Linear Charger Touch OTP MCU

BS45R2141

www.holtek.com

Features

CPU Features

- Operating voltage
    - f_{SYS}=8MHz: 2.0V~5.5V
    - f_{SYS}=12MHz: 2.7V~5.5V
    - f_{SYS}=16MHz: 3.3V~5.5V
- Up to 0.25μs instruction cycle with 16MHz system clock at V_{DD}=5V
- Power down and wake-up functions to reduce power consumption
- Oscillator types
    - Internal High Speed 8/12/16MHz 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 one or two instruction cycles
- Table read instructions
- 61 powerful instructions
- 6-level subroutine nesting
- Bit manipulation instruction

Peripheral Features

- OTP Program Memory: 4K×16
- Data Memory: 384×8
- OTP ROM Parameter Program　– ORPP
- Watchdog Timer function
- 10 bidirectional I/O lines
- 2 external interrupt lines shared with I/O pins
- Programmable I/O port source current for LED applications
- Multiple Timer Modules for time measurement, input capture, compare match output, PWM output or single pulse output function
- I²C Interface
- 3 touch key functions
- Dual Time-Base functions for generation of fixed time interrupt signals
- 3 external channel 12-bit resolution A/D converter with internal reference voltage V_{VR}
- Integrated 16-bit Cyclic Redundancy Check function – CRC
- Low voltage reset function
- Input voltage detect function
    - Input maximum rating 20V
    - Over voltage protection
    - Under voltage lockout

- Linear charger

  - Charging current up to 1A programmed through R_{SET} when V_{IN}=5V

  - Charger status indication

  - ±10% charging current regulation in constant current mode

  - ±1% output voltage regulation in constant voltage mode @ 0°C to 85°C

  - Constant temperature regulation

  - Battery reverse current blocking when VIN input is invalid

- 2.0V±1% reference voltage output

- Package type: 16-pin NSOP

General Description

The device is an OTP type 8-bit high performance RISC architecture microcontroller, designed for 3.7V lithium touch personal care products (electric toothbrushes, shavers, etc.) and 3.7V lithium portable touch electronic products.

For memory features, the device is supplied with One-Time Programmable, OTP memory. Other memory includes an area of RAM Data Memory.

Analog features include a multi-channel 12-bit A/D Converter. Multiple Timer Modules for time measurement, input capture, compare match output or PWM output or single pulse output function. Communication with the outside world is catered for by including fully integrated I²C interface function, the popular interface which provides designers with a means of easy communication with external peripheral hardware. Protective features such as an internal Watchdog Timer and Low Voltage Reset coupled with excellent noise immunity and ESD protection ensure that reliable operation is maintained in hostile electrical environments.

A full choice of internal high and low speed oscillators are provided and the two fully integrated system oscillators 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 optimize microcontroller operation and minimize power consumption. The inclusion of flexible I/O programming features, Time-Base funcitons, a CRC and Sink Current Generator along with many other features ensure that the device will find excellent use.

The device includes 3 touch keys which can detect human body contact using external touch pads. The high level of device integration enable applications to be implemented with a minimum number of external components.

Special internal circuitry is also employed to ensure excellent power noise rejection to reduce the possibility of false detections, increasing the touch switch application reliability under adverse environmental conditions. With auto-calibration, low standby current, excellent resistance to voltage fluctuation and other features, this touch key device provides a simple and effective means of implementing touch key operation in a wide variety of applications.

The device includes the functions of input over/under voltage protection, linear charger, charger status indication, reference voltage output, etc. It supports a maximum input rating of 20V to overcome unexpected high voltage from the USB input. The linear charger provides an up to 1A charging current for the Li-ion battery. The reference voltage function is used by the MCU for measuring battery voltage and current. The device uses a two-line communication interface to select different function, such as to control the linear charger and reference voltage function, or to read the charging status and control the STDBY signal and CHRG pin by using the FN signal.

The device will find excellent use in a huge range of 3.7V lithium touch personal care products (electric toothbrushes, shavers, etc.) and 3.7V lithium portable touch electronic products to name but a few.

Block Diagram
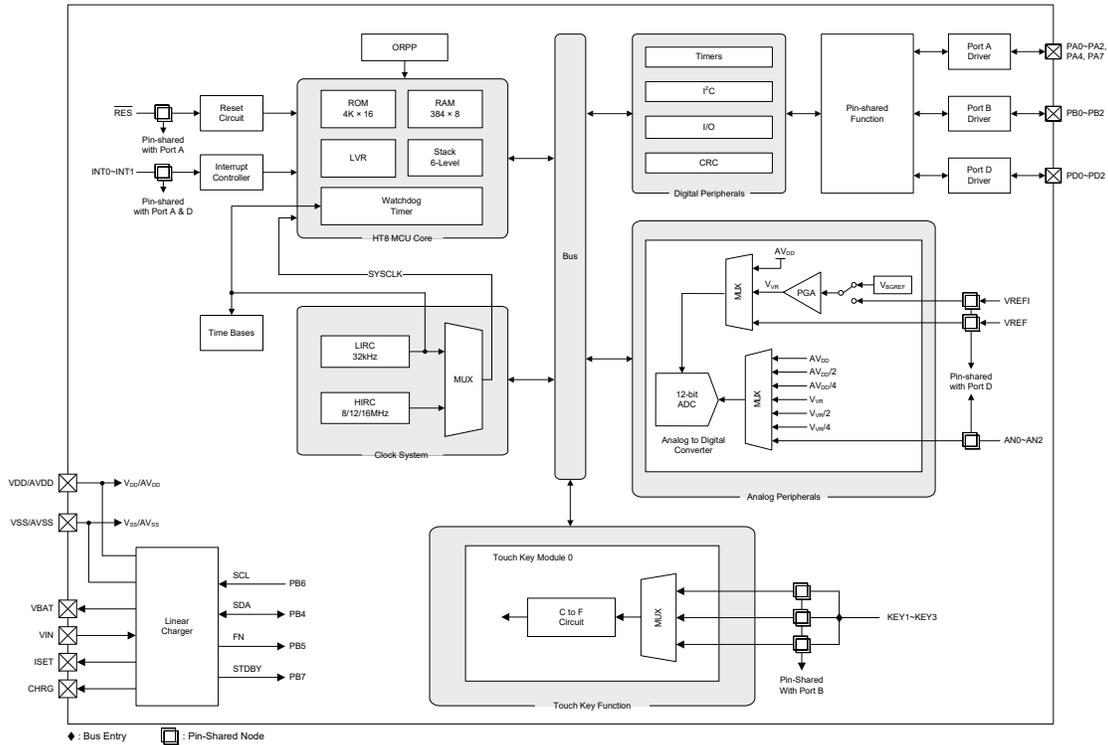
Pin Assignment

PA4/PTPI/SDA/INT0/CTP2/CTP1B & PD2/AN2/CTP2/CTP1B — 1
PA2/SCL/CTP2/ICPCK/OCDSCK — 2
PA1/PTP/CTP0 — 3
PA0/INT1/SDA/CTP1/ICPDA/OCDSDA — 4
PB0/KEY1 — 5
PB1/KEY2 — 6
PB2/KEY3 — 7
ISET — 8

16 — PA7/PTPB/SCL/CTP2B/VPP/$\overline{RES}$
15 — PD1/AN1/VREFI/CTP1/INT0
14 — PD0/AN0/VREF/CTP0
13 — VDD/AVDD
12 — VBAT
11 — VIN
10 — CHRG
9 — VSS/AVSS

BS45R2141/BS45EV2141
16 NSOP-A

Note: 1. On Pin 1 there are two sets of pin functions, the PA4/PTPI/SDA/INT0/CTP2/CTP1B or PD2/AN2/CTP2/CTP1B, however cannot be used simultaneously. If the PA4/PTPI/SDA/INT0/CTP2/CTP1B function set is used, the PD2 should be configured as an input with pull-high function disabled. If the PD2/AN2/CTP2/CTP1B function set is used, the PA4 should be configured as an input with pull-high function disabled.

2. If the pin-shared pin functions have multiple outputs, the desired pin-shared function is determined by the corresponding software control bits.

3. The OCDSDA and OCDSCK pins are the OCDS dedicated pins and only available for the BS45EV2141 device (Flash type) which is the OCDS EV chip for the BS45R2141 device (OTP type).

4. The VPP pin is the High Voltage input for OTP programming and only available for the BS45R2141 device.

5. There are some unbonded pins which should be properly configured to avoid unwanted current consumption resulting from floating input conditions. Refer to the "Standby Current Considerations" and "Input/Output Ports" sections.

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.

Note that the PA4 and PD2 share the same pin location, refer to the Pin Assignment.

Pin Name	Function	OPT	I/T	O/T	Description
PA0/INT1/SDA/CTP1/ ICPDA/OCDSDA	PA0	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	INT1	PAS0 INCT2 INTEG	ST	—	External interrupt input 1
	SDA	PAS0 IFS0	ST	NMOS	I²C data line
	CTP1	PAS0	ST	—	CTM1 output
	ICPDA	—	ST	CMOS	ICP address/data pin
	OCDSDA	—	ST	CMOS	OCDS address/data pin, for EV chip only
PA1/PTP/CTP0	PA1	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	PTP	PAS0	—	CMOS	PTM output
	CTP0	PAS0	—	CMOS	CTM0 output
PA2/SCL/CTP2/ICPCK/ OCDSCK	PA2	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	SCL	PAS0 IFS0	ST	NMOS	I²C clock line
	CTP2	PAS0	–	CMOS	CTM2 output
	ICPCK	—	ST	—	ICP clock
	OCDSCK	—	ST	—	OCDS clock pin, for EV chip only
PA4/PTPI/SDA/INT0/ CTP2/CTP1B	PA4	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	PTPI	PAS1	ST	—	PTM capture input
	SDA	PAS1 IFS0	ST	NMOS	I²C data line
	INT0	INCT0 INTEG PAS1 IFS0	ST	—	External interrupt input 0
	CTP2	PAS1	—	CMOS	CTM2 output
	CTP1B	PAS1	—	CMOS	CTM1 inverted output
PA7/PTPB/SCL/CTP2B/ VPP/RES	PA7	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up. This I/O is pin-shared with VPP and could result in increased current consumption if it is set to output high.
	PTPB	PAS1	—	CMOS	PTM inverting output
	SCL	PAS1 IFS0	ST	NMOS	I²C clock line
	CTP2B	PAS1	—	CMOS	CTM2 inverted output
	VPP	—	PWR	—	High Voltage input for OTP programming pin, not available for EV chip
	$\overline{RES}$	CO	ST	—	External reset input

Pin Name	Function	OPT	I/T	O/T	Description
PB0/KEY1	PB0	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high
	KEY1	PBS0	AN	—	Touch KEY1 input
PB1/KEY2	PB1	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high
	KEY2	PBS0	AN	—	Touch KEY2 input
PB2/KEY3	PB2	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high
	KEY3	PBS0	AN	—	Touch KEY3 input
PD0/AN0/VREF/CTP0	PD0	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high
	AN0	PDS0	AN	—	A/D Converter analog input channel 0
	VREF	PDS0	AN	—	A/D Converter reference voltage input
	CTP0	PDS0	—	CMOS	CTM0 output
PD1/AN1/VREFI/CTP1/ INT0	PD1	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high
	AN1	PDS0	AN	—	A/D Converter analog input channel 1
	VREFI	PDS0	AN	—	A/D Converter reference voltage input
	CTP1	PDS0	—	CMOS	CTM1 output
	INT0	INTEG INTC0 IFS0 PDS0	ST	—	External interrupt input 0
PD2/AN2/CTP2/CTP1B	PD2	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high
	AN2	PDS0	AN	—	A/D Converter analog input channel 2
	CTP2	PDS0	—	CMOS	CTM2 output
	CTP1B	PDS0	—	CMOS	CTM1 inverted output
ISET	ISET	—	—	AN	Charging current setting
CHRG	CHRG	—	—	OD	Open-drain output, refer to the "Linear Charger" section
VIN	VIN	—	PWR	—	Input power
VBAT	VBAT	—	—	AN	Battery power
VDD/AVDD	VDD	—	PWR	—	Digital positive power supply
	AVDD	—	PWR	—	Analog positive power supply
VSS/AVSS	VSS	—	PWR	—	Digital negative power supply, ground
	AVSS	—	PWR	—	Analog negative power supply, ground

Legend: I/T: Input type;   O/T: Output type;

      OPT: Optional by configuration option (CO) or register selection;

      PWR: Power;   ST: Schmitt Trigger input;

      CMOS: CMOS output;   NMOS: NMOS output;

      OD: Open drain;   AN: Analog signal;

      CO: Configuration option.

Internally Connected Signals

Several signals are not connected to external package pins. These signals are interconnection lines between the MCU and the Linear Charger and are listed in the following table. Users should properly configure the relevant pin-shared control bits and I/O control bits to implement correct interconnection.

MCU Signal Name	Linear Charger Signal Name	Function	Description
PB6	SCL	PB6	General purpose I/O Internally connected to the Linear Charger SCL
		SCL	Two-line communication clock line Internally connected to the MCU PB6
PB4	SDA	PB4	General purpose I/O Internally connected to the Linear Charger SDA
		SDA	Two-line communication data line Internally connected to the MCU PB4
PB5	FN	PB5	General purpose I/O Internally connected to the Linear Charger FN
		FN	Different status and functions based on two-line communication setting, refer to the "Interface" section Internally connected to the MCU PB5
PB7	STDBY	PB7	General purpose I/O Internally connected to the Linear Charger STDBY
		STDBY	Open-drain output, refer to the "Linear Charger" and "Interface" sections Internally connected to the MCU PB7

Absolute Maximum Ratings

Supply Voltage .... V_{SS}-0.3V to V_{SS}+6.0V

Input Voltage .... V_{SS}-0.3V to V_{DD}+0.3V

VIN, CHRG.. -0.3V to 20V

VBAT, ISET .... -0.3V to 7.0V

Maximum Junction Temperature.. 160°C

Lead Temperature (Soldering 10sec) .... 260°C

Storage Temperature... -60°C to 150°C

Operating Temperature... -40°C to 85°C

I_{OH} Total .... -80mA

I_{OL} Total 80mA

Total Power Dissipation 500mW

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 device reliability.

Recommended Operating Ranges

VIN, CHRG... 4.5V to 7.0V

VBAT .... 2.5V to 4.2V

I_{BAT} (max.) @ V_{IN}=5V 1A

Ta...-40°C to 85°C

Note: Absolute Maximum Ratings indicate limitations beyond which damage to the device may occur. Recommended Operating Ranges indicate conditions for which the device is intended to be functional, but do not guarantee specified performance limits.

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	$f_{SYS}=f_{HIRC}$=8MHz	2.0	—	5.5	V
		$f_{SY}=f_{HIRC}$=12MHz	2.7	—	5.5	
		$f_{SYS}=f_{HIRC}$=16MHz	3.3	—	5.5	
	Operating Voltage – LIRC	$f_{SYS}=f_{LIRC}$=32kHz	2.0	—	5.5	V

Operating Current Characteristics

Ta=-40°C~85°C

Symbol	Operating Mode	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{DD}	SLOW Mode – LIRC	2V	f_{SYS}=32kHz	—	30	50	μA
		3V		—	40	60	
		5V		—	60	80	
	FAST Mode – HIRC	2V	f_{SYS}=8MHz	—	0.6	1.0	mA
		3V		—	0.8	1.2	mA
		5V		—	1.6	2.4	mA
		2.7V	f_{SYS}=12MHz	—	1.0	1.4	mA
		3V		—	1.2	1.8	mA
		5V		—	2.4	3.6	mA
		3.3V	f_{SYS}=16MHz	—	1.5	3.0	mA
		5V		—	2.5	5.0	mA

Note: When using the characteristic table data, the following notes should be taken into consideration:
1. Any digital inputs are set in a non-floating condition and the I/O pin-shared with VPP is not setup in an output high 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	2V	WDT off	—	0.45	0.80	7.00	µA
		3V		—	0.45	0.90	8.00	
		5V		—	0.5	2.0	10.0	
		2V	WDT on	—	1.5	3.0	5.5	µA
		3V		—	1.8	3.6	6.5	
		5V		—	3	5	10	
	IDLE0 Mode – LIRC	2V	f_{SUB} on	—	2.4	4.0	8.0	µA
		3V		—	3	5	9	
		5V		—	5	10	15	
	IDLE1 Mode – HIRC	2V	f_{SUB} on, f_{SYS}=8MHz	—	288	400	480	µA
		3V		—	360	500	600	µA
		5V		—	600	800	960	µA
		2.7V	f_{SUB} on, f_{SYS}=12MHz	—	432	600	720	µA
		3V		—	540	750	900	µA
		5V		—	800	1200	1440	µA
		3.3V	f_{SUB} on, f_{SYS}=16MHz	—	0.8	1.2	1.44	mA
		5V		—	1.4	2.0	2.4	mA

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 and the I/O pin-shared with VPP is not setup in an output high 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 – HIRC – 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 either 3V or 5V.

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Temp.				
f_{HIRC}	8 MHz Writer Trimmed HIRC Frequency	3V/5V	25°C	-1%	8	+1%	MHz
			-20°C~60°C	-1.5%	8	+1.1%	
			-40°C~85°C	-3.5%	8	+3.5%	
		2.0V~5.5V	25°C	-5%	8	+5%	
			-20°C~60°C	-8.5%	8	+8%	
			-40°C~85°C	-10%	8	+10%	
		2.2V~5.5V	25°C	-3%	8	+3%	
			-20°C~60°C	-2.8%	8	+2.8%	
			-40°C~85°C	-4.5%	8	+4.5%	

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Temp.				
f_{HIRC}	12 MHz Writer Trimmed HIRC Frequency	3V/5V	25°C	-1%	12	+1%	MHz
			-20°C~60°C	-1.5%	12	+1.1%	
			-40°C~85°C	-3.5%	12	+3.5%	
		2.7V~5.5V	25°C	-3%	12	+3%	
			-20°C~60°C	-2.8%	12	+2.8%	
			-40°C~85°C	-4.5%	12	+4.5%	
	16 MHz Writer Trimmed HIRC Frequency	5V	25°C	-1%	16	+1%	MHz
			-20°C~60°C	-1.5%	16	+1.1%	
			-40°C~85°C	-3.5%	16	+3.5%	
		3.3V~5.5V	25°C	-3%	16	+3%	
			-20°C~60°C	-2.8%	16	+2.8%	
			-40°C~85°C	-4.5%	16	+4.5%	

Note: 1. The 3V/5V values for V_{DD} are provided as these are the two selectable fixed voltages at which the HIRC frequency is trimmed by the writer.

2. The row below the 3V/5V 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 2.0V to 3.6V and fixed at 5V for application voltage ranges from 3.3V to 5.5V.

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%.

Internal Low Speed Oscillator Characteristics – LIRC

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Temp.				
f_{LIRC}	LIRC Frequency	2.0V~5.5V	25°C	-20%	32	+20%	kHz
			-40°C~85°C	-50%	32	+60%	
t_{START}	LIRC Start Up Time	—	-40°C~85°C	—	—	500	μs

Operating Frequency Characteristic Curves

Rev. 1.00

September 11, 2025

System Start Up Time Characteristics

Ta=-40°C~85°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
t_{SST}	System Start-up Time Wake-up from condition where f_{SYS} is off	—	$f_{SYS}=f_H~f_H/64$, $f_H=f_{HIRC}$	—	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.	—	$f_{SYS}=f_H~f_H/64$, $f_H=f_{HIRC}$	—	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	—	$RR_{POR}=5V/ms$	5	16	80	ms
	System Reset Delay Time Reset source from WDT overflow or Reset pin reset	—	—	5	16	80	ms

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 $t_{HIRC}=1/f_{HIRC}$, $t_{LIRC}=1/f_{LIRC}$, 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=-40°C~85°C, unless otherwise specify

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{IL}	Input Low Voltage for I/O Ports (Except $\overline{RES}$)	5V	—	0	—	1.5	V
		—	—	0	—	$0.2V_{DD}$	
	Input Low Voltage for $\overline{RES}$ Pin	—	$V_{DD}≥2.7$	0	—	$0.4V_{DD}$	V
		—	$2.0≤V_{DD}<2.7$	0	—	$0.3V_{DD}$	V
V_{IH}	Input High Voltage for I/O Ports (Except $\overline{RES}$)	5V	—	3.5	—	5.0	V
		—	—	$0.8V_{DD}$	—	V_{DD}	
	Input High Voltage for $\overline{RES}$ Pin	—	—	$0.9V_{DD}$	—	V_{DD}	V
I_{OL}	Sink Current for I/O Pins (Except PD2~PD0)	3V	$V_{OL}=0.1V_{DD}$	16	32	—	mA
		5V		32	65	—	
	I/O Port Sink Current (For PD2~PD0 only)	3V	$V_{OL}=0.1V_{DD}$, PDNSn=0	16	32	—	mA
			$V_{OL}=0.1V_{DD}$, PDNSn=1	25	50	—	
		5V	$V_{OL}=0.1V_{DD}$, PDNSn=0	32	65	—	
			$V_{OL}=0.1V_{DD}$, PDNSn=1	50	100	—	

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{OH}	Source Current for PA7 only	3V	V_{OH}=0.9V_{DD}	-4	-8	—	mA
		5V		-8	-16	—	
	Source Current for I/O Pins (except PA7)	3V	V_{OH}=0.9V_{DD}, SLEDCn[m+1:m]=00B (n=0~2, m=0 or 2 or 4 or 6)	-0.7	-1.5	—	mA
		5V	V_{OH}=0.9V_{DD}, SLEDCn[m+1:m]=00B (n=0~2, m=0 or 2 or 4 or 6)	-1.5	-2.9	—	
		3V	V_{OH}=0.9V_{DD}, SLEDCn[m+1:m]=01B (n=0~2, m=0 or 2 or 4 or 6)	-1.3	-2.5	—	
		5V	V_{OH}=0.9V_{DD}, SLEDCn[m+1:m]=01B (n=0~2, m=0 or 2 or 4 or 6)	-2.5	-5.1	—	
		3V	V_{OH}=0.9V_{DD}, SLEDCn[m+1:m]=10B (n=0~2, m=0 or 2 or 4 or 6)	-1.8	-3.6	—	
		5V	V_{OH}=0.9V_{DD}, SLEDCn[m+1:m]=10B (n=0~2, m=0 or 2 or 4 or 6)	-3.6	-7.3	—	
		3V	V_{OH}=0.9V_{DD}, SLEDCn[m+1:m]=11B (n=0~2, m=0 or 2 or 4 or 6)	-4	-8	—	
		5V	V_{OH}=0.9V_{DD}, SLEDCn[m+1:m]=11B (n=0~2, m=0 or 2 or 4 or 6)	-8	-16	—	
R_{PH}	Pull-high Resistance for I/O Ports[1]	3V	—	20	60	100	kΩ
		5V		10	30	50	
I_{LEAK}	Input Leakage Current	5V	V_{IN}=V_{DD} or V_{IN}=V_{SS}	—	—	±1	µA
t_{TPI}	PTM Capture Input Pin Minimum Pulse Width	—	—	0.3	—	—	µs
f_{TMCLK}	PTM Maximum Timer Clock Source Frequency	5V	—	—	—	1	f_{SYS}
t_{CPW}	PTM Minimum Capture Pulse Width	—	—	—	t_{CPW}[2]	—	t_{TMCLK}
t_{INT}	External Interrupt Minimum Pulse Width	—	—	10	—	—	µs
t_{RES}	External Reset Minimum Low Pulse Width	—	—	0.3	—	—	µs

Note: 1. The R_{PH} internal pull high resistance value is calculated by connecting to ground and enabling the input pin with a pull-high resistor and then measuring the pin current at the specified supply voltage level. Dividing the voltage by this measured current provides the RPH value.

2. PTCAPTS=0, then t_{CPW}=max(2×t_{TMCLK}, t_{TPI})

Ex1: If f_{TMCLK}=16MHz, t_{TPI}=0.3µs, then t_{CPW}=max(0.125µs, 0.3µs)=0.3µs

Ex2: If f_{TMCLK}=8MHz, t_{TPI}=0.3µs, then t_{CPW}=max(0.25µs, 0.3µs)=0.3µs

Ex3: If f_{TMCLK}=4MHz, t_{TPI}=0.3µs, then t_{CPW}=max(0.5µs, 0.3µs)=0.5µs

Memory Characteristics

Ta=-40°C~85°C, unless otherwise specify

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
OTP Program Memory							
V_{DD}	V_{DD} for Read – ORPP Memory	—	—	2.0	5.0	5.5	V
	V_{DD} for Write – ORPP Memory	—	—	4.5	5.0	5.5	V
V_{PP}	V_{PP} for Write – ORPP Memory	—	—	8.25	8.50	8.75	V
t_{WR}	Write Cycle Time – ORPP Memory	—	—	—	300	450	μs
E_P	Cell Endurance – ORPP Memory	—	—	1	—	—	W
t_{RETD}	ROM Data Retention Time	—	Ta=25°C	—	40	—	Year
Flash Program Memory – for BS45EV2141 only							
t_{ACTV}	ROM Activation Time	—	Wake-up from Power Down Mode	1	—	2	t_{LIRC}
RAM Data Memory							
V_{DR}	RAM Data Retention Voltage	—	—	1.0	—	—	V

Note: 1. The ROM activation time t_{ACTV} should be added when calculating the total system start-up time of a wake-up from the power down mode.

2. "W" means Write times.

LVR Electrical Characteristics

Ta=-40°C~85°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{LVR}	Low Voltage Reset Voltage	—	LVR enable, voltage select 1.9V	-5%	1.9	+5%	V
			LVR enable, voltage select 2.1V		2.1		
			LVR enable, voltage select 3.15V		3.15		
			LVR enable, voltage select 4.2V		4.2		
I_{LVR}	Operating Current	3V	LVR enable, V_{LVR}=1.9V	—	—	15	μA
		5V	LVR enable, V_{LVR}=1.9V	—	15	30	
t_{LVR}	Minimum Low Voltage Width to Reset	—	—	100	240	1250	μs

A/D Converter Electrical Characteristics

Ta=-40°C~85°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{ADI}	Input Voltage	—	—	0	—	V_{REF}	V
V_{REF}	Reference Voltage	—	—	2.0	—	V_{DD}	V
N_R	Resolution	—	—	—	—	12	Bit

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
DNL	Differential Nonlinearity	2V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=0.5µs	-3	—	3	LSB
		3V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=0.5µs				
		5V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=0.5µs				
		3V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=10µs				
		5V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=10µs				
INL	Integral Nonlinearity	2V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=0.5µs	-4	—	4	LSB
		3V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=0.5µs	-4	—	4	
		5V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=0.5µs	-4	—	4	
		3V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=10µs	-4	—	4	
		5V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=10µs	-4	—	4	
I_{ADC}	Additional Current for A/D Converter Enable	2V	No load (t_{ADCK}=2.0µs)	—	280	400	µA
		3V	No load (t_{ADCK}=0.5µs)	—	450	600	
		5V	No load (t_{ADCK}=0.5µs)	—	850	1000	
t_{ADCK}	A/D Converter Clock Period	—	2.0V≤V_{DD}≤5.5V	0.5	—	10	µs
t_{ON2ST}	A/D Converter On-to-Start Time	—	—	4	—	—	µs
t_{ADS}	A/D Converter Sampling Time	—	—	—	4	—	t_{ADCK}
t_{ADC}	A/D Conversion Time (Including A/D Sample and Hold Time)	—	—	—	16	—	t_{ADCK}
I_{PGA}	Additional Current for PGA Enable	2.2V	No load，PGAIS=1, PGAGS[1:0]=01	—	250	500	µA
		3V	No load，PGAIS=1, PGAGS[1:0]=01	—	300	600	
		5V	No load，PGAIS=1, PGAGS[1:0]=01	—	400	700	
V_{OR}	PGA Maximum Output Voltage Range	2.2V	—	V_{SS}+0.1	—	V_{DD}-0.1	V
		3V	—	V_{SS}+0.1	—	V_{DD}-0.1	
		5V	—	V_{SS}+0.1	—	V_{DD}-0.1	
V_{VR}	PGA Fixed Voltage Output	—	V_{DD}=2.2V~5.5V $V_{RI}=V_{BGREF}$ (PGAIS=1)	-3%	2	+3%	V
		—	V_{DD}=3.2V~5.5V $V_{RI}=V_{BGREF}$ (PGAIS=1)	-3%	3	+3%	
		—	V_{DD}=4.2V~5.5V $V_{RI}=V_{BGREF}$ (PGAIS=1)	-3%	4	+3%	

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{IR}	PGA Input Voltage Range	3V	—	V_{SS}+0.1	—	V_{DD}-1.4	mV
		5V		V_{SS}+0.1	—	V_{DD}-1.4	

Reference Voltage Characteristics

Ta=-40°C~85°C, unless otherwise specified

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{BGREF}	Operating Current	5.5V	—	—	25	35	μA
PSRR	Power Supply Rejection Ratio	—	Ta=25°C, V_{RIPPLE}=1V_{P-P}, f_{RIPPLE}=100Hz	75	—	—	dB
En	Output Noise	—	Ta=25°C, no load current, f=0.1Hz~10Hz	—	300	—	μV_{RMS}
I_{SD}	Shutdown Current	—	V_{BGREN}=0	—	—	0.1	μA
t_{START}	Startup Time	2.0V~5.5V	Ta=25°C	—	—	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 V_{DD} and GND.

