Sub-1GHz Low Current RF Transceiver
Smoke Detector Flash MCU

BA45F5640

Revision: V1.00 Date: December 09, 2019

www.holtek.com

Table of Contents

Features **7**
CPU Features 7
Peripheral Features 7
RF Features 8
General Description **9**
Block Diagram **10**
Pin Assignment **11**
Pin Description **12**
Absolute Maximum Ratings **15**
D.C. Characteristics **15**
Operating Voltage Characteristics 15
Operating Current Characteristics 15
Standby Current Characteristics 16
A.C. Characteristics **16**
High Speed Internal Oscillator Frequency Accuracy 16
Low Speed Internal Oscillator Characteristics – LIRC 17
Operating Frequency Characteristic Curves 17
System Start Up Time Characteristics 17
Input/Output Characteristics **18**
Memory Characteristics **18**
LVD & LVR Electrical Characteristics **19**
A/D Converter Electrical Characteristics **19**
Reference Voltage Characteristics **20**
Sink Current Generator Electrical Characteristics **20**
Sink Current Generator Characteristic Curves 21
Operational Amplifier Electrical Characteristics **22**
For Smoke Detector AFE 22
For Power Line Transceiver 23
D/A Converter Electrical Characteristics **23**
Comparator Electrical Characteristics **24**
RF Electrical Characteristics **25**
Power-on Reset Characteristics **27**
System Architecture **27**
Clocking and Pipelining 27
Program Counter 28
Stack 29
Arithmetic and Logic Unit – ALU 29

Flash Program Memory 30
Structure 30
Special Vectors 30
Look-up Table 30
Table Program Example 31
In Circuit Programming – ICP 32
On Chip Debug Support – OCDS 33
Data Memory 33
Structure 33
Data Memory Addressing 34
General Purpose Data Memory 34
Special Purpose Data Memory 34
Special Function Register Description 36
Indirect Addressing Registers – IAR0, IAR1, IAR2 36
Memory Pointers – MP0, MP1L, MP1H, MP2L, MP2H 36
Accumulator – ACC 37
Program Counter Low Register – PCL 38
Look-up Table Registers – TBLP, TBHP, TBLH 38
Status Register – STATUS 38
EEPROM Data Memory 40
EEPROM Data Memory Structure 40
EEPROM Registers 40
Reading Data from the EEPROM 42
Writing Data to the EEPROM 42
Write Protection 42
EEPROM Interrupt 42
Programming Considerations 43
Oscillators 44
Oscillator Overview 44
System Clock Configurations 44
Internal High Speed RC Oscillator – HIRC 44
Internal 32kHz Oscillator – LIRC 45
Operating Modes and System Clocks 45
System Clocks 45
System Operation Modes 46
Control Registers 47
Operating Mode Switching 48
Standby Current Considerations 51
Wake-up 52
Watchdog Timer 53
Watchdog Timer Clock Source 53
Watchdog Timer Control Register 53
Watchdog Timer Operation 54

Reset and Initialisation 55
Reset Functions 55
Reset Initial Conditions 56
Input/Output Ports 60
Pull-high Resistors 60
Port A Wake-up 61
I/O Port Control Registers 61
I/O Port Source Current Selection 61
Pin-shared Functions 62
I/O Pin Structures 66
Programming Considerations 66
Timer Modules – TM 67
Introduction 67
TM Operation 67
TM Clock Source 67
TM Interrupts 67
TM External Pins 68
Programming Considerations 69
Standard Type TM – STM 70
Standard TM Operation 70
Standard Type TM Register Description 70
Standard Type TM Operation Modes 75
Periodic Type TM – PTM 85
Periodic TM Operation 85
Periodic Type TM Register Description 86
Periodic Type TM Operating Modes 91
Smoke Detector AFE 103
Smoke Detector AFE Registers 103
PowerLine Transceiver − PLT 107
PowerLine Transceiver Registers 108
Offset Calibration Procedure 113
Analog to Digital Converter 115
A/D Converter Overview 115
A/D Converter Register Description 116
A/D Converter Operation 119
A/D Converter Reference Voltage 120
A/D Converter Input Signals 120
Conversion Rate and Timing Diagram 121
Summary of A/D Conversion Steps 122
Programming Considerations 123
A/D Conversion Function 123
A/D Conversion Programming Examples 123
Sink Current Generator 125

Universal Serial Interface Module – USIM 126
SPI Interface 126
I²C Interface 134
UART Interface 144
Low Voltage Detector – LVD 159
LVD Register 159
LVD Operation 160
Interrupts 160
Interrupt Registers 160
Interrupt Operation 164
External Interrupts 165
PLT Comparator Interrupts 166
A/D Converter Interrupt 166
Time Base Interrupts 166
USIM Interrupt 167
LVD Interrupt 168
EEPROM Interrupt 168
TM Interrupts 168
Interrupt Wake-up Function 168
Programming Considerations 169
Memory Mapping 170
Control Register Access 170
SFR Mapping and Bit Definition 171
Common Area Control Register 171
Bank 0 Control Registers 183
Bank 1 Control Registers 190
Bank 2 Control Registers 196
Special Function Description 197
Sub-1GHz RF Transceiver 197
Serial Interface 197
System Clock 199
Frequency Synthesizer 199
Modulator 200
State Machine 200
Calibration 204
AGC & RSSI 204
Packet Handler 205
FIFO Operation Modes 207
Receiving Packet Judgement 210
Continuous RX Mode 211
ARK Mode: Auto-Resend and Auto-Ack 212
ATR Mode: Auto-Transmit-Receive 214
Message Flowchart Examples 217

Abbreviation **220**
Application Circuits **222**
Instruction Set **223**
Introduction 223
Instruction Timing 223
Moving and Transferring Data 223
Arithmetic Operations 223
Logical and Rotate Operation 224
Branches and Control Transfer 224
Bit Operations 224
Table Read Operations 224
Other Operations 224
Instruction Set Summary **225**
Table Conventions 225
Extended Instruction Set 227
Instruction Definition **229**
Extended Instruction Definition 238
Package Information **245**
SAW Type 46-pin QFN (6.5mm×4.5mm×0.75mm) Outline Dimensions 246

Features

CPU Features

- Operating Voltage
  - f_{SYS}=2MHz: 2.2V~3.6V
  - f_{SYS}=4MHz: 2.2V~3.6V
  - f_{SYS}=8MHz: 2.2V~3.6V
- Up to 0.5μs instruction cycle with 8MHz system clock at V_{DD}=5V
- Power down and wake-up functions to reduce power consumption
- Oscillator types
  - Internal High Speed 2/4/8MHz RC – HIRC
  - Internal Low Speed 32kHz RC – LIRC
- Multi-mode operation: FAST, SLOW, IDLE and SLEEP
- Fully integrated internal oscillators require no external components
- All instructions executed in 1~3 instruction cycles
- Table read instructions
- 115 powerful instructions
- 8-level subroutine nesting
- Bit manipulation instruction

Peripheral Features

- Flash Program Memory: 4K×16
- RAM Data Memory: 256×8
- True EEPROM Memory: 64×8
- Watchdog Timer function
- Up to 13 bidirectional I/O lines
- Up to two external interrupt lines shared with I/O pins
- Programmable I/O port source current for LED applications
- Sink current generator for constant current output
- Smoke Detector AFE including two operational amplifiers
- Power Line Data Transceiver including two comparators, one Operational Amplifier and three D/A converters
- Multiple Timer Modules for time measure, input capture, compare match output, PWM output or single pulse output function
- Dual Time-Base functions for generation of fixed time interrupt signals
- Universal Serial Interface Module – USIM for SPI, I^2C or UART communication
- 4 external channels 12-bit resolution A/D converter
- Low voltage reset function
- Low voltage detect function
- Package type: 46-pin QFN

RF Features

- Frequency band: 315/433/470/868/915MHz
- FSK/GFSK modulation
- Supports 3-wire or 4-wire SPI interface
- Wide input voltage range of 2.2V~3.6V
- Programmable data rate: 2Kbps~250Kbps
- Programmable TX output power: 0dBm~13dBm
- Low current consumption
  - 0.5μA deep sleep mode current with data retention
  - RX current consumption (AGC on & low data rate) @ 433.92MHz: 4.2mA
  - RX current consumption (AGC on & low data rate) @ 868.3MHz: 5.5mA
  - TX current consumption @ 433.92MHz: 22mA@10dBm P_{OUT}
  - TX current consumption @ 868.3MHz: 24mA@10dBm P_{OUT}
- High RX sensitivity (433.92MHz)
  - -119dBm at 2Kbps on-air data rate
  - -109dBm at 50Kbps on-air data rate
  - -100dBm at 250Kbps on-air data rate
- High RX sensitivity (868.3MHz)
  - -118dBm at 2Kbps on-air data rate
  - -108dBm at 50Kbps on-air data rate
  - -100dBm at 250Kbps on-air data rate
- On-chip VCO and Fractional-N synthesizer with integrated loop filter
- Supports low cost 16MHz crystal with integrated load capacitor
- Programmable digital channel filter for optimum performance at various data rates
- AGC (Auto Gain Control) to achieve wide input range, up to +10dBm
- AFC (Auto Frequency Compensation) for frequency drift due to X'tal aging
- On-chip low power RC oscillator for WOR (Wake-on-RX) and WOT (Wake-on-TX) functions
- Physical TX/RX FIFO buffers: TX 64 bytes, RX 64 bytes
- Simple FIFO/Block FIFO/Extend FIFO (up to 255 bytes)/Infinite FIFO modes
- Programmable threshold for carrier detection
- Frame synchronization recognition for both FIFO mode and Direct mode
- Packet handling
  - FEC (Forward Error Correction)
  - Data whitening
  - Manchester encoding
  - CRC-16 checking
- ATR (Auto-Transmit-Receive)
  - Auto-resend
  - Auto-acknowledgment
  - WOT + Auto-resend
  - WOR + Auto-acknowledgment
- Packet filtering
  - CRC filtering
  - Address filtering

General Description

The BA45F5640 is a Flash Memory A/D type 8-bit high performance RISC architecture microcontroller, designed especially for smoke detector applications. Offering users the convenience of Flash Memory multi-programming features, the device also includes a wide range of flexible functions and features. Other memory includes an area of RAM Data Memory as well as an area of true EEPROM memory for storage of non-volatile data such as serial numbers, calibration data etc.

Analog features include a multi-channel Analog to Digital converter, two comparators, three operational amplifiers and three D/A converters. With regard to internal timers, the device includes multiple and extremely flexible Timer Modules providing functions for timing, pulse generation and PWM output operations. Communication with the outside world is catered for by including fully integrated SPI, I^2C and UART interface functions, three popular interfaces which provide designers with a means of easy communication with external peripheral hardware. Protective features such as an internal Watchdog Timer, Low Voltage Reset and Low Voltage Detector coupled with excellent noise immunity and ESD protection ensure that reliable operation is maintained in hostile electrical environments.

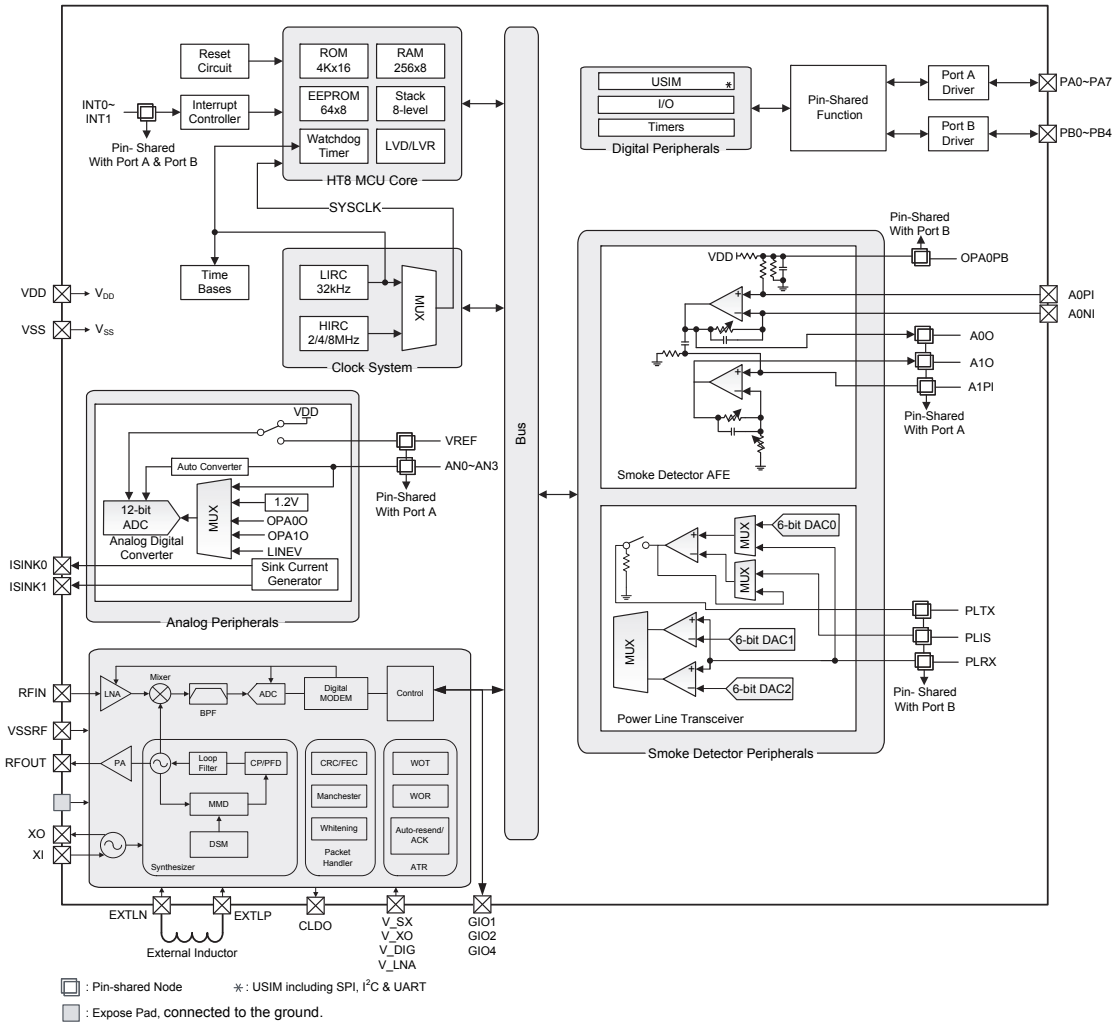
The device also includes fully integrated high and low speed oscillators which require no external components for their implementation. The ability to operate and switch dynamically between a range of operating modes using different clock sources gives users the ability to opetimise microcontroller operation can minimum power consumption.

The RF module is a high performance and low cost FSK/GFSK transceiver for wireless applications in the 315MHz, 433MHz, 470MHz, 868MHz and 915MHz frequency bands. It incorporates a highly integrated sub-1GHz transceiver and a baseband modem with programmable data rates from 2Kbps to 250Kbps. Data handling features include 64-byte TX/RX FIFO and packet handling such as CRC generation, Forward Error Correction and data whitening, Manchester encoding.

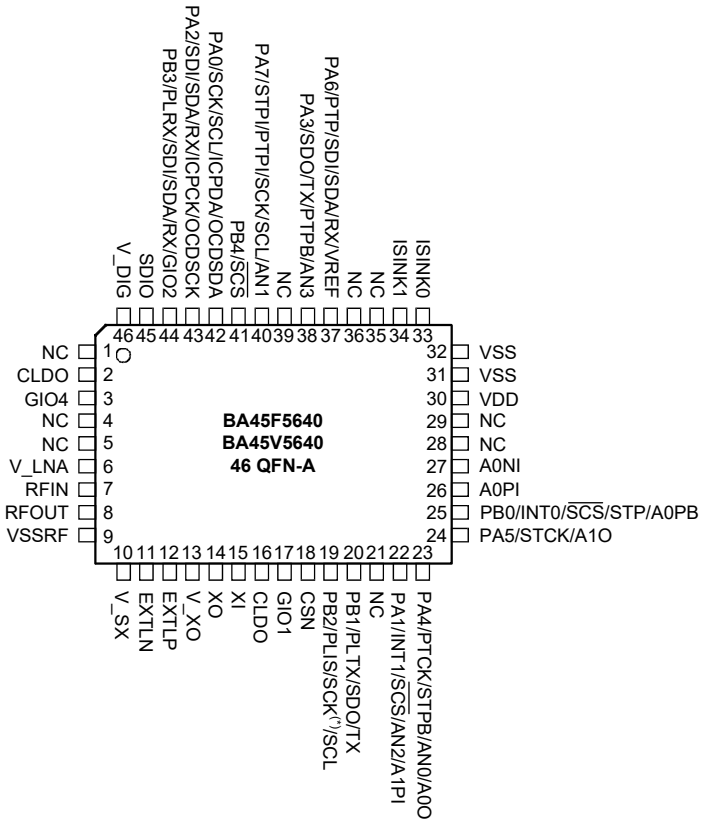
The RF module is optimized for the very low power consumption applications. At 433MHz band, its RX mode is operated at 4.2mA and it delivers +10dBm TX output power at 22mA current consumption. A low-noise low-IF receiver can achieve -119dBm sensitivity of 2Kbps data rate at 433/868MHz bands. A Class-E Power Amplifier can deliver up to +13dBm output power at 433/868MHz bands. A fully integrated Fractional-N synthesizer can support a wide frequency range with a fine resolution. Both loop filter and XO load capacitors are integrated to on-chip to minimize the requirement for external components.

While the inclusion of flexible I/O programming features, Time-Base functions along with many other features ensure that the device is suitable for a wide range of applications such as Smoke Detetor, IoT (smart community), Wireless meter reading (water/gas meter) and Agricultural/Industrial control and so on.

Block Diagram



Pin Assignment



Note: 1. If the pin-shared pin functions have multiple outputs, the desired pin-shared functions are determined by the corresponding pin-shared software control bits.

2. The OCDSDA and OCDSCK pins are supplied for the OCDS dedicated pins and are only available for the BA45V5640 device which is the OCDS EV chip for the BA45F5640 device.

3. The SCK and GIO2 lines, which are pin-shard with the MCU PB2 and PB3 pins respectively.

4. *: The SCK pin pin-shared with PB2 is the RF SPI serial clock line or the MCU SPI serial clock line. It is important not to confuse this with other SCK pin pin-shared with Port A which is the serial clock line of the master MCU.

Pin Description

The function of each pin is listed in the following table, however the details behind how each pin is configured is contained in other sections of the datasheet.

Pin Name	Function	OPT	I/T	O/T	Description
PA0/SCL/SCK/ ICPDA/OCDSDA	PA0	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	SCL	IFS0 PAS0	ST	NMOS	I^2C clock line
	SCK	IFS0 PAS0	ST	CMOS	SPI serial clock
	ICPDA	—	ST	CMOS	ICP data/address
	OCDSDA	—	ST	CMOS	OCDS data/address, for EV chip only
PA1/INT1/$\overline{SCS}$/AN2/A1PI	PA1	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	INT1	PAS0 INTC0 INTEG	ST	—	External Interrupt 1
	$\overline{SCS}$	IFS0 PAS0	ST	CMOS	SPI slave device select
	AN2	PAS0	AN	—	A/D Converter input channel 2
	A1PI	PAS0	AN	—	SD Operational Amplifier 1 positive input
PA2/SDI/SDA/RX/ ICPCK/OCDSCK	PA2	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	SDI	IFS0 PAS0	ST	—	SPI serial data input
	SDA	IFS0 PAS0	ST	NMOS	I^2C data line
	RX	IFS0 PAS0	ST	—	UART receiver pin
	ICPCK	—	ST	—	ICP clock pin
	OCDSCK	—	ST	—	OCDS clock pin, for EV chip only
PA3/SDO/TX/PTPB/AN3	PA3	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	SDO	PAS0	—	CMOS	SPI serial data output
	TX	PAS0	—	CMOS	UART transmitter pin
	PTPB	PAS0	—	CMOS	PTM inverted output
	AN3	PAS0	AN	—	A/D Converter input channel 3
PA4/PTCK/STPB/AN0/ A0O	PA4	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	PTCK	PAS1	ST	—	PTM clock input or capture input
	STPB	PAS1	—	CMOS	STM inverted output
	AN0	PAS1	AN	—	A/D Converter input channel 0
	A0O	PAS1	—	AN	SD Operational Amplifier 0 output

Pin Name	Function	OPT	I/T	O/T	Description
PA5/STCK/A1O	PA5	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	STCK	IFS1 PAS1	ST	—	STM clock input
	A1O	PAS1	—	AN	SD Operational Amplifier 1 output
PA6/PTP/SDI/SDA/RX/ VREF	PA6	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	PTP	PAS1	—	CMOS	PTM output
	SDI	IFS0 PAS1	ST	—	SPI serial data input
	SDA	IFS0 PAS1	ST	NMOS	I^2C data line
	RX	IFS0 PAS1	ST	—	UART receiver pin
	VREF	PAS1	AN	—	A/D Converter external reference voltage
PA7/STPI/PTPI/SCK/ SCL/AN1	PA7	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	STPI	PAS1	ST	—	STM capture
	PTPI	IFS0 PAS1	ST	—	PTM capture input
	SCK	IFS0 PAS1	ST	CMOS	SPI serial clock
	SCL	IFS0 PAS1	ST	NMOS	I^2C clock line
	AN1	PAS1	AN	—	A/D Converter input channel 1
PB0/INT0/$\overline{SCS}$/STP/ A0PB	PB0	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high
	INT0	PBS0 INTEG	ST	—	External Interrupt 0
	$\overline{SCS}$	IFS0 PBS0	ST	CMOS	SPI slave device select
	STP	PBS0	—	CMOS	STM output
	A0PB	PBS0	AN	—	SD Operational Amplifier 0 bias input
PB1/PLTX/SDO/TX	PB1	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high
	PLTX	PBS0	—	AN	Power Line TX
	SDO	PBS0	—	CMOS	SPI serial data output
	TX	PBS0	—	CMOS	UART transmitter pin
PB2/PLIS/SCK/SCL	PB2	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high
	PLIS	PBS0	AN	—	Power Line IS input
	SCK	IFS0 PBS0	ST	CMOS	MCU SPI serial clock RF SPI serial clock
	SCL	IFS0 PBS0	ST	NMOS	I^2C clock line

Pin Name	Function	OPT	I/T	O/T	Description
PB3/PLRX/SDI/SDA/RX/GIO2	PB3	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high
	PLRX	PBS0	AN	—	Power Line RX
	SDI	IFS0 PBS0	ST	—	SPI serial data input
	SDA	IFS0 PBS0	ST	NMOS	I^2C data line
	RX	IFS0 PBS0	ST	—	UART receiver pin
	GIO2	—	ST	CMOS	RF multi-function I/O 2
PB4/$\overline{SCS}$	PB4	PBPU PBS1	ST	CMOS	General purpose I/O. Register enabled pull-high
	$\overline{SCS}$	IFS0 PBS1	ST	CMOS	SPI slave device select
ISINK0	ISINK0	—	—	AN	Sink current 0 source
ISINK1	ISINK1	—	—	AN	Sink current 1 source
A0NI	A0NI	—	AN	—	SD Operational Amplifier 0 negative input
A0PI	A0PI	—	AN	—	SD Operational Amplifier 0 positive input
CLDO	CLDO	—	—	PWR	LDO output, connected to a bypass capacitor
GIO4	GIO4	—	ST	CMOS	RF multi-function I/O 4
V_LNA	V_LNA	—	PWR	—	LNA power supply
RFIN	RFIN	—	AN	—	RF LNA input
RFOUT	RFOUT	—	—	AN	RF Power Amplifier Output
VSSRF	VSSRF	—	PWR	—	RF ground
V_SX	V_SX	—	PWR	—	Synthesizer power supply
EXTLN	EXTLN	—	AN	—	Connected to an external inductor
EXTLP	EXTLP	—	AN	—	Connected to an external inductor
V_XO	V_XO	—	PWR	—	Analog positive power supply
XO	XO	—	—	HXT	Crystal oscillator output
XI	XI	—	HXT	—	Crystal oscillator input
CLDO	CLDO	—	—	PWR	LDO output, connected to a bypass capacitor
GIO1	GIO1	—	ST	CMOS	RF multi-function I/O 1
CSN	CSN	—	ST	—	SPI chip select input, low active
SDIO	SDIO	—	ST	CMOS	SPI data input/output
V_DIG	V_DIG	—	PWR	—	Digital positive power supply
GND	GND	—	PWR	—	Ground
VDD	VDD	—	PWR	—	Positive power supply
VSS	VSS	—	PWR	—	Negative power supply, ground
NC	—	—	—	—	No connection

Legend: I/T: Input type; O/T: Output type;
OPT: Optional by register option; PWR: Power;
ST: Schmitt Trigger input; AN: Analog signal;
CMOS: CMOS output; NMOS: NMOS output;
HXT: High frequency crystal oscillator.

Absolute Maximum Ratings

Supply Voltage ..... V_{SS}−0.3V to V_{SS}+3.6V
Input Voltage ..... V_{SS}−0.3V to V_{DD}+0.3V
Storage Temperature..... -50˚C to 125˚C
Operating Temperature..... -40˚C to 85˚C
I_{OL} Total 80mA
I_{OH} Total ..... -80mA
Total Power Dissipation 500mW
ESD HBM ..... ±2kV

*The device is ESD sensitive. HBM(Human Body Mode) is based on MIL-STD-883.

Note: These are stress ratings only. Stresses exceeding the range specified under "Absolute Maximum Ratings" may cause substantial damage to the device. Functional operation of the device at other conditions beyond those listed in the specification is not implied and prolonged exposure to extreme conditions may affect devices reliability.

D.C. Characteristics

For data in the following tables, note that factors such as oscillator type, operating voltage, operating frequency, pin load conditions, temperature and program instruction type, etc., can all exert an influence on the measured values.

Operating Voltage Characteristics

Ta= -40°C~85°C

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_{DD}	Operating Voltage (HIRC)	f_{SYS}=f_{HIRC}=2MHz	2.2	—	3.6	V
		f_{SYS}=f_{HIRC}=4MHz	2.2	—	3.6	
		f_{SYS}=f_{HIRC}=8MHz	2.2	—	3.6	
	Operating Voltage (LIRC)	f_{SYS}=f_{LIRC}=32kHz	2.2	—	3.6	V

Operating Current Characteristics

Ta=25°C

Symbol	Normal Operation	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{DD}	SLOW Mode (LIRC)	2.2V	f_{SYS}=32kHz	—	8	16	μA
		3V		—	10	20	
	FAST Mode (HIRC)	2.2V	f_{SYS}=2MHz	—	0.15	0.20	mA
		3V		—	0.20	0.30	
		2.2V	f_{SYS}=4MHz	—	0.3	0.5	mA
		3V		—	0.4	0.6	
		2.7V	f_{SYS}=8MHz	—	0.6	1.0	mA
		3V		—	0.8	1.2	

Note: When using the characteristic table data, the following notes should be taken into consideration:

1. Any digital inputs are setup in a non-floating condition.
2. All measurements are taken under conditions of no load and with all peripherals in an off state.
3. There are no DC current paths.
4. All Operating Current values are measured using a continuous NOP instruction program loop.

Standby Current Characteristics

Ta=25°C, unless otherwise specified.

Symbol	Standby Mode	Test Conditions		Min.	Typ.	Max.	Max. @ 85°C	Unit
		V_{DD}	Conditions					
I_{STB}	SLEEP Mode	2.2V	WDT on	—	1.2	2.4	2.9	μA
		3V		—	1.5	3.0	3.6	
	IDLE0 Mode (LIRC)	2.2V	f_{SUB} on	—	2.4	4.0	4.8	μA
		3V		—	3	5	6	
	IDLE1 Mode (HIRC)	2.2V	f_{SUB} on, f_{SYS}=2MHz	—	60	120	140	μA
		3V		—	70	140	160	
		2.2V	f_{SUB} on, f_{SYS}=4MHz	—	90	200	220	μA
		3V		—	110	220	240	
		2.7V	f_{SUB} on, f_{SYS}=8MHz	—	150	300	340	μA
		3V		—	180	360	400	

Note: When using the characteristic table data, the following notes should be taken into consideration:

1. Any digital inputs are setup in a non-floating condition.
2. All measurements are taken under conditions of no load and with all peripherals in an off state.
3. There are no DC current paths.
4. All Standby Current values are taken after a HALT instruction execution thus stopping all instruction execution.

A.C. Characteristics

For data in the following tables, note that factors such as oscillator type, operating voltage, operating frequency and temperature etc., can all exert an influence on the measured values.

High Speed Internal Oscillator Frequency Accuracy

During the program writing operation the writer will trim the HIRC oscillator at a user selected HIRC frequency and user selected voltage of 3V.

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Temp.				
f_{HIRC}	2MHz Writer Trimmed HIRC Frequency	3V	25°C	-1%	2	+1%	MHz
			-20°C~60°C	-2%	2	+2%	
			-40°C~85°C	-3%	2	+3%	
	4MHz Writer Trimmed HIRC Frequency	3V	25°C	-1%	4	+1%	MHz
			-40°C~85°C	-2%	4	+2%	
		2.4V~3.3V	-20°C~50°C	-2%	4	+2%	
	8MHz Writer Trimmed HIRC Frequency	3V	25°C	-1%	8	+1%	MHz
			-40°C~85°C	-10%	8	+2%	

Note: 1. The 3V values for V_{DD} are provided as this is the fixed voltage at which the HIRC frequency is trimmed by the writer.

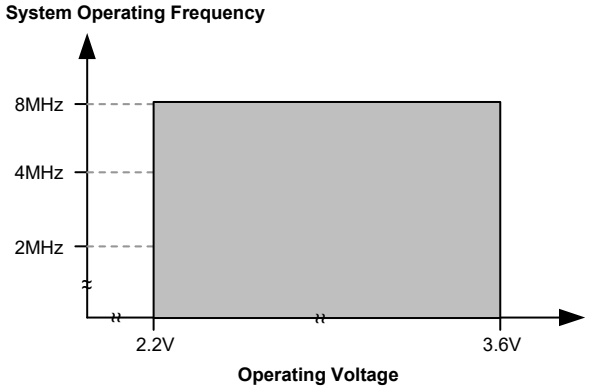
2. The row below the 3V trim voltage row is provided to show the values for the full V_{DD} range operating voltage. It is recommended that the trim voltage is fixed at 3V for application voltage ranges from 2.2V to 3.6V.

3. The minimum and maximum tolerance values provided in the table are only for the frequency at which the writer trims the HIRC oscillator. After trimming at this chosen specific frequency any change in HIRC oscillator frequency using the oscillator register control bits by the application program will give a frequency tolerance to within ±20%.

Low Speed Internal Oscillator Characteristics – LIRC

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Temp.				
f_{LIRC}	LIRC Frequency	2.2V~3.6V	25°C	-5%	32	+5%	kHz
			-40°C~85°C	-10%	32	+10%	
t_{START}	LIRC Start-up Time	—	—	—	—	100	μs

Operating Frequency Characteristic Curves



System Start Up Time Characteristics

Ta=-40°C~85°C

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
t_{SST}	System Start-up Time Wake-up from Condition where f_{SYS} is off	$f_{SYS}=f_H\sim f_H/64$, $f_H=f_{HIRC}$	—	16	—	t_{HIRC}
		$f_{SYS}=f_{SUB}=f_{LIRC}$	—	2	—	t_{LIRC}
	System Start-up Time Wake-up from Condition where f_{SYS} is on	$f_{SYS}=f_H\sim f_H/64$, $f_H=f_{HIRC}$	—	2	—	t_H
		$f_{SYS}=f_{SUB}=f_{LIRC}$	—	2	—	t_{SUB}
	System Speed Switch Time FAST to SLOW Mode or SLOW to FAST Mode	f_{HIRC} switches from off → on	—	16	—	t_{HIRC}
t_{RSTD}	System Reset Delay Time Reset Source from Power-on Reset or LVR Hardware Reset	RR_{POR}=5V/ms	42	48	54	ms
	System Reset Delay Time WDTC Register Software Reset	—				
	System Reset Delay Time WDT Overflow Reset	—	14	16	18	
t_{SRESET}	Minimum Software Reset Width to Reset	—	45	90	120	μs

Note: 1. For the System Start-up time values, whether f_{SYS} is on or off depends upon the mode type and the chosen f_{SYS} system oscillator. Details are provided in the System Operating Modes section.

2. The time units, shown by the symbols t_{HIRC} etc. are the inverse of the corresponding frequency values as provided in the frequency tables. For example $t_{HIRC}=1/f_{HIRC}$, $t_{SYS}=1/f_{SYS}$ etc.

3. If the LIRC is used as the system clock and if it is off when in the SLEEP Mode, then an additional LIRC start up time, t_{START}, as provided in the LIRC frequency table, must be added to the t_{SST} time in the table above.

4. The System Speed Switch Time is effectively the time taken for the newly activated oscillator to start up.

Input/Output Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{IL}	Input Low Voltage for I/O Ports	—	—	0	—	$0.2V_{DD}$	V
V_{IH}	Input High Voltage for I/O Ports	—	—	$0.8V_{DD}$	—	V_{DD}	V
I_{OL}	Sink Current for I/O Ports	3V	$V_{OL}=0.1V_{DD}$	16	32	—	mA
I_{OH}	Source Current for I/O Ports	3V	$V_{OH}=0.9V_{DD}$, SLEDC[m+1, m]=00B (m=0, 2, 4, 6)	-0.7	-1.5	—	mA
		3V	$V_{OH}=0.9V_{DD}$, SLEDC[m+1, m]=01B (m=0, 2, 4, 6)	-1.3	-2.5	—	
		3V	$V_{OH}=0.9V_{DD}$, SLEDC[m+1, m]=10B (m=0, 2, 4, 6)	-1.8	-3.6	—	
		3V	$V_{OH}=0.9V_{DD}$, SLEDC[m+1, m]=11B (m=0, 2, 4, 6)	-4	-8	—	
R_{PH}	Pull-high Resistance for I/O Ports (Note)	3V	—	20	60	100	kΩ
I_{TOL}	Total Sink Current for all I/O Ports	3V	—	40	—	—	mA
I_{TOH}	Total Source Current for all I/O Ports	3V	—	-40	—	—	mA
t_{TPI}	STM STPI Input Pin Minimum Pulse Width	—	—	0.3	—	—	μs
	PTM PTPI Input Pin Minimum Pulse Width			0.1	—	—	
t_{TCK}	STM/PTM TCK Input Pin Minimum Pulse Width	—	—	0.3	—	—	μs
t_{INT}	Interrupt Pin Minimum Pulse Width	—	—	10	—	—	μs
t_{CPW}	PTM Minimum Capture Pulse Width	—	—	2	—	—	t_{TMCLK}

Note: The R_{PH} internal pull-high resistance value is calculated by connecting to ground and enabling input pin with pull-high resistor and then measuring the pin current at the specified supply voltage level. Dividing the voltage by this measured current provides the R_{PH} value.

Memory Characteristics

Ta=-40°C~85°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{RW}	V_{DD} for Read/Write	—	—	V_{DDmin}	—	V_{DDmax}	V
Flash Program / Data EEPROM Memory							
t_{DEW}	Erase/Write Cycle Time – Flash Program Memory	—	—	—	2	3	ms
	Write Cycle Time – Data EEPROM Memory	—	—	—	4	6	ms
I_{DDPGM}	Programming/Erase Current on V_{DD}	—	—	—	—	5.0	mA
E_P	Cell Endurance – Flash Program Memory	—	—	10K	—	—	E/W
	Cell Endurance – Data EEPROM Memory	—	—	100K	—	—	E/W
t_{RETD}	ROM Data Retention Time	—	Ta=25°C	—	40	—	Year
RAM Data Memory							
V_{DR}	RAM Data Retention Voltage	—	Device in SLEEP Mode	1.0	—	—	V

LVD & LVR Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{LVR}	Low Voltage Reset Voltage	—	LVR enable	-5%	2.1	+5%	V
V_{LVD}	Low Voltage Detection Voltage	—	LVD enable, voltage select 2.2V	-5%	2.2	+5%	V
			LVD enable, voltage select 2.4V		2.4		
			LVD enable, voltage select 2.7V		2.7		
			LVD enable, voltage select 3.0V		3.0		
			LVD enable, voltage select 3.3V		3.3		
			LVD enable, voltage select 3.6V		3.6		
			LVD enable, voltage select 4.0V		4.0		
$I_{LVRLVDBG}$	Operating Current	3V	LVD enable, LVR enable, VBGEN=0	—	15	20	μA
		3V	LVD enable, LVR enable, VBGEN=1	—	20	25	μA
t_{LVDS}	LVDO Stable Time	—	For LVR enable, VBGEN=0, LVD off → on	—	—	15	μs
			For LVR disable, VBGEN=0, LVD off → on	—	—	150	
t_{LVR}	Minimum Low Voltage Width to Reset	—	—	120	240	480	μs
t_{LVD}	Minimum Low Voltage Width to Interrupt	—	—	60	120	240	μs
I_{LVR}	Additional Current for LVR Enable	—	LVD disable, VBGEN=0	—	20	25	μA

A/D Converter Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.2	—	3.6	V
V_{ADI}	Input Voltage	—	—	0	—	V_{REF}	V
V_{REF}	Reference Voltage	—	—	2	—	V_{DD}	V
DNL	Differential Non-linearity	3V	V_{REF}=V_{DD}, t_{ADCK}=0.5μs	-3	—	+3	LSB
		3V	V_{REF}=V_{DD}, t_{ADCK}=8μs				
INL	Integral Non-linearity	3V	V_{REF}=V_{DD}, t_{ADCK}=0.5μs	-4	—	+4	LSB
		3V	V_{REF}=V_{DD}, t_{ADCK}=8μs				
I_{ADC}	Additional Current for A/D Converter Enable	3V	No load, t_{ADCK}=0.5μs	—	1	2	mA
t_{ADCK}	Clock Period	—	—	0.5	—	10.0	μs
t_{ON2ST}	A/D Converter On-to-Start Time	—	—	4	—	—	μs
t_{ADS}	Sampling Time	—	—	—	4	—	t_{ADCK}
t_{ADC}	Conversion Time (Include A/D Sample and Hold Time)	—	—	—	16	—	t_{ADCK}

Reference Voltage Characteristics

Ta=25°C, unless otherwise specified.

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.2	—	3.6	V
V_{BGREF}	Bandgap Reference Voltage	—	Ta=25°C	-1%	1.2	+1%	V
			Ta=-40°C~85°C	-2%	1.2	+2%	
PSRR	Power Supply Rejection Ratio	—	V_{RIPPLE}=1V_{P-P}, f_{RIPPLE}=100Hz	75	—	—	dB
En	Output Noise	—	No load current, f=0.1Hz~10Hz	—	300	—	μV_{RMS}
I_{DRV}	Buffer Driving Capability	—	ΔV_{BGREF}=-1%	1	—	—	mA
I_{SD}	Shutdown Current	—	VBGREN=0	—	—	0.1	μA
t_{START}	Startup Time	2.2V~3.6V	—	—	—	400	μs

Note: 1. All the above parameters are measured under conditions of no load condition unless otherwise described.

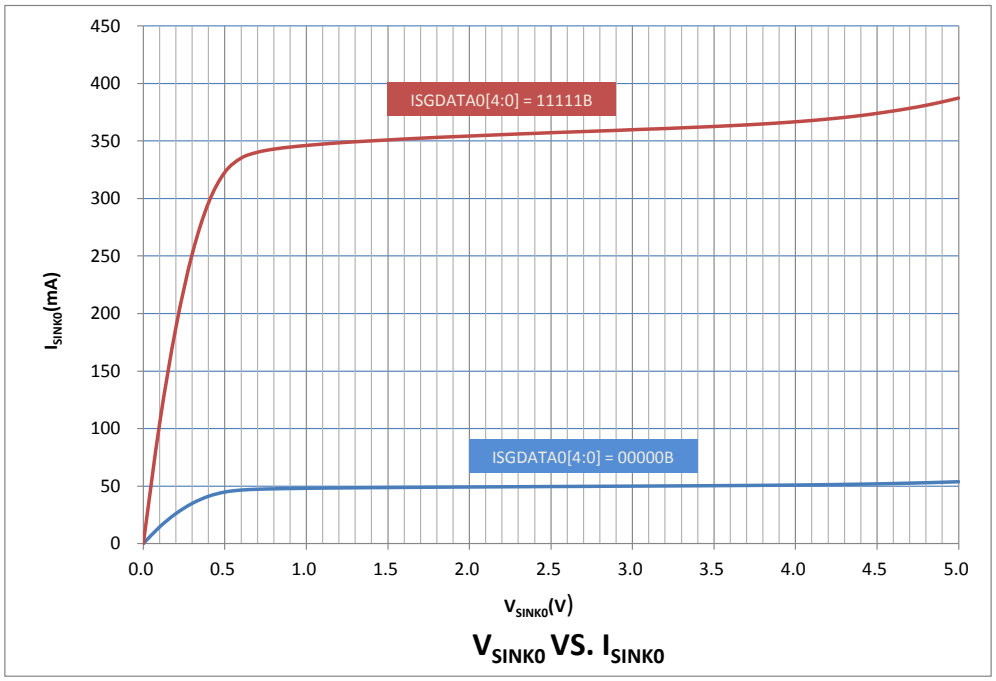
2. A 0.1μF ceramic capacitor should be connected between VDD and GND.

3. The V_{BGREF} voltage can be used as the A/D converter input.

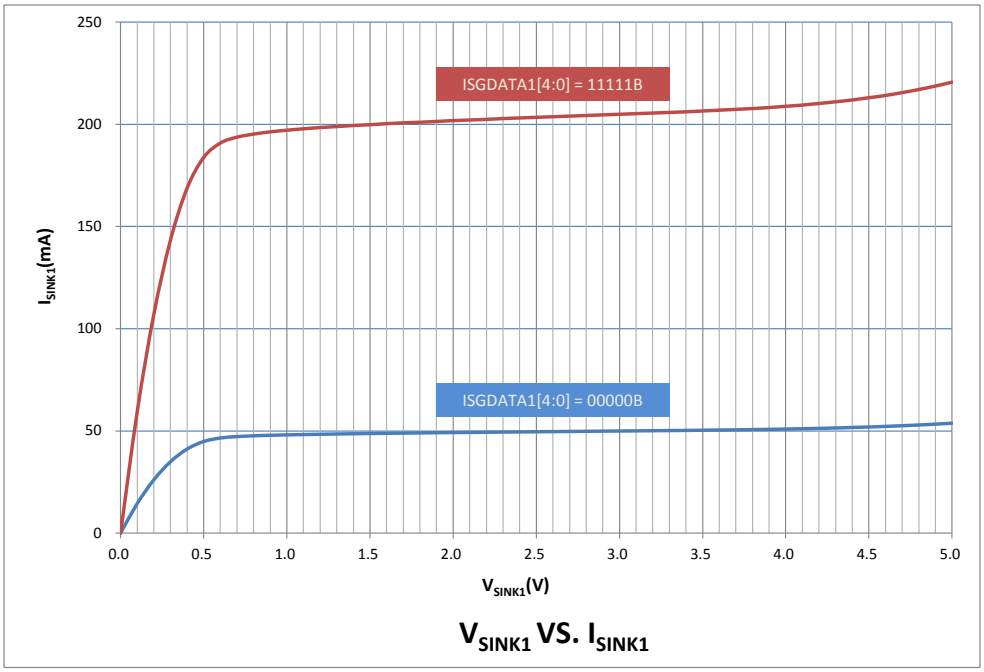
Sink Current Generator Electrical Characteristics

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{SINK0}	Sink Current for ISINK0 Pin	—	Ta=-40°C~85°C, V_{ISINK0}=1.0V~3.6V, ISGDATA0[4:0]=00000B	41	50	59	mA
			Ta=-40°C~85°C, V_{ISINK0}=0.7V~1.0V, ISGDATA0[4:0]=00000B	37.5	50.0	50.0	
			Ta=-40°C~85°C, V_{ISINK0}=1.0V~3.6V, ISGDATA0[4:0]=11111B	295	360	425	
			Ta=-40°C~85°C, V_{ISINK0}=0.7V~1.0V, ISGDATA0[4:0]=11111B	270	360	360	
I_{SINK1}	Sink Current for ISINK1 Pin	—	Ta=-40°C~85°C, V_{ISINK1}=1.0V~3.6V, ISGDATA1[4:0]=00000B	41	50	59	mA
			Ta=-40°C~85°C, V_{ISINK1}=0.7V~1.0V, ISGDATA1[4:0]=00000B	37.5	50.0	50.0	
			Ta=-40°C~85°C, V_{ISINK1}=1.0V~3.6V, ISGDATA1[4:0]=11111B	168	205	242	
			Ta=-40°C~85°C, V_{ISINK1}=0.7V~1.0V, ISGDATA1[4:0]=11111B	154	205	205	

Sink Current Generator Characteristic Curves



I_{SINK0} Characteristic Curve



I_{SINK1} Characteristic Curve

Operational Amplifier Electrical Characteristics

For Smoke Detector AFE

V_{DD}=3V, Ta=-40°C~85°C, typical Ta= 25°C, unless otherwise specified.

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.2	—	3.6	V
I_{OPA}	Operating Current	3V	SDAmBW[1:0]=00B (m=0,1), no load	—	3.0	5.0	μA
			SDAmBW[1:0]=01B (m=0,1), no load	—	10	16	
			SDAmBW[1:0]=10B (m=0,1), no load	—	80	128	
			SDAmBW[1:0]=11B (m=0,1), no load	—	200	320	
V_{OS}	Input Offset Voltage	3V	Without calibration (SDAmOF[5:0]=100000B (m=0,1))	-15	—	15	mV
			With calibration	-4	—	+4	
I_{OS}	Input Offset Current	3V	V_{IN}=1/2V_{CM}	—	1	10	nA
V_{CM}	Common Mode Voltage Range	3V	SDAmBW[1:0]=00,01,10,11 (m=0,1)	V_{SS}	—	V_{DD} -1.4	V
PSRR	Power Supply Rejection Ratio	3V	SDAmBW[1:0]=00,01,10,11 (m=0,1)	50	70	—	dB
CMRR	Common Mode Rejection Ratio	3V	SDAmBW[1:0]=00,01,10,11 (m=0,1)	50	80	—	dB
A_{OL}	Open Loop Gain	3V	SDAmBW[1:0]=00,01,10,11 (m=0,1)	60	80	—	dB
SR	Slew Rate	3V	R_{LOAD}=1MΩ, C_{LOAD}=60pF, SDAmBW[1:0]=00 (m=0,1)	0.5	1.0	—	V/ms
			R_{LOAD}=1MΩ, C_{LOAD}=60pF, SDAmBW[1:0]=01 (m=0,1)	5	10	—	
			R_{LOAD}=1MΩ, C_{LOAD}=60pF, SDAmBW[1:0]=10 (m=0,1)	180	300	—	
			R_{LOAD}=1MΩ, C_{LOAD}=60pF, SDAmBW[1:0]=11 (m=0,1)	600	1200	—	
GBW	Gain Bandwidth	3V	R_{LOAD}=1MΩ, C_{LOAD}=60pF, SDAmBW[1:0]=00 (m=0,1)	2.0	3.5	—	kHz
			R_{LOAD}=1MΩ, C_{LOAD}=60pF, SDAmBW[1:0]=01 (m=0,1)	15	30	—	
			R_{LOAD}=1MΩ, C_{LOAD}=60pF, SDAmBW[1:0]=10 (m=0,1)	250	500	—	
			R_{LOAD}=1MΩ, C_{LOAD}=60pF, SDAmBW[1:0]=11 (m=0,1)	800	1600	—	
V_{OR}	Maximum Output Voltage Range	3V	SDAmBW[1:0]=00,01 (m=0,1) R_{LOAD}=5kΩ to V_{DD}/2	V_{SS} +140	—	V_{DD} -160	mV
			SDAmBW[1:0]=10,11 (m=0,1) R_{LOAD}=5kΩ to V_{DD}/2	V_{SS} +120	—	V_{DD} -140	
I_{SC}	Output Short Circuit Current	3V	R_{LOAD}=5.1Ω, SDAmBW[1:0]=00,01 (m=0,1)	±1.2	±5.0	—	mA
			R_{LOAD}=5.1Ω, SDAmBW[1:0]=10,11 (m=0,1)	±2	±10	—	

Note: These parameters are characterized but not tested.

For Power Line Transceiver

V_{DD}=3V, Ta=-40°C~85°C, typical Ta= 25°C, unless otherwise specified.

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.2	—	3.6	V
I_{OPA}	Operating Current	3V	PLTABW=0, no load	—	80	128	μA
			PLTABW=1, no load	—	200	320	
V_{OS}	Input Offset Voltage	3V	Without calibration (PLTAOF[5:0]=100000B)	-15	—	+15	mV
			With calibration	-4	—	+4	
I_{OS}	Input Offset Current	3V	V_{IN}=1/2V_{CM}	—	1	10	nA
V_{CM}	Common Mode Voltage Range	3V	PLTABW=0, 1	V_{SS}	—	V_{DD} -1.4	V
PSRR	Power Supply Rejection Ratio	3V	PLTABW=0, 1	50	70	—	dB
CMRR	Common Mode Rejection Ratio	3V	PLTABW=0, 1	50	80	—	dB
A_{OL}	Open Loop Gain	3V	PLTABW=0, 1	60	80	—	dB
SR	Slew Rate	3V	R_{LOAD}=1MΩ, C_{LOAD}=60pF, PLTABW=0	180	500	—	V/ms
			R_{LOAD}=1MΩ, C_{LOAD}=60pF, PLTABW=1	600	1800	—	
GBW	Gain Bandwidth	3V	R_{LOAD}=1MΩ, C_{LOAD}=60pF, PLTABW=0	400	600	—	kHz
			R_{LOAD}=1MΩ, C_{LOAD}=60pF, PLTABW=1	1300	2000	—	
V_{OR}	Maximum Output Voltage Range	3V	PLTABW=0, 1, R_{LOAD}=5kΩ to V_{DD}/2	V_{SS} +140	—	V_{DD} -160	mV
I_{SC}	Output Short Circuit Current	3V	R_{LOAD}=5.1Ω, PLTABW=0, 1	±2	±10	—	mA

Note: These parameters are characterized but not tested.

D/A Converter Electrical Characteristics

Ta=-40°C~85°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.2	—	3.6	V
V_{DACO}	Output Voltage Range	—	—	V_{SS}	—	V_{REF}	V
V_{REF}	Reference Voltage	—	—	2	—	V_{DD}	V
I_{DAC}	Additional Current for D/A Converter Enable (DAC0 & DAC1)	3V	—	—	—	12	μA
	Additional Current for D/A Converter Enable (DAC2)	3V	—	—	—	360	μA
t_{ST}	Settling Time	3V	C_{LOAD}=50pF	—	—	5	μs
DNL	Differential Non-Linearity	3V	V_{REF}=V_{DD}	-1	—	+1	LSB
INL	Integral Non-Linearity	3V	V_{REF}=V_{DD}	-1.5	—	+1.5	LSB
R_O	Resistor-String Output Resistor (DAC0 & DAC1)	3V	—	—	1000	—	kΩ
	R2R Output Resistor	3V	—	—	10	—	kΩ
OSRR	Offset Error	3V	—	—	—	6	mV
GERR	Gain Error	3V	—	—	—	12	mV

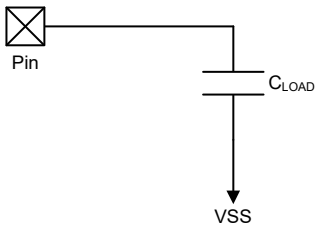
Comparator Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Comparator Operating Voltage	—	—	2.2	—	3.6	V
I_{CMP}	Additional Current for Comparator Enable	—	No load, PLTCmIS[1:0]=00B (m=0, 1)	—	1.7	2.7	μA
			No load, PLTCmIS[1:0]=01B (m=0, 1)	—	14	22	
			No load, PLTCmIS[1:0]=10B (m=0, 1)	—	36	57	
			No load, PLTCmIS[1:0]=11B (m=0, 1)	—	58	92	
V_{CM}	Common Mode Voltage Range	—	—	V_{SS}	—	V_{DD} -1.4	V
t_{RP}	Response Time	3V	With 10mV overdrive(Note) No debounce PLTCmIS[1:0]=00B (m=0,1)	—	—	35	μs
		3V	With 10mV overdrive, (Note) No debounce PLTCmIS[1:0]=01B (m=0,1)	—	—	2.5	
		3V	With 10mV overdrive(Note) No debounce PLTCmIS[1:0]=10B (m=0,1)	—	—	1	
		3V	With 10mV overdrive(Note), No debounce, PLTCmIS[1:0]=11B (m=0,1)	—	—	0.7	
V_{HYS}	Hysteresis	3V	PLTCmHYS[1:0]=00, PLTCmIS[1:0]=00 (m=0,1)	0	0	5	mV
		3V	PLTCmHYS[1:0]=01, PLTCmIS[1:0]=01 (m=0,1)	20	40	60	
		3V	PLTCmHYS[1:0]=10, PLTCmIS[1:0]=10 (m=0,1)	50	100	150	
		3V	PLTCmHYS[1:0]=11, PLTCmIS[1:0]=11 (m=0,1)	80	160	240	

Note: All the above parameters are measured under condition of Comp. input voltage=(V_{DD}-1.4)/2 and remain constant.

Load Conditon: C_{LOAD}=50pF



Load Condition

RF Electrical Characteristics

Ta=25°C, V_{DD}=3.3V, f_{XTAL}=16MHz, FSK modulation with matching circuit and low/high pass filter, RF output is powered by V_{DD} (3.3V), unless otherwise specified.

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
T_{OP}	Operating Temperature	—	-40	—	85	°C
V_{DD}	Supply Voltage	—	2.2	3.3	3.6	V
Digital I/Os						
V_{IH}	High Level Input Voltage	—	0.7×V_{DD}	—	V_{DD}	V
V_{IL}	Low Level Input Voltage	—	0	—	0.3×V_{DD}	V
V_{OH}	High Level Output Voltage	I_{OH}=-5mA	0.8×V_{DD}	—	V_{DD}	V
V_{OL}	Low Level Output Voltage	I_{OL}=5mA	0	—	0.2×V_{DD}	V
Current Consumption						
I_{Sleep}	Deep Sleep Mode Current Consumption	—	—	0.4	1.0	μA
I_{IL}	Idle Mode Current Consumption	LIRC on, X'tal off	—	1.8	—	μA
	Light Sleep Mode Current Consumption	X'tal on	—	0.55	—	mA
$I_{Standby}$	Standby Mode Current Consumption @ 315/433MHz	X'tal on, Synthesizer on	—	2.2	—	mA
	Standby Mode Current Consumption @ 868/915MHz		—	3.0	—	
I_{RX} or I_{TX}	433MHz Band Current Consumption	RX mode @ 50Kbps	—	4.2	—	mA
		RX mode @ 250Kbps	—	4.6	—	
		TX mode @ 0dBm P_{OUT}	—	14	—	
		TX mode @ 10dBm P_{OUT}	—	22	—	
		TX mode @ 13dBm P_{OUT}	—	30	—	
	868MHz Band Current Consumption	RX mode @ 50Kbps	—	5.5	—	mA
		RX mode @ 250Kbps	—	6.1	—	
		TX mode @ 0dBm P_{OUT}	—	15	—	
		TX mode @ 10dBm P_{OUT}	—	24	—	
		TX mode @ 13dBm P_{OUT}	—	32	—	
RF Characteristics						
f_{RF}	RF Frequency Band	315MHz band	—	315	—	MHz
		433MHz band	—	433.92	—	
		470~510MHz band	—	490	—	
		868MHz band	—	868.3	—	
		915MHz band	—	915	—	
DR	Data Rate	GFSK modulation	2	—	250	Kbps
Transmitter						
P_{OUT}	TX Output Power	433MHz band	0	—	13	dBm
		868MHz band	0	—	13	
S.E.$_{TX}$	TX Spurious Emission (P_{OUT}=10dBm)	f<1GHz	—	—	-36	dBm
		47MHz<f<74MHz	—	—	-54	
		87.5MHz<f<118MHz				
		174MHz<f<230MHz				
		470MHz<f<862MHz				
		2nd, 3rd Harmonic	—	—	-30	

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
Receiver						
$t_{ST,RX}$	RX Settling Time	Light Sleep mode to RX mode	—	150	—	μs
P_{Sens}	433MHz RX Sensitivity @ BER=0.1%	2Kbps (f_{DEV}=8kHz)	—	-119	—	dBm
		10Kbps (f_{DEV}=40kHz)	—	-112	—	
		50Kbps (f_{DEV}=18.75kHz)	—	-109	—	
		125Kbps (f_{DEV}=46.875kHz)	—	-104	—	
		250Kbps (f_{DEV}=93.75kHz)	—	-100	—	
	868MHz RX Sensitivity @ BER=0.1%	2Kbps (f_{DEV}=8kHz)	—	-118	—	dBm
		10Kbps (f_{DEV}=40kHz)	—	-112	—	
		50Kbps (f_{DEV}=18.75kHz)	—	-108	—	
		125Kbps (f_{DEV}=46.875kHz)	—	-104	—	
		250Kbps (f_{DEV}=93.75kHz)	—	-100	—	
$P_{IN,max}$	Maxmum Input Power	@ BER<0.1%	—	—	10	dBm
IR	Image Rejection	—	—	25	—	dB
$S.E._{.RX}$	RX Spurious	25MHz~1GHz	—	—	-57	dBm
		Above 1GHz	—	—	-47	
	RSSI Range	AGC on	-110	—	-10	dBm
LO Characteristics						
f_{LO}	RF Frequency Coverage Range	315MHz band	290	—	335	MHz
		433MHz band	415	—	490	
		470~510MHz band	470	—	510	
		868MHz band	830	—	1000	
		915MHz band	870	—	1050	
f_{STEP}	LO Frequency Resolution	—	—	—	1	kHz
PN_{LO}	315MHz Phase Noise	@ 100kHz offset	—	-90	—	dBc/Hz
		@ 1MHz offset	—	-113	—	
	433MHz Phase Noise	@ 100kHz offset	—	-89	—	
		@ 1MHz offset	—	-110	—	
	868MHz Phase Noise	@ 100kHz offset	—	-86	—	
		@ 1MHz offset	—	-109	—	
	915MHz Phase Noise	@ 100kHz offset	—	-86	—	
		@ 1MHz offset	—	-109	—	
Crystal Oscillator						
f_{XTAL}	X'tal Frequency	—	—	16	—	MHz
ESR	X'tal Equivalent Series Resistance	—	—	—	100	Ω
C_{LOAD}	X'tal Capacitor Load	—	12	16	20	pF
TOL	X'tal Tolerance	—	-20	—	+20	dBc/Hz
t_{SU}	X'tal Startup Time	49US XO	—	—	1	ms

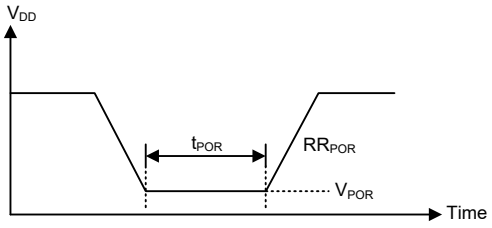
Note: 1. Depend on crystal property.

2. This is the total tolerance including Initial tolerance, Crystal loading, Aging and and Temperature dependence.

Power-on Reset Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{POR}	V_{DD} Start Voltage to Ensure Power-on Reset	—	—	—	—	100	mV
RR_{POR}	V_{DD} Rising Rate to Ensure Power-on Reset	—	—	0.035	—	—	V/ms
t_{POR}	Minimum Time for V_{DD} Stays at V_{POR} to Ensure Power-on Reset	—	—	1	—	—	ms



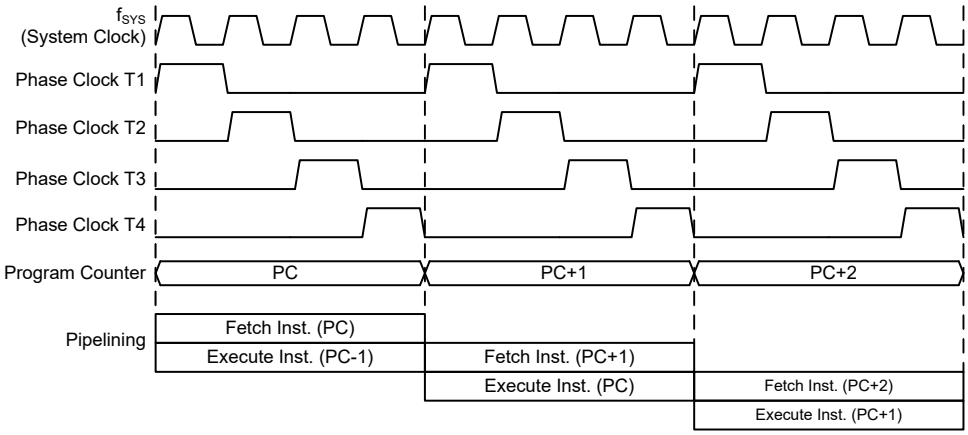
System Architecture

A key factor in the high-performance features of the Holtek range of microcontrollers is attributed to their internal system architecture. The device takes advantage of the usual features found within RISC microcontrollers providing increased speed of operation and enhanced performance. The pipelining scheme is implemented in such a way that instruction fetching and instruction execution are overlapped, hence instructions are effectively executed in one or two cycles for most of the standard or extended instructions respectively. The exceptions to these are branch or call instructions which need one more cycle. An 8-bit wide ALU is used in practically all instruction set operations, which carries out arithmetic operations, logic operations, rotation, increment, decrement, branch decisions, etc. The internal data path is simplified by moving data through the Accumulator and the ALU. Certain internal registers are implemented in the Data Memory and can be directly or indirectly addressed. The simple addressing methods of these registers along with additional architectural features ensure that a minimum of external components is required to provide a functional I/O and A/D control system with maximum reliability and flexibility. This makes the device suitable for low-cost, high-volume production for controller applications.

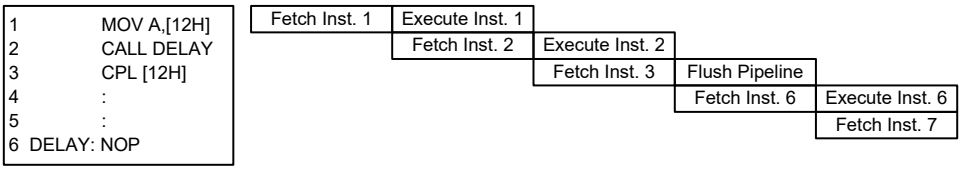
Clocking and Pipelining

The main system clock, derived from either a HIRC or LIRC oscillator is subdivided into four internally generated non-overlapping clocks, T1~T4. The Program Counter is incremented at the beginning of the T1 clock during which time a new instruction is fetched. The remaining T2~T4 clocks carry out the decoding and execution functions. In this way, one T1~T4 clock cycle forms one instruction cycle. Although the fetching and execution of instructions takes place in consecutive instruction cycles, the pipelining structure of the microcontroller ensures that instructions are effectively executed in one instruction cycle. The exception to this are instructions where the contents of the Program Counter are changed, such as subroutine calls or jumps, in which case the instruction will take one more instruction cycle to execute.

For instructions involving branches, such as jump or call instructions, two machine cycles are required to complete instruction execution. An extra cycle is required as the program takes one cycle to first obtain the actual jump or call address and then another cycle to actually execute the branch. The requirement for this extra cycle should be taken into account by programmers in timing sensitive applications.



System Clocking and Pipelining



Instruction Fetching

Program Counter

During program execution, the Program Counter is used to keep track of the address of the next instruction to be executed. It is automatically incremented by one each time an instruction is executed except for instructions, such as "JMP" or "CALL" that demands a jump to a non-consecutive Program Memory address. Only the lower 8 bits, known as the Program Counter Low Register, are directly addressable by the application program.

When executing instructions requiring jumps to non-consecutive addresses such as a jump instruction, a subroutine call, interrupt or reset, etc., the microcontroller manages program control by loading the required address into the Program Counter. For conditional skip instructions, once the condition has been met, the next instruction, which has already been fetched during the present instruction execution, is discarded and a dummy cycle takes its place while the correct instruction is obtained.

Program Counter	
High Byte	**Low Byte (PCL)**
PC11~PC8	PCL7~PCL0

Program Counter

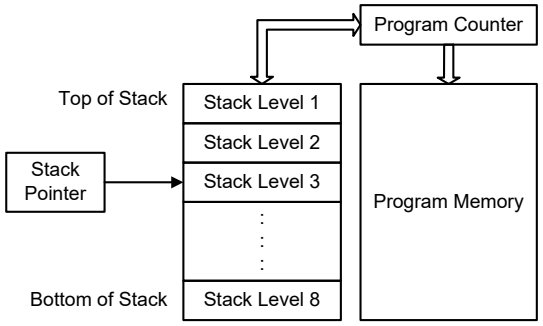
The lower byte of the Program Counter, known as the Program Counter Low register, PCL, is available for program control and is a readable and writeable register. By transferring data directly into this register, a short program jump can be executed directly; however, as only this low byte is available for manipulation, the jumps are limited to the present page of memory that is 256 locations. When such program jumps are executed it should also be noted that a dummy cycle will be inserted. Manipulating the PCL register may cause program branching, so an extra cycle is needed to pre-fetch.

Stack

This is a special part of the memory which is used to save the contents of the Program Counter only. The stack has multiple levels and neither part of the data nor part of the program space, and is neither readable nor writeable. The activated level is indexed by the Stack Pointer, and is neither readable nor writeable. At a subroutine call or interrupt acknowledge signal, the contents of the Program Counter are pushed onto the stack. At the end of a subroutine or an interrupt routine, signaled by a return instruction, RET or RETI, the Program Counter is restored to its previous value from the stack. After a device reset, the Stack Pointer will point to the top of the stack.

If the stack is full and an enabled interrupt takes place, the interrupt request flag will be recorded but the acknowledge signal will be inhibited. When the Stack Pointer is decremented, by RET or RETI, the interrupt will be serviced. This feature prevents stack overflow allowing the programmer to use the structure more easily. However, when the stack is full, a CALL subroutine instruction can still be executed which will result in a stack overflow. Precautions should be taken to avoid such cases which might cause unpredictable program branching.

If the stack is overflow, the first Program Counter save in the stack will be lost.



Arithmetic and Logic Unit – ALU

The arithmetic-logic unit or ALU is a critical area of the microcontroller that carries out arithmetic and logic operations of the instruction set. Connected to the main microcontroller data bus, the ALU receives related instruction codes and performs the required arithmetic or logical operations after which the result will be placed in the specified register. As these ALU calculation or operations may result in carry, borrow or other status changes, the status register will be correspondingly updated to reflect these changes. The ALU supports the following functions:

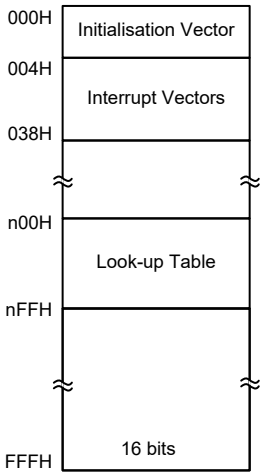
- Arithmetic operations:
  ADD, ADDM, ADC, ADCM, SUB, SUBM, SBC, SBCM, DAA,
  LADD, LADDM, LADC, LADCM, LSUB, LSUBM, LSBC, LSBCM, LDAA
- Logic operations:
  AND, OR, XOR, ANDM, ORM, XORM, CPL, CPLA,
  LAND, LOR, LXOR, LANDM, LORM, LXORM, LCPL, LCPLA
- Rotation:
  RRA, RR, RRCA, RRC, RLA, RL, RLCA, RLC,
  LRR, LRRA, LRRCA, LRRC, LRLA, LRL, LRLCA, LRLC
- Increment and Decrement:
  INCA, INC, DECA, DEC,
  LINCA, LINC, LDECA, LDEC
- Branch decision:
  JMP, SZ, SZA, SNZ, SIZ, SDZ, SIZA, SDZA, CALL, RET, RETI,
  LSZ, LSZA, LSNZ, LSIZ, LSDZ, LSIZA, LSDZA

Flash Program Memory

The Program Memory is the location where the user code or program is stored. For the device the Program Memory is Flash type, which means it can be programmed and re-programmed a large number of times, allowing the user the convenience of code modification on the same device. By using the appropriate programming tools, the Flash device offers users the flexibility to conveniently debug and develop their applications while also offering a means of field programming and updating.

Structure

The Program Memory has a capacity of 4K×16 bits. The Program Memory is addressed by the Program Counter and also contains data, table information and interrupt entries. Table data, which can be setup in any location within the Program Memory, is addressed by a separate table pointer register.



Program Memory Structure

Special Vectors

Within the Program Memory, certain locations are reserved for the reset and interrupts. The location 000H is reserved for use by the device reset for program initialisation. After a device reset is initiated, the program will jump to this location and begin execution.

Look-up Table

Any location within the Program Memory can be defined as a look-up table where programmers can store fixed data. To use the look-up table, the table pointer must first be setup by placing the address of the look up data to be retrieved in the table pointer registers, TBLP and TBHP. These registers define the total address of the look-up table.

After setting up the table pointer, the table data can be retrieved from the Program Memory using the corresponding table read instruction such as “TABRD [m]” or “TABRDL [m]” respectively when the memory [m] is located in sector 0. If the memory [m] is located in other sectors, the data can be retrieved from the program memory using the corresponding extended table read instruction such as “LTABRD [m]” or “LTABRDL [m]” respectively. When the instruction is executed, the lower order table byte from the Program Memory will be transferred to the user defined Data Memory register [m] as specified in the instruction. The higher order table data byte from the Program Memory will be transferred to the TBLH special register. Any unused bits in this transferred higher order byte will be read as “0”.

The accompanying diagram illustrates the addressing data flow of the look-up table.

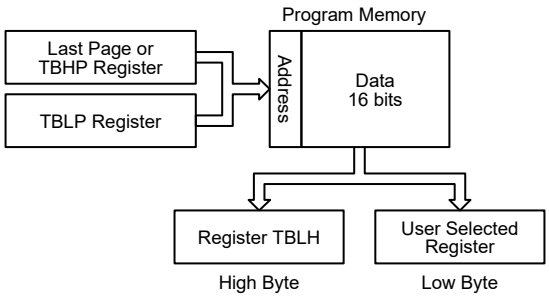


Table Program Example

The following example shows how the table pointer and table data is defined and retrieved from the microcontroller. This example uses raw table data located in the Program Memory which is stored there using the ORG statement. The value at this ORG statement is "0F00H" which refers to the start address of the last page within the 4K Program Memory of the device. The table pointer low byte register is setup here to have an initial value of "06H". This will ensure that the first data read from the data table will be at the Program Memory address "0F06H" or 6 locations after the start of the last page. Note that the value for the table pointer is referenced to the first address specified by TBLP and TBHP if the "TABRD [m]" or "LTABRD [m]" instruction is being used. The high byte of the table data which in this case is equal to zero will be transferred to the TBLH register automatically when the "TABRD [m]" or "LTABRD [m]" instruction is executed.

Because the TBLH register is a read/write register and can be restored, care should be taken to ensure its protection if both the main routine and Interrupt Service Routine use table read instructions. If using the table read instructions, the Interrupt Service Routines may change the value of the TBLH and subsequently cause errors if used again by the main routine. As a rule it is recommended that simultaneous use of the table read instructions should be avoided. However, in situations where simultaneous use cannot be avoided, the interrupts should be disabled prior to the execution of any main routine table-read instructions. Note that all table related instructions require two instruction cycles to complete their operation.

Table Read Program Example

```
tempreg1 db ?   ; temporary register #1
tempreg2 db ?   ; temporary register #2
:
mov a,06h       ; initialise low table pointer - note that this address is referenced
mov tblp,a      ; to the last page or the page that tbhp pointed
mov a,0Fh       ; initialise high table pointer
mov tbhp,a
:
tabrd tempreg1  ; transfers value in table referenced by table pointer data at program
                ; memory address "0F06H" transferred to tempreg1 and TBLH
dec tblp        ; reduce value of table pointer by one
tabrd tempreg2  ; transfers value in table referenced by table pointer data at program
                ; memory address "0F05H" transferred to tempreg2 and TBLH in this example
                ; the data "1AH" is transferred to tempreg1 and data "0FH" to register
                ; tempreg2
:
org 0F00h       ; sets initial address of program memory
dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
:
```

In Circuit Programming – ICP

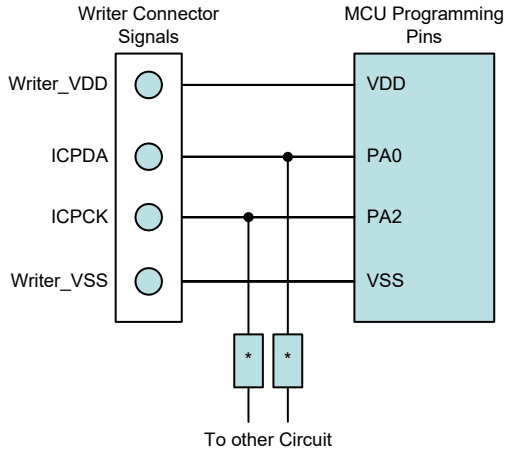
The provision of Flash type Program Memory provides the user with a means of convenient and easy upgrades and modifications to their programs on the same device. As an additional convenience, Holtek has provided a means of programming the microcontroller in-circuit using a 4-pin interface. This provides manufacturers with the possibility of manufacturing their circuit boards complete with a programmed or un-programmed microcontroller, and then programming or upgrading the program at a later stage. This enables product manufacturers to easily keep their manufactured products supplied with the latest program releases without removal and re-insertion of the device.

The Holtek Flash MCU to Writer Programming Pin correspondence table is as follows:

Holtek Writer Pins	MCU Programming Pins	Pin Description
ICPDA	PA0	Programming Serial Data/Address
ICPCK	PA2	Programming Clock
VDD	VDD	Power Supply
VSS	VSS	Ground

The Program Memory can be programmed serially in-circuit using this 4-wire interface. Data is downloaded and uploaded serially on a single pin with an additional line for the clock. Two additional lines are required for the power supply. The technical details regarding the in-circuit programming of the device is beyond the scope of this document and will be supplied in supplementary literature.

During the programming process, the user must take care of the PA0 and PA2 pins for data and clock programming purposes to ensure that no other outputs are connected to these two pins.



Note: * may be resistor or capacitor. The resistance of * must be greater than 1kΩ or the capacitance of * must be less than 1nF.

On Chip Debug Support – OCDS

There is an EV chip which is used to emulate the real MCU device. The EV chip device also provides an "On-Chip Debug" function to debug the real MCU device during the development process. The EV chip and the real MCU device are almost functionally compatible except for "On-Chip Debug" function. Users can use the EV chip device to emulate the real chip device behavior by connecting the OCDSDA and OCDSCK pins to the Holtek HT-IDE development tools. The OCDSDA pin is the OCDS Data/Address input/output pin while the OCDSCK pin is the OCDS clock input pin. When users use the EV chip for debugging, other functions which are shared with the OCDSDA and OCDSCK pins in the device will have no effect in the EV chip. However, the two OCDS pins which are pin-shared with the ICP programming pins are still used as the Flash Memory programming pins for ICP. For more detailed OCDS information, refer to the corresponding document named "Holtek e-Link for 8-bit MCU OCDS User's Guide".

Holtek e-Link Pins	EV Chip Pins	Pin Description
OCDSDA	OCDSDA	On-Chip Debug Support Data/Address input/output
OCDSCK	OCDSCK	On-Chip Debug Support Clock input
VDD	VDD	Power Supply
VSS	VSS	Ground

Data Memory

The Data Memory is a volatile area of 8-bit wide RAM internal memory and is the location where temporary information is stored.

