



Wearable Peripheral Intergrated Flash MCU with Touch

BS45F5830/BS45F5831

BS45F5832/BS45F5833

Revision: V1.10 Date: November 18, 2019

www.holtek.com

Table of Contents

Features	6
CPU Features	6
Peripheral Features.....	6
General Description.....	7
Block Diagram.....	7
Selection Table.....	8
Pin Assignment.....	8
Pin Description	9
Absolute Maximum Ratings.....	10
D.C. Characteristics.....	11
Operating Voltage Characteristics.....	11
Standby Current Characteristics	11
Operating Current Characteristics.....	12
A.C. Characteristics.....	12
High Speed Internal Oscillator – HIRC – Frequency Accuracy	12
Low Speed Internal Oscillator Characteristics – LIRC	13
Operating Frequency Characteristic Curves	13
System Start Up Time Characteristics	13
Input/Output Characteristics	14
Input/Output (without Multi-power) D.C. Characteristics	14
Input/Output (with Multi-power) D.C. Characteristics	15
Memory Characteristics	15
LVR/LVD Electrical Characteristics	16
A/D Converter Electrical Characteristics.....	16
Internal Reference Voltage Characteristics.....	17
PGA Electrical Characteristics	17
LDO Electrical Characteristics	17
Linear Charger Electrical Characteristics	18
Power-on Reset Characteristics.....	19
System Architecture	19
Clocking and Pipelining.....	19
Program Counter.....	20
Stack	21
Arithmetic and Logic Unit – ALU	21
Flash Program Memory	22
Structure.....	22
Special Vectors	22
Look-up Table.....	22
Table Program Example.....	23

In Circuit Programming – ICP	24
On-Chip Debug Support – OCDS	25
Data Memory	25
Structure.....	25
General Purpose Data Memory	26
Special Purpose Data Memory	26
Special Function Register Description.....	28
Indirect Addressing Registers – IAR0, IAR1	28
Memory Pointers – MP0, MP1	28
Bank Pointer – BP.....	29
Accumulator – ACC.....	29
Program Counter Low Register – PCL.....	29
Look-up Table Registers – TBLP, TBHP, TBLH.....	29
Status Register – STATUS.....	30
EEPROM Data Memory.....	31
EEPROM Data Memory Structure	31
EEPROM Registers	31
Reading Data from the EEPROM	33
Writing Data to the EEPROM.....	33
Write Protection.....	33
EEPROM Interrupt	33
Programming Considerations.....	33
Oscillators	35
Oscillator Overview	35
System Clock Configurations.....	35
Internal High Speed RC Oscillator – HIRC	36
Internal 32kHz Oscillator – LIRC.....	36
Operating Modes and System Clocks	36
System Clocks	36
System Operation Modes.....	37
Control Registers	38
Operating Mode Switching.....	40
Standby Current Considerations	43
Wake-up	43
Watchdog Timer.....	44
Watchdog Timer Clock Source.....	44
Watchdog Timer Control Register	44
Watchdog Timer Operation	45
Reset and Initialisation.....	46
Reset Functions	46
Reset Initial Conditions	49
Input/Output Ports	52
Pull-high Resistors	52
Port A Wake-up	53

I/O Port Control Registers	53
I/O Port Source Current Selection.....	54
I/O Port Power Source Control.....	54
Pin-shared Functions	55
I/O Pin Structures.....	58
Programming Considerations.....	58
Timer Modules – TM	59
Introduction	59
TM Operation	59
TM Clock Source.....	59
TM Interrupts.....	60
TM External Pins.....	60
Programming Considerations.....	61
Compact Type TM – CTM	62
Compact TM Operation.....	62
Compact Type TM Register Description.....	62
Compact Type TM Operating Modes	66
Standard Type TM – STM	72
Standard TM Operation.....	72
Standard Type TM Register Description	72
Standard Type TM Operation Modes	77
Touch Key Function	86
Touch Key Structure.....	86
Touch Key Register Definition	86
Touch Key Operation.....	95
Touch Key Interrupts	104
Programming Considerations.....	104
Voltage Regulator – LDO.....	105
Analog to Digital Converter	105
A/D Converter Overview	105
A/D Converter Register Descriptions	106
A/D Converter Reference Voltage.....	109
A/D Converter Input Signals.....	110
A/D Converter Operation.....	110
Conversion Rate and Timing Diagram	111
Summary of A/D Conversion Steps.....	112
Programming Considerations.....	113
A/D Transfer Function	113
A/D Programming Examples.....	114
Linear Charger	115
Linear Charger Control Register	116
Functional Description.....	116

I²C Interface	117
I ² C Interface Operation.....	118
I ² C Registers	119
I ² C Bus Communication	122
I ² C Bus Start Signal.....	122
I ² C Slave Address	123
I ² C Bus Read/Write Signal	123
I ² C Bus Slave Address Acknowledge Signal	123
I ² C Bus Data and Acknowledge Signal	123
I ² C Time-out Control.....	125
Low Voltage Detector – LVD	126
LVD Register	126
LVD Operation.....	127
Interrupts	127
Interrupt Registers.....	127
Interrupt Operation	132
External Interrupt.....	133
I ² C Interface Interrupt.....	134
Time Base Interrupts	134
A/D Converter Interrupt.....	136
LVD Interrupt.....	136
EEPROM Interrupt	136
Multi-function Interrupts.....	136
Touch Key TKRCOV Interrupt.....	137
Touch Key Module TKTH Interrupt.....	137
TM Interrupts.....	137
Interrupt Wake-up Function.....	138
Programming Considerations.....	138
Application Circuits.....	139
Instruction Set.....	140
Introduction	140
Instruction Timing	140
Moving and Transferring Data.....	140
Arithmetic Operations.....	140
Logical and Rotate Operation	141
Branches and Control Transfer	141
Bit Operations	141
Table Read Operations	141
Other Operations.....	141
Instruction Set Summary	142
Table Conventions.....	142
Instruction Definition.....	144
Package Information	153
SAW Type 24-pin QFN (3mm×3mm×0.55mm) Outline Dimensions	154

Features

CPU Features

- Operating voltage
 - ♦ $f_{\text{SYS}}=4\text{MHz}$: 2.2V~5.5V
 - ♦ $f_{\text{SYS}}=8\text{MHz}$: 2.2V~5.5V
 - ♦ $f_{\text{SYS}}=12\text{MHz}$: 2.7V~5.5V
- Up to 0.33 μs instruction cycle with 12MHz system clock at $V_{\text{DD}}=5\text{V}$
- Power down and wake-up functions to reduce power consumption
- Oscillator types
 - ♦ Internal High Speed 4/8/12MHz 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
- 63 powerful instructions
- 4-level subroutine nesting
- Bit manipulation instruction

Peripheral Features

- Flash Program Memory: 2K \times 16
- RAM Data Memory: 128 \times 8
- Touch Key Data Memory: 16 \times 8
- True EEPROM Memory: 32 \times 8
- Watchdog Timer function
- Up to 16 bidirectional I/O lines
- Programmable I/O port source current for LED driver
- Single external interrupt line shared with I/O pin
- Multiple Timer Modules for time measurement, input capture, compare match output or PWM output or single pulse output function
 - ♦ 1 Standard type 10-bit Timer Module – STM
 - ♦ 1 Compact type 10-bit Timer Module – CTM
- 4 external channels 12-bit resolution A/D converter with Programmable Internal Reference Voltage V_{R} .
- I²C Interface
- Dual Time-Base functions for generation of fixed time interrupt signals
- 4 touch key functions
- Internal LDO function
- Linear charger function
- Low voltage reset function
- Low voltage detect function
- Package type: 24-pin QFN

General Description

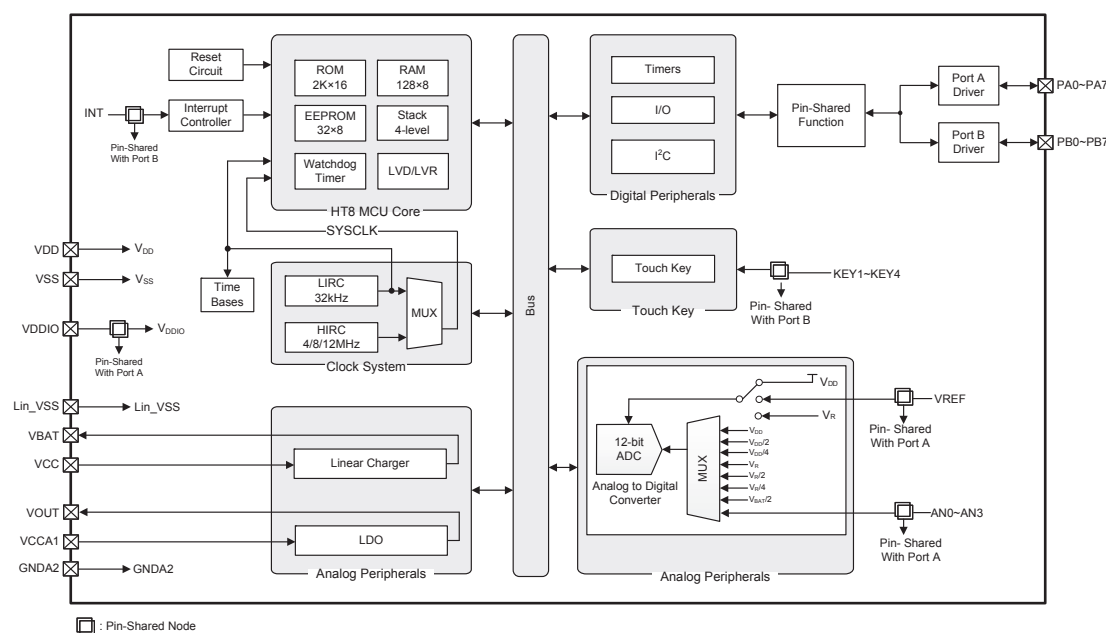
The series of devices are Flash Memory A/D type 8-bit high performance RISC architecture microcontroller, specifically designed for the wearable peripheral circuit applications. With all touch key functions provided internally and with the convenience of Flash Memory multi-programming features, these devices also include a wide range of functions and features. Other memory includes an area of RAM Data Memory as well as an area of true EEPROM memory for storage of non-volatile data such as serial numbers, calibration data etc.

Analog features include a multi-channel 12-bit A/D converter function. Multiple and extremely flexible Timer Modules provide timing, pulse generation and PWM generation functions. 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. In addition, an internal LDO function provides various power options to the I²C interface function. Protective features such as an internal Watchdog Timer, Low Voltage Reset and Low Voltage Detector coupled with excellent noise immunity and ESD protection ensure that reliable operation is maintained in hostile electrical environments.

A full choice of internal high and low oscillators is provided including two fully integrated system oscillators which require no external components for their implementation. The ability to operate and switch dynamically between a range of operating modes using different clock sources gives users the ability to optimize microcontroller operation and minimize power consumption.

The inclusion of flexible I/O programming features, a fully integrated Linear Charger and Time-Base functions along with many other features further enhance devices functionality and flexibility.

Block Diagram



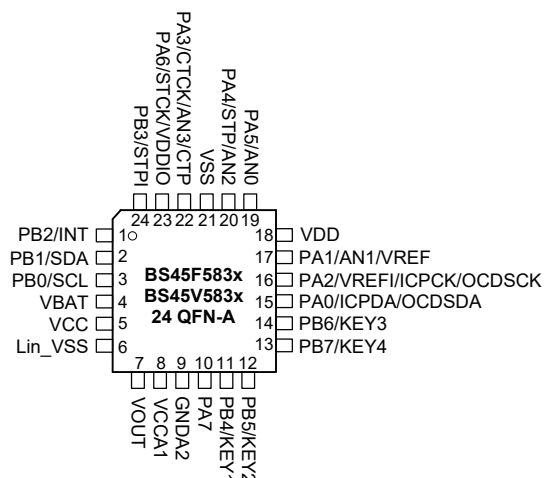
Selection Table

Most features are common to these devices, the main features distinguishing them are LDO output voltage and Linear charger constant voltage. The following table summarises the main features of each device.

Part No.	V _{DD}	Program Memory	Data Memory	Data EEPROM	I/O	External Interrupt	A/D Converter
BS45F5830	2.2V~5.5V	2K×16	128×8	32×8	16	1	12-bit×6
BS45F5831	2.2V~5.5V	2K×16	128×8	32×8	16	1	12-bit×6
BS45F5832	2.2V~5.5V	2K×16	128×8	32×8	16	1	12-bit×6
BS45F5833	2.2V~5.5V	2K×16	128×8	32×8	16	1	12-bit×6

Part No.	Timer Module	Time Base	I ² C	LDO	Linear Charger	Motor Driver	Stacks	Package
BS45F5830	10-bit CTM×1 10-bit STM×1	2	√	3.3V	4.2V	√	4	24QFN
BS45F5831	10-bit CTM×1 10-bit STM×1	2	√	3.3V	4.35V	√	4	24QFN
BS45F5832	10-bit CTM×1 10-bit STM×1	2	√	3.0V	4.2V	√	4	24QFN
BS45F5833	10-bit CTM×1 10-bit STM×1	2	√	3.0V	4.35V	√	4	24QFN

Pin Assignment



- Note: 1. If the pin-shared pin functions have multiple outputs, the desired pin-shared function is determined by the corresponding software control bits.
2. The OCSDA and OCDSCK pins are supplied for the OCDS dedicated pins and as such only available for the BS45V583x devices which are the OCDS EV chip for the BS45F583x devices.

Pin Description

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

Pin Name	Function	OPT	I/T	O/T	Description
PA0/ICPDA/OCSDSA	PA0	PAPU PAWU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	ICPDA	—	ST	CMOS	ICP Data/Address pin
	OCSDSA	—	ST	CMOS	OCDS Data/Address pin, for EV chip only.
PA1/AN1/VREF	PA1	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	AN1	PAS0	AN	—	A/D Converter external analog input
	VREF	PAS0	AN	—	A/D Converter external reference voltage input
PA2/VREFI/ICPCK/ OCDSCK	PA2	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	VREFI	PAS0	AN	—	A/D Converter PGA input
	ICPCK	—	ST	—	ICP clock
	OCDSCK	—	ST	—	OCDS clock pin, for EV chip only.
PA3/CTCK/AN3/CTP	PA3	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	CTCK	PAS0	ST	—	CTM clock input
	AN3	PAS0	AN	—	A/D Converter external analog input
	CTP	PAS0	—	CMOS	CTM output
PA4/AN2/STP	PA4	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	AN2	PAS1	AN	—	A/D Converter external analog input
	STP	PAS1	—	CMOS	STM output
PA5/AN0	PA5	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	AN0	PAS1	AN	—	A/D Converter external analog input
PA6/STCK/VDDIO	PA6	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	STCK	PAS1	ST	—	STM clock input
	VDDIO	PAS1	PWR	—	Positive power supply for PB3~PB0 pin
PA7	PA7	PAPU PAWU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
PB0/SCL	PB0	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	SCL	PBS0	ST	NMOS	I ² C clock line
PB1/SDA	PB1	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	SDA	PBS0	ST	NMOS	I ² C data line
PB2/INT	PB2	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	INT	INTEG INTC0	ST	—	External interrupt input
PB3/STPI	PB3	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	STPI	—	ST	—	STM capture input

Pin Name	Function	OPT	I/T	O/T	Description
PB4/KEY1	PB4	PBPU PBS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	KEY1	PBS1	AN	—	Touch key input
PB5/KEY2	PB5	PBPU PBS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	KEY2	PBS1	AN	—	Touch key input
PB6/KEY3	PB6	PBPU PBS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	KEY3	PBS1	AN	—	Touch key input
PB7/KEY4	PB7	PBPU PBS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	KEY4	PBS1	AN	—	Touch key input
VDD	VDD	—	PWR	—	Positive power supply
VSS	VSS	—	PWR	—	Negative power supply, ground.
VCC	VCC	—	PWR	—	Linear Charger power input
VBAT	VBAT	—	—	PWR	Linear Charger output
Lin_VSS	Lin_VSS	—	PWR	—	Linear Charger negative power supply
VCCA1	VCCA1	—	PWR	—	HV Power for LDO input voltage
VOOUT	VOOUT	—	—	PWR	LDO output voltage
GND A2	GND A2	—	PWR	—	LDO ESD ground

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

Absolute Maximum Ratings

Supply Voltage	$V_{SS}-0.3V$ to $6.0V$
Input Voltage	$V_{SS}-0.3V$ to $V_{DD}+0.3V$
Storage Temperature.....	$-50^{\circ}C$ to $125^{\circ}C$
Operating Temperature.....	$-40^{\circ}C$ to $85^{\circ}C$
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 devices. Functional operation of the devices at other conditions beyond those listed in the specification is not implied and prolonged exposure to extreme conditions may affect devices reliability.

D.C. Characteristics

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

Operating Voltage Characteristics

Ta=-40°C~85°C

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
		Conditions				
V _{DD}	Operating Voltage – HIRC	f _{SYS} =f _{HIRC} =4MHz	2.2	—	5.5	V
		f _{SYS} =f _{HIRC} =8MHz	2.2	—	5.5	
		f _{SYS} =f _{HIRC} =12MHz	2.7	—	5.5	
	Operating Voltage – LIRC	f _{SYS} =f _{LIRC} =32kHz	2.2	—	5.5	V

Standby Current Characteristics

Ta=25°C, unless otherwise specified.

Symbol	Standby Mode	Test Conditions		Min.	Typ.	Max.	Max. @85°C	Unit
		V _{DD}	Conditions					
I _{STB}	SLEEP Mode	2.2V	WDT on	—	1.2	2.4	2.9	μA
		3V		—	1.5	3	3.6	
		5V		—	3	5	6	
	IDLE0 Mode – LIRC	2.2V	f _{SUB} on	—	2.4	4	4.8	μA
		3V		—	3	5	6	
		5V		—	5	10	12	
	IDLE1 Mode – HIRC	2.2V	f _{SUB} on, f _{SYS} =4MHz	—	144	200	240	μA
		3V		—	180	250	300	
		5V		—	400	600	720	
		2.2V	f _{SUB} on, f _{SYS} =8MHz	—	288	400	480	μA
		3V		—	360	500	600	
		5V		—	600	800	960	
		2.7V	f _{SUB} on, f _{SYS} =12MHz	—	432	600	720	μA
		3V		—	540	750	900	
		5V		—	800	1200	1440	

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

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

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	2.2V	f _{sys} =32kHz	—	8	16	μA
		3V		—	10	20	
		5V		—	30	50	
	FAST Mode – HIRC	2.2V	f _{sys} =4MHz	—	0.3	0.5	mA
		3V		—	0.4	0.6	
		5V		—	0.8	1.2	
		2.2V	f _{sys} =8MHz	—	0.6	1.0	mA
		3V		—	0.8	1.2	
		5V		—	1.6	2.4	
		2.7V	f _{sys} =12MHz	—	1.0	1.4	mA
		3V		—	1.2	1.8	
		5V		—	2.4	3.6	

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

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

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}	4MHz Writer Trimmed HIRC Frequency	3V/5V	25°C	-1%	4	+1%	MHz
			-40°C ~ 85°C	-2%	4	+2%	
		2.2V~5.5V	25°C	-2.5%	4	+2.5%	
			-40°C ~ 85°C	-3%	4	+3%	
	8MHz Writer Trimmed HIRC Frequency	3V/5V	25°C	-1%	8	+1%	MHz
			-40°C ~ 85°C	-2%	8	+2%	
		2.2V~5.5V	25°C	-2.5%	8	+2.5%	
			-40°C ~ 85°C	-3%	8	+3%	
	12MHz Writer Trimmed HIRC Frequency	5V	25°C	-1%	12	+1%	MHz
			-40°C ~ 85°C	-2%	12	+2%	
		2.7V~5.5V	25°C	-2.5%	12	+2.5%	
			-40°C ~ 85°C	-3%	12	+3%	

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.

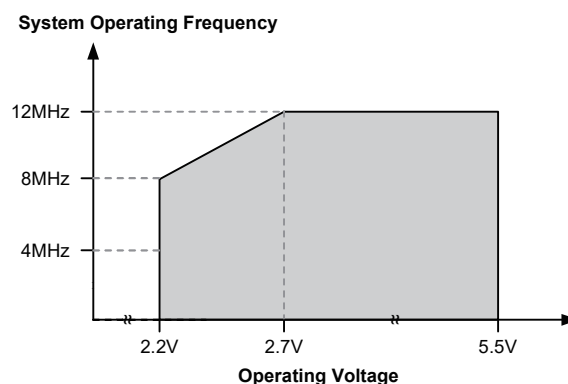
2. The row below the 3V/5V trim voltage row is provided to show the values for the full V_{DD} range operating voltage. It is recommended that the trim voltage is fixed at 3V for application voltage ranges from 2.2 to 3.6V and fixed at 5V for application voltage ranges from 3.3V to 5.5V.

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 $\pm 20\%$.

Low Speed Internal Oscillator Characteristics – LIRC

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V _{DD}	Temp.				
f _{LIRC}	LIRC Frequency	2.2V~5.5V	25°C	-10%	32	+10%	kHz
			-40°C ~ 85°C	-50%	32	+60%	
t _{START}	LIRC Start Up Time	—	—	—	—	100	μs

Operating Frequency Characteristic Curves



System Start Up Time Characteristics

T_a=-40°C~85°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V _{DD}	Conditions				
t _{SST}	System Start-up Time	—	f _{SYS} =f _H ~ f _H /64, f _H =f _{HIRC}	—	16	—	t _{HIRC}
	Wake-up from condition where f _{SYS} is off	—	f _{SYS} =f _{SUB} =f _{LIRC}	—	2	—	t _{LIRC}
	System Start-up Time	—	f _{SYS} =f _H ~ f _H /64, f _H =f _{HIRC}	—	2	—	t _H
	Wake-up from condition where f _{SYS} is on	—	f _{SYS} =f _{SUB} =f _{LIRC}	—	2	—	t _{SUB}
	System Speed Switch Time FAST to SLOW Mode or SLOW to FAST Mode	—	f _{HIRC} switches from off → on	—	16	—	t _{HIRC}
t _{RSTD}	System Reset Delay Time Reset source from Power-on reset or LVR hardware reset	—	RR _{POR} =5V/ms	42	48	54	ms
	System Reset Delay Time LVRC/WDTC/RSTC software reset	—	—				
	System Reset Delay Time Reset source from WDT overflow	—	—	14	16	18	ms
t _{SRESET}	Minimum Software Reset Width to Reset	—	—	45	90	120	μs

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

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

3. If the LIRC is used as the system clock and if it is off when in the SLEEP Mode, then an additional LIRC start up time, t_{START} , as provided in the LIRC frequency table, must be added to the t_{SST} time in the table above.
4. The System Speed Switch Time is effectively the time taken for the newly activated oscillator to start up.

Input/Output Characteristics

Input/Output (without Multi-power) D.C. Characteristics

$T_a = -40^{\circ}\text{C} \sim 85^{\circ}\text{C}$

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{IL}	Input Low Voltage for I/O Ports or Input Pins (except PB0~PB3 pins)	5V	—	0	—	1.5	V
		—	—	0	—	$0.2V_{DD}$	
V_{IH}	Input High Voltage for I/O Ports or Input Pins (except PB0~PB3 pins)	5V	—	3.5	—	5	V
		—	—	$0.8V_{DD}$	—	V_{DD}	
I_{OL}	Sink Current for I/O Pins (except PA3 and PB0~PB3 pins)	3V	$V_{OL} = 0.1V_{DD}$	16	32	—	mA
		5V		32	65	—	
	Sink Current for PA3 Pin	3V	$V_{OL} = 0.1V_{DD}$	150	200	—	mA
		5V		200	250	—	
I_{OH}	Source Current for PA4~PA7 pins	3V	$V_{OH} = 0.9V_{DD}$, SLEDC[1, 0]=00B	-0.7	-1.5	—	mA
		5V		-1.5	-2.9	—	
		3V	$V_{OH} = 0.9V_{DD}$, SLEDC[1, 0]=01B	-1.3	-2.5	—	mA
		5V		-2.5	-5.1	—	
		3V	$V_{OH} = 0.9V_{DD}$, SLEDC[1, 0]=10B	-1.8	-3.6	—	mA
		5V		-3.6	-7.3	—	
		3V	$V_{OH} = 0.9V_{DD}$, SLEDC[1, 0]=11B	-4	-8	—	mA
		5V		-8	-16	—	
R_{PH}	Pull-high Resistance for PA6 and PB0~PB3 pins ^(Note)	3V	LVPU=0	20	60	100	k Ω
		5V		10	30	50	
		3V	PxPU=FFH (x=A or B)	6.67	15	23	
		5V		3.5	7.5	12	
	Pull-high Resistance for I/O Ports (except PA6, PB0~PB3 pins) ^(Note)	3V	—	20	60	100	k Ω
		5V	—	10	30	50	
I_{LEAK}	Input Leakage Current (except PB0~PB3 pins)	5V	$V_{IN} = V_{DD}$ or $V_{IN} = V_{SS}$	—	—	± 1	μA
t_{INT}	Interrupt Input Pin Minimum Pulse Width	—	—	10	—	—	μs
t_{TCK}	CTCK and STCK Clock Input Minimum Pulse Width	—	—	0.3	—	—	μs
t_{TPI}	STPI Capture Input Minimum Pulse Width	—	—	0.3	—	—	μs

Input/Output (with Multi-power) D.C. Characteristics

Ta=-40°C~85°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V _{DD}	Conditions				
V _{DD}	V _{DD} Power Supply for PB0~PB3 Pins	—	—	2.2	5	5.5	V
V _{DDIO}	V _{DDIO} Power Supply for PB0~PB3 Pins	—	—	1.8	—	V _{DD}	V
V _{IL}	Input Low Voltage for PB0~PB3 Pins	5V	—	0	—	1.5	V
		—	Pin power=V _{DD} or V _{DDIO}	0	—	0.2 (V _{DD} /V _{DDIO})	
V _{IH}	Input High Voltage for PB0~PB3 Pins	5V	—	3.5	—	5	V
		—	Pin power=V _{DD} or V _{DDIO}	0.8 (V _{DD} /V _{DDIO})	—	V _{DD} /V _{DDIO}	
I _{OL}	Sink Current for PB0~PB3 Pins	3V	V _{OL} =0.1 (V _{DD} or V _{DDIO}) V _{DDIO} =V _{DD}	16	32	—	mA
		5V	V _{OL} =0.1 (V _{DD} or V _{DDIO}) V _{DDIO} =V _{DD}	32	65	—	mA
			V _{OL} =0.1 V _{DDIO} , V _{DDIO} =3V	20	40	—	mA
I _{OH}	Source Current for PB0~PB3 Pins	3V	V _{OH} =0.9 (V _{DD} or V _{DDIO}), V _{DDIO} =V _{DD}	-4	-8	—	mA
		5V	V _{OH} =0.9 (V _{DD} or V _{DDIO}), V _{DDIO} =V _{DD}	-8	-16	—	mA
			V _{OH} =0.9 V _{DDIO} , V _{DDIO} =3V	-2.5	-5	—	mA
R _{PH}	Pull-high Resistance for PB0~PB3 Pins ^(Note)	3V	V _{DDIO} =V _{DD}	20	60	100	kΩ
		5V	V _{DDIO} =V _{DD}	10	30	50	kΩ
			V _{DDIO} =3V	36	110	180	kΩ
I _{LEAK}	Input Leakage Current for PB0~PB3 Pins	5V	V _{IN} =V _{SS} or V _{IN} =V _{DD} or V _{DDIO}	—	—	±1	μA

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

Memory Characteristics

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

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V _{DD}	Conditions				
Flash Program / Data EEPROM Memory							
t _{DEW}	Erase / Write Time – Flash Program Memory	—	—	—	2	3	ms
	Write Cycle Time – Data EEPROM Memory	—	—	—	4	6	ms
E _P	Cell Endurance – Flash Program Memory	—	—	10K	—	—	E/W
	Cell Endurance – Data EEPROM Memory	—	—	100K	—	—	E/W
t _{RETD}	ROM Data Retention time	—	Ta=25°C	—	40	—	Year
RAM Data Memory							
V _{DR}	RAM Data Retention voltage	—	Device in SLEEP Mode	1.0	—	—	V

LVR/LVD 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 2.1V	-5%	2.1	+5%	V
		—	LVR enable, voltage select 2.55V		2.55		
		—	LVR enable, voltage select 3.15V		3.15		
		—	LVR enable, voltage select 3.8V		3.8		
V _{LVD}	Low Voltage Detection Voltage	—	LVD enable, voltage select 2.0V	-5%	2.0	+5%	V
		—	LVD enable, voltage select 2.2V		2.2		
		—	LVD enable, voltage select 2.4V		2.4		
		—	LVD enable, voltage select 2.7V		2.7		
		—	LVD enable, voltage select 3.0V		3.0		
		—	LVD enable, voltage select 3.3V		3.3		
		—	LVD enable, voltage select 3.6V		3.6		
		—	LVD enable, voltage select 4.0V		4.0		
I _{LVR/LVDBG}	Operating Current	3V	LVD enable, LVR enable, VBGEN=0	—	—	20	μA
		5V	VBGEN=0	—	20	25	μA
		3V	LVD enable, LVR enable, VBGEN=1	—	—	25	μA
		5V	VBGEN=1	—	25	30	μA
t _{LVDS}	LVDO Stable Time	—	For LVR enable, VBGEN=0, LVD off → on	—	—	15	μs
t _{LVR}	Minimum Low Voltage Width to Reset	—	—	120	240	480	μs
t _{LVD}	Minimum Low Voltage Width to Interrupt	—	—	60	120	240	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	—	V _{DD}	V
N _R	Resolution	—	—	—	—	12	Bit
DNL	Differential Non-linearity	—	V _{REF} =V _{DD} , t _{ADCK} =0.5μs	-3	—	3	LSB
INL	Integral Non-linearity	—	V _{REF} =V _{DD} , t _{ADCK} =0.5μs	-4	—	4	LSB
I _{ADC}	Additional Current for A/D Converter Enable	2.2V	No load, t _{ADCK} =0.5μs	—	300	420	μA
		3V		—	340	500	
		5V		—	500	700	
t _{ADCK}	Clock Period	—	—	0.5	—	10	μs
t _{ON2ST}	A/D Converter On-to-Start Time	—	—	4	—	—	μs

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V _{DD}	Conditions				
t _{ADC}	Conversion Time (Include A/D Converter Sample and Hold Time)	—	—	—	16	—	t _{ADCK}

Internal Reference Voltage Characteristics

Ta=-40°C~85°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V _{DD}	Conditions				
V _{BG}	Bandgap Reference Voltage	—	—	-5%	1.04	+5%	V
I _{BG}	Additional Current for Bandgap Reference Enable	—	LVR disable, LVD disable	—	—	25	μA

Note: The V_{BG} voltage is used as the PGA positive input.

PGA Electrical Characteristics

Ta=-40°C~85°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V _{DD}	Conditions				
V _{DD}	Operating Voltage	—	—	2.2	—	5.5	V
I _{PGA}	Additional Current for PGA Enable	3V 5V	No load	— —	270 270	550 600	μA
V _{OR}	PGA Maximum Output Voltage Range	3V 5V	— —	V _{SS} +0.1 V _{SS} +0.1	— —	V _{DD} -0.1 V _{DD} -0.1	V
Ga	PGA Gain Accuracy	—	Gain=1; V _{RI} >0.1V Gain=2, 3, 4; V _{RI} >0.05V	-5	—	+5	%
V _{IR}	PGA Input Voltage Range	3V	Gain=1; Ga <±5%	V _{SS} +0.1	—	V _{DD} -0.1	V
		5V		V _{SS} +0.1	—	V _{DD} -0.1	V
		3V	Gain=2, 3, 4; Ga <±5%	V _{SS} +0.05	—	V _{OR(Max)} /Gain	V
		5V		V _{SS} +0.05	—	V _{OR(Max)} /Gain	V

LDO Electrical Characteristics

V_{IN}=V_{OUT}+1V, C_{LOAD}=1μF, Ta=-40°C~85°C, unless otherwise specify

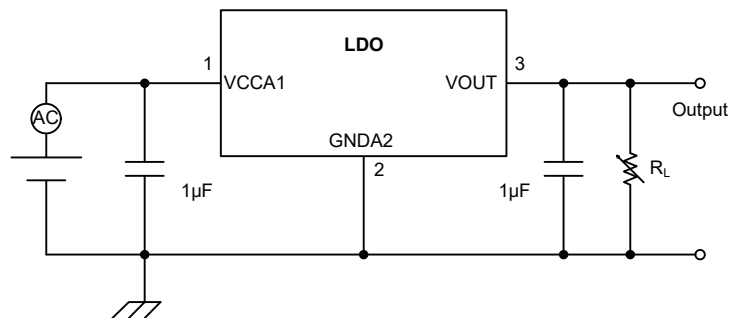
Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V _{IN}	Conditions				
V _{IN}	Input Voltage	—	—	V _{OUT} +0.15	5	5.5	V
V _{OUT}	Output Voltage for BS45F5830/BS45F5831	—	Ta=25°C, I _{LOAD} =1mA, V _{OUT} =3.3V	-2%	3.3	+2%	V
		—	Ta=-40°C ~ 85°C, I _{LOAD} =1mA, V _{OUT} =3.3V	-5%	3.3	+5%	V
	Output Voltage for BS45F5832/BS45F5833	—	Ta=25°C, I _{LOAD} =1mA, V _{OUT} =3.0V	-2%	3.0	+2%	V
		—	Ta=-40°C ~ 85°C, I _{LOAD} =1mA, V _{OUT} =3.0V	-5%	3.0	+5%	V
ΔV _{LOAD}	Load Regulation ⁽¹⁾	—	1mA ≤ I _{LOAD} ≤ 200mA	—	—	0.005	%/mA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V _{IN}	Conditions				
V _{DROP}	Dropout Voltage ⁽²⁾ for BS45F5830/BS45F5831	—	V _{OUT} =3.3V, ΔV _{OUT} =-2% I _{LOAD} =100mA	—	100	150	mV
	Dropout Voltage ⁽²⁾ for BS45F5832/BS45F5833	—	V _{OUT} =3.0V, ΔV _{OUT} =-2% I _{LOAD} =100mA	—	100	150	mV
I _{OUT}	Output Current	—	V _{IN} =V _{OUT} + 0.4V, ΔV _{OUT} =-3%	250	—	—	mA
I _Q	Quiescent Current	5V	No load	—	1	2	μA
ΔV _{LINE}	Line Regulation	—	V _{OUT} + 0.1V ≤ V _{IN} ≤ 5.5V, I _{LOAD} =10mA	—	—	0.2	%/V
V _{NOISE}	Output Voltage Noise	5V	I _{LOAD} =30mA, BW=10Hz to 100kHz	—	60	—	μV
RR	Ripple Rejection ⁽³⁾	—	V _{IN} =5V _{DC} +1V _{P-P(AC)} , I _{LOAD} =50mA, f=1kHz	—	70	—	dB
I _{LIMIT}	Current Limit	—	ΔV _{OUT} =-10%	280	—	—	mA
I _{SHORT}	Fold-back Current	—	V _{OUT} short to GND	75	—	—	mA

Note: 1. Load regulation is measured at a constant junction temperature, using pulse testing with a low ON time and is guaranteed up to the maximum power dissipation. Power dissipation is determined by the input/output differential voltage and the output current. Guaranteed maximum power dissipation will not be available over the full input/output range. The maximum allowable power dissipation at any ambient temperature is $P_D = (T_{J(MAX)} - T_a) / \theta_{JA}$.