3. The V_{BGREF} voltage is used as the A/D converter PGA internal input signal.

I²C Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{I2C}	I²C Standard Mode (100kHz) f_{SYS} Frequency [Note]	—	No clock debounce	2	—	—	MHz
			2 system clock debounce	4	—	—	
			4 system clock debounce	4	—	—	
	I²C Fast Mode (400kHz) f_{SYS} Frequency [Note]	—	No clock debounce	4	—	—	MHz
			2 system clock debounce	8	—	—	
			4 system clock debounce	8	—	—	
f_{SCL}	SCL Clock Frequency	3V/5V	Standard mode	—	—	100	kHz
			Fast mode	—	—	400	
$t_{SCL(H)}$	SCL Clock High Time	3V/5V	Standard mode	3.5	—	—	μs
			Fast mode	0.9	—	—	
$t_{SCL(L)}$	SCL Clock Low Time	3V/5V	Standard mode	3.5	—	—	μs
			Fast mode	0.9	—	—	
t_{FALL}	SCL and SDA Fall Time	3V/5V	Standard mode	—	—	1.3	μs
			Fast mode	—	—	0.34	
t_{RISE}	SCL and SDA Rise Time	3V/5V	Standard mode	—	—	1.3	μs
			Fast mode	—	—	0.34	
$t_{SU(SDA)}$	SDA Data Setup Time	3V/5V	Standard mode	0.25	—	—	μs
			Fast mode	0.1	—	—	
$t_{H(SDA)}$	SDA Data Hold Time	3V/5V	—	0.1	—	—	μs
$t_{VD(SDA)}$	SDA Data Valid Time	3V/5V	—	—	—	0.6	μs
$t_{SU(STA)}$	Start Condition Setup Time	3V/5V	Standard mode	3.5	—	—	μs
			Fast mode	0.6	—	—	

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
$t_{H(STA)}$	Start Condition Hold Time	3V/5V	Standard mode	4.0	—	—	μs
			Fast mode	0.6	—	—	
$t_{SU(STO)}$	Stop Condition Setup Time	3V/5V	Standard mode	3.5	—	—	μs
			Fast mode	0.6	—	—	

Note: Using the debounce function can make the transmission more stable and reduce the probability of communication failure due to interference.
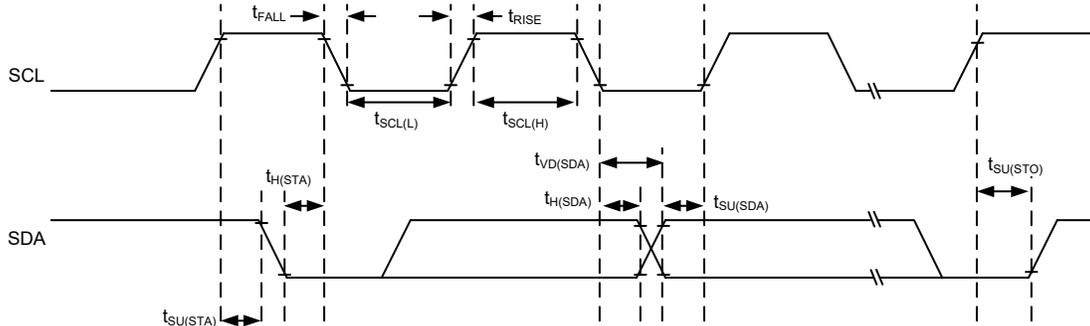
I²C Timing Diagram

High Voltage Linear Charger Electrical Characteristics

V_{IN}=5V, V_{BAT}=3.6V, R_{SET}=20kΩ, Ta= 25°C, unless otherwise specified

Symbol	Parameter	Test Condition	Min.	Typ.	Max.	Unit
Supply and Input						
V_{IN}	Input Voltage	—	4.5	5.0	7.0	V
I_{IN1}	Operation Current (Charging)	R_{SET}=20kΩ, measure I_{VIN}-I_{VBAT}	—	—	2	mA
I_{IN2}	Operation Current (Charging Done)	V_{BAT}=4.2V, measure I_{VIN}	—	—	450	μA
I_{BAT_LK1}	VBAT Leakage Current (Charger Shutdown)	D[6:0]="0000000"	—	-1	-2	μA
I_{BAT_LK2}	VBAT Leakage Current (Invalid Input)	V_{IN}=0V	—	-1	-2	μA
I_{BAT_VREF}	VREF Operation Current	V_{IN}=0V, D[6:0]="0110000"	—	-150	-200	μA
Over Voltage and Under Voltage Detect						
V_{IN_UVLO1}	Input Under Voltage Lockout 1	V_{IN} rising	3.8	4.0	4.2	V
V_{IN_UVLO2}	Input Under Voltage Lockout 2	V_{IN} rising	—	V_{BAT} +0.1V	—	V
$V_{IN\_UVLO\_HYS}$	V_{IN_UVLO1} Hysteresis Voltage	V_{IN} falling	20	100	—	mV
V_{IN_OVP}	Input Over Voltage Threshold	V_{IN} rising	7.0	7.3	7.6	V
$V_{IN\_OVP\_HYS}$	V_{IN_ovp} Hysteresis Voltage	V_{IN} falling	20	100	—	mV
t_{IN_OVP}	Over Voltage Response Time	V_{IN} rising at 10V/1μs	—	350	—	ns
Linear Charger						
V_{BAT_CV1}	Battery Charging Target Voltage	V_{IN}=4.5V, R_{SET}=20kΩ, I_{BAT_LOAD}=20mA, Ta=0°C~85°C	-1%	4.2	+1%	V
V_{BAT_CV2}	Battery Charging Target Voltage	V_{IN}=5.5V, R_{SET}=2kΩ, I_{BAT_LOAD}=200mA, Ta=0°C~85°C				
V_{BAT_RECHG}	Battery Recharge Threshold	V_{BAT} falling	—	V_{BAT_CV} -0.2	—	V
V_{BAT_LOW}	Low Battery Voltage	V_{BAT} rising	2.9	3.0	3.1	V
$V_{BAT\_LOW\_HYS}$	V_{BAT_LOW} Hysteresis Voltage	V_{BAT} falling	—	100	—	mV
V_{IF_EN}	Interface Enable Voltage	V_{BAT} rising	2.2	—	—	V

Symbol	Parameter	Test Condition	Min.	Typ.	Max.	Unit
V_{IF_DIS}	Interface Disable Voltage	V_{BAT} falling	—	—	0.8	V
I_{BAT_CC}	Programmable Constant Charging Current	R_{SET}=2kΩ	900	1000	1100	mA
		R_{SET}=4kΩ	-10%	500	+10%	mA
		R_{SET}=20kΩ	-10%	100	+10%	mA
I_{BAT_TC}	Trickle Charging Current	V_{BAT}=2.7V	-50%	0.1× I_{BAT_CC}	+50%	mA
I_{BAT_TERM}	Charge Termination Current	—	-15%	0.1× I_{BAT_CC}	+15%	mA
T_{CT}	Linear Charger Constant Temperature	—	—	125	—	°C
Others						
V_{REF}	Reference Voltage	V_{BAT}=2.5V~4.2V, I_{LOAD}=1mA	-1%	2.0	+1%	V
f_{SCL}	SCL Frequency	—	—	50	—	kHz
V_{IF}	Interface Operation Voltage	—	2.2	—	5.5	V
I_{OL}	Sink Current for CHRG Pin	V_{OL}=1.5V	2.5	—	—	mA
R_{PL}	Pull-low Resistance for SDA and SCL Signals	—	—	100	—	kΩ
$R_{DS(ON)}$	MOS On Resistance between VIN and VBAT	R_{SET}=1kΩ, V_{IN}=4.5V	—	—	800	mΩ
t_{INOK2}	VIN Status (INOK2) Holding Time on FN Signal	V_{IN}=0V~8V, V_{BAT}=2.5V~4.2V, D[6:0]="1010000"	65	—	180	µs
t_{LCS}	Linear Charger Status Setting Time on FN Signal	V_{IN}=0V~8V, V_{BAT}=2.5V~4.2V, D[6:0]="1010000"	—	—	30	µs
t_{VREF}	VREF Setting Time on FN Signal	V_{IN}=0V, D[6:0]="0110000"	—	—	15	µs

Power-on Reset Characteristics

Ta=-40°C~85°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
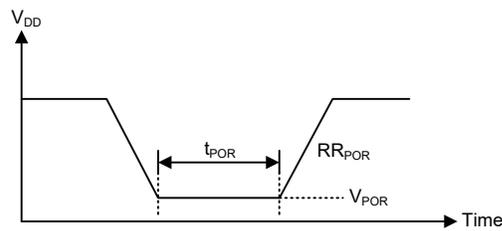
System Architecture

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 cycle, with the exception of branch or call instructions. 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 affordable, high-volume production for controller applications.

Clocking and Pipelining

The main system clock, derived from either an 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.
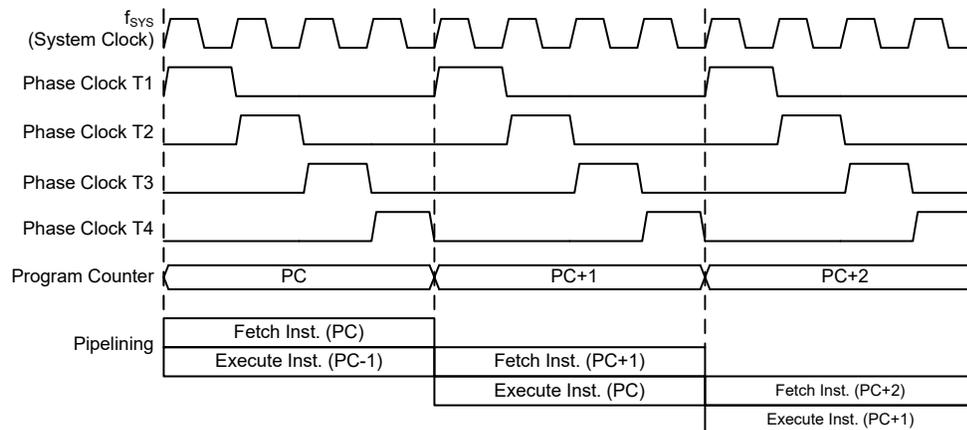
System Clocking and Pipelining
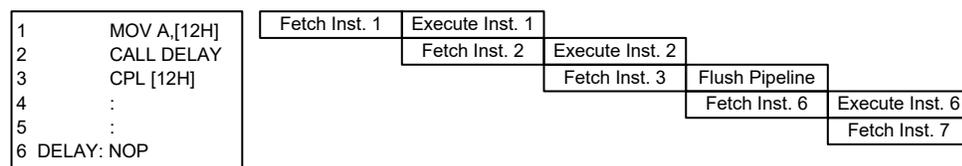
Instruction Fetching

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 demand 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 or 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.

Program Counter Read Registers

The Program Counter Read registers are a read only register pair for reading the program counter value which indicates the current program execution address. Read the low byte register first then the high byte register. Reading the low byte register, PCRL, will read the low byte data of the current program execution address, and place the high byte data of the program counter into the 8-bit PCRH buffer. Then reading the PCRH register will read the corresponding data from the 8-bit PCRH buffer.

The following example shows how to read the current program execution address. When the current program execution address is 123H, the steps to execute the instructions are as follows:

(1) MOV A, PCRL → the ACC value is 23H, and the PCRH value is 01H;

MOV A, PCRH → the ACC value is 01H.

(2) LMOV A, PCRL → the ACC value is 23H, and the PCRH value is 01H;

LMOV A, PCRH → the ACC value is 01H.
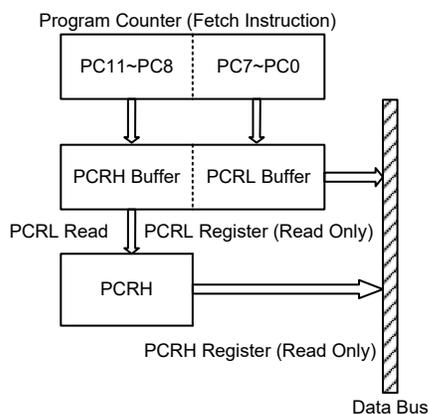
• PCRL Register

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:** Program Counter Read Low byte register bit 7 ~ bit 0

• PCRH Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	D11	D10	D9	D8
R/W	—	—	—	—	R	R	R	R
POR	—	—	—	—	0	0	0	0

Bit 7~4   Unimplemented, read as "0"

Bit 3~0   **D11~D8:** Program Counter Read High byte register bit 3 ~ bit 0

Stack

This is a special part of the memory which is used to save the contents of the Program Counter only. The stack is organized into 6 levels and is 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, STKPTR[2:0]. 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.
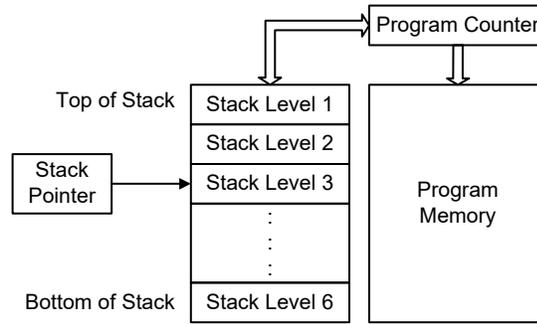
- **STKPTR Register**

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

Bit 7   **OSF:** Stack overflow flag
　　　　　0: No stack overflow occurred
　　　　　1: Stack overflow occurred
　　　　　When the stack is full and a CALL instruction is executed or when the stack is empty
　　　　　and a RET instruction is executed, the OSF bit will be set high. The OSF bit is cleared
　　　　　only by software and cannot be reset automatically by hardware.

Bit 6~3   Unimplemented, read as "0"

Bit 2~0   **D2~D0:** Stack pointer register bit 2 ~ bit 0

The following example shows how the Stack Pointer and Stack Overflow Flag change when program branching conditions occur.

(1) When the CALL subroutine instruction is executed 7 times continuously and the RET instruction is not executed during the period, the corresponding changes of the STKPTR[2:0] and OSF bits are as follows:

CALL Execution Times	0	1	2	3	4	5	6	7
STKPTR[2:0] Bit Value	0	1	2	3	4	5	0	1
OSF Bit Value	0	0	0	0	0	0	0	1

(2) When the OSF bit is set high and not cleared, it will remain high no matter how many times the RET instruction is executed.

(3) When the stack is empty, the RET instruction is executed 6 times continuously, the corresponding changes of the STKPTR[2:0] and OSF bits are as follows:

RET Execution Times	0	1	2	3	4	5	6
STKPTR[2:0] Bit Value	0	5	4	3	2	1	0
OSF Bit Value	0	1	1	1	1	1	1

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

- Logic operations: AND, OR, XOR, ANDM, ORM, XORM, CPL, CPLA

- Rotation: RRA, RR, RRCA, RRC, RLA, RL, RLCA, RLC

- Increment and Decrement: INCA, INC, DECA, DEC

- Branch decision: JMP, SZ, SZA, SNZ, SIZ, SDZ, SIZA, SDZA, CALL, RET, RETI

OTP Program Memory

The Program Memory is the location where the user code or program is stored. The device is supplied with One-Time Programmable, OTP memory where users can program their application code into the device.

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 set in any location within the Program Memory, is addressed by a separate table pointer register.
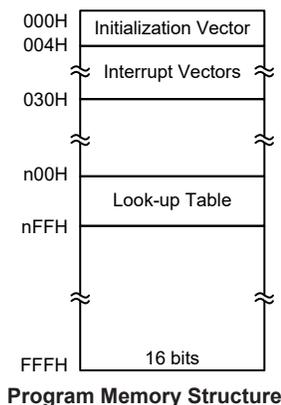
Program Memory Structure

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 configured by placing the address of the look up data to be retrieved in the table pointer register, TBLP and TBHP. This register defines 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 "TABRD [m]" or "TABRDL [m]" instructions 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.
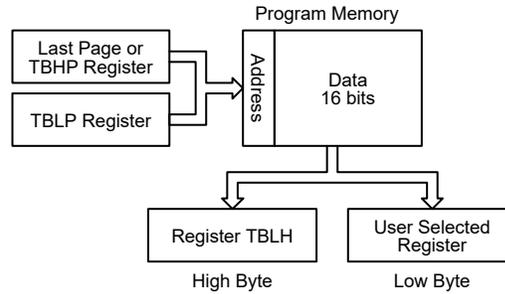
Table Program Example

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 words Program Memory of the device. The table pointer low byte register is set 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 the specific address pointed by the TBHP and TBLP registers if the "TABRD [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] instruction is executed.

Because the TBLH register is a read-only register and cannot 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 present page
:
:
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
                   ; the value "00H" will be transferred to the high byte register
                   ; TBLH
:
:
```

```
org 0F00h        ; sets initial address of program memory
dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
```

In Circuit Programming – ICP

The provision of OTP type Program Memory, users can program their application One-Time into the device. As an additional convenience, Holtek has provided a means of programming the microcontroller in-circuit using a 5-pin interface. This provides manufacturers with the possibility of manufacturing their circuit boards complete with an un-programmed microcontroller, and then programming the program at a later stage.

Holtek Writer Pins	MCU Programming Pins	Pin Description
ICPDA	PA0	Programming Serial Data/Address
ICPCK	PA2	Programming Clock
VPP	VPP	Programming OTP ROM power supply (8.5V)
VDD	VDD	Power Supply. A 0.1µF capacitor is required to be connected between VDD and VSS for programming.
VSS	VSS	Ground

The Program Memory can be programmed serially in-circuit using this 5-wire interface. Data is downloaded and uploaded serially on a single pin with an additional line for the clock. Three additional lines are required for the power supply. The technical details regarding the in-circuit programming of the device are beyond the scope of this document and will be supplied in supplementary literature.

During the programming process, the user must take care of the ICPDA and ICPCK pins for data and clock programming purposes to ensure that no other outputs are connected to these two pins.
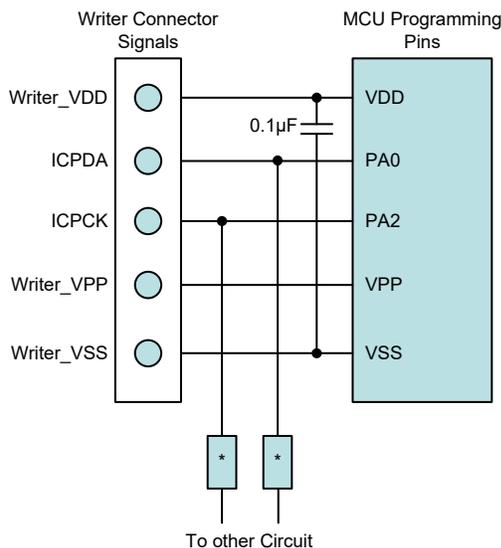
Note: 1. A 0.1µF capacitor is required to be connected between VDD and VSS for ICP programming, and as close to these pins as possible.
    2. * may be resistor or capacitor. The resistance of * must be greater than 1kΩ or the capacitance of * must be less than 1nF.

On-Chip Debug Support – OCDS

There is an EV chip named BS45EV2141 which is used to emulate the device named BS45R2141. 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 and the package type. 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. 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

OTP ROM Parameter Program – ORPP

This device contains an ORPP function. The provision of the ORPP function offers users the convenience of OTP Memory programming features. Note that the Write operation only writes data to the last page of OTP Program Memory, and the data can only be written once and cannot be erased.

Before the write operation is implemented, the VPP pin must be connected to an 8.5V power and after the write operation is completed, the high voltage power should be removed from the VPP pin. If the VPP function is pin-shared with an I/O port, the corresponding I/O port cannot be set as an output when it is used as the VPP function.

ORPP Registers

Three registers control the overall operation of the internal ORPP function. These are data registers ODL and ODH, and a control register OCR.

Register Name	Bit							
	7	6	5	4	3	2	1	0
OCR	—	—	—	—	WREN	WR	—	—
ODL	D7	D6	D5	D4	D3	D2	D1	D0
ODH	D15	D14	D13	D12	D11	D10	D9	D8

ORPP Register List

• ODL 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:** ORPP Program Memory data bit 7 ~ bit 0

• **ODH Register**

Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
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　　　**D15~D8:** ORPP Program Memory data bit 15 ~ bit 8

• **OCR Register**

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

Bit 7~4　　Unimplemented, read as "0"

Bit 3　　**WREN:** ORPP Write Enable
　　　　　　0: Disable
　　　　　　1: Enable
　　　　This is the ORPP Write Enable Bit which must be set high before write operations are carried out. This bit will be automatically reset to zero by the hardware after the write cycle has finished. Clearing this bit to zero will inhibit ORPP write operations.

Bit 2　　**WR:** ORPP Write Control
　　　　　　0: Write cycle has finished
　　　　　　1: Activate a write cycle
　　　　This is the ORPP 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 1~0　　Unimplemented, read as "0"

Note: 1. The WREN and WR cannot be set high at the same time in one instruction.
　　　　2. Note that the CPU will be stopped when a write operation is successfully activated.
　　　　3. Ensure that the f_{SUB} clock is stable before executing the write operation.
　　　　4. Ensure that the write operation is totally complete before executing other operations.

ORPP Writing Data to the OTP Program Memory

For ORPP write operation the data to be written should be placed in the ODH and ODL registers and the desired write address should first be placed in the TBLP register. To write data to the OTP Program Memory, the write enable bit, WREN, in the OCR register must first be set high to enable the write function. After this, the WR bit in the OCR register must be immediately set high to initiate a write cycle. These two instructions must be executed in two consecutive instruction cycles to activate a write operation successfully. The global interrupt bit EMI should also first be cleared before implementing any write operations, and then set again after a valid write activation procedure has completed. Note that the CPU will be stopped when a write operation is successfully activated. When the write cycle terminates, the CPU will resume executing the application program. And the WR bit will be automatically cleared to zero by the microcontroller, informing the user that the data has been written to the OTP Program Memory.

ORPP Reading Data from the OTP Program Memory

For ORPP read operation the desired address should first be placed in the TBLP register. Then the data can be retrieved from the program memory using the "TABRDL [m]" instruction. 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.

Programming Considerations

Care must be taken that data is not inadvertently written to the OTP Program Memory. Protection can be enhanced by ensuring that the Write Enable bit is normally cleared to zero when not writing. 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 set high again after a write activation procedure has completed. Note that the device should not enter the IDLE or SLEEP mode until the ORPP write operation is totally complete. Otherwise, the ORPP write operation will fail.

Programming Examples

ORPP Reading data from the OTP Program Memory

```
Tempreg1 db?              ; temporary register
MOV A, 03H
MOV TBLP, A              ; set read address 03H
TABRDL Tempreg1          ; transfers value in table (last page) referenced by table
                         ; pointer, data at program memory address "0F03H"
                         ; transferred to tempreg1 and TBLH
```

ORPP Writing Data to the OTP Program Memory

