HOLTEK

TinyPower™ 24-Bit Delta Sigma
A/D Flash MCU with LCD & EEPROM

HT67F5630

Revision: V1.20   Date: December 14, 2016

Features

CPU Features

- Operating Voltage
    - f_{SYS}=8MHz: 2.2V~5.5V
    - f_{SYS}=12MHz: 2.7V~5.5V
    - f_{SYS}=20MHz: 4.5V~5.5V
- Up to 0.2μs instruction cycle with 20MHz system clock at V_{DD}=5V
- Power down and wake-up functions to reduce power consumption
- Three oscillators
    - External Crystal – HXT
    - Internal RC – HIRC
    - Internal 32kHz RC – LIRC
- Multi-mode operation: NORMAL, SLOW, IDLE and SLEEP
- Fully integrated internal 4.9152MHz, 4.9152×2MHz and 4.9152×3MHz oscillator requires no external components
- All instructions executed in one or two instruction cycles
- Table read instructions
- 63 powerful instructions
- 6-level subroutine nesting
- Bit manipulation instruction

Peripheral Features

- Flash Program Memory: 2K×16
- RAM Data Memory: 128×8
- True EEPROM Memory: 32×8
- Watchdog Timer function
- LCD COM driver with 1/3 bias
- 35 bidirectional I/O lines
- Dual pin-shared external interrupts
- Multiple Timer Module for time measure, input capture, compare match output, PWM output or single pulse output function
- Dual Time-Base functions for generation of fixed time interrupt signals
- 2 differential channels 24-bit resolution Delta Sigma A/D converter
- Low voltage reset function
- Low voltage detect function
- Internal LDO for PGA, ADC or external sensor power supply
- One comparator
- Package type: 48-pin LQFP

General Description

The device is a Flash Memory A/D type 8-bit high performance RISC architecture microcontroller with multi-channel 24-bit Delta Sigma A/D (ΣΔA/D) converter designed for applications that interface directly to analog signals and which require a low noise and high accuracy analog to digital converter. Offering users the convenience of Flash Memory multi-programming features, this device also includes 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 with 24-bit ΣΔA/D converter and programmable gain amplifier functions. Multiple extremely flexible Timer Modules provides timing, pulse generation and PWM generation functions. In addition, internal LDO function provides various power options to the internal and external devices. 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 external and internal low and high speed oscillator functions are provided including a fully integrated system oscillator which requires no external components for its implementation. The ability to operate and switch dynamically between a range of operating modes using different clock sources gives users the ability to optimise microcontroller operation and minimise power consumption.

The inclusion of flexible I/O programming features, Time-Base functions along with many other features ensure that the device will find excellent use in applications such as weight scales, electronic metering, environmental monitoring, handheld instruments, household appliances, electronically controlled tools, motor driving in addition to many others.

Block Diagram
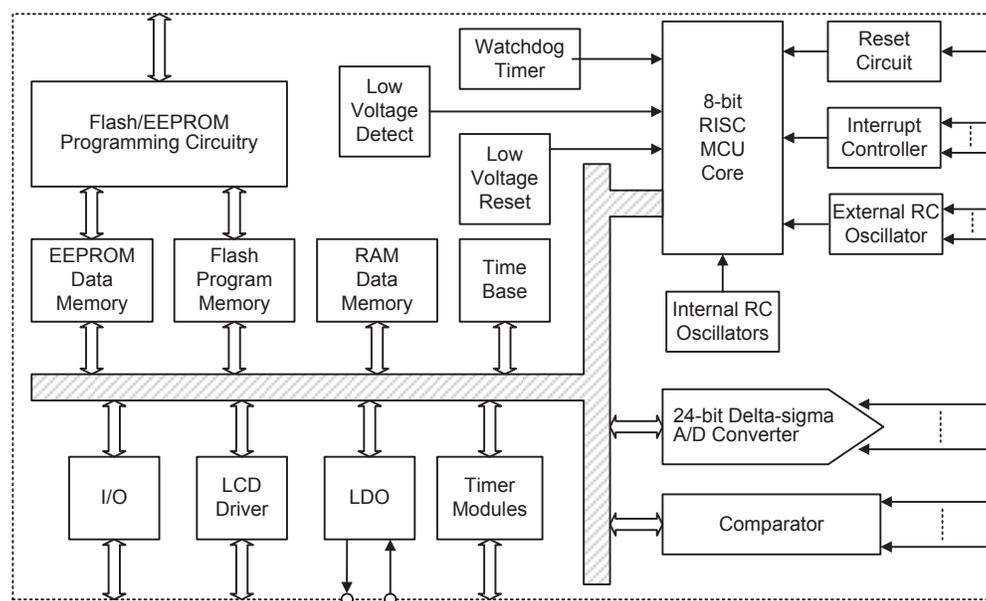
Pin Assignment
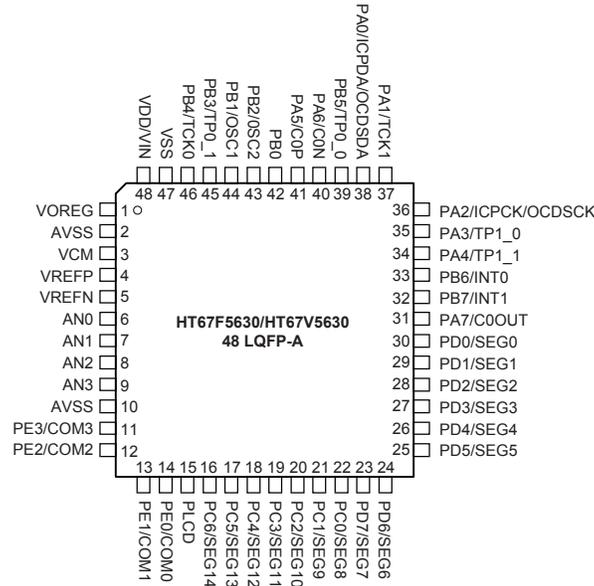
Note: The OCDSDA and OCDSCK pins are the OCDS dedicated pins and only available for the HT67V5630 device which is the OCDS EV chip for the HT67F5630 device.

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	Descriptions
PA0/ICPDA/ OCDSDA	PA0	PAPU PAWU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	ICPDA	—	ST	CMOS	ICP address/data
	OCDSDA	—	ST	CMOS	OCDS address/data, for EV chip only.
PA1/TCK1	PA1	PAPU PAWU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	TCK1	—	ST	—	TM1 clock input
PA2/ICPCK/ OCDSCK	PA2	PAPU PAWU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	ICPCK	—	ST	—	ICP clock
	OCDSCK	—	ST	—	OCDS clock, for EV chip only.
PA3/TP1_0	PA3	PAPU PAWU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	TP1_0	CTRL0 PTM1C0	ST	CMOS	TM1 input/output
PA4/TP1_1	PA4	PAPU PAWU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	TP1_1	CTRL0 PTM1C0	ST	CMOS	TM1 input/output

Pin Name	Function	OPT	I/T	O/T	Descriptions
PA5/C0P	PA5	PAPU PAWU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	C0P	CP0C	AN	—	Comparator 0 postive input
PA6/C0N	PA6	PAPU PAWU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	C0N	CP0C	AN	—	Comparator 0 negtive input
PA7/C0OUT	PA7	PAPU PAWU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	C0OUT	CP0C	—	CMOS	Comparator 0 output pin
PB0	PB0	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
PB1/OSC1	PB1	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	OSC1	CO	HXT	—	HXT Oscillator input
PB2/OSC2	PB2	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	OSC2	CO	—	HXT	HXT Oscillator output
PB3/TP0_1	PB3	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	TP0_1	CTRL0 TM0C0	ST	CMOS	TM0 input/output
PB4/TCK0	PB4	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	TCK0	—	ST	—	TM0 clock input
PB5/TP0_0	PB5	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	TP0_0	CTRL0 TM0C0	ST	CMOS	TM0 input/output
PB6/INT0	PB6	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	INT0	—	ST	—	External Interrupt 0 input
PB7/INT1	PB7	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	INT1	—	ST	—	External Interrupt 1 input
PC0/SEG8	PC0	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG8	LCD2	—	CMOS	LCD Segment output
PC1/SEG9	PC1	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG9	LCD2	—	CMOS	LCD Segment output
PC2/SEG10	PC2	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG10	LCD2	—	CMOS	LCD Segment output
PC3/SEG11	PC3	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG11	LCD2	—	CMOS	LCD Segment output
PC4/SEG12	PC4	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG12	LCD2	—	CMOS	LCD Segment output
PC5/SEG13	PC5	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG13	LCD2	—	CMOS	LCD Segment output

Pin Name	Function	OPT	I/T	O/T	Descriptions
PC6/SEG14	PC6	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG14	LCD2	—	CMOS	LCD Segment output
VOREG	VOREG	—	PWR	—	Supply voltage for ADC, VCM & PGA.
PD0/SEG0	PD0	PDPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG0	LCD0	—	CMOS	LCD Segment output
PD1/SEG1	PD1	PDPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG1	LCD0	—	CMOS	LCD Segment output
PD2/SEG2	PD2	PDPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG2	LCD0	—	CMOS	LCD Segment output
PD3/SEG3	PD3	PDPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG3	LCD0	—	CMOS	LCD Segment output
PD4/SEG4	PD4	PDPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG4	LCD0	—	CMOS	LCD Segment output
PD5/SEG5	PD5	PDPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG5	LCD0	—	CMOS	LCD Segment output
PD6/SEG6	PD6	PDPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG6	LCD0	—	CMOS	LCD Segment output
PD7/SEG7	PD7	PDPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	SEG7	LCD0	—	CMOS	LCD Segment output
PE0/COM0	PE0	PEPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	COM0	LCD4	—	CMOS	LCD Common output
PE1/COM1	PE1	PEPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	COM1	LCD4	—	CMOS	LCD Common output
PE2/COM2	PE2	PEPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	COM2	LCD4	—	CMOS	LCD Common output
PE3/COM3	PE3	PEPU	ST	CMOS	General purpose I/O. Register enabled pull-up.
	COM3	LCD4	—	CMOS	LCD Common output
AN0~AN3	ANn	—	AN	—	A/D input pin

Pin Name	Function	OPT	I/T	O/T	Descriptions
VREFP	VREFP	—	PWR	—	ADC positive reference input
VREFN	VREFN	—	PWR	—	ADC negative reference input
VCM	VCM	—	—	PWR	ADC common mode voltage output
		—	PWR	—	External ADC common mode input voltage
VDD/VIN	VDD	—	PWR	—	Digital Power supply
	VIN	—	PWR	—	LDO input pin
PLCD	PLCD	—	—	PWR	LCD power supply
AVSS	AVSS	—	PWR	—	Analog Ground
VSS	VSS	—	PWR	—	Digital Ground

Note: I/T: Input type

　　　O/T: Output type

　　　OP: Optional by configuration option (CO) or register option

　　　PWR: Power

　　　CO: Configuration option

　　　ST: Schmitt Trigger input

　　　CMOS: CMOS output

　　　AN: Analog input pin

　　　HXT: High frequency crystal oscillator

Absolute Maximum Ratings

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

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

Storage Temperature ....-50˚C to 150˚C

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

I_{OL} Total 150mA

I_{OH} Total ....-100mA

Total Power Dissipation 500mW

Note: These are stress ratings only. Stresses exceeding the range specified under "Absolute Maximum Ratings" may cause substantial damage to these devices. Functional operation of these 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

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage (HXT)	—	f_{SYS}=8MHz	2.2	—	5.5	V
		—	f_{SYS}=12MHz	2.7	—	5.5	V
		—	f_{SYS}=20MHz	4.5	—	5.5	V
	Operating Voltage (HIRC)	—	f_{SYS}=4.9152MHz	2.2	—	5.5	V
I_{DD}	Operating Current (HXT)	3V	f_{SYS}=f_H=8MHz No load, all peripheral off	—	1.0	1.5	mA
		5V		—	2.5	4	mA
		3V	f_{SYS}=f_H=12MHz No load, all peripheral off	—	1.5	2.5	mA
		5V		—	3.5	5.5	mA
		5V	f_{SYS}=f_H=20MHz No load, all peripheral off	—	5.5	8.5	mA
	Operating Current (HIRC)	3V	f_{SYS}=f_H=4.9152x3MHz No load, all peripheral off	—	1.8	3.0	mA
		5V		—	4.0	6.0	mA
	Operating Current (HXT)	3V	f_{SYS}=f_H/2, f_H=12MHz No load, all peripheral off	—	0.9	1.5	mA
		5V		—	2.1	3.3	mA
		3V	f_{SYS}=f_H/64, f_H=12MHz No load, all peripheral off	—	0.36	0.55	mA
		5V		—	1.0	1.5	mA
	Operating Current (LIRC)	3V	f_{SYS}=f_{SUB}=f_{LIRC}=32kHz No load, all peripheral off	—	10	20	µA
		5V		—	30	50	µA
I_{STB}	Standby Current (IDLE0)	3V	f_{SYS} off, f_{SUB} on, No load, all peripheral off	—	1.3	3.0	µA
		5V		—	2.2	5.0	µA
	Standby Current (IDLE1)	3V	f_{SYS}=12MHz on, f_{SUB} on, No load, all peripheral off	—	0.6	1.0	mA
		5V		—	1.2	2.0	mA
		3V	f_{SYS}=12MHz/64 on, f_{SUB} on, No load, all peripheral off	—	0.34	0.6	mA
		5V		—	0.85	1.2	mA
	Standby Current (SLEEP0)	3V	f_{SYS} off, f_{SUB} off, WDT disabled No load, all peripheral off	—	0.1	1	µA
		5V		—	0.3	2	µA
	Stanby Current (SLEEP1)	3V	f_{SYS} off, f_{SUB} on, WDT enabled No load, all peripheral off	—	1.3	5	µA
		5V		—	2.2	10	µA
		—	f_{SYS} off, f_{SUB} on, WDT disabled No load, all peripheral off, LVD enabled	—	90	120	µA
V_{IL}	Input Low Voltage for I/O port	5V	—	0	—	1.5	V
		—	—	0	—	0.2V_{DD}	V
V_{IH}	Input High Voltage for I/O Ports	5V	—	3.5	—	5	V
		—	—	0.8V_{DD}	—	V_{DD}	V
V_{LVR}	Low Voltage Reset Voltage	—	LVR enable, voltage select 2.1V	Typ. -5%	2.1	Typ. +5%	V
		—	LVR enable, voltage select 2.55V		2.55		V
		—	LVR enable, voltage select 3.15V		3.15		V
		—	LVR enable, voltage select 3.8V		3.8		V
I_{LVR}	Low Voltage Reset Current	—	LVR enable, LVD disable	—	60	90	µA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{LVD}	Low Voltage Detector Voltage	—	LVD enable, voltage select 2.0V	Typ. -5%	2.0	Typ. +5%	V
			LVD enable, voltage select 2.2V		2.2		V
			LVD enable, voltage select 2.4V		2.4		V
			LVD enable, voltage select 2.7V		2.7		V
			LVD enable, voltage select 3.0V		3.0		V
			LVD enable, voltage select 3.3V		3.3		V
			LVD enable, voltage select 3.6V		3.6		V
			LVD enable, voltage select 4.0V		4.0		V
I_{LVD}	Low Voltage Detector Current	—	LVR disable, LVD enable	—	75	120	µA
		—	LVR enable, LVD enable	—	90	150	µA
I_{OL}	Sink Current for I/O Ports	—	V_{DD}=3V, V_{OL}=0.1V_{DD}	4	8	—	mA
		—	V_{DD}=5V, V_{OL}=0.1V_{DD}	10	20	—	mA
	Sink Current for LCD pins	—	V_{LCD}=3V, V_{OL}=0.1V_{LCD}	210	420	—	µA
		—	V_{LCD}=5V, V_{OL}=0.1V_{LCD}	350	700	—	µA
I_{OH}	Source Current for I/O Ports	—	V_{DD}=3V, V_{OH}=0.9V_{DD}	-2	-4	—	mA
		—	V_{DD}=5V, V_{OH}=0.9V_{DD}	-5	-10	—	mA
	Source Current for LCD pins	—	V_{LCD}=3V, V_{OH}=0.9V_{LCD}	-80	-160	—	µA
		—	V_{LCD}=5V, V_{OH}=0.9V_{LCD}	-180	-360	—	µA
R_{PH}	Pull-high Resistance for I/O Ports	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ

A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS}	System Clock (HXT)	2.2V~ 5.5V	—	0.4	—	8	MHz
		2.7V~ 5.5V		0.4	—	12	MHz
		4.5V~ 5.5V		0.4	—	20	MHz
	System Clock (HIRC)	5V	Ta=25°C	Typ. -2%	4.9152 ×2	Typ. +2%	MHz
	System Clock (LIRC)	5V	Ta=25°C	Typ. -10%	32	Typ. +10%	kHz
		2.2V~ 5.5V	Ta= -40°C ~ +85°C	Typ. -50%	32	Typ. +60%	kHz
t_{INT}	Interrupt Pin Minimum Input Pulse Width	—	—	10	—	—	µs
t_{TCK}	TCKn and TPn Pin Minimum Input Pulse Width	—	—	0.3	—	—	µs
t_{SST}	System Start-up Timer Period (Wake-up from Power Down Mode, f_{SYS} off)	—	$f_{SYS}=f_{HXT}$	—	1024	—	t_{SYS}
			$f_{SYS}=f_{HIRC}$	—	16	—	
			$f_{SYS}=f_{LIRC}$	—	1~2	—	
t_{RSTD}	System Reset Delay Time (Power On Reset)	—	—	25	50	100	ms
	System Reset Delay Time (Any Reset except Power On Reset)	—	—	8.3	16.7	33.3	ms
t_{LVR}	Minimum Low Voltage Width to Reset	—	—	120	240	480	µs
t_{LVD}	Minimum Low Voltage Width to Interrupt	—	—	60	120	240	µs
t_{LVDS}	LVDO Stable Time	—	LVR enable, LVD disable→enable	—	—	15	µs
			LVR disable, LVD disable→enable	—	—	150	µs
t_{EERD}	EEPROM Read Time	—	—	—	—	4	t_{SYS}
t_{EEWR}	EEPROM Write Timet	—	—	1	2	4	ms

