as hosted applications. The **ISM14585-L35** is supported by a complete development environment with Dialog's SmartSnippets™ software that helps customers optimize software for power consumption.

The **ISM14585-L35** supports several fully qualified profiles embedded in ROM (see "Typical Applications" below), and the option of loading additional profiles into RAM. The low cost, small footprint (6.0mm x 8.6mm x 1.2mm), LGA 35 pin package and ease of design-in make the **ISM14585-L35** ideal for a wide range of embedded applications.

The **ISM14585-L35** enables wireless connectivity to the simplest existing sensor products with minimal engineering effort. **ISM14585-L35** reduces development time, lowers manufacturing costs, saves board space, simplifies certification compliance, and minimizes customer RF expertise required during development of target applications.

The **ISM14585-L35** provides the highest level of integration for a wireless system, with market leading and integrated BLE 5.0 technology based on Dialog's DA14585 SoC.
The **ISM14585-L35** is also fully supported by Dialog's Smartbond product family Development Kit-Pro evaluation board and Dialog SmartSnippets Studio SDK.

3 FEATURES

The **ISM14585-L35** provides a reduced boot time and supports up to 8 connections.
It has a fully integrated radio transceiver and baseband processor for Bluetooth® Low Energy. It can be used as a standalone application processor or as a data pump in hosted systems.

The **ISM14585-L35** is optimized for remote control units (RCU) requiring support for voice commands and motion/gesture recognition. Its integrated Audio Unit (AU) offers easy interface for MEMS microphones over PDM, external codecs over PCM/I2S and a Sample Rate Converter unit.

The **ISM14585-L35** Bluetooth Low Energy firmware includes the L2CAP service layer protocols, Security Manager (SM), Attribute Protocol (ATT), the Generic Attribute Profile (GATT) and the Generic Access Profile (GAP). All profiles published by the Bluetooth SIG as well as custom profiles are supported.

The transceiver interfaces directly to the antenna and is fully compliant with the Bluetooth 5.0 standard. The **ISM14585-L35** has dedicated hardware for the Link Layer

implementation of Bluetooth Low Energy and interface controllers for enhanced connectivity capabilities.

3.1 Feature Highlights:

- Frequency Band: 2.4GHz
- Complies to the Bluetooth 5 core specification
- Supports up to 8 Bluetooth LE connections
- Network Standard: Bluetooth Low Energy
- Longest battery life
- Operating voltage 3.3V
- Operating Temperature: -40℃ to 85℃
- MSL level 3
- Low system Bill of Materials
- FCC, CE, IE and Japan certification in-process
- Certifications will comply with Bluetooth V5.0, ETSI EN 300, 328 and EN 300 440 Class 2 (Europe), FCC CFR47 Part 15 (US) and ARIB STD-T66 (Japan)
- Processing power
    - 16 MHz 32 bit ARM Cortex-M0 with SWD
    - interface
    - Dedicated Link Layer Processor
    - AES-128 bit encryption Processor
- Memory Resources
    - Large memory to build complex applications
    - Integrated One-Time-Programmable memory 64 kB (OTP)
    - 96 kB Data/Retention SRAM
    - 128 kB ROM
    - 8Mb integrated Flash
- Power Management
    - Integrated Buck DCDC converter
    - P0, P1 and P2 ports with 3.3 V tolerance
    - Internal decoupling of the supply pins
    - Supports Coin (typ. 3.3 V)
    - 10-bit ADC for battery voltage measurement
    - Integrated Power Amplifier for maximum radio performance
- Digital controlled oscillators
    - 16 MHz crystal (±20 ppm max) and RC oscillator
    - 32 kHz crystal (±50 ppm, ±500 ppm max) and RCX oscillator
- Flexible Reset Circuitry
    - System & Power-On Reset in a single pin

- Fast cold boot in less than 50ms
- General-purpose, Capture and Sleep timers
- Digital interfaces
  - Gen. Purpose I/Os: 14
  - 2 x UARTs with hardware flow control up to 1 MBd
  - SPI+™ interface
  - I2C bus at 100 kHz, 400 kHz
  - 3-axes capable Quadrature Decoder
- Analog interfaces
  - 4-channel 10-bit ADC
- Radio transceiver
  - Fully integrated 2.4 GHz CMOS transceiver
  - Single wire antenna: no RF matching, or RX/TX switching required
  - Supply current at VBAT3V:
    - TX: 3.4 mA
    - RX: 3.7 mA (with ideal DCDC)
  - Integrated 10dBm PA
  - -20 dBm output power in “Near Field Mode”
  - -93 dBm receiver sensitivity
- Package:
  - LGA 35, 6.0mm x 8.6mm x 1.2mm

3.2 Application Examples

- Voice-controlled remote controls
- Beacons
- (Multi-sensor) Wearable devices
  - Fitness trackers
  - Consumer health
- Smartwatches
- Human interface devices
  - Keyboard
  - Mouse
- Toys
- Consumer appliances

3.3 Key Benefits

- Lowest power consumption
- Smallest system size
- Lowest system cost

3.4 Limitations

Inventek Systems products are not authorized for use in safety-critical applications (such as life support) where a failure of the Inventek Systems product would reasonably be expected to cause severe personal injury or death.

3.5 Regulatory Compliance

CE FCC

Regulator	Status
FCC	**O7P-14585**
IC	**10147A-14585**
RoHS	Compliant

Inventek Systems FCC, IC, and CE module transmitter certifications for the **ISM14585-L35-P8** module can be used to the advantage of any manufacturer developing a product using these devices. In order to take full advantage of the certifications and remain in compliance, Developers may not interfere, modify, replace and/or enhance the SiP antenna design, layout and power settings. For FCC compliance, End Customer finished products will still need to meet the Declaration of Conformity (SDoC) requirements according to 47 CFR Chapter 1, part 15, subpart B.

The testing required for the Declaration of Conformity (SDoC) requirements is specified in sections 15.107 and 15.109. The official documents can be obtained from the U.S Government Printing Office online. U.S. Government Printing Office CFR 47.

Any changes to the Inventek certified antenna options such as a different antenna or adding an antenna diversity switch will require filing for a Class 2 permissive change. Any Class 2 permissive changes must be performed under Inventek's grant, and therefore must be done in cooperation with Inventek. In addition to this document, Inventek recommends verifying the schematic board design with Inventek Engineering once the schematic is complete for further review and validation.

If it is desired to add a connector or U.FL connector in the RF path or change the antenna to one of the same type (chip) with equal or less gain, customers can do so without refiling.
Other changes such as a different antenna or adding an antenna diversity switch will require filing for a Class 2 permissive change. Any Class 2 permissive changes must be performed under Inventek's grant, and therefore must be done in cooperation with Inventek. In addition to this document, Inventek recommends verifying the schematic board design with Inventek Engineering once the schematic is complete for further review and validation.

Inventek also provides customers additional certifications for specific countries upon request and an agreed upon service fee.

3.5.1 FCC and IC Regulatory Information

- Model: ISM14585-L35-P8
- FCC ID: 07P-14585
- IC: 10147A-14585

This module is limited to OEM installation only.
OEM integrators must ensure that the end-user has no manual instructions to remove or
install the module. OEM's must comply with FCC marking regulation part 15 declaration of conformity (Section 2.925(e)).

This module is to be installed only in mobile or fixed applications (Please refer to FCC CFR
47 Part 2.1091(b) for a definition of mobile and fixed devices).

Separate approval is required for all other operating configurations, including portable configurations with respect to FCC CFR 47 Part 2.1093, and different antenna configurations.

The antennas used with this module must be installed to provide a separation distance of at
least 20cm from all persons, and must not be co-located or transmit simultaneously with any
other antenna or transmitter, except in accordance with FCC multi transmitter product procedures. The ISM14585-L35-P8 Module has been designed to operate with the following antennas and gains. Use with other antenna types or with these antenna types at higher gains is strictly prohibited.

Manufacturer	Type of Antenna	Model	Gain dBi	Type of Connector
Inventek	Integrated	N/A	-1	embedded
Inventek	Trace	B24P-W	3.2	w.fl

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions:

(1) This device may not cause harmful interference
(2) This device must accept any interference received, including interference that may cause undesired operation.

Warning: Changes or modifications not expressly approved by the party responsible is prohibited.

NOTE: This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications.

However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/ TV technician for help.

This equipment complies with FCC RF Exposure requirements and should be installed and operated with a minimum distance of 20cm between the radiator and any part of the human body.

A clearly visible label is required on the outside of the user's (OEM) enclosure with the following text:

- Contains FCC ID: O7P-14585
- Contains IC: 10147A-14585

This transmitter module is certified for FCC Part 15 operation; when installed in a host device, the host manufacturer is responsible for making sure that the host device with the transmitter installed continues to be compliant with Part 15B unintentional radiator requirements.

RSS-210/RSS-Gen Notices:
Operation is subject to the following two conditions:
(1) this device may not cause interference, and (2) this device must accept any interference, including interference that may cause undesired operation of this device.
L'opè ration est soumise aux deux conditions suivantes: (1) cet appareil ne peut pas provoquer d'interfè rences et (2) cet apparial doit accepter toute interfè rence, y compris les interfè rences qui peuvent causer un mauvis fonctionment de l'appareil.

Under Industry Canada regulations, this radio transmitter may only operate using an antenna of a type and maximum (or lesser) gain approved for the transmitter by Industry Canada. To reduce potential radio interference to other users, the antenna type and its gain should be so chosen that the equivalent isotropically radiated power (e.i.r.p.) is not more than that necessary for successful communication.

Sous la rè glementation d'Industrie Canada, ce transmetteur radio ne peut fonctionner en utilisant une antenne d'un type et un maximum (ou moins) gain approuvè es pour l'è metteur
ISM14585-L35 Specification par Industrie Canada. Pour rè duire le risqué d'interference aux autres utilisateures, le type d'antenne et son gain doivent être choisis de maniè re que la puissance isotrpe rayonnè e è quivalente (PIRE) ne dè passe pas ce qui est nè cessaire pour une communication rè ussie.

The radio transmitter has been approved by Industry Canada to operate with the antenna types listed above with the maximum permissible gain and required antenna impedance for each antenna type indicated. Antenna types not included in this list having a gain greater than the maximum gain indicated for that type, are strictly prohibited for use with this device.

Cet è metteur de radio a è tè approuvè par Industrie Canada pour fonctionner avec les types d'antennes è numè rè es ci-dessus avec le gain maximal admissible et impè dance d'antenna requise pour chaque type d'antenne indiquè . Types d'antennes ne figurant pas dans cette liste, ayant un gain supè rieur au gain maximum indiquè pour ce type, sontstrictement interdites pour l'utilisation avec cet appareil.

4 COMPLEMENTARY DOCUMENTATION

4.1 EVB

- The Inventek ISM14585-L35 Evaluation Board is the **ISM14585-L35-EVB**

- Please reference the **ISM14585-L35-EVB User's Manual**
  - Evaluation Board Specification
  - EVB User's Guide
  - Design Guidelines

5 ISM14585-L35 SoC & Module BLOCK DIAGRAMS

5.1 ISM14585-L35 SoC Block Diagram:

DIALOG DA14585 Radio w/Audio I/F SoC:

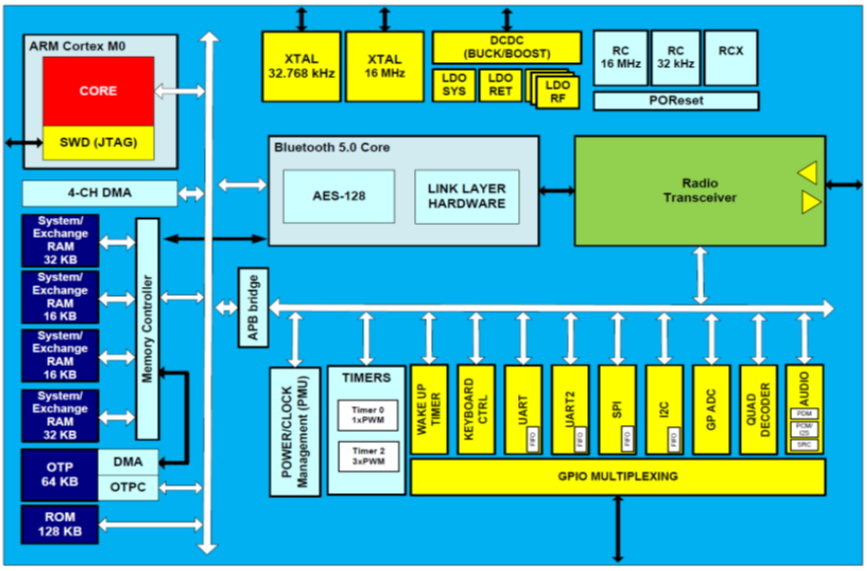
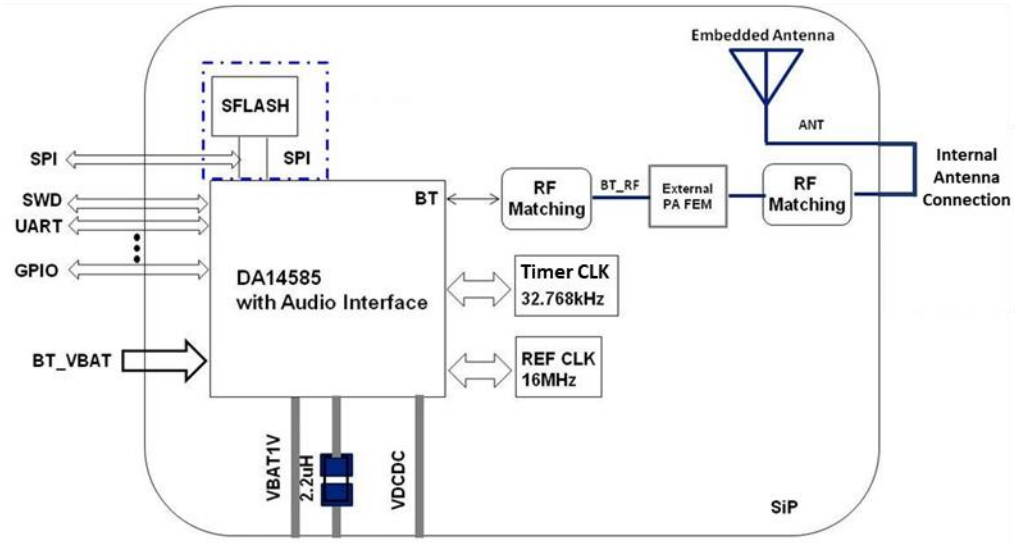


Figure 1 Dialog DA14585 SoC Block Diagram

Please Note: The Audio Unit feature requires a version of the ISM14585 module without the integrated 8Mb of Flash. The Microphone feature requires I2S or PDM via the SPI interface.

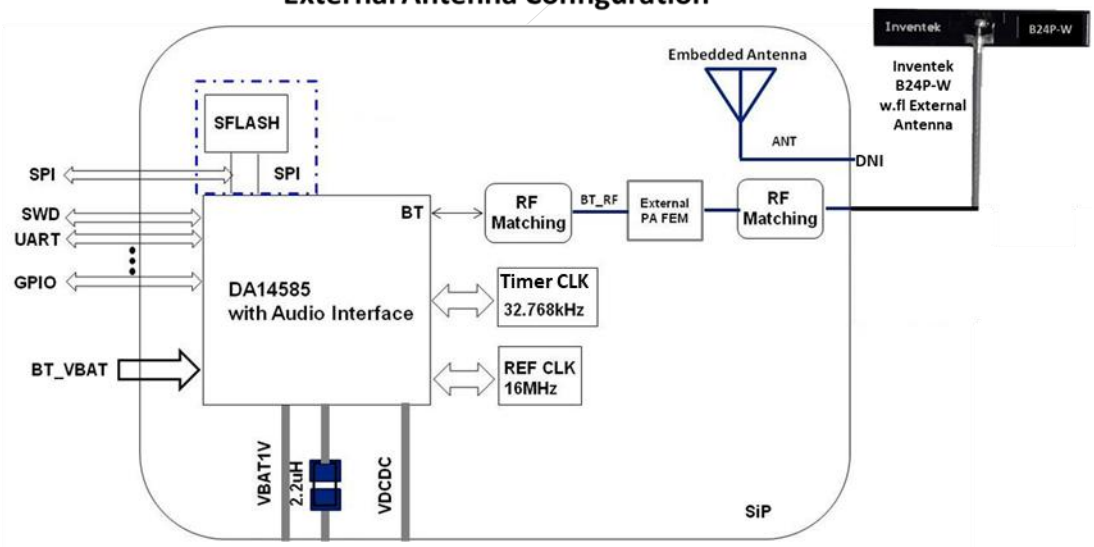
5.2 ISM14585-L35 Module Block Diagram Configuration Options:

Internal Antenna Configuration



Power Mode: The ISM14585 module is configured for Buck mode only and the "Switch" Pin requires the Synchronous DC-DC converter to be configured for 3.3V or higher.

External Antenna Configuration



Power Mode: The ISM14585 module is configured for Buck mode only and the "Switch" Pin requires the Synchronous DC-DC converter to be configured for 3.3V or higher.

- UART: Universal synchronous/asynchronous receiver transmitters
- SPI: Serial Peripheral Interface
- GPIO: General-purpose input/output
- SWD: Serial Wire Debug

6 Electrical Specification

6.1 Absolute Maximum Rating

Supply Power	Max +4 Volt		
Storage Temperature	- 40° to 125° Celsius		
Voltage ripple	+/- 2%	Max. Values not exceeding Operating voltage	
	Power	min	Max
Power Supply Absolute Maximum Ratings	BT_VBAT	-	3.6
	VDD_PA	-	3.6
	VDD_FLASH	-	3.6
Voltage on input or output pin		VSS-0.3	VBAT+0.3

6.2 Recommendable Operation Condition

6.2.1 Temperature, Humidity

The **ISM14585-L35** will withstand the operational requirements listed in the table below.

Operating Temperature	-30° to 85° Celsius	
Humidity range	Max 95%	Non condensing, relative humidity

6.2.2 Voltage

The Power supply for the **ISM14585-L35** will be provided by the host via the power pins

Symbol	Parameter	Min	Typ	Max	Unit
BT_VBAT		**3**	**3.3**	**3.6**	**V**
VDD_PA		**3**	**3.3**	**3.6**	**V**
VDD_Flash		**3**	**3.3**	**3.6**	**V**

6.3 Current Consumption

6.3.1 BLUETOOTH LOW ENERGY

Condition: Condition: 25deg.C

Item	Condition	Min	Nom	Max	Unit
Tx Mode	**Transmitter and baseband are both operating, 100%**		**TBD**		**mA**
RX Mode	**Receiver and baseband are both operating, 100%**		**TBD**		**mA**

7 RF Specification

7.1 RF Transmitter Specification

7.1.1 BLE RF SPECIFICATION

Parameter	Mode and Condition	Min.	Typ.	Max.	Unit
Frequency Range		2402		2480	MHz
RX sense[a]	LE GFSK, 0.1% BER, 1 Mbps		TBD	-80	dBm
TX Power	N/A		TBD	TBD	dBm
Mod char: delta f1 average		225	225	275	kHz
Mod char: delta f2 max[c]		99.9			%
Mod char: ratio		0.8	0.95		%

- Dirty TX is Off.
- Up to 1dB of variation may potentially be seen from typical sensitivity specs due to the chip, board, and associated variations.
- At least 99.9% of all delta F2 max frequency values recorded over 10 packets must be greater than 185 kHz.
- Additional specifications to reference is SIG

8 Pin Definition

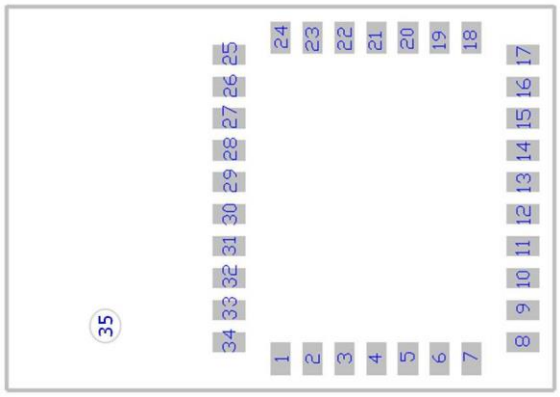
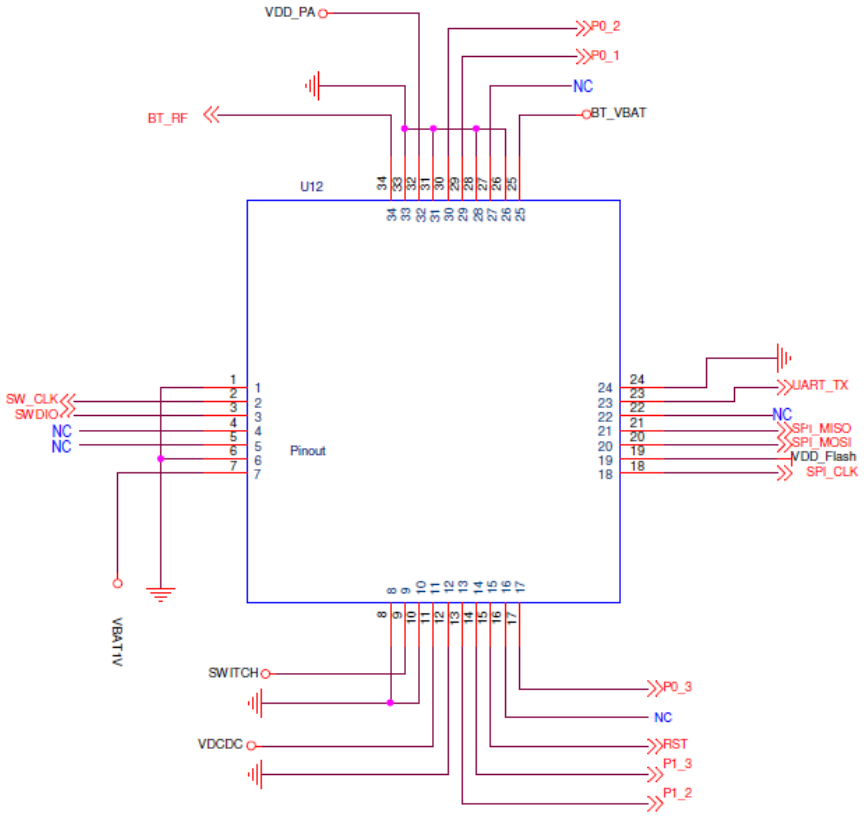


Figure 3. TOP VIEW

8.1 Module Pin Out Schematic (Internal Antenna configuration)



8.2 Detail Pin definition information

Please reference the Dialog **DA14585** Bluetooth 5.0 SoC Data Sheet for any additional information.

Pin Name	Pin Number	I/O Type	Description
Radio			
BT_RF	**34**	**I/O**	RF I/O antenna port
Antenna			
ANT	35	I/O	Embedded Antenna Port. Enable: tie to BT_RF pin Disable: tie to GND when not use.
Voltage Regulators			
BT_VBAT	25	AI	VBAT input Pin, Bypass with a 1uF
VDD_PA	32	AI	PA VDD Input Pin, Bypass with a 1uF
VDD_FLASH	19	AI	Flash VDD input pin
VDCDC	11	AO	Bypass with 2.2uF and 100nF. Reference Section 9.2
SWITCH	9	AIO	Reference Section 9.2
VBAT1V	7	AI	RESERVED, connect to Ground
Straps			
RST	15	DI	INPUT. Reset signal (active high). Must be connected to GND if not used.
Grounds			
GND	1,6,8,10,12,24, 26, 28,31,33	GND	Module Grounds

Pin Name	Pin Number	I/O Type	Description
Digital I/O			
P0_0 (Note2)	18	DIO / ADC 0 / SPI CLK	INPUT/OUTPUT with selectable pull up/down resistor. Pulldown enabled during and after reset. General purpose I/O port bit or alternate function nodes. Contains state retention mechanism during power down. Drive current is 3.5mA Note 1: An increased Packet Error rate might occur if GPIO P1-2 and P1-3 is toggling. The root cause of this problem is the close location of a few GPIO pins to the on-chip 16 MHz XTAL oscillator circuitry. Toggling of these GPIOs might corrupt the clock, which will be visible in the Packet Error Rate. Please reference the Dialog DA14585 Errata for more details. Note 2: All Port Pins can be PWM Note 3: There is no execute in place from the SPI Flash, therefore UART is available after Boot or the UART can be interleaved with SPI. If configured, the SPI Flash can be used to load FW.
P0_1 (Note2)	29	DIO / ADC 1	
P0_2 (Note2)	30	DIO / ADC 2 / I2C SCL	
P0_3 (Note2)	17	DIO / ADC 3 / I2C SDA	
P0_4 (Note2)	23	DIO / UART TX	
P0_5 (Note2 & 3)	21	DIO / UART RX / SPI MISO	
P0_6 (Note2)	20	DIO / SPI MOSI	
P1_2 (Note 1 & 2)	13	DIO	
P1_3 (Note 1 & 2)	14	DIO	
Debug Interface			
SWCLK	2	DIO	This signal is the JTAG clock by default
SWDIO	3	DIO	This signal is the JTAG data I/O by default
No Connects			
NC	4,5,16, 22,27	NA	RESERVED
Internal Flash			
P0_0	18	SPI CLK	Shared SPI clock, internally connected to SPI Flash
P0_5	21	UART RX / SPI MISO	Shared SPI MISO internally connected to SPI Flash. See (Note 3)
P0_6	20	SPI MOSI	Shared SPI MOSI internally connected to SPI Flash
P0_7	N/A	SPI CS	Internally connected to the CS of the SPI Flash (Note: In Dialog SmartSnippet Studio, set Flash Enable to this Pin

9 Addition Information

9.1 Microcontroller Unit

The **ISM14585-L35** includes a Cortex-M0 32-bit Reduced Instruction Set Computing (RISC) processor with a von Neumann architecture (single bus interface). It uses an instruction set called Thumb, which was first supported in the ARM7TDMI processor.

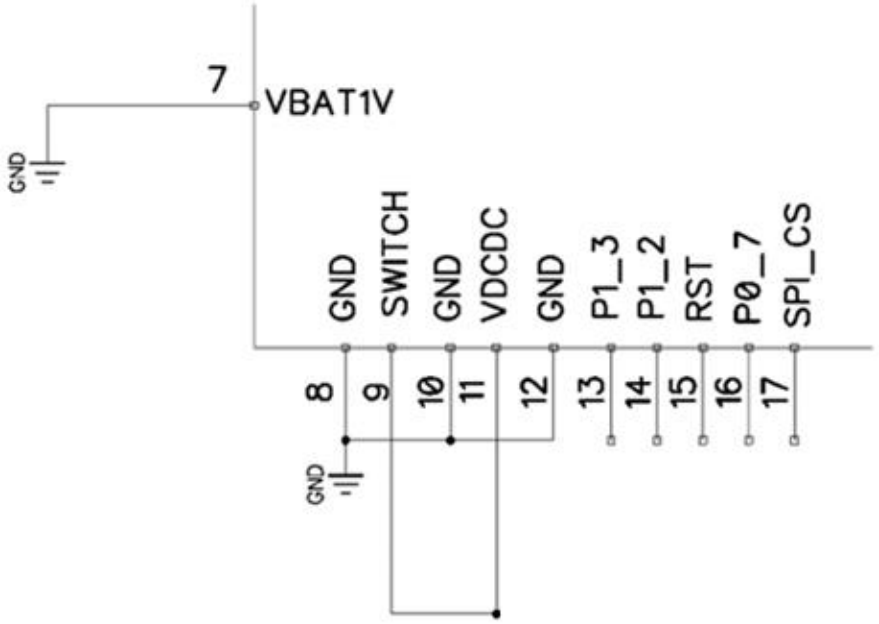
However, several newer instructions from the ARMv6 architecture and a few instructions from the Thumb-2 technology are also included. Thumb-2 technology extends the previous Thumb instruction set to allow all operations to be carried out in one CPU state. The instruction set in Thumb-2 includes both 16-bit and 32-bit instructions. Most instructions generated by the C compiler use the 16-bit instructions, and the 32-bit instructions are used when the 16-bit version cannot carry out the required operations. This results in high code density and avoids the overhead of switching between two instruction sets.

In total, the Cortex-M0 processor supports 56 base instructions, although some instructions can have more than one form. Although the instruction set is small, the Cortex-M0 processor is highly capable because the Thumb instruction set is highly optimized.
Academically, the Cortex-M0 processor is classified as load-store architecture, as it has separate instructions for reading and writing to memory, and instructions for arithmetic or logical operations that use registers.

9.2 Buck Power Configuration

The **ISM14585** module is configured for Buck mode only and the “Switch” Pin requires the Synchronous DC-DC converter to be configured for 3.3V or higher.

The **ISM14585** module is configured for Buck mode only and the “Switch” Pin requires the Synchronous DC-DC converter to be configured for 3.3V or higher.



9.3 ANTENNA CONFIGURATION OPTIONS

9.3.1 Integrated Antenna

- Pin 34 is connected to Pin 35.
- Please reference the **ISM14585-L35-P8-EVB** Evaluation Board User’s Manual for Layout specification.

9.3.2 External w.fl Antenna

- Pin 34 is connected to a w.fl external antenna connector.
- Please reference the **ISM14585-L35-P8-EVB** Evaluation Board User's Manual for Layout specification.

9.4 External Reset

The **ISM14585-L35** comprises an RST pad which is active high. It contains an RC filter for spikes suppression with 400kΩ and 2.8pF for the resistor and the capacitor respectively. It also contains a 25kΩ pull-down resistor. This pad should be connected to ground if not needed by the application.

The response is illustrated in the **Figure 4** which displays the voltage (V) on the vertical axis and the time (µs) on the horizontal axis:.