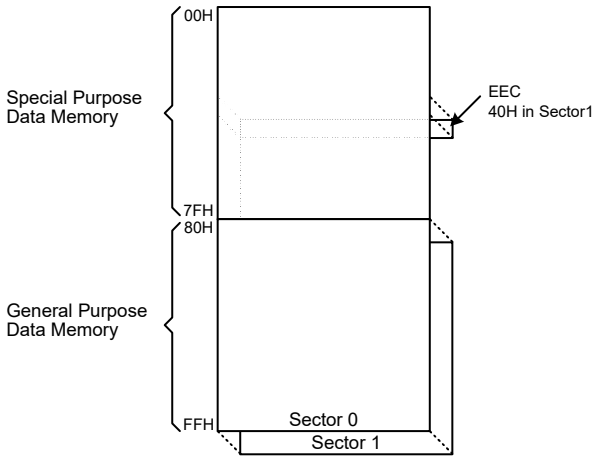
Categorized into two types, the first of these is an area of RAM where special function registers are located. These registers have fixed locations and are necessary for correct operation of the device. Many of these registers can be read from and written to directly under program control, however, some remain protected from user manipulation. The second area of Data Memory is reserved for general purpose use. All locations within this area are read and write accessible under program control.

Switching between the different Data Memory sectors is achieved by properly setting the Memory Pointers to correct value.

Structure

The Data Memory is subdivided into several sectors, all of which are implemented in 8-bit wide RAM. Each of the Data Memory Sector is categorized into two types, the special Purpose Data Memory and the General Purpose Data Memory. The address range of the Special Purpose Data Memory for the device is from 00H to 7FH while the General Purpose Data Memory address range is from 80H to FFH.

Special Purpose Data Memory	General Purpose Data Memory	
Located Sectors	Capacity	Sector: Address
0: 00H~7FH 1: 40H (EEC only)	256×8	0: 80H~FFH 1: 80H~FFH



Data Memory Structure

Data Memory Addressing

For the device that supports the extended instructions, there is no Bank Pointer for Data Memory addressing. The desired Sector is pointed by the MP1H or MP2H register and the certain Data Memory address in the selected sector is specified by the MP1L or MP2L register when using indirect addressing access.

Direct Addressing can be used in all sectors using the corresponding instruction which can address all available data memory space. For the accessed data memory which is located in any data memory sectors except sector 0, the extended instructions can be used to access the data memory instead of using the indirect addressing access. The main difference between standard instructions and extended instructions is that the data memory address “m” in the extended instructions has 9 valid bits, the high byte indicates a sector and the low byte indicates a specific address within the sector.

General Purpose Data Memory

All microcontroller programs require an area of read/write memory where temporary data can be stored and retrieved for use later. It is this area of RAM memory that is known as General Purpose Data Memory. This area of Data Memory is fully accessible by the user programing for both reading and writing operations. By using the bit operation instructions individual bits can be set or reset under program control giving the user a large range of flexibility for bit manipulation in the Data Memory.

Special Purpose Data Memory

This area of Data Memory is where registers, necessary for the correct operation of the microcontroller, are stored. Most of the registers are both readable and writeable but some are protected and are readable only, the details of which are located under the relevant Special Function Register section. Note that for locations that are unused, any read instruction to these addresses will return the value “00H”.

	Sector 0	Sector 1
00H	IAR0	
01H	MP0	
02H	IAR1	
03H	MP1L	
04H	MP1H	
05H	ACC	
06H	PCL	
07H	TBLP	
08H	TBLH	
09H	TBHP	
0AH	STATUS	
0BH	VBGRC	
0CH	IAR2	
0DH	MP2L	
0EH	MP2H	
0FH	RSTFC	
10H	TB0C	
11H	TB1C	
12H	SCC	
13H	HIRCC	
14H	PA	
15H	PAC	
16H	PAPU	
17H	PAWU	
18H	PB	
19H	PBC	
1AH	PBPU	
1BH	SLEDC	
1CH		
1DH	PSCR	
1EH	LVDC	
1FH		
20H	SDSW	
21H	SDPGAC0	
22H	SDPGAC1	
23H	SDA0C	
24H	SDA0VOS	
25H	SDA1C	
26H	SDA1VOS	
27H	STMC0	
28H	STMC1	
29H	STMDL	
2AH	STMDH	
2BH	STMAL	
2CH	STMAH	
2DH	SADOL	
2EH	SADOH	
2FH	SADC0	
30H	SADC1	
31H	PLTSW	
32H	PLTDACC	
33H	PLTDA0L	
34H	PLTDA1L	
35H	PLTDA2L	
36H	PLTC0C	
37H	PLTC0VOS	
38H	PLTC1C	
39H	PLTC1VOS	
3AH	PLTCHYC	
3BH	PLTAC	
3CH	PLTAVOS	
3DH	WDTC	
3EH	EEA	
3FH	EED	

	Sector 0	Sector 1
40H		EEC
41H	SIMC0	
42H	SIMC1/UUCR1	
43H	SIMC2/SIMA/UUCR2	
44H	SIMD/UTXR_RXR	
45H	SIMTOC/UBRG	
46H	UUSR	
47H	INTEG	
48H	INTC0	
49H	INTC1	
4AH	INTC2	
4BH	INTC3	
4CH	PAS0	
4DH	PAS1	
4EH	PBS0	
4FH	PBS1	
50H	IFS0	
51H		
52H	PTMC0	
53H	PTMC1	
54H	PTMC2	
55H	PTMDL	
56H	PTMDH	
57H	PTMAL	
58H	PTMAH	
59H	PTMBL	
5AH	PTMBH	
5BH	PTMRPL	
5CH	PTMRPH	
5DH	ISGENC	
5EH	ISGDATA0	
5FH	ISGDATA1	
60H		
61H		
62H		
63H		
64H		
65H		
66H		
67H		
68H		
69H		
6AH		
6BH		
6CH		
6DH		
6EH		
6FH		
70H		
71H		
72H		
73H		
74H		
75H		
76H		
77H		
78H		
79H		
7AH		
7BH		
7CH		
7DH		
7EH		
7FH		

: Unused, read as 00H

Special Purpose Data Memory Structure

Special Function Register Description

Most of the Special Function Register details will be described in the relevant functional sections however several registers require a separate description in this section.

Indirect Addressing Registers – IAR0, IAR1, IAR2

The Indirect Addressing Registers, IAR0, IAR1 and IAR2, although having their locations in normal RAM register space, do not actually physically exist as normal registers. The method of indirect addressing for RAM data manipulation uses these Indirect Addressing Registers and Memory Pointers, in contrast to direct memory addressing, where the actual memory address is specified. Actions on the IAR0, IAR1 and IAR2 registers will result in no actual read or write operation to these registers but rather to the memory location specified by their corresponding Memory Pointers, MP0, MP1L/MP1H or MP2L/MP2H. Acting as a pair, IAR0 and MP0 can together access data only from Sector 0 while the IAR1 register together with the MP1L/MP1H register pair and IAR2 register together with the MP2L/MP2H register pair can access data from any Data Memory Sector. As the Indirect Addressing Registers are not physically implemented, reading the Indirect Addressing Registers will return a result of "00H" and writing to the registers will result in no operation.

Memory Pointers – MP0, MP1L, MP1H, MP2L, MP2H

Five Memory Pointers, known as MP0, MP1L, MP1H, MP2L, MP2H, are provided. These Memory Pointers are physically implemented in the Data Memory and can be manipulated in the same way as normal registers providing a convenient way with which to address and track data. When any operation to the relevant Indirect Addressing Registers is carried out, the actual address that the microcontroller is directed to is the address specified by the related Memory Pointer. MP0, together with Indirect Addressing Register, IAR0, are used to access data from Sector 0, while MP1L/MP1H together with IAR1 and MP2L/MP2H together with IAR2 are used to access data from all sectors according to the corresponding MP1H or MP2H register. Direct Addressing can be used in all sectors using the corresponding instruction which can address all available data memory space.

The following example shows how to clear a section of four Data Memory locations already defined as locations adres1 to adres4.

Indirect Addressing Program Example 1

```
data .section ´data´
adres1   db ?
adres2   db ?
adres3   db ?
adres4   db ?
block    db ?
code .section at 0 ´code´
org 00h
start:
     mov a, 04h              ; setup size of block
     mov block, a
     mov a, offset adres1    ; Accumulator loaded with first RAM address
     mov mp0, a              ; setup memory pointer with first RAM address
loop:
     clr IAR0                ; clear the data at address defined by MP0
     inc mp0                 ; increment memory pointer
     sdz block               ; check if last memory location has been cleared
     jmp loop
continue:
```

Indirect Addressing Program Example 2

```
data .section ´data´
adres1   db ?
adres2   db ?
adres3   db ?
adres4   db ?
block    db ?
code .section at 0 ´code´
org 00h
start:
     mov a, 04h              ; setup size of block
     mov block, a
     mov a, 01h              ; setup the memory sector
     mov mp1h, a
     mov a, offset adres1    ; Accumulator loaded with first RAM address
     mov mp1l, a             ; setup memory pointer with first RAM address
loop:
     clr IAR1                ; clear the data at address defined by MP1L
     inc mp1l                ; increment memory pointer MP1L
     sdz block               ; check if last memory location has been cleared
     jmp loop
continue:
```

The important point to note here is that in the example shown above, no reference is made to specific Data Memory addresses.

Direct Addressing Program Example using extended instructions

```
data .section ´data´
temp  db ?
code .section at 0 ´code´
org 00h
start:
     lmov a, [m]             ; move [m] data to acc
     lsub a, [m+1]           ; compare [m] and [m+1] data
     snz  c                  ; [m]>[m+1]?
     jmp  continue           ; no
     lmov a, [m]             ; yes, exchange [m] and [m+1] data
     mov  temp, a
     lmov a, [m+1]
     lmov [m], a
     mov  a, temp
     lmov [m+1], a
continue:
```

Note: here "m" is a data memory address located in any data memory sectors. For example, m=1F0H, it indicates address 0F0H in Sector 1.

Accumulator – ACC

The Accumulator is central to the operation of any microcontroller and is closely related with operations carried out by the ALU. The Accumulator is the place where all intermediate results from the ALU are stored. Without the Accumulator it would be necessary to write the result of each calculation or logical operation such as addition, subtraction, shift, etc., to the Data Memory resulting in higher programming and timing overheads. Data transfer operations usually involve the temporary storage function of the Accumulator; for example, when transferring data between one user-defined register and another, it is necessary to do this by passing the data through the Accumulator as no direct transfer between two registers is permitted.

Program Counter Low Register – PCL

To provide additional program control functions, the low byte of the Program Counter is made accessible to programmers by locating it within the Special Purpose area of the Data Memory. By manipulating this register, direct jumps to other program locations are easily implemented. Loading a value directly into this PCL register will cause a jump to the specified Program Memory location, however, as the register is only 8-bit wide, only jumps within the current Program Memory page are permitted. When such operations are used, note that a dummy cycle will be inserted.

Look-up Table Registers – TBLP, TBHP, TBLH

These three special function registers are used to control operation of the look-up table which is stored in the Program Memory. TBLP and TBHP are the table pointers and indicate the location where the table data is located. Their value must be setup before any table read commands are executed. Their value can be changed, for example using the "INC" or "DEC" instructions, allowing for easy table data pointing and reading. TBLH is the location where the high order byte of the table data is stored after a table read data instruction has been executed. Note that the lower order table data byte is transferred to a user defined location.

Status Register – STATUS

This 8-bit register contains the SC flag, CZ flag, zero flag (Z), carry flag (C), auxiliary carry flag (AC), overflow flag (OV), power down flag (PDF), and watchdog time-out flag (TO). These arithmetic/logical operation and system management flags are used to record the status and operation of the microcontroller.

With the exception of the TO and PDF flags, bits in the status register can be altered by instructions like most other registers. Any data written into the status register will not change the TO or PDF flag. In addition, operations related to the status register may give different results due to the different instruction operations. The TO flag can be affected only by a system power-up, a WDT time-out or by executing the "CLR WDT" or "HALT" instruction. The PDF flag is affected only by executing the "HALT" or "CLR WDT" instruction or during a system power-up.

The Z, OV, AC, C, SC and CZ flags generally reflect the status of the latest operations.

- C is set if an operation results in a carry during an addition operation or if a borrow does not take place during a subtraction operation; otherwise C is cleared. C is also affected by a rotate through carry instruction.
- AC is set if an operation results in a carry out of the low nibbles in addition, or no borrow from the high nibble into the low nibble in subtraction; otherwise AC is cleared.
- Z is set if the result of an arithmetic or logical operation is zero; otherwise Z is cleared.
- OV is set if an operation results in a carry into the highest-order bit but not a carry out of the highest-order bit, or vice versa; otherwise OV is cleared.
- PDF is cleared by a system power-up or executing the "CLR WDT" instruction. PDF is set by executing the "HALT" instruction.
- TO is cleared by a system power-up or executing the "CLR WDT" or "HALT" instruction. TO is set by a WDT time-out.
- CZ is the operational result of different flags for different instructions. Refer to register definitions for more details.
- SC is the result of the "XOR" operation which is performed by the OV flag and the MSB of the current instruction operation result.

In addition, on entering an interrupt sequence or executing a subroutine call, the status register will not be pushed onto the stack automatically. If the contents of the status registers are important and if the subroutine can corrupt the status register, precautions must be taken to correctly save it.

- **STATUS Register**

Bit	7	6	5	4	3	2	1	0
Name	SC	CZ	TO	PDF	OV	Z	AC	C
R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W
POR	x	x	0	0	x	x	x	x

"x": unknown

Bit 7 **SC**: The result of the "XOR" operation which is performed by the OV flag and the MSB of the instruction operation result.

Bit 6 **CZ**: The operational result of different flags for different instructions.
For SUB/SUBM/LSUB/LSUBM instructions, the CZ flag is equal to the Z flag.
For SBC/SBCM/LSBC/LSBCM instructions, the CZ flag is the "AND" operation result which is performed by the previous operation CZ flag and current operation zero flag.
For other instructions, the CZ flag will not be affected.

Bit 5 **TO**: Watchdog Time-out flag
0: After power up or executing the "CLR WDT" or "HALT" instruction
1: A watchdog time-out occurred.

Bit 4 **PDF**: Power down flag
0: After power up or executing the "CLR WDT" instruction
1: By executing the "HALT" instruction

Bit 3 **OV**: Overflow flag
0: No overflow
1: An operation results in a carry into the highest-order bit but not a carry out of the highest-order bit or vice versa.

Bit 2 **Z**: Zero flag
0: The result of an arithmetic or logical operation is not zero
1: The result of an arithmetic or logical operation is zero

Bit 1 **AC**: Auxiliary flag
0: No auxiliary carry
1: An operation results in a carry out of the low nibbles in addition, or no borrow from the high nibble into the low nibble in subtraction

Bit 0 **C**: Carry flag
0: No carry-out
1: An operation results in a carry during an addition operation or if a borrow does not take place during a subtraction operation
The "C" flag is also affected by a rotate through carry instruction.

EEPROM Data Memory

The device contains an area of internal EEPROM Data Memory. EEPROM is by its nature a non-volatile form of re-programmable memory, with data retention even when its power supply is removed. By incorporating this kind of data memory, a whole new host of application possibilities are made available to the designer. The availability of EEPROM storage allows information such as product identification numbers, calibration values, specific user data, system setup data or other product information to be stored directly within the product microcontroller. The process of reading and writing data to the EEPROM memory has been reduced to a very trivial affair.

EEPROM Data Memory Structure

The device EEPROM Data Memory capacity is 64×8 bits. Unlike the Program Memory and RAM Data Memory, the EEPROM Data Memory is not directly mapped into memory space and is therefore not directly addressable in the same way as the other types of memory. Read and Write operations to the EEPROM are carried out in single byte operations using an address and a data register in Sector 0 and a single control register in Sector 1.

EEPROM Registers

Three registers control the overall operation of the internal EEPROM Data Memory. These are the address register, EEA, the data register, EED and a single control register, EEC. As both the EEA and EED registers are located in Sector 0, they can be directly accessed in the same way as any other Special Function Register. The EEC register however, being located in Sector 1, can only be read from or written to indirectly using the MP1L/MP1H or MP2L/MP2H Memory Pointer and Indirect Addressing Register, IAR1/IAR2. Because the EEC control register is located at address 40H in Sector 1, the MP1L or MP2L Memory Pointer must first be set to the value 40H and the MP1H or MP2H Memory Pointer high byte set to the value, 01H, before any operations on the EEC register are executed.

Register Name	Bit							
	7	6	5	4	3	2	1	0
EEA	—	—	EEA5	EEA4	EEA3	EEA2	EEA1	EEA0
EED	D7	D6	D5	D4	D3	D2	D1	D0
EEC	—	—	—	—	WREN	WR	RDEN	RD

EEPROM Register List

- **EEA Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	EEA5	EEA4	EEA3	EEA2	EEA1	EEA0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5~0 **EEA5~EEA0**: Data EEPROM address
Data EEPROM address bit 5 ~ bit 0

• **EED Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **D7~D0**: Data EEPROM data
Data EEPROM data bit 7 ~ bit 0

• **EEC Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	WREN	WR	RDEN	RD
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4 Unimplemented, read as "0"

Bit 3 **WREN**: Data EEPROM Write Enable
0: Disable
1: Enable
This is the Data EEPROM Write Enable Bit which must be set high before Data EEPROM write operations are carried out. Clearing this bit to zero will inhibit Data EEPROM write operations.

Bit 2 **WR**: EEPROM Write Control
0: Write cycle has finished
1: Activate a write cycle
This is the Data EEPROM Write Control Bit and when set high by the application program will activate a write cycle. This bit will be automatically reset to zero by the hardware after the write cycle has finished. Setting this bit high will have no effect if the WREN has not first been set high.

Bit 1 **RDEN**: Data EEPROM Read Enable
0: Disable
1: Enable
This is the Data EEPROM Read Enable Bit which must be set high before Data EEPROM read operations are carried out. Clearing this bit to zero will inhibit Data EEPROM read operations.

Bit 0 **RD**: EEPROM Read Control
0: Read cycle has finished
1: Activate a read cycle
This is the Data EEPROM Read Control Bit and when set high by the application program will activate a read cycle. This bit will be automatically reset to zero by the hardware after the read cycle has finished. Setting this bit high will have no effect if the RDEN has not first been set high.

Note: 1. The WREN, WR, RDEN and RD cannot be set high at the same time in one instruction. The WR and RD cannot be set high at the same time.
2. Ensure that the f_{SUB} clock is stable before executing the write operation.
3. Ensure that the write operation is totally complete before changing the EEC register content.

Reading Data from the EEPROM

To read data from the EEPROM, the read enable bit, RDEN, in the EEC register must first be set high to enable the read function. The EEPROM address of the data to be read must then be placed in the EEA register. If the RD bit in the EEC register is now set high, a read cycle will be initiated. Setting the RD bit high will not initiate a read operation if the RDEN bit has not been set. When the read cycle terminates, the RD bit will be automatically cleared to zero, after which the data can be read from the EED register. The data will remain in the EED register until another read or write operation is executed. The application program can poll the RD bit to determine when the data is valid for reading.

Writing Data to the EEPROM

The EEPROM address of the data to be written must first be placed in the EEA register and the data placed in the EED register. To write data to the EEPROM, the write enable bit, WREN, in the EEC register must first be set high to enable the write function. After this, the WR bit in the EEC register must be immediately set high to initiate a write cycle. These two instructions must be executed in two consecutive instruction cycles. The global interrupt bit EMI should also first be cleared before implementing any write operations, and then set again after the write cycle has started. Note that setting the WR bit high will not initiate a write cycle if the WREN bit has not been set. As the EEPROM write cycle is controlled using an internal timer whose operation is asynchronous to microcontroller system clock, a certain time will elapse before the data will have been written into the EEPROM. Detecting when the write cycle has finished can be implemented either by polling the WR bit in the EEC register or by using the EEPROM interrupt. When the write cycle terminates, the WR bit will be automatically cleared to zero by the microcontroller, informing the user that the data has been written to the EEPROM. The application program can therefore poll the WR bit to determine when the write cycle has ended.

Write Protection

Protection against inadvertent write operation is provided in several ways. After the device is powered-on the Write Enable bit in the control register will be cleared preventing any write operations. Also at power-on the Memory Pointer high byte register, MP1H or MP2H, will be reset to zero, which means that Data Memory Sector 0 will be selected. As the EEPROM control register is located in Sector 1, this adds a further measure of protection against spurious write operations. During normal program operation, ensuring that the Write Enable bit in the control register is cleared will safeguard against incorrect write operations.

EEPROM Interrupt

The EEPROM write interrupt is generated when an EEPROM write cycle has ended. The EEPROM interrupt must first be enabled by setting the DEE bit in the relevant interrupt register. When an EEPROM write cycle ends, the DEF request flag will be set. If the global and EEPROM interrupts are enabled and the stack is not full, a jump to the Data EEPROM Interrupt vector will take place. When the interrupt is serviced the EEPROM interrupt flag will be automatically reset. More details can be obtained in the Interrupt section.

Programming Considerations

Care must be taken that data is not inadvertently written to the EEPROM. Protection can be enhanced by ensuring that the Write Enable bit is normally cleared to zero when not writing. Also the Memory Pointer high byte register, MP1H or MP2H, could be normally cleared to zero as this would inhibit access to Sector 1 where the EEPROM control register exists. Although certainly not necessary, consideration might be given in the application program to the checking of the validity of new write data by a simple read back process.

When writing data the WR bit must be set high immediately after the WREN bit has been set high, to ensure the write cycle executes correctly. The global interrupt bit EMI should also be cleared before a write cycle is executed and then re-enabled after the write cycle starts. Note that the device should not enter the IDLE or SLEEP mode until the EEPROM read or write operation is totally complete. Otherwise, the EEPROM read or write operation will fail.

Programming Examples

Reading data from the EEPROM – polling method

```
MOV  A, EEPROM_ADRES      ; user defined address
MOV  EEA, A
MOV  A, 040H              ; setup memory pointer MP1L
MOV  MP1L, A              ; MP1L points to EEC register
MOV  A, 01H               ; setup memory pointer MP1H
MOV  MP1H, A
SET  IAR1.1               ; set RDEN bit, enable read operations
SET  IAR1.0               ; start Read Cycle - set RD bit
BACK:
SZ   IAR1.0               ; check for read cycle end
JMP  BACK
CLR  IAR1                 ; disable EEPROM read if no more read operations are required
CLR  MP1H
MOV  A, EED               ; move read data to register
MOV  READ_DATA, A
```

Note: For each read operation, the address register should be re-specified followed by setting the RD bit high to activate a read cycle even if the target address is consecutive.

Writing Data to the EEPROM – polling method

```
MOV  A, EEPROM_ADRES      ; user defined address
MOV  EEA, A
MOV  A, EEPROM_DATA       ; user defined data
MOV  EED, A
MOV  A, 040H              ; setup memory pointer MP1L
MOV  MP1L, A              ; MP1L points to EEC register
MOV  A, 01H               ; setup memory pointer MP1H
MOV  MP1H, A
CLR  EMI
SET  IAR1.3               ; set WREN bit, enable write operations
SET  IAR1.2               ; start Write Cycle - set WR bit – executed immediately
                          ; after setting WREN bit
SET  EMI
BACK:
SZ   IAR1.2               ; check for write cycle end
JMP  BACK
CLR  MP1H
```

Oscillators

Various oscillator options offer the user a wide range of functions according to their various application requirements. The flexible features of the oscillator functions ensure that the best optimisation can be achieved in terms of speed and power saving. Oscillator selections and operation are selected only through the application program by using some control registers.

Oscillator Overview

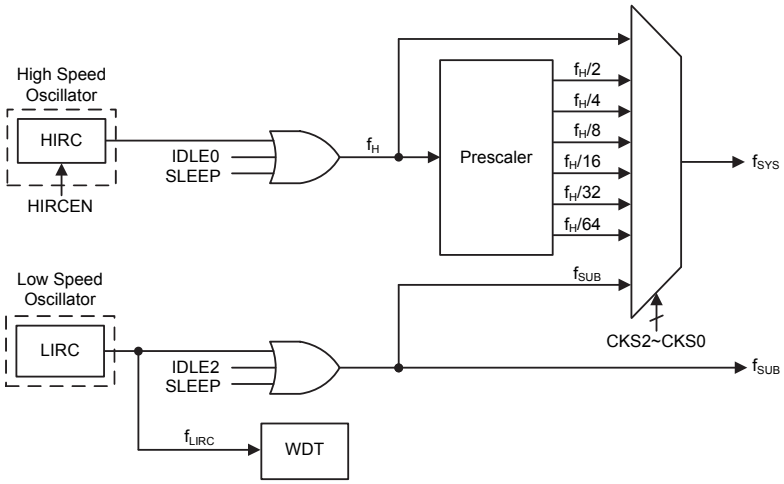
In addition to being the source of the main system clock the oscillators also provide clock sources for the Watchdog Timer and Time Base Interrupts. Two fully integrated internal oscillators, requiring no external components, are provided to form a wide range of both fast and slow system oscillators. All oscillator options all can be selected through register programming. The higher frequency oscillators provide higher performance but carry with it the disadvantage of higher power requirements, while the opposite is of course true for the lower frequency oscillators. With the capability of dynamically switching between fast and slow system clock, the device have the flexibility to optimize the performance/power ratio, a feature especially important in power sensitive portable applications.

Type	Name	Frequency
Internal High Speed RC	HIRC	2/4/8MHz
Internal Low Speed RC	LIRC	32kHz

Oscillator Types

System Clock Configurations

There are two oscillator sources, one high speed oscillator and one low speed oscillator. The high speed system clocks is sourced from the internal 2/4/8MHz RC oscillator, HIRC. The low speed oscillator is the internal 32kHz RC oscillator, LIRC. Selecting whether the low or high speed oscillator is used as the system oscillator is implemented using the CKS2~CKS0 bits in the SCC register and as the system clock can be dynamically selected.



System Clock Configurations

Internal High Speed RC Oscillator – HIRC

The internal RC oscillator is a fully integrated system oscillator requiring no external components. The internal RC oscillator has three fixed frequencies of 2MHz, 4MHz and 8MHz. Device trimming during the manufacturing process and the inclusion of internal frequency compensation circuits are used to ensure that the influence of the power supply voltage, temperature and process variations on the oscillation frequency are minimised.

Internal 32kHz Oscillator – LIRC

The Internal 32kHz System Oscillator is also a fully integrated RC oscillator with a typical frequency of 32kHz, requiring no external components for its implementation. Device trimming during the manufacturing process and the inclusion of internal frequency compensation circuits are used to ensure that the influence of the power supply voltage, temperature and process variations on the oscillation frequency are minimised.

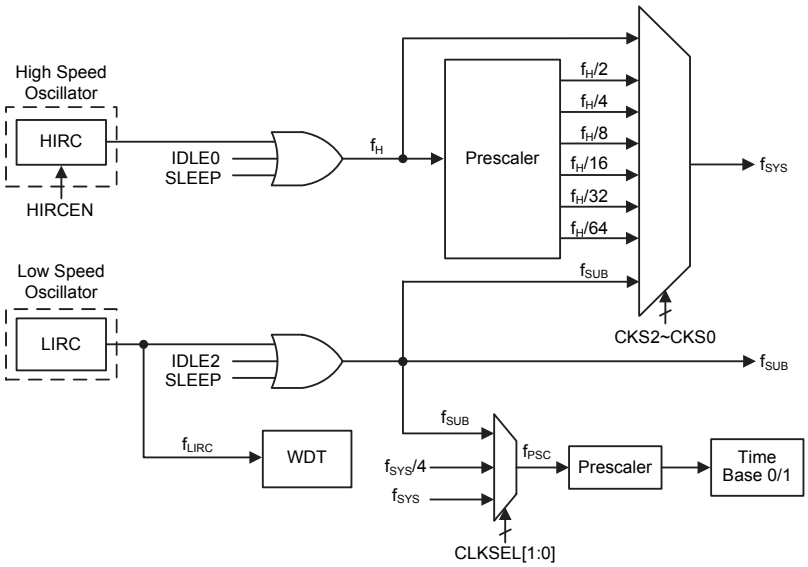
Operating Modes and System Clocks

Present day applications require that their microcontrollers have high performance but often still demand that they consume as little power as possible, conflicting requirements that are especially true in battery powered portable applications. The fast clocks required for high performance will by their nature increase current consumption and of course vice versa, lower speed clocks reduce current consumption. As Holtek has provided the device with both high and low speed clock sources and the means to switch between them dynamically, the user can optimise the operation of their microcontroller to achieve the best performance/power ratio.

System Clocks

The device has many different clock sources for both the CPU and peripheral function operation. By providing the user with a wide range of clock options using register programming, a clock system can be configured to obtain maximum application performance.

The main system clock, can come from a high frequency, f_H, or low frequency, f_{SUB}, source, and is selected using the CKS2~CKS0 bits in the SCC register. The high speed system clock is sourced from the HIRC oscillator, while the low speed system clock source is sourced from the internal clock f_{SUB} which is sourced by the LIRC oscillator. The other choice, which is a divided version of the high speed system oscillator has a range of $f_H/2$~$f_H/64$.



Device Clock Configurations

Note: When the system clock source f_{SYS} is switched to f_{SUB} from f_H, the high speed oscillator will stop to conserve the power or continue to oscillate to provide the clock source, f_H~$f_H/64$, for peripheral circuit to use, which is determined by configuring the corresponding high speed oscillator enable control bit.

System Operation Modes

There are six different modes of operation for the microcontroller, each one with its own special characteristics and which can be chosen according to the specific performance and power requirements of the application. There are two modes allowing normal operation of the microcontroller, the FAST Mode and SLOW Mode. The remaining four modes, the SLEEP, IDLE0, IDLE1 and IDLE2 Mode are used when the microcontroller CPU is switched off to conserve power.

Operation Mode	CPU	Register Setting			f_{SYS}	f_H	f_{SUB}	f_{LIRC}
		FHIDEN	FSIDEN	CKS2~CKS0				
FAST	On	x	x	000~110	f_H~f_H/64	On	On	On
SLOW	On	x	x	111	f_{SUB}	On/Off (1)	On	On
IDLE0	Off	0	1	000~110	Off	Off	On	On
				111	On			
IDLE1	Off	1	1	xxx	On	On	On	On
IDLE2	Off	1	0	000~110	On	On	Off	On
				111	Off			
SLEEP	Off	0	0	xxx	Off	Off	Off	On (2)

"x": Don't care

Note: 1. The f_H clock will be switched on or off by configuring the corresponding oscillator enable bit in the SLOW mode.
2. The f_{LIRC} clock will be switched on since the WDT function is always enabled even in the SLEEP mode.

FAST Mode

This is one of the main operating modes where the microcontroller has all of its functions operational and where the system clock is provided by the high speed oscillator. This mode operates allowing the microcontroller to operate normally with a clock source from the HIRC high speed oscillator. The high speed oscillator will however first be divided by a ratio ranging from 1 to 64, the actual ratio being selected by the CKS2~CKS0 bits in the SCC register. Although a high speed oscillator is used, running the microcontroller at a divided clock ratio reduces the operating current.

SLOW Mode

This is also a mode where the microcontroller operates normally although now with a slower speed clock source. The clock source used will be from f_{SUB}. The f_{SUB} clock is derived from the LIRC oscillator.

SLEEP Mode

The SLEEP Mode is entered when a HALT instruction is executed and when the FHIDEN and FSIDEN bit both are low. In the SLEEP mode the CPU will be stopped. The f_{SUB} clock provided to the peripheral function will also be stopped. However the f_{LIRC} clock still continues to operate since the WDT function is enabled.

IDLE0 Mode

The IDLE0 Mode is entered when a HALT instruction is executed and when the FHIDEN bit in the SCC register is low and the FSIDEN bit in the SCC register is high. In the IDLE0 Mode the CPU will be switched off but the low speed oscillator will be turned on to drive some peripheral functions.

IDLE1 Mode

The IDLE1 Mode is entered when a HALT instruction is executed and when the FHIDEN bit in the SCC register is high and the FSIDEN bit in the SCC register is high. In the IDLE1 Mode the CPU will be switched off but both the high and low speed oscillators will be turned on to provide a clock source to keep some peripheral functions operational.

IDLE2 Mode

The IDLE2 Mode is entered when a HALT instruction is executed and when the FHIDEN bit in the SCC register is high and the FSIDEN bit in the SCC register is low. In the IDLE2 Mode the CPU will be switched off but the high speed oscillator will be turned on to provide a clock source to keep some peripheral functions operational.

Control Registers

The registers, SCC and HIRCC, are used to control the system clock and the HIRC oscillator configurations.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SCC	CKS2	CKS1	CKS0	—	—	—	FHIDEN	FSIDEN
HIRCC	—	—	—	—	HIRC1	HIRC0	HIRCF	HIRCEN

System Operating Mode Control Register List

• SCC Register

Bit	7	6	5	4	3	2	1	0
Name	CKS2	CKS1	CKS0	—	—	—	FHIDEN	FSIDEN
R/W	R/W	R/W	R/W	—	—	—	R/W	R/W
POR	0	0	0	—	—	—	0	0

Bit 7~5 **CKS2~CKS0**: System clock selection
000: f_H
001: $f_H/2$
010: $f_H/4$
011: $f_H/8$
100: $f_H/16$
101: $f_H/32$
110: $f_H/64$
111: f_{SUB}
These three bits are used to select which clock is used as the system clock source. In addition to the system clock source directly derived from f_H or f_{SUB}, a divided version of the high speed system oscillator can also be chosen as the system clock source.

Bit 4~2 Unimplemented, read as "0"

Bit 1 **FHIDEN**: High Frequency oscillator control when CPU is switched off
0: Disable
1: Enable
This bit is used to control whether the high speed oscillator is activated or stopped when the CPU is switched off by executing an "HALT" instruction.

Bit 0 **FSIDEN**: Low Frequency oscillator control when CPU is switched off
0: Disable
1: Enable
This bit is used to control whether the low speed oscillator is activated or stopped when the CPU is switched off by executing an "HALT" instruction. The LIRC oscillator is controlled by this bit together with the WDT function enable control. If this bit is cleared to 0 but the WDT function is enabled, the f_{LIRC} clock will also be enabled.

• HIRCC Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	HIRC1	HIRC0	HIRCF	HIRCEN
R/W	—	—	—	—	R/W	R/W	R	R/W
POR	—	—	—	—	0	0	0	1

Bit 7~4 Unimplemented, read as "0"

Bit 3~2 **HIRC1~HIRC0**: HIRC frequency selection
00: 2MHz
01: 4MHz
10: 8MHz
11: 2MHz
When the HIRC oscillator is enabled, the HIRC frequency is changed by changing these two bits, the clock frequency will automatically be changed after the HIRCF flag is set to 1.

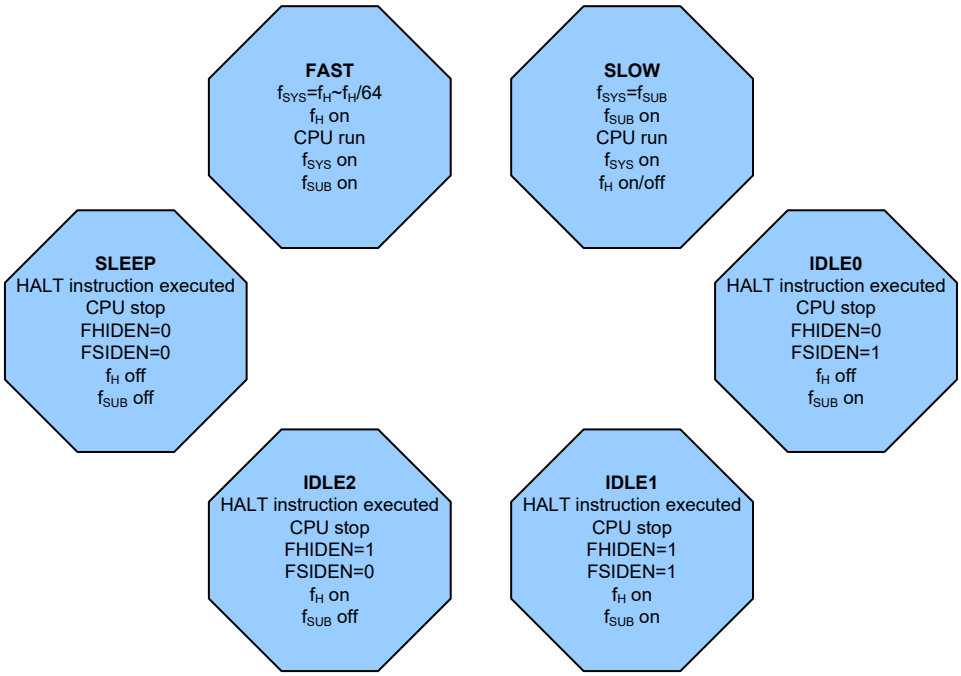
Bit 1 **HIRCF**: HIRC oscillator stable flag
0: HIRC unstable
1: HIRC stable
This bit is used to indicate whether the HIRC oscillator is stable or not. When the HIRCEN bit is set to 1 to enable the HIRC oscillator or the HIRC frequency selection is changed by application program, the HIRCF bit will first be cleared to 0 and then set to 1 after the HIRC oscillator is stable.

Bit 0 **HIRCEN**: HIRC oscillator enable control
0: Disable
1: Enable

Operating Mode Switching

The device can switch between operating modes dynamically allowing the user to select the best performance/power ratio for the present task in hand. In this way microcontroller operations that do not require high performance can be executed using slower clocks thus requiring less operating current and prolonging battery life in portable applications.

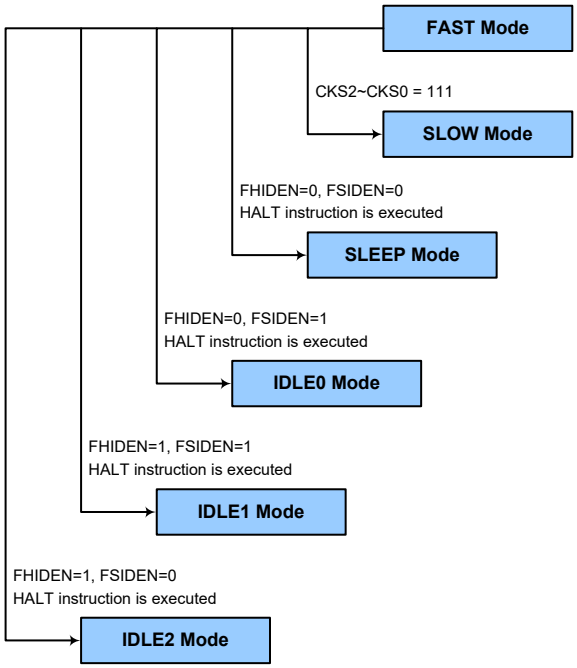
In simple terms, Mode Switching between the FAST Mode and SLOW Mode is executed using the CKS2~CKS0 bits in the SCC register while Mode Switching from the FAST/SLOW Modes to the SLEEP/IDLE Modes is executed via the HALT instruction. When an HALT instruction is executed, whether the device enter the IDLE Mode or the SLEEP Mode is determined by the condition of the FHIDEN and FSIDEN bits in the SCC register.



FAST Mode to SLOW Mode Switching

When running in the FAST Mode, which uses the high speed system oscillator, and therefore consumes more power, the system clock can switch to run in the SLOW Mode by setting the CKS2~CKS0 bits to "111" in the SCC register. This will then use the low speed system oscillator which will consume less power. Users may decide to do this for certain operations which do not require high performance and can subsequently reduce power consumption.

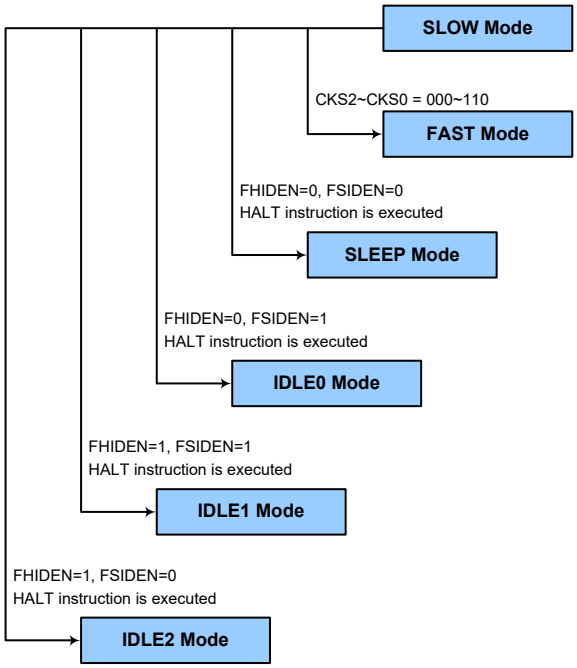
The SLOW Mode is sourced from the LIRC oscillator and therefore requires this oscillator to be stable before full mode switching occurs.



SLOW Mode to FAST Mode Switching

In SLOW mode the system clock is derived from f_{SUB}. When system clock is switched back to the FAST mode from f_{SUB}, the CKS2~CKS0 bits should be set to "000"~"110" and then the system clock will respectively be switched to $f_H \sim f_H/64$.

However, if f_H is not used in SLOW mode and thus switched off, it will take some time to re-oscillate and stabilise when switching to the FAST mode from the SLOW Mode. This is monitored using the HIRCF bit in the HIRCC register. The time duration required for the high speed system oscillator stabilization is specified in the System Start Up Time Characteristics.



Entering the SLEEP Mode

There is only one way for the device to enter the SLEEP Mode and that is to execute the "HALT" instruction in the application program with both the FHIDEN and FSIDEN bits in the SCC register equal to "0". In this mode all the clocks and functions will be switched off except the WDT function. When this instruction is executed under the conditions described above, the following will occur:

- The system clock will be stopped and the application program will stop at the "HALT" instruction.
- The Data Memory contents and registers will maintain their present condition.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.
- The WDT will be cleared and resume counting as the WDT function is always enabled.

Entering the IDLE0 Mode

There is only one way for the device to enter the IDLE0 Mode and that is to execute the "HALT" instruction in the application program with the FHIDEN bit in the SCC register equal to "0" and the FSIDEN bit in the SCC register equal to "1". When this instruction is executed under the conditions described above, the following will occur:

- The f_H clock will be stopped and the application program will stop at the "HALT" instruction, but the f_{SUB} clock will be on.
- The Data Memory contents and registers will maintain their present condition.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.
- The WDT will be cleared and resume counting as the WDT function is always enabled.

Entering the IDLE1 Mode

There is only one way for the device to enter the IDLE1 Mode and that is to execute the "HALT" instruction in the application program with both the FHIDEN and FSIDEN bits in the SCC register equal to "1". When this instruction is executed under the conditions described above, the following will occur:

- The f_H and f_{SUB} clocks will be on but the application program will stop at the "HALT" instruction.
- The Data Memory contents and registers will maintain their present condition.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.
- The WDT will be cleared and resume counting as the WDT function is always enabled.

Entering the IDLE2 Mode

There is only one way for the device to enter the IDLE2 Mode and that is to execute the "HALT" instruction in the application program with the FHIDEN bit in the SCC register equal to "1" and the FSIDEN bit in the SCC register equal to "0". When this instruction is executed under the conditions described above, the following will occur:

- The f_H clock will be on but the f_{SUB} clock will be off and the application program will stop at the "HALT" instruction.
- The Data Memory contents and registers will maintain their present condition.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.
- The WDT will be cleared and resume counting as the WDT function is always enabled.

Standby Current Considerations

As the main reason for entering the SLEEP or IDLE Mode is to keep the current consumption of the device to as low a value as possible, perhaps only in the order of several micro-amps in the IDLE1 and IDLE2 Mode, there are other considerations which must also be taken into account by the circuit designer if the power consumption is to be minimised. Special attention must be made to the I/O pins on the device. All high-impedance input pins must be connected to either a fixed high or low level as any floating input pins could create internal oscillations and result in increased current consumption. This also applies to the device which has different package types, as there may be unbonded pins. These must either be setup as outputs or if setup as inputs must have pull-high resistors connected.

Care must also be taken with the loads, which are connected to I/O pins, which are setup as outputs. These should be placed in a condition in which minimum current is drawn or connected only to external circuits that do not draw current, such as other CMOS inputs. Also note that additional standby current will also be required if the LIRC oscillator has enabled.

In the IDLE1 and IDLE2 Mode the high speed oscillator is on, if the peripheral function clock source is derived from the high speed oscillator, the additional standby current will also be perhaps in the order of several hundred micro-amps.

Wake-up

To minimise power consumption the device can enter the SLEEP or any IDLE Mode, where the CPU will be switched off. However, when the device is woken up again, it will take a considerable time for the original system oscillator to restart, stabilise and allow normal operation to resume.

After the system enters the SLEEP or IDLE Mode, it can be woken up from one of various sources listed as follows:

- An external falling edge on Port A
- A system interrupt
- A WDT overflow

When the device executes the "HALT" instruction, it will enter the IDLE or SLEEP mode and the PDF flag will be set high. The PDF flag is cleared to 0 if the device experiences a system power-up or executes the clear Watchdog Timer instruction.

If the system is woken up by a WDT overflow, a Watchdog Timer Time-out reset will be initiated and the TO flag will be set to 1. The TO flag is set high if a WDT time-out occurs, and causes a wake-up that only resets the Program Counter and Stack Pointer, the other flags remain in their original status.

Each pin on Port A can be setup using the PAWU register to permit a negative transition on the pin to wake-up the system. When a pin wake-up occurs, the program will resume execution at the instruction following the "HALT" instruction.

If the system is woken up by an interrupt, then two possible situations may occur. The first is where the related interrupt is disabled or the interrupt is enabled but the stack is full, in which case the program will resume execution at the instruction following the "HALT" instruction. In this situation, the interrupt which woke-up the device will not be immediately serviced, but will rather be serviced later when the related interrupt is finally enabled or when a stack level becomes free. The other situation is where the related interrupt is enabled and the stack is not full, in which case the regular interrupt response takes place. If an interrupt request flag is set high before entering the SLEEP or IDLE Mode, the wake-up function of the related interrupt will be disabled.

Watchdog Timer

The Watchdog Timer is provided to prevent program malfunctions or sequences from jumping to unknown locations, due to certain uncontrollable external events such as electrical noise.

Watchdog Timer Clock Source

The Watchdog Timer clock source is provided by the internal clock, f_{LIRC} which is sourced from the LIRC oscillator. The LIRC internal oscillator has an approximate frequency of 32kHz and this specified internal clock period can vary with V_{DD}, temperature and process variations. The Watchdog Timer source clock is then subdivided by a ratio of 2^8 to 2^{18} to give longer timeouts, the actual value being chosen using the WS2~WS0 bits in the WDTC register.

Watchdog Timer Control Register

A single register, WDTC, controls the required timeout period as well as the enable and reset MCU operation.

- **WDTC Register**

Bit	7	6	5	4	3	2	1	0
Name	WE4	WE3	WE2	WE1	WE0	WS2	WS1	WS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	1	0	1	0	0	1	1

Bit 7~3 **WE4~WE0**: WDT function software control
01010/10101: Enable
Others: Reset MCU
When these bits are changed by the environmental noise or software setting to reset the microcontroller, the reset operation will be activated after a delay time, t_{SRESET} and the WRF bit in the RSTFC register will be set high.

Bit 2~0 **WS2~WS0**: WDT time-out period selection
000: $2^8/f_{LIRC}$
001: $2^{10}/f_{LIRC}$
010: $2^{12}/f_{LIRC}$
011: $2^{14}/f_{LIRC}$
100: $2^{15}/f_{LIRC}$
101: $2^{16}/f_{LIRC}$
110: $2^{17}/f_{LIRC}$
111: $2^{18}/f_{LIRC}$
These three bits determine the division ratio of the watchdog timer source clock, which in turn determines the time-out period.

- **RSTFC Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	LVRF	—	WRF
R/W	—	—	—	—	—	R/W	—	R/W
POR	—	—	—	—	—	x	—	0

"x": unknown

Bit 7~3 Unimplemented, read as "0"

Bit 2 **LVRF**: LVR function reset flag
Described elsewhere

Bit 1 Unimplemented, read as "0"

Bit 0 **WRF**: WDTC register software reset flag
0: Not occurred
1: Occurred
This bit is set to 1 by the WDTC register software reset and cleared to zero by the application program. Note that this bit can be cleared to 0 only by the application program.

Watchdog Timer Operation

The Watchdog Timer operates by providing a device reset when its timer overflows. This means that in the application program and during normal operation the user has to strategically clear the Watchdog Timer before it overflows to prevent the Watchdog Timer from executing a reset. This is done using the clear watchdog instruction. If the program malfunctions for whatever reason, jumps to an unknown location, or enters an endless loop, the clear instruction will not be executed in the correct manner, in which case the Watchdog Timer will overflow and reset the device. There are five bits, WE4~WE0, in the WDTC register to offer the enable control and reset control of the Watchdog Timer. The WDT function will be enabled if the WE4~WE0 bits are equal to 01010B or 10101B. If the WE4~WE0 bits are set to any other values, other than 01010B and 10101B, it will reset the device after a delay time, t_{SRESET}. After power on these bits will have a value of 01010B.

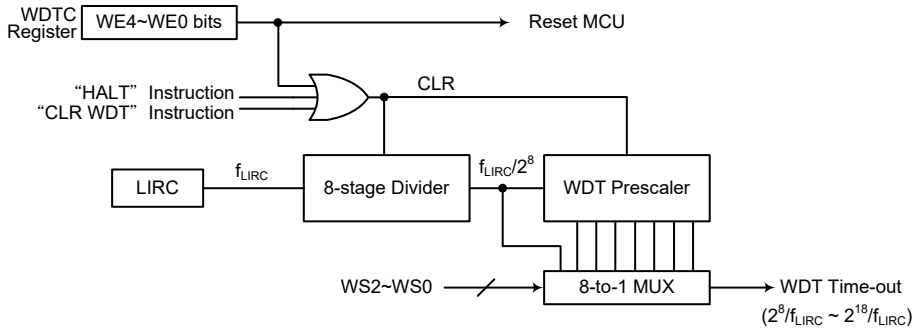
WE4~WE0 Bits	WDT Function
01010B or 10101B	Enable
Any other values	Reset MCU

Watchdog Timer Enable/Reset Control

Under normal program operation, a Watchdog Timer time-out will initialise a device reset and set the status bit TO high. However, if the system is in the SLEEP or IDLE Mode, when a Watchdog Timer time-out occurs, the TO bit in the status register will be set high and only the Program Counter and Stack Pointer will be reset. Three methods can be adopted to clear the contents of the Watchdog Timer. The first is a WDTC software reset, which means a certain value except 01010B and 10101B written into the WE4~WE0 bits, the second is using the Watchdog Timer software clear instruction and the third is via a HALT instruction.

There is only one method of using software instruction to clear the Watchdog Timer. That is to use the single "CLR WDT" instruction to clear the WDT.

The maximum time out period is when the 2^{18} division ratio is selected. As an example, with a 32kHz LIRC oscillator as its source clock, this will give a maximum watchdog period of around 8 seconds for the 2^{18} division ratio, and a minimum timeout of 8ms for the 2^8 division ration.



Watchdog Timer

Reset and Initialisation

A reset function is a fundamental part of any microcontroller ensuring that the device can be set to some predetermined condition irrespective of outside parameters. The most important reset condition is after power is first applied to the microcontroller. In this case, internal circuitry will ensure that the microcontroller, after a short delay, will be in a well-defined state and ready to execute the first program instruction. After this power-on reset, certain important internal registers will be set to defined states before the program commences. One of these registers is the Program Counter, which will be reset to zero forcing the microcontroller to begin program execution from the lowest Program Memory address.

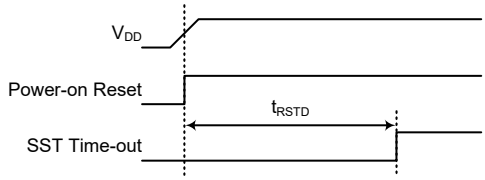
In addition to the power-on reset, another reset exists in the form of a Low Voltage Reset, LVR, where a full reset is implemented in situations where the power supply voltage falls below a certain threshold. Another type of reset is when the Watchdog Timer overflows and resets the microcontroller. All types of reset operations result in different register conditions being setup.

Reset Functions

There are several ways in which a microcontroller reset can occur, through events occurring internally:

Power-on Reset

The most fundamental and unavoidable reset is the one that occurs after power is first applied to the microcontroller. As well as ensuring that the Program Memory begins execution from the first memory address, a power-on reset also ensures that certain other registers are preset to known conditions. All the I/O port and port control registers will power up in a high condition ensuring that all pins will be first set to inputs.



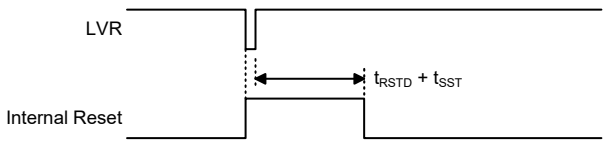
Power-on Reset Timing Chart

Low Voltage Reset – LVR

The microcontrollers contain a low voltage reset circuit in order to monitor the supply voltage of the device and provide an MCU reset when the value falls below a certain predefined level.

The LVR function is always enabled in normal operation with a specific LVR voltage V_{LVR}. For the device the V_{LVR} value is fixed at 2.1V. If the supply voltage of the device drop to within a range of 0.9V~V_{LVR} such as might occur when changing the battery in battery powered applications, the LVR will automatically reset the device internally and the LVRF bit in the RSTFC register will also be set to 1. For a valid LVR signal, a low supply voltage, i.e., a voltage in the range between 0.9V~V_{LVR} must exist for a time greater than that specified by t_{LVR} in the LVD & LVR Electrical Characteristics. If the low supply voltage state does not exceed this value, the LVR will ignore the low supply voltage and will not perform a reset function.

Note that the LVR function will be automatically disabled when the device enters the SLEEP or IDLE mode.



Low Voltage Reset Timing Chart

• **RSTFC Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	LVRF	—	WRF
R/W	—	—	—	—	—	R/W	—	R/W
POR	—	—	—	—	—	x	—	0

"x": unknown

Bit 7~3 Unimplemented, read as "0"

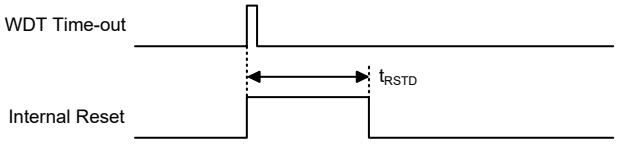
Bit 2 **LVRF**: LVR function reset flag
0: Not occur
1: Occurred
This bit is set to 1 when an actual Low Voltage Reset situation condition occurs. This bit can be cleared to 0 only by the application program.

Bit 1 Unimplemented, read as "0"

Bit 0 **WRF**: WDTC register software reset flag
Refer to the Watchdog Timer Control Register section.

Watchdog Time-out Reset during Normal Operation

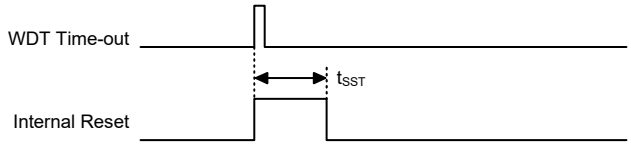
The Watchdog time-out Reset during normal operation in the FAST or SLOW mode is the same as a Power On reset except that the Watchdog time-out flag TO will be set to "1".



WDT Time-out Reset during Normal Operation Timing Chart

Watchdog Time-out Reset during SLEEP or IDLE Mode

The Watchdog time-out Reset during SLEEP or IDLE Mode is a little different from other kinds of reset. Most of the conditions remain unchanged except that the Program Counter and the Stack Pointer will be cleared to "0" and the TO flag will be set to "1". Refer to the System Start Up Time Characteristics for t_{SST} details.



WDT Time-out Reset during SLEEP or IDLE Mode Timing Chart

Reset Initial Conditions

The different types of reset described affect the reset flags in different ways. These flags, known as PDF and TO are located in the status register and are controlled by various microcontroller operations, such as the SLEEP or IDLE Mode function or Watchdog Timer. The reset flags are shown in the table:

TO	PDF	RESET Conditions
0	0	Power-on reset
u	u	LVR reset during FAST or SLOW Mode operation
1	u	WDT time-out reset during FAST or SLOW Mode operation
1	1	WDT time-out reset during IDLE or SLEEP Mode

"u" stands for unchanged

The following table indicates the way in which the various components of the microcontroller are affected after a power-on reset occurs.

Item	Condition after RESET
Program Counter	Reset to zero
Interrupts	All interrupts will be disabled
WDT, Time Bases	Clear after reset, WDT begins counting
Timer Modules	Timer Modules will be turned off
Input/Output Ports	I/O ports will be setup as inputs
Stack Pointer	Stack Pointer will point to the top of the stack

The different kinds of resets all affect the internal registers of the microcontroller in different ways. To ensure reliable continuation of normal program execution after a reset occurs, it is important to know what condition the microcontroller is in after a particular reset occurs. The following table describes how each type of reset affects each of the microcontroller internal registers.

Register	Power On Reset	LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (IDLE/SLEEP)
IAR0	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP0	0000 0000	0000 0000	0000 0000	uuuu uuuu
IAR1	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP1L	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP1H	0000 0000	0000 0000	0000 0000	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	---- xxxx	---- uuuu	---- uuuu	---- uuuu
STATUS	xx00 xxxx	xxuu uuuu	xx1u uuuu	uu11 uuuu
VBGRC	---- ---0	---- ---0	---- ---0	---- ---u
IAR2	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2L	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2H	0000 0000	0000 0000	0000 0000	uuuu uuuu
RSTFC	---- -x-0	---- -1-u	---- -u-u	---- -u-u
TB0C	0--- -000	0--- -000	0--- -000	u--- -uuu
TB1C	0--- -000	0--- -000	0--- -000	u--- -uuu
SCC	000- --00	000- --00	000- --00	uuu- --uu
HIRCC	---- 0001	---- 0001	---- 0001	---- uuuu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PB	---1 1111	---1 1111	---1 1111	---u uuuu
PBC	---1 1111	---1 1111	---1 1111	---u uuuu
PBPU	---0 0000	---0 0000	---0 0000	---u uuuu
SLEDC	0000 0000	0000 0000	0000 0000	uuuu uuuu
PSCR	---- --00	---- --00	---- --00	---- --uu
LVDC	--00 0000	--00 0000	--00 0000	--uu uuuu
SDSW	-000 0000	-000 0000	-000 0000	-uuu uuuu
SDPGAC0	--00 0000	--00 0000	--00 0000	--uu uuuu
SDPGAC1	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	Power On Reset	LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (IDLE/SLEEP)
SDA0C	-00- --00	-00- --00	-00- --00	-uu- --uu
SDA0VOS	0010 0000	0010 0000	0010 0000	uuuu uuuu
SDA1C	-00- --00	-00- --00	-00- --00	-uu- --uu
SDA1VOS	0010 0000	0010 0000	0010 0000	uuuu uuuu
STMC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDH	---- --00	---- --00	---- --00	---- --uu
STMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAH	---- --00	---- --00	---- --00	---- --uu
SADOL	xxxx ----	xxxx ----	xxxx ----	uuuu ---- (ADRFS=0)
				uuuu uuuu (ADRFS=1)
SADOH	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu (ADRFS=0)
				---- uuuu (ADRFS=1)
SADC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PLTSW	---- -001	---- -001	---- -001	---- -uuu
PLTDACC	---- -000	---- -000	---- -000	---- -uuu
PLTDA0L	--00 0000	--00 0000	--00 0000	--uu uuuu
PLTDA1L	--00 0000	--00 0000	--00 0000	--uu uuuu
PLTDA2L	--00 0000	--00 0000	--00 0000	--uu uuuu
PLTC0C	000- 0000	000- 0000	000- 0000	uuu- uuuu
PLTC0VOS	-001 0000	-001 0000	-001 0000	-uuu uuuu
PLTC1C	000- 0000	000- 0000	000- 0000	uuu- uuuu
PLTC1VOS	-001 0000	-001 0000	-001 0000	-uuu uuuu
PLTCHYC	-000 0000	-000 0000	-000 0000	-uuu uuuu
PLTAC	-00- ---0	-00- ---0	-00- ---0	-uu- ---u
PLTAVOS	0010 0000	0010 0000	0010 0000	uuuu uuuu
WDTC	0101 0011	0101 0011	0101 0011	uuuu uuuu
EEA	--00 0000	--00 0000	--00 0000	--uu uuuu
EED	0000 0000	0000 0000	0000 0000	uuuu uuuu
EEC	---- 0000	---- 0000	---- 0000	---- uuuu
SIMC0	1110 0000	1110 0000	1110 0000	uuuu uuuu
SIMC1 (UMD=0)	1000 0001	1000 0001	1000 0001	uuuu uuuu
UUCR1* (UMD=1)	0000 00x0	0000 00x0	0000 00x0	uuuu uuuu
SIMD/UTXR_RXR	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SIMA/SIMC2/UUCR2	0000 0000	0000 0000	0000 0000	uuuu uuuu
SIMTOC (UMD=0)	0000 0000	0000 0000	0000 0000	uuuu uuuu
UBRG* (UMD=1)	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
UUSR	0000 1011	0000 1011	0000 1011	uuuu uuuu
INTEG	---- 0000	---- 0000	---- 0000	---- uuuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	Power On Reset	LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (IDLE/SLEEP)
INTC2	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC3	-000 -000	-000 -000	-000 -000	-uuu -uuu
PAS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBS1	---- --00	---- --00	---- --00	---- --uu
IFS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMC0	0000 0---	0000 0---	0000 0---	uuuu u---
PTMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMC2	---- -000	---- -000	---- -000	---- -uuu
PTMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDH	---- --00	---- --00	---- --00	---- --uu
PTMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMAH	---- --00	---- --00	---- --00	---- --uu
PTMBL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMBH	---- --00	---- --00	---- --00	---- --uu
PTMRPL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMRPH	---- --00	---- --00	---- --00	---- --uu
ISGENC	0--- --00	0--- --00	0--- --00	u--- --uu
ISGDATA0	---0 0000	---0 0000	---0 0000	---u uuuu
ISGDATA1	---0 0000	---0 0000	---0 0000	---u uuuu

Note: “u” stands for unchanged
“x” stands for unknown
“-” stands for unimplemented
“*” The UUCR1 and SIMC1 registers share the same memory address while the UBRG and SIMTOC registers share the same memory address. The default value of the UUCR1 or UBRG register can be obtained when the UMD bit is set high by application program after a reset.

Input/Output Ports

Holtek microcontrollers offer considerable flexibility on their I/O ports. With the input or output designation of every pin fully under user program control, pull-high selections for all ports and wake-up selections on certain pins, the user is provided with an I/O structure to meet the needs of a wide range of application possibilities.

The device provides bidirectional input/output lines labeled with port names PA~PB. These I/O ports are mapped to the RAM Data Memory with specific addresses as shown in the Special Purpose Data Memory table. All of these I/O ports can be used for input and output operations. For input operation, these ports are non-latching, which means the inputs must be ready at the T2 rising edge of instruction "MOV A, [m]", where m denotes the port address. For output operation, all the data is latched and remains unchanged until the output latch is rewritten.

Register Name	Bit							
	7	6	5	4	3	2	1	0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PB	—	—	—	PB4	PB3	PB2	PB1	PB0
PBC	—	—	—	PBC4	PBC3	PBC2	PBC1	PBC0
PBPU	—	—	—	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0

"—": Unimplemented, read as "0"

I/O Logic Function Register List

Pull-high Resistors

Many product applications require pull-high resistors for their switch inputs usually requiring the use of an external resistor. To eliminate the need for these external resistors, all I/O pins, when configured as an input have the capability of being connected to an internal pull-high resistor. These pull-high resistors are selected using registers PAPU~PBPU, and are implemented using weak PMOS transistors.

Note that the pull-high resistor can be controlled by the relevant pull-high control register only when the pin-shared functional pin is selected as a digital input or NMOS output. Otherwise, the pull-high resistors cannot be enabled.

- **PxPU Register**

Bit	7	6	5	4	3	2	1	0
Name	PxPU7	PxPU6	PxPU5	PxPU4	PxPU3	PxPU2	PxPU1	PxPU0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

PxPUn: I/O Port x Pin pull-high function control
0: Disable
1: Enable

The PxPUn bit is used to control the pin pull-high function. Here the "x" can be A or B. However, the actual available bits for each I/O Port may be different.

Port A Wake-up

The HALT instruction forces the microcontroller into the SLEEP or IDLE Mode which preserves power, a feature that is important for battery and other low-power applications. Various methods exist to wake-up the microcontroller, one of which is to change the logic condition on one of the Port A pins from high to low. This function is especially suitable for applications that can be woken up via external switches. Each pin on Port A can be selected individually to have this wake-up feature using the PAWU register.

Note that the wake-up function can be controlled by the wake-up control registers only when the pin is selected as a general purpose input and the MCU enters the IDLE or SLEEP mode.

• **PAWU Register**

Bit	7	6	5	4	3	2	1	0
Name	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **PAWU7~PAWU0**: PA7~PA0 wake-up function control
0: Disable
1: Enable

I/O Port Control Registers

Each I/O port has its own control register known as PAC~PBC, to control the input/output configuration. With this control register, each CMOS output or input can be reconfigured dynamically under software control. Each pin of the I/O ports is directly mapped to a bit in its associated port control register. For the I/O pin to function as an input, the corresponding bit of the control register must be written as a "1". This will then allow the logic state of the input pin to be directly read by instructions. When the corresponding bit of the control register is written as a "0", the I/O pin will be setup as a CMOS output. If the pin is currently setup as an output, instructions can still be used to read the output register. However, it should be noted that the program will in fact only read the status of the output data latch and not the actual logic status of the output pin.

• **PxC Register**

Bit	7	6	5	4	3	2	1	0
Name	PxC7	PxC6	PxC5	PxC4	PxC3	PxC2	PxC1	PxC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

PxCn: I/O Port x Pin type selection
0: Output
1: Input

The PxCn bit is used to control the pin type selection. Here the "x" can be A or B. However, the actual available bits for each I/O Port may be different.

It is important to note that the PBC2 and PBC3 bits in the PBC register must be properly configured if the RF pin function shared with the MCU PB2 and PB3 pins respectively. It is recommended to set the PB2 as output if both the MCU and RF pin functions are needed.

I/O Port Source Current Selection

Each pin in this device can be configured with different output source current which is selected by the corresponding source current selection bits. These source current selection bits are available when the corresponding pin is configured as a CMOS output. Otherwise, these select bits have no effect. Users should refer to the Input/Output Characteristics section to obtain the exact value for different applications.

• **SLEDC Register**

Bit	7	6	5	4	3	2	1	0
Name	SLEDC7	SLEDC6	SLEDC5	SLEDC4	SLEDC3	SLEDC2	SLEDC1	SLEDC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **SLEDC7~SLEDC6**: PB4 source current selection
00: Source current=Level 0 (Min.)
01: Source current=Level 1
10: Source current=Level 2
11: Source current=Level 3 (Max.)

Bit 5~4 **SLEDC5~SLEDC4**: PB3~PB0 source current selection
00: Source current=Level 0 (Min.)
01: Source current=Level 1
10: Source current=Level 2
11: Source current=Level 3 (Max.)

Bit 3~2 **SLEDC3~SLEDC2**: PA7~PA4 source current selection
00: Source current=Level 0 (Min.)
01: Source current=Level 1
10: Source current=Level 2
11: Source current=Level 3 (Max.)

Bit 1~0 **SLEDC1~SLEDC0**: PA3~PA0 source current selection
00: Source current=Level 0 (Min.)
01: Source current=Level 1
10: Source current=Level 2
11: Source current=Level 3 (Max.)

Pin-shared Functions

The flexibility of the microcontroller range is greatly enhanced by the use of pins that have more than one function. Limited numbers of pins can force serious design constraints on designers but by supplying pins with multi-functions, many of these difficulties can be overcome. For these pins, the desired function of the multi-function I/O pins is selected by a series of registers via the application program control.

Pin-shared Function Selection Registers

The limited number of supplied pins in a package can impose restrictions on the amount of functions a certain device can contain. However by allowing the same pins to share several different functions and providing a means of function selection, a wide range of different functions can be incorporated into even relatively small package sizes. The device includes Port “x” Output Function Selection register “n”, labeled as PxSn, and Input Function Selection register, labeled as IFS0, which can select the desired functions of the multi-function pin-shared pins.

The most important point to note is to make sure that the desired pin-shared function is properly selected and also deselected. For most pin-shared functions, to select the desired pin-shared function, the pin-shared function should first be correctly selected using the corresponding pin-shared control register. After that the corresponding peripheral functional setting should be configured and then the peripheral function can be enabled.

However, special point must be noted for some digital input pins, such as INTn, xTCKn, xTPnI, etc, which share the same pin-shared control configuration with their corresponding general purpose I/O functions when setting the revelant pin-shared control bit fields. To select these pin functions, in addition to the necessary pin-shared control and peripheral functional setup forementioned, they must also be setup as input by setting the corresponding bit in the I/O port control register.

To correctly deselect the pin-shared function, the peripheral function should first be disabled and then the corresponding pin-shared function control register can be modified to select other pin-shared functions.

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	PAS17	PAS16	PAS15	PAS14	PAS13	PAS12	PAS11	PAS10
PBS0	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
PBS1	—	—	—	—	—	—	PBS11	PBS10
IFS0	IFS07	IFS06	IFS05	IFS04	IFS03	IFS02	IFS01	IFS00

Pin-shared Function Selection Register List

- **PAS0 Register**

Bit	7	6	5	4	3	2	1	0
Name	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PAS07~PAS06**: PA3 Pin-Shared function selection
00: PA3
01: SDO/TX
10: PTPB
11: AN3

Bit 5~4 **PAS05~PAS04**: PA2 Pin-Shared function selection
00: PA2
01: SDI/SDA/RX
10: PA2
11: PA2

Bit 3~2 **PAS03~PAS02**: PA1 Pin-Shared function selection
00: PA1/INT1
01: $\overline{SCS}$
10: AN2
11: A1PI

Bit 1~0 **PAS01~PAS00**: PA0 Pin-Shared function selection
00: PA0
01: SCL/SCK
10: PA0
11: PA0

- **PAS1 Register**

Bit	7	6	5	4	3	2	1	0
Name	PAS17	PAS16	PAS15	PAS14	PAS13	PAS12	PAS11	PAS10
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PAS17~PAS16**: PA7 Pin-Shared function selection
00: PA7/STPI/PTPI
01: SCK/SCL
10: AN1
11: PA7/STPI/PTPI

Bit 5~4 **PAS15~PAS14**: PA6 Pin-Shared function selection
00: PA6
01: PTP
10: SDI/SDA/RX
11: VREF

Bit 3~2 **PAS13~PAS12**: PA5 Pin-Shared function selection
00: PA5/STCK
01: A1O
10: PA5/STCK
11: PA5/STCK

Bit 1~0 **PAS11~PAS10**: PA4 Pin-Shared function selection
00: PA4/PTCK
01: STPB
10: AN0
11: A0O

- **PBS0 Register**

Bit	7	6	5	4	3	2	1	0
Name	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PBS07~PBS06**: PB3 Pin-Shared function selection
00: PB3
01: PLRX
10: SDI/SDA/RX
11: PB3

As these pin functions share the same pin location with the GIO2 pin of the RF transceiver, to use any one of these pin functions by setting the PBS07~PBS06 bits, it is recommended to avoid using the RF transceiver by disconnecting the RF power supply, or first ensure the RF transceiver is in the Deep Sleep Mode. If the GIO2 pin of the RF transceiver is to be used, the PBS07~PBS06 bits must be fixed at 00 or 11.

Bit 5~4 **PBS05~PBS04**: PB2 Pin-Shared function selection
00: PB2
01: PLIS
10: SCK/SCL
11: PB2

As these pin functions share the same pin location with the SCK pin of the RF transceiver, to use any one of these pin functions by setting the PBS05~PBS04 bits, it is recommended to avoid using the RF transceiver by disconnecting the RF power supply, or first ensure the RF transceiver is in the Deep Sleep Mode. If the SCK pin of the RF transceiver is to be used, the PBS05~PBS04 bits must be fixed at 00 or 11.

Bit 3~2 **PBS03~PBS02**: PB1 Pin-Shared function selection
00: PB1
01: PLTX
10: SDO/TX
11: PB1

Bit 1~0 **PBS01~PBS00**: PB0 Pin-Shared function selection
00: PB0/INT0
01: $\overline{\text{SCS}}$
10: STP
11: A0PB

• **PBS1 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PBS11	PBS10
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **PBS11~PBS10**: PB4 Pin-Shared function selection
00: PB4
01: $\overline{SCS}$
10: PB4
11: PB4

• **IFS0 Register**

Bit	7	6	5	4	3	2	1	0
Name	IFS07	IFS06	IFS05	IFS04	IFS03	IFS02	IFS01	IFS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **IFS07~IFS06**: PTPI input source selection
00: CXCAP
01: PA7
10: CXCAP
11: CXCAP
Note: CXCAP is the PowerLine Transceiver comparator output signal, refer to "Powerline Transceiver – PLT" section for details.

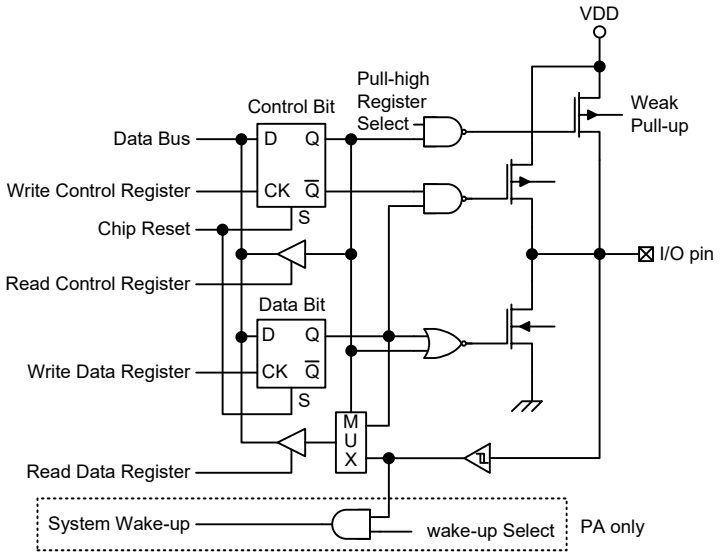
Bit 5~4 **IFS05~IFS04**: $\overline{SCS}$ input source pin selection
00: PB0
01: PB4
10: PA1
11: PB0

Bit 3~2 **IFS03~IFS02**: SCK/SCL input source pin selection
00: PA0
01: PB2
10: PA7
11: PB2
Note: If the SPI Master mode is selected, when the SIMEN bit is set high, the PA0 and PB2 pins both can be used as the SCK pin function ignoring the IFS0[3:2] bit settings.

Bit 1~0 **IFS01~IFS00**: SDI/SDA/RX input source pin selection
00: PA2
01: PA6
10: PB3
11: PA2

I/O Pin Structures

The accompanying diagram illustrates the internal structures of the I/O logic function. As the exact logical construction of the I/O pin will differ from this diagram, it is supplied as a guide only to assist with the functional understanding of the logic function I/O pins. The wide range of pin-shared structures does not permit all types to be shown.



Logic Function Input/Output Structure

Programming Considerations

Within the user program, one of the first things to consider is port initialisation. After a reset, all of the I/O data and port control registers will be set high. This means that all I/O pins will default to an input state, the level of which depends on the other connected circuitry and whether pull-high selections have been chosen. If the port control registers are then programmed to setup some pins as outputs, these output pins will have an initial high output value unless the associated port data registers are first programmed. Selecting which pins are inputs and which are outputs can be achieved byte-wide by loading the correct values into the appropriate port control register or by programming individual bits in the port control register using the "SET [m].i" and "CLR [m].i" instructions. Note that when using these bit control instructions, a read-modify-write operation takes place. The microcontroller must first read in the data on the entire port, modify it to the required new bit values and then rewrite this data back to the output ports.

Port A has the additional capability of providing wake-up functions. When the device is in the SLEEP or IDLE Mode, various methods are available to wake the device up. One of these is a high to low transition of any of the Port A pins. Single or multiple pins on Port A can be setup to have this function.

Timer Modules – TM

One of the most fundamental functions in any microcontroller the device is the ability to control and measure time. To implement time related functions the device includes several Timer Modules, abbreviated to the name TM. The TMs are multi-purpose timing units and serve to provide operations such as Timer/Counter, Input Capture, Compare Match Output and Single Pulse Output as well as being the functional unit for the generation of PWM signals. Each of the TMs has two individual interrupts. The addition of input and output pins for each TM ensures that users are provided with timing units with a wide and flexible range of features.