Note: $t_{SYS} = 1/f_{SYS}$

LDO+PGA+ADC+VCM Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{IN}	LDO Supply Voltage	—	—	2.6	—	5.5	V
I_{VOREG}	LDO Operating Current	—	No load, LDOVS[1:0]=00	—	400	520	µA
V_{OREG}	LDO Output Voltage (I_L=0.1mA, $V_{IN}>V_{OREG}$+0.2V)	—	LDOVS[1:0]=00	-5%	2.4	+5%	V
		—	LDOVS[1:0]=01		2.6		
		—	LDOVS[1:0]=10		2.9		
		—	LDOVS[1:0]=11		3.3		
	Dropout Voltage (I_L=10mA)	—	LDOVS[1:0]=00	—	—	100	mV
		—	LDOVS[1:0]=01	—	—	130	
		—	LDOVS[1:0]=10	—	—	180	
		—	LDOVS[1:0]=11	—	—	200	
	Temperature Drift	LDOVS[1:0]=00 I_L=100µA	—	Ta=-40°C~85°C	—	—	200	ppm/°C
	V_{OREG} Voltage Drift		2.7V~5.5V	—	-0.3	—	+0.3	%/V
ΔV_{LOAD}	Load Regulation	2.6V	Load=0mA~10mA, MCU HALT, LDO=2.4V, LDO enable, other function disable	—	25	50	mV
V_{CM}	V_{CM} Output Voltage	—	No load	-5%	1.05	+5%	V
		—	I_L = 200µA	0.98	1.25	1.02	V_{CM}
	Temperature Drift	—	I_L = 10µA, Ta=-40°C~85°C	—	—	200	ppm/°C
	AV_{DD} Voltage Drift	—	No load, AV_{DD}=2.5V~3.3V	—	100	—	µV/V
t_{VCM}	V_{CM} Turn on Stable Time	—	—	10	—	—	ms
I_{CMSRC}	V_{CM} Source Current	—	V_{CM} drop 2% of V_{CM}	1	—	—	mA
I_{CMSNK}	V_{CM} Sink Current	—	V_{CM} raise 2% of V_{CM}	1	—	—	mA

ADC & ADC Internal Reference Voltage (Delta Sigma ADC)

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
AV_{DD}	Supply Voltage for VCM, ADC, PGA	—	—	2.4	—	3.3	V
I_{CM}+ I_{PGIA}+ I_{ADC}	Operating Current for VCM, PGA and ADC	—	VCM enable, VRBUFP=1 and VRBUFN=1	—	—	900	µA
		—	VCM enable, VRBUFP=0 and VRBUFN=0	—	600	750	
		—	VCM disable, VRBUFP=0 and VRBUFN=0	—	500	650	
I_{ADSTB}	Standby Current	—	System HALT, no load	—	—	1	µA
RS_{AD}	ADC Resolution	—	—	—	—	24	bits
NNFC	Noise Free Code	—	PGA Gain=128 Data Rate=12.5Hz	—	15.4	—	bits
ENOB	Effective Number of Bits	—	PGA Gain=128 Data Rate=12.5Hz	—	18.1	—	bits

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{AD}	A/D Clock Frequency (f_{MCLK})	—	—	—	4.9152	—	MHz
f_{ADO}	ADC Output Data Rate	—	f_{MCLK}= 4.9152MHz FLMS[2:0]=000, ADC CLK=f_{MCLK}/30	5	—	640	Hz
			f_{MCLK}= 4.9152MHz FLMS[2:0]=010, ADC CLK=f_{MCLK}/12	12.5	—	1600	
V_{REF+}	Reference Input Voltage	—	VREFS=1, VRBUFP=0 and VRBUFN=0	0.96	1.25	2.2	V
V_{REF-}				0	0	1.0	
V_{REF}		—	V_{REF}=(V_{REF+})-(V_{REF-})	0.96	1.25	1.44	
PGA							
V_{DI+}, V_{DI-}	Absolute/Common Input Voltage	—	—	0.4	—	AV_{DD}-1.3	V
$\Delta DI\pm$	Differentail Input Voltage Range	—	Gain = PGS×AGS	$\Delta VR-$/Gain	—	$\Delta VR+$/Gain	V
TC_{PGA}	Gain Temperature Drift	—	Ta=-40°C~85°C	—	5	—	ppm/°C
Temperature Sensor							
TC_S	Sensor Temperature Drift	—	Ta=-40°C~85°C ΔVR=1.25V, VGS[1:0]=00 (Gain=1), VRBUFP=0 and VRBUFN=0	—	175	—	µV/°C

Effective Number of Bits (ENOB)

AV_{DD}=3.3V, V_{REF}=1.25V, f_{MCLK}=4.9152MHz, FLMS[2:0]=000

Data Rate (SPS)	PGA Gain							
	1	2	4	8	16	32	64	128
5	19.7	19.8	19.6	19.7	19.7	19.6	19.2	18.6
10	19.4	19.3	19.3	19.3	19.3	19.1	18.7	18.1
20	19.0	18.8	18.7	18.9	18.8	18.6	18.2	17.5
40	18.4	18.3	18.3	18.3	18.3	18.1	17.7	17.0
80	18.1	17.9	18.0	17.9	17.9	17.6	17.2	16.5
160	17.6	17.4	17.4	17.4	17.3	17.1	16.6	15.9
320	15.8	15.8	15.9	15.8	15.9	15.9	15.8	15.3
640	14.1	14.0	14.0	14.1	14.1	14.0	14.1	14.4

AV$_{DD}$=3.3V, V$_{REF}$=1.25V, f$_{MCLK}$=4.9152MHz, FLMS[2:0]=010

Data Rate (SPS)	PGA Gain							
	1	2	4	8	16	32	64	128
12.5	19.4	18.8	18.7	18.8	18.8	18.7	18.9	18.1
25	19.0	18.3	18.3	18.3	18.3	18.2	17.9	17.3
50	18.5	17.8	17.8	17.8	17.9	17.7	17.4	16.8
100	18.2	18.2	18.1	18.2	18.1	17.8	17.2	16.4
200	17.9	17.8	17.8	17.8	17.6	17.3	16.7	15.9
400	17.4	17.2	17.2	17.2	17.1	16.8	16.2	15.4
800	16.2	16.1	16.1	16.1	16.1	15.9	15.5	14.8
1600	14.5	14.5	14.5	14.4	14.5	14.5	14.3	14.0

Comparator Electrical Characteristics

Ta=25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V$_{DD}$	Conditions				
V$_{DD}$	Comparator Operating Voltage	—	—	2.2	—	5.5	V
I$_{CMP}$	Additional Current for Comparator Enabled	3V	—	—	37	56	μA
		5V	—	—	130	200	
V$_{OS}$	Input Offset Voltage Range	—	—	-10	—	10	mV
V$_{HYS}$	Hysteresis Voltage	—	—	20	40	80	mV
V$_{CM}$	Common Mode Voltage Range	—	—	V$_{SS}$	—	V$_{DD}$-1.4V	V
A$_{OL}$	Open Loop Gain	—	—	60	80	—	dB
t$_{RP}$	Response Time	—	With 100mV overdrive[Note]	—	200	450	ns

Note: Measured with comparator one input pin at V$_{CM}$ = (V$_{DD}$-1.4)/2 while the other pin input transition from V$_{SS}$ to (V$_{CM}$ + 100mV) or from V$_{DD}$ to (V$_{CM}$ − 100mV).

LCD Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{LCD}	Additional Current for LCD Enabled (R type)	5V	V_{LCD}=5V, LCD clock = 4kHz 1/3 Bias, no load	—	25	37.5	μA
				—	50	75	
				—	100	150	
				—	200	300	
V_{PLCD}	LCD Power Supply	—	LCDPR=0	—	—	V_{DD}	V
		2.6V~5.5V	LCDPR=1, CPVS[1:0]=00	-10%	3.3	+10%	V
		2.3V~5.5V	LCDPR=1, CPVS[1:0]=01		3.0		
		2.2V~5.5V	LCDPR=1, CPVS[1:0]=10		2.7		
		3.8V~5.5V	LCDPR=1, CPVS[1:0]=11		4.5		
I_{PUMP}	Additional Current Consumption for Charge Pump Enabled	3V	LCDPR=1, CPVS[1:0]=00, LCDIS[1:0]=01	—	150	200	μA

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
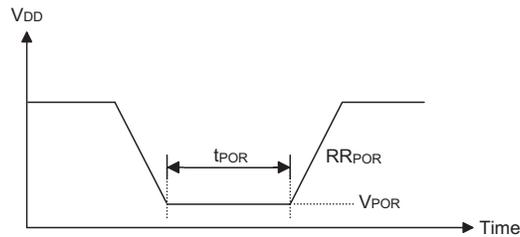
System Architecture

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

Clocking and Pipelining

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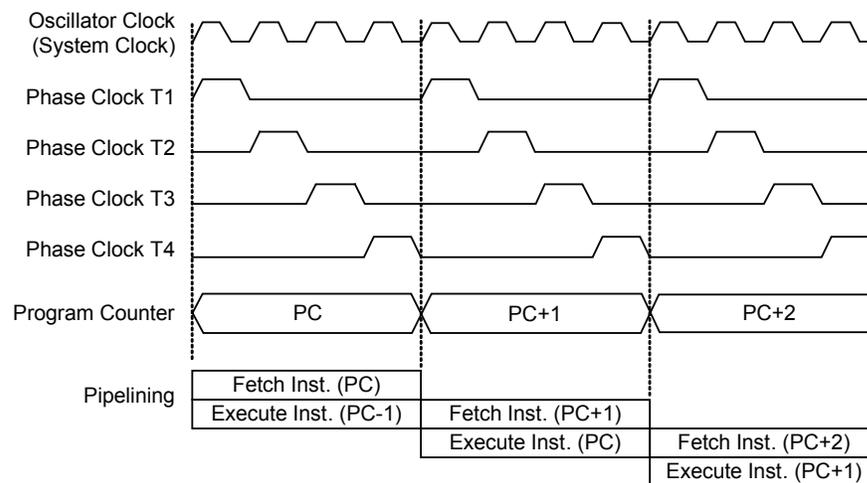
System Clocking and Pipelining

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.

```
1        MOV A,[12H]
2        CALL DELAY
3        CPL [12H]
4        :
5        :
6        NOP

    DELAY:
```

Fetch Inst. 1	Execute Inst. 1			
	Fetch Inst. 2	Execute Inst. 2		
		Fetch Inst. 3	Flush Pipeline	
			Fetch Inst. 6	Execute Inst. 6
				Fetch Inst. 7
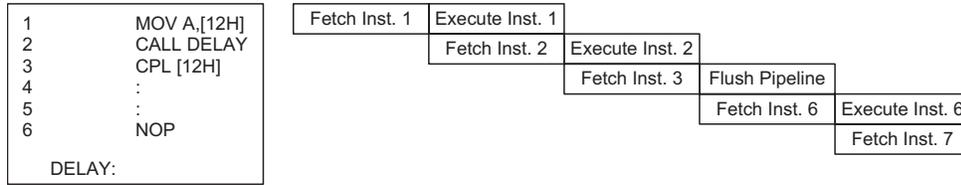
Instruction Fetching

Program Counter

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

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

Program Counter	
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 6 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.
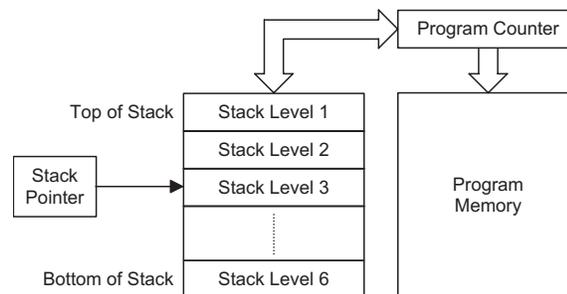
Arithmetic and Logic Unit – ALU

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

Structure

The Program Memory has a capacity of 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.

000H	Initialisation Vector
004H	
	Interrupt Vectors
024H	
028H	
n00H	Look-up Table
nFFH	
7FFH	16 bits
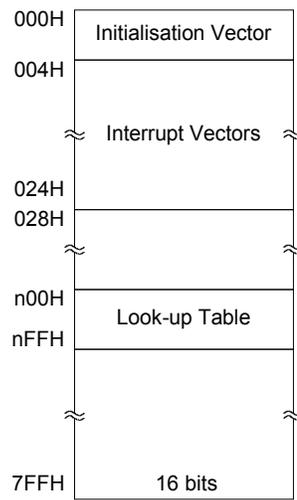
Program Memory Structure

Special Vectors

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

Look-up Table

Any location within the Program Memory can be defined as a look-up table where programmers can store fixed data. To use the look-up table, the table pointer must first be setup by placing the address of the look up data to be retrieved in the table pointer register, 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.
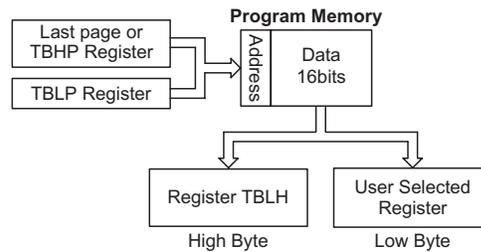
Table Program Example

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 Program Memory of the microcontroller. The table pointer is setup here to have an initial value of "06H". This will ensure that the first data read from the data table will be at the Program Memory address "706H" or 6 locations after the start of the last page. Note that the value for the table pointer is referenced to the first address specified by TBLP and TBHP if the "TABRD [m]" instruction is being used. The high byte of the table data which in this case is equal to zero will be transferred to the TBLH register automatically when the "TABRD [m]" instruction is executed.

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

Table Read Program Example

```
tempreg1 db ?        ; temporary register #1
tempreg2 db ?        ; temporary register #2
:
:
mov a,06h            ; initialise low table pointer - note that this address
                     ; is referenced
mov tblp,a           ; to the last page or present page
mov a,07h            ; initialise high table pointer
mov tbhp,a
:
:
tabrdl 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
tabrdl 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
:
:
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.

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

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

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

During the programming process, taking control of the PA0 and PA2 pins for data and clock programming purposes. The user must there take care to ensure that no other outputs are connected to these two pins.
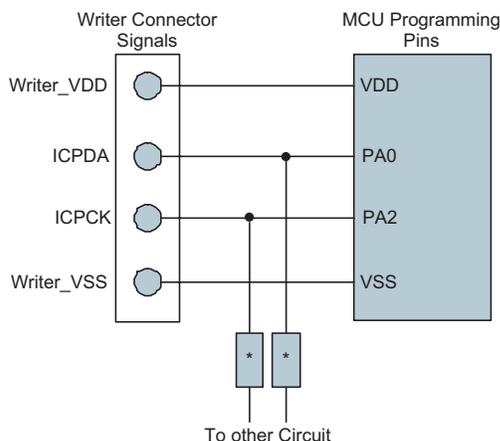
Note: * may be resistor or capacitor. The resistance of * must be greater than 1k or the capacitance of * must be less than 1nF.

On Chip Debug Support – OCDS

There is an EV chip named HT67V5630 which is used to emulate the HT67F5630 device. The EV chip device also provides an "On-Chip Debug" function to debug the real MCU device during the development process. The EV chip and the real MCU device are almost functionally compatible except for "On-Chip Debug" function. Users can use the EV chip device to emulate the real chip device behavior by connecting the OCDSDA and OCDSCK pins to the Holtek HT-IDE development tools. The OCDSDA pin is the OCDS Data/Address input/output pin while the OCDSCK pin is the OCDS clock input pin. When users use the EV chip for debugging, other functions which are shared with the OCDSDA and OCDSCK pins in the device will have no effect in the EV chip. However, the two OCDS pins which are pin-shared with the ICP programming pins are still used as the Flash Memory programming pins for ICP. For more detailed OCDS information, refer to the corresponding document named "Holtek e-Link for 8-bit MCU OCDS User's Guide".

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

RAM 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.

Divided into three types, the first of these is an area of RAM where special function registers are located. These registers have fixed locations and are necessary for correct operation of the device. Many of these registers can be read from and written to directly under program control, however, some remain protected from user manipulation. The second area of Data Memory is reserved for general purpose use. All locations within this area are read and write accessible under program control. The third area is reserved for the LCD Memory. This special area of Data Memory is mapped directly to the LCD display so data written into this memory area will directly affect the displayed data.

Structure

The Data Memory is subdivided into several banks, all of which are implemented in 8-bit wide RAM. Each of the Data Memory Bank is categorized into two types, the special Purpose Data Memory and the General Purpose Data Memory. While the 80H~8FH of Bank 1 is LCD Memory.

The start address of the Data Memory for the device is the address 00H while the start address of the General Purpose Data Memory and LCD Memory is the address 80H. The Special Purpose Data Memory registers which are addressed from 00H to 7FH in Data Memory are common to all banks and are accessible in all banks, with the exception of the EEC register at address 40H, which are only accessible in Bank 1. Switching between the different Data Memory banks is achieved by setting the Memory Pointers to the correct value.

Capacity	Banks
General Purpose: 128×8	0: 80H~FFH 1: 80H~8FH (for LCD)
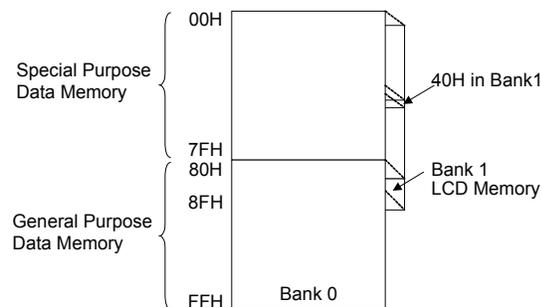
Data Memory Structure

	Bank 0~1			Bank 0	Bank 1
00H	IAR0		40H	Unused	EEC
01H	MP0		41H	EEA	
02H	IAR1		42H	EED	
03H	MP1		43H	Unused	
04H	BP		44H	Unused	
05H	ACC		45H	Unused	
06H	PCL		46H	Unused	
07H	TBLP		47H	Unused	
08H	TBLH		48H	Unused	
09H	TBHP		49H	CTRL0	
0AH	STATUS		4AH	PTM1RPL	
0BH	SMOD		4BH	PTM1RPH	
0CH	LVDC		4CH	Unused	
0DH	INTEG		4DH	Unused	
0EH	WDTC		4EH	Unused	
0FH	TBC		4FH	CP0C	
10H	INTC0		50H	PWRC	
11H	INTC1		51H	PGAC0	
12H	INTC2		52H	PGAC1	
13H	Unused		53H	PGACS	
14H	MFI0		54H	Unused	
15H	MFI1		55H	Unused	
16H	MFI2		56H	Unused	
17H	Unused		57H	Unused	
18H	PAWU		58H	Unused	
19H	PAPU		59H	Unused	
1AH	PA		5AH	LCDCP	
1BH	PAC		5BH	LCDC	
1CH	PBPU		5CH	LCD1	
1DH	PB		5DH	LCD2	
1EH	PBC		5EH	LCD3	
1FH	PCPU		5FH	Unused	
20H	PC		60H	Unused	
21H	PCC		61H	Unused	
22H	PDPU		62H	Unused	
23H	PD		63H	Unused	
24H	PDC		64H	Unused	
25H	PEPU		65H	Unused	
26H	PE		66H	Unused	
27H	PEC		67H	Unused	
28H	Unused		68H	Unused	
29H	Unused		69H	Unused	
2AH	Unused		6AH	Unused	
2BH	LVRC		6BH	Unused	
2CH	CTRL		6CH	Unused	
2DH	Unused		6DH	Unused	
2EH	ADRL		6EH	Unused	
2FH	ADRH		6FH	Unused	
30H	ADCR0		70H	Unused	
31H	ADCR1		71H	Unused	
32H	ADCS		72H	Unused	
33H	ADRM		73H	Unused	
34H	PTM1C0		74H	Unused	
35H	PTM1C1		75H	Unused	
36H	PTM1DL		76H	Unused	
37H	PTM1DH		77H	Unused	
38H	PTM1AL		78H	Unused	
39H	PTM1AH		79H	Unused	
3AH	TM0C0		7AH	Unused	
3BH	TM0C1		7BH	Unused	
3CH	TM0DL		7CH	Unused	
3DH	TM0DH		7DH	Unused	
3EH	TM0AL		7EH	Unused	
3FH	TM0AH		7FH	Unused	

■ : 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 and MP1 register pair can access data from any bank. As the Indirect Addressing Registers are not physically implemented, reading the Indirect Addressing Registers indirectly will return a result of "00H" and writing to the registers indirectly 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 and IAR1 are used to access data from all banks according to BP register. Direct Addressing can only be used with Bank 0, all other Banks must be addressed indirectly using MP1 and IAR1.

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

Indirect Addressing Program Example

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

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

Bank Pointer – BP

The Data Memory is divided into several banks. Selecting the required Data Memory area is achieved using the Bank Pointer. Bit 0 of the Bank Pointer is used to select Data Memory Banks 0~1.

The Data Memory is initialised to Bank 0 after a reset, except for a WDT time-out reset in the Power Down Mode, in which case, the Data Memory bank remains unaffected. It should be noted that the Special Function Data Memory is not affected by the bank selection, which means that the Special Function Registers can be accessed from within any bank. 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	—	—	—	—	—	—	—	DMBP0
R/W	—	—	—	—	—	—	—	R/W
POR	—	—	—	—	—	—	—	0

Bit 7~1   Unimplemented, read as "0"

Bit 0   **DMBP0**: Select Data Memory Banks
         0: Bank 0
         1: Bank 1

Accumulator – ACC

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

Program Counter Low Register – PCL

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

Look-up Table Registers – TBLP, TBHP, TBLH

These three special function registers are used to control operation of the look-up table which is stored in the Program Memory. TBLP and TBHP are the table pointer and indicates 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
C is also affected by a rotate through carry instruction.

EEPROM Data Memory

This device contains an area of internal EEPROM Data Memory. EEPROM, which stands for Electrically Erasable Programmable Read Only Memory, 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 the device. 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 data register in Bank 0 and a single control register in Bank 1.

Capacity	Address
32×8	00H~1FH

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 was as any other Special Function Register. The EEC register however, being located in Bank1, cannot be directly addressed directly and 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.

Name	Bit							
	7	6	5	4	3	2	1	0
EEA	D7	D6	D5	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	D7	D6	D5	EEA4	EEA3	EEA2	EEA1	EEA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~5   Undefined, used as general data bits

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	x	x	x	x	x	x	x	x

"x" unknown

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

EEC Register

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

Bit 7~4   Unimplemented, read as "0"

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

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

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

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

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

Reading Data from the EEPROM

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

Writing Data to the EEPROM

The EEPROM address of the data to be written must first be placed in the EEA register and the data placed in the EED register. To write data to the EEPROM, the write enable bit, WREN, in the EEC register must first be set high to enable the write function. After this, the WR bit in the EEC register must be immediately set high to initiate a write cycle. These two instructions must be executed consecutively. 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, 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. However as the EEPROM is contained within a Multi-function Interrupt, the associated multi-function interrupt enable bit must also be set. When an EEPROM write cycle ends, the DEF request flag and its associated multi-function interrupt request flag will both be set. If the global, EEPROM and Multi-function interrupts are enabled and the stack is not full, a jump to the associated Multi-function Interrupt vector will take place. When the interrupt is serviced only the Multi-function interrupt flag will be automatically reset, the EEPROM interrupt flag must be manually reset by the application program. More details can be obtained in the Interrupt section.

Programming Considerations

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

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

Programming Examples

- **Reading data from the EEPROM – polling method**