2. Dropout voltage is defined as the input voltage minus the output voltage that produces a 2% change in the output voltage from the value at appointed V_{IN}.

3. Ripple rejection ratio measurement circuit. $RR = 20 \times \log(\Delta V_{IN} / \Delta V_{OUT})$.



Linear Charger Electrical Characteristics

V_{CC}=5V, T_a=-40°C~85°C

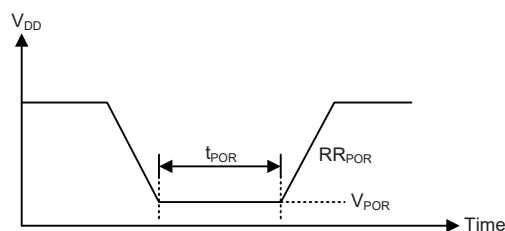
Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V _{CC}	Conditions				
V _{IN}	Input Voltage	—	V _{BAT} <4.0V	4.0	5	5.5	V
		—	V _{BAT} >4.0V	V _{BAT} +0.1V	5	5.5	
V _{CV}	V _{BAT} on Constant Voltage for BS45F5830/BS45F5832	—	—	-1%	4.2	1%	V
	V _{BAT} on Constant Voltage for BS45F5831/BS45F5833	—	—	-1%	4.35	1%	V
V _{SCC}	V _{BAT} on Trickle Current Threshold	—	—	-3%	3	+3%	V
V _{Recharge}	V _{BAT} Recharging Threshold	—	—	-3%	4.0	+3%	V
I _{BAT_SCC}	I _{BAT} in Trickle Current Mode	—	I _{BAT_CC} set by LCS[3:0]	—	I _{BAT_CC} /10	—	mA
I _{BAT_CC}	I _{BAT} in Constant Current mode	—	I _{BAT_CC} set by LCS[3:0]	-10%	I _{BAT_CC}	+10%	mA
I _{BAT_Cut}	I _{BAT} Charge Cut-off Current	—	I _{BAT_CC} set by LCS[3:0]	-10%	I _{BAT_CC} /10	—	mA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V _{CC}	Conditions				
I _{VCC_OP}	I _{VCC} Operating Current	—	V _{CC} =5V	—	100	150	μA
I _{BAT_LK}	I _{BAT} Leak Current	—	V _{BAT} =4.2V, V _{CC} pin is floating	—	-1	-2	μA
		—	V _{BAT} =4.2V, V _{CC} =0V	—	-1	-2	

Power-on Reset Characteristics

Ta=25°C

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



System Architecture

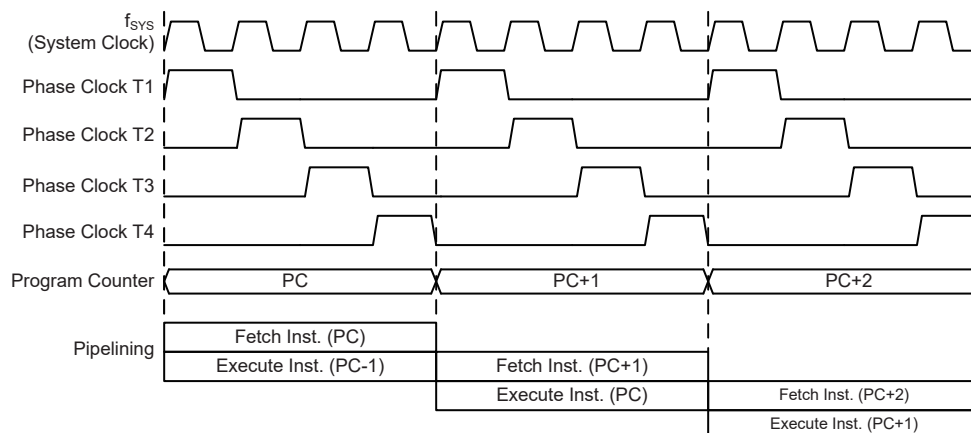
A key factor in the high-performance features of the Holtek range of microcontrollers is attributed to their internal system architecture. The devices take advantage of the usual features found within RISC microcontrollers providing increased speed of operation and enhanced performance. The pipelining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

Clocking and Pipelining

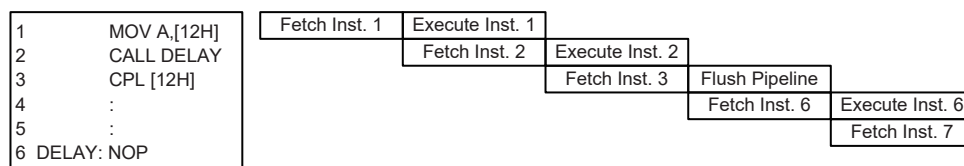
The main system clock, derived from either a HIRC or LIRC oscillator is subdivided into four internally generated non-overlapping clocks, T1~T4. The Program Counter is incremented at the beginning of the T1 clock during which time a new instruction is fetched. The remaining T2~T4 clocks carry out the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

contents of the Program Counter are changed, such as subroutine calls or jumps, in which case the instruction will take one more instruction cycle to execute.

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



System Clocking and Pipelining



Instruction Fetching

Program Counter

During program execution, the Program Counter is used to keep track of the address of the next instruction to be executed. It is automatically incremented by one each time an instruction is executed except for instructions, such as “JMP” or “CALL” that 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	
Program Counter High Byte	PCL Register
PC10~PC8	PCL7~PCL0

Program Counter

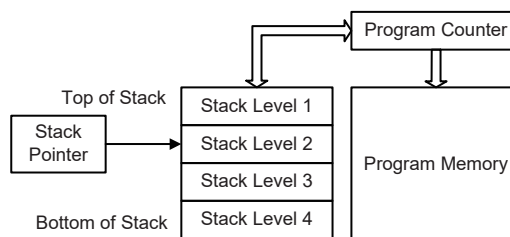
The lower byte of the Program Counter, known as the Program Counter Low register or PCL, is available for program control and is a readable and writeable register. By transferring data directly into this register, a short program jump can be executed directly; however, as only this low byte is available 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.

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

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

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



Arithmetic and Logic Unit – ALU

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

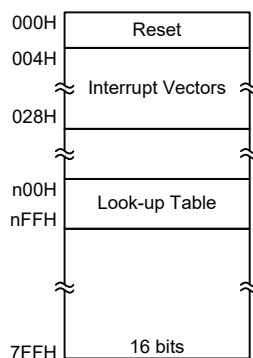
- Arithmetic operations: ADD, ADDM, ADC, ADCM, SUB, SUBM, SBC, SBCM, DAA
- Logic operations: AND, OR, XOR, ANDM, ORM, XORM, CPL, CPLA
- Rotation: RRA, RR, RRCA, RRC, RLA, RL, RLCA, RLC
- Increment and Decrement: INCA, INC, DECA, DEC
- Branch decision: JMP, SZ, SZA, SNZ, SIZ, SDZ, SIZA, SDZA, CALL, RET, RETI

Flash Program Memory

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

Structure

The Program Memory has a capacity of 2K×16 bits. The Program Memory is addressed by the Program Counter and also contains data, table information and interrupt entries. Table data, which can be setup in any location within the Program Memory, is addressed by a separate table pointer register.



Program Memory Structure

Special Vectors

Within the Program Memory, certain locations are reserved for the reset and interrupts. The location 000H is reserved for use by these devices 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 registers, TBLP and TBHP. These registers define the total address of the look-up table.

After setting up the table pointer, the table data can be retrieved from the Program Memory using the “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.

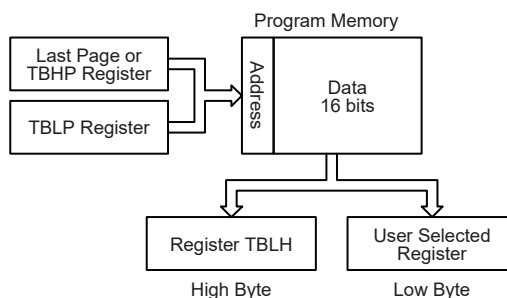


Table Program Example

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 “700H” which refers to the start address of the last page within the 2K words Program Memory of these devices. 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 “706H” or 6 locations after the start of the last page. Note that the value for the table pointer is referenced to the specific address pointed by the TBLP and TBHP 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 the page that tbhp pointed
mov a,07h          ; initialise high table pointer
mov tbhp,a
:
:
tabrd tempreg1      ; transfers value in table referenced by table pointer,
                   ; data at program memory address "706H" 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 "705H" 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 700h          ; sets initial address of program memory
dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh

```

In Circuit Programming – ICP

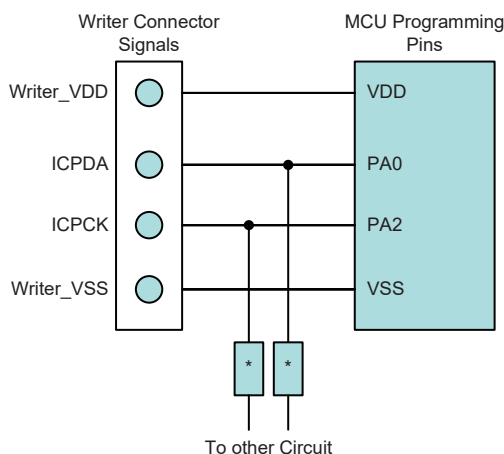
The provision of Flash type Program Memory provides the user with a means of convenient and easy upgrades and modifications to their programs on the same device.

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

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

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

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



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

On-Chip Debug Support – OCDS

There is an EV chip named BS45V583x which is used to emulate the BS45F583x device. The EV chip device also provides an “On-Chip Debug” function to debug the real MCU device during the development process. The EV chip and the real MCU device are almost functionally compatible except for “On-Chip Debug” function. Users can use the EV chip device to emulate the real chip device behavior by connecting the OCDSDA and OCDSCCK pins to the Holtek HT-IDE development tools. The OCDSDA pin is the OCDS Data/Address input/output pin while the OCDSCCK pin is the OCDS clock input pin. When users use the EV chip for debugging, other functions which are shared with the OCDSDA and OCDSCCK pins in the device will have no effect in the EV chip. However, the two OCDS pins which are pin-shared with the ICP programming pins are still used as the Flash Memory programming pins for ICP. For more detailed OCDS information, refer to the corresponding document named “Holtek e-Link for 8-bit MCU OCDS User’s Guide”.

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

Data Memory

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

Structure

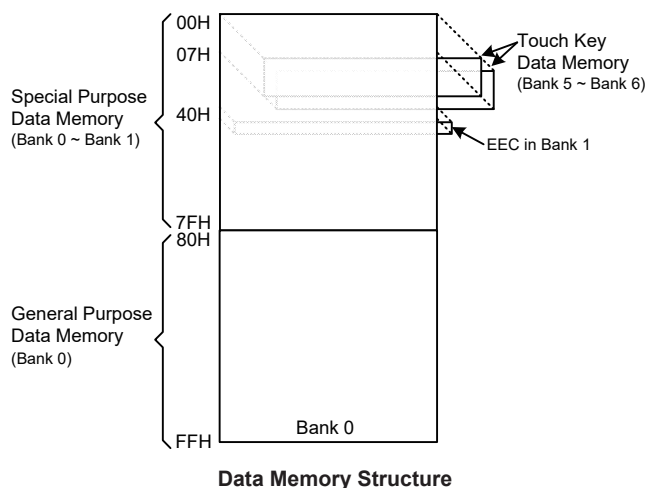
Categorised into two types, the first of these is an area of RAM, known as the Special Function Data Memory. These registers have fixed locations and are necessary for correct operation of these devices. Many of these registers can be read from and written to directly under program control, however, 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. There is another area of Data Memory reserved for the Touch Key Data Memory.

The overall Data Memory is subdivided into several banks, which are implemented in 8-bit wide Memory. The Special Purpose Data Memory registers are accessible in Bank 0, with the exception of the EEC register at address 40H, which is only accessible in Bank 1. Switching between the different Data Memory banks is achieved by properly setting the Bank Pointer to the correct value.

The address range of the Special Purpose Data Memory for these devices is from 00H to 7FH while the address range of the General Purpose Data Memory is from 80H to FFH. The Touch Key Data Memory is located from 00H to 07H in Bank 5 and Bank 6. The start address of the Data Memory for these devices is the address 00H.

Special Purpose Data Memory	General Purpose Data Memory		Touch Key Data Memory	
Located Bank	Capacity	Bank: Address	Capacity	Bank: Address
0, 1	128×8	0: 80H~FFH	16×8	5: 00H~07H 6: 00H~07H

Data Memory Summary



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 programming 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	Bank 1
00H	IAR0	40H	EEC
01H	MP0	41H	EED
02H	IAR1	42H	EEA
03H	MP1	43H	
04H	BP	44H	IICC0
05H	ACC	45H	IICC1
06H	PCL	46H	IICD
07H	TBLP	47H	IICA
08H	TBLH	48H	IICTOC
09H	TBHP	49H	
0AH	STATUS	4AH	CTMC0
0BH		4BH	CTMC1
0CH		4CH	CTMDL
0DH		4DH	CTMDH
0EH		4EH	CTMAL
0FH	RSTFC	4FH	CTMAH
10H	PB	50H	INTC0
11H	PBC	51H	INTC1
12H	PBPU	52H	INTC2
13H	WDTC	53H	MF10
14H	PA	54H	MF11
15H	PAC	55H	MF12
16H	PAPU	56H	
17H	PAWU	57H	
18H	LVRC	58H	
19H	LVDC	59H	INTEG
1AH		5AH	PAS0
1BH		5BH	PAS1
1CH		5CH	PBS0
1DH	TB0C	5DH	PBS1
1EH	TB1C	5EH	
1FH	PSC0R	5FH	
20H	STMC0	60H	
21H	STMC1	61H	
22H	STMDL	62H	
23H	STMDH	63H	
24H	STMAL	64H	
25H	STMAH	65H	
26H		66H	
27H		67H	
28H		68H	
29H		69H	PSC1R
2AH		6AH	TKTMR
2BH	RSTC	6BH	TKC0
2CH	REGC	6CH	TKC1
2DH	LCS	6DH	TKC2
2EH	LVPUC	6EH	TK16DL
2FH	SCC	6FH	TK16DH
30H	HIRCC	70H	TKM0C0
31H		71H	TKM0C1
32H		72H	TKM0C2
33H		73H	TKM016DL
34H		74H	TKM016DH
35H		75H	TKM0ROL
36H	SADOL	76H	TKM0ROH
37H	SADOH	77H	TK1M0TH16L
38H	SADC0	78H	TK1M0TH16H
39H	SADC1	79H	TK2M0TH16L
3AH	SADC2	7AH	TK2M0TH16H
3BH		7BH	TK3M0TH16L
3CH		7CH	TK3M0TH16H
3DH		7DH	TK4M0TH16L
3EH	SLEDC	7EH	TK4M0TH16H
3FH	PMPS	7FH	TKM0THS

 : Unused, read as 00H

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 Registers – IAR0, IAR1

The Indirect Addressing Registers, IAR0 and IAR1, although having their locations in normal RAM register space, do not actually physically exist as normal registers. The method of indirect addressing for RAM data manipulation uses these Indirect Addressing Registers and Memory Pointers, in contrast to direct memory addressing, where the actual memory address is specified. Actions on the IAR0 and IAR1 registers will result in no actual read or write operation to these registers but rather to the memory location specified by their corresponding Memory Pointers, MP0 or MP1. Acting as a pair, IAR0 and MP0 can together access data from Bank 0 while the IAR1 register together with the MP1 register can access data from any Data Memory Bank. As the Indirect Addressing Registers are not physically implemented, reading the Indirect Addressing Registers will return a result of “00H” and writing to the registers will result in no operation.

Memory Pointers – MP0, MP1

Two Memory Pointers, known as MP0 and MP1 are provided. These Memory Pointers are physically implemented in the Data Memory and can be manipulated in the same way as normal registers providing a convenient way with which to address and track data. When any operation to the relevant Indirect Addressing Registers is carried out, the actual address that the microcontroller is directed to is the address specified by the related Memory Pointer. MP0, together with Indirect Addressing Register, IAR0, are used to access data from Bank 0, while MP1 together with IAR1 are used to access data from all data banks according to the BP register. Direct Addressing can only be used with Bank 0, Bank 1 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             ; increment 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

For these devices, 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 SLEEP or IDLE 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	—	—	—	—	—	DMBP2	DMBP1	DMBP0
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 Unimplemented, read as “0”

Bit 2~0 **DMBP2~DMBP0**: Data memory banks selection

000: Bank 0

001: Bank 1

010: Unimplemented

011: Unimplemented

100: Unimplemented

101: Bank 5

110: Bank 6

111: Unimplemented

Accumulator – ACC

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

Program Counter Low Register – PCL

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

Look-up Table Registers – TBLP, TBHP, TBLH

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

Status Register – STATUS

This 8-bit register contains the 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.

EEPROM Data Memory

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

EEPROM Data Memory Structure

The EEPROM Data Memory capacity is 32×8 bits for these devices. Unlike the Program Memory and RAM Data Memory, the EEPROM Data Memory is not directly mapped into memory space and is therefore not directly addressable in the same way as the other types of memory. Read and Write operations to the EEPROM are carried out in single byte operations using an address and a data register in Bank 0 and a single control register in Bank 1.

EEPROM Registers

Three registers control the overall operation of the internal EEPROM Data Memory. These are the address register, EEA, the data register, EED and a single control register, EEC. As both the EEA and EED registers are located in Bank 0, they can be directly accessed in the same way as any other Special Function Register. The EEC register however, being located in Bank 1, can only be read from or written to indirectly using the MP1 Memory Pointer and Indirect Addressing Register, IAR1. Because the EEC control register is located at address 40H in Bank 1, the MP1 Memory Pointer must first be set to the value 40H and the Bank Pointer register, BP, set to the value, 01H, before any operations on the EEC register are executed.

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

EEPROM Register List

• **EEA Register**

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

Bit 7~5 Unimplemented, read as “0”

Bit 4~0 **EEA4~EEA0**: Data EEPROM address bit 4 ~ bit 0

• **EED Register**

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

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

• **EEC Register**

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

Bit 7~4 Unimplemented, read as “0”

Bit 3 **WREN**: Data EEPROM write enable

0: Disable

1: Enable

This is the Data EEPROM Write Enable Bit which must be set high before Data EEPROM write operations are carried out. Clearing this bit to zero will inhibit Data EEPROM write operations.

Bit 2 **WR**: EEPROM write control

0: Write cycle has finished

1: Activate a write cycle

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

Bit 1 **RDEN**: Data EEPROM read enable

0: Disable

1: Enable

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

Bit 0 **RD**: EEPROM read control

0: Read cycle has finished

1: Activate a read cycle

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

Note: 1. The WREN, WR, RDEN and RD cannot be set high at the same time in one instruction.
The WR and RD cannot be set high at the same time.

2. Ensure that the f_{SUB} clock is stable before executing the write operation.

3. Ensure that the write operation is totally complete before changing the EEC register content.

Reading Data from the EEPROM

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

Writing Data to the EEPROM

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

Write Protection

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

EEPROM Interrupt

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

Programming Considerations

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

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

Programming Examples

Reading data from the EEPROM – polling method

```
MOV A, EEPROM_ADRES      ; user defined address
MOV EEA, A
MOV A, 40H                ; set memory pointer MP1
MOV MP1, A                ; MP1 points to EEC register
MOV A, 01H                ; set bank pointer BP
MOV BP, A
SET IAR1.1                ; set RDEN bit, enable read operations
SET IAR1.0                ; start Read Cycle - set RD bit
BACK:
SZ IAR1.0                 ; check for read cycle end
JMP BACK
CLR IAR1                  ; disable EEPROM read if no more read operations are required
CLR BP
MOV A, EED                ; move read data to register
MOV READ_DATA, A
```

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

Writing Data to the EEPROM – polling method

```
MOV A, EEPROM_ADRES      ; user defined address
MOV EEA, A
MOV A, EEPROM_DATA        ; user defined data
MOV EED, A
MOV A, 40H                ; set memory pointer MP1
MOV MP1, A                ; MP1 points to EEC register
MOV A, 01H                ; set bank pointer BP
MOV BP, A
CLR EMI
SET IAR1.3                ; set WREN bit, enable write operations
SET IAR1.2                ; start Write Cycle - set WR bit - executed immediately
                        ; after set WREN bit