The common features of the different TM types are described here with more detailed information provided in the individual Standard and Periodic TM sections.

Introduction

The device contains two TMs and each individual TM can be categorised as a certain type, namely Standard Type TM or Periodic Type TM. Although similar in nature, the different TM types vary in their feature complexity. The common features to all of the Standard and Periodic TMs will be described in this section. The detailed operation regarding each of the TM types will be described in separate sections. The main features and differences between the two types of TMs are summarised in the accompanying table.

Function	STM	PTM
Timer/Counter	√	√
Input Capture	√	√
Compare Match Output	√	√
PWM Output	√	√
Single Pulse Output	√	√
PWM Alignment	Edge	Edge
PWM Adjustment Period & Duty	Duty or Period	Duty or Period

TM Function Summary

TM Operation

The different types of TM offer a diverse range of functions, from simple timing operations to PWM signal generation. The key to understanding how the TM operates is to see it in terms of a free running counter whose value is then compared with the value of pre-programmed internal comparators. When the free running counter has the same value as the pre-programmed comparator, known as a compare match situation, a TM interrupt signal will be generated which can clear the counter and perhaps also change the condition of the TM output pin. The internal TM counter is driven by a user selectable clock source, which can be an internal clock or an external pin.

TM Clock Source

The clock source which drives the main counter in each TM can originate from various sources. The selection of the required clock source is implemented using the xTCK2~xTCK0 bits in the xTM control registers, where "x" stands for S or P type TM. The clock source can be a ratio of the system clock f_{SYS} or the internal high clock f_H, the f_{SUB} clock source or the external xTCK pin. The xTCK pin clock source is used to allow an external signal to drive the TM as an external clock source or for event counting.

TM Interrupts

The Standard and Periodic type TMs each have two internal interrupts, one for each of the internal comparator A or comparator P, which generate a TM interrupt when a compare match condition occurs. When a TM interrupt is generated it can be used to clear the counter and also to change the state of the TM output pin.

TM External Pins

Each of the TMs, irrespective of what type, has two TM input pins, with the label xTCK and xTPI respectively. The xTM input pin, xTCK, is essentially a clock source for the xTM and is selected using the xTCK2~xTCK0 bits in the xTMC0 register. This external xTCK input pin allows an external clock source to drive the internal TM. The xTCK input pin can be chosen to have either a rising or falling active edge. The xTCK pin is also used as the external trigger input pin in single pulse output mode.

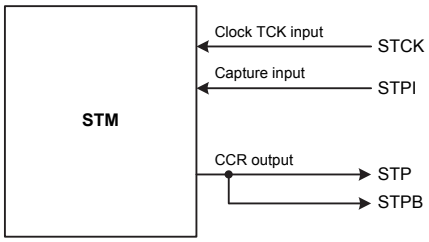
The other xTM input pin, xTPI, is the capture input whose active edge can be a rising edge, a falling edge or both rising and falling edges and the active edge transition type is selected using the xTIO1~xTIO0 bits in the xTMC1 register. There is another capture input, PTCK, for PTM capture input mode, which can be used as the external trigger input source.

The TMs each have two output pins with the label xTP and xTPB. The xTPB pin outputs the inverted signal of the xTP. When the TM is in the Compare Match Output Mode, these pins can be controlled by the TM to switch to a high or low level or to toggle when a compare match situation occurs. The external output pin is also the pin where the TM generates the PWM output waveform.

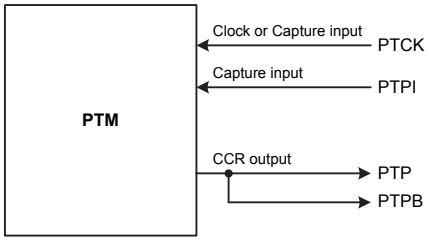
As the TM input and output pins are pin-shared with other functions, the TM input or output function must first be setup using relevant pin-shared function selection register. The details of the pin-shared function selection are described in the pin-shared function section.

STM		PTM	
Input	Output	Input	Output
STCK, STPI	STP, STPB	PTCK, PTPI	PTP, PTPB

TM External Pins



STM Function Pin Block Diagram

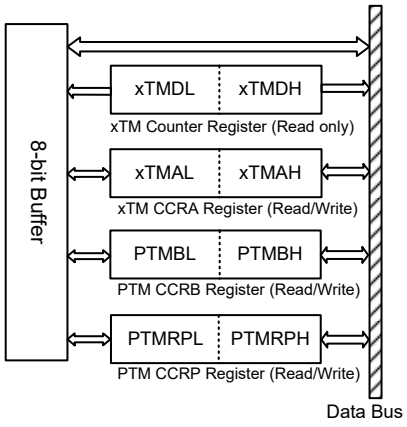


PTM Function Pin Block Diagram

Programming Considerations

The TM Counter Registers and the Capture/Compare CCRA and CCRP registers as well as the PTM CCRB register, all have a low and high byte structure. The high bytes can be directly accessed, but as the low bytes can only be accessed via an internal 8-bit buffer, reading or writing to these register pairs must be carried out in a specific way. The important point to note is that data transfer to and from the 8-bit buffer and its related low byte only takes place when a write or read operation to its corresponding high byte is executed.

As the CCRA, CCRP and CCRB registers are implemented in the way shown in the following diagram and accessing these register pairs is carried out in a specific way as described above, it is recommended to use the "MOV" instruction to access the CCRA, CCRP and CCRB low byte registers, named xTMAL, PTMRPL, PTMBL, using the following access procedures. Accessing the CCRA, CCRB or CCRP low byte register without following these access procedures will result in unpredictable values.

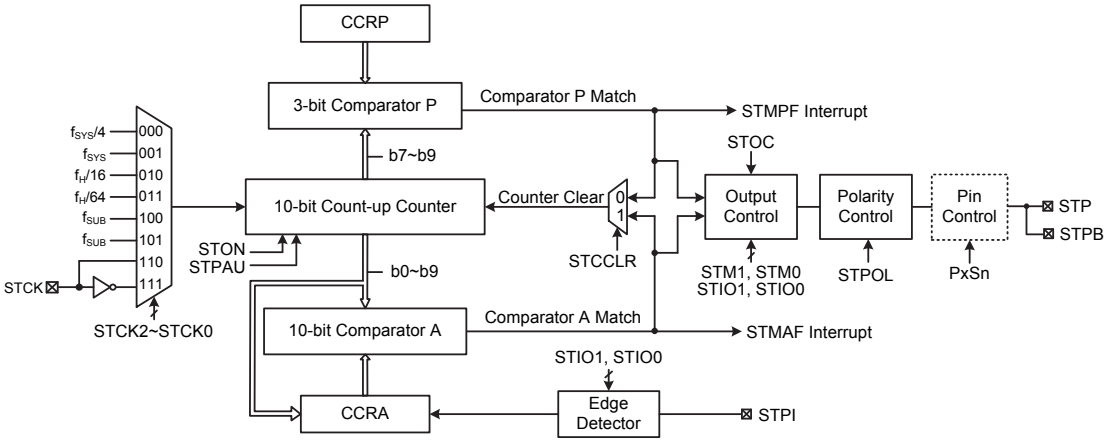


The following steps show the read and write procedures:

- Writing Data to CCRA, CCRB or CCRP
  - Step 1. Write data to Low Byte xTMAL, PTMBL or PTMRPL
    - Note that here data is only written to the 8-bit buffer.
  - Step 2. Write data to High Byte xTMAH, PTMBH or PTMRPH
    - Here data is written directly to the high byte registers and simultaneously data is latched from the 8-bit buffer to the Low Byte registers.
- Reading Data from the Counter Registers, CCRA, CCRB or CCRP
  - Step 1. Read data from the High Byte xTMDH, xTMAH, PTMBH or PTMRPH
    - Here data is read directly from the High Byte registers and simultaneously data is latched from the Low Byte register into the 8-bit buffer.
  - Step 2. Read data from the Low Byte xTMDL, xTMAL, PTMBL or PTMRPL
    - This step reads data from the 8-bit buffer.

Standard Type TM – STM

The Standard Type TM contains five operating modes, which are Compare Match Output, Timer/Event Counter, Capture Input, Single Pulse Output and PWM Output modes. The Standard TM can also be controlled with two external input pins and can drive two external output pins.



Note: 1. The STPB is the inverted output of the STP.

2. The STM external pins are pin-shared with other functions and can input or output on different pins, so before using the STM function, the pin-shared function registers must be set properly.

Standard Type TM Block Diagram

Standard TM Operation

The size of Standard TM is 10-bit wide and its core is a 10-bit count-up counter which is driven by a user selectable internal or external clock source. There are also two internal comparators with the names, Comparator A and Comparator P. These comparators will compare the value in the counter with CCRP and CCRA registers. The CCRP comparator is 3-bit wide whose value is compared with the highest 3 bits in the counter while the CCRA is 10 bits and therefore compares all counter bits.

The only way of changing the value of the 10-bit counter using the application program, is to clear the counter by changing the STON bit from low to high. The counter will also be cleared automatically by a counter overflow or a compare match with one of its associated comparators. When these conditions occur, a STM interrupt signal will also usually be generated. The Standard Type TM can operate in a number of different operational modes, can be driven by different clock sources including an input pin and can also control two output pins. All operating setup conditions are selected using relevant internal registers.

Standard Type TM Register Description

Overall operation of the Standard TM is controlled using a series of registers. A read only register pair exists to store the internal counter 10-bit value, while a read/write register pair exists to store the internal 10-bit CCRA value. The remaining two registers are control registers which setup the different operating and control modes as well as three CCRP bits.

Register Name	Bit							
	7	6	5	4	3	2	1	0
STMC0	STPAU	STCK2	STCK1	STCK0	STON	STRP2	STRP1	STRP0
STMC1	STM1	STM0	STIO1	STIO0	STOC	STPOL	STDPX	STCCLR
STMDL	D7	D6	D5	D4	D3	D2	D1	D0
STMDH	—	—	—	—	—	—	D9	D8
STMAL	D7	D6	D5	D4	D3	D2	D1	D0
STMAH	—	—	—	—	—	—	D9	D8

10-bit Standard TM Register List

- **STMC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	STPAU	STCK2	STCK1	STCK0	STON	STRP2	STRP1	STRP0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **STPAU**: STM Counter Pause Control
0: Run
1: Pause
The counter can be paused by setting this bit high. Clearing the bit to zero restores normal counter operation. When in a Pause condition the STM will remain powered up and continue to consume power. The counter will retain its residual value when this bit changes from low to high and resume counting from this value when the bit changes to a low value again.

Bit 6~4 **STCK2~STCK0**: Select STM Counter Clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: f_{SUB}
110: STCK rising edge clock
111: STCK falling edge clock
These three bits are used to select the clock source for the STM. The external pin clock source can be chosen to be active on the rising or falling edge. The clock source f_{SYS} is the system clock, while f_H and f_{SUB} are other internal clocks, the details of which can be found in the oscillator section.

Bit 3 **STON**: STM Counter On/Off Control
0: Off
1: On
This bit controls the overall on/off function of the STM. Setting the bit high enables the counter to run while clearing the bit disables the STM. Clearing this bit to zero will stop the counter from counting and turn off the STM which will reduce its power consumption. When the bit changes state from low to high the internal counter value will be reset to zero, however when the bit changes from high to low, the internal counter will retain its residual value until the bit returns high again. If the STM is in the Compare Match Output Mode or the PWM output Mode or Single Pulse Output Mode then the STM output pin will be reset to its initial condition, as specified by the STOC bit, when the STON bit changes from low to high.

Bit 2~0 **STRP2~STRP0**: STM CCRP 3-bit register, compared with the STM counter bit 9~bit 7 Comparator P Match Period =
000: 1024 STM clocks
001: 128 STM clocks
010: 256 STM clocks
011: 384 STM clocks
100: 512 STM clocks
101: 640 STM clocks
110: 768 STM clocks
111: 896 STM clocks

These three bits are used to setup the value on the internal CCRP 3-bit register, which are then compared with the internal counter's highest three bits. The result of this comparison can be selected to clear the internal counter if the STCCLR bit is set to zero. Setting the STCCLR bit to zero ensures that a compare match with the CCRP values will reset the internal counter. As the CCRP bits are only compared with the highest three counter bits, the compare values exist in 128 clock cycle multiples. Clearing all three bits to zero is in effect allowing the counter to overflow at its maximum value.

• STMC1 Register

Bit	7	6	5	4	3	2	1	0
Name	STM1	STM0	STIO1	STIO0	STOC	STPOL	STDPX	STCCLR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **STM1~STM0**: Select STM Operating Mode
00: Compare Match Output Mode
01: Capture Input Mode
10: PWM Output Mode or Single Pulse Output Mode
11: Timer/Counter Mode

These bits setup the required operating mode for the STM. To ensure reliable operation the STM should be switched off before any changes are made to the STM1 and STM0 bits. In the Timer/Counter Mode, the STM output pin state is undefined.

Bit 5~4 **STIO1~STIO0**: Select STM External Pins Function
Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output
PWM Output Mode/Single Pulse Output Mode
00: PWM output inactive state
01: PWM output active state
10: PWM output
11: Single Pulse Output
Capture Input Mode
00: Input capture at rising edge of STPI
01: Input capture at falling edge of STPI
10: Input capture at both rising and falling edges of STPI
11: Input capture disabled
Timer/Counter Mode
Unused

These two bits are used to determine how the STM functions when a certain condition is reached. The function that these bits select depends upon in which mode the STM is running.

In the Compare Match Output Mode, the STIO1 and STIO0 bits determine how the STM output pin changes state when a compare match occurs from the Comparator A. The STM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the bits are both zero, then no change will take place on the output. The initial value of the STM output pin should be setup using the STOC bit in the STMC1 register. Note that the output level requested by the STIO1 and STIO0 bits must be different from the initial value setup using the STOC bit otherwise no change will occur on the STM output pin when a compare match occurs. After the STM output pin changes state, it can be reset to its initial level by changing the level of the STON bit from low to high.

In the PWM Output Mode, the STIO1 and STIO0 bits determine how the STM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to only change the values of the STIO1 and STIO0 bits only after the STM has been switched off. Unpredictable PWM outputs will occur if the STIO1 and STIO0 bits are changed when the STM is running.

Bit 3 **STOC**: STP Output Control

Compare Match Output Mode
0: Initial low
1: Initial high

PWM Output Mode/Single Pulse Output Mode
0: Active low
1: Active high

This is the output control bit for the STM output. Its operation depends upon whether STM is being used in the Compare Match Output Mode or in the PWM Output Mode/ Single Pulse Output Mode. It has no effect if the STM is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the STM output pin before a compare match occurs. In the PWM Output Mode/Single Pulse Output Mode it determines if the PWM signal is active high or active low.

Bit 2 **STPOL**: STP Output Polarity Control
0: Non-invert
1: Invert

This bit controls the polarity of the STP output. When the bit is set high the STM output will be inverted and not inverted when the bit is zero. It has no effect if the STM is in the Timer/Counter Mode.

Bit 1 **STDPX**: STM PWM Duty/Period Control
0: CCRP – period; CCRA – duty
1: CCRP – duty; CCRA – period

This bit determines which of the CCRA and CCRP registers are used for period and duty control of the PWM waveform.

Bit 0 **STCCLR**: STM Counter Clear Condition Selection
0: Comparator P match
1: Comparator A match

This bit is used to select the method which clears the counter. Remember that the Standard TM contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the STCCLR bit set high, the counter will be cleared when a compare match occurs from the Comparator A. When the bit is low, the counter will be cleared when a compare match occurs from the Comparator P or with a counter overflow. A counter overflow clearing method can only be implemented if the CCRP bits are all cleared to zero. The STCCLR bit is not used in the PWM Output, Single Pulse Output or Capture Input Mode.

• **STMDL Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 **D7~D0**: STM Counter Low Byte Register bit 7 ~ bit 0
STM 10-bit Counter bit 7 ~ bit 0

• **STMDH Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R	R
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **D9~D8**: STM Counter High Byte Register bit 1 ~ bit 0
STM 10-bit Counter bit 9 ~ bit 8

• **STMAL Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **D7~D0**: STM CCRA Low Byte Register bit 7 ~ bit 0
STM 10-bit CCRA bit 7 ~ bit 0

• **STMAH Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **D9~D8**: STM CCRA High Byte Register bit 1 ~ bit 0
STM 10-bit CCRA bit 9 ~ bit 8

Standard Type TM Operation Modes

The Standard Type TM can operate in one of five operating modes, Compare Match Output Mode, PWM Output Mode, Single Pulse Output Mode, Capture Input Mode or Timer/Counter Mode. The operating mode is selected using the STM1 and STM0 bits in the STMC1 register.

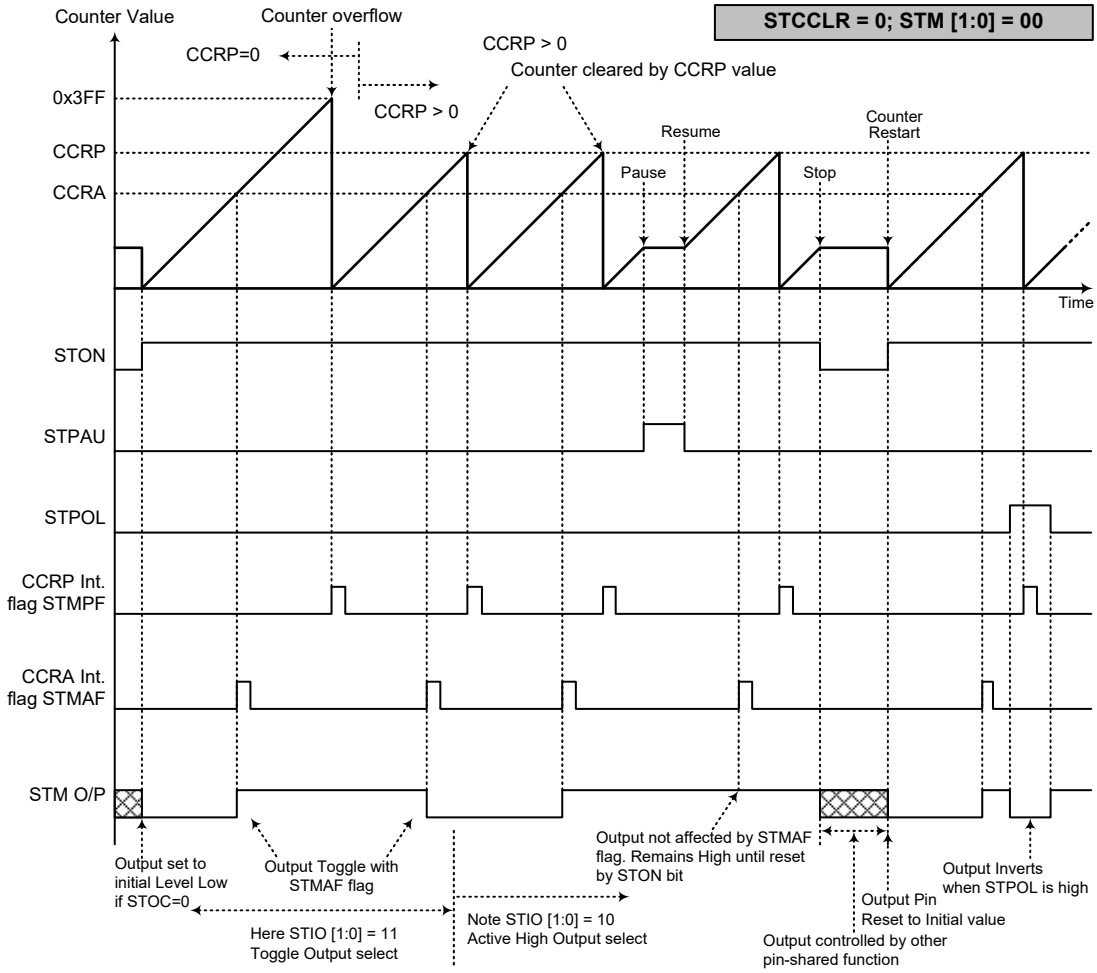
Compare Match Output Mode

To select this mode, bits STM1 and STM0 in the STMC1 register, should be set to 00 respectively. In this mode once the counter is enabled and running it can be cleared by three methods. These are a counter overflow, a compare match from Comparator A and a compare match from Comparator P. When the STCCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both STMAF and STMPF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

If the STCCLR bit in the STMC1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the STMAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when STCCLR is high no STMPF interrupt request flag will be generated. In the Compare Match Output Mode, the CCRA can not be set to “0”.

If the CCRA bits are all zero, the counter will overflow when it reaches its maximum 10-bit, 3FF Hex, value, however here the STMAF interrupt request flag will not be generated.

As the name of the mode suggests, after a comparison is made, the STM output pin, will change state. The STM output pin condition however only changes state when a STMAF interrupt request flag is generated after a compare match occurs from Comparator A. The STMPF interrupt request flag, generated from a compare match occurs from Comparator P, will have no effect on the STM output pin. The way in which the STM output pin changes state are determined by the condition of the STIO1 and STIO0 bits in the STMC1 register. The STM output pin can be selected using the STIO1 and STIO0 bits to go high, to go low or to toggle from its present condition when a compare match occurs from Comparator A. The initial condition of the STM output, which is setup after the STON bit changes from low to high, is setup using the STOC bit. Note that if the STIO1 and STIO0 bits are zero then no pin change will take place.

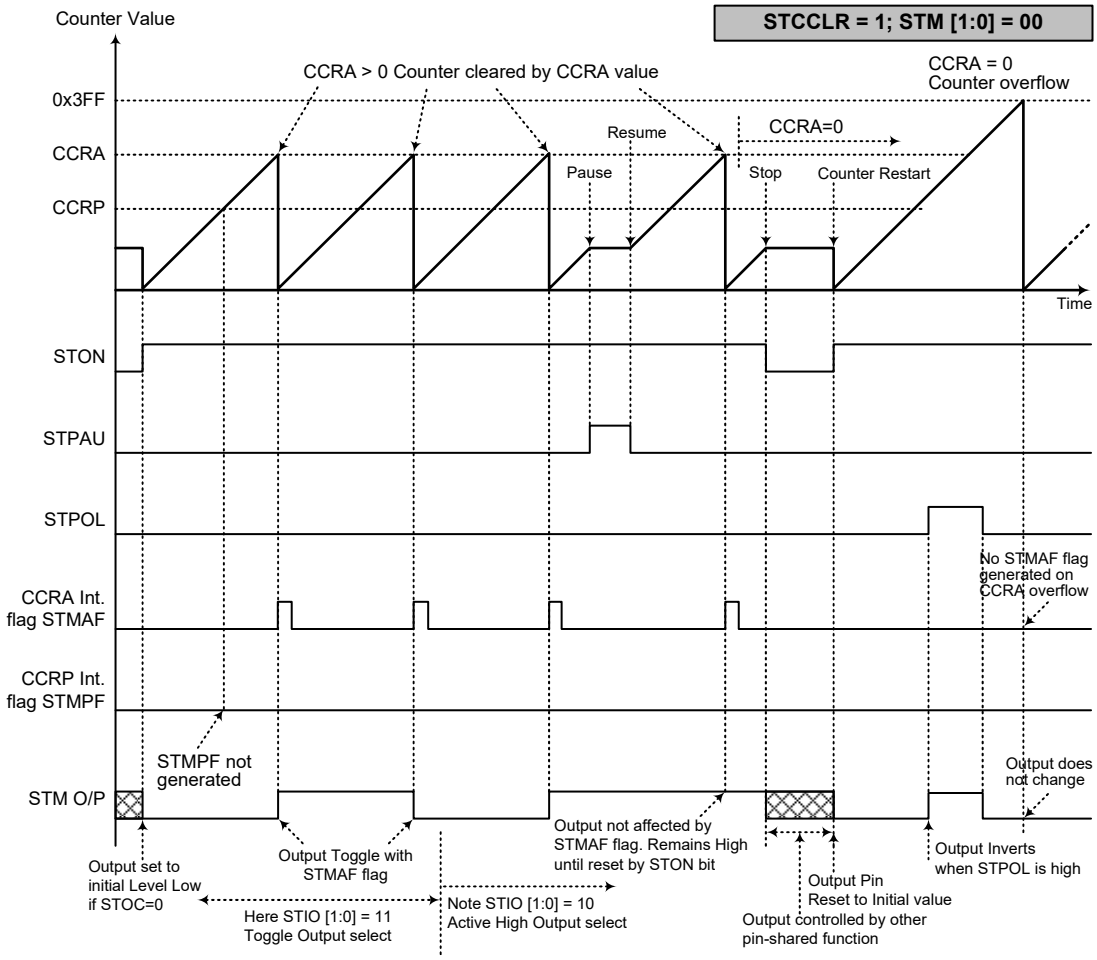


Compare Match Output Mode – STCCLR=0

Note: 1. With STCCLR=0 a Comparator P match will clear the counter

2. The STM output is controlled only by the STMAF flag

3. The STM output is reset to itsinitial state by a STON bit rising edge



Compare Match Output Mode – STCCLR=1

Note: 1. With STCCLR=1 a Comparator A match will clear the counter

2. The STM output is controlled only by the STMAF flag

3. The STM output is reset to its initial state by a STON bit rising edge

4. A STMPF flag is not generated when STCCLR=1

Timer/Counter Mode

To select this mode, bits STM1 and STM0 in the STMC1 register should be set to 11 respectively. The Timer/Counter Mode operates in an identical way to the Compare Match Output Mode generating the same interrupt flags. The exception is that in the Timer/Counter Mode the STM output pin is not used. Therefore the above description and Timing Diagrams for the Compare Match Output Mode can be used to understand its function. As the STM output pin is not used in this mode, the pin can be used as a normal I/O pin or other pin-shared function.

PWM Output Mode

To select this mode, bits STM1 and STM0 in the STMC1 register should be set to 10 respectively and also the STIO1 and STIO0 bits should be set to 10 respectively. The PWM function within the STM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the STM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM Output mode, the STCCLR bit has no effect as the PWM period. Both of the CCRA and CCRP registers are used to generate the PWM waveform, one register is used to clear the internal counter and thus control the PWM waveform frequency, while the other one is used to control the duty cycle. Which register is used to control either frequency or duty cycle is determined using the STDPX bit in the STMC1 register. The PWM waveform frequency and duty cycle can therefore be controlled by the values in the CCRA and CCRP registers.

An interrupt flag, one for each of the CCRA and CCRP, will be generated when a compare match occurs from either Comparator A or Comparator P. The STOC bit in the STMC1 register is used to select the required polarity of the PWM waveform while the two STIO1 and STIO0 bits are used to enable the PWM output or to force the STM output pin to a fixed high or low level. The STPOL bit is used to reverse the polarity of the PWM output waveform.

- **10-bit STM, PWM Output Mode, Edge-aligned Mode, STDPX=0**

CCRP	1~7	0
Period	CCRP×128	1024
Duty	CCRA	

If f_{SYS}=4MHz, STM clock source is f_{SYS}/4, CCRP=4 and CCRA =128,

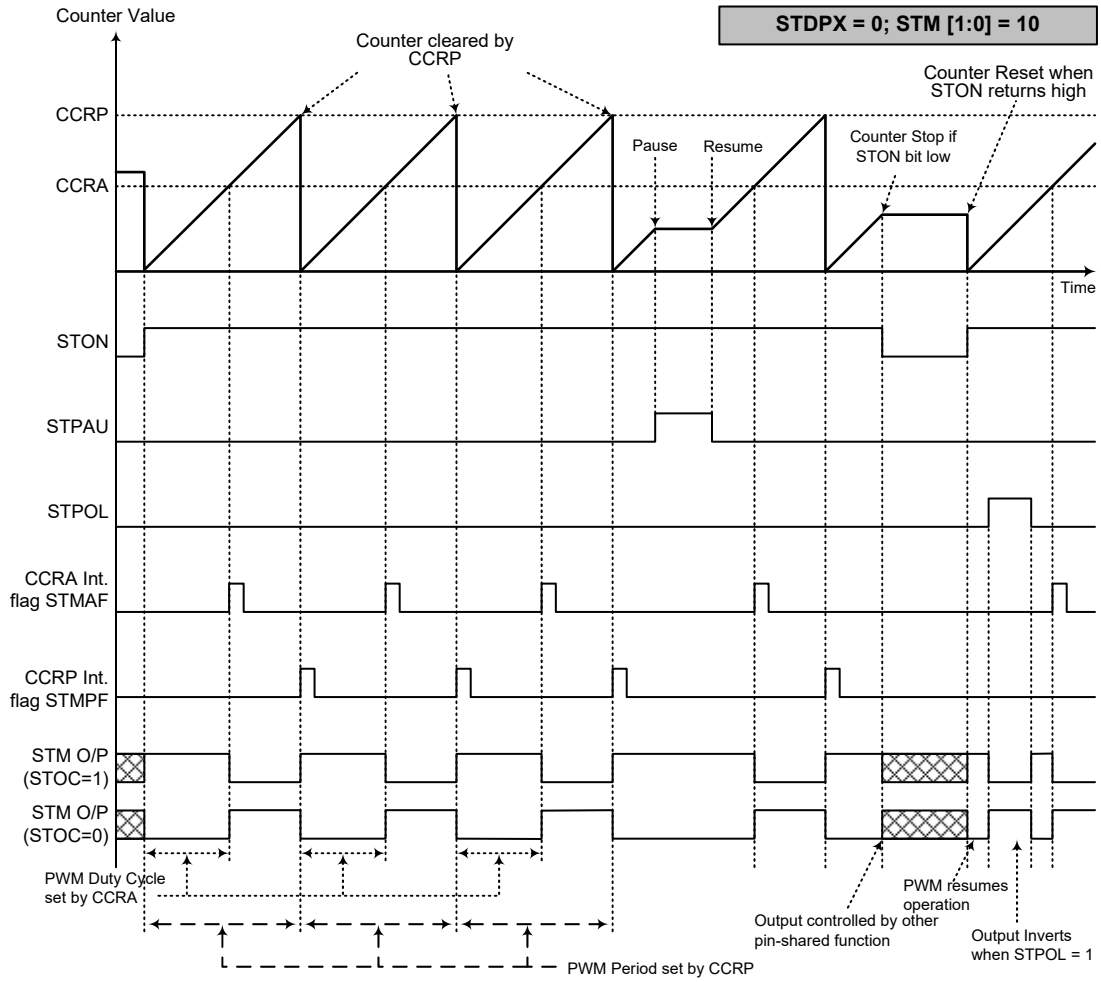
The STM PWM output frequency=(f_{SYS}/4)/(4×128)=f_{SYS}/2048=2kHz, duty=128/(4×128)=25%.

If the Duty value defined by the CCRA register is equal to or greater than the Period value, then the PWM output duty is 100%.

- **10-bit STM, PWM Output Mode, Edge-aligned Mode, STDPX=1**

CCRP	1~7	0
Period	CCRA	
Duty	CCRP×128	1024

The PWM output period is determined by the CCRA register value together with the STM clock while the PWM duty cycle is defined by the CCRP register value.



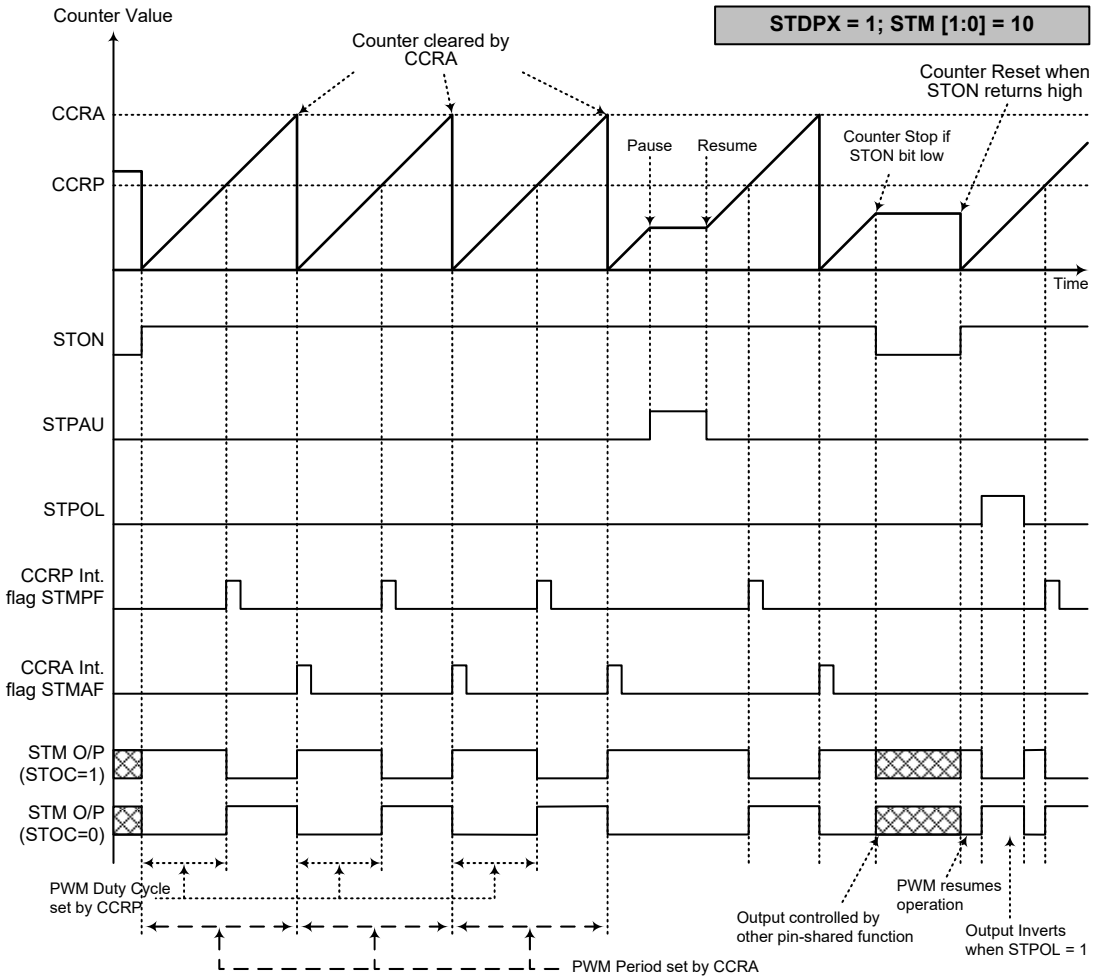
PWM Output Mode – STDPX=0

Note: 1. Here STDPX=0 – Counter cleared by CCRP

2. A counter clear sets the PWM Period

3. The internal PWM function continues running even when STIO[1:0]=00 or 01

4. The STCCLR bit has no influence on PWM operation



PWM Output Mode – STDPX=1

Note: 1. Here STDPX=1 – Counter cleared by CCRA

2. A counter clear sets the PWM Period

3. The internal PWM function continues even when STIO[1:0]=00 or 01

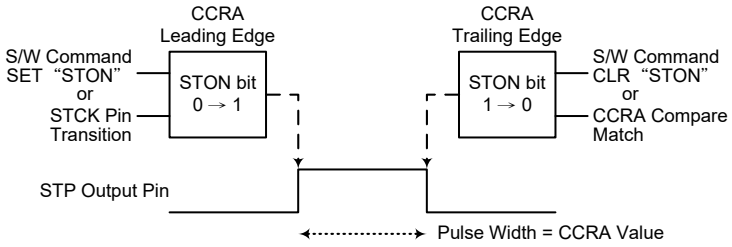
4. The STCCLR bit has no influence on PWM operation

Single Pulse Output Mode

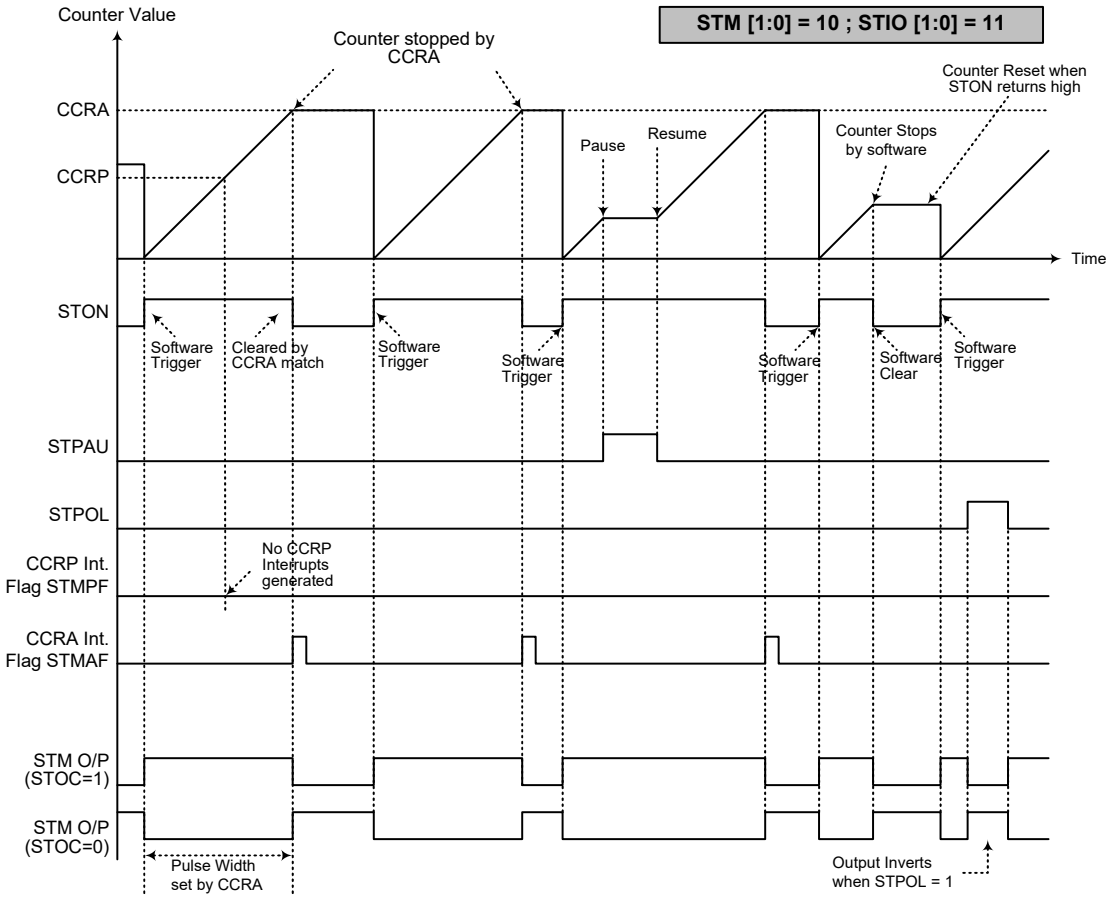
To select this mode, bits STM1 and STM0 in the STMC1 register should be set to 10 respectively and also the STIO1 and STIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the STM output pin.

The trigger for the pulse output leading edge is a low to high transition of the STON bit, which can be implemented using the application program. However in the Single Pulse Mode, the STON bit can also be made to automatically change from low to high using the external STCK pin, which will in turn initiate the Single Pulse output. When the STON bit transitions to a high level, the counter will start running and the pulse leading edge will be generated. The STON bit should remain high when the pulse is in its active state. The generated pulse trailing edge will be generated when the STON bit is cleared to zero, which can be implemented using the application program or when a compare match occurs from Comparator A.

However a compare match from Comparator A will also automatically clear the STON bit and thus generate the Single Pulse output trailing edge. In this way the CCRA value can be used to control the pulse width. A compare match from Comparator A will also generate a STM interrupt. The counter can only be reset back to zero when the STON bit changes from low to high when the counter restarts. In the Single Pulse Mode CCRP is not used. The STCCLR and STDPX bits are not used in this Mode.



Single Pulse Generation



Single Pulse Mode

Note: 1. Counter stopped by CCRA

2. CCRP is not used

3. The pulse triggered by the STCK pin or by setting the STON bit high

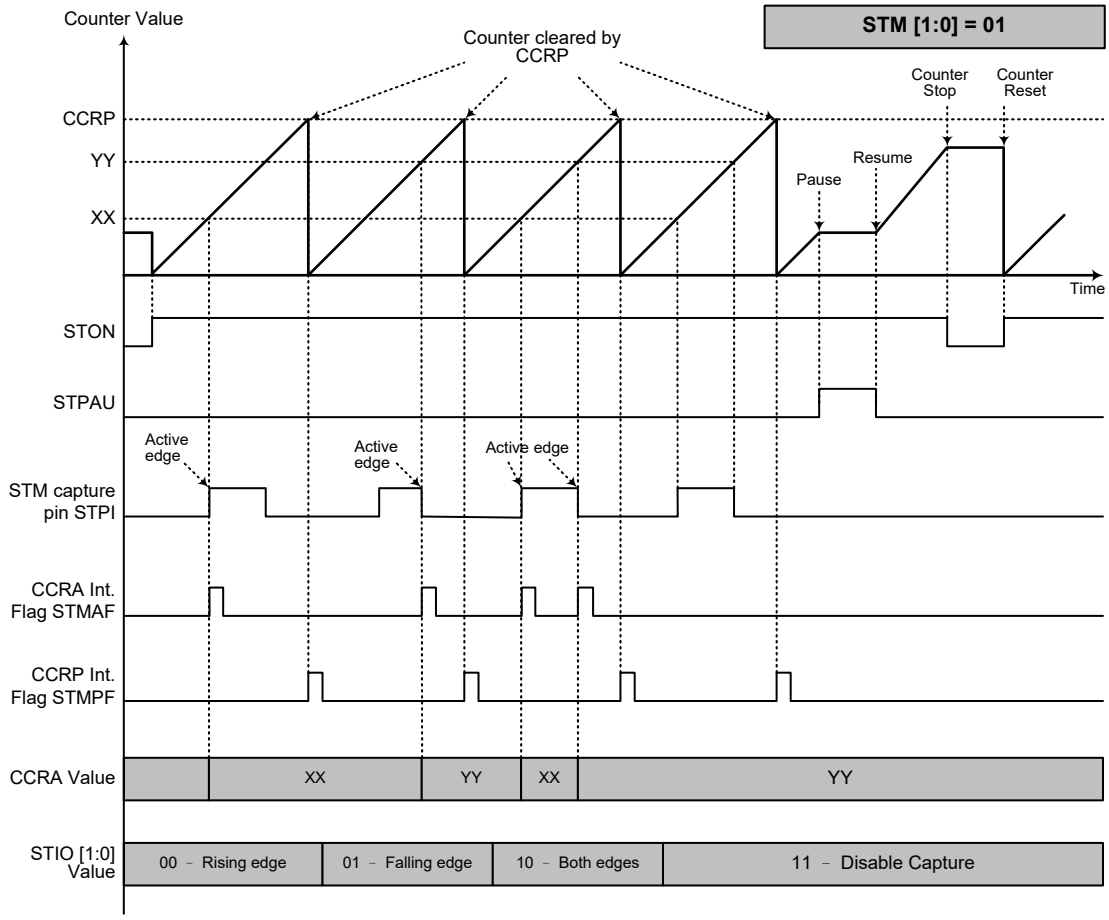
4. A STCK pin active edge will automatically set the STON bit high

5. In the Single Pulse Mode, STIO[1:0] must be set to “11” and can not be changed

Capture Input Mode

To select this mode bits STM1 and STM0 in the STMC1 register should be set to 01 respectively. This mode enables external signals to capture and store the present value of the internal counter and can therefore be used for applications such as pulse width measurements. The external signal is supplied on the STPI pin, whose active edge can be a rising edge, a falling edge or both rising and falling edges; the active edge transition type is selected using the STIO1 and STIO0 bits in the STMC1 register. The counter is started when the STON bit changes from low to high which is initiated using the application program.

When the required edge transition appears on the STPI pin the present value in the counter will be latched into the CCRA registers and a STM interrupt generated. Irrespective of what events occur on the STPI pin the counter will continue to free run until the STON bit changes from high to low. When a CCRP compare match occurs the counter will reset back to zero; in this way the CCRP value can be used to control the maximum counter value. When a CCRP compare match occurs from Comparator P, a STM interrupt will also be generated. Counting the number of overflow interrupt signals from the CCRP can be a useful method in measuring long pulse widths. The STIO1 and STIO0 bits can select the active trigger edge on the STPI pin to be a rising edge, falling edge or both edge types. If the STIO1 and STIO0 bits are both set high, then no capture operation will take place irrespective of what happens on the STPI pin, however it must be noted that the counter will continue to run. The STCCLR and STDPX bits are not used in this Mode.



Capture Input Mode

Note: 1. STM[1:0]=01 and active edge set by the STIO[1:0] bits

2. A STM Capture input pin active edge transfers the counter value to CCRA

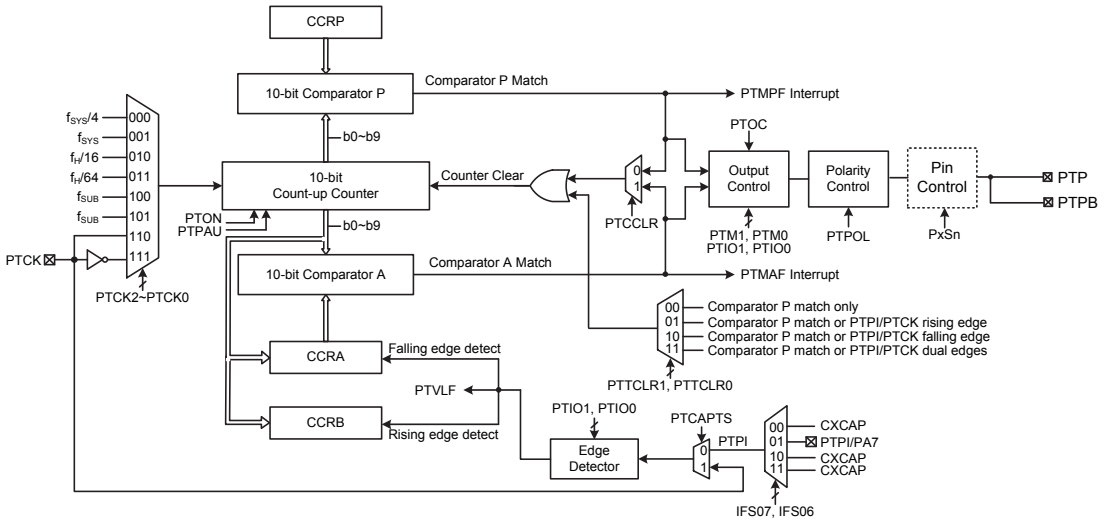
3. STCCLR bit not used

4. No output function – STOC and STPOL bits are not used

5. CCRP determines the counter value and the counter has a maximum count value when CCRP is equal to zero

Periodic Type TM – PTM

The Periodic Type TM contains five operating modes, which are Compare Match Output, Timer/Event Counter, Capture Input, Single Pulse Output and PWM Output modes. The Periodic TM can be controlled with external input pins and can drive some external output pins.



Note: 1. CXCAP is the PowerLine Transceiver comparator output signal.

2. The PTM PTPI signal can be from the external PTPI pin input or from the internal CXCAP signal, which is selected using the IFS0[7:6] bits.

3. If the PTM external pins will be used and as these pins are pin-shared with other functions, before using the PTM function, the pin-shared function registers should be set properly.

Periodic Type TM Block Diagram

Periodic TM Operation

The Periodic Type TM core is a 10-bit count-up counter which is driven by a user selectable internal or external clock source. There are also two internal comparators with the names, Comparator A and Comparator P. These comparators will compare the value in the counter with CCRP and CCRA registers. The CCRP comparator is 10-bit wide.

The only way of changing the value of the 10-bit counter using the application program, is to clear the counter by changing the PTON bit from low to high. The counter will also be cleared automatically by a counter overflow or a compare match with one of its associated comparators or the active trigger edge in Capture input mode by PTTCLR[1:0]. When these conditions occur, a PTM interrupt signal will also usually be generated. The Periodic Type TM can operate in a number of different operational modes and can be driven by different clock sources including two input pins and also control more than one output pins. All operating setup conditions are selected using relevant internal registers.

Periodic Type TM Register Description

Overall operation of the Periodic Type TM is controlled using a series of registers. A read only register pair exists to store the internal 10-bit counter value, while three read/write register pairs exist to store the internal 10-bit CCRA value, CCRP value and CCRB value. The remaining three registers are control registers which setup the different operating and control modes.

Register Name	Bit							
	7	6	5	4	3	2	1	0
PTMC0	PTPAU	PTCK2	PTCK1	PTCK0	PTON	—	—	—
PTMC1	PTM1	PTM0	PTIO1	PTIO0	PTOC	PTPOL	PTCAPTS	PTCCLR
PTMC2	—	—	—	—	—	PTTCLR1	PTTCLR0	PTVLF
PTMDL	D7	D6	D5	D4	D3	D2	D1	D0
PTMDH	—	—	—	—	—	—	D9	D8
PTMAL	D7	D6	D5	D4	D3	D2	D1	D0
PTMAH	—	—	—	—	—	—	D9	D8
PTMBL	D7	D6	D5	D4	D3	D2	D1	D0
PTMBH	—	—	—	—	—	—	D9	D8
PTMRPL	D7	D6	D5	D4	D3	D2	D1	D0
PTMRPH	—	—	—	—	—	—	D9	D8

10-bit Periodic TM Register List

- **PTMC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	PTPAU	PTCK2	PTCK1	PTCK0	PTON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

Bit 7 **PTPAU**: PTM Counter Pause Control
0: Run
1: Pause
The counter can be paused by setting this bit high. Clearing the bit to zero restores normal counter operation. When in a Pause condition the PTM will remain powered up and continue to consume power. The counter will retain its residual value when this bit changes from low to high and resume counting from this value when the bit changes to a low value again.

Bit 6~4 **PTCK2~PTCK0**: Select PTM Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: f_{SUB}
110: PTCK rising edge clock
111: PTCK falling edge clock
These three bits are used to select the clock source for the PTM. The external pin clock source can be chosen to be active on the rising or falling edge. The clock source f_{SYS} is the system clock, while f_H and f_{SUB} are other internal clocks, the details of which can be found in the oscillator section.

Bit 3 **PTON**: PTM Counter On/Off Control
0: Off
1: On
This bit controls the overall on/off function of the PTM. Setting the bit high enables the counter to run, clearing the bit disables the PTM. Clearing this bit to zero will stop the counter from counting and turn off the PTM which will reduce its power consumption. When the bit changes state from low to high the internal counter value will be reset to zero, however when the bit changes from high to low, the internal counter will retain its residual value until the bit returns high again.
If the PTM is in the Compare Match Output Mode, PWM output Mode or Single Pulse Output Mode then the PTM output pin will be reset to its initial condition, as specified by the PTOC bit, when the PTON bit changes from low to high.

Bit 2~0 Unimplemented, read as "0"

- **PTMC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	PTM1	PTM0	PTIO1	PTIO0	PTOC	PTPOL	PTCAPTS	PTCCLR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PTM1~PTM0**: Select PTM Operating Mode
00: Compare Match Output Mode
01: Capture Input Mode
10: PWM Output Mode or Single Pulse Output Mode
11: Timer/Counter Mode
These bits setup the required operating mode for the PTM. To ensure reliable operation the PTM should be switched off before any changes are made to the PTM1 and PTM0 bits. In the Timer/Counter Mode, the PTM output pin state is undefined.

Bit 5~4 **PTIO1~PTIO0**: Select PTM External Pins Function
Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output
PWM Output Mode/Single Pulse Output Mode
00: PWM output inactive state
01: PWM output active state
10: PWM output
11: Single pulse output
Capture Input Mode
PTTCLR[1:0]=00B:
00: Input capture at rising edge of PTPI or PTCK, and the counter value will be latched into CCRA
01: Input capture at falling edge of PTPI or PTCK, and the counter value will be latched into CCRA
10: Input capture at both falling and rising edges of PTPI or PTCK, and the counter value will be latched into CCRA
11: Input capture disabled
PTTCLR[1:0]=01B,10B or 11B:
00: Input capture at rising edge of PTPI or PTCK, and the counter value will be latched into CCRB
01: Input capture at falling edge of PTPI or PTCK, and the counter value will be latched into CCRA
10: Input capture at both falling and rising edges of PTPI or PTCK, and the counter value will be latched into CCRA at falling edge or CCRB at rising edge
11: Input capture disabled

Timer/Counter Mode
Unused

These two bits are used to determine how the PTM functions when a certain condition is reached. The function that these bits select depends upon in which mode the PTM is running.

In the Compare Match Output Mode, the PTIO1 and PTIO0 bits determine how the PTM output changes state when a compare match occurs from the Comparator A. The PTM output can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the bits are both zero, then no change will take place on the output. The initial value of the PTM output should be setup using the PTOC bit in the PTMC1 register. Note that the output level requested by the PTIO1 and PTIO0 bits must be different from the initial value setup using the PTOC bit otherwise no change will occur on the PTM output when a compare match occurs. After the PTM output changes state, it can be reset to its initial level by changing the level of the PTON bit from low to high.

In the PWM Output Mode, the PTIO1 and PTIO0 bits determine how the PTM output changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to only change the values of the PTIO1 and PTIO0 bits only after the TM has been switched off. Unpredictable PWM outputs will occur if the PTIO1 and PTIO0 bits are changed when the PTM is running.

Bit 3 **PTOC**: PTM PTP Output control bit

Compare Match Output Mode
0: Initial low
1: Initial high

PWM Output Mode/Single Pulse Output Mode
0: Active low
1: Active high

This is the output control bit for the PTM output. Its operation depends upon whether PTM is being used in the Compare Match Output Mode or in the PWM Output Mode/ Single Pulse Output Mode. It has no effect if the PTM is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the PTM output before a compare match occurs. In the PWM Output Mode it determines if the PWM signal is active high or active low. In the Single Pulse Output Mode it determines the logic level of the PTM output when the PTON bit changes from low to high.

Bit 2 **PTPOL**: PTM PTP Output polarity Control
0: Non-invert
1: Invert

This bit controls the polarity of the PTP output. When the bit is set high the PTM output will be inverted and not inverted when the bit is zero. It has no effect if the PTM is in the Timer/Counter Mode.

Bit 1 **PTCAPTS**: PTM Capture Trigger Source Selection
0: From PTPI input signal
1: From PTCK input

Bit 0 **PTCCLR**: Select PTM Counter clear condition
0: PTM Comparator P match
1: PTM Comparator A match

This bit is used to select the method which clears the counter. Remember that the Periodic TM contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the PTCCLR bit set high, the counter will be cleared when a compare match occurs from the Comparator A. When the bit is low, the counter will be cleared when a compare match occurs from the Comparator P or with a counter overflow. A counter overflow clearing method can only be implemented if the CCRP bits are all cleared to zero. The PTCCLR bit is not used in the PWM Output Mode, Single Pulse Output Mode or Capture Input Mode.

• **PTMC2 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	PTTCLR1	PTTCLR0	PTVLF
R/W	—	—	—	—	—	R/W	R/W	R
POR	—	—	—	—	—	0	0	0

Bit 7~3 Unimplemented, read as "0"

Bit 2~1 **PTTCLR1~PTTCLR0**: Select PTM Counter clear condition in capture input mode only
00: Comparator P match
01: Comparator P match or PTCK/PTPI rising edge
10: Comparator P match or PTCK/PTPI falling edge
11: Comparator P match or PTCK/PTPI dual edges
Note that these bits selection can be available only when the PTM operates in the Capture Input Mode.

Bit 0 **PTVLF**: PTM counter value latch edge flag
0: Falling edge trigger the counter value latch
1: Rising edge trigger the counter value latch
Note: When the PTTCLR1~PTTCLR0 bits equal to 00B, ignore this flag status.

• **PTMDL Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 **D7~D0**: PTM Counter Low Byte Register bit 7 ~ bit 0
PTM 10-bit Counter bit 7 ~ bit 0

• **PTMDH Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R	R
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **D9~D8**: PTM Counter High Byte Register bit 1 ~ bit 0
PTM 10-bit Counter bit 9 ~ bit 8

• **PTMAL Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **D7~D0**: PTM CCRA Low Byte Register bit 7 ~ bit 0
PTM 10-bit CCRA bit 7 ~ bit 0

• **PTMAH Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **D9~D8**: PTM CCRA High Byte Register bit 1 ~ bit 0
PTM 10-bit CCRA bit 9 ~ bit 8

• **PTMBL Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **D7~D0**: PTM CCRB Low Byte Register bit 7 ~ bit 0
PTM 10-bit CCRB bit 7 ~ bit 0

• **PTMBH Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **D9~D8**: PTM CCRB High Byte Register bit 1 ~ bit 0
PTM 10-bit CCRB bit 9 ~ bit 8

• **PTMRPL Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **D7~D0**: PTM CCRP Low Byte Register bit 7 ~ bit 0
PTM 10-bit CCRP bit 7 ~ bit 0

• **PTMRPH Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **D9~D8**: PTM CCRP High Byte Register bit 1 ~ bit 0
PTM 10-bit CCRP bit 9 ~ bit 8

Periodic Type TM Operating Modes

The Periodic Type TM can operate in one of five operating modes, Compare Match Output Mode, PWM Output Mode, Single Pulse Output Mode, Capture Input Mode or Timer/Counter Mode. The operating mode is selected using the PTM1 and PTM0 bits in the PTMC1 register.

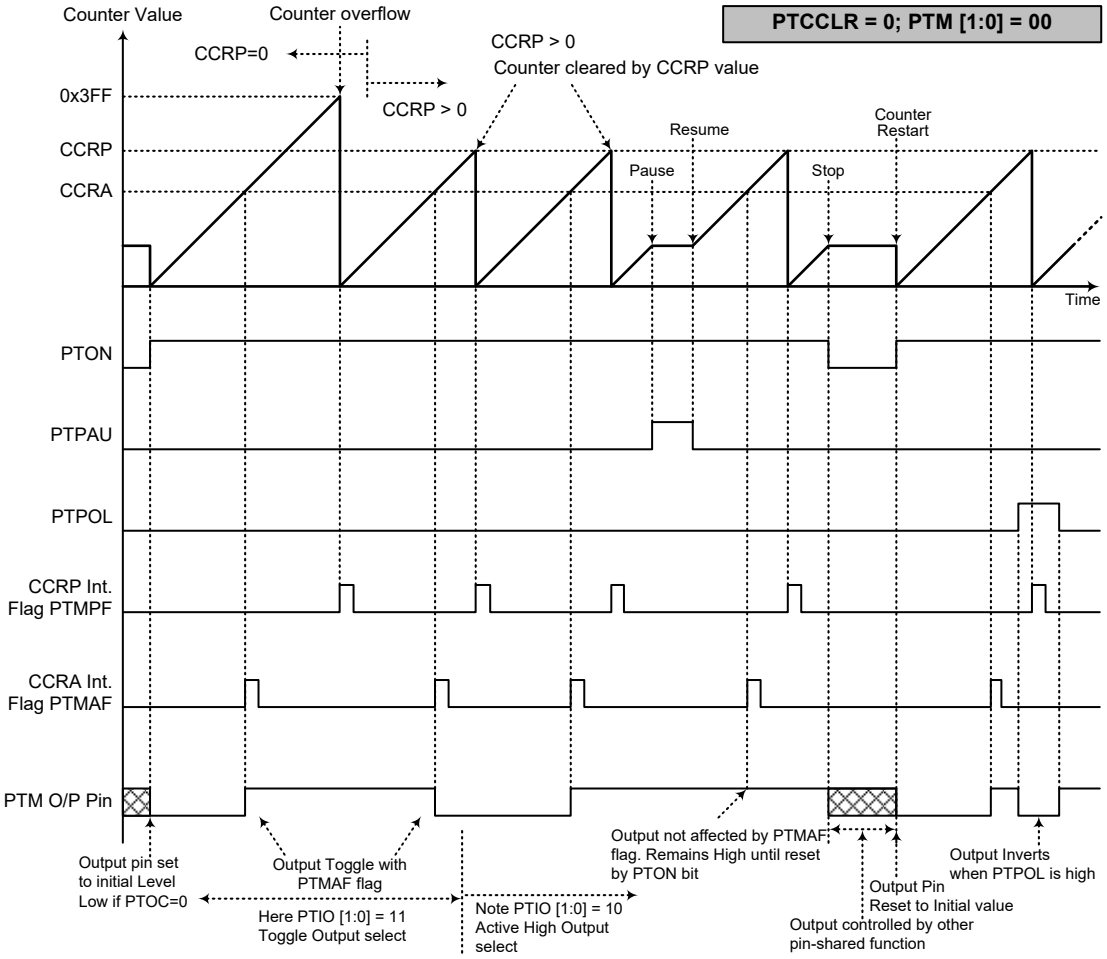
Compare Match Output Mode

To select this mode, bits PTM1 and PTM0 in the PTMC1 register, should be set to 00 respectively. In this mode once the counter is enabled and running it can be cleared by three methods. These are a counter overflow, a compare match from Comparator A and a compare match from Comparator P. When the PTCCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both PTMAF and PTMPF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

If the PTCCLR bit in the PTMC1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the PTMAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when PTCCLR is high no PTMPF interrupt request flag will be generated. In the Compare Match Output Mode, the CCRA can not be cleared to zero.

If the CCRA bits are all zero, the counter will overflow when its reaches its maximum 3FF Hex value, however here the PTMAF interrupt request flag will not be generated.

As the name of the mode suggests, after a comparison is made, the PTM output will change state. The PTM output condition however only changes state when a PTMAF interrupt request flag is generated after a compare match occurs from Comparator A. The PTMPF interrupt request flag, generated from a compare match occurs from Comparator P, will have no effect on the PTM output. The way in which the PTM output changes state are determined by the condition of the PTIO1 and PTIO0 bits in the PTMC1 register. The PTM output can be selected using the PTIO1 and PTIO0 bits to go high, to go low or to toggle from its present condition when a compare match occurs from Comparator A. The initial condition of the PTM output, which is setup after the PTON bit changes from low to high, is setup using the PTOC bit. Note that if the PTIO1 and PTIO0 bits are zero then no output change will take place.

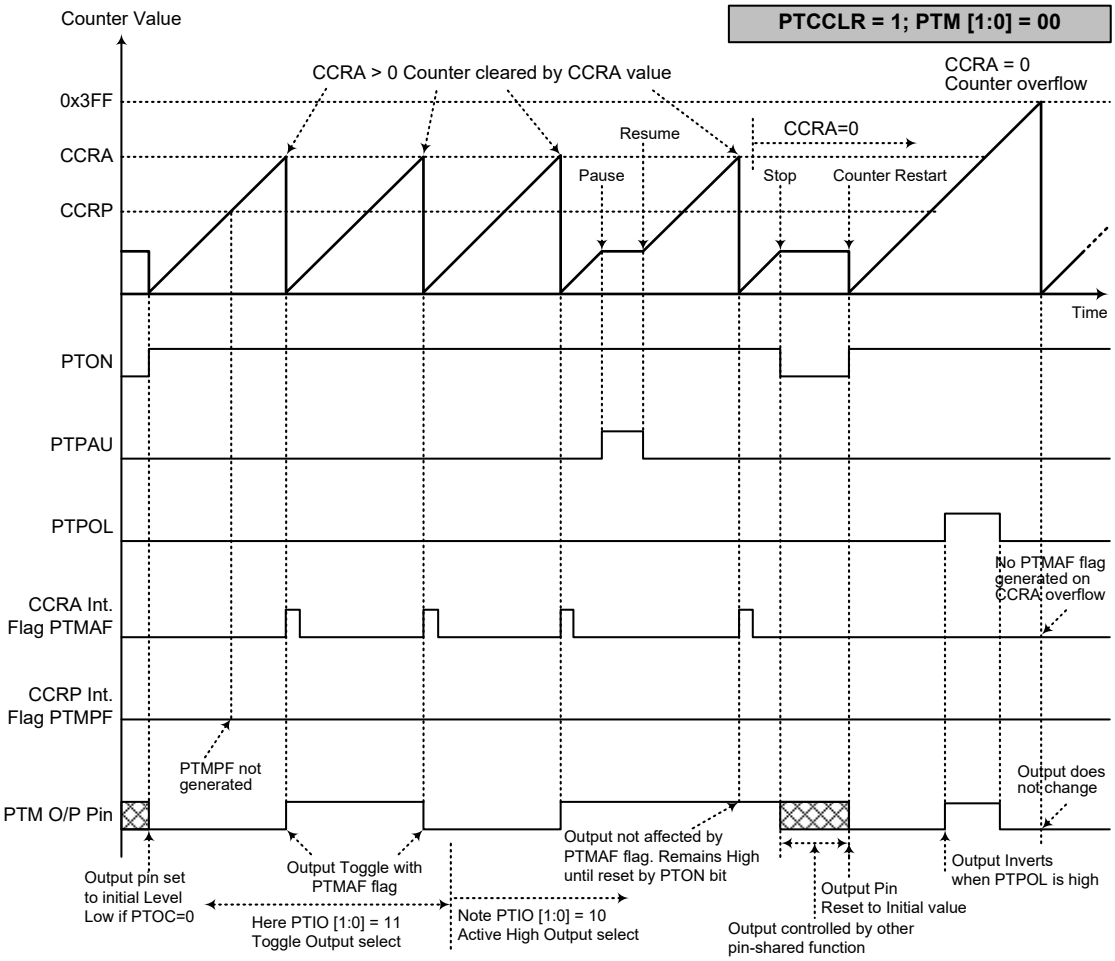


Compare Match Output Mode – PTCCLR=0

Note: 1. With PTCCLR=0 a Comparator P match will clear the counter

2. The PTM output is controlled only by the PTMAF flag

3. The output is reset to its initial state by a PTON bit rising edge



Compare Match Output Mode – PTCCLR=1

Note: 1. With PTCCLR=1 a Comparator A match will clear the counter

2. The PTM output is controlled only by the PTMAF flag

3. The output is reset to its initial state by a PTON bit rising edge

4. A PTMPF flag is not generated when PTCCLR=1

Timer/Counter Mode

To select this mode, bits PTM1 and PTM0 in the PTMC1 register should be set to 11 respectively. The Timer/Counter Mode operates in an identical way to the Compare Match Output Mode generating the same interrupt flags. The exception is that in the Timer/Counter Mode the PTM output pins are not used. Therefore the above description and Timing Diagrams for the Compare Match Output Mode can be used to understand its function. As the PTM output pins are not used in this mode, the pins can be used as normal I/O pins or other pin-shared function.

PWM Output Mode

To select this mode, bits PTM1 and PTM0 in the PTMC1 register should be set to 10 respectively. The PWM function within the PTM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the PTM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM Output Mode, the PTCCLR bit has no effect on the PWM operation. Both of the CCRA and CCRP registers are used to generate the PWM waveform, one register is used to clear the internal counter and thus control the PWM waveform frequency, while the other one is used to control the duty cycle. The PWM waveform frequency and duty cycle can therefore be controlled by the values in the CCRA and CCRP registers.

An interrupt flag, one for each of the CCRA and CCRP, will be generated when a compare match occurs from either Comparator A or Comparator P. The PTOC bit in the PTMC1 register is used to select the required polarity of the PWM waveform while the two PTIO1 and PTIO0 bits are used to enable the PWM output or to force the PTM output to a fixed high or low level. The PTPOL bit is used to reverse the polarity of the PWM output waveform.

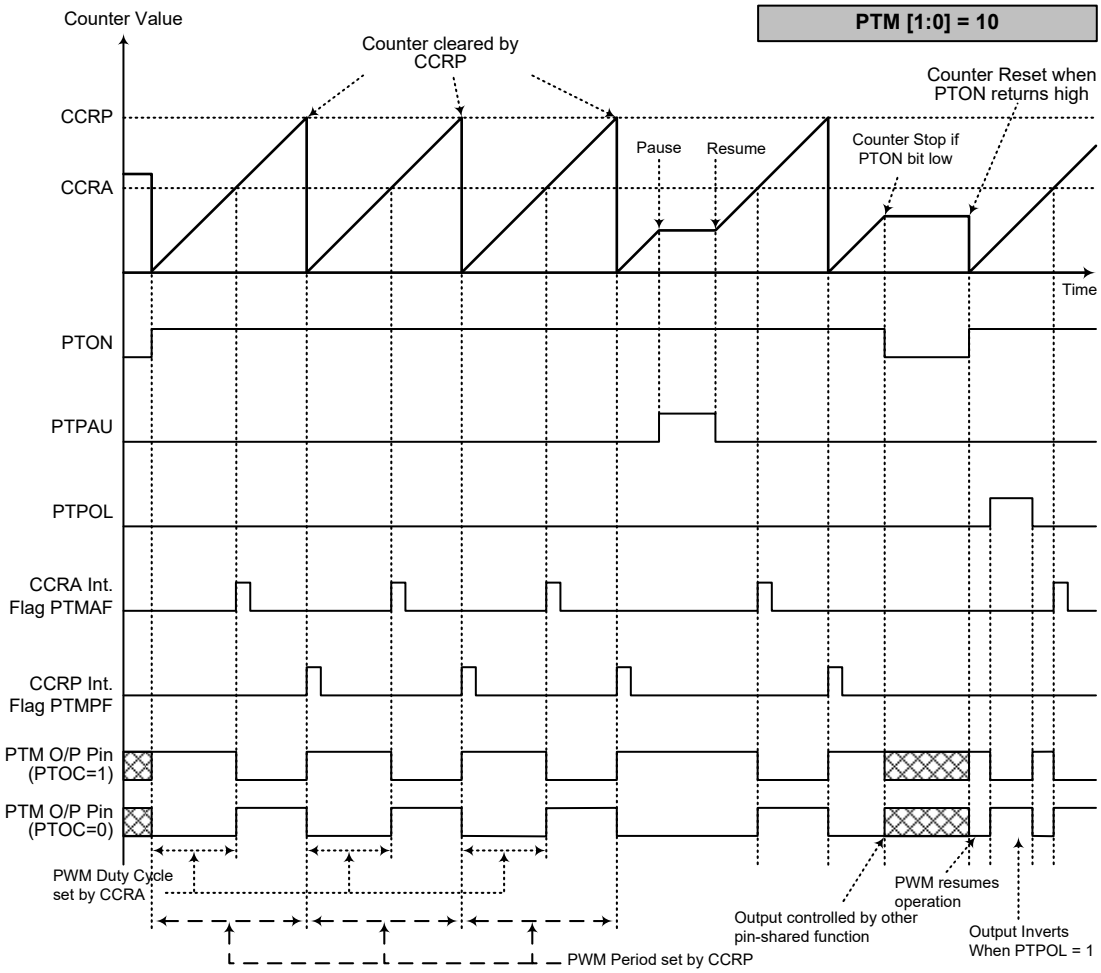
- **10-bit PTM, PWM Output Mode, Edge-aligned Mode**

CCRP	1~1023	0
Period	1~1023	1024
Duty	CCRA	

If f_{SYS}=8MHz, PTM clock source select f_{SYS}/4, CCRP=512 and CCRA=128,

The PTM PWM output frequency= (f_{SYS}/4)/512=f_{SYS}/2048=4kHz, duty=128/512=25%.

If the Duty value defined by the CCRA register is equal to or greater than the Period value, then the PWM output duty is 100%.



PWM Output Mode

Note: 1. Counter cleared by CCRP

2. A counter clear sets the PWM Period

3. The internal PWM function continues running even when PTIO[1:0]=00 or 01

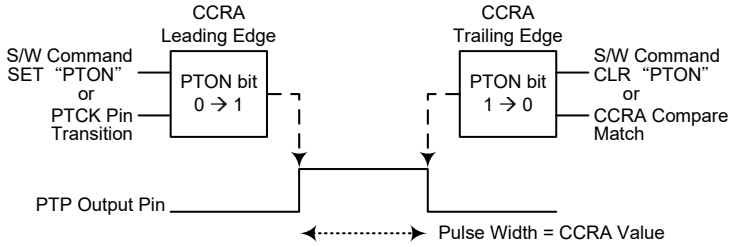
4. The PTCCLR bit has no influence on PWM operation

Single Pulse Output Mode

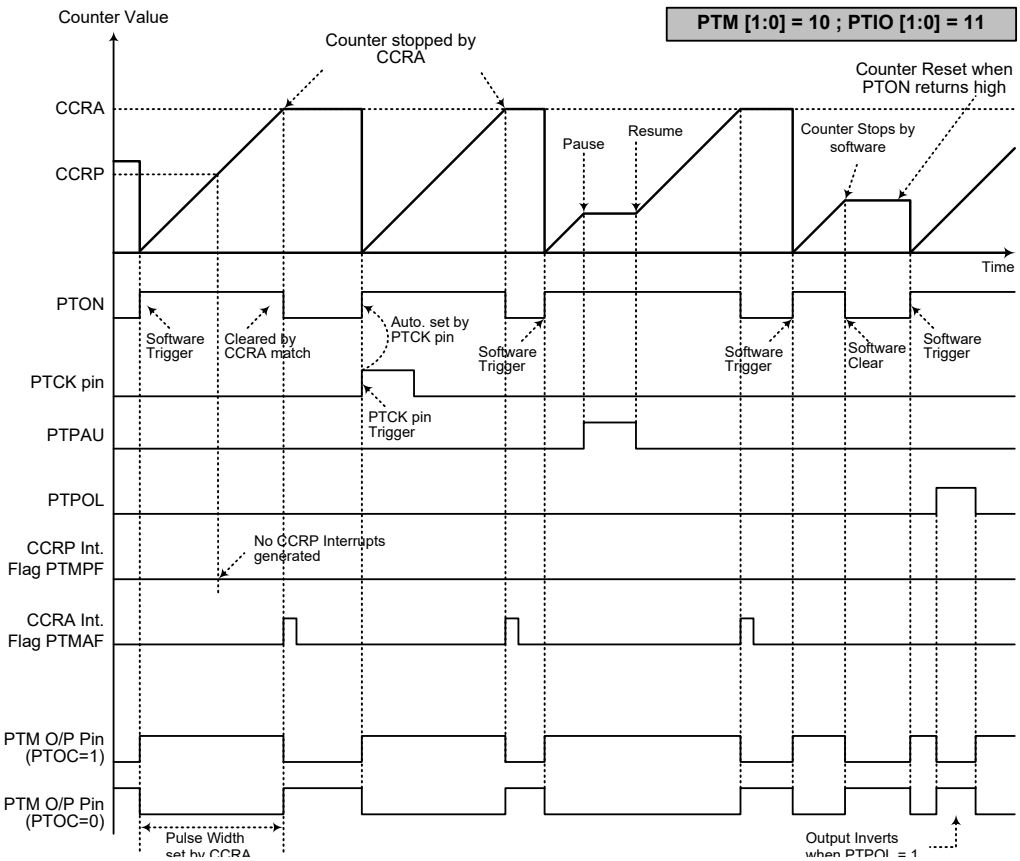
To select this mode, bits PTM1 and PTM0 in the PTMC1 register should be set to 10 respectively and also the PTIO1 and PTIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the PTM output pin.

The trigger for the pulse output leading edge is a low to high transition of the PTON bit, which can be implemented using the application program. However in the Single Pulse Mode, the PTON bit can also be made to automatically change from low to high using the external PTCK pin, which will in turn initiate the Single Pulse output. When the PTON bit transitions to a high level, the counter will start running and the pulse leading edge will be generated. The PTON bit should remain high when the pulse is in its active state. The generated pulse trailing edge will be generated when the PTON bit is cleared to zero, which can be implemented using the application program or when a compare match occurs from Comparator A.

However a compare match from Comparator A will also automatically clear the PTON bit and thus generate the Single Pulse output trailing edge. In this way the CCRA value can be used to control the pulse width. A compare match from Comparator A will also generate a PTM interrupt. The counter can only be reset back to zero when the PTON bit changes from low to high when the counter restarts. In the Single Pulse Output Mode CCRP is not used. The PTCCLR bit is not used in this Mode.