```
MOV  A, EEPROM_ADRES      ; user defined address
MOV  EEA, A
MOV  A, 040H              ; setup memory pointer MP1
MOV  MP1, A               ; MP1 points to EEC register
MOV  A, 01H               ; setup Bank Pointer
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/write
CLR  BP
MOV  A, EED              ; move read data to register
MOV  READ_DATA, A
```

- **Writing Data to the EEPROM – polling method**

```
MOV  A, EEPROM_ADRES      ; user defined address
MOV  EEA, A
MOV  A, EEPROM_DATA       ; user defined data
MOV  EED, A
MOV  A, 040H              ; setup memory pointer MP1
MOV  MP1, A               ; MP1 points to EEC register
MOV  A, 01H               ; setup Bank Pointer
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  IAR1                ; disable EEPROM read/write
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 a combination of configuration options and registers.

Oscillator Overview

In addition to being the source of the main system clock the oscillators also provide clock sources for the Watchdog Timer and Time Base Interrupts. External oscillators requiring some external components as well as fully integrated internal oscillators, requiring no external components, are provided to form a wide range of both fast and slow system oscillators. All oscillator options are selected through the configuration options. 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	Freq.	Pins
External Crystal	HXT	400kHz~20MHz	OSC1/OSC2
Internal High Speed RC	HIRC	4.9152, 4.9152×2 or 4.9152×3MHz	—
Internal Low Speed RC	LIRC	32kHz	—

Oscillator Types

System Clock Configurations

There are three methods of generating the system clock, two high speed oscillators and one low speed oscillator. The high speed oscillators are the external crystal oscillator and the internal 4.9152MHz, 4.9152×2MHz or 4.9152×3MHz RC oscillator. The low speed oscillator is the internal 32kHz RC oscillator. Selecting whether the low or high speed oscillator is used as the system oscillator is implemented using the HLCLK bit and CKS2~CKS0 bits in the SMOD register and as the system clock can be dynamically selected.

The actual source clock used for each of the high speed oscillators is chosen via configuration options. The frequency of the high speed system clock is also determined using the HLCLK bit and CKS2~CKS0 bits in the SMOD register. Note that two oscillator selections must be made namely one high speed and one low speed system oscillators. It is not possible to choose a no-oscillator selection for either the high or low speed oscillator.
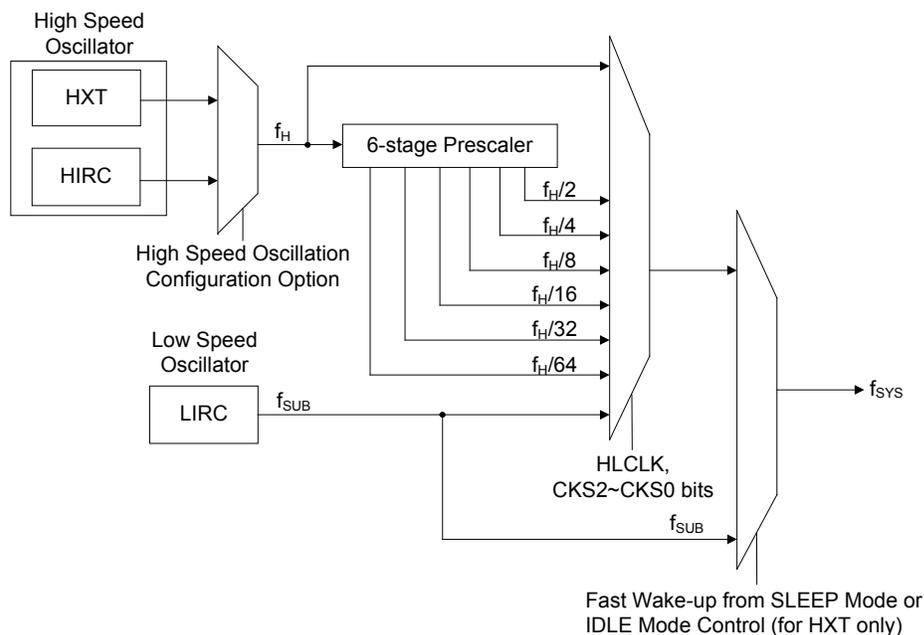
System Clock Configurations

External Crystal/Ceramic Oscillator – HXT

The External Crystal/Ceramic System Oscillator is one of the high frequency oscillator choices, which is selected via configuration option. For most crystal oscillator configurations, the simple connection of a crystal across OSC1 and OSC2 will create the necessary phase shift and feedback for oscillation, without requiring external capacitors. However, for some crystal types and frequencies, to ensure oscillation, it may be necessary to add two small value capacitors, C1 and C2. Using a ceramic resonator will usually require two small value capacitors, C1 and C2, to be connected as shown for oscillation to occur. The values of C1 and C2 should be selected in consultation with the crystal or resonator manufacturer's specification.

For oscillator stability and to minimise the effects of noise and crosstalk, it is important to ensure thatthe crystal and any associated resistors andcapacitors along with interconnectinglines are all located as close to the MCUas possible.
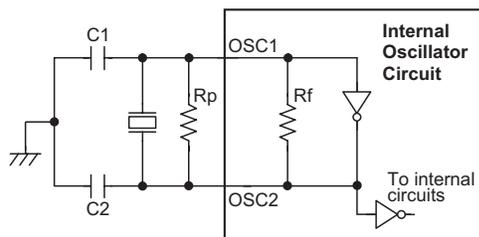
Note: 1. Rp is normally not required. C1 and C2 are required.
2. Although not shown OSC1/OSC2 pins have a parasitic capacitance of around 7pF.

Crystal/Resonator Oscillator – HXT

Crystal Oscillator C1 and C2 Values		
Crystal Frequency	C1	C2
20MHz	0pF	0pF
12MHz	0pF	0pF
8MHz	0pF	0pF
4MHz	0pF	0pF
1MHz	100pF	100pF
Note: C1 and C2 values are for guidance only.		

Crystal Recommended Capacitor Values

Internal 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 either 4.9152MHz, 4.9152×2MHz or 4.9152×3MHz. 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. As a result, at a power supply of 5V and at a temperature of 25°C degrees, the fixed oscillation frequency of 4.9152×2MHz will have a tolerance within 2%. Note that if this internal system clock option is selected, as it requires no external pins for its operation, I/O pins PB1 and PB2 are free for use as normal I/O pins.

Internal 32kHz Oscillator – LIRC

The Internal 32kHz System Oscillator is one of the low frequency oscillator choices, which is selected via configuration option. It is a fully integrated RC oscillator with a typical frequency of 32kHz at 5V, 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. As a result, at a power supply of 5V and at a temperature of 25°C degrees, the fixed oscillation frequency of 32kHz will have a tolerance within 10%.

Supplementary Oscillators

The low speed oscillators, in addition to providing a system clock source are also used to provide a clock source to three other device functions. These are the Watchdog Timer, the Time Base Interrupts function, and the LCD driver.

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 this device with both high and low speed clock sources and the means to switch between them dynamically, the user can optimise the operation of their microcontroller to achieve the best performance/power ratio.

System Clocks

The device has many different clock sources for both the CPU and peripheral function operation. By providing the user with a wide range of clock options using configuration options and 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 HLCLK bit and CKS2~CKS0 bits in the SMOD register. The high speed system clock can be sourced from either an HXT or HIRC oscillator, selected via a configuration option. The low speed system clock source can be sourced from internal clock f_{SUB}. If f_{SUB} is selected then it can be sourced by the LIRC oscillator. The other choice, which is a divided version of the high speed system oscillator has a range of $f_H/2 \sim f_H/64$.

The f_{SUB} clock is used to provide a substitute clock for the microcontroller just after a wake-up has occurred to enable faster wake-up times. The f_{SUB} is used as a clock source for the Watchdog timer, the Time Base interrupt, the TMs, and the LCD functions.
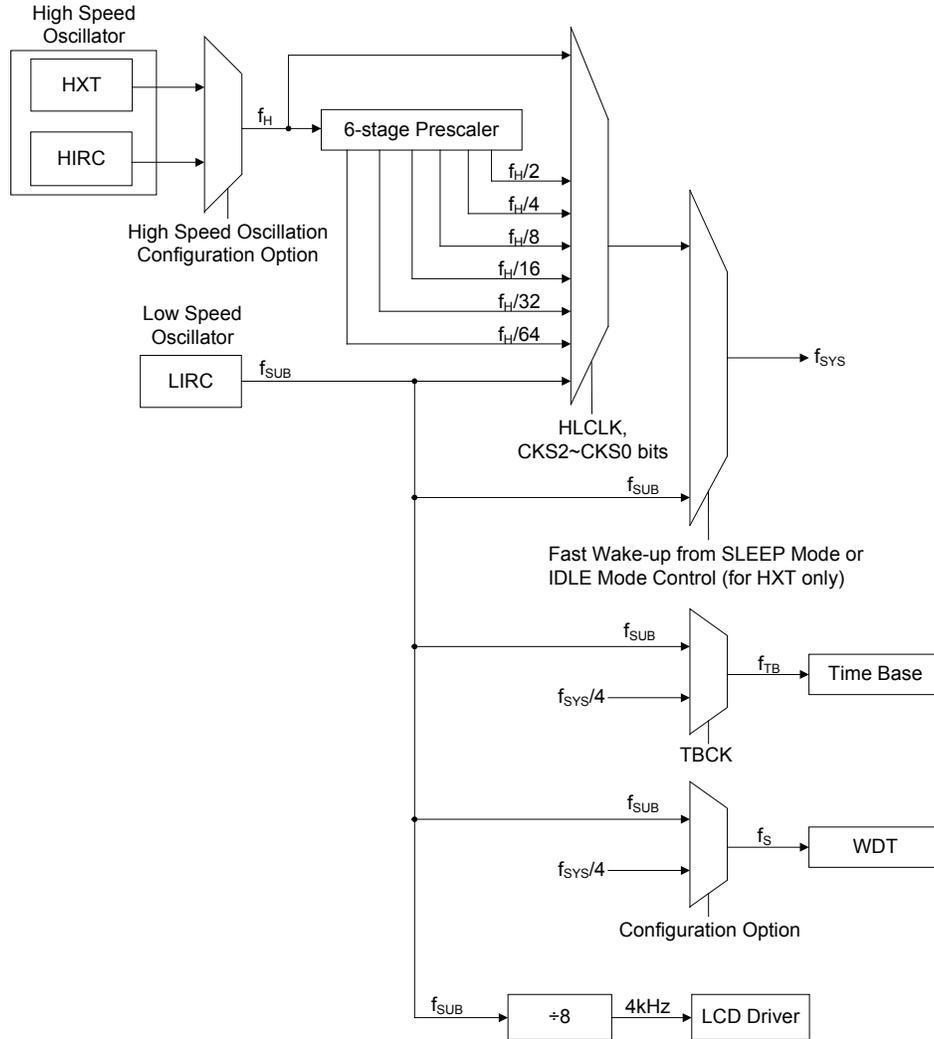
System Clock Configurations

Note: When the system clock source f_{SYS} is switched to f_{SUB} from f_H, the high speed oscillation will stop to conserve the power. Thus there is no $f_H \sim f_H/64$ for peripheral circuit to use.

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 NORMAL Mode and SLOW Mode. The remaining four modes, the SLEEP0, SLEEP1, IDLE0 and IDLE1 Mode are used when the microcontroller CPU is switched off to conserve power.

Operating Mode	Description			
	CPU	f_{SYS}	f_{SUB}	f_S
NORMAL Mode	on	$f_H \sim f_H/64$	on	on
SLOW Mode	on	f_{SUB}	on	on
IDLE0 Mode	off	off	on	on/off
IDLE1 Mode	off	on	on	on
SLEEP0 Mode	off	off	off	off
SLEEP1 Mode	off	off	on	on

NORMAL Mode

As the name suggests 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 one of the high speed oscillators. This mode operates allowing the microcontroller to operate normally with a clock source will come from one of the high speed oscillators, either the HXT or HIRC oscillators. 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 and HLCLK bits in the SMOD 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 the low speed oscillator, LIRC. Running the microcontroller in this mode allows it to run with much lower operating currents. In the SLOW Mode, the f_H is off.

SLEEP0 Mode

The SLEEP Mode is entered when an HALT instruction is executed and when the IDLEN bit in the SMOD register is low. In the SLEEP0 mode the CPU will be stopped, and the f_{SUB} and f_S clocks will be stopped too, and the Watchdog Timer function is disabled. In this mode, the LVDEN is must cleared to zero. If the LVDEN is set high, it won't enter the SLEEP0 Mode.

SLEEP1 Mode

The SLEEP Mode is entered when an HALT instruction is executed and when the IDLEN bit in the SMOD register is low. In the SLEEP1 mode the CPU will be stopped. However the f_{SUB} and f_S clocks will continue to operate if the LVDEN is "1" or the Watchdog Timer function is enabled and if its clock source is chosen via configuration option to come from the f_{SUB}.

IDLE0 Mode

The IDLE0 Mode is entered when a HALT instruction is executed and when the IDLEN bit in the SMOD register is high and the FSYSON bit in the CTRL register is low. In the IDLE0 Mode the system oscillator will be inhibited from driving the CPU but some peripheral functions will remain operational such as the Watchdog Timer, TMs, and LCD driver. In the IDLE0 Mode, the system oscillator will be stopped. In the IDLE0 Mode the Watchdog Timer clock, f_S, will either be on or off depending upon the f_S clock source. If the source is $f_{SYS}/4$ then the f_S clock will be off, and if the source comes from f_{SUB} then f_S will be on.

IDLE1 Mode

The IDLE1 Mode is entered when an HALT instruction is executed and when the IDLEN bit in the SMOD register is high and the FSYSON bit in the CTRL register is high. In the IDLE1 Mode the system oscillator will be inhibited from driving the CPU but may continue to provide a clock source to keep some peripheral functions operational such as the Watchdog Timer, TMs, and LCD driver. In the IDLE1 Mode, the system oscillator will continue to run, and this system oscillator may be high speed or low speed system oscillator. In the IDLE1 Mode the Watchdog Timer clock, f_S, will be on. If the source is $f_{SYS}/4$ then the f_S clock will be on, and if the source comes from f_{SUB} then f_S will be on.

Control Register

The registers, SMOD and CTRL, are used for overall control of the internal clocks within the device.

SMOD Register

Bit	7	6	5	4	3	2	1	0
Name	CKS2	CKS1	CKS0	FSTEN	LTO	HTO	IDLEN	HLCLK
R/W	R/W	R/W	R/W	R/W	R/W	R	R	R/W	R/W
POR	0	0	0	0	0	0	1	1

Bit 7~5   **CKS2~CKS0**: The system clock selection when HLCLK is "0"

     000: f_{SUB} (f_{LIRC})
     001: f_{SUB} (f_{LIRC})
     010: $f_H/64$
     011: $f_H/32$
     100: $f_H/16$
     101: $f_H/8$
     110: $f_H/4$
     111: $f_H/2$

These three bits are used to select which clock is used as the system clock source. In addition to the system clock source, which can be the LIRC, a divided version of the high speed system oscillator can also be chosen as the system clock source.