```
MOV A, ORPP_ADRES        ; user defined address
MOV TBLP, A
MOV A, ORPP_DATA_L       ; user defined data
MOV ODL, A
MOV A, ORPP_DATA_H
MOV ODH, A
MOV A, 00H
MOV OCR, A
CLR EMI
SET WREN                 ; set WREN bit, enable write operation
SET WR                   ; start Write Cycle – set WR bit – executed immediately
                         ; after setting WREN bit
SET EMI
BACK:
SZ  WR                   ; check for write cycle end
JMP BACK
NOP
```

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.

Structure

Categorised into two types, the first of these is an area of RAM, known as the Special Function Data Memory. 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 known as the General Purpose Data Memory, which is reserved for general purpose use. All locations within this area are read and write accessible under program control.

The start address of the Data Memory for the device is 00H. For the device, the address range of the Special Purpose Data Memory is from 00H to 7FH while the address range of the General Purpose Data Memory is from 80H to FFH. Switching between the different Data Memory banks is achieved by setting the Data Memory Bank Pointer to the correct value.

Special Purpose Data Memory	General Purpose Data Memory	
Located Bank	Capacity	Bank: Address
0	384×8	0: 80H~FFH 1: 80H~FFH 2: 80H~FFH
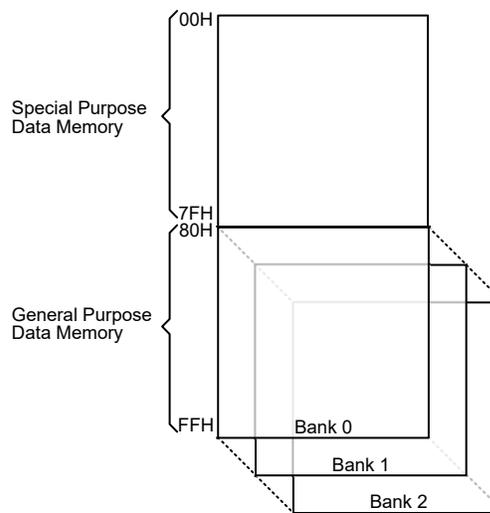
Data Memory Structure

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".

	Bank 0			Bank 0
00H	IAR0		40H	IECC
01H	MP0		41H	PTMC0
02H	IAR1		42H	PTMC1
03H	MP1		43H	PTMDL
04H	BP		44H	PTMDH
05H	ACC		45H	PTMAL
06H	PCL		46H	PTMAH
07H	TBLP		47H	PTMRPL
08H	TBLH		48H	PTMRPH
09H	TBHP		49H	SIMC0
0AH	STATUS		4AH	SIMC1
0BH			4BH	SIMD
0CH	INTEG		4CH	SIMA
0DH	WDTC		4DH	SIMTOC
0EH	TB0C		4EH	
0FH			4FH	SADC0
10H	SCC		50H	SADC1
11H	HIRCC		51H	SADC2
12H	INTC0		52H	SADOL
13H	INTC1		53H	SADOH
14H	PA		54H	VBGRC
15H	PAC		55H	CRCCR
16H	PAPU		56H	CRCIN
17H	PAWU		57H	CRCDL
18H	TB1C		58H	CRCDH
19H	TKTMR		59H	PCRL
1AH	TKC0		5AH	PCRH
1BH	TKC1		5BH	CTM0C0
1CH	TK16DL		5CH	CTM0C1
1DH	TK16DH		5DH	CTM0DL
1EH	TKM0C0		5EH	CTM0DH
1FH	TKM0C1		5FH	CTM0AL
20H	TKM016DL		60H	CTM0AH
21H	TKM016DH		61H	CTM1C0
22H	TKM0ROL		62H	CTM1C1
23H	TKM0ROH		63H	CTM1DL
24H	INTC2		64H	CTM1DH
25H	MFI		65H	CTM1AL
26H	PB		66H	CTM1AH
27H	PBC		67H	CTM2C0
28H	PBPU		68H	CTM2C1
29H	SLEDC0		69H	CTM2DL
2AH	SLEDC1		6AH	CTM2DH
2BH	SLEDC2		6BH	CTM2AL
2CH	PDNS		6CH	CTM2AH
2DH	OCR		6DH	STKPTR
2EH	ODL		6EH	INTC3
2FH	ODH		6FH	LVRC
30H	PD		70H	
31H	PDC			
32H	PDPU			
33H	PAS0			
34H	PAS1			
35H	PBS0			
36H	PDS0			
37H				
38H	IFS0			
39H				
3AH				
3BH				
3CH				
3DH				
3EH				
3FH	ORMC		7FH	

: Unused, read as 00H   : Reserved, cannot be changed

Special Purpose Data Memory

Special Function Register Description

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

Indirect Addressing Register – IAR0, IAR1

The Indirect Addressing Register, IAR0 and IAR1, although having its location in normal RAM register space, does not actually physically exist as normal registers. The method of indirect addressing for RAM data manipulation uses this Indirect Addressing Register and Memory Pointer, in contrast to direct memory addressing, where the actual memory address is specified. Actions on the IAR0 and IAR1 registers will result in no actual read or write operation to this register but rather to the memory location specified by the corresponding Memory Pointer, MP0 or MP1. Acting as a pair, IAR0 and MP0 or IAR1 and MP1 can together access data from Bank 0. As the Indirect Addressing Register is not physically implemented, reading the Indirect Addressing Register will return a result of "00H" and writing to the register will result in no operation.

Memory Pointer – MP0 , MP1

Two Memory Pointers, known as MP0 and MP1 are provided. This Memory Pointer is 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 Indirect Addressing Register is carried out, the actual address that the microcontroller is directed to is the address specified by the Memory Pointer. MP0, together with the Indirect Addressing Register, IAR0, are used to access data from Bank 0, while MP1 and IAR1 are used to access data from all banks according to the desired data memory bank selected by the DMBP0~DMBP1 bits in the BP register. Direct Addressing can only be used within Bank 0, all other Banks must be addressed indirectly using MP1 and IAR1.

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

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

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

Bank Pointer – BP

The Data Memory is divided into several banks. Selecting the required Data Memory area is achieved using the Bank Pointer.

The Data Memory is initialised to Bank 0 after a reset, except for a WDT time-out reset in the IDLE or SLEEP mode, in which case, the Data Memory bank remains unaffected. Directly addressing the Data Memory will always result in Bank 0 being accessed irrespective of the value of the Bank Pointer. Accessing data from banks other than Bank 0 must be implemented using Indirect Addressing.

• **BP Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	DMBP1	DMBP0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2   Unimplemented, read as "0"

Bit 1~0   **DMBP1~DMBP0:** Data Memory Bank Selection
          00: Bank 0
          01: Bank 1
          10: Bank 2
          11: Undefined

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 Byte 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. The TBLP and TBHP registers are the table pointer pair and indicates the location where the table data is located. Their value must be setup before any table read instructions 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.

Option Memory Mapping Register – ORMC

The ORMC register is used to enable Option Memory Mapping function. The Option Memory capacity is 64 words. When a specific pattern of 55H and AAH is consecutively written into this register, the Option Memory Mapping function will be enabled and then the Option Memory code can be read by using the table read instruction. The Option Memory addresses 000H~03FH will be mapped to Program Memory last page addresses 0C0H~0FFH.

To successfully enable the Option Memory Mapping function, the specific pattern of 55H and AAH must be written into the ORMC register in two consecutive instruction cycles. It is therefore recommended that the global interrupt bit EMI should first be cleared before writing the specific pattern, and then set high again at a proper time according to users' requirements after the pattern is successfully written. An internal timer will be activated when the pattern is successfully written. The mapping operation will be automatically finished after a period of $4 \times t_{LIRC}$. Therefore, users should read the data in time, otherwise the Option Memory Mapping function needs to be restarted. After the completion of each consecutive write operation to the ORMC register, the timer will recount.

When the table read instructions are used to read the Option Memory code, both "TABRD [m]" and "TABRDL [m]" instructions can be used. However, care must be taken if the "TABRD [m]" instruction is used, the table pointer defined by the TBHP register must be referenced to the last page. Refer to corresponding sections about the table read instruction for more details.

- **ORMC Register**

Bit	7	6	5	4	3	2	1	0
Name	ORMC7	ORMC6	ORMC5	ORMC4	ORMC3	ORMC2	ORMC1	ORMC0
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 **ORMC7~ORMC0:** Option Memory Mapping specific pattern

When a specific pattern of 55H and AAH is written into this register, the Option Memory Mapping function will be enabled. Note that the register content will be cleared after the MCU is woken up from the IDLE/SLEEP mode.

Status Register – STATUS

This 8-bit register contains the 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 and C 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.

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	—	—	TO	PDF	OV	Z	AC	C
R/W	—	—	R	R	R/W	R/W	R/W	R/W
POR	—	—	0	0	x	x	x	x

"x": unknown

Bit 7~6   Unimplemented, read as "0"

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.

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 through a combination of configuration options and 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. The fully integrated internal oscillators, requiring no external components, are provided to form a wide range of both fast and slow system oscillators. The higher frequency oscillator provides higher performance but carry with it the disadvantage of higher power requirements, while the opposite is of course true for the lower frequency oscillator. With the capability of dynamically switching between fast and slow system clock, the device has 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	8/12/16MHz
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 clock is sourced from the internal 8/12/16MHz 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 the system clock can be dynamically selected.

System Clock Configuration
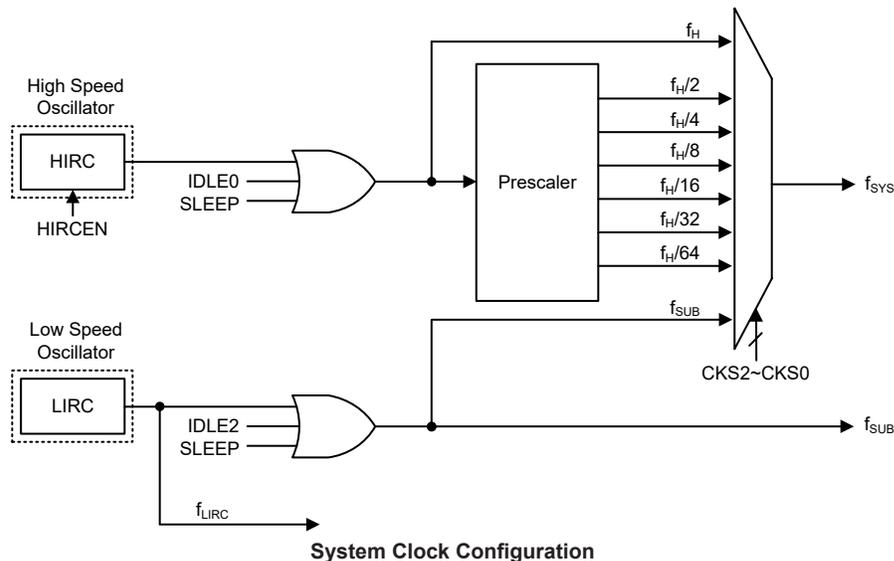
Internal High Speed RC Oscillator – HIRC

The internal RC oscillator is a fully integrated system oscillator requiring no external components. The internal RC oscillator has three fixed frequencys of 8MHz, 12MHz and 16MHz, which is selected using a configuration option. The HIRC1~HIRC0 bits in the HIRCC register must also be setup to match the selected configuration option frequency. Setting up these bits is necessary to ensure that the HIRC frequency accuracy specified in the A.C. Characteristics is achieved. 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 a fully integrated low frequency 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 either 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. The low speed system clock source is sourced from the LIRC oscillator. The other choice, which is a divided version of the high speed system oscillator has a range of $f_H/2 \sim f_H/64$.
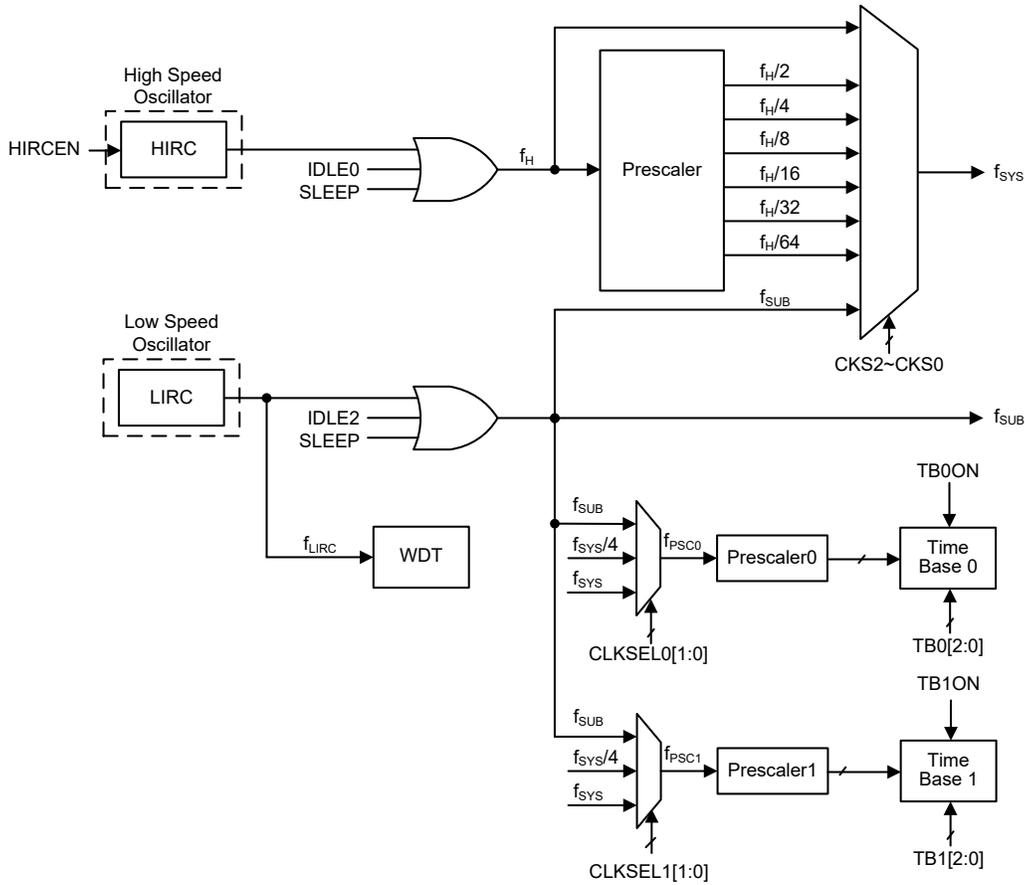
Device Clock Configurations

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, $f_H \sim f_H/64$, 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	$f_H \sim f_H/64$	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/Off [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 can be switched on or off which is controlled by the WDT function being enabled or disabled 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 which will come from the high speed oscillator, HIRC. 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}, which is derived from the LIRC oscillator.

SLEEP Mode

The SLEEP Mode is entered when an HALT instruction is executed and when the FHIDEN and FSIDEN bits are low. In the SLEEP mode the CPU will be stopped. The f_{SUB} clock provided to the peripheral function will also be stopped, too. However the f_{LIRC} clock can continues to operate if the WDT function is enabled.

IDLE0 Mode

The IDLE0 Mode is entered when an 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 an 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 an 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 Register

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

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

"—": Unimplemented, read as "0"

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.

Note: A certain delay is required before the relevant clock is successfully switched to the target clock source after any clock switching setup using the CKS2~CKS0 bits. A proper delay time must be arranged before executing the following operations which require immediate reaction with the target clock source.

Clock switching delay time$=4 \times t_{SYS} + [0 \sim (1.5 \times t_{Curr.} + 0.5 \times t_{Tar.})]$, where $t_{Curr.}$ indicates the current clock period, $t_{Tar.}$ indicates the target clock period and the t_{SYS} indicates the current system clock period.

• **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: 8MHz
        01: 12MHz
        10: 16MHz
        11: 8MHz

When the HIRC oscillator is enabled or the HIRC frequency selection is changed by application program, the clock frequency will automatically be changed after the HIRCF flag is set high.

It is recommended that the HIRC frequency selected by these two bits should be the same with the frequency determined by the configuration option to achieve the HIRC frequency accuracy specified in the A.C. Characteristics.

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 high to enable the HIRC oscillator or the HIRC frequency selection is changed by application program, the HIRCF bit will first be cleared to zero and then set high 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 enters the IDLE Mode or the SLEEP Mode is determined by the condition of the FHIDEN and FSIDEN bits in the SCC register.
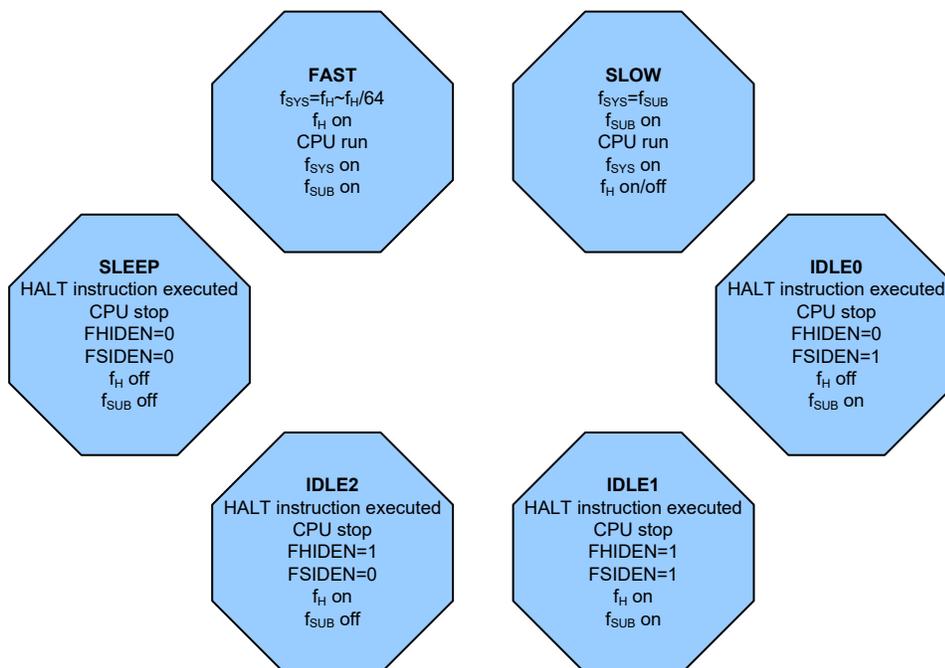
FAST Mode to SLOW Mode Switching

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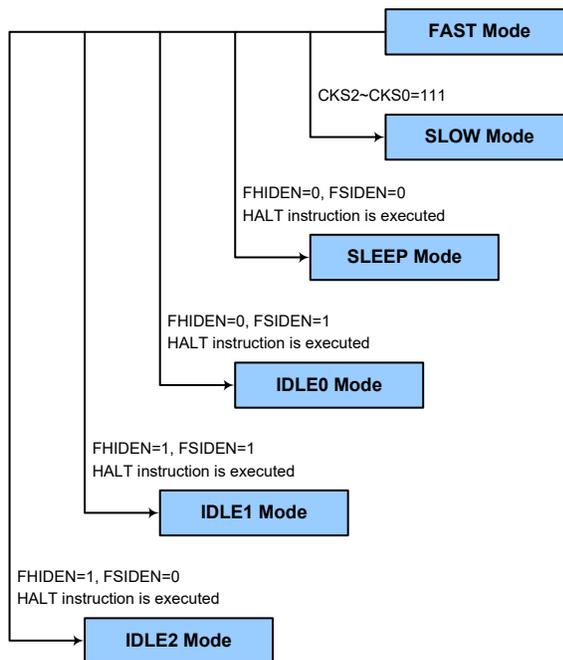
SLOW Mode to FAST Mode Switching

In the 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~$f_H/64$.

However, if f_H is not used in the 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 stabilisation is specified in the System Start Up Time Characteristics.
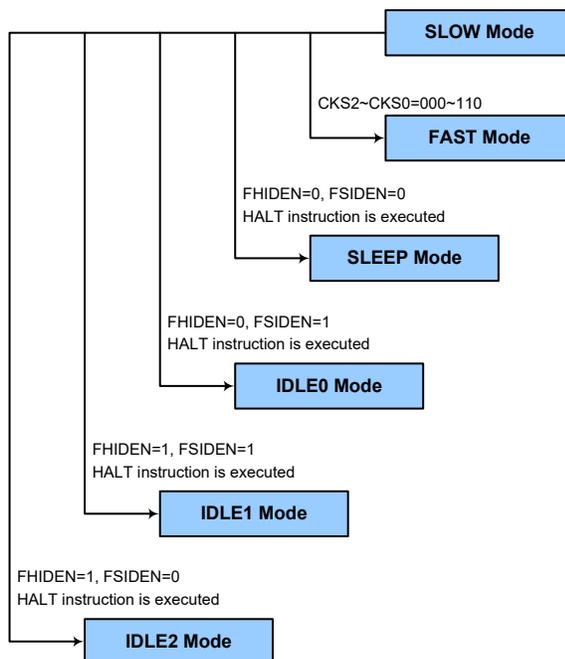
Entering the SLEEP Mode

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 if the WDT function is enabled. If the WDT function is disabled, the WDT will be cleared and then stopped.

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 if the WDT function is enabled. If the WDT function is disabled, the WDT will be cleared and then stopped.

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 if the WDT function is enabled. If the WDT function is disabled, the WDT will be cleared and then stopped.

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 if the WDT function is enabled. If the WDT function is disabled, the WDT will be cleared and then stopped.

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 except 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 set as outputs or if set as inputs must have pull-high resistors connected. In addition, the I/O pin-shared with VPP must not be set to output high, as this could

result in increased current consumption.

Care must also be taken with the loads, which are connected to I/O pins, which are set 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
- An external $\overline{RES}$ pin reset
- A system interrupt
- A WDT overflow

When the device execute the "HALT" instruction, it will enter the IDLE or SLEEP mode and the PDF flag will be set to 1. The PDF flag will be cleared to 0 if the device experience a system power-up or executes the clear Watchdog Timer instruction.

If the system is woken up by an external $\overline{RES}$ pin reset, the device will experience a full system reset, however, if the device is woken up by a WDT overflow, a Watchdog Timer reset will be initiated. Although both of these wake-up methods will initiate a reset operation, the actual source of the wake-up can be determined by examining the TO and PDF flag. The PDF flag is cleared by a system power-up or executing the clear Watchdog Timer instructions and is set when executing the "HALT" instruction. The TO flag is set if a WDT time-out occurs and causes a wake-up that only resets the Program Counter and Stack Pointer, other flags remain in their original status.

Each pin on Port A can be set 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 $[(2^8-2^0)\sim2^8]\sim[(2^{15}-2^7)\sim2^{15}]$ 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 time-out period as well as the enable/disable operation.

- **WDTC Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	WDTEN	WS2	WS1	WS0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	1	1	1	1

Bit 7~4   Unimplemented, read as "0"

Bit 3   **WDTEN:** WDT function enable control
    0: Disable
    1: Enable

Bit 2~0   **WS2~WS0:** WDT time-out period selection
    000: $[(2^8-2^0)\sim2^8]/f_{LIRC}$
    001: $[(2^9-2^1)\sim2^9]/f_{LIRC}$
    010: $[(2^{10}-2^2)\sim2^{10}]/f_{LIRC}$
    011: $[(2^{11}-2^3)\sim2^{11}]/f_{LIRC}$
    100: $[(2^{12}-2^4)\sim2^{12}]/f_{LIRC}$
    101: $[(2^{13}-2^5)\sim2^{13}]/f_{LIRC}$
    110: $[(2^{14}-2^6)\sim2^{14}]/f_{LIRC}$
    111: $[(2^{15}-2^7)\sim2^{15}]/f_{LIRC}$

These three bits determine the division ratio of the Watchdog Timer source clock, which in turn determines the timeout period.

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. With regard to the Watchdog Timer enable/disable function, there is one bit, WDTEN, in the WDTC register to offer the enable/disable control.

Under normal program operation, a Watchdog Timer time-out will initialise a device reset and set the status bit TO. 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 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 using the Watchdog Timer software clear instruction, the second is via a HALT instruction. The third is an external hardware reset, which means a low level on the external reset pin.

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^{15} 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 1 second for the 2^{15} division ratio, and a minimum timeout of 8ms for the 2^8 division ration.

Watchdog Timer
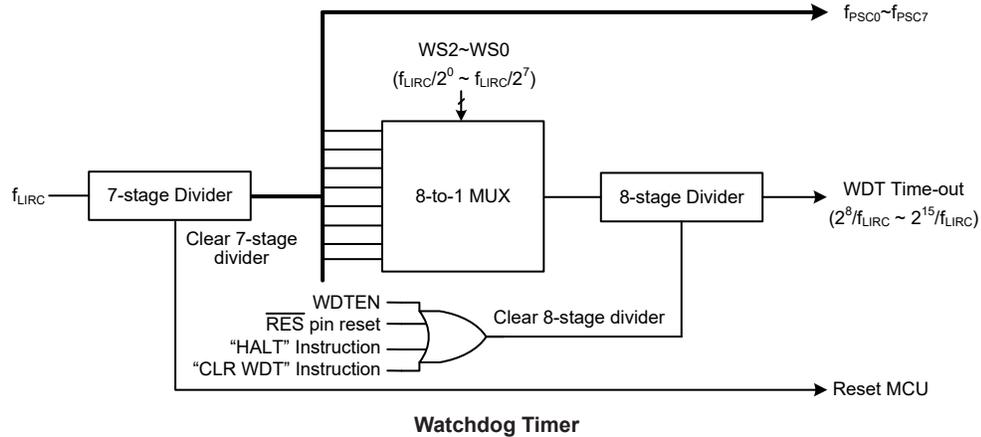
Reset and Initialisation

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, situations may arise where it is necessary to forcefully apply a reset condition when the device is running. One example of this is where after power has been applied and the device is already running, the $\overline{\text{RES}}$ line is forcefully pulled low. In such a case, known as a normal operation reset, some of the registers remain unchanged allowing the device to proceed with normal operation after the reset line is allowed to return high.

Another reset exists in the form of a Low Voltage Reset, LVR, where a full reset, similar to the $\overline{\text{RES}}$ 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 both internally and externally.

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.
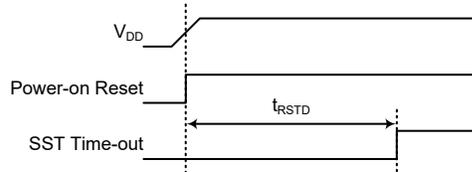

Power-on Reset Timing Chart

$\overline{\text{RES}}$ Pin Reset

As the reset pin is shared with an I/O pin, the reset function must be selected using a configuration option. Although the microcontroller has an internal RC reset function, if the V_{DD} power supply rise time is not fast enough or does not stabilise quickly at power-on, the internal reset function may be incapable of providing proper reset operation. For this reason it is recommended that an external RC network is connected to the $\overline{\text{RES}}$ pin, whose additional time delay will ensure that the $\overline{\text{RES}}$ pin remains low for an extended period to allow the power supply to stabilise. During this time delay, normal operation of the microcontroller will be inhibited. After the $\overline{\text{RES}}$ line reaches a certain voltage value, the reset delay time t_{RSTD} is invoked to provide an extra delay time after which the microcontroller will begin normal operation. The abbreviation SST in the figures stands for System Start-up Timer.

For most applications a resistor connected between VDD and the $\overline{\text{RES}}$ pin and a capacitor connected between VSS and the $\overline{\text{RES}}$ pin will provide a suitable external reset circuit. Any wiring connected to the $\overline{\text{RES}}$ pin should be kept as short as possible to minimise any stray noise interference.

For applications that operate within an environment where more noise is present the Enhanced Reset Circuit shown is recommended.
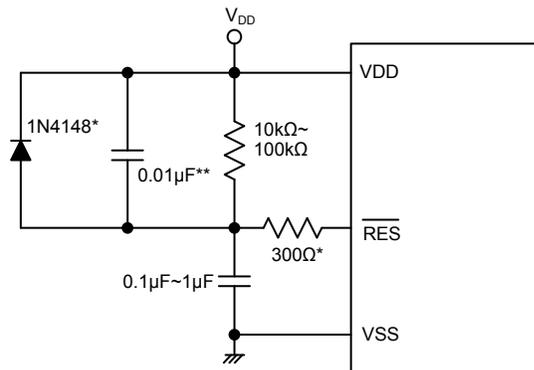

Note: * It is recommended that this component is added for added ESD protection.

   ** It is recommended that this component is added in environments where power line noise is significant.

**External $\overline{\text{RES}}$ Circuit**

Pulling the $\overline{\text{RES}}$ pin low using external hardware will also execute a device reset. In this case, as in the case of other resets, the Program Counter will reset to zero and program execution initiated from this point.

RES Reset Timing Chart
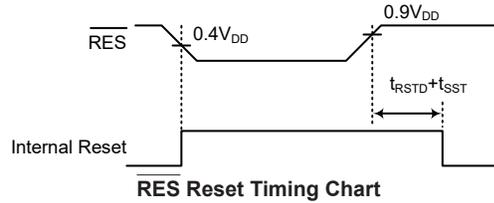
Low Voltage Reset – LVR

The microcontroller contains a low voltage reset circuit in order to monitor the supply voltage of the device and provide an MCU reset should the value fall below a certain predefined level.

The LVR function can be enabled or disabled by the LVRC control register. If the LVRC control register is configured to enable the LVR function, the LVR function is always enabled except in the SLEEP/IDLE mode. If the supply voltage of the device drop to within a range of $0.9V \sim V_{LVR}$ such as might occur when changing the battery, the LVR will automatically reset the device internally. For a valid LVR signal, a low voltage, i.e., a voltage in the range between $0.9V \sim V_{LVR}$ must exist for greater than the value t_{LVR} specified in the LVR Electrical Characteristics. If the low voltage state does not exceed this value, the LVR will ignore the low supply voltage and will not perform a reset function. The actual V_{LVR} value can be selected by the LVS7~LVS0 bits in the LVRC register. After power on the register will have the value of 01100110B. Note that the LVR function will be automatically disabled when the device enter the IDLE or SLEEP mode.

Low Voltage Reset Timing Chart
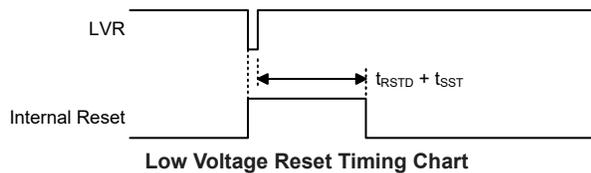
• LVRC Register

| Bit	7	6	5	4	3	2	1	0
Name	LVS7	LVS6	LVS5	LVS4	LVS3	LVS2	LVS1	LVS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	1	1	0	0	1	1	0

Bit 7~0 **LVS7~LVS0:** LVR voltage selection
01100110: 1.9V
01010101: 2.1V
00110011: 3.15V
10011001: 4.2V
11110000: LVR disable
Other values: 1.9V

When an actual low voltage condition occurs, as specified by one of the defined LVR voltage values above, an MCU reset will be generated. The reset operation will be activated after the low voltage condition keeps more than a t_{LVR} time. In this situation the register contents will remain the same after such a reset occurs.

### Watchdog Time-out Reset during Normal Operation

When the Watchdog time-out Reset during normal operations in the FAST or SLOW mode occurs, the Watchdog time-out flag TO will be set to "1".
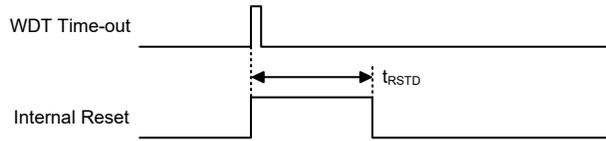
WDT Time-out Reset during Normal Operation Timing Chart

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 and PDF flags will be set to "1". Refer to the System Start Up Time Characteristics for t_{SST} details.
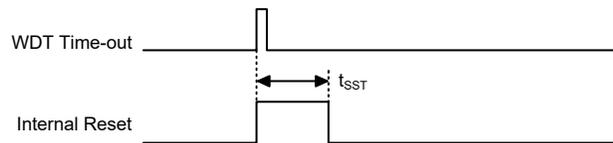
WDT Time-out Reset during SLEEP or IDLE Timing Chart

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	$\overline{RES}$ or 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 operation

"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 Base	Cleared after reset, WDT begins counting
Timer Modules	Timer Modules will be turned off
Input/Output Ports	I/O ports will be set 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.

Registers	Reset (Power On)	WDT Time-out (Normal Operation)	RES Reset (Normal Operation)	WDT Time-out (IDLE/SLEEP)
IAR0	x x x x x x x x	u u u u u u u u	u u u u u u u u	u u u u u u u u
MP0	x x x x x x x x	u u u u u u u u	u u u u u u u u	u u u u u u u u
IAR1	x x x x x x x x	u u u u u u u u	u u u u u u u u	u u u u u u u u
MP1	x x x x x x x x	u u u u u u u u	u u u u u u u u	u u u u u u u u
BP	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
ACC	x x x x x x x x	u u u u u u u u	u u u u u u u u	u u u u u u u u
PCL	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0
TBLP	x x x x x x x x	u u u u u u u u	u u u u u u u u	u u u u u u u u
TBLH	x x x x x x x x	u u u u u u u u	u u u u u u u u	u u u u u u u u
TBHP	- - - - x x x x	- - - - u u u u	- - - - u u u u	- - - - u u u u
STATUS	- - 0 0 x x x x	- - 1 u u u u u	- - u u u u u u	- - 1 1 u u u u
INTEG	- - - - 0 0 0 0	- - - - 0 0 0 0	- - - - 0 0 0 0	- - - - u u u u
WDTC	- - - - 1 1 1 1	- - - - 1 1 1 1	- - - - 1 1 1 1	- - - - u u u u
TB0C	0 - 0 0 - 0 0 0	0 - 0 0 - 0 0 0	0 - 0 0 - 0 0 0	u - u u - u u u
SCC	0 0 0 - - - 0 0	0 0 0 - - - 0 0	0 0 0 - - - 0 0	u u u - - - u u
HIRCC	- - - - 0 0 0 1	- - - - 0 0 0 1	- - - - 0 0 0 1	- - - - u u u u
INTC0	- 0 0 0 0 0 0 0	- 0 0 0 0 0 0 0	- 0 0 0 0 0 0 0	- u u u u u u u
INTC1	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
PA	1 - - 1 1 1 1 1	1 - - 1 1 1 1 1	1 - - 1 1 1 1 1	u - - u u u u u
PAC	1 - - 1 1 1 1 1	1 - - 1 1 1 1 1	1 - - 1 1 1 1 1	u - - u u u u u
PAPU	0 - - 0 0 0 0 0	0 - - 0 0 0 0 0	0 - - 0 0 0 0 0	u - - u u u u u
PAWU	0 - - 0 0 0 0 0	0 - - 0 0 0 0 0	0 - - 0 0 0 0 0	u - - u u u u u
TB1C	0 - 0 0 - 0 0 0	0 - 0 0 - 0 0 0	0 - 0 0 - 0 0 0	u - u u - u u u
TKTMR	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
TKC0	- 0 0 0 0 0 0 0	- 0 0 0 0 0 0 0	- 0 0 0 0 0 0 0	- u u u u u u u
TKC1	- - - - - - 1 1	- - - - - - 1 1	- - - - - - 1 1	- - - - - - u u
TK16DL	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
TK16DH	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
TKM0C0	0 0 0 - 0 0 0 0	0 0 0 - 0 0 0 0	0 0 0 - 0 0 0 0	u u u - u u u u
TKM0C1	0 - 0 0 0 0 0 0	0 - 0 0 0 0 0 0	0 - 0 0 0 0 0 0	u - u u u u u u
TKM016DL	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
TKM016DH	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
TKM0ROL	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
TKM0ROH	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
INTC2	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
MFI	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
PB	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	u u u u u u u u
PBC	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	u u u u u u u u
PBPU	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
SLEDC0	- - 0 0 0 0 0 0	- - 0 0 0 0 0 0	- - 0 0 0 0 0 0	- - u u u u u u
SLEDC1	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
SLEDC2	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
PDNS	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
OCR	- - - - 0 0 - -	- - - - 0 0 - -	- - - - 0 0 - -	- - - - u u - -
ODL	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
ODH	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	u u u u u u u u
PD	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	u u u u u u u u

Registers	Reset (Power On)	WDT Time-out (Normal Operation)	RES Reset (Normal Operation)	WDT Time-out (IDLE/SLEEP)
PDC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS1	00-- --00	00-- --00	00-- --00	uu-- --uu
PBS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS0	---- 0000	---- 0000	---- 0000	---- uuuu
ORMC	0000 0000	0000 0000	0000 0000	0000 0000
IECC	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
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
PTMRPL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMRPH	---- --00	---- --00	---- --00	---- --uu
SIMC0	111- --00	111- --00	111- --00	uuu- --uu
SIMC1	1000 0001	1000 0001	1000 0001	uuuu uuuu
SIMD	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SIMA	0000 0000	0000 0000	0000 0000	uuuu uuuu
SIMTOC	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADC1	0000 -000	0000 -000	0000 -000	uuuu -uuu
SADC2	0--0 0000	0--0 0000	0--0 0000	u--u uuuu
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)
VBGRC	---- ---0	---- ---0	---- ---0	---- ---u
CRCCR	---- ---0	---- ---0	---- ---0	---- ---u
CRCIN	0000 0000	0000 0000	0000 0000	uuuu uuuu
CRCDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
CRCDH	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCRL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCRH	---- -000	---- -000	---- -000	---- -uuu
CTM0C0	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM0C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM0DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM0DH	---- --00	---- --00	---- --00	---- --uu
CTM0AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM0AH	---- --00	---- --00	---- --00	---- --uu
CTM1C0	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM1C1	0000 0000	0000 0000	0000 0000	uuuu uuuu

Registers	Reset (Power On)	WDT Time-out (Normal Operation)	RES Reset (Normal Operation)	WDT Time-out (IDLE/SLEEP)
CTM1DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM1DH	---- --00	---- --00	---- --00	---- --uu
CTM1AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM1AH	---- --00	---- --00	---- --00	---- --uu
CTM2C0	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM2C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM2DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM2DH	---- --00	---- --00	---- --00	---- --uu
CTM2AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM2AH	---- --00	---- --00	---- --00	---- --uu
STKPTR	0--- -000	0--- -000	0--- -000	u--- -000
INTC3	---0 ---0	---0 ---0	---0 ---0	---u ---u
LVRC	0110 0110	0110 0110	0110 0110	uuuu uuuu

Note: "u" stands for unchanged

"x" stands for unknown

"-" stands for unimplemented

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 and PD. These I/O ports are mapped to the RAM Data Memory with specific addresses as shown in the Special Purpose Data Memory Structure diagram. 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	—	—	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	—	—	PAC4	PAC3*	PAC2	PAC1	PAC0
PAPU	PAPU7	—	—	PAPU4	PAPU3**	PAPU2	PAPU1	PAPU0
PAWU	PAWU7	—	—	PAWU4	PAWU3**	PAWU2	PAWU1	PAWU0
PB	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PBC	PBC7	PBC6	PBC5	PBC4	PBC3*	PBC2	PBC1	PBC0
PBPU	PBPU7	PBPU6	PBPU5	PBPU4	PBPU3**	PBPU2	PBPU1	PBPU0
PD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
PDC	PDC7*	PDC6*	PDC5*	PDC4*	PDC3*	PDC2	PDC1	PDC0
PDPU	PDPU7**	PDPU6**	PDPU5**	PDPU4**	PDPU3**	PDPU2	PDPU1	PDPU0

"—": Unimplemented, read as "0"

I/O Logic Function Register List

Note: The control bit with an asterisk * should be cleared to 0 and the control bit with an asterisk ** should remain the POR value after power-on reset. This can set these unbonded and not internally used lines as outputs to prevent additional power consumption.

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 a digital input have the capability of being connected to an internal pull-high resistor. These pull-high resistors are selected using the relevant pull-high control registers 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

PxPU7~PxPU0: I/O Port x pull-high function control
        0: Disable
        1: Enable
        The PxUn bit is used to control the pin pull-high function. Here the "x" is the Port name which can be A, B and D. 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	—	—	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
R/W	R/W	—	—	R/W	R/W	R/W	R/W	R/W
POR	0	—	—	0	0	0	0	0

Bit 7, 4, 2~0 **PAWU7, PAWU4, PAWU2~PAWU0**: PA7, PA4, PA2~PA0 pin Wake-up function control
        0: Disable
        1: Enable
Bit 3   **PAWU3:** This bit should remain unchanged after power-on reset
Bit 6~5   Unimplemented, read as "0"

I/O Port Control Registers

Each I/O port has its own control register known as PAC, PBC and PDC, 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 set as a CMOS output. If the pin is currently set 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 when the IECM is cleared to "0".