Single Pulse Generation



Single Pulse Output Mode

Note: 1. Counter stopped by CCRA

2. CCRP is not used

3. The pulse is triggered by the PTCK pin or by setting the PTON bit high

4. A PTCK pin active edge will automatically set the PTON bit high

5. In the Single Pulse Mode, PTIO[1:0] must be set to “11” and cannot be changed

Capture Input Mode

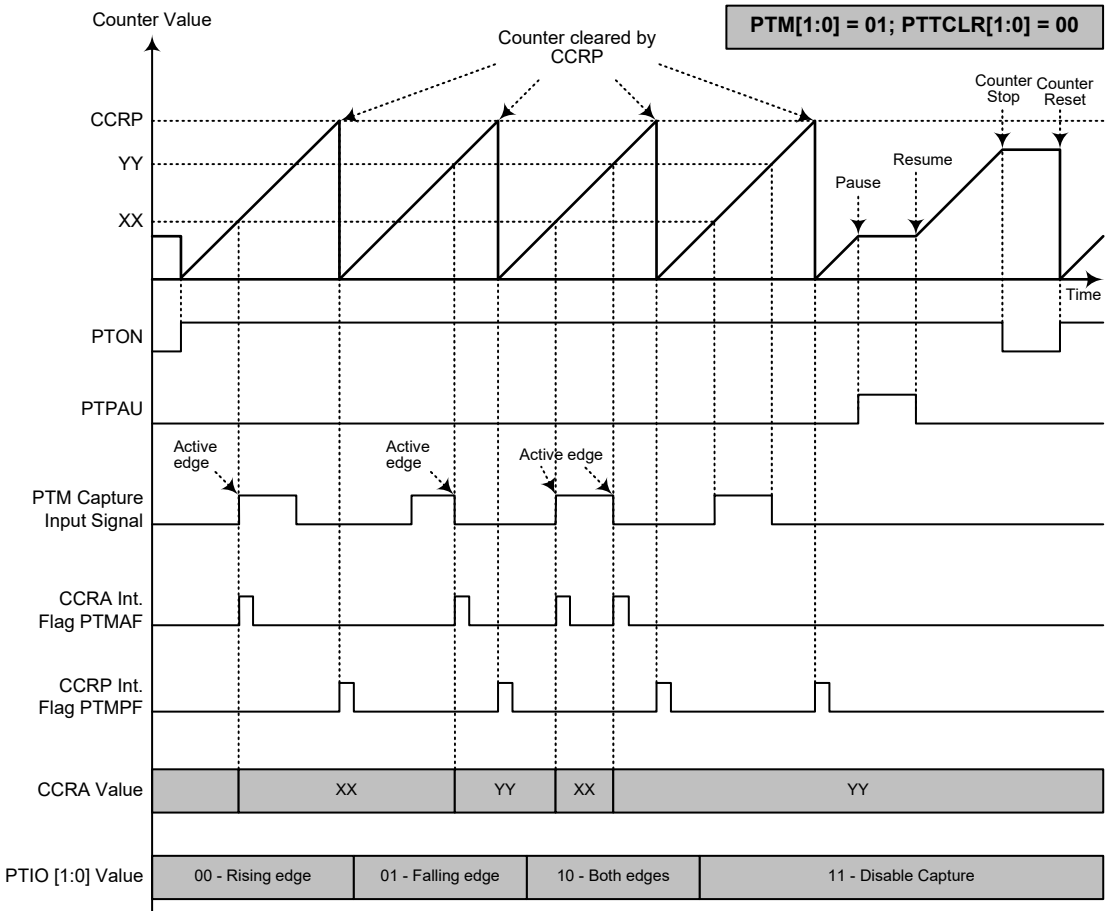
To select this mode bits PTM1 and PTM0 in the PTMC1 register should be set to 01 respectively. This mode enables external or internal signals to capture and store the present value of the internal counter and can therefore be used for applications such as pulse width measurements. The external or internal signal is supplied on the PTPI or PTCK pin which is selected using the IFS07~IFS06 bits in the IFS0 register and the PTCAPTS bit in the PTMC1 register. The input signal active edge can be either a rising edge, a falling edge or both rising and falling edges; the active edge transition type is selected using the PTIO1 and PTIO0 bits in the PTMC1 register. The counter is started when the PTON bit changes from low to high which is initiated using the application program.

The PTIO1 and PTIO0 bits decide which active edge transition type to be latched and interrupted. The PTTCLR1 and PTTCLR0 bits decide the condition that the counter reset back to zero. The present counter value latched into CCRA or CCRB is decided by both PTIO1~PTIO0 and PTTCLR1~PTTCLR0 setting. The PTIO1~PTIO0 and PTTCLR1~PTTCLR0 are independent on and uninfluenced each other.

When the required edge transition appears on the input signal, the present value in the counter will be latched into the CCRA registers or CCRB registers and a PTM interrupt generated. Irrespective of what events occur on the input signal, the counter will continue to free run until the PTON bit changes from high to low. When a CCRP compare match occurs the counter will reset back to zero; in this way the CCRP value can be used to control the maximum counter value. When a CCRP compare match occurs from Comparator P, a PTM interrupt will also be generated. Counting the number of overflow interrupt signals from the CCRP can be a useful method in measuring long pulse widths. The PTIO1 and PTIO0 bits can select the active trigger edge on the input signal to be a rising edge, falling edge or both edge types. If the PTIO1 and PTIO0 bits are both set high, then no capture operation will take place irrespective of what happens on the input signal, however it must be noted that the counter will continue to run.

If the capture pulse width is less than two timer clock cycles, it may be ignored by hardware. The timer clock source must be equal to or less than 50MHz, otherwise the counter may fail to count.

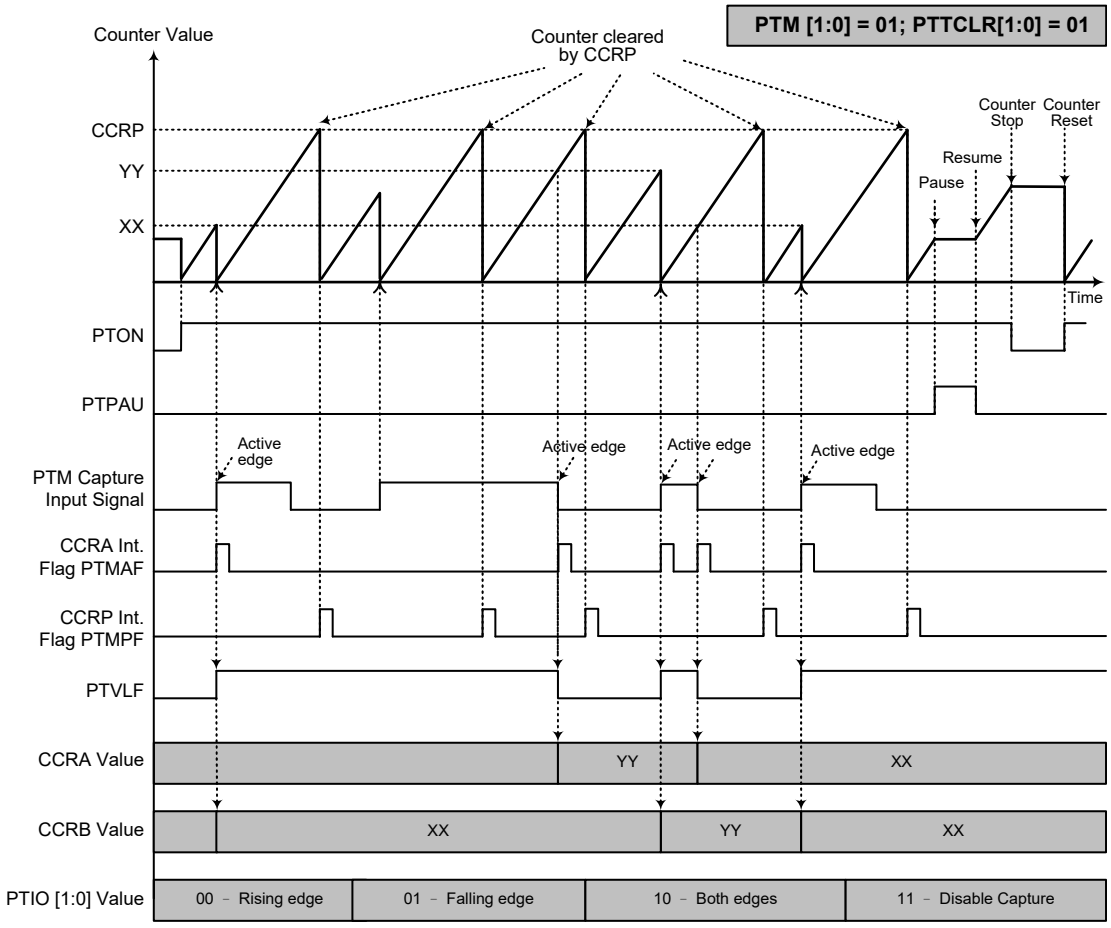
As the PTCK or PTPI pin is pin shared with other functions, care must be taken if the PTM is in the Capture Input Mode. This is because if the pin is setup as an output, then any transitions on this pin may cause an input capture operation to be executed. The PTCCLR, PTOC and PTPOL bits are not used in this Mode.



Capture Input Mode – PTTCLR[1:0]=00

Note: 1. PTM[1:0]=01, PTTCLR[1:0]=00 and active edge set by the PTIO[1:0] bits

2. A PTM Capture input active edge transfers the counter value to CCRA
3. Comparator P match will clear the counter
4. PTCCLR bit is not used
5. No output function – PTOC and PTPOL bits are not used
6. CCRP determines the counter value and the counter has a maximum count value when CCRP is equal to zero
7. Ignore the PTVLF bit status when PTTCLR[1:0]=00



Capture Input Mode – PTTCLR[1:0]=01

Note: 1. PTM[1:0]=01, PTTCLR[1:0]=01 and active edge set by the PTIO[1:0] bits

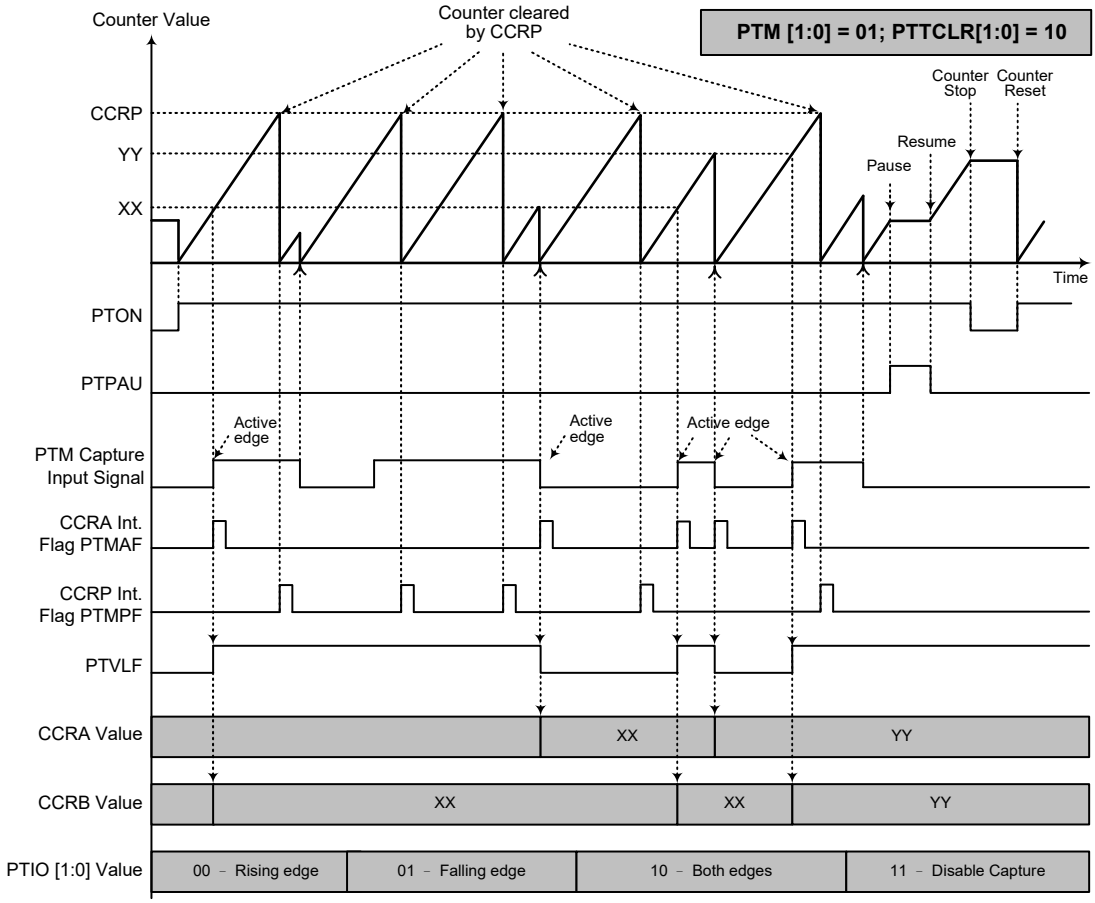
2. A PTM Capture input active edge transfers the counter value to CCRA or CCRB

3. Comparator P match or PTM capture input rising edge will clear the counter

4. PTCCLR bit is not used

5. No output function – PTOC and PTPOL bits are not used

6. CCRP determines the counter value and the counter has a maximum count value when CCRP is equal to zero



Capture Input Mode – PTTCLR[1:0]=10

Note: 1. PTM[1:0]=01, PTTCLR[1:0]=10 and active edge set by the PTIO[1:0] bits

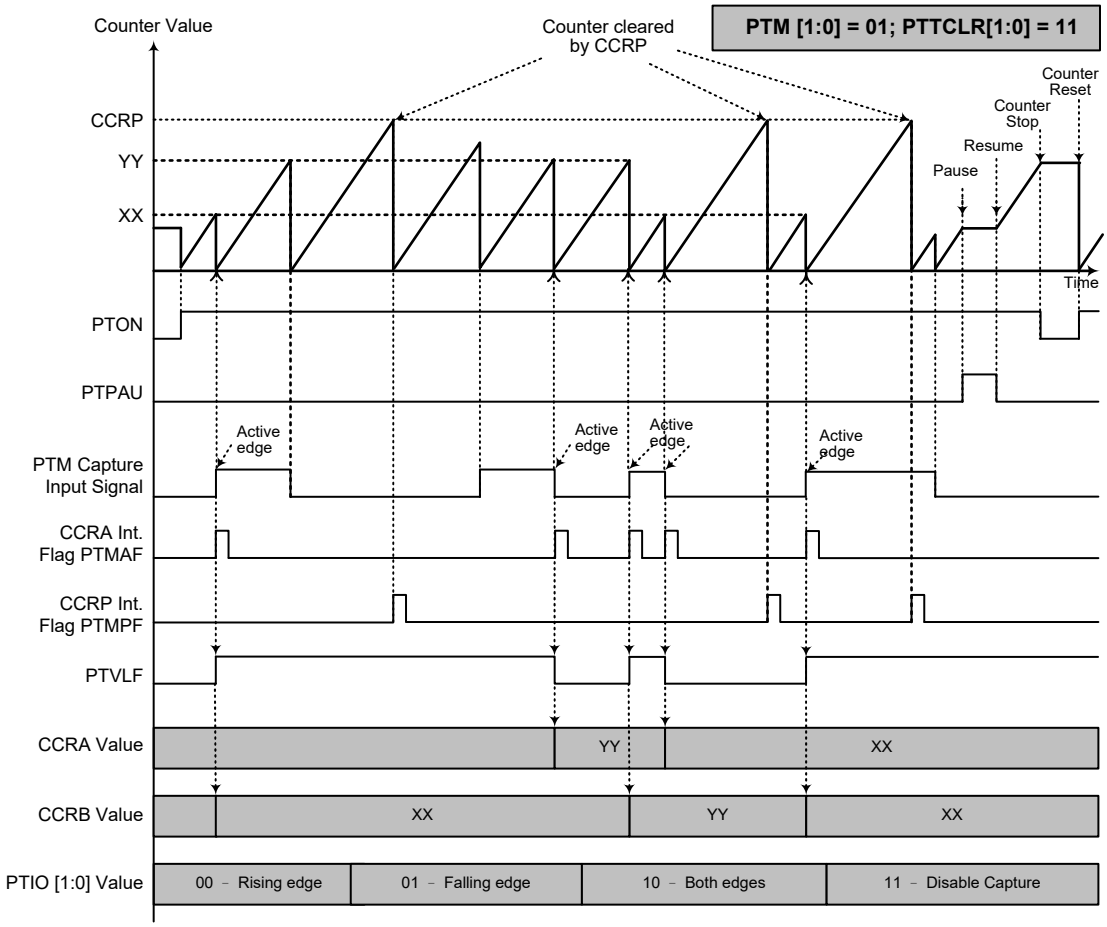
2. A PTM Capture input active edge transfers the counter value to CCRA or CCRB

3. Comparator P match or PTM capture input falling edge will clear the counter

4. PTCCLR bit is not used

5. No output function – PTOC and PTPOL bits are not used

6. CCRP determines the counter value and the counter has a maximum count value when CCRP is equal to zero



Capture Input Mode – PTTCLR[1:0]=11

Note: 1. PTM[1:0]=01, PTTCLR[1:0]=11 and active edge set by the PTIO[1:0] bits

2. A PTM Capture input active edge transfers the counter value to CCRA or CCRB

3. Comparator P match or PTM capture input rising or falling edge will clear the counter

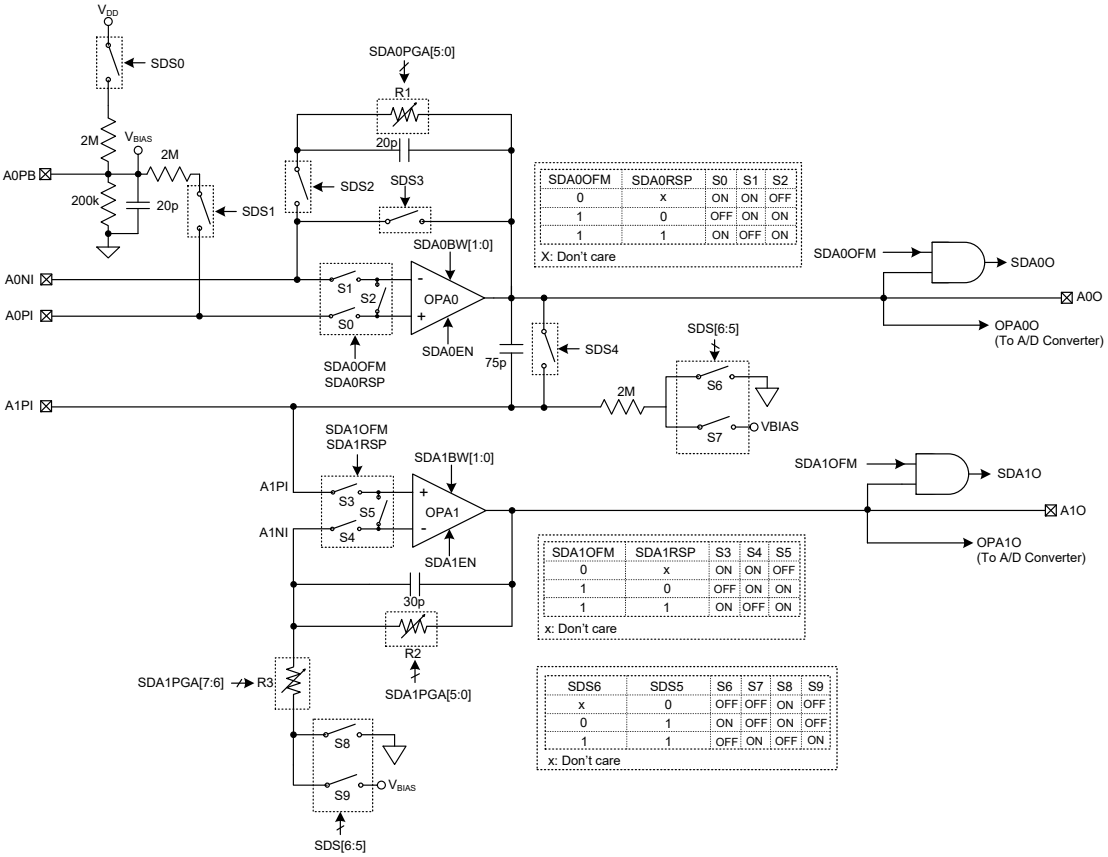
4. PTCCLR bit is not used

5. No output function – PTOC and PTPOL bits are not used

6. CCRP determines the counter value and the counter has a maximum count value when CCRP is equal to zero

Smoke Detector AFE

The device provides a Smoke Detector AFE circuit which can be used for optical signal detection in Smoke Detector applications. The circuit consists of two fully integrated Operational Amplifiers. The optical signal can be detected and processed by the operational amplifiers.



SDA0OFM	SDA0RSP	S0	S1	S2
0	x	ON	ON	OFF
1	0	OFF	ON	ON
1	1	ON	OFF	ON

X: Don't care

SDA1OFM	SDA1RSP	S3	S4	S5
0	x	ON	ON	OFF
1	0	OFF	ON	ON
1	1	ON	OFF	ON

x: Don't care

SDS6	SDS5	S6	S7	S8	S9
x	0	OFF	OFF	ON	OFF
0	1	ON	OFF	ON	OFF
1	1	OFF	ON	OFF	ON

x: Don't care

Smoke Detector AFE Block Diagram

Smoke Detector AFE Registers

Overall operation of the Smoke Detector AFE circuit is controlled using a series of registers. The SDSW register is used to control the switches on or off thus controlling the OPAs operating mode. The SDPGAC0 and SDPGAC1 registers are used to select the R1, R2 and R3 resistance. The SDAnC register where n=0~1, is used to control the SD OPAn enable/disable and bandwidth functions as well as stores the output status. The SDAnVOS register is used to select and control the SD OPAn input offset voltage calibration function.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SDSW	—	SDS6	SDS5	SDS4	SDS3	SDS2	SDS1	SDS0
SDPGAC0	—	—	SDA0PGA5	SDA0PGA4	SDA0PGA3	SDA0PGA2	SDA0PGA1	SDA0PGA0
SDPGAC1	SDA1PGA7	SDA1PGA6	SDA1PGA5	SDA1PGA4	SDA1PGA3	SDA1PGA2	SDA1PGA1	SDA1PGA0
SDA0C	—	SDA0EN	SDA0O	—	—	—	SDA0BW1	SDA0BW0
SDA1C	—	SDA1EN	SDA1O	—	—	—	SDA1BW1	SDA1BW0
SDA0VOS	SDA0OFM	SDA0RSP	SDA0OF5	SDA0OF4	SDA0OF3	SDA0OF2	SDA0OF1	SDA0OF0
SDA1VOS	SDA1OFM	SDA1RSP	SDA1OF5	SDA1OF4	SDA1OF3	SDA1OF2	SDA1OF1	SDA1OF0

Smoke Detector AFE Register List

• **SDSW Register**

Bit	7	6	5	4	3	2	1	0
Name	—	SDS6	SDS5	SDS4	SDS3	SDS2	SDS1	SDS0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7 Unimplemented, read as "0"

Bit 6~5 **SDS6~SDS5**: Mode control
00: External mode
01: AC coupling mode
10: External mode
11: DC coupling mode

Bit 4 **SDS4**: SDS4 switch on/off control
0: Off
1: On

Bit 3 **SDS3**: SDS3 switch on/off control
0: Off
1: On

Bit 2 **SDS2**: SDS2 switch on/off control
0: Off
1: On

Bit 1 **SDS1**: SDS1 switch on/off control
0: Off
1: On

Bit 0 **SDS0**: SDS0 switch on/off control
0: Off
1: On

• **SDPGAC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	SDA0PGA5	SDA0PGA4	SDA0PGA3	SDA0PGA2	SDA0PGA1	SDA0PGA0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5~0 **SDA0PGA5~SDA0PGA0**: R1 resistance control
R1=SDA0PGA[5:0]×100kΩ

• **SDPGAC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	SDA1PGA7	SDA1PGA6	SDA1PGA5	SDA1PGA4	SDA1PGA3	SDA1PGA2	SDA1PGA1	SDA1PGA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **SDA1PGA7~SDA1PGA6**: R3 resistance control
00: 10kΩ
01: 20kΩ
10: 30kΩ
11: 40kΩ

Bit 5~0 **SDA1PGA5~SDA1PGA0**: R2 resistance control
R2=SDA1PGA[5:0]×100kΩ

• **SDA0C Register**

Bit	7	6	5	4	3	2	1	0
Name	—	SDA0EN	SDA0O	—	—	—	SDA0BW1	SDA0BW0
R/W	—	R/W	R	—	—	—	R/W	R/W
POR	—	0	0	—	—	—	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **SDA0EN**: SD OPA0 enable or disable control
0: Disable
1: Enable

Bit 5 **SDA0O**: SD OPA0 output status (positive logic)
This bit is read only.
When SDA0OFM bit is set to 1, SDA0O is defined as SD OPA0 output status, refer to the "Operational Amplifier Input Offset Calibration" section for details.
When SDA0OFM bit is cleared to 0, this bit will be fixed at a low level.

Bit 4~2 Unimplemented, read as "0"

Bit 1~0 **SDA0BW1~SDA0BW0**: SD OPA0 bandwidth control
00: 5kHz
01: 40kHz
10: 600kHz
11: 2MHz
Refer to "Operational Amplifier Electrical Characteristics" for details.

• **SDA1C Register**

Bit	7	6	5	4	3	2	1	0
Name	—	SDA1EN	SDA1O	—	—	—	SDA1BW1	SDA1BW0
R/W	—	R/W	R	—	—	—	R/W	R/W
POR	—	0	0	—	—	—	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **SDA1EN**: SD OPA1 enable or disable control
0: Disable
1: Enable

Bit 5 **SDA1O**: SD OPA1 output status (positive logic)
This bit is read only.
When SDA1OFM bit is set to 1, SDA1O is defined as SD OPA1 output status, refer to the "Operational Amplifier Input Offset Calibration" section for details.
When SDA1OFM bit is cleared to 0, this bit will be fixed at a low level.

Bit 4~2 Unimplemented, read as "0"

Bit 1~0 **SDA1BW1~SDA1BW0**: SD OPA1 bandwidth control
00: 5kHz
01: 40kHz
10: 600kHz
11: 2MHz
Refer to "Operational Amplifier Electrical Characteristics" for details.

• **SDA0VOS Register**

Bit	7	6	5	4	3	2	1	0
Name	SDA0OFM	SDA0RSP	SDA0OF5	SDA0OF4	SDA0OF3	SDA0OF2	SDA0OF1	SDA0OF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	0	0	0	0	0

Bit 7 **SDA0OFM**: SD OPA0 normal operation or input offset voltage calibration mode selection
0: Normal operation
1: Offset calibration mode

Bit 6 **SDA0RSP**: SD OPA0 input offset voltage calibration reference selection
0: Input reference voltage comes from A0NI
1: Input reference voltage comes from A0PI

Bit 5~0 **SDA0OF5~SDA0OF0**: SD OPA0 input offset voltage calibration control
This 6-bit field is used to perform the operational amplifier input offset calibration operation and the value for the SD OPA0 input offset Calibration can be restored into this bit field. More detailed information is described in the "Operational Amplifier Input Offset Calibration" section.

• **SDA1VOS Register**

Bit	7	6	5	4	3	2	1	0
Name	SDA1OFM	SDA1RSP	SDA1OF5	SDA1OF4	SDA1OF3	SDA1OF2	SDA1OF1	SDA1OF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	0	0	0	0	0

Bit 7 **SDA1OFM**: SD OPA1 normal operation or input offset voltage calibration mode selection
0: Normal operation
1: Offset calibration mode

Bit 6 **SDA1RSP**: SD OPA1 input offset voltage calibration reference selection
0: Input reference voltage comes from A1NI
1: Input reference voltage comes from A1PI

Bit 5~0 **SDA1OF5~SDA1OF0**: SD OPA1 input offset voltage calibration control
This 6-bit field is used to perform the operational amplifier input offset calibration operation and the value for the SD OPA1 input offset Calibration can be restored into this bit field. More detailed information is described in the "Operational Amplifier Input Offset Calibration" section.

Operational Amplifier Operation

There are two fully integrated Operational Amplifiers in the device, OPA1 and OPA0. These OPAs can be used for signal amplification according to specific user requirements. The OPAs can be disabled or enabled entirely under software control using the internal registers. With specific control registers, some OPA related applications can be more flexible and easier to be implemented, such as Unit Gain Buffer, Non-Inverting Amplifier, Inverting Amplifier and various kinds of filters, etc.

Operational Amplifier Input Offset Calibration

Note that if the SD Operational Amplifier inputs are pin-shared with I/O pins, they should be configured as the SD Operational Amplifier input function before the Input Offset Calibration.

Step 1. Set SDAnOFM=1 and SDAnRSP=1, the SD Operational Amplifier n is now under the input offset Calibration mode. To make sure the V_{AnOS} as minimize as possible after calibration, the input reference voltage in calibration should be the same as input DC operating voltage in normal operation.

Step 2. Set SDAnOF[5:0]=000000 and then read the SDAnO bit.

Step 3. Increase the SDAnOF[5:0] value by 1 and then read the SDAnO bit.
If the SDAnO bit state has not changed, then repeat Step 3 until the SDAnO bit state has changed.
If the SDAnO bit state has changed, record the SDAnOF[5:0] value as V_{AnOS1} and then go to Step 4.

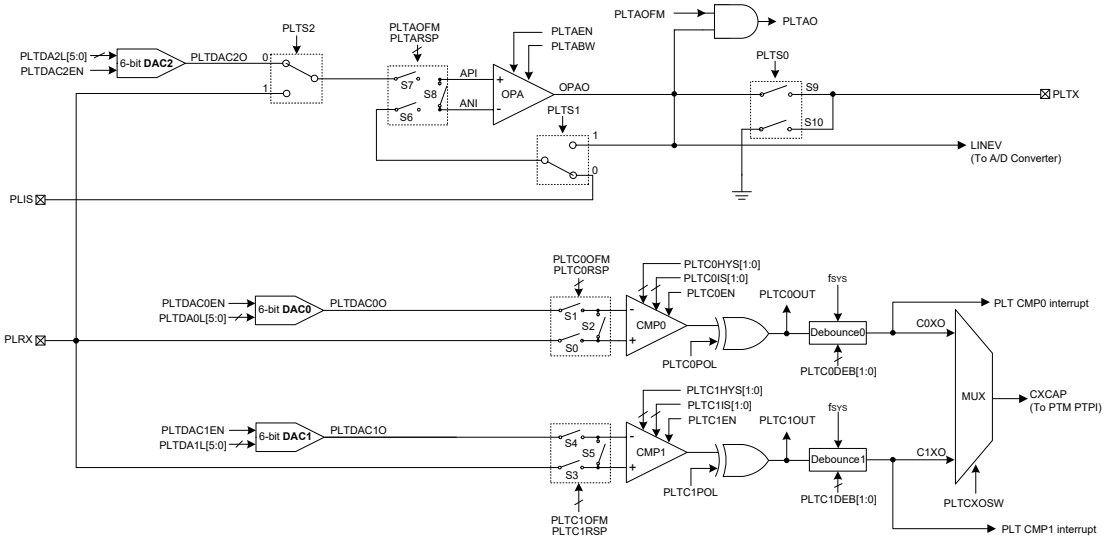
Step 4. Set SDAnOF[5:0]=111111 and read the SDAnO bit.

Step 5. Decrease the SDAnOF[5:0] value by 1 and then read the SDAnO bit.
If the SDAnO bit state has not changed, then repeat Step 5 until the SDAnO bit state has changed.
If the SDAnO bit state has changed, record the SDAnOF[5:0] value as V_{AnOS2} and then go to Step 6.

Step 6. Restore the SD Operational Amplifier n input offset calibration value V_{AnOS} into the SDAnOF[5:0] bit field. The offset Calibration procedure is now finished.
$V_{AnOS}=(V_{AnOS1}+V_{AnOS2})/2$.
If $(V_{AnOS1}+V_{AnOS2})/2$ is not integral, discard the decimal.

PowerLine Transceiver − PLT

The device provides a powerline transceiver circuit which can be used for power line data transmission and reception. The circuit consists of three 6-bit D/A converters, one fully integrated Operational Amplifier and two Comparators.



PowerLine Transceiver Block Diagram

PowerLine Transceiver Registers

Overall operation of the PowerLine Transceiver circuit is controlled using a series of registers. The DACn outputs, the OPA, Comparator input signal selection, operating modes, output signals all can be setup using these registers by application program.

Register Name	Bit							
	7	6	5	4	3	2	1	0
PLTSW	—	—	—	—	—	PLTS2	PLTS1	PLTS0
PLTDACC	—	—	—	—	—	PLTDAC2EN	PLTDAC1EN	PLTDAC0EN
PLTDA0L	—	—	D5	D4	D3	D2	D1	D0
PLTDA1L	—	—	D5	D4	D3	D2	D1	D0
PLTDA2L	—	—	D5	D4	D3	D2	D1	D0
PLTC0C	PLTC0OUT	PLTC0EN	PLTC0O	—	PLTC0DEB1	PLTC0DEB0	PLTC0IS1	PLTC0IS0
PLTC0VOS	—	PLTC0OFM	PLTC0RSP	PLTC0OF4	PLTC0OF3	PLTC0OF2	PLTC0OF1	PLTC0OF0
PLTC1C	PLTC1OUT	PLTC1EN	PLTC1O	—	PLTC1DEB1	PLTC1DEB0	PLTC1IS1	PLTC1IS0
PLTC1VOS	—	PLTC1OFM	PLTC1RSP	PLTC1OF4	PLTC1OF3	PLTC1OF2	PLTC1OF1	PLTC1OF0
PLTCHYC	—	PLTCXOSW	PLTC1POL	PLTC0POL	PLTC1HYS1	PLTC1HYS0	PLTC0HYS1	PLTC0HYS0
PLTAC	—	PLTAEN	PLTAO	—	—	—	—	PLTABW
PLTAVOS	PLTAOFM	PLTARSP	PLTAOF5	PLTAOF4	PLTAOF3	PLTAOF2	PLTAOF1	PLTAOF0

PowerLine Transceiver Register List

• PLTSW Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	PLTS2	PLTS1	PLTS0
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	1

Bit 7~3 Unimplemented, read as "0"

Bit 2 **PLTS2**: PLTS2 switch selection
0: Connect to PLTDAC2O
1: Connect to PLRX

Bit 1 **PLTS1**: PLTS1 switch selection
0: Connect to PLIS
1: Connect to LINEV

Bit 0 **PLTS0**: PLTX connection selection
0: Connect to GND
1: Connect to OPA output, OPAO

• PLTDACC Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	PLTDAC2EN	PLTDAC1EN	PLTDAC0EN
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 Unimplemented, read as "0"

Bit 2 **PLTDAC2EN**: PLT DAC2 enable or disable control
0: Disable – PLTDAC2O high impedance
1: Enable

Bit 1 **PLTDAC1EN**: PLT DAC1 enable or disable control
0: Disable – PLTDAC1O high impedance
1: Enable

Bit 0 **PLTDAC0EN**: PLT DAC0 enable or disable control
0: Disable – PLTDAC0O high impedance
1: Enable

• **PLTDA0L Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	D5	D4	D3	D2	D1	D0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5~0 **D5~D0**: PLT DAC0 output control code
PLTDAC0O=(DAC $V_{DD}/2^6$)×PLTDA0L[5:0]

• **PLTDA1L Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	D5	D4	D3	D2	D1	D0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5~0 **D5~D0**: PLT DAC1 output control code
PLTDAC1O=(DAC $V_{DD}/2^6$)×PLTDA1L[5:0]

• **PLTDA2L Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	D5	D4	D3	D2	D1	D0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5~0 **D5~D0**: PLT DAC2 output control code
PLTDAC2O=(DAC $V_{DD}/2^6$)×PLTDA2L[5:0]

• **PLTC0C Register**

Bit	7	6	5	4	3	2	1	0
Name	PLTC0OUT	PLTC0EN	PLTC0O	—	PLTC0DEB1	PLTC0DEB0	PLTC0IS1	PLTC0IS0
R/W	R	R/W	R	—	R/W	R/W	R/W	R/W
POR	0	0	0	—	0	0	0	0

Bit 7 **PLTC0OUT**: PLT Comparator 0 output bit
If PLTC0POL=0 and input voltages of the comparator are
C0PI > C0NI → PLTC0OUT=1
C0NI > C0PI → PLTC0OUT=0
If PLTC0POL=1 and input voltages of the comparator are
C0PI < C0NI → PLTC0OUT=1
C0NI < C0PI → PLTC0OUT=0

Bit 6 **PLTC0EN**: PLT Comparator 0 enable or disable control
0: Disable
1: Enable
This is the PLT Comparator 0 on/off control bit. The Comparator output will be set to 0 when it is disabled. Therefore PLTC0OUT is set to 0 when PLTC0POL=0 or PLTC0OUT is set to 1 when PLTC0POL=1.

Bit 5 **PLTC0O**: PLT Comparator 0 debounced output
The PLTC0O is de-bounced version of PLTC0OUT
If PLTC0POL=0, the PLTC0O outputs "1" only when the current and the previous N samples of PLTC0OUT are all "1".
If PLTC0POL=1, The PLTC0O outputs "0" only when the current and the previous N samples of PLTC0OUT are all "0". The sampling frequency is depend on the PLTC0DEB[1:0] bit configuration.

Bit 4 Unimplemented, read as "0"

Bit 3~2 **PLTC0DEB1~PLTC0DEB0**: PLT Comparator 0 debounce time control
00: No debounce
01: (31~32)×t_{SYS}
10: (63~64)×t_{SYS}
11: (126~127)×t_{SYS}
Note: $t_{SYS}=1/f_{SYS}$

Bit 1~0 **PLTC0IS1~PLTC0IS0**: PLT Comparator 0 current control
Refer to the "Comparator Electrical Characteristics" table for details.

• PLTC1C Register

Bit	7	6	5	4	3	2	1	0
Name	PLTC1OUT	PLTC1EN	PLTC1O	—	PLTC1DEB1	PLTC1DEB0	PLTC1IS1	PLTC1IS0
R/W	R	R/W	R	—	R/W	R/W	R/W	R/W
POR	0	0	0	—	0	0	0	0

Bit 7 **PLTC1OUT**: PLT Comparator 1 output bit
If PLTC1POL=0 and input voltages of the comparator are
C1PI > C1NI → PLTC1OUT=1
C1NI > C1PI → PLTC1OUT=0
If PLTC1POL=1 and input voltages of the comparator are
C1PI < C1NI → PLTC1OUT=1
C1NI < C1PI → PLTC1OUT=0

Bit 6 **PLTC1EN**: PLT Comparator 1 enable or disable control
0: Disable
1: Enable
This is the PLT Comparator 1 on/off control bit. The Comparator output will be set to 0 when it is disabled. Therefore PLTC1OUT is set to 0 when PLTC1POL=0 or PLTC1OUT is set to 1 when PLTC1POL=1.

Bit 5 **PLTC1O**: PLT Comparator 1 debounced output
The PLTC1O is de-bounced version of PLTC1OUT
If PLTC1POL=0, the PLTC1O outputs "1" only when the current and the previous N samples of PLTC1OUT are all "1". If PLTC1POL=1, The PLTC1O outputs "0" only when the current and the previous N samples of PLTC1OUT are all "0". The sampling frequency is depend on the PLTC1DEB[1:0] bit configuration.

Bit 4 Unimplemented, read as "0"

Bit 3~2 **PLTC1DEB1~PLTC1DEB0**: PLT Comparator 1 debounce time control
00: No debounce
01: (31~32)×t_{SYS}
10: (63~64)×t_{SYS}
11: (126~127)×t_{SYS}
Note: $t_{SYS}=1/f_{SYS}$

Bit 1~0 **PLTC1IS1~PLTC1IS0**: PLT Comparator 1 current control
Refer to the "Comparator Electrical Characteristics" table for details.

• **PLTC0VOS Register**

Bit	7	6	5	4	3	2	1	0
Name	—	PLTC0OFM	PLTC0RSP	PLTC0OF4	PLTC0OF3	PLTC0OF2	PLTC0OF1	PLTC0OF0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	1	0	0	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **PLTC0OFM**: PLT Comparator 0 normal operation or input offset voltage calibration mode selection
0: Normal operation
1: Offset calibration mode

Bit 5 **PLTC0RSP**: PLT Comparator 0 input offset voltage calibration reference selection
0: Input reference voltage comes from C0NI
1: Input reference voltage comes from C0PI

Bit 4~0 **PLTC0OF4~PLTC0OF0**: PLT Comparator 0 input offset voltage calibration control
This 5-bit field is used to perform the PLT comparator 0 input offset calibration operation and the value for the PLT Comparator 0 input offset calibration can be restored into this bit field. More detailed information is described in the "Comparator Input Offset Calibration" section.

• **PLTC1VOS Register**

Bit	7	6	5	4	3	2	1	0
Name	—	PLTC1OFM	PLTC1RSP	PLTC1OF4	PLTC1OF3	PLTC1OF2	PLTC1OF1	PLTC1OF0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	1	0	0	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **PLTC1OFM**: PLT Comparator 1 normal operation or input offset voltage calibration mode selection
0: Normal operation
1: Offset calibration mode

Bit 5 **PLTC1RSP**: PLT Comparator 1 input offset voltage calibration reference selection
0: Input reference voltage comes from C1NI
1: Input reference voltage comes from C1PI

Bit 4~0 **PLTC1OF4~PLTC1OF0**: PLT Comparator 1 input offset voltage calibration control
This 5-bit field is used to perform the PLT Comparator 1 input offset calibration operation and the value for the PLT Comparator 1 input offset calibration can be restored into this bit field. More detailed information is described in the "Comparator Input Offset Calibration" section.

• **PLTCHYC Register**

Bit	7	6	5	4	3	2	1	0
Name	—	PLTCXOSW	PLTC1POL	PLTC0POL	PLTC1HYS1	PLTC1HYS0	PLTC0HYS1	PLTC0HYS0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **PLTCXOSW**: Comparator 0 or Comparator 1 output selection
0: Comparator 0 Output
1: Comparator 1 Output
This is the Comparator 0 or Comparator 1 output control bit. If the bit is zero then the PLTC0O bit will be output. If the bit is high the PLTC1O bit will be output.

Bit 5 **PLTC1POL**: PLT Comparator 1 output polarity control
0: Non-invert
1: Invert
This is the PLT Comparator 1 polarity control bit. If the bit is zero then the PLTC1OUT bit will reflect the non-inverted output condition of the comparator 1. If the bit is high the comparator PLTC1OUT bit will be the inverted output condition of the comparator 1.

Bit 4 **PLTC0POL**: PLT Comparator 0 output polarity control
0: Non-invert
1: Invert
This is the PLT Comparator 0 polarity control bit. If the bit is zero then the PLTC0OUT bit will reflect the non-inverted output condition of the comparator 0. If the bit is high the comparator PLTC0OUT bit will be the inverted output condition of the comparator 0.

Bit 3~2 **PLTC1HYS1~PLTC1HYS0**: PLT Comparator 1 hysteresis voltage window control
Refer to "Comparator Electrical Characteristics" table for details.

Bit 1~0 **PLTC0HYS1~PLTC0HYS0**: PLT Comparator 0 hysteresis voltage window control
Refer to "Comparator Electrical Characteristics" table for details.

• **PLTAC Register**

Bit	7	6	5	4	3	2	1	0
Name	—	PLTAEN	PLTAO	—	—	—	—	PLTABW
R/W	—	R/W	R	—	—	—	—	R/W
POR	—	0	0	—	—	—	—	0

Bit 7 Unimplemented, read as "0"

Bit 6 **PLTAEN**: PLT OPA enable or disable control
0: Disable
1: Enable

Bit 5 **PLTAO**: PLT OPA output status (positive logic)
When PLTAOFM bit is set to 1, PLTAO is defined as PLT OPA output status, please refer to Offset calibration procedure. When PLTAOFM bit is cleared to 0, this bit will be fixed at a low level.

Bit 4~1 Unimplemented, read as "0"

Bit 0 **PLTABW**: PLT OPA Gain bandwidth control bit
0: 600kHz
1: 2MHz
Refer to "Operational Amplifier Electrical Characteristics" table for more details.

• **PLTAVOS Register**

Bit	7	6	5	4	3	2	1	0
Name	PLTAOFM	PLTARSP	PLTAOF5	PLTAOF4	PLTAOF3	PLTAOF2	PLTAOF1	PLTAOF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	0	0	0	0	0

Bit 7 **PLTAOFM**: PLT OPA normal operation or input offset voltage calibration mode selection
0: Normal operation
1: Offset calibration mode

Bit 6 **PLTARSP**: PLT OPA input offset voltage calibration reference selection
0: Input reference voltage comes from ANI
1: Input reference voltage comes from API

Bit 5~0 **PLTAOF5~PLTAOF0**: PLT OPA input offset voltage calibration control
This 6-bit field is used to perform the PLT OPA input offset calibration operation and the value for the PLT OPA input offset calibration can be restored into this bit field. More detailed information is described in the "Operational Amplifier Input Offset Calibration" section.

Offset Calibration Procedure

To operate in the input offset calibration mode for the PLT Operational Amplifier or the Comparators, the PLTAOFM or PLTCnOFM bit should first be set to "1" to select the input offset voltage calibration mode. Note that as the comparator or OPA input is from the PLRX or PLIS pin which is pin shared with I/O or other functions, before the calibration, they should be configured as PLT comparator or operational amplifier input pin function first.

Comparator Input Offset Calibration

- Step 1
  Set PLTCnOFM=1, PLTCnRSP=1, the PLT Comparator n is now operating in the comparator input offset calibration mode, S0 and S2 on or S3 and S5 on. To make sure V_{CnOS} as minimal as possible after calibration, the input reference voltage in calibration should be the same as input DC operating voltage in normal operation.
- Step 2
  Set PLTCnOF[4:0]=00000 and read the PLTCnOUT bit.
- Step 3
  Increase the PLTCnOF[4:0] value by 1 and then read the PLTCnOUT bit.
  If the PLTCnOUT bit state has not changed, then repeat Step 3 until the PLTCnOUT bit state has changed.
  If the PLTCnOUT bit state has changed, record the PLTCnOF[4:0] value as V_{CnOS1} and then go to Step 4.
- Step 4
  Set PLTCnOF[4:0]=11111 and then read the PLTCnOUT bit.
- Step 5
  Decrease the PLTCnOF[4:0] value by 1 and then read the PLTCnOUT bit.
  If the PLTCnOUT bit state has not changed, then repeat Step 5 until the PLTCnOUT bit state has changed.
  If the PLTCnOUT bit state has changed, record the PLTCnOF[4:0] value as V_{CnOS2} and then go to Step 6.
- Step 6
  Restore the PLT Comparator n input offset calibration value V_{CnOS} into the PLTCnOF[4:0] bit field. The offset Calibration procedure is now finished.
  Where $V_{CnOS}= (V_{CnOS1}+V_{CnOS2})/2$. If $(V_{CnOS1}+V_{CnOS2})/2$ is not integral, discard the decimal.

Operational Amplifier Input Offset Calibration

- Step 1
  Set PLTAOFM=1, PTLARSP=1, the PLT Operational Amplifier is now under the input offset Calibration mode, S6 and S8 on. To make sure the V_{AOS} as minimal as possible after calibration, the input reference voltage in calibration should be the same as input DC operating voltage in normal operation.
- Step 2
  Set PLTAOF[5:0]=000000 and then read the PLTAO bit.
- Step 3
  Increase the PLTAOF[5:0] value by 1 and then read the PLTAO bit.
  If the PLTAO bit state has not changed, then repeat Step 3 until the PLTAO bit state has changed.
  If the PLTAO bit state has changed, record the PLTAOF[5:0] value as V_{AOS1} and then go to Step 4.
- Step 4
  Set PLTAOF[5:0]=111111 and read the PLTAO bit.
- Step 5
  Decrease the PLTAOF[5:0] value by 1 and then read the PLTAO bit.
  If the PLTAO bit state has not changed, then repeat Step 5 until the PLTAO bit state has changed.
  If the PLTAO bit state has changed, record the PLTAOF[5:0] value as V_{AOS2} and then go to Step 6.
- Step 6
  Restore the PLT Operational Amplifier input offset calibration value V_{AOS} into the PLTAOF[5:0] bit field. The offset Calibration procedure is now finished.
  $V_{AOS} = (V_{AOS1}+V_{AOS2})/2$.
  If $(V_{AOS1}+V_{AOS2})/2$ is not integral, discard the decimal.

Analog to Digital Converter

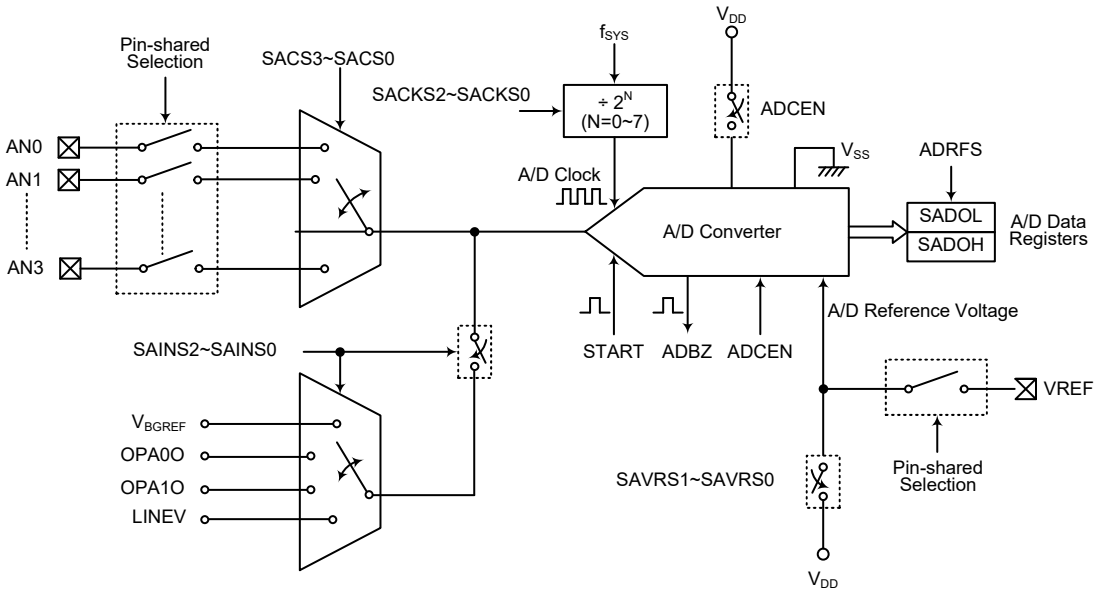
The need to interface to real world analog signals is a common requirement for many electronic systems. However, to properly process these signals by a microcontroller, they must first be converted into digital signals by A/D converters. By integrating the A/D conversion electronic circuitry into the microcontroller, the need for external components is reduced significantly with the corresponding follow-on benefits of lower costs and reduced component space requirements.

A/D Converter Overview

The device contains a multi-channel analog to digital converter which can directly interface to external analog signals, such as that from sensors or other control signals and convert these signals directly into a 12-bit digital value. It also can convert the internal signals, the high performance bandgap reference voltage, V_{BGREF}, the SD operational amplifier 0 output signal OPA0O, SD operational amplifier 1 output signal OPA1O and PLT operational amplifier output signal, LINEV, into a 12-bit digital value. The external or internal analog signal to be converted is determined by the SAINS2~SAINS0 bits together with the SACS3~SACS0 bits. More detailed information about the A/D input signal is described in the "A/D Converter Control Registers" and "A/D Converter Input Signals" sections respectively.

External Input Channels	Internal Signals	Channel Select Bits
AN0~AN3	V_{BGREF}, OPA0O, OPA1O, LINEV	SAINS2~SAINS0, SACS3~SACS0

The accompanying block diagram shows the overall internal structure of the A/D converter, together with its associated registers.



A/D Converter Structure

A/D Converter Register Description

Overall operation of the A/D converter is controlled using several registers. A read only register pair exists to store the A/D converter data 12-bit value. The remaining two registers are control registers which setup the operating and control function of the A/D converter.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SADOH (ADRFS=0)	D11	D10	D9	D8	D7	D6	D5	D4
SADOH (ADRFS=1)	—	—	—	—	D11	D10	D9	D8
SADOL (ADRFS=0)	D3	D2	D1	D0	—	—	—	—
SADOL (ADRFS=1)	D7	D6	D5	D4	D3	D2	D1	D0
SADC0	START	ADBZ	ADCEN	ADRFS	SACS3	SACS2	SACS1	SACS0
SADC1	SAINS2	SAINS1	SAINS0	SAVRS1	SAVRS0	SACKS2	SACKS1	SACKS0

A/D Converter Register List

A/D Converter Data Registers – SADOL, SADOH

As the device contains an internal 12-bit A/D converter, it requires two data registers to store the converted value. These are a high byte register, known as SADOH, and a low byte register, known as SADOL. After the conversion process takes place, these registers can be directly read by the microcontroller to obtain the digitised conversion value. As only 12 bits of the 16-bit register space is utilised, the format in which the data is stored is controlled by the ADRFS bit in the SADC0 register as shown in the accompanying table. D0~D11 are the A/D conversion result data bits. Any unused bits will be read as zero. Note that A/D Converter data register contents will be unchanged if the A/D converter is disabled.

ADRFS	SADOH								SADOL							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0
1	0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

A/D Data Registers

A/D Converter Control Registers – SADC0, SADC1

To control the function and operation of the A/D converter, two control registers known as SADC0 and SADC1 are provided. These 8-bit registers define functions such as the selection of which analog channel is connected to the internal A/D converter, the digitised data format, the A/D clock source as well as controlling the start function and monitoring the A/D converter busy status. As the device contains only one actual analog to digital converter hardware circuit, each of the external or internal analog signal inputs must be routed to the converter. The SACS3~SACS0 bits in the SADC0 register are used to determine which external channel input is selected to be converted. The SAINS2~SAINS0 bits in the SADC1 register are used to determine that the analog signal to be converted comes from the internal analog signal or external analog channel input.

The relevant pin-shared function selection bits determine which pins on I/O Ports are used as analog inputs for the A/D converter input and which pins are not to be used as the A/D converter input. When the pin is selected to be an A/D input, its original function whether it is an I/O or other pin-shared function will be removed. In addition, any internal pull-high resistor connected to the pin will be automatically removed if the pin is selected to be an A/D converter input.

• **SADC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	START	ADBZ	ADCEN	ADRFS	SACS3	SACS2	SACS1	SACS0
R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **START**: Start the A/D conversion
0→1→0: Start
This bit is used to initiate an A/D conversion process. The bit is normally low but if set high and then cleared low again, the A/D converter will initiate a conversion process.

Bit 6 **ADBZ**: A/D converter busy flag
0: No A/D conversion is in progress
1: A/D conversion is in progress
This read only flag is used to indicate whether the A/D conversion is in progress or not. When the START bit is set from low to high and then to low again, the ADBZ flag will be set to 1 to indicate that the A/D conversion is initiated. The ADBZ flag will be cleared to 0 after the A/D conversion is complete.

Bit 5 **ADCEN**: A/D converter function enable control
0: Disable
1: Enable
This bit controls the A/D internal function. This bit should be set to one to enable the A/D converter. If the bit is set low, then the A/D converter will be switched off reducing the device power consumption. When the A/D converter function is disabled, the contents of the A/D data register pair known as SADOH and SADOL will be unchanged.

Bit 4 **ADRFS**: A/D converter data format select
0: A/D converter data format → SADOH=D[11:4]; SADOL=D[3:0]
1: A/D converter data format → SADOH=D[11:8]; SADOL=D[7:0]
This bit controls the format of the 12-bit converted A/D value in the two A/D data registers. Details are provided in the A/D data register section.

Bit 3~0 **SACS3~SACS0**: A/D converter external analog channel input select
0000: AN0
0001: AN1
0010: AN2
0011: AN3
0100~1111: Non-existed channel, the input will be floating

• **SADC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	SAINS2	SAINS1	SAINS0	SAVRS1	SAVRS0	SACKS2	SACKS1	SACKS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~5 **SAINS2~SAINS0**: A/D converter input signal select
000: External input – External analog channel input
001: Internal input – Internal high performance bandgap reference voltage, V_{BGREF}
010: Internal input – Internal SD operational amplifier 0 output signal, OPA0O
011: Internal input – Internal SD operational amplifier 1 output signal, OPA1O
100: Internal input – Internal PLT operational amplifier output signal, LINEV
101~111: External input – External analog channel input

Care must be taken if the SAINS2~SAINS0 bits are set from “001” to “100” to select the internal analog signal to be converted. When the internal analog signal is selected to be converted, the external input pin must never be selected as the A/D input signal by properly setting the SACS3~SACS0 bits. Otherwise, the external channel input will be connected together with the internal analog signal. This will result in unpredictable situations such as an irreversible damage.

Bit 4~3 **SAVRS1~SAVRS0**: A/D converter reference voltage select
00: From external VREF pin
01: Internal A/D converter power, V_{DD}
1x: From external VREF pin

These bits are used to select the A/D converter reference voltage. Care must be taken if the SAVRS1~SAVRS0 bits are set to "01" to select the internal A/D converter power as the reference voltage source. When the internal A/D converter power is selected as the reference voltage, the VREF pin cannot be configured as the reference voltage input by properly configuring the corresponding pin-shared function control bits. Otherwise, the external input voltage on VREF pin will be connected to the internal A/D converter power.

Bit 2~0 **SACKS2~SACKS0**: A/D conversion clock source select
000: f_{SYS}
001: $f_{SYS}/2$
010: $f_{SYS}/4$
011: $f_{SYS}/8$
100: $f_{SYS}/16$
101: $f_{SYS}/32$
110: $f_{SYS}/64$
111: $f_{SYS}/128$

These three bits are used to select the clock source for the A/D converter.

A/D Converter Operation

The START bit in the SADC0 register is used to start the AD conversion. When the microcontroller sets this bit from low to high and then low again, an analog to digital conversion cycle will be initiated.

The ADBZ bit in the SADC0 register is used to indicate whether the analog to digital conversion process is in progress or not. This bit will be automatically set to 1 by the microcontroller after an A/D conversion is successfully initiated. When the A/D conversion is complete, the ADBZ will be cleared to 0. In addition, the corresponding A/D interrupt request flag will be set in the interrupt control register, and if the interrupts are enabled, an appropriate internal interrupt signal will be generated. This A/D internal interrupt signal will direct the program flow to the associated A/D internal interrupt address for processing. If the A/D internal interrupt is disabled, the microcontroller can poll the ADBZ bit in the SADC0 register to check whether it has been cleared as an alternative method of detecting the end of an A/D conversion cycle.

The clock source for the A/D converter, which originates from the system clock f_{SYS}, can be chosen to be either f_{SYS} or a subdivided version of f_{SYS}. The division ratio value is determined by the SACKS2~SACKS0 bits in the SADC1 register. Although the A/D clock source is determined by the system clock f_{SYS} and by bits SACKS2~SACKS0, there are some limitations on the maximum A/D clock source speed that can be selected. As the recommended range of permissible A/D clock period, t_{ADCK}, is from 0.5μs to 10μs, care must be taken for system clock frequencies. For example, as the system clock operates at a frequency of 8MHz, the SACKS2~SACKS0 bits should not be set to 000, 001 or 111. Doing so will give A/D clock periods that are less or larger than the minimum or maximum A/D clock period which may result in inaccurate A/D conversion values. Refer to the following table for examples, where values marked with an asterisk * show where, depending upon the device, special care must be taken, as the values may be less or larger than the specified A/D Clock Period range.

f_{SYS}	A/D Clock Period (t_{ADCK})							
	SACKS [2:0]=000 (f_{SYS})	SACKS [2:0]=001 ($f_{SYS}/2$)	SACKS [2:0]=010 ($f_{SYS}/4$)	SACKS [2:0]=011 ($f_{SYS}/8$)	SACKS [2:0]=100 ($f_{SYS}/16$)	SACKS [2:0]=101 ($f_{SYS}/32$)	SACKS [2:0]=110 ($f_{SYS}/64$)	SACKS [2:0]=111 ($f_{SYS}/128$)
1MHz	1μs	2μs	4μs	8μs	16μs *	32μs *	64μs *	128μs *
2MHz	500ns	1μs	2μs	4μs	8μs	16μs *	32μs *	64μs *
4MHz	250ns *	500ns	1μs	2μs	4μs	8μs	16μs *	32μs *
8MHz	125ns *	250ns *	500ns	1μs	2μs	4μs	8μs	16μs *

A/D Clock Period Examples

Controlling the power on/off function of the A/D converter circuitry is implemented using the ADCEN bit in the SADC0 register. This bit must be set high to power on the A/D converter. When the ADCEN bit is set high to power on the A/D converter internal circuitry a certain delay, as indicated in the timing diagram, must be allowed before an A/D conversion is initiated. Even if no pins are selected for use as A/D inputs, if the ADCEN bit is high, then some power will still be consumed. In power conscious applications it is therefore recommended that the ADCEN is set low to reduce power consumption when the A/D converter function is not being used.

A/D Converter Reference Voltage

The reference voltage supply to the A/D converter can be supplied from the power supply V_{DD}, or from an external reference source supplied on pin VREF. The desired selection is made using the SAVRS1 and SAVRS0 bits. When the SAVRS bit field is set to "01", the A/D converter reference voltage will come from the V_{DD}. Otherwise, if the SAVRS bit field is set to any other value except "01", the A/D converter reference voltage will come from the VREF pin. As the VREF pin is pin-shared with other functions, when the VREF pin is selected as the reference voltage supply pin, the VREF pin should be configured as the reference voltage input function by properly configuring the corresponding pin-shared function selection bits PAS15~PAS14. However, if the internal A/D converter power V_{DD} is selected as the reference voltage, the VREF pin must not be configured as the reference voltage input function to avoid the internal connection between the VREF pin to A/D converter power V_{DD}. The analog input values must not be allowed to exceed the value of the selected A/D reference voltage.

A/D Converter Input Signals

All the external A/D analog channel input pins are pin-shared with the I/O pins as well as other functions. The corresponding control bits for each A/D external input pin in the PxS0 and PxS1 registers determine whether the input pins are setup as A/D converter analog input channel or whether they have other functions. If the pin is setup to be as an A/D analog channel input, the original pin functions will be disabled. In this way, pins can be changed under program control to change their function between A/D inputs and other functions. All pull high resistors, which are setup through register programming, will be automatically disconnected if the pins are setup as A/D inputs. Note that it is not necessary to first setup the A/D pin as an input in the port control register to enable the A/D input as when the pin-shared function control bits enable an A/D input, the status of the port control register will be overridden.

There are some internal analog signals derived from the high performance bandgap reference voltage V_{BGREF}, the SD operational amplifier 0 output signal OPA0O, the SD operational amplifier 1 output signal OPA1O and PLT operational amplifier output signal, LINEV, which can be connected to the A/D converter as the analog input signal by configuring the SAINS2~SAINS0 bits. If the external channel input is selected to be converted, the SAINS2~SAINS0 bits should be set to "000" or "101~111" and the SACS3~SACS0 bits can determine which external channel is selected. If the internal analog signal is selected to be converted, the SACS3~SACS0 bits must be configured with an appropriate value to switch off the external analog channel input. Otherwise, the internal analog signal will be connected together with the external channel input. This will result in unpredictable situations.

This V_{BGREF} is the internal high performance bandgap voltage reference with driver capability. It has accurate voltage reference output when input supply voltage V_{DD} change or temperature variation. And, this bandgap will startup at a low supply voltage. Therefore, this voltage reference has high power supply rejection ratio (PSRR) for low dropout regulator (LDO) is presented.

SAINS[2:0]	SACS[3:0]	Input Signals	Description
000, 101~111	0000~0011	AN0~AN3	External pin analog input
	0100~1111	—	Non-existed channel, input is floating.
001	0100~1111	V_{BGREF}	Internal high performance Bandgap reference voltage
010	0100~1111	OPA0O	Internal SD operational amplifier 0 output signal
011	0100~1111	OPA1O	Internal SD operational amplifier 1 output signal
100	0100~1111	LINEV	Internal PLT operational amplifier output signal

A/D Converter Input Signal Selection

• **VBGRC Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	VBGREN
R/W	—	—	—	—	—	—	—	R/W
POR	—	—	—	—	—	—	—	0

Bit 7~1 Unimplemented, read as "0"

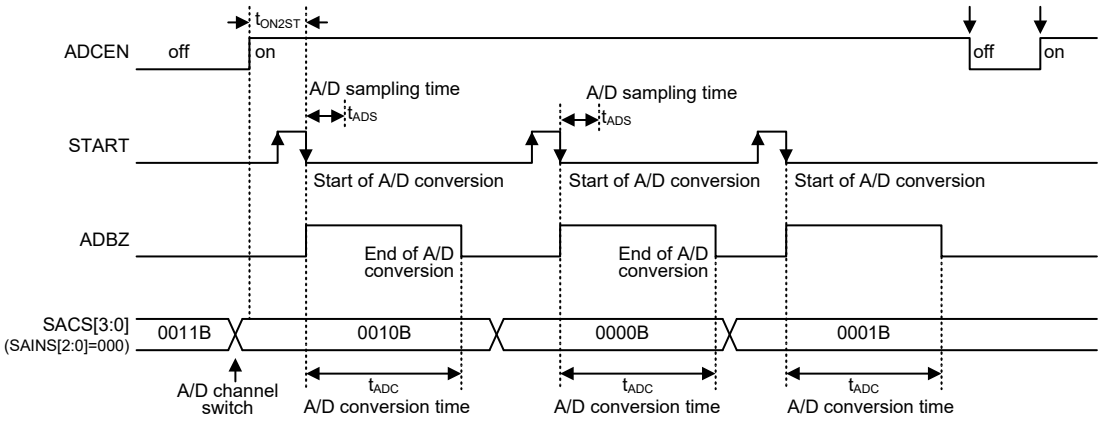
Bit 0 **VBGREN**: Bandgap enable or disable control
0: Disable
1: Enable
When the VBGREN bit is cleared to zero, the Bandgap voltage output is in a high impedance state.

Conversion Rate and Timing Diagram

A complete A/D conversion contains two parts, data sampling and data conversion. The data sampling which is defined as t_{ADS} takes 4 A/D clock cycles and the data conversion takes 12 A/D clock cycles. Therefore a total of 16 A/D clock cycles for an external input A/D conversion which is defined as t_{ADC} are necessary.

Maximum single A/D conversion rate = A/D clock period ÷ 16

The accompanying diagram shows graphically the various stages involved in an analog to digital conversion process and its associated timing. After an A/D conversion process has been initiated by the application program, the microcontroller internal hardware will begin to carry out the conversion, during which time the program can continue with other functions. The time taken for the A/D conversion is 16 t_{ADCK} clock cycles where t_{ADCK} is equal to the A/D clock period.



A/D Conversion Timing

Summary of A/D Conversion Steps

The following summarises the individual steps that should be executed in order to implement an A/D conversion process.

- Step 1
  Select the required A/D conversion clock by correctly programming bits SACKS2~SACKS0 in the SADC1 register.
- Step 2
  Enable the A/D by setting the ADCEN bit in the SADC0 register to 1.
- Step 3
  Select which signal is to be connected to the internal A/D converter by correctly configuring the SAINS2~SAINS0 bits
  Select the external channel input to be converted, go to Step 4.
  Select the internal analog signal to be converted, go to Step 5.
- Step 4
  If the A/D input signal comes from the external channel input selected by configuring the SAINS2~SAINS0 bits field, the corresponding pins should be configured as A/D input function by configuring the relevant pin-shared function control bits. The desired analog channel then should be selected by configuring the SACS3~SACS0 bits field. After this step, go to Step 6.
- Step 5
  Before the A/D input signal is selected to come from the internal analog signal by configuring the SAINS2~SAINS0 bits field, the corresponding external input pin must be switched to a non-existed channel input by properly configured the SACS3~SACS0 bits. The desired internal analog signal then can be selected by configuring the SAINS2~SAINS0 bits field. After this step, go to Step 6.
- Step 6
  Select the reference voltage source by configuring the SAVRS1~SAVRS0 bits in the SADC1 register.
- Step 7
  Select A/D converter output data format by setting the ADRFS bit in the SADC0 register.
- Step 8
  If A/D conversion interrupt is used, the interrupt control registers must be correctly configured to ensure the A/D interrupt function is active. The master interrupt control bit, EMI, and the A/D conversion interrupt control bit, ADE, must both be set high in advance.
- Step 9
  The A/D conversion procedure can now be initialized by setting the START bit from low to high and then low again.
- Step 10
  If A/D conversion is in progress, the ADBZ flag will be set high. After the A/D conversion process is complete, the ADBZ flag will go low and then the output data can be read from SADOH and SADOL registers.

Note: When checking for the end of the conversion process, if the method of polling the ADBZ bit in the SADC0 register is used, the interrupt enable step above can be omitted.

Programming Considerations

During microcontroller operations where the A/D converter is not being used, the A/D internal circuitry can be switched off to reduce power consumption, by clearing bit ADCEN to 0 in the SADC0 register. When this happens, the internal A/D converter circuits will not consume power irrespective of what analog voltage is applied to their input lines. If the A/D converter input lines are used as normal I/O pins, then care must be taken as if the input voltage is not at a valid logic level, then this may lead to some increase in power consumption.

A/D Conversion Function

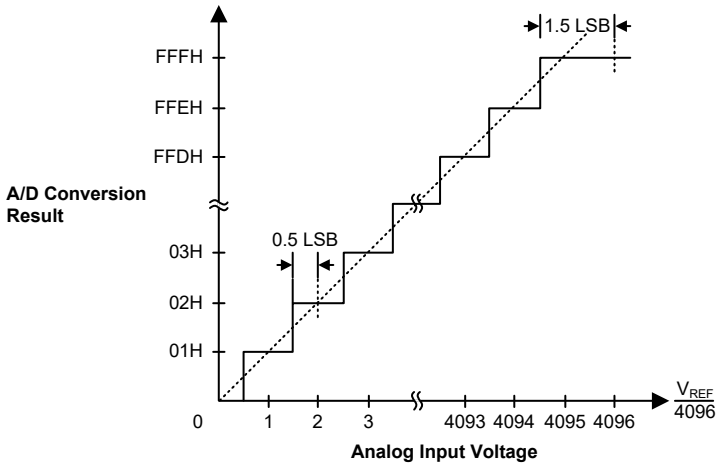
As the device contains a 12-bit A/D converter, its full-scale converted digitised value is equal to FFFH. Since the full-scale analog input value is equal to the actual A/D converter reference voltage, V_{REF}, this gives a single bit analog input value of V_{REF} divided by 4096.

$$1\ LSB = V_{REF} \div 4096$$

The A/D Converter input voltage value can be calculated using the following equation:

$$A/D\ input\ voltage = A/D\ output\ digital\ value \times V_{REF} \div 4096$$

The diagram shows the ideal transfer function between the analog input value and the digitised output value for the A/D converter. Except for the digitised zero value, the subsequent digitised values will change at a point 0.5 LSB below where they would change without the offset, and the last full scale digitised value will change at a point 1.5 LSB below the V_{REF} level. Note that here the V_{REF} voltage is the actual A/D converter reference voltage determined by the SAVRS field.



Ideal A/D Transfer Function

A/D Conversion Programming Examples

The following two programming examples illustrate how to setup and implement an A/D conversion. In the first example, the method of polling the ADBZ bit in the SADC0 register is used to detect when the conversion cycle is complete, whereas in the second example, the A/D interrupt is used to determine when the conversion is complete.

Example: using an ADBZ polling method to detect the end of conversion

```
clr  ADE                ; disable ADC interrupt
mov  a,03H
mov  SADC1,a            ; select fSYS/8 as A/D clock
set  ADCEN
mov  a,02h              ; setup PAS1 register to configure pin AN0
mov  PAS1,a
mov  a,20h
mov  SADC0,a            ; enable and connect AN0 channel to A/D converter
:
start_conversion:
clr  START              ; high pulse on start bit to initiate conversion
set  START              ; reset A/D
clr  START              ; start A/D
polling_EOC:
sz   ADBZ               ; poll the SADC0 register ADBZ bit to detect end of A/D conversion
jmp  polling_EOC        ; continue polling
mov  a,SADOL            ; read low byte conversion result value
mov  SADOL_buffer,a     ; save result to user defined register
mov  a,SADOH            ; read high byte conversion result value
mov  SADOH_buffer,a     ; save result to user defined register
:
:
jmp  start_conversion   ; start next A/D conversion
```

Example: using the interrupt method to detect the end of conversion

```
clr  ADE                ; disable ADC interrupt
mov  a,03H
mov  SADC1,a            ; select fSYS/8 as A/D clock
set  ADCEN
mov  a, 02h             ; setup PAS1 register to configure pin AN0
mov  PAS1,a
mov  a,20h
mov  SADC0,a            ; enable and connect AN0 channel to A/D converter
Start_conversion:
clr  START              ; high pulse on START bit to initiate conversion
set  START              ; reset A/D
clr  START              ; start A/D
clr  ADF                ; clear ADC interrupt request flag
set  ADE                ; enable ADC interrupt
set  EMI                ; enable global interrupt
:
:
; ADC interrupt service routine
ADC_ISR:
mov  acc_stack,a        ; save ACC to user defined memory
mov  a,STATUS
mov  status_stack,a     ; save STATUS to user defined memory
:
:
mov  a,SADOL            ; read low byte conversion result value
mov  SADOL_buffer,a     ; save result to user defined register
mov  a,SADOH            ; read high byte conversion result value
mov  SADOH_buffer,a     ; save result to user defined register
:
:
EXIT_INT_ISR:
mov  a,status_stack
mov  STATUS,a           ; restore STATUS from user defined memory
mov  a,acc_stack        ; restore ACC from user defined memory
reti
```

Sink Current Generator

The sink current source generator could provide constant current no matter what V_{ISINK} voltage is from 1.0V~3.6V. The constant current value is controlled by the ISGDATA0/ISGDATA1 register, and the sink current range is 50mA~360mA.

• ISGENC Register

Bit	7	6	5	4	3	2	1	0
Name	ISGEN	—	—	—	—	—	ISGS1	ISGS0
R/W	R/W	—	—	—	—	—	R/W	R/W
POR	0	—	—	—	—	—	0	0

Bit 7 **ISGEN**: Sink current generator enable control
0: Disable
1: Enable
When the ISGEN bit is cleared to zero to disable the sink current generator, the ISINK0&ISINK1 pin status are $V_{ISINK0/1}=V_{DD}$, $I_{SINK0/1}=0$.

Bit 6~2 Unimplemented, read as "0"

Bit 1 **ISGS1**: ISINK1 pin sink current control
0: Disable
1: Enable

Bit 0 **ISGS0**: ISINK0 pin sink current control
0: Disable
1: Enable

• ISGDATA0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	D4	D3	D2	D1	D0
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	0	0	0	0	0

Bit 7~5 Unimplemented, read as "0"

Bit 4~0 **D4~D0**: Sink current generator control for ISINK0 pin
Current value (mA)=50+10×(ISGDATA0[4:0])
Refer to "Sink Current Generator Electrical Characteristics" table for more details.

• ISGDATA1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	D4	D3	D2	D1	D0
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	0	0	0	0	0

Bit 7~5 Unimplemented, read as "0"

Bit 4~0 **D4~D0**: Sink current generator control for ISINK1 pin
Current value (mA)=50+5×(ISGDATA1[4:0])
Refer to "Sink Current Generator Electrical Characteristics" table for more details.

Universal Serial Interface Module – USIM

The device contains a Universal Serial Interface Module, which includes the four-line SPI interface, the two-line I²C interface and the two-line UART interface types, to allow an easy method of communication with external peripheral hardware. Having relatively simple communication protocols, these serial interface types allow the microcontroller to interface to external SPI, I²C or UART based hardware such as sensors, Flash or EEPROM memory, etc. The USIM interface pins are pin-shared with other I/O pins therefore the USIM interface functional pins must first be selected using the corresponding pin-shared function selection bits. As all the interface types share the same pins and registers, the choice of whether the UART, SPI or I²C type is used is made using the UART mode selection bit, named UMD, and the SPI/I²C operating mode control bits, named SIM2~SIM0, in the SIMC0 register. These pull-high resistors of the USIM pin-shared I/O are selected using pull-high control registers when the USIM function is enabled and the corresponding pins are used as USIM input pins.

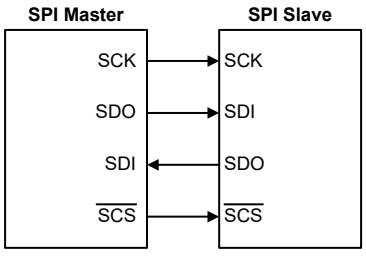
SPI Interface

The SPI interface is often used to communicate with external peripheral devices such as sensors, Flash or EEPROM memory devices etc. Originally developed by Motorola, the four line SPI interface is a synchronous serial data interface that has a relatively simple communication protocol simplifying the programming requirements when communicating with external hardware devices.