Bit 4   **FSTEN**: Fast Wake-up Control (only for HXT)

     0: Disable
     1: Enable

This is the Fast Wake-up Control bit which determines if the f_{SUB} clock source is initially used after the device wakes up. When the bit is high, the f_{SUB} clock source can be used as a temporary system clock to provide a faster wake up time as the f_{SUB} clock is available.

Bit 3   **LTO**: Low speed system oscillator ready flag
        0: Not ready
        1: Ready

This is the low speed system oscillator ready flag which indicates when the low speed system oscillator is stable after power on reset or a wake-up has occurred. The flag will be low when in the SLEEP0 Mode but after a wake-up has occurred, the flag will change to a high level after 1~2 clock cycles if the LIRC oscillator is used.

Bit 2   **HTO**: High speed system oscillator ready flag
        0: Not ready
        1: Ready

This is the high speed system oscillator ready flag which indicates when the high speed system oscillator is stable. This flag is cleared to zero by hardware when the device is powered on and then changes to a high level after the high speed system oscillator is stable.

Therefore this flag will always be read as "1" by the application program after device power-on. The flag will be low when in the SLEEP or IDLE0 Mode but after a wake-up has occurred, the flag will change to a high level after 1024 clock cycles if the HXT oscillator is used and after 15~16 clock cycles if the HIRC oscillator is used.

Bit 1   **IDLEN**: IDLE Mode control
        0: Disable
        1: Enable

This is the IDLE Mode Control bit and determines what happens when the HALT instruction is executed. If this bit is high, when a HALT instruction is executed the device will enter the IDLE Mode. In the IDLE1 Mode the CPU will stop running but the system clock will continue to keep the peripheral functions operational, if FSYSON bit is high. If FSYSON bit is low, the CPU and the system clock will all stop in IDLE0 mode. If the bit is low the device will enter the SLEEP Mode when a HALT instruction is executed.

Bit 0   **HLCLK**: system clock selection
        0: $f_H/2 \sim f_H/64$ or f_{SUB}
        1: f_H

This bit is used to select if the f_H clock or the $f_H/2 \sim f_H/64$ or f_{SUB} clock is used as the system clock. When the bit is high the f_H clock will be selected and if low the $f_H/2 \sim f_H/64$ or f_{SUB} clock will be selected. When system clock switches from the f_H clock to the f_{SUB} clock and the f_H clock will be automatically switched off to conserve power.

CTRL Register

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

"x" unknown

Bit 7   **FSYSON**: f_{SYS} Control in IDLE Mode
        0: Disable
        1: Enable

Bit 6~3   Unimplemented, read as "0"

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

Bit 1   **LRF**: LVR Control register software reset flag
        Described elsewhere.

Bit 0   **WRF**: WDT Control register software reset flag
        Described elsewhere.

Fast Wake-up

To minimise power consumption the device can enter the SLEEP or IDLE0 Mode, where the system clock source to the device will be stopped. However when the device is woken up again, it can take a considerable time for the original system oscillator to restart, stabilise and allow normal operation to resume. To ensure the device is up and running as fast as possible a Fast Wake-up function is provided, which allows f_{SUB}, namely LIRC oscillator, to act as a temporary clock to first drive the system until the original system oscillator has stabilised. As the clock source for the Fast Wake-up function is f_{SUB}, the Fast Wake-up function is only available in the SLEEP1 and IDLE0 modes. When the device is woken up from the SLEEP0 mode, the Fast Wake-up function has no effect because the f_{SUB} clock is stopped. The Fast Wake-up enable/disable function is controlled using the FSTEN bit in the SMOD register.

If the HXT oscillator is selected as the NORMAL Mode system clock, and if the Fast Wake-up function is enabled, then it will take one to two t_{SUB} clock cycles of the LIRC oscillator for the system to wake-up. The system will then initially run under the f_{SUB} clock source until 1024 HXT clock cycles have elapsed, at which point the HTO flag will switch high and the system will switch over to operating from the HXT oscillator.

If the HIRC oscillator or LIRC oscillator is used as the system oscillator then it will take 15~16 clock cycles of the HIRC or 1~2 cycles of the LIRC to wake up the system from the SLEEP or IDLE0 Mode. The Fast Wake-up bit, FSTEN will have no effect in these cases.

System Oscillator	FSTEN Bit	Wake-up Time (SLEEP0 Mode)	Wake-up Time (SLEEP1 Mode)	Wake-up Time (IDLE0 Mode)	Wake-up Time (IDLE1 Mode)
HXT	0	1024 HXT cycles	1024 HXT cycles		1~2 HXT cycles
	1	1024 HXT cycles	1~2 f_{SUB} cycles (System runs with f_{SUB} first for 1024 HXT cycles and then switches over to run with the HXT clock)		1~2 HXT cycles
HIRC	×	15~16 HIRC cycles	15~16 HIRC cycles		1~2 HIRC cycles
LIRC	×	1~2 LIRC cycles	1~2 LIRC cycles		1~2 LIRC cycles

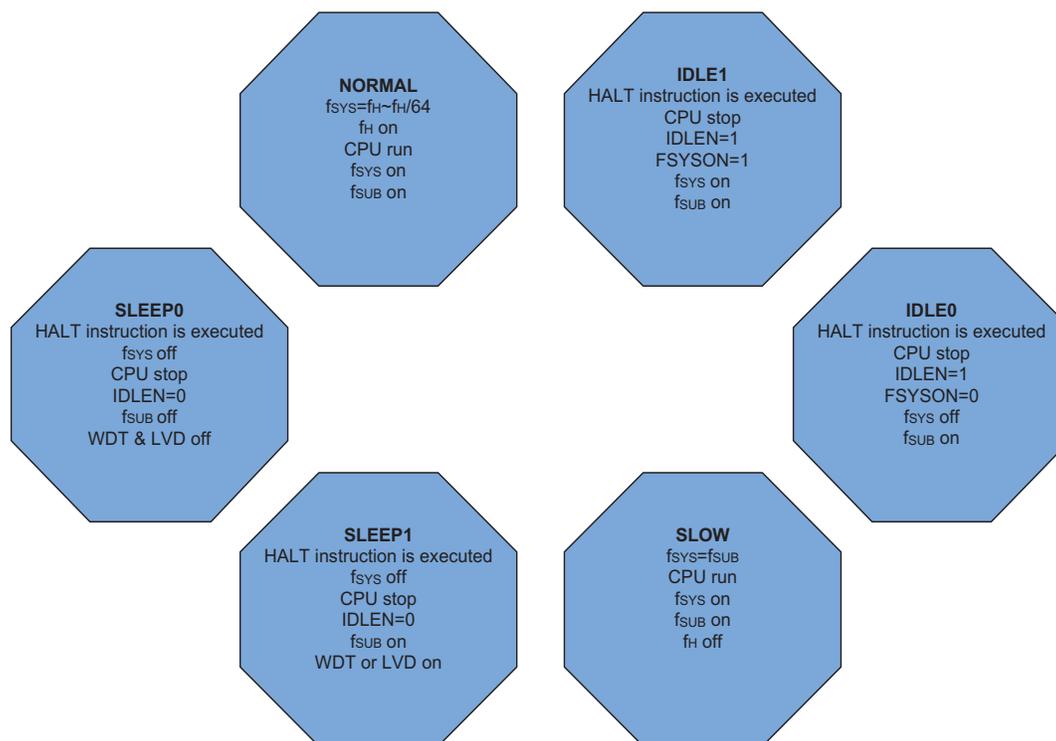"×": don't care

Wake-Up Times

Note that if the Watchdog Timer is disabled, which means that the LIRC is off, then there will be no Fast Wake-up function available when the device wake-up from the SLEEP0 Mode.

Operating Mode Switching

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

In simple terms, Mode Switching between the NORMAL Mode and SLOW Mode is executed using the HLCLK bit and CKS2~CKS0 bits in the SMOD register while Mode Switching from the NORMAL/SLOW Modes to the SLEEP/IDLE Modes is executed via the HALT instruction. When a HALT instruction is executed, whether the device enters the IDLE Mode or the SLEEP Mode is determined by the condition of the IDLEN bit in the SMOD register and FSYSON in the CTRL register.
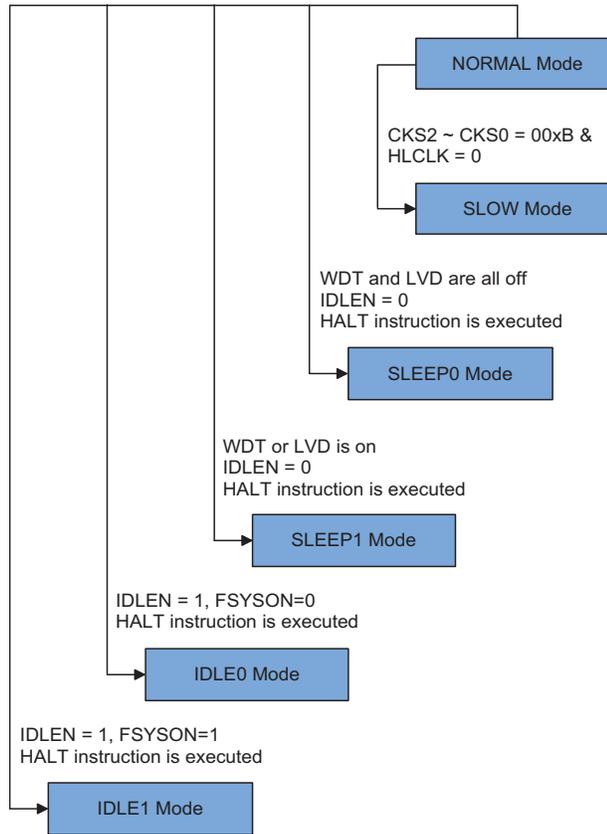
When the HLCLK bit switches to a low level, which implies that clock source is switched from the high speed clock source, f_H, to the clock source, $f_H/2 \sim f_H/64$ or f_{SUB}. If the clock is from the f_{SUB}, the high speed clock source will stop running to conserve power. When this happens it must be noted that the $f_H/16$ and $f_H/64$ internal clock sources will also stop running, which may affect the operation of other internal functions such as the TMs. The accompanying flowchart shows what happens when the device moves between the various operating modes.

NORMAL
$f_{SYS}=f_H \sim f_H/64$
f_H on
CPU run
f_{SYS} on
f_{SUB} on

IDLE1
HALT instruction is executed
CPU stop
IDLEN=1
FSYSON=1
f_{SYS} on
f_{SUB} on

SLEEP0
HALT instruction is executed
f_{SYS} off
CPU stop
IDLEN=0
f_{SUB} off
WDT & LVD off

IDLE0
HALT instruction is executed
CPU stop
IDLEN=1
FSYSON=0
f_{SYS} off
f_{SUB} on

SLEEP1
HALT instruction is executed
f_{SYS} off
CPU stop
IDLEN=0
f_{SUB} on
WDT or LVD on

SLOW
$f_{SYS}=f_{SUB}$
CPU run
f_{SYS} on
f_{SUB} on
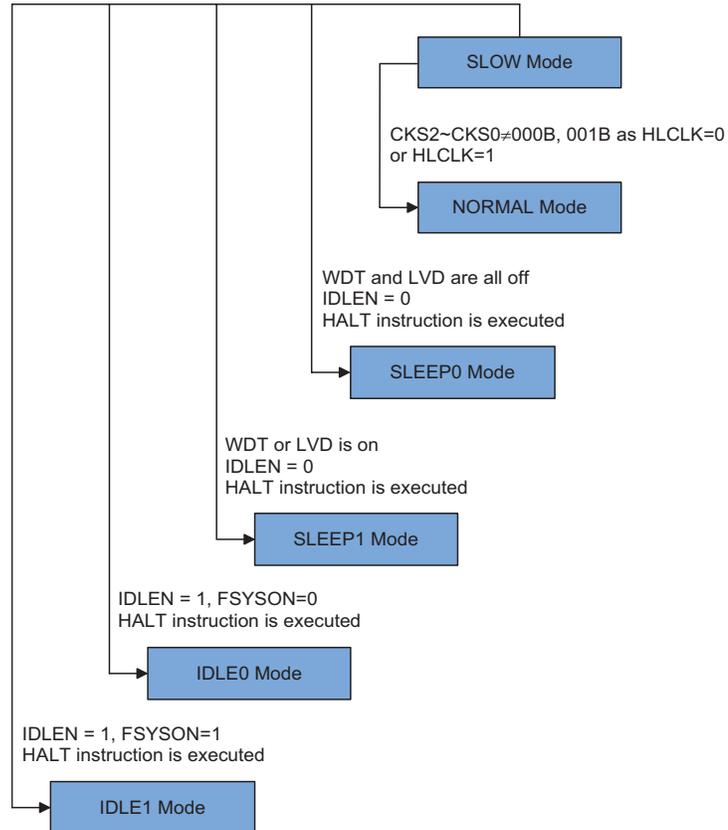f_H off

NORMAL Mode to SLOW Mode Switching

When running in the NORMAL 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 HLCLK bit to "0" and set the CKS2~CKS0 bits to "000" or "001" in the SMOD register. This will then use the low speed system oscillator which will consume less power. Users may decide to do this for certain operations which do not require high performance and can subsequently reduce power consumption.

The SLOW Mode is sourced from the LIRC oscillator and therefore requires the oscillator to be stable before full mode switching occurs. This is monitored using the LTO bit in the SMOD register.

NORMAL Mode

CKS2 ~ CKS0 = 00xB &
HLCLK = 0

SLOW Mode

WDT and LVD are all off
IDLEN = 0
HALT instruction is executed

SLEEP0 Mode

WDT or LVD is on
IDLEN = 0
HALT instruction is executed

SLEEP1 Mode

IDLEN = 1, FSYSON=0
HALT instruction is executed

IDLE0 Mode

IDLEN = 1, FSYSON=1
HALT instruction is executed

IDLE1 Mode

SLOW Mode to NORMAL Mode Switching

In SLOW Mode the system uses the LIRC low speed system oscillator. To switch back to the NORMAL Mode, where the high speed system oscillator is used, the HLCLK bit should be set high or HLCLK bit is "0", but CKS2~CKS0 is set to "010", "011", "100", "101", "110" or "111". As a certain amount of time will be required for the high frequency clock to stabilise, the status of the HTO bit is checked. The amount of time required for high speed system oscillator stabilization depends upon which high speed system oscillator type is used.

SLOW Mode

CKS2~CKS0≠000B, 001B as HLCLK=0
or HLCLK=1

NORMAL Mode

WDT and LVD are all off
IDLEN = 0
HALT instruction is executed

SLEEP0 Mode

WDT or LVD is on
IDLEN = 0
HALT instruction is executed

SLEEP1 Mode

IDLEN = 1, FSYSON=0
HALT instruction is executed

IDLE0 Mode

IDLEN = 1, FSYSON=1
HALT instruction is executed

IDLE1 Mode

Entering the SLEEP0 Mode

There is only one way for the device to enter the SLEEP0 Mode and that is to execute the "HALT" instruction in the application program with the IDLEN bit in SMOD register equal to "0" and the WDT and LVD both off. When this instruction is executed under the conditions described above, the following will occur:

- The system clock and the f_{SUB} 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 WDT will be cleared and stopped no matter if the WDT clock source originates from the f_{SUB} clock or from the system clock.
- 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.

Entering the SLEEP1 Mode

There is only one way for the device to enter the SLEEP1 Mode and that is to execute the "HALT" instruction in the application program with the IDLEN bit in SMOD register equal to "0" and the WDT or LVD on. 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, but the WDT or LVD will remain with the clock source coming from the f_{SUB} clock.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting if the WDT clock source is selected to come from the f_{SUB} clock as the WDT is enabled.
- 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.

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 IDLEN bit in SMOD register equal to "1" and the FSYSON bit in CTRL register equal to "0". 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, but the f_{SUB} clock will be on.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting if the WDT clock source is selected to come from the f_{SUB} clock and the WDT is enabled. The WDT will stop if its clock source originates from the system clock.
- 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.

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 the IDLEN bit in SMOD register equal to "1" and the FSYSON bit in CTRL register equal to "1". When this instruction is executed under the conditions described above, the following will occur:

- The system clock and the f_{SUB} clock will be on and the application program will stop at the "HALT" instruction.

- The Data Memory contents and registers will maintain their present condition.

- The WDT will be cleared and resume counting if the WDT is enabled regardless of the WDT clock source which originates from the f_{SUB} clock or from the system clock.

- 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.

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 Mode, there are other considerations which must also be taken into account by the circuit designer if the power consumption is to be minimised. Special attention must be made to the I/O pins on the device. All high-impedance input pins must be connected to either a fixed high or low level as any floating input pins could create internal oscillations and result in increased current consumption. This also applies to the device which has different package types, as there may be unbonded pins. These must either be setup as outputs or if setup as inputs must have pull-high resistors connected.

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

In the IDLE1 Mode the system oscillator is on, if the system oscillator is from the high speed system oscillator, the additional standby current will also be perhaps in the order of several hundred micro-amps.

Wake-up

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

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

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

Each pin on Port A can be setup using the PAWU register to permit a negative transition on the pin to wake-up the system. When a 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.

Programming Considerations

The HXT oscillator uses the same SST counter. For example, if the system is woken up from the SLEEP0 Mode and the HXT oscillator needs to start-up from an off state.

- If the device is woken up from the SLEEP0 Mode to the NORMAL Mode, the high speed system oscillator needs an SST period. The device will execute first instruction after HTO is set to 1.
- If the device is woken up from the SLEEP1 Mode to NORMAL Mode, and the system clock source is from HXT oscillator and FSTEN is 1, the system clock can be switched to the LIRC oscillator after wake up.
- There are peripheral functions, such as WDT, TMs, and LCD driver, for which the f_{SYS} is used. If the system clock source is switched from f_H to f_{SUB}, the clock source to the peripheral functions mentioned above will change accordingly.
- The on/off condition of f_{SUB} and f_S depends upon whether the WDT is enabled or disabled as the WDT clock source is selected from f_{SUB}.

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_S, which is in turn supplied by one of two sources selected by configuration option: f_{SUB} or $f_{SYS}/4$. The f_{SUB} clock is sourced from the LIRC oscillator. 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. The LIRC internal oscillator has an approximate period of 32kHz at a supply voltage of 5V. However, it should be noted that this specified internal clock period can vary with V_{DD}, temperature and process variations. The other Watchdog Timer clock source option is the $f_{SYS}/4$ clock.

Watchdog Timer Control Register

A single register, WDTC, controls the required timeout period as well as the enable/disable operation. This register together with several configuration options control the overall operation of the Watchdog Timer.

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
    If the WDT configuration option is "always enable":
        10101 or 01010: enable
        Others: reset MCU
    If the WDT configuration option is "controlled by the WDT control register":
        10101: disable
        01010: enable
        Others: reset MCU
    When these bits are changed by the environmental noise or software setting to reset the microcontroller, the reset operation will be activated after 2~3 LIRC clock cycles and the WRF bit in the CTRL register will be set high.