SET EMI
BACK:
SZ IAR1.2                 ; check for write cycle end
JMP BACK
CLR BP
```

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 the application program and relevant control registers.

Oscillator Overview

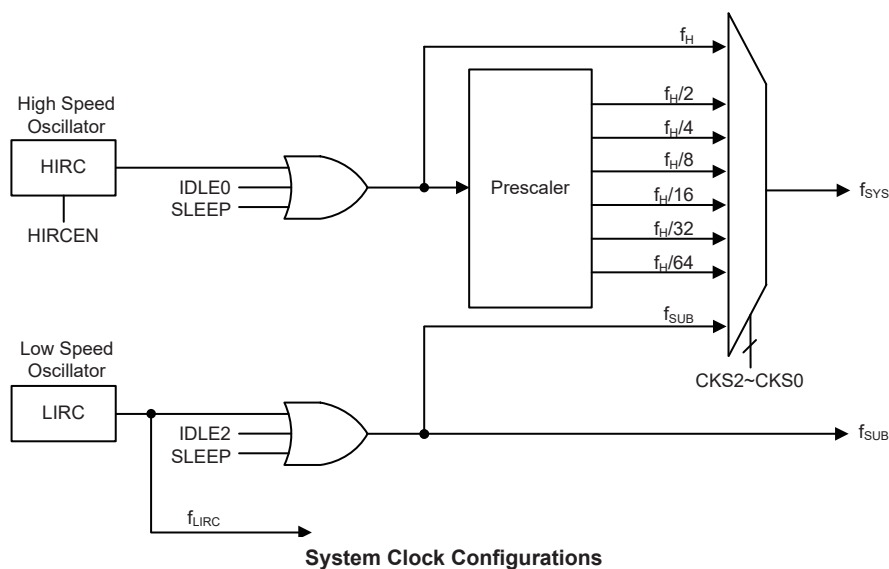
In addition to being the source of the main system clock the oscillators also provide clock sources for the Watchdog Timer and Time Base Interrupts. The fully integrated internal oscillators, requiring no external components, are provided to form a wide range of both fast and slow system oscillators. The higher frequency oscillators provide higher performance but carry with it the disadvantage of higher power requirements, while the opposite is of course true for the lower frequency oscillators. With the capability of dynamically switching between fast and slow system clock, the device 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	4/8/12MHz
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 4/8/12MHz RC oscillator, HIRC. The low speed oscillator is the internal 32kHz RC oscillator, LIRC. Selecting whether the low or high speed oscillator is used as the system oscillator is implemented using the CKS2~CKS0 bits in the SCC register and as the system clock can be dynamically selected.



Internal High Speed RC Oscillator – HIRC

The internal RC oscillator is a fully integrated system oscillator requiring no external components. The internal RC oscillator has three fixed frequencies of 4MHz, 8MHz and 12MHz. 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. Note that this internal system clock option requires no external pins for its operation.

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.

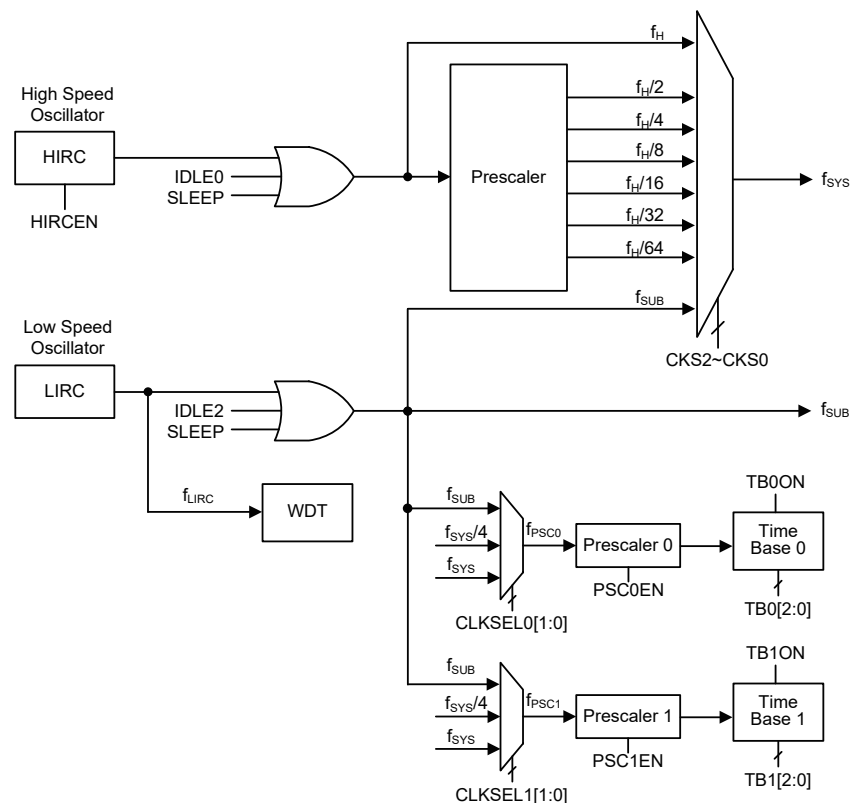
Operating Modes and System Clocks

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 these devices 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

These devices have 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 Mode	On	x	x	000~110	$f_H \sim f_H/64$	On	On	On
SLOW Mode	On	x	x	111	f_{SUB}	On/Off ⁽¹⁾	On	On
IDLE0 Mode	Off	0	1	000~110	Off	Off	On	On
IDLE1 Mode	Off	1	1	xxx	On			
IDLE2 Mode	Off	1	0	000~110	On	On	Off	On
SLEEP Mode	Off	0	0	xxx	Off			

"x": Don't care

Note: 1. The f_H clock will be switched on or off by configuring the corresponding oscillator enable bit in the SLOW mode.

2. The f_{LIRC} clock will be switched on since the WDT function is always enabled 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} . The f_{SUB} clock 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 bit are low. In the SLEEP mode the CPU will be stopped, and the f_{SUB} clock to peripheral will be stopped too. However the f_{LIRC} clock still continues to operate since the WDT function is always 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 Registers

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

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.

• 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: 4MHz

01: 8MHz

10: 12MHz

11: 4MHz

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 to 1.

Bit 1 **HIRCF**: HIRC oscillator stable flag

0: HIRC unstable

1: HIRC stable

This bit is used to indicate whether the HIRC oscillator is stable or not. When the HIRCEN bit is set to 1 to enable the HIRC oscillator, the HIRCF bit will first be cleared to 0 and then set to 1 after the HIRC oscillator is stable.

Bit 0 **HIRCEN**: HIRC oscillator enable control

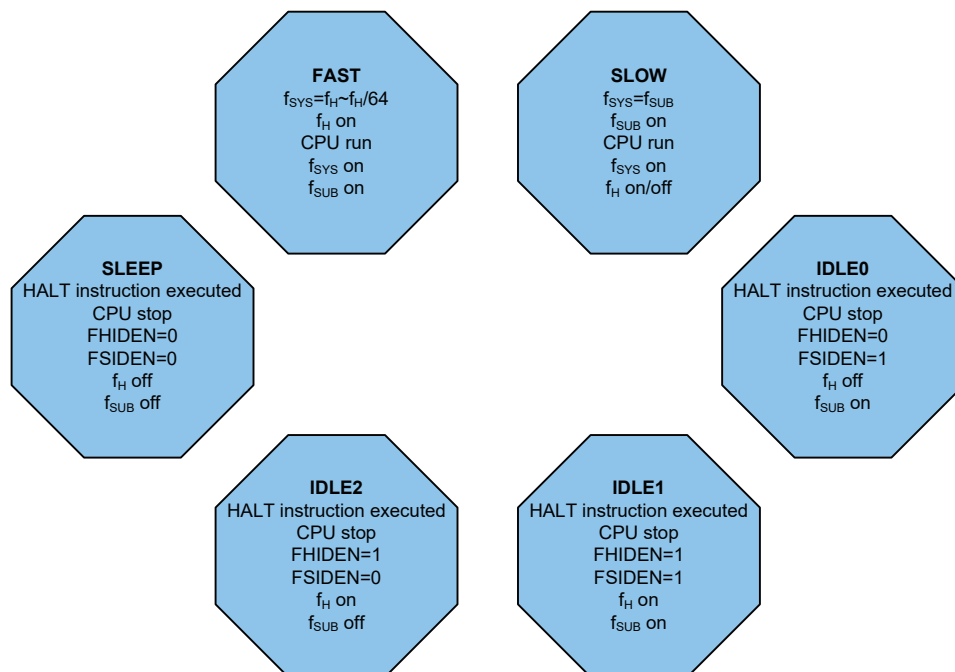
0: Disable

1: Enable

Operating Mode Switching

These devices 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.

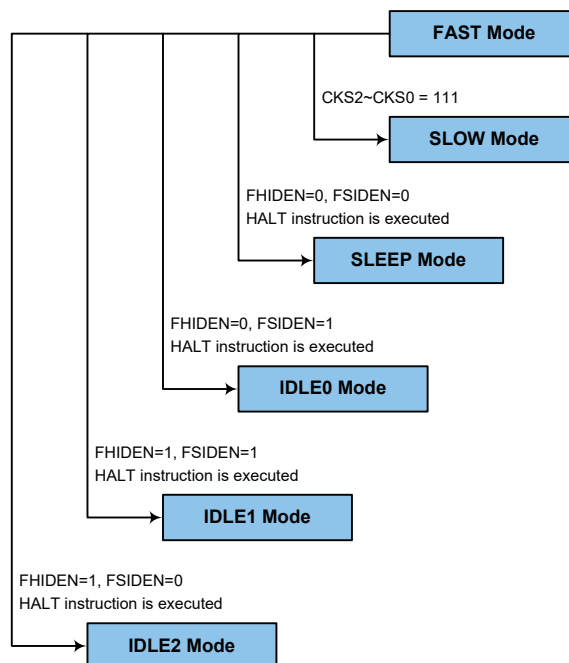
In simple terms, mode switching between the FAST Mode and SLOW Mode is executed using the CKS2~CKS0 bits in the SCC register while mode switching from the FAST/SLOW Modes to the SLEEP/IDLE Modes is executed via the HALT instruction. When an HALT instruction is executed, whether the device enters the 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 set 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.

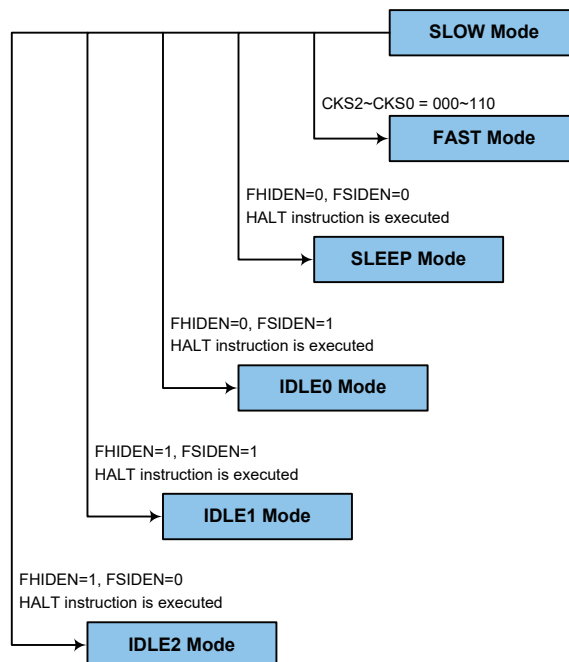
The SLOW Mode system clock is sourced from the LIRC oscillator and therefore requires this oscillator to be stable before full mode switching occurs.



SLOW Mode to FAST Mode Switching

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

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



Entering the SLEEP Mode

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

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

Entering the IDLE0 Mode

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

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

Entering the IDLE1 Mode

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

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

Entering the IDLE2 Mode

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

- The f_H clock will be on but the f_{SUB} clock will be off and the application program will stop at the “HALT” instruction.
- The Data Memory contents and registers will maintain their present condition.
- The I/O ports will maintain their present conditions.

- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.
- The WDT will be cleared and resume counting as the WDT function is always enabled.

Standby Current Considerations

As the main reason for entering the SLEEP or IDLE Mode is to keep the current consumption of the device to as low a value as possible, perhaps only in the order of several micro-amps 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.

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
- A system interrupt
- A WDT overflow

When the device executes the “HALT” instruction, the PDF flag will be set to 1. The PDF flag will be cleared to 0 if the device experiences a system power-up or executes the clear Watchdog Timer instruction. If the system is woken up by a WDT overflow, a Watchdog Timer reset will be initiated and the TO flag will be set to 1. 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 setup using the PAWU register to permit a negative transition on the pin to wake up the system. When a Port A pin wake-up occurs, the program will resume execution at the instruction following the “HALT” instruction. If the system is woken up by an interrupt, then two possible situations may occur. The first is where the related interrupt is disabled or the interrupt is enabled but the stack is full, in which case the program will resume execution at the instruction following the “HALT” instruction. In this situation, the interrupt which woke up the device will not be immediately serviced, but will rather be serviced later when the related interrupt is finally enabled or when a stack level becomes free. The other situation is where the related interrupt is enabled and the stack is not full, in which case the regular interrupt response takes place. If an interrupt request flag is set high before entering the SLEEP or IDLE Mode, the wake-up function of the related interrupt will be disabled.

Watchdog Timer

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

Watchdog Timer Clock Source

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

Watchdog Timer Control Register

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

• WDTC Register

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

Bit 7~3 **WE4~WE0**: WDT function software control

01010/10101: Enabled

Other values: Reset MCU

When these bits are changed to any other values due to environmental noise the microcontroller will be reset; this reset operation will be activated after a delay time, t_{SRESET} , and the WRF bit in the RSTFC register will be set high.

Bit 2~0 **WS2~WS0**: WDT time-out period selection

000: $2^8/f_{LIRC}$

001: $2^{10}/f_{LIRC}$

010: $2^{12}/f_{LIRC}$

011: $2^{14}/f_{LIRC}$

100: $2^{15}/f_{LIRC}$

101: $2^{16}/f_{LIRC}$

110: $2^{17}/f_{LIRC}$

111: $2^{18}/f_{LIRC}$

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

• RSTFC Register

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

"x": unknown

Bit 7~4 Unimplemented, read as "0"

Bit 3 **RSTF**: Reset control register software reset flag

Refer to Internal Reset Control section.

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

Refer to the Low Voltage Reset section.

- Bit 1 **LRF**: LVR control register software reset flag
Refer to the Low Voltage Reset section.
- Bit 0 **WRF**: WDT control register software reset flag
0: Not occurred
1: Occurred
This bit is set high by the WDT Control register software reset and cleared to zero by the application program. Note that this bit can only be cleared to zero by the application program.

Watchdog Timer Operation

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

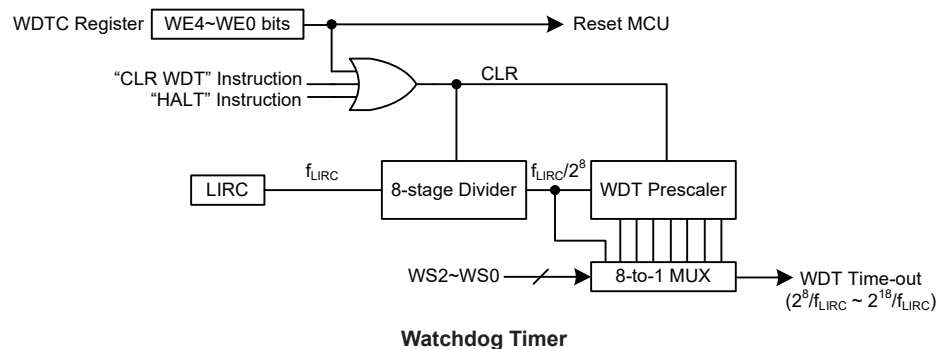
WE4~WE0 Bits	WDT Function
01010B/10101B	Enable
Any other values	Reset MCU

Watchdog Timer Function 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 a WDT software reset, which means a certain value except 01010B and 10101B written into the WE4~WE0 bit filed, the second is using the Watchdog Timer software clear instruction and the third is via a HALT instruction.

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

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



Reset and Initialisation

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

In addition to the power-on reset, another reset exists in the form of a Low Voltage Reset, LVR, where a full reset is implemented in situations where the power supply voltage falls below a certain threshold.

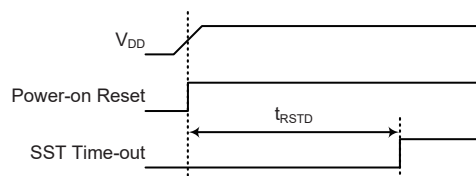
Another type of reset is when the Watchdog Timer overflows and resets the microcontroller. All types of reset operations result in different register conditions being set.

Reset Functions

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

Power-on Reset

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



Power-on Reset Timing Chart

Internal Reset Control

There is an internal reset control register, RSTC, which is used to provide a reset when the device operates abnormally due to the environmental noise interference. If the content of the RSTC register is set to any value other than 01010101B or 10101010B, it will reset the device after a delay time, t_{SRESET} . After power on the register will have a value of 01010101B.

RSTC7~RSTC0 Bits	Reset Function
01010101B	No operation
10101010B	No operation
Any other value	Reset MCU

Internal Reset Function Control

• RSTC Register

Bit	7	6	5	4	3	2	1	0
Name	RSTC7	RSTC6	RSTC5	RSTC4	RSTC3	RSTC2	RSTC1	RSTC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	1	0	1	0	1	0	1

Bit 7~0 **RSTC7~RSTC0**: Reset function control

01010101: No operation

10101010: No operation

Other values: Reset MCU

If these bits are changed due to adverse environmental conditions, the microcontroller will be reset. The reset operation will be activated after a delay time, t_{SRESET} and the RSTF bit in the RSTFC register will be set to 1.

• RSTFC Register

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

"x": unknown

Bit 7~4 Unimplemented, read as "0"

Bit 3 **RSTF**: Reset control register software reset flag

0: Not occurred

1: Occurred

This bit is set high by the RSTC control register software reset and cleared to zero by the application program. Note that this bit can only be cleared to 0 by the application program.

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

Refer to the Low Voltage Reset section.

Bit 1 **LRF**: LVR control register software reset flag

Refer to the Low Voltage Reset section.

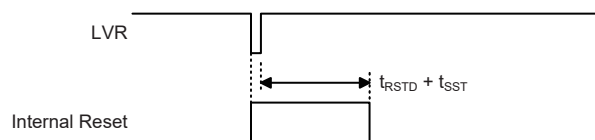
Bit 0 **WRF**: WDT control register software reset flag

Refer to the Watchdog Timer Control Register section.

Low Voltage Reset – LVR

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

The LVR function is always enabled in FAST and SLOW Mode with a specific LVR voltage V_{LVR} . If the supply voltage of the device drops to within a range of $0.9V \sim V_{LVR}$ such as might occur when changing the battery in battery powered applications, the LVR will automatically reset the device internally and the LVRF bit in the RSTFC register will also be set to 1. For a valid LVR signal, a low supply voltage, i.e., a voltage in the range between $0.9V \sim V_{LVR}$ must exist for a time greater than that specified by t_{LVR} in the LVR/LVD Electrical Characteristics. If the low supply voltage state does not exceed this value, the LVR will ignore the low supply voltage and will not perform a reset function. The actual V_{LVR} value can be selected by the LVS bits in the LVRC register. If the LVS7~LVS0 bits are changed to some different values by environmental noise, the LVR will reset the device after a delay time, t_{SRESET} . When this happens, the LRF bit in the RSTFC register will be set to 1. After power on the register will have the value of 01010101B. Note that the LVR function will be automatically disabled when the device enters the SLEEP or IDLE 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	0	1	0	1	0	1

Bit 7~0 **LVS7~LVS0**: LVR Voltage Select control

01010101: 2.1V

00110011: 2.55V

10011001: 3.15V

10101010: 3.8V

Any other value: Generates MCU reset – register is reset to POR value

When an actual low voltage condition occurs, as specified by one of the four 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.

Any register value, other than the four defined LVR values above, will also result in the generation of an MCU reset. The reset operation will be activated after a delay time, t_{SRESET} . However in this situation the register contents will be reset to the POR value.

• **RSTFC Register**

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

“x”: unknown

Bit 7~4 Unimplemented, read as “0”

Bit 3 **RSTF**: Reset control register software reset flag

Refer to the Internal Reset Control section.

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

0: Not occur

1: Occurred

This bit is set high when a specific Low Voltage Reset situation condition occurs. This bit can only be cleared to zero by the application program.

Bit 1 **LRF**: LVR control register software reset flag

0: Not occur

1: Occurred

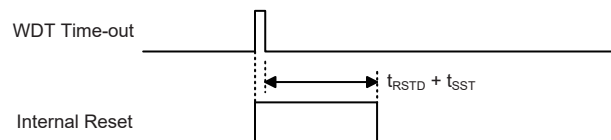
This bit is set high if the LVRC register contains any non-defined LVR voltage register values. This in effect acts like a software-reset function. This bit can only be cleared to zero by the application program.

Bit 0 **WRF**: WDT control register software reset flag

Refer to the Watchdog Timer Control Register section.

Watchdog Time-out Reset during Normal Operation

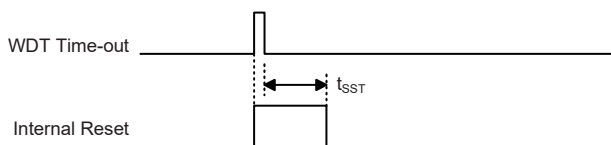
The Watchdog time-out Reset during normal operations in the FAST or SLOW mode is the same as a LVR reset except that the Watchdog time-out flag TO will be set to “1”.



WDT Time-out Reset during Normal Operation Timing Chart

Watchdog Time-out Reset during SLEEP or IDLE Mode

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



WDT Time-out Reset during SLEEP or IDLE Timing Chart

Reset Initial Conditions

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

TO	PDF	RESET Conditions
0	0	Power-on reset
u	u	LVR reset during FAST or SLOW Mode operation
1	u	WDT time-out reset during FAST or SLOW Mode operation
1	1	WDT time-out reset during IDLE or SLEEP Mode 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 Bases	Clear after reset, WDT begins counting
Timer Modules	Timer Modules will be turned off
Input/Output Ports	I/O ports will be setup as inputs
Stack Pointer	Stack Pointer will point to the top of the stack

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

Register	Power On Reset	LVR Reset (Normal Operation)	WDT Time-out (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	- - - - - 0 0 0	- - - - - u 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	- - - - - u u u	- - - - - u u u	- - - - - u u u
STATUS	- - 0 0 x x x x	- - u u u u u u	- - 1 u u u u u	- - 1 1 u u u u
RSTFC	- - - - - 0 x 0 0	- - - - - u 1 u u	- - - - - 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
WDTC	0 1 0 1 0 0 1 1	0 1 0 1 0 0 1 1	0 1 0 1 0 0 1 1	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 1 1 1 1 1 1	u u 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 1 1 1 1 1 1	u u 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 0 0 0 0 0 0	u u 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 0 0 0 0 0 0	u u u u u u u u
LVRC	0 1 0 1 0 1 0 1	u u u u u u u u	0 1 0 1 0 1 0 1	u u u u u u u u
LVDC	- - 0 0 0 0 0 0	- - 0 0 0 0 0 0	- - 0 0 0 0 0 0	- - u u u u u u
TB0C	0 - - - - 0 0 0	0 - - - - 0 0 0	0 - - - - 0 0 0	u - - - - u u u
TB1C	0 - - - - 0 0 0	0 - - - - 0 0 0	0 - - - - 0 0 0	u - - - - u u u
PSC0R	- - - - - 0 0 0	- - - - - 0 0 0	- - - - - 0 0 0	- - - - - u u u
STMC0	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
STMC1	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
STMDL	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
STMDH	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
STMAL	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
STMAH	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
RSTC	0 1 0 1 0 1 0 1	0 1 0 1 0 1 0 1	0 1 0 1 0 1 0 1	u u u u u u u u
REGC	- - - - - - - 1	- - - - - - - 1	- - - - - - - 1	- - - - - - - u
LCS	x - x 0 0 0 0 0	x - x 0 0 0 0 0	x - x 0 0 0 0 0	x - x u u u u u
LVPUC	- - - - - - - 0	- - - - - - - 0	- - - - - - - 0	- - - - - - - 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
SADOL	x x x x - - - -	x x x x - - - -	x x x x - - - -	u u u u - - - - (ADRF5=0)
				u u u u u u u u (ADRF5=1)
SADOH	x x x x x x x x	x x x x x x x x	x x x x x x x x	u u u u u u u u (ADRF5=0)
				- - - - - u u u u (ADRF5=1)
SADC0	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
SADC1	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
SADC2	0 - - 0 0 0 0 0	0 - - 0 0 0 0 0	0 - - 0 0 0 0 0	u - - u u u u u

Register	Power On Reset	LVR Reset (Normal Operation)	WDT Time-out (Normal Operation)	WDT Time-out (IDLE/SLEEP)
SLEDC	---- --00	---- --00	---- --00	---- --uu
PMP5	---- --00	---- --00	---- --00	---- --uu
EEC	---- 0000	---- 0000	---- 0000	---- uuuu
EED	0000 0000	0000 0000	0000 0000	uuuu uuuu
EEA	---0 0000	---0 0000	---0 0000	---u uuuu
IICC0	---- 000-	---- 000-	---- 000-	---- uuu-
IICC1	1000 0001	1000 0001	1000 0001	uuuu uuuu
IICD	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
IICA	0000 000-	0000 000-	0000 000-	uuuu uuu-
IICTOC	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMDH	---- --00	---- --00	---- --00	---- --uu
CTMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMAH	---- --00	---- --00	---- --00	---- --uu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC2	-000 -000	-000 -000	-000 -000	-uuu -uuu
MF10	--00 --00	--00 --00	--00 --00	--uu --uu
MF11	--00 --00	--00 --00	--00 --00	--uu --uu
MF12	--00 --00	--00 --00	--00 --00	--uu --uu
INTEG	---- --00	---- --00	---- --00	---- --uu
PAS0	0000 00--	0000 00--	0000 00--	uuuu uu--
PAS1	--00 0000	--00 0000	--00 0000	--uu uuuu
PBS0	---- 0000	---- 0000	---- 0000	---- uuuu
PBS1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PSC1R	---- -000	---- -000	---- -000	---- -uuu
TKTMR	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKC0	0000 0010	0000 0010	0000 0010	uuuu uuuu
TKC1	0000 0011	0000 0011	0000 0011	uuuu uuuu
TKC2	---- --01	---- --01	---- --01	---- --uu
TK16DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TK16DH	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM0C0	--00 0000	--00 0000	--00 0000	--uu uuuu
TKM0C1	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
TKM0C2	1110 0100	1110 0100	1110 0100	uuuu uuuu
TKM016DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM016DH	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM0ROL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM0ROH	---- --00	---- --00	---- --00	---- --uu
TK1M0TH16L	0000 0000	0000 0000	0000 0000	uuuu uuuu
TK1M0TH16H	0000 0000	0000 0000	0000 0000	uuuu uuuu
TK2M0TH16L	0000 0000	0000 0000	0000 0000	uuuu uuuu
TK2M0TH16H	0000 0000	0000 0000	0000 0000	uuuu uuuu
TK3M0TH16L	0000 0000	0000 0000	0000 0000	uuuu uuuu
TK3M0TH16H	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	Power On Reset	LVR Reset (Normal Operation)	WDT Time-out (Normal Operation)	WDT Time-out (IDLE/SLEEP)
TK4M0TH16L	0000 0000	0000 0000	0000 0000	uuuu uuuu
TK4M0TH16H	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM0THS	0000 0000	0000 0000	0000 0000	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.

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

Register Name	Bit							
	7	6	5	4	3	2	1	0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PB	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

I/O Logic Function Register List

Pull-high Resistors

Many product applications require pull-high resistors for their switch inputs usually requiring the use of an external resistor. To eliminate the need for these external resistors, all I/O pins, when configured as an input have the capability of being connected to an internal pull-high resistor. These pull-high resistors are selected using the PxPU register, and are implemented using weak PMOS transistors. The PxPU register is used to determine whether the pull-high function is enabled or not, while for the PA6 and PB0~PB3 ports, another register LVPUC is used to select the pull-high resistor value for low voltage power supply applications.

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.

Not that the LVPU bit in the LVPUC register is only available when the corresponding pin pull-high function is enabled. If the pull-high function is disabled, the LVPU bit has no effect on selecting the pull-high resistor value.

• PxPU Register

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

PxPUn: I/O Port x Pin pull-high function control

0: Disable

1: Enable

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

• LVPUC Register

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

Bit 7~1 Unimplemented, read as “0”

Bit 0 **LVPU:** Pull-high resistor selection for low voltage power supply

0: All pin pull high resistor is 60kΩ@3V

1: All pin pull high resistor is 15kΩ@3V

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 general purpose input and the MCU enters the IDLE or SLEEP mode.

• PAWU Register

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

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

0: Disable

1: Enable

I/O Port Control Registers

Each I/O port has its own control register known as PAC~PBC, to control the input/output configuration. With this control register, each CMOS output or input can be reconfigured dynamically under software control. Each pin of the I/O ports is directly mapped to a bit in its associated port control register. For the I/O pin to function as an input, the corresponding bit of the control register must be written as a “1”. This will then allow the logic state of the input pin to be directly read by instructions. When the corresponding bit of the control register is written as a “0”, the I/O pin will be setup as a CMOS output. If the pin is currently setup as an output, instructions

can still be used to read the output register. However, it should be noted that the program will in fact only read the status of the output data latch and not the actual logic status of the output pin.

• PxC Register

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

PxCn: I/O Port x Pin type selection

0: Output

1: Input

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

I/O Port Source Current Selection