The communication is full duplex and operates as a slave/master type, where the device can be either master or slave. Although the SPI interface specification can control multiple slave devices from a single master, but the device provides only one $\overline{SCS}$ pin. If the master needs to control multiple slave devices from a single master, the master can use I/O pin to select the slave devices.

SPI Interface Operation

The SPI interface is a full duplex synchronous serial data link. It is a four line interface with pin names SDI, SDO, SCK and $\overline{SCS}$. Pins SDI and SDO are the Serial Data Input and Serial Data Output lines, the SCK pin is the Serial Clock line and $\overline{SCS}$ is the Slave Select line. As the SPI interface pins are pin-shared with normal I/O pins and with the I²C/UART function pins, the SPI interface pins must first be selected by configuring the pin-shared function selection bits and setting the correct bits in the SIMC0 and SIMC2 registers. Communication between devices connected to the SPI interface is carried out in a slave/master mode with all data transfer initiations being implemented by the master. The Master also controls the clock signal. As the device only contains a single $\overline{SCS}$ pin only one slave device can be utilized. The $\overline{SCS}$ pin is controlled by software, set CSEN bit to 1 to enable $\overline{SCS}$ pin function, set CSEN bit to 0 the $\overline{SCS}$ pin will be floating state.



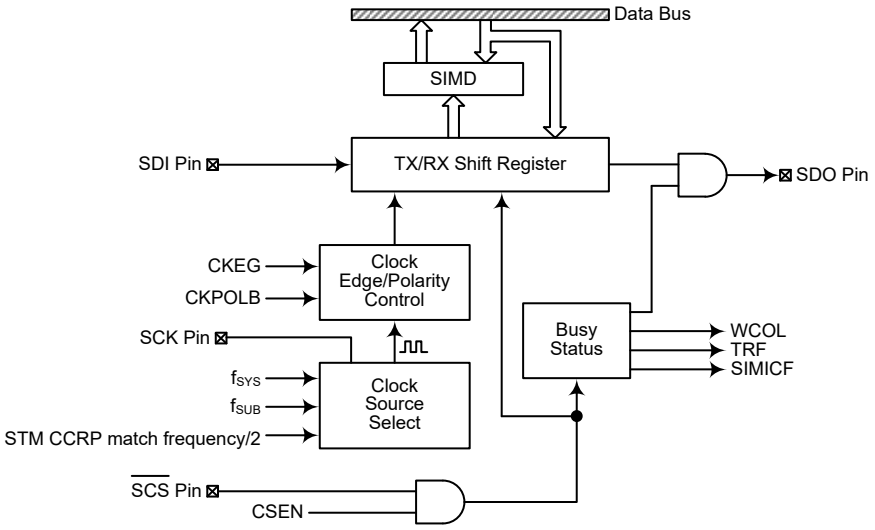
SPI Master/Slave Connection

The SPI function in the device offers the following features:

- Full duplex synchronous data transfer
- Both Master and Slave modes

- LSB first or MSB first data transmission modes
- Transmission complete flag
- Rising or falling active clock edge

The status of the SPI interface pins is determined by a number of factors such as whether the device is in the master or slave mode and upon the condition of certain control bits such as CSEN and SIMEN.



SPI Block Diagram

SPI Registers

There are three internal registers which control the overall operation of the SPI interface. These are the SIMD data register and two control registers, SIMC0 and SIMC2. Note that the SIMC2 and SIMD registers and their POR values are only available when the SPI mode is selected by properly configuring the UMD and SIM2~SIM0 bits in the SIMC0 register.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	UMD	SIMDEB1	SIMDEB0	SIMEN	SIMICF
SIMC2	D7	D6	CKPOLB	CKEG	MLS	CSEN	WCOL	TRF
SIMD	D7	D6	D5	D4	D3	D2	D1	D0

SPI Register List

SPI Data Register

The SIMD register is used to store the data being transmitted and received. The same register is used by both the SPI and I^2C functions. Before the device writes data to the SPI bus, the actual data to be transmitted must be placed in the SIMD register. After the data is received from the SPI bus, the device can read it from the SIMD register. Any transmission or reception of data from the SPI bus must be made via the SIMD register.

• **SIMD Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x": unknown

Bit 7~0 **D7~D0**: USIM SPI/I²C data register bit 7 ~ bit 0

SPI Control Registers

There are also two control registers for the SPI interface, SIMC0 and SIMC2. The SIMC0 register is used to control the enable/disable function and to set the data transmission clock frequency. The SIMC2 register is used for other control functions such as LSB/MSB selection, write collision flag etc.

• **SIMC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	UMD	SIMDEB1	SIMDEB0	SIMEN	SIMICF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	0	0	0	0	0

Bit 7~5 **SIM2~SIM0**: USIM SPI/I²C Operating Mode Control
000: SPI master mode; SPI clock is $f_{SYS}/4$
001: SPI master mode; SPI clock is $f_{SYS}/16$
010: SPI master mode; SPI clock is $f_{SYS}/64$
011: SPI master mode; SPI clock is f_{SUB}
100: SPI master mode; SPI clock is STM CCRP match frequency/2
101: SPI slave mode
110: I²C slave mode
111: Unused mode
When the UMD bit is cleared to zero, these bits setup the SPI or I²C operating mode of the USIM function. As well as selecting if the I²C or SPI function, they are used to control the SPI Master/Slave selection and the SPI Master clock frequency. The SPI clock is a function of the system clock but can also be chosen to be sourced from STM and f_{SUB}. If the SPI Slave Mode is selected then the clock will be supplied by an external Master device.

Bit 4 **UMD**: UART mode selection bit
0: SPI or I²C mode
1: UART mode
This bit is used to select the UART mode. When this bit is cleared to zero, the actual SPI or I²C mode can be selected using the SIM2~SIM0 bits. Note that the UMD bit must be cleared to zero for SPI or I²C mode.

Bit 3~2 **SIMDEB1~SIMDEB0**: I²C Debounce Time Selection
These bits are only available when the USIM is configured to operate in the I²C mode. Refer to the I²C register section.

Bit 1 **SIMEN**: USIM SPI/I²C Enable Control
0: Disable
1: Enable
The bit is the overall on/off control for the USIM SPI/I²C interface. When the SIMEN bit is cleared to zero to disable the USIM SPI/I²C interface, the SDI, SDO, SCK and $\overline{SCS}$, or SDA and SCL lines will lose their SPI or I²C function and the USIM operating current will be reduced to a minimum value. When the bit is high the USIM SPI/I²C interface is enabled. If the USIM is configured to operate as an SPI interface via the UMD and SIM2~SIM0 bits, the contents of the SPI control registers will remain at the previous settings when the SIMEN bit changes from low to high and should therefore

be first initialised by the application program. If the USIM is configured to operate as an I²C interface via the UMD and SIM2~SIM0 bits and the SIMEN bit changes from low to high, the contents of the I²C control bits such as HTX and TXAK will remain at the previous settings and should therefore be first initialised by the application program while the relevant I²C flags such as HCF, HAAS, HBB, SRW and RXAK will be set to their default states.

Bit 0 **SIMICF**: USIM SPI Incomplete Flag
0: USIM SPI incomplete condition is not occurred
1: USIM SPI incomplete condition is occurred
This bit is only available when the USIM is configured to operate in an SPI slave mode. If the SPI operates in the slave mode with the SIMEN and CSEN bits both being set to 1 but the $\overline{SCS}$ line is pulled high by the external master device before the SPI data transfer is completely finished, the SIMICF bit will be set to 1 together with the TRF bit. When this condition occurs, the corresponding interrupt will occur if the interrupt function is enabled. However, the TRF bit will not be set to 1 if the SIMICF bit is set to 1 by software application program.

• SIMC2 Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	CKPOLB	CKEG	MLS	CSEN	WCOL	TRF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **D7~D6**: Undefined bits
These bits can be read or written by the application program.

Bit 5 **CKPOLB**: SPI clock line base condition selection
0: The SCK line will be high when the clock is inactive
1: The SCK line will be low when the clock is inactive
The CKPOLB bit determines the base condition of the clock line, if the bit is high, then the SCK line will be low when the clock is inactive. When the CKPOLB bit is low, then the SCK line will be high when the clock is inactive.

Bit 4 **CKEG**: SPI SCK clock active edge type selection
CKPOLB=0
0: SCK is high base level and data capture at SCK rising edge
1: SCK is high base level and data capture at SCK falling edge
CKPOLB=1
0: SCK is low base level and data capture at SCK falling edge
1: SCK is low base level and data capture at SCK rising edge
The CKEG and CKPOLB bits are used to setup the way that the clock signal outputs and inputs data on the SPI bus. These two bits must be configured before data transfer is executed otherwise an erroneous clock edge may be generated. The CKPOLB bit determines the base condition of the clock line, if the bit is high, then the SCK line will be low when the clock is inactive. When the CKPOLB bit is low, then the SCK line will be high when the clock is inactive. The CKEG bit determines active clock edge type which depends upon the condition of CKPOLB bit.

Bit 3 **MLS**: SPI data shift order
0: LSB first
1: MSB first
This is the data shift select bit and is used to select how the data is transferred, either MSB or LSB first. Setting the bit high will select MSB first and low for LSB first.

Bit 2 **CSEN**: SPI $\overline{SCS}$ pin control
0: Disable
1: Enable
The CSEN bit is used as an enable/disable for the $\overline{SCS}$ pin. If this bit is low, then the $\overline{SCS}$ pin will be disabled and placed into a floating condition. If the bit is high the $\overline{SCS}$ pin will be enabled and used as a select pin.

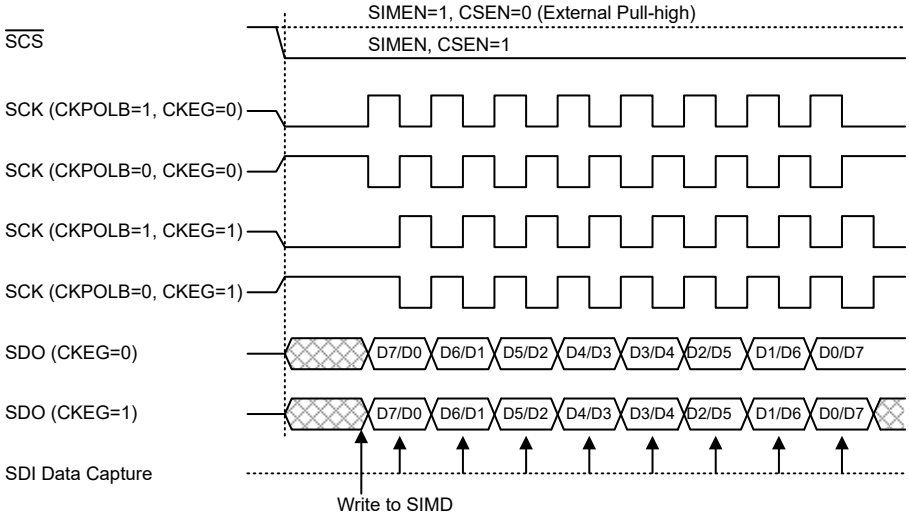
Bit 1 **WCOL**: SPI write collision flag
0: No collision
1: Collision
The WCOL flag is used to detect if a data collision has occurred. If this bit is high it means that data has been attempted to be written to the SIMD register during a data transfer operation. This writing operation will be ignored if data is being transferred. The bit can be cleared by the application program.

Bit 0 **TRF**: SPI Transmit/Receive complete flag
0: SPI data is being transferred
1: SPI data transmission is completed
The TRF bit is the Transmit/Receive Complete flag and is set "1" automatically when an SPI data transmission is completed, but must set to "0" by the application program. It can be used to generate an interrupt.

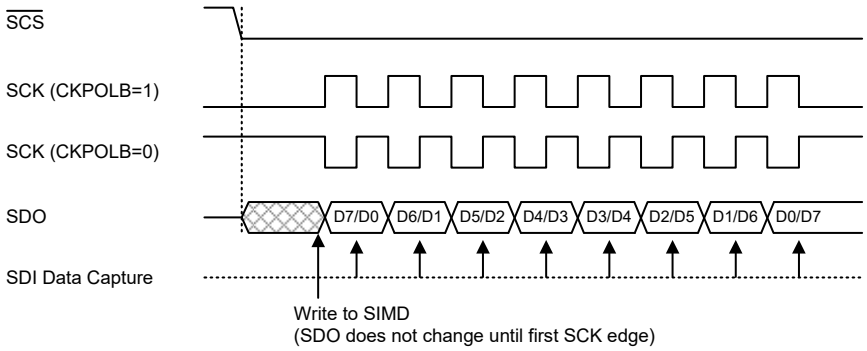
SPI Communication

After the SPI interface is enabled by setting the SIMEN bit high, then in the Master Mode, when data is written to the SIMD register, transmission/reception will begin simultaneously. When the data transfer is completed, the TRF flag will be set high automatically, but must be cleared using the application program. In the Slave Mode, when the clock signal from the master has been received, any data in the SIMD register will be transmitted and any data on the SDI pin will be shifted into the SIMD register. The master should output an $\overline{SCS}$ signal to enable the slave devices before a clock signal is provided. The slave data to be transferred should be well prepared at the appropriate moment relative to the SCK signal depending upon the configurations of the CKPOLB bit and CKEG bit. The accompanying timing diagram shows the relationship between the slave data and SCK signal for various configurations of the CKPOLB and CKEG bits.

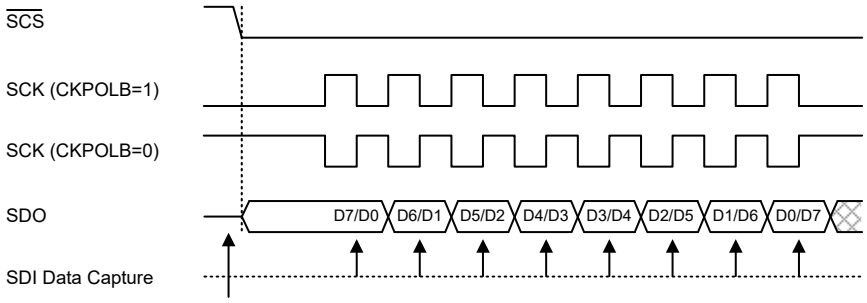
The SPI will continue to function in certain IDLE Modes if the clock source used by the SPI interface is still active.



SPI Master Mode Timing

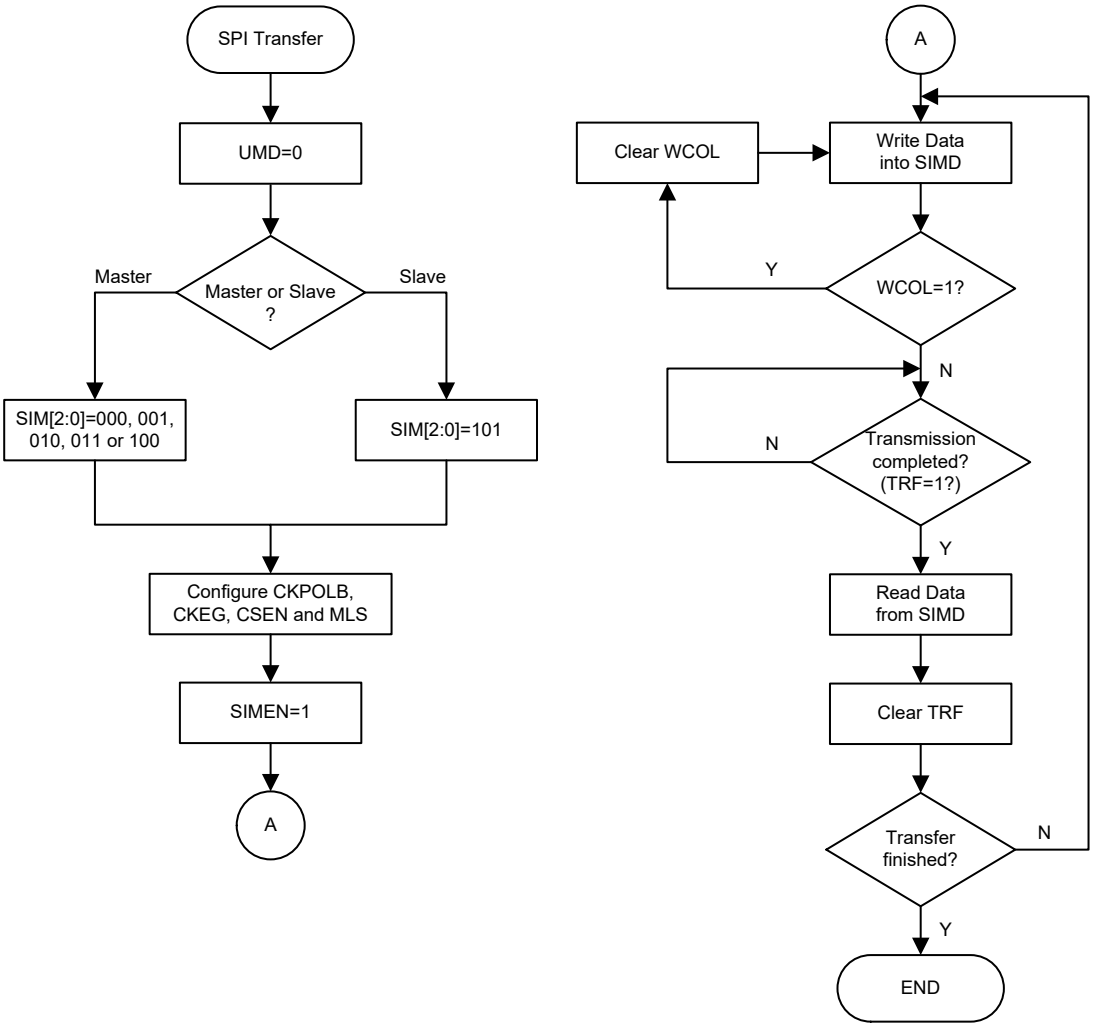


SPI Slave Mode Timing – CKEG=0



Note: For SPI slave mode, if SIMEN=1 and CSEN=0, SPI is always enabled and ignores the $\overline{SCS}$ level.

SPI Slave Mode Timing – CKEG=1



SPI Transfer Control Flowchart

SPI Bus Enable/Disable

To enable the SPI bus, set CSEN=1 and $\overline{SCS}$=0, then wait for data to be written into the SIMD (TXRX buffer) register. For the Master Mode, after data has been written to the SIMD (TXRX buffer) register, then transmission or reception will start automatically. When all the data has been transferred, the TRF bit should be set. For the Slave Mode, when clock pulses are received on SCK, data in the TXRX buffer will be shifted out or data on SDI will be shifted in.

When the SPI bus is disabled, SCK, SDI, SDO and $\overline{SCS}$ can become I/O pins or other pin-shared functions using the corresponding pin-shared control bits.

SPI Operation Steps

All communication is carried out using the 4-line interface for either Master or Slave Mode.

The CSEN bit in the SIMC2 register controls the overall function of the SPI interface. Setting this bit high will enable the SPI interface by allowing the $\overline{SCS}$ line to be active, which can then be used to control the SPI interface. If the CSEN bit is low, the SPI interface will be disabled and the $\overline{SCS}$ line will be in a floating condition and can therefore not be used for control of the SPI interface. If the CSEN bit and the SIMEN bit in the SIMC0 are set high, this will place the SDI line in a

floating condition and the SDO line high. If in Master Mode the SCK line will be either high or low depending upon the clock polarity selection bit CKPOLB in the SIMC2 register. If in Slave Mode the SCK line will be in a floating condition. If the SIMEN bit is low, then the bus will be disabled and $\overline{SCS}$, SDI, SDO and SCK will all become I/O pins or the other functions using the corresponding pin-shared control bits. In the Master Mode the Master will always generate the clock signal. The clock and data transmission will be initiated after data has been written into the SIMD register. In the Slave Mode, the clock signal will be received from an external master device for both data transmission and reception. The following sequences show the order to be followed for data transfer in both Master and Slave Mode.

Master Mode

- Step 1
  Select the SPI Master mode and clock source using the UMD and SIM2~SIM0 bits in the SIMC0 control register.
- Step 2
  Setup the CSEN bit and setup the MLS bit to choose if the data is MSB or LSB first, this setting must be the same with the Slave devices.
- Step 3
  Setup the SIMEN bit in the SIMC0 control register to enable the SPI interface.
- Step 4
  For write operations: write the data to the SIMD register, which will actually place the data into the TXRX buffer. Then use the SCK and $\overline{SCS}$ lines to output the data. After this, go to step 5.
  For read operations: the data transferred in on the SDI line will be stored in the TXRX buffer until all the data has been received at which point it will be latched into the SIMD register.
- Step 5
  Check the WCOL bit if set high then a collision error has occurred so return to step 4. If equal to zero then go to the following step.
- Step 6
  Check the TRF bit or wait for a USIM SPI serial bus interrupt.
- Step 7
  Read data from the SIMD register.
- Step 8
  Clear TRF.
- Step 9
  Go to step 4.

Slave Mode

- Step 1
  Select the SPI Slave mode using the UMD and SIM2~SIM0 bits in the SIMC0 control register
- Step 2
  Setup the CSEN bit and setup the MLS bit to choose if the data is MSB or LSB first, this setting must be the same with the Master devices.
- Step 3
  Setup the SIMEN bit in the SIMC0 control register to enable the SPI interface.
- Step 4
  For write operations: write the data to the SIMD register, which will actually place the data into the TXRX buffer. Then wait for the master clock SCK and $\overline{SCS}$ signal. After this, go to step 5.

For read operations: the data transferred in on the SDI line will be stored in the TXRX buffer until all the data has been received at which point it will be latched into the SIMD register.

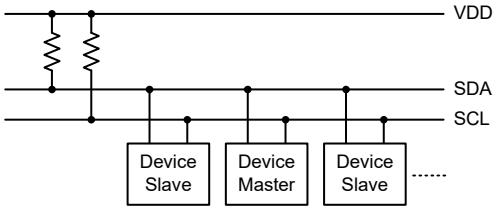
- Step 5
  Check the WCOL bit if set high then a collision error has occurred so return to step 4. If equal to zero then go to the following step.
- Step 6
  Check the TRF bit or wait for a USIM SPI serial bus interrupt.
- Step 7
  Read data from the SIMD register.
- Step 8
  Clear TRF.
- Step 9
  Go to step 4.

Error Detection

The WCOL bit in the SIMC2 register is provided to indicate errors during data transfer. The bit is set by the SPI serial Interface but must be cleared by the application program. This bit indicates that a data collision has occurred which happens if a write to the SIMD register takes place during a data transfer operation and will prevent the write operation from continuing.

I²C Interface

The I²C interface is used to communicate with external peripheral devices such as sensors, EEPROM memory etc. Originally developed by Philips, it is a two line low speed serial interface for synchronous serial data transfer. The advantage of only two lines for communication, relatively simple communication protocol and the ability to accommodate multiple devices on the same bus has made it an extremely popular interface type for many applications.

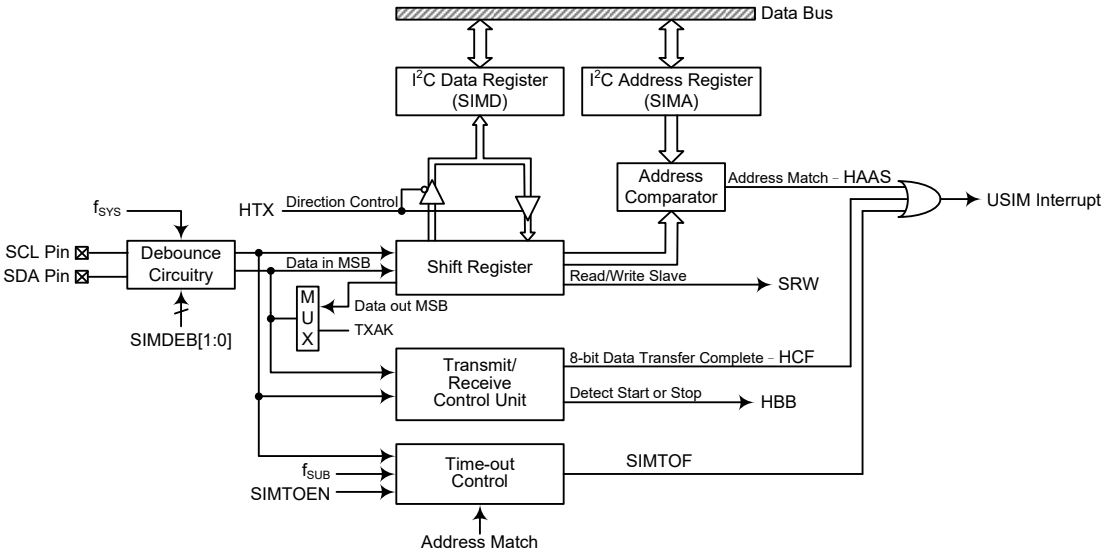


I²C Master Slave Bus Connection

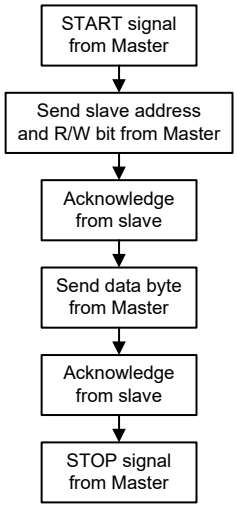
I²C Interface Operation

The I²C serial interface is a two line interface, a serial data line, SDA, and serial clock line, SCL. As many devices may be connected together on the same bus, their outputs are both open drain types. For this reason it is necessary that external pull-high resistors are connected to these outputs. Note that no chip select line exists, as each device on the I²C bus is identified by a unique address which will be transmitted and received on the I²C bus.

When two devices communicate with each other on the bidirectional I²C bus, one is known as the master device and one as the slave device. Both master and slave can transmit and receive data, however, it is the master device that has overall control of the bus. For the device, which only operates in slave mode, there are two methods of transferring data on the I²C bus, the slave transmit mode and the slave receive mode. The pull-high control function pin-shared with SCL/SDA pin is still applicable even if I²C device is activated and the related internal pull-high register could be controlled by its corresponding pull-high control register.



I²C Block Diagram



I²C Interface Operation

The SIMDEB1 and SIMDEB0 bits determine the debounce time of the I²C interface. This uses the internal clock to in effect add a debounce time to the external clock to reduce the possibility of glitches on the clock line causing erroneous operation. The debounce time, if selected, can be chosen to be either 2 or 4 system clocks. To achieve the required I²C data transfer speed, there exists a relationship between the system clock, f_{SYS}, and the I²C debounce time. For either the I²C Standard or Fast mode operation, users must take care of the selected system clock frequency and the configured debounce time to match the criterion shown in the following table.

I²C Debounce Time Selection	I²C Standard Mode (100kHz)	I²C Fast Mode (400kHz)
No Debounce	f_{SYS} > 2MHz	f_{SYS} > 5MHz
2 system clock debounce	f_{SYS} > 4MHz	f_{SYS} > 10MHz
4 system clock debounce	f_{SYS} > 8MHz	f_{SYS} > 20MHz

I²C Minimum f_{SYS} Frequency Requirements

I²C Registers

There are three control registers associated with the I²C bus, SIMC0, SIMC1 and SIMTOC, one address register SIMA and one data register, SIMD. Note that the SIMC1, SIMD, SIMA and SIMTOC registers and their POR values are only available when the I²C mode is selected by properly configuring the UMD and SIM2~SIM0 bits in the SIMC0 register.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	UMD	SIMDEB1	SIMDEB0	SIMEN	SIMICF
SIMC1	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
SIMD	D7	D6	D5	D4	D3	D2	D1	D0
SIMA	SIMA6	SIMA5	SIMA4	SIMA3	SIMA2	SIMA1	SIMA0	D0
SIMTOC	SIMTOEN	SIMTOF	SIMTOS5	SIMTOS4	SIMTOS3	SIMTOS2	SIMTOS1	SIMTOS0

I²C Register List

I²C Data Register

The SIMD register is used to store the data being transmitted and received. The same register is used by both the SPI and I²C functions. Before the device writes data to the I²C bus, the actual data to be transmitted must be placed in the SIMD register. After the data is received from the I²C bus, the device can read it from the SIMD register. Any transmission or reception of data from the I²C bus must be made via the SIMD register.

- **SIMD Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x": unknown

Bit 7~0 **D7~D0**: USIM SPI/I²C data register bit 7 ~ bit 0

I²C Address Register

The SIMA register is also used by the SPI interface but has the name SIMC2. The SIMA register is the location where the 7-bit slave address of the slave device is stored. Bits 7~1 of the SIMA register define the device slave address. Bit 0 is not defined. When a master device, which is connected to the I²C bus, sends out an address, which matches the slave address in the SIMA register, the slave device will be selected.

- **SIMA Register**

Bit	7	6	5	4	3	2	1	0
Name	SIMA6	SIMA5	SIMA4	SIMA3	SIMA2	SIMA1	SIMA0	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~1 **SIMA6~SIMA0**: I²C slave address
SIMA6~SIMA0 is the I²C slave address bit 6~bit 0.

Bit 0 **D0**: Reserved bit, can be read or written

I²C Control Registers

There are three control registers for the I²C interface, SIMC0, SIMC1 and SIMTOC. The SIMC0 register is used to control the enable/disable function and to set the data transmission clock frequency. The SIMC1 register contains the relevant flags which are used to indicate the I²C communication status. Another register, SIMTOC, is used to control the I²C time-out function and is described in the corresponding section.

- **SIMC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	UMD	SIMDEB1	SIMDEB0	SIMEN	SIMICF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	0	0	0	0	0

Bit 7~5 **SIM2~SIM0**: USIM SPI/I²C Operating Mode Control
000: SPI master mode; SPI clock is $f_{SYS}/4$
001: SPI master mode; SPI clock is $f_{SYS}/16$
010: SPI master mode; SPI clock is $f_{SYS}/64$
011: SPI master mode; SPI clock is f_{SUB}
100: SPI master mode; SPI clock is STM CCRP match frequency/2
101: SPI slave mode
110: I²C slave mode
111: Unused mode
When the UMD bit is cleared to zero, these bits setup the SPI or I²C operating mode of the USIM function. As well as selecting if the I²C or SPI function, they are used to control the SPI Master/Slave selection and the SPI Master clock frequency. The SPI clock is a function of the system clock but can also be chosen to be sourced from STM and f_{SUB}. If the SPI Slave Mode is selected then the clock will be supplied by an external Master device.

Bit 4 **UMD**: UART mode selection bit
0: SPI or I²C mode
1: UART mode
This bit is used to select the UART mode. When this bit is cleared to zero, the actual SPI or I²C mode can be selected using the SIM2~SIM0 bits. Note that the UMD bit must be set low for SPI or I²C mode.

Bit 3~2 **SIMDEB1~SIMDEB0**: I²C Debounce Time Selection
00: No debounce
01: 2 system clock debounce
1x: 4 system clock debounce
These bits are used to select the I²C debounce time when the USIM is configured as the I²C interface function by setting the UMD bit to "0" and the SIM2~SIM0 bits to "110".

Bit 1 **SIMEN**: USIM SPI/I²C Enable Control
0: Disable
1: Enable
The bit is the overall on/off control for the USIM SPI/I²C interface. When the SIMEN bit is cleared to zero to disable the USIM SPI/I²C interface, the SDI, SDO, SCK and $\overline{SCS}$, or SDA and SCL lines will lose their SPI or I²C function and the USIM operating current will be reduced to a minimum value. When the bit is high the USIM SPI/I²C interface is enabled. If the USIM is configured to operate as an SPI interface via the UMD and SIM2~SIM0 bits, the contents of the SPI control registers will remain at the previous settings when the SIMEN bit changes from low to high and should therefore be first initialised by the application program. If the USIM is configured to operate as an I²C interface via the UMD and SIM2~SIM0 bits and the SIMEN bit changes from low to high, the contents of the I²C control bits such as HTX and TXAK will remain

at the previous settings and should therefore be first initialised by the application program while the relevant I²C flags such as HCF, HAAS, HBB, SRW and RXAK will be set to their default states.

Bit 0 **SIMICF**: USIM SPI Incomplete Flag
This bit is only available when the USIM is configured to operate in an SPI slave mode. Refer to the SPI register section.

• **SIMC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
R/W	R	R	R	R/W	R/W	R	R/W	R
POR	1	0	0	0	0	0	0	1

Bit 7 **HCF**: I²C Bus data transfer completion flag
0: Data is being transferred
1: Completion of an 8-bit data transfer
The HCF flag is the data transfer flag. This flag will be zero when data is being transferred. Upon completion of an 8-bit data transfer the flag will go high and an interrupt will be generated.

Bit 6 **HAAS**: I²C Bus address match flag
0: Not address match
1: Address match
The HAAS flag is the address match flag. This flag is used to determine if the slave device address is the same as the master transmit address. If the addresses match then this bit will be high, if there is no match then the flag will be low.

Bit 5 **HBB**: I²C Bus busy flag
0: I²C Bus is not busy
1: I²C Bus is busy
The HBB flag is the I²C busy flag. This flag will be "1" when the I²C bus is busy which will occur when a START signal is detected. The flag will be set to "0" when the bus is free which will occur when a STOP signal is detected.

Bit 4 **HTX**: I²C slave device is transmitter or receiver selection
0: Slave device is the receiver
1: Slave device is the transmitter

Bit 3 **TXAK**: I²C Bus transmit acknowledge flag
0: Slave send acknowledge flag
1: Slave do not send acknowledge flag
The TXAK bit is the transmit acknowledge flag. After the slave device receipt of 8 bits of data, this bit will be transmitted to the bus on the 9th clock from the slave device. The slave device must always set TXAK bit to "0" before further data is received.

Bit 2 **SRW**: I²C Slave Read/Write flag
0: Slave device should be in receive mode
1: Slave device should be in transmit mode
The SRW flag is the I²C Slave Read/Write flag. This flag determines whether the master device wishes to transmit or receive data from the I²C bus. When the transmitted address and slave address is match, that is when the HAAS flag is set high, the slave device will check the SRW flag to determine whether it should be in transmit mode or receive mode. If the SRW flag is high, the master is requesting to read data from the bus, so the slave device should be in transmit mode. When the SRW flag is zero, the master will write data to the bus, therefore the slave device should be in receive mode to read this data.

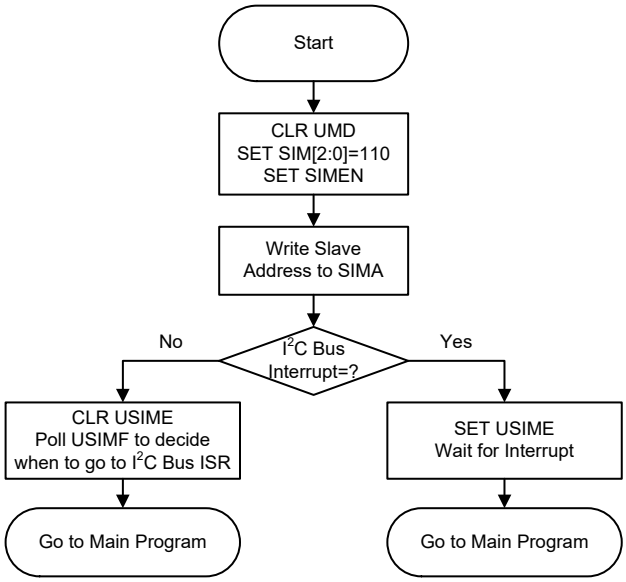
Bit 1 **IAMWU**: I²C Address Match Wake-up control
0: Disable
1: Enable
This bit should be set to 1 to enable the I²C address match wake up from the SLEEP or IDLE Mode. If the IAMWU bit has been set before entering either the SLEEP or IDLE mode to enable the I²C address match wake up, then this bit must be cleared by the application program after wake-up to ensure correction device operation.

Bit 0 **RXAK**: I²C Bus Receive acknowledge flag
0: Slave receive acknowledge flag
1: Slave does not receive acknowledge flag
The RXAK flag is the receiver acknowledge flag. When the RXAK flag is "0", it means that a acknowledge signal has been received at the 9th clock, after 8 bits of data have been transmitted. When the slave device in the transmit mode, the slave device checks the RXAK flag to determine if the master receiver wishes to receive the next byte. The slave transmitter will therefore continue sending out data until the RXAK flag is "1". When this occurs, the slave transmitter will release the SDA line to allow the master to send a STOP signal to release the I²C Bus.

I²C Bus Communication

Communication on the I²C bus requires four separate steps, a START signal, a slave device address transmission, a data transmission and finally a STOP signal. When a START signal is placed on the I²C bus, all devices on the bus will receive this signal and be notified of the imminent arrival of data on the bus. The first seven bits of the data will be the slave address with the first bit being the MSB. If the address of the slave device matches that of the transmitted address, the HAAS bit in the SIMC1 register will be set and an USIM interrupt will be generated. After entering the interrupt service routine, the slave device must first check the condition of the HAAS and SIMTOF bits to determine whether the interrupt source originates from an address match or from the completion of an 8-bit data transfer completion or from the I²C bus time-out occurrence. During a data transfer, note that after the 7-bit slave address has been transmitted, the following bit, which is the 8th bit, is the read/write bit whose value will be placed in the SRW bit. This bit will be checked by the slave device to determine whether to go into transmit or receive mode. Before any transfer of data to or from the I²C bus, the microcontroller must initialise the bus, the following are steps to achieve this:

- Step 1
  Set the UMD, SIM2~SIM0 and SIMEN bits in the SIMC0 register to "0", "110" and "1" respectively to enable the I²C bus.
- Step 2
  Write the slave address of the device to the I²C bus address register SIMA.
- Step 3
  Set the USIME interrupt enable bit of the interrupt control register to enable the USIM interrupt.



I²C Bus Initialisation Flow Chart

I²C Bus Start Signal

The START signal can only be generated by the master device connected to the I²C bus and not by the slave device. This START signal will be detected by all devices connected to the I²C bus. When detected, this indicates that the I²C bus is busy and therefore the HBB bit will be set. A START condition occurs when a high to low transition on the SDA line takes place when the SCL line remains high.

I²C Slave Address

The transmission of a START signal by the master will be detected by all devices on the I²C bus. To determine which slave device the master wishes to communicate with, the address of the slave device will be sent out immediately following the START signal. All slave devices, after receiving this 7-bit address data, will compare it with their own 7-bit slave address. If the address sent out by the master matches the internal address of the microcontroller slave device, then an internal USIM I²C bus interrupt signal will be generated. The next bit following the address, which is the 8th bit, defines the read/write status and will be saved to the SRW bit of the SIMC1 register. The slave device will then transmit an acknowledge bit, which is a low level, as the 9th bit. The slave device will also set the status flag HAAS when the addresses match.

As an USIM I²C bus interrupt can come from three sources, when the program enters the interrupt subroutine, the HAAS and SIMTOF bits should be examined to see whether the interrupt source has come from a matching slave address or from the completion of a data byte transfer or from the I²C bus time-out occurrence. When a slave address is matched, the device must be placed in either the transmit mode and then write data to the SIMD register, or in the receive mode where it must implement a dummy read from the SIMD register to release the SCL line.

I²C Bus Read/Write Signal

The SRW bit in the SIMC1 register defines whether the master device wishes to read data from the I²C bus or write data to the I²C bus. The slave device should examine this bit to determine if it is to be a transmitter or a receiver. If the SRW flag is “1” then this indicates that the master device wishes to read data from the I²C bus, therefore the slave device must be setup to send data to the I²C bus as a transmitter. If the SRW flag is “0” then this indicates that the master wishes to send data to the I²C bus, therefore the slave device must be setup to read data from the I²C bus as a receiver.

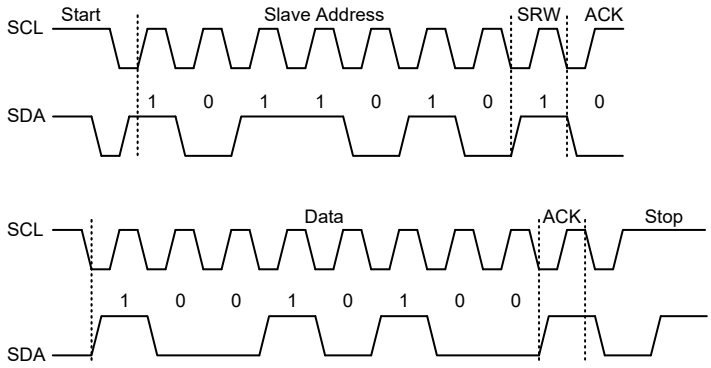
I²C Bus Slave Address Acknowledge Signal

After the master has transmitted a calling address, any slave device on the I²C bus, whose own internal address matches the calling address, must generate an acknowledge signal. The acknowledge signal will inform the master that a slave device has accepted its calling address. If no acknowledge signal is received by the master then a STOP signal must be transmitted by the master to end the communication. When the HAAS flag is high, the addresses have matched and the slave device must check the SRW flag to determine if it is to be a transmitter or a receiver. If the SRW flag is high, the slave device should be setup to be a transmitter so the HTX bit in the SIMC1 register should be set to "1". If the SRW flag is low, then the microcontroller slave device should be setup as a receiver and the HTX bit in the SIMC1 register should be set to "0".

I²C Bus Data and Acknowledge Signal

The transmitted data is 8-bit wide and is transmitted after the slave device has acknowledged receipt of its slave address. The order of serial bit transmission is the MSB first and the LSB last. After receipt of 8 bits of data, the receiver must transmit an acknowledge signal, level "0", before it can receive the next data byte. If the slave transmitter does not receive an acknowledge bit signal from the master receiver, then the slave transmitter will release the SDA line to allow the master to send a STOP signal to release the I²C Bus. The corresponding data will be stored in the SIMD register. If setup as a transmitter, the slave device must first write the data to be transmitted into the SIMD register. If setup as a receiver, the slave device must read the transmitted data from the SIMD register.

When the slave receiver receives the data byte, it must generate an acknowledge bit, known as TXAK, on the 9th clock. The slave device, which is setup as a transmitter will check the RXAK bit in the SIMC1 register to determine if it is to send another data byte, if not then it will release the SDA line and await the receipt of a STOP signal from the master.

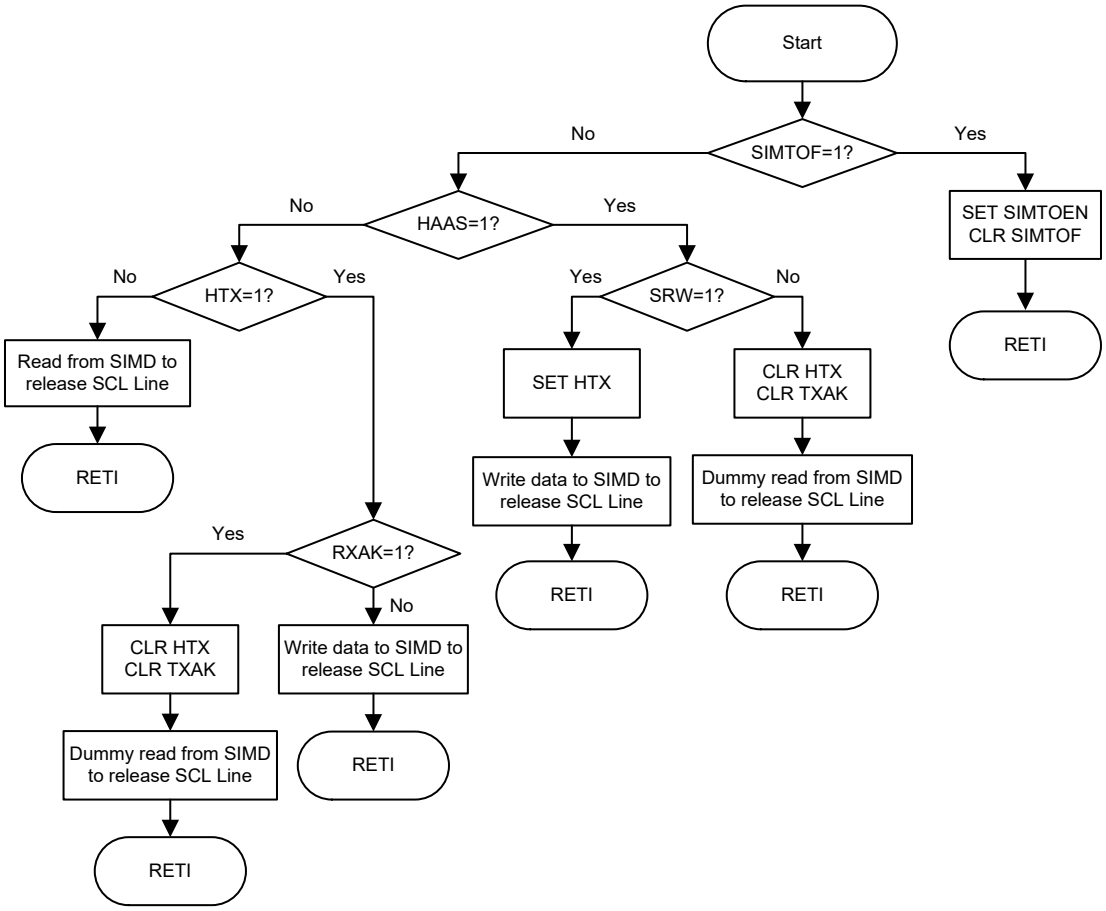


S=Start (1 bit)
SA=Slave Address (7 bits)
SR=SRW bit (1 bit)
M=Slave device send acknowledge bit (1 bit)
D=Data (8 bits)
A=ACK (RXAK bit for transmitter, TXAK bit for receiver, 1 bit)
P=Stop (1 bit)

S	SA	SR	M	D	A	D	A	·····	S	SA	SR	M	D	A	D	A	·····	P

I²C Communication Timing Diagram

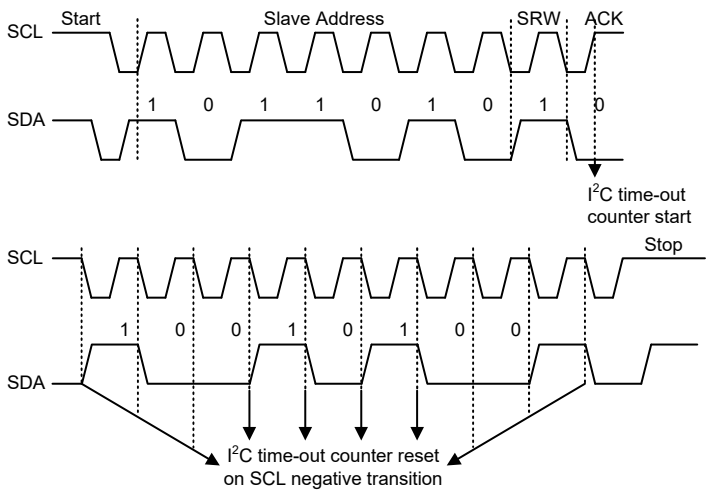
Note: When a slave address is matched, the device must be placed in either the transmit mode and then write data to the SIMD register, or in the receive mode where it must implement a dummy read from the SIMD register to release the SCL line.



I²C Bus ISR Flow Chart

I²C Time-out Control

In order to reduce the problem of I²C lockup due to reception of erroneous clock sources, a time-out function is provided. If the clock source to the I²C is not received for a while, then the I²C circuitry and registers will be reset after a certain time-out period. The time-out counter starts counting on an I²C bus "START" & "address match" condition, and is cleared by an SCL falling edge. Before the next SCL falling edge arrives, if the time elapsed is greater than the time-out setup by the SIMTOC register, then a time-out condition will occur. The time-out function will stop when an I²C "STOP" condition occurs.



I²C Time-out

When an I²C time-out counter overflow occurs, the counter will stop and the SIMTOEN bit will be cleared to zero and the SIMTOF bit will be set high to indicate that a time-out condition has occurred. The time-out condition will also generate an interrupt which uses the USIM interrupt vector. When an I²C time-out occurs, the I²C internal circuitry will be reset and the registers will be reset into the following condition:

Registers	After I²C Time-out
SIMD, SIMA, SIMC0	No change
SIMC1	Reset to POR condition

I²C Registers after Time-out

The SIMTOF flag can be cleared by the application program. There are 64 time-out periods which can be selected using SIMTOS bit field in the SIMTOC register. The time-out time is given by the formula: $((1\sim64)\times32)/f_{SUB}$. This gives a time-out period which ranges from about 1ms to 64ms.

- **SIMTOC Register**

Bit	7	6	5	4	3	2	1	0
Name	SIMTOEN	SIMTOF	SIMTOS5	SIMTOS4	SIMTOS3	SIMTOS2	SIMTOS1	SIMTOS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **SIMTOEN**: USIM I²C Time-out control
0: Disable
1: Enable

Bit 6 **SIMTOF**: USIM I²C Time-out flag
0: No time-out occurred
1: Time-out occurred
This bit is set high when time-out occurs and can only be cleared by application program.

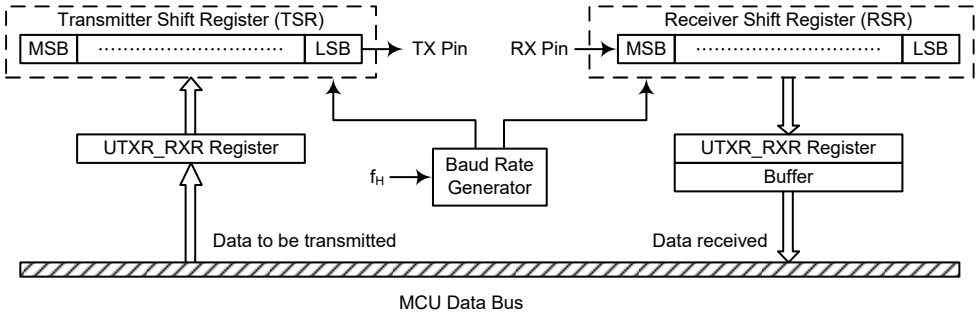
Bit 5~0 **SIMTOS5~SIMTOS0**: USIM I²C Time-out period selection
I²C time-out clock source is $f_{SUB}/32$.
I²C time-out time is equal to $(SIMTOS[5:0]+1)\times(32/f_{SUB})$.

UART Interface

The device contains an integrated full-duplex asynchronous serial communications UART interface that enables communication with external devices that contain a serial interface. The UART function has many features and can transmit and receive data serially by transferring a frame of data with eight or nine data bits per transmission as well as being able to detect errors when the data is overwritten or incorrectly framed. The UART function shares the same internal interrupt vector with the SPI and I^2C interfaces which can be used to indicate when a reception occurs or when a transmission terminates.

The integrated UART function contains the following features:

- Full-duplex, asynchronous communication
- 8 or 9 bits character length
- Even, odd or no parity options
- One or two stop bits
- Baud rate generator with 8-bit prescaler
- Parity, framing, noise and overrun error detection
- Support for interrupt on address detect (last character bit=1)
- Separately enabled transmitter and receiver
- 2-byte Deep FIFO Receive Data Buffer
- RX pin wake-up function
- Transmit and receive interrupts
- Interrupts can be initialized by the following conditions:
  - Transmitter Empty
  - Transmitter Idle
  - Receiver Full
  - Receiver Overrun
  - Address Mode Detect



UART Data Transfer Block Diagram

UART External Pins

To communicate with an external serial interface, the internal UART has two external pins known as TX and RX. The TX and RX pins are the UART transmitter and receiver pins respectively. The TX and RX pin function should first be selected by the corresponding pin-shared function selection register before the UART function is used. Along with the UMD bit, the UREN bit, the UTXEN and URXEN bits, if set, will setup these pins to their respective TX output and RX input conditions and disable any pull-high resistor option which may exist on the TX and RX pins. When the TX or RX

pin function is disabled by clearing the UMD, UREN, UTXEN or URXEN bit, the TX or RX pin will be set to a floating state. At this time whether the internal pull-high resistor is connected to the TX or RX pin or not is determined by the corresponding I/O pull-high function control bit.

UART Data Transfer Scheme

The above block diagram shows the overall data transfer structure arrangement for the UART. The actual data to be transmitted from the MCU is first transferred to the UTXR_RXR register by the application program. The data will then be transferred to the Transmit Shift Register from where it will be shifted out, LSB first, onto the TX pin at a rate controlled by the Baud Rate Generator. Only the UTXR_RXR register is mapped onto the MCU Data Memory, the Transmit Shift Register is not mapped and is therefore inaccessible to the application program.

Data to be received by the UART is accepted on the external RX pin, from where it is shifted in, LSB first, to the Receiver Shift Register at a rate controlled by the Baud Rate Generator. When the shift register is full, the data will then be transferred from the shift register to the internal UTXR_RXR register, where it is buffered and can be manipulated by the application program. Only the UTXR_RXR register is mapped onto the MCU Data Memory, the Receiver Shift Register is not mapped and is therefore inaccessible to the application program.

It should be noted that the actual register for data transmission and reception only exists as a single shared register in the Data Memory. This shared register known as the UTXR_RXR register is used for both data transmission and data reception.

UART Status and Control Registers

There are six control registers associated with the UART function. The UMD bit in the SIMC0 register can be used to select the UART mode. The UUSR, UUCR1 and UUCR2 registers control the overall function of the UART, while the UBRG register controls the Baud rate. The actual data to be transmitted and received on the serial interface is managed through the UTXR_RXR data register. Note that UART related registers and their POR values are only available when the UART mode is selected by setting the UMD bit in the SIMC0 register to "1".

Register Name	Bit							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	UMD	SIMDEB1	SIMDEB0	SIMEN	SIMICF
UUSR	UPERR	UNF	UFERR	UOERR	URIDLE	URXIF	UTIDLE	UTXIF
UUCR1	UREN	UBNO	UPREN	UPRT	USTOPS	UTXBRK	URX8	UTX8
UUCR2	UTXEN	URXEN	UBRGH	UADDEN	UWAKE	URIE	UTIIE	UTEIE
UTXR_RXR	UTXRX7	UTXRX6	UTXRX5	UTXRX4	UTXRX3	UTXRX2	UTXRX1	UTXRX0
UBRG	UBRG7	UBRG6	UBRG5	UBRG4	UBRG3	UBRG2	UBRG1	UBRG0

UART Register List

- **SIMC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	UMD	SIMDEB1	SIMDEB0	SIMEN	SIMICF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	0	0	0	0	0

Bit 7~5 **SIM2~SIM0**: USIM SPI/I^2C Operating Mode Control

When the UMD bit is cleared to zero, these bits setup the SPI or I^2C operating mode of the USIM function. Refer to the SPI or I^2C register section for more details.

Bit 4 **UMD**: UART mode selection bit
0: SPI or I²C mode
1: UART mode
This bit is used to select the UART mode. When this bit is cleared to zero, the actual SPI or I²C mode can be selected using the SIM2~SIM0 bits. Note that the UMD bit must be set low for SPI or I²C mode.

Bit 3~2 **SIMDEB1~SIMDEB0**: I²C Debounce Time Selection
Refer to the I²C register section.

Bit 1 **SIMEN**: USIM SPI/I²C Enable Control
This bit is only available when the USIM is configured to operate in an SPI or I²C mode with the UMD bit set low. Refer to the SPI or I²C register section for more details.

Bit 0 **SIMICF**: USIM SPI Incomplete Flag
Refer to the SPI register section.

• UUSR Register

The UUSR register is the status register for the UART, which can be read by the program to determine the present status of the UART. All flags within the UUSR register are read only. Further explanation on each of the flags is given below:

Bit	7	6	5	4	3	2	1	0
Name	UPERR	UNF	UFERR	UOERR	URIDLE	URXIF	UTIDLE	UTXIF
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	1	0	1	1

Bit 7 **UPERR**: Parity error flag
0: No parity error is detected
1: Parity error is detected
The UPERR flag is the parity error flag. When this read only flag is "0", it indicates a parity error has not been detected. When the flag is "1", it indicates that the parity of the received word is incorrect. This error flag is applicable only if Parity mode (odd or even) is selected. The flag can also be cleared by a software sequence which involves a read to the status register UUSR followed by an access to the UTXR_RXR data register.

Bit 6 **UNF**: Noise flag
0: No noise is detected
1: Noise is detected
The UNF flag is the noise flag. When this read only flag is "0", it indicates no noise condition. When the flag is "1", it indicates that the UART has detected noise on the receiver input. The UNF flag is set during the same cycle as the URXIF flag but will not be set in the case of as overrun. The UNF flag can be cleared by a software sequence which will involve a read to the status register UUSR followed by an access to the UTXR_RXR data register.

Bit 5 **UFERR**: Framing error flag
0: No framing error is detected
1: Framing error is detected
The UFERR flag is the framing error flag. When this read only flag is "0", it indicates that there is no framing error. When the flag is "1", it indicates that a framing error has been detected for the current character. The flag can also be cleared by a software sequence which will involve a read to the status register UUSR followed by an access to the UTXR_RXR data register.

Bit 4 **UOERR**: Overrun error flag
0: No overrun error is detected
1: Overrun error is detected
The UOERR flag is the overrun error flag which indicates when the receiver buffer has overflowed. When this read only flag is "0", it indicates that there is no overrun error. When the flag is "1", it indicates that an overrun error occurs which will inhibit further transfers to the UTXR_RXR receive data register. The flag is cleared by a software sequence, which is a read to the status register UUSR followed by an access to the UTXR_RXR data register.

Bit 3 **URIDLE**: Receiver status
0: Data reception is in progress (Data being received)
1: No data reception is in progress (Receiver is idle)
The URIDLE flag is the receiver status flag. When this read only flag is "0", it indicates that the receiver is between the initial detection of the start bit and the completion of the stop bit. When the flag is "1", it indicates that the receiver is idle. Between the completion of the stop bit and the detection of the next start bit, the URIDLE bit is "1" indicating that the UART receiver is idle and the RX pin stays in logic high condition.

Bit 2 **URXIF**: Receive UTXR_RXR data register status
0: UTXR_RXR data register is empty
1: UTXR_RXR data register has available data
The URXIF flag is the receive data register status flag. When this read only flag is "0", it indicates that the UTXR_RXR read data register is empty. When the flag is "1", it indicates that the UTXR_RXR read data register contains new data. When the contents of the shift register are transferred to the UTXR_RXR register, an interrupt is generated if URIE=1 in the UUCR2 register. If one or more errors are detected in the received word, the appropriate receive-related flags UNF, UFERR, and/or UPERR are set within the same clock cycle. The URXIF flag will eventually be cleared when the UUSR register is read with URXIF set, followed by a read from the UTXR_RXR register, and if the UTXR_RXR register has no more new data available.

Bit 1 **UTIDLE**: Transmission idle
0: Data transmission is in progress (Data being transmitted)
1: No data transmission is in progress (Transmitter is idle)
The UTIDLE flag is known as the transmission complete flag. When this read only flag is "0", it indicates that a transmission is in progress. This flag will be set high when the UTXIF flag is "1" and when there is no transmit data or break character being transmitted. When UTIDLE is equal to "1", the TX pin becomes idle with the pin state in logic high condition. The UTIDLE flag is cleared by reading the UUSR register with UTIDLE set and then writing to the UTXR_RXR register. The flag is not generated when a data character or a break is queued and ready to be sent.

Bit 0 **UTXIF**: Transmit UTXR_RXR data register status
0: Character is not transferred to the transmit shift register
1: Character has transferred to the transmit shift register (UTXR_RXR data register is empty)
The UTXIF flag is the transmit data register empty flag. When this read only flag is "0", it indicates that the character is not transferred to the transmitter shift register. When the flag is "1", it indicates that the transmitter shift register has received a character from the UTXR_RXR data register. The UTXIF flag is cleared by reading the UART status register (UUSR) with UTXIF set and then writing to the UTXR_RXR data register. Note that when the UTXEN bit is set, the UTXIF flag bit will also be set since the transmit data register is not yet full.

• UUCR1 Register

The UUCR1 register together with the UUCR2 register are the two UART control registers that are used to set the various options for the UART function, such as overall on/off control, parity control, data transfer bit length etc. Further explanation on each of the bits is given below:

Bit	7	6	5	4	3	2	1	0
Name	UREN	UBNO	UPREN	UPRT	USTOPS	UTXBRK	URX8	UTX8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	W
POR	0	0	0	0	0	0	x	0

"x": unknown

Bit 7 **UREN**: UART function enable control
0: Disable UART TX and RX pins are in a floating state
1: Enable UART TX and RX pins function as UART pins

The UREN bit is the UART enable bit. When this bit is equal to "0", the UART will be disabled and the RX pin as well as the TX pin will be set in a floating state. When the bit is equal to "1", the UART will be enabled if the UMD bit is set and the TX and RX pins will function as defined by the UTXEN and URXEN enable control bits.

When the UART is disabled, it will empty the buffer so any character remaining in the buffer will be discarded. In addition, the value of the baud rate counter will be reset. If the UART is disabled, all error and status flags will be reset. Also the UTXEN, URXEN, UTXBRK, URXIF, UOERR, UFERR, UPERR and UNF bits will be cleared, while the UTIDLE, UTXIF and URIDLE bits will be set. Other control bits in UUCR1, UUCR2 and UBRG registers will remain unaffected. If the UART is active and the UREN bit is cleared, all pending transmissions and receptions will be terminated and the module will be reset as defined above. When the UART is re-enabled, it will restart in the same configuration.

Bit 6 **UBNO**: Number of data transfer bits selection
0: 8-bit data transfer
1: 9-bit data transfer

This bit is used to select the data length format, which can have a choice of either 8-bit or 9-bit format. When this bit is equal to "1", a 9-bit data length format will be selected. If the bit is equal to "0", then an 8-bit data length format will be selected. If 9-bit data length format is selected, then bits URX8 and UTX8 will be used to store the 9th bit of the received and transmitted data respectively.

Bit 5 **UPREN**: Parity function enable control
0: Parity function is disabled
1: Parity function is enabled

This is the parity enable bit. When this bit is equal to "1", the parity function will be enabled. If the bit is equal to "0", then the parity function will be disabled.

Bit 4 **UPRT**: Parity type selection bit
0: Even parity for parity generator
1: Odd parity for parity generator

This bit is the parity type selection bit. When this bit is equal to "1", odd parity type will be selected. If the bit is equal to "0", then even parity type will be selected.

Bit 3 **USTOPS**: Number of Stop bits selection
0: One stop bit format is used
1: Two stop bits format is used

This bit determines if one or two stop bits are to be used. When this bit is equal to "1", two stop bits are used. If this bit is equal to "0", then only one stop bit is used.

Bit 2 **UTXBRK**: Transmit break character
0: No break character is transmitted
1: Break characters transmit

The UTXBRK bit is the Transmit Break Character bit. When this bit is "0", there are no break characters and the TX pin operates normally. When the bit is "1", there are

transmit break characters and the transmitter will send logic zeros. When this bit is equal to "1", after the buffered data has been transmitted, the transmitter output is held low for a minimum of a 13-bit length and until the UTXBRK bit is reset.

Bit 1 **URX8**: Receive data bit 8 for 9-bit data transfer format (read only)
This bit is only used if 9-bit data transfers are used, in which case this bit location will store the 9th bit of the received data known as URX8. The UBNO bit is used to determine whether data transfers are in 8-bit or 9-bit format.

Bit 0 **UTX8**: Transmit data bit 8 for 9-bit data transfer format (write only)
This bit is only used if 9-bit data transfers are used, in which case this bit location will store the 9th bit of the transmitted data known as UTX8. The UBNO bit is used to determine whether data transfers are in 8-bit or 9-bit format.

- **UUCR2 Register**

The UUCR2 register is the second of the two UART control registers and serves several purposes. One of its main functions is to control the basic enable/disable operation of the UART Transmitter and Receiver as well as enabling the various USIM UART mode interrupt sources. The register also serves to control the baud rate speed, receiver wake-up enable and the address detect enable. Further explanation on each of the bits is given below:

Bit	7	6	5	4	3	2	1	0
Name	UTXEN	URXEN	UBRGH	UADDEN	UWAKE	URIE	UTIIE	UTEIE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **UTXEN**: UART Transmitter enabled control
0: UART transmitter is disabled
1: UART transmitter is enabled
The bit named UTXEN is the Transmitter Enable Bit. When this bit is equal to "0", the transmitter will be disabled with any pending data transmissions being aborted. In addition the buffers will be reset. In this situation the TX pin will be set in a floating state.
If the UTXEN bit is equal to "1" and the UMD and UREN bit are also equal to "1", the transmitter will be enabled and the TX pin will be controlled by the UART. Clearing the UTXEN bit during a transmission will cause the data transmission to be aborted and will reset the transmitter. If this situation occurs, the TX pin will be set in a floating state.

Bit 6 **URXEN**: UART Receiver enabled control
0: UART receiver is disabled
1: UART receiver is enabled
The bit named URXEN is the Receiver Enable Bit. When this bit is equal to "0", the receiver will be disabled with any pending data receptions being aborted. In addition the receive buffers will be reset. In this situation the RX pin will be set in a floating state. If the URXEN bit is equal to "1" and the UMD and UREN bit are also equal to "1", the receiver will be enabled and the RX pin will be controlled by the UART. Clearing the URXEN bit during a reception will cause the data reception to be aborted and will reset the receiver. If this situation occurs, the RX pin will be set in a floating state.

Bit 5 **UBRGH**: Baud Rate speed selection
0: Low speed baud rate
1: High speed baud rate
The bit named UBRGH selects the high or low speed mode of the Baud Rate Generator. This bit, together with the value placed in the baud rate register UBRG, controls the Baud Rate of the UART. If this bit is equal to "1", the high speed mode is selected. If the bit is equal to "0", the low speed mode is selected.

Bit 4 **UADDEN**: Address detect function enable control
0: Address detect function is disabled
1: Address detect function is enabled

The bit named UADDEN is the address detect function enable control bit. When this bit is equal to "1", the address detect function is enabled. When it occurs, if the 8th bit, which corresponds to URX7 if UBNO=0 or the 9th bit, which corresponds to URX8 if UBNO=1, has a value of "1", then the received word will be identified as an address, rather than data. If the corresponding interrupt is enabled, an interrupt request will be generated each time the received word has the address bit set, which is the 8th or 9th bit depending on the value of UBNO. If the address bit known as the 8th or 9th bit of the received word is "0" with the address detect function being enabled, an interrupt will not be generated and the received data will be discarded.

Bit 3 **UWAKE**: RX pin wake-up UART function enable control
0: RX pin wake-up UART function is disabled
1: RX pin wake-up UART function is enabled

This bit is used to control the wake-up UART function when a falling edge on the RX pin occurs. Note that this bit is only available when the UART clock (f_H) is switched off. There will be no RX pin wake-up UART function if the UART clock (f_H) exists. If the UWAKE bit is set to 1 as the UART clock (f_H) is switched off, a UART wake-up request will be initiated when a falling edge on the RX pin occurs. When this request happens and the corresponding interrupt is enabled, an RX pin wake-up UART interrupt will be generated to inform the MCU to wake up the UART function by switching on the UART clock (f_H) via the application program. Otherwise, the UART function cannot resume even if there is a falling edge on the RX pin when the UWAKE bit is cleared to 0.

Bit 2 **URIE**: Receiver interrupt enable control
0: Receiver related interrupt is disabled
1: Receiver related interrupt is enabled

This bit enables or disables the receiver interrupt. If this bit is equal to "1" and when the receiver overrun flag UOERR or receive data available flag URXIF is set, the USIM interrupt request flag USIMF will be set. If this bit is equal to "0", the USIM interrupt request flag USIMF will not be influenced by the condition of the UOERR or URXIF flags.

Bit 1 **UTIIE**: Transmitter Idle interrupt enable control
0: Transmitter idle interrupt is disabled
1: Transmitter idle interrupt is enabled

This bit enables or disables the transmitter idle interrupt. If this bit is equal to "1" and when the transmitter idle flag UTIDLE is set, due to a transmitter idle condition, the USIM interrupt request flag USIMF will be set. If this bit is equal to "0", the USIM interrupt request flag USIMF will not be influenced by the condition of the UTIDLE flag.

Bit 0 **UTEIE**: Transmitter Empty interrupt enable control
0: Transmitter empty interrupt is disabled
1: Transmitter empty interrupt is enabled

This bit enables or disables the transmitter empty interrupt. If this bit is equal to "1" and when the transmitter empty flag UTXIF is set, due to a transmitter empty condition, the USIM interrupt request flag USIMF will be set. If this bit is equal to "0", the USIM interrupt request flag USIMF will not be influenced by the condition of the UTXIF flag.

• **UTXR_RXR Register**

The UTXR_RXR register is the data register which is used to store the data to be transmitted on the TX pin or being received from the RX pin.

Bit	7	6	5	4	3	2	1	0
Name	UTXRX7	UTXRX6	UTXRX5	UTXRX4	UTXRX3	UTXRX2	UTXRX1	UTXRX0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x": unknown

Bit 7~0 **UTXRX7~UTXRX0**: UART Transmit/Receive Data bit 7 ~ bit 0

• **UBRG Register**

Bit	7	6	5	4	3	2	1	0
Name	UBRG7	UBRG6	UBRG5	UBRG4	UBRG3	UBRG2	UBRG1	UBRG0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x": unknown

Bit 7~0 **UBRG7~UBRG0**: Baud Rate values

By programming the UBRGH bit in UUCR2 Register which allows selection of the related formula described above and programming the required value in the UBRG register, the required baud rate can be setup.

Note: Baud rate=f_H/[64×(N+1)] if UBRGH=0.

Baud rate=f_H/[16×(N+1)] if UBRGH=1.

Baud Rate Generator

To setup the speed of the serial data communication, the UART function contains its own dedicated baud rate generator. The baud rate is controlled by its own internal free running 8-bit timer, the period of which is determined by two factors. The first of these is the value placed in the baud rate register UBRG and the second is the value of the UBRGH bit with the control register UUCR2. The UBRGH bit decides if the baud rate generator is to be used in a high speed mode or low speed mode, which in turn determines the formula that is used to calculate the baud rate. The value N in the UBRG register which is used in the following baud rate calculation formula determines the division factor. Note that N is the decimal value placed in the UBRG register and has a range of between 0 and 255.

UUCR2 UBRGH Bit	0	1
Baud Rate (BR)	f_H/[64 (N+1)]	f_H/[16 (N+1)]

By programming the UBRGH bit which allows selection of the related formula and programming the required value in the UBRG register, the required baud rate can be setup. Note that because the actual baud rate is determined using a discrete value, N, placed in the UBRG register, there will be an error associated between the actual and requested value. The following example shows how the UBRG register value N and the error value can be calculated.

Calculating the Baud Rate and Error Values

For a clock frequency of 4MHz, and with UBRGH cleared to zero determine the UBRG register value N, the actual baud rate and the error value for a desired baud rate of 4800.

From the above table the desired baud rate BR=f_H/[64 (N+1)]

Re-arranging this equation gives N=[f_H/(BR×64)]-1

Giving a value for N=[4000000/(4800×64)]-1=12.0208

To obtain the closest value, a decimal value of 12 should be placed into the UBRG register. This gives an actual or calculated baud rate value of BR=4000000/[64×(12+1)]=4808

Therefore the error is equal to (4808-4800)/4800=0.16%

UART Setup and Control

For data transfer, the UART function utilizes a non-return-to-zero, more commonly known as NRZ, format. This is composed of one start bit, eight or nine data bits, and one or two stop bits. Parity is supported by the UART hardware, and can be setup to be even, odd or no parity. For the most common data format, 8 data bits along with no parity and one stop bit, denoted as 8, N, 1, is used as the default setting, which is the setting at power-on. The number of data bits and stop bits, along with the parity, are setup by programming the corresponding UBNO, UPRT, UPREN, and USTOPS bits in the UUCR1 register. The baud rate used to transmit and receive data is setup using the internal 8-bit baud rate generator, while the data is transmitted and received LSB first. Although the UART transmitter and receiver are functionally independent, they both use the same data format and baud rate. In all cases stop bits will be used for data transmission.

Enabling/Disabling the UART Interface

The basic on/off function of the internal UART function is controlled using the UREN bit in the UUCR1 register. When the UART mode is selected by setting the UMD bit in the SIMC0 register to "1", if the UREN, UTXEN and URXEN bits are set, then these two UART pins will act as normal TX output pin and RX input pin respectively. If no data is being transmitted on the TX pin, then it will default to a logic high value.

Clearing the UREN bit will disable the TX and RX pins and allow these two pins to be used as normal I/O or other pin-shared functional pins by configuring the corresponding pin-shared control bits. When the UART function is disabled the buffer will be reset to an empty condition, at the same time discarding any remaining residual data. Disabling the UART will also reset the error and status flags with bits UTXEN, URXEN, UTXBRK, URXIF, UOERR, UFERR, UPERR and UNF being cleared while bits UTIDLE, UTXIF and URIDLE will be set. The remaining control bits in the UUCR1, UUCR2 and UBRG registers will remain unaffected. If the UREN bit in the UUCR1 register is cleared while the UART is active, then all pending transmissions and receptions will be immediately suspended and the UART will be reset to a condition as defined above. If the UART is then subsequently re-enabled, it will restart again in the same configuration.

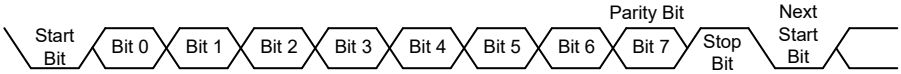
Data, Parity and Stop Bit Selection

The format of the data to be transferred is composed of various factors such as data bit length, parity on/off, parity type, address bits and the number of stop bits. These factors are determined by the setup of various bits within the UUCR1 register. The UBNO bit controls the number of data bits which can be set to either 8 or 9, the UPRT bit controls the choice of odd or even parity, the UPREN bit controls the parity on/off function and the USTOPS bit decides whether one or two stop bits are to be used. The following table shows various formats for data transmission. The address bit, which is the MSB of the data byte, identifies the frame as an address character or data if the address detect function is enabled. The number of stop bits, which can be either one or two, is independent of the data length and is only used for the transmitter. There is only one stop bit for the receiver.

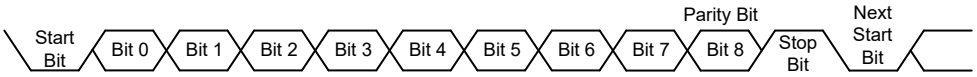
Start Bit	Data Bits	Address Bit	Parity Bit	Stop Bit
Example of 8-bit Data Formats				
1	8	0	0	1
1	7	0	1	1
1	7	1	0	1
Example of 9-bit Data Formats				
1	9	0	0	1
1	8	0	1	1
1	8	1	0	1

Transmitter Receiver Data Format

The following diagram shows the transmit and receive waveforms for both 8-bit and 9-bit data formats.



8-bit data format



9-bit data format

UART Transmitter

Data word lengths of either 8 or 9 bits can be selected by programming the UBNO bit in the UUCR1 register. When UBNO bit is set, the word length will be set to 9 bits. In this case the 9th bit, which is the MSB, needs to be stored in the UTX8 bit in the UUCR1 register. At the transmitter core lies the Transmitter Shift Register, more commonly known as the TSR, whose data is obtained from the transmit data register, which is known as the UTXR_RXR register. The data to be transmitted is loaded into this UTXR_RXR register by the application program. The TSR register is not written to with new data until the stop bit from the previous transmission has been sent out. As soon as this stop bit has been transmitted, the TSR can then be loaded with new data from the UTXR_RXR register, if it is available. It should be noted that the TSR register, unlike many other registers, is not directly mapped into the Data Memory area and as such is not available to the application program for direct read/write operations. An actual transmission of data will normally be enabled when the UTXEN bit is set, but the data will not be transmitted until the UTXR_RXR register has been loaded with data and the baud rate generator has defined a shift clock source. However, the transmission can also be initiated by first loading data into the UTXR_RXR register, after which the UTXEN bit can be set. When a transmission of data begins, the TSR is normally empty, in which case a transfer to the UTXR_RXR register will result in an immediate transfer to the TSR. If during a transmission the UTXEN bit is cleared, the transmission will immediately cease and the transmitter will be reset. The TX output pin can then be configured as the I/O or other pin-shared function by configuring the corresponding pin-shared control bits.