Bit 2~0   **WS2~WS0**: WDT time-out period selection
        000: $2^8/f_S$
        001: $2^{10}/f_S$
        010: $2^{12}/f_S$
        011: $2^{14}/f_S$
        100: $2^{15}/f_S$
        101: $2^{16}/f_S$
        110: $2^{17}/f_S$
        111: $2^{18}/f_S$

CTRL Register

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

"x" unknown

Bit 7   **FSYSON**: f_{SYS} Control in IDLE Mode
        Described elsewhere.

Bit 6~3   Unimplemented, read as "0"

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

Bit 1   **LRF**: LVR Control register software reset flag
        Described elsewhere.

Bit 0   **WRF**: WDT Control register software reset flag
        0: not occur
        1: occurred
        This bit is set high by the WDT Control register software reset and cleared 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 instructions. If the program malfunctions for whatever reason, jumps to an unknown location, or enters an endless loop, these clear instructions will not be executed in the correct manner, in which case the Watchdog Timer will overflow and reset the device. Some of the Watchdog Timer options, such as always on select and clear instruction type are selected using configuration options. With regard to the Watchdog Timer enable/disable function, there are also five bits, WE4~WE0, in the WDTC register to offer additional enable/disable and reset control of the Watchdog Timer. If the WDT configuration option is determined that the WDT function is always enabled, the WE4~WE0 bits still have effects on the WDT function. When the WE4~WE0 bits value is equal to 01010B or 10101B, the WDT function is enabled. However, if the WE4~WE0 bits are changed to any other values except 01010B and 10101B, which is caused by the environmental noise or software setting, it will reset the microcontroller after 2~3 LIRC clock cycles. If the WDT configuration option is determined that the WDT function is controlled by the WDT control register, the WE4~WE0 values can determine which mode the WDT operates in. The WDT function will be disabled when the WE4~WE0 bits are set to a value of 10101B. The WDT function will be enabled if the WE4~WE0 bits value is equal to 01010B. If the WE4~WE0 bits are set to any other values by the environmental noise or software setting, except 01010B and 10101B, it will reset the device after 2~3 LIRC clock cycles. After power on these bits will have the value of 01010B.

WDT Configuration Option	WE4 ~ WE0 Bits	WDT Function
Always Enable	01010B or 10101B	Enable
	Any other values	Reset MCU
Controlled by WDT Control Register	10101B	Disable
	01010B	Enable
	Any other values	Reset MCU

Watchdog Timer Enable/Disable Control

Under normal program operation, a Watchdog Timer time-out will initialise a device reset and set the status bit TO. However, if the system is in the SLEEP or IDLE Mode, when a Watchdog Timer time-out occurs, the TO bit in the status register will be set and only the Program Counter and Stack Pointer will be reset. Three methods can be adopted to clear the contents of the Watchdog Timer. The first is a WDT 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 instructions 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 second for the 2^{18} division ratio, and a minimum timeout of 7.8ms for the 2^8 division ration.

Watchdog Timer
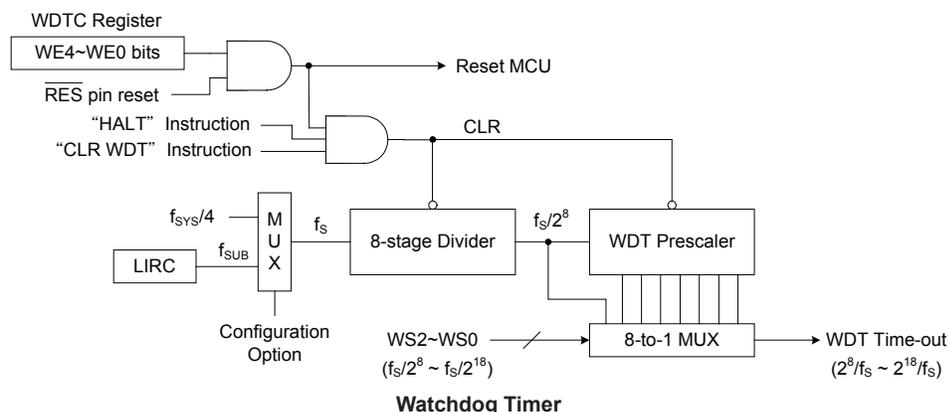
Reset and Initialisation

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

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

Reset Functions

There are five ways in which a reset can occur, through events occurring both internally and externally:

Power-on Reset

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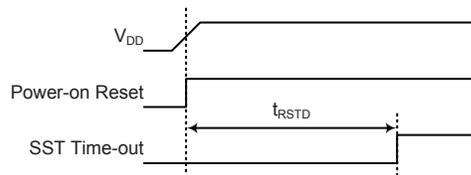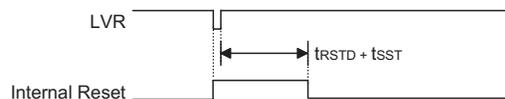

Note: t_{RSTD} is power-on delay, typical time=50ms
Power-On Reset Timing Chart

Low Voltage Reset – LVR

The microcontroller contains a low voltage reset circuit in order to monitor the supply voltage of the device. The LVR function is always enabled 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, the LVR will automatically reset the device internally and the LVRF bit in the CTRL register will also be set high. 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 A.C. 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 LVS7~LVS0 bits in the LVRC register. If the LVS7~LVS0 bits are changed to some certain values by the environmental noise or software setting, the LVR will reset the device after 2~3 f_{SUB} clock cycles. When this happens, the LRF bit in the CTRL register will be set high. 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 power down mode.

Note: t_{RSTD} is power-on delay, typical time=50ms
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
　　　　　　01010101: 2.1V
　　　　　　00110011: 2.55V
　　　　　　10011001: 3.15V
　　　　　　10101010: 3.8V
　　　　　　Other values: 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 2~3 f_{SUB} clock cycles. 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 2~3 f_{SUB} clock cycles. However in this situation the register contents will be reset to the POR value.

- **CTRL Register**

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

"x" unknown

Bit 7　　　**FSYSON**: f_{SYS} Control in IDLE Mode
　　　　　Described elsewhere.

Bit 6~3　 Unimplemented, read as "0"

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
　　　　　Described elsewhere.

Watchdog Time-out Reset during Normal Operation

The Watchdog time-out Reset during normal operation is the same as a hardware Low Voltage reset except that the Watchdog time-out flag TO will be set high.
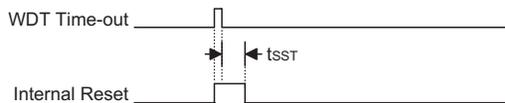
WDT Time-out

$t_{RSTD} + t_{SST}$

Internal Reset
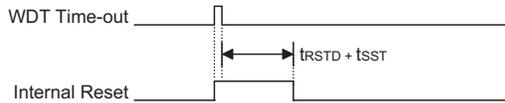
Note: t_{RSTD} is power-on delay, typical time=16.7ms
WDT Time-out Reset during Normal Operation Timing Chart

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 zero and the TO flag will be set high. Refer to the A.C. Characteristics for t_{SST} details.

WDT Time-out

t_{SST}

Internal Reset

Note: The t_{SST} is 15~16 clock cycles if the system clock source is provided by HIRC.
　　　The t_{SST} is 1024 clock for HXT. The t_{SST} is 1~2 clock for LIRC.
WDT Time-out Reset during Sleep 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 Normal or SLOW Mode operation
1	u	WDT time-out reset during Normal or SLOW Mode operation
1	1	WDT time-out reset during IDLE or SLEEP Mode operation

Note: "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	Clear after reset, WDT begins counting
Timer/Event Counter	Timer Counter will be turned off
Input/Output Ports	I/O ports will be setup as inputs, and AN0~AN3 as A/D input pin.
Stack Pointer	Stack Pointer will point to the top of the stack

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

Register	Power On Reset	LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
MP0	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
MP1	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	---- -xxx	---- -uuu	---- -uuu	---- -uuu
STATUS	--00 xxxx	--uu uuuu	--1u uuuu	--11 uuuu
BP	---- ---0	---- ---0	---- ---0	---- ---u
SMOD	0000 0011	0000 0011	0000 0011	uuuu uuuu
INTEG	---- 0000	---- 0000	---- 0000	---- uuuu
LVDC	--00 -000	--00 -000	--00 -000	--uu -uuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC2	-0-0 -0-0	-0-0 -0-0	-0-0 -0-0	-u-u -u-u
MFI0	--00 --00	--00 --00	--00 --00	--uu --uu
MFI1	--00 --00	--00 --00	--00 --00	--uu --uu
MFI2	--00 --00	--00 --00	--00 --00	--uu --uu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PE	---- 1111	---- 1111	---- 1111	uuuu uuuu
PEC	---- 1111	---- 1111	---- 1111	uuuu uuuu
WDTC	0101 0011	0101 0011	0101 0011	uuuu uuuu
TBC	0011 -111	0011 -111	0011 -111	uuuu -uuu
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PEPU	---- 0000	---- 0000	---- 0000	uuuu uuuu
TM0C0	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0C1	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	Power On Reset	LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
TM0DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0DH	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
TM0AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0AH	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
EEA	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
EED	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
LCDC	0 - - - - - 0 0	0 - - - - - 0 0	0 - - - - - 0 0	u - - - - - u u
LCD1	0000 0000	0000 0000	0000 0000	uuuu uuuu
LCD2	- 000 0000	- 000 0000	- 000 0000	- uuu uuuu
LCD3	- - - - 0000	- - - - 0000	- - - - 0000	- - - - uuuu
CP0C	1000 0 - - 1	1000 0 - - 1	1000 0 - - 1	uuuu u - - u
CTRL0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
LCDCP	- - - - 0 - 0 0	- - - - 0 - 0 0	- - - - 0 - 0 0	- - - - u - u u
PWRC	00 - - - 000	00 - - - 000	00 - - - 000	uu - - - uuu
PGAC0	- 000 0000	- 000 0000	- 000 0000	- uuu uuuu
PGAC1	10 - - 000 -	10 - - 000 -	10 - - 000 -	uu - - uuu -
PGACS	- - 00 0000	- - 00 0000	- - 00 0000	- - uu uuuu
ADCR0	0010 00 - 0	0010 00 - 0	0010 00 - 0	uuuu uu - u
ADCR1	0000 000 -	0000 000 -	0000 000 -	uuuu uuu -
ADRL	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRH	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCS	- - - 0 0000	- - - 0 0000	- - - 0 0000	- - - u uuuu
ADRM	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
LVRC	0101 0101	0101 0101	0101 0101	uuuu uuuu
CTRL	0 - - - - x00	0 - - - - x00	0 - - - - x00	u - - - - uuu
PTM1C0	0000 0 - - -	0000 0 - - -	0000 0 - - -	uuuu u - - -
PTM1C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1DH	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
PTM1AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1AH	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
PTM1RPL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1RPH	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
EEC	- - - - 0000	- - - - 0000	- - - - 0000	- - - - uuuu

Note: "u" stands for unchanged

"x" stands for unknown

"-" stands for unimplemented

Input/Output Ports

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

The device provides bidirectional input/output lines labeled with port names PA~PE. 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.

I/O Register List

Register Name	Bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PAWU	D7	D6	D5	D4	D3	D2	D1	D0
PAPU	D7	D6	D5	D4	D3	D2	D1	D0
PA	D7	D6	D5	D4	D3	D2	D1	D0
PAC	D7	D6	D5	D4	D3	D2	D1	D0
PBPU	D7	D6	D5	D4	D3	D2	D1	D0
PB	D7	D6	D5	D4	D3	D2	D1	D0
PBC	D7	D6	D5	D4	D3	D2	D1	D0
PCPU	—	D6	D5	D4	D3	D2	D1	D0
PC	—	D6	D5	D4	D3	D2	D1	D0
PCC	—	D6	D5	D4	D3	D2	D1	D0
PDPU	D7	D6	D5	D4	D3	D2	D1	D0
PD	D7	D6	D5	D4	D3	D2	D1	D0
PDC	D7	D6	D5	D4	D3	D2	D1	D0
PEPU	—	—	—	—	D3	D2	D1	D0
PE	—	—	—	—	D3	D2	D1	D0
PEC	—	—	—	—	D3	D2	D1	D0

Pull-high Resistors

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

PAPU 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   Port A bit 7 ~ bit 0 Pull-high Control
        0: Disable
        1: Enable

PBPU 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   Port B bit 7 ~ bit 0 Pull-high Control
        0: Disable
        1: Enable

PCPU Register

Bit	7	6	5	4	3	2	1	0
Name	—	D6	D5	D4	D3	D2	D1	D0
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 6~0   Port C bit 7 ~ bit 0 Pull-high Control
        0: Disable
        1: Enable

PDPU 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   Port D bit 7 ~ bit 0 Pull-high Control
        0: Disable
        1: Enable

PEPU Register

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

Bit 7~4   Unimplemented, read as "0"

Bit 3~0   Port E bit 3 ~ bit 0 Pull-high Control
      0: Disable
      1: Enable

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.

PAWU 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   Port A bit 7~bit 0 Wake-up Control
      0: Disable
      1: Enable

I/O Port Control Registers

Each I/O port has its own control register known as PAC~PEC, 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.

PAC 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	1	1	1	1	1	1	1	1

Bit 7~0   Port A bit 7 ~ bit 0 Input/Output Control
      0: Output
      1: Input

PBC 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	1	1	1	1	1	1	1	1

Bit 7~0   Port B bit 7 ~ bit 0 Input/Output Control
        0: Output
        1: Input

PCC Register

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

Bit 7   Unimplemented, read as "0"

Bit 6~0   Port C bit 7 ~ bit 0 Input/Output Control
        0: Output
        1: Input

PDC 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	1	1	1	1	1	1	1	1

Bit 7~0   Port D bit 7 ~ bit 0 Input/Output Control
        0: Output
        1: Input

PEC Register

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

Bit 7~4   Unimplemented, read as "0"

Bit 3~0   Port E bit 3 ~ bit 0 Input/Output Control
        0: Output
        1: Input

I/O Pin Structures

The accompanying diagrams illustrate the internal structures of some generic I/O pin types. As the exact logical construction of the I/O pin will differ from these drawings, they are supplied as a guide only to assist with the functional understanding of the I/O pins. The wide range of pin-shared structures does not permit all types to be shown.

Generic Input/Output Structure
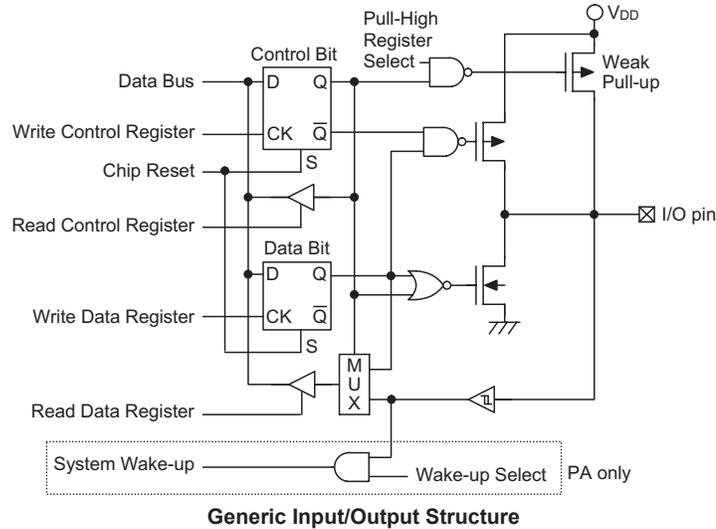
Programming Considerations

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

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

Timer Modules – TM

One of the most fundamental functions in any microcontroller device 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 Periodic TM sections.

Introduction

The device contains two TMs having a reference name of TM0 and TM1. Each individual TM can be categorised as a certain type, namely Compact Type TM or Periodic Type TM. Although similar in nature, the different TM types vary in their feature complexity. The common features to all of the Compact and Periodic 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	PTM
Timer/Counter	√	√
I/P Capture	—	√
Compare Match Output	√	√
PWM Channels	1	1
Single Pulse Output	—	1
PWM Alignment	Edge	Edge
PWM Adjustment Period & Duty	Duty or Period	Duty or Period

TM Function Summary

TM0	TM1
10-bit CTM	10-bit PTM

TM Name/Type Reference

TM Operation

The two 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 TnCK2~TnCK0 bits in the TM control registers. The clock source can be a ratio of either the system clock f_{SYS} or the internal high clock f_H, the f_{SUB} clock source or the external TCKn pin. The TCKn 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 Type and Periodic Type TMs each have two internal interrupts, one for each of the internal comparator A or comparator P, which generate a TM interrupt when a compare match condition occurs. When a TM interrupt is generated it can be used to clear the counter and also to change the state of the TM output pin.

TM External Pins

Each of the TMs, irrespective of what type, has one TM input pin, with the label TCKn. The TM input pin is essentially a clock source for the TM and is selected using the TnCK2~TnCK0 bits in the TMnC0/PTMnC0 register. This external TM input pin allows an external clock source to drive the internal TM. This external TM input pin is shared with other functions but will be connected to the internal TM if selected using the TnCK2~TnCK0 bits. The TM input pin can be chosen to have either a rising or falling active edge.

The TMs each have two output pins with the label TPn. 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 TPn output pin is also the pin where the TM generates the PWM output waveform. As the TM output pins are pin-shared with other function, the TM output function must first be setup using the CTRL0 register. A single bit in one of the registers determines if its associated pin is to be used as an external TM output pin or if it is to have another function. The number of output pins for each TM type is different, the details are provided in the accompanying table. All TM output pin names have a "_n" suffix. Pin names that include a "_0" or "_1" suffix indicate that they are from a TM with multiple output pins. This allows the TM to generate a complimentary output pair, selected using the I/O register data bits.

TM0	TM1	Register
TP0_0, TP0_1	TP1_0, TP1_1	CTRL0

TM Output Pins

TM Input/Output Pin Control Registers

Selecting to have a TM input/output or whether to retain its other shared functions is implemented using one register with a single bit in each register corresponding to a TM input/output pin. When the TMn is enabled, if the corresponding pin is setup as a TM input/output, and the complimentary output will be as a normal I/O pin.

CTRL0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	TP1CPS	TP0CPS
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2   Unimplemented, read as "0"
Bit 1   **TP1CPS**: TP1_0, TP1_1 pin selection
    0: TP1_0
    1: TP1_1
When TM1 is enabled, the output function of TP1_0 is timer, then the TP1_1 is I/O, and vice versa.
Bit 0   **TP0CPS**: TP0_0, TP0_1 pin selection
    0: TP0_0
    1: TP0_1
When TM0 is enabled, the output function of TP0_0 is timer, then the TP0_1 is I/O, and vice versa.