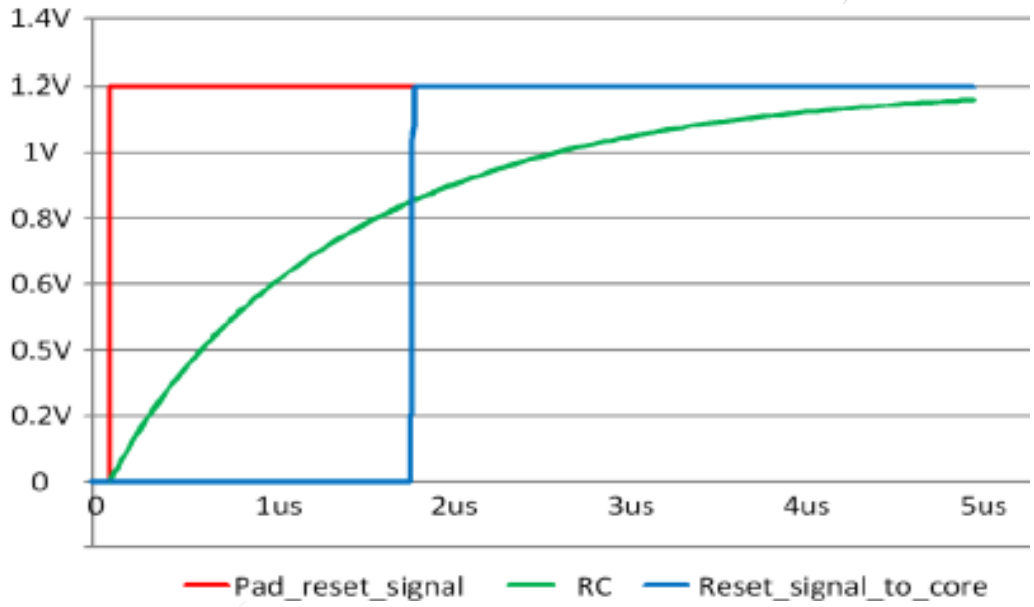


Figure 4. RST Pad Latency

The typical latency of the RST pad is in the range of 2us.

9.4.1 POR, HW AND SW RESET

The Power-On Reset (POR) signal is generated:

● Internally and will release the system's flip flops as soon as the VDD voltage crosses the minimum threshold value.

● Externally by a Power-On Reset source (RST pad or GPIO).

There are three main reset signals in the **ISM14585-L35:**

1. The PWR-On reset which is triggered by a GPIO set as POR source with selectable polarity and/or the RST pad after a programmable time delay.
2. The HW reset which is basically triggered by the RST pad when it becomes active for a short period of time (less than the programmable delay for POR).
3. The SW reset which is triggered by writing the SYS_CTRL_REG[SW_RESET] bit.

- The HW reset can also be automatically activated upon waking up of the system from the Extended or Deep Sleep mode by programming bit PMU_CTRL_REG [RESET_ON_WAKEUP].
- The PWR-On reset as well as the HW reset will basically run the cold start-up sequence and the BootROM code will be executed.

The SW reset is the logical OR of a signal from the ARM CPU (triggered by writing SCB->AIRCR = 0x05FA0004) and the SYS_CTRL_REG[SW_RESET] bit.

This is mainly used to reboot the system after the base address has been remapped. The block diagram of the reset block is depicted in **Figure 5**.

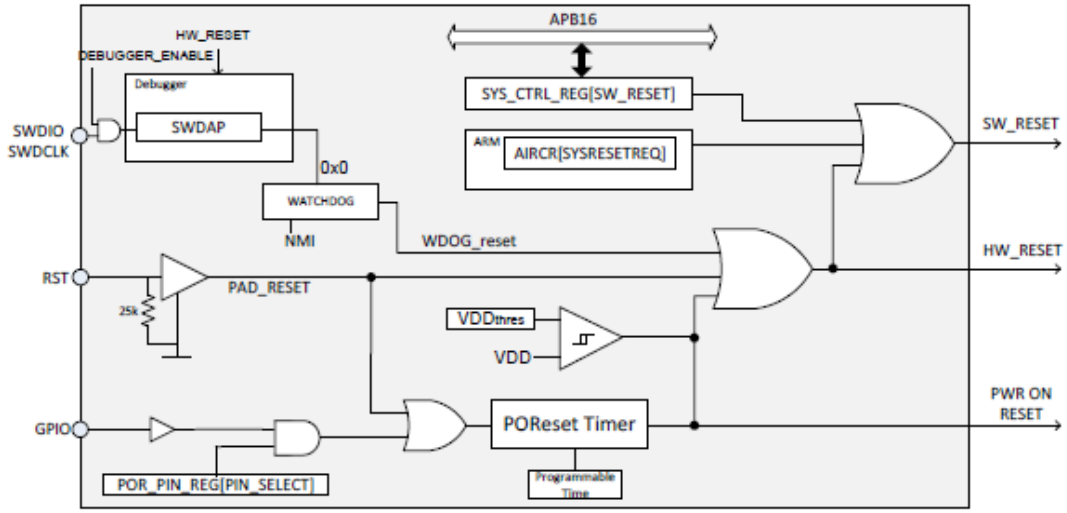


Figure 5. Reset Block Diagram

Certain registers are reset by POR only or by POR and the HW reset signal, but not by the SW reset. These registers are listed in the table below.

Reset by POR Only	Reset by POR or HW Reset	Reset by POR, HW or SW Reset
BANDGAP_REG	OTPC_NWORDS_REG	The rest of the Register File
POR_PIN_REG	CLK_FREQ_TRIM_REG	
POR_TIMER_REG	CLK_RADIO_REG	
RF registers containing the trimming values for the VCO, the LNA and the I/O capacitance	All RF calibration registers	
	BLE_CNTL2_REG	
	CLK_CTRL_REG	
	PMU_CTRL_REG	
	SYS_CTRL_REG	
	CLK_32K_REG_I	
	CLK_16M_REG_I	
	CLK_RCX32K_REG	
	TRIM_CTRL_REG	
	DEBUG_REG[DEBUGS_FREEZE_EN]	
	GP_CONTROL_REG[EM_MAP]	

9.4.2 POWER-ON RESET FUNCTIONALITY

Power-On Reset functionality is available by two sources:

- Reset Pad: Reset pad is always capable of producing a Power-On Reset.
- GPIO Pin: A GPIO can be selected by the user application to act as POR source.

The time needed for the POReset pin to be active is stored in the POR_TIMER_REG. The register field POR_TIME is a 7-bits field which holds time factor that the total time for POR is calculated. The maximum value of the field is 0x7F. The total time for POR is calculated by the following formula:

Total time = POR_TIME x 4096 x RC32k clock period

where RC32k clock period = 31.25μs at 25oC .

The maximum time that a POR can be performed is ~16.2 seconds at 25oC.

The RC32k clock is temperature dependent so based on the temperature span of -10oC to 50oC, clock frequency range is calculated to be 25kHz to 39kHz. Then,

TPORcold = 13s

TPORhot = 20.8s

9.4.2.1 POR TIMER CLOCK

The Power-On Reset timer is clocked by the RC32k clock. If the application disables the RC32k, then the hardware takes care of enabling the RC32k clock when the POR source (Reset pad or GPIO) is asserted. It should be noted that if POR is generated from the Reset pad the RC32 will operate with the reset trimming value. If a GPIO is used as POR source, the RC32 clock will be trimmed. The deviation between both cases in terms of timing is expected to be minor.

9.4.2.2 RESET PAD

The Reset pad will produce a HW Reset if the pin active time is less than the programmed value in the POR_TIMER_REG register and a Power-On Reset if it is greater or equal the value. Reset pad is always Active High.

9.4.2.3 POR FROM GPIO

When a GPIO is used as a Power-on Reset source, the selected pin retains its capability to act as GPIO. The POR_PIN_REG[PIN_SELECT] field holds the required GPIO pin number. If the value of the PIN_SELECT field equals to 0 the POR over GPIO functionality is disabled. The polarity of the pin can be configured by the POR_PIN_REG [POR_POLARITY] bit where 0 means Active Low and 1 Active High.

9.4.2.4 POWER-ON RESET TIMING DIAGRAM

The operation of the Power-On Reset for both Reset pad and GPIO is depicted in Figure 6.

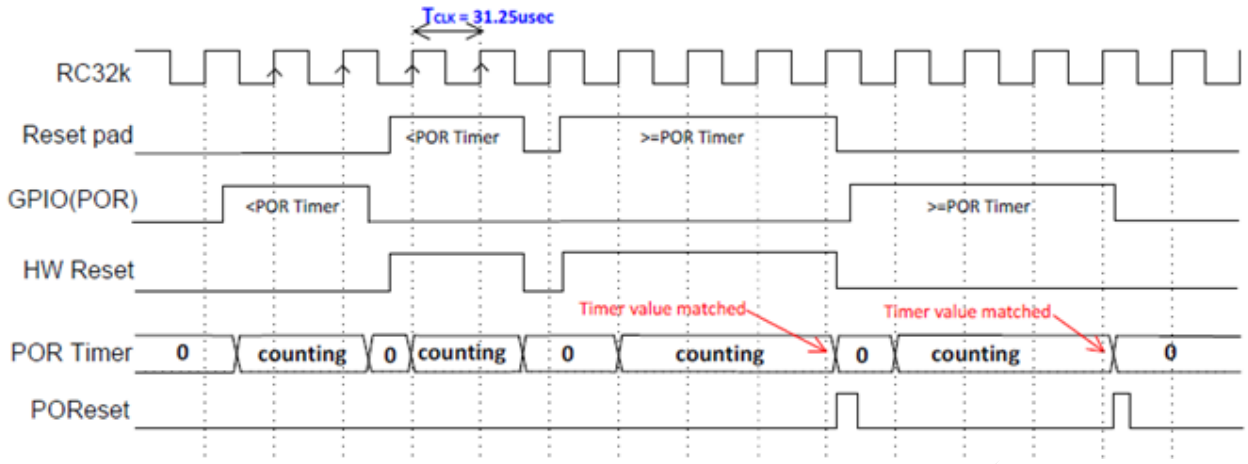


Figure 6 Power-On Reset Timing Diagram

9.4.2.5 POWER-ON RESET CONSIDERATIONS

If any of the POR sources is asserted then the POR timer starts to count. When a POR source is released before the timer has expired, POR timer will reset to 0. If a second source is asserted while the first is already asserted and the first is released after that point, POR will occur; assuming that the total time of both sources kept asserted is larger or equal than the POR_TIME.

The POR_PIN_REG[PIN_SELECT] field cannot survive any Reset (POR, HW, SW) and as such the user must take special care on setting up the GPIO POR source right after a reset. This also applies for the POR_TIMER_REG[POR_TIME] field after a Power-On Reset.

The user must also take into account that if a GPIO is used as POR source, the dynamic current of the system increases due to the dynamic current consumed by the RC32k oscillator. This increase is calculated to be 100nA to 120nA and it is also present during sleep time period. POR from Reset pin does not add this dynamic current consumption.

9.5 DMA Controller

The DMA controller has 4 Direct Memory Access (DMA) channels for fast data transfers from/to SPI, UART, I2C, PDM and PCM to/from any on-chip RAM. The DMA controller off-loads the ARM interrupt rate if an interrupts is given after a number of transfers. More peripherals DMA requests are multiplexed on the 4 available channels, to increase utilization of the DMA. The block diagram of the DMA controller is depicted in Figure 7.

Features:

- 4 channels with optional peripheral trigger
- Full 32 bit source and destination pointers.
- Flexible interrupt generation.
- Programmable length.
- Flexible peripheral request per channel.
- Option to initialize memory.
- Programmable Edge-Sensitive request support(suggested for UART and I2C service)

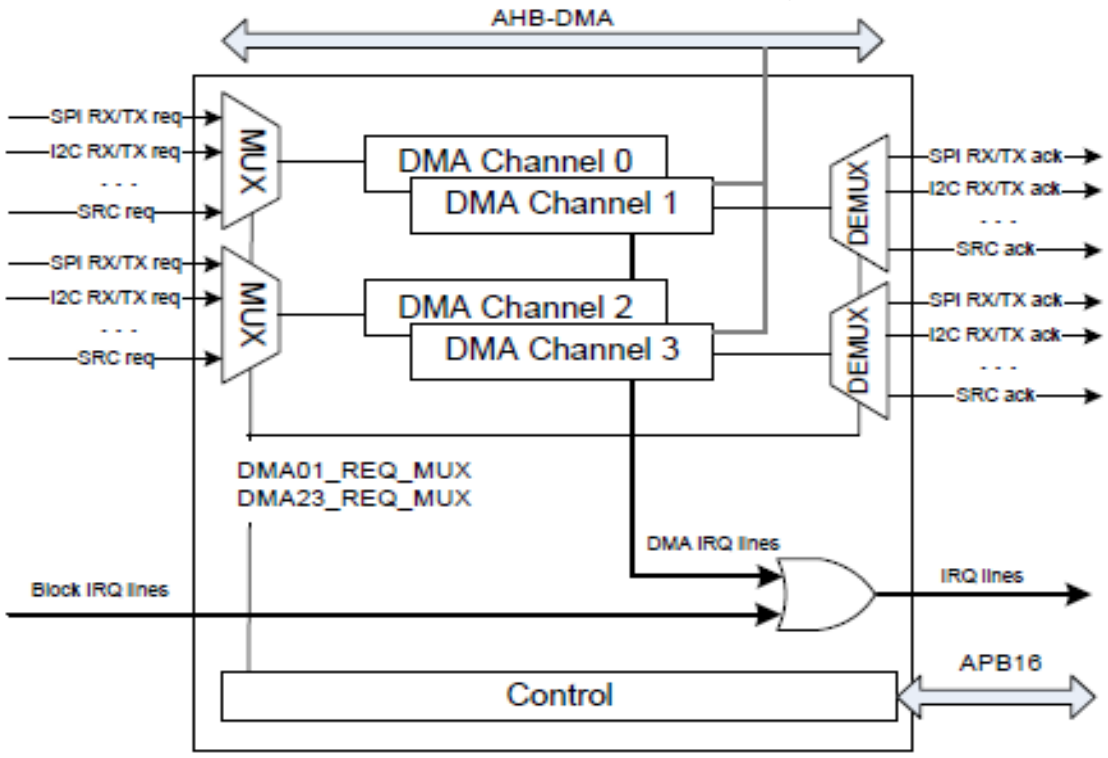


Figure 7 DMA Controller Block Diagram

9.5.1 DMA PERIPHERALS

The list of peripherals that can request for a DMA service is presented in below table

Name	Direction
SPI	RX
SPI	TX
UART	RX
UART	TX
UART2	RX
UART2	TX
I2C	RX
I2C	TX
PCM	RX
PCM	TX
SRC	RX
SRC	TX

9.5.2 INPUT/OUTPUT MULTIPLEXER

The multiplexing of peripheral requests is controlled by DMA_REQ_MUX_REG. Thus, if DMA_REQ_MUX_REG [DMAxy_SEL] is set to a certain (non-reserved) value, the TX/RX request from the corresponding peripheral will be routed to DMA channels y (TX request) and x (RX request) respectively.

Similarly, an acknowledging de-multiplexing mechanism is applied.

However, when two or more bit-fields (peripheral selectors) of DMA_REQ_MUX_REG have the same value, the lesser significant selector will be given priority (see also the register's description).

9.5.3 DMA CHANNEL OPERATION

A DMA channel is switched on with bit DMA_ON. This bit is automatically reset if the dma transfer is finished. The DMA channels can either be triggered by software or by a peripheral DMA request. If DREQ_MODE is 0, then a DMA channel is immediately triggered. If DREQ_MODE is 1 the DMA channel can be triggered by a Hardware interrupt.

If DMA starts, data is transferred from address DMAx_A_START_REG to address DMAx_B_START_REG for a length of DMAx_LEN_REG, which can be 8, 16 or 32 bits

wide. The address increment is realized with an internal 13 bits counter DMAx_IDX_REG, which is set to 0 if the DMA transfer starts and is compared with the DMA_LEN_REG after each transfer. The register value is multiplied according to the AINC and BINC and BW values before it is added to DMA_B_START_REG and DMA_B_START_REG. AINC or BINC must be 0 for register access.

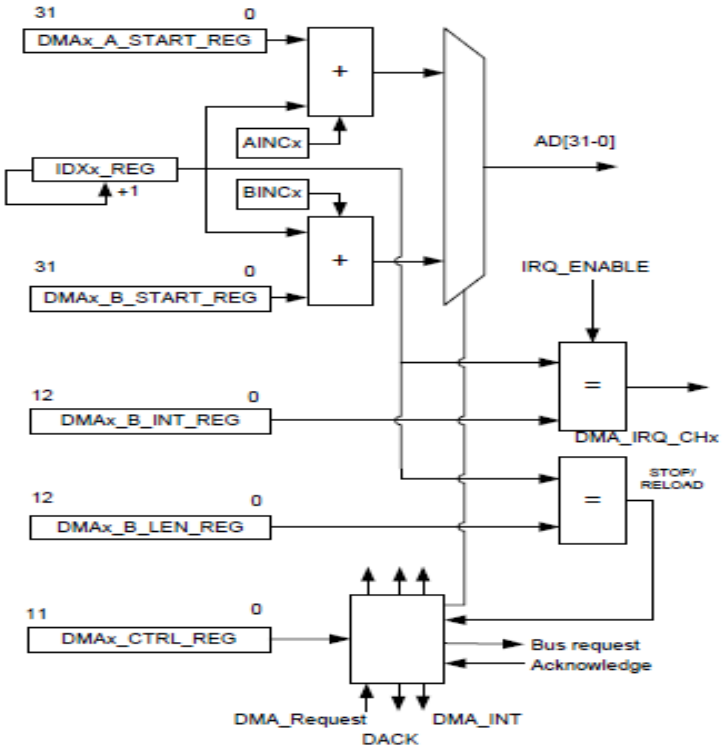


Figure 8 DMA Channel Diagram

If at the end of a DMA cycle, the DMA start condition is still true, the DMA continues. The DMA stops if DREQ_MODE is low or if DMAx_LEN_REG is equal to the internal index register. This condition also clears the DMA_ON bit.

If bit CIRCULAR is set to 1, the DMA controller automatically resets the internal index registers and continues from its starting address without intervention of the ARM CortexTM M0. If the DMA controller is started with DREQ_MODE =0, the DMA will always stop, regardless of the state of CIRCULAR.

Each DMA channel can generate an interrupt if DMAx_INT_REG if equal to DMAx_IDX_REG. After the transfer and before DMAx_IDX_REG is incremented, the interrupt is generated. Example: if DMA_x_INT_REG=0 and DMA_x_LEN_REG=0, there will be one transfer and an interrupt.

9.5.4 DMA ARBITRATION

The priority level of a DMA channel can be set with bits DMA_PRIO[2-0]. These bits determine which DMA channel will be activated in case more than one DMA channel requests DMA. If two or more channels have the same priority, an inherent priority applies, (see register description).

With DREQ_MODE = 0, a DMA can be interrupted by a channel with a higher priority if the DMA_IDLE bit is set. When DMA_INIT is set, however, the DMA channel currently performing the transfer locks the bus and cannot be interrupted by any other channel, until the transfer is completed, regardless if DMA_IDLE is set. The purpose of DMA_INIT is to initialize a specific memory block with a certain value, fetched also from memory, without any interruption from other active DMA channels that may request the bus at the same time. Consequently, it should be used only for memory initialization, while when the DMA transfers data to/ from peripherals, it should be set to '0'. Note that AINC must be set to '0' and BINC to '1', when DMA_INIT is enabled.

It should be noted that memory initialization could also be performed without having the DMA_INIT enabled and by simply setting AINC to '0' and BINC to '1', provided that the source address memory value will not change during the transfer. However, it is not guaranteed that the DMA transfer will not be interrupted by other channels of higher priority, when these request access to the bus at the same time.

9.5.5 FREEZING DMA CHANNELS

Each channel of the DMA controller can be temporarily disabled by writing a 1 to freeze all channels at SET_FREEZE_REG.

To enable the channels again, a 1 to bits at the RESET_FREEZE_REG must be written.

There is no hardware protection from erroneous programming of the DMA registers.

It is noted that the on-going Memory-to-Memory transfers (DREQ_MODE='0') cannot be interrupted. Thus, in that case, the corresponding DMA channels will be frozen after any on-going Memory-to memory transfer is completed.

9.6 I2C Interface

The I2C Interface is a programmable control bus that provides support for the communications link between Integrated Circuits in a system. It is a simple two-wire bus with a software-defined protocol for system control, which is used in temperature sensors and voltage level translators to EEPROMs, general-purpose I/O, A/D and D/A converters.

Features

- Two-wire I2C serial interface consists of a serial data line (SDA) and a serial clock (SCL)
- Two speeds are supported:
  - Standard mode (0 to 100kbit/s)
  - Fast mode (<= 400kbit/s)
- Clock synchronization
- 32B deep transmit/receive FIFOs
- Master transmit, Master receive operation
- 7-bit or 10-bit addressing
- 7-bit or 10-bit combined format transfers
- Bulk transmit mode
- Default slave address of 0x055
- Interrupt or polled-mode operation
- Handles Bit and Byte waiting at both bus speeds
- Programmable SDA hold time
- DMA support

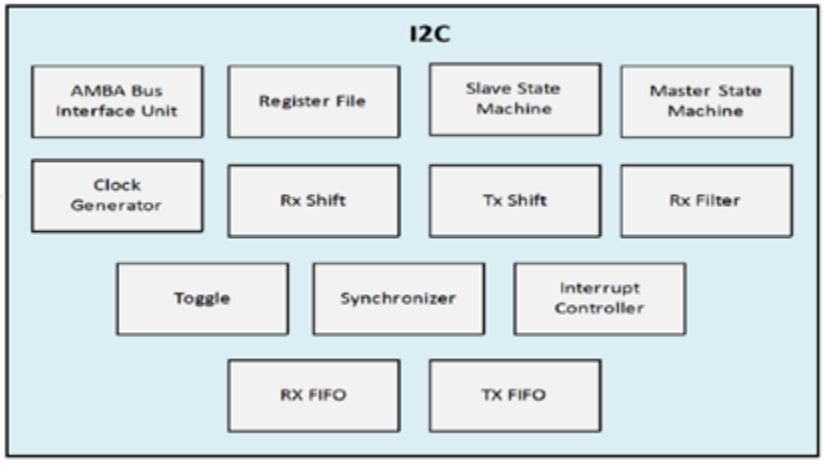


Figure 9 I2C Controller Block Diagram

The I2C Controller block diagram shown in Figure 9 contains the following sub-blocks:

- AMBA Bus Interface Unit: Interfacing via the APB interface to access the register file.
- Register File: Contains configuration registers and is the interface with software.
- Master State Machine: Generates the I2C protocol for the master transfers.
- Clock Generator: Calculates the required timing to do the following:
  - Generate the SCL clock when configured as a master
  - Check for bus idle
  - Generate a START and a STOP
  - Setup the data and hold the data
- Rx Shift: Takes data into the design and extracts it in byte format.
- Tx Shift: Presents data supplied by CPU for transfer on the I2C bus.
- Rx Filter: Detects the events in the bus; for example, start, stop and arbitration lost.
- Toggle: Generates pulses on both sides and toggles to transfer signals across clock domains.
- Synchronizer: Transfers signals from one clock domain to another.
- Interrupt Controller: Generates the raw interrupt and interrupt flags, allowing them to be set and cleared.
- RX FIFO/TX: Holds the RX FIFO and TX FIFO register banks and controllers, along with their status levels.

9.7 I2C BUS TERMS

- The following terms relate to how the role of the I2C device and how it interacts with other I2C devices on the bus.

- Transmitter. The device that sends data to the bus. A transmitter can either be a device that initiates the data transmission to the bus (a master-transmitter) or responds to a request from the master to send data to the bus (a slave-transmitter).

- Receiver. The device that receives data from the bus. A receiver can either be a device that receives data on its own request (a master-receiver) or in response to a request from the master (a slave-receiver).

- Master. The component that initializes a transfer (START command), generates the clock (SCL) signal and terminates the transfer (STOP command). A master can be either a transmitter or a receiver.

- Slave. The device addressed by the master. A slave can be either receiver or transmitter. These concepts are illustrated in Figure 10.

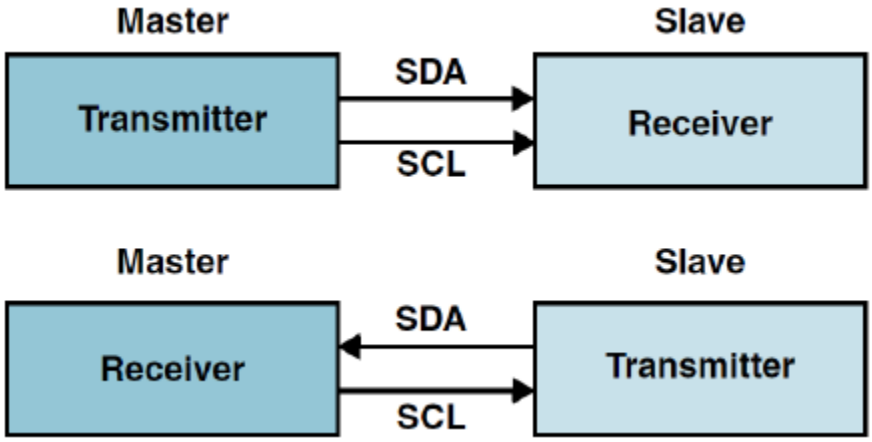


Figure 10 Master/Slave and Transmitter/Receiver Relationships

- Multi-master. The ability for more than one master to co-exist on the bus at the same time without collision or data loss.

- Arbitration. The predefined procedure that authorizes only one master at a time to take control of the bus. For more information about this behavior, refer to Multiple Master Arbitration section.

- Synchronization. The predefined procedure that synchronizes the clock signals provided by two or more masters. For more information about this feature, refer to Clock Synchronization.

- SDA. Data signal line (Serial Data)

- SCL. Clock signal line (Serial Clock)

9.7.1.1BUS TRANSFER TERMS

The following terms are specific to data transfers that occur to/from the I2C bus:

- START (RESTART). Data transfer begins with a START or RESTART condition. The level of the SDA data line changes from high to low, while the SCL clock line remains high. When this occurs, the bus becomes busy.

- STOP. Data transfer is terminated by a STOP condition. This occurs when the level on the SDA data line passes from the low state to the high state, while the SCL clock line remains high. When the data transfer has been terminated, the bus is free or idle once again. The bus stays busy if a RESTART is generated instead of a STOP condition.

Note: START and RESTART conditions are functionally identical.

9.7.2 I2C BEHAVIOR

The I2C can be only be controlled via software to be an I2C master only, communicating with other I2C slaves;

The master is responsible for generating the clock and controlling the transfer of data. The slave is responsible for either transmitting or receiving data to/from the master. The acknowledgement of data is sent by the device that is receiving data, which can be either a master or a slave. As mentioned previously, the I2C protocol also allows multiple masters to reside on the I2C bus and uses an arbitration procedure to determine bus ownership.

Each slave has a unique address that is determined by the system designer. When a master wants to communicate with a slave, the master transmits a START/RESTART condition that is then followed by the slave's address and a control bit (R/W) to determine if the master wants to transmit data or receive data from the slave. The slave then sends an acknowledge pulse (ACK) after the address.

If the master (master-transmitter) is writing to the slave (slave-receiver), the receiver gets one byte of data. This transaction continues until the master terminates the

transmission with a STOP condition. If the master is reading from a slave (master-receiver), the slave transmits (slave-transmitter) a byte of data to the master, and the

master then acknowledges the transaction with the ACK pulse. This transaction continues until the master terminates the transmission by not acknowledging (NACK) the transaction after the last byte is received, and then the master issues a STOP condition or addresses another slave after issuing a RESTART condition. This behavior is illustrated in Figure 11.

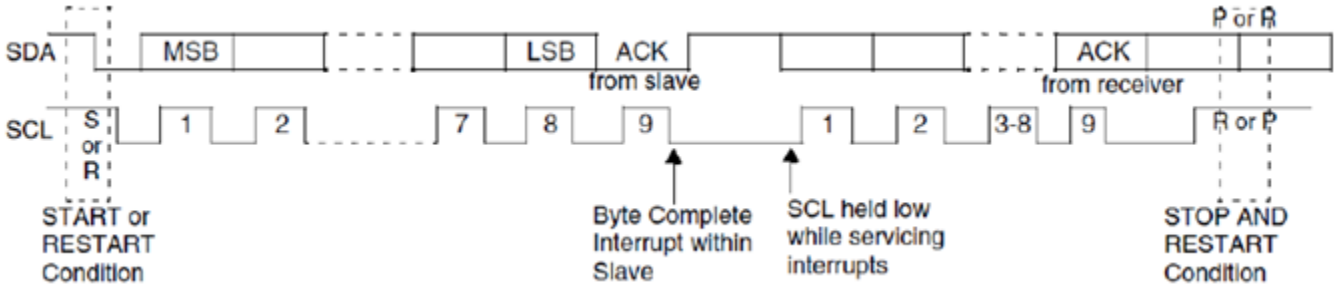


Figure 11 Data Transfer on the I2C Bus

The I2C is a synchronous serial interface. The SDA line is a bidirectional signal and changes only while the SCL line is low, except for STOP, START, and RESTART conditions. The output drivers are open-drain or open-collector to perform wire-AND functions on the bus. The maximum number of devices on the bus is limited by only the maximum capacitance specification of 400pF. Data is transmitted in byte packages.

9.7.2.1 START AND STOP GENERATION

When operating as an I2C master, putting data into the transmit FIFO causes the I2C Controller to generate a START condition on the I2C bus. Allowing the transmit FIFO to empty causes the I2C Controller to generate a STOP condition on the I2C bus.

When operating as a slave, the I2C Controller does not generate START and STOP conditions, as per the protocol. However, if a read request is made to the I2C Controller, it holds the SCL line low until read data has been supplied to it. This stalls the I2C bus until read data is provided to the slave I2C Controller, or the I2C Controller slave is disabled by writing a 0 to I2C_ENABLE.

9.7.2.2 COMBINED FORMATS

The I2C Controller supports mixed read and write combined format transactions in both 7-bit and 10-bit addressing modes.

The I2C Controller does not support mixed address and mixed address format - that is, a 7-bit address transaction followed by a 10-bit address transaction or vice versa - combined format transactions.