• 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 ,B or D. However, the actual available bits for each I/O Port may be different.
  Note that for the PB4~PB7 lines, which are internally connected to the Linear Charger SDA, FN, SCL and STDBY respectively, the corresponding control bits should be properly configured depending on the function that each line is actually used for internal connection. Set the line as an output or an input with pull-high if necessary.

I/O Port Source Current Selection

The device supports different output source current driving capability for each I/O port. With the selection registers, SLEDC0~SLEDC2, specific I/O port can support four levels of the source current driving capability. These source current selection bits are available when the corresponding pin is configured as a CMOS output. Otherwise, these selection bits have no effect. Users should refer to the Input/Output Characteristics section to select the desired output source current for different applications.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SLEDC0	—	—	SLEDC05	SLEDC04	SLEDC03	SLEDC02	SLEDC01	SLEDC00
SLEDC1	D7	D6	D5	D4	SLEDC13	SLEDC12	SLEDC11	SLEDC10
SLEDC2	D7	D6	D5	D4	SLEDC23	SLEDC22	SLEDC21	SLEDC20

"—": Unimplemented, read as "0"

I/O Port Source Current Selection Register List

• SLEDC0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	SLEDC05	SLEDC04	SLEDC03	SLEDC02	SLEDC01	SLEDC00
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 5~4 **SLEDC5~SLEDC4:** 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 3~2 **SLEDC3~SLEDC2:** PA2 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:** PA1, 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.)

• **SLEDC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	SLEDC13	SLEDC12	SLEDC11	SLEDC10
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 7~4 **D7~D4:** Reserved bits, should remain unchanged after power-on reset
Bit 3~2 **SLEDC13~SLEDC12:** PB2 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 **SLEDC11~SLEDC10:** PB1, 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.)

• **SLEDC2 Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	SLEDC23	SLEDC22	SLEDC21	SLEDC20
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 7~4 **D7~D4:** Reserved bits, should remain unchanged after power-on reset
Bit 3~2 **SLEDC3~SLEDC2:** PD2 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:** PD1, PD0 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.)

I/O Port Sink Current Selection

The device supports different output sink current driving capability for PD port. With the selection registers, PDNS, specific I/O port can support two levels of the sink current driving capability. These sink current selection bits are available when the corresponding pin is configured as a CMOS output. Otherwise, these selection bits have no effect. Users should refer to the Input/Output Characteristics section to select the desired output sink current for different applications.

• **PDNS Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	PDNS2	PDNS1	PDNS0
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 7~3   **D7~D3:** Reserved bits, should remain unchanged after power-on reset

Bit 2   **PDNS2:** PD2 sink current selection (NMOS adjust)
           0: Sink current=Level 1 (Min.)
           1: Sink current=Level 2 (Max.)

Bit 1   **PDNS1:** PD1 sink current selection (NMOS adjust)
           0: Sink current=Level 1 (Min.)
           1: Sink current=Level 2 (Max.)

Bit 0   **PDNS0:** PD0 sink current selection (NMOS adjust)
           0: Sink current=Level 1 (Min.)
           1: Sink current=Level 2 (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 a Port "x" Output Function Selection register"n", labeled as PxSn, and Input Function Selection register, labeled as iFSn, 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, a special point must be noted for the digital input pin PTPI and INTn, etc, which shares the same pin-shared control configuration with its corresponding general purpose I/O function when setting the relevant pin-shared control bit. To select this pin function, in addition to the necessary pin-shared control and peripheral functional setup aforementioned, it must also be set as an 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	Bit							
Name	7	6	5	4	3	2	1	0
PAS0	D7	D6	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	PAS17	PAS16	—	—	—	—	PAS11	PAS10
PBS0	D7	D6	D5	D4	D3	PBS02	PBS01	PBS00
PDS0	D7	D6	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
IFS0	—	—	—	—	INT0PS1	INT0PS0	SCLPS	SDAPS

"—": Unimplemented, read as "0"

Pin-shared Function Selection Register List

• **PAS0 Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	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   **D7~D6:** Reserved bits, should remain unchanged after power-on reset

Bit 5~4   **PAS05~PAS04:** PA2 pin-shared function selection
      00: PA2
      01: SCL
      10: CTP2
      11: PA2

Bit 3~2   **PAS03~PAS02:** PA1 pin-shared function selection
      00: PA1
      01: Reserved
      10: PTP
      11: CTP0

Bit 1~0   **PAS01~PAS00:** PA0 pin-shared function selection
      00: PA0/INT1
      01: SDA
      10: CTP1
      11: PA0/INT1

• **PAS1 Register**

Bit	7	6	5	4	3	2	1	0
Name	PAS17	PAS16	—	—	—	—	PAS11	PAS10
R/W	R/W	R/W	—	—	—	—	R/W	R/W
POR	0	0	—	—	—	—	0	0

Bit 7~6   **PAS17~PAS16:** PA7 pin-shared function selection
      00: PA7/VPP/$\overline{RES}$
      01: SCL
      10: PTPB
      11: CTP2B

Bit 5~2   Unimplemented, read as "0"

Bit 1~0   **PAS11~PAS10:** PA4 pin-shared function selection
      00: PA4/PTPI/INT0
      01: SDA
      10: CTP2
      11: CTP1B

Note: The PA4 and PD2 share one external pin, it is important to note that PD2 should be configured as an input with the pull-high function disabled when using these bits to select the pin function on PA4 pin.

• **PBS0 Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	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 7~3   **D7~D3:** Reserved bits, should remain unchanged after power-on reset

Bit 2   **PBS02:** PB2 Pin-Shared function selection
          0: PB2
          1: KEY3

Bit 1   **PBS01:** PB1 Pin-Shared function selection
          0: PB1
          1: KEY2

Bit 0   **PBS00:** PB0 Pin-Shared function selection
          0: PB0
          1: KEY1

• **PDS0 Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
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:** Reserved bits, should remain unchanged after power-on reset

Bit 5~4   **PDS05~PDS04:** PD2 pin-shared function selection
          00: PD2
          01: AN2
          10: CTP2
          11: CTP1B
        Note: The PA4 and PD2 share one external pin, it is important to note that PA4 should
             be configured as an input with the pull-high function disabled when using these
             bits to select the pin function on PD2 pin.

Bit 3~2   **PDS03~PDS02:** PD1 pin-shared function selection
          00: PD1/INT0
          01: AN1
          10: VREFI
          11: CTP1

Bit 1~0   **PDS01~PDS00:** PD0 pin-shared function selection
          00: PD0
          01: AN0
          10: VREF
          11: CTP0

• **IFS0 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	INT0PS1	INT0PS0	SCLPS	SDAPS
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   **INT0PS1~INT0PS0:** INT0 input source pin selection
          00: PA4
          01: PD1
          10: Reserved
          11: Reserved

Bit 1   **SCLPS:** SCL input source pin selection
         0: PA2
         1: PA7

Bit 0   **SDAPS:** SDA input source pin selection
         0: PA0
         1: PA4

I/O Pin Structure

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

Logic Function Input/Output Structure
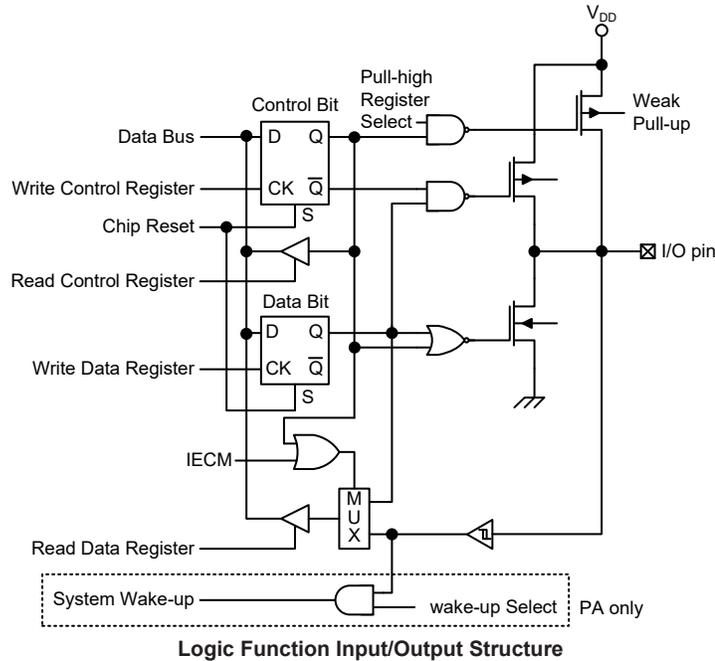
READ PORT Function

The READ PORT function is used to read data from I/O pins, which is specially designed for the IEC 60730 self-diagnostic test on the I/O function and A/D paths. After the self-diagnostic test on the I/O function and A/D paths are completed, users must disable the READ PORT function immediately. In cases other than the I/O function and A/D path self-diagnostic test, it is strongly recommended to disable the READ PORT function to avoid affecting other peripheral functions and causing unexpected consequences.

There is a register, IECC, which is used to control the READ PORT function. When a specific data pattern, "11001010", is written into the IECC register, the internal signal named IECM will be set high to enable the READ PORT function. If the READ PORT function is enabled, the reading path is from the I/O pins. The value on the corresponding pins will be passed to the accumulator ACC when the read port instruction "mov a, Px" is executed, where the "x" stands for the corresponding I/O port name. However, when the IECC register content is set to any other values rather than "11001010", the IECM internal signal will be cleared to 0 to disable the READ PORT function, and the reading path will be from the data latch or I/O pins. If the READ PORT function is disabled, the pin function will operate as the selected pin-shared function.

• **IECC Register**

Bit	7	6	5	4	3	2	1	0
Name	IECS7	IECS6	IECS5	IECS4	IECS3	IECS2	IECS1	IECS0
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   **IECS7~IECS0:** READ PORT function enable control bit 7~ bit 0
            11001010: IECM=1 – READ PORT function is enabled
            Others: IECM=0 – READ PORT function is disabled

READ PORT Function	Disabled		Enabled	
Port Control Register Bit – PxC.n	1	0	1	0
I/O Function	Pin value	Data latch value	Pin value	
A/D Function	0			

Note: The value on the above table is the content of the ACC register after "mov a, Px" instruction is executed where "x" means the relevant port name.

The additional function of the READ PORT mode is to check the A/D path. When the READ PORT function is disabled, the A/D path from the external pin to the internal analog input will be switched off if the A/D input pin function is not selected by the corresponding selection bits. For the MCU with A/D converter channels, the desired A/D channel can be switched on by properly configuring the external analog input channel selection bits in the A/D Control Register together with the corresponding analog input pin function is selected. However, the additional function of the READ PORT mode is to force the A/D path to be switched on. As shown in the following example, when the AN0 is selected as the analog input channel as the READ PORT function is enabled, the AN0 analog input path will be switched on even if the AN0 analog input pin function is not selected. In this way, the AN0 analog input path can be examined by internally connecting the digital output on this shared pin with the AN0 analog input pin switch and then converting the corresponding digital data without any external analog input voltage connected.

Note that the A/D converter reference voltage should be equal to the I/O power supply voltage when examining the A/D path using the READ PORT function.
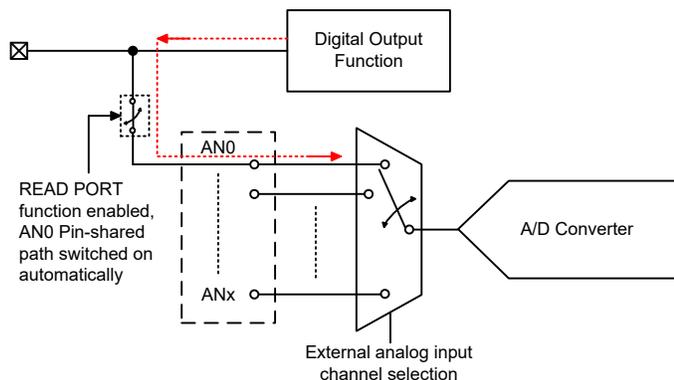
A/D Channel Input Path Internal Connection

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 set 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 function. 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 set to have this function.

Timer Modules – TM

One of the most fundamental functions in any microcontroller devices is the ability to control and measure time. To implement time related functions the device includes several Timer Modules, generally 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 interrupts. The addition of input and output pins for 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 Compact and Periodic TM sections.

Introduction

The device contains several TM units and each individual TM can be categorised as a certain type, namely Compact 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 Compact and Periodic Type TMs will be described in this section and 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.

TM Function	CTM	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 TMs 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 comparator. 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.

TM Clock Source

The clock source which drives the main counter in the each TM can originate from various sources. The selection of the required clock source is implemented using the xTnCK2~xTnCK0 bits in the xTMn control registers, where "x" stands for C or P type TM and "n" stands for the specific TM serial number. For the PTM there is no serial number "n" in the relevant pins, registers and control bits since there is only one PTM in the device. 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.

TM Interrupts

The Compact or Periodic type TM each has two internal interrupt, 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

The CTM has no external input pins, while the PTM has one input pin with the label PTPI. The PTM input pin, PTPI, 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 PTIO1~PTIO0 bits in the PTMC1 register.

The TMs each has two output pins with the label xTPn and xTPnB. 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 xTPnB is the inverted signal of the xTPn output. The external output pins are also the pins where the TM generates the PWM output waveform.

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

CTM		PTM	
Input	**Output**	**Input**	**Output**
—	CTP0, CTP1, CTP1B CTP2, CTP2B	PTPI	PTP,PTPB

**TM External Pins**
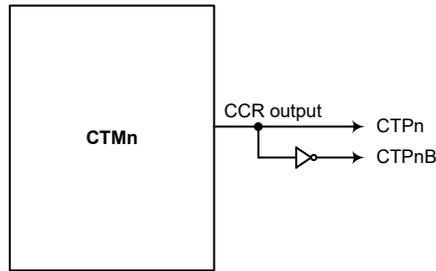
CTMn Function Pin Block Diagram (n=0~2)
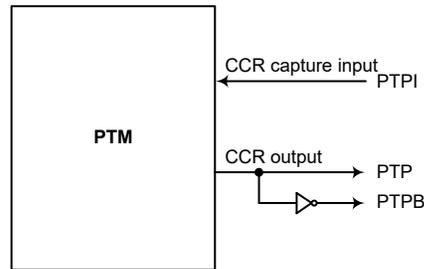
PTM Function Pin Block Diagram

Programming Considerations

The TM Counter Registers and the Capture/Compare CCRA and CCRP 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 and CCRP 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 and CCRP low byte registers, named xTMnAL and PTMRPL, using the following access procedures. Accessing the CCRA or CCRP low byte registers without following these access procedures will result in unpredictable values.
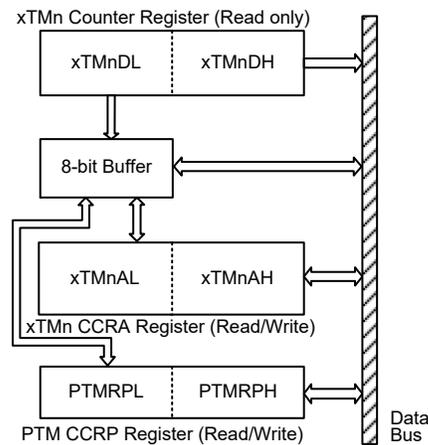
The following steps show the read and write procedures:

- Writing Data to CCRA or CCRP
  - Step 1. Write data to Low Byte xTMAL or PTMRPL
    - Note that here data is only written to the 8-bit buffer.
  - Step 2. Write data to High Byte xTMAH 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 or CCRP
  - Step 1. Read data from the High Byte xTMDH, xTMAH 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 or PTMRPL
    - This step reads data from the 8-bit buffer.

Compact Type TM – CTM

The Compact Type TM contains three operating modes, which are Compare Match Output, Timer Counter and PWM Output modes.
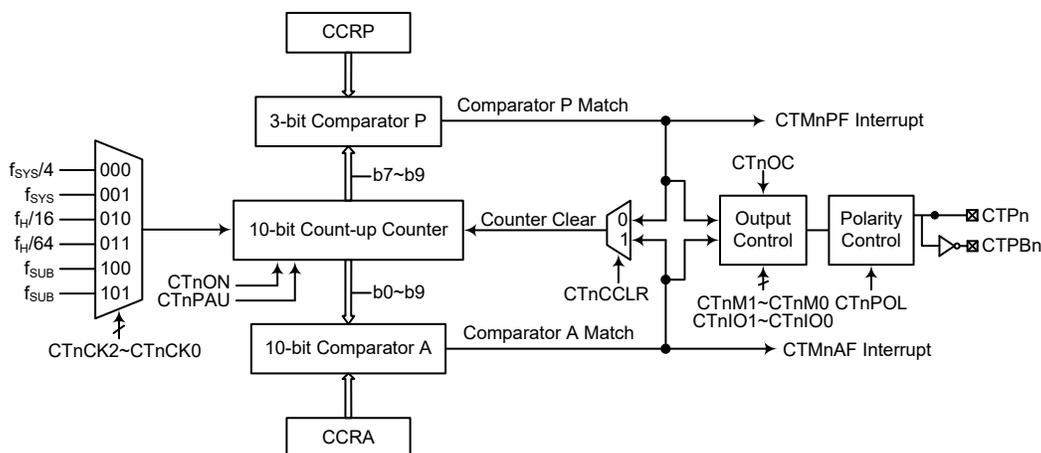
Note: 1. The CTMn external pins are pin-shared with other functions, therefore before using the CTMn function, ensure that the pin-shared function registers have been set properly to enable the CTMn pin function.
2. The CTPnB is the inverted signal of the CTPn, where the CTP0B signal is not connected to the external package.
3. n=0~2.

10-bit Compact Type TM Block Diagram

Compact Type TM Operation

The size of Compact 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 the 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 CTnON 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 CTMn interrupt signal will also usually be generated. The Compact Type TM can operate in a number of different operational modes, can be driven by different clock sources and can also control two output pins. All operating setup conditions are selected using relevant internal registers.

Compact Type TM Register Description

Overall operation of the Compact Type 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 the CCRP bits.

Register Name	Bit							
	7	6	5	4	3	2	1	0
CTMnC0	CTPnAU	CTnCK2	CTnCK1	CTnCK0	CTnON	CTnRP2	CTnRP1	CTnRP0
CTMnC1	CTnM1	CTnM0	CTnIO1	CTnIO0	CTnOC	CTnPOL	CTnDPX	CTnCCLR
CTMnDL	D7	D6	D5	D4	D3	D2	D1	D0
CTMnDH	—	—	—	—	—	—	D9	D8
CTMnAL	D7	D6	D5	D4	D3	D2	D1	D0
CTMnAH	—	—	—	—	—	—	D9	D8

"—": Unimplemented, read as "0"

10-bit Compact Type TM Register List (n=0~2)

- **CTMnC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	CTnPAU	CTnCK2	CTnCK1	CTnCK0	CTnON	CTnRP2	CTnRP1	CTnRP0
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   **CTnPAU:** CTMn 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 CTMn 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   **CTnCK2~CTnCK0:** CTMn counter clock selection

000: $f_{SYS}/4$

001: f_{SYS}

010: $f_H/16$

011: $f_H/64$

100: f_{SUB}

101: f_{SUB}

110: Reserved

111: Reserved

These three bits are used to select the clock source for the CTMn. 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 Operating Modes and System Clocks section.

Bit 3   **CTnON:** CTMn counter on/off control

0: Off

1: On

This bit controls the overall on/off function of the CTMn. Setting the bit high enables the counter to run while clearing the bit disables the CTMn. Clearing this bit to zero will stop the counter from counting and turn off the CTMn 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 CTMn is in the Compare Match Output Mode or the PWM Output Mode then the CTMn output pin will be reset to its initial condition, as specified by the CTnOC bit, when the CTnON bit changes from low to high.

Bit 2~0   **CTnRP2~CTnRP0:** CTMn CCRP 3-bit register, compared with the CTMn counter bit 9 ~ bit 7

Comparator P match period=
  000: 1024 CTMn clocks
  001: 128 CTMn clocks
  010: 256 CTMn clocks
  011: 384 CTMn clocks
  100: 512 CTMn clocks
  101: 640 CTMn clocks
  110: 768 CTMn clocks
  111: 896 CTMn 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 CTnCCLR bit is set to zero. Setting the CTnCCLR 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

- **CTMnC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	CTnM1	CTnM0	CTnIO1	CTnIO0	CTnOC	CTnPOL	CTnDPX	CTnCCLR
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   **CTnM1~CTnM0:** CTMn operating mode selection
  00: Compare Match Output Mode
  01: Undefined
  10: PWM Output Mode
  11: Timer/Counter Mode

These bits setup the required operating mode for the CTMn. To ensure reliable operation the CTMn should be switched off before any changes are made to the CTnM1 and CTnM0 bits. In the Timer/Counter Mode, the CTMn output pin state is undefined.

Bit 5~4   **CTnIO1~CTnIO0:** CTMn external pin function selection

Compare Match Output Mode
  00: No change
  01: Output low
  10: Output high
  11: Toggle output

PWM Output Mode
  00: PWM output inactive state
  01: PWM output active state
  10: PWM output
  11: Undefined

Timer/Counter Mode
   Unused

These two bits are used to determine how the CTMn external pins change state when a certain condition is reached. The function that these bits select depends upon in which mode the CTMn is running.

In the Compare Match Output Mode, the CTnIO1 and CTnIO0 bits determine how the CTMn output pin changes state when a compare match occurs from the Comparator A. The CTMn 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 CTMn output pin should be setup using the CTnOC bit in the CTMnC1 register. Note that the output level requested by the CTnIO1 and CTnIO0 bits must be different from the initial value setup using the CTnOC bit otherwise no change will occur on the CTMn output pin when a compare match occurs. After the CTMn output pin changes state, it can be reset to its initial level by changing the level of the CTnON bit from low to high.

In the PWM Output Mode, the CTnIO1 and CTnIO0 bits determine how the CTMn 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 CTnIO1 and CTnIO0 bits only after the CTMn has been switched off. Unpredictable PWM outputs will occur if the CTnIO1 and CTnIO0 bits are changed when the CTMn is running.

Bit 3   **CTnOC:** CTMn CTPn 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 CTMn output pin. Its operation depends upon whether CTMn is being used in the Compare Match Output Mode or in the PWM Output Mode. It has no effect if the CTMn is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the CTMn output pin before a compare match occurs. In the PWM Output Mode it determines if the PWM signal is active high or active low.

Bit 2   **CTnPOL:** CTMn CTPn output polarity control
   0: Non-inverted
   1: Inverted

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

Bit 1   **CTnDPX:** CTMn 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   **CTnCCLR:** CTMn 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 CTMn contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the CTnCCLR 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 CTnCCLR bit is not used in the PWM Output mode.

• **CTMnDL 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:** CTMn Counter Low Byte Register bit 7~ bit 0
CTMn 10-bit Counter bit 7 ~ bit 0

• **CTMnDH 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:** CTMn Counter High Byte Register bit 1 ~ bit 0
CTMn 10-bit Counter bit 9 ~ bit 8

• **CTMnAL 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:** CTMn CCRA Low Byte Register bit 7 ~ bit 0
CTMn 10-bit CCRA bit 7 ~ bit 0

• **CTMnAH 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:** CTMn CCRA High Byte Register bit 1 ~ bit 0
CTMn 10-bit CCRA bit 9 ~ bit 8

Compact Type TM Operation Modes

The Compact Type TM can operate in one of three operating modes, Compare Match Output Mode, PWM Output Mode or Timer/Counter Mode. The operating mode is selected using the CTnM1 and CTnM0 bits in the CTMnC1 register.

Compare Match Output Mode

To select this mode, bits CTnM1 and CTnM0 in the CTMnC1 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 CTnCCLR 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 CTMnAF and CTMnPF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

If the CTnCCLR bit in the CTMnC1 register is high then the counter will be cleared when a compare

match occurs from Comparator A. However, here only the CTMnAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when CTnCCLR is high no CTMnPF interrupt request flag will be generated. If the CCRA bits are all zero, the counter will overflow when it reaches its maximum 10-bit, 3FF Hex, value. However, here the CTMnAF interrupt request flag will not be generated.

As the name of the mode suggests, after a comparison is made, the CTMn output pin, will change state. The CTMn output pin condition however only changes state when a CTMnAF interrupt request flag is generated after a compare match occurs from Comparator A. The CTMnPF interrupt request flag, generated from a compare match occurs from Comparator P, will have no effect on the CTMn output pin. The way in which the CTMn output pin changes state are determined by the condition of the CTnIO1 and CTnIO0 bits in the CTMnC1 register. The CTMn output pin can be selected using the CTnIO1 and CTnIO0 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 CTMn output pin, which is setup after the CTnON bit changes from low to high, is setup using the CTnOC bit. Note that if the CTnIO1 and CTnIO0 bits are zero then no pin change will take place.

Compare Match Output Mode – CTnCCLR=0
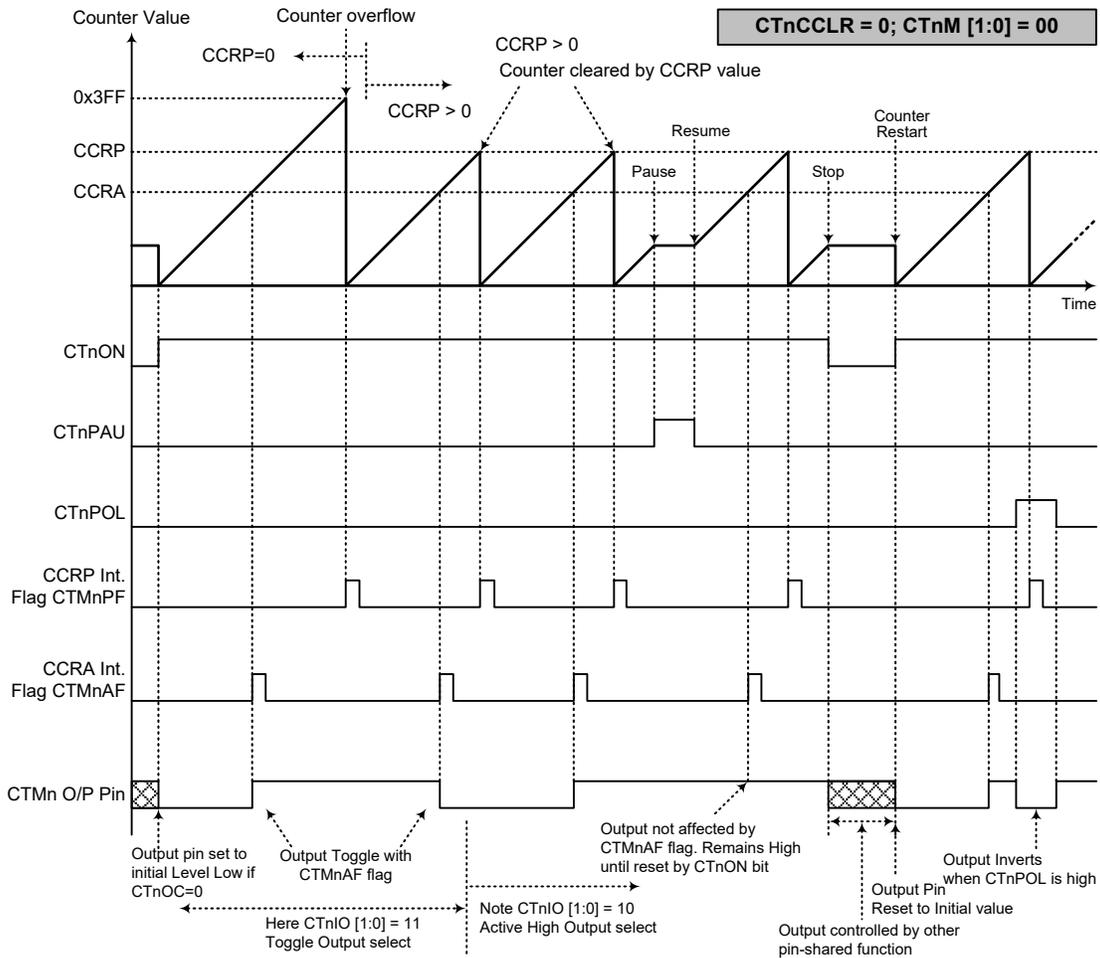
Note: 1. With CTnCCLR=0 a Comparator P match will clear the counter

2. The CTMn output pin is controlled only by the CTMnAF flag

3. The output pin is reset to initial state by a CTnON bit rising edge

4. n=0~2
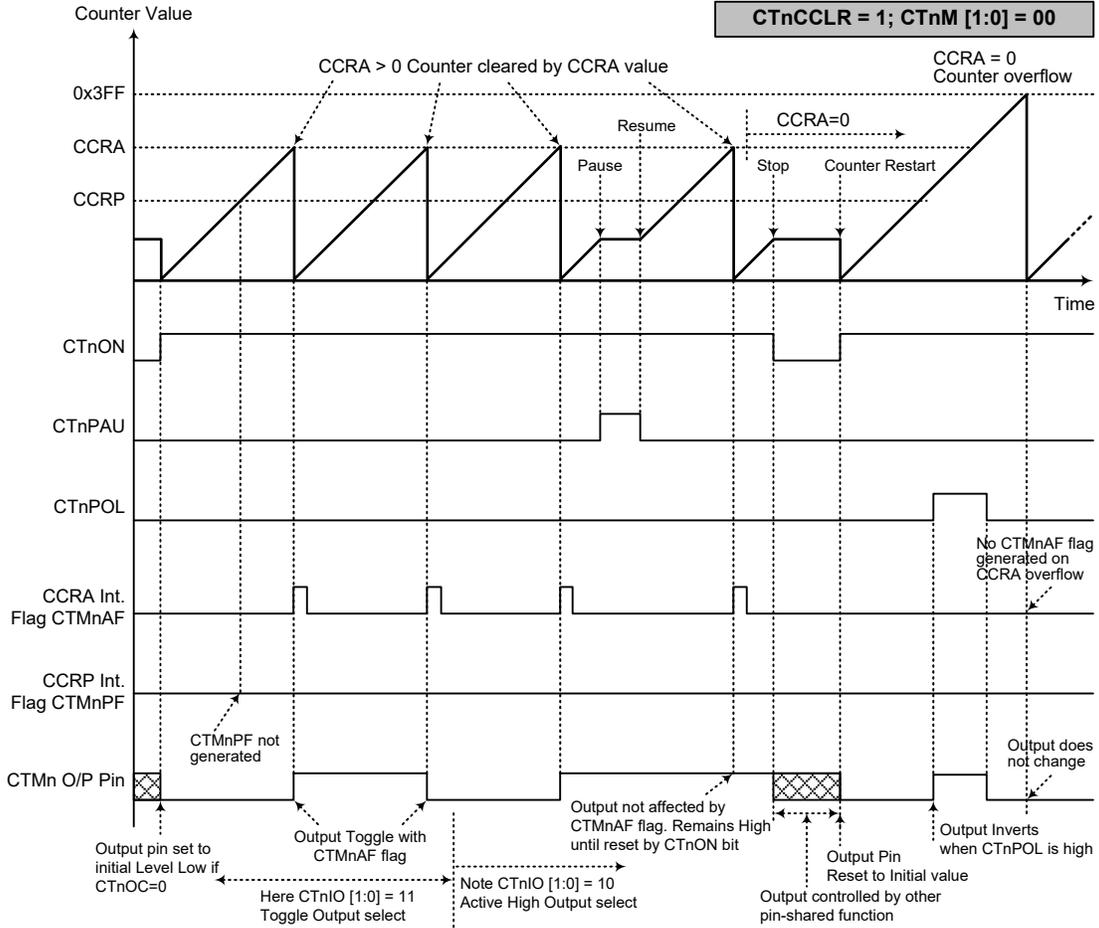
Compare Match Output Mode – CTnCCLR=1

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

    2. The CTMn output pin is controlled only by the CTMnAF flag

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

    4. A CTMnPF flag is not generated when CTnCCLR=1

    5. n=0~2

Timer/Counter Mode

To select this mode, bits CTnM1 and CTnM0 in the CTMnC1 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 CTMn 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 CTMn 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 CTnM1 and CTnM0 in the CTMnC1 register should be set to 10 respectively. The PWM function within the CTMn 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 CTMn 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 CTnCCLR 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 CTnDPX bit in the CTMnC1 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 CTnOC bit in the CTMnC1 register is used to select the required polarity of the PWM waveform while the two CTnIO1 and CTnIO0 bits are used to enable the PWM output or to force the CTMn output pin to a fixed high or low level. The CTnPOL bit is used to reverse the polarity of the PWM output waveform.

- **10-bit CTMn, PWM Output Mode, Edge-aligned Mode, CTnDPX=0**

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

If f_{SYS}=8MHz, CTMn clock source is f_{SYS}/4, CCRP=4 and CCRA=128,

The CTMn PWM output frequency=(f_{SYS}/4)/(4×128)=f_{SYS}/2048=4kHz, 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 CTMn, PWM Output Mode, Edge-aligned Mode, CTnDPX=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 CTMn clock while the PWM duty cycle is defined by the CCRP register value.

PWM Output Mode – CTnDPX=0
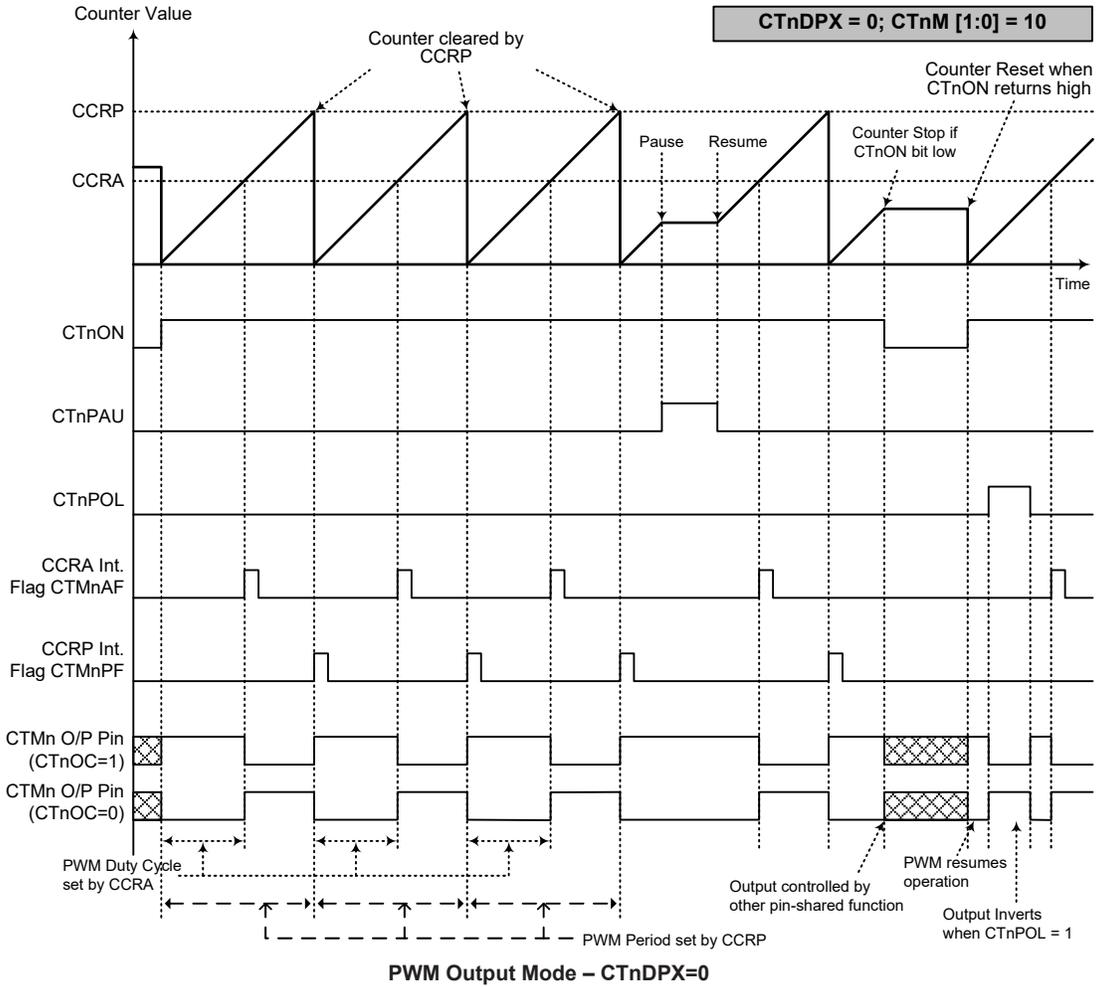
Note: 1. Here CTnDPX=0 – Counter cleared by CCRP

2. A counter clear sets the PWM Period

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

4. The CTnCCLR bit has no influence on PWM operation

5. n=0~2

PWM Output Mode – CTnDPX=1
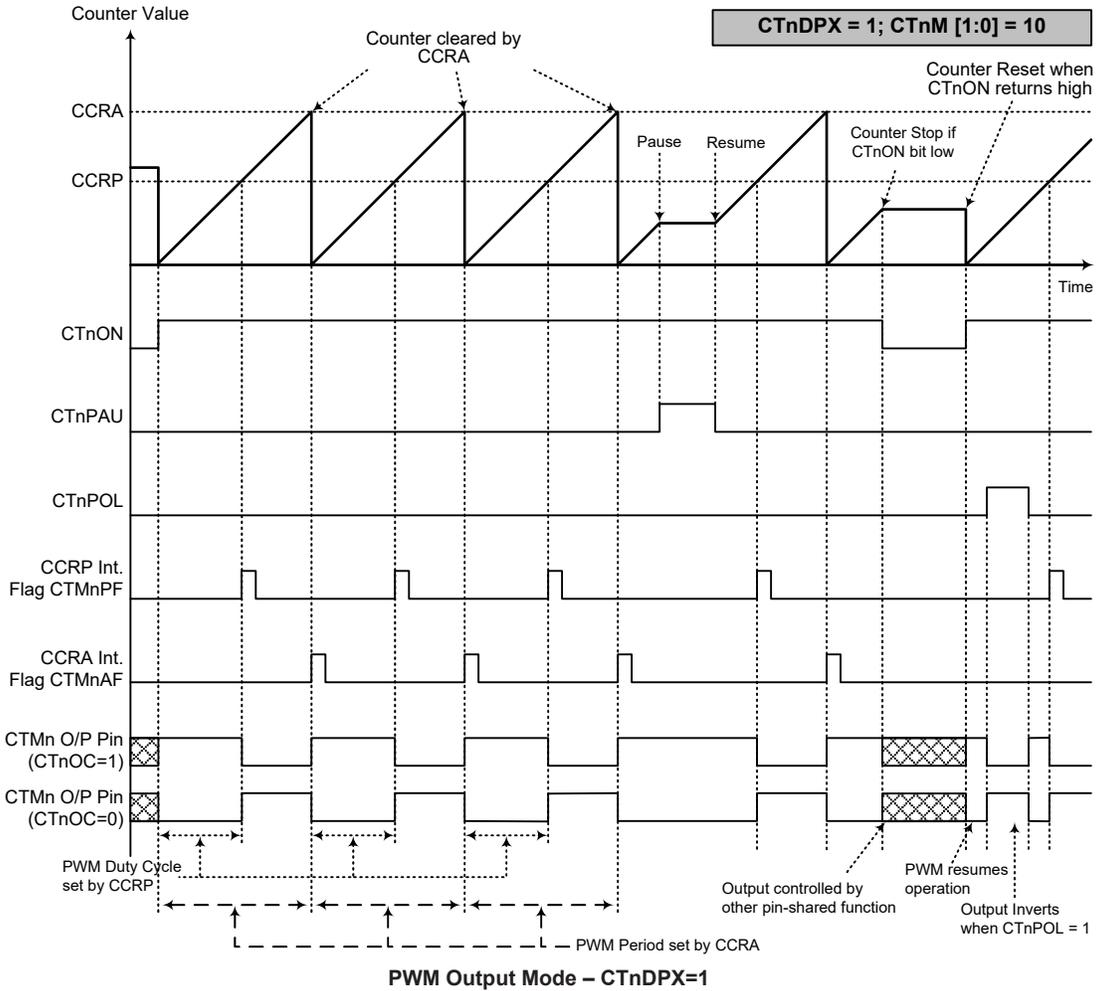
Note: 1. Here CTnDPX=1 – Counter cleared by CCRP

    2. A counter clear sets the PWM Period

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

    4. The CTnCCLR bit has no influence on PWM operation

    5. n=0~2

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 also be controlled with an external input pin and can drive two external output pins.
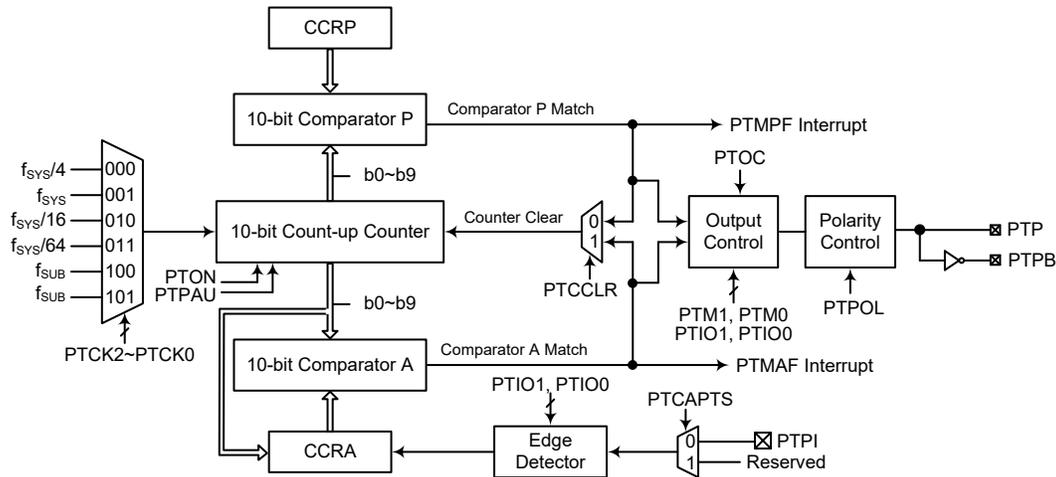
Note: 1. The PTM external pins are pin-shared with other functions, therefore before using the PTM function the pin-shared function registers must be set properly to enable the PTM pin function. The PTPI pin, if used, must also be set as an input by setting the corresponding bits in the port control register.

   2. The PTPB is the inverted signal of the PTP.

10-bit Periodic Type TM Block Diagram

Periodic Type TM Operation

The size of Periodic Type 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 and CCRA comparators are 10-bit wide whose value is respectively compared with 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 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. 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, can be driven by different clock sources and can also control two output pins. All operating setup conditions are selected using relevant internal registers.

Periodic Type TM Register Description

Overall operation of the Periodic TM is controlled using a series of registers. A read only register pair exists to store the internal counter 10-bit value, while two read/write register pairs exist to store the internal 10-bit CCRA and CCRP value. The remaining two 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
PTMDL	D7	D6	D5	D4	D3	D2	D1	D0
PTMDH	—	—	—	—	—	—	D9	D8
PTMAL	D7	D6	D5	D4	D3	D2	D1	D0
PTMAH	—	—	—	—	—	—	D9	D8
PTMRPL	D7	D6	D5	D4	D3	D2	D1	D0
PTMRPH	—	—	—	—	—	—	D9	D8

"—": Unimplemented, read as "0"

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:** PTM counter clock selection

    000: $f_{SYS}/4$
    001: f_{SYS}
    010: $f_H/16$
    011: $f_H/64$
    100: f_{SUB}
    101: f_{SUB}
    110: Reserved
    111: Reserved

These three bits are used to select the clock source for the PTM. 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 "Operating Modes and System Clocks" 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 while 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:** PTM operating mode selection

         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:** PTM external pin function selection

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 PTPI

         01: Input capture at falling edge of PTPI

         10: Input capture at rising/falling edge of PTPI

         11: Input capture disabled

Timer/Counter Mode

         Unused

These two bits are used to determine how the PTM external pins change state 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 pin changes state when a compare match occurs from the Comparator A. The PTM 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 PTM output pin 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 pin when a compare match occurs. After the PTM output pin 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 TM output pin changes state when a certain compare match condition occurs. The PTM 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 PTM 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

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 pin. 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 pin 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 pin when the PTON bit changes from low to high.

Bit 2   **PTPOL:** PTM PTP output polarity control

    0: Non-inverted

    1: Inverted

This bit controls the polarity of the PTP output pin. When the bit is set high the PTM output pin 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 pin

    1: Reserved

Bit 0   **PTCCLR:** PTM 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 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, Single Pulse Output or Capture Input Mode.

• **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

• **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 Operation 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 cannot be cleared 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 PTMAF interrupt request flag will not be generated.

As the name of the mode suggests, after a comparison is made, the PTM output pin will change state. The PTM output pin 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 pin. The way in which the PTM output pin changes state are determined by the condition of the PTIO1 and PTIO0 bits in the PTMC1 register. The PTM output pin 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 pin, 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 pin change will take place.
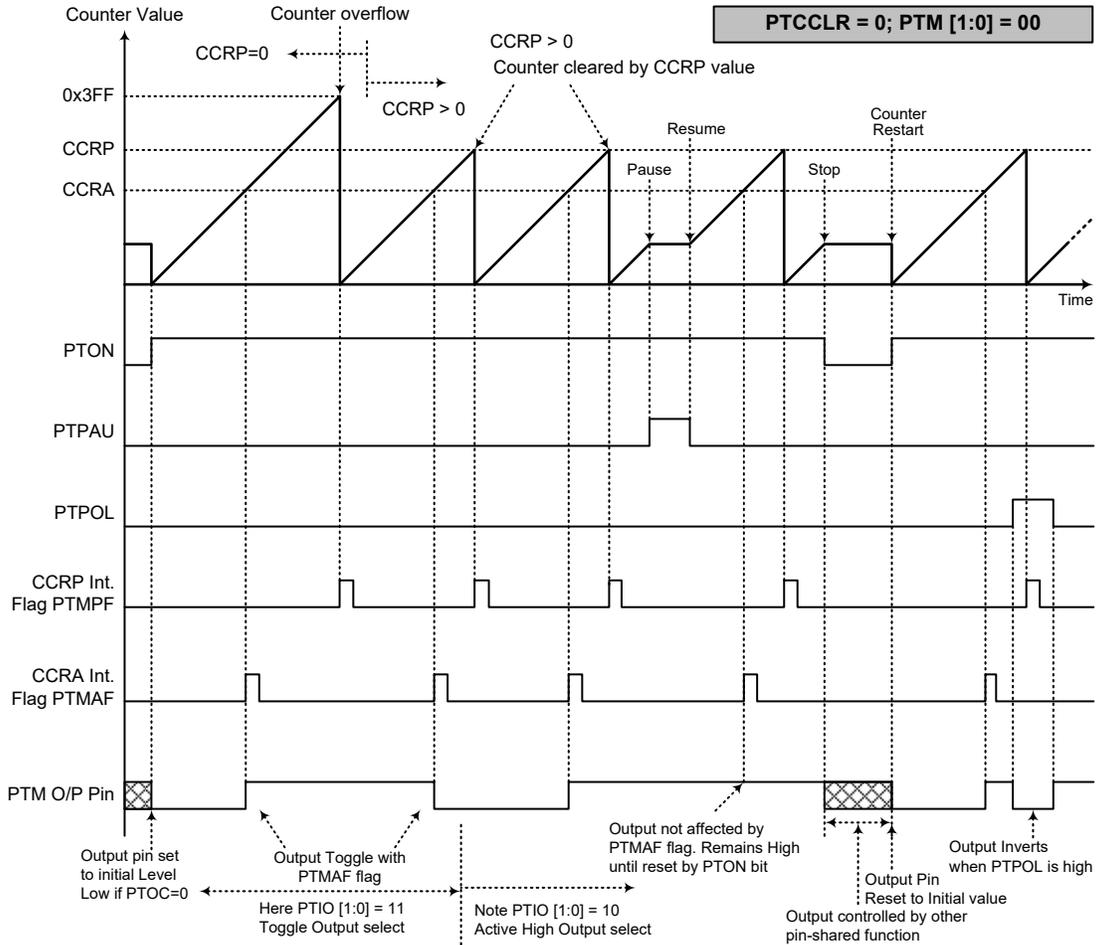
Compare Match Output Mode – PTCCLR=0

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

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

3. The output pin is reset to its initial state by a PTON bit rising edge
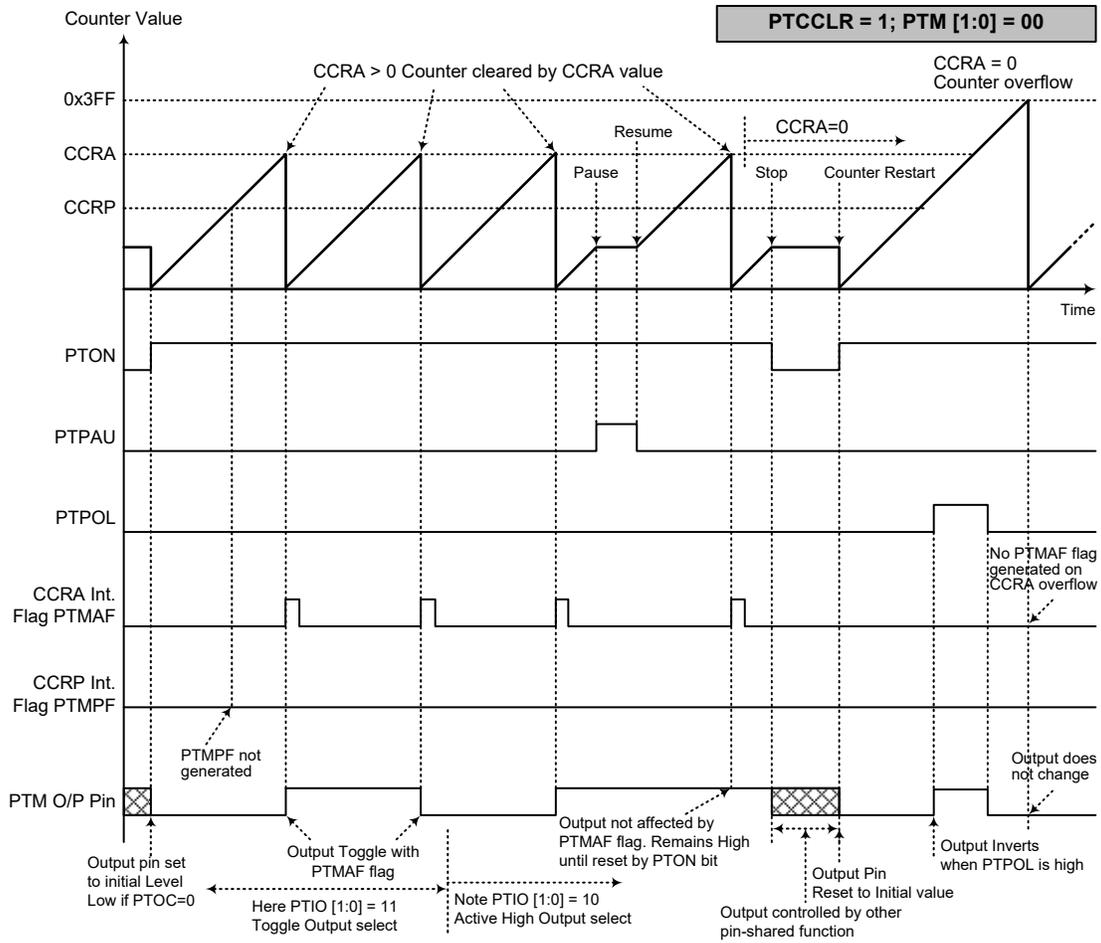
Compare Match Output Mode – PTCCLR=1

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

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

3. The output pin 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 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 PTM 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 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 as the PWM period. Both of the CCRP and CCRA registers are used to generate the PWM waveform, the CCRP register is used to clear the internal counter and thus control the PWM waveform frequency, while the CCRA register 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 pin 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=2kHz, 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. The counter is 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