The source current of PA4~PA7 pins in these devices can be configured with different source current which is selected by the corresponding pin source current select bits. These source current bits are available when the corresponding pin is configured as a CMOS output. Otherwise, these select bits have no effect. Users should refer to the Input/Output Characteristics section to obtain the exact value for different applications.

• SLEDC Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	SLEDC1	SLEDC0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as “0”

Bit 1~0 **SLEDC1~SLEDC0:** PA4~PA7 pins source current selection

00: Level 0 (Min.)

01: Level 1

10: Level 2

11: Level 3 (Max.)

I/O Port Power Source Control

These devices support different I/O port power source selections for PB0~PB3 pins. The port power can come from either the power pin VDD or VDDIO which is determined using the PMPS bit field in the PMPS register. The VDDIO power pin function should first be selected using the corresponding pin-shared function selection bits if the port power is supposed to come from the VDDIO pin. An important point to know is that the input power voltage on the VDDIO pin should be equal to or less than the device supply power voltage when the VDDIO pin is selected as the port power supply pin.

• PMPS Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PMPS1	PMPS0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as “0”

Bit 1~0 **PMPS1~PMPS0**: PB3~PB0 pin power source selection

0x: VDD

1x: VDDIO

Pin-shared Functions

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

Pin-shared Function Selection Registers

The limited number of supplied pins in a package can impose restrictions on the amount of functions a certain device can contain. However by allowing the same pins to share several different functions and providing a means of function selection, a wide range of different functions can be incorporated into even relatively small package sizes. The device includes Port “x” Output Function Selection register “n”, labeled as PxSn, 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 some digital input pins, such as INT, xTCK, STPI, etc, which share the same pin-shared control configuration with their corresponding general purpose I/O functions when setting the relevant pin-shared control bit fields. To select these pin functions, in addition to the necessary pin-shared control and peripheral functional setup aforementioned, they must also be setup 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 Name	Bit							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	—	—
PAS1	—	—	PAS15	PAS14	PAS13	PAS12	PAS11	PAS10
PBS0	—	—	—	—	PBS03	PBS02	PBS01	PBS00
PBS1	PBS17	PBS16	PBS15	PBS14	PBS13	PBS12	PBS11	PBS10

Pin-shared Function Selection Register List

• **PAS0 Register**

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

Bit 7~6 **PAS07~PAS06:** PA3 Pin-Shared function selection

00: PA3/CTCK
01: PA3/CTCK
10: CTP
11: AN3

Bit 5~4 **PAS05~PAS04:** PA2 Pin-Shared function selection

00: PA2
01: PA2
10: VREFI
11: PA2

Bit 3~2 **PAS03~PAS02:** PA1 Pin-Shared function selection

00: PA1
01: PA1
10: VREF
11: AN1

Bit 1~0 Unimplemented, read as “0”

• **PAS1 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PAS15	PAS14	PAS13	PAS12	PAS11	PAS10
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 **PAS15~PAS14:** PA6 Pin-Shared function selection

00: PA6/STCK
01: PA6/STCK
10: PA6/STCK
11: VDDIO

Bit 3~2 **PAS13~PAS12:** PA5 Pin-Shared function selection

00: PA5
01: PA5
10: PA5
11: AN0

Bit 1~0 **PAS11~PAS10:** PA4 Pin-Shared function selection

00: PA4
01: PA4
10: STP
11: AN2

• **PBS0 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	PBS03	PBS02	PBS01	PBS00
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 **PBS03~PBS02**: PB1 Pin-Shared function selection

00: PB1

01: SDA

10: PB1

11: PB1

Bit 1~0 **PBS01~PBS00**: PB0 Pin-Shared function selection

00: PB0

01: SCL

10: PB0

11: PB0

• **PBS1 Register**

Bit	7	6	5	4	3	2	1	0
Name	PBS17	PBS16	PBS15	PBS14	PBS13	PBS12	PBS11	PBS10
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 **PBS17~PBS16**: PB7 Pin-Shared function selection

00: PB7

01: PB7

10: PB7

11: KEY4

Bit 5~4 **PBS15~PBS14**: PB6 Pin-Shared function selection

00: PB6

01: PB6

10: PB6

11: KEY3

Bit 3~2 **PBS13~PBS12**: PB5 Pin-Shared function selection

00: PB5

01: PB5

10: PB5

11: KEY2

Bit 1~0 **PBS11~PBS10**: PB4 Pin-Shared function selection

00: PB4

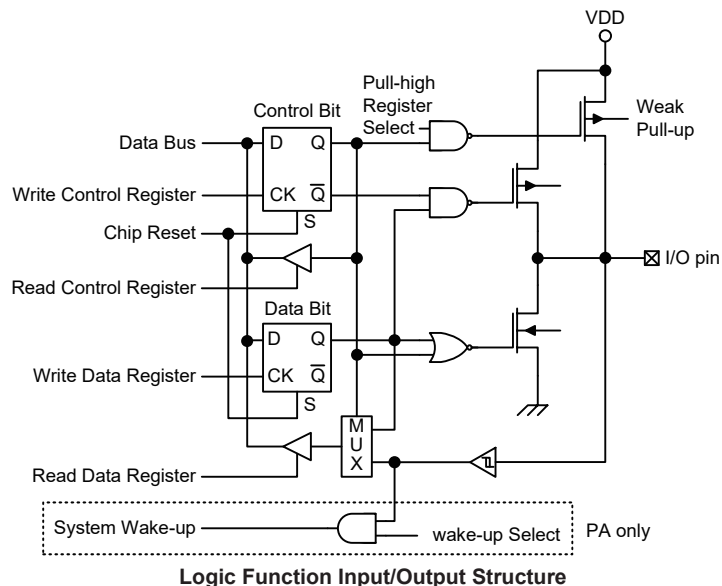
01: PB4

10: PB4

11: KEY1

I/O Pin Structures

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.



Programming Considerations

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

Port A has the additional capability of providing wake-up 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 setup 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 each device includes several Timer Modules, abbreviated to the name TM. The TMs are multi-purpose timing units and serve to provide operations such as Timer/Counter, Input Capture, Compare Match Output and Single Pulse Output as well as being the functional unit for the generation of PWM signals. Each of the TMs has two individual interrupts. The addition of input and output pins for each TM ensures that users are provided with timing units with a wide and flexible range of features.

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

Introduction

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

Function	CTM	STM
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

STM	CTM
10-bit STM	10-bit CTM

TM Name/Type Reference

TM Operation

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

TM Clock Source

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

TM Interrupts

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

TM External Pins

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

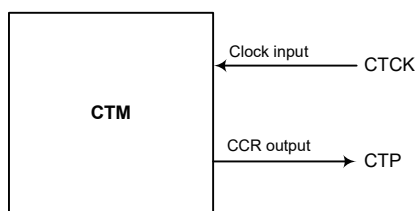
The Standard type TM has another input pin, STPI, which is the capture input whose active edge can be a rising edge, a falling edge or both rising and falling edges and the active edge transition type is selected using the STIO1~STIO0 bits in the STMC1 register.

The TMs each has an output pin, xTP. The TM output pin can be selected using the corresponding pin-shared function selection bits described in the Pin-shared Function section. When the TM is in the Compare Match Output Mode, these pins can be controlled by the TM to switch to a high or low level or to toggle when a compare match situation occurs. The external xTP output pin is also the pin where the TM generates the PWM output waveform.

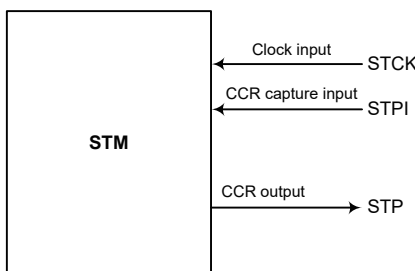
As the TM input and output pins are pin-shared with other functions, the TM input and output functions must first be setup using the relevant pin-shared function selection bits described in the Pin-shared Function section. The details of the pin-shared function selection are described in the pin-shared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

CTM		STM	
Input	Output	Input	Output
CTCK	CTP	STCK, STPI	STP

TM External Pins



CTM Function Pin Block Diagram

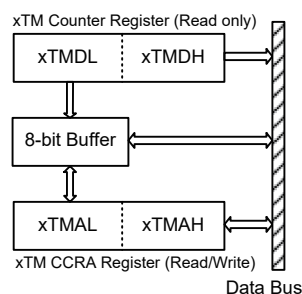


STM Function Pin Block Diagram

Programming Considerations

The TM Counter Registers and the Capture/Compare CCRA registers, 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 register is 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 low byte registers, named xTMAL, using the following access procedures. Accessing the CCRA low byte registers without following these access procedures will result in unpredictable values.

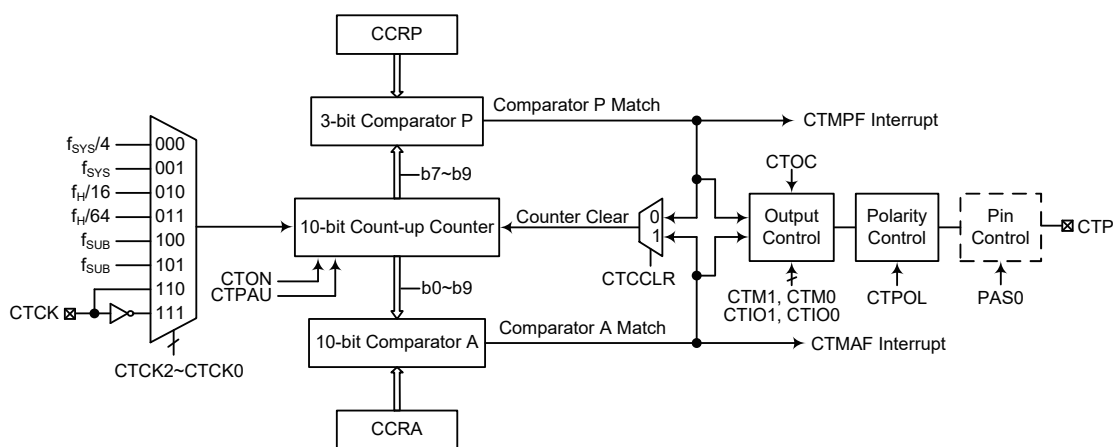


The following steps show the read and write procedures:

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

Compact Type TM – CTM

Although a simple TM type, the Compact TM type still contains three operating modes, which are Compare Match Output, Timer/Event Counter and PWM Output modes. The Compact TM can also be controlled with an external input pin and can drive an external output pin.



Compact Type TM Block Diagram

Compact TM Operation

At 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 is three bits wide whose value is compared with the highest three bits in the counter while the CCRA is the ten bits and therefore compares 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 CTON 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 CTM 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 including an input pin and can also control a output pin. All operating setup conditions are selected using relevant internal registers.

Compact Type TM Register Description

Overall operation of each Compact TM is controlled using several 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 three CCRP bits.

Register Name	Bit							
	7	6	5	4	3	2	1	0
CTMC0	CTPAU	CTCK2	CTCK1	CTCK0	CTON	CTRP2	CTRP1	CTRP0
CTMC1	CTM1	CTM0	CTIO1	CTIO0	CTOC	CTPOL	CTDPX	CTCCLR
CTMDL	D7	D6	D5	D4	D3	D2	D1	D0
CTMDH	—	—	—	—	—	—	D9	D8
CTMAL	D7	D6	D5	D4	D3	D2	D1	D0
CTMAH	—	—	—	—	—	—	D9	D8

10-bit Compact TM Register List

• CTMC0 Register

Bit	7	6	5	4	3	2	1	0
Name	CTPAU	CTCK2	CTCK1	CTCK0	CTON	CTRP2	CTRP1	CTRP0
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 **CTPAU**: CTM 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 CTM 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 **CTCK2~CTCK0**: Select CTM counter clock

000: $f_{SYS}/4$

001: f_{SYS}

010: $f_H/16$

011: $f_H/64$

100: f_{SUB}

101: f_{SUB}

110: CTCK rising edge clock

111: CTCK falling edge clock

These three bits are used to select the clock source for the CTM. The external pin clock source can be chosen to be active on the rising or falling edge. The clock source f_{SYS} is the system clock, while f_H and f_{SUB} are other internal clocks, the details of which can be found in the oscillator section.

Bit 3 **CTON**: CTM Counter on/off control

0: Off

1: On

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

Bit 2~0 **CTRP2~CTRP0**: CTM CCRP 3-bit register, compared with the CTM counter bit 9~bit 7

Comparator P Match Period:

000: 1024 CTM clocks

001: 128 CTM clocks

010: 256 CTM clocks

011: 384 CTM clocks

100: 512 CTM clocks

101: 640 CTM clocks

110: 768 CTM clocks

111: 896 CTM 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 CTCCLR bit is set to zero. Setting the CTCCLR 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.

• **CTMC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	CTM1	CTM0	CTIO1	CTIO0	CTOC	CTPOL	CTDPX	CTCCLR
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 **CTM1~CTM0**: Select CTM operating mode

- 00: Compare Match Output Mode
- 01: Undefined
- 10: PWM Output Mode
- 11: Timer/Counter Mode

These bits setup the required operating mode for the CTM. To ensure reliable operation the CTM should be switched off before any changes are made to the CTM1 and CTM0 bits. In the Timer/Counter Mode, the CTM output pin state is undefined.

Bit 5~4 **CTIO1~CTIO0**: Select CTM external pin CTP function

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 CTM output pin changes state when a certain condition is reached. The function that these bits select depends upon in which mode the CTM is running.

In the Compare Match Output Mode, the CTIO1~CTIO0 bits determine how the CTM output pin changes state when a compare match occurs from the Comparator A. The CTM 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 CTIO1~CTIO0 bits are both zero, then no change will take place on the output. The initial value of the CTM output pin should be setup using the CTOC bit. Note that the output level requested by the CTIO1~CTIO0 bits must be different from the initial value setup using the CTOC bit otherwise no change will occur on the CTM output pin when a compare match occurs. After the CTM output pin changes state it can be reset to its initial level by changing the level of the CTON bit from low to high.

In the PWM Output Mode, the CTIO1 and CTIO0 bits determine how the CTM 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 change the values of the CTIO1 and CTIO0 bits only after the CTM has been switched off. Unpredictable PWM outputs will occur if the CTIO1 and CTIO0 bits are changed when the CTM is running.

Bit 3 **CTOC**: CTM CTP output control bit

Compare Match Output Mode

- 0: Initial low
- 1: Initial high

PWM Output Mode

0: Active low

1: Active high

This is the output control bit for the CTM output pin. Its operation depends upon whether CTM is being used in the Compare Match Output Mode or in the PWM Output Mode. It has no effect if the CTM is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the CTM 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 **CTPOL**: CTM CTP output polarity control

0: Non-invert

1: Invert

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

Bit 1 **CTDPX**: CTM PWM period/duty 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 **CTCCLR**: CTM counter clear condition selection

0: CTM Comparatror P match

1: CTM Comparatror A match

This bit is used to select the method which clears the counter. Remember that the Compact TM contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the CTCCLR 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 CTCCLR bit is not used in the PWM Output Mode.

• CTMDL 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**: CTM counter low byte register bit 7~bit 0

CTM 10-bit Counter bit 7~bit 0

• CTMDH 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**: CTM counter high byte register bit 1~bit 0

CTM 10-bit Counter bit 9~bit 8

• **CTMAL 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:** CTM CCRA low byte register bit 7~bit 0
 CTM 10-bit CCRA bit 7~bit 0

• **CTMAH 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:** CTM CCRA high byte register bit 1~bit 0
 CTM 10-bit CCRA bit 9~bit 8

Compact Type TM Operating 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 CTM1 and CTM0 bits in the CTMC1 register.

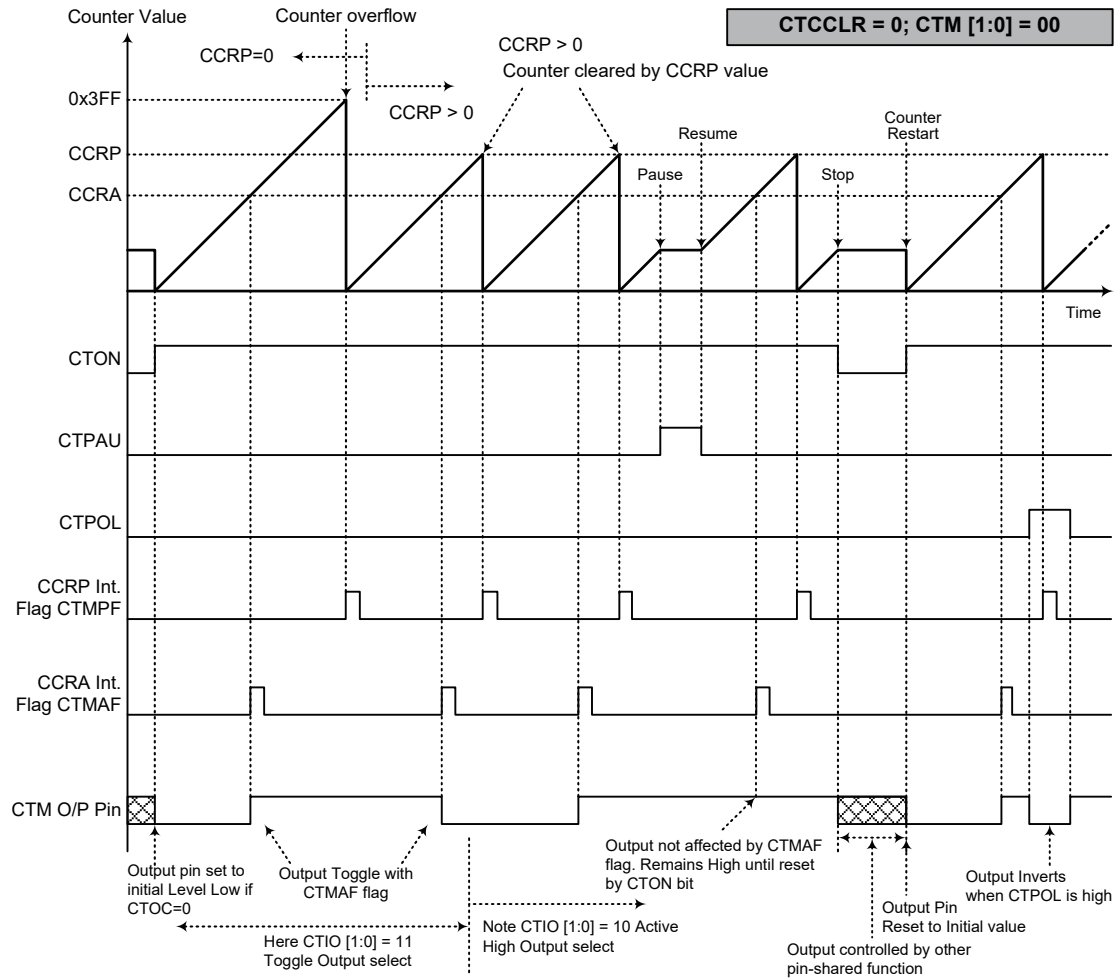
Compare Match Output Mode

To select this mode, bits CTM1 and CTM0 in the CTMC1 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 CTCCLR 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 CTMAF and CTMPF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

If the CTCCLR bit in the CTMC1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the CTMAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when CTCCLR is high no CTMPF interrupt request flag will be generated.

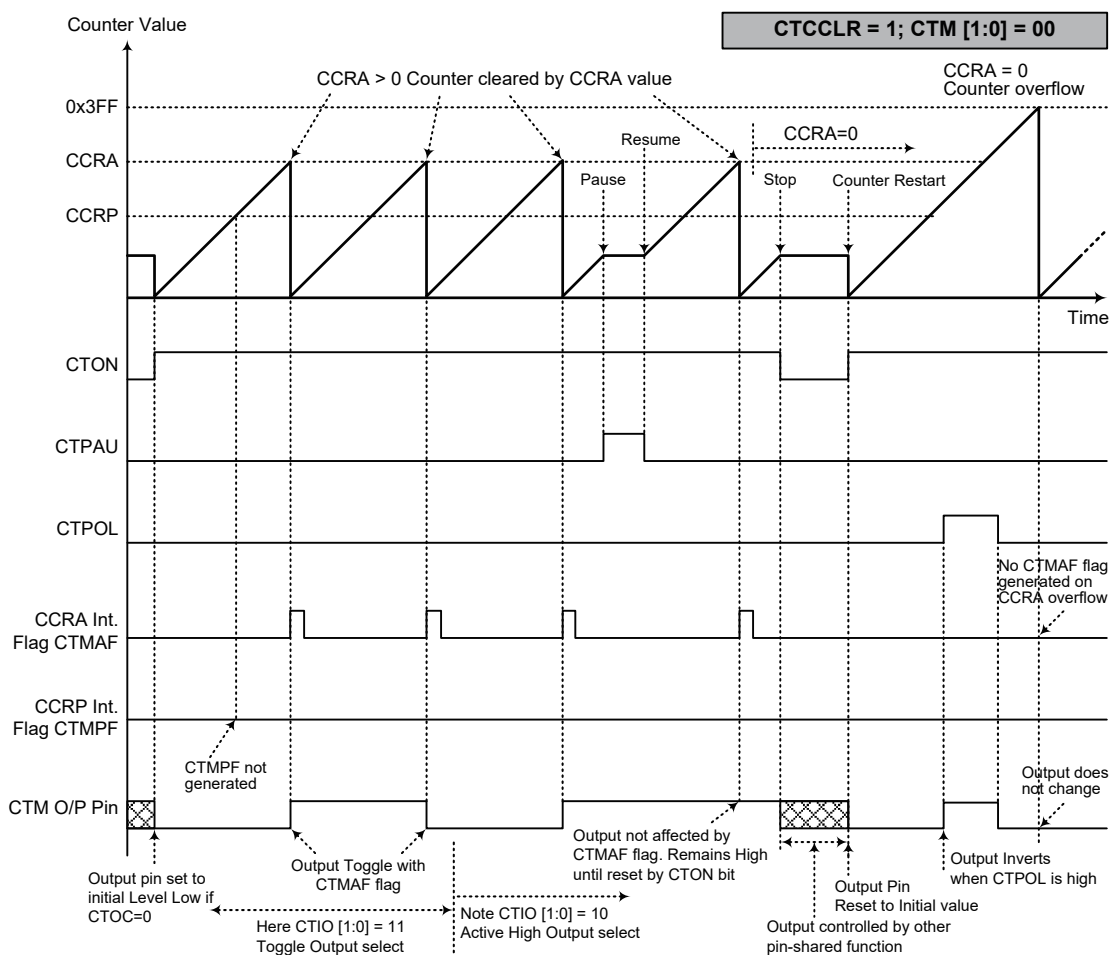
If the CCRA bits are all zero, the counter will overflow when it reaches its maximum 10-bit, 3FF Hex, value, however here the CTMAF interrupt request flag will not be generated.

As the name of the mode suggests, after a comparison is made, the CTM output pin, will change state. The CTM output pin condition however only changes state when a CTMAF interrupt request flag is generated after a compare match occurs from Comparator A. The CTMPF interrupt request flag, generated from a compare match occurs from Comparator P, will have no effect on the CTM output pin. The way in which the CTM output pin changes state are determined by the condition of the CTIO1 and CTIO0 bits in the CTMC1 register. The CTM output pin can be selected using the CTIO1 and CTIO0 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 CTM output pin, which is setup after the CTON bit changes from low to high, is setup using the CTOC bit. Note that if the CTIO1 and CTIO0 bits are zero then no pin change will take place.



Compare Match Output Mode – CTCCLR=0

- Note:
1. With CTCCLR=0, a Comparator P match will clear the counter
 2. The CTM output pin controlled only by the CTMAF flag
 3. The output pin reset to initial state by a CTON bit rising edge



Compare Match Output Mode – CTCCLR=1

- Note: 1. With CTCCLR=1, a Comparator A match will clear the counter
 2. The CTM output pin controlled only by the CTMAF flag
 3. The output pin reset to initial state by a CTON rising edge
 4. The CTMPF flags is not generated when CTCCLR=1

Timer/Counter Mode

To select this mode, bits CTM1 and CTM0 in the CTMC1 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 CTM 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 CTM 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 CTM1 and CTM0 in the CTMC1 register should be set to 10 respectively. The PWM function within the CTM 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 CTM 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 CTCCLR 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 CTD PX bit in the CTMC1 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 CTC bit in the CTMC1 register is used to select the required polarity of the PWM waveform while the two CTIO1 and CTIO0 bits are used to enable the PWM output or to force the CTM output pin to a fixed high or low level. The CTPOL bit is used to reverse the polarity of the PWM output waveform.

• 10-bit CTM, PWM output Mode, Edge-aligned Mode, CTD PX=0

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

If $f_{SYS}=8\text{MHz}$, CTM clock source is $f_{SYS}/4$, CCRP=2 and CCRA=128,

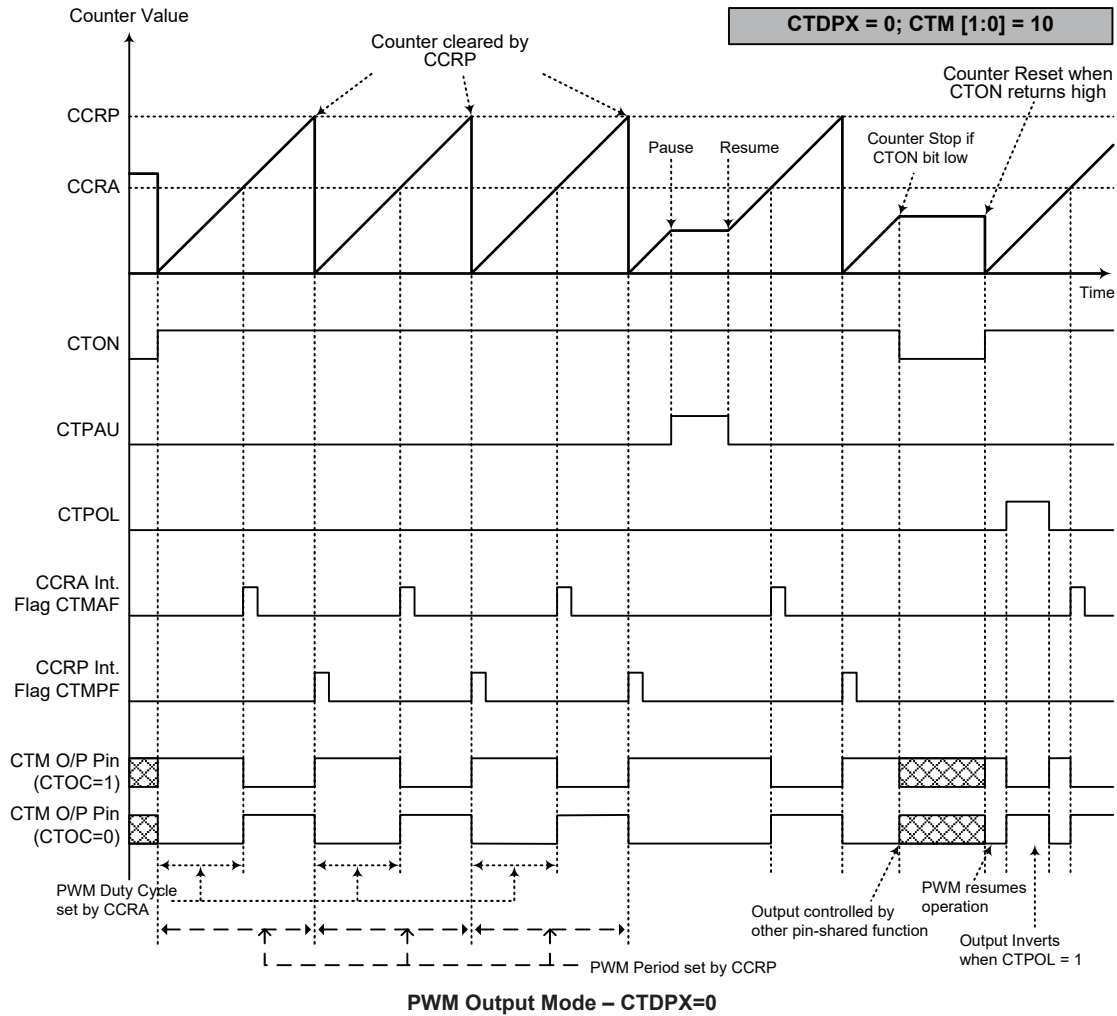
The CTM PWM output frequency= $(f_{SYS}/4)/(2\times 128)=f_{SYS}/1024=7.812\text{kHz}$, duty= $128/(2\times 128)=50\%$.

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%.

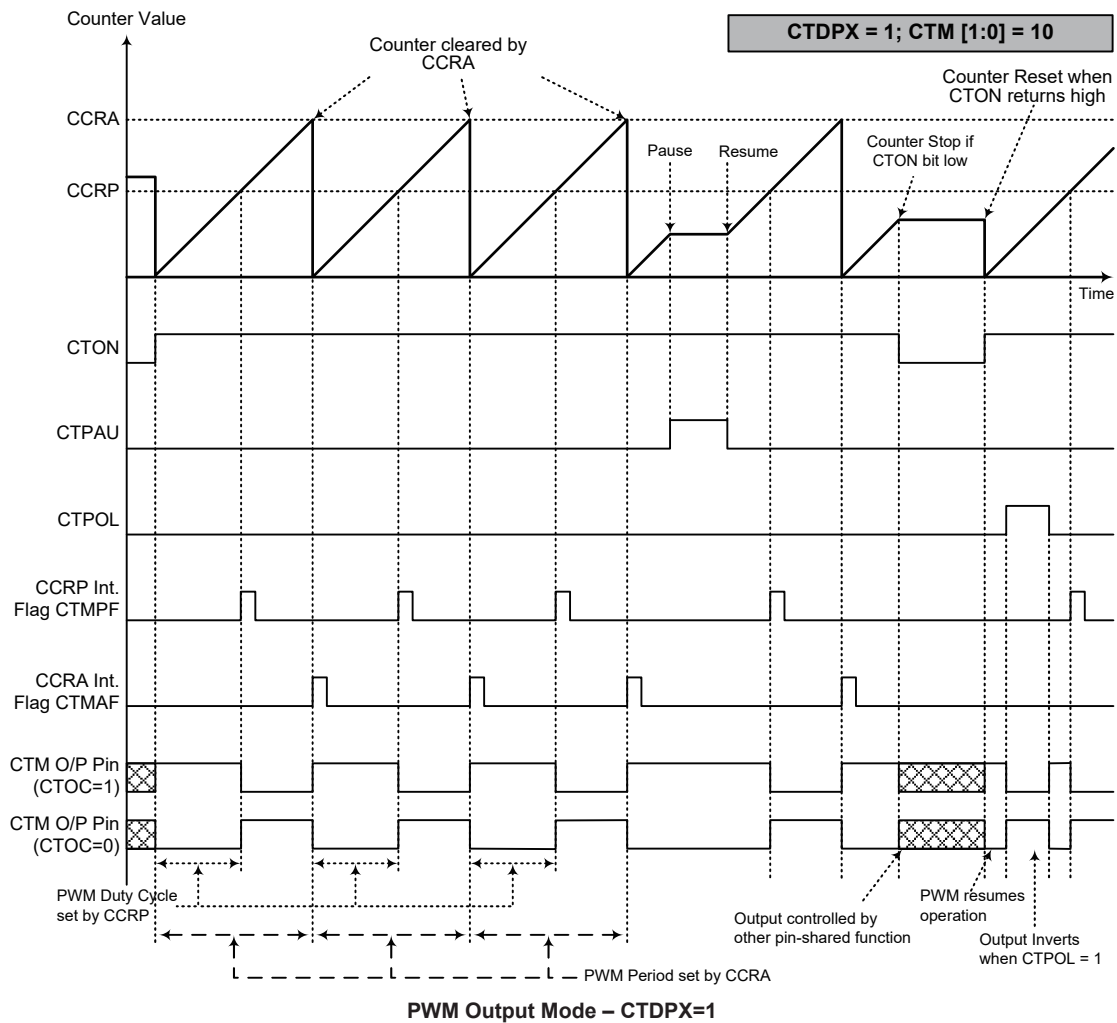
• 10-bit CTM, PWM output Mode, Edge-aligned Mode, CTD PX=1

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

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



- Note: 1. Here CTD PX=0 – Counter cleared by CCRP
 2. A counter clear sets PWM Period
 3. The internal PWM function continues running even when CTIO[1:0]=00 or 01
 4. The CTCCLR bit has no influence on PWM operation



Note: 1. Here CTD PX=1 – Counter cleared by CCRA

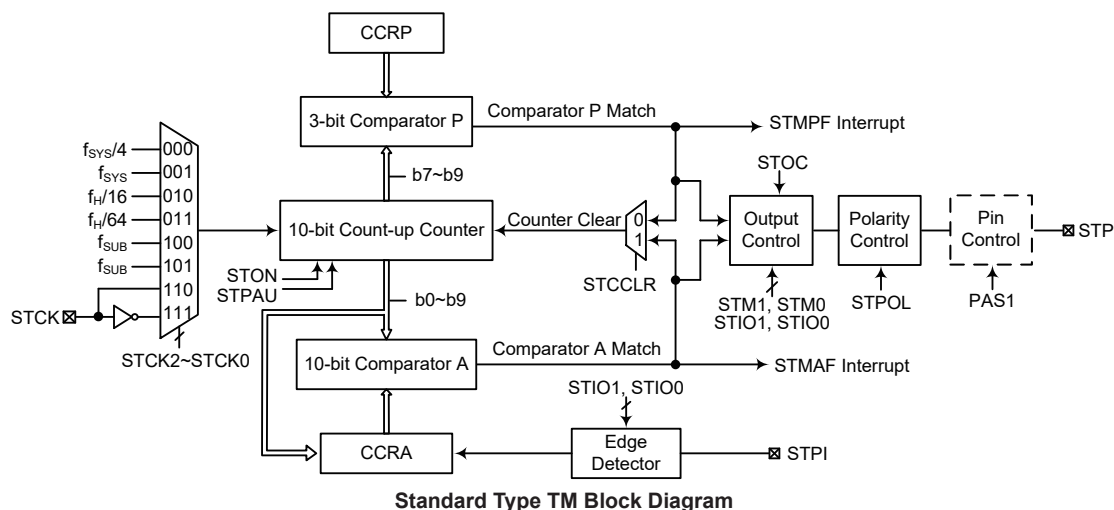
2. A counter clear sets PWM Period

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

4. The CTCCLR bit has no influence on PWM operation

Standard Type TM – STM

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



Standard TM Operation

The size of Standard TM is 10-bit wide and its core is a 10-bit count-up counter which is driven by a user selectable internal or external clock source. There are also two internal comparators with the names, Comparator A and Comparator P. These comparators will compare the value in the counter with CCRP and CCRA registers. The CCRP comparator is three bits wide whose value is compared with the highest three bits in the counter while the CCRA is the ten bits and therefore compares 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 STON bit from low to high. The counter will also be cleared automatically by a counter overflow or a compare match with one of its associated comparators. When these conditions occur, a STM interrupt signal will also usually be generated. The Standard Type TM can operate in a number of different operational modes, can be driven by different clock sources including an input pin and can also control a output pin. All operating setup conditions are selected using relevant internal registers.

Standard Type TM Register Description

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

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

10-bit Standard TM Register List

• STMC0 Register

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

Bit 7 **STPAU**: STM counter pause control

0: Run
1: Pause

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

Bit 6~4 **STCK2~STCK0**: Select STM counter clock

000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: f_{SUB}
110: STCK rising edge clock
111: STCK falling edge clock

These three bits are used to select the clock source for the STM. The external pin clock source can be chosen to be active on the rising or falling edge. The clock source f_{SYS} is the system clock, while f_H and f_{SUB} are other internal clocks, the details of which can be found in the oscillator section.

Bit 3 **STON**: STM counter on/off control

0: Off
1: On

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

Bit 2~0 **STRP2~STRP0**: STM CCRP 3-bit register, compared with the STM counter bit 9~bit 7

Comparator P Match Period

000: 1024 STM clocks
001: 128 STM clocks

010: 256 STM clocks
 011: 384 STM clocks
 100: 512 STM clocks
 101: 640 STM clocks
 110: 768 STM clocks
 111: 896 STM clocks

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

• **STMC1 Register**

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

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

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

Bit 5~4 **STIO1~STIO0**: Select STM external pin function

Compare Match Output Mode

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

PWM Output Mode/Single Pulse Output Mode

00: PWM output inactive state
 01: PWM output active state
 10: PWM output
 11: Single Pulse Output

Capture Input Mode

00: Input capture at rising edge of STPI
 01: Input capture at falling edge of STPI
 10: Input capture at rising/falling edge of STPI
 11: Input capture disabled

Timer/Counter Mode

Unused

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

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

level requested by the STIO1 and STIO0 bits must be different from the initial value setup using the STOC bit otherwise no change will occur on the STM output pin when a compare match occurs. After the STM output pin changes state, it can be reset to its initial level by changing the level of the STON bit from low to high.

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

Bit 3 **STOC**: STM STP output control

Compare Match Output Mode

0: Initial low

1: Initial high

PWM Output Mode/Single Pulse Output Mode

0: Active low

1: Active high

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

Bit 2 **STPOL**: STM STP output polarity control

0: Non-invert

1: Invert

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

Bit 1 **STDPX**: STM PWM duty/period control

0: CCRP – period; CCRA – duty

1: CCRP – duty; CCRA – period

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

Bit 0 **STCCLR**: STM counter clear condition selection

0: Comparator P match

1: Comparator A match

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

• **STMDL Register**

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

Bit 7~0 **D7~D0**: STM counter low byte register bit 7 ~ bit 0
 STM 10-bit Counter bit 7 ~ bit 0

• **STMDH Register**

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

Bit 7~2 Unimplemented, read as “0”
 Bit 1~0 **D9~D8**: STM counter high byte register bit 1 ~ bit 0
 STM 10-bit Counter bit 9 ~ bit 8

• **STMAL Register**

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

Bit 7~0 **D7~D0**: STM CCRA low byte register bit 7 ~ bit 0
 STM 10-bit CCRA bit 7 ~ bit 0

• **STMAH Register**

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

Bit 7~2 Unimplemented, read as “0”
 Bit 1~0 **D9~D8**: STM CCRA high byte register bit 1 ~ bit 0
 STM 10-bit CCRA bit 9 ~ bit 8

Standard Type TM Operation Modes

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

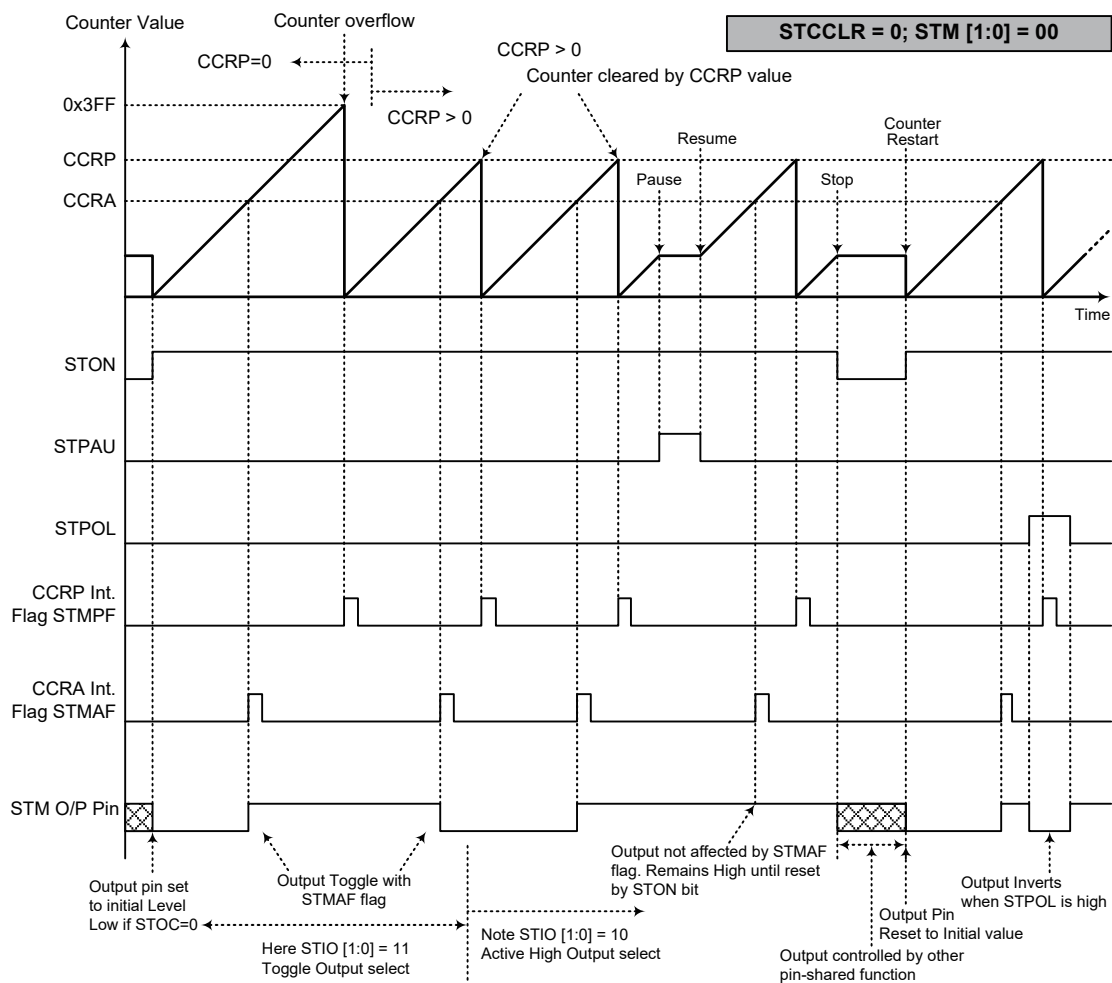
Compare Match Output Mode

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

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

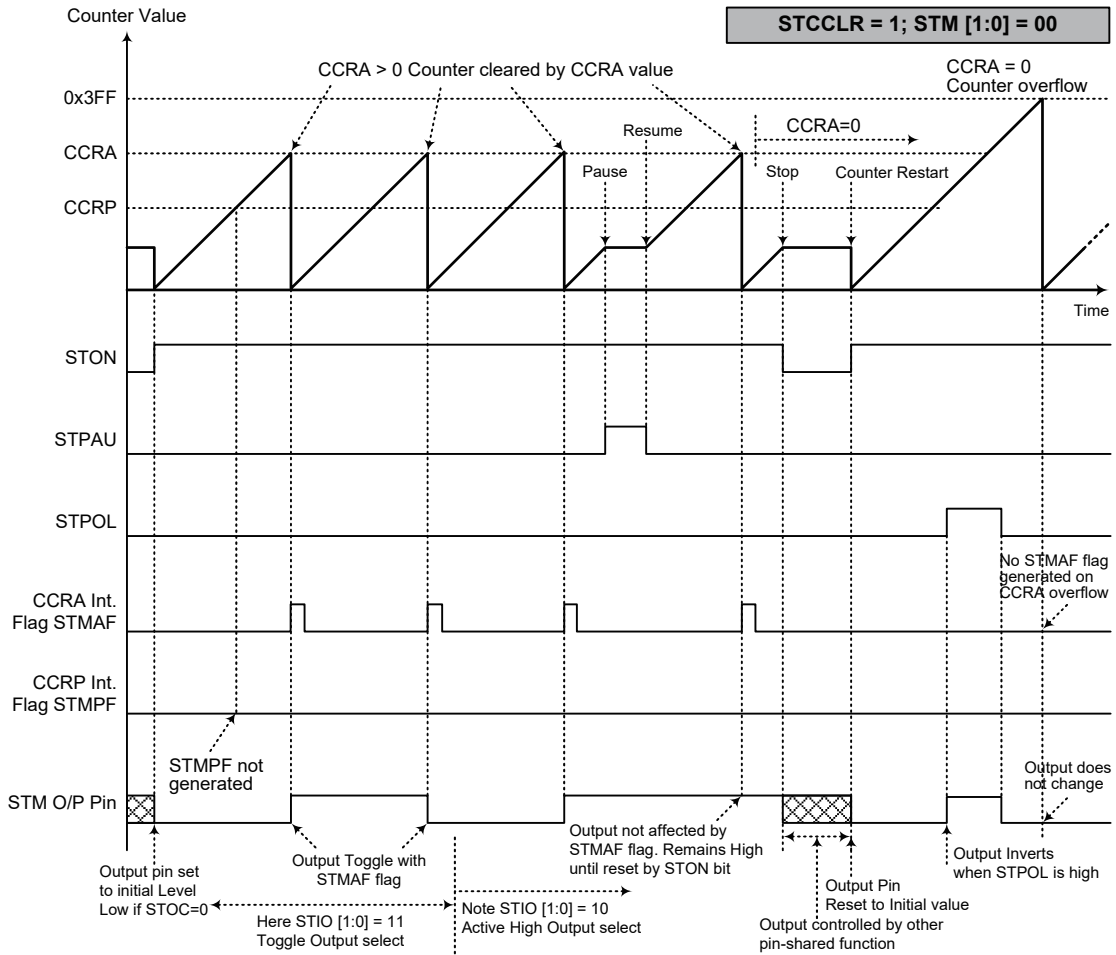
If the CCRA bits are all zero, the counter will overflow when it reaches its maximum 10-bit, 3FF Hex, value, however here the STMAF interrupt request flag will not be generated.