Transmitting Data

When the UART is transmitting data, the data is shifted on the TX pin from the shift register, with the least significant bit first. In the transmit mode, the UTXR_RXR register forms a buffer between the internal bus and the transmitter shift register. It should be noted that if 9-bit data format has been selected, then the MSB will be taken from the UTX8 bit in the UUCR1 register. The steps to initiate a data transfer can be summarized as follows:

- Make the correct selection of the UBNO, UPRT, UPREN and USTOPS bits to define the required word length, parity type and number of stop bits.

- Setup the UBRG register to select the desired baud rate.
- Set the UTXEN bit to ensure that the TX pin is used as a UART transmitter pin.
- Access the UUSR register and write the data that is to be transmitted into the UTXR_RXR register. Note that this step will clear the UTXIF bit.

This sequence of events can now be repeated to send additional data.

It should be noted that when UTXIF=0, data will be inhibited from being written to the UTXR_RXR register. Clearing the UTXIF flag is always achieved using the following software sequence:

1. A UUSR register access
2. A UTXR_RXR register write execution

The read-only UTXIF flag is set by the UART hardware and if set indicates that the UTXR_RXR register is empty and that other data can now be written into the UTXR_RXR register without overwriting the previous data. If the UTEIE bit is set then the UTXIF flag will generate an interrupt.

During a data transmission, a write instruction to the UTXR_RXR register will place the data into the UTXR_RXR register, which will be copied to the shift register at the end of the present transmission. When there is no data transmission in progress, a write instruction to the UTXR_RXR register will place the data directly into the shift register, resulting in the commencement of data transmission, and the UTXIF bit being immediately set. When a frame transmission is complete, which happens after stop bits are sent or after the break frame, the UTIDLE bit will be set. To clear the UTIDLE bit the following software sequence is used:

1. A UUSR register access
2. A UTXR_RXR register write execution

Note that both the UTXIF and UTIDLE bits are cleared by the same software sequence.

Transmitting Break

If the UTXBRK bit is set high and the state keeps for a time greater than $[(UBRG+1)\times t_H]$ while UTIDLE=1, then break characters will be sent on the next transmission. Break character transmission consists of a start bit, followed by 13×N '0' bits and stop bits, where N=1, 2, etc. If a break character is to be transmitted then the UTXBRK bit must be first set by the application program, and then cleared to generate the stop bits. Transmitting a break character will not generate a transmit interrupt. Note that a break condition length is at least 13 bits long. If the UTXBRK bit is continually kept at a logic high level then the transmitter circuitry will transmit continuous break characters. After the application program has cleared the UTXBRK bit, the transmitter will finish transmitting the last break character and subsequently send out one or two stop bits. The automatic logic highs at the end of the last break character will ensure that the start bit of the next frame is recognized.

UART Receiver

The UART is capable of receiving word lengths of either 8 or 9 bits. If the UBNO bit is set, the word length will be set to 9 bits with the MSB being stored in the URX8 bit of the UUCR1 register. At the receiver core lies the Receive Serial Shift Register, commonly known as the RSR. The data which is received on the RX external input pin is sent to the data recovery block. The data recovery block operating speed is 16 times that of the baud rate, while the main receive serial shifter operates at the baud rate. After the RX pin is sampled for the stop bit, the received data in RSR is transferred to the receive data register, if the register is empty. The data which is received on the external RX input pin is sampled three times by a majority detect circuit to determine the logic level that has been placed onto the RX pin. It should be noted that the RSR register, unlike many other registers, is not directly mapped into the Data Memory area and as such is not available to the application program for direct read/write operations.

Receiving Data

When the UART receiver is receiving data, the data is serially shifted in on the external RX input pin, LSB first. In the read mode, the UTXR_RXR register forms a buffer between the internal bus and the receiver shift register. The UTXR_RXR register is a two byte deep FIFO data buffer, where two bytes can be held in the FIFO while a third byte can continue to be received. Note that the application program must ensure that the data is read from UTXR_RXR before the third byte has been completely shifted in, otherwise this third byte will be discarded and an overrun error UOERR will be subsequently indicated. The steps to initiate a data transfer can be summarized as follows:

- Make the correct selection of UBNO, UPRT and UPREN bits to define the word length, parity type.
- Setup the UBRG register to select the desired baud rate.
- Set the URXEN bit to ensure that the RX pin is used as a UART receiver pin.

At this point the receiver will be enabled which will begin to look for a start bit.

When a character is received the following sequence of events will occur:

- The URXIF bit in the UUSR register will be set when the UTXR_RXR register has data available. There will be at most one more character available before an overrun error occurs.
- When the contents of the shift register have been transferred to the UTXR_RXR register, then if the URIE bit is set, an interrupt will be generated.
- If during reception, a frame error, noise error, parity error, or an overrun error has been detected, then the error flags can be set.

The URXIF bit can be cleared using the following software sequence:

1. A UUSR register access
2. A UTXR_RXR register read execution

Receiving Break

Any break character received by the UART will be managed as a framing error. The receiver will count and expect a certain number of bit times as specified by the values programmed into the UBNO bit plus one stop bit. If the break is much longer than 13 bit times, the reception will be considered as complete after the number of bit times specified by UBNO plus one stop bit. The URXIF bit is set, UFERR is set, zeros are loaded into the receive data register, interrupts are generated if appropriate and the URIDLE bit is set. A break is regarded as a character that contains only zeros with the UFERR flag set. If a long break signal has been detected, the receiver will regard it as a data frame including a start bit, data bits and the invalid stop bit and the UFERR flag will be set. The receiver must wait for a valid stop bit before looking for the next start bit. The receiver will not make the assumption that the break condition on the line is the next start bit. The break character will be loaded into the buffer and no further data will be received until stop bits are received. It should be noted that the URIDLE read only flag will go high when the stop bits have not yet been received. The reception of a break character on the UART registers will result in the following:

- The framing error flag, UFERR, will be set.
- The receive data register, UTXR_RXR, will be cleared.
- The UOERR, UNF, UPERR, URIDLE or URXIF flags will possibly be set.

Idle Status

When the receiver is reading data, which means it will be in between the detection of a start bit and the reading of a stop bit, the receiver status flag in the UUSR register, otherwise known as the URIDLE flag, will have a zero value. In between the reception of a stop bit and the detection of the next start bit, the URIDLE flag will have a high value, which indicates the receiver is in an idle condition.

Receiver Interrupt

The read only receive interrupt flag URXIF in the UUSR register is set by an edge generated by the receiver. An interrupt is generated if URIE=1, when a word is transferred from the Receive Shift Register, RSR, to the Receive Data Register, UTXR_RXR. An overrun error can also generate an interrupt if URIE=1.

Managing Receiver Errors

Several types of reception errors can occur within the UART module, the following section describes the various types and how they are managed by the UART.

Overrun Error – UOERR

The UTXR_RXR register is composed of a two byte deep FIFO data buffer, where two bytes can be held in the FIFO register, while a third byte can continue to be received. Before this third byte has been entirely shifted in, the data should be read from the UTXR_RXR register. If this is not done, the overrun error flag UOERR will be consequently indicated.

In the event of an overrun error occurring, the following will happen:

- The UOERR flag in the UUSR register will be set.
- The UTXR_RXR contents will not be lost.
- The shift register will be overwritten.
- An interrupt will be generated if the URIE bit is set.

The UOERR flag can be cleared by an access to the UUSR register followed by a read to the UTXR_RXR register.

Noise Error – UNF

Over-sampling is used for data recovery to identify valid incoming data and noise. If noise is detected within a frame the following will occur:

- The read only noise flag, UNF, in the UUSR register will be set on the rising edge of the URXIF bit.
- Data will be transferred from the Shift register to the UTXR_RXR register.
- No interrupt will be generated. However this bit rises at the same time as the URXIF bit which itself generates an interrupt.

Note that the UNF flag is reset by a UUSR register read operation followed by a UTXR_RXR register read operation.

Framing Error – UFERR

The read only framing error flag, UFERR, in the UUSR register, is set if a zero is detected instead of stop bits. If two stop bits are selected, both stop bits must be high; otherwise the UFERR flag will be set. The UFERR flag and the received data will be recorded in the UUSR and UTXR_RXR registers respectively, and the flag is cleared in any reset.

Parity Error – UPERR

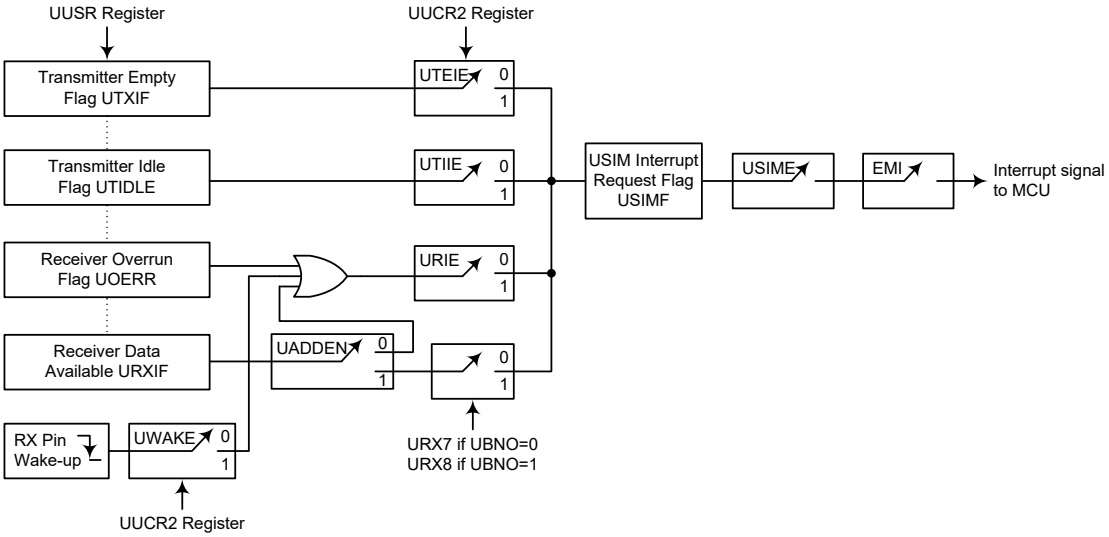
The read only parity error flag, UPERR, in the UUSR register, is set if the parity of the received word is incorrect. This error flag is only applicable if the parity is enabled, UPREN = 1, and if the parity type, odd or even is selected. The read only UPERR flag and the received data will be recorded in the UUSR and UTXR_RXR registers respectively. It is cleared on any reset, it should be noted that the flags, UFERR and UPERR, in the UUSR register should first be read by the application program before reading the data word.

UART Interrupt Structure

Several individual UART conditions can trigger an USIM interrupt. When these conditions exist, a low pulse will be generated to get the attention of the microcontroller. These conditions are a transmitter data register empty, transmitter idle, receiver data available, receiver overrun, address detect and an RX pin wake-up. When any of these conditions are created, if the global interrupt enable bit and the USIM interrupt control bit are enabled and the stack is not full, the program will jump to its corresponding interrupt vector where it can be serviced before returning to the main program. Four of these conditions have the corresponding UUSR register flags which will generate an USIM interrupt if its associated interrupt enable control bit in the UUCR2 register is set. The two transmitter interrupt conditions have their own corresponding enable control bits, while the two receiver interrupt conditions have a shared enable control bit. These enable bits can be used to mask out individual USIM UART mode interrupt sources.

The address detect condition, which is also an USIM UART mode interrupt source, does not have an associated flag, but will generate an USIM interrupt when an address detect condition occurs if its function is enabled by setting the UADDEN bit in the UUCR2 register. An RX pin wake-up, which is also an USIM UART mode interrupt source, does not have an associated flag, but will generate an USIM interrupt if the UART clock (f_H) source is switched off and the UWAKE and URIE bits in the UUCR2 register are set when a falling edge on the RX pin occurs. Note that in the event of an RX wake-up interrupt occurring, there will be a certain period of delay, commonly known as the System Start-up Time, for the oscillator to restart and stabilize before the system resumes normal operation.

Note that the UUSR register flags are read only and cannot be cleared or set by the application program, neither will they be cleared when the program jumps to the corresponding interrupt servicing routine, as is the case for some of the other interrupts. The flags will be cleared automatically when certain actions are taken by the UART, the details of which are given in the UART register section. The overall UART interrupt can be disabled or enabled by the USIM interrupt enable control bit in the interrupt control register of the microcontroller to decide whether the interrupt requested by the UART module is masked out or allowed.



UART Interrupt Structure

Address Detect Mode

Setting the Address Detect Mode bit, UADDEN, in the UUCR2 register, enables this special mode. If this bit is enabled then an additional qualifier will be placed on the generation of a Receiver Data Available interrupt, which is requested by the URXIF flag. If the UADDEN bit is enabled, then when data is available, an interrupt will only be generated, if the highest received bit has a high value. Note that the USIME and EMI interrupt enable bits must also be enabled for correct interrupt generation. This highest address bit is the 9th bit if UBNO=1 or the 8th bit if UBNO=0. If this bit is high, then the received word will be defined as an address rather than data. A Data Available interrupt will be generated every time the last bit of the received word is set. If the UADDEN bit is not enabled, then a Receiver Data Available interrupt will be generated each time the URXIF flag is set, irrespective of the data last bit status. The address detect mode and parity enable are mutually exclusive functions. Therefore if the address detect mode is enabled, then to ensure correct operation, the parity function should be disabled by resetting the parity enable bit UPREN to zero.

UADDEN	Bit 9 if UBNO=1, Bit 8 if UBNO=0	USIM Interrupt Generated
0	0	√
	1	√
1	0	×
	1	√

UADDEN Bit Function

UART Power Down and Wake-up

When the UART clock (f_H) is off, the UART will cease to function, all clock sources to the module are shutdown. If the UART clock (f_H) is off while a transmission is still in progress, then the transmission will be paused until the UART clock source derived from the microcontroller is activated. In a similar way, if the MCU enters the IDLE or SLEEP Mode while receiving data, then the reception of data will likewise be paused. When the MCU enters the IDLE or SLEEP Mode, note that the UUSR, UUCR1, UUCR2, transmit and receive registers, as well as the UBRG register will not be affected. It is recommended to make sure first that the UART data transmission or reception has been finished before the microcontroller enters the IDLE or SLEEP mode.

The UART function contains a receiver RX pin wake-up function, which is enabled or disabled by the UWAKE bit in the UUCR2 register. If this bit, along with the UART mode selection bit, UMD, the UART enable bit, UREN, the receiver enable bit, URXEN and the receiver interrupt bit, URIE, are all set when the UART clock (f_H) is off, then a falling edge on the RX pin will trigger an RX pin wake-up UART interrupt. Note that as it takes certain system clock cycles after a wake-up, before normal microcontroller operation resumes, any data received during this time on the RX pin will be ignored.

For a UART wake-up interrupt to occur, in addition to the bits for the wake-up being set, the global interrupt enable bit, EMI, and the USIM interrupt enable bit, USIME, must be set. If the EMI and USIME bits are not set then only a wake up event will occur and no interrupt will be generated. Note also that as it takes certain system clock cycles after a wake-up before normal microcontroller resumes, the USIM interrupt will not be generated until after this time has elapsed.

Low Voltage Detector – LVD

The device has a Low Voltage Detector function, also known as LVD. This enables the device to monitor the power supply voltage, V_{DD}, and provide a warning signal should it fall below a certain level. This function may be especially useful in battery applications where the supply voltage will gradually reduce as the battery ages, as it allows an early warning battery low signal to be generated. The Low Voltage Detector also has the capability of generating an interrupt signal.

LVD Register

The Low Voltage Detector function is controlled using a single register with the name LVDC. Three bits in this register, VLVD2~VLVD0, are used to select one of eight fixed voltages below which a low voltage condition will be determined. A low voltage condition is indicated when the LVDO bit is set. If the LVDO bit is low, this indicates that the V_{DD} voltage is above the preset low voltage value. The LVDEN bit is used to control the overall on/off function of the low voltage detector. Setting the bit high will enable the low voltage detector. Clearing the bit to zero will switch off the internal low voltage detector circuits. As the low voltage detector will consume a certain amount of power, it may be desirable to switch off the circuit when not in use, an important consideration in power sensitive battery powered applications.

- **LVDC Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	LVDO	LVDEN	VBGEN	VLVD2	VLVD1	VLVD0
R/W	—	—	R	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5 **LVDO**: LVD Output Flag
0: No Low Voltage Detected
1: Low Voltage Detected

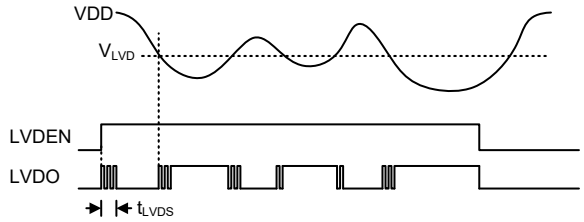
Bit 4 **LVDEN**: Low Voltage Detector Control
0: Disable
1: Enable

Bit 3 **VBGEN**: Bandgap Buffer Control
0: Disable
1: Enable
Note that the Bandgap circuit is enabled when the LVD or the LVR function is enabled or when the VBGEN bit is set high.

Bit 2~0 **VLVD2~VLVD0**: Select LVD Voltage
000: 2.0V
001: 2.2V
010: 2.4V
011: 2.7V
100: 3.0V
101: 3.3V
110: 3.6V
111: 4.0V

LVD Operation

The Low Voltage Detector function operates by comparing the power supply voltage, V_{DD}, with a pre-specified voltage level stored in the LVDC register. This has a range of between 2.0V and 4.0V. When the power supply voltage, V_{DD}, falls below this pre-determined value, the LVDO bit will be set high indicating a low power supply voltage condition. The Low Voltage Detector function is supplied by a reference voltage which will be automatically enabled. When the device is in the SLEEP mode, the low voltage detector will be disabled even if the LVDEN bit is high. After enabling the Low Voltage Detector, a time delay t_{LVDS} should be allowed for the circuitry to stabilise before reading the LVDO bit. Note also that as the V_{DD} voltage may rise and fall rather slowly, at the voltage nears that of V_{LVD}, there may be multiple bit LVDO transitions.



LVD Operation

The Low Voltage Detector also has its own interrupt, providing an alternative means of low voltage detection, in addition to polling the LVDO bit. The interrupt will only be generated after a delay of t_{LVD} after the LVDO bit has been set high by a low voltage condition. In this case, the LVF interrupt request flag will be set, causing an interrupt to be generated if V_{DD} falls below the preset LVD voltage. This will cause the device to wake-up from the IDLE Mode, however if the Low Voltage Detector wake up function is not required then the LVF flag should be first set high before the device enter the IDLE Mode. The LVD function is always disabled in the SLEEP mode.

Interrupts

Interrupts are an important part of any microcontroller system. When an external event or an internal function such as a Timer Module or an A/D converter requires microcontroller attention, their corresponding interrupt will enforce a temporary suspension of the main program allowing the microcontroller to direct attention to their respective needs. The device contains several external interrupt and internal interrupts functions. The external interrupts are generated by the action of the external INT0~INT1 pins, while the internal interrupts are generated by various internal functions such as the TMs, Time Bases, LVD, EEPROM and the A/D converter.

Interrupt Registers

Overall interrupt control, which basically means the setting of request flags when certain microcontroller conditions occur and the setting of interrupt enable bits by the application program, is controlled by a series of registers, located in the Special Purpose Data Memory. The first is the INTC0~INTC3 registers which setup the primary interrupts, the second is the INTEG register which setup the external interrupt trigger edge type.

Each register contains a number of enable bits to enable or disable individual registers as well as interrupt flags to indicate the presence of an interrupt request. The naming convention of these follows a specific pattern. First is listed an abbreviated interrupt type, then the (optional) number of that interrupt followed by either an “E” for enable/disable bit or “F” for request flag.

Function	Enable Bit	Request Flag	Notes
Global	EMI	—	—
INTn Pin	INTnE	INTnF	n=0~1
PLT Comparator	PLTCnE	PLTCnF	n=0~1
A/D Converter	ADE	ADF	—
Time Base	TBnE	TBnF	n=0~1
USIM	USIME	USIMF	—
LVD	LVE	LVF	—
EEPROM	DEE	DEF	—
STM	STMPE	STMPF	—
	STMAE	STMAF	
PTM	PTMPE	PTMPF	—
	PTMAE	PTMAF	

Interrupt Register Bit Naming Conventions

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	INT1F	INT0F	PLTC0F	INT1E	INT0E	PLTC0E	EMI
INTC1	DEF	ADF	LVF	USIMF	DEE	ADE	LVE	USIME
INTC2	STMAF	STMPF	PTMAF	PTMPF	STMAE	STMPE	PTMAE	PTMPE
INTC3	—	PLTC1F	TB1F	TB0F	—	PLTC1E	TB1E	TB0E

Interrupt Register List

- **INTEG Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4 Unimplemented, read as "0"

Bit 3~2 **INT1S1~INT1S0**: Interrupt edge control for INT1 pin
00: Disable
01: Rising edge
10: Falling edge
11: Rising and falling edges

Bit 1~0 **INT0S1~INT0S0**: Interrupt edge control for INT0 pin
00: Disable
01: Rising edge
10: Falling edge
11: Rising and falling edges

• **INTC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	INT1F	INT0F	PLTC0F	INT1E	INT0E	PLTC0E	EMI
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **INT1F**: INT1 interrupt request flag
0: No request
1: Interrupt request

Bit 5 **INT0F**: INT0 interrupt request flag
0: No request
1: Interrupt request

Bit 4 **PLTC0F**: PLT Comparator 0 interrupt request flag
0: No request
1: Interrupt request

Bit 3 **INT1E**: INT1 interrupt control
0: Disable
1: Enable

Bit 2 **INT0E**: INT0 interrupt control
0: Disable
1: Enable

Bit 1 **PLTC0E**: PLT Comparator 0 interrupt control
0: Disable
1: Enable

Bit 0 **EMI**: Global interrupt control
0: Disable
1: Enable

• **INTC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	DEF	ADF	LVF	USIMF	DEE	ADE	LVE	USIME
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **DEF**: Data EEPROM interrupt request flag
0: No request
1: Interrupt request

Bit 6 **ADF**: A/D Converter interrupt request flag
0: No request
1: Interrupt request

Bit 5 **LVF**: LVD interrupt request flag
0: No request
1: Interrupt request

Bit 4 **USIMF**: USIM interrupt request flag
0: No request
1: Interrupt request

Bit 3 **DEE**: Data EEPROM interrupt control
0: Disable
1: Enable

Bit 2 **ADE**: A/D Converter interrupt control
0: Disable
1: Enable

Bit 1 **LVE**: LVD interrupt control
0: Disable
1: Enable

Bit 0 **USIME**: USIM interrupt control
0: Disable
1: Enable

• **INTC2 Register**

Bit	7	6	5	4	3	2	1	0
Name	STMAF	STMPF	PTMAF	PTMPF	STMAE	STMPE	PTMAE	PTMPE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **STMAF**: STM Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 6 **STMPF**: STM Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 5 **PTMAF**: PTM Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **PTMPF**: PTM Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 3 **STMAE**: STM Comparator A match interrupt control
0: Disable
1: Enable

Bit 2 **STMPE**: STM Comparator P match interrupt control
0: Disable
1: Enable

Bit 1 **PTMAE**: PTM Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **PTMPE**: PTM Comparator P match interrupt control
0: Disable
1: Enable

• **INTC3 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	PLTC1F	TB1F	TB0F	—	PLTC1E	TB1E	TB0E
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **PLTC1F**: PLT Comparator 1 interrupt request flag
0: No request
1: Interrupt request

Bit 5 **TB1F**: Time Base 1 interrupt request flag
0: No request
1: Interrupt request

Bit 4 **TB0F**: Time Base 0 interrupt request flag
0: No request
1: Interrupt request

Bit 3 Unimplemented, read as "0"

Bit 2 **PLTC1E**: PLT Comparator 1 interrupt control
0: Disable
1: Enable

Bit 1 **TB1E**: Time Base 1 interrupt control
0: Disable
1: Enable

Bit 0 **TB0E**: Time Base 0 interrupt control
0: Disable
1: Enable

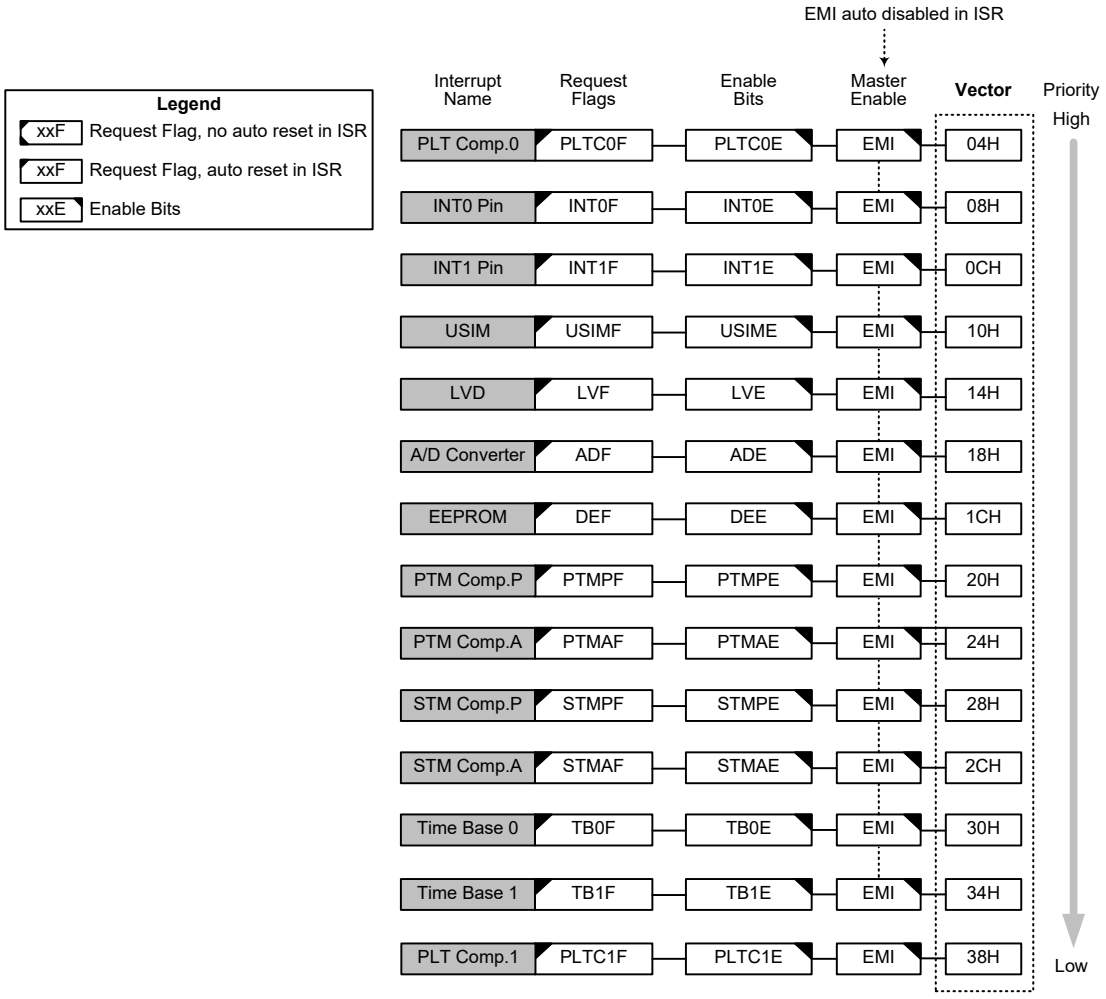
Interrupt Operation

When the conditions for an interrupt event occur, such as a TM Comparator P, Comparator A match or A/D conversion completion etc., the relevant interrupt request flag will be set. Whether the request flag actually generates a program jump to the relevant interrupt vector is determined by the condition of the interrupt enable bit. If the enable bit is set high then the program will jump to its relevant vector; if the enable bit is zero then although the interrupt request flag is set an actual interrupt will not be generated and the program will not jump to the relevant interrupt vector. The global interrupt enable bit, if cleared to zero, will disable all interrupts.

When an interrupt is generated, the Program Counter, which stores the address of the next instruction to be executed, will be transferred onto the stack. The Program Counter will then be loaded with a new address which will be the value of the corresponding interrupt vector. The microcontroller will then fetch its next instruction from this interrupt vector. The instruction at this vector will usually be a "JMP" which will jump to another section of program which is known as the interrupt service routine. Here is located the code to control the appropriate interrupt. The interrupt service routine must be terminated with a "RETI", which retrieves the original Program Counter address from the stack and allows the microcontroller to continue with normal execution at the point where the interrupt occurred.

The various interrupt enable bits, together with their associated request flags, are shown in the accompanying diagrams with their order of priority. All interrupt sources have their own individual vector. Once an interrupt subroutine is serviced, all the other interrupts will be blocked, as the global interrupt enable bit, EMI bit will be cleared automatically. This will prevent any further interrupt nesting from occurring. However, if other interrupt requests occur during this interval, although the interrupt will not be immediately serviced, the request flag will still be recorded.

If an interrupt requires immediate servicing while the program is already in another interrupt service routine, the EMI bit should be set after entering the routine, to allow interrupt nesting. If the stack is full, the interrupt request will not be acknowledged, even if the related interrupt is enabled, until the Stack Pointer is decremented. If immediate service is desired, the stack must be prevented from becoming full. In case of simultaneous requests, the accompanying diagram shows the priority that is applied. All of the interrupt request flags when set will wake-up the device if it is in SLEEP or IDLE Mode, however to prevent a wake-up from occurring the corresponding flag should be set before the device are in SLEEP or IDLE Mode.



Interrupt Structure

External Interrupts

The external interrupt is controlled by signal transitions on the INTn pin. An external interrupt request will take place when the external interrupt request flag, INTnF, is set, which will occur when a transition, whose type is chosen by the edge select bits, appears on the external interrupt pin. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and respective external interrupt enable bit, INTnE, must first be set. Additionally the correct interrupt edge type must be selected using the INTEG register to enable the external interrupt function and to choose the trigger edge type. As the external interrupt pin is pin-shared with I/O pin, it can only be configured as external interrupt pin if its external interrupt enable bit in the corresponding interrupt register has been set and the external interrupt pin is selected by the corresponding pin-shared function selection bits. The pin must also be setup as an input by setting the corresponding bit in the port control register. When the interrupt is enabled, the stack is not full and the correct transition type appears on the external interrupt pin, a subroutine call to the external interrupt vector, will take place. When the interrupt is serviced, the external interrupt request flag, INTnF, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts. Note that any pull-high resistor selections on the external interrupt pin will remain valid even if the pin is used as an external interrupt input. The INTEG register is used to select the type

of active edge that will trigger the external interrupt. A choice of either rising or falling or both edge types can be chosen to trigger an external interrupt. Note that the INTEG register can also be used to disable the external interrupt function.

PLT Comparator Interrupts

The PLT comparator interrupts are controlled by the Powerline Transceiver circuit internal comparators. A PLT comparator n interrupt request will take place when the PLT comparator n interrupt request flag, PLTCnF, is set, a situation that will occur when the PLT comparator n output bit changes state. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and PLT comparator n interrupt enable bit, PLTCnE, must first be set. When the interrupt is enabled, the stack is not full and the PLT comparator n inputs generate a comparator output transition, a subroutine call to the PLT comparator n interrupt vector, will take place. When the interrupt is serviced, the PLT comparator interrupt request flag, PLTCnF, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts.

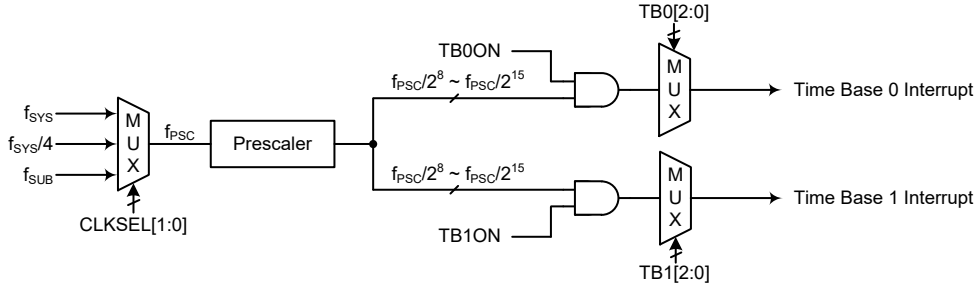
A/D Converter Interrupt

An A/D Converter Interrupt request will take place when the A/D Converter Interrupt request flag, ADF, is set, which occurs when the A/D conversion process finishes. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and A/D Interrupt enable bit, ADE, must first be set. When the interrupt is enabled, the stack is not full and the A/D conversion process has ended, a subroutine call to the A/D Converter Interrupt vector, will take place. When the interrupt is serviced, the A/D Converter Interrupt flag, ADF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

Time Base Interrupts

The function of the Time Base Interrupts is to provide regular time signal in the form of an internal interrupt. They are controlled by the overflow signals from their respective timer functions. When these happens their respective interrupt request flags, TBnF will be set. To allow the program to branch to their respective interrupt vector addresses, the global interrupt enable bit, EMI and Time Base enable bits, TBnE, must first be set. When the interrupt is enabled, the stack is not full and the Time Base overflows, a subroutine call to their respective vector locations will take place. When the interrupt is serviced, the respective interrupt request flag, TBnF, will be automatically reset and the EMI bit will be cleared to disable other interrupts.

The purpose of the Time Base Interrupt is to provide an interrupt signal at fixed time periods. Its clock source, f_{PSC}, originates from the internal clock source f_{SYS}, $f_{SYS}/4$ or f_{SUB} and then passes through a divider, the division ratio of which is selected by programming the appropriate bits in the TBnC register to obtain longer interrupt periods whose value ranges. The clock source which in turn controls the Time Base interrupt period is selected using the CLKSEL1~CLKSEL0 bits in the PSCR register.



Time Base Interrupt

• **PSCR Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CLKSEL1	CLKSEL0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"
Bit 1~0 **CLKSEL1~CLKSEL0**: Prescaler clock source selection
00: f_{SYS}
01: $f_{SYS}/4$
1x: f_{SUB}

• **TBnC Register (n=0 or 1)**

Bit	7	6	5	4	3	2	1	0
Name	TBnON	—	—	—	—	TBn2	TBn1	TBn0
R/W	R/W	—	—	—	—	R/W	R/W	R/W
POR	0	—	—	—	—	0	0	0

Bit 7 **TBnON**: Time Base n Control
0: Disable
1: Enable
Bit 6~3 Unimplemented, read as "0"
Bit 2~0 **TBn2~TBn0**: Select Time Base n Time-out Period
000: $2^8/f_{PSC}$
001: $2^9/f_{PSC}$
010: $2^{10}/f_{PSC}$
011: $2^{11}/f_{PSC}$
100: $2^{12}/f_{PSC}$
101: $2^{13}/f_{PSC}$
110: $2^{14}/f_{PSC}$
111: $2^{15}/f_{PSC}$

USIM Interrupt

The Universal Serial Interface Module Interrupt, also known as the USIM interrupt, will take place when the USIM Interrupt request flag, USIMF, is set. As the USIM interface can operate in three modes which are SPI mode, I^2C mode and UART mode, the USIMF flag can be set by different conditions depending on the selected interface mode.

If the SPI or I^2C mode is selected, the USIM interrupt can be triggered when a byte of data has been received or transmitted by the USIM SPI or I^2C interface, or an I^2C slave address match occurs, or an I^2C bus time-out occurs. If the UART mode is selected, several individual UART conditions including a transmitter data register empty, transmitter idle, receiver data available, receiver overrun, address detect and an RX pin wake-up, can generate a USIM interrupt with the USIMF flag bit set high.

To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the Universal Serial Interface Module Interrupt enable bit, USIME, must first be set. When the interrupt is enabled, the stack is not full and any of the above described situations occurs, a subroutine call to the respective Interrupt vector, will take place. When the interrupt is serviced, the Universal Serial Interface Module Interrupt flag, USIMF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

Note that if the USIM interrupt is triggered by the UART interface, after the interrupt has been serviced, the UUSR register flags will only be cleared when certain actions are taken by the UART, the details of which are given in the UART section.

LVD Interrupt

An LVD Interrupt request will take place when the LVD Interrupt request flag, LVF, is set, which occurs when the Low Voltage Detector function detects a low power supply voltage. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and Low Voltage Interrupt enable bit, LVE, must first be set. When the interrupt is enabled, the stack is not full and a low voltage condition occurs, a subroutine call to the LVD Interrupt vector, will take place. When the Low Voltage Interrupt is serviced, the LVD Interrupt flag, LVF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

EEPROM Interrupt

An EEPROM Interrupt request will take place when the EEPROM Interrupt request flag, DEF, is set, which occurs when an EEPROM Write cycle ends. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and EEPROM Interrupt enable bit, DEE, must first be set. When the interrupt is enabled, the stack is not full and an EEPROM Write cycle ends, a subroutine call to the EEPROM Interrupt vector will take place. When the EEPROM Interrupt is serviced, the EEPROM Interrupt flag, DEF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

TM Interrupts

The Standard and Periodic Type TMs each has two interrupts, one comes from the comparator A match situation and the other comes from the comparator P match situation. For each of the Standard and Periodic Type TM there are two interrupt request flags xTMPF and xTMAF and two enable bits xTMPE and xTMAE. A TM interrupt request will take place when any of the TM request flags are set, a situation which occurs when a TM comparator P or A match situation happens.

To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, respective TM Interrupt enable bit must first be set. When the interrupt is enabled, the stack is not full and a TM comparator match situation occurs, a subroutine call to the TM comparator P or comparator A Interrupt vector locations, will take place. When the TM interrupt is serviced, the TM interrupt request flag will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

Interrupt Wake-up Function

Each of the interrupt functions has the capability of waking up the microcontroller when in the SLEEP or IDLE Mode. A wake-up is generated when an interrupt request flag changes from low to high and is independent of whether the interrupt is enabled or not. Therefore, even though the device is in the SLEEP or IDLE Mode and its system oscillator stopped, situations such as external edge transitions on the external interrupt pins or a low power supply voltage may cause their respective interrupt flag to be set high and consequently generate an interrupt. Care must therefore be taken if spurious wake-up situations are to be avoided. If an interrupt wake-up function is to be disabled then the corresponding interrupt request flag should be set high before the device enter the SLEEP or IDLE Mode. The interrupt enable bits have no effect on the interrupt wake-up function.

Programming Considerations

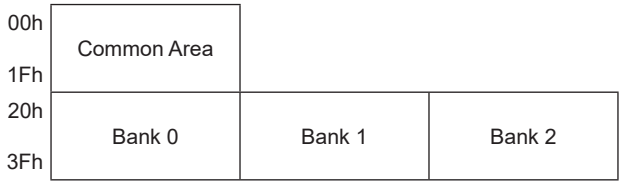
By disabling the relevant interrupt enable bits, a requested interrupt can be prevented from being serviced, however, once an interrupt request flag is set, it will remain in this condition in the interrupt register until the corresponding interrupt is serviced or until the request flag is cleared by the application program.

It is recommended that programs do not use the "CALL" instruction within the interrupt service subroutine. Interrupts often occur in an unpredictable manner or need to be serviced immediately. If only one stack is left and the interrupt is not well controlled, the original control sequence will be damaged once a CALL subroutine is executed in the interrupt subroutine.

Every interrupt has the capability of waking up the microcontroller when it is in SLEEP or IDLE Mode, the wake up being generated when the interrupt request flag changes from low to high. If it is required to prevent a certain interrupt from waking up the microcontroller then its respective request flag should be first set high before enter SLEEP or IDLE Mode.

As only the Program Counter is pushed onto the stack, then when the interrupt is serviced, if the contents of the accumulator, status register or other registers are altered by the interrupt service program, their contents should be saved to the memory at the beginning of the interrupt service routine. To return from an interrupt subroutine, either a RET or RETI instruction may be executed. The RETI instruction in addition to executing a return to the main program also automatically sets the EMI bit high to allow further interrupts. The RET instruction however only executes a return to the main program leaving the EMI bit in its present zero state and therefore disabling the execution of further interrupts.

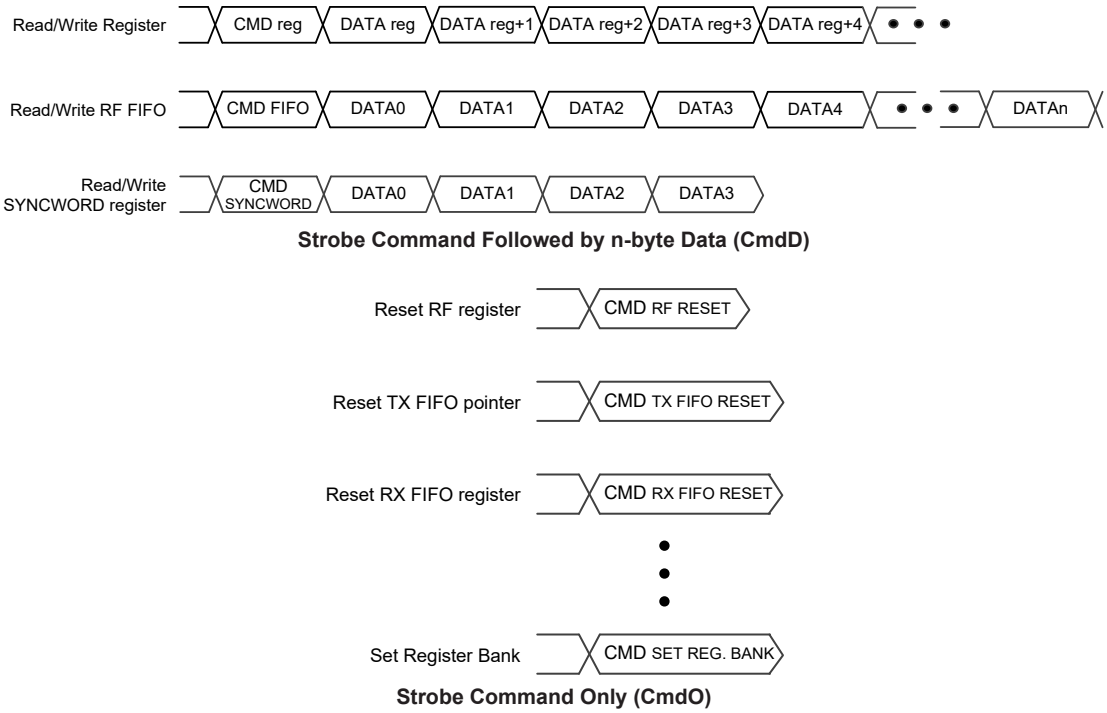
Memory Mapping



- Common Area: It contains 32 bytes space. Accessing addresses 00h~1Fh always means to access the Common Area regardless of Bank Pointer configuration.
- Bank 0~2: Each bank contains 32 bytes space. They are selected by the Bank Pointer.

The Bank Pointer, BANK[1:0], which is defined in the Common Area, can be set directly by the Set Register Bank command and read/written by the Control Register command.

Control Register Access



SFR Mapping and Bit Definition

Common Area Control Register

All control registers will be set to their initial value by power-on reset (POR). The software reset will set the control registers to their initial value except the FSYCK_EN, FSYCK_DIV[1:0], PWRON, GIO1S[2:0], GIO2S[2:0], PADDS[1:0], GIO4S[3:0], GIOPU[4:1], SPIPU, SDO_TEN bits in the RC1, IO1, IO2 and IO3 registers. These bits keep unchanged after software reset.

Addr.	Name	Bit							
		7	6	5	4	3	2	1	0
00h	CFG1	—	AGC_EN	RXCON_EN	DIR_EN	—	—	BANK[1:0]	
01h	RC1	PWRON	FSYCK_RDY	XCLK_RDY	XCLK_EN	FSYCK_DIV[1:0]		FSYCK_EN	RST_LL
02h	IRQ1	RXTO	RXFFOW	—	—	RXDETS[1:0]		IRQCPOR	IRQPOR
03h	IRQ2	ARKTFIE	ATRCTIE	FIFOLTIE	RXERRIE	RXDETIE	CALCMPIE		TXCMPIE
04h	IRQ3	ARKTFIF	ATRCTIF	FIFOLTIF	RXERRIF	RXDETIF	CALCMPIF	RXCMPIF	TXCMPIF
06h	IO1	PADDS[1:0]		GIO2S[2:0]			GIO1S[2:0]		
07h	IO2	GIO4S[3:0]				D3	D2	D1	D0
08h	IO3	SDO_TEN	SPIPU	—	GIOPU[4:1]				—
09h	FIFO1	—	—	TXFFSA[5:0]					
0Ah	FIFO2	—	—	—	RXPL2F_EN	FFINF_EN	FFMG_EN	FFMG[1:0]	
0Bh	PKT1	TXPMLEN[7:0]							
0Ch	PKT2	PID[1:0]		TRAILER_EN	WHTFMT	SYNCLEN[1:0]		RXPMLEN[1:0]	
0Dh	PKT3	MCH_EN	FEC_EN	CRC_EN	CRCFMT	PLLEN_EN	PLHAC_EN	PLHLEN	PLH_EN
0Eh	PKT4	WHT_EN	WHTSD[6:0]						
0Fh	PKT5	TXDLEN[7:0]							
10h	PKT6	RXDLEN[7:0]							
11h	PKT7	RXPID[1:0]		DLY_RXS[2:0]			DLY_TXS[2:0]		
12h	PKT8	—		PLHA[5:0]					
13h	PKT9	PLHEA[7:0]							
14h	MOD1	DTR[7:0]							
15h	MOD2	RXIFOS[11:8]				DITHER[1:0]		—	DTR[8]
16h	MOD3	RXIFOS[7:0]							
17h	DM1	—	—	MDIV[5:0]					
18h	DM2	PREAMBLE_CFO_EN1	PREAMBLE_CFO_EN0	SDR[5:0]					
19h	DM3	CSF_SW_EN	FD_MOD[6:0]						
1Ah	DM4	THOLD[3:0]				CFO_DSEL	—	PH_DIFF_MOD	PRE_CSF_EN
1Bh	DM5	FD_HOLD[7:0]							
1Eh	DM8	M_RATIO[7:0]							

Note: Addresses 05h, 1Ch, 1Dh and 1Fh, which are not listed in this table are reserved for future use, it is suggested not to change their initial values by any methods.

The reset value shown in the following register description tables means the software reset results of strobe command.

• **CFG1: Configuration Control Register 1**

Bit	7	6	5	4	3	2	1	0
Name	—	AGC_EN	RXCON_EN	DIR_EN	—	—	BANK[1:0]	
R/W	—	R/W	R/W	R/W	—	—	R/W	
Reset	0	0	0	0	0	0	0	0

Bit 7 Reserved, must be "0"

Bit 6 **AGC_EN**: AGC enable
0: Disable
1: Enable

Bit 5 **RXCON_EN**: RX continue mode enable
0: Disable
1: Enable
Note that this bit only affects normal RX mode and ATR RX mode without ARK function.

Bit 4 **DIR_EN**: Direct mode enable
0: TX/RX data from packet handling hardware
1: TX/RX data from/to external MCU directly

Bit 3~2 Reserved, must be "00"

Bit 1~0 **BANK[1:0]**: Control register bank selection
00: Bank 0
01: Bank 1
10: Bank 2
11: Reserved
This selection can be set by both the Set Register Bank command and Control Register command.

• **RC1: Reset/Clock Control Register 1**

Bit	7	6	5	4	3	2	1	0
Name	PWRON	FSYCK_RDY	XCLK_RDY	XCLK_EN	FSYCK_DIV[1:0]		FSYCK_EN	RST_LL
R/W	R/W	R	R	R/W	R/W		R/W	R/W
POR	1	—	—	—	0	0	0	—
Reset	—	0	0	1	—	—	—	0

Bit 7 **PWRON**: 3.3V power on flag
This bit is only set to 1 by power on reset and not affected by software reset of strobe command. After being set high, this bit should be cleared by application program. The firmware can check this flag status and determine whether to execute auto calibration in the Light Sleep mode.

Bit 6 **FSYCK_RDY**: FSYCK clock ready flag (ready only)
0: Not ready
1: Ready
This bit is used to indicate that whether the FSYCK clock is ready for operation. This bit will be automatically cleared when FSYCK_EN=0, when power on reset occurs or when a Deep Sleep command or an Idle command is received.

Bit 5 **XCLK_RDY**: XCLK clock ready flag (ready only)
0: Not ready
1: Ready
This bit is used to indicate whether the XCLK debounce counter is full and XCLK is ready for operation. Note that when exiting the Deep Sleep state, this flag may need a certain period before being set high. This bit will be automatically cleared to zero when XCLK_EN=0, when RST_LL=1, when power on reset occurs or when a software reset command, a Deep Sleep command or an Idle command is received.

Bit 4 **XCLK_EN**: XCLK clock enable
0: Disable
1: Enable
Setting this bit high will enable the XCLK path to the baseband block while clearing this bit to zero can save power if required. The XCLK clock should be enabled when writing data to the FIFO.

Bit 3~2 **FSYCK_DIV[1:0]**: FSYCK clock (XCLK division) selection
00: 1/1 XCLK
01: 1/2 XCLK
10: 1/4 XCLK
11: 1/8 XCLK

Bit 1 **FSYCK_EN**: FSYCK clock enable
0: Disable
1: Enable

Bit 0 **RST_LL**: Low voltage (1.2V) logic reset control
0: Release reset
1: Reset

• IRQ1: Interrupt Control Register 1

Bit	7	6	5	4	3	2	1	0
Name	RXTO	RXFFOW	—	—	RXDETS[1:0]		IRQCPOR	IRQPOR
R/W	R	R	—	—	R/W		R/W	R/W
Reset	0	0	0	0	1	0	0	1

Bit 7 **RXTO**: RX time-out flag
0: RX time-out does not occur
1: RX time-out occurs
This flag will be set high by hardware when the RX time-out condition occurs and automatically cleared when a Light Sleep strobe command is received, when the device enters the RX continuous mode, when WOR/WOT wake up occurs or when the device enters the ARK TX/RX mode.

Bit 6 **RXFFOW**: RX FIFO overwrite flag
0: RX FIFO overwrite does not occur
1: RX FIFO overwrite occurs
This flag will be set high by hardware when the RX FIFO overwrite condition occurs and automatically cleared when a RX FIFO reset strobe command or a RX strobe command is received.

Bit 5~4 Reserved, must be “00”

Bit 3~2 **RXDETS[1:0]**: RX detect selection
00: Detect carry
01: Detect preamble
10/11: Detect SYNCWORD

Bit 1 **IRQCPOR**: IRQ flags clearing polarity selection
0: IRQ flags are cleared by writing 0 to the corresponding bits
1: IRQ flags are cleared by writing 1 to the corresponding bits

Bit 0 **IRQPOR**: IRQ signal polarity selection
0: Active low
1: Active high
When an IRQ flag in the IRQ3 register is set high and the corresponding IRQ function is enabled, the active level of the IRQ signal is determined by this configuration.

• **IRQ2: Interrupt Control Register 2**

Bit	7	6	5	4	3	2	1	0
Name	ARKTFIE	ATRCTIE	FIFOLTIE	RXERRIE	RXDETIE	CALCMPIE	RXCMPIE	TXCMPIE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 7 **ARKTFIE**: ARK TX Failure IRQ Enable
0: Disable
1: Enable

Bit 6 **ATRCTIE**: ATR Cycle Timer IRQ Enable
0: Disable
1: Enable

Bit 5 **FIFOLTIE**: FIFO Low Threshold IRQ Enable
0: Disable
1: Enable

Bit 4 **RXERRIE**: RX Error IRQ Enable
0: Disable
1: Enable

Bit 3 **RXDETIE**: RX Event Detected IRQ Enable
0: Disable
1: Enable

Bit 2 **CALCMPIE**: Calibration Complete IRQ Enable
0: Disable
1: Enable

Bit 1 **RXCMPIE**: RX Complete IRQ Enable
0: Disable
1: Enable

Bit 0 **TXCMPIE**: TX Complete IRQ Enable
0: Disable
1: Enable

• **IRQ3: Interrupt Control Register 3**

Bit	7	6	5	4	3	2	1	0
Name	ARKTFIF	ATRCTIF	FIFOLTIF	RXERRIF	RXDETIF	CALCMPIF	RXCMPIF	TXCMPIF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

When the individual flag within this register is set high by the hardware, the corresponding IRQ will be generated. These flags can be cleared by writing 0 or 1 to the corresponding flag which is determined by the IRQCPOR bit configuration.

Bit 7 **ARKTFIF**: ARK TX Failure IRQ Flag
0: No request
1: Interrupt request

Bit 6 **ATRCTIF**: ATR Cycle Timer IRQ Flag
0: No request
1: Interrupt request
This flag will be set high when the ATRCT timer is full.

Bit 5 **FIFOLTIF**: FIFO Low Threshold IRQ Flag
0: No request
1: Interrupt request
When in the Burst TX mode, if this flag is set high, it means that TX FIFO data length is less than FFMG setting threshold and there are TX data to be written into the FIFO. When in the Burst RX mode, if this flag is set high, it means that RX FIFO remaining space is less than FFMG setting threshold and the remaining RX data length is longer than FFMG setting threshold.

Bit 4 **RXERRIF**: RX Error IRQ Flag
0: No request
1: Interrupt request
The RX error conditions include RX failure, CRC failure (CRC_EN=1) or RX FIFO overwrite.

Bit 3 **RXDETIF**: RX Event Detected IRQ Flag
0: No request
1: Interrupt request
The RX events include carry, preamble and syncword and the actual trigger source is determined by the RXDETS[1:0] configuration.

Bit 2 **CALCMPIF**: Calibration Complete IRQ Flag
0: No request
1: Interrupt request
If ACAL_EN=0, the LIRC calibration is enabled by its individual calibration enable bit and the calibration completion will trigger IRQ. If ACAL_EN=1, VCO and RC calibrations are enabled and both completion will trigger IRQ.

Bit 1 **RXCMPIF**: RX Complete IRQ Flag
0: No request
1: Interrupt request
When the RX operation is completed without any error, this flag will be set high by hardware.

Bit 0 **TXCMPIF**: TX Complete IRQ Flag
0: No request
1: Interrupt request

• IO1: I/O Control Register 1

Bit	7	6	5	4	3	2	1	0
Name	PADDS[1:0]		GIO2S[2:0]			GIO1S[2:0]		
R/W	R/W		R/W			R/W		
POR	0	1	0	0	0	0	0	0

Bit 7~6 **PADDS[1:0]**: PAD driving strength selection (only reset by POR)
00: 0.5mA
01: 1mA
10: 5mA
11: 10mA

Bit 5~3 **GIO2S[2:0]**: GIO2 pin function selection (only reset by POR)
000/111: No function, input
001: SDO, 4-wire SPI data, output
010: TRXD, direct mode TXD/RXD, input/output
011: TXD, direct mode TXD, input
100: RXD, direct mode RXD, output
101: IRQ, interrupt request, output
110: ROSCi, ATR clock external input

Bit 2~0 **GIO1S[2:0]**: GIO1 pin function selection (only reset by POR)
000/111: No function, input
001: SDO, 4-wire SPI data, output
010: TRXD, direct mode TXD/RXD, input/output
011: TXD, direct mode TXD, intput
100: RXD, direct mode RXD, output
101: IRQ, interrupt request, output
110: ROSCi, ATR clock external input

• **IO2: I/O Control Register 2**

Bit	7	6	5	4	3	2	1	0
Name	GIO4S[3:0]				D3	D2	D1	D0
R/W	R/W				R/W			
POR	0	0	0	0	0	0	0	0

Bit 7~4 **GIO4S[3:0]**: GIO4 pin function selection (only reset by POR)
0000/0111/1111: No function, input
0001: SDO, 4-wire SPI data, output
0010: TRXD, direct mode TXD/RXD, input/output
0011: TXD, direct mode TXD, input
0100: RXD, direct mode RXD, output
0101: IRQ, interrupt request, output
0110: ROSCi, ATR clock external input
1000: TBCLK, TX bit (data) clock, output
1001: RBCLK, RX bit (recovery) clock, output
1010: FSYCK, i.e. XCLK 1/1, 1/2, 1/4, 1/8 output
1011: LIRCCLK, internal LIRC clock with debounce, output
1100: EPA_EN, external PA enable, output
1101: ELAN_EN, external LNA enable, output
1110: TRBCLK, TBCLK in TX mode or RBCLK in RX mode, output

Bit 3~0 **D3~D0**: Reserved, must be fixed at "0000", "0111" or "1111"

• **IO3: I/O Control Register 3**

Bit	7	6	5	4	3	2	1	0
Name	SDO_TEN	SPIPU	—	GIOPU[4:1]				—
R/W	R/W	R/W	—	R/W				—
POR	0	1	1	1	1	1	1	1

Bit 7 **SDO_TEN**: SDO tri-state enable (only reset by POR)
0: Disable
1: Enable

Bit 6 **SPIPU**: 3-wire SPI pull-up enable (only reset by POR)
0: Disable
1: Enable
When this bit is set high, it only controls the pull-up function for the CSN, SCK and SDIO pins. Note that the pull-up function of the SDO pin for the 4-wire SPI is configured using the GIOPU[4:1] bits.

Bit 5 Reserved, must be "1"

Bit 4~1 **GIOPU[4:1]**: GIO pin function pull-up enable control (only reset by POR)
These bits control the pull-high function of the GIO4~GIO1 pins respectively.

Bit 0 Reserved, must be "1"

• **FIFO1: FIFO Control Register 1**

Bit	7	6	5	4	3	2	1	0
Name	—	—	TXFFSA[5:0]					
R/W	—	—	R/W					
Reset	0	0	0	0	0	0	0	0

Bit 7~6 Reserved, must be "00"

Bit 5~0 **TXFFSA[5:0]**: TX FIFO start address, used for Block FIFO mode

• **FIFO2: FIFO Control Regsiter 2**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	RXPL2F_EN	FFINF_EN	FFMG_EN	FFMG[1:0]	
R/W	—	—	—	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	1

Bit 7~5 Reserved, must be "000"

Bit 4 **RXPL2F_EN**: RX payload length byte to FIFO enable
0: Disable
1: Enable
Setting this bit high will place the payload length byte in the packet to RX FIFO. In the RX continue mode (RXCON_EN=1), this bit should be set to 1 to support multiple payload in single RX FIFO.

Bit 3 **FFINF_EN**: FIFO infinite length mode enable
0: Disable
1: Enable

Bit 2 **FFMG_EN**: FIFO length margin detect enable
0: Disable
1: Enable

Bit 1~0 **FFMG[1:0]**: FIFO length margin selection
Threshold of remaining data in TX FIFO:
00: 4 bytes
01: 8 bytes
10: 16 bytes
11: 32 bytes
Threshold of remaining space in RX FIFO:
00: 4 bytes
01: 8 bytes
10: 16 bytes
11: 32 bytes
After the FIFO length margin detect function has been enabled by setting the FFMG_EN bit high and the required FIFO length margin has been selected by setting these bits, when the selected condition occurs the FIFOLTIF flag will be set high. In this case, an interrupt signal will also be generated if the corresponding interrrpt function has been enabled.

• **PKT1: Packet Control Register 1**

Bit	7	6	5	4	3	2	1	0
Name	TXPMLEN[7:0]							
R/W	R/W							
Reset	0	0	0	0	0	0	0	1

Bit 7~0 **TXPMLEN[7:0]**: TX preamble length
Transmit preamble length = (TXPMLEN[7:0]+1) bytes

• **PKT2: Packet Control Register 2**

Bit	7	6	5	4	3	2	1	0
Name	PID[1:0]		TRAILER_EN	WHTFMT	SYNCLEN[1:0]		RXPMLEN[1:0]	
R/W	R/W		R/W	R/W	R/W		R/W	
Reset	0	0	1	0	0	1	1	0

Bit 7~6 **PID[1:0]**: TX Packet ID
This ID will be placed in the highest two bits of the payload header field when the header option is enabled using the PLH_EN bit.

Bit 5 **TRAILER_EN**: Trailer field enable
0: Disable
1: Enable

Bit 4 **WHTFMT**: Whitening format selection
0: PN7
1: G(D)=D^7+D^4+1

Bit 3~2 **SYNCLEN[1:0]**: TX/RX mode SYNCWORD length selection
00: Reserved
01: 4 bytes
10: 6 bytes
11: 8 bytes

Bit 1~0 **RXPMLEN[1:0]**: RX preamble detection length selection
00: 0 byte – no preamble detection
01: 1 byte
10: 2 bytes
11: 4 bytes

- **PKT3: Packet Control Register 3**

Bit	7	6	5	4	3	2	1	0
Name	MCH_EN	FEC_EN	CRC_EN	CRCFMT	PLLEN_EN	PLHAC_EN	PLHLEN	PLH_EN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	1	0	0	0	0	0

Bit 7 **MCH_EN**: Manchester code enable
0: Disable
1: Enable

Bit 6 **FEC_EN**: FEC enable
0: Disable
1: Enable

Bit 5 **CRC_EN**: CRC field enable
0: Disable
1: Enable

Bit 4 **CRCFMT**: CRC format selection
0: CCITT-16-CRC G(X) = X^{16}+X^{12}+X^5+1
1: IBC-16-CRC G(X) = X^{16}+X^{15}+X^2+1

Bit 3 **PLLEN_EN**: Payload length field enable
0: Disable
1: Enable

Bit 2 **PLHAC_EN**: Payload header address correction enable control
0: Disable, PLHA[5:0] in the PKT8 register can be used as software flags defined by users.
1: Enable, PLHA[5:0] of TX/RX devices must include the same address, otherwise the packet will be regarded as a failed packet.
Note: If PLHLEN =1 and PLHAC_EN =1, the PLHEA[7:0] is still used as the address function.

Bit 1 **PLHLEN**: Payload header length
0: 1 byte
1: 2 bytes

Bit 0 **PLH_EN**: Payload header field enable
0: Disable
1: Enable

• **PKT4: Packet Control Register 4**

Bit	7	6	5	4	3	2	1	0
Name	WHT_EN	WHTSD[6:0]						
R/W	R/W	R/W						
Reset	0	0	1	1	0	1	1	0

Bit 7 **WHT_EN**: Data whitening enable
0: Disable
1: Enable

Bit 6~0 **WHTSD[6:0]**: Data whitening seed

• **PKT5: Packet Control Register 5**

Bit	7	6	5	4	3	2	1	0
Name	TXDLEN[7:0]							
R/W	R/W							
Reset	0	1	0	0	0	0	0	0

Bit 7~0 **TXDLEN[7:0]**: TX data length (unit: byte, used in burst mode only)

• **PKT6: Packet Control Register 6**

Bit	7	6	5	4	3	2	1	0
Name	RXDLEN[7:0]							
R/W	R/W							
Reset	0	1	0	0	0	0	0	0

Bit 7~0 **RXDLEN[7:0]**: RX data length (unit: byte; used in burst mode only)
When the PLLEN_EN bit is cleared to 0, the received data length is determined by this field.
When this register is read, the read value indicates the RX data length in FIFO. The default read value is 00h.

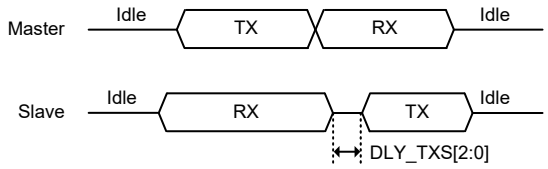
• **PKT7: Packet Control Register 7**

Bit	7	6	5	4	3	2	1	0
Name	RXPID[1:0]		DLY_RXS[2:0]			DLY_TXS[2:0]		
R/W	R		R/W			R/W		
Reset	0	0	1	0	0	0	0	0

Bit 7~6 **RXPID[1:0]**: Received packet PID (read only)

Bit 5~3 **DLY_RXS[2:0]**: RX block stable time after RX is enabled
000: 4μs
001: 8μs
010: 12μs
011: 16μs
100: 20μs
101: 32μs
110: 64μs
111: 100μs
These bits are used to select the waiting time between RX enable and RX stable. This time should be configured to a value greater than the default RX DCOC turbo mode delay time of 6μs.

Bit 2~0 **DLY_TXS[2:0]**: TX start (delay) time before entering the TX mode
000: 0μs
001: 10μs
010: 20μs
011: 40μs
100: 60μs
101: 80μs
110: 100μs
111: 120μs
It is used to align the timing between transmitter and receiver in ARK mode.



• PKT8: Packet Control Register 8

Bit	7	6	5	4	3	2	1	0
Name	—	—	PLHA[5:0]					
R/W	—	—	R/W					
Reset	0	0	0	0	0	0	0	0

Bit 7~6 Reserved, must be "00"

Bit 5~0 **PLHA[5:0]**: Payload header address to support broadcast
Address=0 in RX mode means not doing correction check.
Write: write data to TX PLHA[5:0]. Read: read data from RX PLHA[5:0].

• PKT9: Packet Control Register 9

Bit	7	6	5	4	3	2	1	0
Name	PLHEA[7:0]							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

Bit 7~0 **PLHEA[7:0]**: Payload header extended address to support broadcast
Address=0 in RX mode means not doing correction check.

• MOD1: Modulator Control Register 1

Bit	7	6	5	4	3	2	1	0
Name	DTR[7:0]							
R/W	R/W							
Reset	0	0	0	0	0	0	0	1

Bit 7~0 **DTR[7:0]**
DTR[8:0]: Data rate divider, DTR[8] is loaced in the MOD2 register.
Data Rate=f_{XTAL}/[(XODIV2+1)×32×(DTR[8:0]+1)], XODIV2=0, here data rate indicates TBCLK. Note that DTR[8:0] can only be an odd number.

• **MOD2: Modulator Control Register 2**

Bit	7	6	5	4	3	2	1	0
Name	RXIFOS[11:8]				DITHER[1:0]		—	DTR[8]
R/W	R/W				R/W		—	R/W
Reset	1	0	0	1	0	0	0	0

Bit 7~4 **RXIFOS[11:8]**

RXIFOS[11:0]: RX intermediate frequency offset, RXIFOS[7:0] is located in the MOD3 register.

Write to RXIFOS[11:8] first and then write to RXIFOS[7:0] to fully update RXIFOS[11:0].

RXIFOS[11:0]= floor{f_{IF}/[f_{XTAL}/(XODIV2+1)]×2^{17}}, XODIV2=0

Bit 3~2 **DITHER[1:0]**: Dither value

Bit 1 Reserved, must be "0"

Bit 0 **DTR[8]**

DTR[8:0]: Data rate divider, DTR[7:0] is located in the MOD1 register.

Data Rate=f_{XTAL}/[(XODIV2+1)×32×(DTR[8:0]+1)], XODIV2=0, here data rate indicates TBCLK. Note that DTR[8:0] can only be an odd number.

• **MOD3: Modulator Control Register 3**

Bit	7	6	5	4	3	2	1	0
Name	RXIFOS[7:0]							
R/W	R/W							
Reset	1	0	0	1	1	0	1	0

Bit 7~0 **RXIFOS[7:0]**

RXIFOS[11:0]: RX intermediate frequency offset, RXIFOS[11:8] is loaced in the MOD2 register.

Write to RXIFOS[11:8] first and then write to RXIFOS[7:0] to fully update RXIFOS[11:0].

RXIFOS[11:0]=floor{f_{IF}/[f_{XTAL}/(XODIV2+1)]×2^{17}}, XODIV2=0

• **DM1: Demodulator Control Register 1**

Bit	7	6	5	4	3	2	1	0
Name	—	—	MDIV[5:0]					
R/W	—	—	R/W					
Reset	0	0	0	0	0	0	1	1

Bit 7~6 Reserved, must be "00"

Bit 5~0 **MDIV[5:0]**: Demodulator operation clock divider

DMCLK=ADCLK/(MDIV[5:0]+1)

• **DM2: Demodulator Control Register 2**

Bit	7	6	5	4	3	2	1	0
Name	PREAMBLE_CFO_EN1	PREAMBLE_CFO_EN0	SDR[5:0]					
R/W	R/W	R/W	R/W					
Reset	0	1	0	0	0	0	0	0

Bit 7 **PREAMBLE_CFO_EN1**: Enable 2nd stage CFO correction in preamble
0: Disable
1: Enable
Note that this bit can only be set if the preamble length is 4 bytes, i.e, RXPMLEN[1:0]=11b.

Bit 6 **PREAMBLE_CFO_EN0**: Enable 1st stage CFO correction in preamble
0: Disable
1: Enable

Bit 5~0 **SDR[5:0]**: Decimator operation clock after phase extract
SDR[5:0]+1=DMCLK/(8×DATA_RATE), here DATA_RATE indicates RBCLK.

• **DM3: Demodulator Control Register 3**

Bit	7	6	5	4	3	2	1	0
Name	CSF_SW_EN	FD_MOD[6:0]						
R/W	R/W	R/W						
Reset	1	1	1	0	0	0	0	0

Bit 7 **CSF_SW_EN**: Channel selection filter auto bandwidth switch enable
0: Disable
1: Enable

Bit 6~0 **FD_MOD[6:0]**: Frequency deviation modifier
FD_MOD=Round((h/(SDR[5:0]+1))×128); h=modulation index
SDR[5:0]+1=DMCLK/(8×DATA_RATE)

• **DM4: Demodulator Control Register 4**

Bit	7	6	5	4	3	2	1	0
Name	THOLD[3:0]				CFO_DSEL	—	PH_DIFF_MOD	PRE_CSF_EN
R/W	R/W				R/W	—	R/W	R/W
Reset	0	0	0	1	1	0	0	0

Bit 7~4 **THOLD[3:0]**: Detection errors threshold
THOLD[3:2]: Preamble detection errors bit number threshold
THOLD[1:0]: SYNCWORD detection errors bit number

Bit 3 **CFO_DSEL**: CFO correction domain selection
0: Analog domain
1: Digital domain

Bit 2 Reserved, must be "0"

Bit 1 **PH_DIFF_MOD**: Phase difference extract mode setting
0: Phase extract range [-pi/2, pi/2]
1: Phase extract range [-pi, pi]

Bit 0 **PRE_CSF_EN**: Switch receiver filter bandwidth when preamble is matched
0: Disable
1: Enable

• **DM5: Demodulator Control Register 5**

Bit	7	6	5	4	3	2	1	0
Name	FD_HOLD[7:0]							
R/W	R/W							
Reset	0	0	1	1	0	0	0	0

Bit 7~0 **FD_HOLD[7:0]**: Frequency deviation threshold for preamble detection

• **DM8: Demodulator Control Register 8**

Bit	7	6	5	4	3	2	1	0
Name	M_RATIO[7:0]							
R/W	R/W							
Reset	0	1	0	0	0	0	0	0

Bit 7~0 **M_RATIO[7:0]**: For CFO calulation
M_RATIO=round(1/(MDIV[5:0]+1)×2^8)

Bank 0 Control Registers

All control registers will be set to their initial value by power-on reset (POR). The software reset will set the control registers to their initial value except the LIRC_EN, LIRC_OP[4:0], LIRC_OW and LIRCCAL_EN bits in the XO3 register. These bits keep unchanged after software reset.

Addr.	Name	Bit							
		7	6	5	4	3	2	1	0
20h	OM	PWR_SOFT	BAND_SEL[1:0]		—	ACAL_EN	RTX_EN	RTX_SEL	SX_EN
22h	SX1	—	D_N[6:0]						
23h	SX2	D_K[7:0]							
24h	SX3	D_K[15:8]							
25h	SX4	—	—	—	—	D_K[19:16]			
26h	STA1	—	—	—	CD_FLAG	—	OMST[2:0]		
28h	RSSI2	—				RSSI_CTHD[3:0]			
29h	RSSI3	RSSI_NEGDB[7:0]							
2Ah	RSSI4	RSSI_SYNC_OK[7:0]							
2Bh	ATR1	ATRCLK_DIV[1:0]		ATRCLKS	ATRTU	ATRCTM	ATRM[1:0]		ATR_EN
2Ch	ATR2	ATRCYC[7:0]							
2Dh	ATR3	ATRCYC[15:8]							
2Eh	ATR4	ATRRXAP[7:0]							
2Fh	ATR5	ATRRXEP[7:0]							
30h	ATR6	ATRRXEP[15:8]							
31h	ATR7	ARKNM[3:0]				—	ATR_WDLY[1:0]		ARK_EN
32h	ATR8	ARKRXAP[7:0]							
33h	ATR9	ATRCT[7:0]							
34h	ATR10	ATRCT[15:8]							
35h	ATR11	—					ATRRXAP[10:8]		
3Ch	XO1	XSHIFT[1:0]		—	XO_TRIM[4:0]				
3Dh	XO2	—	—	—	—	XODIV2	—		
3Eh	XO3	LIRCCAL_EN	LIRC_OW	LIRC_OP[4:0]					LIRC_EN
3Fh	TX2	—				CT_PAD[3:0]			

Note: Addresses 21h, 27h and 36h~3Bh which are not listed in this table are reserved for future use, it is suggested not to change their initial values by any methods.

The reset value shown in the following register description tables means the software reset results of strobe command.

• **OM: Operation Mode Control Register**

Bit	7	6	5	4	3	2	1	0
Name	PWR_SOFT	BAND_SEL[1:0]		—	ACAL_EN	RTX_EN	RTX_SEL	SX_EN
R/W	R/W	R/W		—	R/W	R/W	R/W	R/W
Reset	0	0	1	0	0	0	0	0

Bit 7 **PWR_SOFT**: RF operation mode selection
0: RF normal operation mode
1: RF engineering mode

Bit 6~5 **BAND_SEL[1:0]**: Band selection (when PWR_SOFT=0)
00: 315MHz band
01: 433MHz band
10: 470~510MHz band
11: 868/915MHz band

Bit 4 Reserved, must be "0"

Bit 3 **ACAL_EN**: Auto calibration enable
0: Disable
1: Enable
When this bit is set high, both the VCO and RC calibrations will be enabled. When the VCO and RC calibrations are completed, this bit will be cleared to zero by hardware.

Bit 2 **RTX_EN**: RX or TX mode enable
0: Disable
1: Enable
After the RX or TX mode has been selected by the RTX_SEL bit, setting this bit high will enable the selected mode.

Bit 1 **RTX_SEL**: RX or TX mode selection
0: RX mode
1: TX mode

Bit 0 **SX_EN**: Synthesizer enable (standby mode enable control)
0: Disable
1: Enable
Setting this bit high will enable the PFD, CP and VCO functions.

• **SX1: Fractional-N Synthesizer Control Register 1**

Bit	7	6	5	4	3	2	1	0
Name	—	D_N[6:0]						
R/W	—	R/W						
Reset	0	0	0	1	1	0	1	1

Bit 7 Reserved, must be "0"

Bit 6~0 **D_N[6:0]**: RF channel integer number code
D_N[6:0]=floor{f_{RF}/[f_{XTAL}/(XODIV2+1)]}
For example, XO=16MHz and RF band=433.92MHz which are initial setup:
→ 433.92MHz/16MHz=27.12
→ D_N=27
→ Dec2Hex(27)=1B
→ Dec2Bin(27)=001_1011

• **SX2: Fractional-N Synthesizer Control Register 2**

Bit	7	6	5	4	3	2	1	0
Name	D_K[7:0]							
R/W	R/W							
Reset	1	0	0	0	0	1	0	1

Bit 7~0 **D_K[7:0]**: RF channel fractional number code lowest byte

• **SX3: Fractional-N Synthesizer Control Register 3**

Bit	7	6	5	4	3	2	1	0
Name	D_K[15:8]							
R/W	R/W							
Reset	1	1	1	0	1	0	1	1

Bit 7~0 **D_K[15:8]**: RF channel fractional number code medium byte

• **SX4: Fractional-N Synthesizer Control Register 4**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	D_K[19:16]			
R/W	—	—	—	—	R/W			
Reset	0	0	0	0	0	0	0	1

Bit 7~4 Reserved, must be "0000"

Bit 3~0 **D_K[19:16]**: RF channel fractional number code highest byte

$D\_K[19:0]=floor\{(f_{RF}/[f_{XTAL}/(XODIV2+1)]-D\_N[6:0])\times 2^{20}\}$

For example, XO=16MHz and RF band=433.92MHz which are initial setup:

→ 433.92MHz/16MHz=27.12

→ $D\_K=0.12\times 2^{20}=125829$

→ Dec2Hex(125829)=1EB85

→ Dec2Bin(125829)=0001_1110_1011_1000_0101

• **STA1: Status Control Register 1**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	CD_FLAG	—	OMST[2:0]		
R/W	—	—	—	R	—	R		
Reset	0	0	0	0	0	0	0	0

Bit 7~5 Reserved, must be "000"

Bit 4 **CD_FLAG**: Carrier detection flag (read only)

This flag will be set high by hardware when carrier detection is okey after pulling DEMOD_EN high. Here DEMOD_EN high level is an internal signal which is generated by the internal state machine when in the Direct mode (DIR_EN=1) or after the RX strobe command is received when in the Burst mode (DIR_EN=0). The flag will be automatically cleared when RX_EN rising edge occurs. Here RX_EN rising edge is generated after setting RTX_SEL=0 and RTX_EN=1 when in the Direct mode or by the internal state machine after the RX strobe command is received when in the Burst mode.