Programming Considerations

The TM Counter Registers and the Capture/Compare CCRA and CCRP registers, being 10-bit, all have a low and high byte structure. The high bytes can be directly accessed, but as the low bytes can only be accessed via an internal 8-bit buffer, reading or writing to these register pairs must be carried out in a specific way. The important point to note is that data transfer to and from the 8-bit buffer and its related low byte only takes place when a write or read operation to its corresponding high byte is executed. As the CCRA and CCRP registers are implemented in the way shown in the following diagram and accessing the register is carried out in a specific way described above, it is recommended to use the "MOV" instruction to access the CCRA and CCRP low byte registers, named TMxAL/PTMxAL and PTMxRPL, using the following access procedures. Accessing the CCRA or CCRP low byte register without following these access procedures will result in unpredictable values.
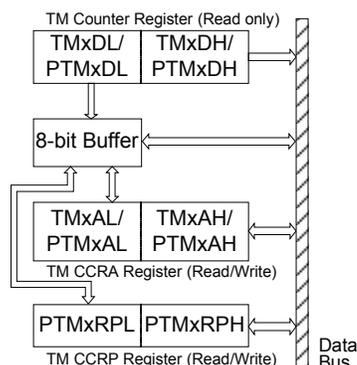
The following steps show the read and write procedures:

- Writing Data to CCRA or CCRP
    - Step 1. Write data to Low Byte TMxAL/PTMxAL or PTMxRPL
        - Note that here data is only written to the 8-bit buffer.
    - Step 2. Write data to High Byte TMxAH/PTMxAH or PTMxRPH
        - 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 or CCRP
    - Step 1. Read data from the High Byte TMxDH/PTMxDH, TMxAH/PTMxAH or PTMxRPH
        - 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 TMxDL/PTMxDL, TMxAL/PTMxAL or PTMxRPL
        - This step reads data from the 8-bit buffer.
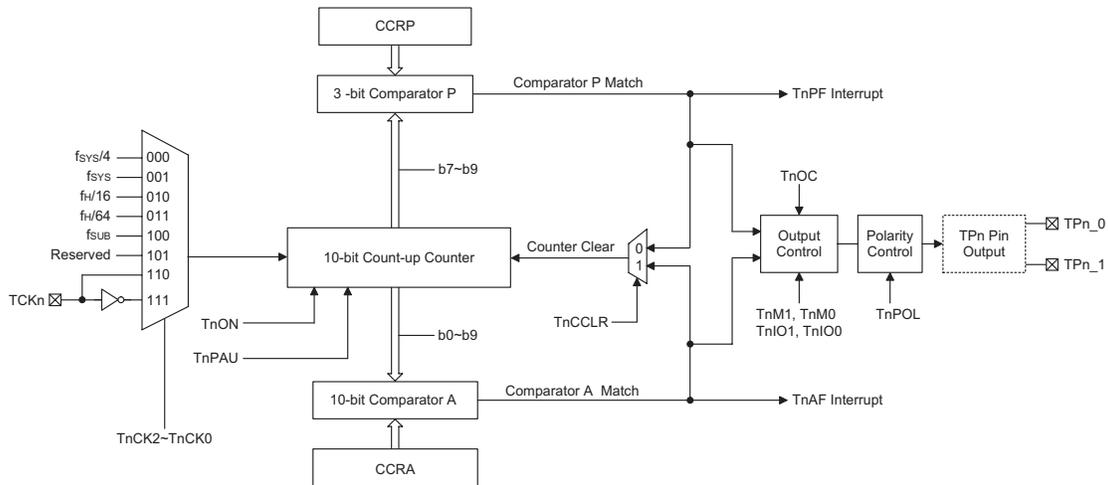
Compact Type TM – CTM

Although the simplest form of the two TM types, 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 be controlled with an external input pin and can drive two external output pins.

Name	TM No.	TM Input Pin	TM Output Pin
10-bit CTM	0	TCK0	TP0_0, TP0_1

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 TnON 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 TM 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 an output pin. All operating setup conditions are selected using relevant internal registers.

Compact Type TM Block Digram (n=0)

Compact Type TM Register Description

Overall operation of the Compact TM is controlled using six 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.

Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TMnC0	TnPAU	TnCK2	TnCK1	TnCK0	TnON	TnRP2	TnRP1	TnRP0
TMnC1	TnM1	TnM0	TnIO1	TnIO0	TnOC	TnPOL	TnDPX	TnCCLR
TMnDL	D7	D6	D5	D4	D3	D2	D1	D0
TMnDH	—	—	—	—	—	—	D9	D8
TMnAL	D7	D6	D5	D4	D3	D2	D1	D0
TMnAH	—	—	—	—	—	—	D9	D8

Compact TM Register List (n=0)

TMnC0 Register (n=0)

Bit	7	6	5	4	3	2	1	0
Name	TnPAU	TnCK2	TnCK1	TnCK0	TnON	TnRP2	TnRP1	TnRP0
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   **TnPAU**: TMn 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 TM 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   **TnCK2~TnCK0**: Select TMn Counter clock

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

These three bits are used to select the clock source for the TM. Selecting the Reserved clock input will effectively disable the internal counter. 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   **TnON**: TMn Counter On/Off Control

       0: Off
       1: On

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

Bit 2~0   **TnRP2~TnRP0**: TMn CCRP 3-bit register, compared with the TMn Counter bit 9~bit 7 Comparator P Match Period

    000: 1024 TMn clocks
    001: 128 TMn clocks
    010: 256 TMn clocks
    011: 384 TMn clocks
    100: 512 TMn clocks
    101: 640 TMn clocks
    110: 768 TMn clocks
    111: 896 TMn 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 TnCCLR bit is set to zero. Setting the TnCCLR 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.

TMnC1 Register (n=0)

Bit	7	6	5	4	3	2	1	0
Name	TnM1	TnM0	TnIO1	TnIO0	TnOC	TnPOL	TnDPX	TnCCLR
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   **TnM1~TnM0**: Select TMn Operating Mode

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

These bits setup the required operating mode for the TM. To ensure reliable operation the TM should be switched off before any changes are made to the TnM1 and TnM0 bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4   **TnIO1~TnIO0**: Select TPn_0, TPn_1 output function

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

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

In the Compare Match Output Mode, the TnIO1 and TnIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM 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 TM output pin should be setup using the TnOC bit in the TMnC1 register. Note that the output level requested by the TnIO1 and TnIO0 bits must be different from the initial value setup using the TnOC bit otherwise no change will occur on the TM output pin when a compare match occurs. After the TM output pin changes state, it can be reset to its initial level by changing the level of the TnON bit from low to high.

In the PWM Mode, the TnIO1 and TnIO0 bits determine how the TM 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 TnIO1 and TnIO0 bits only after the TMn has been switched off. Unpredictable PWM outputs will occur if the TnIO1 and TnIO0 bits are changed when the TM is running.

Bit 3   **TnOC**: TPn_0, TPn_1 Output control bit

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

PWM Mode
  0: Active low
  1: Active high

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

Bit 2   **TnPOL**: TPn_0, TPn_1 Output polarity Control
  0: Non-invert
  1: Invert

This bit controls the polarity of the TPn_0 or TPn_1 output pin. When the bit is set high the TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in the Timer/Counter Mode.

Bit 1   **TnDPX**: TMn 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   **TnCCLR**: Select TMn Counter clear condition
  0: TMn Comparatror P match
  1: TMn 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 TnCCLR 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 TnCCLR bit is not used in the PWM Mode.

TMnDL Register (n=0)

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**: TMn Counter Low Byte Register bit 7 ~ bit 0
TMn 10-bit Counter bit 7 ~ bit 0

TMnDH Register (n=0)

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**: TMn Counter High Byte Register bit 1 ~ bit 0
TMn 10-bit Counter bit 9 ~ bit 8

TMnAL Register (n=0)

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**: TMn CCRA Low Byte Register bit 7 ~ bit 0
TMn 10-bit CCRA bit 7 ~ bit 0

TMnAH Register (n=0)

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**: TMn CCRA High Byte Register bit 1 ~ bit 0
TMn 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 Mode or Timer/Counter Mode. The operating mode is selected using the TnM1 and TnM0 bits in the TMnC1 register.

Compare Match Output Mode

To select this mode, bits TnM1 and TnM0 in the TMnC1 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 TnCCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match occurs from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both TnAF and TnPF interrupt request flags for the Comparator A and Comparator P respectively, will both be generated.

If the TnCCLR bit in the TMnC1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the TnAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when TnCCLR is high no TnPF interrupt request flag will be generated. If the CCRA bits are all zero, the counter will overflow when its reaches its maximum 10-bit, 3FF Hex, value, however here the TnAF interrupt request flag will not be generated.

As the name of the mode suggests, after a comparison is made, the TM output pin will change state. The TM output pin condition however only changes state when an TnAF interrupt request flag is generated after a compare match occurs from Comparator A. The TnPF interrupt request flag, generated from a compare match occurs from Comparator P, will have no effect on the TM output pin. The way in which the TM output pin changes state are determined by the condition of the TnIO1 and TnIO0 bits in the TMnC1 register. The TM output pin can be selected using the TnIO1 and TnIO0 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 TM output pin, which is setup after the TnON bit changes from low to high, is setup using the TnOC bit. Note that if the TnIO1 and TnIO0 bits are zero then no pin change will take place.
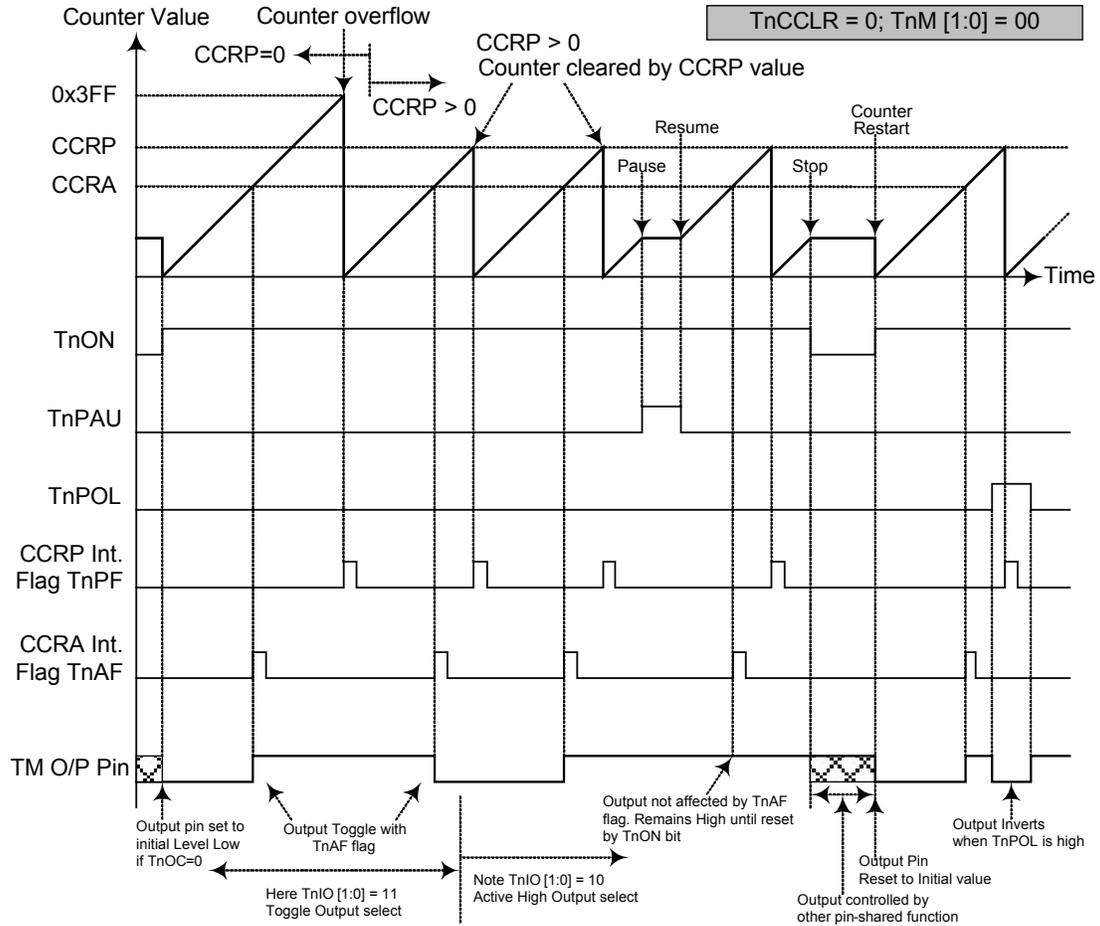
Compare Match Output Mode – TnCCLR = 0 (n=0)

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

2. The TM output pin is controlled only by the TnAF flag

3. The output pin is reset to its initial state by a TnON bit rising edge
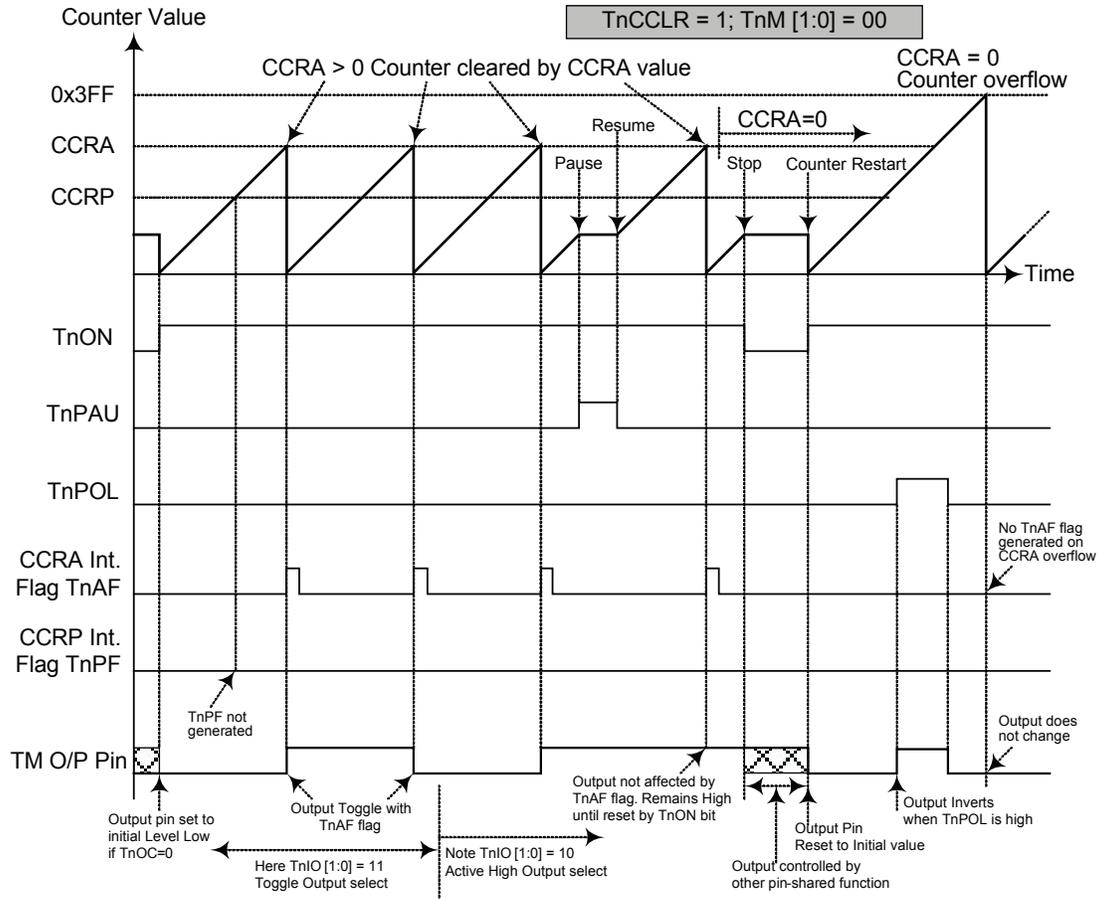
Compare Match Output Mode – TnCCLR = 1 (n=0)

Note: 1. With TnCCLR=1, a Comparator A match will clear the counter
2. The TM output pin is controlled only by the TnAF flag
3. The output pin is reset to its initial state by a TnON bit rising edge
4. The TnPF flag is not generated when TnCCLR=1

Timer/Counter Mode

To select this mode, bits TnM1 and TnM0 in the TMnC1 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 TM 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 TM 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 TnM1 and TnM0 in the TMnC1 register should be set to 10 respectively. The PWM function within the TM 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 TM 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 mode, the TnCCLR bit has no effect on the PWM operation. Both of the CCRA and CCRP registers are used to generate the PWM waveform, one register is used to clear the internal counter and thus control the PWM waveform frequency, while the other one is used to control the duty cycle. Which register is used to control either frequency or duty cycle is determined using the TnDPX bit in the TMnC1 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 TnOC bit in the TMnC1 register is used to select the required polarity of the PWM waveform while the two TnIO1 and TnIO0 bits are used to enable the PWM output or to force the TM output pin to a fixed high or low level. The TnPOL bit is used to reverse the polarity of the PWM output waveform.

- **CTM, PWM Mode, Edge-aligned Mode, TnDPX=0**

CCRP	001b	010b	011b	100b	101b	110b	111b	000b
Period	128	256	384	512	640	768	896	1024
Duty	CCRA							

If f_{SYS} = 16MHz, TM clock source is $f_{SYS}/4$, CCRP = 100b and CCRA = 128,

The CTM PWM output frequency = $(f_{SYS}/4)/512 = f_{SYS}/2048 = 7.8125$ kHz, duty = $128/512 = 25\%$.