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



Compare Match Output Mode – STCCLR=0

- Note: 1. With STCCLR=0 a Comparator P match will clear the counter
2. The STM output pin is controlled only by the STMAF flag
3. The output pin is reset to its initial state by a STON bit rising edge



Compare Match Output Mode – STCCLR=1

- Note:
1. With STCCLR=1 a Comparator A match will clear the counter
 2. The STM output pin is controlled only by the STMAF flag
 3. The output pin is reset to its initial state by a STON bit rising edge
 4. The STMPF flag is not generated when STCCLR=1

Timer/Counter Mode

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

PWM Output Mode

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

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

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

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

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

If $f_{SYS}=8\text{MHz}$, STM clock source is $f_{SYS}/4$, CCRP=2 and CCRA=128,

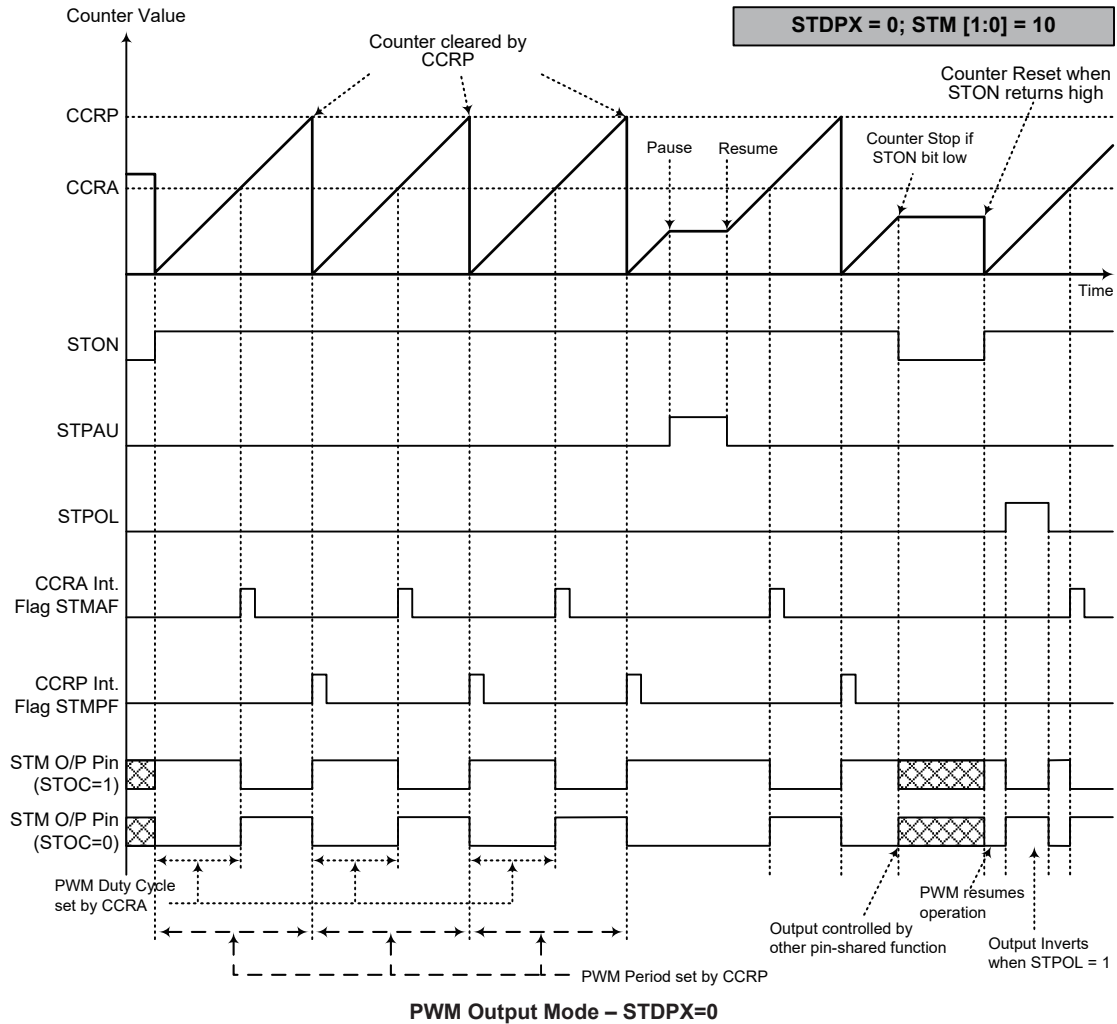
The STM PWM output frequency= $(f_{SYS}/4)/(2 \times 128)=f_{SYS}/1024=7.81\text{kHz}$, duty= $128/(2 \times 128)=50\%$.

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

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

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

The PWM output period is determined by the CCRA register value together with the STM clock while the PWM duty cycle is defined by the CCRP register value except when the CCRP value is equal to 0.

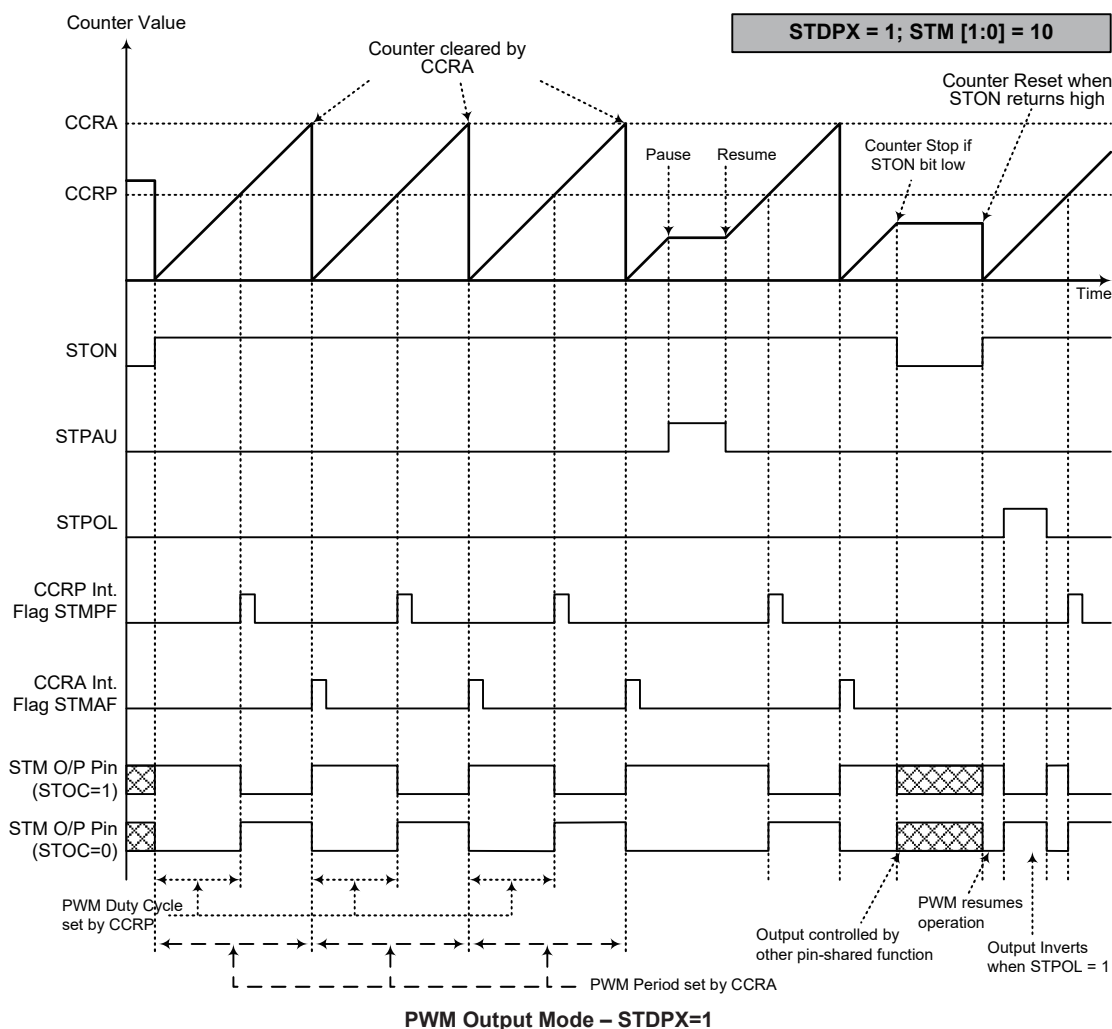


Note: 1. Here STDPX=0 – Counter cleared by CCRP

2. A counter clear sets the PWM Period

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

4. The STCCLR bit has no influence on PWM operation



- Note:
1. Here STDPX=1 – Counter cleared by CCRA
 2. A counter clear sets the PWM Period
 3. The internal PWM function continues even when STIO [1:0]=00 or 01
 4. The STCCLR bit has no influence on PWM operation

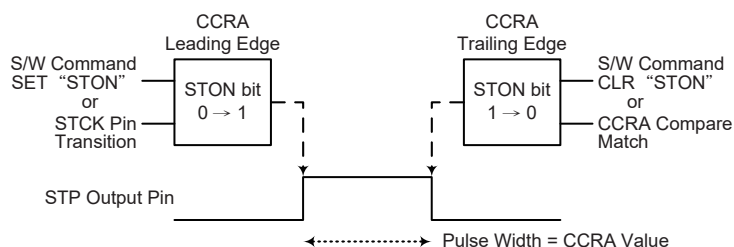
Single Pulse Output Mode

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

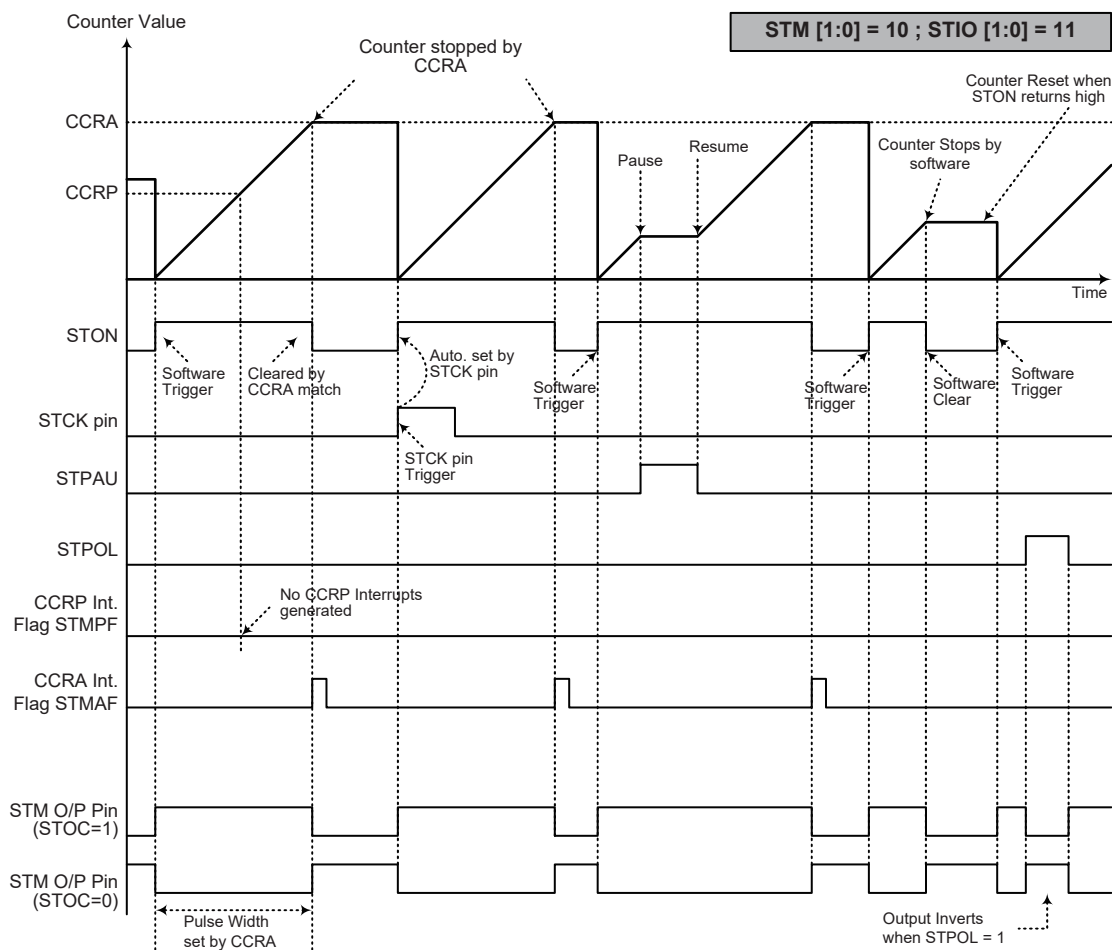
The trigger for the pulse output leading edge is a low to high transition of the STON bit, which can be implemented using the application program. However in the Single Pulse Output Mode, the STON bit can also be made to automatically change from low to high using the external STCK pin, which will in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

However a compare match from Comparator A will also automatically clear the STON bit and thus generate the Single Pulse output trailing edge. In this way the CCRA value can be used to

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



Single Pulse Generation



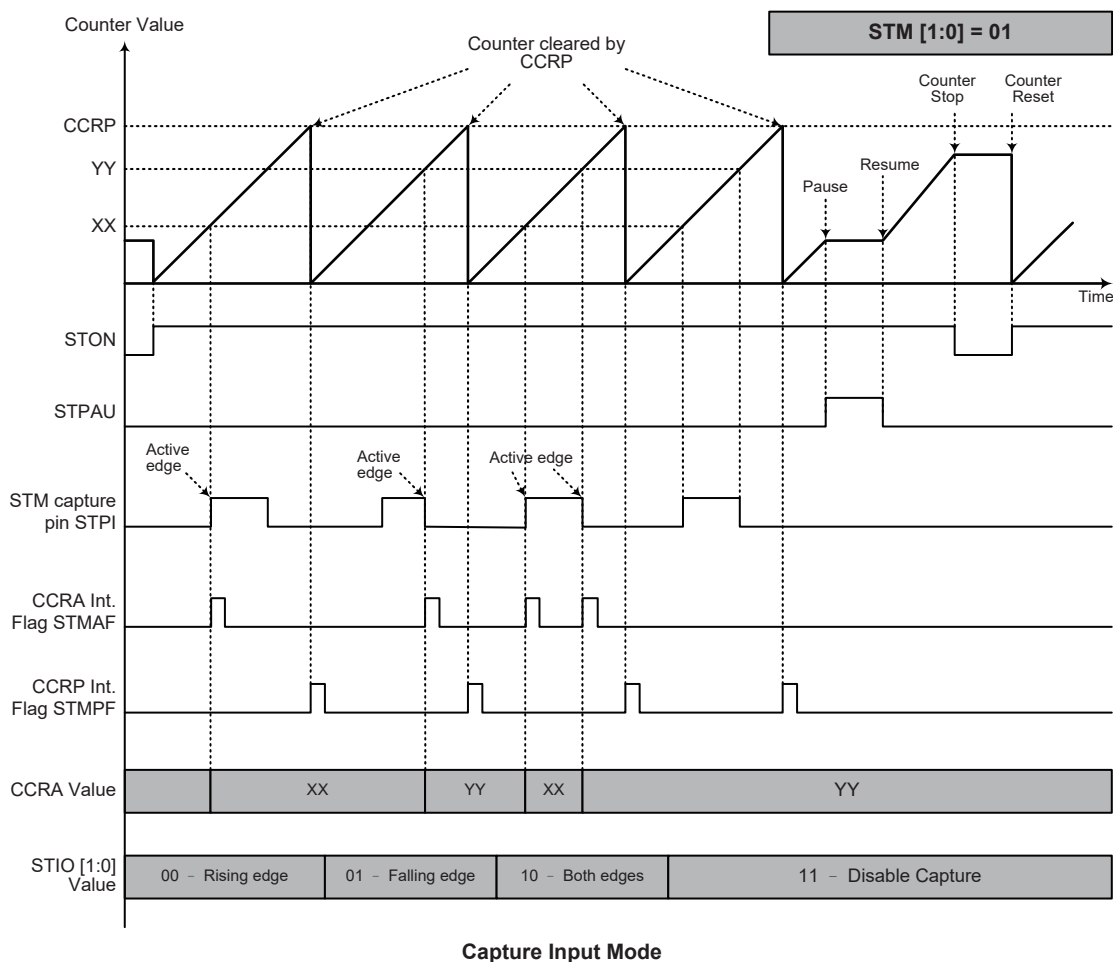
Single Pulse Output Mode

- Note:
1. Counter stopped by CCRA
 2. CCRP is not used
 3. The pulse triggered by the STCK pin or by setting the STON bit high
 4. A STCK pin active edge will automatically set the STON bit high.
 5. In the Single Pulse Output Mode, STIO[1:0] must be set to "11" and can not be changed.

Capture Input Mode

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

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



- Note: 1. STM [1:0]=01 and active edge set by the STIO [1:0] bits
2. A STM Capture input pin active edge transfers the counter value to CCRA
3. STCCLR bit not used
4. No output function – STOC and STPOL bits are not used
5. CCRP determines the counter value and the counter has a maximum count value when CCRP is equal to zero.

Touch Key Function

These devices provide 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 manipulation of internal registers.

Touch Key Structure

The touch keys are pin shared with the I/O pins, with the desired function chosen via the pin-shared selection register bit. Keys are organised into one group, known as a module. The module is a fully independent set of four Touch Keys and has its own oscillator. The module contains its own control logic circuits and register set.

Total Key Number	Touch Key	Shared I/O Pin
4	KEY1~KEY4	PB4~PB7

Touch Key Structure

Touch Key Register Definition

The touch key module, which contains four touch key functions, has its own suite registers. The following table shows the register set for the 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
TKC2	Touch key function control register 2
TK16DL	Touch key function 16-bit counter low byte
TK16DH	Touch key function 16-bit counter high byte
TKM016DL	Touch key module 0 16-bit C/F counter low byte
TKM016DH	Touch key module 0 16-bit C/F counter high byte
TKM0ROL	Touch key module 0 reference oscillator capacitor select low byte
TKM0ROH	Touch key module 0 reference oscillator capacitor select high byte
TKM0C0	Touch key module 0 control register 0
TKM0C1	Touch key module 0 control register 1
TKM0C2	Touch key module 0 control register 2
TK1M0TH16L	Touch key module 0 KEY1 16-bit threshold low byte
TK1M0TH16H	Touch key module 0 KEY1 16-bit threshold high byte
TK2M0TH16L	Touch key module 0 KEY2 16-bit threshold low byte
TK2M0TH16H	Touch key module 0 KEY2 16-bit threshold high byte
TK3M0TH16L	Touch key module 0 KEY3 16-bit threshold low byte
TK3M0TH16H	Touch key module 0 KEY3 16-bit threshold high byte
TK4M0TH16L	Touch key module 0 KEY4 16-bit threshold low byte
TK4M0TH16H	Touch key module 0 KEY4 16-bit threshold high byte
TKM0THS	Touch key module 0 threshold comparison flag

Touch Key Function Register Definition

Register Name	Bit							
	7	6	5	4	3	2	1	0
TKTMR	D7	D6	D5	D4	D3	D2	D1	D0
TKC0	TKRAMC	TKRCOV	TKST	TKCFOV	TK16OV	TKMOD1	TKMOD0	TKBUSY
TKC1	D7	D6	D5	D4	TK16S1	TK16S0	TKFS1	TKFS0
TKC2	—	—	—	—	—	—	ASMP1	ASMP0
TK16DL	D7	D6	D5	D4	D3	D2	D1	D0
TK16DH	D15	D14	D13	D12	D11	D10	D9	D8
TKM0C0	—	—	M0DFEN	M0FILEN	M0SOFC	M0SOF2	M0SOF1	M0SOF0
TKM0C1	M0TSS	—	M0ROEN	M0KOEN	M0K4EN	M0K3EN	M0K2EN	M0K1EN
TKM0C2	M0SK31	M0SK30	M0SK21	M0SK20	M0SK11	M0SK10	M0SK01	M0SK00
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
TK1M0TH16L	D7	D6	D5	D4	D3	D2	D1	D0
TK1M0TH16H	D15	D14	D13	D12	D11	D10	D9	D8
TK2M0TH16L	D7	D6	D5	D4	D3	D2	D1	D0
TK2M0TH16H	D15	D14	D13	D12	D11	D10	D9	D8
TK3M0TH16L	D7	D6	D5	D4	D3	D2	D1	D0
TK3M0TH16H	D15	D14	D13	D12	D11	D10	D9	D8
TK4M0TH16L	D7	D6	D5	D4	D3	D2	D1	D0
TK4M0TH16H	D15	D14	D13	D12	D11	D10	D9	D8
TKM0THS	M0K4THF	M0K3THF	M0K2THF	M0K1THF	M0K4THS	M0K3THS	M0K2THS	M0K1THS

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

The touch key time slot counter preload register is used to determine the touch key time slot overflow time. The time slot unit period is obtained by a 5-bit counter and equal to 32 time slot clock cycles. Therefore, the time slot counter overflow time is equal to the following equation shown.

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

• TKC0 Register

Bit	7	6	5	4	3	2	1	0
Name	TKRAMC	TKRCOV	TKST	TKCFOV	TK16OV	TKMOD1	TKMOD0	TKBUSY
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
POR	0	0	0	0	0	0	1	0

Bit 7

TKRAMC: Touch key data memory access control

0: Accessed by MCU

1: Accessed by touch key module

This bit determines that the touch key data memory is used by the MCU or the touch key module. However, the touch key module will have the priority to access the touch

key data memory when the touch key module operates in the auto scan mode or the periodic auto scan mode, i.e., the TKST bit state is changed from 0 to 1 when the TKMOD1~TKMOD0 bits are set to 00, 10 or 11. After the touch key auto scan or the periodic auto scan operation is completed, i.e., the TKBUSY bit state is changed from 1 to 0, the touch key data memory access will be controlled by the TKRAMC bit. Therefore, it is recommended to set the TKRAMC bit to 1 when the touch key module operates in the auto scan mode or the periodic auto scan mode. Otherwise, the contents of the touch key data memory may be modified as this data memory space is configured by the touch key module followed by the MCU access.

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 TKRCOV interrupt request flag will be set. However, if this bit is set by application program, the touch key TKRCOV interrupt request flag will not be affected. Therefore, this bit can not be set by application program but must be cleared to 0 by application program.

In the auto scan mode, if the time slot counter overflows but the touch key auto scan operation is not completed, the TKRCOV bit will not be set. At this time, the touch key module 16-bit C/F counter, touch key function 16-bit counter and 5-bit time slot unit period counter will be automatically cleared but the 8-bit time slot counter will be reloaded from the 8-bit time slot counter preload register. When the touch key auto scan operation is completed, the TKRCOV bit and the Touch Key TKRCOV Interrupt request flag, TKRCOVF, will be set and all module keys 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.

In the periodic auto scan mode, the TKRCOV bit is cleared to zero during the auto scan operation period. Only at the end of the last scan operation in the WDT time-out cycle, the 16-bit C/F counter content will be written into the corresponding touch key data memory, and then the TKRCOV bit will be set high by the hardware circuit. The other actions in this mode are the same as those in the auto scan mode except the above mentioned.

In the manual scan mode, if the time slot counter overflows, the TKRCOV bit and the Touch Key TKRCOV Interrupt request flag, TKRCOVF, will be set and all module keys 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

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 by touch key module 16-bit C/F counter overflow and must be cleared to 0 by application program.

- Bit 3 **TK16OV**: Touch key function 16-bit counter overflow flag
 0: No overflow occurs
 1: Overflow occurs
 This bit is set by touch key function 16-bit counter overflow and must be cleared to 0 by application program.
- Bit 2~1 **TKMOD1~TKMOD0**: Touch key scan mode select
 00: Auto scan mode
 01: Manual scan mode
 1x: Periodic auto scan mode
 In the manual scan mode the reference oscillator capacitor value should be properly configured before the scan operation begins and the touch key module 16-bit C/F counter value should be read after the scan operation finishes by application program.
 In the auto scan mode the data movement which is described above is implemented by hardware. The individual reference oscillator capacitor value and 16-bit C/F counter content for all scanned keys will be read from and written into a dedicated Touch Key Data Memory area. In the auto scan mode the keys to be scanned can be arranged in a specific sequence which is determined by the M0SK3[1:0]~M0SK0[1:0] bits in the TKM0C2 register. The scan operation will not be stopped until all arranged keys are scanned.
 In the periodic auto scan mode the touch key scan operation will be implemented automatically on a periodic basis, which can be determined by the ASMP1~ASMP0 bits in the TKC2 register. Only at the end of the last scan operation in the WDT time-out cycle, the 16-bit C/F counter content for all scanned keys will be written into the corresponding touch key data memory. In addition, when any key C/F counter value is less than the lower threshold if M0KnTHS=0, or larger than the upper threshold if M0KnTHS=1, the TKTH signal will be set high. The other actions in this mode are the same as those in the auto scan mode except the above mentioned.
- Bit 0 **TKBUSY**: Touch key scan operation busy flag
 0: Not busy – no scan operation is executed or scan operation is completed
 1: Busy – scan operation is executing
 This bit indicates whether the touch key scan operation is executing or not. It is set to 1 when the TKST bit is set high to start the scan operation for all touch key scan modes. In the manual scan mode this bit is cleared to 0 automatically when the touch key time slot counter overflows. In the auto scan mode this bit is cleared to 0 automatically when the touch key scan operation is completed. In the periodic auto scan mode this bit is cleared to 0 automatically when the last scan operation in the WDT time-out cycle is completed, or when any key C/F counter value is less than the lower threshold if M0KnTHS=0, or when the value is larger than the upper threshold if M0KnTHS=1.

• TKC1 Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	TK16S1	TK16S0	TKFS1	TKFS0
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	1	1

- Bit 7~5 **D7~D5**: Data bits for test only
 These bits are used for test purpose only and must be kept as “000” for normal operations.
- Bit 4 **D4**: Reserved, cannot be used
- Bit 3~2 **TK16S1~TK16S0**: Touch key function 16-bit counter clock source select
 00: f_{SYS}
 01: $f_{SYS}/2$
 10: $f_{SYS}/4$
 11: $f_{SYS}/8$

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

• **TKC2 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	ASMP1	ASMP0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	1

Bit 7~2 Unimplemented, read as “0”

Bit 1~0 **ASMP1~ASMP0**: Periodic auto scan mode period selection
 00: $2^{14}/f_{LIRC}$
 01: $2^{13}/f_{LIRC}$
 10: $2^{12}/f_{LIRC}$
 11: $2^{11}/f_{LIRC}$

These bits is used to determine the touch key scan period and only available when the touch key function is configured to operate in the periodic auto scan mode. The number of touch key scan times is obtained by the WDT time-out period, t_{WDT} , and the periodic auto scan mode period, t_{KEY} , using the equation, $N=t_{WDT}/t_{KEY}$.

For example, if the WDT time-out period is $2^{15}/f_{LIRC}$ by setting the WS[2:0] bits to 100, then t_{WDT} is equal to 1.024s. Therefore, the number of touch key scan times is 2/4/8/16 times in a WDT time-out cycle when the ASMP[1:0] bits are set to 00/01/10/11 respectively. It is extremely important to ensure that the periodic auto scan mode period t_{KEY} does not exceed the WDT time-out period t_{WDT} in applications.

• **TK16DL 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**: Touch key function 16-bit counter low byte contents

• **TK16DH Register**

Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 **D15~D8**: Touch key function 16-bit counter high byte contents

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 in the manual scan mode, this 16-bit counter will be stopped and the counter content will be unchanged. However, this 16-bit counter content will be cleared to zero at the end of the time slot 0, slot 1 and slot 2 but kept unchanged at the end of the time slot 3 in the auto scan mode or the periodic auto scan mode. This register pair will be cleared to zero when the TKST bit is cleared to zero.

• **TKM016DL 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:** Touch key module 0 16-bit C/F counter low byte contents

• **TKM016DH Register**

Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 **D15~D8:** Touch key module 0 16-bit C/F counter high byte contents

This register pair is used to store the touch key module 0 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 in the manual scan mode. However, this 16-bit C/F counter content will be cleared to zero at the end of the time slot 0, slot 1 and slot 2 but kept unchanged at the end of the time slot 3 when the auto scan mode or the periodic auto scan mode is selected. This register pair will be cleared to zero when the TKST bit is cleared to zero.

• **TKM0ROL 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 module 0 reference oscillator internal capacitor select

• **TKM0ROH 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:** Touch key module 0 reference oscillator internal capacitor select

This register pair is used to store the touch key module 0 reference oscillator capacitor value. This register pair will be loaded with the corresponding next time slot capacitor value from the dedicated touch key data memory at the end of the current time slot when the auto scan mode or the periodic auto scan mode is selected.

The reference oscillator internal capacitor value = $(TKM0RO[9:0] \times 50pF) / 1024$

• **TKM0C0 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	M0DFEN	M0FILEN	M0SOFC	M0SOF2	M0SOF1	M0SOF0
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 **M0DFEN**: Touch key module 0 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 **M0FILEN**: Touch key module 0 filter function control
0: Disable
1: Enable
- Bit 3 **M0SOFC**: Touch key module 0 C-to-F oscillator frequency hopping function control select
0: Controlled by the M0SOF2~M0SOF0
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 M0SOF2~M0SOF0 bits value.
- Bit 2~0 **M0SOF2~M0SOF0**: Touch key module 0 Reference and Key oscillators hopping frequency select
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 M0SOFC 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.

• **TKM0C1 Register**

Bit	7	6	5	4	3	2	1	0
Name	M0TSS	—	M0ROEN	M0KOEN	M0K4EN	M0K3EN	M0K2EN	M0K1EN
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 **M0TSS**: Touch key module 0 time slot counter clock source select
0: Touch key module 0 reference oscillator
1: $f_{SYS}/4$
- Bit 6 Unimplemented, read as “0”
- Bit 5 **M0ROEN**: Touch key module 0 Reference oscillator enable control
0: Disable
1: Enable
This bit is used to enable the touch key module reference oscillator. In the auto scan mode or the periodic auto scan mode, the reference oscillator will automatically be enabled by setting the M0ROEN bit high when the TKST bit is set from low to high if the reference oscillator is selected as the time slot clock source. The combination of the M0TSS and M0K4EN~M0K1EN bits determines whether the reference oscillator is used or not. When the TKBUSY bit is changed from high to low, the M0ROEN bit will automatically be cleared to zero to disable the reference oscillator.
In the manual scan mode the reference oscillator should first be enabled before setting the TKST bit from low to high if the reference oscillator is selected to be used and will be disabled when the TKBUSY bit is changed from high to low.

- Bit 4 **M0KOEN**: Touch key module 0 Key oscillator enable control
 0: Disable
 1: Enable
- This bit is used to enable the touch key module key oscillator. In the auto scan mode or the periodic auto scan mode, the key oscillator will automatically be enabled by setting the M0KOEN bit high when the TKST bit is set from low to high. When the TKBUSY bit is changed from high to low, the M0KOEN bit will automatically be cleared to zero to disable the key oscillator.
- In the manual scan mode the key oscillator should first be enabled before setting the TKST bit from low to high if the relevant key is enabled to be scanned and will be disabled when the TKBUSY bit is changed from high to low.
- Bit 3 **M0K4EN**: Touch key module 0 KEY4 enable control
 0: Disable
 1: Enable
- Bit 2 **M0K3EN**: Touch key module 0 KEY3 enable control
 0: Disable
 1: Enable
- Bit 1 **M0K2EN**: Touch key module 0 KEY2 enable control
 0: Disable
 1: Enable
- Bit 0 **M0K1EN**: Touch key module 0 KEY1 enable control
 0: Disable
 1: Enable

• TKM0C2 Register

Bit	7	6	5	4	3	2	1	0
Name	M0SK31	M0SK30	M0SK21	M0SK20	M0SK11	M0SK10	M0SK01	M0SK00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	0	0	1	0	0

- Bit 7~6 **M0SK31~M0SK30**: Touch key module 0 time slot 3 key scan select
 00: KEY1
 01: KEY2
 10: KEY3
 11: KEY4
- These bits are used to select the desired scan key in time slot 3 and only available in the auto scan mode or the periodic auto scan mode.
- Bit 5~4 **M0SK21~M0SK20**: Touch key module 0 time slot 2 key scan select
 00: KEY1
 01: KEY2
 10: KEY3
 11: KEY4
- These bits are used to select the desired scan key in time slot 2 and only available in the auto scan mode or the periodic auto scan mode.
- Bit 3~2 **M0SK11~M0SK10**: Touch key module 0 time slot 1 key scan select
 00: KEY1
 01: KEY2
 10: KEY3
 11: KEY4
- These bits are used to select the desired scan key in time slot 1 and only available in the auto scan mode or the periodic auto scan mode.

Bit 1~0 **M0SK01~M0SK00**: Touch key module 0 time slot 0 key scan select
 00: KEY1
 01: KEY2
 10: KEY3
 11: KEY4

These bits are used to select the desired scan key in time slot 0 in the auto scan mode or the periodic auto scan mode or used as the multiplexer for scan key select in the manual mode.

• **TKnM0TH16L Register (n=1~4)**

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 module 0 KEYn 16-bit threshold low byte contents

• **TKnM0TH16H Register (n=1~4)**

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**: Touch key module 0 KEYn 16-bit threshold high byte contents

These four register pairs are used to store the touch key module 0 KEY1~KEY4 16-bit upper/lower threshold value respectively. After the touch key module 0 KEYn scan operation is completed, the 16-bit C/F counter content, TKM016DH/TKM016DL, will be compared with the TKnM0TH16H/TKnM0TH16L value by the hardware. When this value is less than the the lower threshold if M0KnTHS=0, or larger than the upper threshold if M0KnTHS=1, then the M0KnTHF flag will be set high, and an interrupt signal will be generated.

• **TKM0THS Register**

Bit	7	6	5	4	3	2	1	0
Name	M0K4THF	M0K3THF	M0K2THF	M0K1THF	M0K4THS	M0K3THS	M0K2THS	M0K1THS
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 **M0K4THF**: Touch key module 0 KEY4 upper/lower threshold comparison flag
 0: Not less than lower threshold or not larger than upper threshold
 1: Less than lower threshold or larger than upper threshold

Bit 6 **M0K3THF**: Touch key module 0 KEY3 upper/lower threshold comparison flag
 0: Not less than lower threshold or not larger than upper threshold
 1: Less than lower threshold or larger than upper threshold

Bit 5 **M0K2THF**: Touch key module 0 KEY2 upper/lower threshold comparison flag
 0: Not less than lower threshold or not larger than upper threshold
 1: Less than lower threshold or larger than upper threshold

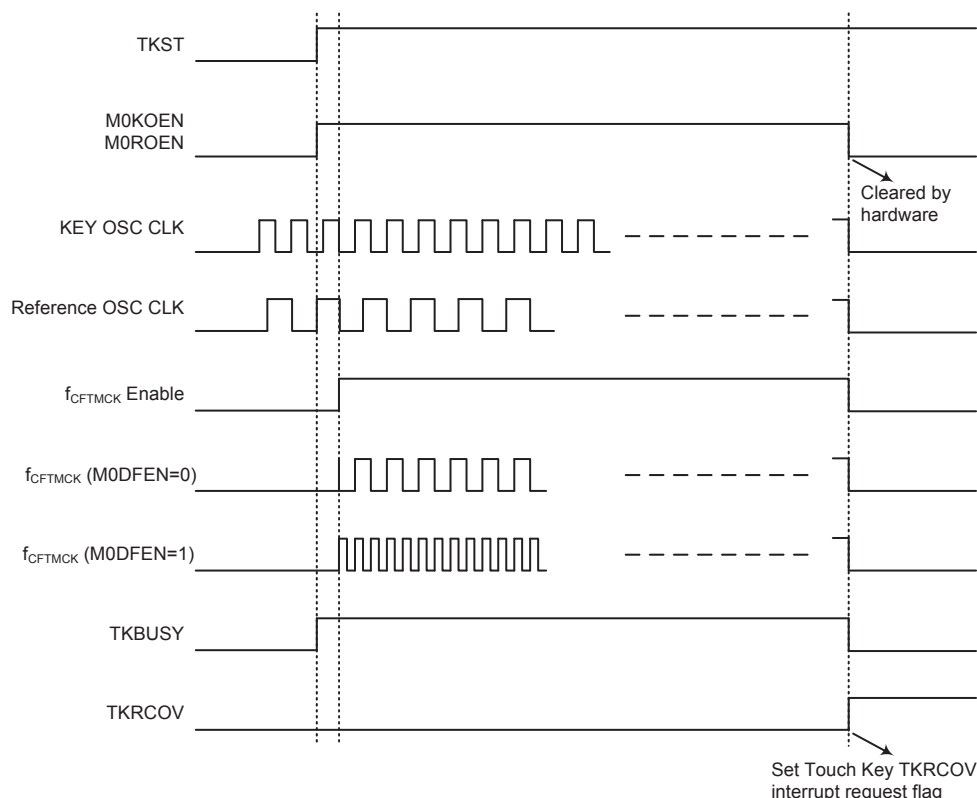
Bit 4 **M0K1THF**: Touch key module 0 KEY1 upper/lower threshold comparison flag
 0: Not less than lower threshold or not larger than upper threshold
 1: Less than lower threshold or larger than upper threshold

Bit 3 **M0K4THS**: Touch key module 0 KEY4 upper or lower threshold comparison selection
 0: Lower threshold comparison
 1: Upper threshold comparison

Bit 2	M0K3THS : Touch key module 0 KEY3 upper or lower threshold comparison selection 0: Lower threshold comparison 1: Upper threshold comparison
Bit 1	M0K2THS : Touch key module 0 KEY2 upper or lower threshold comparison selection 0: Lower threshold comparison 1: Upper threshold comparison
Bit 0	M0K1THS : Touch key module 0 KEY1 upper or lower threshold comparison selection 0: Lower threshold comparison 1: Upper threshold comparison

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 Manual Scan Mode Timing Diagram

The touch key module contains four touch key inputs, namely KEY1~KEY4, which are shared with logical I/O pins, and the desired function is selected using register bits. The touch key has its own independent sense oscillator. There are therefore four 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 TKRCOV interrupt signal will be generated in the manual scan mode.

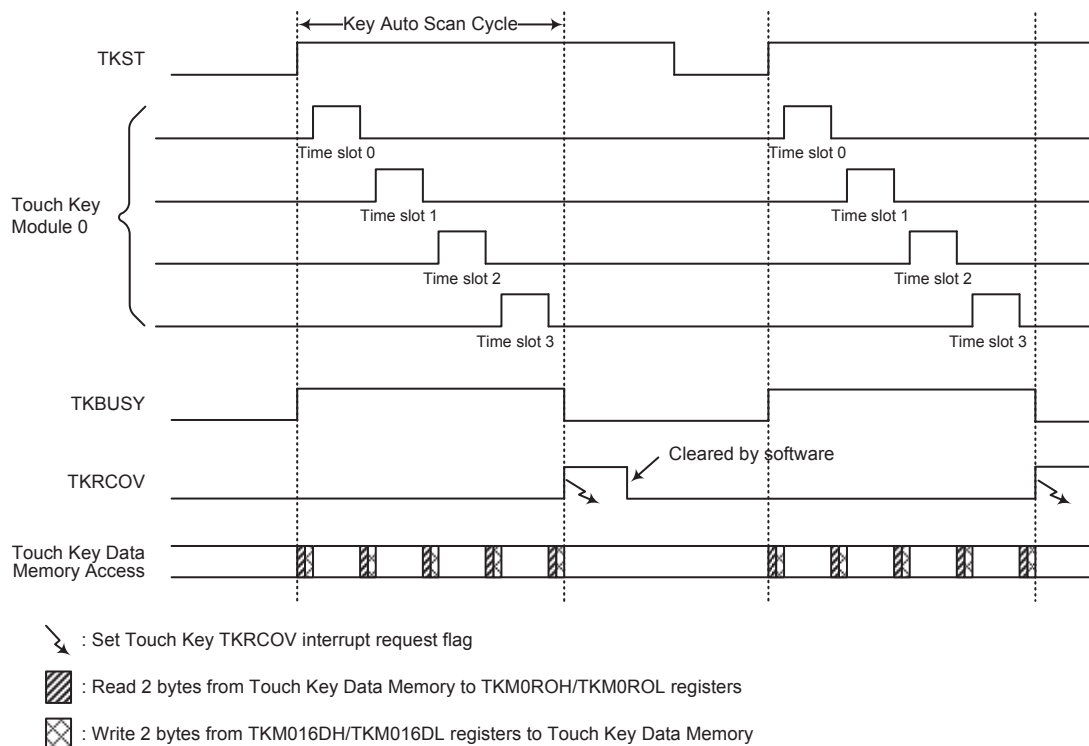