To initiate combined format transfers, I2C_CON.I2C_RESTART_EN should be set to 1. With this value set and operating as a master, when the I2C Controller completes an I2C transfer, it checks the transmit FIFO and executes the next transfer. If the direction of this transfer differs from the previous transfer, the combined format is used to issue the transfer. If the transmit FIFO is empty when the current I2C transfer completes, a STOP is issued, and the next transfer is issued following a START condition.

9.7.3 I2C PROTOCOLS

The I2C Controller has the following protocols:

- START and STOP Conditions
- Addressing Slave Protocol
- Transmitting and Receiving Protocol
- START BYTE Transfer Protocol

9.7.3.1 START AND STOP CONDITIONS

When the bus is idle, both the SCL and SDA signals are pulled high through external pull-up resistors on the bus. When the master wants to start a transmission on the bus, the master issues a START condition. This is defined to be a high-to-low transition of the SDA signal while SCL is 1. When the master wants to terminate the transmission, the master issues a STOP condition. This is defined to be a low-to-high transition of the SDA line while SCL is 1. Figure 12 shows the timing of the START and STOP conditions. When data is being transmitted on the bus, the SDA line must be stable when SCL is 1.

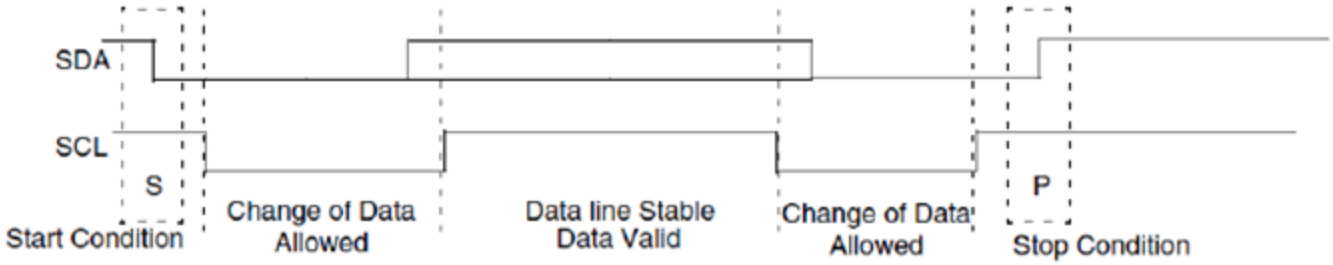


Figure 12 START and STOP Conditions

Note: The signal transitions for the START/STOP conditions, as depicted in Figure 12, reflect those observed at the output signals of the Master driving the I2C bus. Care should be taken when observing the SDA/SCL signals at the input signals of the Slave(s), because unequal line delays may result in an incorrect SDA/SCL timing relationship.

9.7.3.2 ADDRESSING SLAVE PROTOCOL

There are two address formats: 7-bit address format and 10-bit address format.

7-bit Address Format

During the 7-bit address format, the first seven bits (bits 7:1) of the first byte set the slave address and the LSB bit (bit 0) is the R/W bit as shown in Figure 13. When bit 0 (R/W) is set to 0, the master writes to the slave. When bit 0 (R/W) is set to 1, the master reads from the slave.

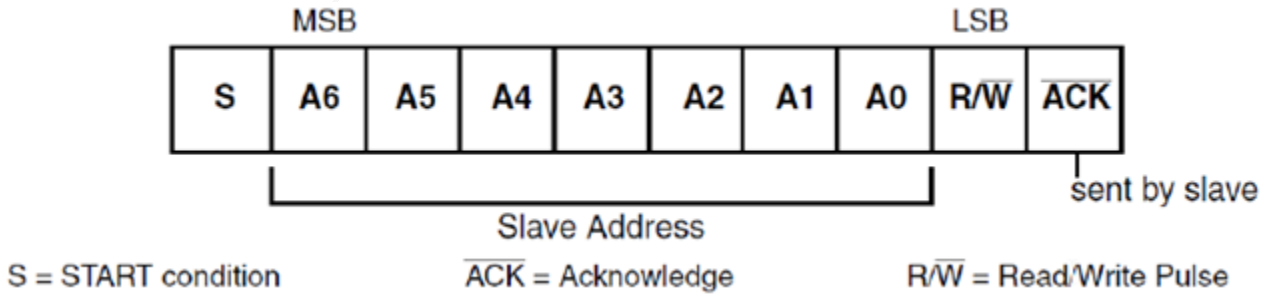


Figure 13 7-bit Address Format

10-bit Address Format

During 10-bit addressing, two bytes are transferred to set the 10-bit address. The transfer of the first byte contains the following bit definition. The first five bits (bits 7:3) notify the slaves that this is a 10-bit transfer followed by the next two bits (bits 2:1), which set the slaves address bits 9:8, and the LSB bit (bit 0) is the R/W bit. The second byte transferred sets bits 7:0 of the slave address. Figure 14 shows the 10-bit address format, and below table defines the special purpose and reserved first byte addresses.

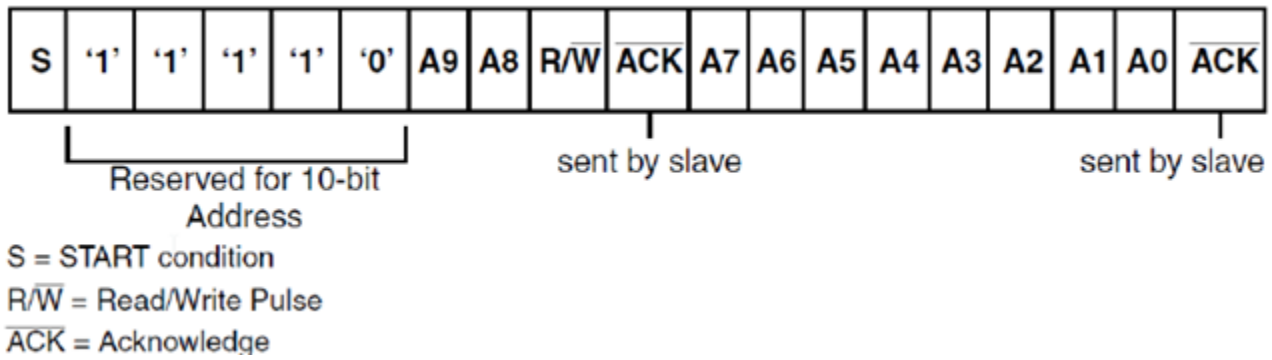


Figure 14 10-bit Address Format

Slave Address	R/W Bits	Description
0000 000	0	General Call Address. I2C Controller places the data in the receive buffer and issues a General Call interrupt
0000 000	1	START byte. For more details, refer to "START BYTE Transfer Protocol" 0000
0000 001	X	CBUS address. I2C Controller ignores these accesses
0000 010	X	Reserved
0000 011	X	Reserved
0000 1XX	X	High-speed master code (for more information, refer to "Multiple Master Arbitration"
1111 1XX	X	Reserved
1111 0XX	X	10-bit slave addressing

The I2C Controller does not restrict you from using these reserved addresses. However, if you use these reserved addresses, you may run into incompatibilities with other I2C components.

9.7.3.3TRANSMITTING AND RECEIVING PROTOCOLS

The master can initiate data transmission and reception to/from the bus, acting as either a master-transmitter or master-receiver. A slave responds to requests from the master to either transmit data or receive data to/from the bus, acting as either a slave-transmitter or slave-receiver, respectively.

Master-Transmitter and Slave-Receiver
All data is transmitted in byte format, with no limit on the number of bytes transferred per data transfer. After the master sends the address and R/W bit or the master transmits a byte of data to the slave, the slave-receiver must respond with the acknowledge signal (ACK). When a slave-receiver does not respond with an ACK pulse, the master aborts the transfer by issuing a STOP condition.

The slave must leave the SDA line high so that the master can abort the transfer. If the master-transmitter is transmitting data as shown in Figure 15, then the slave-receiver responds to the master-transmitter with an acknowledge pulse after every byte of data is received.

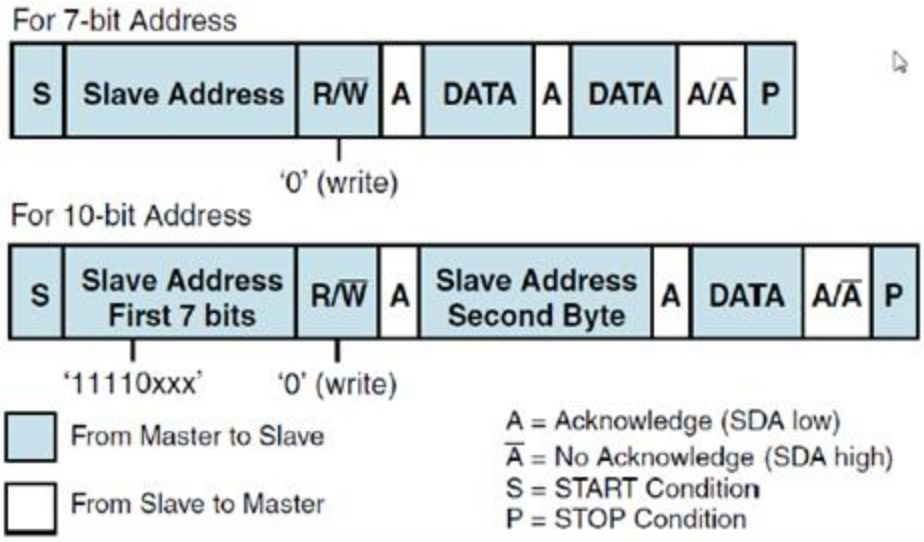


Figure 15 Master-Transmitter Protocol

Master-Receiver and Slave-Transmitter

If the master is receiving data as shown in Figure 16 then the master responds to the slave-transmitter with an acknowledge pulse after a byte of data has been received, except for the last byte. This is the way the master-receiver notifies the slave-transmitter that this is the last byte. The slave-transmitter relinquishes the SDA line after detecting the No Acknowledge (NACK) so that the master can issue a STOP condition.

When a master does not want to relinquish the bus with a STOP condition, the master can issue a RESTART condition. This is identical to a START condition except it occurs after the ACK pulse. The master can then communicate with the same slave or a different slave.

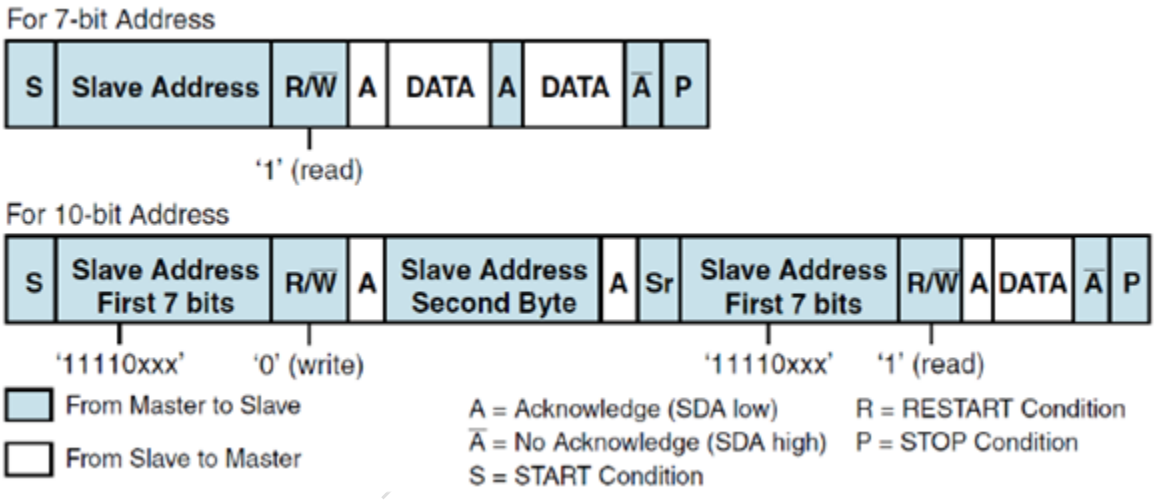


Figure 16 Master-Receiver Protocol

START BYTE Transfer Protocol

The START BYTE transfer protocol is set up for systems that do not have an on-board dedicated I2C hardware module. When the I2C Controller is addressed as a slave, it always samples the I2C bus at the highest speed supported so that it never requires a START BYTE transfer. However, when I2C Controller is a master, it supports the generation of START BYTE transfers at the beginning of every transfer in case a slave device requires it. This protocol consists of seven zeros being transmitted followed by a 1, as illustrated in Figure 17. This allows the processor that is polling the bus to under-sample the address phase until 0 is detected. Once the microcontroller detects a 0, it switches from the under sampling rate to the correct rate of the master.

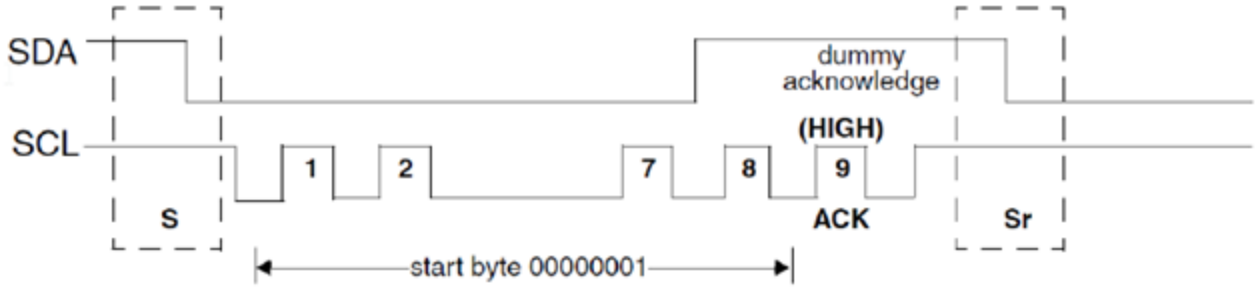


Figure 17 START BYTE Transfer

The START BYTE procedure is as follows:

1. Master generates a START condition.
2. Master transmits the START byte (0000 0001).
3. Master transmits the ACK clock pulse. (Present only to conform with the byte handling format used on the bus)
4. No slave sets the ACK signal to 0.
5. Master generates a RESTART (R) condition.

A hardware receiver does not respond to the START BYTE because it is a reserved address and resets after the RESTART condition is generated.

9.7.4 MULTIPLE MASTER ARBITRATION

The I2C Controller bus protocol allows multiple masters to reside on the same bus. If there are two masters on the same I2C-bus, there is an arbitration procedure if both try to take control of the bus at the same time by generating a START condition at the same time. Once a master (for example, a microcontroller) has control of the bus, no other master can take control until the first master sends a STOP condition and places the bus in an idle state.

Arbitration takes place on the SDA line, while the SCL line is 1. The master, which transmits a 1 while the other master transmits 0, loses arbitration and turns off its data output stage. The master that lost arbitration can continue to generate clocks until the end of the byte transfer. If both masters are addressing the same slave device, the arbitration could go into the data phase. Figure 18 illustrates the timing of when two masters are arbitrating on the bus.

For high-speed mode, the arbitration cannot go into the data phase because each master is programmed with a unique high-speed master code. This 8-bit code is defined by the system designer and is set by writing to the High Speed Master Mode Code Address Register, I2C_HS_MADDR. Because the codes are unique, only one master can win arbitration, which occurs by the end of the transmission of the high-speed master code.

Control of the bus is determined by address or master code and data sent by competing masters, so there is no central master or any order of priority on the bus. Arbitration is not allowed between the following conditions:

- A RESTART condition and a data bit
- A STOP condition and a data bit
- A RESTART condition and a STOP condition

Slaves are not involved in the arbitration process.

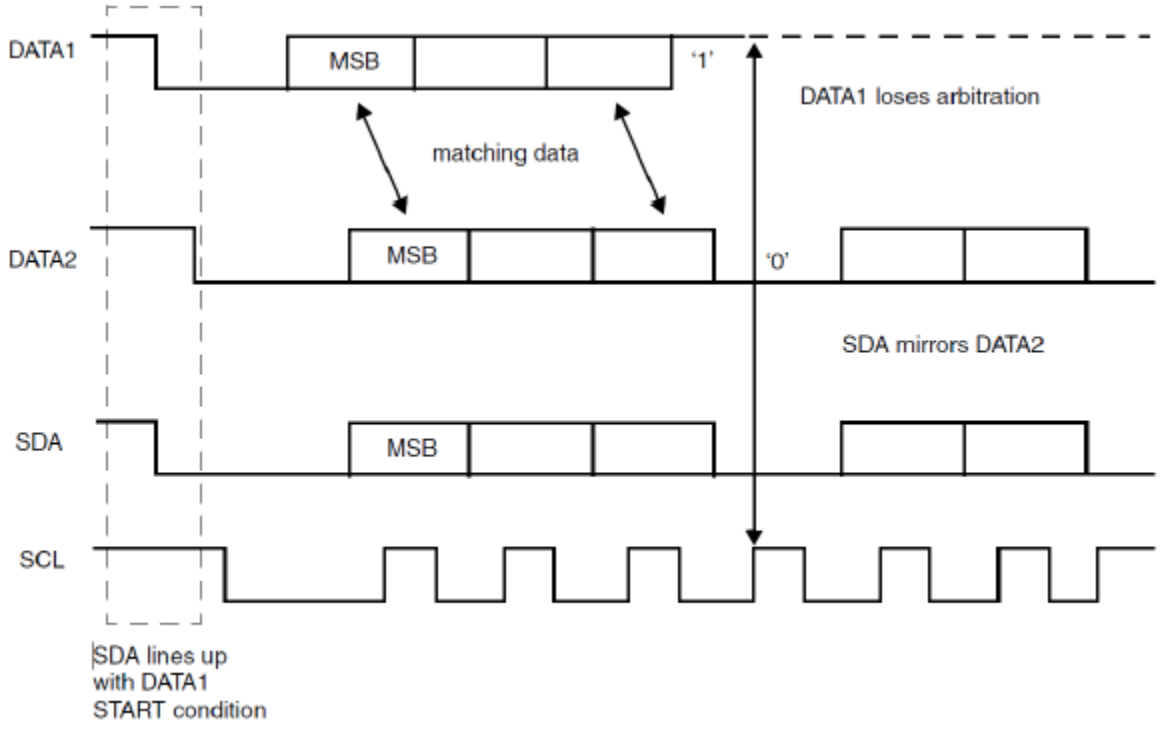


Figure 18 Multiple Master Arbitration

9.7.5 CLOCK SYNCHRONIZATION

When two or more masters try to transfer information on the bus at the same time, they must arbitrate and synchronize the SCL clock. All masters generate their own clock to transfer messages. Data is valid only during the high period of SCL clock. Clock synchronization is performed using the wired-AND connection to the SCL signal. When the master transitions the SCL clock to 0, the master starts counting the low time of the SCL clock and transitions the SCL clock signal to 1 at the beginning of the next clock period. However, if another master is holding the SCL line to 0, then the master goes into a HIGH wait state until the SCL clock line transitions to 1.

All masters then count off their high time and the master with the shortest high time transitions the SCL line to 0. The masters then counts out their low time and the one with the longest low time forces the other master into a HIGH wait state. Therefore, a synchronized SCL clock is generated, which is illustrated in Figure 19. Optionally, slaves may hold the SCL line low to slow down the timing on the I2C bus.

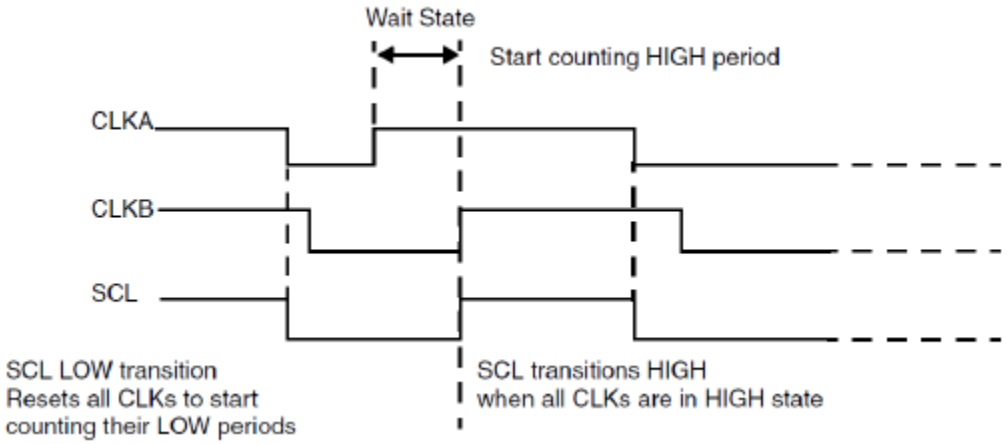


Figure 19 Multiple Master Clock Synchronization

9.7.6 OPERATION MODES

This section provides information on the following topics:

- Slave mode operation
- Master mode operation

Note: It is important that the I2C Controller should only be set to operate as an I2C Master, or I2C Slave, but not both simultaneously. This is achieved by ensuring that bit 6 (I2C_SLAVE_DISABLE) and bit 0 (I2C_MASTER_MODE) of the I2C_CON register are never set to 0 and 1, respectively.

9.7.6.1 SLAVE MODE OPERATION

This section includes the following procedures:

- Initial Configuration
- Slave-Transmitter Operation for a Single Byte
- Slave-Receiver Operation for a Single Byte
- Slave-Transfer Operation for Bulk Transfers

Initial Configuration

To use the I2C Controller as a slave, perform the following steps:

1. Disable the I2C Controller by writing a '0' to bit 0 of the I2C_ENABLE register.
2. Write to the I2C_SAR register (bits 9:0) to set the slave address. This is the address to which the I2C Controller responds.
3. Write to the I2C_CON register to specify which type of addressing is supported (7-bit or 10-bit by setting bit 3). Enable the I2C Controller in slave-only mode by writing a '0' into bit 6 (I2C_SLAVE_DISABLE) and a '0' to bit 0 (MASTER_MODE).
    a. Note: Slaves and masters do not have to be programmed with the same type of addressing 7-bit or 10-bit address. For instance, a slave can be programmed with 7-bit addressing and a master with 10-bit addressing, and vice versa.
4. Enable the I2C Controller by writing a '1' in bit 0 of the I2C_ENABLE register.
    a. Note: Depending on the reset values chosen, steps 2 and 3 may not be necessary because the reset values can be configured. For instance, if the device is only going to be a master, there would be no need to set the slave address because you can configure I2C Controller to have the slave disabled after reset and to enable the master after reset. The values stored are static and do not need to be reprogrammed if the I2C Controller is disabled.

Slave-Transmitter Operation for a Single Byte

When another I2C master device on the bus addresses the I2C Controller and requests data, the I2C Controller acts as a slave-transmitter and the following steps occur:

1. The other I2C master device initiates an I2C transfer with an address that matches the slave address in the I2C_SAR register of the I2C Controller.
2. The I2C Controller acknowledges the sent address and recognizes the direction of the transfer to indicate that it is acting as a slave-transmitter.

3. The I2C Controller asserts the RD_REQ interrupt (bit 5 of the I2C_RAW_INTR_STAT register) and holds the SCL line low. It is in a wait state until software responds. If the RD_REQ interrupt has been masked, due to I2C_INTR_MASK[5] register (M_RD_REQ bit field) being set to 0, then it is recommended that a hardware and/or software timing routine be used to instruct the CPU to perform periodic reads of the I2C_RAW_INTR_STAT register.
    a. Reads that indicate I2C_RAW_INTR_STAT[5] (R_RD_REQ bit field) being set to 1 must be treated as the equivalent of the RD_REQ interrupt being asserted.
    b. Software must then act to satisfy the I2C transfer.
    c. The timing interval used should be in the order of 10 times the fastest SCL clock period the I2C Controller can handle. For example, for 400 kb/s, the timing interval is 25us.
        i. Note: The value of 10 is recommended here because this is approximately the amount of time required for a single byte of data transferred on the I2C bus.
4. If there is any data remaining in the TX FIFO before receiving the read request, then the I2C Controller asserts a TX_ABRT interrupt (bit 6 of the I2C_RAW_INTR_STAT register) to flush the old data from the TX FIFO.
    a. Note: Because the I2C Controller's TX FIFO is forced into a flushed/reset state whenever a TX_ABRT event occurs, it is necessary for software to release the I2C Controller from this state by reading the I2C_CLR_TX_ABRT register before attempting to write into the TX FIFO. See register I2C_RAW_INTR_STAT for more details.
5. If the TX_ABRT interrupt has been masked, due to of I2C_INTR_MASK[6] register (M_TX_ABRT bit field) being set to 0, then it is recommended that re-using the timing routine (described in the previous step), or a similar one, be used to read the I2C_RAW_INTR_STAT register.
    a. Reads that indicate bit 6 (R_TX_ABRT) being set to 1 must be treated as the equivalent of the TX_ABRT interrupt being asserted.
    b. There is no further action required from software.
    c. The timing interval used should be similar to that described in the previous step for the I2C_RAW_INTR_STAT[5] register.
6. Software writes to the I2C_DATA_CMD register with the data to be written (by writing a '0' in bit 8).
7. Software must clear the RD_REQ and TX_ABRT interrupts (bits 5 and 6, respectively) of the I2C_RAW_INTR_STAT register before proceeding.

8. If the RD_REQ and/or TX_ABRT interrupts have been masked, then clearing of the I2C_RAW_INTR_STAT register will have already been performed when either the R_RD_REQ or R_TX_ABRT bit has been read as 1.
9. The I2C Controller releases the SCL and transmits the byte.
10. The master may hold the I2C bus by issuing a RESTART condition or release the bus by issuing a STOP condition.

Slave-Receiver Operation for a Single Byte

When another I2C master device on the bus addresses the I2C Controller and is sending data, the I2C Controller acts as a slave-receiver and the following steps occur:

1. The other I2C master device initiates an I2C transfer with an address that matches the I2C Controller's slave address in the I2C_SAR register.
2. The I2C Controller acknowledges the sent address and recognizes the direction of the transfer to indicate that the I2C Controller is acting as a slave-receiver.
3. I2C Controller receives the transmitted byte and places it in the receive buffer.
4. If the RX FIFO is completely filled with data when a byte is pushed, then an overflow occurs, and the I2C Controller continues with subsequent I2C transfers. Because a NACK is not generated, software must recognize the overflow when indicated by the I2C Controller (by the R_RX_OVER bit in the I2C_INTR_STAT register) and take appropriate actions to recover from lost data. Hence, there is a real time constraint on software to service the RX FIFO before the latter overflow as there is no way to re-apply pressure to the remote transmitting master. You must select a deep enough RX FIFO depth to satisfy the interrupt service interval of their system.
5. I2C Controller asserts the RX_FULL interrupt (I2C_RAW_INTR_STAT[2] register).
6. If the RX_FULL interrupt has been masked, due to setting I2C_INTR_MASK[2] register to 0 or setting I2C_TX_TL to a value larger than 0, then it is recommended that a timing routine (described in "Slave-Transmitter Operation for a Single Byte") be implemented for periodic reads of the I2C_STATUS register. Reads of the I2C_STATUS register, with bit 3 (RFNE) set at 1, must then be treated by software as the equivalent of the RX_FULL interrupt being asserted.
7. Software may read the byte from the I2C_DATA_CMD register (bits 7:0).
8. The other master device may hold the I2C bus by issuing a RESTART condition or release the bus by issuing a STOP condition.

Slave-Transfer Operation for Bulk Transfers

In the standard I2C protocol, all transactions are single byte transactions and the programmer responds to a remote master read request by writing one byte into the slave's TX FIFO. When a slave (slave-transmitter) is issued with a read request

(RD_REQ) from the remote master (master-receiver), at a minimum there should be at least one entry placed into the slave-transmitter's TX FIFO.

The I2C Controller is designed to handle more data in the TX FIFO so that subsequent read requests can take that data without raising an interrupt to get more data. Ultimately, this eliminates the possibility of significant latencies being incurred between raising the interrupt for data each time had there been a restriction of having only one entry placed in the TX FIFO.

This mode only occurs when I2C Controller is acting as a slave-transmitter. If the remote master acknowledges the data sent by the slave-transmitter and there is no data in the slave's TX FIFO, the I2C Controller holds the I2C SCL line low while it raises the read request interrupt (RD_REQ) and waits for data to be written into the TX FIFO before it can be sent to the remote master.

If the RD_REQ interrupt is masked, due to bit 5 (M_RD_REQ) of the I2C_INTR_STAT register being set to 0, then it is recommended that a timing routine be used to activate periodic reads of the I2C_RAW_INTR_STAT register. Reads of I2C_RAW_INTR_STAT that return bit 5 (R_RD_REQ) set to 1 must be treated as the equivalent of the RD_REQ interrupt referred to in this section. This timing routine is similar to that described in "Slave-Transmitter Operation for a Single Byte"

The RD_REQ interrupt is raised upon a read request, and like interrupts, must be cleared when exiting the interrupt service handling routine (ISR). The ISR allows you to either write 1 byte or more than 1 byte into the TX FIFO. During the transmission of these bytes to the master, if the master acknowledges the last byte, then the slave must raise the RD_REQ again because the master is requesting for more data.

If the programmer knows in advance that the remote master is requesting a packet of n bytes, then when another master addresses I2C Controller and requests data, the TX FIFO could be written with n number bytes and the remote master receives it as a continuous stream of data. For example, the I2C Controller slave continues to send data to the remote master as long as the remote master is acknowledging the data sent and there is data available in the TX FIFO. There is no need to hold the SCL line low or to issue RD_REQ again.

If the remote master is to receive n bytes from the I2C Controller but the programmer wrote a number of bytes larger than n to the TX FIFO, then when the slave finishes sending the requested n bytes, it clears the TX FIFO and ignores any excess bytes.

The I2C Controller generates a transmit abort (TX_ABRT) event to indicate the clearing of the TX FIFO in this example. At the time an ACK/NACK is expected, if a NACK is received, then the remote master has all the data it wants. At this time, a flag is raised within the slave's state machine to clear the leftover data in the TX FIFO. This flag is

transferred to the processor bus clock domain where the FIFO exists and the contents of the TX FIFO is cleared at that time.