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

- **CTM, PWM Mode, Edge-aligned Mode, TnDPX=1**

CCRP	001b	010b	011b	100b	101b	110b	111b	000b
Period	CCRA							
Duty	128	256	384	512	640	768	896	1024

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

PWM Mode – TnDPX = 0 (n=0)
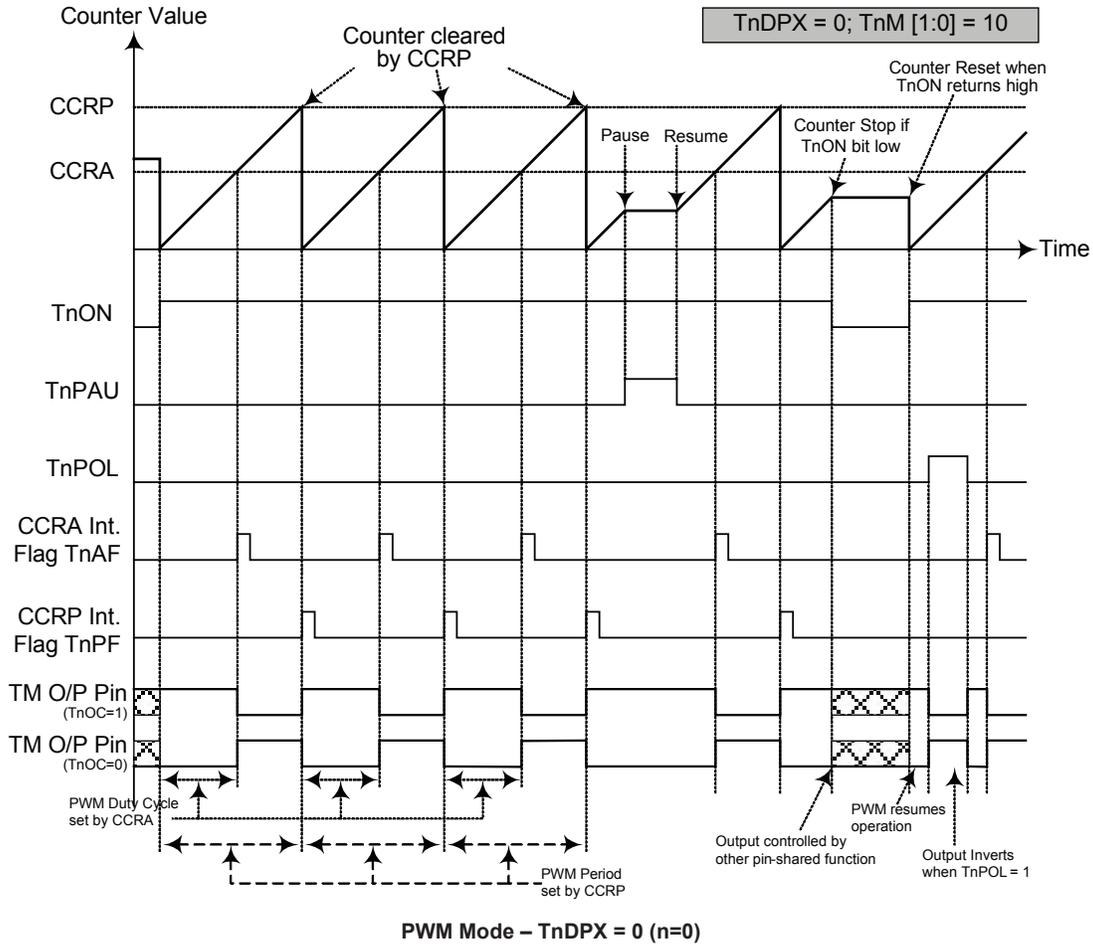
Note: 1. Here TnDPX=0 – Counter cleared by CCRP
  2. A counter clear sets the PWM Period
  3. The internal PWM function continues even when TnIO [1:0] = 00 or 01
  4. The TnCCLR bit has no influence on PWM operation

PWM Mode – TnDPX = 1 (n=0)
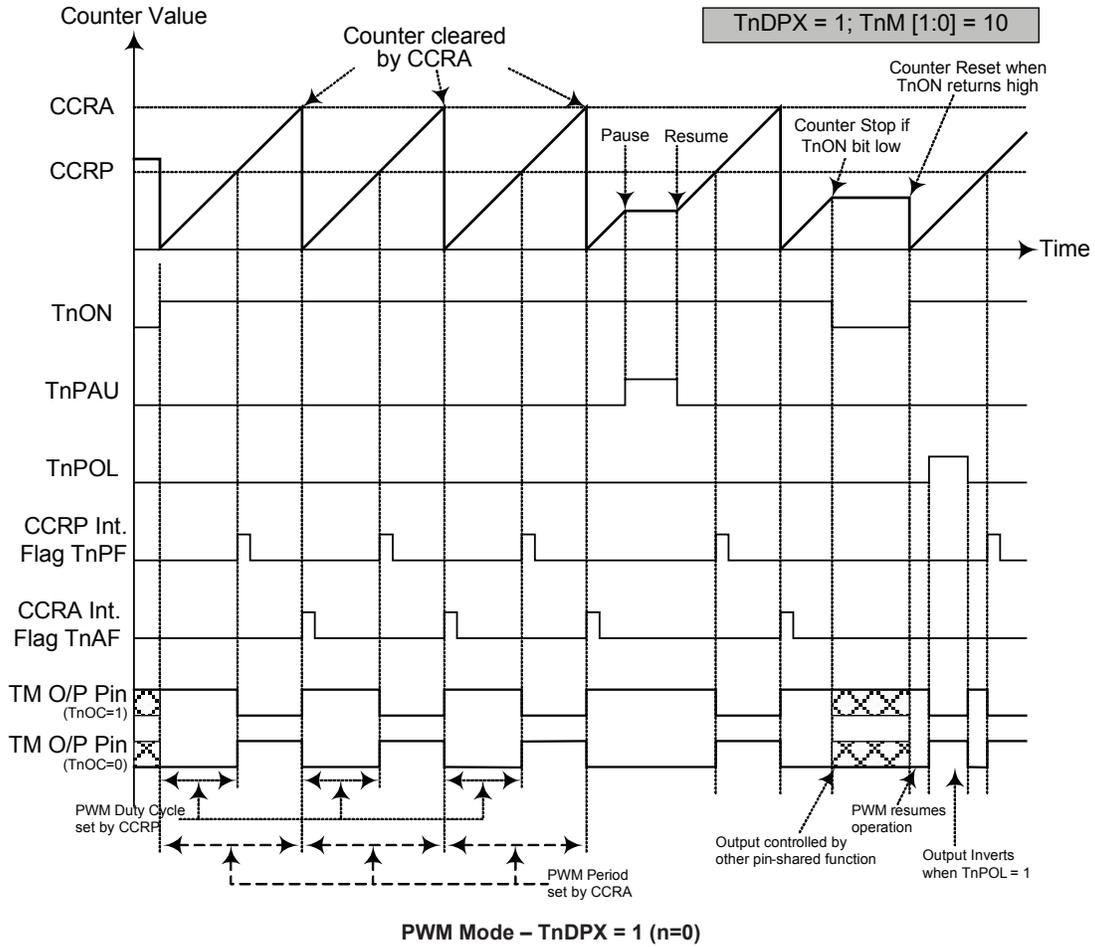
Note: 1. Here TnDPX = 1 – Counter cleared by CCRA
2. A counter clear sets the PWM Period
3. The internal PWM function continues even when TnIO [1:0] = 00 or 01
4. The TnCCLR bit has no influence on PWM operation
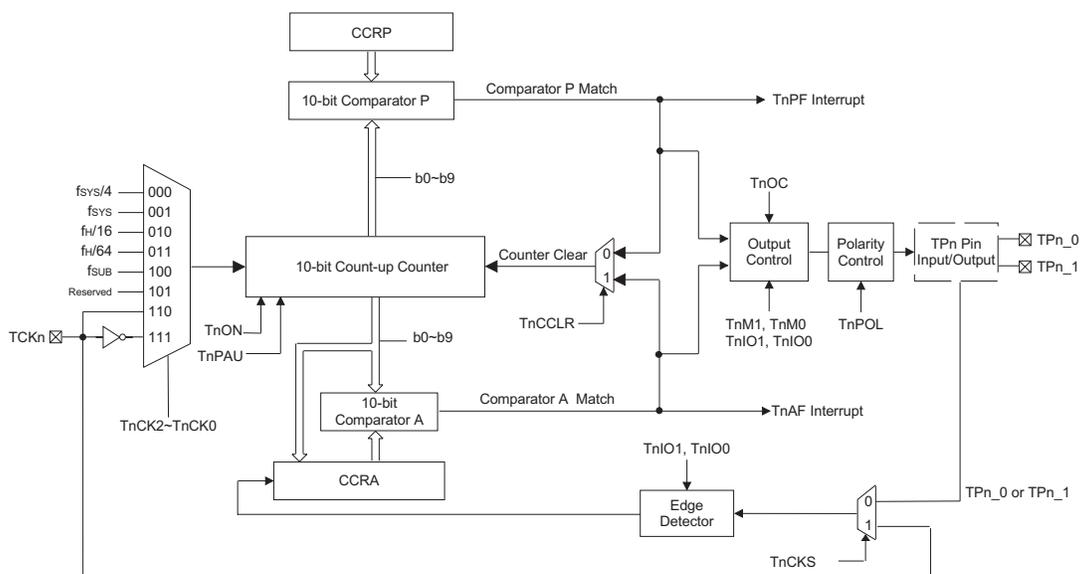
Periodic Type TM – PTM

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

Name	TM No.	TM Input Pin	TM Output Pin
10-bit PTM	1	TCK1	TP1_0, TP1_1

Periodic 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 two internal comparators with the names, Comparator A and Comparator P. These
comparators will compare the value in the counter with the CCRA and CCRP registers.

The only way of changing the value of the 10-bit counter using the application program, is to
clear the counter by changing the TnON 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 TM interrupt signal will also usually be generated. The Periodic
Type TM can operate in a number of different operational modes, can be driven by different clock
sources including an input pin and can also control the output pin. All operating setup conditions are
selected using relevant internal registers.

Periodic Type TM Block Diagram (n=1)

Periodic Type TM Register Description

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

Register Name	Bit							
	7	6	5	4	3	2	1	0
PTMnC0	TnPAU	TnCK2	TnCK1	TnCK0	TnON	—	—	—
PTMnC1	TnM1	TnM0	TnIO1	TnIO0	TnOC	TnPOL	TnCAPTS	TnCCLR
PTMnDL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnDH	—	—	—	—	—	—	D9	D8
PTMnAL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnAH	—	—	—	—	—	—	D9	D8
PTMnRPL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnRPH	—	—	—	—	—	—	D9	D8

10-bit Periodic TM Register List (n=1)

PTMnC0 Register (n=1)

Bit	7	6	5	4	3	2	1	0
Name	TnPAU	TnCK2	TnCK1	TnCK0	TnON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

Bit 7   **TnPAU**: TMn 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 TM 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   **TnCK2~TnCK0**: Select TMn Counter clock

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

These three bits are used to select the clock source for the TM. Selecting the Reserved clock input will effectively disable the internal counter. 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   **TnON**: TMn Counter On/Off Control

0: Off

1: On

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

Bit 2~0   Unimplemented, read as "0"

PTMnC1 Register (n=1)

Bit	7	6	5	4	3	2	1	0
Name	TnM1	TnM0	TnIO1	TnIO0	TnOC	TnPOL	TnCAPTS	TnCCLR
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   **TnM1~TnM0**: Select TMn Operation Mode

00: Compare Match Output Mode

01: Capture Input Mode

10: PWM Mode or Single Pulse Output Mode

11: Timer/Counter Mode

These bits setup the required operating mode for the TM. To ensure reliable operation the TM should be switched off before any changes are made to the TnM1 and TnM0 bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4   **TnIO1~TnIO0**: Select TPn_0, TPn_1 output function

Compare Match Output Mode

00: No change

01: Output low

10: Output high

11: Toggle output

PWM 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 TPn_0, TPn_1, TCKn

01: Input capture at falling edge of TPn_0, TPn_1, TCKn

10: Input capture at falling/rising edge of TPn_0, TPn_1, TCKn

11: Input capture disabled

Timer/counter Mode

Unused

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

In the Compare Match Output Mode, the TnIO1 and TnIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM 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 these bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the TnOC bit. Note that the output level requested by the TnIO1 and TnIO0 bits must be different from the initial value setup using the TnOC bit otherwise no change will occur on the TM output pin when a compare match occurs. After the TM output pin changes state, it can be reset to its initial level by changing the level of the TnON bit from low to high.

In the PWM Mode, the TnIO1 and TnIO0 bits determine how the TM 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 TnIO1 and TnIO0 bits only after the TM has been switched off. Unpredictable PWM outputs will occur if the TnIO1 and TnIO0 bits are changed when the TM is running.

Bit 3   **TnOC**: TPn_0, TPn_1 Output control bit

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

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

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

Bit 2   **TnPOL**: TPn_0, TPn_1 Output polarity Control
  0: Non-invert
  1: Invert

This bit controls the polarity of the TPn_0, TPn_1 output pin. When the bit is set high the TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in the Timer/Counter Mode.

Bit 1   **TnCAPTS**: TMn capture trigger source select
  0: From TPn_0, TPn_1 pin
  1: From TCKn pin

Bit 0   **TnCCLR**: Select TMn Counter clear condition
  0: TMn Comparatror P match
  1: TMn Comparatror A match

This bit is used to select the method which clears the counter. Remember that the Periodic TM contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the TnCCLR 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 TnCCLR bit is not used in the PWM, Single Pulse or Input Capture Mode.

PTMnDL Register (n=1)

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   **PTMnDL**: TMn Counter Low Byte Register bit 7 ~ bit 0
TMn 10-bit Counter bit 7 ~ bit 0

PTMnDH Register (n=1)

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   **PTMnDH**: TMn Counter High Byte Register bit 1 ~ bit 0
TMn 10-bit Counter bit 9 ~ bit 8

PTMnAL Register (n=1)

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   **PTMnAL**: TMn CCRA Low Byte Register bit 7 ~ bit 0
TMn 10-bit CCRA bit 7 ~ bit 0

PTMnAH Register (n=1)

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   **PTMnAH**: TMn CCRA High Byte Register bit 1 ~ bit 0
TMn 10-bit CCRA bit 9 ~ bit 8

PTMnRPL Register (n=1)

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   **PTMnRPL**: TMn CCRP Low Byte Register bit 7 ~ bit 0
TMn 10-bit CCRP bit 7 ~ bit 0

PTMnRPH Register (n=1)

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   **PTMnRPH**: TMn CCRP High Byte Register bit 1 ~ bit 0
                     TMn 10-bit CCRP bit 9 ~ bit 8

Periodic Type TM Operating Modes

The Periodic Type TM can operate in one of five operating modes, Compare Match Output Mode, PWM Output Mode, Single Pulse Output Mode, Capture Input Mode or Timer/Counter Mode. The operating mode is selected using the TnM1 and TnM0 bits in the PTMnC1 register.

Compare Match Output Mode

To select this mode, bits TnM1 and TnM0 in the PTMnC1 register, should be all cleared 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 TnCCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match occurs from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both the TnAF and TnPF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

If the TnCCLR bit in the PTMnC1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the TnAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when TnCCLR is high no TnPF interrupt request flag will be generated. In the Compare Match Output Mode, the CCRA can not be cleared to zero.

As the name of the mode suggests, after a comparison is made, the TM output pin, will change state. The TM output pin condition however only changes state when a TnAF interrupt request flag is generated after a compare match occurs from Comparator A. The TnPF interrupt request flag, generated from a compare match from Comparator P, will have no effect on the TM output pin. The way in which the TM output pin changes state are determined by the condition of the TnIO1 and TnIO0 bits in the PTMnC1 register. The TM output pin can be selected using the TnIO1 and TnIO0 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 TM output pin, which is setup after the TnON bit changes from low to high, is setup using the TnOC bit. Note that if the TnIO1, TnIO0 bits are zero then no pin change will take place.

Compare Match Output Mode – TnCCLR = 0 (n=1)

Note: 1. With TnCCLR = 0 – a Comparator P match will clear the counter
2. The TM output pin is controlled only by the TnAF flag
3. The output pin is reset to initial state by a TnON bit rising edge
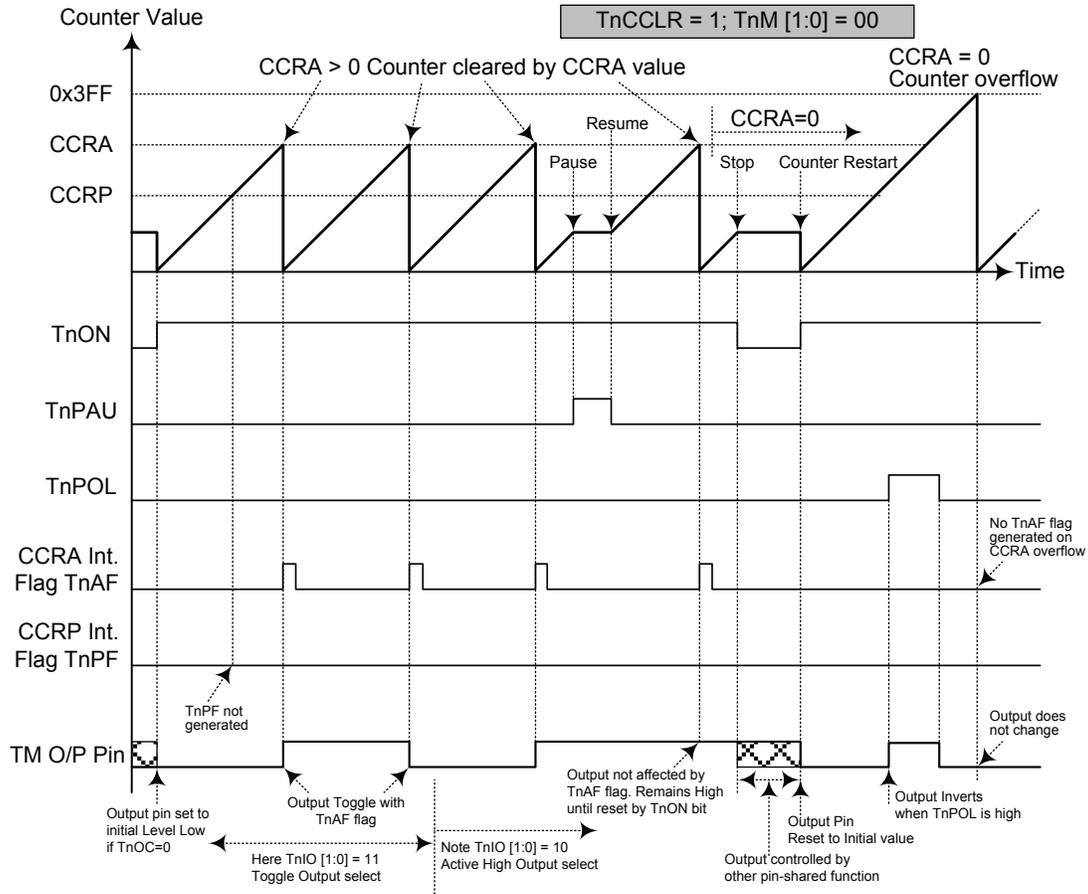
Compare Match Output Mode – TnCCLR = 1 (n=1)

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

2. The TM output pin is controlled only by the TnAF flag

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

4. The TnPF flag is not generated when TnCCLR = 1

Timer/Counter Mode

To select this mode, bits TnM1 and TnM0 in the PTMnC1 register should all 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 TM 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 TM 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 TnM1 and TnM0 in the PTMnC1 register should be set to 10 respectively and also the TnIO1 and TnIO0 bits should be set to 10 respectively. The PWM function within the TM 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 TM 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 mode, the TnCCLR bit has no effect as the PWM period. Both of the CCRP and CCRA registers are used to generate the PWM waveform, one register is used to clear the internal counter and thus control the PWM waveform frequency, while the other one is used to control the duty cycle. The PWM waveform frequency and duty cycle can therefore be controlled by the values in the CCRA and CCRP registers.

An interrupt flag, one for each of the CCRA and CCRP, will be generated when a compare match occurs from either Comparator A or Comparator P. The TnOC bit in the PTMnC1 register is used to select the required polarity of the PWM waveform while the two TnIO1 and TnIO0 bits are used to enable the PWM output or to force the TM output pin to a fixed high or low level. The TnPOL bit is used to reverse the polarity of the PWM output waveform.