Bit 3 Reserved, must be "0"

Bit 2~0 **OMST[2:0]**: Operation mode state indication (read only)

000: Deep Sleep mode
001: Idle mode
010: Light Sleep mode
011: Standby mode
100: TX mode
101: RX mode
110: VCO calibration mode
111: Undefined

• **RSSI2: RSSI Control Register 2**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	RSSI_CTHD[3:0]			
R/W	—	—	—	—	R/W			
Reset	0	0	0	0	1	0	1	0

Bit 7~4 Reserved, must be "0000"

Bit 3~0 **RSSI_CTHD[3:0]**: RSSI threshold for carrier detection
(RSSI_CTHD[3:0]×2+1)+74=RSSI threshold for carrier detection

• **RSSI3: RSSI Control Register 3**

Bit	7	6	5	4	3	2	1	0
Name	RSSI_NEGDB[7:0]							
R/W	R							
Reset	0	0	0	0	0	0	0	0

Bit 7~0 **RSSI_NEGDB[7:0]**: RSSI value (unit: -dB)
It is a real time measurement value.

• **RSSI4: RSSI Control Register 4**

Bit	7	6	5	4	3	2	1	0
Name	RSSI_SYNC_OK[7:0]							
R/W	R							
Reset	0	0	0	0	0	0	0	0

Bit 7~0 **RSSI_SYNC_OK[7:0]**: RSSI snapshot when SYNCWORD is detected correct

• **ATR1: Auto TX/RX Control Register 1**

Bit	7	6	5	4	3	2	1	0
Name	ATRCLK_DIV[1:0]		ATRCLKS	ATRTU	ATRCTM	ATRM[1:0]		ATR_EN
R/W	R/W		R/W	R/W	R/W	R/W		R/W
Reset	1	1	0	0	1	0	0	0

Bit 7~6 **ATRCLK_DIV[1:0]**: ATR clock frequency selection
00: 1/1, ATRCLK=32768Hz
01: 1/4, ATRCLK=8192Hz
10: 1/8, ATRCLK=4096Hz
11: 1/16, ATRCLK=2048Hz

Bit 5 **ATRCLKS**: ATRCLK clock source selection
0: From the internal LIRC clock
1: From the exteral ROSCi clock input on the GIOn pin

Bit 4 **ATRTU**: Auto TRX unit time selection
0: 250μs
1: 1ms, used to support low data rate applications
This bit is used to select the unit time for the ATR RX active period (ATRRXAP[10:0]), ATR RX extended period (ATRRXEP[15:0]) and ARK RX active period (ARKRXAP[7:0]).

Bit 3 **ATRCTM**: Auto TRX timer mode selection
0: Single mode, restart ATRCT timer when every ATR transaction occurs.
1: Continuous mode, start ATRCT timer upon receiving Idle command; stop ATRCT timer when ATR_EN=0 or ATRCTM=0 and exit the ATR active period.

Bit 2~1 **ATRM[1:0]**: Auto TRX mode selection
00: ATR WOT mode
01: ATR WOR mode
10/11: ATR WTM mode

Bit 0 **ATR_EN**: Auto TRX enable
0: Disable
1: Enable
Note that the ATR functions are activated by operation state transition from Deep Sleep/Light Sleep mode to Idle mode.

• **ATR2: Auto TX/RX Control Register 2**

Bit	7	6	5	4	3	2	1	0
Name	ATRCYC[7:0]							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

Bit 7~0 **ATRCYC[7:0]**: ATRCT timer expire value low byte

• **ATR3: Auto TX/RX Control Register 3**

Bit	7	6	5	4	3	2	1	0
Name	ATRCYC[15:8]							
R/W	R/W							
Reset	0	0	0	0	1	1	1	1

Bit 7~0 **ATRCYC[15:8]**: ATRCT timer expire value high byte
Wake up period=ATRCLK×(ATRCYC[15:0]+1). Default period is 1s.

• **ATR4: Auto TX/RX Control Register 4**

Bit	7	6	5	4	3	2	1	0
Name	ATRRXAP[7:0]							
R/W	R/W							
Reset	0	0	1	0	0	1	1	1

Bit 7~0 **ATRRXAP[7:0]**: ATR RX active period low byte
ATR RX active period high byte ATRRXAP[10:8] is located in the ATR11 register.
Active period=unit time×(ATRRXAP[10:0]+1); the unit time can be 250μs or 1ms which is determined by the ATRTU bit. The default ATR RX active period is 10ms with a default time unit of 250μs.

• **ATR5: Auto TX/RX Control Register 5**

Bit	7	6	5	4	3	2	1	0
Name	ATRRXEP[7:0]							
R/W	R/W							
Reset	1	0	0	0	1	1	1	1

Bit 7~0 **ATRRXEP[7:0]**: ATR RX extend period low byte

• **ATR6: Auto TX/RX Control Register 6**

Bit	7	6	5	4	3	2	1	0
Name	ATRRXEP[15:8]							
R/W	R/W							
Reset	0	0	0	0	0	0	0	1

Bit 7~0 **ATRRXEP[15:8]**: ATR RX extended period high byte
Extend period=unit time×(ATRRXEP[15:0]+1); the unit time can be 250μs or 1ms which is determined by the ATRTU bit. The default ATR RX extended period is 100ms with a default time unit of 250μs.

• **ATR7: Auto TX/RX Control Register 7**

Bit	7	6	5	4	3	2	1	0
Name	ARKNM[3:0]				—	ATR_WDLY[1:0]		ARK_EN
R/W	R/W				—	R/W		R/W
Reset	0	1	1	1	0	0	1	0

Bit 7~4 **ARKNM[3:0]**: ARK repeat cycle number
Maximum repeat cycle number=ARKNM[3:0]+1

Bit 3 Reserved, must be "0"

Bit 2~1 **ATR_WDLY[1:0]**: Auto wake up delay time
00: 244μs
01: 488μs
10: 732μs
11: 976μs

Bit 0 **ARK_EN**: Auto-Resend/ACK enable
0: Disable
1: Enable

• **ATR8: Auto TX/RX Control Register 8**

Bit	7	6	5	4	3	2	1	0
Name	ARKRXAP[7:0]							
R/W	R/W							
Reset	0	0	1	0	0	1	1	1

Bit 7~0 **ARKRXAP[7:0]**: ARK RX active period
Active period=unit time×(ARKRXAP[7:0]+1); the unit time can be 250μs or 1ms which is determined by the ATRTU bit. The default ARK RX active period is 10ms with a default time unit of 250μs.

• **ATR9: Auto TX/RX Control Register 9**

Bit	7	6	5	4	3	2	1	0
Name	ATRCT[7:0]							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

Bit 7~0 **ATRCT[7:0]**: ATR cycle timer low byte

• **ATR10: Auto TX/RX Control Register 10**

Bit	7	6	5	4	3	2	1	0
Name	ATRCT[15:8]							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

Bit 7~0 **ATRCT[15:8]**: ATR cycle timer high byte
Reading ATRCT[15:0] will get the current count value. Due to the limitation of SPI 8-bit data length, reading the ATR9 register will take a snapshot of the whole 16-bit data into the read register buffer. Users should read ATR9 and ATR10 continuously (non-interrupted) to get correct data.
Writing to ATRCT[15:0] will update the count value. Write to ATR9 first and then write to ATR10 to trigger the ATRCT write function. This timer update mechanism is used to align the time slot for the master and slave in a two-way RF system.

• **ATR11: Auto TX/RX Control Register 11**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	ATRRXAP[10:8]		
R/W	—	—	—	—	—	R/W		
Reset	0	0	0	0	0	0	0	0

Bit 7~3 Reserved, must be "00000"

Bit 2~0 **ATRRXAP[10:8]**: ATR RX active period high byte

ATR RX active period low byte ATRRXAP[7:0] is located in the ATR4 register.

Active period=unit time×(ATRRXAP[10:0]+1); the unit time can be 250μs or 1ms which is determined by the ATRTU bit. The default ATR RX active period is 10ms with a default time unit of 250μs.

• **XO1: XO Control Register 1**

Bit	7	6	5	4	3	2	1	0
Name	XSHIFT[1:0]		—	XO_TRIM[4:0]				
R/W	R/W		—	R/W				
Reset	0	0	0	1	0	0	0	0

Bit 7~6 **XSHIFT[1**:0]: Capacitor load coarse shift

Bit 5 Reserved, must be "0"

Bit 4~0 **XO_TRIM[4**:0]: Trim value for the internal capacitor load for the crystal

Default setting=2.4pF, step=0.15pF. The larger the trim value is the larger the capacitor load will be, vice versa.

Note: The recommended values for the XO1 register are listed below:

C_{LOAD}	12pF	16pF	20pF
XO1	1Eh	50h	92h

• **XO2: XO Control Register 2**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	XODIV2	—	—	—
R/W	—	—	—	—	R/W	—	—	—
Reset	0	0	0	0	0	0	1	1

Bit 7~4 Reserved, must be "0000"

Bit 3 **XODIV2**: XO output divided by 2 enable

0: Disable

1: Enable

Note: f_{XTAL}=16MHz, XODIV2 must be "0".

Bit 2~0 Reserved, must be "011"

• **XO3: XO Control 5Register 3**

Bit	7	6	5	4	3	2	1	0
Name	LIRCCAL_EN	LIRC_OW	LIRC_OP[4:0]					LIRC_EN
R/W	R/W	R/W	R/W					R/W
POR	0	0	0	1	1	0	1	0

Bit 7 **LIRCCAL_EN**: LIRC calibration enable

0: Disable

1: Enable

Bit 6 **LIRC_OW**: LIRC overwrite control

0: LIRC_OP[4:0] from calibration engine

1: LIRC_OP[4:0] from control register

Bit 5~1 **LIRC_OP[4:0]**: LIRC trim
After writing data to LIRC_OP[4:0], this trim will become active when the LIRC_OW bit is set high. When reading data from LIRC_OP[4:0], the actual data source is determined by the LIRC_OW bit setting.

Bit 0 **LIRC_EN**: LIRC enable
0: Disable
1: Enable

- **TX2: TX Control Register 2**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	CT_PAD[3:0]			
R/W	—	—	—	—	R/W			
Reset	0	0	0	0	1	0	0	0

Bit 7~4 Reserved, must be "0000"

Bit 3~0 **CT_PAD[3:0]**: TX PAD linear power control

Bank 1 Control Registers

All control registers will be set to their initial value by power-on reset (POR). The software reset will set the control registers to their initial value.

Addr.	Name	Bit							
		7	6	5	4	3	2	1	0
21h	AGC2	SAT_SEL[1:0]		—				AGC_CMP_THD[1:0]	
22h	AGC3	CDRST_THD_SEL[1:0]		ENVAVG_SEL[1:0]		—	IF_DETOK_THD[2:0]		
23h	AGC4	GAIN_SEL[3:0]				—	AGC_ST[2:0]		
24h	AGC5	—						AGC_FSEL[1:0]	
26h	AGC7	GAIN_STB[7:0]							
2Ch	FCF1	—	—	SFRATIO[1:0]		—			
2Dh	FCF2	FSCALE[7:0]							
2Eh	FCF3	—				FSCALE[11:8]			
2Fh	FCF4	CF_B12[7:0]							
30h	FCF5	—						CF_B12[9:8]	
31h	FCF6	CF_B13[7:0]							
32h	FCF7	—						CF_B13[9:8]	
33h	FCF8	CF_A12[7:0]							
34h	FCF9	—						CF_A12[9:8]	
35h	FCF10	CF_A13[7:0]							
36h	FCF11	—						CF_A13[9:8]	
37h	FCF12	CF_B22[7:0]							
38h	FCF13	—						CF_B22[9:8]	
39h	FCF14	CF_B23[7:0]							
3Ah	FCF15	—						CF_B23[9:8]	
3Bh	FCF16	CF_A22[7:0]							
3Ch	FCF17	—						CF_A22[9:8]	
3Dh	FCF18	CF_A23[7:0]							
3Eh	FCF19	—						CF_A23[9:8]	

Note: Addresses 20h, 25h, 27h~2Bh and 3Fh which are not listed in this table are reserved for future use, it is suggested not to change their initial values by any methods.

The reset value shown in the following register description tables means the software reset results of strobe command.

• **AGC2: AGC Control Register 2**

Bit	7	6	5	4	3	2	1	0
Name	SAT_SEL[1:0]		—	—	—	—	AGC_CMP_THD[1:0]	
R/W	R/W		—	—	—	—	R/W	
Reset	0	1	0	0	0	0	0	0

Bit 7~6 **SAT_SEL[1:0]**: Saturation detection threshold selection
00: -6 dBFS
01: -8 dBFS
10: -10 dBFS
11: -12 dBFS
Note: "FS" indicates the ADC output full-scale.

Bit 5~2 Reserved, must be "0000"

Bit 1~0 **AGC_CMP_THD[1:0]**: AGC comparison number threshold
00: Continuous AGC comparison until SYNCWORD is detected
01~11: Comparison number threshold

• **AGC3: AGC Control Register 3**

Bit	7	6	5	4	3	2	1	0
Name	CDRST_THD_SEL[1:0]		ENVAVG_SEL[1:0]		—	IF_DETOK_THD[2:0]		
R/W	R/W		R/W		—	R/W		
Reset	0	0	1	0	0	1	0	0

Bit 7~6 **CDRST_THD_SEL[1:0]**: Carrier signal threshold to reset AGC

CDRST_THD_SEL[1:0]	GAIN_SEL[3:0]		
	0010b	0011b	Other Values
00b	-32 dBFS	-41 dBFS	-48 dBFS
01b	-35 dBFS	-44 dBFS	-48 dBFS
10b	-38 dBFS	-47 dBFS	-48 dBFS
11b	-41 dBFS	-48 dBFS	-48 dBFS

If the AGC completion state is reached and the signal strength detected is below the preset threshold, the AGC flow will be reset and then restarted.

Bit 5~4 **ENVAVG_SEL[1:0]**: Envelop detection average ratio selection
00: 1/16
01: 1/32
10: 1/64
11: 1/128

Bit 3 Reserved, must be "0"

Bit 2~0 **IF_DETOK_THD[2:0]**: IF detection OK threshold
After the gain stable time which determined by the AGC7 register, the AGC circuit will wait for (IF_DETOK_THD[2:0]×8) ADCLK cycles before starting to detect the IF signal strength.

• **AGC4: AGC Control Register 4**

Bit	7	6	5	4	3	2	1	0
Name	GAIN_SEL[3:0]				—	AGC_ST[2:0]		
R/W	R				—	R		
Reset	0	0	0	1	0	0	0	1

Bit 7~4 **GAIN_SEL[3:0]**: Gain curve selection
0000: Gain curve is not selected
0001: Maximum gain is selected
0111: Minimum gain is selected
The available field value is from 0000 to 0111. The gain will automatically be selected by the hardware. This bit field and the CDRST_THD_SEL[1:0] bit field together determine the carrier signal strength threshold to reset AGC, refer to the AGC3 register.

Bit 3 Reserved, must be "0"

Bit 2~0 **AGC_ST[2:0]**: AGC state machine state
000~001: AGC is not completed
111: AGC is completed

• **AGC5: AGC Control Register 5**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	AGC_FSEL[1:0]	
R/W	—	—	—	—	—	—	R/W	
Reset	0	0	0	0	0	0	0	0

Bit 7~2 Reserved, must be "000000"

Bit 1~0 **AGC_FSEL[1:0]**: AGC filter configuration
AGC_FSEL[1]: HPF pass band set point
0: 5/32 ADCLK
1: 6/32 ADCLK
AGC_FSEL[0]: LPF pass band set point
0: 17/320 ADCLK
1: 17/256 ADCLK
ADCLK=0.5×XCLK. It is recommended to set AGC_FSEL[1:0]=00b for XCLK=16MHz.

• **AGC7: AGC Control Register 7**

Bit	7	6	5	4	3	2	1	0
Name	GAIN_STB[7:0]							
R/W	R/W							
Reset	0	0	1	1	0	0	0	0

Bit 7~0 **GAIN_STB[7:0]**: Gain stable count
Gain stable count delay in ADCLK period=GAIN_STB[7:0]×2

• **FCF1: Filter Coefficient Control Register 1**

Bit	7	6	5	4	3	2	1	0
Name	—	—	SFRATIO[1:0]		—	—	—	—
R/W	—	—	R/W		—	—	—	—
Reset	0	0	0	0	0	1	1	0

Bit 7~6 Reserved, must be "00"

Bit 5~4 **SFRATIO[1:0]**: Smooth filter ratio selection
00: 1/1
01: 1/16
10: 1/64
11: 1/128

Bit 3~0 Reserved, must be "0110"

• **FCF2: Filter Coefficient Control Register 2**

Bit	7	6	5	4	3	2	1	0
Name	FSCALE[7:0]							
R/W	R/W							
Reset	0	1	0	0	0	1	0	0

Bit 7~0 **FSCALE[7:0]**: Frequency deviation scale parameter low byte

• **FCF3: Filter Coefficient Control Register 3**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	FSCALE[11:8]			
R/W	—	—	—	—	R/W			
Reset	0	0	0	0	0	1	0	0

Bit 7~4 Reserved, must be "0000"

Bit 3~0 **FSCALE[11:8]**: Frequency deviation scale parameter high byte
If the data rate is in the range of 250Kbps~100Kbps, the pre-filter is required.
FSCALE[11:0]=round{h×f_S/[f_{XTAL}/(XODIV2+1)]×2^{15}}
where h=(2×frequency deviation)/(data rate).
Here "h" is the modulation index calculated from frequency deviation and data rate.

• **FCF4: Filter Coefficient Control Register 4**

Bit	7	6	5	4	3	2	1	0
Name	CF_B12[7:0]							
R/W	R/W							
Reset	1	0	0	0	0	1	0	1

• **FCF5: Filter Coefficient Control Register 5**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CF_B12[9:8]	
R/W	—	—	—	—	—	—	R/W	
Reset	0	0	0	0	0	0	1	0

• **FCF6: Filter Coefficient Control Register 6**

Bit	7	6	5	4	3	2	1	0
Name	CF_B13[7:0]							
R/W	R/W							
Reset	1	0	0	0	1	0	1	0

• **FCF7: Filter Coefficient Control Register 7**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CF_B13[9:8]	
R/W	—	—	—	—	—	—	R/W	
Reset	0	0	0	0	0	0	0	0

• **FCF8: Filter Coefficient Control Register 8**

Bit	7	6	5	4	3	2	1	0
Name	CF_A12[7:0]							
R/W	R/W							
Reset	0	0	0	1	0	0	1	0

• **FCF9: Filter Coefficient Control Register 9**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CF_A12[9:8]	
R/W	—	—	—	—	—	—	R/W	
Reset	0	0	0	0	0	0	0	0

When the data rate is in the range of 49Kbps~2Kbps, the following smooth filter is needed.

$CF\_A12[9:0]=mod(2^{10}+[(SFRATIO[1:0]-1)\times 2^{8}], 2^{10})$

• **FCF10: Filter Coefficient Control Register 10**

Bit	7	6	5	4	3	2	1	0
Name	CF_A13[7:0]							
R/W	R/W							
Reset	0	0	1	0	1	0	1	1

• **FCF11: Filter Coefficient Control Register 11**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CF_A13[9:8]	
R/W	—	—	—	—	—	—	R/W	
Reset	0	0	0	0	0	0	1	1

• **FCF12: Filter Coefficient Control Register 12**

Bit	7	6	5	4	3	2	1	0
Name	CF_B22[7:0]							
R/W	R/W							
Reset	0	0	0	1	0	1	0	0

• **FCF13: Filter Coefficient Control Register 13**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CF_B22[9:8]	
R/W	—	—	—	—	—	—	R/W	
Reset	0	0	0	0	0	0	0	1

• **FCF14: Filter Coefficient Control Register 14**

Bit	7	6	5	4	3	2	1	0
Name	CF_B23[7:0]							
R/W	R/W							
Reset	0	0	1	0	0	0	0	1

• **FCF15: Filter Coefficient Control Register 15**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CF_B23[9:8]	
R/W	—	—	—	—	—	—	R/W	
Reset	0	0	0	0	0	0	0	0

• **FCF16: Filter Coefficient Control Register 16**

Bit	7	6	5	4	3	2	1	0
Name	CF_A22[7:0]							
R/W	R/W							
Reset	0	1	1	1	1	0	0	0

• **FCF17: Filter Coefficient Control Register 17**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CF_A22[9:8]	
R/W	—	—	—	—	—	—	R/W	
Reset	0	0	0	0	0	0	0	0

• **FCF18: Filter Coefficient Control Register 18**

Bit	7	6	5	4	3	2	1	0
Name	CF_A23[7:0]							
R/W	R/W							
Reset	0	0	1	0	1	0	0	0

• **FCF19: Filter Coefficient Control Register 19**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CF_A23[9:8]	
R/W	—	—	—	—	—	—	R/W	
Reset	0	0	0	0	0	0	0	0

The FCF4~FCF19 regsiters define eight groups of IIR coefficients, their recommended settings for different XTAL clock conditions are listed below.

f_{XTAL}	16MHz	16MHz	16MHz	16MHz	16MHz
f_S	250Kbps	125Kbps	50Kbps	10Kbps	2Kbps
f_D	93.75kHz	46.875kHz	18.75kHz	40kHz	8kHz
D_K[19:0] (H)	$f_{RF}/[f_{XTAL}/(XODIV2+1)]$, take decimal number				
D_N[6:0] (H)	$f_{RF}/[f_{XTAL}/(XODIV2+1)]$, take integer number				
SFRATIO[1:0] (D)	0	0	0	1	3
FSCALE[11:0] (H)	444	119	4C	A4	20
CF_B12[9:0] (H)	285	01D	0	0	0
CF_B13[9:0] (H)	8A	346	0	0	0
CF_A12[9:0] (H)	12	022	0	310	302
CF_A13[9:0] (H)	32B	331	0	0	0
CF_B22[9:0] (H)	114	386	0	0	0
CF_B23[9:0] (H)	21	012	0	0	0
CF_A22[9:0] (H)	78	008	0	0	0
CF_A23[9:0] (H)	28	008	0	0	0

Bank 2 Control Registers

All control registers will be set to their initial value by power-on reset (POR). The software reset will set the control registers to their initial value.

Addr.	Name	Bit							
		7	6	5	4	3	2	1	0
26h	RSV1	Reserved							
27h	RSV2	Reserved							
28h	RSV3	Reserved							
29h	RSV4	Reserved							
2Dh	RSV5	Reserved							
2Eh	RSV6	Reserved							
2Fh	RVS7	Reserved							
30h	RSV8	Reserved							
31h	RSV9	Reserved							
34h	RSV10	Reserved							
3Ah	RSV11	Reserved							

Note: The addresses which are not listed in this table are reserved for future use, it is suggested not to change their initial values by any methods.

The recommended values for the Bank 2 registers are listed below:

Addr.	Name	Frequency Band	
		433MHz	868MHz
26h	RSV1	02h/C2h (125Kbps/250Kbps: C2h)	
27h	RSV2	66h/33h (125Kbps/250Kbps: 33h)	
28h	RSV3	AAh	
29h	RSV4	00h	
2Dh	RSV5	16h	
2Eh	RSV6	64h	74h
2Fh	RSV7	44h/54h (≥100Kbps: 54h)	
30h	RSV8	00h	
31h	RSV9	64h	
34h	RSV10	BCh	9Ch
3Ah	RSV11	94h	

Special Function Description

Sub-1GHz RF Transceiver

The BA45F5640 adopts a fully-integrated, low-IF receiver architrecture. The received RF signal is first amplified by a low noise amplifier (LNA), after which the frequency is down-converted to an intermediate frequency (IF) by a quadrature mixer. The mixer output is filtered by a channel-selected filter which rejects the unwanted out-of-band (OOB) interference and image signals. After filtering, the IF signal is amplified by a analog programmable gain amplifier (PGA). Then the IF signal is digitized by a 10-bit ΣΔ ADC.

The RF features an Automatic Gain Control (AGC) unit to adjust the receiver gain according to the RSSI, generated at the digital modem. The AGC enables the RF to operate from sensitivity level to +10dBm input power.

The BA45F5640 adopts a fully integrated fractional-N synthesizer which includes RF VCO, loop filter, digital controlled XO (DCXO) and integrated load capacitors for XO. Placing VCO load inductor on the PCB is to lower VCO resonant frequency to achieve 4.2mA RX mode current consumption. The fractional-N synthesizer architecture allows the users to extend their potential usage to a wider frequency range.

The transmit session is a VCO direct modulation architecture. Different from the conventional direct up-conversion transmitters, the GFSK modulation signal is fed into the VCO directly to take advantange of fractional-N synthesizer. As a result, both layout area and current consumption are much smaller compared with direct up-conversion transmitters. The fine resolution can generate a low FSK error GFSK signal. The modulated signal is fed into a Class-E Power Amplifier (PA) and the maximum output power can be up to +13dBm.

Serial Interface

The RF communicates with a MCU via a 3-wire SPI interface (CSN, SCK, SDIO) or a 4-wire SPI interface (SDO from GIO1 or GIO2) with a data rate up to 250Kbps. An SPI transmission is an (8+8×n) bits sequence which consists of an 8-bit command and n×8 bits of data, where n can be 0 or any natural number. If the number n is greater than the address boundary, the address will return to zero. The MCU should pull the CSN (SPI chip select) pin low in order to access the RF portion. Using the SPI interface, user can access the control registers and issue Strobe commands. When writing data to the RF chip, the SPI data will be latched into the registers at the rising edge of the SCK signal. When reading data from the RF chip registers, the bit data will be transferred at the falling edge of the SCK signal after the target register address has been input.

Command (8 bits)								Data (8 bits)							
C7	C6	C5	C4	C3	C2	C1	C0	D7	D6	D5	D4	D3	D2	D1	D0

SPI Command Format

Two kinds of command are defined. One is 1-byte command only, named CmdO, and the other is 1-byte command followed by n-byte data, named CmdD.

C7	C6	C5	C4	C3	C2	C1	C0	Description	CmdO	CmdD
0	1	A5	A4	A3	A2	A1	A0	Write to control registers		√
1	1	A5	A4	A3	A2	A1	A0	Read from control registers		√
0	0	1	x	x	x	B1	B0	Set register bank	√	
0	0	0	1	x	x	x	0	Write SYNCWORD command		√
1	0	0	1	x	x	x	0	Read SYNCWORD command		√
0	0	0	1	x	x	x	1	TX FIFO write command		√

C7	C6	C5	C4	C3	C2	C1	C0	Description	CmdO	CmdD
1	0	0	1	x	x	x	1	RX FIFO read command		√
1	0	0	1	1	1	1	1	Read Chip ID command		√
0	0	0	0	1	0	0	0	Software reset command	√	
0	0	0	0	1	0	0	1	TX FIFO address pointer reset command	√	
1	0	0	0	1	0	0	1	RX FIFO address pointer reset command	√	
0	0	0	0	1	0	1	0	Deep Sleep mode	√	
0	0	0	0	1	0	1	1	Idle mode	√	
0	0	0	0	1	1	0	0	Light Sleep mode	√	
0	0	0	0	1	1	0	1	Standby mode	√	
0	0	0	0	1	1	1	0	TX mode	√	
1	0	0	0	1	1	1	0	RX mode	√	

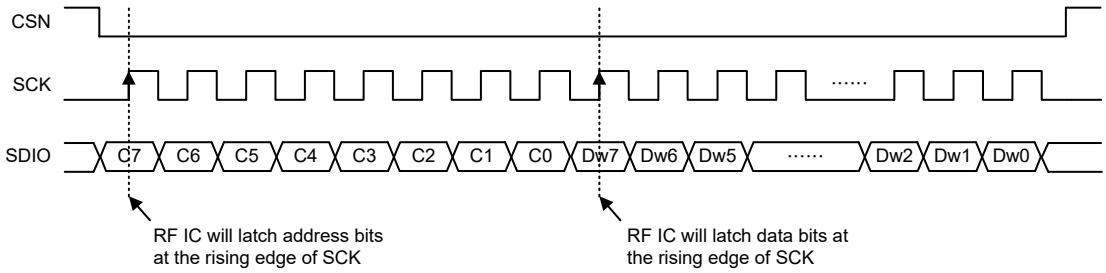
A5~A0: The address of control registers.

x: Hardware doesn't care but it is recommended to set to 0 by software.

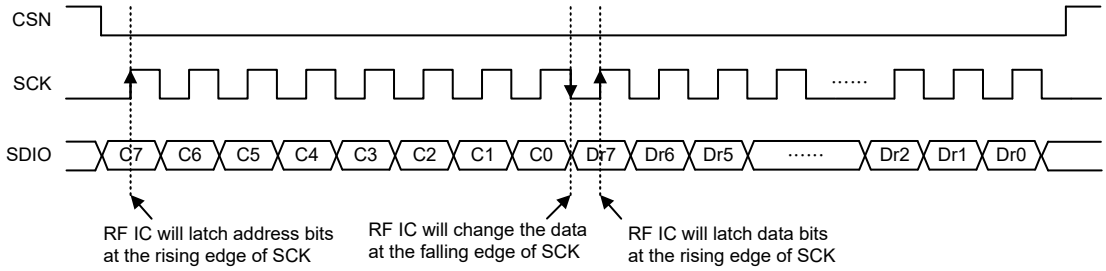
B1~B0: Bank number.

Note: 1. The chip supports multi-byte read/write operations and the address is increased automatically after each read or write operation.
2. Using software to read/write multiple bytes is allowed after one read/write command in a single CSN enabled cycle.
3. In the sleep mode, GIOs will keep the same level of the last operation mode.
4. Chip ID is a 2-byte data.

SPI Timing



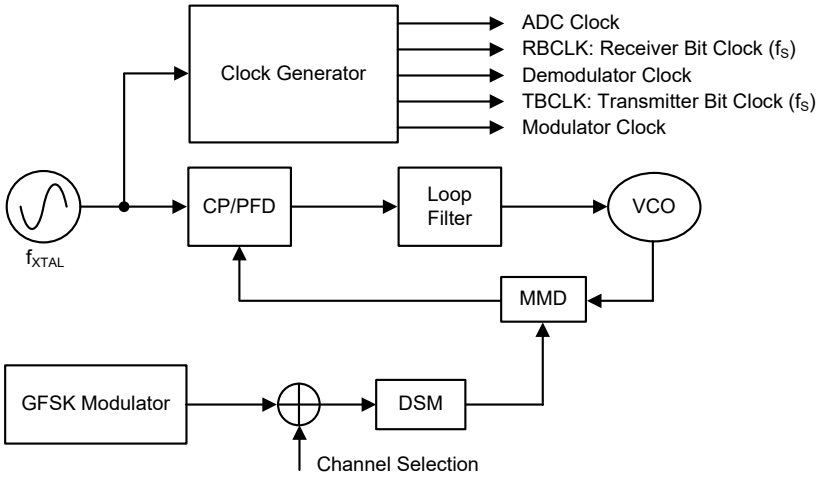
3-Wire SPI Interface Write 1-byte Data Operation



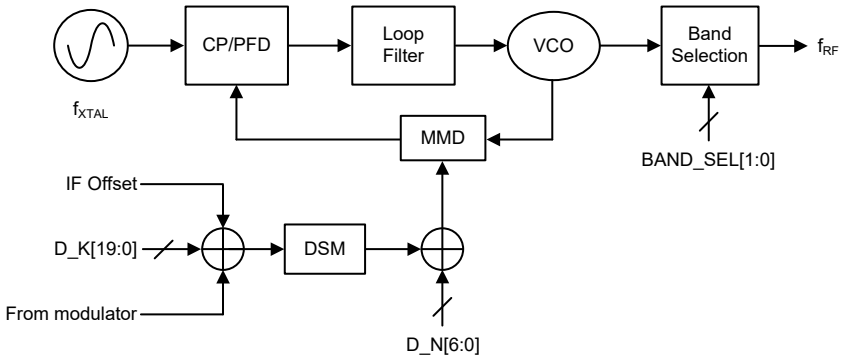
3-Wire SPI Interface Read 1-byte Data Operation

System Clock

The main system clock of the RF comes from the X'tal oscillator. All internal operation clocks of various functional blocks are derived from the X'tal oscillator.



Frequency Synthesizer



The RF transceiver frequency is generated by a high resolution fractional-N Delta Sigma frequency synthesizer. By appropriate setting on the configuration parameters D_N[6:0] and D_K[19:0], a low-noise LO frequency can be generated to comply with various radio regulatory standards including ETSI EN, FCC, etc. In the RX mode, the synthesizer would generate an LO-IF frequency for the RX mixer operation, RXIFOS[11:0] is used to generate the required IF(Intermedia Frequency) offset. For data rate equal to or larger than 200Kbps, IF should be set to 300kHz, otherwise IF should be set to 200kHz. In the TX mode, there is extra input from the modulator to provide extra frequency deviation waveform of baseband data.

$$D\_N[6:0]=\text{Floor}\left(\frac{f_{RF}}{f_{XTAL}/(XODIV2+1)}\right)$$

$$D\_K[19:0]=\text{Floor}\left(\left(\frac{f_{RF}}{f_{XTAL}/(XODIV2+1)}-D\_N[6:0]\right)\times 2^{20}\right)$$

$$RXIFOS[11:0]=\text{Floor}\left(\left(\frac{f_{IF}}{f_{XTAL}/(XODIV2+1)}\right)\times 2^{17}\right)$$

Modulator

The BA45F5640 supports GFSK modulation. A BT=0.5 Gaussian filter for pulse smoothing is implemented in the RF portion. The frequency deviation, f_{DEV}, of the transmitter is programmed using the FSCALE[11:0] field. The value of FSCALE[11:0] is determined by the modulation index h, the XO output divided by 2 control bit XODIV2, data rate f_S and f_{XTAL}.

$$h=\frac{2\times f_{DEV}}{f_S}$$

$$FSCALE[11:0]=round\left((h\times\frac{f_S}{f_{XTAL}/(XODIV2+1)})\times 2^{15}\right), \text{ take the least significant 12 bits}$$

For low data rate applications (≤10Kbps), a modulation index of 8 is recommended. For high data rate applications (≥50Kbps), a modulation index of 0.75 is recommended. For applications where data rate is between the aforementioned boundaries, keeping the frequency deviation above 20kHz is recommended.

The FSCALE bit field needs to multiply a scaling factor for data rate equal to or larger than 100Kbps. The recommended FSACLE values for data rates equal to or larger than 100Kbps are provided in the filter coefficients tables following the Filter Coefficient Control Registers.

State Machine

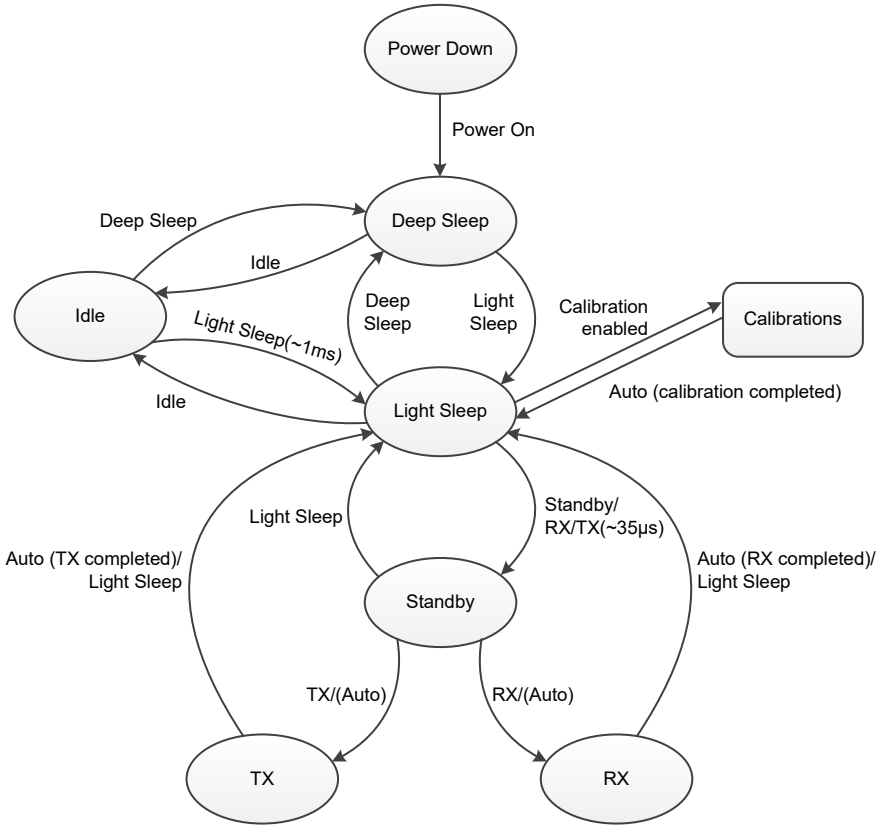
There are seven operating modes in the RF. The operation modes and key funcitons on/off state in the corresponding mode are listed below.

1. Power Down mode
2. Deep Sleep mode
3. Light Sleep mode
4. Standby mode
5. Idle mode
6. TX mode
7. RX mode

Mode	Register Retention	3.3V	LIRC	Regulator	XO	Standby+VCO	TX	RX	Strobe Command
Power Down	No	OFF	OFF	OFF	OFF	OFF	OFF	OFF	—
Deep Sleep	Yes	ON	OFF	OFF	OFF	OFF	OFF	OFF	0000_1010
Light Sleep	Yes	ON	OFF	ON	ON	OFF	OFF	OFF	0000_1100
Idle	Yes	ON	ON	OFF	OFF	OFF	OFF	OFF	0000_1011
Standby	Yes	ON	OFF	ON	ON	ON	OFF	OFF	0000_1101
TX	Yes	ON	OFF	ON	ON	ON	ON	OFF	0000_1110
RX	Yes	ON	OFF	ON	ON	ON	OFF	ON	1000_1110

TX/RX FIFO Mode (DIR_EN=0) State Machine

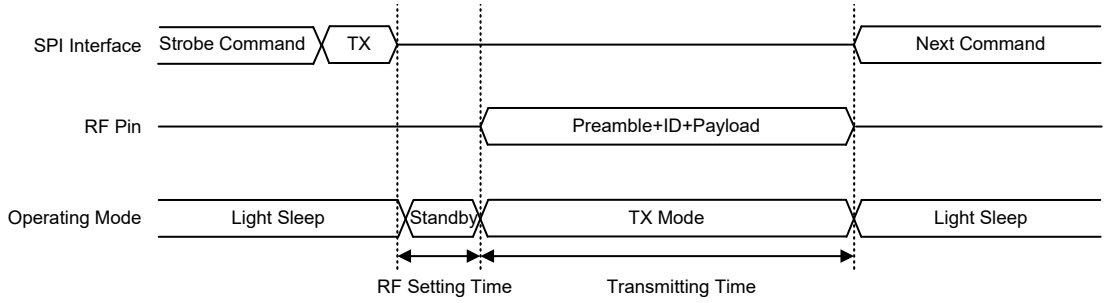
If the DIR_EN bit is cleared to 0, the device mode transactions are implemented by strobe command from the MCU and the TX/RX data are derived from the packet handling hardware.



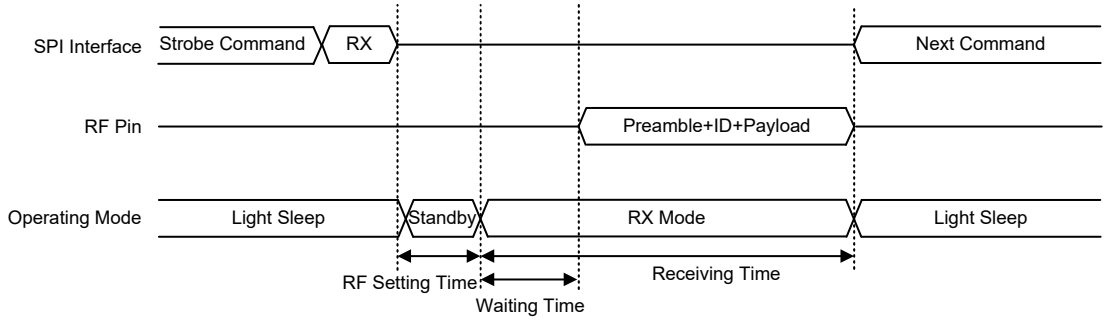
FIFO Mode State Diagram

Initially, the RF is in the Power Down mode. After the device completes the internal power on reset, it will enter the Deep Sleep mode and wait for further strobe commands from the MCU. If the Light Sleep command is received, the device will enable the internal LDO, oscillate the XO and enter the Light Sleep mode. In this state, the MCU can have the RF execute calibration process if necessary. For normal TRX operations, the MCU can issue a RX or TX command to the RF. After receiving the TX or RX command, the device will first enter the Standby mode which lasts a certain period known as TX/RX settling time. After the settling time has escaped, the device will finally enter the RX or TX mode. The device will stay in the TX/RX sate until the TX/RX event is completed, after which the device will return to the Light Sleep mode automatically.

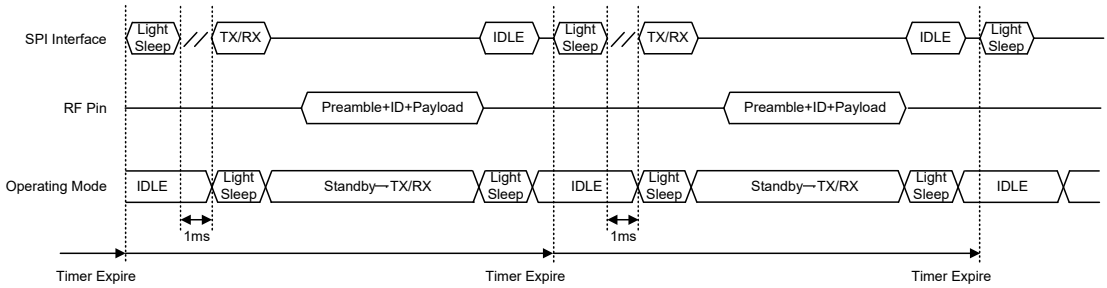
For low power periodical wireless transmission, the device supports low power Idle mode where the LIRC and wake-up timer are turned on. By appropriate timer setting and issuing the Idle mode command, the device will turn off the LDO and XO and enter the Idle mode. The device stays in the Idle mode until the timer expires and then an IRQ will be asserted on the GIO to wake-up the MCU. Then the MCU can have the device enter the Light Sleep mode and continue to execute normal TX/RX operations. After the TX/RX event is completed, the MCU can issue the Idle command to have the device return to the Idle mode again.



TX Timing in FIFO Mode



RX Timing in FIFO Mode



Periodical TX/RX Timing

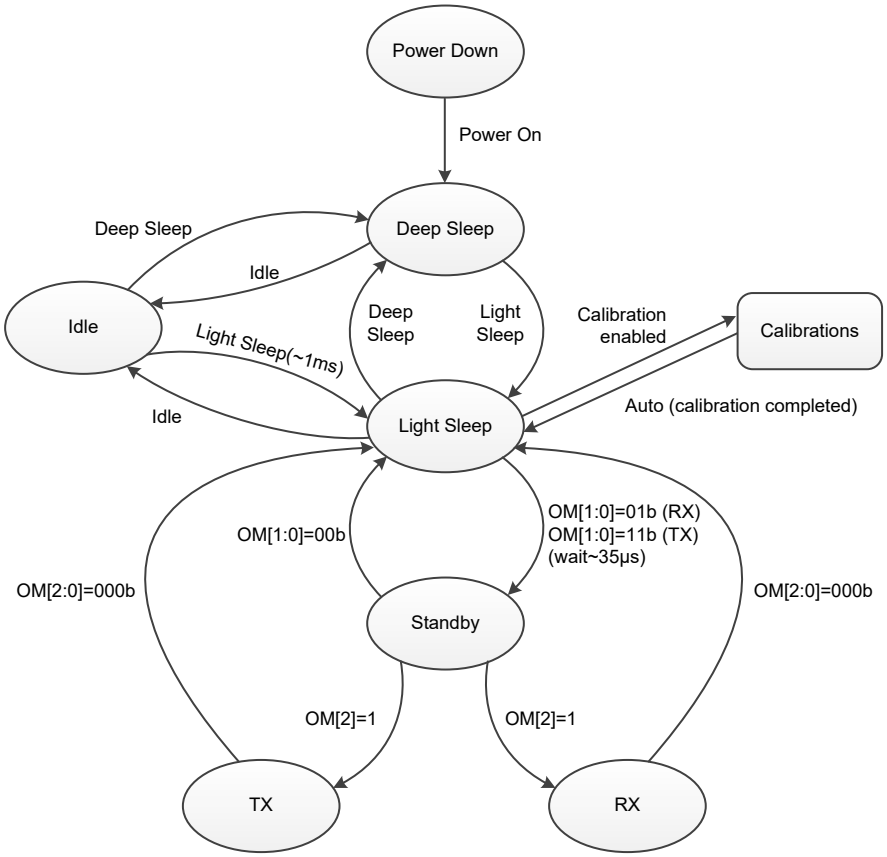
TX/RX Direct Mode (DIR_EN=1) State Machine

If the DIR_EN bit is set to 1, TX data is derived directly from the MCU to RF and RX data is sent directly from the RF to the MCU. In order to simplify the data bit clock synchronization between the RF and the MCU, the RF outputs the TBCLK/RCLK from GIO4 by setting GIO4S[3:0]. Both TBCLK and RBCLK are in 50/50 duty cycle. In the transmitting mode, the MCU outputs bit data at the rising edge of the TBCLK signal and the RF samples the TX bit data at the falling edge of the TBCLK signal. In the receiving mode, the MCU receives data at the rising edge of the RBCLK signal and the RF outputs bit data at the falling edge of the RBCLK signal. The MCU can select GIO1 or GIO2 for the TX/RX bit data transmission by setting GIO1S[2:0] or GIO2S[2:0].

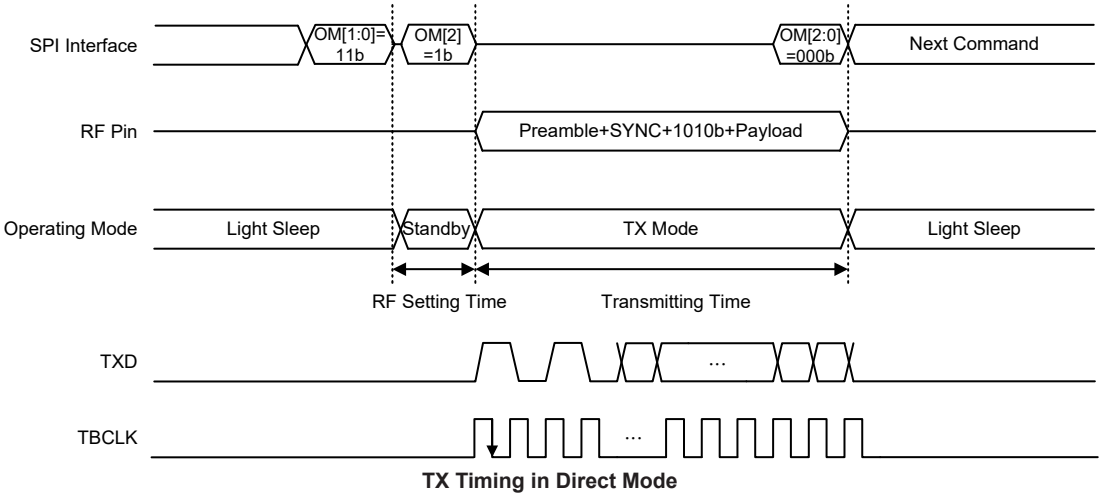
For TX operations in the direct mode, the MCU needs to set the OM[1:0] bits, i.e. RTX_SEL and SX_EN, to 11b to select the TX mode and have the RF enter standby mode first, then set the OM[2] bit, RTX_EN, to 1 to have the RF start to transmit the TX data. As long as the MCU sets OM[2:0] to 000b, the RF will return to the Light Sleep mode.

For RX operations in the direct mode, the MCU needs to set OM[1:0] to 01b first, then set OM[2] to 1 to have the RF start to receive data from the air. After the RF receives the matched SYNCWORD code, it will output the RBCLK clock, receive data bit (payload part) and then transmit to the MCU.

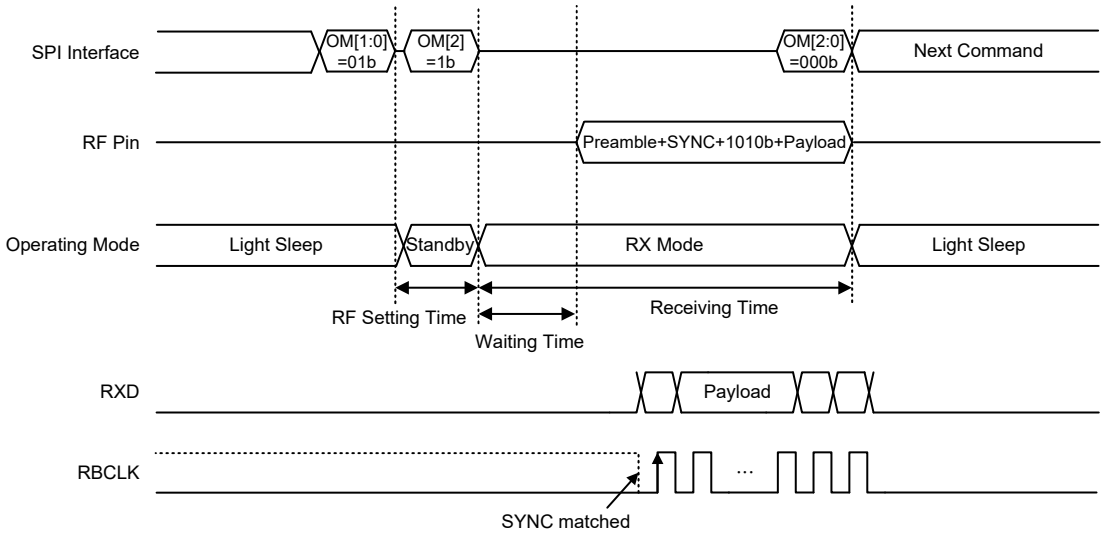
In direct mode, the transmission data length has no limit.



Direct Mode State Diagram



TX Timing in Direct Mode



RX Timing in Direct Mode

Calibration

The device has three calibration functions, VCO, RC and LIRC calibrations, allowing users to auto select proper setting to compensate the PVT (Process-Voltage-Temperature) variation effect. The control bit, ACAL_EN, is used to enable the VCO and RC calibration functions at the same time and both calibration functions will be automatically implemented after this bit is set high. When the calibrations are completed, the ACAL_EN bit is cleared to zero by hardware. The MCU can poll the ACAL_EN bit status or use the calibration complete interrupt flag CALCMPIF to check the calibration status. The device also has an independent enable bit, LIRCCAL_EN, for the LIRC calibration function, allowing to independently implementing the LIRC calibration function.

LIRC Calibration

There is an internal low frequency RC oscillator in the RF providing a clock source for the wake-up timer in the Idle mode. After calibration, the internal low frequency RC oscillator supports a precision of ±2% for the PVT variation. The calibration process chooses the curve setting of LIRC frequency to an approximation of 32768Hz. Then an extra LIRC correction process is used to tune the wake-up timer accuracy error to be less than ±1%.

The MCU need to configure LIRC_OW=0 and LIRC_EN=1 before the LIRC calibration. Then the RF will do LIRC calibration when LIRCCAL_EN is set to 1 by the MCU during the Light Sleep mode. The LIRCCAL_EN bit is reset to 0 by hardware on the completion of LIRC calibration. The LIRC calibration process would take about 4ms.

AGC & RSSI

In order to enhance the receiving dynamic range and ensure the signal SNR be no less than the demodulator minimum SNR requirement, an AGC (Auto-Gain-Control) function block is embedded. The AGC would tune the receiver gain to get valid signal level before ADC staying between the set point and -26 dBFS. The set point is in the range of -6 dBFS to -12 dBFS which is adjusted by the SAT_SEL[1:0] bit field. FS is the full-scale of ADC.

There is an integrated RSSI (Receiver Signal Strength Indicator) measurement function block in the RF. The RSSI calculation engine calculates the receiving signal strength after ADC. By combining the calculated ADC signal strength value and receiver chain total gain, the RSSI value is induced. The valid RSSI reading value is from -110dBm to -10dBm. The RSSI measurement error is normally

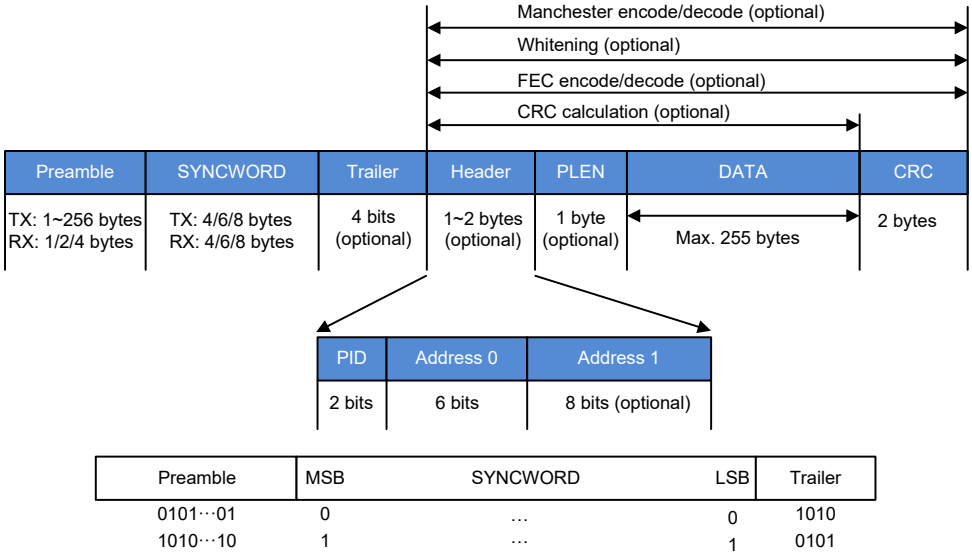
below ±6dBm. The unit of the reading value is -dBm. Two RSSI reading values are available, one is RSSI_SYNC_OK[7:0] that is a RSSI measurement value snapshot when the valid SYNCWORD is detected OK, the other reading value is RSSI_NEGDB[7:0] that is a real time RSSI calculation result. In the receiving mode, the MCU can access the RSSI_NEGDB bit field. After the receiving is completed, the MCU should poll the RSSI_SYNC_OK[7:0] field for receiving signal strength assessment.

Packet Handler

In the TX mode, the packet handler is used to move the transmitting data out of FIFO and implement channel coding according to the packet format, then sends the packet to the modulator. In the RX mode, the packet handler is used to implement channel decoding with data from the demodulator and store the payload data into FIFO.

The packet handler performs several tasks such as Preamble and SYNCWORD insertion, Forward Error Correction, CRC calculation/checking, whitening/de-whiteing and Manchester encode/decode.

Packet Format



Preamble	MSB	SYNCWORD	LSB	Trailer
0101···01	0	…	0	1010
1010···10	1	…	1	0101

Note: 1. Preamble format will follow SYNCWORD MSB to inverse.
If MSB=0, Preamble format=0101…01
If MSB=1, Preamble format=1010…10
2. Trailer format will follow SYNCWORD LSB to inverse.
If LSB=0, Trailer format=1010
If LSB=1, Trailer format=0101
3. The Trailer field contains 4 bits and is an optional field which is enabled by TRAILER_EN.

Preamble

The packet starts with a preamble with a length of 1~256 bytes set by TXPMLEN[7:0] in the TX mode. In the RX mode, preamble detection length is limited to 1, 2 or 4 bytes selected by RXPMLEN[1:0].

SYNCWORD

The SYNCWORD length which is set by SYNCLEN[1:0] can be 4, 6 or 8 bytes in the TX mode. In the RX mode, the detection length is also 4, 6 or 8 bytes. When the RX side receives a matched SYNCWORD packet, the DATA field will be stored in the FIFO.

Trailer

The trailer field length is fixed at 4 bits which is a concatenating field between SYNCWORD and the latter payload.

Header

The header (Payload Header) is optional and enabled by PLH_EN. The payload header length can be 1 or 2 bytes set by PLHLEN. When the PLHLEN bit is 0, only PID[1:0] and PLHA[5:0] (address 0) fields are used in the packet. PID[1:0] is located in bit 7~6 of the payload header field. If PLHAC_EN=0, PLHA[5:0] can be used as software flags and the actual function can be defined by users. If PLHAC_EN=1, the device will compare the local PLHA[5:0] field with the received PLHA[5:0] field. If matched, the receiving data will be moved into the RX FIFO, otherise the following incoming data will be abandoned. The PLHA[5:0] field is used to support broadcast function and PLHA[5:0]=0 is a special preserved address permitting the RF not to implement the address filtering mechanism.

When the PLHLEN bit is set to 1, the address field length is extended to 14 bits which is formed by address 0 (PLHA[5:0]) and address 1 (PLHEA[7:0]).

PLEN (Payload Length)

The PLEN field is optional and its length is fixed at 1 byte once being enabled by PLLEN_EN. When this bit is set high, the DATA field length is variable and is determined by the PLEN field in each TX/RX packet.

DATA

When in the TX mode, the TX data length is determined by the TXDLEN[7:0] field. The maximum length is 255 bytes in the extend FIFO mode. In the special case of infinite FIFO mode, the length can exceed 255 with an infinite length. If PLLEN_EN=1, the PLEN field in the TX packet is enabled and the PLEN content is equal to TXDLEN[7:0]. When in the RX mode, the RX data length is set by RXDLEN[7:0] if PLLEN_EN=0 and by PLEN field in the receiving packet if PLLEN_EN=1.

CRC

The CRC field is optional and is enabled by CRC_EN. It is recommend to always set CRC_EN to 1 for data correctness checking. There are two CRC formulas selected by setting the CRCFMT bit.

CRCFMT=0: CCITT-16-CRC $G(X) = X^{16} + X^{12} + X^5 + 1$

CRCFMT=1: IBC-16-CRC $G(X) = X^{16} + X^{15} + X^2 + 1$

FEC

The optional data encode/decode function can be enabled by FEC_EN. Use (7,4) Hamming code to correct 1-bit error and more than 1-bit error detect for each 4-bit data. After FEC, the data length for each data will be (4+3)×2=14 bits.

- **Hamming Code Function Table**

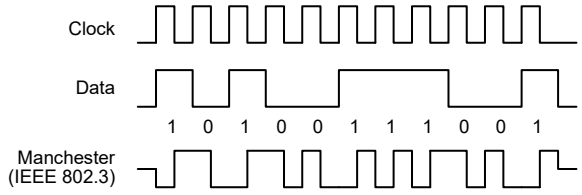
Bit	7	6	5	4	3	2	1
Transmitted Bit	D3	D2	D1	P2	D0	P1	P0
P0	Y	N	Y	N	Y	N	Y
P1	Y	Y	N	N	Y	Y	N
P2	Y	Y	Y	Y	N	N	N

Data Whitening

The optional data whitening/de-whitening function can be enabled by WHT_EN. Use PN7 code to implement XOR operation with the transmitted data. The whitening seed is set by WHTSD[6:0].

Manchester Code

The optional Manchester encode/decode function can be enabled by MCH_EN. Each bit after Manchester encoding will be extended into two bits and recovered to one bit data after decoding.



Manchester Code Example

FIFO Operation Modes

In Burst mode, data transmission to the RF transmitter is derived from FIFO and is pre-written by the MCU. There are 4 FIFO modes to support various applications. They are the Simple FIFO mode, Block FIFO mode, Extend FIFO mode and Infinite FIFO mode.

FIFO Reset

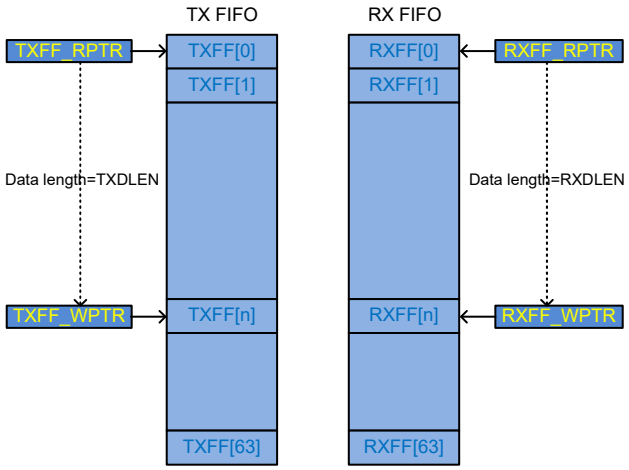
To use the FIFO in the burst mode, issue the TX FIFO address pointer reset command and RX FIFO address pointer reset command to reset the FIFO pointer and buffer first. After this, the FIFO is in the initial state same as reset.

Simple FIFO Mode

This FIFO mode is used for general applications with a TX/RX data length less then or equal to 64 bytes. The data length should not exceed 64 bytes. To use the simple FIFO mode, the MCU must write the transmitting data to FIFO by the SPI write FIFO command. The transmitting sequence is first written byte first out and the MSB in each byte first out to the transmitter. Users should determine all transmitting data packet format including the preamble, SYNCWORD and packet encoding such as FEC, CRC, whiting. After FIFO data filling out is completed, clear the TXFFSA[5:0] field and set TXDLEN[7:0]/RXDLEN[7:0] field to the desired transmitting/receiving length in bytes. Then issue the TX command to start the transmission. After the current transmitting is completed, the data will be kept in FIFO to wait for the next transmission.

Programming procedure:

1. Reset TX FIFO by the SPI reset TX FIFO command.
2. Reset RX FIFO by the SPI reset RX FIFO command.
3. TXFFSA[5:0] must be cleared to 0.
4. Fill out TX FIFO by the SPI write FIFO command.
5. Set TXDLEN[7:0]/RXDLEN[7:0] to control the TX/RX length in bytes.
6. Issue the TX command for transmitter and RX command for receiver.
7. TX/RX completion is acknowledged by the TX/RX complete IRQ.
8. Re-transmitting TX packet with the same data will auto-reset TXFF_RPTR to 0.

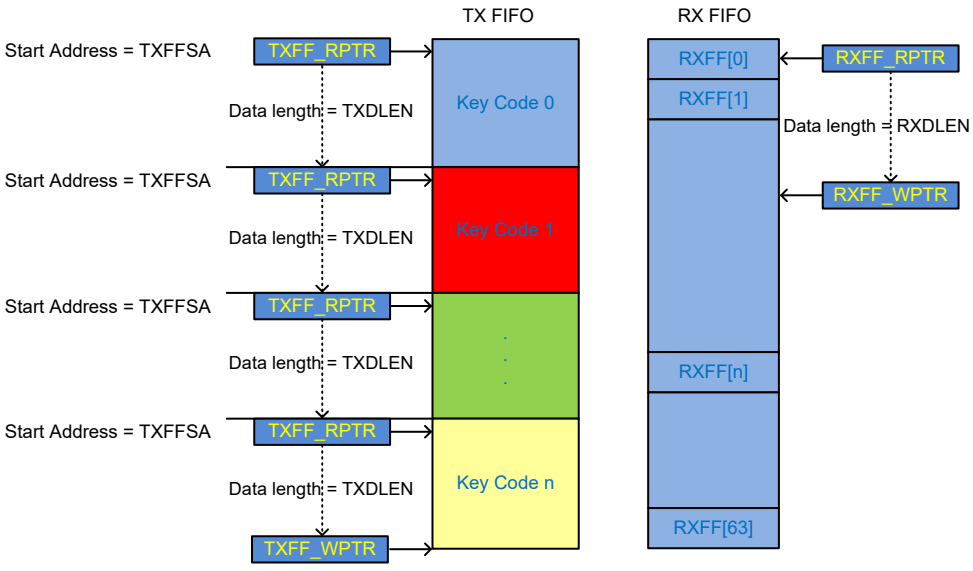


Block FIFO Mode

The Bock FIFO mode is used to support multi-key code applications. Users should write all the key codes to FIFO first. When a key is pressed, the MCU will detect the key and set TXFFSA[5:0] to the target key code start address and set TXDLEN[7:0] to indicate the key code length and issue the TX strobe command to start the transmission. The maximum FIFO length is also limited to 64 bytes.

Programming procedure:

1. TX: Write key code 0~n to TX FIFO by SPI write FIFO command.
2. TX: Set TXDLEN[7:0] for key code length.
3. TX: When a key is pressed, the MCU will set TXFFSA[5:0] to the start address of the corresponding key code.
4. RX: Set the RXDLEN[7:0] to key code length and then enter the RX mode by SPI command.
5. TX: Issue TX command for transmitter.
6. RX: Issue RX command for receiver.
7. TX/RX completion is acknowledged by the TX/RX complete IRQ.



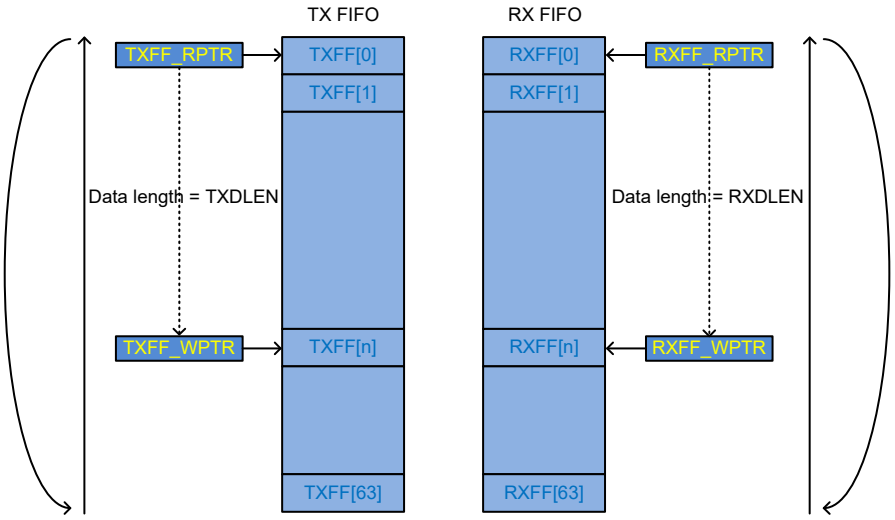
Extend FIFO Mode

The Extend FIFO mode is used for transmissions with a long payload data packet. The maximum length is 255 bytes. As the physical FIFO length is 64 bytes, to extend the available transmitting length in one packet, a handshake mechanism is needed between the MCU and the FIFO controller.

Set FFMG[1:0] to determine the FIFO data length margin and set FFMG_EN to enable the margin detect function to inform the MCU when the TX FIFO data fullness level is less than the margin. The MCU should write data to TX FIFO fast enough when receiving this reminding signal to avoid transmission being terminated by TX FIFO underflow.

Programming procedure:

1. Set FFMG_EN to enable FIFO depth low threshold detect function and set FFMG[1:0] to select the threshold, 4, 8, 16 or 32 bytes.
2. Set the FIFOLTIE bit to 1 to enable the FIFO low threshold IRQ.
3. Set GIOnS field (n=1~4) = 101b to output IRQ on GIO1~GIO4.
4. TX: If MCU detects the FIFO low threshold IRQ signal, it will move data into TX FIFO with a data length less than or equal to (64-FFMG[1:0]). Then the MCU clears the FIFO low threshold IRQ flag FIFOLTIF and repeats the same routine until all TX data are completely written to TX FIFO.
5. RX: If MCU detects the FIFO low threshold IRQ signal, it will read data from RX FIFO with a data length equal to FFMG[1:0]. Then the MCU clears the FIFO low threshold IRQ flag FIFOLTIF and repeats the same routine until receiving the RX completion IRQ to read the remaining data from RX FIFO.



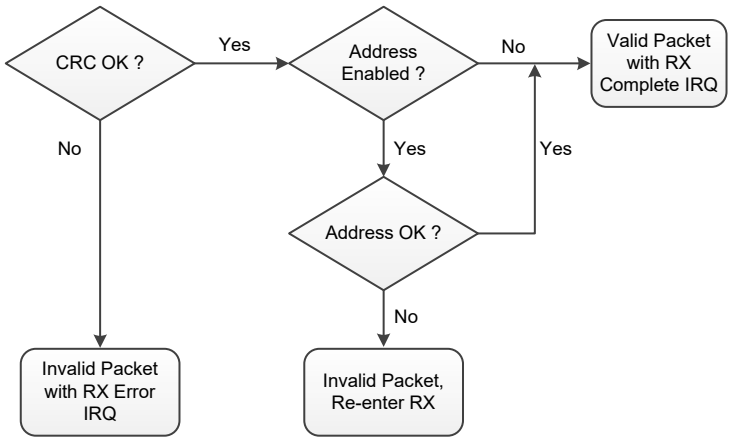
Infinite FIFO Mode

Programming procedure:

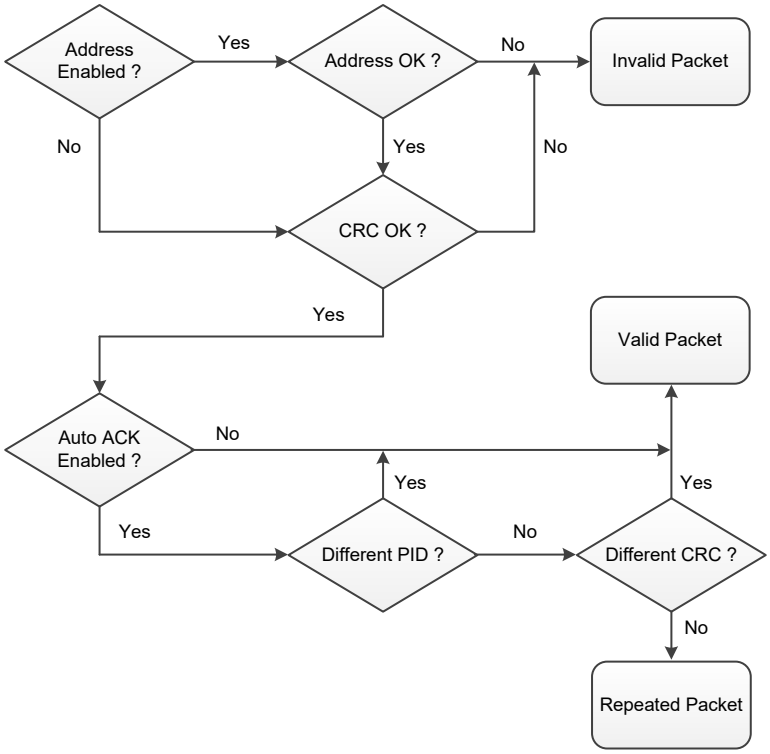
1. Set FFINF_EN to 1 to enable the Infinite FIFO mode.
2. The handshaking and IRQ function are identical with the Extend FIFO mode.
3. TX: If receiving the FIFO low threshold IRQ, the MCU continues to write TX data to TX FIFO with a data length less than or equal to (64-FFMG[1:0]). Then the MCU clears the FIFO low threshold IRQ flag and repeats the same routine until it wants to terminate the infinite FIFO mode. To terminate the infinite FIFO mode, after receiving IRQ and moving data to TX FIFO, the MCU should clear FFINF_EN to zero and set TXDLEN[7:0] to the remaining data length if the remaining transmitting length is less than 192 bytes and longer than 64 bytes. The terminating configuration should be programmed only once for one transmission. The packet will be terminated when all of the target data are transmitted completely.
4. RX: If receiving the FIFO low threshold IRQ, the MCU reads data from RX FIFO with a data length equal to FFMG[1:0]. Then the MCU clears the FIFO low threshold IRQ flag and repeats the same routine until it wants to terminate the infinite FIFO mode. To terminate the infinite FIFO mode, after receiving IRQ and reading data from RX FIFO, the MCU should clear FFINF_EN to zero and set RXDLEN[7:0] to the remaining data length if the remaining receiving length is less than 192 bytes and longer than 64 bytes. The terminating configuration should be programmed only once for one receiption. The packet will be terminated when all of the target data are received completely.

Receiving Packet Judgement

In normal RX operating mode, package reception follows the following judgement criteria.

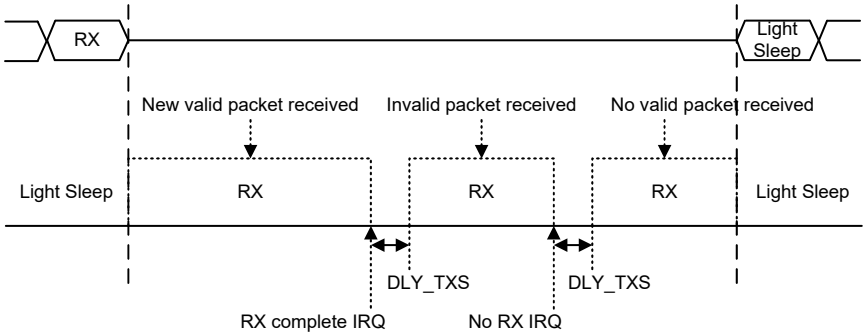


The RF adopts extra receiver packet judgment for the continuous RX mode and auto-acknowledge mode. The main purpose of these special link layer functions are used to alleviate MCU loading when handling TRX packet transaction.



Continuous RX Mode

There is a special continuous RX operating mode supported in the RF. The MCU can enable this continuous RX mode by setting the RXCON_EN bit high and start the continuous RX mode by issuing the RX strobe command to the device. If there is a valid RX packet received, the RF will issue a RX completion IRQ to the MCU. The device then repeats the RX operation after a duration defined by DLY_TXS[2:0] to keep listening for incoming packets. If an invalid packet is received, the RF would only repeat the RX operation without issuing the RX completion IRQ to the MCU. The MCU stops the continuous RX by issuing the Light Sleep strobe command to the RF. In the continuous RX mode, only simple FIFO mode can be used. In order to prevent the receiving packet data length field from being corrupted by new incoming packets before the MCU reads data from RX FIFO, users should set RXPL2F_EN=1 and PLLEN_EN=1 to have the PLEN information stored into the RX FIFO. Because of the existence of PLEN byte, the maximum packet data length becomes 63 bytes. If a new incoming packet arrives before the MCU reads RX FIFO, a FIFO overflow error will happen, in which condition the RF will issue a RX error IRQ to the MCU. At this moment, the MCU should exit the continuous RX mode and reset the RX FIFO pointer.

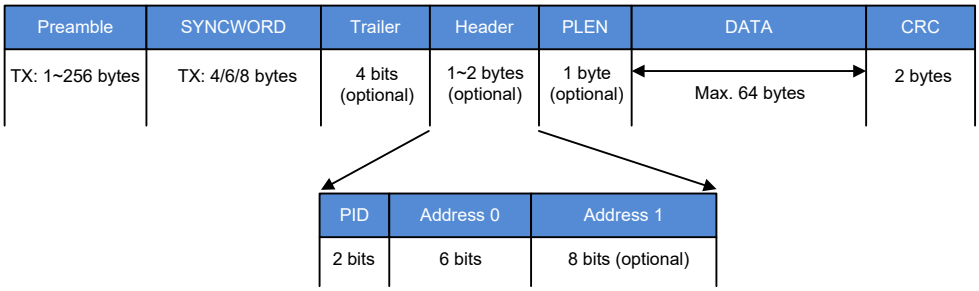


Continuous RX Mode

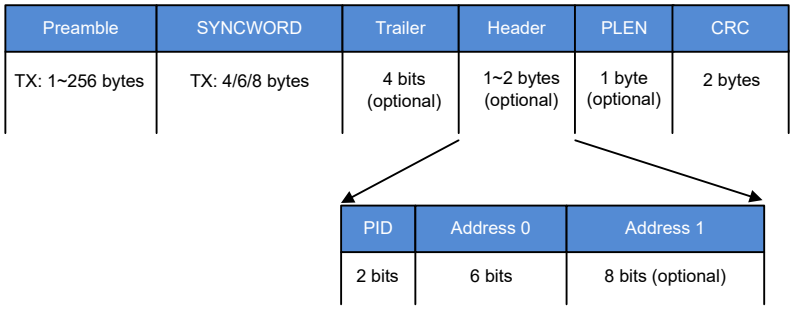
ARK Mode: Auto-Resend and Auto-Ack

The RF supports auto-resend and auto-ack mechanism by setting the ARK_EN bit high. This mechanism enables an easy two-way communication implementation however can only be operated in the simple FIFO mode.