9.7.6.2 MASTER MODE OPERATION

This section includes the following topics:

- Initial Configuration
- Master Transmit and Master Receive

Initial Configuration
The procedures are very similar and are only different with regard to where the I2C_10BITADDR_MASTER bit is set (either bit 4 of I2C_CON register or bit 12 of I2C_TAR register).

To use the I2C Controller as a master perform the following steps:

1. Disable the I2C Controller by writing 0 to the I2C_ENABLE register.

2. Write to the I2C_CON register to set the maximum speed mode supported (bits 2:1) and the desired speed of the I2C Controller master-initiated transfers, either 7-bit or 10-bit addressing (bit 4).

3. Ensure that bit 6 I2C_SLAVE_DISABLE = 1 and bit 0 MASTER_MODE =1
    a. Note: Slaves and masters do not have to be programmed with the same type of addressing 7-bit or 10-bit address. For instance, a slave can be programmed with 7-bit addressing and a master with 10-bit addressing, and vice versa.

4. Write to the I2C_TAR register the address of the I2C device to be addressed (bits 9:0). This register also indicates whether a General Call or a START BYTE command is going to be performed by I2C.

5. Only applicable for high-speed mode transfers. Write to the I2C_HS_MADDR register the desired master code for the I2C Controller. The master code is programmer-defined.

6. Enable the I2C Controller by writing a 1 in bit 0 of the I2C_ENABLE register.

7. Now write transfer direction and data to be sent to the I2C_DATA_CMD register. If the I2C_DATA_CMD register is written before the I2C Controller is enabled, the data and commands are lost as the buffers are kept cleared when I2C Controller is disabled.

8. This step generates the START condition and the address byte on the I2C Controller. Once I2C Controller is enabled and there is data in the TX FIFO, I2C Controller starts reading the data.

Note: Depending on the reset values chosen, steps 2, 3, 4, and 5 may not be necessary because the reset values can be configured. The values stored are static and do not need to be reprogrammed if the I2C Controller is disabled, with the exception of the transfer direction and data.

Master Transmit and Master Receive
The I2C Controller supports switching back and forth between reading and writing dynamically. To transmit data, write the data to be written to the lower byte of the I2C_DATA_CMD_REG register. The CMD bit [8] should be written to 0 for I2C write operations.

Subsequently, a read command may be issued by writing "don't cares" to the lower byte of the I2C_DATA_CMD_REG register, and a 1 should be written to the CMD bit.

The I2C Controller master continues to initiate transfers as long as there are commands present in the transmit FIFO. If the transmit FIFO becomes empty, the master inserts a STOP condition after completing the current transfer.

9.7.7 DISABLING THE I2C CONTROLLER

The register I2C_ENABLE_STATUS is added to allow software to unambiguously determine when the hardware has completely shut down in response to the I2C_ENABLE register being set from 1 to 0. Only one register is required to be monitored.

Procedure

1. Define a timer interval (ti2c_poll) equal to the 10 times the signaling period for the highest I2C transfer speed used in the system and supported by I2C Controller. For example, if the highest I2C transfer mode is 400kbit/s, then this ti2c_poll is 25 µs.
2. Define a maximum time-out parameter, MAX_T_POLL_COUNT, such that if any repeated polling operation exceeds this maximum value, an error is reported.
3. Execute a blocking thread/process/function that prevents any further I2C master transactions to be started by software but allows any pending transfers to be completed.
    a. Note: This step can be ignored if I2C Controller is programmed to operate as an I2C slave only.
4. The variable POLL_COUNT is initialized to zero.
5. Set I2C_ENABLE to 0.
6. Read the I2C_ENABLE_STATUS register and test the I2C_EN bit (bit 0). Increment POLL_COUNT by one. If POLL_COUNT >= MAX_T_POLL_COUNT, exit with the relevant error code.
7. If I2C_ENABLE_STATUS[0] is 1, then sleep for ti2c_poll and proceed to the previous step.

Otherwise, exit with a relevant success code.

9.8 UART

The **ISM14585-L35** contains two identical instances of this block, i.e. UART and UART2. The UART is compliant to the industry-standard 16550 and is used for serial communication with a peripheral. Data is written from a master (CPU) over the APB bus to the UART and it is converted to serial form and transmitted to the destination device. Serial data is also received by the UART and stored for the master (CPU) to read back. There is also DMA support on the UART block thus the internal FIFOs can be used. Both UARTs support hardware flow control signals (RTS, CTS).

Features:

- 16 bytes Transmit and receive FIFOs
- Hardware flow control support (CTS/RTS)
- Shadow registers to reduce software overhead and also include a software programmable reset
- Transmitter Holding Register Empty (THRE) interrupt mode
- IrDA 1.0 SIR mode supporting low power mode.
- Functionality based on the 16550 industry standard:
- Programmable character properties, such as number of data bits per character (5-8), optional parity bit (with odd or even select) and number of stop bits (1, 1.5 or 2)

- Line break generation and detection
- Prioritized interrupt identification
- Programmable serial data baud rate as calculated by the following: baud rate = (serial clock frequency)/(divisor).

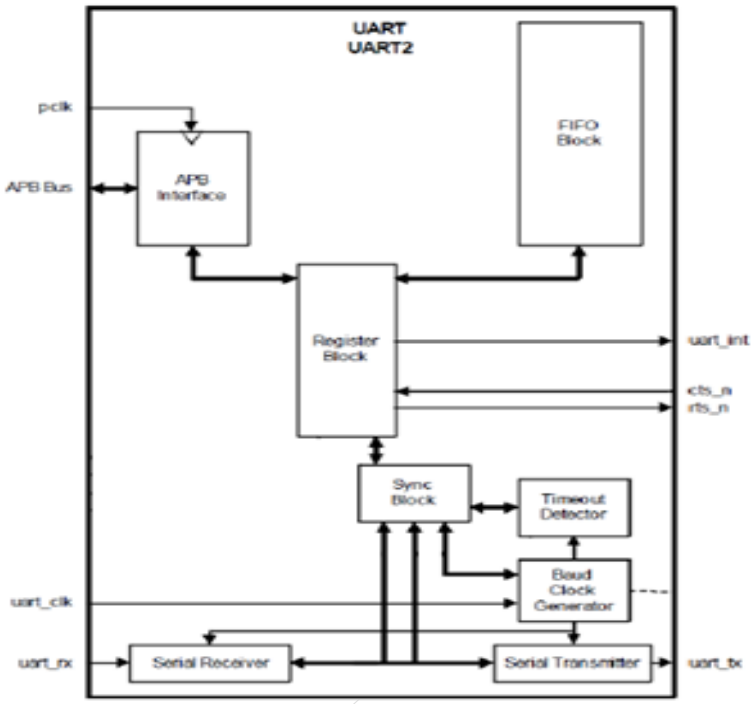


Figure 20 UART Block Diagram

9.8.1 UART (RS232) SERIAL PROTOCOL

Because the serial communication between the UART and the selected device is asynchronous, additional bits (start and stop) are added to the serial data to indicate the beginning and end. Utilizing these bits allows two devices to be synchronized. This structure of serial data accompanied by start and stop bits is referred to as a character, as shown in Figure 21.

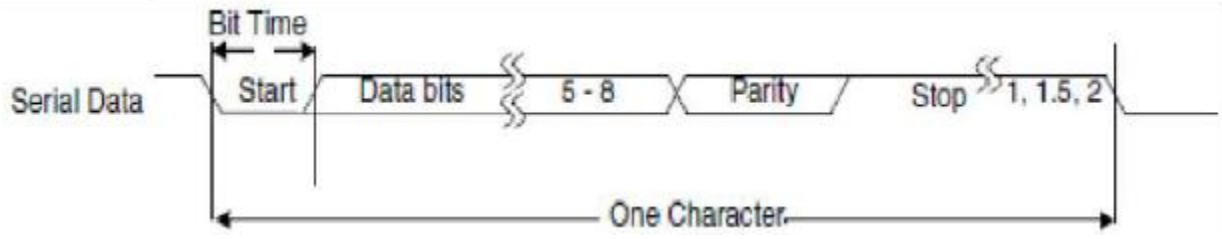


Figure 21 Serial Data Format

An additional parity bit may be added to the serial character. This bit appears after the last data bit and before the stop bit(s) in the character structure to provide the UART with the ability to perform simple error checking on the received data.

The UART Line Control Register (UART_LCR_REG) is used to control the serial character characteristics. The individual bits of the data word are sent after the start bit, starting with the least significant bit (LSB). These are followed by the optional parity bit, followed by the stop bit(s), which can be 1, 1.5 or 2.

All the bits in the transmission (with exception to the half stop bit when 1.5 stop bits are used) are transmitted for exactly the same time duration. This is referred to as a Bit Period or Bit Time. One BitTime equals 16 baud clocks. To ensure stability on the line the receiver samples the serial input data at approximately the mid-point of the Bit Time once the start bit has been detected. As the exact number of baud clocks that each bit was transmitted for is known, calculating the mid-point for sampling is not difficult, that is every 16 baud clocks after the mid-point sample of the start bit. Figure 22 shows the sampling points of the first couple of bits in a serial character.

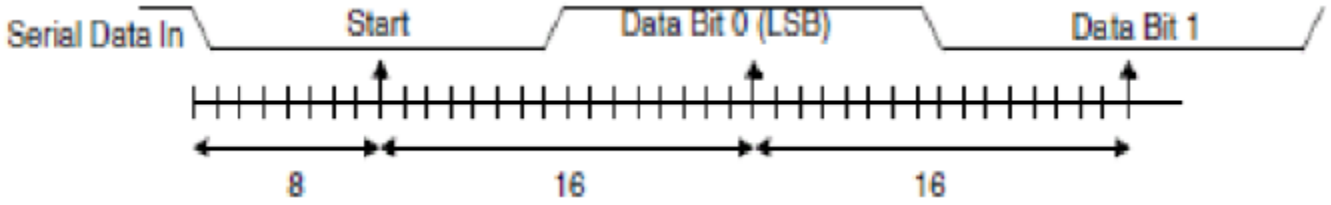


Figure 22 Receiver Serial Data Sampling Points

As part of the 16550 standard an optional baud clock reference output signal (baudout_n) is supplied to provide timing information to receiving devices that require it. The baud rate of the UART is controlled by the serial clock (sclk or pclk in a single clock implementation) and the Divisor Latch Register (DLH and DLL).

9.8.2 IRDA 1.0 SIR PROTOCOL

The Infrared Data Association (IrDA) 1.0 Serial Infrared (SIR) mode supports bidirectional data communications with remote devices using infrared radiation as the transmission medium. IrDA 1.0 SIR mode specifies a maximum baud rate of 115.2kBd.

Note: Information provided on IrDA SIR mode in this section assumes that the reader is fully familiar with the IrDA Serial Infrared Physical Layer Specifications. This specification can be obtained from the following website: http://www.irda.org.

The data format is similar to the standard serial (sout and sin) data format. Each data character is sent serially, beginning with a start bit, followed by 8 data bits, and ending with at least one stop bit.

Thus, the number of data bits that can be sent is fixed. No parity information can be supplied and only one stop bit is used while in this mode.

Trying to adjust the number of data bits sent or enable parity with the Line Control Register (LCR) has no effect. When the UART is configured to support IrDA 1.0 SIR it can be enabled with Mode Control Register (MCR) bit 6. When the UART is not configured to support IrDA SIR mode, none of the logic is implemented and the mode cannot be activated, reducing total gate counts. When SIR mode is enabled, and active, serial data is transmitted and received on the sir_out_n and sir_in ports, respectively.

Transmitting a single infrared pulse signals a logic zero, while a logic one is represented by not sending a pulse. The width of each pulse is 3/16th of a normal serial bit time. Thus, each new character begins with an infrared pulse for the start bit. However, received data is inverted from transmitted data due to the infrared pulses energizing the photo transistor base of the IrDA receiver, pulling its output low. This inverted transistor output is then fed to the UART sir_in port, which then has correct UART polarity. Figure 23 shows the timing diagram for the IrDA SIR data format in comparison to the standard serial format.

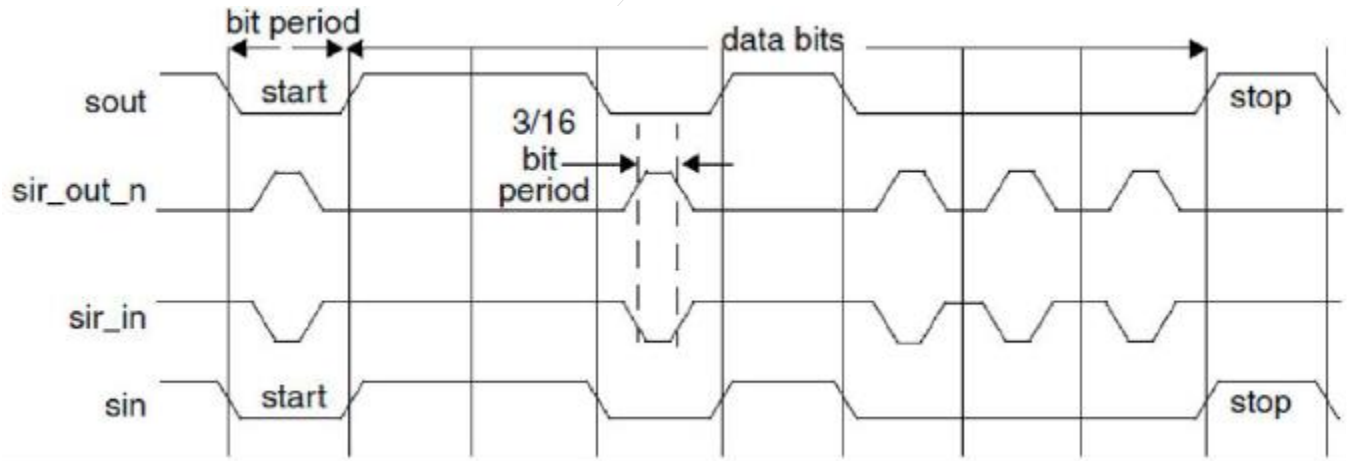


Figure 23 IrDA SIR Data Format

As detailed in the IrDA 1.0 SIR, the UART can be configured to support a low-power reception mode. When the UART is configured in this mode, the reception of SIR pulses of 1.41µs (minimum pulse duration) is possible, as well as nominal 3/16th of a normal serial bit time. Using this low-power reception mode requires programming the Low Power Divisor Latch (LPDLL/LPDLH) registers.

It should be noted that for all sclk frequencies greater than or equal to 7.37MHz (and obey the requirements of the Low Power Divisor Latch registers), pulses of 1.41µs are detectable. However, there are several values of sclk that do not allow the detection of such a narrow pulse and these are as follows:

Table: Low Power Divisor Latch Register Values:

SCLK	Low Power Divisor Latch Register Value	Min Pulse Width for Detection
1.84 MHz	1	3.77 μs
3.69 MHz	2	2.086 μs
5.33 MHz	3	1.584 μs

When IrDA SIR mode is enabled, the UART operation is similar to when the mode is disabled, with one exception; data transfers can only occur in half-duplex fashion when IrDA SIR mode is enabled. This is because the IrDA SIR physical layer specifies a minimum of 10ms delay between transmission and reception. This 10ms delay must be generated by software.

9.8.3 CLOCK SUPPORT

The UART has two system clocks (pclk and sclk). Having the second asynchronous serial clock (sclk) implemented accommodates accurate serial baud rate settings, as well as APB bus interface requirements.

With the two clock design a synchronization module is implemented for synchronization of all control and data across the two system clock boundaries.

A serial clock faster than four-times the pclk does not leave enough time for a complete incoming character to be received and pushed into the receiver FIFO. However, in most cases, the pclk signal is faster than the serial clock and this should never be an issue.

The serial clock modules must have time to see new register values and reset their respective state machines. This total time is guaranteed to be no more than eight clock cycles of the slower of the two system clocks. Therefore, no data should be transmitted or received before this maximum time expires, after initial configuration.

9.8.4 CLOCK SUPPORT

The assertion of the UART interrupt (UART_INT) occurs whenever one of the several prioritized interrupt types are enabled and active. The following interrupt types can be enabled with the IER register:

- Receiver Error
- Receiver Data Available
- Character Timeout (in FIFO mode only)
- Transmitter Holding Register Empty at/below threshold (in Programmable THRE interrupt mode)

When an interrupt occurs, the master accesses the UART_IIR_REG to determine the source of the interrupt before dealing with it accordingly. These interrupt types are described in more detail in below table

Interrupt ID Bits [3-0]	Interrupt Set and Reset Functions			
	Priority	**Interrupt Type**	**Interrupt Source**	**Interrupt Reset Control**
0001	-	None		
0110	Highest	Receiver Line status	Overrun/parity/ framing errors or break interrupt	Reading the line status register
0100	1	Receiver Data Available	Receiver data available (non-FIFO mode or FIFOs disabled) or RCVR FIFO trigger level reached (FIFO mode and FIFOs enabled)	Reading the receiver buffer register (non-FIFO mode or FIFOs disabled) or the FIFO drops below the trigger level (FIFO mode and FIFOs enabled)
1100	2	Character timeout indication	No characters in or out of the RCVR FIFO during the last 4 character times and there is at least 1 character in it during this time.	Reading the receiver buffer register
0010	3	Transmitter holding register empty	Transmitter holding register empty (Prog. THRE Mode disabled) or XMIT FIFO at or below threshold (Prog. THRE Mode enabled).	Reading the IIR register (if source of interrupt); or, writing into THR (FIFOs or THRE Mode not selected or disabled) or XMIT FIFO above threshold (FIFOs and THRE Mode selected and enabled).
0000	4	Reserved		
0111	Lowest	Reserved	-	-

9.8.5 PROGRAMMABLE THRE INTERRUPT

The UART can be configured to have a Programmable THRE Interrupt mode available to increase system performance.

When Programmable THRE Interrupt mode is selected it can be enabled via the Interrupt Enable Register (IER[7]). When FIFOs and the THRE Mode are implemented and enabled, THRE Interrupts are active at, and below, a programmed transmitter FIFO empty threshold level, as opposed to empty, as shown in the flowchart in Figure 24.

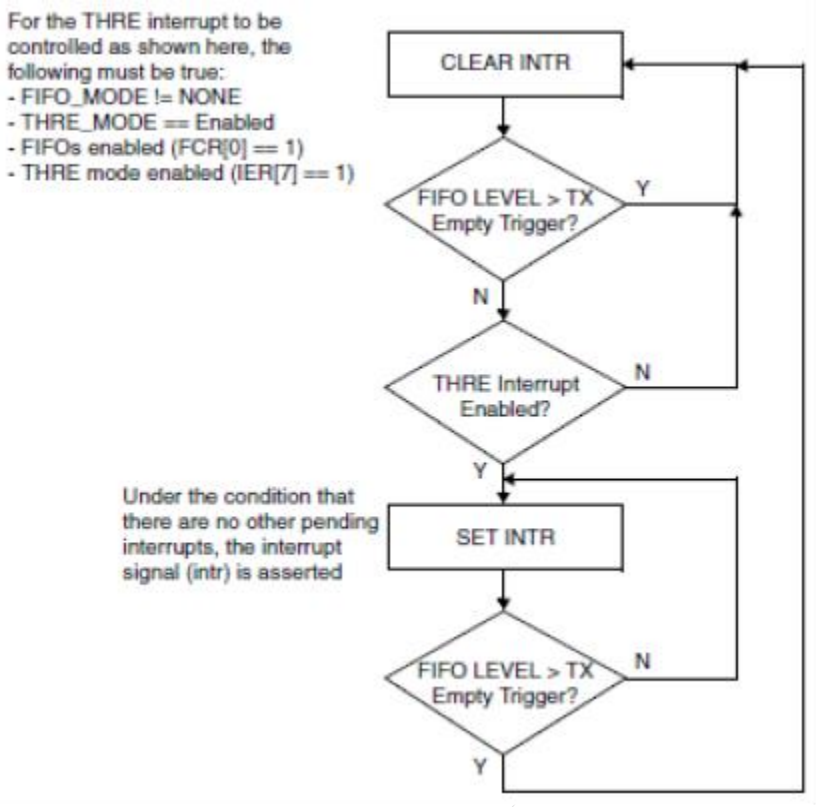


Figure 24 Flowchart of Interrupt Generation for Programmable THRE Interrupt Mode

This threshold level is programmed into FCR[5:4]. The available empty thresholds are: empty, 2, ¼ and ½. See UART_FCR_REG for threshold setting details. Selection of the best threshold value depends on the system's ability to begin a new transmission sequence in a timely manner. However, one of these thresholds should prove optimum in increasing system performance by preventing the transmitter FIFO from running empty.

In addition to the interrupt change, Line Status Register (LSR[5]) also switches function from indicating transmitter FIFO empty, to FIFO full. This allows software to fill the FIFO each transmit sequence by polling LSR[5] before writing another character. The flow then becomes, “fill transmitter FIFO whenever an interrupt occurs and there is data to transmit”, instead of waiting until the FIFO is completely empty. Waiting until the FIFO is empty causes a performance hit whenever the system is too busy to respond immediately.

Even if everything else is selected and enabled, if the FIFOs are disabled via FCR[0], the Programmable THRE Interrupt mode is also disabled. When not selected or disabled, THRE interrupts and LSR[5] function normally (both reflecting an empty THR

or FIFO). The flowchart of THRE interrupt generation when not in programmable THRE interrupt mode is shown in Figure 25.

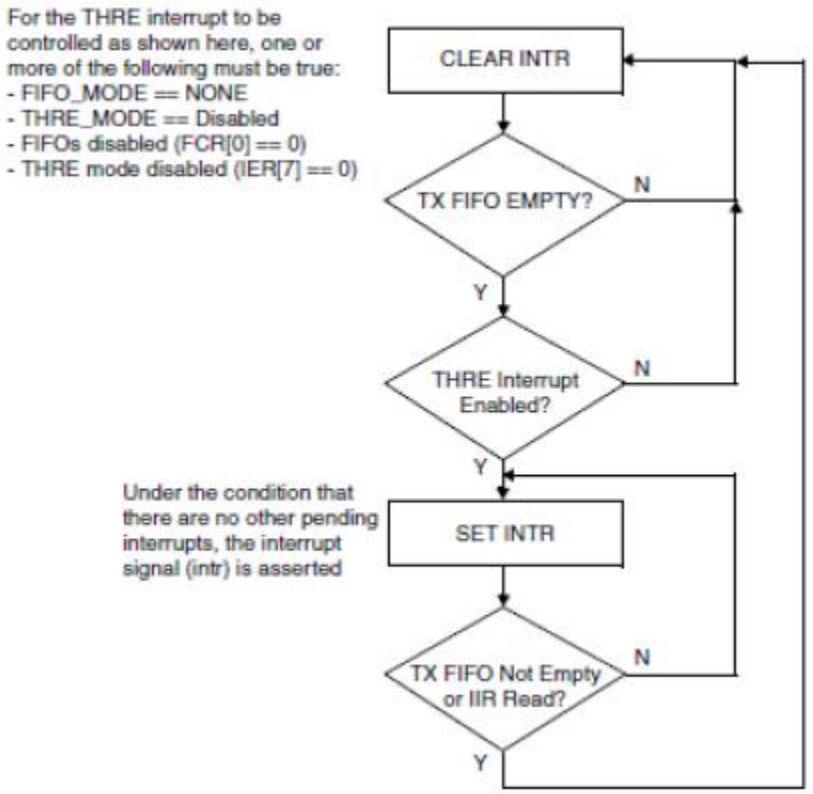


Figure 25 Flowchart of Interrupt Generation When Not in Programmable THRE Interrupt Mode

9.8.6 SHADOW REGISTERS

The shadow registers shadow some of the existing register bits that are regularly modified by software. These can be used to reduce the software overhead that is introduced by having to perform read-modify-writes.

- UART_SRBR_REG support a host burst mode where the host increments its address but still accesses the same Receive buffer register.
- UART_STHR support a host burst mode where the host increments its address but still accesses the same transmit holding register.
- UART_SFE_REG accesses the FCR[0] register without accessing the other UART_FCR_REG bits.
- UART_SRT_REG accesses the FCR[7-6] register without accessing the other UART_FCR_REG bits.

- UART_STER_REG accesses the FCR[5-4] register without accessing the other UART_FCR_REG bits.

9.8.7 DIRECT TEST MODE

The on-chip UARTS can be used for the Direct Test Mode required for the final product PHY layer testing. It can be done either over the HCI layer, which engages a full CTS/RTS UART or using a 2- wire UART directly as described in the Bluetooth Low Energy Specification (Volume 6, Part F).

9.9 SPI+ Interface

This interface supports a subset of the Serial Peripheral Interface (SPI™). The serial interface can transmit and receive 8, 16 or 32 bits in master/slave mode and transmit 9 bits in master mode. The SPI+ interface has enhanced functionality with bidirectional 2x16-bit word FIFOs.

Features

- Slave and Master mode
- 8 bit, 9 bit, 16 bit or 32 bit operation
- Clock speeds up to 16 MHz for the SPI controller. Programmable output frequencies of SPI source clock divided by 1, 2, 4, 8
- SPI clock line speed up to 8 MHz
- SPI mode 0, 1, 2, 3 support (clock edge and phase)
- Programmable SPI_DO idle level
- Maskable Interrupt generation
- Bus load reduction by unidirectional writes-only and reads-only modes.
- Built-in RX/TX FIFOs for continuous SPI bursts.
- DMA support

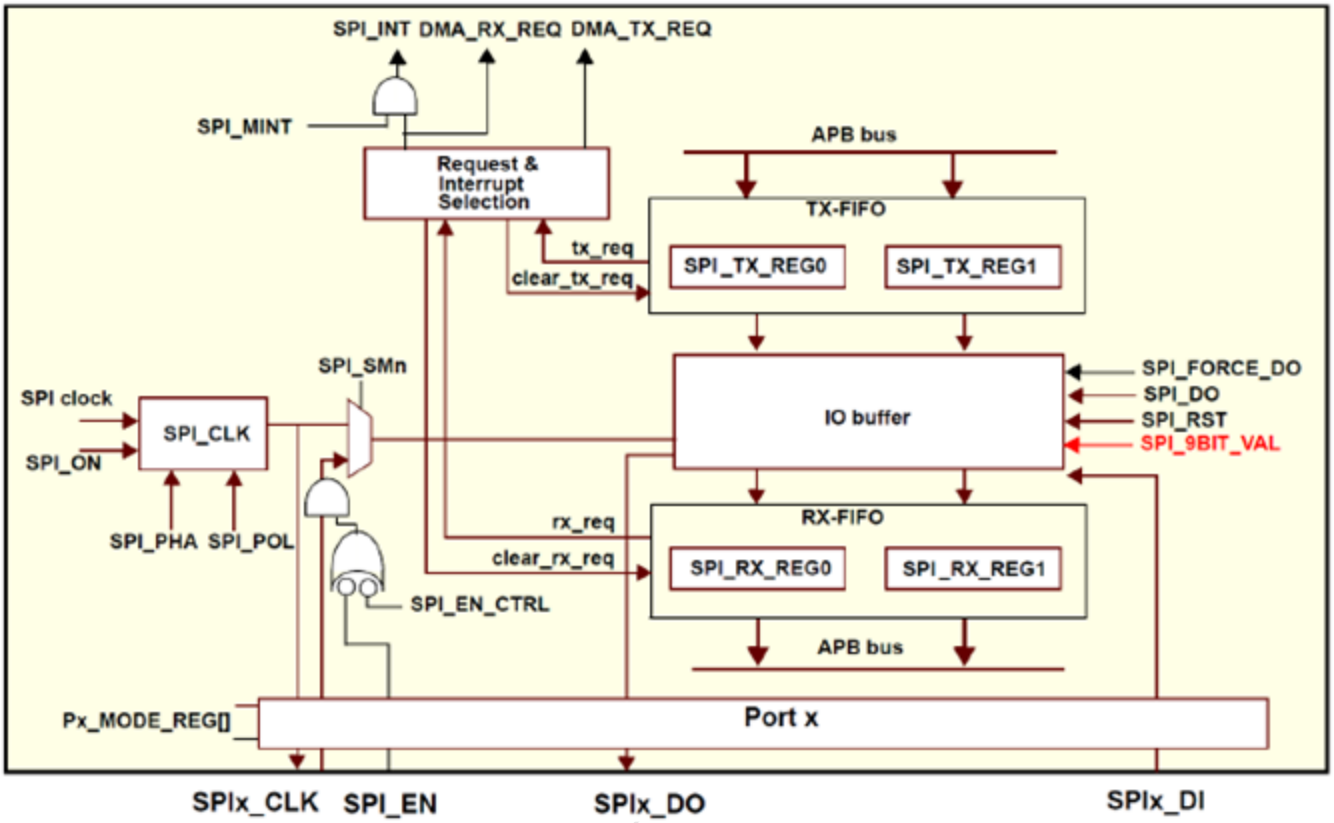


Figure 26 SPI Block Diagram

9.9.1 OPERATION WITHOUT FIFOS

This mode is the default mode.

Master Mode

To enable SPITM operation, first the individual port signal must be enabled. Next the SPI must be configured in SPI_CTRL_REG, for the desired mode. Finally bit SPI_ON must be set to 1.

A SPI transfer cycle starts after writing to the SPI_RX_TX_REG0. In case of 32 bits mode, the PI_RX_TX_REG1 must be written first. Writing to SPI_RX_TX_REG0 also sets the SPI_TXH. As soon as the holding register is copied to the IO buffer, the SPI_TXH is reset and a serial transfer cycle of 8/9/16/32 clock-cycles is started which causes 8/9/16/32 bits to be transmitted on SPI_DO. Simultaneously, data is received on SPI_DI and shifted into the IO buffer. The transfer cycle finishes after the 8th/9th/16th/32nd clock cycle and SPI_INT_BIT bit is set in the SPI_CTRL_REG and

SPI_INT_PEND bit in (RE)SET_INT_PENDING_REG is set. The received bits in the IO buffer are copied to the SPI_RX_TX_REG0 (and SPI_RX_TX_REG1 in case of 32 bits mode) were they can be
read by the CPU.