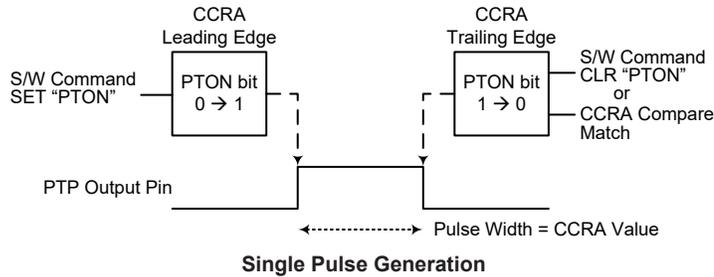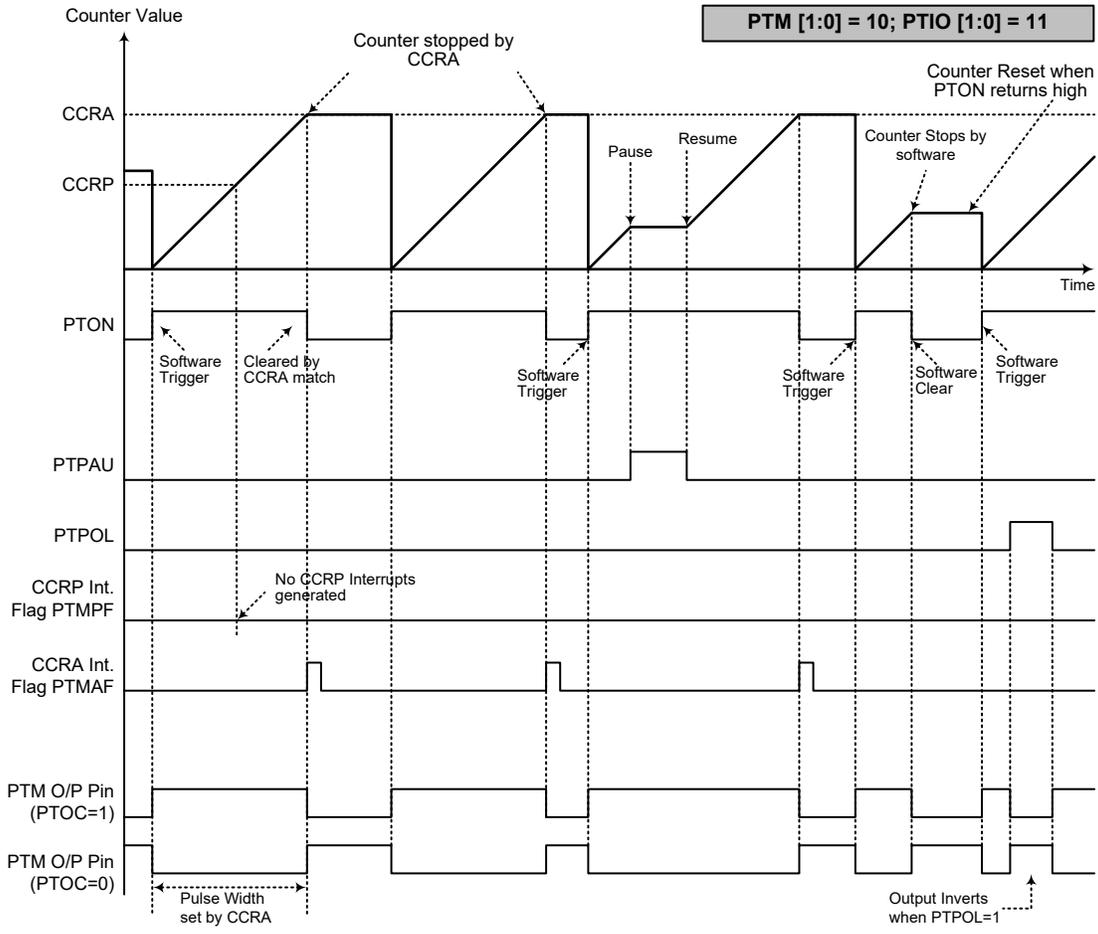
4. The PTCCLR bit has no influence on PWM operation

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. 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 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 triggered by setting the PTON bit high

4. In the Single Pulse Output 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 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 PTPI pin, selected by the PTCAPTS bit in the PTMC1 register. The input pin 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.

When the required edge transition appears on the PTPI pin the present value in the counter will be latched into the CCRA registers and a PTM interrupt generated. Irrespective of what events occur on the PTPI pin 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 PTPI pin 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 PTPI pin, however it must be noted that the counter will continue to run. The PTCCLR, PTOC and PTPOL bits are not used in this Mode.

There are some considerations that should be noted. If the captured pulse width is less than 2 timer clock periods, it may be ignored by hardware. After the counter value is latched to the CCRA registers by an active capture edge, the PTMAF flag will be set high after 0.5 timer clock periods. The delay time from the active capture edge received to the action of latching counter value to CCRA registers is less than 1.5 timer clock periods.
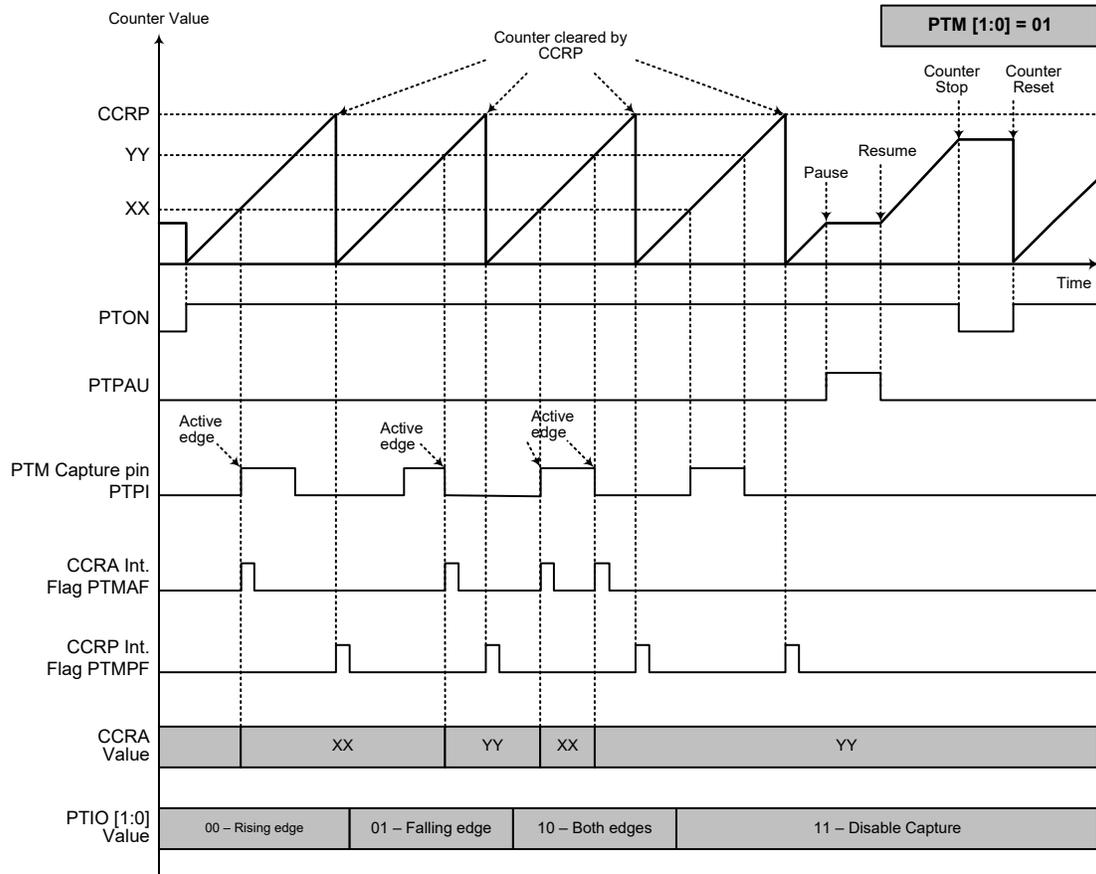
Capture Input Mode

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

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

3. PTCCLR bit not used

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

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

6. The capture input mode cannot be used if the selected PTM counter clock is not available

Touch Key Function

The device provides multiple touch key functions. The touch key function is fully integrated and requires no external components, allowing touch key functions to be implemented by the simple

Touch Key Structure

The touch keys are pin-shared with the I/O pins, with the desired function chosen via the corresponding selection register bits. Keys are organised into 1 group known as a module and having a module number, M0. The module is a fully independent set of three Touch Keys and each Touch Key has its own oscillator. The module contains its own control logic circuits and register set. Examination of the register names will reveal the module number it is referring to.

Total Key Number	Touch Key Module	Touch Key	Shared I/O Pin
3	M0	KEY1~KEY3	PB0~PB2

Touch Key Structure

Touch Key Register Definition

The touch key module, which contains three touch key functions, has its own suite registers. The following table shows the register set for the touch key module. The Mn within the register name refers to the Touch Key module number. The device has one Touch Key Module.

Register Name	Description
TKTMR	Touch key time slot 8-bit counter preload register
TKC0	Touch key function Control register 0
TKC1	Touch key function Control register 1
TK16DL	Touch key function 16-bit counter low byte
TK16DH	Touch key function 16-bit counter high byte
TKMn16DL	Touch key module n 16-bit C/F counter low byte
TKMn16DH	Touch key module n 16-bit C/F counter high byte
TKMnROL	Touch key module n reference oscillator capacitor select low byte
TKMnROH	Touch key module n reference oscillator capacitor select high byte
TKMnC0	Touch key module n Control register 0
TKMnC1	Touch key module n Control register 1

Touch Key Function Register Definition (n=0)

Register Name	Bit							
	7	6	5	4	3	2	1	0
TKTMR	D7	D6	D5	D4	D3	D2	D1	D0
TKC0	—	TKRCOV	TKST	TKCFOV	TK16OV	TSCS	TK16S1	TK16S0
TKC1	—	—	—	—	—	—	TKFS1	TKFS0
TK16DL	D7	D6	D5	D4	D3	D2	D1	D0
TK16DH	D15	D14	D13	D12	D11	D10	D9	D8
TKM0C0	M0MXS1	M0MXS0	M0DFEN	—	M0SOFC	M0SOF2	M0SOF1	M0SOF0
TKM0C1	M0TSS	—	M0ROEN	M0KOEN	D3	M0K3EN	M0K2EN	M0K1EN
TKM016DL	D7	D6	D5	D4	D3	D2	D1	D0
TKM016DH	D15	D14	D13	D12	D11	D10	D9	D8
TKM0ROL	D7	D6	D5	D4	D3	D2	D1	D0
TKM0ROH	—	—	—	—	—	—	D9	D8

"—": Unimplemented, read as "0"

Touch Key Function Register List

- **TKTMR 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:** Touch key time slot 8-bit counter preload register

Time slot counter overflow time=(256 - TKTMR[7:0])×32t_{TSC}, where t_{TSC} is the time slot counter clock period.

- **TKC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	TKRCOV	TKST	TKCFOV	TK16OV	TSCS	TK16S1	TK16S0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R
POR	—	0	0	0	0	0	0	0

Bit 7   Unimplemented, read as "0"

Bit 6   **TKRCOV:** Touch key time slot counter overflow flag

     0: No overflow occurs

     1: Overflow occurs

This bit can be accessed by application program. When this bit is set by touch key time slot counter overflow, the corresponding touch key interrupt request flag will be set. However, if this bit is set by application program, the touch key interrupt request flag will not be affected. Therefore, this bit cannot be set by application program but must be cleared to 0 by application program.

If the module 0 or all module time slot counter, selected by the TSCS bit, overflows, the TKRCOV bit and the Touch Key Interrupt request flag, TKMF, will be set and all module key oscillators and reference oscillators will automatically stop. The touch key module 16-bit C/F counter, touch key function 16-bit counter, 5-bit time slot unit period counter and 8-bit time slot counter will be automatically switched off.

Bit 5   **TKST:** Touch key detection Start control

     0: Stopped or no operation

     0→1: Start detection

In all modules the touch key module 16-bit C/F counter, touch key function 16-bit counter and 5-bit time slot unit period counter will automatically be cleared when this bit is cleared to zero. However, the 8-bit programmable time slot counter will not be cleared. When this bit is changed from low to high, the touch key module 16-bit C/F counter, touch key function 16-bit counter, 5-bit time slot unit period counter and 8-bit time slot counter will be switched on together with the key and reference oscillators to drive the corresponding counters.

Bit 4   **TKCFOV:** Touch key module 16-bit C/F counter overflow flag

     0: No overflow occurs

     1: Overflow occurs

This bit is set high by the touch key module 16-bit C/F counter overflow and must be cleared to 0 by application programs.

Bit 3   **TK16OV:** Touch key function 16-bit counter overflow flag

     0: No overflow occurs

     1: Overflow occurs

This bit is set high by the touch key function 16-bit counter overflow and must be cleared to 0 by application programs.

Bit 2   **TSCS:** Touch key time slot counter select

     0: Each module use own time slot counter

     1: All touch key module use module 0 time slot counter

Bit 1~0   **TK16S1~TK16S0:** Touch key function 16-bit counter clock source select
00: f_{SYS}
01: $f_{SYS}/2$
10: $f_{SYS}/4$
11: $f_{SYS}/8$

- **TKC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	TKFS1	TKFS0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	1	1

Bit 7~2   Unimplemented, read as "0"

Bit 1~0   **TKFS1~TKFS0:** Touch Key oscillator and Reference oscillator frequency select
00: 1MHz
01: 3MHz
10: 7MHz
11: 11MHz

- **TK16DH/TK16DL – Touch Key Function 16-bit Counter Register Pair**

Register	TK16DH								TK16DL							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

This register pair is used to store the touch key function 16-bit counter value. This 16-bit counter can be used to calibrate the reference or key oscillator frequency. When the touch key time slot counter overflows, this 16-bit counter will be stopped and the counter content will be unchanged. This register pair will be cleared to zero when the TKST bit is set low.

- **TKMn16DH/TKMn16DL – Touch Key Module n 16-bit C/F Counter Register Pair**

Register	TKMn16DH								TKMn16DL							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

This register pair is used to store the touch key module n 16-bit C/F counter value. This 16-bit C/F counter will be stopped and the counter content will be kept unchanged when the touch key time slot counter overflows. This register pair will be cleared to zero when the TKST bit is set low.

- **TKMnROH/TKMnROL – Touch Key Module n Reference Oscillator Capacitor Select Register Pair**

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

This register pair is used to store the touch key module n reference oscillator capacitor value.

The reference oscillator internal capacitor value=(TKMnRO[9:0]×50pF)/1024

• **TKMnC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	MnMXS1	MnMXS0	MnDFEN	—	MnSOFC	MnSOF2	MnSOF1	MnSOF0
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 7~6   **MnMXS1~MnMXS0:** Multiplexer Key Select

Bit	Touch Key Module Number
MnMXS[1:0]	M0
00	KEY1
01	KEY2
10	KEY3
11	Reserved

Bit 5   **MnDFEN:** Touch key module n multi-frequency control
  0: Disable
  1: Enable

This bit is used to control the touch key oscillator frequency doubling function. When this bit is set to 1, the key oscillator frequency will be doubled.

Bit 4   Unimplemented, read as "0"

Bit 3   **MnSOFC:** Touch key module n C-to-F oscillator frequency hopping function control select
  0: Controlled by the MnSOF2~MnSOF0
  1: Controlled by hardware circuit

This bit is used to select the touch key oscillator frequency hopping function control method. When this bit is set to 1, the key oscillator frequency hopping function is controlled by the hardware circuit regardless of the MnSOF2~MnSOF0 bits value.

Bit 2~0   **MnSOF2~MnSOF0:** Touch key module n Reference and Key oscillators hopping frequency select (MnSOFC=0)
  000: 1.020MHz
  001: 1.040MHz
  010: 1.059MHz
  011: 1.074MHz
  100: 1.085MHz
  101: 1.099MHz
  110: 1.111MHz
  111: 1.125MHz

These bits are used to select the touch key oscillator frequency for the hopping function. Note that these bits are only available when the MnSOFC bit is cleared to 0.

The frequency mentioned here will be changed when the external or internal capacitor is with different values. If the touch key operates at 1MHz frequency, users can adjust the frequency in scale when any other frequency is selected.

• **TKMnC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	MnTSS	—	MnROEN	MnKOEN	D3	MnK3EN	MnK2EN	MnK1EN
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   **MnTSS:** Touch key module n time slot counter clock source select
  0: Touch key module n reference oscillator
  1: $f_{SYS}/4$

Bit 6   Unimplemented, read as "0"

Bit 5 **MnROEN:** Touch key module n Reference oscillator enable control
      0: Disable
      1: Enable

Bit 4 **MnKOEN:** Touch key module n Key oscillator enable control
      0: Disable
      1: Enable

Bit 3 **D3:** Reserved bit, should remain unchanged after power-on reset

Bit 2 **MnK3EN:** Touch key module n Key 3 enable control

MnK3EN	Touch Key Module n – Mn
	M0
0: Disable	I/O or other functions
1: Enable	KEY3

Bit 1 **MnK2IO:** Touch key module n Key 2 enable control

MnK2IO	Touch Key Module n – Mn
	M0
0: Disable	I/O or other functions
1: Enable	KEY2

Bit 0 **MnK1EN:** Touch key module n Key 1 enable control

MnK1EN	Touch Key Module n – Mn
	M0
0: Disable	I/O or other functions
1: Enable	KEY1

Touch Key Operation

When a finger touches or is in proximity to a touch pad, the capacitance of the pad will increase. By using this capacitance variation to change slightly the frequency of the internal sense oscillator, touch actions can be sensed by measuring these frequency changes. Using an internal programmable divider the reference clock is used to generate a fixed time period. By counting a number of generated clock cycles from the sense oscillator during this fixed time period touch key actions can be determined.
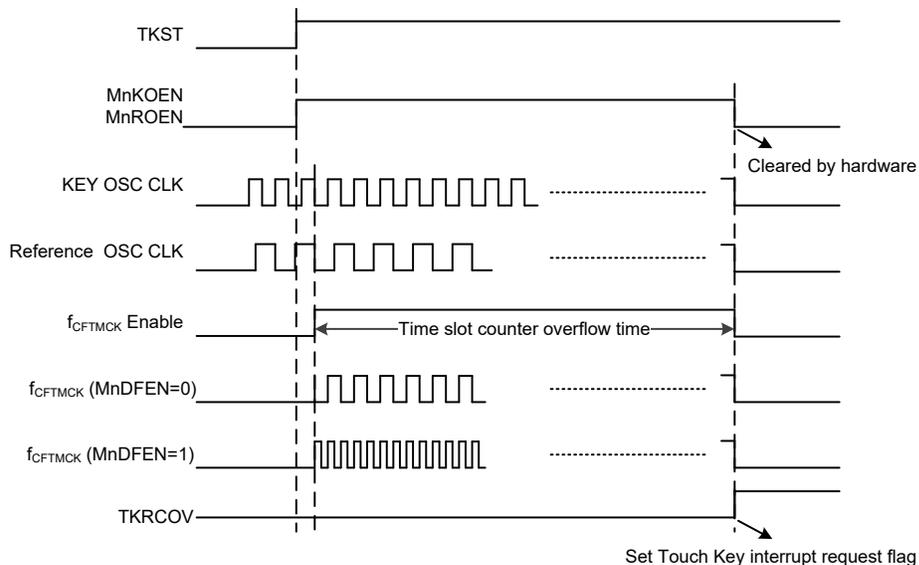
Touch Key Scan Mode Timing Diagram

The touch key module contains three touch key inputs which are shared with logical I/O pins, and the desired function is selected using register bits. Each touch key has its own independent sense oscillator. Therefore, there are three sense oscillators within the touch key module.

During this reference clock fixed interval, the number of clock cycles generated by the sense oscillator is measured, and it is this value that is used to determine if a touch action has been made or not. At the end of the fixed reference clock time interval a Touch Key interrupt signal will be generated.

Using the TSCS bit in the TKC0 register can select the module 0 time slot counter as the time slot counter for all modules. All modules use the same started signal, TKST, in the TKC0 register. The touch key module 16-bit C/F counter, touch key function 16-bit counter, 5-bit time slot unit period counter in all modules will be automatically cleared when the TKST bit is cleared to zero, but the 8-bit programmable time slot counter will not be cleared. The overflow time is setup by user. When the TKST bit changes from low to high, the 16-bit C/F counter, touch key function 16-bit counter, 5-bit time slot unit period counter and 8-bit time slot timer counter will be automatically switched on.

The key oscillator and reference oscillator in all modules will be automatically stopped and the 16-bit C/F counter, touch key function 16-bit counter, 5-bit time slot unit period counter and 8-bit time slot timer counter will be automatically switched off when the time slot counter overflows. The clock source for the time slot counter is sourced from the reference oscillator or $f_{SYS}/4$ which is selected using the MnTSS bit in the TKMnC1 register. The reference oscillator and key oscillator will be enabled by setting the MnROEN bit and MnKOEN bits in the TKMnC1 register.

When the time slot counter in all the touch key modules or in the touch key module 0 overflows, an actual touch key interrupt will take place. The touch keys mentioned here are the keys which are enabled.

Touch Key Interrupt

The touch key only has single interrupt, when the time slot counter in all the touch key modules or in the touch key module 0 overflows, an actual touch key interrupt will take place. The touch keys mentioned here are the keys which are enabled. The 16-bit C/F counter, 16-bit counter, 5-bit time slot unit period counter and 8-bit time slot counter in all modules will be automatically cleared. More details regarding the touch key interrupt is located in the interrupt section of the datasheet.

Programming Considerations

After the relevant registers are setup, the touch key detection process is initiated by changing the TKST bit from low to high. This will enable and synchronise all relevant oscillators. The TKRCOV flag which is the time slot counter flag will go high when the counter overflows. When this happens an interrupt signal will be generated. As the TKRCOV flag will not be automatically cleared, it has to be cleared by the application program.

The TKCFOV flag which is the 16-bit C/F counter overflow flag will go high when any of the Touch Key Module 16-bit C/F counter overflows. As this flag will not be automatically cleared, it has to be cleared by the application program.The TK16OV flag which is the 16-bit counter overflow flag will go high when the 16-bit counter overflows. As this flag will not be automatically cleared, it has to be cleared by the application program.

When the external touch key size and layout are defined, their related capacitances will then determine the sensor oscillator frequency.

# 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, such as the internal reference voltage, into a 12-bit digital value. The external or internal analog signal to be converted is determined by the SAINS and SACS bit fields. Note that when the internal analog signal is selected to be converted using the SAINS field, the external channel analog input will automatically be switched off. More detailed information about the A/D input signal selection will be described in the "A/D Converter Input Signals" section.

External Input Channels	Internal Signal	A/D Channel Select
3: AN0~AN2	V_{DD}, $V_{DD}/2$, $V_{DD}/4$, V_{VR}, $V_{VR}/2$, $V_{VR}/4$, AV_{SS}	SAINS3~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
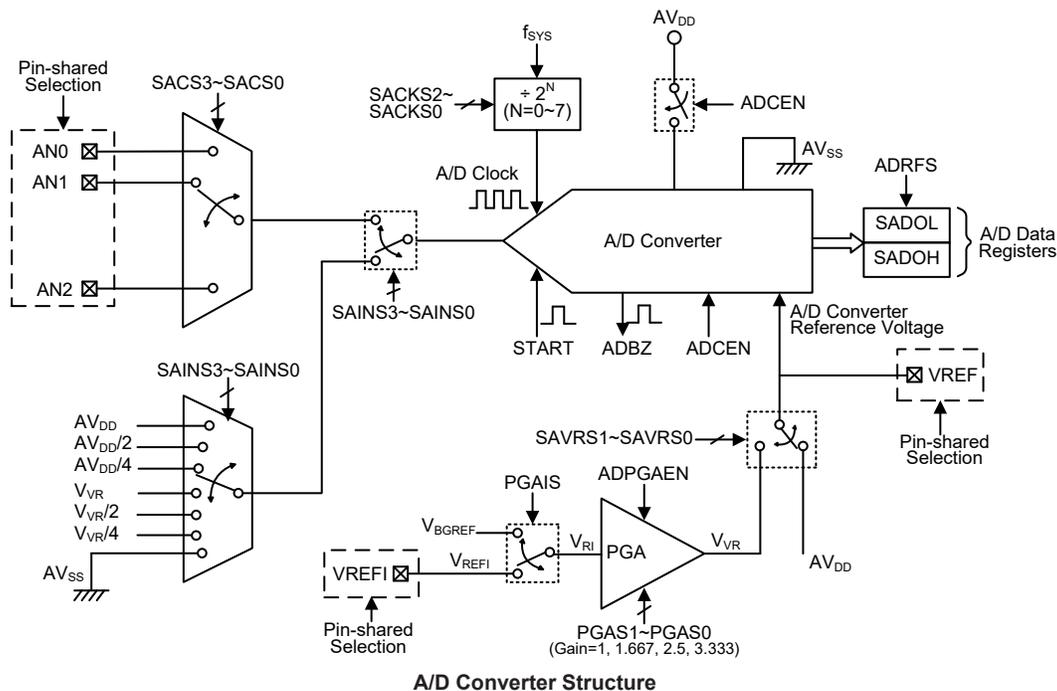
A/D Converter Register Description

Overall operation of the A/D converter is controlled using six registers. A read only register pair exists to store the A/D Converter data 12-bit value. Three registers, SADC0, SADC1 and SADC2, are the control registers which setup the operating conditions and control function of the A/D converter. The VBGRC register contains the VBGREN bit to control the bandgap reference voltage.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SADOL (ADRFS=0)	D3	D2	D1	D0	—	—	—	—
SADOL (ADRFS=1)	D7	D6	D5	D4	D3	D2	D1	D0
SADOH (ADRFS=0)	D11	D10	D9	D8	D7	D6	D5	D4
SADOH (ADRFS=1)	—	—	—	—	D11	D10	D9	D8
SADC0	START	ADBZ	ADCEN	ADRFS	SACS3	SACS2	SACS1	SACS0
SADC1	SAINS3	SAINS2	SAINS1	SAINS0	—	SACKS2	SACKS1	SACKS0
SADC2	ADPGAEN	—	—	PGAIS	SAVRS1	SAVRS0	PGAGS1	PGAGS0
VBGRC	—	—	—	—	—	—	—	VBGREN

"—": Unimplemented, read as "0"

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. The A/D data registers 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	—	—	—	—
1	—	—	—	—	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

A/D Converter Data Registers

A/D Converter Control Registers – SADC0, SADC1, SADC2

To control the function and operation of the A/D converter, three control registers known as SADC0, SADC1 and SADC2 are provided. These 8-bit registers define functions such as the selection of which analog signal 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 and internal analog signals must be routed to the converter. The SAINS3~SAINS0 bits in the SADC1 register and the SACS3~SACS0 bits in the SADC0 register are used to determine which analog signal derived from the external or internal signals will be connected to the A/D converter. The A/D converter also contains a programmable gain amplifier, PGA, to generate the A/D converter internal reference voltage. The overall operation of the PGA is controlled using the SADC2 register.

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. 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 A/D conversion

　　　　　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 high to indicate that the A/D conversion is initiated. The ADBZ flag will be cleared to zero 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 one to enable the A/D converter. If the bit is cleared to zero, 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, SADOH and SADOL, will be unchanged.

Bit 4　　　**ADRFS:** A/D Converter data format control

　　　　　　0: ADC output data format → SADOH=D[11:4]; SADOL=D[3:0]

　　　　　　1: ADC output 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 converter data register section.

Bit 3~0　　**SACS3~SACS0:** A/D converter external analog channel input selection

　　　　　　0000: AN0

　　　　　　0001: AN1

　　　　　　0010: AN2

　　　　　　0011~0111: Reserved

　　　　　　1000~1111: Undefined, input floating

• **SADC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	SAINS3	SAINS2	SAINS1	SAINS0	—	SACKS2	SACKS1	SACKS0
R/W	R/W	R	R/W	R/W	—	R/W	R/W	R/W
POR	0	0	0	0	—	0	0	0

Bit 7~4　　**SAINS3~SAINS0:** A/D converter input signal selection

　　　　　　0000: External source – External analog channel intput, ANn

　　　　　　0001: Internal source – Internal signal derived from AV_{DD}

　　　　　　0010: Internal source – Internal signal derived from $AV_{DD}/2$

　　　　　　0011: Internal source – Internal signal derived from $AV_{DD}/4$

　　　　　　0100: External source – External analog channel intput, ANn

　　　　　　0101: Internal source – Internal A/D converter PGA output voltage V_{VR}

　　　　　　0110: Internal source – Internal A/D converter PGA output voltage $V_{VR}/2$

　　　　　　0111: Internal source – Internal A/D converter PGA output voltage $V_{VR}/4$

　　　　　　10xx: Internal source – Connected to ground, V_{SS}

　　　　　　11xx: External source – External analog channel intput, ANn

　　　　　When the internal analog signal is selected to be converted, the external channel signal input will automatically be switched off regardless of the SACS field value. It will

prevent the external channel input from being connected together with the internal analog signal.

Bit 3   Unimplemented, read as "0"

Bit 2~0   **SACKS2~SACKS0:** A/D conversion clock source selection

         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: $f_{SYS}/128$

• **SADC2 Register**

Bit	7	6	5	4	3	2	1	0
Name	ADPGAEN	—	—	PGAIS	SAVRS1	SAVRS0	PGAGS1	PGAGS0
R/W	R/W	—	—	R/W	R/W	R/W	R/W	R/W
POR	0	—	—	0	0	0	0	0

Bit 7   **ADPGAEN:** A/D converter PGA enable/disable control

         0: Disable
         1: Enable

     This bit is used to control the A/D converter internal PGA function. When the PGA output voltage is selected as A/D input or A/D reference voltage, the PGA needs to be enabled by setting this bit high. Otherwise the PGA needs to be disabled by clearing the ADPGAEN bit to zero to conserve power.

Bit 6~5   Unimplemented, read as "0"

Bit 4   **PGAIS:** PGA input voltage selection

         0: From VREFI pin
         1: From internal reference voltage V_{BGREF}

     When the internal independent reference voltage V_{BGREF} is selected as the PGA input, the external reference voltage on the VREFI pin will be automatically switched off. In addition, the internal bandgap reference V_{BGREF} should be enabled by setting the V_{BGREN} bit in the VBGRC register to "1".

Bit 3~2   **SAVRS1~SAVRS0:** A/D converter reference voltage selection

         00: Positive power supply, AV_{DD}
         01: External VREF pin
         1x: Internal PGA output voltage, V_R

     These bits are used to select the A/D converter reference voltage source. When the internal reference voltage source is selected, the reference voltage derived from the external VREF pin will automatically be switched off.

Bit 1~0   **PGAGS1~PGAGS0:** PGA gain selection

         00: Gain=1
         01: Gain=1.667 – V_{VR}=2V as V_{RI}=1.2V
         10: Gain=2.5 – V_{VR}=3V as V_{RI}=1.2V
         11: Gain=3.333 – V_{VR}=4V as V_{RI}=1.2V

     These bits are used to select the PGA gain. Note that here the gain is guaranteed only when the PGA input voltage is equal to 1.2V.

Bandgap Referenc Voltage Control Register – VBGRC

A high performance bandgap voltage reference is included in the device. It has an accurate voltage reference output, V_{BGREF}, when input supply voltage changes or temperature variates. The VBGRC register is used to control the bandgap reference voltage circuit enable or disable.

• **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 reference voltage control
        0: Disable
        1: Enable
        This bit is used to enable the internal Bandgap reference circuit. The internal Bandgap reference circuit should first be enabled before the V_{BGREF} voltage is selected to be used. A specific start-up time is necessary for the Bandgap circuit to become stable and accurate. When this bit is cleared to 0, the Bandgap voltage output V_{BGREF} is in a low state.

A/D Converter Reference Voltage

The actual reference voltage supply to the A/D Converter can be supplied from the internal A/D converter power, AV_{DD}, an external reference source supplied on pin VREF or an internal reference source derived from the PGA output V_{VR}. The desired selection is made using the SAVRS1~SAVRS0 bits in the SADC2 register. The internal reference voltage V_{VR} is an amplified output signal through a programmable gain amplifier, PGA, which is controlled by the ADPGAEN bit in the SADC2 register. The PGA gain can be equal to 1, 1.667, 2.5 or 3.333 and selected using the PGAGS1~PGAGS0 bits in the SADC2 register. The PGA input can come from the external reference input pin, VREFI, or an internal Bandgap reference voltage, V_{BGREF}, selected by the PGAIS bit in the SADC2 register. The internal Bandgap reference circuit should first be enabled before the V_{BGREF} is selected to be used. A specific start-up time is necessary for the Bandgap circuit to become stable and accurate.

As the VREFI and VREF pins both are pin-shared with other functions, when the VREFI or VREF pin is selected as the reference voltage pin, the VREFI or VREF pin-shared function selection bits should first be properly configured to disable other pin-shared functions. However, if the internal reference signal is selected as the reference source, the external reference input from the VREFI or VREF pin will automatically be switched off by hardware.

Note that the analog input values must not be allowed to exceed the value of the selected reference voltage.

SAVRS[1:0]	Reference Source	Description
00	AV_{DD}	Internal A/D converter power supply
01	VREF Pin	External A/D converter reference pin
10 or 11	V_{VR}	Internal A/D converter PGA output voltage

A/D Converter Reference Voltage Selection

A/D Converter Input Signals

All of the external A/D analog input pins are pin-shared with the I/O pins as well as other functions. The corresponding pin-shared function selection bits in the PxS1 and PxS0 registers, determine whether the external input pins are setup as A/D converter analog channel inputs or whether they have other functions. If the corresponding pin is setup to be an A/D converter analog channel input, the original pin function 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 relevant A/D input function selection bits enable an A/D input, the status of the port control register will be overridden.

As the device contains only one actual analog to digital converter hardware circuit, each of the external and internal analog signals must be routed to the converter. The SAINS3~SAINS0 bits in the SADC1 register are used to determine that the analog signal to be converted comes from the external channel input or internal analog signal. The SACS3~SACS0 bits in the SADC0 register are used to determine which external channel input is selected to be converted. If the SAINS3~SAINS0 bits are set to "0000", "0100", or "1100~1111", the external channel input will be selected to be converted and the SACS3~SACS0 bits can determine which external channel is selected.

When the SAINS3~SAINS0 bits is set to the value of "0001~0011", "0101~0111", the internal analog signal will be selected. If the internal analog signal is selected to be converted, the external channel signal input will automatically be switched off regardless of the SACS3~SACS0 bit values. It will prevent the external channel input from being connected together with the internal analog signal.

SAINS[3:0]	SACS[3:0]	Input Signals	Description
0000, 0100, 1100~1111	0000~0010	AN0~AN2	External channel analog input ANn
	1000~1111	—	Floating
0001	xxxx	AV_{DD}	Internal A/D converter power supply voltage AV_{DD}
0010	xxxx	$AV_{DD}/2$	Internal A/D converter power supply voltage $AV_{DD}/2$
0011	xxxx	$AV_{DD}/4$	Internal A/D converter power supply voltage $AV_{DD}/4$
0101	xxxx	V_{VR}	Internal A/D converter PGA output V_{VR}
0110	xxxx	$V_{VR}/2$	Internal A/D converter PGA output $V_{VR}/2$
0111	xxxx	$V_{VR}/4$	Internal A/D converter PGA output $V_{VR}/4$
10xx	xxxx	AVss	Connected to ground

A/D Converter Input Signal Selection

A/D Conversion 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 bit 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 associated 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 A/D clock source speed that can be selected. As the recommended range of permissible A/D clock period, t_{ADCK}, for different situations is specified in the "A/D Converter Electrical Characteristics", care must be taken for system clock frequencies. For example, when the t_{ADCK} is from 0.5μs to 10μs at 2.0V≤V_{DD}≤5.5V, if 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 than the minimum A/D clock period or greater than the maximum A/D clock period, which may result in inaccurate A/D conversion values. Refer to the following table for examples, special care must be taken to values marked with an asterisk *, as these values may be beyond 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 *
12MHz	83ns *	167ns *	333ns *	667ns *	1.33μs	2.67μs	5.33μs	10.67μs *
16MHz	62.5ns *	125ns *	250ns *	500ns	1μs	2μs	4μs	8μs

A/D Clock Period Examples @ 2.0V≤V_{DD}≤5.5V

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.

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 periods and the data conversion takes 12 A/D clock periods. Therefore a total of 16 A/D clock periods for an analog signal A/D conversion which is defined as t_{ADC} are necessary.

Maximum single A/D conversion rate=1/(A/D clock period×16)

The accompanying diagram shows graphically the various stages involved in an external channel input signal 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} where t_{ADCK} is equal to the A/D clock period.

A/D Conversion Timing – External Channel Input
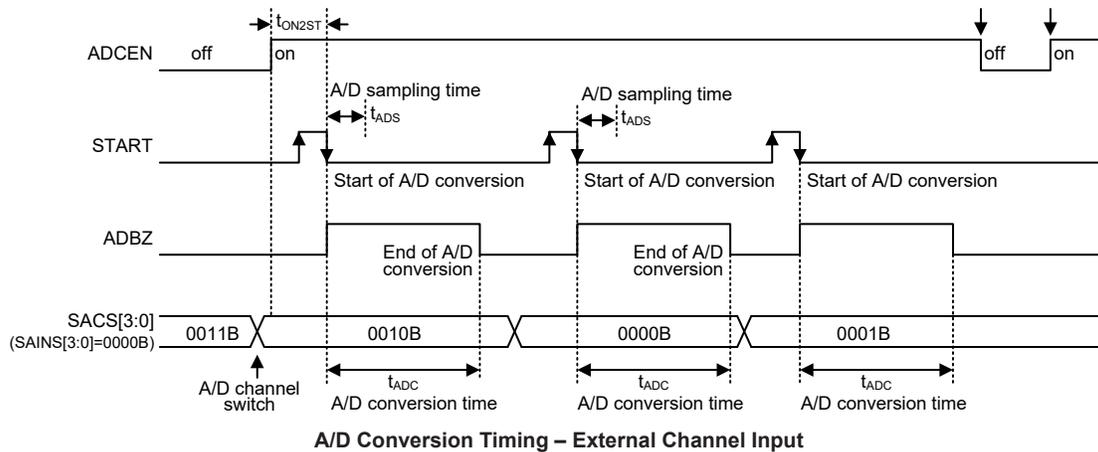
Summary of A/D Conversion Steps

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 properly programming the SACKS2~SACKS0 bits in the SADC1 register.

• Step 2

Enable the A/D converter 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 SACS3~SACS0 and SAINS3~SAINS0 bits

Selecting the external channel input to be converted, go to Step 4.

Selecting 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 SAINS3~SAINS0 bits, the corresponding pin 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. After this step, go to Step 6.

• Step 5

If the value of SAINS3 to SAINS0 is "0001~0011"or"0101~1011", the internal analog signal is selected, the external channel analog input is automatically disconnected. After this step, go to Step 6.

• Step 6

Select the A/D converter output data format by configuring the ADRFS bit.

• Step 7

Select the A/D converter reference voltage source by configuring the SAVRS1~SAVRS0 bits in the SADC1 register.

Select the PGA input signal and the desired PGA gain if the PGA output voltage, VR, is selected as the A/D converter reference voltage.

- 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 initialised 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 Transfer 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 reference voltage value divided by 4096.

$$1 \text{ LSB} = V_{REF}/4096$$

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

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

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 SAVRS1~SAVRS0 bits.

Ideal A/D Transfer Function
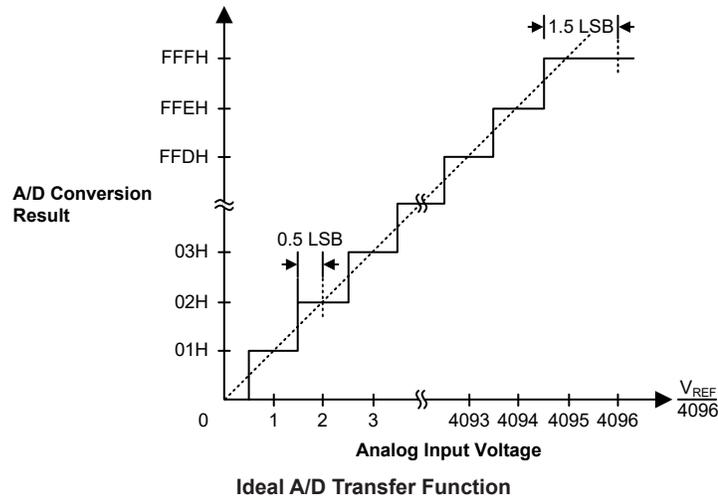
A/D Programming Examples

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.

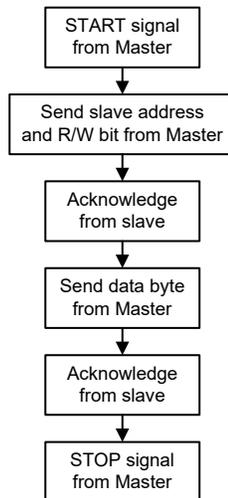**Example: using an ADBZ polling method to detect the end of conversion fot the BS45R2141**