The touch key module 16-bit C/F counter, 16-bit counter, 5-bit time slot unit period counter in the module will be automatically cleared when the TKST bit is cleared to zero, but the 8-bit programmable time slot counter will not be cleared. The overflow time is setup by user. When the TKST bit changes from low to high, the 16-bit C/F counter, 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 the module will be automatically stopped and the 16-bit C/F counter, 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 M0TSS bit in the TKM0C1 register. The reference oscillator and key oscillator will be enabled by setting the M0ROEN bit and M0KOEN bits in the TKM0C1 register.

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

Auto Scan Mode

There are three scan modes contained for the touch key function. The auto scan mode, the periodic auto scan mode and the manual scan mode are selected using the TKMOD1~TKMOD0 bits in the TKC0 register. The auto scan mode can minisize the load of the application program and improve the touch key scan operation performance. When the TKMOD1~TKMOD0 bits are set to 00, the auto scan mode is selcted to scan the module keys in a specific sequence determined by the M0SK3[1:0]~M0SK0[1:0] bits in the TKM0C2 register.

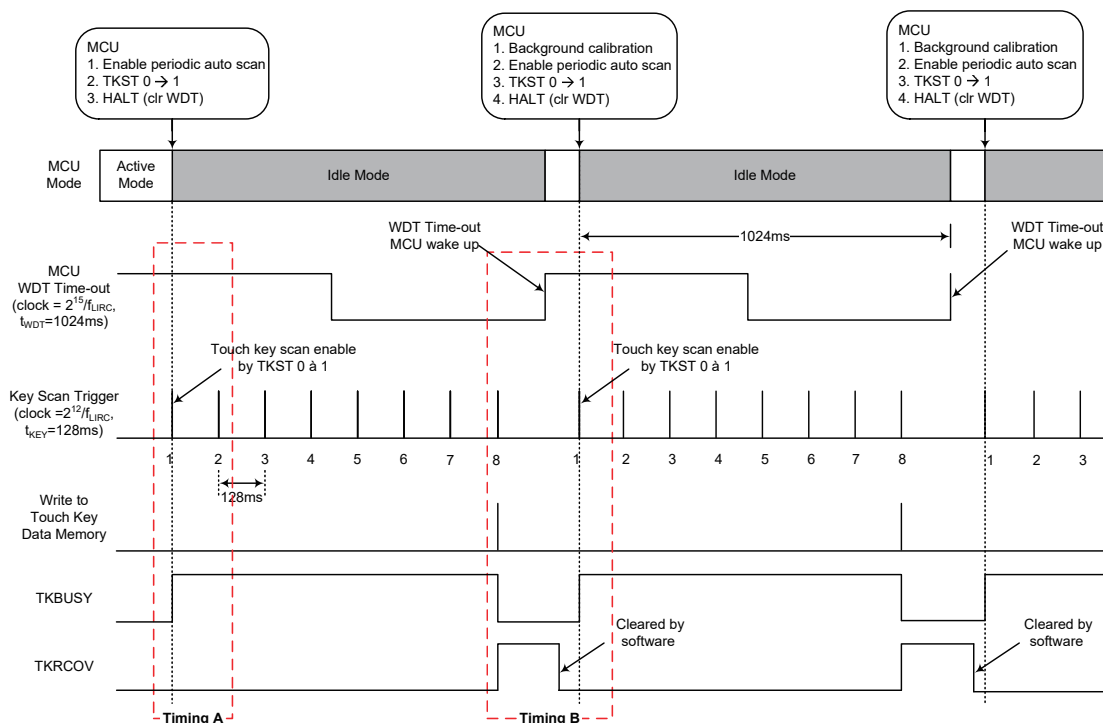


Touch Key Auto Scan Mode Timing Diagram

In the auto scan mode the key oscillator and reference oscillator will automatically be enabled when the TKST bit is set from low to high and disabled automatically when the TKBUSY bit changes from high to low. When the TKST bit is set from low to high in the auto scan mode, the internal capacitor value of the reference oscillator for the selected key to be scanned in the time slot 0 will first be read from a specific location of the dedicated touch key data memory and loaded into the corresponding TKM0ROH/TKM0ROL registers. Then the 16-bit C/F counter value will be written into the corresponding location of the time slot 3 scanned key in the touch key data memory. After this, the selected key will start to be scanned in time slot 0. At the end of the time slot 0 key scan operation, the reference oscillator internal capacitor value for the next selected key will be read from the touch key data memory and loaded into the next TKM0ROH/TKM0ROL registers. Then the 16-bit C/F counter value of the current scanned key will be written into the corresponding touch key data memory. The whole auto scan operation will sequentially be carried out in the above specific way from time slot 0 to time slot 3. At the end of the time slot 3 key scan operation, the reference oscillator internal capacitor value for the time slot 0 selected key will again be read from the touch key data memory and loaded into the corresponding TKM0ROH/TKM0ROL registers. Then the 16-bit C/F counter value will be written into the relevant location of the time slot 3 scanned key in the touch key data memory. After four selected keys are scanned, the TKRCOV bit will be set high and the TKBUSY bit will be cleared to zero as well as an auto scan mode operation is completed.

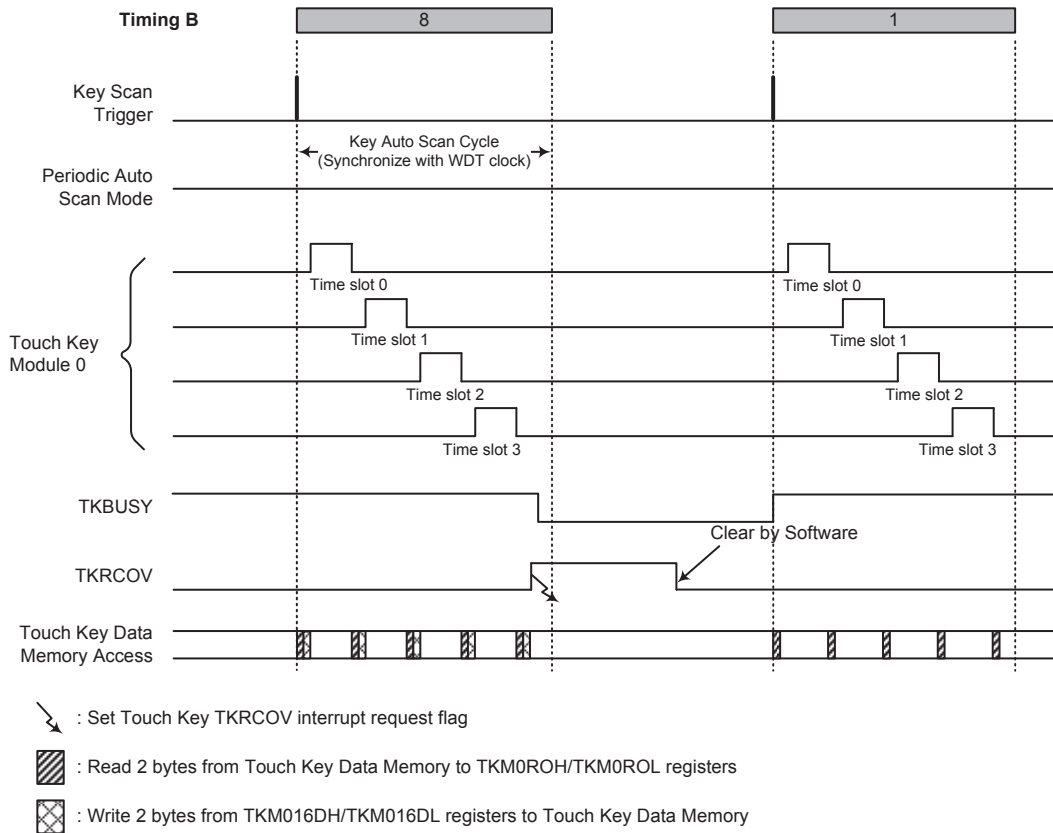
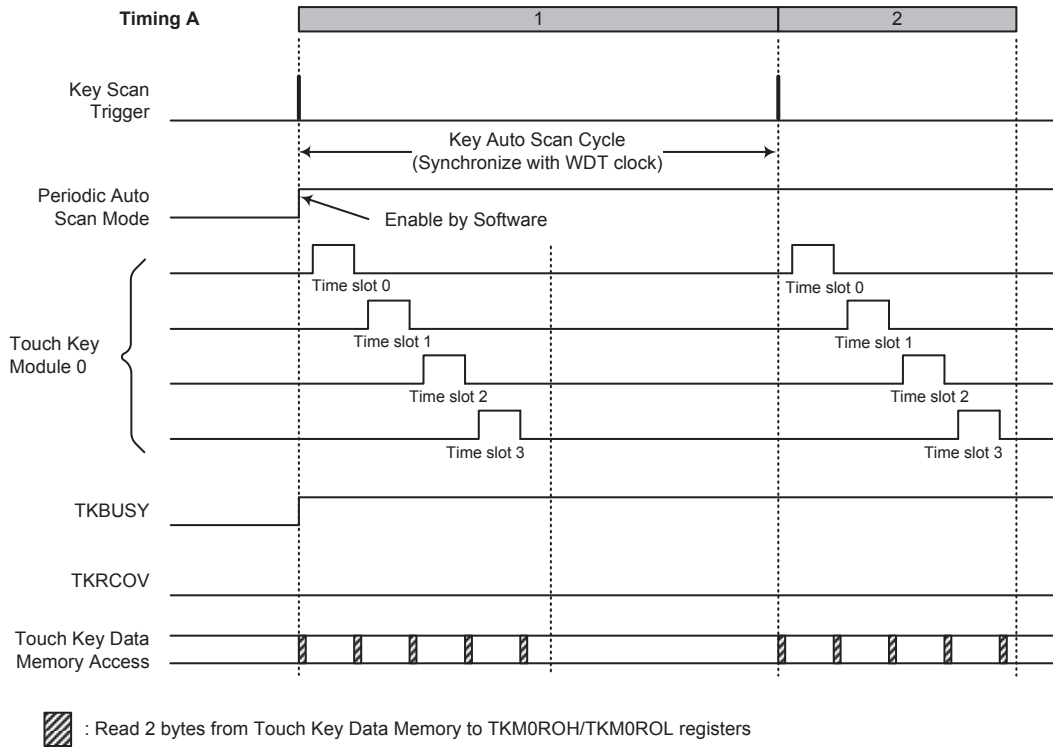
Periodic Auto Scan Mode

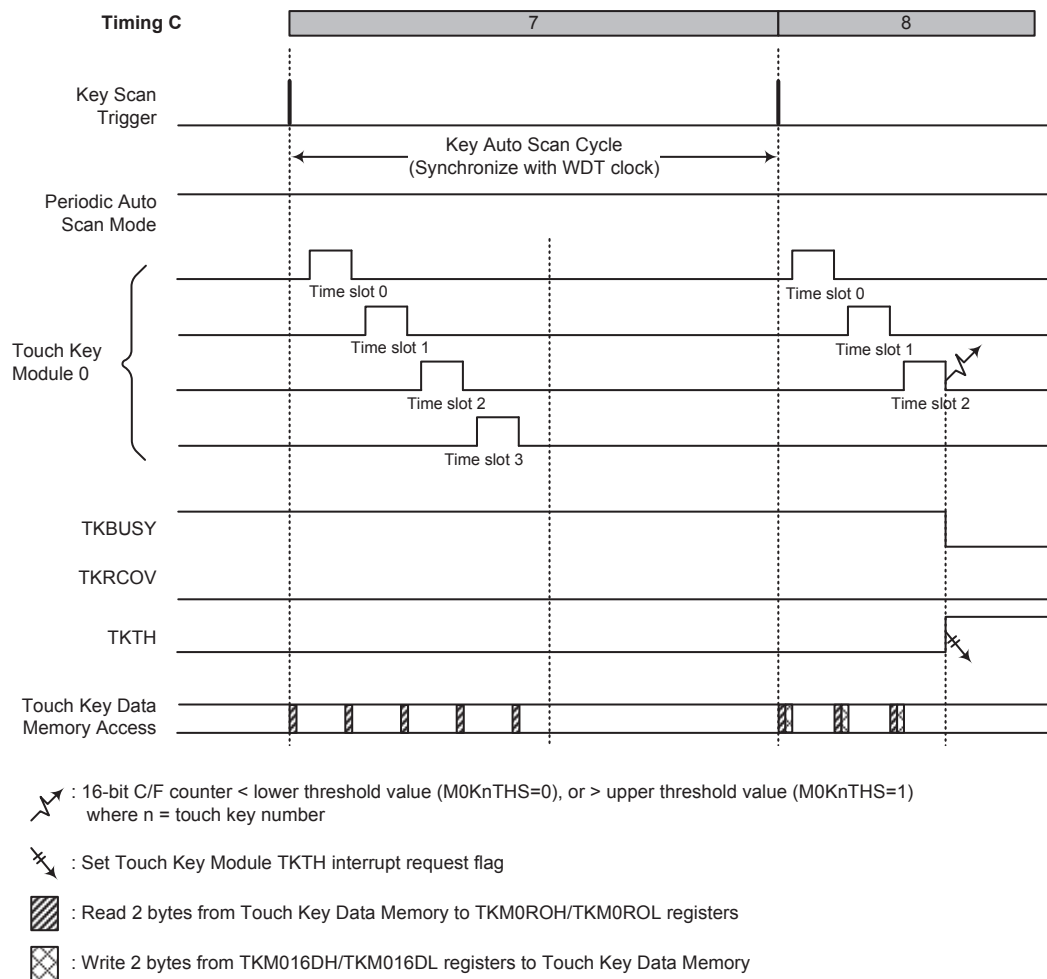
In addition to those actions mentioned in the auto scan mode, the periodic auto scan mode provides periodic auto scan and C/F counter upper/lower threshold comparison functions. When the TKMOD1~TKMOD0 bits are set to 10 or 11, the periodic auto scan mode is selected to scan the module keys automatically and periodically. Note that this mode is generally used in the IDLE mode, in order to monitor the touch key state and minimise power consumption.



Note: Special Timing A and Timing B are shown in the Touch Key Periodic Auto Scan Mode Timing Diagram.

Touch Key Periodic Auto Scan Function in the IDLE Mode





Note: 1. Timing A and Timing B are the specified timing diagram in the Touch Key Periodic Auto Scan Function in the IDLE Mode.

2. Timing C is the timing diagram when a certain touch key threshold comparison condition occurs.

Touch Key Periodic Auto Scan Mode Timing Diagrams

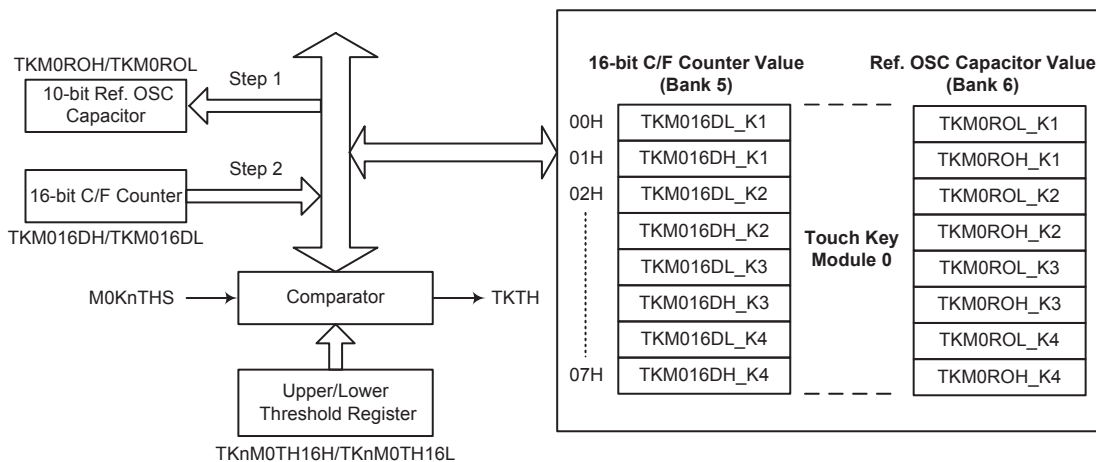
In the periodic auto scan mode the touch key scan operation will be implemented automatically on a periodic basis, which can be determined by the ASMP1~ASMP0 bits in the TKC2 register. The number of touch key scan times depends upon the WDT time-out period and the periodic auto scan mode period. Each auto scan operation will sequentially be carried out in a specific way from time slot 0 to time slot 3 like the auto scan mode. The reference oscillator internal capacitor value for each time slot selected key will be read from the touch key data memory and loaded into the corresponding TKM0ROH/TKM0ROL registers. However, only at the end of the last scan operation in the WDT time-out cycle, the 16-bit C/F counter value for all scanned keys will be written into the corresponding touch key data memory.

In addition, each touch key has its own independent upper/lower threshold comparaor. The upper/lower threshold comparison function will automatically be enabled in the periodic auto scan mode. When any key C/F counter value is less than the lower threshold if M0KnTHS=0, or larger than the upper threshold if M0KnTHS=1, this indicates that the touch key state changes, then the M0KnTHF flag will be set high by the hardware, and an interrupt signal will be generated.

As the periodic auto scan operation is implemented using the WDT counter clock to reduce power consumption, when the WDT is cleared the WDT counter will be reset, the periodic auto scan operation time will be affected but the number of touch key times will not be affected.

Touch Key Data Memory

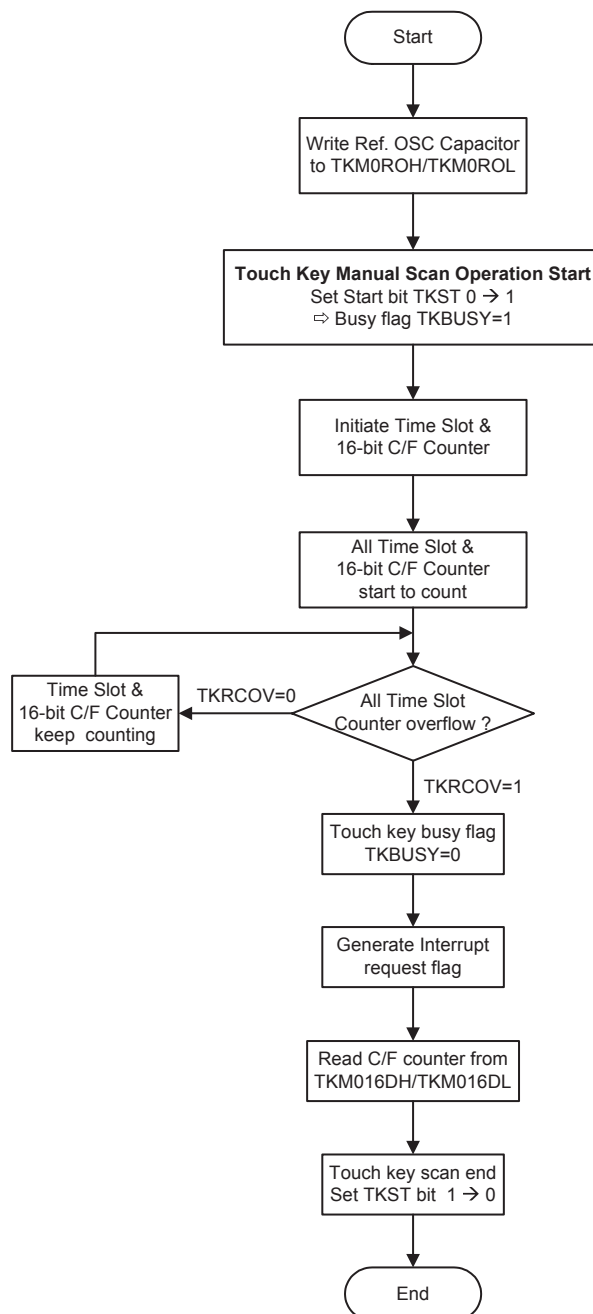
These devices provide two dedicated Data Memory area for the touch key auto scan mode. One area is used to store the 16-bit C/F counter values of the touch key module and located in Data Memory Bank 5. The other area is used to store the reference oscillator internal capacitor values of the touch key module and located in Data Memory Bank 6.



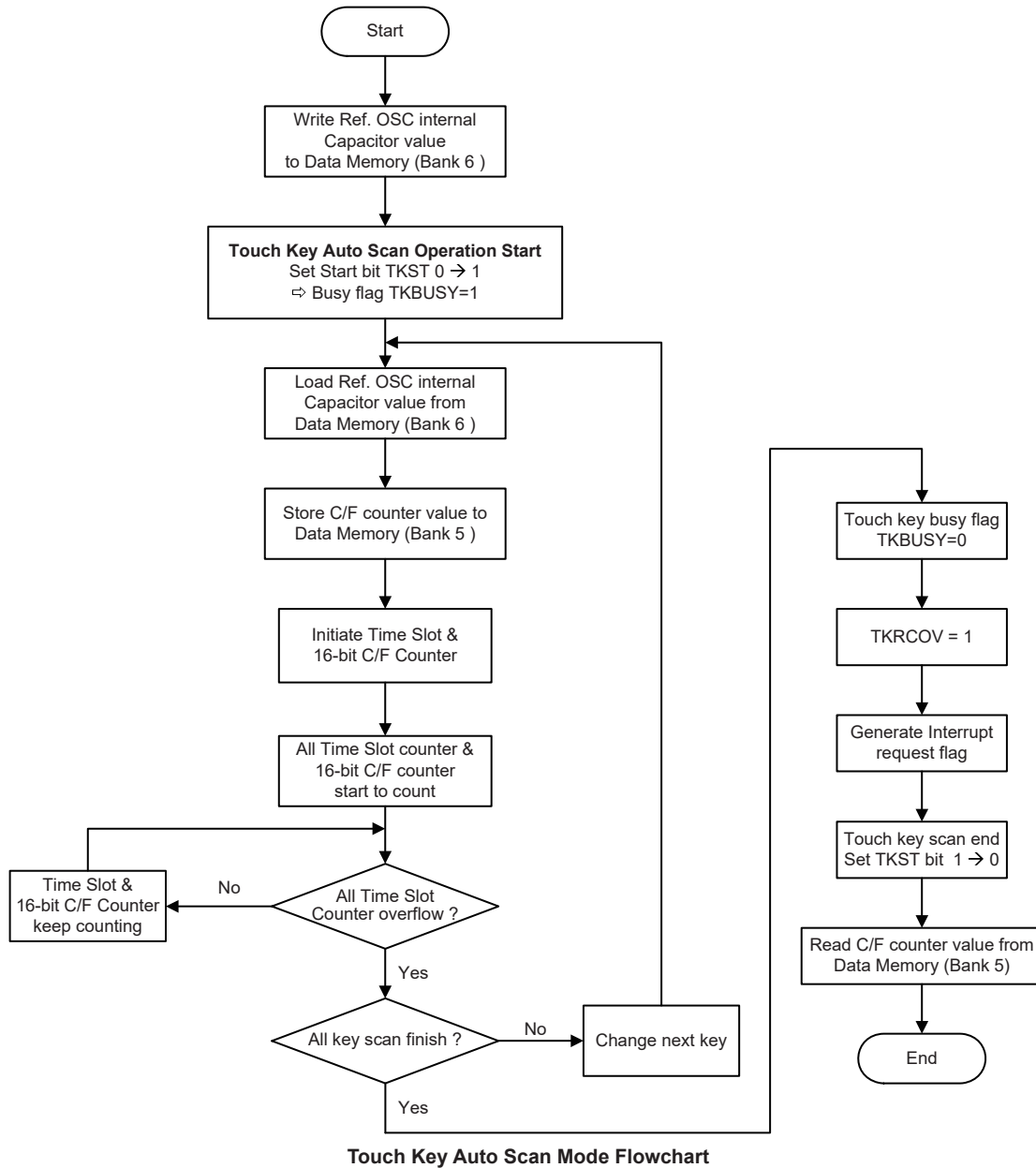
Note: n = Touch Key number, 1~4.

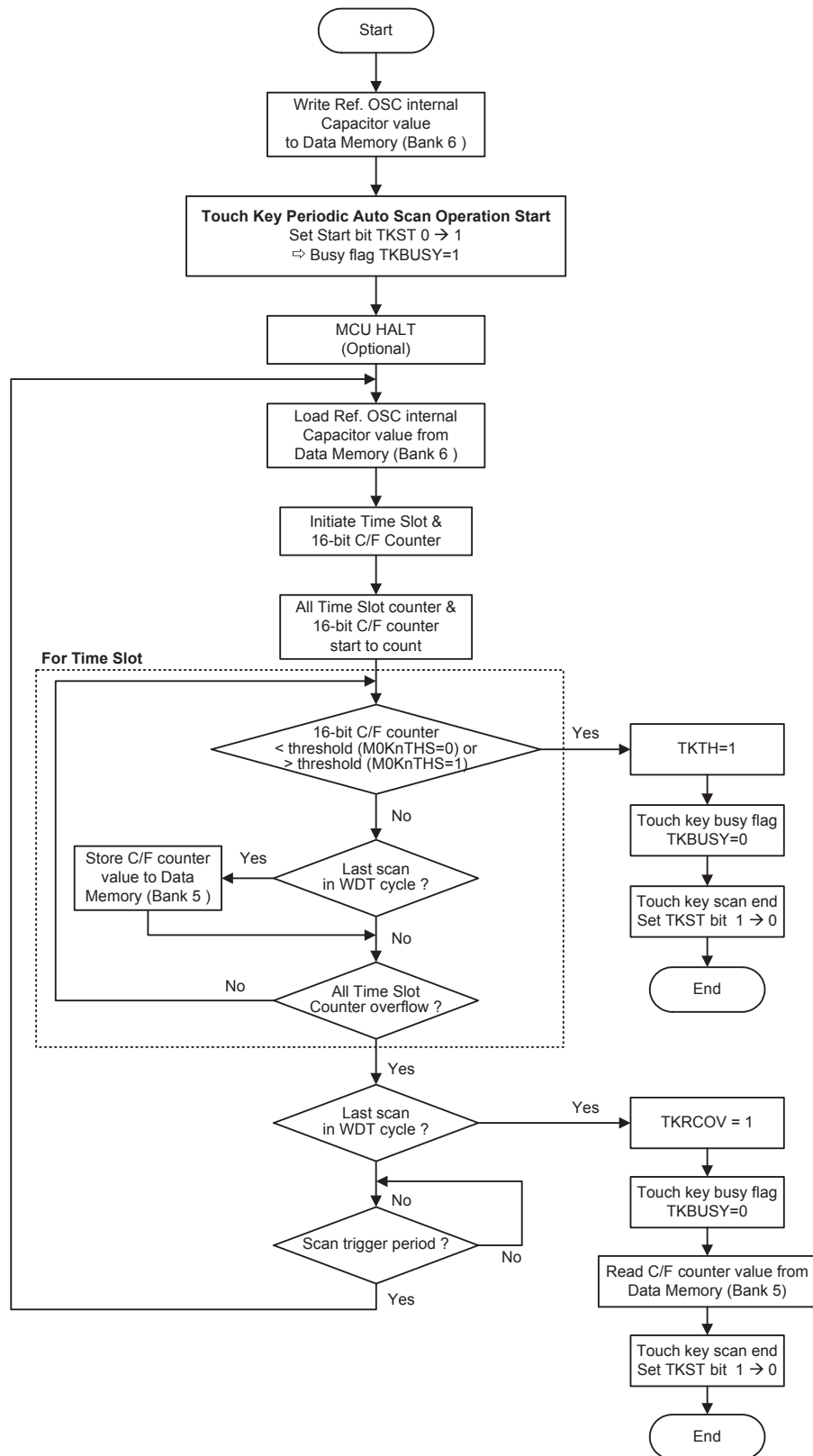
Touch Key Data Memory Map

Touch Key Scan Operation Flowchart



Touch Key Manual Scan Mode Flowchart





Touch Key Periodic Auto Scan Mode Flowchart

Touch Key Interrupts

The touch key has two independent interrupts, known as touch key TKRCOV interrupt and touch key module TKTH interrupt. In the manual scan mode, when the touch key module time slot counter overflows, an actual touch key TKRCOV interrupt will take place. In the auto scan mode, when the touch key auto scan operation is completed, the Touch Key TKRCOV Interrupt request flag, TKRCOVF, will be set. In the periodic auto scan mode, after the last scan operation in the WDT time-out cycle completes the 16-bit C/F counter content is written into the corresponding touch key data memory, then the Touch Key TKRCOV Interrupt request flag, TKRCOVF, will be set. 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 the module will be automatically cleared. When any key C/F counter value is less than the lower threshold if M0KnTHS=0, or larger than the upper threshold if M0KnTHS=1, a touch key threshold interrupt will take place. 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 when 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 in the manual scan mode. When this happens an interrupt signal will be generated. In the auto scan mode, if the time slot counter overflows but the touch key auto scan operation is not completed, the TKRCOV bit will not be set. When the touch key auto scan operation is completed, the TKRCOV bit and the Touch Key TKRCOV Interrupt request flag, TKRCOVF, will be set. In the periodic auto scan mode, the TKRCOV bit is cleared to zero during the auto scan operation period. Only at the end of the last scan operation in the WDT time-out cycle, the 16-bit C/F counter content will be written into the corresponding touch key data memory, and then the TKRCOV bit will be set high by the hardware circuit. The TKTH signal which is the threshold comparison indication signal will go high when a certain threshold comparison condition occurs. When this happens an interrupt signal will also be generated. As the TKRCOV flag will not be automatically cleared, it has to be cleared by the application program.

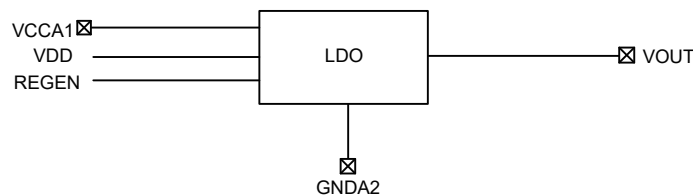
The TKCFOV flag which is the 16-bit C/F counter overflow flag will go high when any of the Touch Key Module 16-bit C/F counter overflows. As this flag will not be automatically cleared, it has to be cleared by the application program.

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.

Voltage Regulator – LDO

These devices include a voltage regulator, LDO. The REGC register controls the regulator module to work in two modes. In the Hi-impedance mode, the LDO will enter power down mode and the VOUT pin will be floating. In the second mode the regulator is turned on, the LDO will output a fixed voltage of 3.3V or 3.0V on the VOUT pin, which can provide power for internal circuits and external application systems. The minimum input voltage= $V_{OUT} + 0.15V$, while the maximum input voltage can be up to 5.5V.



LDO Regulator

Part No.	LDO output voltage V_{OUT}
BS45F5830	3.3V
BS45F5831	3.3V
BS45F5832	3.0V
BS45F5833	3.0V

• REGC Register

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

Bit 7~1 Unimplemented, read as “0”

Bit 0 **REGEN**: Regulator on/off control

0: Regulator off, the VOUT pin is in the Hi-impedance mode

1: Regulator on, the LDO output voltage V_{OUT} is 3.3V or 3.0V

Analog to Digital Converter

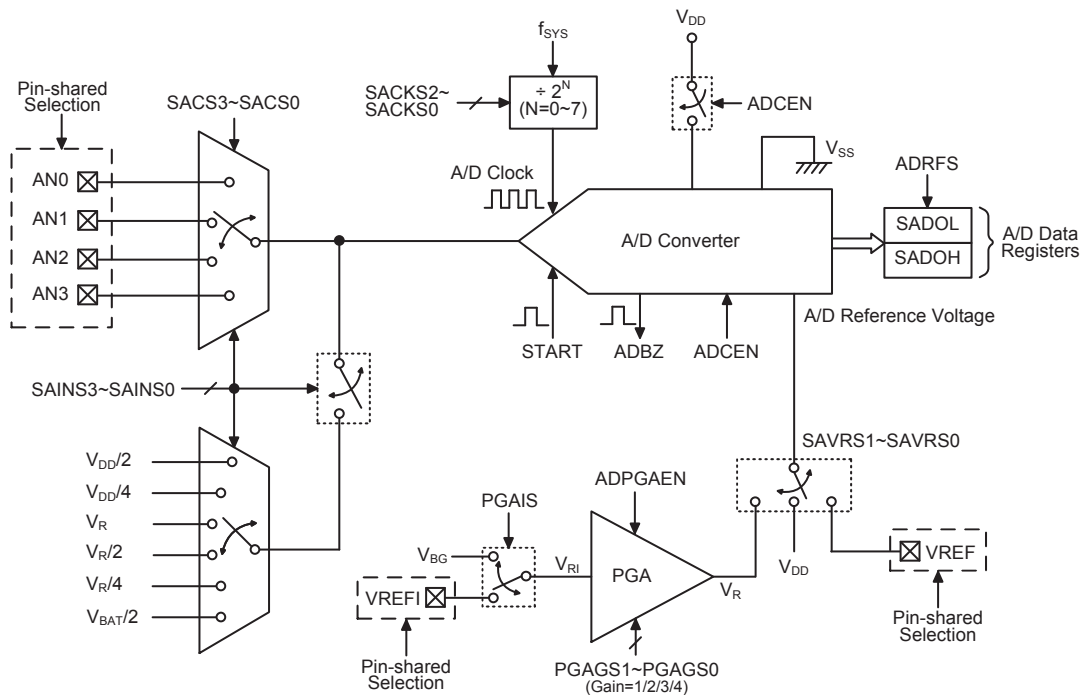
The need to interface to real world analog signals is a common requirement for many electronic systems. However, to properly process these signals by a microcontroller, they must first be converted into digital signals by A/D converters. By integrating the A/D conversion electronic circuitry into the 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

These devices contain 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 Converter input signal selection will be described in the “A/D Converter Input Signals” section.

The accompanying block diagram shows the internal structure of the A/D converter with temperature sensor together with its associated registers and control bits.

External Input Channels	Internal Signal	A/D Signal Select
AN0~AN3	V_{DD} , $V_{DD}/2$, $V_{DD}/4$, V_R , $V_R/2$, $V_R/4$, $V_{BAT}/2$	SAINS3~SAINS0 SACS3~SACS0



Note: The $V_{BAT}/2$ is derived from a divided voltage of the Linear Charger output voltage V_{BAT} .

A/D Converter Structure

A/D Converter Register Descriptions

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

Register Name	Bit							
	7	6	5	4	3	2	1	0
SADOL (ADRF5=0)	D3	D2	D1	D0	—	—	—	—
SADOL (ADRF5=1)	D7	D6	D5	D4	D3	D2	D1	D0
SADOH (ADRF5=0)	D11	D10	D9	D8	D7	D6	D5	D4
SADOH (ADRF5=1)	—	—	—	—	D11	D10	D9	D8
SADC0	START	ADBZ	ADCEN	ADRF5	SACS3	SACS2	SACS1	SACS0
SADC1	SAINS3	SAINS2	SAINS1	SAINS0	—	SACKS2	SACKS1	SACKS0
SADC2	ADPGAEN	—	—	PGAIS	SAVRS1	SAVRS0	PGAGS1	PGAGS0

A/D Converter Register List

A/D Converter Data Registers – SADOL, SADOH

As the internal A/D converter provides a 12-bit digital conversion value, 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. Any unused bits will be read as zero. Note that 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	0	0	0	0
1	0	0	0	0	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 these devices contain only one actual analog to digital converter hardware circuit, each of the external and internal analog signals must be routed to the converter. The SAINS field in the SADC1 register and SACS field 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 to be used as the A/D converter input. When the pin is selected to be an A/D input, its original function whether it is an I/O or other pin-shared function will be removed. In addition, any internal pull-high resistor connected to the pin will be automatically removed if the pin is selected to be an A/D converter input.

• SADC0 Register

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

Bit 7 **START:** Start the A/D conversion

0→1→0: Start

This bit is used to initiate an A/D conversion process. The bit is normally low but if set high and then cleared low again, the A/D converter will initiate a conversion process.

Bit 6 **ADBZ:** A/D converter busy flag

0: No A/D conversion is in progress

1: A/D conversion is in progress

This read only flag is used to indicate whether the A/D conversion is in progress or not. When the START bit is set from low to high and then to low again, the ADBZ flag will be set to 1 to indicate that the A/D conversion is initiated. The ADBZ flag will be cleared to 0 after the A/D conversion is complete.

- Bit 5 **ADCEN**: A/D converter function enable control
0: Disable
1: Enable
This bit controls the A/D internal function. This bit should be set to one to enable the A/D converter. If the bit is 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 known as SADOH and SADOL will be unchanged.
- Bit 4 **ADRF5**: A/D conversion data format select
0: A/D converter data format → SADOH=D [11:4]; SADOL=D[3:0]
1: A/D converter data format → SADOH=D [11:8]; SADOL=D[7:0]
This bit controls the format of the 12-bit converted A/D value in the two A/D data registers. Details are provided in the A/D converter data register section.
- Bit 3~0 **SACS3~SACS0**: A/D converter external analog input channel select
0000: AN0
0001: AN1
0010: AN2
0011: AN3
0100~1111: Undefined, input floating.
These bits are used to select which external analog input channel is to be converted. When the external analog input channel is selected, the SAINS bit field must set to “0000”, “0100” or “11xx”. Details are summarized in the “A/D Converter Input Signal Selection” table.

• **SADC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	SAINS3	SAINS2	SAINS1	SAINS0	—	SACKS2	SACKS1	SACKS0
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 7~4 **SAINS3~SAINS0**: A/D converter input signal select
0000: External source – External analog channel input
0001: A/D converter input signal floating
0010: Internal source – Internal signal derived from $V_{DD}/2$
0011: Internal source – Internal signal derived from $V_{DD}/4$
0100: External source – External analog channel input
0101: Internal source – Internal signal derived from PGA output V_R
0110: Internal source – Internal signal derived from PGA output $V_R/2$
0111: Internal source – Internal signal derived from PGA output $V_R/4$
1000: Internal source – Internal signal derived from $V_{BAT}/2$
1001~1011: Reserved, connected to ground
11xx: External source – External analog channel input
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 select
000: f_{SYS}
001: $f_{SYS}/2$
010: $f_{SYS}/4$
011: $f_{SYS}/8$
100: $f_{SYS}/16$
101: $f_{SYS}/32$
110: $f_{SYS}/64$
111: $f_{SYS}/128$
These bits are used to select the clock source for the A/D converter.

• 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**: PGA enable/disable control

0: Disable

1: Enable

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

Bit 6~5 Unimplemented, read as “0”

Bit 4 **PGAIS**: PGA input (V_{RI}) select

0: External VREFI pin

1: Internal reference voltage, V_{BG}

When the internal reference voltage V_{BG} is selected as the PGA input, the external reference voltage on the VREFI pin will be automatically switched off. When this bit is set high to select V_{BG} as PGA input, the internal bandgap reference V_{BG} should be enabled by setting the VBGEN bit in the LVDC register to “1”.

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

00: Internal A/D converter power, V_{DD}

01: VREF pin

1x: Internal PGA output voltage, V_R

These bits are used to select the A/D converter reference voltage. When the internal A/D converter power or the internal PGA output voltage is selected as the reference voltage, the hardware will automatically disconnect the external VREF input.

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

00: Gain=1

01: Gain=2

10: Gain=3

11: Gain=4

A/D Converter Reference Voltage

The reference voltage supply to the A/D converter can be supplied from the internal power supply, V_{DD} , an external reference source supplied on pin VREF or the internal PGA output voltage, V_R . The desired selection is made using the SAVRS1 and SAVRS0 bits in the SADC2 register. The internal reference voltage is amplified 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, 2, 3 or 4 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_{BG} , selected by the PGAIS bit in the SADC2 register. As the VREFI and VREF pin 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.

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

SAVRS[1:0]	Reference	Description
00	V _{DD}	Internal A/D converter power supply voltage
01	VREF pin	External A/D converter reference pin VREF
1x	V _R	Internal A/D converter PGA output voltage

A/D Converter Reference Voltage Selection

A/D Converter Input Signals

All the external A/D analog channel input pins are pin-shared with the I/O pins as well as other functions. The corresponding control bits for each A/D external input pin in the PAS0 register determines whether the input pins are setup as A/D converter analog inputs or whether they have other functions. If the pin is setup to be as an A/D analog channel input, the original pin functions will be disabled. In this way, pins can be changed under program control to change their function between A/D inputs and other functions. All pull high resistors, which are setup through register programming, will be automatically disconnected if the pins are setup as A/D inputs. Note that it is not necessary to first setup the A/D pin as an input in the port control register to enable the A/D input as when the pin-shared function control bits enable an A/D input, the status of the port control register will be overridden.

As these devices contain 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 SACS bit field 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 SACS bit field can determine which external channel is selected.

When the SAINS field is set to the value of “0101”, “0x10”, “0x11” or “1000”, 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 SACS field value. 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	0000~0011	AN0~AN3	External channel analog input ANn
0001	xxxx	—	Input signal floating
0010	xxxx	V _{DD} /2	Internal signal derived from V _{DD} /2
0011	xxxx	V _{DD} /4	Internal signal derived from V _{DD} /4
0101	xxxx	V _R	Internal signal derived from PGA output V _R
0110	xxxx	V _R /2	Internal signal derived from PGA output V _R /2
0111	xxxx	V _R /4	Internal signal derived from PGA output V _R /4
1000	xxxx	V _{BAT} /2	Internal signal derived from a divided voltage of the Linear Charger output voltage V _{BAT}
1001~1011	xxxx	—	Reserved, connected to ground

“x”: Don't care

A/D Converter Input Signal Selection

A/D Converter Operation

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

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

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

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	$1\mu s$	$2\mu s$	$4\mu s$	$8\mu s$	$16\mu s$ *	$32\mu s$ *	$64\mu s$ *	$128\mu s$ *
2MHz	500ns	$1\mu s$	$2\mu s$	$4\mu s$	$8\mu s$	$16\mu s$ *	$32\mu s$ *	$64\mu s$ *
4MHz	250ns *	500ns	$1\mu s$	$2\mu s$	$4\mu s$	$8\mu s$	$16\mu s$ *	$32\mu s$ *
8MHz	125ns *	250ns *	500ns	$1\mu s$	$2\mu s$	$4\mu s$	$8\mu s$	$16\mu s$ *
12MHz	83ns *	167ns *	333ns *	667ns	$1.33\mu s$	$2.67\mu s$	$5.33\mu s$	$10.67\mu s$ *

A/D Clock Period Examples

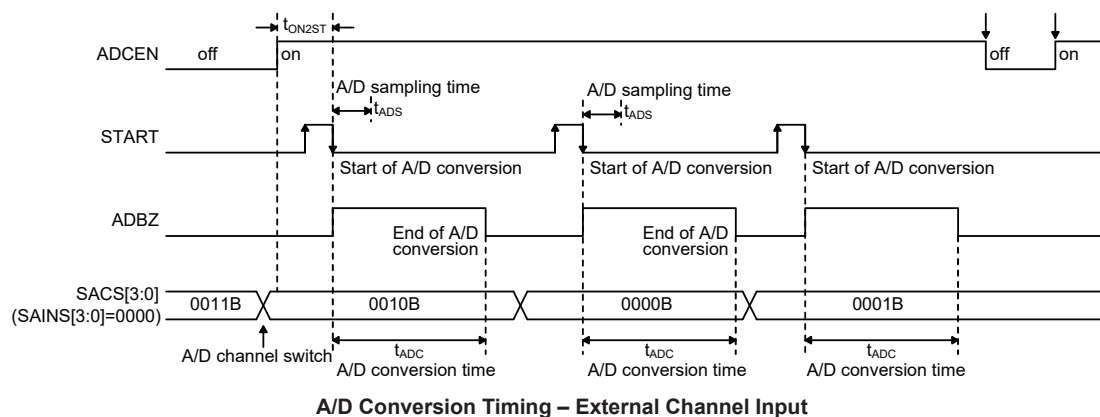
Controlling the power on/off function of the A/D converter circuitry is implemented using the ADCEN bit in the SADC0 register. This bit must be set high to power on the A/D converter. When the ADCEN bit is set high to power on the A/D converter internal circuitry a certain delay, as indicated in the timing diagram, must be allowed before an A/D conversion is initiated. Even if no pins are selected for use as A/D inputs, if the ADCEN bit is high, then some power will still be consumed. In power conscious applications it is therefore recommended that the ADCEN is cleared to zero 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. A total of 16 A/D clock cycles for an external input A/D conversion which is defined as t_{ADC} are necessary.

$$\text{Maximum single A/D conversion rate} = \text{A/D clock period} / 16$$

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



Summary of A/D Conversion Steps

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

- Step 1
Select the required A/D conversion clock by correctly programming bits SACKS2~SACKS0 in the SADC1 register.
- Step 2
Enable the A/D 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 SAINS bit fields.
Select the external channel input to be converted, go to Step 4.
Select the internal analog signal to be converted, go to Step 5.
- Step 4
If the A/D input signal comes from the external channel input selected by configuring the SAINS bit field, 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 SACS bit field. After this step, go to Step 6.
- Step 5
Before the A/D input signal is selected to come from the internal analog signal by configuring the SAINS bit field, the external input pin must be disabled by properly configuring the relevant pin-shared function control bits. The desired internal analog signal then can be selected by configuring the SAINS bit field. After this step, go to Step 6.
- Step 6
Select the reference voltage source by configuring the SAVRS1~SAVRS0 bits in the SADC2 register. Select the PGA input signal and the desired PGA gain if the PGA output voltage is selected as the A/D converter reference voltage.
- Step 7
Select A/D converter output data format by setting the ADRFS bit in the SADC0 register.
- Step 8
If A/D conversion interrupt is used, the interrupt control registers must be correctly configured to ensure the A/D interrupt function is active. The master interrupt control bit, EMI, and the A/D conversion interrupt control bit, ADE, must both be set high in advance.

- Step 9

The A/D conversion procedure can now be initialized by setting the START bit from low to high and then low again.

- Step 10

If A/D conversion is in progress, the ADBZ flag will be set high. After the A/D conversion process is complete, the ADBZ flag will go low and then the output data can be read from SADOH and SADOL registers.

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

Programming Considerations

During microcontroller operations where the A/D converter is not being used, the A/D internal circuitry can be switched off to reduce power consumption, by clearing the ADCEN bit 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/Os, 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 these devices contain a 12-bit A/D converter, its full-scale converted digitised value is equal to FFFH. Since the full-scale analog input value is equal to the actual A/D converter reference voltage, V_{REF} , this gives a single bit analog input value of V_{REF} divided by 4096.

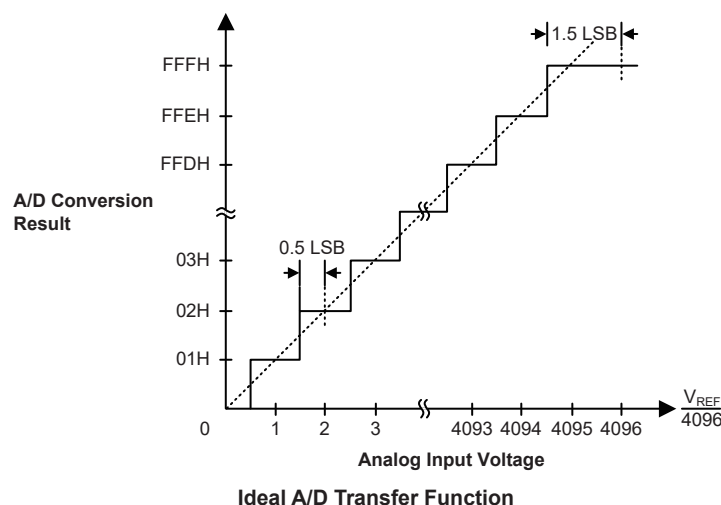
$$1 \text{ LSB} = V_{REF} \div 4096$$

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

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

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

Note that here the V_{REF} voltage is the actual A/D converter reference voltage determined by the SAVRS bit field.



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