Interrupts to the CPU can be disabled using the SPI_MINT bit. To clear the SPI interrupt source, any value to SPI_CLEAR_INT_REG must be written. Note however that SPI_INT will be set as long as the RX-FIFO contains unread data.

Slave Mode
The slave mode is selected with SPI_SMn set to 1 and the Px_MODE_REG must also select SPI_CLK as input. The functionality of the IO buffer in slave and master mode is identical. The SPI module clocks data in on SPI_DI and out on SPI_DO on every active edge of SPI_CLK. As shown in Figure 24 to Figure 27, Figure 27: SPI Master/slave, Mode 3: SPI_POL=1 and SPI_PHA=1. The SPI has an active low clock enable SPI_EN, which can be enabled with bit SPI_EN_CTRL=1.

In slave mode the internal SPI clock must be more than four times the SPI_CLK

In slave mode the SPI_EN serves as a clock enable and bit synchronization If enabled with bit SPI_EN_CTRL. As soon as SPI_EN is deactivated between the MSB and LSB bits, the I/O buffer is reset.

SPI_POL and SPI_PHA
The phase and polarity of the serial clock can be changed with bits SPI_POL and SPI_PHA in the SPI_CTRL_REG.

SPI_DO Idle Levels
The idle level of signal SPI_DO depends on the master or slave mode and polarity and phase mode of the clock.

In master mode pin SPI_DO gets the value of bit SPI_DO if the SPI is idle in all modes. Also, if slave in SPI modes 0 and 2, SPI_DO is the initial and final idle level.

In SPI modes 1 and 3 however there is no clock edge after the sampled lsb and pin SPI_DO gets the lsb value of the IO buffer. If required, the SPI_DO can be forced to the SPI_DO bit level by resetting the SPI to the idle state by shortly setting bit SPI_RST to 1. (Optionally SPI_FORCE_DO can be set, but this does not reset the IO buffer). The following diagrams show the timing of the SPITM interface.

Writes Only Mode
In “writes only” mode (SPI_FIFO_MODE = “10“) only the TX-FIFO is used. Received data will be copied to the SPI_RX_TX_REGx, but if a new SPI transfer is finished before the old data is read from the memory, this register will be overwritten.

SPI_INT acts as a tx_request signal, indicating that there is still place in the FIFO. It will be ‘0’ when the FIFO is full or else ‘1’ when it’s not full. This is also indicated in the SPI_CTRL_REG[SPI_TXH], which is ‘1’ if the TX-FIFO is full. Writing to the FIFO if this bit is still 1, will result in transmission of undefined data. If all data has been transferred, SPI_CTRL_REG1 [SPI_BUSY] will become ‘0’.

Reads Only Mode
In “reads only” mode (SPI_FIFO_MODE = “01“) only the RX-FIFO is used. Transfers will start immediately when the SPI is turned on in this mode. In transmit direction the SPI_DO pin will transmit the IO buffer contents being the actual value of the SPI_TX_REGx (all 0’s after reset). This means that no dummy writes are needed for reads only transfers.

In Slave mode transfers only take place if the external master initiates them, but in master mode this means that transfers will continue until the RX-FIFO is full. If this happens SPI_CTRL_REG1[SPI_BUSY] will become ‘0’. If exactly N words need to be read from SPI device, first read (N - fifosize+1) words. Then wait until the SPI_BUSY becomes ‘0’, set SPI_FIFO_MODE to “00” and finally read the remaining (fifosize +1) words. Here fifosize is 4/2/1 words for 8/16/32 bits mode respectively.

If this is not done, more data will be read from the SPI device until the FIFO is completely filled, or the SPI is turned off.

Bidirectional transfers with FIFO
If SPI_FIFO_MODE is “00“, both registers are used as a FIFO. SPI_TXH indicates that TX-FIFO is full, SPI_INT indicates that there is data in the RX-FIFO.

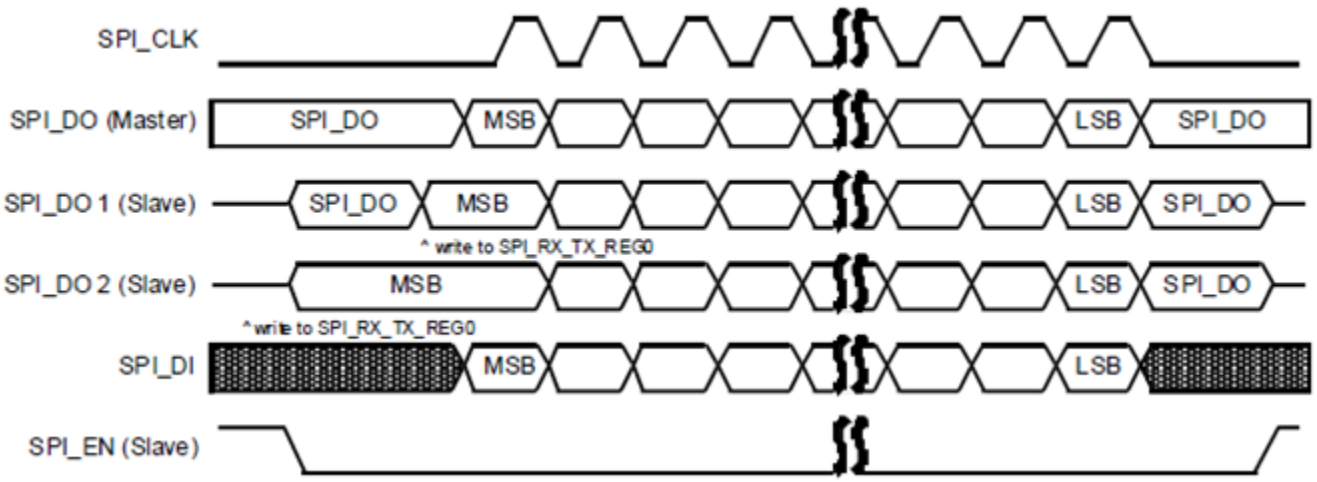


Figure 27 SPI Master/Slave, Mode 0: SPI_POL=0 and SPI_PHA=0

Note 1: If 9 bits SPI mode, the MSB bit in transmit direction is determined by bit SPI_CTRL_REG[SPI_9BIT_VAL]. In receive direction, the MSB is received but not stored.

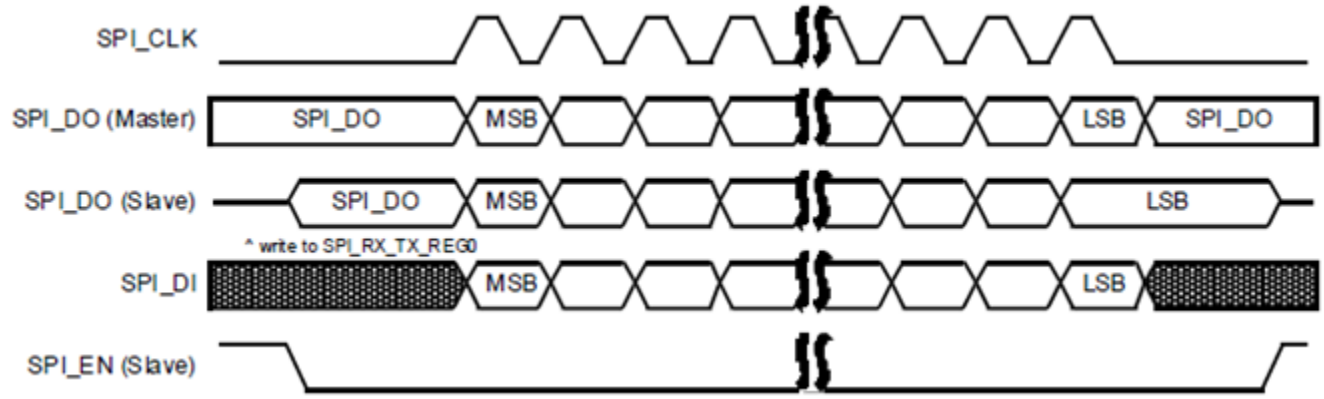


Figure 28 SPI Master/Slave, Mode 1: SPI_POL=0 and SPI_PHA=1

For the MSB bit refer to Note 1

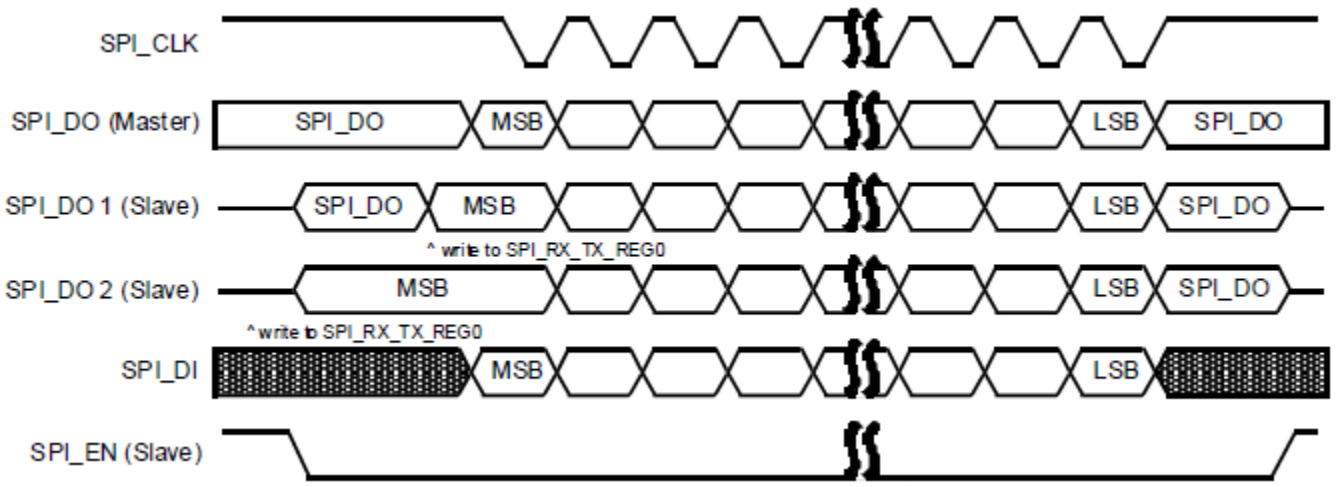


Figure 29 SPI Master/Slave, Mode 2: SPI_POL=1 and SPI_PHA=0

For the MSB bit refer to Note 1.

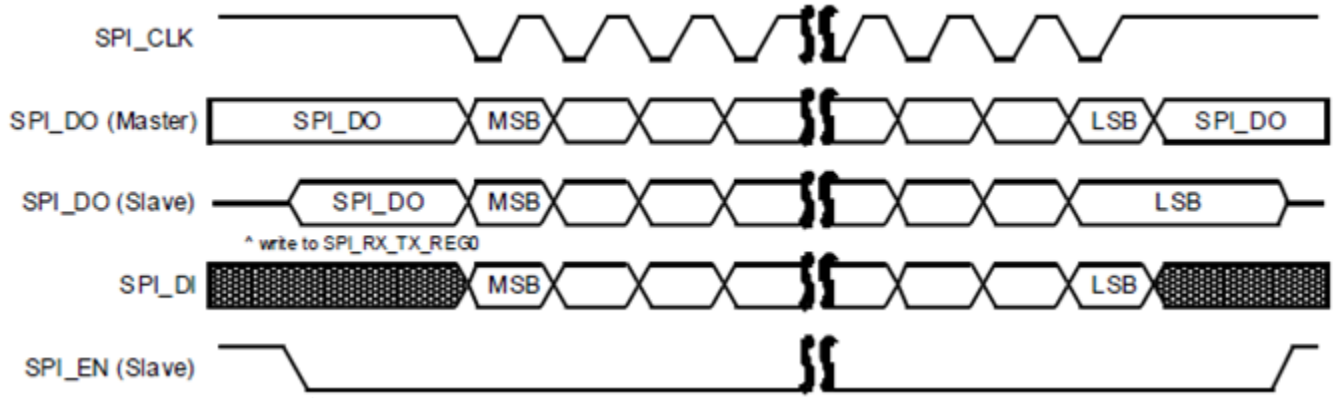


Figure 30 SPI Master/slave, Mode 3: SPI_POL=1 and SPI_PHA=1

For the MSB bit refer to Note 1

9.10 Wake-Up Timer

The Wake-up timer can be programmed to wake up the ISM14585 from power down mode after a pre-programmed number of GPIO events.

Each of the GPIO inputs can be selected to generate an event by programming the corresponding WKUP_SELECT_Px_REG register. When all WKUP_SELECT_Px_REG registers are configured to generate a wake-up interrupt, a toggle on any GPIO will wake up the system.

- The input signal edge can be selected by programming the WKUP_POL_Px_REG register.
- The block diagram illustrating the Wake-up function is shown in Figure 31.

Features

- Monitors any GPIO state change
- Implements debouncing time from 0ms up to 63ms
- Accumulates external events and compares the number to a programmed value
- Generates an interrupt to the CPU

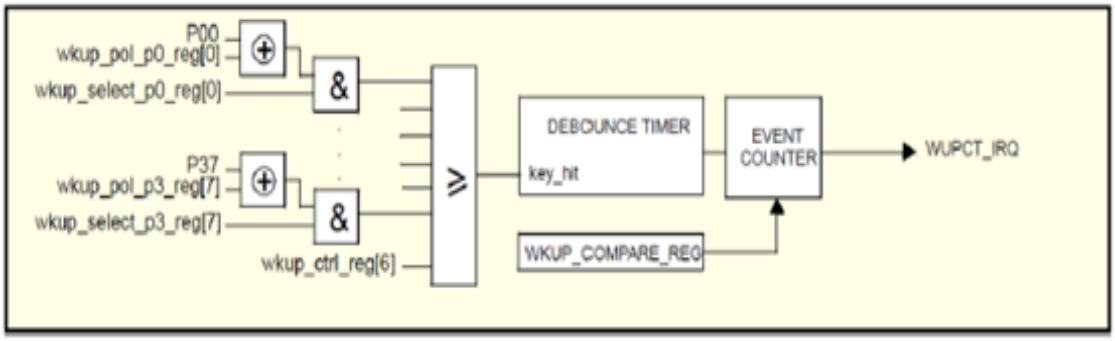


Figure 31 Wake-Up Timer Block Diagram

A LOW to HIGH level transition on the selected input port, while WKUP_POL_Px_REG[y] = 0, sets internal signal “key_hit” to ‘1’. This signal triggers the event counter state machine as shown in Figure 31. The debounce timer is loaded with value WKUP_CTRL_REG[WKUP_DEB_VALUE]. The timer counts down every 1ms. If the timer reaches 0 and the “key_hit” signal is still ‘1’, the event counter will be incremented.

The event counter is edge sensitive. After detecting an active edge, a reverse edge must be detected first before it goes back to the IDLE state and from there starts waiting for a new active edge.

If the event counter is equal to the value set in the WKUP_COMPARE_REG register, the counter will be reset, and an interrupt will be generated, if it was enabled by WKUP_CTRL_REG[ENABLE_IRQ].

The interrupt can be cleared by writing any value to register WKUP_RESET_IRQ_REG. The event counter can be reset by writing any value to register

WKUP_RESET_CNTR_REG. The value of the event counter can be read at any time by reading register WKUP_COUNTER_REG.

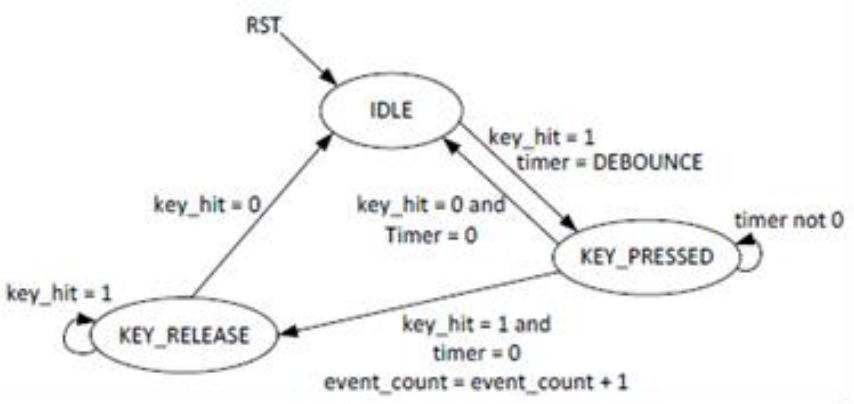


Figure 32 Event Counter State Machine for the Wake-Up Interrupt Generator

9.11 General Purpose Timers

The Timer block contains two timer modules that are software controlled, programmable and can be used for various tasks. Timer 0 is a 16-bit general purpose timer with a PWM output capability. Timer 2 is a 14-bit counter that generates three identical PWM signals in a quite flexible manner.

9.11.1 TIMER 0

Timer 0 is a 16-bit general purpose software programmable timer, which has the ability of generating Pulse Width Modulated signals, namely PWM0 and PWM1. It also generates the SWTIM_IRQ interrupt to the ARM Cortex-M0. It can be configured in various modes regarding output frequency, duty cycle and the modulation of the PWM signals.

Features:

- 16-bit general purpose timer
- Ability to generate 2 Pulse Width Modulated signals (PWM0 and PWM1)
- Programmable output frequency: $f = \frac{(16,\ 8,\ 4,\ 2\ MHz\ or\ 32\ kHz)}{(M+1)+(N+1)}$ with N = 0 to (216-1), M = 0 to (216-1)
- Programmable duty cycle: $\delta = \frac{M+1}{(M+1)+(N+1)} \times 100\ \%$
- Separately programmable interrupt timer: $T = \frac{(16,\ 8,\ 4,\ 2\ MHz\ or\ 32\ kHz)}{(ON+1)}$

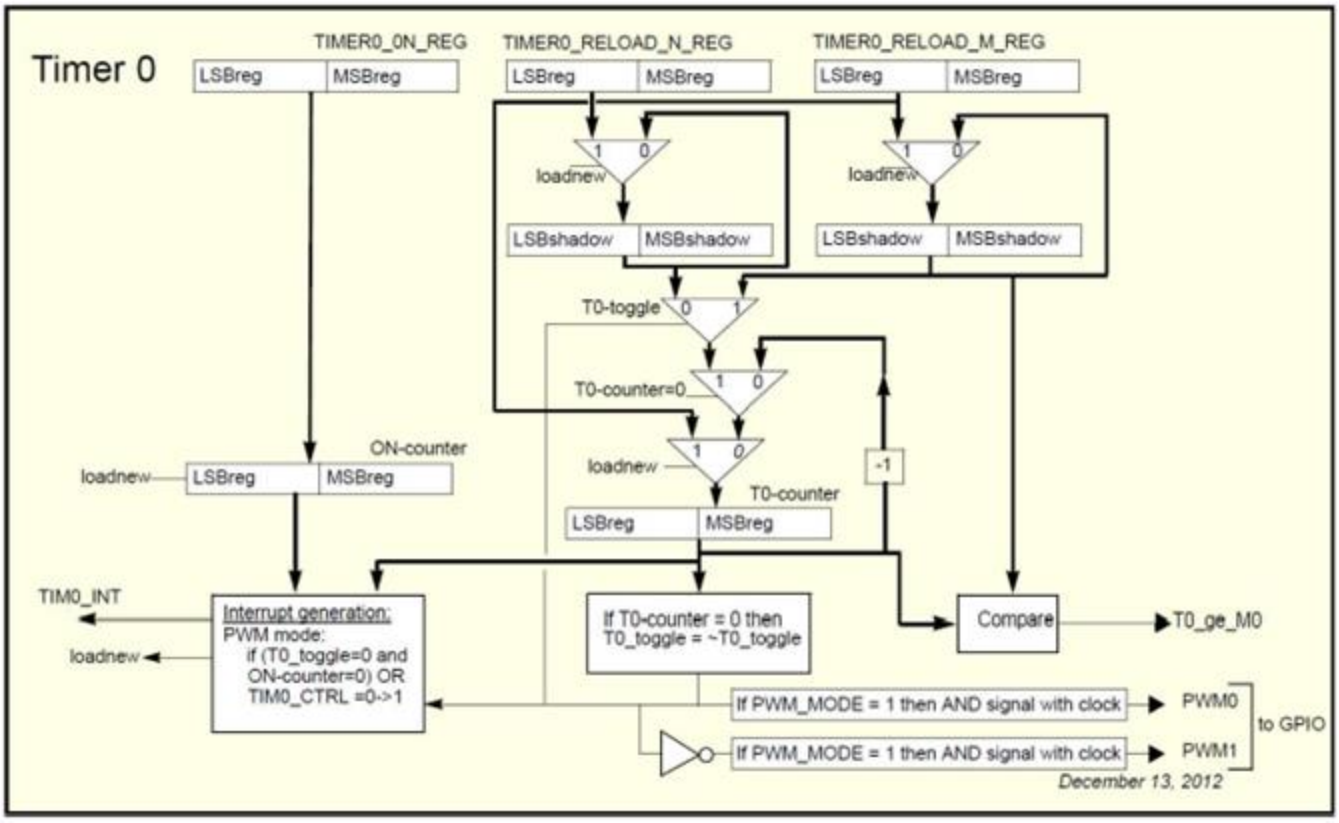


Figure 33 Timer 0 Block Diagram

Figure 33 shows the block diagram of Timer 0. The 16 bits timer consists of two counters: T0-counter and ON-counter, and three registers: TIMER0_RELOAD_M_REG, TIMER0_RELOAD_N_REG and TIMER0_ON_REG. Upon reset, the counter and register values are 0x0000. Timer 0 will generate a Pulse Width Modulated signal PWM0. The frequency and duty cycle of PWM0 are determined by the contents of the TIMER0_RELOAD_N_REG and the TIMER0_RELOAD_M_REG registers.
The timer can run at five different clocks: 16 MHz, 8 MHz, 4 MHz, 2 MHz or 32 kHz. The 32 kHz clock is selected by default with bit TIM0_CLK_SEL in the TIMER0_CTRL_REG register. This 'slow' clock has no enabling bit. The other four options can be selected by setting the TIM0_CLK_SEL bit and the TMR_ENABLE bit in the CLK_PER_REG (default disabled). This register also controls the frequency via the TMR_DIV bits. An extra clock divider is available that can be activated via bit TIM0_CLK_DIV of the timer control register TIMER0_CTRL_REG. This clock divider is only used for the ON-counter and always divides by 10. Timer 0 operates in PWM mode. The signals PWM0 and PWM1 can be mapped to any GPIOs.

Timer 0 PWM Mode

If bit TIM0_CTRL in the TIMER0_CTRL_REG is set, Timer 0 will start running. SWTIM_IRQ will be generated and the T0-counter will load its start value from the TIMER0_RELOAD_M_REG register, and will decrement on each clock. The ON-counter also loads its start value from the TIMER0_ON_REG register and decrements with the selected clock.

When the T0-counter reaches zero, the internal signal T0-toggle will be toggled to select the TIMER0_RELOAD_N_REG whose value will be loaded in the T0-counter. Each time the T0-counter reaches zero it will alternately be reloaded with the values of the M0- and N0-shadow registers respectively. PWM0 will be high when the M0-value decrements and low when the N0-value decrements. For PWM1 the opposite is applicable since it is inverted. If bit PWM_MODE in the TIMER0_CTRL_REG register is set, the PWM signals are not HIGH during the ‘high time’ but output a clock in that stage. The frequency is based on the clock settings defined in the CLK_PER_REG register (also in 32 kHz mode), but the selected clock frequency is divided by two to get a 50 % duty cycle.

If the ON-counter reaches zero it will remain zero until the T0-counter also reaches zero, while decrementing the value loaded from the TIMER0_RELOAD_N_REGregister (PWM0 is low). The counter will then generate an interrupt (SWTIM_IRQ). The ON-counter will be reloaded with the value of the TIMER0_ON_REG register. The T0-counter as well as the M0-shadow register will be loaded with the value of the TIMER0_RELOAD_M_REG register. At the same time, the N0-shadow register will be loaded by the TIMER0_RELOAD_N_REG register. Both counters will be decremented on the next clock again and the sequence will be repeated.

Note that it is possible to generate interrupts at a high rate, when selecting a high clock frequency in combination with low counter values. This could result in missed interrupt events.

During the time that the ON-counter is non-zero, new values for the ON-register, M0-register and N0-register can be written, but they are not used by the T0-counter until a full cycle is finished. More specifically, the newly written values in the TIMER0_RELOAD_M_REG and TIMER0_RELOAD_N_REG registers are only stored into the shadow registers when the ON-counter and the T0-counter have both reached zero and the T0-counter was decrementing the value loaded from the TIMER0_RELOAD_N_REG register (see Figure 34).

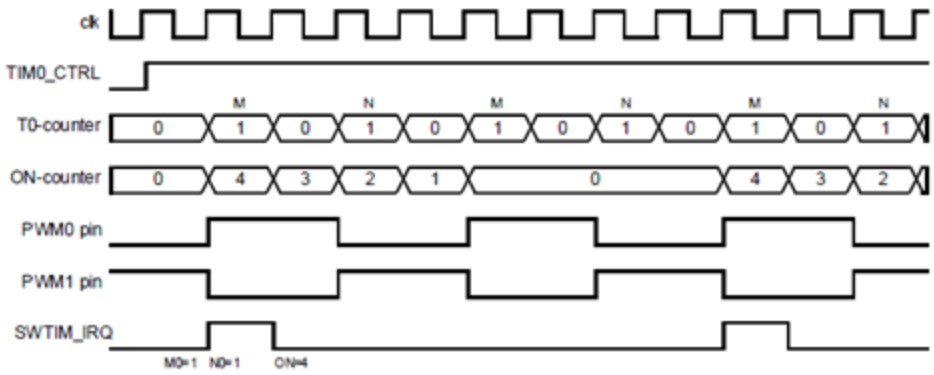


Figure 34 Timer 0 PWM Mode

At start-up both counters and the PWM0 signal are LOW so also at start-up an interrupt is generated. If Timer 0 is disabled all flip-flops, counters and outputs are in reset state except for the ON-register, the TIMER0_RELOAD_N_REG register and the TIMER0_RELOAD_M_REG register.

The timer input registers ON-register, TIMER0_RELOAD_N_REGand TIMER0_RELOAD_M_REG can be written, and the counter registers ON-counter and T0-counter can be read. When reading from the address of the ON-register, the value of the ON-counter is returned. Reading from the address of either the TIMER0_RELOAD_N_REG or the TIMER0_RELOAD_M_REG register, returns the value of the T0-counter.

It is possible to freeze Timer 0 with bit FRZ_SWTIM of the register SET_FREEZE_REG. When the timer is frozen the timer counters are not decremented. This will freeze all the timer registers at their last value. The timer will continue its operation again when bit FRZ_SWTIM is cleared via register RESET_FREEZE_REG.

9.11.2 TIMER 2

Timer 2 has three Pulse Width Modulated (PWM) outputs. The block diagram is shown in Figure 35

Features:

- 14-bit general purpose timer

- Ability to generate 3 Pulse Width Modulated signals (PWM2, PWM3 and PWM4)

- Input clock frequency: $f_{IN} = \frac{sys\_clk}{N}$ with N = 1, 2, 4 or 8 and sys_clk = 16 MHz or 32 kHz
- Programmable output frequency: $f_{OUT} = \left(\frac{f_{IN}}{2}\right) to \left(\frac{f_{IN}}{2^{14}-1}\right)$
- Three outputs with programmable duty cycle from 0% to 100%
- Used for white LED intensity (on/off) control

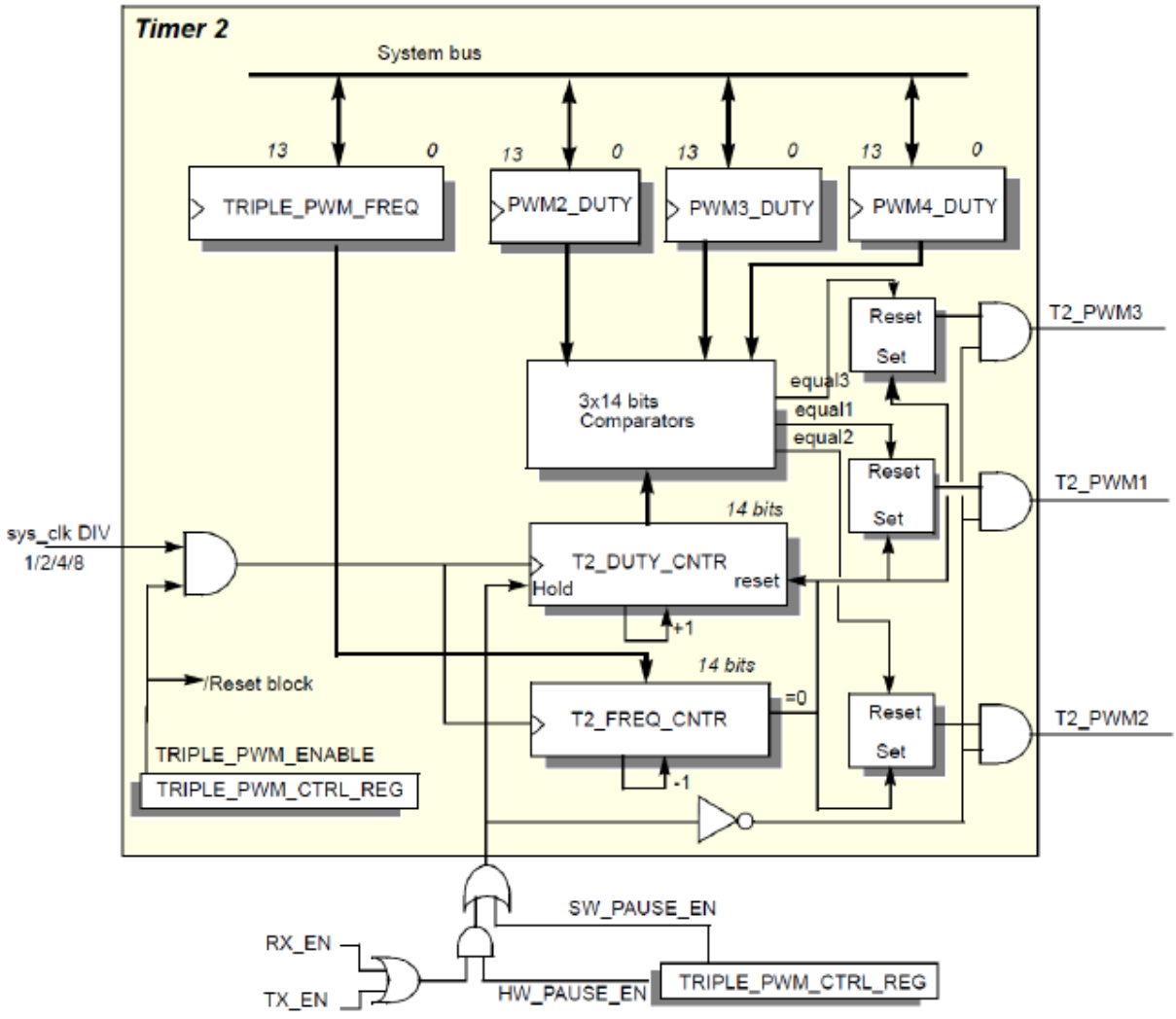


Figure 35 Timer 2 PWM Block Diagram

The Timer 2 is clocked with the system clock divided by TMR_DIV (1, 2, 4 or 8) and can be enabled with TRIPLE_PWM_CTRL_REG[TRIPLE_PWM_ENABLE].