```
clr ADE              ; disable ADC interrupt
mov a,03H            ; select f_SYS/8 as A/D clock and A/D input
mov SADC1,a          ; signal comes from external channel
mov a,00H            ; select AV_DD as the A/D reference voltage source
mov SADC2,a
mov a,01H            ; setup PDS0 to configure pin AN0
mov PDS0,a
mov a,20H            ; enable A/D converter and select AN0 as
mov SADC0,a          ; the A/D external channel input
:
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 fot the BS45R2141

```
    clr ADE             ; disable ADC interrupt
    mov a,03H           ; select f_SYS/8 as A/D clock and A/D input
    mov SADC1,a         ; signal comes from external channel
    mov a,00H           ; select AV_DD as the A/D reference voltage source
    mov SADC2,a
    mov a,01H           ; setup PDS0 to configure pin AN0
    mov PDS0,a
    mov a,20H           ; enable A/D converter and select AN0 as
    mov SADC0,a         ; the A/D external channel input
    :
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_ISR:                ; ADC interrupt service routine
    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
```

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 function 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	$f_{SYS}>2MHz$	$f_{SYS}>4MHz$
2 system clock debounce	$f_{SYS}>4MHz$	$f_{SYS}>8MHz$
4 system clock debounce	$f_{SYS}>4MHz$	$f_{SYS}>8MHz$