Set ARK_EN to 1 to enable the device to enter the auto-resend and auto-ack ready mode. Then, auto-resend is triggered by the TX strobe command from the MCU and auto-ack is triggered by RX strobe command from the MCU. Packet format transmitted from the master to the slave in the auto-resend mode are illustrated below.



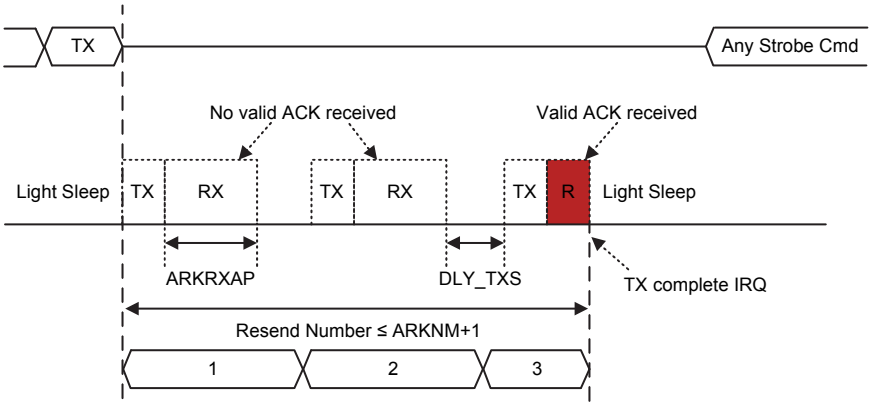
The slave side in the auto_ack mode uses the packet format as the following to be an acknowledge packet transmitted to master. Note that there is no payload data field used in the acknowledge packet.



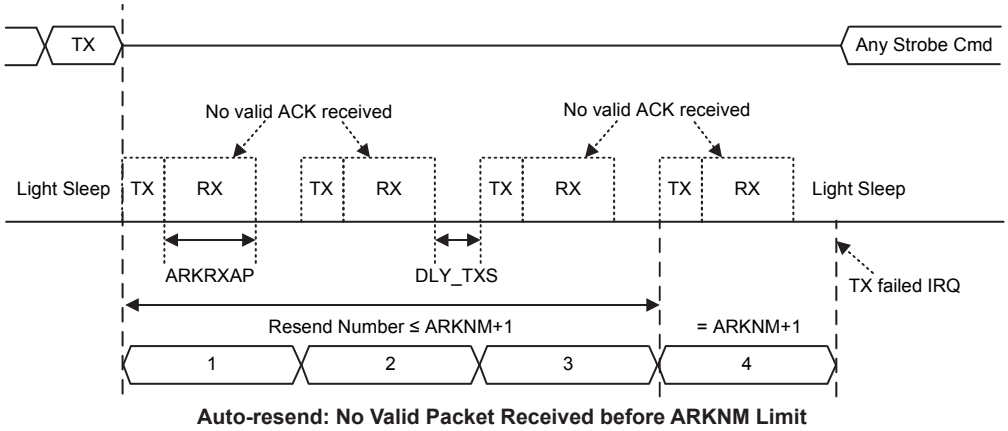
If the address field is used for the ARK mode, the auto-resend (master) side should configure the same address as the auto-ack (slave) side.

After configuring ARKNM[3:0], ARK_EN and ARKRXAP[7:0], the MCU starts the auo-resend process by issuing the TX strobe command. The RF starts to transmit data from the TX FIFO and then enters the RX mode after the TX completion. The RX period is in multiples of 250μs which is determined by ARKRXAP[7:0] plusing one. If the RF receives a valid acknowledge packet from the

slave side within the RX period with CRC checked correct, it will return to the Light Sleep mode and issue a TX completion IRQ to the MCU. Otherwise, the RF will check if the resend number has reached the limit set by ARKNM[3:0], if not, it will go to the TX mode to transmit the same TX data from the TX FIFO and the resend number will be increased by one.



Auto-resend: ACK Packet Received before ARKNM Limit

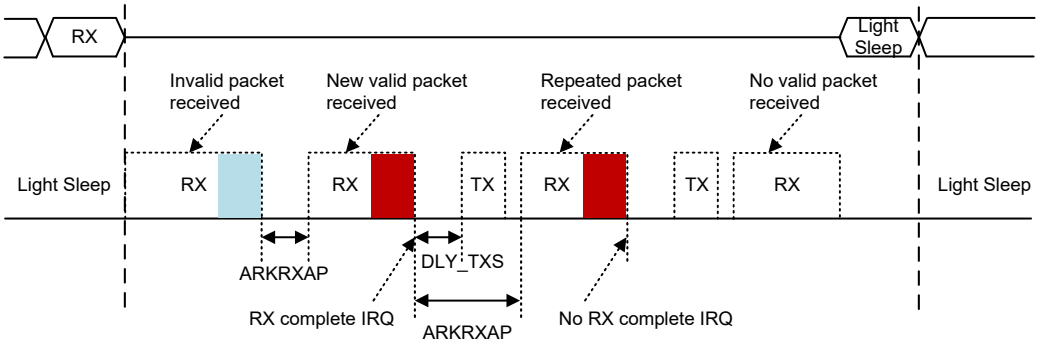


Auto-resend: No Valid Packet Received before ARKNM Limit

Regarding the auto-ack in the slave side, the MCU issues the RX strobe command to start the auto-ack process and issues the Light Sleep strobe command to stop the auto-ack process. In the auto-ack mode, an extra PID/CRC filtering function will be applied for the slave side to check the packet received. If the PID/CRC of the new incoming packet is same as the stored PID/CRC of the last packet, then the newly received packet would be treated as a repeated packet.

During the auto-ack process, if the device receives a valid packet with different PID/CRC and CRC/address checked correct, it will issue a RX completion IRQ to the MCU and auto-transmit the ACK packet to the master. If the device receives a packet with the same PID/CRC and CRC/address checked correct, it will treat this packet as the repeated packet. Then the device will not issue the RX IRQ to the MCU but still auto-transmit the ACK packet to the master. If the device receives a packet with CRC/address checked failed, no IRQ is issued and the device will automatically re-do the RX operation to continuely listening for incoming packets.

The gap period for the device to restart the next RX operation after the current RX completion is defined by ARKRXAP[7:0]. In general cases, the MCU should fetch the receiver FIFO data within this period after receiving the RX completion IRQ. Besides, the MCU needs to wait for a same duration if it wants to leave the ARK mode after receiving the RX completion IRQ.



Auto-ACK Process

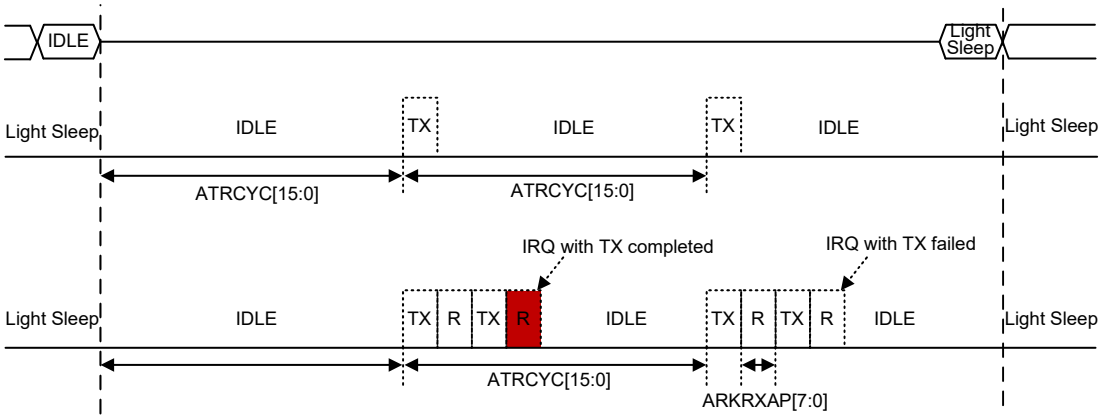
ATR Mode: Auto-Transmit-Receive

There is a special ATR operation mode in the RF to reduce the external host's loading. Two ATR functions are implemented within the device, one is WOR (Wake-On-RX) and the other is WOT (Wake-On-TX). They can only be operated with simple FIFO mode. These two operating modes need to co-work with an Idle mode timer which operates at a low frequency. The low frequency clock can be sourced from the internal LIRC or from the external ROSCi clock by setting the ATRCLKS bit in the ATR1 register. There are two operation modes for the ATRCT timer which is selected using the ATRCTM bit. Clearing the ATRCTM bit to 0 will select the single mode, where the ATRCT timer will restart upon every ATR transation when entering the Idle state. The ATRCT timer will stop and leave the ATR mode upon receiving the Light Sleep command. Setting the ATRCTM bit to 1 will select the continuous mode, where the ATRCT timer will start to operate upon receiving the Idle command and continuously run until the ATR_EN bit or the ATRCTM bit is cleared to zero.

After entering the ATR mode, only the Idle, Light Sleep, Set Register Bank and control register read/write commands can be recognized by the RF.

WOT (Wake-On-TX) Function

When the WOT function is enabled by setting the ATR_EN bit to 1 and the ATRM[1:0] bits to 00b, the device will periodically wake-up from the Idle mode and transmit TX FIFO contents without interaction with the MCU. The device starts the WOT process upon receiving the Idle strobe command from the MCU and stops the WOT process upon receiving the Light Sleep strobe command from the MCU. The ATRCYC[15:0] bits are used to set the wake-up period for the WOT function. At the moment of timer expiration, the wake-up timer will trigger the device to leave the Idle state and enter the active state to transmit data, at the same time the ATRCYC[15:0] content will be reloaded into the timer's counter. After finishing the TX operation, the device will return to the Idle mode and stay in this state until next wake-up timer expiration occurs. In the active state, the device only implements wake-up transmission once by default. Users can extend the wake-up transmitting mechanism by combining with the ARK function. The repeated transmitting number is controlled by the ARKNM[3:0] bits in the ATR7 register. The time duration between the repeated transmitting packets is inserted with one RX slot and controlled by ARKRXAP[7:0] in the ATR8 register. If the device receives ACK in the RX slot, a TX completion IRQ will be issued to inform the MCU.

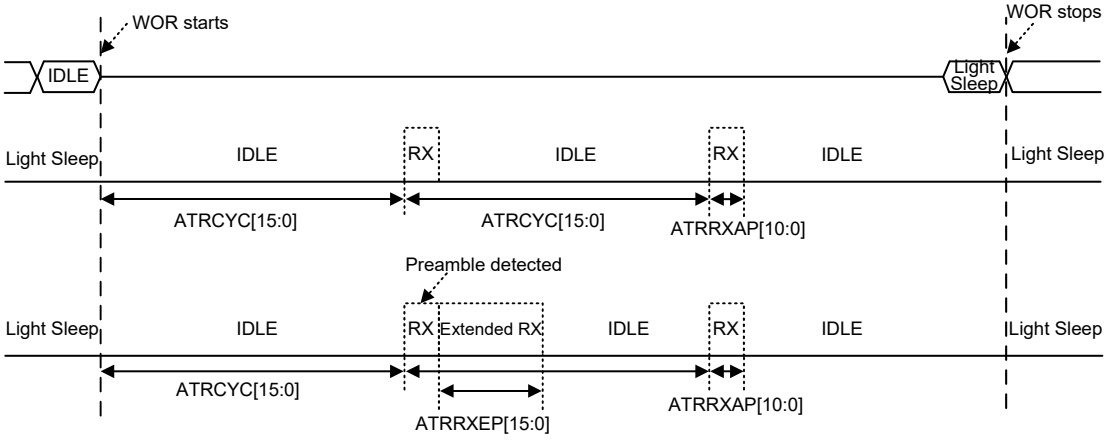


WOT Process

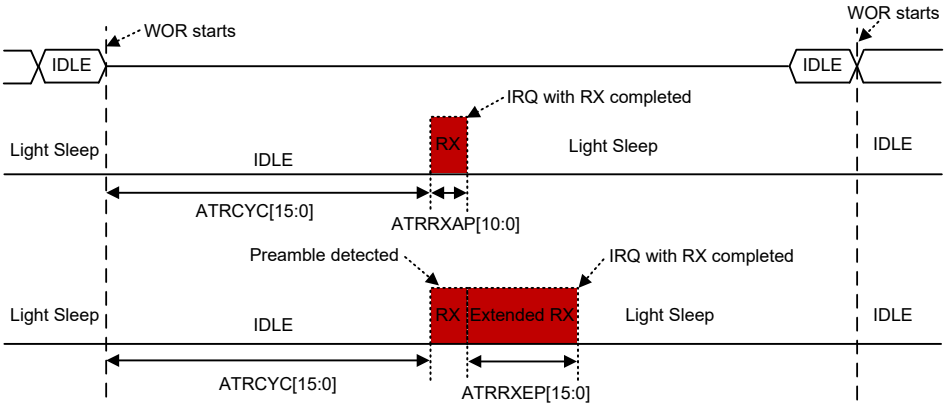
WOR (Wake-On-RX) Function

When the WOR function is enabled by setting the ATR_EN bit to 1 and the ATRM[1:0] bits to 01b, the device will periodically wake-up from the Idle mode and listen for the incoming packets without interaction with the host MCU. The device starts the WOR process upon receiving the Idle strobe command from the MCU and stops the WOR process upon receiving the Ligh Sleep strobe command from the MCU. The ATRCYC[15:0] bits are used to set the wake-up period for the WOR function. At the moment of ATR timer expiration, the wake-up timer will trigger the device to leave the Idle mode and enter the active state to listen for the incoming packet, at the same time the ATRCYC[15:0] content will be reloaded into the timer's counter. The receiving active period is defined by the ATRRXAP[10:0] bits. The active period is in multiples of 250μs and starts from 250μs. If there is no incoming packet received in the RX active period, the device will return to the Idle mode and wait for the next WOR cycle.

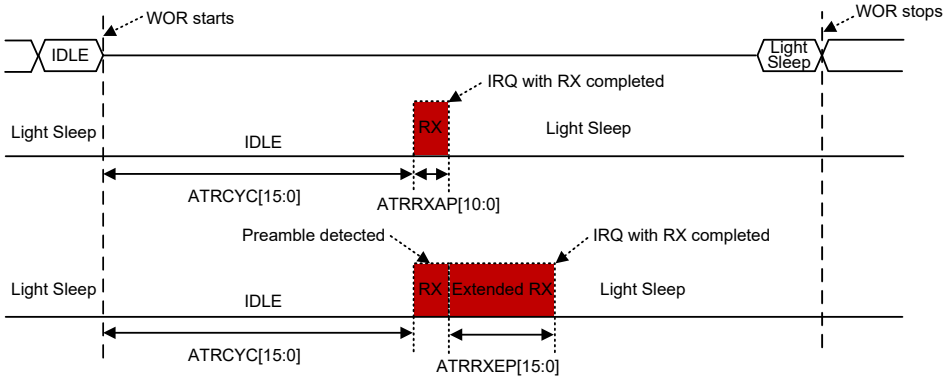
The active period is auto-extended when the preamble is detected. The extend period is defined by ATRRXEP[15:0]. The extend period is also in multiples of 250μs and starts from 250μs. Once the SYNCWORD is received, the receiving period would be auto-extended until the whole packet is completely received. After the RX receiving is done with CRC checked correct, the RF would acknowledge the MCU with RX complete IRQ and stay at light sleep mode. MCU can read the incoming packet from the RX FIFO and then restart the next WOR session by issuing Idle strobe command. If MCU want to leave WOR mode, MCU still need to issue light sleep command to the RF.



WOR without Incoming Packet Received

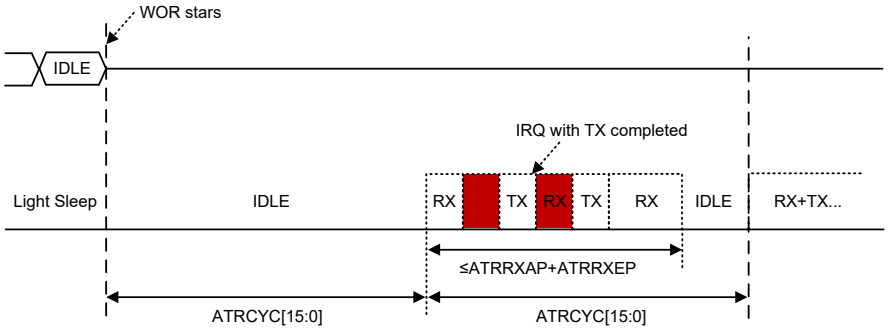


WOR with Incoming Packet Received



WOR Stops after Receiving Incoming Packet

In the WOR active period, the device only implements RX operation once by default. Users can extend the wake-up receiving mechanism by combining with the ARK function. In WOR+ARK mode, the time duration between the repeated receiving packets is inserted with one TX slot for acknowledgement. The TX duration depends on the transmitting data rate. The device stays in the RX mode for a maximum period of time defined by ATRRXAP+ATRRXEP. If a valid imcoming packet, with CRC checked OK and a different PID/CRC, is received before the timer expires, the device will issue a RX completion IRQ to the MCU and automatically enter the TX mode. If a repeated packet, with CRC checked correct and a same PID/CRC, is received, the device will only automatically enter the TX mode with no IRQ to the MCU. After the TX completion, the device will return to the RX mode again and listen for the incoming packets until the timer expires if no incoming packet is received.



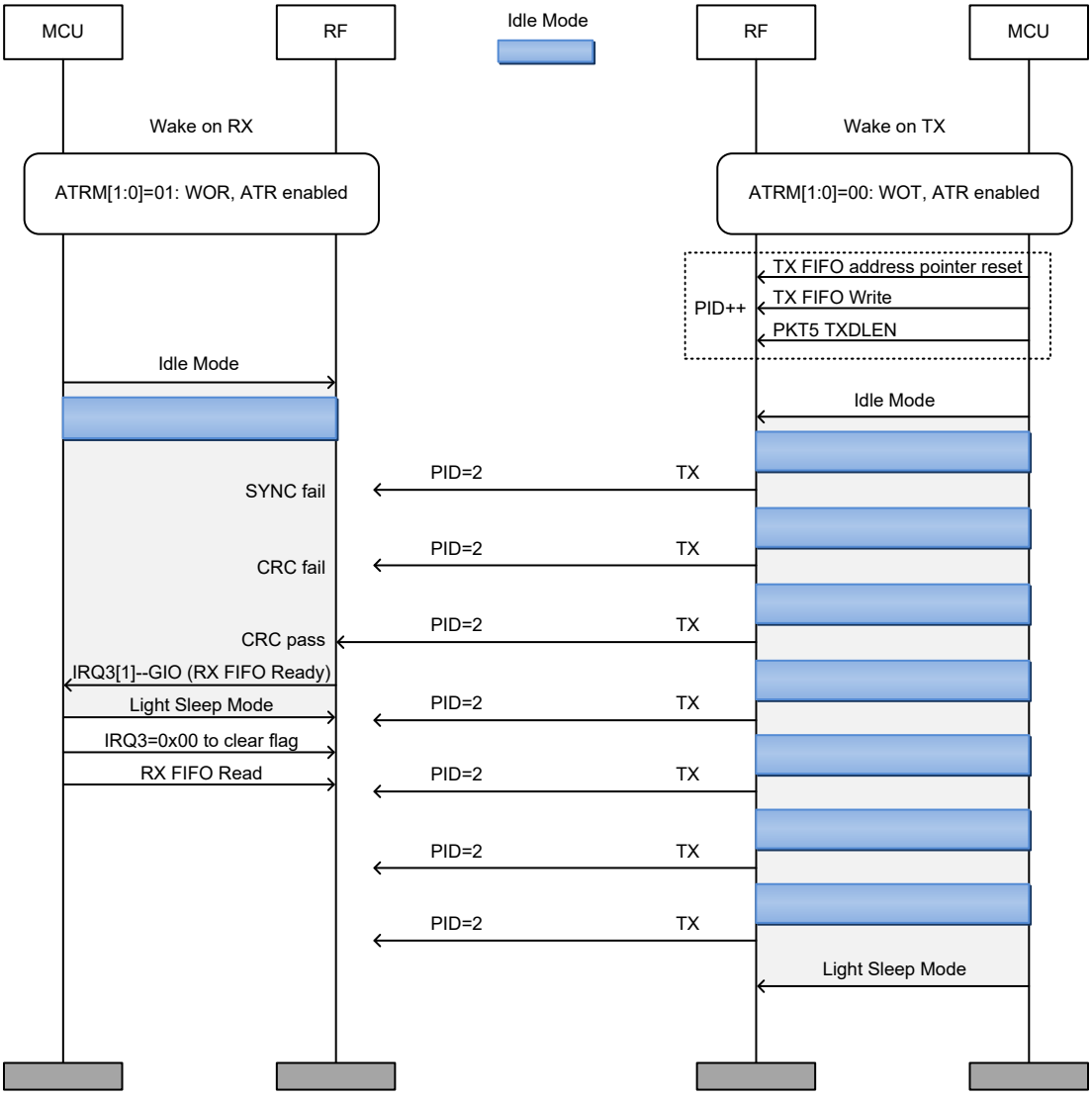
WOR+ARK Process

WTM (Wake up Timer Mode)

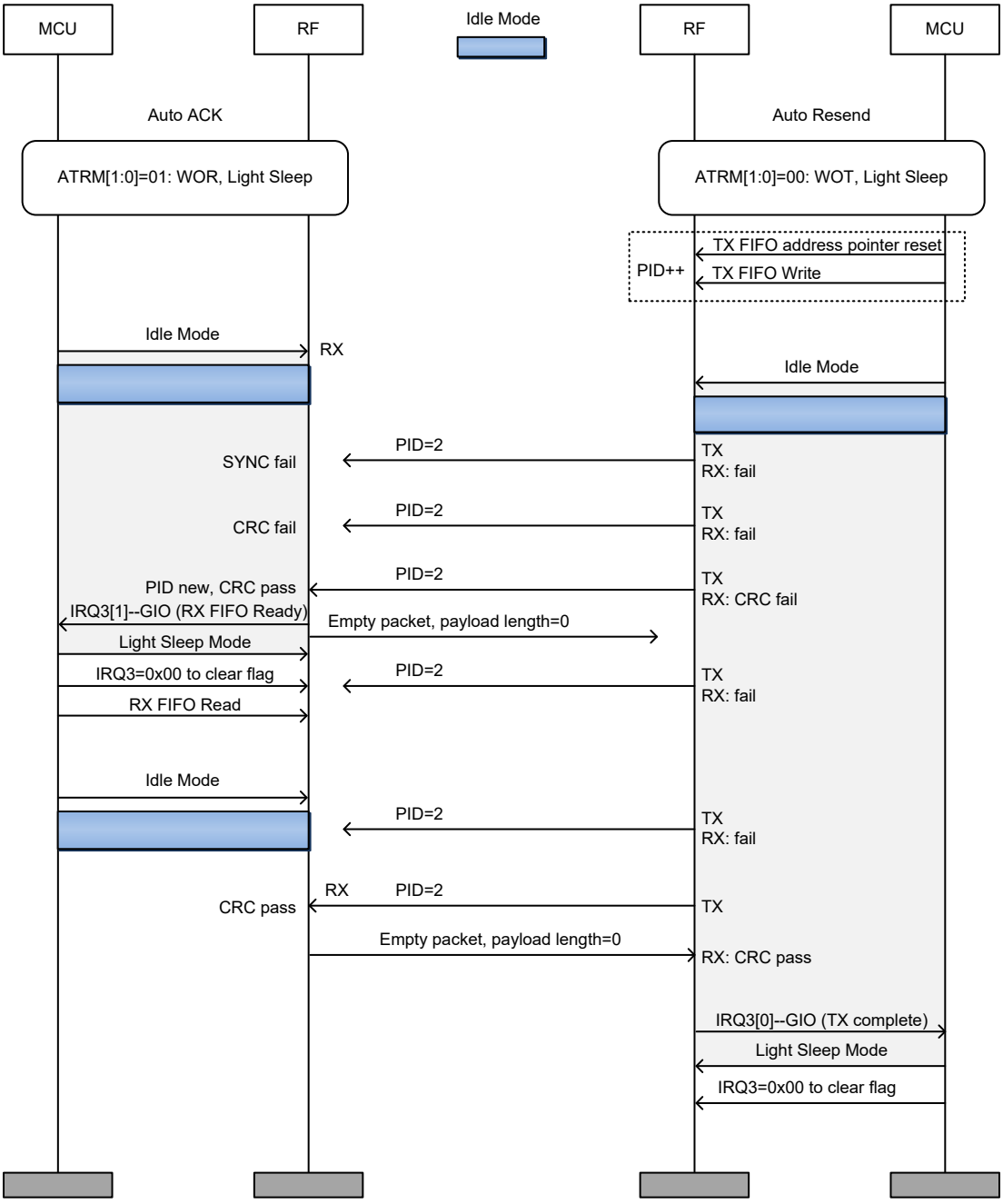
The RF can be set as a programmable timer to output a periodical waveform on GIOs. User can use this signal to wake up the CPU. Set ATR_EN=1 and ATRM=10b/11b to enable the WTM mode. The device starts the WTM mode upon receiving the Idle strobe command from the MCU and stops the WTM mode upon receiving the Light Sleep strobe command. The device will stay in the Idle mode for the whole WTM process.

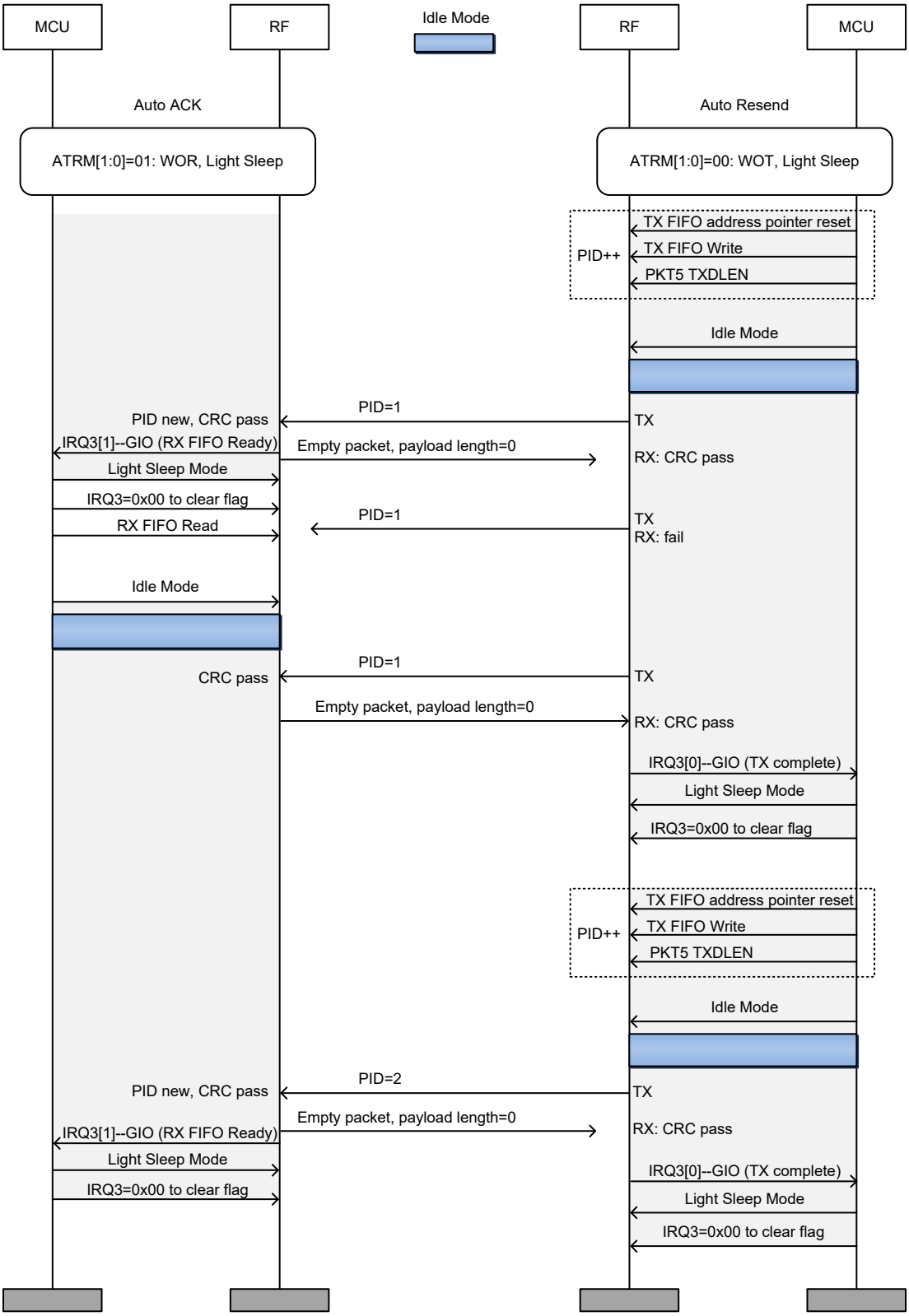
Message Flowchart Examples

ATR: WOT & WOR



ATR+ARK: WOT+Auto-Resend & WOR+Auto-Ack




MCU
RF
Idle Mode
RF
MCU
Auto ACK
ATRM[1:0]=01: WOR, Light Sleep
Auto Resend
ATRM[1:0]=00: WOT, Light Sleep
TX FIFO address pointer reset
PID++
TX FIFO Write
PKT5 TXDLEN
Idle Mode
PID=1
TX
PID new, CRC pass
IRQ3[1]--GIO (RX FIFO Ready)
Empty packet, payload length=0
RX: CRC pass
Light Sleep Mode
IRQ3=0x00 to clear flag
PID=1
TX
RX FIFO Read
RX: fail
Idle Mode
PID=1
TX
CRC pass
Empty packet, payload length=0
RX: CRC pass
IRQ3[0]--GIO (TX complete)
Light Sleep Mode
IRQ3=0x00 to clear flag
TX FIFO address pointer reset
PID++
TX FIFO Write
PKT5 TXDLEN
Idle Mode
PID=2
TX
PID new, CRC pass
Empty packet, payload length=0
IRQ3[1]--GIO (RX FIFO Ready)
RX: CRC pass
Light Sleep Mode
IRQ3[0]--GIO (TX complete)
IRQ3=0x00 to clear flag
Light Sleep Mode
IRQ3=0x00 to clear flag

Abbreviation

ADC: Analog to Digital Converter

AFC: Automatic Frequency Compensation

AGC: Automatic Gain Control

ARK: Auto-Resend and Auto-Ack

ATR: Automatic-Transmit-Receive

BER: Bit Error Rate

BPF: Band Pass Filter

CD: Carrier Detect

CFO: Carrier Frequency Offset

CP: Charge Pump

CRC: Cyclic Redundancy Check

DCOC: DC Offset Correct

DSM: Delta Sigma Modulator

FEC: Forward Error Correction

FIFO: First In First Out

GFSK: Gaussian Frequency Shift Keying

HPF: High Pass Filter

ID: Identifier

IF: Intermedia Frequency

IIR: Infinite Impulse Response

IRQ: Interrupt Request

ISM: Industrial, Scientific and Medical

LNA: Low Noise Amplifier

LO: Local Oscillator

LPF: Low Pass Filter

MCU: Mico Controller Unit

MMD: Multi-Mode Divider

OW: Overwrite

PA: Power Amplifier

PD: Power Down

PFD: Phase Frequency Detector (for PLL)

PLL: Phase Lock Loop

POR: Power On Reset

PVT: Process-Voltage-Temperature

RBCLK: RX Bit Clock

RSSI: Received Signal Strength Indicator

RX: Receiver

SNR: Signal Noise Ratio

SPI: Serial Port Interface

SX: Synthesizer

SYCK: System Clock for digital circuit

SYNC/SYNCWORD: Synchronization Word

TBCLK: TX Bit Clock

TRX: TX/RX

TX: Transmitter

VCO: Voltage Controlled Oscillator

WOR: Wake-on-RX

WOT: Wake-on-TX

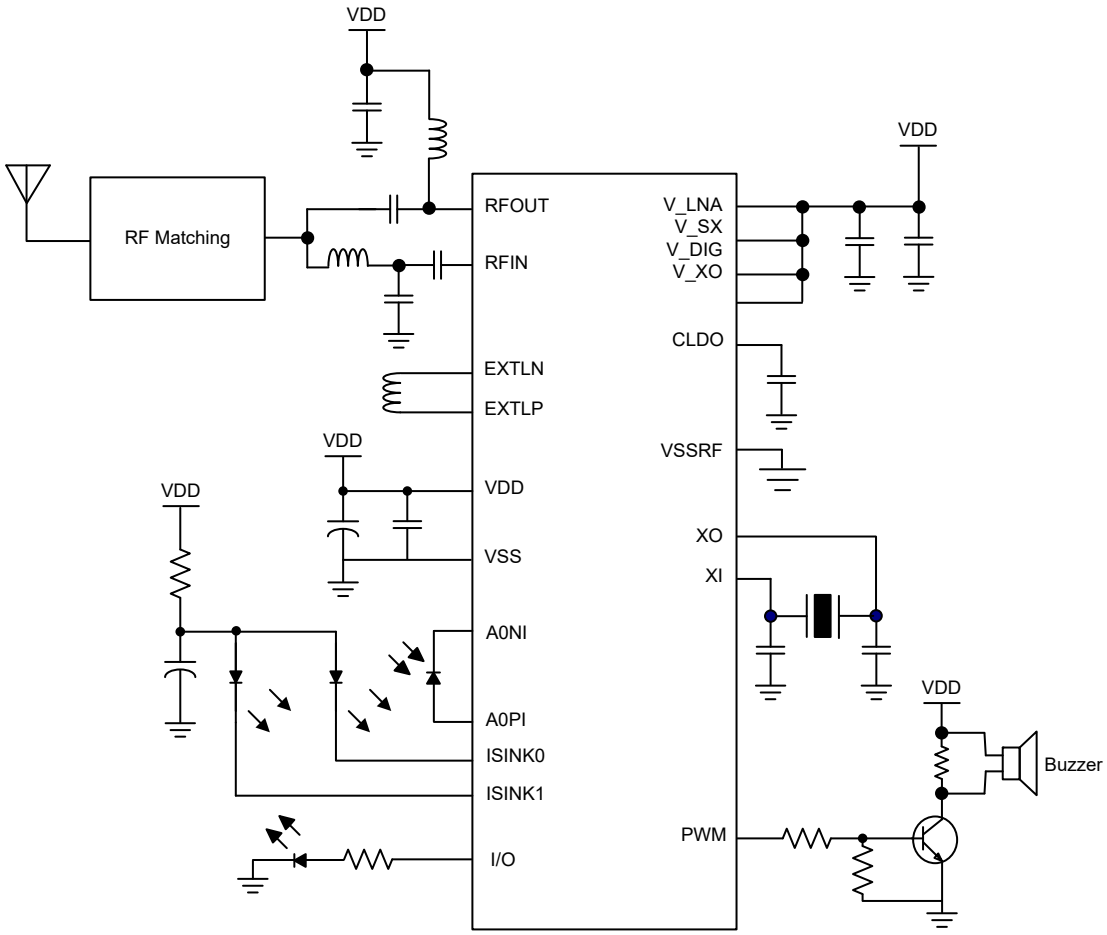
WTM: Wake-up Timer Mode

XCLK: Crystal Clock

XO/XOSC: Crystal Oscillator

XTAL: Crystal

Application Circuits



Instruction Set

Introduction

Central to the successful operation of any microcontroller is its instruction set, which is a set of program instruction codes that directs the microcontroller to perform certain operations. In the case of Holtek microcontroller, a comprehensive and flexible set of over 60 instructions is provided to enable programmers to implement their application with the minimum of programming overheads.

For easier understanding of the various instruction codes, they have been subdivided into several functional groupings.

Instruction Timing

Most instructions are implemented within one instruction cycle. The exceptions to this are branch, call, or table read instructions where two instruction cycles are required. One instruction cycle is equal to 4 system clock cycles, therefore in the case of an 8MHz system oscillator, most instructions would be implemented within 0.5μs and branch or call instructions would be implemented within 1μs. Although instructions which require one more cycle to implement are generally limited to the JMP, CALL, RET, RETI and table read instructions, it is important to realize that any other instructions which involve manipulation of the Program Counter Low register or PCL will also take one more cycle to implement. As instructions which change the contents of the PCL will imply a direct jump to that new address, one more cycle will be required. Examples of such instructions would be "CLR PCL" or "MOV PCL, A". For the case of skip instructions, it must be noted that if the result of the comparison involves a skip operation then this will also take one more cycle, if no skip is involved then only one cycle is required.

Moving and Transferring Data

The transfer of data within the microcontroller program is one of the most frequently used operations. Making use of three kinds of MOV instructions, data can be transferred from registers to the Accumulator and vice-versa as well as being able to move specific immediate data directly into the Accumulator. One of the most important data transfer applications is to receive data from the input ports and transfer data to the output ports.

Arithmetic Operations

The ability to perform certain arithmetic operations and data manipulation is a necessary feature of most microcontroller applications. Within the Holtek microcontroller instruction set are a range of add and subtract instruction mnemonics to enable the necessary arithmetic to be carried out. Care must be taken to ensure correct handling of carry and borrow data when results exceed 255 for addition and less than 0 for subtraction. The increment and decrement instructions INC, INCA, DEC and DECA provide a simple means of increasing or decreasing by a value of one of the values in the destination specified.

Logical and Rotate Operation

The standard logical operations such as AND, OR, XOR and CPL all have their own instruction within the Holtek microcontroller instruction set. As with the case of most instructions involving data manipulation, data must pass through the Accumulator which may involve additional programming steps. In all logical data operations, the zero flag may be set if the result of the operation is zero. Another form of logical data manipulation comes from the rotate instructions such as RR, RL, RRC and RLC which provide a simple means of rotating one Bit right or left. Different rotate instructions exist depending on program requirements. Rotate instructions are useful for serial port programming applications where data can be rotated from an internal register into the Carry Bit from where it can be examined and the necessary serial Bit set high or low. Another application which rotate data operations are used is to implement multiplication and division calculations.

Branches and Control Transfer

Program branching takes the form of either jumps to specified locations using the JMP instruction or to a subroutine using the CALL instruction. They differ in the sense that in the case of a subroutine call, the program must return to the instruction immediately when the subroutine has been carried out. This is done by placing a return instruction "RET" in the subroutine which will cause the program to jump back to the address right after the CALL instruction. In the case of a JMP instruction, the program simply jumps to the desired location. There is no requirement to jump back to the original jumping off point as in the case of the CALL instruction. One special and extremely useful set of branch instructions are the conditional branches. Here a decision is first made regarding the condition of a certain data memory or individual Bits. Depending upon the conditions, the program will continue with the next instruction or skip over it and jump to the following instruction. These instructions are the key to decision making and branching within the program perhaps determined by the condition of certain input switches or by the condition of internal data Bits.

Bit Operations

The ability to provide single Bit operations on Data Memory is an extremely flexible feature of all Holtek microcontrollers. This feature is especially useful for output port Bit programming where individual Bits or port pins can be directly set high or low using either the "SET [m].i" or "CLR [m].i" instructions respectively. The feature removes the need for programmers to first read the 8-Bit output port, manipulate the input data to ensure that other Bits are not changed and then output the port with the correct new data. This read-modify-write process is taken care of automatically when these Bit operation instructions are used.

Table Read Operations

Data storage is normally implemented by using registers. However, when working with large amounts of fixed data, the volume involved often makes it inconvenient to store the fixed data in the Data Memory. To overcome this problem, Holtek microcontrollers allow an area of Program Memory to be set as a table where data can be directly stored. A set of easy to use instructions provides the means by which this fixed data can be referenced and retrieved from the Program Memory.

Other Operations

In addition to the above functional instructions, a range of other instructions also exist such as the “HALT” instruction for Power-down operations and instructions to control the operation of the Watchdog Timer for reliable program operations under extreme electric or electromagnetic environments. For their relevant operations, refer to the functional related sections.

Instruction Set Summary

The instructions related to the data memory access in the following table can be used when the desired data memory is located in Data Memory sector 0.

Table Conventions

x: Bits immediate data
m: Data Memory address
A: Accumulator
i: 0~7 number of bits
addr: Program memory address

Mnemonic	Description	Cycles	Flag Affected
Arithmetic			
ADD A,[m]	Add Data Memory to ACC	1	Z, C, AC, OV, SC
ADDM A,[m]	Add ACC to Data Memory	1Note	Z, C, AC, OV, SC
ADD A,x	Add immediate data to ACC	1	Z, C, AC, OV, SC
ADC A,[m]	Add Data Memory to ACC with Carry	1	Z, C, AC, OV, SC
ADCM A,[m]	Add ACC to Data memory with Carry	1Note	Z, C, AC, OV, SC
SUB A,x	Subtract immediate data from the ACC	1	Z, C, AC, OV, SC, CZ
SUB A,[m]	Subtract Data Memory from ACC	1	Z, C, AC, OV, SC, CZ
SUBM A,[m]	Subtract Data Memory from ACC with result in Data Memory	1Note	Z, C, AC, OV, SC, CZ
SBC A,x	Subtract immediate data from ACC with Carry	1	Z, C, AC, OV, SC, CZ
SBC A,[m]	Subtract Data Memory from ACC with Carry	1	Z, C, AC, OV, SC, CZ
SBCM A,[m]	Subtract Data Memory from ACC with Carry, result in Data Memory	1Note	Z, C, AC, OV, SC, CZ
DAA [m]	Decimal adjust ACC for Addition with result in Data Memory	1Note	C
Logic Operation			
AND A,[m]	Logical AND Data Memory to ACC	1	Z
OR A,[m]	Logical OR Data Memory to ACC	1	Z
XOR A,[m]	Logical XOR Data Memory to ACC	1	Z
ANDM A,[m]	Logical AND ACC to Data Memory	1Note	Z
ORM A,[m]	Logical OR ACC to Data Memory	1Note	Z
XORM A,[m]	Logical XOR ACC to Data Memory	1Note	Z
AND A,x	Logical AND immediate Data to ACC	1	Z
OR A,x	Logical OR immediate Data to ACC	1	Z
XOR A,x	Logical XOR immediate Data to ACC	1	Z
CPL [m]	Complement Data Memory	1Note	Z
CPLA [m]	Complement Data Memory with result in ACC	1	Z
Increment & Decrement			
INCA [m]	Increment Data Memory with result in ACC	1	Z
INC [m]	Increment Data Memory	1Note	Z
DECA [m]	Decrement Data Memory with result in ACC	1	Z
DEC [m]	Decrement Data Memory	1Note	Z
Rotate			
RRA [m]	Rotate Data Memory right with result in ACC	1	None
RR [m]	Rotate Data Memory right	1Note	None
RRCA [m]	Rotate Data Memory right through Carry with result in ACC	1	C
RRC [m]	Rotate Data Memory right through Carry	1Note	C
RLA [m]	Rotate Data Memory left with result in ACC	1	None
RL [m]	Rotate Data Memory left	1Note	None
RLCA [m]	Rotate Data Memory left through Carry with result in ACC	1	C
RLC [m]	Rotate Data Memory left through Carry	1Note	C

Mnemonic	Description	Cycles	Flag Affected
Data Move			
MOV A,[m]	Move Data Memory to ACC	1	None
MOV [m],A	Move ACC to Data Memory	1Note	None
MOV A,x	Move immediate data to ACC	1	None
Bit Operation			
CLR [m].i	Clear bit of Data Memory	1Note	None
SET [m].i	Set bit of Data Memory	1Note	None
Branch Operation			
JMP addr	Jump unconditionally	2	None
SZ [m]	Skip if Data Memory is zero	1Note	None
SZA [m]	Skip if Data Memory is zero with data movement to ACC	1Note	None
SZ [m].i	Skip if bit i of Data Memory is zero	1Note	None
SNZ [m]	Skip if Data Memory is not zero	1Note	None
SNZ [m].i	Skip if bit i of Data Memory is not zero	1Note	None
SIZ [m]	Skip if increment Data Memory is zero	1Note	None
SDZ [m]	Skip if decrement Data Memory is zero	1Note	None
SIZA [m]	Skip if increment Data Memory is zero with result in ACC	1Note	None
SDZA [m]	Skip if decrement Data Memory is zero with result in ACC	1Note	None
CALL addr	Subroutine call	2	None
RET	Return from subroutine	2	None
RET A,x	Return from subroutine and load immediate data to ACC	2	None
RETI	Return from interrupt	2	None
Table Read Operation			
TABRD [m]	Read table (specific page) to TBLH and Data Memory	2Note	None
TABRDL [m]	Read table (last page) to TBLH and Data Memory	2Note	None
ITABRD [m]	Increment table pointer TBLP first and Read table to TBLH and Data Memory	2Note	None
ITABRDL [m]	Increment table pointer TBLP first and Read table (last page) to TBLH and Data Memory	2Note	None
Miscellaneous			
NOP	No operation	1	None
CLR [m]	Clear Data Memory	1Note	None
SET [m]	Set Data Memory	1Note	None
CLR WDT	Clear Watchdog Timer	1	TO, PDF
SWAP [m]	Swap nibbles of Data Memory	1Note	None
SWAPA [m]	Swap nibbles of Data Memory with result in ACC	1	None
HALT	Enter power down mode	1	TO, PDF

Note: 1. For skip instructions, if the result of the comparison involves a skip then up to three cycles are required, if no skip takes place only one cycle is required.

2. Any instruction which changes the contents of the PCL will also require 2 cycles for execution.

3. For the “CLR WDT” instruction the TO and PDF flags may be affected by the execution status. The TO and PDF flags are cleared after the “CLR WDT” instructions is executed. Otherwise the TO and PDF flags remain unchanged.

Extended Instruction Set

The extended instructions are used to support the full range address access for the data memory. When the accessed data memory is located in any data memory sector except sector 0, the extended instruction can be used to directly access the data memory instead of using the indirect addressing access. This can not only reduce the use of Flash memory space but also improve the CPU execution efficiency.

Mnemonic	Description	Cycles	Flag Affected
Arithmetic			
LADD A,[m]	Add Data Memory to ACC	2	Z, C, AC, OV, SC
LADDM A,[m]	Add ACC to Data Memory	2Note	Z, C, AC, OV, SC
LADC A,[m]	Add Data Memory to ACC with Carry	2	Z, C, AC, OV, SC
LADCM A,[m]	Add ACC to Data memory with Carry	2Note	Z, C, AC, OV, SC
LSUB A,[m]	Subtract Data Memory from ACC	2	Z, C, AC, OV, SC, CZ
LSUBM A,[m]	Subtract Data Memory from ACC with result in Data Memory	2Note	Z, C, AC, OV, SC, CZ
LSBC A,[m]	Subtract Data Memory from ACC with Carry	2	Z, C, AC, OV, SC, CZ
LSBCM A,[m]	Subtract Data Memory from ACC with Carry, result in Data Memory	2Note	Z, C, AC, OV, SC, CZ
LDAA [m]	Decimal adjust ACC for Addition with result in Data Memory	2Note	C
Logic Operation			
LAND A,[m]	Logical AND Data Memory to ACC	2	Z
LOR A,[m]	Logical OR Data Memory to ACC	2	Z
LXOR A,[m]	Logical XOR Data Memory to ACC	2	Z
LANDM A,[m]	Logical AND ACC to Data Memory	2Note	Z
LORM A,[m]	Logical OR ACC to Data Memory	2Note	Z
LXORM A,[m]	Logical XOR ACC to Data Memory	2Note	Z
LCPL [m]	Complement Data Memory	2Note	Z
LCPLA [m]	Complement Data Memory with result in ACC	2	Z
Increment & Decrement			
LINCA [m]	Increment Data Memory with result in ACC	2	Z
LINC [m]	Increment Data Memory	2Note	Z
LDECA [m]	Decrement Data Memory with result in ACC	2	Z
LDEC [m]	Decrement Data Memory	2Note	Z
Rotate			
LRRA [m]	Rotate Data Memory right with result in ACC	2	None
LRR [m]	Rotate Data Memory right	2Note	None
LRRCA [m]	Rotate Data Memory right through Carry with result in ACC	2	C
LRRC [m]	Rotate Data Memory right through Carry	2Note	C
LRLA [m]	Rotate Data Memory left with result in ACC	2	None
LRL [m]	Rotate Data Memory left	2Note	None
LRLCA [m]	Rotate Data Memory left through Carry with result in ACC	2	C
LRLC [m]	Rotate Data Memory left through Carry	2Note	C
Data Move			
LMOV A,[m]	Move Data Memory to ACC	2	None
LMOV [m],A	Move ACC to Data Memory	2Note	None
Bit Operation			
LCLR [m].i	Clear bit of Data Memory	2Note	None
LSET [m].i	Set bit of Data Memory	2Note	None

Mnemonic	Description	Cycles	Flag Affected
Branch			
LSZ [m]	Skip if Data Memory is zero	2Note	None
LSZA [m]	Skip if Data Memory is zero with data movement to ACC	2Note	None
LSNZ [m]	Skip if Data Memory is not zero	2Note	None
LSZ [m].i	Skip if bit i of Data Memory is zero	2Note	None
LSNZ [m].i	Skip if bit i of Data Memory is not zero	2Note	None
LSIZ [m]	Skip if increment Data Memory is zero	2Note	None
LSDZ [m]	Skip if decrement Data Memory is zero	2Note	None
LSIZA [m]	Skip if increment Data Memory is zero with result in ACC	2Note	None
LSDZA [m]	Skip if decrement Data Memory is zero with result in ACC	2Note	None
Table Read			
LTABRD [m]	Read table to TBLH and Data Memory	3Note	None
LTABRDL [m]	Read table (last page) to TBLH and Data Memory	3Note	None
LITABRD [m]	Increment table pointer TBLP first and Read table to TBLH and Data Memory	3Note	None
LITABRDL [m]	Increment table pointer TBLP first and Read table (last page) to TBLH and Data Memory	3Note	None
Miscellaneous			
LCLR [m]	Clear Data Memory	2Note	None
LSET [m]	Set Data Memory	2Note	None
LSWAP [m]	Swap nibbles of Data Memory	2Note	None
LSWAPA [m]	Swap nibbles of Data Memory with result in ACC	2	None

Note: 1. For these extended skip instructions, if the result of the comparison involves a skip then up to four cycles are required, if no skip takes place two cycles is required.

2. Any extended instruction which changes the contents of the PCL register will also require three cycles for execution.

Instruction Definition

ADC A,[m] Add Data Memory to ACC with Carry

Description The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the Accumulator.

Operation ACC ← ACC + [m] + C

Affected flag(s) OV, Z, AC, C, SC

ADCM A,[m] Add ACC to Data Memory with Carry

Description The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the specified Data Memory.

Operation [m] ← ACC + [m] + C

Affected flag(s) OV, Z, AC, C, SC

ADD A,[m] Add Data Memory to ACC

Description The contents of the specified Data Memory and the Accumulator are added. The result is stored in the Accumulator.

Operation ACC ← ACC + [m]

Affected flag(s) OV, Z, AC, C, SC

ADD A,x Add immediate data to ACC

Description The contents of the Accumulator and the specified immediate data are added. The result is stored in the Accumulator.

Operation ACC ← ACC + x

Affected flag(s) OV, Z, AC, C, SC

ADDM A,[m] Add ACC to Data Memory

Description The contents of the specified Data Memory and the Accumulator are added. The result is stored in the specified Data Memory.

Operation [m] ← ACC + [m]

Affected flag(s) OV, Z, AC, C, SC

AND A,[m] Logical AND Data Memory to ACC

Description Data in the Accumulator and the specified Data Memory perform a bitwise logical AND operation. The result is stored in the Accumulator.

Operation ACC ← ACC "AND" [m]

Affected flag(s) Z

AND A,x Logical AND immediate data to ACC

Description Data in the Accumulator and the specified immediate data perform a bit wise logical AND operation. The result is stored in the Accumulator.

Operation ACC ← ACC "AND" x

Affected flag(s) Z

ANDM A,[m] Logical AND ACC to Data Memory

Description Data in the specified Data Memory and the Accumulator perform a bitwise logical AND operation. The result is stored in the Data Memory.

Operation [m] ← ACC "AND" [m]

Affected flag(s) Z

CALL addr Subroutine call

Description: Unconditionally calls a subroutine at the specified address. The Program Counter then increments by 1 to obtain the address of the next instruction which is then pushed onto the stack. The specified address is then loaded and the program continues execution from this new address. As this instruction requires an additional operation, it is a two cycle instruction.

Operation: Stack ← Program Counter + 1
Program Counter ← addr

Affected flag(s): None

CLR [m] Clear Data Memory

Description: Each bit of the specified Data Memory is cleared to 0.

Operation: [m] ← 00H

Affected flag(s): None

CLR [m].i Clear bit of Data Memory

Description: Bit i of the specified Data Memory is cleared to 0.

Operation: [m].i ← 0

Affected flag(s): None

CLR WDT Clear Watchdog Timer

Description: The TO, PDF flags and the WDT are all cleared.

Operation: WDT cleared
TO ← 0
PDF ← 0

Affected flag(s): TO, PDF

CPL [m] Complement Data Memory

Description: Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa.

Operation: [m] ← $\overline{[m]}$

Affected flag(s): Z

CPLA [m] Complement Data Memory with result in ACC

Description: Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa. The complemented result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation: ACC ← $\overline{[m]}$

Affected flag(s): Z

DAA [m] Decimal-Adjust ACC for addition with result in Data Memory

Description: Convert the contents of the Accumulator value to a BCD (Binary Coded Decimal) value resulting from the previous addition of two BCD variables. If the low nibble is greater than 9 or if AC flag is set, then a value of 6 will be added to the low nibble. Otherwise the low nibble remains unchanged. If the high nibble is greater than 9 or if the C flag is set, then a value of 6 will be added to the high nibble. Essentially, the decimal conversion is performed by adding 00H, 06H, 60H or 66H depending on the Accumulator and flag conditions. Only the C flag may be affected by this instruction which indicates that if the original BCD sum is greater than 100, it allows multiple precision decimal addition.

Operation: [m] ← ACC + 00H or
[m] ← ACC + 06H or
[m] ← ACC + 60H or
[m] ← ACC + 66H

Affected flag(s): C

DEC [m] Decrement Data Memory

Description: Data in the specified Data Memory is decremented by 1.

Operation: [m] ← [m] − 1

Affected flag(s): Z

DECA [m] Decrement Data Memory with result in ACC

Description: Data in the specified Data Memory is decremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation: ACC ← [m] − 1

Affected flag(s): Z

HALT Enter power down mode

Description: This instruction stops the program execution and turns off the system clock. The contents of the Data Memory and registers are retained. The WDT and prescaler are cleared. The power down flag PDF is set and the WDT time-out flag TO is cleared.

Operation: TO ← 0
PDF ← 1

Affected flag(s): TO, PDF

INC [m] Increment Data Memory

Description: Data in the specified Data Memory is incremented by 1.

Operation: [m] ← [m] + 1

Affected flag(s): Z

INCA [m] Increment Data Memory with result in ACC

Description: Data in the specified Data Memory is incremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation: ACC ← [m] + 1

Affected flag(s): Z

JMP addr Jump unconditionally

Description: The contents of the Program Counter are replaced with the specified address. Program execution then continues from this new address. As this requires the insertion of a dummy instruction while the new address is loaded, it is a two cycle instruction.

Operation: Program Counter ← addr

Affected flag(s): None

MOV A,[m] Move Data Memory to ACC

Description: The contents of the specified Data Memory are copied to the Accumulator.

Operation: ACC ← [m]

Affected flag(s): None

MOV A,x Move immediate data to ACC

Description: The immediate data specified is loaded into the Accumulator.

Operation: ACC ← x

Affected flag(s): None

MOV [m],A Move ACC to Data Memory

Description: The contents of the Accumulator are copied to the specified Data Memory.

Operation: [m] ← ACC

Affected flag(s): None

NOP No operation

Description No operation is performed. Execution continues with the next instruction.

Operation No operation

Affected flag(s) None

OR A,[m] Logical OR Data Memory to ACC

Description Data in the Accumulator and the specified Data Memory perform a bitwise logical OR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″OR″ [m]

Affected flag(s) Z

OR A,x Logical OR immediate data to ACC

Description Data in the Accumulator and the specified immediate data perform a bitwise logical OR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″OR″ x

Affected flag(s) Z

ORM A,[m] Logical OR ACC to Data Memory

Description Data in the specified Data Memory and the Accumulator perform a bitwise logical OR operation. The result is stored in the Data Memory.

Operation [m] ← ACC ″OR″ [m]

Affected flag(s) Z

RET Return from subroutine

Description The Program Counter is restored from the stack. Program execution continues at the restored address.

Operation Program Counter ← Stack

Affected flag(s) None

RET A,x Return from subroutine and load immediate data to ACC

Description The Program Counter is restored from the stack and the Accumulator loaded with the specified immediate data. Program execution continues at the restored address.

Operation Program Counter ← Stack
ACC ← x

Affected flag(s) None

RETI Return from interrupt

Description The Program Counter is restored from the stack and the interrupts are re-enabled by setting the EMI bit. EMI is the master interrupt global enable bit. If an interrupt was pending when the RETI instruction is executed, the pending Interrupt routine will be processed before returning to the main program.

Operation Program Counter ← Stack
EMI ← 1

Affected flag(s) None

RL [m] Rotate Data Memory left

Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0.

Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← [m].7

Affected flag(s) None

RLA [m] Rotate Data Memory left with result in ACC

Description	The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.
Operation	ACC.(i+1) ← [m].i; (i=0~6) ACC.0 ← [m].7
Affected flag(s)	None

RLC [m] Rotate Data Memory left through Carry

Description	The contents of the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into bit 0.
Operation	[m].(i+1) ← [m].i; (i=0~6) [m].0 ← C C ← [m].7
Affected flag(s)	C

RLCA [m] Rotate Data Memory left through Carry with result in ACC

Description	Data in the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into the bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.
Operation	ACC.(i+1) ← [m].i; (i=0~6) ACC.0 ← C C ← [m].7
Affected flag(s)	C

RR [m] Rotate Data Memory right

Description	The contents of the specified Data Memory are rotated right by 1 bit with bit 0 rotated into bit 7.
Operation	[m].i ← [m].(i+1); (i=0~6) [m].7 ← [m].0
Affected flag(s)	None

RRA [m] Rotate Data Memory right with result in ACC

Description	Data in the specified Data Memory is rotated right by 1 bit with bit 0 rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.
Operation	ACC.i ← [m].(i+1); (i=0~6) ACC.7 ← [m].0
Affected flag(s)	None

RRC [m] Rotate Data Memory right through Carry

Description	The contents of the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7.
Operation	[m].i ← [m].(i+1); (i=0~6) [m].7 ← C C ← [m].0
Affected flag(s)	C

RRCA [m] Rotate Data Memory right through Carry with result in ACC

Description: Data in the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation:
ACC.i ← [m].(i+1); (i=0~6)
ACC.7 ← C
C ← [m].0

Affected flag(s): C

SBC A,[m] Subtract Data Memory from ACC with Carry

Description: The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation: ACC ← ACC − [m] − $\overline{C}$

Affected flag(s): OV, Z, AC, C, SC, CZ

SBC A, x Subtract immediate data from ACC with Carry

Description: The immediate data and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation: ACC ← ACC - [m] – $\overline{C}$

Affected flag(s): OV, Z, AC, C, SC, CZ

SBCM A,[m] Subtract Data Memory from ACC with Carry and result in Data Memory

Description: The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation: [m] ← ACC − [m] − $\overline{C}$

Affected flag(s): OV, Z, AC, C, SC, CZ

SDZ [m] Skip if decrement Data Memory is 0

Description: The contents of the specified Data Memory are first decremented by 1. If the result is 0 the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation:
[m] ← [m] − 1
Skip if [m]=0

Affected flag(s): None

SDZA [m] Skip if decrement Data Memory is zero with result in ACC

Description: The contents of the specified Data Memory are first decremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.

Operation:
ACC ← [m] − 1
Skip if ACC=0

Affected flag(s): None

SET [m] Set Data Memory

Description Each bit of the specified Data Memory is set to 1.

Operation [m] ← FFH

Affected flag(s) None

SET [m].i Set bit of Data Memory

Description Bit i of the specified Data Memory is set to 1.

Operation [m].i ← 1

Affected flag(s) None

SIZ [m] Skip if increment Data Memory is 0

Description The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation [m] ← [m] + 1
Skip if [m]=0

Affected flag(s) None

SIZA [m] Skip if increment Data Memory is zero with result in ACC

Description The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation ACC ← [m] + 1
Skip if ACC=0

Affected flag(s) None

SNZ [m].i Skip if Data Memory is not 0

Description If the specified Data Memory is not 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is 0 the program proceeds with the following instruction.

Operation Skip if [m].i ≠ 0

Affected flag(s) None

SNZ [m] Skip if Data Memory is not 0

Description If the specified Data Memory is not 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is 0 the program proceeds with the following instruction.

Operation Skip if [m]≠ 0

Affected flag(s) None

SUB A,[m] Subtract Data Memory from ACC

Description The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC − [m]

Affected flag(s) OV, Z, AC, C, SC, CZ

SUBM A,[m] Subtract Data Memory from ACC with result in Data Memory

Description The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation [m] ← ACC − [m]

Affected flag(s) OV, Z, AC, C, SC, CZ

SUB A,x Subtract immediate data from ACC

Description The immediate data specified by the code is subtracted from the contents of the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC − x

Affected flag(s) OV, Z, AC, C, SC, CZ

SWAP [m] Swap nibbles of Data Memory

Description The low-order and high-order nibbles of the specified Data Memory are interchanged.

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s) None

SWAPA [m] Swap nibbles of Data Memory with result in ACC

Description The low-order and high-order nibbles of the specified Data Memory are interchanged. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC.3~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0

Affected flag(s) None

SZ [m] Skip if Data Memory is 0

Description If the contents of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation Skip if [m]=0

Affected flag(s) None

SZA [m] Skip if Data Memory is 0 with data movement to ACC

Description The contents of the specified Data Memory are copied to the Accumulator. If the value is zero, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation ACC ← [m]
Skip if [m]=0

Affected flag(s) None

SZ [m].i Skip if bit i of Data Memory is 0

Description If bit i of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.

Operation Skip if [m].i=0

Affected flag(s) None

TABRD [m] Read table (specific page or current page) to TBLH and Data Memory

Description The low byte of the program code (specific page) addressed by the table pointer pair (TBHP and TBLP or only TBLP if no TBHP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

TABRDL [m] Read table (last page) to TBLH and Data Memory

Description The low byte of the program code (last page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

ITABRD [m] Increment table pointer low byte first and read table to TBLH and Data Memory

Description Increment table pointer low byte, TBLP, first and then the program code addressed by the table pointer (TBHP and TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

ITABRDL [m] Increment table pointer low byte first and read table (last page) to TBLH and Data Memory

Description Increment table pointer low byte, TBLP, first and then the low byte of the program code (last page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

XOR A,[m] Logical XOR Data Memory to ACC

Description Data in the Accumulator and the specified Data Memory perform a bitwise logical XOR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″XOR″ [m]

Affected flag(s) Z

XORM A,[m] Logical XOR ACC to Data Memory

Description Data in the specified Data Memory and the Accumulator perform a bitwise logical XOR operation. The result is stored in the Data Memory.

Operation [m] ← ACC ″XOR″ [m]

Affected flag(s) Z

XOR A,x Logical XOR immediate data to ACC

Description Data in the Accumulator and the specified immediate data perform a bitwise logical XOR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″XOR″ x

Affected flag(s) Z

Extended Instruction Definition

The extended instructions are used to directly access the data stored in any data memory sections.

LADC A,[m] Add Data Memory to ACC with Carry

Description The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the Accumulator.

Operation ACC ← ACC + [m] + C

Affected flag(s) OV, Z, AC, C, SC

LADCM A,[m] Add ACC to Data Memory with Carry

Description The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the specified Data Memory.

Operation [m] ← ACC + [m] + C

Affected flag(s) OV, Z, AC, C, SC

LADD A,[m] Add Data Memory to ACC

Description The contents of the specified Data Memory and the Accumulator are added. The result is stored in the Accumulator.

Operation ACC ← ACC + [m]

Affected flag(s) OV, Z, AC, C, SC

LADDM A,[m] Add ACC to Data Memory

Description The contents of the specified Data Memory and the Accumulator are added. The result is stored in the specified Data Memory.

Operation [m] ← ACC + [m]

Affected flag(s) OV, Z, AC, C, SC

LAND A,[m] Logical AND Data Memory to ACC

Description Data in the Accumulator and the specified Data Memory perform a bitwise logical AND operation. The result is stored in the Accumulator.

Operation ACC ← ACC "AND" [m]

Affected flag(s) Z

LANDM A,[m] Logical AND ACC to Data Memory

Description Data in the specified Data Memory and the Accumulator perform a bitwise logical AND operation. The result is stored in the Data Memory.

Operation [m] ← ACC "AND" [m]

Affected flag(s) Z

LCLR [m] Clear Data Memory

Description Each bit of the specified Data Memory is cleared to 0.

Operation [m] ← 00H

Affected flag(s) None

LCLR [m].i Clear bit of Data Memory

Description Bit i of the specified Data Memory is cleared to 0.

Operation [m].i ← 0

Affected flag(s) None

LCPL [m] Complement Data Memory

Description Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa.

Operation $[m] \leftarrow \overline{[m]}$

Affected flag(s) Z

LCPLA [m] Complement Data Memory with result in ACC

Description Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa. The complemented result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation $ACC \leftarrow \overline{[m]}$

Affected flag(s) Z

LDAA [m] Decimal-Adjust ACC for addition with result in Data Memory

Description Convert the contents of the Accumulator value to a BCD (Binary Coded Decimal) value resulting from the previous addition of two BCD variables. If the low nibble is greater than 9 or if AC flag is set, then a value of 6 will be added to the low nibble. Otherwise the low nibble remains unchanged. If the high nibble is greater than 9 or if the C flag is set, then a value of 6 will be added to the high nibble. Essentially, the decimal conversion is performed by adding 00H, 06H, 60H or 66H depending on the Accumulator and flag conditions. Only the C flag may be affected by this instruction which indicates that if the original BCD sum is greater than 100, it allows multiple precision decimal addition.

Operation $[m] \leftarrow ACC + 00H$ or
$[m] \leftarrow ACC + 06H$ or
$[m] \leftarrow ACC + 60H$ or
$[m] \leftarrow ACC + 66H$

Affected flag(s) C

LDEC [m] Decrement Data Memory

Description Data in the specified Data Memory is decremented by 1.

Operation $[m] \leftarrow [m] - 1$

Affected flag(s) Z

LDECA [m] Decrement Data Memory with result in ACC

Description Data in the specified Data Memory is decremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation $ACC \leftarrow [m] - 1$

Affected flag(s) Z

LINC [m] Increment Data Memory

Description Data in the specified Data Memory is incremented by 1.

Operation $[m] \leftarrow [m] + 1$

Affected flag(s) Z

LINCA [m] Increment Data Memory with result in ACC

Description Data in the specified Data Memory is incremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation $ACC \leftarrow [m] + 1$

Affected flag(s) Z

LMOV A,[m] Move Data Memory to ACC

Description The contents of the specified Data Memory are copied to the Accumulator.

Operation ACC ← [m]

Affected flag(s) None

LMOV [m],A Move ACC to Data Memory

Description The contents of the Accumulator are copied to the specified Data Memory.

Operation [m] ← ACC

Affected flag(s) None

LOR A,[m] Logical OR Data Memory to ACC

Description Data in the Accumulator and the specified Data Memory perform a bitwise logical OR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″OR″ [m]

Affected flag(s) Z

LORM A,[m] Logical OR ACC to Data Memory

Description Data in the specified Data Memory and the Accumulator perform a bitwise logical OR operation. The result is stored in the Data Memory.

Operation [m] ← ACC ″OR″ [m]

Affected flag(s) Z

LRL [m] Rotate Data Memory left

Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0.

Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← [m].7

Affected flag(s) None

LRLA [m] Rotate Data Memory left with result in ACC

Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← [m].7

Affected flag(s) None

LRLC [m] Rotate Data Memory left through Carry

Description The contents of the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into bit 0.

Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← C
C ← [m].7

Affected flag(s) C

LRLCA [m] Rotate Data Memory left through Carry with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into the bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← C
C ← [m].7

Affected flag(s) C

LRR [m] Rotate Data Memory right

Description The contents of the specified Data Memory are rotated right by 1 bit with bit 0 rotated into bit 7.

Operation [m].i ← [m].(i+1); (i=0~6)
[m].7 ← [m].0

Affected flag(s) None

LRRA [m] Rotate Data Memory right with result in ACC

Description Data in the specified Data Memory is rotated right by 1 bit with bit 0 rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.i ← [m].(i+1); (i=0~6)
ACC.7 ← [m].0

Affected flag(s) None

LRRC [m] Rotate Data Memory right through Carry

Description The contents of the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7.

Operation [m].i ← [m].(i+1); (i=0~6)
[m].7 ← C
C ← [m].0

Affected flag(s) C

LRRCA [m] Rotate Data Memory right through Carry with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.i ← [m].(i+1); (i=0~6)
ACC.7 ← C
C ← [m].0

Affected flag(s) C

LSBC A,[m] Subtract Data Memory from ACC with Carry

Description The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC − [m] − $\overline{C}$

Affected flag(s) OV, Z, AC, C, SC, CZ

LSBCM A,[m] Subtract Data Memory from ACC with Carry and result in Data Memory

Description The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation [m] ← ACC − [m] − $\overline{C}$

Affected flag(s) OV, Z, AC, C, SC, CZ

LSDZ [m] Skip if decrement Data Memory is 0

Description The contents of the specified Data Memory are first decremented by 1. If the result is 0 the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation [m] ← [m] − 1
Skip if [m]=0

Affected flag(s) None

LSDZA [m] Skip if decrement Data Memory is zero with result in ACC

Description The contents of the specified Data Memory are first decremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.

Operation ACC ← [m] − 1
Skip if ACC=0

Affected flag(s) None

LSET [m] Set Data Memory

Description Each bit of the specified Data Memory is set to 1.

Operation [m] ← FFH

Affected flag(s) None

LSET [m].i Set bit of Data Memory

Description Bit i of the specified Data Memory is set to 1.

Operation [m].i ← 1

Affected flag(s) None

LSIZ [m] Skip if increment Data Memory is 0

Description The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation [m] ← [m] + 1
Skip if [m]=0

Affected flag(s) None

LSIZA [m] Skip if increment Data Memory is zero with result in ACC

Description The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation ACC ← [m] + 1
Skip if ACC=0

Affected flag(s) None

LSNZ [m].i Skip if Data Memory is not 0

Description If the specified Data Memory is not 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is 0 the program proceeds with the following instruction.

Operation Skip if [m].i ≠ 0

Affected flag(s) None

LSNZ [m] Skip if Data Memory is not 0

Description If the content of the specified Data Memory is not 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is 0 the program proceeds with the following instruction.

Operation Skip if [m] ≠ 0

Affected flag(s) None

LSUB A,[m] Subtract Data Memory from ACC

Description The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC − [m]

Affected flag(s) OV, Z, AC, C, SC, CZ

LSUBM A,[m] Subtract Data Memory from ACC with result in Data Memory

Description The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation [m] ← ACC − [m]

Affected flag(s) OV, Z, AC, C, SC, CZ

LSWAP [m] Swap nibbles of Data Memory

Description The low-order and high-order nibbles of the specified Data Memory are interchanged.

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s) None

LSWAPA [m] Swap nibbles of Data Memory with result in ACC

Description The low-order and high-order nibbles of the specified Data Memory are interchanged. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC.3~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0

Affected flag(s) None

LSZ [m] Skip if Data Memory is 0

Description If the contents of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation Skip if [m]=0

Affected flag(s) None

LSZA [m] Skip if Data Memory is 0 with data movement to ACC

Description The contents of the specified Data Memory are copied to the Accumulator. If the value is zero, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation ACC ← [m]
Skip if [m]=0

Affected flag(s) None

LSZ [m].i Skip if bit i of Data Memory is 0

Description If bit i of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.

Operation Skip if [m].i=0

Affected flag(s) None

LTABRD [m] Read table (current page) to TBLH and Data Memory

Description The low byte of the program code (current page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

LTABRDL [m] Read table (last page) to TBLH and Data Memory

Description The low byte of the program code (last page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

LITABRD [m] Increment table pointer low byte first and read table to TBLH and Data Memory

Description Increment table pointer low byte, TBLP, first and then the program code addressed by the table pointer (TBHP and TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

LITABRDL [m] Increment table pointer low byte first and read table (last page) to TBLH and Data Memory

Description Increment table pointer low byte, TBLP, first and then the low byte of the program code (last page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

LXOR A,[m] Logical XOR Data Memory to ACC

Description Data in the Accumulator and the specified Data Memory perform a bitwise logical XOR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″XOR″ [m]

Affected flag(s) Z

LXORM A,[m] Logical XOR ACC to Data Memory

Description Data in the specified Data Memory and the Accumulator perform a bitwise logical XOR operation. The result is stored in the Data Memory.

Operation [m] ← ACC ″XOR″ [m]

Affected flag(s) Z

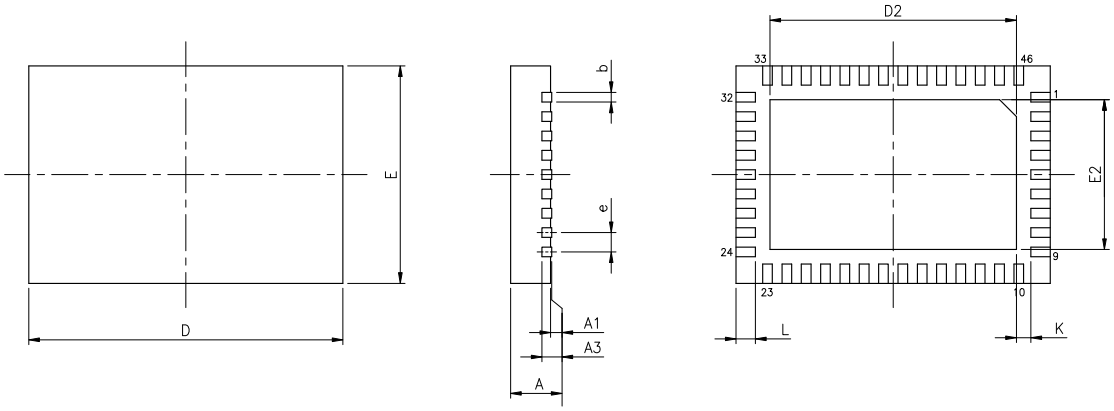
Package Information

Note that the package information provided here is for consultation purposes only. As this information may be updated at regular intervals users are reminded to consult the Holtek website for the latest version of the Package/Carton Information.

Additional supplementary information with regard to packaging is listed below. Click on the relevant section to be transferred to the relevant website page.

- Package Information (include Outline Dimensions, Product Tape and Reel Specifications)
- The Operation Instruction of Packing Materials
- Carton information

SAW Type 46-pin QFN (6.5mm×4.5mm×0.75mm) Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	0.028	0.030	0.031
A1	0.000	0.001	0.002
A3	—	0.008 BSC	—
b	0.006	0.008	0.010
D	—	0.256 BSC	—
E	—	0.177 BSC	—
e	—	0.016 BSC	—
D2	0.199	0.201	0.203
E2	0.120	0.122	0.124
L	0.014	0.016	0.018
K	0.008	—	—

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	0.70	0.75	0.80
A1	0.00	0.02	0.05
A3	—	0.203 BSC	—
b	0.15	0.20	0.25
D	—	6.50 BSC	—
E	—	4.50 BSC	—
e	—	0.40 BSC	—
D2	5.05	5.10	5.15
E2	3.05	3.10	3.15
L	0.35	0.40	0.45
K	0.20	—	—