T2_FREQ_CNTR determines the output frequency of the T2_PMWn output. This counter counts down from the value stored in register TRIPLE_PWM_FREQUENCY. At counter value 0, T2_FREQ_CNTR sets the T2_PWMn output to '1' and the counter is reloaded again.

T2_DUTY_CNTR is an up-counter that determines the duty cycle of the T2_PWMn output signal. After the block is enabled, the counter starts from 0. If T2_DUTY_CNTR is equal to the value stored in the respective PWMn_DUTY_CYCLE register, this resets the T2_PWMn output to 0. T2_DUTY_CNTR is reset when TRIPLE_PWM_FREQUENCY is 0.

Note that the value of PWMn_DUTY_CYCLE must be less or equal than

TRIPLE_PWM_FREQUENCY.

The Timer 2 is enabled/disabled by programming the TRIPLE_PWM_CTRL_REG[TRIPLE_PWM_EN] bit.
The timing diagram of Timer 2 is shown in Figure 36

Freeze function

During RF activity it may be desirable to temporarily suppress the PWM switching noise. This can be done by setting TRIPLE_PWM_CTRL_REG[HW_PAUSE_EN] = 1. The effect is that whenever there is a transmission or a reception process from the Radio, T2_DUTY_CNTR is frozen and T2_PWMx output is switched to '0' to disable the selected T2_PWM1, T2_PWM2, T2_PWM3. As soon as the Radio is idle (i.e. RX_EN or TX_EN signals are zero), T2_DUTY_CNTR resumes counting and finalizes the remaining part of the PWM duty cycle.

TRIPLE_PWM_CTRL_REG[SW_PAUSE_EN] can be set to '0' to disable the automatic, hardware driven freeze function of the duty counter and keep the duty cycle constant.

Note that the RX_EN and TX_EN signals are not software driven but controlled by the BLE core hardware.

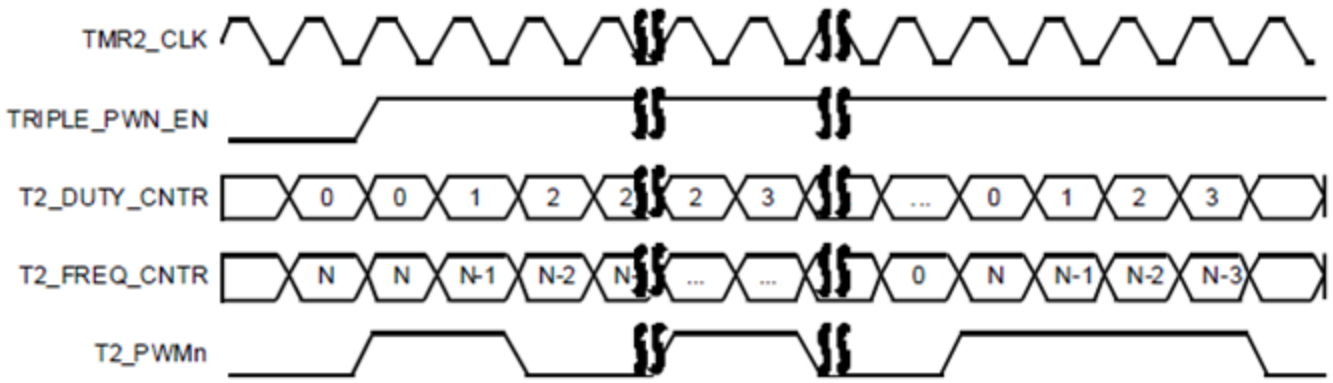


Figure 36 Timer 2 PWM Timing Diagram

9.12 Watchdog Timer

The Watchdog Timer is an 8-bit timer with sign bit that can be used to detect an unexpected execution sequence caused by a software run-away and can generate a full system reset or a Non- Maskable Interrupt (NMI).

Features:

- 8 bits down counter with sign bit, clocked with a 10.24ms clock for a maximum 2.6s time-out.
- Non-Maskable Interrupt (NMI) or WDOG reset.
- Optional automatic WDOG reset if NMI handler fails to update the Watchdog register.
- Non-maskable Watchdog freeze of the Cortex-M0 Debug module when the Cortex-M0 is halted in Debug state. Maskable Watchdog freeze by user program.

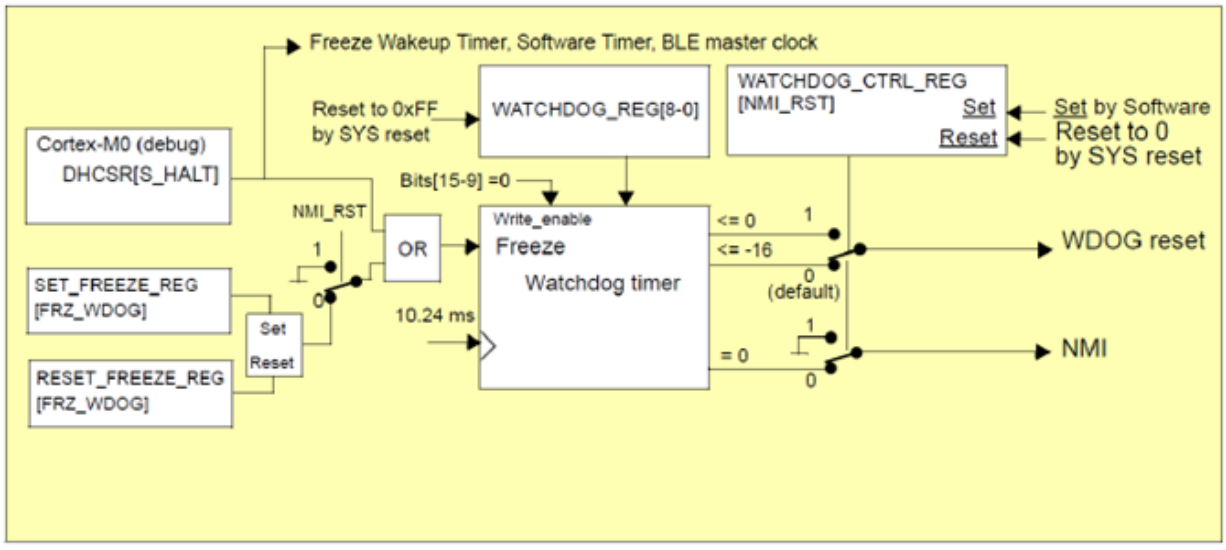


Figure 37 Watchdog Timer Block Diagram

The 8 bits watchdog timer is decremented by 1 every 10.24ms. The timer value can be accessed through the WATCHDOG_REG register which is set to 255 (FF16) at reset. This results in a maximum watchdog time-out of ~2.6s. During write access the WATCHDOG_REG[WDOG_WEN] bits must be 0. This provides extra filtering for a software run-away writing all ones to the WATCHDOG_REG. If the watchdog counter reaches 0, the counter value will get a negative value by setting bit 8. The counter sequence becomes 1, 0, 1FF16 (-1), 1FE16(-2),...1F016 (-16).

If WATCHDOG_CTRL_REG[NMI_RST] = 0, the watchdog timer will generate an NMI if the watchdog timer reaches 0 and a WDOG reset if the counter becomes less or equal to -16 (1F016). The NMI handler must write any value > -16 to the WATCHDOG_REG to prevent the generation of a WDOG reset at counter value -16 after 16*10.24 = 163.8ms.

If WATCHDOG_CTRL_REG[NMI_RST] = 1, the watchdog timer generates a WDOG reset if the timer becomes less or equal than 0.

The WDOG reset is one of the SYS (system) reset sources and resets the whole device, including setting the WATCHDOG_REG register to 255, except for the RST pin, the Power On reset, the HW reset and the DBG (debug module) reset. Since the HW reset is not triggered, the SYS_CTRL_REG[REMAP_ADR0] bits will retain their value and the Cortex-M0 will start executing again from the current selected memory at address zero. Refer to the POR, HW and SW Reset section for an overview of the complete reset circuit and conditions.

For debugging purposes, the Cortex-M0 Debug module can always freeze the watchdog by setting the DHCSR[DBGKEY | C_HALT | C_DEBUGEN] control bits (reflected by the status bit S_HALT). This is automatically done by the debug tool, e.g. during step-by-step debugging. Note that this bit also freezes the Wake-up Timer, the Software Timer and the BLE master clock. For additional information also see the DEBUG_REG[DEBUGS_FREEZE_EN] mask register. The C_DEBUGEN bit is not accessible by the user software to prevent freezing the watchdog.

In addition to the S_HALT bit, the watchdog timer can also be frozen if NMI_RST=0 and SET_FREEZE_REG[FRZ_WDOG] is set to ‘1’. The watchdog timer resumes counting when RESET_FREEZE_REG[FRZ_WDOG] is set to ‘1’. The WATCHDOG_CTRL_REG[NMI_RST] bit can only be set by software and will only be reset on a SYS reset. Note that if the system is not remapped, i.e. SysRAM is at address 0x20000000, then a watchdog fire will trigger the BootROM code to be executed again.

9.13 Input/Output Ports

The **ISM14585-L35** has software-configurable I/O pin assignment, organized into ports Port 0, Port 1.
.
Features:

- Port 0: 8 pins, Port 1: 4 pins
- Fully programmable pin assignment
- Selectable 25kΩ pull-up, pull-down resistors per pin
- Pull-up voltage is VBAT3V (BUCK mode)
- Fixed assignment for analog pin ADC[3:0]
- Pins can retain their last state when system enters the Extended or Deep Sleep mode.

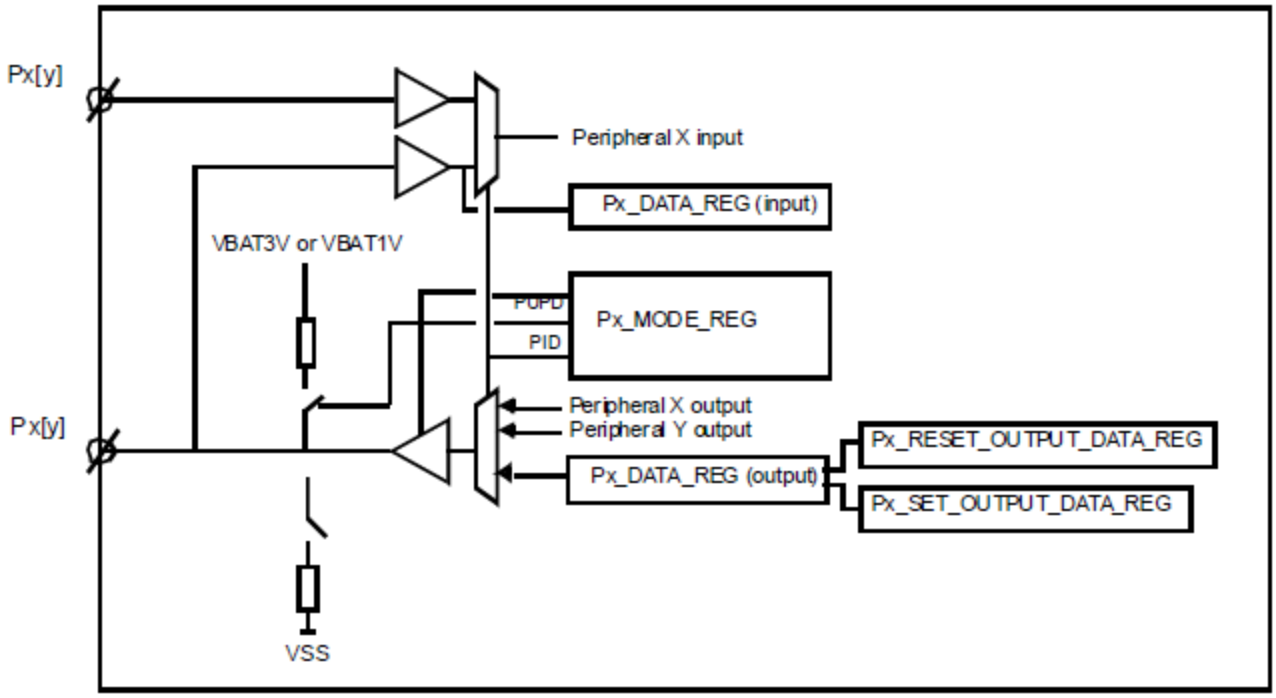


Figure 38 Port P0, and P1 with Programmable Pin Assignment

9.13.1 PROGRAMMABLE PIN ASSIGNMENT

The Programmable Pin Assignment (PPA) provides a multiplexing function to the I/O pins of on-chip peripherals. Any peripheral input or output signal can be freely mapped to any I/O port bit by setting Pxy_MODE_REG[4-0]:

0x00 to 0x1F: Peripheral IO ID (PID)

Refer to the Px_MODE_REGs for an overview of the available PIDs. Analog ADC has fixed pin assignment in order to limit interference with the digital domain. The SWD interface (JTAG) is mapped on P1_4 and P1_5.

Priority

The firmware has the possibility to assign the same peripheral output to more than one pin. It is the responsibility of the user to make a unique assignment.

In case more than one input signal is assigned to a peripheral input, the left most pin in the lowest port pin number has priority. (e.g P00_MODE_REG has priority over P01_MODE_REG).

Direction Control
The port direction is controlled by setting:

Pxy_MODE_REG[9-8]

00 = Input, no resistors selected
01 = Input, pull-up selected
10 = Input, Pull-down selected
11 = Output, no resistors selected

In output mode and analog mode, the pull-up/down resistors are automatically disabled.

9.13.2 GENERAL PURPOSE PORT REGISTERS

The general purpose ports are selected with PID=0. The port function is accessible through registers:

- Px_DATA_REG: Port data input/output register
- Px_SET_OUTPUT_DATA_REG: Port set output register
- Px_RESET_OUTPUT_DATA_REG: Port reset output register

9.13.2.1 PORT DATA REGISTER

The registers input Px_DATA_REG and output Px_DATA_REG are mapped on the same address. The data input register (Px_DATA_REG) is a read-only register that returns the current state on each port pin even if the output direction is selected, regardless of the programmed PID, unless the analog function is selected (in this case it reads 0). The ARM CPU can read this register at any time even when the pin is configured as an output.

The data output register (Px_DATA_REG) holds the data to be driven on the output port pins. In this configuration, writing to the register changes the output value.

9.13.2.2 PORT SET DATA OUTPUT REGISTER

Writing a 1 in the set data output register (Px_SET_OUTPUT_DATA_REG) sets the corresponding output pin. Writing a 0 is ignored.

9.13.2.3 PORT RESET DATA OUTPUT REGISTER

Writing a 1 in the reset data output register (Px_RESET_OUTPUT_DATA_REG) resets the corresponding output pin. Writing a 0 is ignored.

9.13.2.4 FIXED ASSIGNMENT FUNCTIONALITY

There are certain signals that have a fixed mapping on specific general purpose IOs. This assignment is illustrated in the following table:

GPIO	QUAD DEC (Note 1)	ADC (Note 2)
P0_0	CHY_A/CHX_A/CHZ_A	ADC_0
P0_1	CHY_B/CHX_B/CHZ_B	ADC_1
P0_2	CHY_A/CHX_A/CHZ_A	ADC_2
P0_3	CHY_B/CHX_B/CHZ_B	ADC_3
P0_4	CHY_A/CHX_A/CHZ_A	
P0_5	CHY_B/CHX_B/CHZ_B	
P0_6	CHY_A/CHX_A/CHZ_A	
P0_7	CHY_B/CHX_B/CHZ_B	
P1_0	CHY_A/CHX_A/CHZ_A	
P1_1	CHY_B/CHX_B/CHZ_B	
P1_2	CHY_A/CHX_A/CHZ_A	
P1_3	CHY_B/CHX_B/CHZ_B	

- Note 1 The mapping of the Quad Decoder signals on the respective pins, is overruled by the QDEC_CTRL2_REG[CHx_PORT_SEL] register.
- Note 2 The ADC case can be selected by the PID bit field on the respective Px port.

9.14 General Purpose ADC

The **ISM14585-L35** is equipped with a high-speed ultra-low power 10-bit general purpose Analog-to-Digital Converter (GPADC). It can operate in unipolar (single ended) mode as well as in bipolar (differential) mode. The ADC has its own voltage regulator (LDO) of 1.2 V, which represents the full scale reference voltage.

Features:

- 10-bit dynamic ADC with 65 ns conversion time
- Maximum sampling rate 3.3 Msample/s
- Ultra-low power (5µA typical supply current at 100ksample/s)
- Single-ended as well as differential input with two input scales
- Four single-ended or two differential external input channels
- Battery monitoring function
- Chopper function
- Offset and zero scale adjust
- Common-mode input level adjust

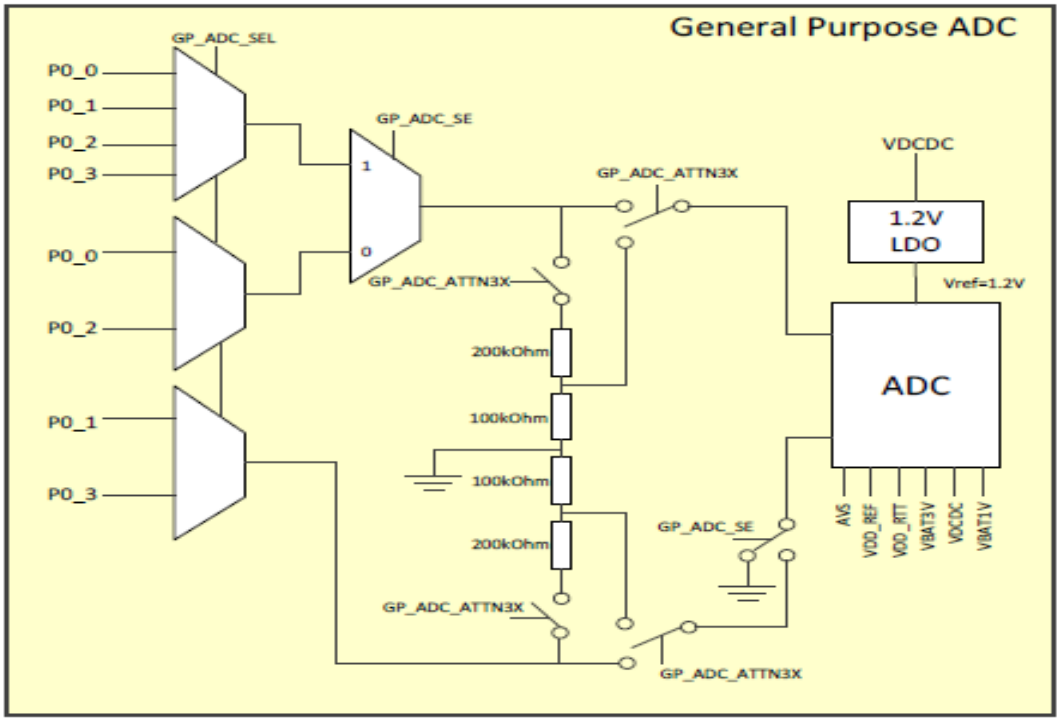


Figure 39 Block Diagram of the General Purpose ADC

9.14.1 INPUT CHANNELS AND INPUT SCALE

The **ISM14585-L35** has a multiplexer between the ADC and four specific GPIO ports (P0_0 to P0_3). Furthermore, the ADC can also be used to monitor the battery voltage and several internal voltages of the system (see GP_ADC_CTRL_REG).

Single-ended or differential operation is selected via bit GP_ADC_CTRL_REG[GP_ADC_SE]. In differential mode the voltage difference between two GPIO input ports will be converted. Via bit GP_ADC_CTRL2_REG[GP_ADC_ATTN3X] the input scale can be enlarged by a factor of three, as summarized in below table.

GP_ADC_ATTN3X	GP_ADC_SE	Input Channels	Input Scale	Input Limits
0	1	P0_0, P0_1, P0_2, P0_3	0 V to +1.2 V	-0.1 V to +1.3 V
0	0	[P0_0, P0_1], [P0_2, P0_3]	-1.2 V to +1.2 V	-1.3 V to +1.3 V
1	1	P0_0, P0_1, P0_2, P0_3	0 V to +3.6 V	-0.1 V to +3.45 V
1	0	[P0_0, P0_1], [P0_2, P0_3] -	-3.6 V to +3.6 V	-3.45 V to +3.45 V

GPADC Input Channels and Voltage Scale

9.14.2 STARTING THE ADC AND SAMPLING RATE

The GPADC is a dynamic ADC and consumes no static power, except for the LDO which consumes less than 5µA. Enabling/disabling of the ADC is triggered by configuring bit GP_ADC_CTRL_REG[GP_ADC_LDO_EN]. After enabling the LDO, a settling time of 20 µs is required before an AD-conversion can be started.

Each conversion has two phases: the sampling phase and the conversion phase. When bit GP_ADC_CTRL_REG[GP_ADC_EN] is set to ‘1’, the ADC continuously tracks (samples) the selected input voltage. Writing a '1' at bit GP_ADC_CTRL_REG[GP_ADC_START] ends the sampling phase and triggers the conversion phase. When the conversion is ready, the ADC resets bit GP_ADC_START to ‘0’ and returns to the sampling phase.

The conversion itself is fast and takes approximately one clock cycle of 16 MHz, though the data handling will require several additional clock cycles, depending on the software code style. The fastest code can handle the data in four clock cycles of 16 MHz, resulting to a highest sampling rate of 16 MHz/5 = 3.3 Msample/s.

At full speed the ADC consumes approximately 50µA. If the data rate is less than 100 ksample/s, the current consumption will be in the range of 5µA.

9.14.3 NON-IDEAL EFFECTS

Besides Differential Non-Linearity (DNL) and Integral Non-Linearity (INL), each ADC has a gain error (linear) and an offset error (linear). The gain error of the GPADC slightly reduces the effective input scale (up to 50 mV). The offset error causes the effective input scale to become non-centered. The offset error of the GPADC is less than 20 mV and can be reduced by chopping or by offset calibration.

The ADC result will also include some noise. If the input signal itself is noise free (inductive effects included), the average noise level will be ±1 LSB. Taking more samples and calculating the average value will reduce the noise and increase the resolution.

With a 'perfect' input signal (e.g. if a filter capacitor is placed close to the input pin) most of the noise comes from the low-power voltage regulator (LDO) of the ADC. Since the DA14585 is targeted for ultra-compact applications, there is no pin available to add a capacitor at this voltage regulator output.

The dynamic current of the ADC causes extra noise at the regulator output. This noise can be reduced by setting bits GP_ADC_CTRL2_REG[GP_ADC_I20U] and GP_ADC_CTRL2_REG[GP_ADC_IDYN] to ‘1’. Bit GP_ADC_I20U enables a constant 20µA load current at the regulator output so that the current will not drop to zero. Bit GP_ADC_IDYN enables a 10µA load current during sampling phase so that the load current during sampling and conversion phase becomes approximately the same.

9.14.4 CHOPPING

Chopping is a technique to cancel offset by taking two samples with opposite signal polarity. This method also smooths out other non-ideal effects and is recommended for DC and slowly changing signals.

Chopping is enabled by setting bit GP_ADC_CTRL_REG[GP_ADC_CHOP] to ‘1’.

The mid-scale value of the ADC is the 'natural' zero point of the ADC (ADC result = 511.5 = 1FF or 200 Hex = 01.1111.1111 or 10.0000.0000 Bin). Ideally this corresponds to Vi = 1.2 V/2 = 0.6 V in single-ended mode and Vi = 0.0 V in differential mode.

If bit GP_ADC_CTRL2_REG[GP_ADC_ATTN3X] is set to ‘1’, the zero point is 3 times higher (1.8 V single-ended and 0.0 V differential).

With bit GP_ADC_CTRL_REG[GP_ADC_MUTE], the ADC input is switched to the center scale input level, so the ADC result ideally is 511.5. If instead a value of 515 is observed, the output offset is +3.5 (adc_off_p = 3.5).

With bit GP_ADC_CTRL_REG[GP_ADC_SIGN] the sign of the ADC input and output is changed. Two sign changes have no effect on the signal path, though the sign of the ADC offset will change.

If adc_off_p = 3.5 the ADC_result with opposite GP_ADC_SIGN will be 508. The sum of these equals 515 + 508 = 1023. This is the mid-scale value of an 11-bit ADC, so one extra bit due to the over-sampling by a factor of two.

The LSB of this 11-bit word should be ignored if a 10-bit word is preferred. In that case the result is 511.5, so the actual output value will be 511 or 512.

9.14.5 OFFSET CALIBRATION

A relative high offset caused by a very small dynamic comparator (up to 20 mV, so approximately 20 LSB). This offset can be cancelled with the chopping function, but it still causes unwanted saturation effects at zero scale or full scale. With the GP_ADC_OFFP and GP_ADC_OFFN registers the offset can be compensated in the ADC network itself.

To calibrate the ADC follow the steps in table below:

GPADC Calibration Procedure for Single-Ended and Differential Modes

Step	Single-Ended Mode (GP_ADC_SE = 1)	Differential Mode (GP_ADC_SE = 0)
1	Set GP_ADC_OFFP = GP_ADC_OFFN=0x200; GP_ADC_MUTE = 0x1; GP_ADC_SIGN = 0x0	Set GP_ADC_OFFP=GP_ADC_OFFN = 0x200; GP_ADC_MUTE = 0x1; GP_ADC_SIGN = 0x0
2	Start conversion	Start conversion
3	adc_off_p = GP_ADC_RESULT - 0x200	adc_off_p = GP_ADC_RESULT - 0x200
4	Set GP_ADC_SIGN = 0x1	Set GP_ADC_SIGN = 0x1
5	Start conversion	Start conversion
6	adc_off_n = GP_ADC_RESULT - 0x200	adc_off_n = GP_ADC_RESULT - 0x200
7	GP_ADC_OFFP = 0x200 - 2*adc_off_p GP_ADC_OFFN = 0x200 - 2*adc_off_n (Note 4)	GP_ADC_OFFP = 0x200 - adc_off_p GP_ADC_OFFN = 0x200 - adc_off_n (Note 4)

Note 4 The average of GP_ADC_OFFP and GP_ADC_OFFN should be 0x200 (with a margin of 20 LSB). It is recommended to implement the above calibration routine during the initialization phase of the **ISM14585-L35**. To verify the calibration results, check whether the GP_ADC_RESULT value is close to 0x200 while bit GP_ADC_MUTE = 1.

9.14.6 ZERO-SCALE ADJUSTMENT

The GP_ADC_OFFP and GP_ADC_OFFN registers can also be used to set the zero-scale or full-scale input level at a certain target value. For instance, they can be used to calibrate GP_ADC_RESULT to 0x000 at an input voltage of exactly 0.0 V, or to calibrate the zero scale of a sensor.

9.14.7 COMMON MODE ADJUSTMENT

The common mode level of the differential signal must be 0.6 V (or 1.8 V with GP_ADC_ATTN3X = 1). If the common mode input level of 0.6 V cannot be achieved, the common mode level of the GP_ADC can be adjusted (the GP_ADC can tolerate a common mode margin up to 50 mV) according to below table.

CM Voltage (Vcmm)	GP_ADC_OFFP = GP_ADC_OFFN
0.3 V	0x300
0.6 V	0x200
0.9 V	0x100

Common Mode Adjustment

Any other common mode level between 0.0 V and 1.2 V can be calculated from the table above. Offset calibration can be combined with common mode adjustment by replacing the 0x200 value in the offset calibration routine by the value required to get the appropriate common mode level.

Note: The input voltage limits for the ADC in differential modes are: -1.3 V to +1.3 V (for GP_ADC_ATTN3X = 0, see Table for GPADC Input Channels and Voltage Scale). The differential input range of the ADC is: -1.2 V < V[P0_0,P0_1] < +1.2 V. Therefore, if Vcmm < 0.5 V or Vcmm > 0.7 V, the input can no longer cover the whole ADC range.

9.14.8 INPUT IMPEDANCE, INDUCTANCE, AND INPUT SETTLING

The GPADC has no input buffer stage. During sampling phase a capacitor of 0.2 pF is switched to the input line. The precharge of this capacitor is at mid-scale level so the input impedance is infinite. At 100 ksample/s, zero or full-scale single-ended input signal, this sampling capacitor will load the input with:
ILOAD = V * C * fS = ±0.6 V * 0.2 pF * 100 kHz = ±12 nA (differential: ±1.2 V * 0.2 pF * 100 kHz = ±24 nA at both pins).

During sampling phase a certain settling time is required. A 10-bit accuracy requires at least 7 time constants of the output impedance of the input signal source and the 0.2 pF

sampling capacitor. The conversion time is approximately one clock cycle of 16 MHz (62.5 ns).