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.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	—	SIMDEB1	SIMDEB0	SIMEN	D0
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. 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:** I²C data register bit 7 ~ bit 0

I²C Address Register

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 select the I²C slave mode and debounce time. 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	—	SIMDEB1	SIMDEB0	SIMEN	D0
R/W	R/W	R/W	R/W	—	R/W	R/W	R/W	R/W
POR	1	1	1	—	0	0	0	0

Bit 7~5   **SIM2~SIM0:** I²C operating mode control
                  000~101, 111: Unused mode
                  110: I²C slave mode
              These bits setup the I²C operating mode.
Bit 4   Unimplemented, read as "0"
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 I²C interface function is selected by setting the SIM2~SIM0 bits to "110".
Bit 1   **SIMEN:** I²C enable control
                  0: Disable
                  1: Enable

The bit is the overall on/off control for the I²C interface. When the SIMEN bit is cleared to zero to disable the I²C interface, the SDA and SCL lines will lose their I²C function and the I²C operating current will be reduced to a minimum value. When the bit is high the I²C interface is enabled. If the I²C interface function is selected via the 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   **D0:** Reserved, cannot be used

- **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. Below is an example of the flow of a two-byte I²C data transfer.

First, the I²C slave device receives a start signal from I²C master and then HCF bit is automatically cleared to zero. Second, the I²C slave device finishes receiving the 1st data byte and then the HCF bit is automatically set high. Third, users read the 1st data byte from SIMD register by the application program and then the HCF bit is automatically cleared to zero. Fourth, the I²C slave device finishes receiving the 2nd data byte and then HCF bit is automatically set high and so on. Finally, the I²C slave device receives a stop signal from the I²C master and then the HCF bit is automatically set high.

Bit 6   **HAAS:** I²C bus address match flag
                    0: Not address match
                    1: Address match

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 I²C 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 SIM2~SIM0 and SIMEN bits in the SIMC0 register to "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 I²C interrupt enable bit of the interrupt control register to enable the I²C interrupt.
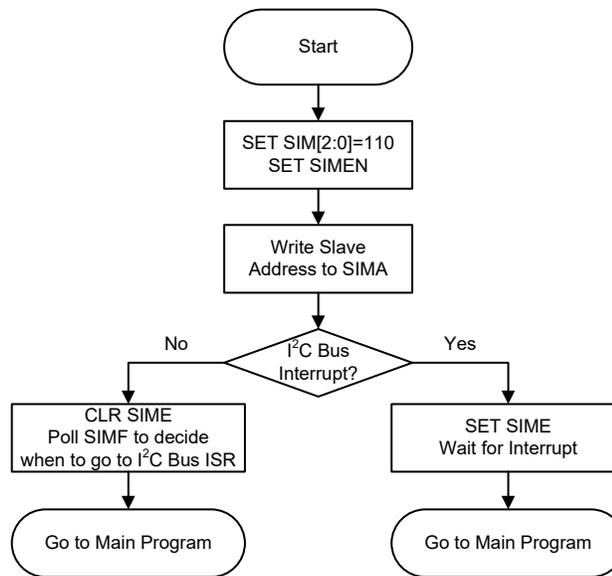
I²C Bus Initialisation Flow Chart

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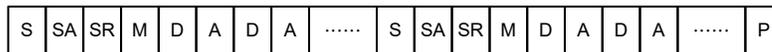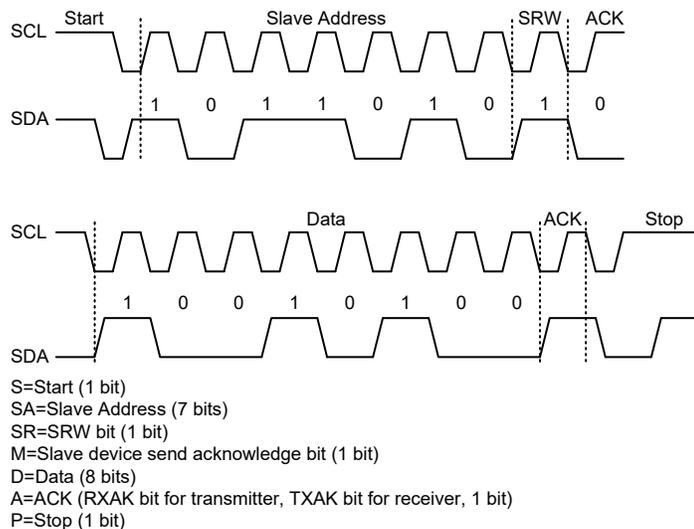
Note: When a slave address is matched, the device must be placed in either the transmit mode and then write data to the SIMD register, or in the receive mode where it must implement a dummy read from the SIMD register to release the SCL line.

I²C Communication Timing Diagram
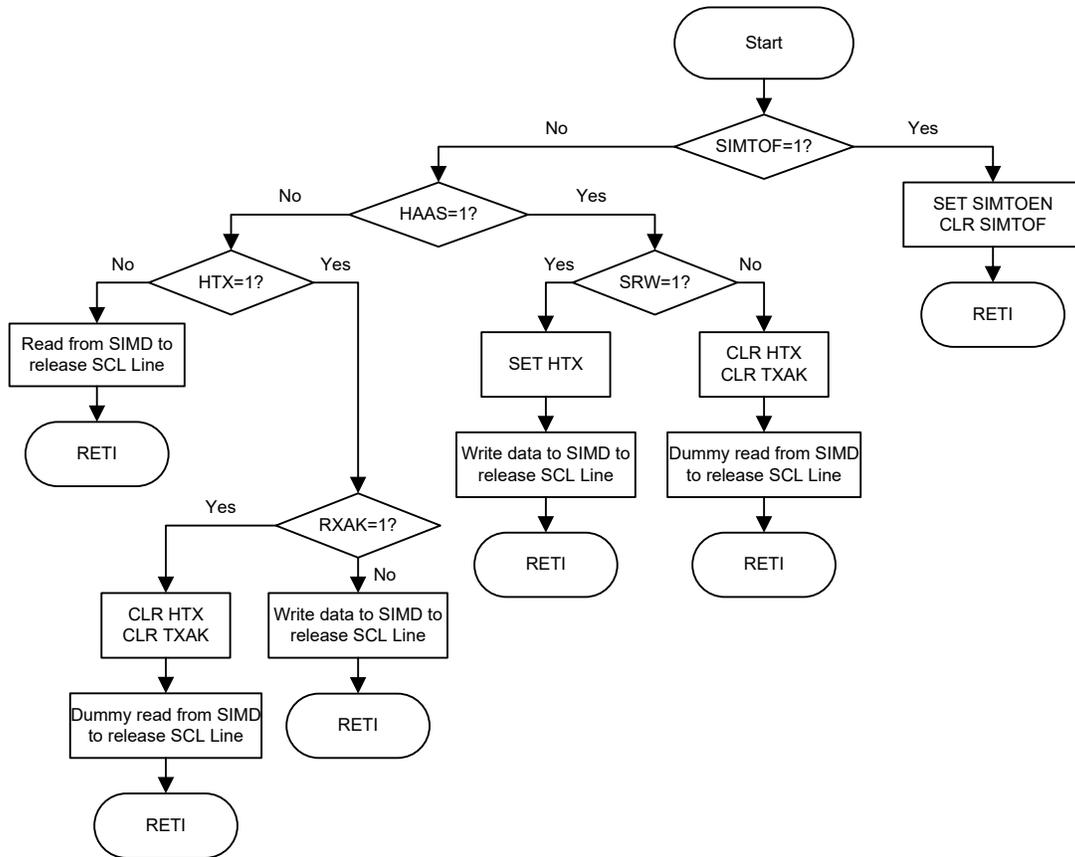
I²C Bus ISR Flow Chart

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.
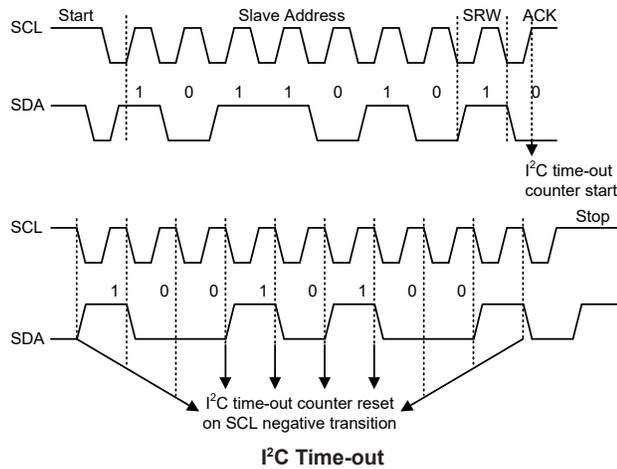
I²C Time-out

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 I²C 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: ((1~64)×32)/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:** I²C time-out control
         0: Disable
         1: Enable

Bit 6   **SIMTOF:** I²C time-out flag
         0: No time-out occurred
         1: Time-out occurred

Bit 5~0   **SIMTOS5~SIMTOS0:** 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)×(32/f$_{SUB}$).

Cyclic Redundancy Check – CRC

The Cyclic Redundancy Check, CRC, calculation unit is an error detection technique test algorithm and uses to verify data transmission or storage data correctness. A CRC calculation takes a data stream or a block of data as input and generates a 16-bit output remainder. Ordinarily, a data stream is suffixed by a CRC code and used as a checksum when being sent or stored. Therefore, the received or restored data stream is calculated by the same generator polynomial as described in the following section.

CRC Block Diagram
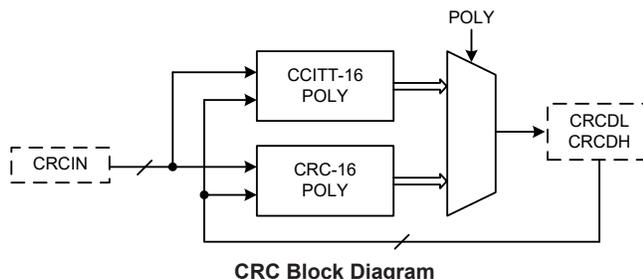
CRC Registers

The CRC generator contains an 8-bit CRC data input register, CRCIN, and a CRC checksum register pair, CRCDH and CRCDL. The CRCIN register is used to input new data and the CRCDH and CRCDL registers are used to hold the previous CRC calculation result. A CRC control register, CRCCR, is used to select which CRC generating polynomial is used.

Register Name	Bit							
	7	6	5	4	3	2	1	0
CRCCR	—	—	—	—	—	—	—	POLY
CRCIN	D7	D6	D5	D4	D3	D2	D1	D0
CRCDL	D7	D6	D5	D4	D3	D2	D1	D0
CRCDH	D15	D14	D13	D12	D11	D10	D9	D8

"—": Unimplemented, read as "0"

CRC Register List

• CRCCR Register

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

Bit 7~1   Unimplemented, read as "0"

Bit 0   **POLY:** 16-bit CRC generating polynomial selection
       0: CRC-CCITT: $X^{16}+X^{12}+X^5+1$
       1: CRC-16: $X^{16}+X^{15}+X^2+1$

• CRCIN 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:** CRC input data register

• **CRCDL 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:** 16-bit CRC checksum low byte data register

• **CRCDH 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:** 16-bit CRC checksum high byte data register

CRC Operation

The CRC generator provides the 16-bit CRC result calculation based on the CRC16 and CCITT CRC16 polynomials. In this CRC generator, there are only these two polynomials available for the numeric values calculation. It cannot support the 16-bit CRC calculations based on any other polynomials.

The following two expressions can be used for the CRC generating polynomial which is determined using the POLY bit in the CRC control register, CRCCR. The CRC calculation result is called as the CRC checksum, CRCSUM, and stored in the CRC checksum register pair, CRCDH and CRCDL.

- CRC-CCITT: $X^{16}+X^{12}+X^5+1$.
- CRC-16: $X^{16}+X^{15}+X^2+1$.

CRC Computation

Each write operation to the CRCIN register creates a combination of the previous CRC value stored in the CRCDH and CRCDL registers and the new data input. The CRC unit calculates the CRC data register value is based on byte by byte. It will take one MCU instruction cycle to calculate the CRC checksum.

CRC Calculation Procedures:

1. Clear the checksum register pair, CRCDH and CRCDL.

2. Execute an "Exclusive OR" operation with the 8-bit input data byte and the 16-bit CRCSUM high byte. The result is called the temporary CRCSUM.

3. Shift the temporary CRCSUM value left by one bit and move a "0" into the LSB.

4. Check the shifted temporary CRCSUM value after procedure 3.

   If the MSB is 0, then this shifted temporary CRCSUM will be considered as a new temporary CRCSUM.

   Otherwise, execute an "Exclusive OR" operation with the shifted temporary CRCSUM in procedure 3 and a data "8005H". Then the operation result will be regarded as the new temporary CRCSUM.

   Note that the data to be perform an "Exclusive OR" operation is "8005H" for the CRC-16 polynomial while for the CRC-CCITT polynomial the data is "1021H".

5. Repeat the procedure 3 ~ procedure 4 until all bits of the input data byte are completely calculated.

6. Repeat the procedure 2 ~ procedure 5 until all of the input data bytes are completely calculated. Then, the latest calculated result is the final CRC checksum, CRCSUM.

CRC Calculation Examples:

- Write 1 byte input data into the CRCIN register and the corresponding CRC checksum are individually calculated as the following table shown.

CRC Data Input CRC Polynomial	00H	01H	02H	03H	04H	05H	06H	07H
CRC-CCITT ($X^{16}+X^{12}+X^5+1$)	0000H	1021H	2042H	3063H	4084H	50A5H	60C6H	70E7H
CRC-16 ($X^{16}+X^{15}+X^2+1$)	0000H	8005H	800FH	000AH	801BH	001EH	0014H	8011H

Note: The initial value of the CRC checksum register pair, CRCDH and CRCDL, is zero before each CRC input data is written into the CRCIN register.

- Write 4 bytes input data into the CRCIN register sequentially and the CRC checksum are sequentially listed in the following table.

CRC Data Input CRC Polynomial	CRCIN=78H→56H→34H→12H
CRC-CCITT ($X^{16}+X^{12}+X^5+1$)	(CRCDH, CRCDL)=FF9FH→BBC3H→A367H→D0FAH
CRC-16 ($X^{16}+X^{15}+X^2+1$)	(CRCDH, CRCDL)=0110h→91F1h→F2DEh→5C43h

Note: The initial value of the CRC checksum register pair, CRCDH and CRCDL, is zero before the sequential CRC data input operation.

Program Memory CRC Checksum Calculation Example:

1. Clear the checksum register pair, CRCDH and CRCDL.

2. Select the CRC-CCITT or CRC-16 polynomial as the generating polynomial using the POLY bit in the CRCCR register.

3. Execute the table read instruction to read the program memory data value.

4. Write the table data low byte into the CRCIN register and execute the CRC calculation with the current CRCSUM value. Then a new CRCSUM result will be obtained and stored in the CRC checksum register pair, CRCDH and CRCDL.

5. Write the table data high byte into the CRCIN register and execute the CRC calculation with the current CRCSUM value. Then a new CRCSUM result will be obtained and stored in the CRC checksum register pair, CRCDH and CRCDL.

6. Repeat the procedure 3 ~ procedure 5 to read the next program memory data value and execute the CRC calculation until all program memory data are read followed by the sequential CRC calculation. Then the value in the CRC checksum register pair is the final CRC calculation result.

High Voltage Linear Charger

Input Voltage Detect

The device integrates an input voltage detect function. The maximum input rating is 20V to overcome unpredictable high voltages from the USB input. The input voltage goes through two switches, both of which will turn on when the V_{IN} is a normal voltage between 4.5V and 7.0V. When the input voltage exceeds 7.0V, the VIN MOSFET will quickly turn off to protect the back-end circuit. When the input voltage is lower than 4V or $V_{BAT}+0.1V$, the linear charger MOSFET will turn off to prevent VBAT current from flowing back to VIN to reduce V_{BAT} leakage.
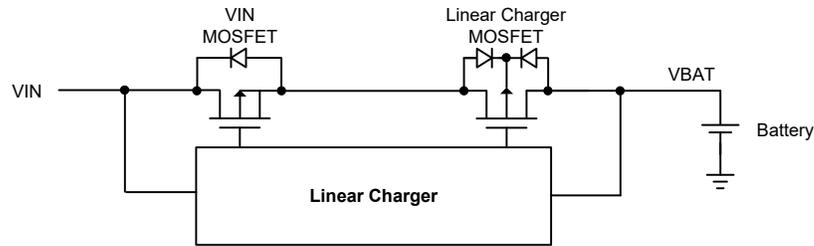
Input Voltage Detection Diagram
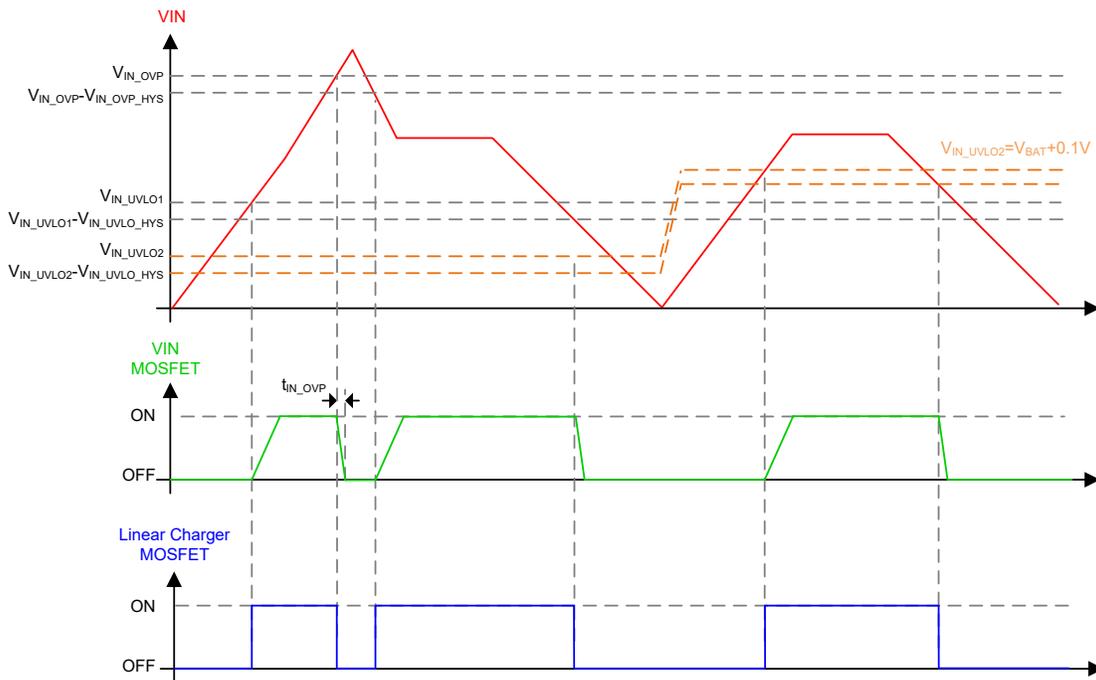
Input Voltage Detection Waveform

Linear Charger

The device integrates a linear charger, the maximum charging current of which can be set to 1A through the ISET pin.
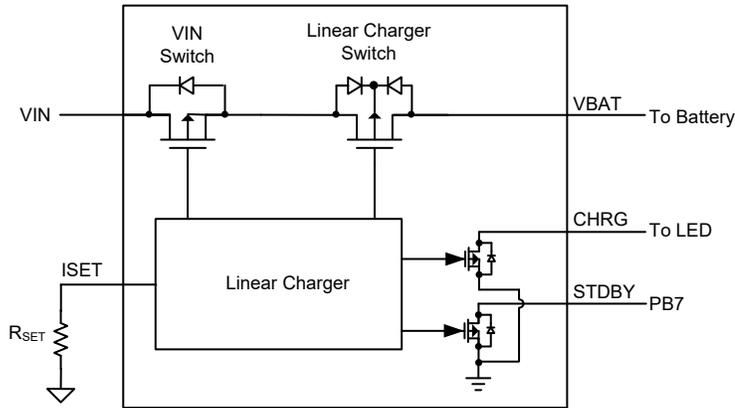
The external resistor, R_{SET}, is used to set the charging current:

$$I_{BAT\_CC} \text{ (Charging Current)} = 2000/R_{SET}$$

The battery charging cycle curve is shown in the figure below. If the V_{BAT} is lower than V_{BAT_LOW} - $V_{BAT\_LOW\_HYS}$, the linear charger is in the trickle chargemode and the charging current is $I_{BAT\_CC}/10$. Once the V_{BAT} is higher than V_{BAT_LOW}, the constant current charge (CC) mode is applied and the charging current is I_{BAT_CC}. When V_{BAT} has been charged to the battery charging target voltage of $V_{BAT\_CV1/2}$, the device enters the constant voltage charge (CV) mode. Once the charging current drops to $I_{BAT\_CC}/10$, the device is in a terminated state and stops charging. When V_{BAT} is depleted and falls below the charging voltage of V_{BAT_RECHG}, the device will recharge the battery. When the temperature is higher than the protection temperature during charging, the charging current will automatically reduce, and the device enters the constant temperature charge (CT) mode.

The linear charger has two open-drain status indication outputs: the CHRG pin and STDBY signal. When the charger is in a charging status, the CHRG pin is pulled low and the STDBY outputs a high-impedance state. When the charger is fully charged, the STDBY signal is pulled low and the CHRG outputs a high-impedance state. In other status, both the CHRG and the STDBY output with a high-impedance state. STDBY can only be powered by VIN.

Battery Charging Cycle
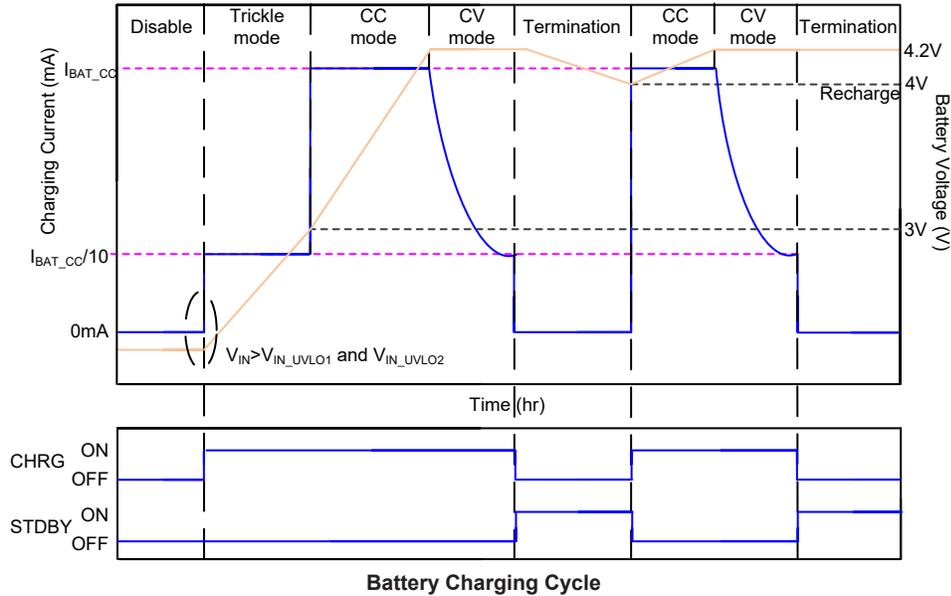
When the battery is not installed, the charger will charge the capacitance on the VBAT pin. When the external capacitance of the battery connection terminal BAT is 10μF, the CHRG/STDBY will flash once every 1~4 seconds.

The linear charger has a temperature control function that automatically reduces the current to prevent temperature increase when the junction temperature exceeds $T_{J(MAX)}$. This means that the current in the constant current charge mode may be lower than the set I_{BAT_CC}. This feature can protect the device from damage caused by larger charging power and operating environment.

Reference Voltage

The linear charger integrates a reference voltage function. The output of the reference voltage is shared with the FN signal. When the D_VREN is 1, the reference voltage function is enabled and the voltage is output by the FN signal. When the D_VREN is 0, the reference voltage function is disabled and the FN signal is used as other function. The linear charger status will affect the reference voltage function, refer to the "Application Information" section.

Multi-function FN Output Signal

The FN signal contains three functions, VIN status, linear charger status and reference voltage. It is necessary to configure the D_SELFN, D_VREN and D_SELLC via the communication interface to enable the corresponding function.

Two-Line Control			FN Signal Function
D_SELLC	**D_VREN**	**D_SELFN**	
x	0	0	VIN status (INOK1)
0	0	1	Floating
1	0	1	Linear charger status
0→1[(1)]	0	1	<t_{INOK2}: VIN status (INOK2)
			>t_{INOK2}+t_{LCS}: Linear charger status
x	1	x[(2)]	Reference voltage

"x": Don't care

Note: 1. D_SELLC=1, D_VREN=0 and D_SELFN=1 are configured in the same communication, whereas the D_SELLC was previously cleared to 0.

2. The FN signal function corresponding to the D_SELFN setting is still valid, the appropriate status must be set according to the current conditions to avoid causing an abnormal reference voltage, refer to the "Application Information" section.
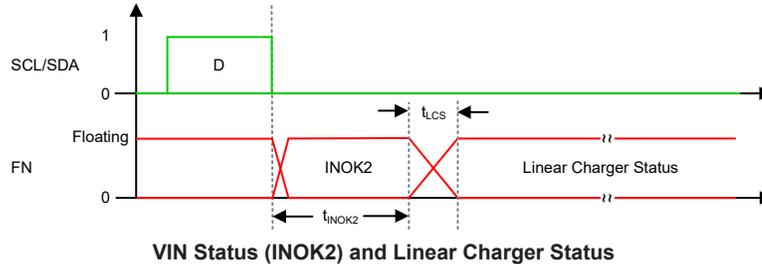
Linear Charger Status		
FN Signal Status	**VIN Input Voltage Status**	**Description**
Floating or pulled low	Invalid	The linear charger is disabled
Floating	Valid [(Note)]	The linear charger is disabled
		The linear charger is enabled and charging
Pulled low		The linear charger is enabled and has stopped

Note: Meet the following conditions: $V_{IN\_UVLO1}<V_{IN}$, $V_{IN\_UVLO2}<V_{IN}$ and $V_{IN}<V_{IN\_OVP}$.

VIN Input Voltage Status		
FN Signal Status		**Description**
INOK1	**INOK2**	
Floating	Floating	VIN input voltage is invalid
Floating	Pulled low	VIN input voltage is invalid
Pulled low	Floating	VIN input voltage is valid [(Note)]
Pulled low	Pulled low	VIN input voltage is invalid

Note: Meet the following conditions: $V_{IN\_UVLO1} < V_{IN}$, $V_{IN\_UVLO2} < V_{IN}$ and $V_{IN} < V_{IN\_OVP}$.

In the case of D_SELLC=0, after configuring D_SELLC=1, D_VREN=0 and D_SELFN=1 via the D value shown in the following figure, the FN signal outputs the VIN status (INOK2) and the linear charger status. The linear charger status must be read more than 8 times at intervals of 20μs or more, and then compare the number of different logic statuses, and the logic with more is the read result.

VIN Status (INOK2) and Linear Charger Status

Interface

The linear charger uses the SDA/SCL signal of the two-line communication interface for functional control and status reading with FN signal, which can enable/disable the linear charger and reference voltage function, or directly control the CHRG and STDBY output status, or use the FN signal to read whether the input voltage is valid and whether the battery charged is complete.

The communication code is composed of one start bit, seven data bits, one even parity bit and one stop bit. The code must meet the definition to be effectively written into the linear charger, an incorrect operation will be ignored directly. The communication code requirements are as follows:
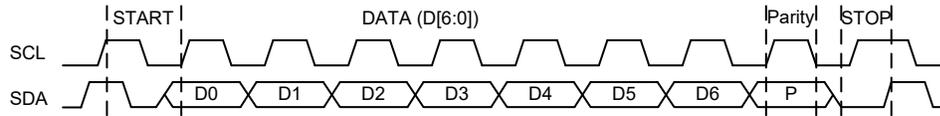
In order to ensure that the linear charger is operated at a correct voltage when powered on, the two-line communication interface is valid after the V_{BAT} is higher than V_{IF_EN}, then the linear charger can be operated via the two-line communication interface. When the V_{BAT} is lower than V_{IF_DIS}, the two-line communication interface will automatically return to the default value, and it cannot be used to operate the linear charger. The functions and state of the two-line communication interface are described as follows:

Data	Name	Function	Description	Default Value
D0	D_SETSTDBY	STDBY signal function setting	This setting is valid only when D1 is set to 1 　0: STDBY signal is floating 　1: STDBY signal is pulled low	0
D1	D_SELSTDBY	STDBY signal function selection	0: STDBY signal outputs the linear charging status 　STDBY signal is pulled low: Fully charged 　STDBY signal is floating: Not fully charged 1: STDBY signal outputs a status based on the D0 bit setting	0
D2	D_SETCHRG	CHRG pin function setting	This setting is valid only when D3 is set to 1 　0: CHRG pin is floating 　1: CHRG pin is pulled low	0
D3	D_SELCHRG	CHRG pin function selection	0: CHRG pin outputs the linear charging status 　CHRG pin is pulled low: Charging 　CHRG pin is floating: Not charging 1: CHRG pin outputs a status based on the D2 bit setting	0

Data	Name	Function	Description	Default Value
D4	D_SELFN	FN signal function selection	0: FN signal outputs VIN status (INOK1) 　FN signal is pulled low: VIN input voltage is valid 　FN signal is floating: VIN input voltage is invalid 1: FN signal outputs VIN status (INOK2) and linear charger status 　$<t_{INOK2}$: VIN status (INOK2) 　$>t_{INOK2}+t_{LCS}$: Linear charger status VIN status 　The FN signal is pulled low for the INOK1 status and is floating for the INOK2 status: VIN input voltage is valid 　Others: VIN input voltage is invalid Linear charger status (it is only valid when there is a valid VIN input voltage) 　FN signal is pulled low: The linear charger is enabled and has stopped 　FN signal is floating: The linear charger is disabled, or the linear charger input power is abnormal, or the linear charger is enabled and charging	0
D5	D_VREN	Reference voltage function enable bit	0: Disable the reference voltage circuit 1: Enable the reference voltage circuit and output by the FN signal	0
D6	D_SELLC	Linear charger function enable bit	0: Disable the linear charger 1: Enable the linear charger, charging is controlled by hardware	1

Power Consumption Considerations

The linear charger requires an ultra-low VBAT quiescent current when fully charged or not charging, it needs to be set up correctly to ensure that the battery is maintained at a fully charged state as much as possible. When the linear charger enters the normal charging mode, if the input voltage does not meet the input voltage range of the linear charger or triggers the charge termination state, the linear charging circuit and related circuits will be automatically shut down. At this time, the VBAT quiescent current is I_{BAT_LK1}. When the linear charger function enable bit, D_SELLC, is cleared to 0, the linear charger will be forced to shut down and the VBAT quiescent current will be maintained at I_{BAT_LK1}, even though the input voltage meets the input voltage range of the linear charger.