```

clr ADE                ; disable ADC interrupt
mov a,03H
mov SADC1,a            ; select fsys/8 as A/D converter clock
mov a,00H
mov SADC2,a;          ; select VDD as A/D converter reference voltage source
mov a,0CH              ; setup PAS1 to configure pin AN0
mov PAS1,a
mov a,20H
mov SADC0,a            ; enable A/D converter and select AN0 as 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

```

clr ADE                ; disable ADC interrupt
mov a,03H
mov SADC1,a            ; select fsys/8 as A/D converter clock
mov a,00H
mov SADC2,a            ; select VDD as A/D converter reference voltage source
mov a,0CH              ; setup PAS1 to configure pin AN0
mov PAS1,a
mov a,20H
mov SADC0,a            ; enable A/D converter and select AN0 as 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 interrupt service routine
ADC_ISR:

```

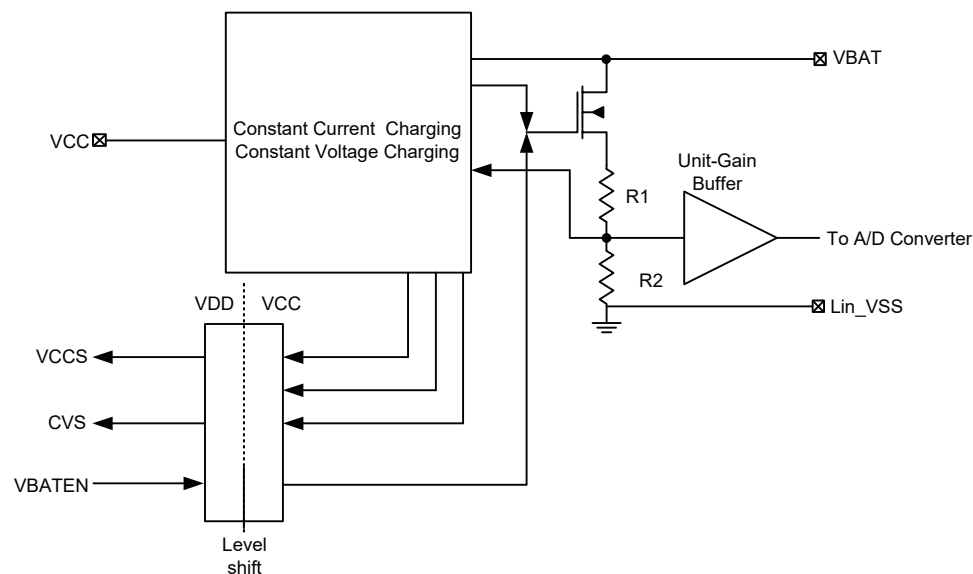
```

mov acc_stack,a      ; save ACC to user defined memory
mov a,STATUS
mov status_stack,a   ; save STATUS to user defined memory
:
:
mov a,SADOL           ; read low byte conversion result value
mov SADOL_buffer,a   ; save result to user defined register
mov a,SADOH           ; read high byte conversion result value
mov SADOH_buffer,a   ; save result to user defined register
:
:
EXIT_INT_ISR:
mov a,status_stack
mov STATUS,a         ; restore STATUS from user defined memory
mov a,acc_stack      ; restore ACC from user defined memory
reti

```

Linear Charger

These devices include a Linear Charger. The LCS register controls the Linear Charger module to work in three modes.



Linear Charger Structure

Part No.	Linear Charger CV Voltage
BS45F5830	4.2V
BS45F5831	4.35V
BS45F5832	4.2V
BS45F5833	4.35V

Linear Charger Control Register

• LCS Register

Bit	7	6	5	4	3	2	1	0
Name	VCCS	—	CVS	VBATEN	D3	D2	D1	D0
R/W	R	—	R	R/W	R/W	R/W	R/W	R/W
POR	x	—	x	0	0	0	0	0

“x”: unknown

- Bit 7 **VCCS**: Linear charger input voltage V_{CC} status
0: Input voltage V_{CC} is abnormal
1: Input voltage V_{CC} is within $(V_{BAT}+0.1V) \sim 5.5V$
- Bit 6 Unimplemented, read as “0”
- Bit 5 **CVS**: The battery charging full flag
0: The battery is not full
1: The battery has been full
- Bit 4 **VBATEN**: Voltage divider resistance control
0: Off, the A/D Conveter will read a voltage of 0V
1: On, the A/D Conveter will read a $V_{BAT}/2$ as internal signal
- Bit 3~0 **D3~D0**: The charging current I_{CC} setting when the constant current mode
0000: 40mA
0001: 60mA
0010: 80mA
0011: 100mA
0100: 120mA
0101: 140mA
0110: 160mA
0111: 180mA
1000: 200mA
1001: 220mA
1010: 240mA
1011: 260mA
1100: 280mA
1101: 300mA
1110: 360mA
1111: 400mA

Note: Before the MCU is powered on reset, the bit setting is invalid, and the I_{CC} value is 40mA in constant current mode.

Functional Description

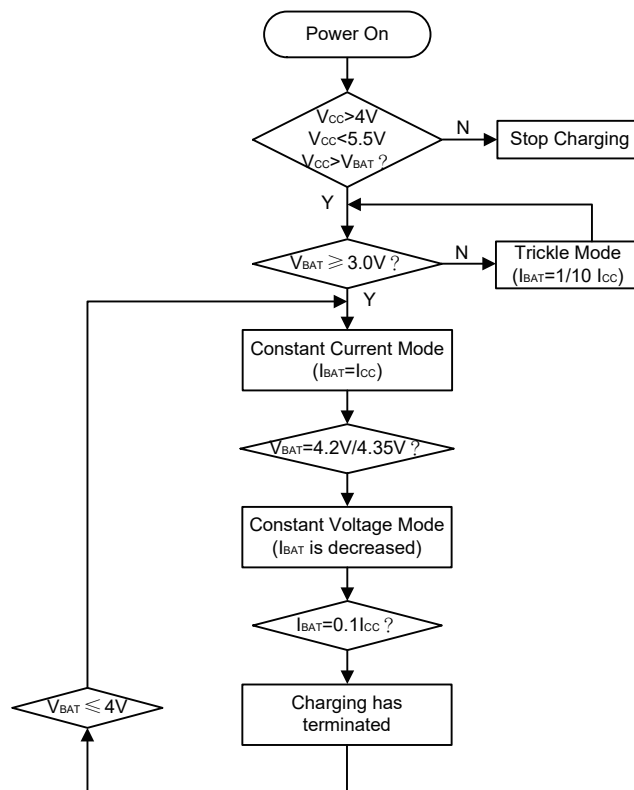
There are three charge control modes for the Linear Charger, trickle mode, constant current charge mode and constant voltage charge mode. The required charge voltage and current for these modes differ according to different user designs. These charge modes are described below.

Trickle Mode: used for completely discharged batteries which have a battery voltage of less than 3V. In this mode the battery will be pre-charged using a typical 0.1 constant current I_{CC} during this first stage of charging.

Constant Current Charging Mode: when the battery voltage is equal to or larger than 3V and less than 4.2V or 4.35V, the charging current will then switch to a typical constant current I_{CC} during this second charging stage. The charging current is determined by the D3~D0 bits in the LCS register, and is within the range of 40mA to 400mA.

Constant Voltage Charging Mode: once the battery voltage has reached 4.2V or 4.35V, it will be charged using a constant voltage during this third stage. The charging voltage should be fixed at 4.2V

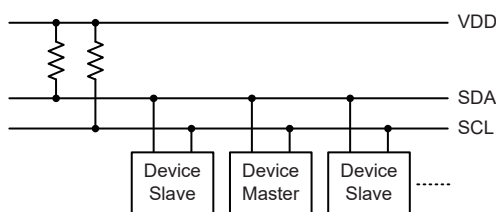
or 4.35V with a tolerance within 1%. The charge current gradually decreases as the constant voltage charge time increases. Typically, the constant voltage charge stage is completed when the charge current reduces to a value of less than $0.1 I_{CC}$. At this time the battery is full, and the CVS bit will be set high. After charging, the battery voltage will always be monitored. When the battery voltage is equal to or less than 4.0V, the battery will be recharged and the next charging cycle will be entered.



Linear Charger Flow Chart

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

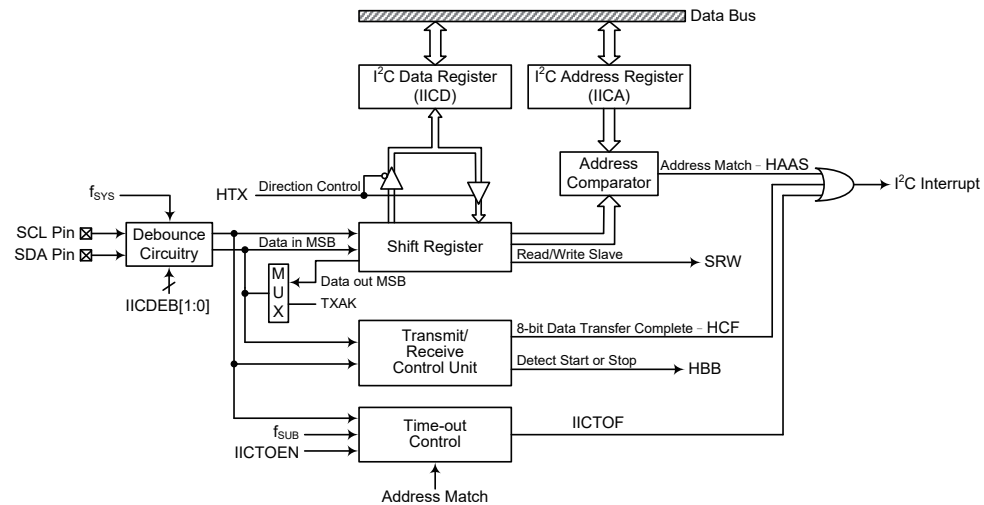
I²C Interface Operation

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.

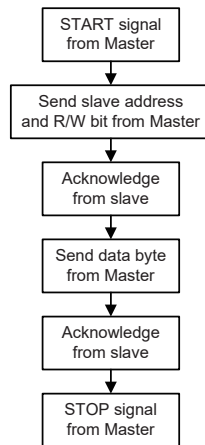
When two devices communicate with each other on the bidirectional I²C bus, one is known as the master device and one as the slave device. Both master and slave can transmit and receive data, however, it is the master device that has overall control of the bus. For the device, which only operates in slave mode, there are two methods of transferring data on the I²C bus, the slave transmit mode and the slave receive mode.

It is suggested that the device shall not enter the IDLE or SLEEP mode during the I²C communication is in progress.

If the related pin is configured as SDA or SCL function, the pin must be configured to open-collect input/output port and its pull-high function can be enabled by programming the related pull-high control register.



I²C Block Diagram



I²C Interface Operation

The IICDEB1 and IICDEB0 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 the I²C Standard 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.

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

I²C Minimum f_{SYS} Frequency Requirements

I²C Registers

There are three control registers associated with the I²C bus, IICC0, IICC1 and IICTOC, one address register IICA and one data register, IICD.

Register Name	Bit							
	7	6	5	4	3	2	1	0
IICC0	—	—	—	—	IICDEB1	IICDEB0	IICEN	—
IICC1	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
IICD	D7	D6	D5	D4	D3	D2	D1	D0
IICA	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	—
IICTOC	IICTOEN	IICTOF	IICTOS5	IICTOS4	IICTOS3	IICTOS2	IICTOS1	IICTOS0

I²C Register List

I²C Data Register

The IICD 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 IICD register. After the data is received from the I²C bus, the device can read it from the IICD register. Any transmission or reception of data from the I²C bus must be made via the IICD register.

• IICD 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 IICA register is the location where the 7-bit slave address of the slave device is stored. Bits 7~1 of the IICA 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 IICA register, the slave device will be selected.

• IICA Register

Bit	7	6	5	4	3	2	1	0
Name	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	—
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 7~1 **IICA6~IICA0**: I²C slave address
IICA6~IICA0 is the I²C slave address bit 6~bit 0.
- Bit 0 Unimplemented, read as “0”

I²C Control Registers

There are three control registers for the I²C interface, IICC0, IIC1 and IICTOC. The IICC0 register is used to control the enable/disable function and to set the data transmission clock frequency. The IICC1 register contains the relevant flags which are used to indicate the I²C communication status. Another register, IICTOC, is used to control the I²C time-out function and will be described in the I²C Time-out section.

• IICC0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	IICDEB1	IICDEB0	IICEN	—
R/W	—	—	—	—	R/W	R/W	R/W	—
POR	—	—	—	—	0	0	0	—

- Bit 7~4 Unimplemented, read as “0”
- Bit 3~2 **IICDEB1~IICDEB0**: I²C Debounce Time Selection
00: No debounce
01: 2 system clock debounce
1x: 4 system clock debounce
- Bit 1 **IICEN**: I²C Enable Control
0: Disable
1: Enable

The bit is the overall on/off control for the I²C interface. When the IICEN 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 IICEN 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 Unimplemented, read as “0”

• IICC1 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 the I²C master and then the 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 to one. Third, users read the 1st data byte from the IICD 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 the 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

The HAAS flag is the address match flag. This flag is used to determine if the slave device address is the same as the master transmit address. If the addresses match then this bit will be high, if there is no match then the flag will be low.

Bit 5 **HBB:** I²C Bus busy flag

- 0: I²C Bus is not busy
- 1: I²C Bus is busy

The HBB flag is the I²C busy flag. This flag will be “1” when the I²C bus is busy which will occur when a START signal is detected. The flag will be set to “0” when the bus is free which will occur when a STOP signal is detected.

Bit 4 **HTX:** I²C slave device is transmitter or receiver selection

- 0: Slave device is the receiver
- 1: Slave device is the transmitter

Bit 3 **TXAK:** I²C Bus transmit acknowledge flag

- 0: Slave send acknowledge flag
- 1: Slave do not send acknowledge flag

The TXAK bit is the transmit acknowledge flag. After the slave device receipt of 8 bits of data, this bit will be transmitted to the bus on the 9th clock from the slave device. The slave device must always set TXAK bit to “0” before further data is received.

Bit 2 **SRW:** I²C Slave Read/Write flag

- 0: Slave device should be in receive mode
- 1: Slave device should be in transmit mode

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

Bit 1 **IAMWU:** I²C Address Match Wake-up control

- 0: Disable
- 1: Enable

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

Bit 0 **RXAK:** I²C Bus Receive acknowledge flag

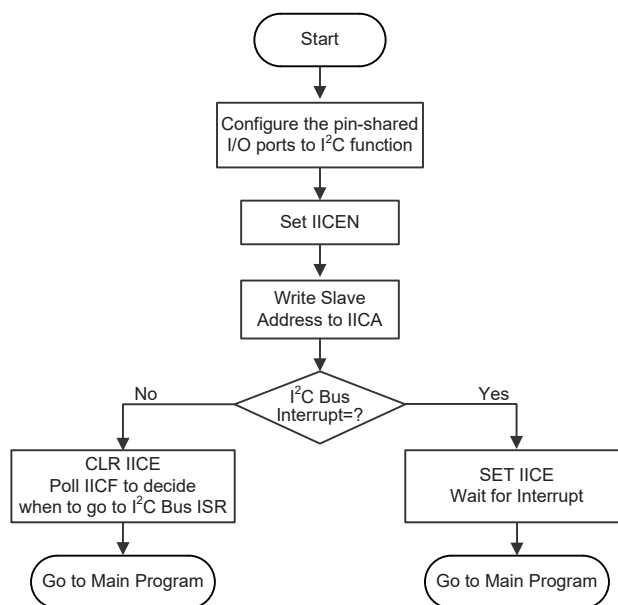
- 0: Slave receive acknowledge flag
- 1: Slave does not receive acknowledge flag

The RXAK flag is the receiver acknowledge flag. When the RXAK flag is “0”, it means that a acknowledge signal has been received at the 9th clock, after 8 bits of data have been transmitted. When the slave device in the transmit mode, the slave device checks the RXAK flag to determine if the master receiver wishes to receive the next byte. The slave transmitter will therefore continue sending out data until the RXAK flag is “1”. When this occurs, the slave transmitter will release the SDA line to allow the master to send a STOP signal to release the I²C Bus.

I²C Bus Communication

Communication on the I²C bus requires four separate steps, a START signal, a slave device address transmission, a data transmission and finally a STOP signal. When a START signal is placed on the I²C bus, all devices on the bus will receive this signal and be notified of the imminent arrival of data on the bus. The first seven bits of the data will be the slave address with the first bit being the MSB. If the address of the slave device matches that of the transmitted address, the HAAS bit in the IICC1 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 IICTOF 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
Configure the related pin-shared I/O pins as I²C pin functions. Set the IICEN bits in the IICCC0 register to “1” to enable the I²C bus.
- Step 2
Write the slave address of the device to the I²C bus address register IICA.
- Step 3
Set the IICE 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 IICC1 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 ICTOF 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 IICD register, or in the receive mode where it must implement a dummy read from the IICD 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 configured 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 configured to read data from the I²C bus as a receiver.

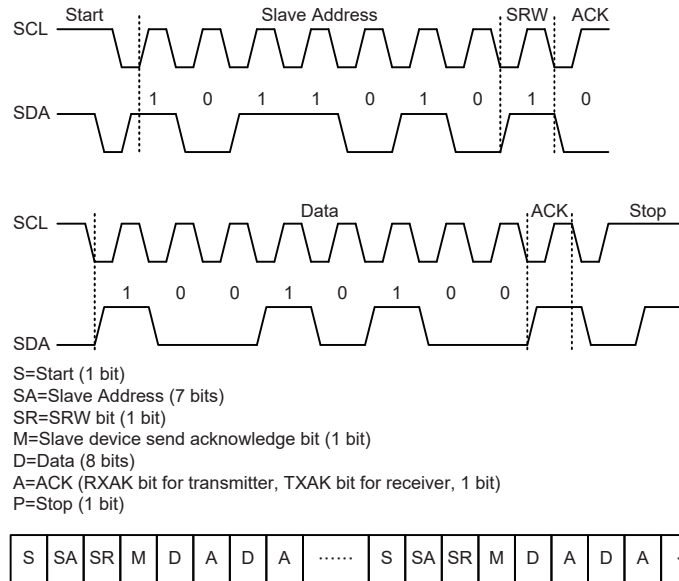
I²C Bus Slave Address Acknowledge Signal

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 configured to be a transmitter so the HTX bit in the IICC1 register should be set to “1”. If the SRW flag is low, then the microcontroller slave device should be configured as a receiver and the HTX bit in the IICC1 register should be set to “0”.

I²C Bus Data and Acknowledge Signal

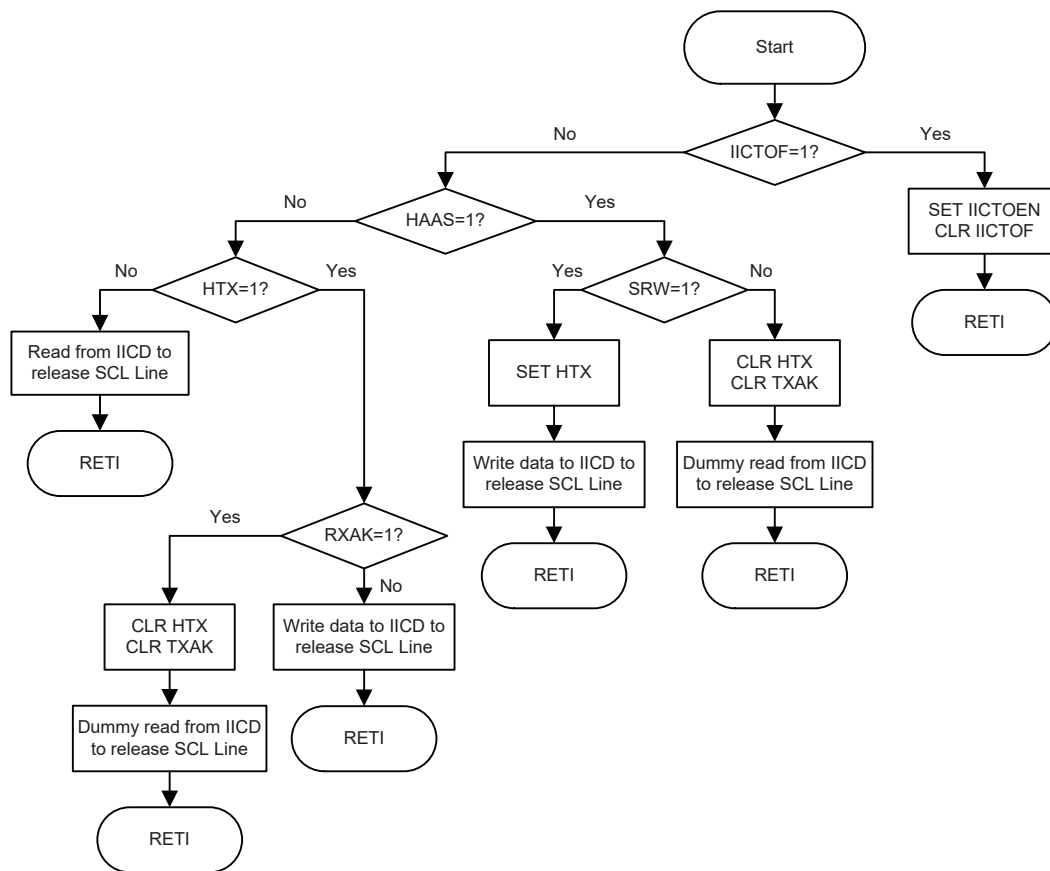
The transmitted data is 8-bit wide and is transmitted after the slave device has acknowledged receipt of its slave address. The order of serial bit transmission is the MSB first and the LSB last. After receipt of 8 bits of data, the receiver must transmit an acknowledge signal, level “0”, before it can 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 IICD register. If configured as a transmitter, the slave device must first write the data to be transmitted into the SIMD register. If configured as a receiver, the slave device must read the transmitted data from the IICD 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 configured as a transmitter will check the RXAK bit in the IICC1 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.



I²C Communication Timing Diagram

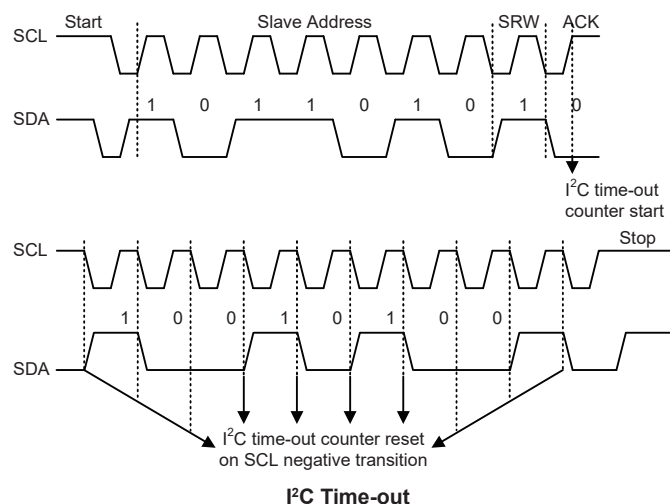
Note: When a slave address is matched, the device must be placed in either the transmit mode and then write data to the IICD register, or in the receive mode where it must implement a dummy read from the IICD register to release the SCL line.