7 * ROUT * 0.2 pF - 62.5 ns < 1/fS
=> ROUT < (1 + 62.5 ns * fS) / (7 * 0.2 pF * fS)

Examples:
ROUT < 7.2 MΩ at fS = 100 kHz
ROUT < 760 kΩ at fS = 1 MHz

The inductance from the signal source to the ADC input pin must be very small. Otherwise, filter capacitors are required from the input pins to ground (differential mode: from pin to pin). To observe the noise level of the ADC and the voltage regulator, bit GP_ADC_CTRL_REG[GP_ADC_MUTE] must be set to '1'. The noise should be less than ±1 LSB on average, with occasionally a ±2 LSB peak value. If a higher noise level is observed on the input channel(s), applying filter capacitor(s) will reduce the noise.

The 3x input attenuator is realized with a resistor divider network. When bit GP_ADC_CTRL_REG2[GP_ADC_ATTN3X] is set to '1', the input impedance of the selected ADC input channel becomes 300 kΩ (typical) instead of infinite. In addition, the resistor divider network will require more settling time in the sampling phase. The general guideline with bit GP_ADC_ATTN3X = 1 is: select the input channel, then wait 1 µs (16 clock cycles) before starting the conversion. Only the required sampling time is affected by the attenuator, the conversion time remains approximately one clock cycle of 16 MHz (62.5 ns).

Note: Selecting the battery measurement channel automatically activates the 3x input attenuator (bit GP_ADC_ATTN3X = 1). Therefore the 1 µs waiting time also applies when measuring the battery voltage, otherwise the resulting Vbat level will be too low.

9.14.9 COMMON MODE ADJUSTMENT

The GPADC has a delay counter that can be used to add delays to several ADC control signals. A delay of up to 32µs can be added for the bits GP_ADC_LDO_EN, GP_ADC_START and GP_ADC_EN via registers GP_ADC_DELAY_REG and GP_ADC_DELAY2_REG.

The reset values of these two registers are the recommended values for a correct start-up, since it is not allowed to activate all signals at once.

To make use of the delay counter for a certain signal, the corresponding bit has to be set in register GP_ADC_CTRL_REG and the delay counter must be enabled via bit

GP_ADC_DELAY_EN in register GP_ADC_CTRL2_REG. The delay counter starts counting when the GP_ADC_START bit is programmed while the GP_ADC_DELAY_EN bit is set. The counter is stopped after the conversion is finished.

The delay counter must be reset before reuse, which is typically only required after the LDO was disabled. Bit GP_ADC_DELAY_EN must be made zero to reset the counter. It is recommended to check that this bit is zero before (re)activating it.

9.14.10 Sample Rate Converter (SRC)

The SRC is a HW accelerator used to convert the sample rate of audio samples between various interfaces. Its primary purpose is to directly connect PCM and PDM channels while converting the rate accordingly.

Features:

- Supported conversions:
  - SRC_IN (24 bits) to SRC_OUT (24 bits)
  - PDM_IN (1bit) to SRC_OUT (24 bits)
  - SRC_IN (24 bits) to PDM_OUT (1 bit)
- SRC_IN, SRC_OUT Sample rates 8 kHz to 192 kHz
- SNR > 100 dB
- Single Buffer I/O with DMA support
- Automatic mode to adjust sample rate to the applied frame sync (e.g. PCM_FSC)
- Manual mode to generate interrupts at the programmed sample rate. Adjustment is done by SW based on buffer pointers
- SRC runs at 16 MHz

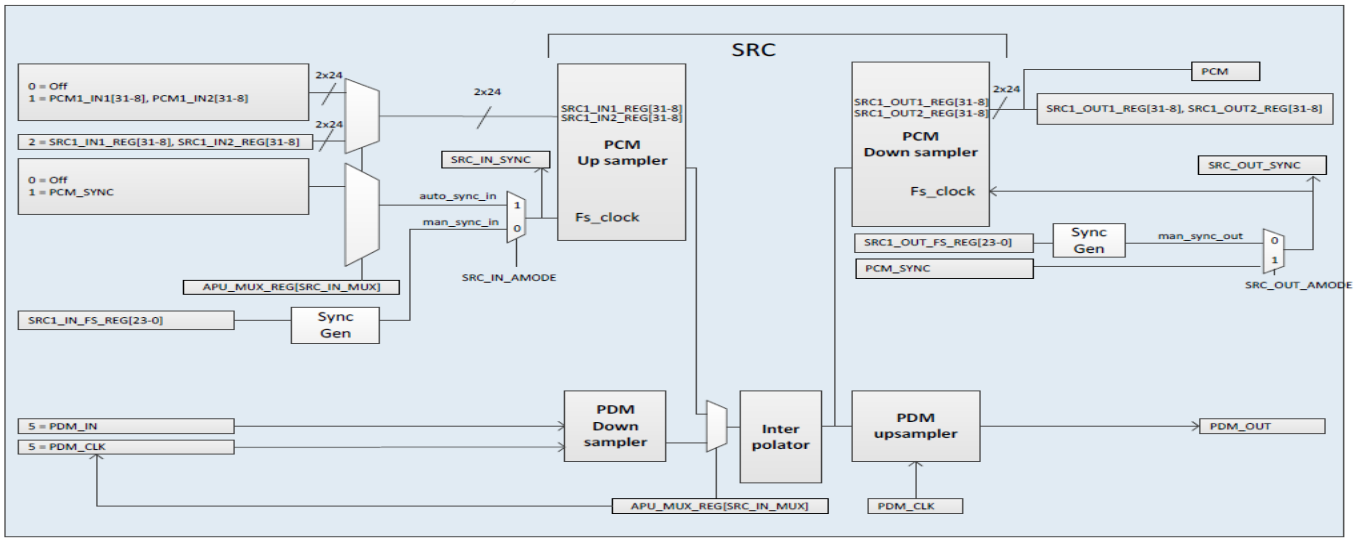


Figure 40 Sample Rate Converter block diagram

9.14.11 SRC ARCHITECTURE

9.14.11.1 I/O CHANNELS

The SRC block converts two 24 bits channels either as a stereo pair or as two mono channels. The PCM linear data pairs are received on SRC_IN and the output is 2x24 bits left aligned on SRC_OUT. The two 1 bit PDM data inputs are received on PDM_IN and are converted to 2x24 bits, left aligned to SRC_OUT.

9.14.11.2 I/O MULTIPLEXERS

The SRCx_IN input multiplexer (Figure 40) is controlled by APU_MUX_REG. The input of these multiplexers either comes from the audio interfaces of from registers SRC1_IN1_REG and SRC1_IN2_REG. The data to these register is left aligned, bits 31-8 are mapped on bits 23-0 of the SRC.

The 24 bits SRCs outputs can be read in SRC1_OUT1_REG and SRC1_OUT2_REG and is also routed to the PCM. This input selection of these multiplexers is also controlled by APU_MUX_REG.

9.14.11.3 INPUT AND OUTPUT SAMPLE RATE CONVERSION

Depending on the use case the Sample Rate Converter operate in either manual or automatic conversion mode. This mode can be set in the SRC1_CTRL_REG bits SRC_IN_AMODE and bit SRC_OUT_AMODE.

9.14.11.4 SRC CONVERSION MODES OF OPERATION

Both at the input and at the output side, the SRC can operate in two mode of operation:

- Manual mode
- Automatic mode

In manual mode, the input/output sample rate is determined by SRC1_IN_FS_REG /SRC1_OUT_FS_REG registers.

In automatic mode, the input/output sample rate is derived from the external synchronization signals and can be read back from SRC1_IN_FS_REG / SRC1_OUT_FS_REG registers. When the PDM is used (input/output), the SRC operates in automatic mode. The sample rate reported in SRC1_IN_FS_REG register is PDM_CLK/64. Typical Use Cases of the SRC are given on the table below:.

9.14.11.5 DMA OPERATION

If more than one sample must be transfer to/from the CPU or the sample rate is so high that it interrupts the CPU too often, the DMA controller must be engaged to perform the transactions.

9.14.11.6 INTERRUPTS

After a Sample Rate Conversion, the input up-sampler and output down-sampler generate edge triggered interrupts on SRC_IN_SYNC and SRC_OUT_SYNC to the CPU which do not have to be cleared. Note that only one sample shall be read from or written to a single register at a time (i.e. there are no FIFOs included).

9.14.11.7 SRC USE CASES

The Sample Rate Converter supports any sample rate between 8 kHz and 48 kHz. Below table shows typical use cases of the Sample Rate Converter.

Use case	SRCx_IN_AMODE DATA path SRCx_IN_SYNC	SRCx_IN_SYNC (out)	SRCx_OUT_AMODE DATA path SRCx_OUT_SYNC	SRCx_OUT_SYNC (out)
PCM/PDM to BLE	Automatic PDMx_IN_DATA PCMx_SYNC PCMx_IN_DATA PCMx_SYNC		Manual SRCx_OUT_REG	up-to 192kHz
BLE to PCM / PDM	Manual SRCx_IN_REG	up-to 192kHz	Automatic PDMx_IN_DATA PCMx_SYNC PCMx_IN_DATA PCMx_SYNC	
PCM to PCM resampler (output must be a multiple of 8 kHz)				
PCM1_IN to PCM2_OUT	Automatic PCM1_IN PCM1_FSC (input)		Automatic PCM2_OUT PCM2_FSC (output)	
PCM2_IN to PCM1_OUT	Automatic PCM2_IN PCM2_FSC (input)		Automatic PCM1_IN PCM1_FSC (output)	

Table of Typical SRC use case

9.15 PDM Interface

The Pulse Density Modulation (PDM) interface provides a serial connection for up-to 2 input devices (e.g MEMS microphones) or output devices. The interfaces have a common clock PDM_CLK and one input PDM_DI which is capable of carrying two channels. Figure 41 shows a typical connection of two microphones sharing one data line.

The PDM input data has a 1-bit data length and is encoded so that the left channel is clocked in on the falling edge of PDM_CLK and the right channel is clocked on the rising edge of PDM_CLK as shown in Figure 42.

The 1 bit data stream is down-sampled to 24 bits PCM samples in the HW Sample Rate Converter (SRC) for further processing in the DSP.

The interface supports MEMS microphone sleep mode by disabling the PDM_CLK.

The PDM interface signals are available through the PPA multiplexer. The interface levels are determined by the IO group on which the PDM signals are mapped. Those signals can be hard wired to 1.8 V and 3.3 V.

Features:

- PDM_CLK frequency 62.5 kHz - 4 MHz
- Down-sampling to 24 bits in SRC
- PDM_CLK on/off to support Sleep mode
- PDM_DATA (input): 1 Channel in stereo format
- PDM_DATA (output): 2 Channels in mono format, 1 Channel in stereo format
- Programmable Left/Right channel selection

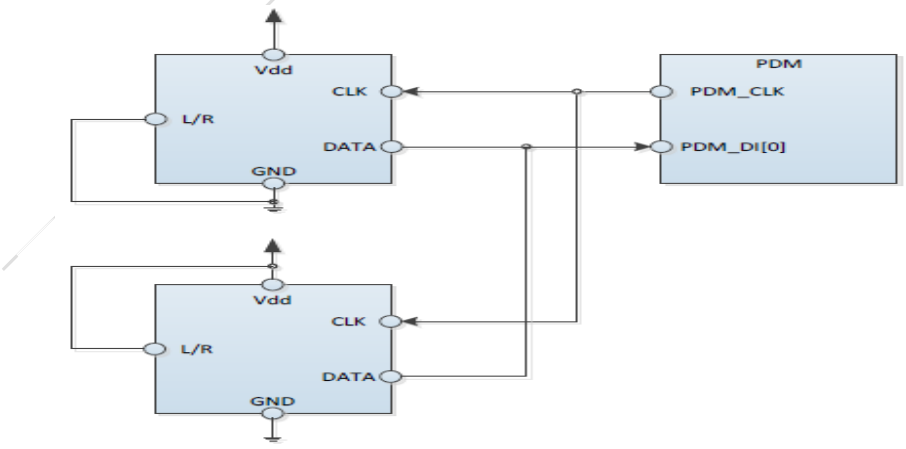


Figure 41 PDM with dual mic interface

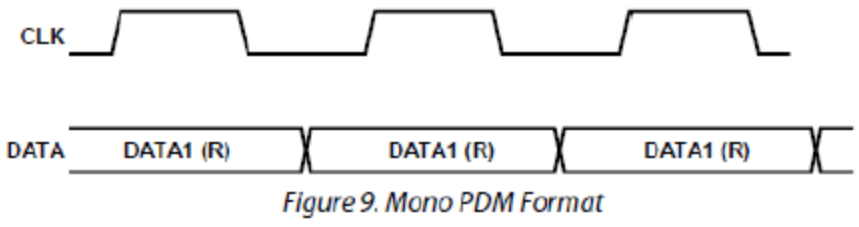


Figure 9. Mono PDM Format

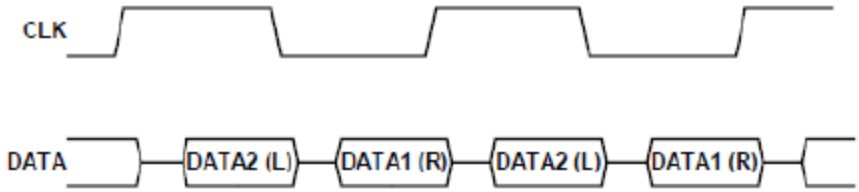


Figure 42 PDM formats

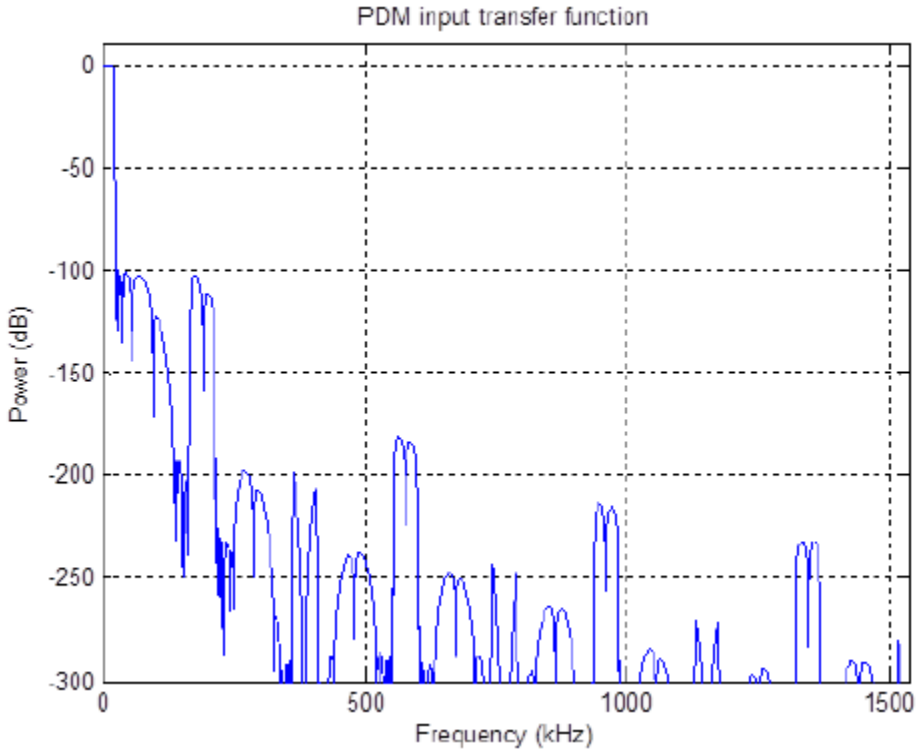


Figure 43 PDM input transfer functions

It should be noted that the audio quality degrades when the oversampling ratio is less than 64. For an 8 kHz sample rate the minimum recommended PDM clock rate is 64 x 8 kHz = 512 kHz.

9.16 PCM Controller

The PCM controller is implementing an up-to 192kHz synchronous interface to external audio devices, ISDN circuits and serial data interfaces.

It is accessed through the APB32 interface. PCM can individually operate in master or slave mode. In slave mode, the phase between the external and internal frame sync can be measured and used to compensate for drift.

The data IO registers have DMA support in order to reduce the interrupt overhead to the CPU. Up-to 8 channels of 8 bits with a programmable delay are supported in received and transmit direction.

The controller supports PCM, I2S, TDM and IOM2 formats.

Features:

- PCM_CLK Master/slave
- PCM_FSC
  - Master/slave 4 kHz to 96 kHz
  - Strobe Length 1, 8, 16, 24, 32, 40, 48 and 64 bits
  - PCM_FSC before or on the first bit. (In Master mode)
- 2x32 channels
- Programmable slot delay up-to 31*8bits
- Formats
  - PCM mode
  - I2S mode (Left/Right channel selection) with N*8 for Left and N*8 for Right
  - IOM2 mode (double clock per bit)
- Programmable clock and frame sync inversion
- Direct connection to Sample Rate Converter (SRC)
- Interrupt line to the CPU
- DMA support

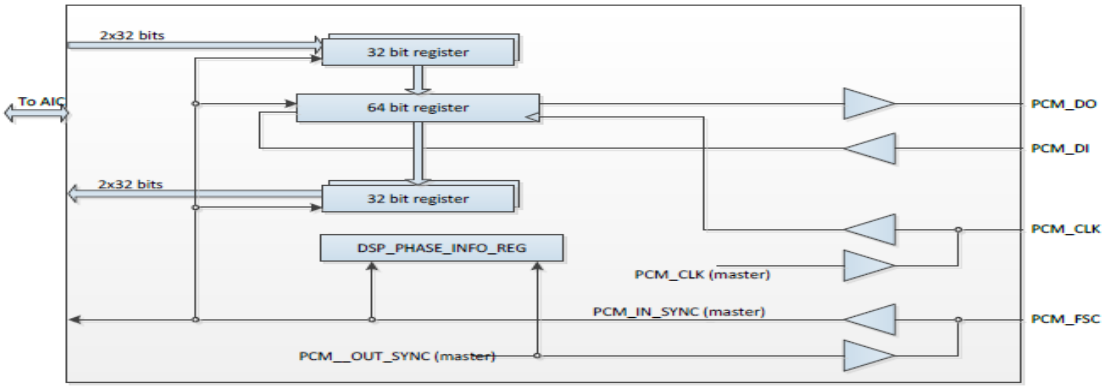


Figure 44 PCM controller

9.16.1 PCM ARCHITECTURE

9.16.1.1 INTERFACE SIGNALS

- PCM_FSC, strobe signal input, output. Supports 8/16/32/48/96/128/192kHz. Can generate an interrupt to the CPU.
- PCM_CLK, PCM clock input, output.
- PCM_DO, PCM Data output, push pull or open drain with external pull-up resistor.
- PCM_DI, PCM Data input.

PCM interface can be powered down by the PCM1_CTRL_REG[PCM_EN] = 0.

9.16.1.2 CHANNEL ACCESS

The PCM interface has two 32-bit channels for TX and RX. Channels are accessed through 32 bits registers:

- PCM1_OUT1_REG and PCM1_OUT2_REG,
- PCM1_IN1_REG and PCM1_IN2_REG.

The registers are only word-wise (32 bits) accessible by the CPU or the DMA via the APB-32 bridge

.

The 32 bits registers are arranged as 8 channels of 8 bits, named channel 1 to channel8.

By a flexible clock inversion, channel delay and strobe length adjustment various format like PCM, I2S, TDM and IOM2 can be made

9.16.1.3 CHANNEL DELAY

The 8 PCM channels can be delayed with a maximum delay of 31x8bits using the bit field PCM1_CTRL_REG[PCM_CH_DEL]. Note that a high delay count in combination with a slow clock, can lead to the PCM_FSC sync occurring before all channels are shifted in or out. The received bits of the current channel may not be properly aligned in that case.

9.16.1.4 CLOCK GENERATION

Figure 45 shows the PCM clock generation block and the Table below, the PCM_DIV_REG and PCM_FDIV_REG values for given PCM_FSC and PCM_CLK in master mode. The PCM_DIV_REG[PCM_DIV] is a 12 bits field which holds the integer part of the desired clock divider.

The fractional part of the divider is stored in the 16 bits PCM_FDIV_REG register. The value of the register is calculated in the following way. The position of the left most '1' of the value in binary format defines the denominator and the number of '1' bits define the numerator of the fraction as shown in the Table below.

PCM_FDIV_REG (Hex)	PCM_FDIV_REG (Binary)	Numerator	Denominator	Fraction
0x0110	0b100010000	2	9	2/9
0x0101	0b100000001	2	9	2/9
0x1abc	0b1101010111100	8	13	8/13
0xbeef	0b1011111011101111	13	16	13/16
0xfeee	0b1111111011101110	13	16	13/16

PCM_FDIV_REG Example Calculations

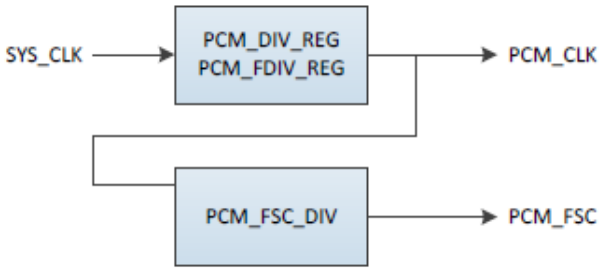


Figure 45 PCM clock generation

The PCM_DIV_REG calculations show the following use cases, with 8 bits, 16 bits, 32 bits and 48 bits.

			XTAL 16000 kHz			Alternatively
Sample Rate	**Bits**	**Desired Bit Clock kHz**	**Desired Divider**			**Actual Divider**
8	**1*8**	64	**250**	**250**	**Actual Word size**	**Divider: Integer and Fractional Part**
8	1*16	128	125	125	8	=250
8	1*24	192	83,33333333	80	16	=125
8	1*32	256	62,5	50	25	=83 + 1/3
8	**2*8**	128	125	125	40	=62 + 1/2
8	2*16	256	62,5	50	8	=125
8	2*24	384	41,66666667	40	20	=62 + 1/2
8	2*32	512	31,25	25	25	=41 + 2/3
16	**1*8**	128	125	125	40	=31 + 1/4
16	1*16	256	62,5	50	8	=125
16	1*24	384	41,66666667	40	20	=62 + 1/2
16	1*32	512	31,25	25	25	=41 + 2/3
16	**2*8**	256	62,5	50	40	=31 + 1/4
16	2*16	512	31,25	25	10	=62 + 1/2
16	2*24	768	20,83333333	20	20	=31 + 1/4
16	2*32	1024	15,625	10	25	=20 + 5/6
32	**1*8**	256	62,5	50	50	=15 + 5/8
32	1*16	512	31,25	25	10	=62 + 1/2
32	1*24	768	20,83333333	20	20	=31 + 1/4
32	1*32	1024	15,625	10	25	=20 + 5/6
32	**2*8**	512	31,25	25	50	=15 + 5/8
32	2*16	1024	15,625	10	10	=31 + 1/4
32	2*24	1536	10,41666667	10	25	=15 + 5/8
32	2*32	2048	7,8125	5	25	=10 + 5/12
48	**1*8**	384	41,66666667	N/A	50	=7 + 13/16
48	1*16	768	20,83333333	N/A	N/A	=41 + 2/3
48	1*24	1152	13,88888889	N/A	N/A	=20 + 5/6
48	1*32	1536	10,41666667	N/A	N/A	=13 + 8/9
48	**2*8**	768	20,83333333	N/A	N/A	=10 + 5/12
48	2*16	1536	10,41666667	N/A	N/A	=20 + 5/6
48	2*24	2304	6,944444444	N/A	N/A	=10 + 5/12
48	2*32	3072	5,208333333	N/A	N/A	N/A

Integer PCM_DIV_REG values for given frequencies and sample rates

9.16.1.5 DATA FORMATS

9.16.1.5.1 PCM MASTER MODE

Master mode is selected if PCM1_CTRL_REG[PCM_MASTER] = 1.

In master mode PCM_FSC is output and falls always over Channel 0. The duration of PCM_FSC is programmable with PCM1_CTRL_REG[PCM_FSCLEN]= 1 or 8,16, 24, 32 clock pulses high. The start position is programmable with PCM1_CTRL_REG[PCM_FSCDEL] and can be placed before or on the first bit of channel 0.

The repetition frequency of PCM_FSC is programmable in PCM1_CTRL_REG[PCM_FSC_DIV] to from 8-192kHz.
If master mode selected, PCM_CLK is output and provides one or two clocks per data bit programmable in PCM1_CTRL_REG[PCM_CLK_BIT].

The polarity of the signal can be inverted with bit PCM1_CTRL_REG[PCM_CLKINV].

The PCM_CLK frequency selection is described in Section 9.14.1.4.

9.16.1.5.2 PCM SLAVE MODE

In slave mode (bit MASTER = 0) PCM_FSC is input and determines the starting point of channel 0.

The repetition rate of PCM_FSC must be equal to PCM_SYNC and must be high for at least one PCM_CLK cycle. Within one frame, PCM_FSC must be low for at least PCM_CLK cycle. Bit PCM_FSCDEL sets the start position of PCM_FSC before or on the first bit (MSB).

In slave mode PCM_CLK is input. The minimum received frequency is 256 kHz, the maximum is 12.288MHz.

In slave mode the main counter can be stopped and resumed on a PCM1_FSC or PCM2_FSC rising edge.

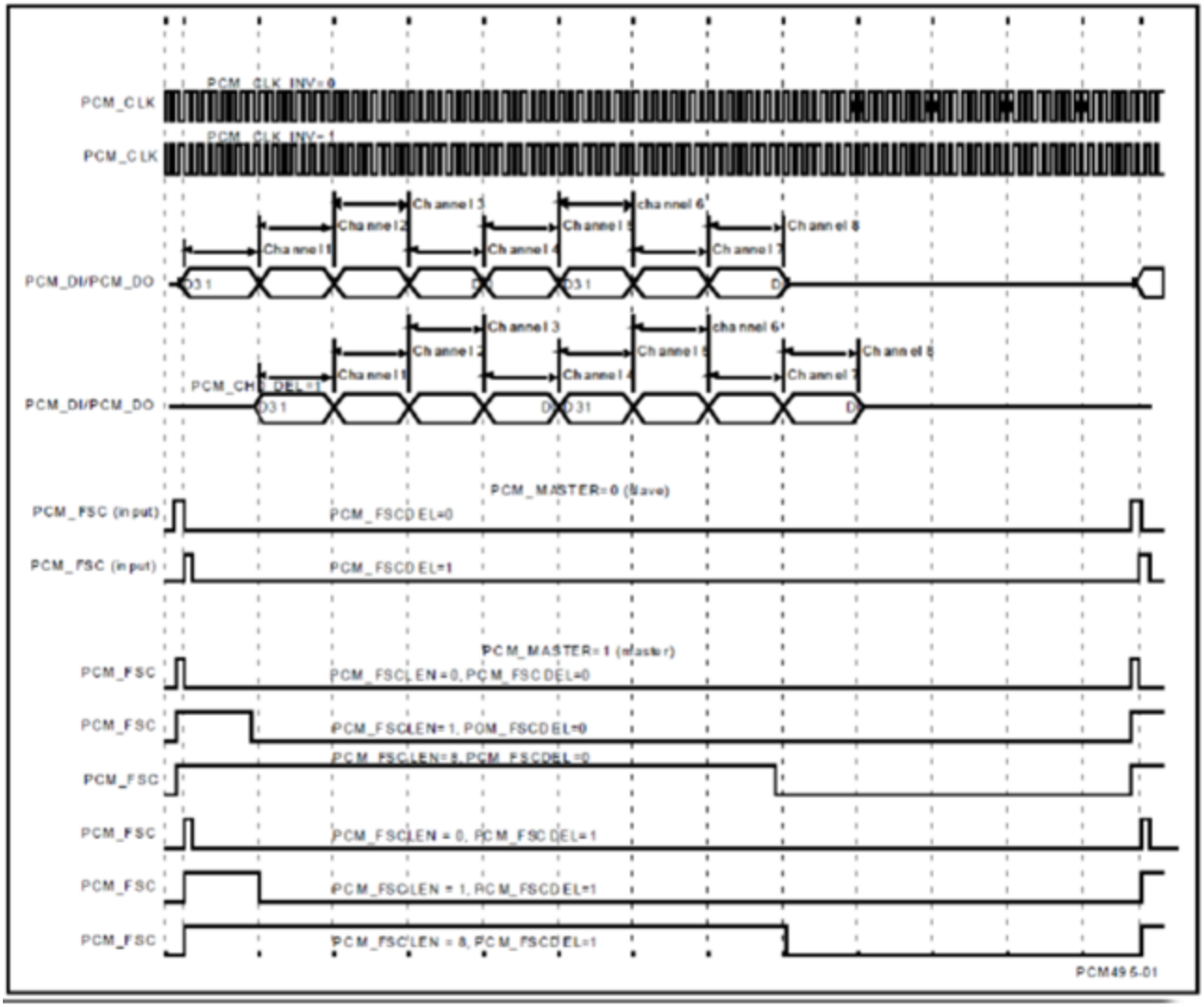


Figure 46 PCM interface formats

9.16.1.5.3 I2S FORMATS

The digital audio interface supports I2S mode, Left Justified mode, Right Justified mode and TDM mode.

I2S mode

To support I2S mode, the MSB of the right channel is valid on the second rising edge of the bit clock after the rising edge of the PCM_FSC, and the MSB of the left channel is valid on the second rising edge of the bit clock after the falling edge of the PCM_FSC.