In addition, during the shutdown of the linear charger, it can still be set via the communication interface. When the D_VREN is set to 1, the FN signal will output a fixed reference voltage, V_{REF}, at this time, an additional power consumption will be generated on the device. In addition to the device operating state, the power consumption of the pull-low resistor of the SDA/SCL signal should also be considered into the VBAT quiescent current.

Application Information

When the FN signal outputs the V_{REF} voltage, do not pull the FN signal low to avoid causing the V_{REF} reference voltage to be invalid. The following table can be referenced in each status to ensure that the FN signal maintains a normal V_{REF} output.

It is recommended to apply the control according to this table after the main firmware has completed the FN function reading and the use of the V_{REF} reference voltage.

Status		Two-Line Control	
VIN	Linear Charger	D_SELLC	D_SELFN
Not connected	Disable	0	1
Invalid	Disable	0	1
Valid	Disable	0	1
Valid	Charging	1[Note]	1
Valid	Fully charged	0	1

Note: During the use of the V_{REF} reference voltage, when the linear charger status changes from charging to fully charged, it will cause the V_{REF} reference voltage to be invalid. To avoid this situation, it is therefore necessary to confirm the linear charger status via the FN signal before and after using the V_{REF} reference voltage. If the statuses are different, it means that the V_{REF} reference voltage was invalidated during its use, so re-execute the above operations or ignore the current V_{REF} output.

Component Selection

VIN Power Supply and Bypass Capacitors

The VIN power supply capacitor, C_{VIN2}, is suggested to have a value of 4.7μF. The power supply bypass capacitor, C_{VIN1}, can be used to filter out high-frequency noises on VIN when the circuit board is powered by the mains and the suggested capacitance value is 0.1μF.

VBAT Output Capacitor

The VBAT output capacitor, C_{VBAT}, is suggested to have a value of 10μF.

Linear Charger Current Regulation Resistor

The linear charger can select the output current according to the needs. It is recommended to use the grounding resistor R_{SET} with the recommended resistance value in the table below.

I_{BAT_CC} (A)	Resistor Package	R_{SET} (kΩ)
1		2
0.5	0603	4
0.1		20

Layout Guideline

To achieve better efficiency and reduce conduction noise, important PCB layout considerations are as follows:

1. Keep the main current path traces as short and wide as possible to reduce parasitic inductance and resistance.

2. Place the input/output capacitors as close as possible to the device pins.

3. Ensure that all feedback connections are short and straight. Components should be placed as close to the pins as possible.

Thermal Consideration

The maximum power dissipation depends upon the thermal resistance of the device package, PCB layout, rate of surrounding airflow and difference between the junction and ambient temperature. The maximum power dissipation can be calculated by the following formula:

$$P_{D(MAX)}=(T_{J(MAX)} - Ta)/\theta_{JA} \qquad (W)$$

Where $T_{J(MAX)}$ is the maximum junction temperature, Ta is the ambient temperature and θ_{JA} is the junction-to-ambient thermal resistance of device package.

For maximum operating rating conditions, the maximum junction temperature is 160°C. However, it is recommended that the maximum junction temperature does not exceed 125°C during normal operation to maintain high reliability. The de-rating curve of the maximum power dissipation is shown below:

$$P_{D(MAX)}=(160°C - 25°C)/(100°C/W)=1.35W$$

For a fixed $T_{J(MAX)}$ of 160°C, the maximum power dissipation depends upon the operating ambient temperature and the package's thermal resistance, θ_{JA}. The de-rating curve below shows the effect of rising ambient temperature on the maximum recommended power dissipation.
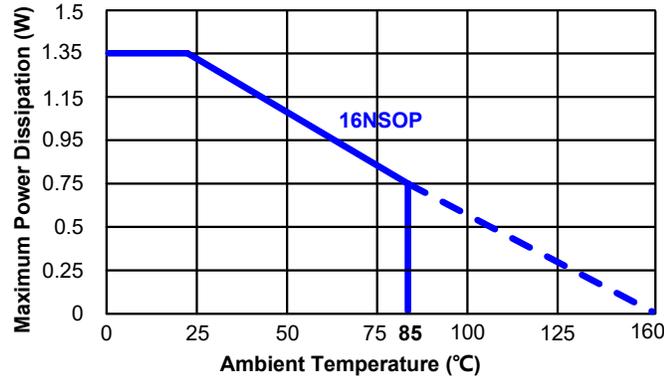
Interrupts

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 interrupt 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 I²C Interface, Touch Keys, TMs, Time Bases, A/D converter and the Multi-function, etc.

The various interrupt enable bits, together with their associated request flags, are shown in the accompanying diagrams with their order of priority. Some interrupt sources have their own individual vector while others share the same multi-function interrupt vector.
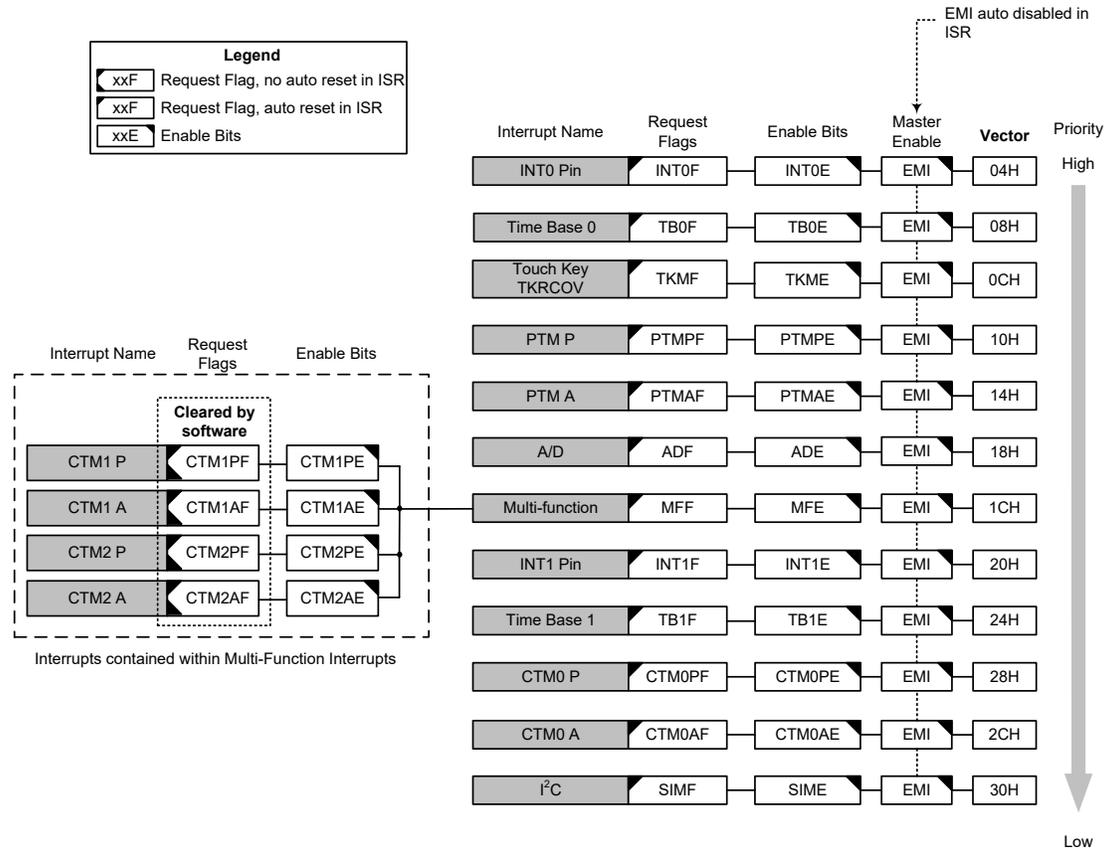
Interrupt Structure

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, as shown in the accompanying table. The number of registers falls into three categories. The first is the INTC0~INTC3 registers which setup the primary interrupts, the second is the MFI register which setup the Multi-function interrupts. Finally there is an INTEG register to 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
I²C Interface	SIME	SIMF	—
Time Base	TBnE	TBnF	n=0~1
Touch Key	TKME	TKMF	—
A/D Converter	ADE	ADF	—

Function	Enable Bit	Request Flag	Notes
CTM	CTMnPE	CTMnPF	n=0~2
	CTMnAE	CTMnAF	
PTM	PTMPE	PTMPF	—
	PTMAE	PTMAF	
Multi-function	MFE	MFF	—

Interrupt Register Bit Naming Conventions

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTC0	—	TKMF	TB0F	INT0F	TKME	TB0E	INT0E	EMI
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC1	MFF	ADF	PTMAF	PTMPF	MFE	ADE	PTMAE	PTMPE
INTC2	CTM0AF	CTM0PF	TB1F	INT1F	CTM0AE	CTM0PE	TB1E	INT1E
INTC3	—	—	—	SIMF	—	—	—	SIME
MFI	CTM2AF	CTM2PF	CTM1AF	CTM1PF	CTM2AE	CTM2PE	CTM1AE	CTM1PE

"—": Unimplemented, read as "0"

Interrupt Register List

- **INTC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	TKMF	TB0F	INT0F	TKME	TB0E	INT0E	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   **TKMF:** Touch key interrupt request flag
         0: No request
         1: Interrupt request

Bit 5   **TB0F:** Time Base 0 interrupt request flag
         0: No request
         1: Interrupt request

Bit 4   **INT0F:** INT0 interrupt request flag
         0: No request
         1: Interrupt request

Bit 3   **TKME:** Touch key interrupt control
         0: Disable
         1: Enable

Bit 2   **TB0E:** Time Base 0 interrupt control
         0: Disable
         1: Enable

Bit 1   **INT0E:** INT0 interrupt control
         0: Disable
         1: Enable

Bit 0   **EMI:** Global interrupt control
         0: Disable
         1: Enable

• **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

• **INTC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	MFF	ADF	PTMAF	PTMPF	MFE	ADE	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   **MFF:** Multi-function 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   **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   **MFE:** Multi-function interrupt control
　　　　0: Disable
　　　　1: Enable

Bit 2   **ADE:** A/D Converter 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

• **INTC2 Register**

Bit	7	6	5	4	3	2	1	0
Name	CTM0AF	CTM0PF	TB1F	INT1F	CTM0AE	CTM0PE	TB1E	INT1E
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 **CTM0AF:** CTM0 comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 6 **CTM0PF:** CTM0 comparator P match 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 **INT1F:** INT1 interrupt request flag
0: No request
1: Interrupt request

Bit 3 **CTM0AE:** CTM0 comparator A match interrupt control
0: Disable
1: Enable

Bit 2 **CTM0PE:** CTM0 comparator P match interrupt control
0: Disable
1: Enable

Bit 1 **TB1E:** Time base 1 interrupt control
0: Disable
1: Enable

Bit 0 **INT1E:** INT1 interrupt control
0: Disable
1: Enable

• **INTC3 Register**

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

Bit 7~5 Unimplemented, read as "0"

Bit 4 **SIMF:** I²C interrupt request flag
0: No request
1: Interrupt request

Bit 3~1 Unimplemented, read as "0"

Bit 0 **SIME:** I²C interrupt control
0: Disable
1: Enable

• **MFI Register**

Bit	7	6	5	4	3	2	1	0
Name	CTM2AF	CTM2PF	CTM1AF	CTM1PF	CTM2AE	CTM2PE	CTM1AE	CTM1PE
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   **CTM2AF:** CTM2 comparator A match interrupt request flag
         0: No request
         1: Interrupt request
     Note that this bit must be cleared to zero by the application program when the interrupt is serviced.

Bit 6   **CTM2PF:** CTM2 comparator P match interrupt request flag
         0: No request
         1: Interrupt request
     Note that this bit must be cleared to zero by the application program when the interrupt is serviced.

Bit 5   **CTM1AF:** CTM1 comparator A match interrupt request flag
         0: No request
         1: Interrupt request
     Note that this bit must be cleared to zero by the application program when the interrupt is serviced.

Bit 4   **CTM1PF:** CTM1 comparator P match interrupt request flag
         0: No request
         1: Interrupt request
     Note that this bit must be cleared to zero by the application program when the interrupt is serviced.

Bit 3   **CTM2AE:** CTM2 comparator A match interrupt control
         0: Disable
         1: Enable

bit 2   **CTM2PE:** CTM2 comparator P match interrupt control
         0: Disable
         1: Enable

Bit 1   **CTM1AE:** CTM1 comparator A match interrupt control
         0: Disable
         1: Enable

Bit 0   **CTM1PE:** CTM1 comparator P match interrupt control
         0: Disable
         1: Enable

Interrupt Operation

When the conditions for an interrupt event occur, such as a TM Comparator P or 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.

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 interrupt structure 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 is in SLEEP or IDLE Mode.

External Interrupt

The external interrupts are controlled by signal transitions on the pins INT0~INT1. An external interrupt request will take place when the external interrupt request flags, INT0F~INT1F, are set, which will occur when a transition, whose type is chosen by the edge select bits, appears on the external interrupt pins. To allow the program to branch to their respective interrupt vector address, the global interrupt enable bit, EMI, and respective external interrupt enable bits, INT0E~INT1E, 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 pins are pin-shared with I/O pins, they can only be configured as external interrupt pins if their 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 pins, a subroutine call to the corresponding external interrupt vector, will take place. When the interrupt is serviced, the external interrupt request flags, INT0F~INT1F, 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 pins 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.

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 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 Interrupt flag, ADF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

Touch Key Interrupt

For a Touch Key interrupt to occur, the global interrupt enable bit, EMI, and the Touch Key interrupt enable bit, TKME, must be first set. An actual Touch Key interrupt will take place when the Touch Key interrupt request flag, TMKF, is set, a situation that will occur when the time slot counter overflows. When the interrupt is enabled, the stack is not full and the Touch Key time slot counter overflow occurs, a subroutine call to the relevant interrupt vector, will take place. When the interrupt is serviced, the Touch Key interrupt request flag will be automatically reset and the EMI bit will also be automatically cleared to disable other interrupts.

Multi-function Interrupts

Within the device there is a Multi-function interrupt. Unlike the other independent interrupts, the interrupt has no independent source, but rather is formed from other existing interrupt sources, namely the CTM1 interrupt and CTM2 interrupt.

A Multi-function interrupt request will take place when the Multi-function interrupt request flag MFF is set. The Multi-function interrupt flag will be set when any of its included functions generate an interrupt request flag. When the Multi-function interrupt is enabled and the stack is not full, and one of the interrupts contained within the Multi-function interrupt occurs, a subroutine call to the Multi-function interrupt vector will take place. When the interrupt is serviced, the related Multi-Function request flag will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts.

However, it must be noted that, although the Multi-function Interrupt request flag will be automatically reset when the interrupt is serviced, the request flag from the original source of the Multi-function interrupt will not be automatically reset and must be manually reset by the application program.

I²C Interrupt

An I²C Interrupt request will take place when the I²C Interrupt request flag, SIMF, is set, which occurs when a byte of data has been received or transmitted by the I²C interface, an I²C slave address match or I²C bus time-out occurrence. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI and the I²C Interrupt enable bit, SIME, 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 I²C Interrupt vector, will take place. When the I²C Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts. The SIMF flag will also be automatically cleared.

Time Base Interrupt

The function of the Time Base Interrupt 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 happen their respective interrupt request flags, TB0F or TB1F, 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 bit, TB0E or TB1E, 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, TB0F or TB1F, 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. Their clock sources f_{PSC0} or f_{PSC1}, originate 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 TB0C or TB1C register to obtain longer interrupt periods whose value ranges. The clock which in

turn controls the Time Base interrupt period is selected using the CLKSEL0[1:0] and CLKSEL1[1:0] bits respectively.

Time Base Interrupt
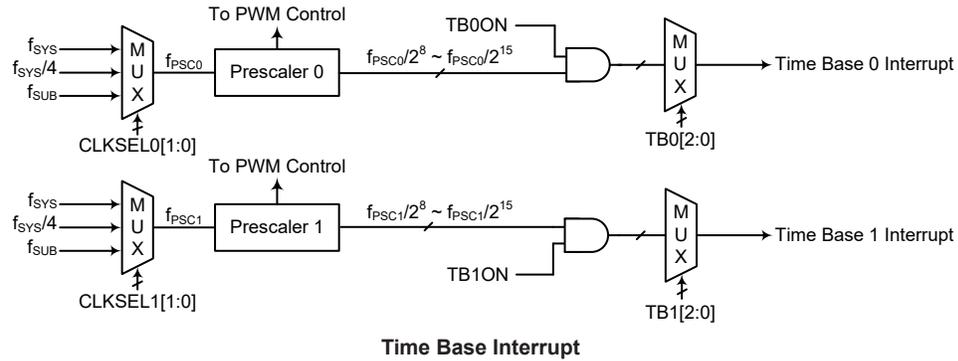
• **TB0C Register**

Bit	7	6	5	4	3	2	1	0
Name	TB0ON	—	CLKSEL01	CLKSEL00	—	TB02	TB01	TB00
R/W	R/W	—	R/W	R/W	—	R/W	R/W	R/W
POR	0	—	0	0	—	0	0	0

Bit 7   **TB0ON:** Time Base 0 enable control
        0: Disable
        1: Enable

Bit 6   Unimplemented, read as "0"

Bit 5~4   **CLKSEL01~CLKSEL00:** Prescaler 0 clock source f_{PSC0} selection
        00: f_{SYS}
        01: $f_{SYS}/4$
        1x: f_{SUB}

Bit 3   Unimplemented, read as "0"

Bit 2~0   **TB02~TB00:** Time Base 0 time-out period selection
        000: $2^4/f_{PSC}$
        001: $2^5/f_{PSC}$
        010: $2^6/f_{PSC}$
        011: $2^7/f_{PSC}$
        100: $2^8/f_{PSC}$
        101: $2^9/f_{PSC}$
        110: $2^{10}/f_{PSC}$
        111: $2^{11}/f_{PSC}$

• **TB1C Register**

Bit	7	6	5	4	3	2	1	0
Name	TB1ON	—	CLKSEL11	CLKSEL10	—	TB12	TB11	TB10
R/W	R/W	—	R/W	R/W	—	R/W	R/W	R/W
POR	0	—	0	0	—	0	0	0

Bit 7   **TB1ON:** Time Base 1 enable control
        0: Disable
        1: Enable

Bit 6   Unimplemented, read as "0"

Bit 5~4   **CLKSEL11~CLKSEL10:** Prescaler 1 clock source f_{PSC1} selection
        00: f_{SYS}
        01: $f_{SYS}/4$
        1x: f_{SUB}

Bit 3   Unimplemented, read as "0"

Bit 2~0   **TB12~TB10:** Time Base 1 time-out period selection

        000: $2^4/f_{PSC}$
        001: $2^5/f_{PSC}$
        010: $2^6/f_{PSC}$
        011: $2^7/f_{PSC}$
        100: $2^8/f_{PSC}$
        101: $2^9/f_{PSC}$
        110: $2^{10}/f_{PSC}$
        111: $2^{11}/f_{PSC}$

TM Interrupts

The Compact 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. The CTM0 and PTM interrupts are independent interrupt and the CTM1 and CTM2 interrupts are contained within the Multi-function Interrupt. For all of the TM types there are two interrupt request flags and two enable control bits. 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.

For the CTM0 and PTM interrupts, to allow the program to branch to the CTMn or PTM 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 the CTMn or PTM comparator match situation occurs, a subroutine call to the CTMn or PTM Interrupt vector location, will take place. When the TM interrupt is serviced, the CTMn or PTM interrupt request flag will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

For the CTM1 and CTM2 interrupts, to allow the program to branch to their respective interrupt vector addresses, the global interrupt enable bit, EMI, respective CTMn Interrupt enable bit, and relevant Multi-function Interrupt enable bit, MFE, must first be set. When the interrupt is enabled, the stack is not full and a CTMn comparator match situation occurs, a subroutine call to the Multi-function Interrupt vector location, will take place. When the CTMn interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the related MFnF flag will be automatically cleared. As the CTMn interrupt request flags will not be automatically cleared, they have to be cleared by the application program.

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 gene rated 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 pin 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 enters 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.

Where a certain interrupt is contained within a Multi-function interrupt, then when the interrupt service routine is executed, as only the Multi-function interrupt request flags, MFF, will be automatically cleared, the individual request flag for the function needs to be 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 the 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.

Configuration Options

Configuration options refer to certain options within the MCU that are programmed into the device during the programming process. During the development process, these options are selected using the HT-IDE software development tools. All options must be defined for proper system function, the details of which are shown in the table.

No.	Options
I/O Pin-Shared Options	
1	$\overline{\text{RES}}$ pin reset function selection: I/O port or $\overline{\text{RES}}$ pin
Oscillator Option	
2	HIRC Frequency Selection – f_{HIRC}: 8MHz, 12MHz or 16MHz

Note: When the HIRC has been configured at a frequency shown in this table, the HIRC1 and HIRC0 bits should also be setup to select the same frequency to achieve the HIRC frequency accuracy specified in the A.C. Characteristics.

Application Circuits
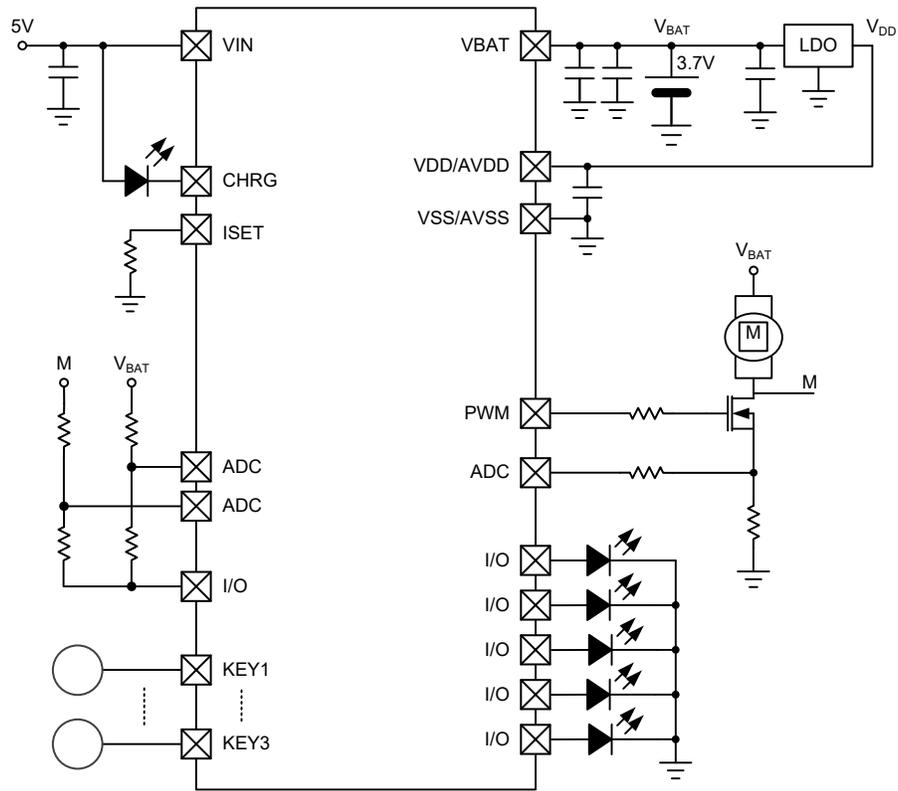
Instruction Set

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 following table depicts a summary of the instruction set categorised according to function and can be consulted as a basic instruction reference using the following listed conventions.

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
ADDM A,[m]	Add ACC to Data Memory	1[Note]	Z, C, AC, OV
ADD A,x	Add immediate data to ACC	1	Z, C, AC, OV
ADC A,[m]	Add Data Memory to ACC with Carry	1	Z, C, AC, OV
ADCM A,[m]	Add ACC to Data memory with Carry	1[Note]	Z, C, AC, OV
SUB A,x	Subtract immediate data from the ACC	1	Z, C, AC, OV
SUB A,[m]	Subtract Data Memory from ACC	1	Z, C, AC, OV
SUBM A,[m]	Subtract Data Memory from ACC with result in Data Memory	1[Note]	Z, C, AC, OV
SBC A,[m]	Subtract Data Memory from ACC with Carry	1	Z, C, AC, OV
SBCM A,[m]	Subtract Data Memory from ACC with Carry, result in Data Memory	1[Note]	Z, C, AC, OV
DAA [m]	Decimal adjust ACC for Addition with result in Data Memory	1[Note]	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	1[Note]	Z
ORM A,[m]	Logical OR ACC to Data Memory	1[Note]	Z
XORM A,[m]	Logical XOR ACC to Data Memory	1[Note]	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	1[Note]	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	1[Note]	Z
DECA [m]	Decrement Data Memory with result in ACC	1	Z
DEC [m]	Decrement Data Memory	1[Note]	Z
Rotate			
RRA [m]	Rotate Data Memory right with result in ACC	1	None
RR [m]	Rotate Data Memory right	1[Note]	None
RRCA [m]	Rotate Data Memory right through Carry with result in ACC	1	C
RRC [m]	Rotate Data Memory right through Carry	1[Note]	C
RLA [m]	Rotate Data Memory left with result in ACC	1	None
RL [m]	Rotate Data Memory left	1[Note]	None
RLCA [m]	Rotate Data Memory left through Carry with result in ACC	1	C
RLC [m]	Rotate Data Memory left through Carry	1[Note]	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	1^Note	None
MOV A,x	Move immediate data to ACC	1	None
Bit Operation			
CLR [m].i	Clear bit of Data Memory	1^Note	None
SET [m].i	Set bit of Data Memory	1^Note	None
Branch Operation			
JMP addr	Jump unconditionally	2	None
SZ [m]	Skip if Data Memory is zero	1^Note	None
SZA [m]	Skip if Data Memory is zero with data movement to ACC	1^Note	None
SZ [m].i	Skip if bit i of Data Memory is zero	1^Note	None
SNZ [m].i	Skip if bit i of Data Memory is not zero	1^Note	None
SIZ [m]	Skip if increment Data Memory is zero	1^Note	None
SDZ [m]	Skip if decrement Data Memory is zero	1^Note	None
SIZA [m]	Skip if increment Data Memory is zero with result in ACC	1^Note	None
SDZA [m]	Skip if decrement Data Memory is zero with result in ACC	1^Note	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 or current page) to TBLH and Data Memory	2^Note	None
TABRDL [m]	Read table (last page) to TBLH and Data Memory	2^Note	None
Miscellaneous			
NOP	No operation	1	None
CLR [m]	Clear Data Memory	1^Note	None
SET [m]	Set Data Memory	1^Note	None
CLR WDT	Clear Watchdog Timer	1	TO, PDF
SWAP [m]	Swap nibbles of Data Memory	1^Note	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 two 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.

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

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

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

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

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

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	$[m] \leftarrow \overline{[m]}$
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	$ACC \leftarrow \overline{[m]}$
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	$[m] \leftarrow ACC + 00H$ or $[m] \leftarrow ACC + 06H$ or $[m] \leftarrow ACC + 60H$ or $[m] \leftarrow ACC + 66H$
Affected flag(s)	C

DEC [m]	Decrement Data Memory
Description	Data in the specified Data Memory is decremented by 1.
Operation	$[m] \leftarrow [m] - 1$
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	$ACC \leftarrow [m] - 1$
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	$TO \leftarrow 0$ $PDF \leftarrow 1$
Affected flag(s)	TO, PDF

INC [m]	Increment Data Memory
Description	Data in the specified Data Memory is incremented by 1.
Operation	$[m] \leftarrow [m] + 1$
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	$ACC \leftarrow [m] + 1$
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 $ACC.(i+1) \leftarrow [m].i; (i=0\sim6)$
$ACC.0 \leftarrow C$
$C \leftarrow [m].7$

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 $[m].i \leftarrow [m].(i+1); (i=0\sim6)$
$[m].7 \leftarrow [m].0$

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 $ACC.i \leftarrow [m].(i+1); (i=0\sim6)$
$ACC.7 \leftarrow [m].0$

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 $[m].i \leftarrow [m].(i+1); (i=0\sim6)$
$[m].7 \leftarrow C$
$C \leftarrow [m].0$

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 $ACC.i \leftarrow [m].(i+1); (i=0\sim6)$
$ACC.7 \leftarrow C$
$C \leftarrow [m].0$

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 $ACC \leftarrow ACC - [m] - \overline{C}$

Affected flag(s) OV, Z, AC, C

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	$[m] \leftarrow ACC - [m] - \overline{C}$
Affected flag(s)	OV, Z, AC, C

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	$[m] \leftarrow [m] - 1$ 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	$ACC \leftarrow [m] - 1$ 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	$[m] \leftarrow FFH$
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	$[m].i \leftarrow 1$
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	$[m] \leftarrow [m] + 1$ 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 $ACC \leftarrow [m] + 1$
Skip if ACC=0

Affected flag(s)   None

SNZ [m].i   Skip if bit i of Data Memory is not 0

Description   If bit i 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 $[m].i \neq 0$

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 $ACC \leftarrow ACC - [m]$

Affected flag(s)   OV, Z, AC, C

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 $[m] \leftarrow ACC - [m]$

Affected flag(s)   OV, Z, AC, C

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 $ACC \leftarrow ACC - x$

Affected flag(s)   OV, Z, AC, C

SWAP [m]   Swap nibbles of Data Memory

Description   The low-order and high-order nibbles of the specified Data Memory are interchanged.

Operation $[m].3 \sim [m].0 \leftrightarrow [m].7 \sim [m].4$

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   The contents of the specified Data Memory are read out and then written to the specified Data Memory again. 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 addressed by the table pointer (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

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

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

16-pin NSOP (150mil) Outline Dimensions
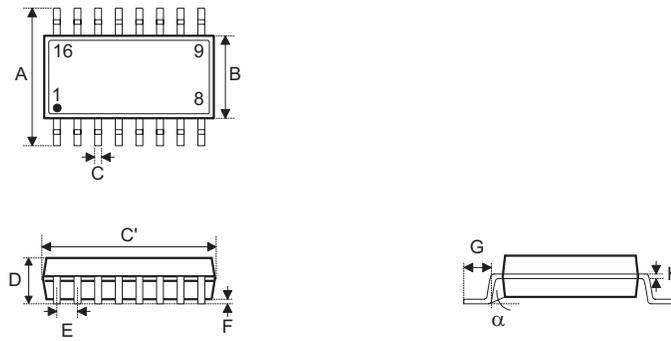
Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A		0.236 BSC	
B		0.154 BSC	
C	0.012	—	0.020
C'		0.390 BSC	
D	—	—	0.069
E		0.050 BSC	
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A		6.00 BSC	
B		3.90 BSC	
C	0.31	—	0.51
C'		9.90 BSC	
D	—	—	1.75
E		1.27 BSC	
F	0.10	—	0.25
G	0.40	—	1.27
H	0.10	—	0.25
α	0°	—	8°