I²C Bus ISR Flow Chart

I²C Time-out Control

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



When an I²C time-out counter overflow occurs, the counter will stop and the IICTOEN bit will be cleared to zero and the IICTOF 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 I ² C Time-out
IICD, IICA, IICC0	No change
IICC1	Reset to POR condition

I²C Registers after Time-out

The IICTOF flag can be cleared by the application program. There are 64 time-out periods which can be selected using IICTOS bit field in the IICTOC register. The time-out time is given by the formula: $((1 \sim 64) \times 32) / f_{\text{SUB}}$. This gives a time-out period which ranges from about 1ms to 64ms.

• IICTOC Register

Bit	7	6	5	4	3	2	1	0
Name	IICTOEN	IICTOF	IICTOS5	IICTOS4	IICTOS3	IICTOS2	IICTOS1	IICTOS0
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 **IICTOEN**: I²C Time-out control
 0: Disable
 1: Enable

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

This bit is set high when time-out occurs and can only be cleared to zero by application program.

Bit 5~0 **IICTOS5~IICTOS0**: I²C Time-out period selection
I²C time-out clock source is $f_{SUB}/32$.
I²C time-out time is equal to $(IICTOS[5:0]+1) \times (32/f_{SUB})$.

Low Voltage Detector – LVD

These devices have a Low Voltage Detector function, also known as LVD. This enabled the devices to monitor the power supply voltage, V_{DD} , and provide a warning signal should it fall below a certain level. This function may be especially useful in battery applications where the supply voltage will gradually reduce as the battery ages, as it allows an early warning battery low signal to be generated. The Low Voltage Detector also has the capability of generating an interrupt signal.

LVD Register

The Low Voltage Detector function is controlled using a single register with the name LVDC. Three bits in this register, VLVD2~VLVD0, are used to select one of eight fixed voltages below which a low voltage condition will be determined. A low voltage condition is indicated when the LVDO bit is set. If the LVDO bit is low, this indicates that the V_{DD} voltage is above the preset low voltage value. The LVDEN bit is used to control the overall on/off function of the low voltage detector. Setting the bit high will enable the low voltage detector. Clearing the bit to zero will switch off the internal low voltage detector circuits. As the low voltage detector will consume a certain amount of power, it may be desirable to switch off the circuit when not in use, an important consideration in power sensitive battery powered applications.

• LVDC Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	LVDO	LVDEN	VBGEN	VLVD2	VLVD1	VLVD0
R/W	—	—	R	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 Unimplemented, read as “0”

Bit 5 **LVDO**: LVD output flag
0: No Low Voltage Detected
1: Low Voltage Detected

Bit 4 **LVDEN**: Low voltage detector enable control
0: Disable
1: Enable

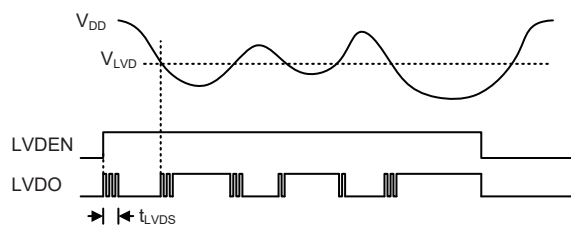
Bit 3 **VBGEN**: Bandgap voltage output enable control
0: Disable
1: Enable

Note that the Bandgap circuit is enabled when the LVD or LVR function is enabled or when the VBGEN bit is set high.

Bit 2~0 **VLVD2~VLVD0**: LVD voltage selection
000: 2.0V
001: 2.2V
010: 2.4V
011: 2.7V
100: 3.0V
101: 3.3V
110: 3.6V
111: 4.0V

LVD Operation

The Low Voltage Detector function operates by comparing the power supply voltage, V_{DD} , with a pre-specified voltage level stored in the LVDC register. This has a range of between 2.0V and 4.0V. When the power supply voltage, V_{DD} , falls below this pre-determined value, the LVDO bit will be set high indicating a low power supply voltage condition. When the device is in the SLEEP mode, the low voltage detector will be disabled even if the LVDEN bit is high. After enabling the Low Voltage Detector, a time delay t_{LVDS} should be allowed for the circuitry to stabilise before reading the LVDO bit. Note also that as the V_{DD} voltage may rise and fall rather slowly, at the voltage nears that of V_{LVD} , there may be multiple bit LVDO transitions.



LVD Operation

The Low Voltage Detector also has its own interrupt, providing an alternative means of low voltage detection, in addition to polling the LVDO bit. The interrupt will only be generated after a delay of t_{LVD} after the LVDO bit has been set high by a low voltage condition. In this case, the LVF interrupt request flag will be set, causing an interrupt to be generated if V_{DD} falls below the preset LVD voltage. This will cause the device to wake-up from the IDLE Mode, however if the Low Voltage Detector wake up function is not required then the LVF flag should be first set high before the device enters the IDLE Mode.

Interrupts

Interrupts are an important part of any microcontroller system. When an external event or an internal function such as a Timer Module or an A/D converter requires microcontroller attention, their corresponding interrupt will enforce a temporary suspension of the main program allowing the microcontroller to direct attention to their respective needs. These devices contain several external interrupt and internal interrupt functions. The external interrupts are generated by the action of the external INT pin, while the internal interrupts are generated by various internal functions such as the Touch Keys, TMs and the A/D converter, etc.

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~INTC2 registers which setup the primary interrupts, the second is the MFI0~MFI2 registers 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	—	—
INT Pin	INTE	INTF	—
I ² C	IICE	IICF	—
Time Base	TBnE	TBnF	n=0 or 1
A/D Converter	ADE	ADF	—
EEPROM	DEE	DEF	—
LVD	LVE	LVF	—
Multi-function	MFnE	MFnF	n=0~2
Touch key TKRCOV	TKRCOVE	TKRCOVF	—
Touch key module TKTH	TKTHE	TKTHF	—
STM	STMPE	STMPF	—
	STMAE	STMAF	—
CTM	CTMPE	CTMPF	—
	CTMAE	CTMAF	

Interrupt Register Bit Naming Conventions

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	—	—	INTS1	INTS0
INTC0	—	IICF	MF0F	INTF	IICE	MF0E	INTE	EMI
INTC1	MF2F	MF1F	TB1F	TB0F	MF2E	MF1E	TB1E	TB0E
INTC2	—	ADF	DEF	LVF	—	ADE	DEE	LVE
MF10	—	—	TKTHF	TKRCOVF	—	—	TKTHE	TKRCOVE
MF11	—	—	CTMAF	CTMPF	—	—	CTMAE	CTMPE
MF12	—	—	STMAF	STMPF	—	—	STMAE	STMPE

Interrupt Register List

• **INTEG Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	INTS1	INTS0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as “0”

Bit 1~0 **INTS1~INTS0**: Interrupt edge control for INT pin

00: Disable

01: Rising edge

10: Falling edge

11: Rising and falling edges

• **INTC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	IICF	MF0F	INTF	IICE	MF0E	INTE	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 **IICF**: I²C interrupt request flag

0: No request

1: Interrupt request

Bit 5	MF0F : Multi-function interrupt 0 request flag 0: No request 1: Interrupt request
Bit 4	INTF : INT interrupt request flag 0: No request 1: Interrupt request
Bit 3	IICE : I ² C interrupt control 0: Disable 1: Enable
Bit 2	MF0E : Multi-function interrupt 0 control 0: Disable 1: Enable
Bit 1	INTE : INT interrupt control 0: Disable 1: Enable
Bit 0	EMI : Global interrupt control 0: Disable 1: Enable

• INTC1 Register

Bit	7	6	5	4	3	2	1	0
Name	MF2F	MF1F	TB1F	TB0F	MF2E	MF1E	TB1E	TB0E
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	MF2F : Multi-function interrupt 2 request flag 0: No request 1: Interrupt request
Bit 6	MF1F : Multi-function interrupt 1 request flag 0: No request 1: Interrupt request
Bit 5	TB1F : Time Base 1 interrupt request flag 0: No request 1: Interrupt request
Bit 4	TB0F : Time Base 0 interrupt request flag 0: No request 1: Interrupt request
Bit 3	MF2E : Multi-function interrupt 2 control 0: Disable 1: Enable
Bit 2	MF1E : Multi-function interrupt 1 control 0: Disable 1: Enable
Bit 1	TB1E : Time Base 1 interrupt control 0: Disable 1: Enable
Bit 0	TB0E : Time Base 0 interrupt control 0: Disable 1: Enable

• **INTC2 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	ADF	DEF	LVF	—	ADE	DEE	LVE
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

- Bit 7 Unimplemented, read as “0”
- Bit 6 **ADF**: A/D Converter interrupt request flag
0: No request
1: Interrupt request
- Bit 5 **DEF**: Data EEPROM interrupt request flag
0: No request
1: Interrupt request
- Bit 4 **LVF**: LVD interrupt request flag
0: No request
1: Interrupt request
- Bit 3 Unimplemented, read as “0”
- Bit 2 **ADE**: A/D Converter interrupt control
0: Disable
1: Enable
- Bit 1 **DEE**: Data EEPROM interrupt control
0: Disable
1: Enable
- Bit 0 **LVE**: LVD interrupt control
0: Disable
1: Enable

• **MFIO Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	TKTHF	TKRCOVF	—	—	TKTHE	TKRCOVE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

- Bit 7~6 Unimplemented, read as “0”
- Bit 5 **TKTHF**: Touch key module TKTH interrupt request flag
0: No request
1: Interrupt request
- Bit 4 **TKRCOVF**: Touch key TKRCOV interrupt request flag
0: No request
1: Interrupt request
- Bit 3~2 Unimplemented, read as “0”
- Bit 1 **TKTHE**: Touch key module TKTH interrupt control
0: Disable
1: Enable
- Bit 0 **TKRCOVE**: Touch key TKRCOV interrupt control
0: Disable
1: Enable

• MFI1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	CTMAF	CTMPF	—	—	CTMAE	CTMPE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 Unimplemented, read as “0”

Bit 5 **CTMAF**: CTM Comparator A match interrupt request flag
 0: No request
 1: Interrupt request

Bit 4 **CTMPF**: CTM Comparator P match interrupt request flag
 0: No request
 1: Interrupt request

Bit 3~2 Unimplemented, read as “0”

Bit 1 **CTMAE**: CTM Comparator A match interrupt control
 0: Disable
 1: Enable

Bit 0 **CTMPE**: CTM Comparator P match interrupt control
 0: Disable
 1: Enable

• MFI2 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	STMAF	STMPF	—	—	STMAE	STMPE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 Unimplemented, read as “0”

Bit 5 **STMAF**: STM Comparator A match interrupt request flag
 0: No request
 1: Interrupt request

Bit 4 **STMPF**: STM Comparator P match interrupt request flag
 0: No request
 1: Interrupt request

Bit 3~2 Unimplemented, read as “0”

Bit 1 **STMAE**: STM Comparator A match interrupt control
 0: Disable
 1: Enable

Bit 0 **STMPE**: STM Comparator P match interrupt control
 0: Disable
 1: Enable

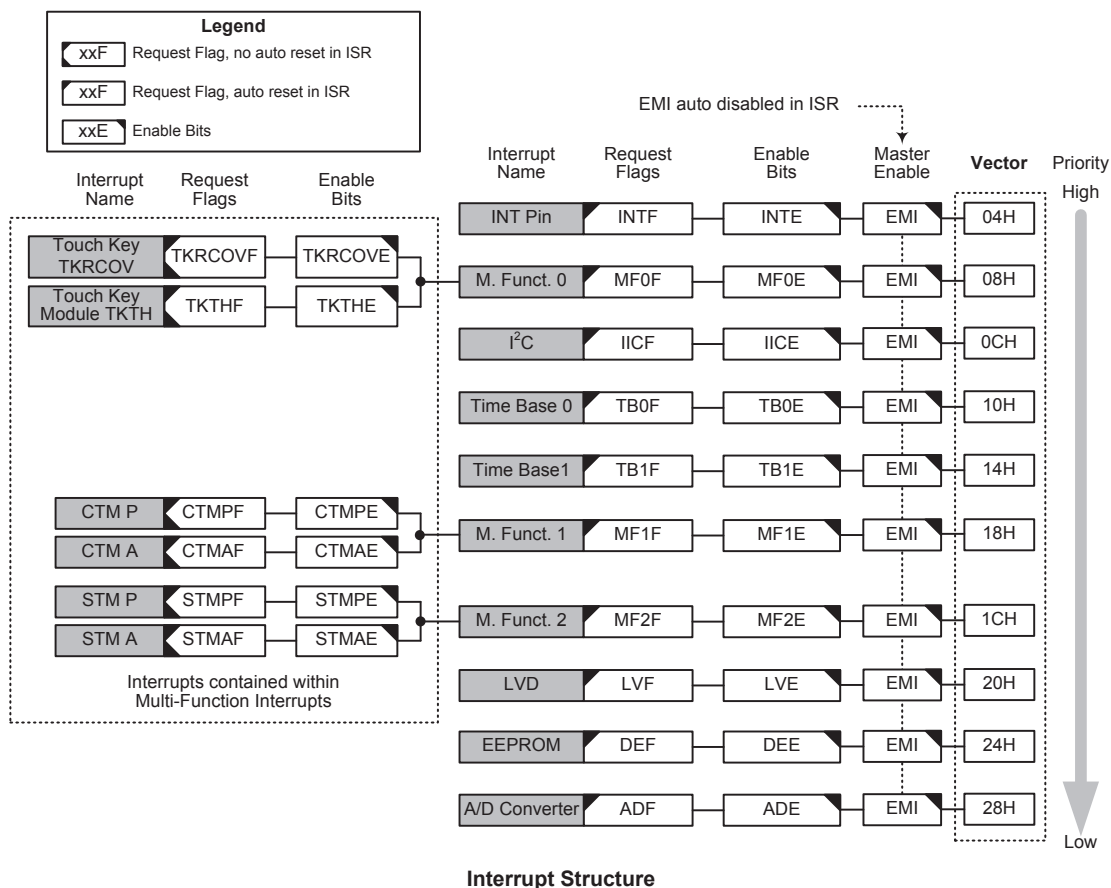
Interrupt Operation

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.

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. Once an interrupt subroutine is serviced, all the other interrupts will be blocked, as the global interrupt enable bit, EMI bit will be cleared automatically. This will prevent any further interrupt nesting from occurring. However, if other interrupt requests occur during this interval, although the interrupt will not be immediately serviced, the request flag will still be recorded.

If an interrupt requires immediate servicing while the program is already in another interrupt service routine, the EMI bit should be set after entering the routine, to allow interrupt nesting. If the stack is full, the interrupt request will not be acknowledged, even if the related interrupt is enabled, until the Stack Pointer is decremented. If immediate service is desired, the stack must be prevented from becoming full. In case of simultaneous requests, the accompanying diagram shows the priority that is applied. All of the interrupt request flags when set will wake-up the device if it is in SLEEP or IDLE Mode, however to prevent a wake-up from occurring the corresponding flag should be set before the device is in SLEEP or IDLE Mode.



External Interrupt

The external interrupt is controlled by signal transitions on the pin INT. An external interrupt request will take place when the external interrupt request flag, INTF, is set, which will occur when a transition, whose type is chosen by the edge select bits, appears on the external interrupt pin. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and respective external interrupt enable bit, INTE, must first be set. Additionally the correct interrupt edge type must be selected using the INTEG register to enable the external interrupt function and to choose the trigger edge type. As the external interrupt pin is pin-shared with I/O pin, they can only be configured as external interrupt pin if its external interrupt enable bit in the corresponding interrupt register has been set and the external interrupt pin is selected by the corresponding pin-shared function selection bits. The pin must also be setup as an input by setting the corresponding bit in the port control register. When the interrupt is enabled, the stack is not full and the correct transition type appears on the external interrupt pin, a subroutine call to the external interrupt pin, will take place. When the interrupt is serviced, the external interrupt request flag, INTF, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts. Note that any pull-high resistor selections on the external interrupt pin will remain valid even if the pin is used as an external interrupt input.

The INTEG register is used to select the type of active edge that will trigger the external interrupt. A choice of either rising or falling or both edge types can be chosen to trigger an external interrupt. Note that the INTEG register can also be used to disable the external interrupt function.

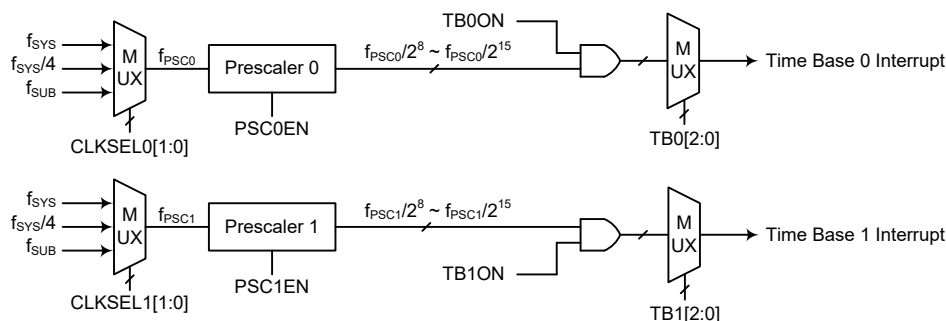
I²C Interface Interrupt

The I²C Interface Interrupt will take place when the I²C Interrupt request flag, IICF, is set, which occurs when a byte of data has been received or transmitted by the I²C interface, or an I²C slave address match occurs, or an I²C bus time-out occurs. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the I²C Interface Interrupt enable bit, IICE, must first be set. When the interrupt is enabled, the stack is not full and any of the above described situations occurs, a subroutine call to the respective Interrupt vector, will take place. When the interrupt is serviced, the I²C Interface Interrupt request flag, IICF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

Time Base Interrupts

The function of the Time Base Interrupts is to provide regular time signal in the form of an internal interrupt. They are controlled by the overflow signals from their respective timer functions. When these happens their respective interrupt request flags, 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 bits, 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. Its clock source, f_{PSC0} or f_{PSC1} , originates from the internal clock source f_{SYS} , $f_{SYS}/4$ or f_{SUB} and then passes through a divider, the division ratio of which is selected by programming the appropriate bits in the TB0C and TB1C registers to obtain longer interrupt periods whose value ranges. The clock source which in turn controls the Time Base interrupt period is selected using the CLKSEL0[1:0] and CLKSEL1[1:0] bits in the PSC0R and PSC1R registers respectively.



Time Base Interrupts

• PSC0R Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	PSC0EN	CLKSEL01	CLKSEL00
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 Unimplemented, read as “0”

Bit 2 **PSC0EN**: Prescaler 0 clock enabled control
0: Disable
1: Enable

This PSC0EN bit is used for Prescaler 0 clock enabled or disabled control. When the Prescale 0 clock is disabled, it can reduce extra power consumption.

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

• PSC1R Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	PSC1EN	CLKSEL11	CLKSEL10
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 Unimplemented, read as “0”

Bit 2 **PSC1EN**: Prescaler 1 clock enabled control
 0: Disable
 1: Enable

This PSC1EN bit is used for Prescaler 1 clock enabled or disabled control. When the Prescale 1 clock is disabled, it can reduce extra power consumption.

Bit 1~0 **CLKSEL11~CLKSEL10**: Prescaler 1 clock source f_{PSC1} selection
 00: f_{SYS}
 01: $f_{SYS}/4$
 10: f_{SUB}
 11: Reserved

• TB0C Register

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

Bit 7 **TB0ON**: Time Base 0 Control
 0: Disable
 1: Enable

Bit 6~3 Unimplemented, read as “0”

Bit 2~0 **TB02~TB00**: Select Time Base 0 Time-out Period
 000: $2^8/f_{PSC0}$
 001: $2^9/f_{PSC0}$
 010: $2^{10}/f_{PSC0}$
 011: $2^{11}/f_{PSC0}$
 100: $2^{12}/f_{PSC0}$
 101: $2^{13}/f_{PSC0}$
 110: $2^{14}/f_{PSC0}$
 111: $2^{15}/f_{PSC0}$

• TB1C Register

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

Bit 7 **TB1ON**: Time Base 1 Control
 0: Disable
 1: Enable

Bit 6~3 Unimplemented, read as “0”

Bit 2~0	TB12~TB10: Select Time Base 1 Time-out Period
	000: $2^8/f_{PSC1}$
	001: $2^9/f_{PSC1}$
	010: $2^{10}/f_{PSC1}$
	011: $2^{11}/f_{PSC1}$
	100: $2^{12}/f_{PSC1}$
	101: $2^{13}/f_{PSC1}$
	110: $2^{14}/f_{PSC1}$
	111: $2^{15}/f_{PSC1}$

A/D Converter Interrupt

An A/D Converter Interrupt request will take place when the A/D Converter Interrupt request flag, ADF, is set, which occurs when the A/D conversion process finishes. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and A/D Interrupt enable bit, ADE, must first be set. When the interrupt is enabled, the stack is not full and the A/D conversion process has ended, a subroutine call to the A/D Interrupt vector, will take place. When the A/D Converter 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.

LVD Interrupt

An LVD Interrupt request will take place when the LVD Interrupt request flag, LVF, is set, which occurs when the Low Voltage Detector function detects a low power supply voltage. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the Low Voltage Interrupt enable bit, LVE, must first be set. When the interrupt is enabled, the stack is not full and a low voltage condition occurs, a subroutine call to the LVD Interrupt vector, will take place. When the Low Voltage Interrupt is serviced, the LVF flag will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

EEPROM Interrupt

An EEPROM Interrupt request will take place when the EEPROM Interrupt request flag, DEF, is set, which occurs when an EEPROM Write cycle ends. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and EEPROM Interrupt enable bit, DEE, must first be set. When the interrupt is enabled, the stack is not full and an EEPROM Write cycle ends, a subroutine call to the EEPROM Interrupt vector will take place. When the EEPROM Interrupt is serviced, the EEPROM Interrupt flag, DEF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

Multi-function Interrupts

Within these devices there are up to three Multi-function interrupts. Unlike the other independent interrupts, these interrupts have no independent source, but rather are formed from other existing interrupt sources, namely the touch key TKRCOV interrupt, touch key module TKTH interrupt and TM Interrupts.

A Multi-function interrupt request will take place when any of the Multi-function interrupt request flags, MFnF are set. The Multi-function interrupt flags will be set when any of their included functions generate an interrupt request flag. To allow the program to branch to its respective interrupt vector address, when the Multi-function interrupt is enabled and the stack is not full, and either one of the interrupts contained within each of Multi-function interrupt occurs, a subroutine call to one of the Multi-function interrupt vectors 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 flags will be automatically reset when the interrupt is serviced, the request flags from the original source of the Multi-function interrupts will not be automatically reset and must be manually reset by the application program.

Touch Key TKRCOV Interrupt

The Touch Key TKRCOV Interrupt is contained within the Multi-function Interrupt. An Touch Key TKRCOV Interrupt request will take place when the Touch Key TKRCOV Interrupt request flag, TKRCOVF, is set, which occurs when the touch key time slot counter overflow flag is set high. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, Touch Key TKRCOV Interrupt enable bit, TKRCOVE, and associated Multi-function interrupt enable bit, must first be set. When the interrupt is enabled, the stack is not full and the touch key time slot counter overflows or all the scan operations are completed, a subroutine call to the Multi-function Interrupt vector, will take place. When the Touch Key TKRCOV Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared. As the TKRCOVF flag will not be automatically cleared, it has to be cleared by the application program.

Touch Key Module TKTH Interrupt

The Touch Key Module TKTH Interrupt is contained within the Multi-function Interrupt. An Touch Key Module TKTH Interrupt request will take place when the Touch Key Module TKTH Interrupt request flag, TKTHF, is set, which occurs when the Touch Key Module 16-bit C/F counter is less than the lower threshold if M0KnTHS=0, or larger than the upper threshold if M0KnTHS=1. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, Touch Key Module TKTH Interrupt enable bit, TKTHE, and associated Multi-function interrupt enable bit, must first be set. When the interrupt is enabled, the stack is not full and any of the above described threshold comparison conditions occurs, a subroutine call to the Multi-function Interrupt vector, will take place. When the Touch Key Module TKTH Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared. As the TKTHF flag will not be automatically cleared, it has to be cleared by the application program.

TM Interrupts

The Compact and Standard Type TMs each have two interrupts, one comes from the comparator A match situation and the other comes from the comparator P match situation. All of the TM interrupts are contained within the Multi-function Interrupts. 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.

To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, respective TM Interrupt enable bit, and relevant Multi-function Interrupt enable bit, MFnE, must first be set. When the interrupt is enabled, the stack is not full and a TM comparator match situation occurs, a subroutine call to the relevant Multi-function Interrupt vector locations, will take place. When the TM 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 TM 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 generated when an interrupt request flag changes from low to high and is independent of whether the interrupt is enabled or not. Therefore, even though the device is in the SLEEP or IDLE Mode and its system oscillator stopped, situations such as external edge transition on the external interrupt pin or a low power supply voltage may cause their respective interrupt flag to be set high and consequently generate an interrupt. Care must therefore be taken if spurious wake-up situations are to be avoided. If an interrupt wake-up function is to be disabled then the corresponding interrupt request flag should be set high before the device 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, MFnF, will be automatically cleared, the individual request flag for the function needs to be cleared by the application program.

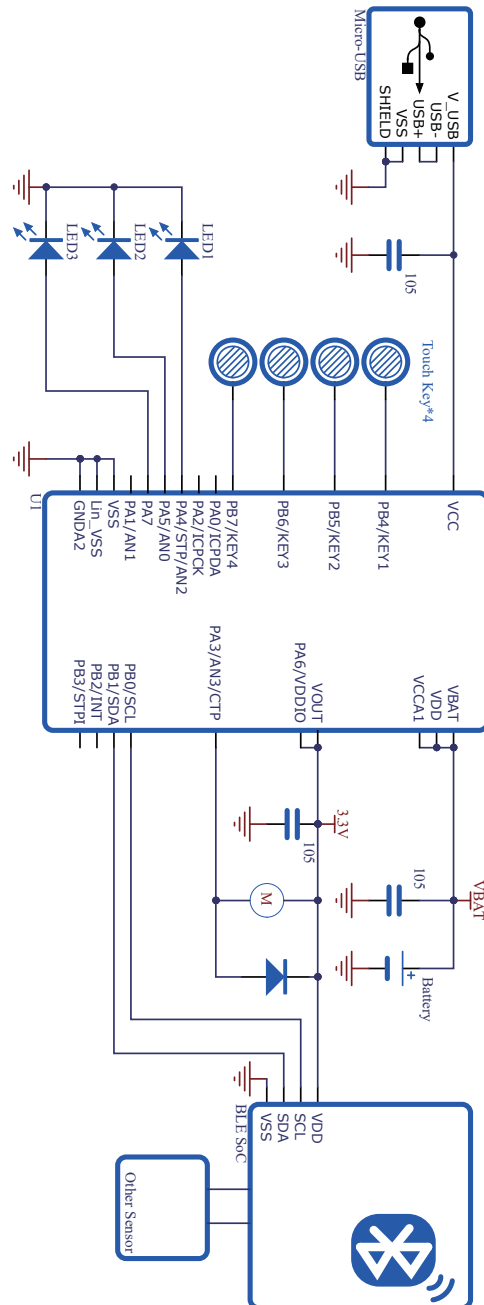
It is recommended that programs do not use the “CALL” instruction within the interrupt service subroutine. Interrupts often occur in an unpredictable manner or need to be serviced immediately. If only one stack is left and the interrupt is not well controlled, the original control sequence will be damaged 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.

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 0.5 μ s and branch or call instructions would be implemented within 1 μ s. 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) to TBLH and Data Memory	2 ^{Note}	None
TABRDC [m]	Read table (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
CLR WDT1	Pre-clear Watchdog Timer	1	TO, PDF
CLR WDT2	Pre-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.

3. For the "CLR WDT1" and "CLR WDT2" instructions the TO and PDF flags may be affected by the execution status. The TO and PDF flags are cleared after both "CLR WDT1" and "CLR WDT2" instructions are consecutively executed. Otherwise the TO and PDF flags remain unchanged.

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	$ACC \leftarrow ACC + [m] + C$
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	$[m] \leftarrow ACC + [m] + C$
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	$ACC \leftarrow ACC + [m]$
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	$ACC \leftarrow ACC + x$
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	$[m] \leftarrow ACC + [m]$
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	$ACC \leftarrow ACC \text{ "AND" } [m]$
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	$ACC \leftarrow ACC \text{ "AND" } x$
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	$[m] \leftarrow ACC \text{ "AND" } [m]$
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 \leftarrow Program Counter + 1 Program Counter \leftarrow addr
Affected flag(s)	None
CLR [m]	Clear Data Memory
Description	Each bit of the specified Data Memory is cleared to 0.
Operation	[m] \leftarrow 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 \leftarrow 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 \leftarrow 0 PDF \leftarrow 0
Affected flag(s)	TO, PDF
CLR WDT1	Pre-clear Watchdog Timer
Description	The TO, PDF flags and the WDT are all cleared. Note that this instruction works in conjunction with CLR WDT2 and must be executed alternately with CLR WDT2 to have effect. Repetitively executing this instruction without alternately executing CLR WDT2 will have no effect.
Operation	WDT cleared TO \leftarrow 0 PDF \leftarrow 0
Affected flag(s)	TO, PDF
CLR WDT2	Pre-clear Watchdog Timer
Description	The TO, PDF flags and the WDT are all cleared. Note that this instruction works in conjunction with CLR WDT1 and must be executed alternately with CLR WDT1 to have effect. Repetitively executing this instruction without alternately executing CLR WDT1 will have no effect.
Operation	WDT cleared TO \leftarrow 0 PDF \leftarrow 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 [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 \leftarrow 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 \leftarrow [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 \leftarrow 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] \leftarrow 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 \leftarrow 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 \leftarrow 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] \leftarrow 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 \leftarrow 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 \leftarrow Stack ACC \leftarrow 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 \leftarrow Stack EMI \leftarrow 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) \leftarrow [m].i; (i=0~6) [m].0 \leftarrow [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) \leftarrow [m].i; (i=0~6) ACC.0 \leftarrow [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) \leftarrow [m].i; (i=0~6) [m].0 \leftarrow C C \leftarrow [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=0~6) 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=0~6) [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	$ACC \leftarrow ACC - [m] - \bar{C}$
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	$[m] \leftarrow ACC - [m] - \bar{C}$
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	$ACC.3 \sim ACC.0 \leftarrow [m].7 \sim [m].4$ $ACC.7 \sim ACC.4 \leftarrow [m].3 \sim [m].0$
Affected flag(s)	None
SZ [m]	Skip if Data Memory is 0
Description	If the contents of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.
Operation	Skip if $[m]=0$
Affected flag(s)	None
SZA [m]	Skip if Data Memory is 0 with data movement to ACC
Description	The contents of the specified Data Memory are copied to the Accumulator. If the value is zero, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.
Operation	$ACC \leftarrow [m]$ 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

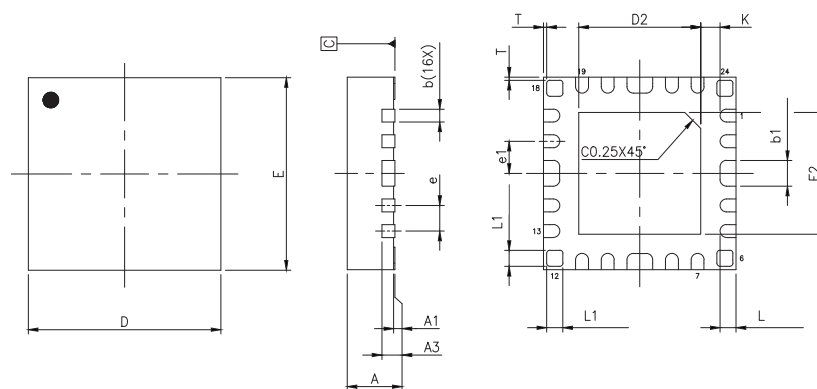
TABRD [m]	Read table (specific page) to TBLH and Data Memory
Description	The low byte of the program code (specific page) addressed by the table pointer pair (TBHP and TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.
Operation	[m] ← program code (low byte) TBLH ← program code (high byte)
Affected flag(s)	None
TABRDC [m]	Read table (current page) to TBLH and Data Memory
Description	The low byte of the program code (current page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.
Operation	[m] ← program code (low byte) TBLH ← program code (high byte)
Affected flag(s)	None
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

SAW Type 24-pin QFN (3mm×3mm×0.55mm) Outline Dimensions


Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	0.020	0.022	0.024
A1	0.000	0.001	0.002
A3	—	0.006 BSC	—
b	0.006	0.008	0.010
b1	0.014	0.016	0.018
D	—	0.118 BSC	—
E	—	0.118 BSC	—
e	—	0.016 BSC	—
e1	—	0.020 BSC	—
D2	0.073	0.075	0.077
E2	0.073	0.075	0.077
L	0.006	0.010	0.014
L1	0.008	0.010	0.012
K	0.008	—	—

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	0.50	0.55	0.60
A1	0.00	0.02	0.05
A3	—	0.150 BSC	—
b	0.15	0.20	0.25
b1	0.35	0.40	0.45
D	—	3.00 BSC	—
E	—	3.00 BSC	—
e	—	0.40 BSC	—
e1	—	0.50 BSC	—
D2	1.85	1.90	1.95
E2	1.85	1.90	1.95
L	0.15	0.25	0.35
L1	0.20	0.25	0.30
K	0.20	—	—

Copyright© 2019 by HOLTEK SEMICONDUCTOR INC.

The information appearing in this Data Sheet is believed to be accurate at the time of publication. However, Holtek assumes no responsibility arising from the use of the specifications described. The applications mentioned herein are used solely for the purpose of illustration and Holtek makes no warranty or representation that such applications will be suitable without further modification, nor recommends the use of its products for application that may present a risk to human life due to malfunction or otherwise. Holtek's products are not authorized for use as critical components in life support devices or systems. Holtek reserves the right to alter its products without prior notification. For the most up-to-date information, please visit our web site at <http://www.holtek.com>.