Settings for I2S mode:

- PCM_FSC_EDGE: 1 (all after PCM_FSC)
- PCM_FSCLEN: 4 (4x8 High, 4x8 Low)
- PCM_FSC_DEL: 0 (one bit delayed)
- PCM_CLK_INV: 1 (output on falling edge)
- PCM_CH0_DEL: 0 (no channel delay)

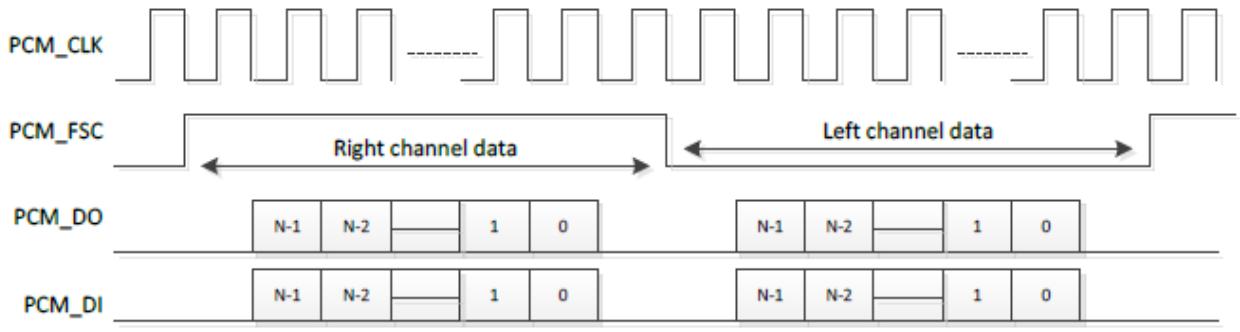


Figure 47 I2S Mode

TDM mode

A time is specified from the normal ‘start of frame’ condition using register bits PCM_CH_DEL. In the left-justified TDM example illustrated in Figure 48, the left channel data is valid PCM_CH_DEL clock cycles after the rising edge of the PCM_FSC, and the right channel data is valid the same PCM_CH_DEL number of clock cycles after the falling edge of the PCM_FSC.

By delaying the channels, also left and right alignment can be achieved.

Settings for TDM mode:

- PCM_FSC_EDGE: 1 (rising and falling PCM_FSC)
- PCM_FSCLEN: Master 1 to 4 Slave waiting for edge.
- PCM_FSC_DEL: 1 (no bit delay)
- PCM_CLK_INV: 1 (output on falling edge)
- PCM_CH0_DEL: Slave 0-31 (channel delay) Master 1-3

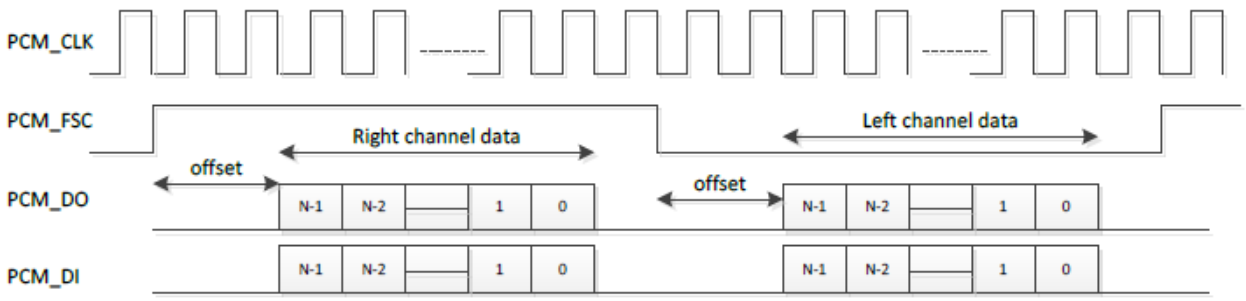


Figure 48 I2S TDM mode (left justified mode)

9.16.1.5.4 IOM MODE

In the IOM format, the PCM_CLK frequency is twice the data bit cell duration. In slave mode synchronization is on the first rising edge of PCM_FSC while data is clock in on the second falling edge.

Settings for IOM mode:

- PCM_FSC_EDGE: 0 (rising edge PCM_FSC)
- PCM_FSCLEN: 0 (one cycle)
- PCM_FSC_DEL: 0 (no bit delay)
- PCM_CLK_INV: 0 (output on rising edge)
- PCM_CH0_DEL: 0 (no delay)
- PCM_CLK_BIT: 1

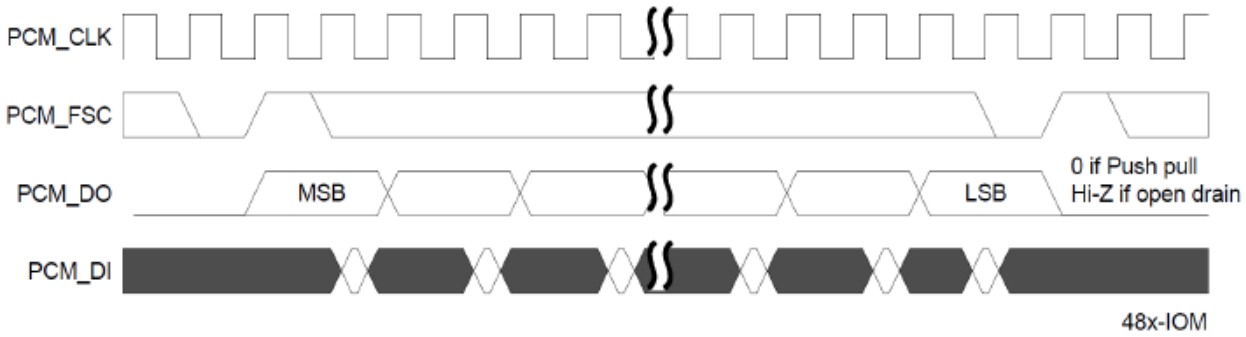


Figure 49 IOM format

9.16.1.6 EXTERNAL SYNCHRONIZATION

With the PCM interface in slave mode, the PCM interface supports direct routing through the sample rate converter (SRC). Any drift in PCM_FSC or other frame sync frequencies like 44.1 kHz can be directly resampled to e.g 48kHz internal sample rate.

10 MECHANICAL SPECIFICATION

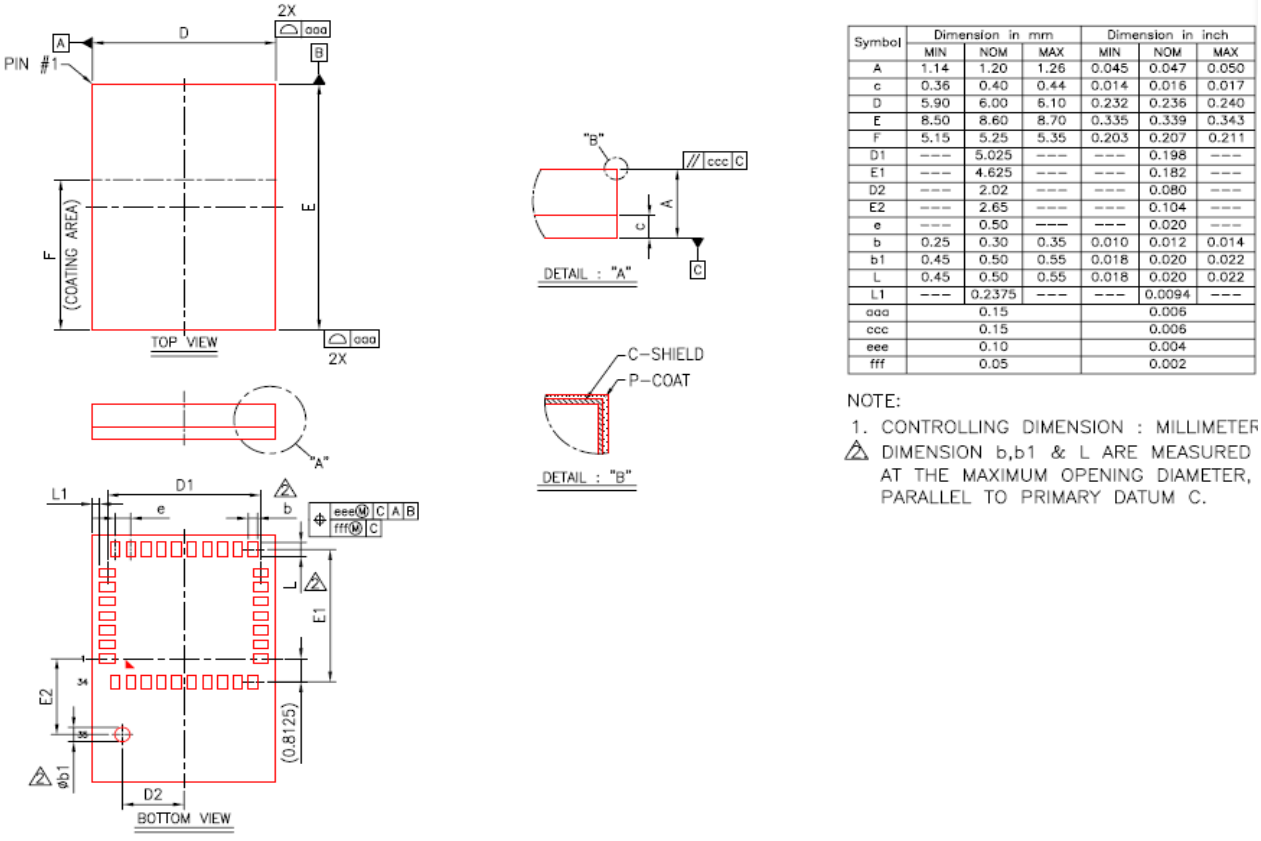
10.1 Size of the Module

The following paragraphs provide the requirements for the size and weight.
The size and thickness of the **ISM14585-L35** SiP is:

- 6mm (W) x 8.6mm (L) x 1.2mm (H):
- (Tolerance: +/- 0.1mm)

10.2 Mechanical Dimension

Dimension: 6 x 8.6 x 1.2 mm³



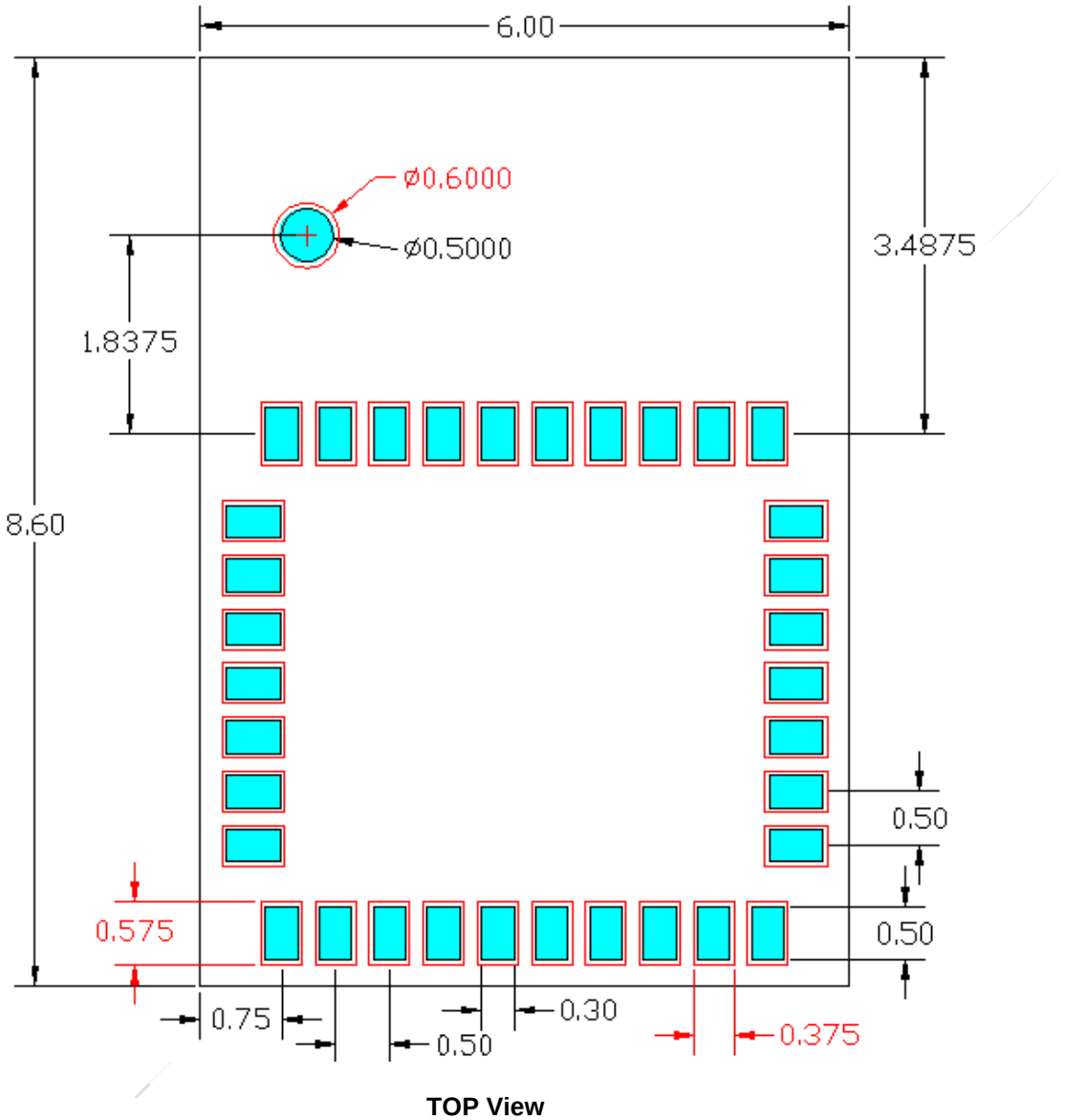
Symbol	Dimension in mm			Dimension in inch		
	MIN	NOM	MAX	MIN	NOM	MAX
A	1.14	1.20	1.26	0.045	0.047	0.050
c	0.36	0.40	0.44	0.014	0.016	0.017
D	5.90	6.00	6.10	0.232	0.236	0.240
E	8.50	8.60	8.70	0.335	0.339	0.343
F	5.15	5.25	5.35	0.203	0.207	0.211
D1	---	5.025	---	---	0.198	---
E1	---	4.625	---	---	0.182	---
D2	---	2.02	---	---	0.080	---
E2	---	2.65	---	---	0.104	---
e	---	0.50	---	---	0.020	---
b	0.25	0.30	0.35	0.010	0.012	0.014
b1	0.45	0.50	0.55	0.018	0.020	0.022
L	0.45	0.50	0.55	0.018	0.020	0.022
L1	---	0.2375	---	---	0.0094	---
aaa		0.15			0.006	
ccc		0.15			0.006	
eee		0.10			0.004	
fff		0.05			0.002	

NOTE:
1. CONTROLLING DIMENSION : MILLIMETER
2. DIMENSION b,b1 & L ARE MEASURED AT THE MAXIMUM OPENING DIAMETER, PARALLEL TO PRIMARY DATUM C.

11 Recommend Footprint (Board Design)

11.1 Module Dimension Measurement

Unit: mm



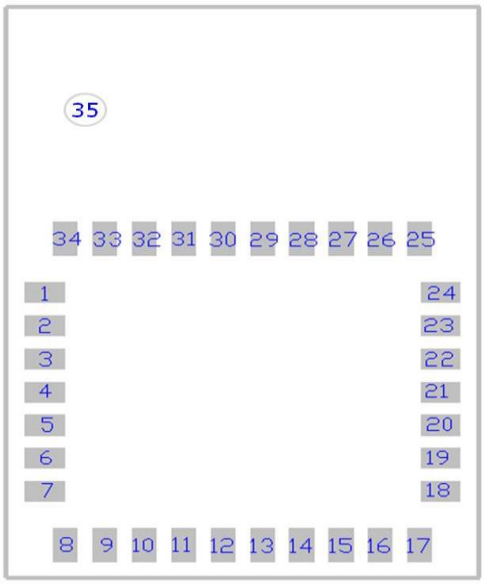
TOP View

Note:

- Please use Un-Solder Mask to design the Module Footprint.
- There are two types pad size in the Module.
  - Rectangle : Pad size: 0.3 x 0.5 mm & Solder Mask size: 0.375 x 0.575 mm
  - Circle: Pad size (Φ): 0.5 mm & Solder Mask size (Φ): 0.6 mm
- Please contact Inventek Systems for interest in a chip-down design.

11.2 The X-Y Central Location Coordinates

Unit: mm (Drawn dimensions with chip 0,0 at bottom right corner)



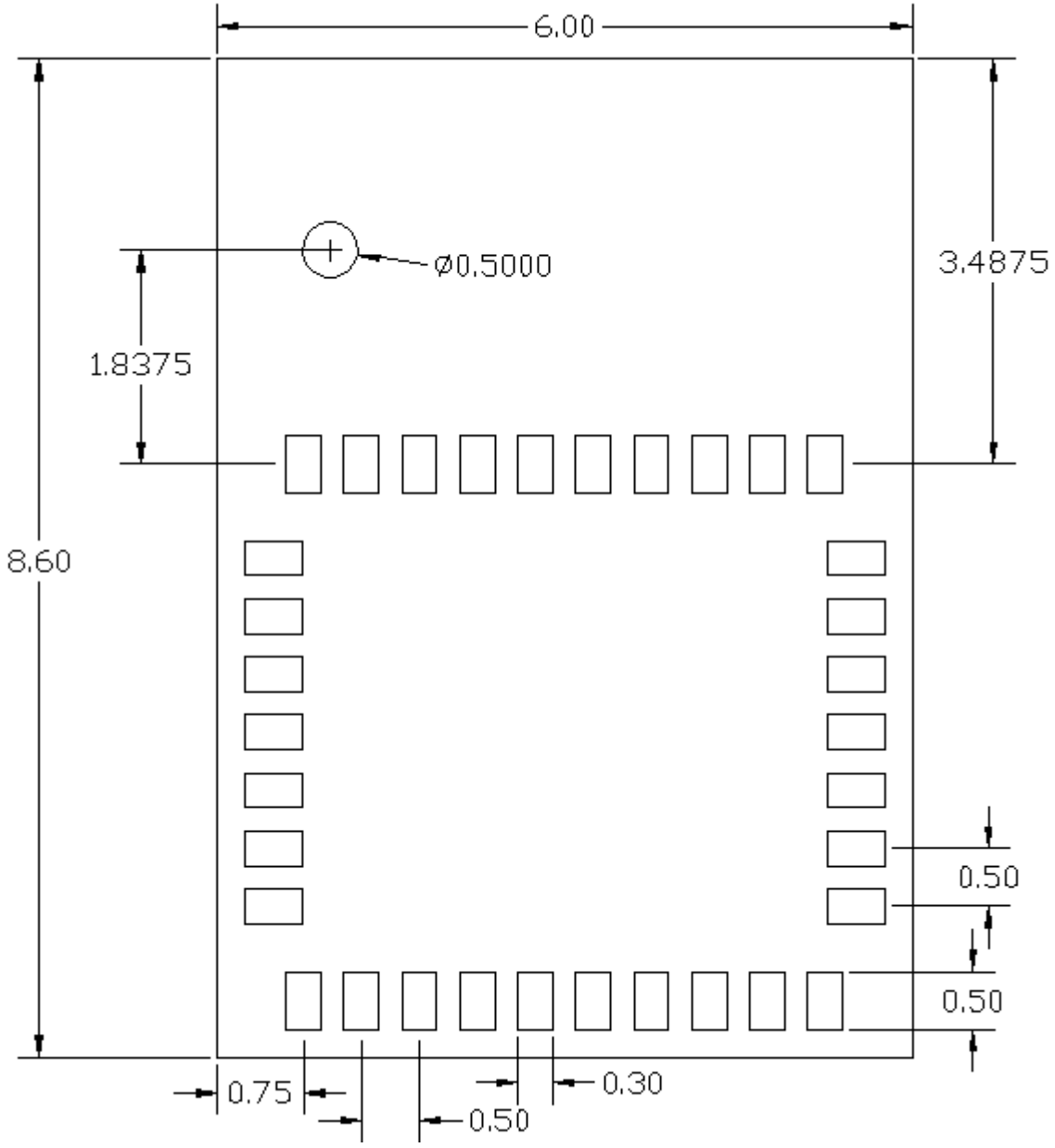
(0,0)

PIN_NUMBER	PAD_Size (mm)	Solder Mask_Size (mm)	PIN_X(mm)	PIN_Y(mm)
1	0.5 x 0.3	0.575 x 0.375	0.4875	4.3
2	0.5 x 0.3	0.575 x 0.375	0.4875	3.8
3	0.5 x 0.3	0.575 x 0.375	0.4875	3.3
4	0.5 x 0.3	0.575 x 0.375	0.4875	2.8
5	0.5 x 0.3	0.575 x 0.375	0.4875	2.3
6	0.5 x 0.3	0.575 x 0.375	0.4875	1.8
7	0.5 x 0.3	0.575 x 0.375	0.4875	1.3
8	0.5 x 0.3	0.575 x 0.375	0.75	0.4875
9	0.5 x 0.3	0.575 x 0.375	1.25	0.4875
10	0.5 x 0.3	0.575 x 0.375	1.75	0.4875
11	0.5 x 0.3	0.575 x 0.375	2.25	0.4875
12	0.5 x 0.3	0.575 x 0.375	2.75	0.4875
13	0.5 x 0.3	0.575 x 0.375	3.25	0.4875
14	0.5 x 0.3	0.575 x 0.375	3.75	0.4875

PIN_NUMBER	PAD_Size (mm)	Solder Mask_Size (mm)	PIN_X(mm)	PIN_Y(mm)
15	0.5 x 0.3	0.575 x 0.375	4.25	0.4875
16	0.5 x 0.3	0.575 x 0.375	4.75	0.4875
17	0.5 x 0.3	0.575 x 0.375	5.25	0.4875
18	0.5 x 0.3	0.575 x 0.375	5.5125	1.3
19	0.5 x 0.3	0.575 x 0.375	5.5125	1.8
20	0.5 x 0.3	0.575 x 0.375	5.5125	2.3
21	0.5 x 0.3	0.575 x 0.375	5.5125	2.8
22	0.5 x 0.3	0.575 x 0.375	5.5125	3.3
23	0.5 x 0.3	0.575 x 0.375	5.5125	3.8
24	0.5 x 0.3	0.575 x 0.375	5.5125	4.3
25	0.5 x 0.3	0.575 x 0.375	5.25	5.1125
26	0.5 x 0.3	0.575 x 0.375	4.75	5.1125
27	0.5 x 0.3	0.575 x 0.375	4.25	5.1125
28	0.5 x 0.3	0.575 x 0.375	3.75	5.1125
29	0.5 x 0.3	0.575 x 0.375	3.25	5.1125
30	0.5 x 0.3	0.575 x 0.375	2.75	5.1125
31	0.5 x 0.3	0.575 x 0.375	2.25	5.1125
32	0.5 x 0.3	0.575 x 0.375	1.75	5.1125
33	0.5 x 0.3	0.575 x 0.375	1.25	5.1125
34	0.5 x 0.3	0.575 x 0.375	0.75	5.1125
35	0.5 mm (Φ)	0.6 mm (Φ)	0.98	6.95

12 Recommend Stencil

Unit: mm

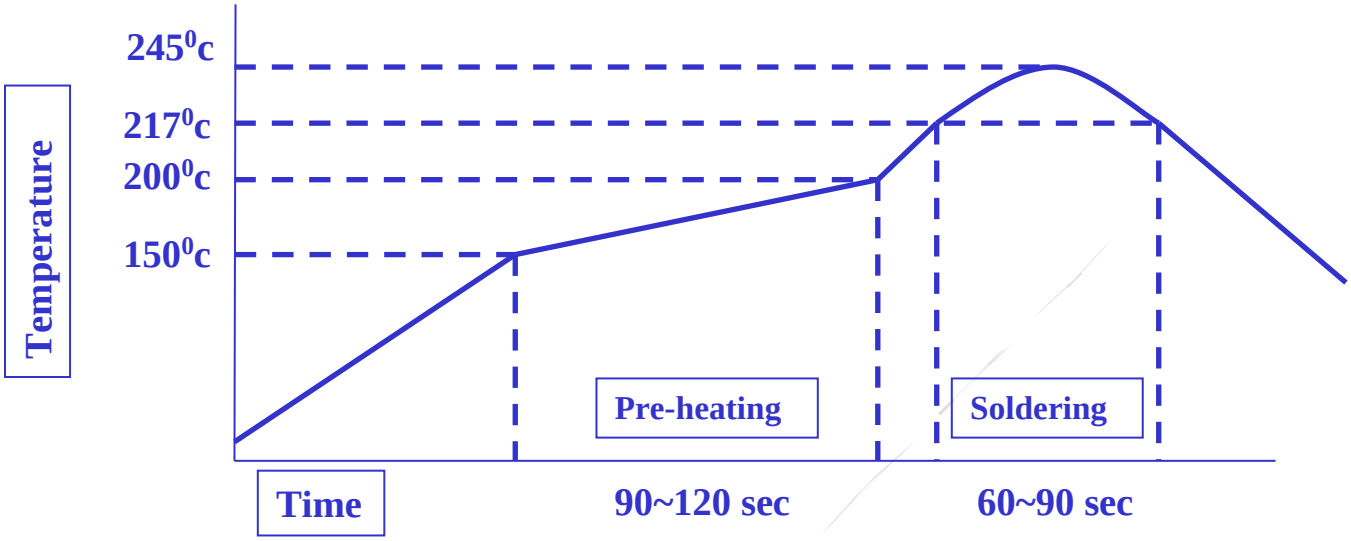


TOP View

Recommend:

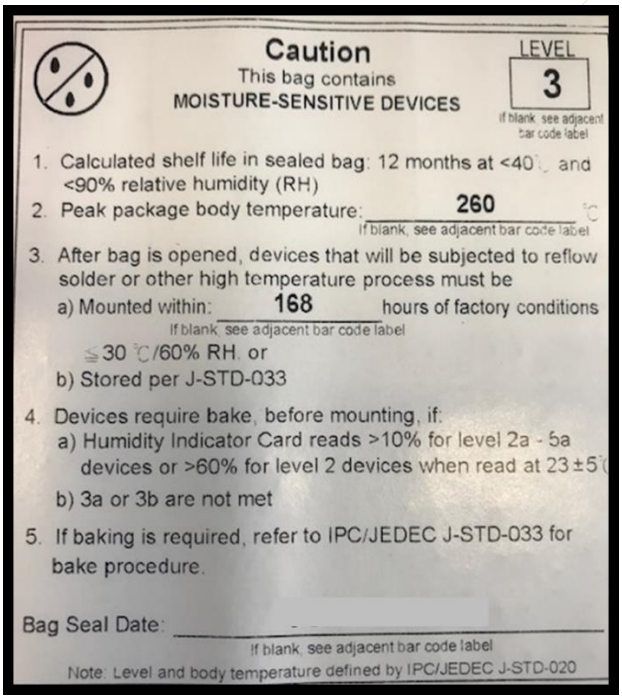
1. ≦ 0.08mm THK stencil will has better solder paste deposit.
2. Type 4 or 5 solder (fine powder size) will has better solution no matter clean or non-clean solder paste.
3. Nitrogen reflow oven.

13 Recommended Reflow Profile



14 Storage Requirements

14.1 MSD Specification


Caution
This bag contains
MOISTURE-SENSITIVE DEVICES
LEVEL 3
if blank, see adjacent bar code label

1. Calculated shelf life in sealed bag: 12 months at <40℃ and <90% relative humidity (RH)
2. Peak package body temperature: 260 ℃
   If blank, see adjacent bar code label
3. After bag is opened, devices that will be subjected to reflow solder or other high temperature process must be
   a) Mounted within: 168 hours of factory conditions
   If blank, see adjacent bar code label
   ≦30 ℃/60% RH, or
   b) Stored per J-STD-033
4. Devices require bake, before mounting, if:
   a) Humidity Indicator Card reads >10% for level 2a - 5a devices or >60% for level 2 devices when read at 23±5℃
   b) 3a or 3b are not met
5. If baking is required, refer to IPC/JEDEC J-STD-033 for bake procedure.

Bag Seal Date: _____
If blank, see adjacent bar code label
Note: Level and body temperature defined by IPC/JEDEC J-STD-020

15 REVISION CONTROL

Document: **ISM14585-L35**	5.0 BLE + Cortex M0 Module
External Release	DOC-DS-14585-201911-3.7

Date	Author	Revision	Comment
7/04/2018	AS	1.0	Preliminary
7/04/2019	AS	2.0	Schematics, Antenna Options and Buck Power Update
10/9/19	AS	2.5	MSD Specification
10/10/2019	AS	3.0	w.fl Antenna Ordering P/N
10/23/19	AS	3.1	Module and Buck Schematic Update
11/19/19	AS	3.5	Added CERT IDs & Note 1 Pin Definitions update
1/28/20	AS	3.6	Clarified VDCDC & Switch Pins
7/9/20	AS	3.7	Audio Note in Sections 2 & 5.1 and Section 1.1 for additional details on Ordering Information
8/24/20	AS	3.8	Internal Flash configuration added to Section 8.2 Detail Pin definition information

16 CONTACT INFORMATION

Inventek Systems
2 Republic Road
Billerica Ma, 01862
Tel: 978-667-1962
Sales@inventeksys.com
www.inventeksys.com

X-ON Electronics

Largest Supplier of Electrical and Electronic Components

Click to view similar products for Bluetooth Modules - 802.15.1 ***category:***

Click to view products by Inventek ***manufacturer:***

Other Similar products are found below :

BM83SM1-00AA ESP32-S2-MINI-2U-N4R2 ESP32-S2-SOLO-2-N4R2 ESP32-S3-MINI-1U-N8 ATWINC1510-MR210PB1976 VG3751T240NFS1 PB-02 PB-03F BT3L BT2S BTU PB-01 PB-02-Kit TB-05 E73-2G4M04S1AX E330-900T13S E73-2G4M08S1EX E83-2G4M03S E104-BT52 E104-BT5005A E73-2G4M04S1F E73-2G4M04S1FX E104-BT40 E104-BT08 E104-BT5010A E72-2G4M05S1G E72-2G4M20S1C E104-BT54S DL-CC2340-B ESP8684-WROOM-02UC-N4 HLK-B40-I HLK-B40 VG6328A Core52840 WCH-BSU BLE-SER-A-ANT WS8000-M6 WL6601-TC E104-BT09 E73-2G4M04S1BX ESP32-H2-MINI-1U-H4 RN4678-VB/RM122 ESP32-C6-WROOM-1-N16 WT52810-S1 WT52840-S1 ESP8684-MINI-1U-H4 EYSHSNZWZ CYBT-483056-02 nRF52840-QIAA-F-R7 60434