- **10-bit PTM, PWM Mode**

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

If f_{SYS} = 16MHz, TM clock source select $f_{SYS}/4$, CCRP = 100b and CCRA = 128,

The PTM PWM output frequency = $(f_{SYS}/4) / 512 = f_{SYS}/2048$ =7.8125kHz, duty = 128/512 = 25%,

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

PWM Mode (n=1)

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

Single Pulse Output Mode

To select this mode, the required bit pairs, TnM1 and TnM0 should be set to 10 respectively and also the corresponding TnIO1 and TnIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the TM output pin.

The trigger for the pulse output leading edge is a low to high transition of the TnON bit, which can be implemented using the application program. However in the Single Pulse Mode, the TnON bit can also be made to automatically change from low to high using the external TCKn pin, which will in turn initiate the Single Pulse output. When the TnON bit transitions to a high level, the counter will start running and the pulse leading edge will be generated. The TnON bit should remain high when the pulse is in its active state. The generated pulse trailing edge will be generated when the TnON 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 TnON 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 TM interrupts. The counter can only be reset back to zero when the TnON bit changes from low to high when the counter restarts. In the Single Pulse Mode CCRP is not used. The TnCCLR bit is also not used.

**Single Pulse Generation (n=1)**
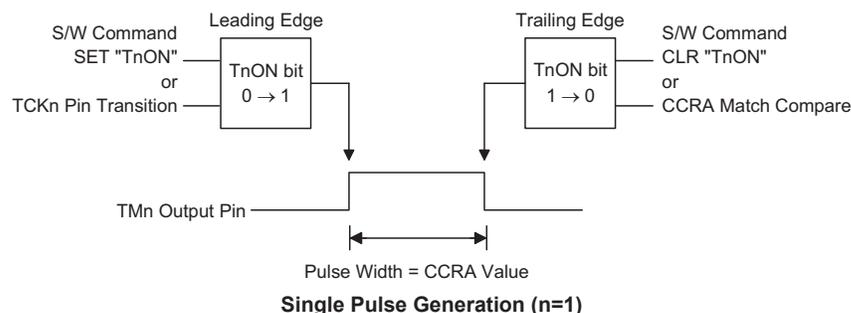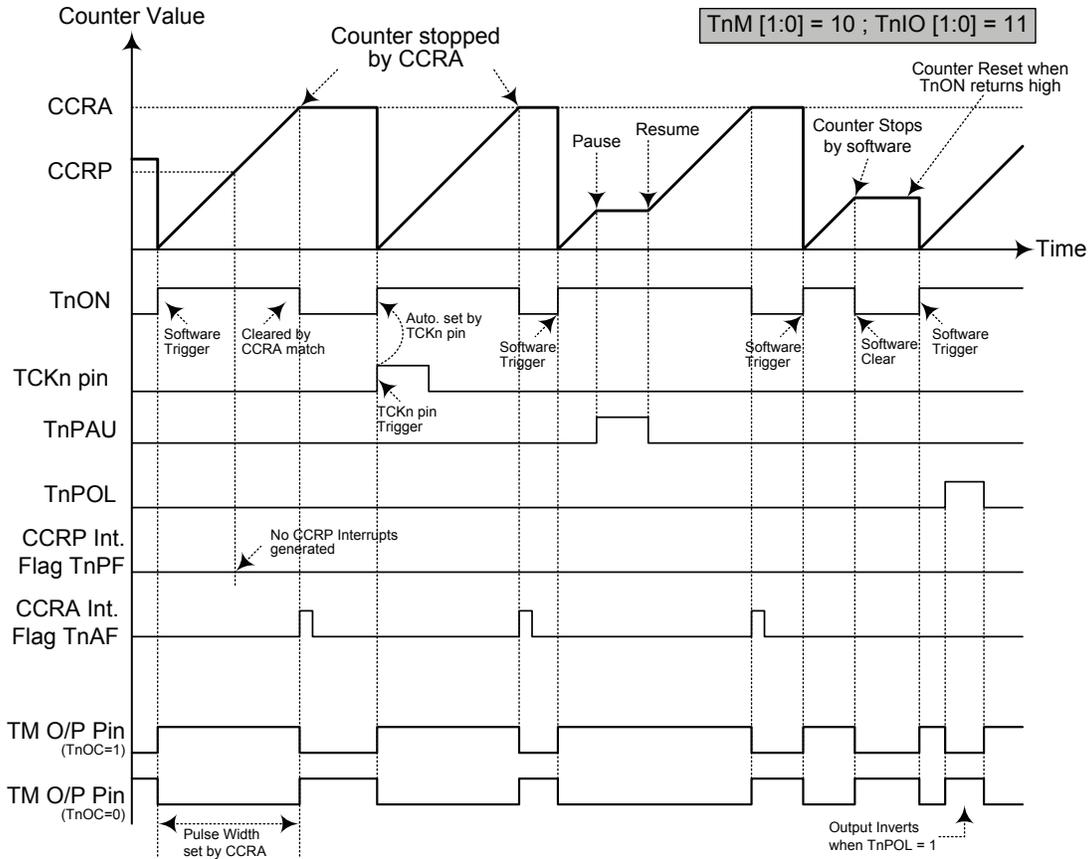
Single Pulse Mode (n=1)

Note: 1. Counter stopped by CCRA

2. CCRP is not used

3. The pulse is triggered by the TCKn pin or by setting the TnON bit high

4. A TCKn pin active edge will automatically set the TnON bit high

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

Capture Input Mode

To select this mode bits TnM1 and TnM0 in the PTMnC1 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 TPn_0, TPn_1 or TCKn pin, selected by the TnCAPTS bit in the PTMnC1 register. The input pin active edge can be either a rising edge, a falling edge or both rising and falling edges; the active edge transition type is selected using the TnIO1 and TnIO0 bits in the PTMnC1 register. The counter is started when the TnON bit changes from low to high which is initiated using the application program.

When the required edge transition appears on the TPn_0, TPn_1 or TCKn pin the present value in the counter will be latched into the CCRA register and a TM interrupt generated. Irrespective of what events occur on the TPn_0, TPn_1 or TCKn pin the counter will continue to free run until the TnON 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 TM 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 TnIO1 and TnIO0 bits can select the active trigger edge on the TPn_0, TPn_1 or TCKn pin to be a rising edge, falling edge or both edge types. If the TnIO1 and TnIO0 bits are both set high, then no capture operation will take place irrespective of what happens on the TPn_0, TPn_1 or TCKn pin, however it must be noted that the counter will continue to run.

As the TPn_0, TPn_1 or TCKn pin is pin shared with other functions, care must be taken if the TMn is in the Capture Input Mode. This is because if the pin is setup as an output, then any transitions on this pin may cause an input capture operation to be executed. The TnCCLR, TnOC and TnPOL bits are not used in this Mode.
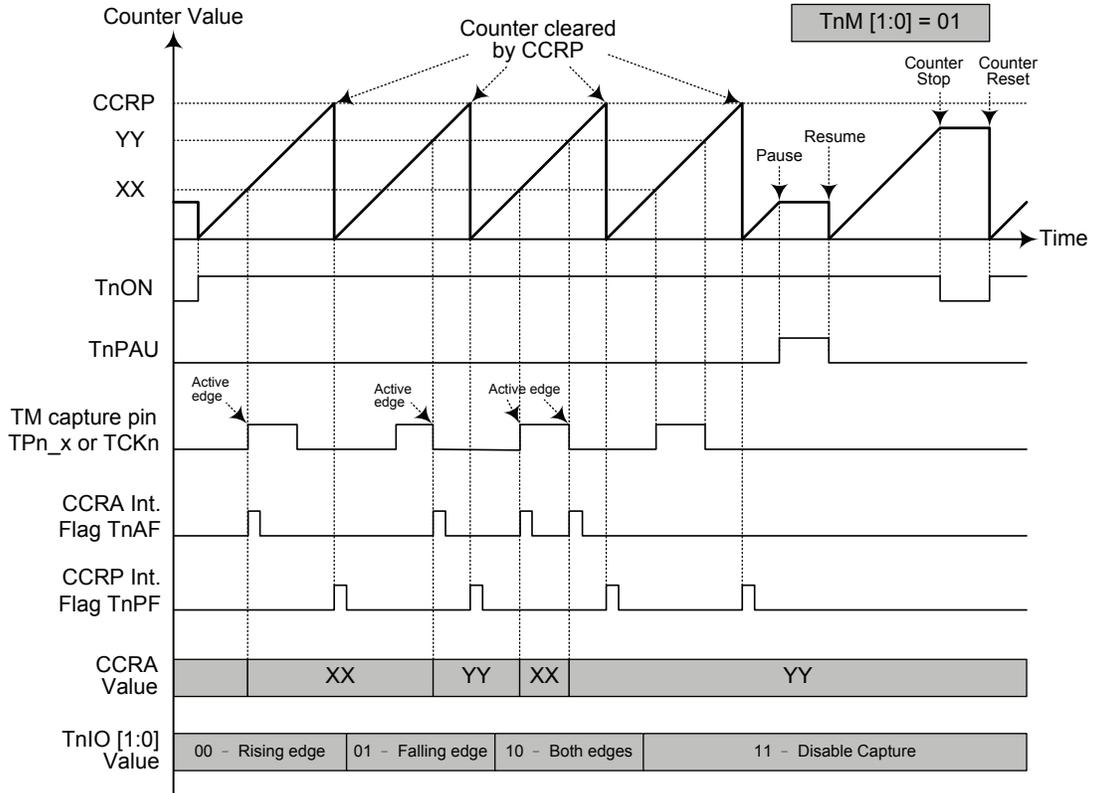
Capture Input Mode (n=1)

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

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

3. The TnCCLR bit is not used

4. No output function – TnOC and TnPOL bits are not used

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

Internal Power Supply

This device contains the LDO and VCM for the regulated power supply. The accompanying block diagram illustrates the basic functional operation. The internal LDO can provide the fixed voltage for PGA, ADC or the external components; as well the VCM can be used as the reference voltage for ADC module. There are four LDO voltage levels, 2.4V, 2.6V, 2.9V or 3.3V, decided by LDOVS1~LDOVS0 bits in the PWRC register, as well the VCM has two output voltage levels, 1.05V or 1.25V, selected by the VCMS bit in the PGAC1 register. The LDO and VCM functions can be controlled by the ENLDO, ENVCM and ADOFF bits respectively and can be powered off to reduce the power consumption.
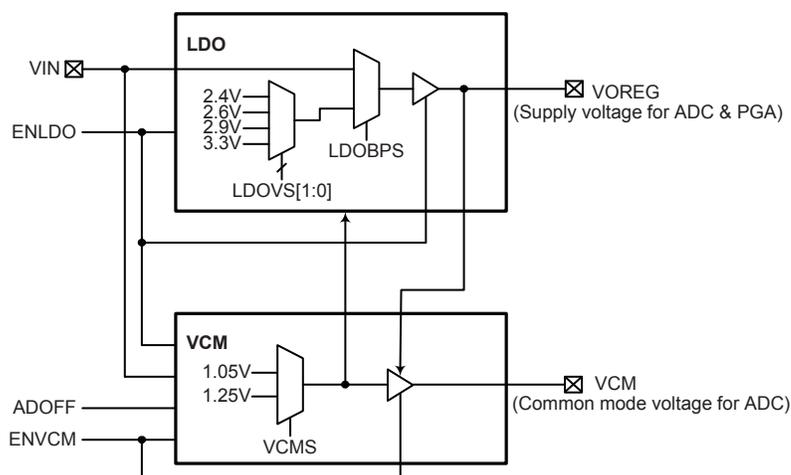
Internal Power Supply Block Diagram

Register Bits			Output Voltage	
ENLDO	ADOFF	ENVCM	VOREG	VCM
0	1	×	Disable	Enable
1	1	×	Enable	Disable
0	0	0	Disable	Disable
1	0	0	Enable	Disable (No external supply voltage on VOREG pin)
0	0	1	Disable	Enable (No external supply voltage on VOREG pin)
1	0	1	Enable	Enable

"x" means don't care

Power Control Table

PWRC Register

Bit	7	6	5	4	3	2	1	0
Name	ENLDO	ENVCM	—	—	—	LDOBPS	LDOVS1	LDOVS0
R/W	R/W	R/W	—	—	—	R/W	R/W	R/W
POR	0	0	—	—	—	0	0	0

Bit 7   **ENLDO**: LDO function control bit
         0: Disable
         1: Enable
         If the LDO is disabled, there will be no power consumption and LDO output pin is floating.

Bit 6   **ENVCM**: V_{CM} function control bit
         0: Disable
         1: Enable
         If the V_{CM} is disabled, there will be no power consumption and VCM output pin is floating.

Bit 5~3   Unimplemented, read as "0"

Bit 2   **LDOBPS**: LDO bypass function enable control
         0: Disable
         1: Enable

Bit 1~0   **LDOVS1~LDOVS0**: LDO output voltage selection
         00: 2.4V
         01: 2.6V
         10: 2.9V
         11: 3.3V

PGAC1 Register

Bit	7	6	5	4	3	2	1	0
Name	VCMS	INIS	—	—	DCSET2	DCSET1	DCSET0	—
R/W	R/W	R/W	—	—	R/W	R/W	R/W	—
POR	1	0	—	—	0	0	0	—

Bit 7   **VCMS**: Analog Common mode voltage selection
         0: 1.05V
         1: 1.25V

Bit 6   **INIS**: Selected input ends DI+/DI- internal connection control bit
         0: Not connected
         1: Connected

Bit 5~4   Unimplemented, read as "0"

Bit 3~1   **DCSET2~DCSET0**: DI+/DI- differential channel input offset selection
         000: DCSET = +0V
         001: DCSET = +0.25 × ΔVR_I
         010: DCSET = +0.5 × ΔVR_I
         011: DCSET = +0.75 × ΔVR_I
         100: DCSET = +0V
         101: DCSET = -0.25 × ΔVR_I
         110: DCSET = -0.5 × ΔVR_I
         111: DCSET = -0.75 × ΔVR_I
         The voltage, ΔVR_I, is the differential reference voltage which is amplified by specific gain selection based on the selected inputs.

Bit 0   Unimplemented, read as "0"

Analog to Digital Converter – ADC

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 Overview

This device contains a high accuracy multi-channel 24-bit Delta Sigma analog-to-digital ($\Sigma\Delta$A/D) 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 24-bit digital value.

In addition, the PGA gain control, ADC gain control and ADC reference gain control determine the amplification gain for ADC input signal. The designer can select the best gain combination for the desired amplification applied to the input signal. The following block diagram illustrates the ADC basic operational function. The ADC input channel can be arranged as two differential input channels. The input signal can be amplified by PGA before entering the 24-bit Delta Sigma ADC. The $\Sigma\Delta$ADC modulator will output one bit converted data to SINC filter which can transform the converted one-bit data to 24 bits and store them into the specific data registers. Additionally, this device also provides a temperature sensor to compensate the A/D converter deviation caused by the temperature. With high accuracy and performance, this device is very suitable for differential output sensor applications such as that found in weight measurement scales and other related products.

A/D Data Rate Definition

The Delta Sigma ADC data rate can be calculated from the equation below:

$$\text{Data Rate} = \frac{f_{ADCK}}{CHOP \times OSR} = \frac{f_{MCLK} / N}{CHOP \times OSR} = \frac{f_{MCLK}}{N \times CHOP \times OSR}$$

f_{ADCK}: A/D clock input, derived from f_{MCLK}/N

f_{MCLK}: A/D clock source, derived from f_{SYS} or f_{SYS}/2 (ADCK+1) using the ADCK bit field.

N: a constant divided factor and can be equal to 30 or 12 determined by the FLMS bit field.

CHOP: Sampling data amount doubling function control and can be equal to 2 or 1 determined by the FLMS bit field.

OSR: Oovrsampling rate determined by the ADOR field.

For example, if a data rate of 10Hz is desired, an f_{MCLK} clock source with a frequency of 4.9152MHz ADC can be selected. Then set the FLMS field to "000" to obtain an "N" equal to 30 and "CHOP" equal to 2. Finally, set the ADOR field to "001" to select an oversampling rate equal to 8192. Therefore, the Data Rate = 4.9152MHz / (30 × 2 × 8192) = 10Hz.

Note that the A/D converter has a notch rejection function for an A/C power supply with a frequency of 50Hz or 60Hz when the data rate is equal to 10Hz.
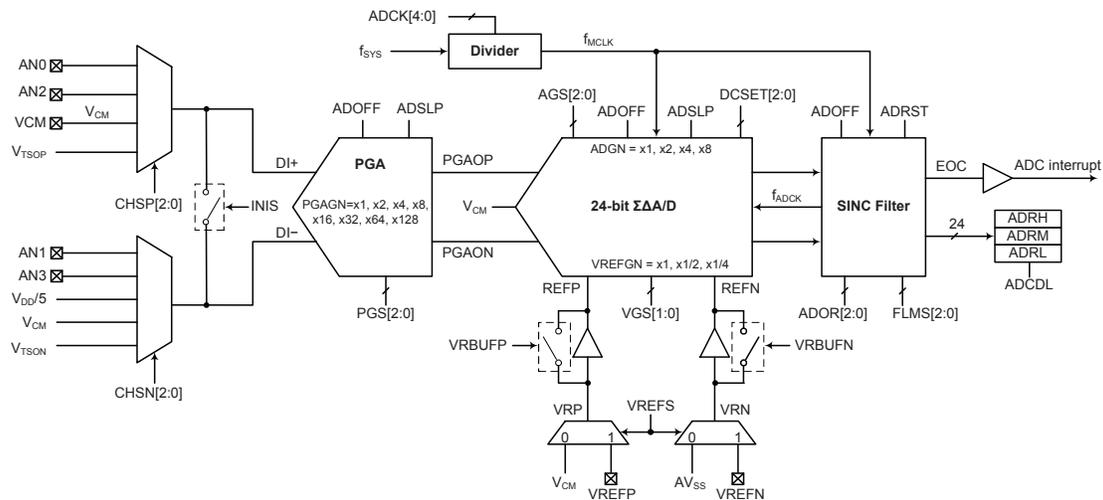
A/D Converter Structure

A/D Converter Register Description

Overall operation of the A/D converter is controlled by using 10 registers. Three read only registers exist to store the ADC data 24-bit value. A control register named as PWRC is used to control the required bias and supply voltages for PGA and ADC and described in the "Internal Power Supply" section. The remaining 6 registers are control registers which set up the gain selections and control functions of the A/D converter.

Register Name	Bit							
	7	6	5	4	3	2	1	0
PGAC0	—	VGS1	VGS0	AGS1	AGS0	PGS2	PGS1	PGS0
PGAC1	VCMS	INIS	—	—	DCSET2	DCSET1	DCSET0	—
PGACS	—	—	CHSN2	CHSN1	CHSN0	CHSP2	CHSP1	CHSP0
ADRL	D7	D6	D5	D4	D3	D2	D1	D0
ADRM	D15	D14	D13	D12	D11	D10	D9	D8
ADRH	D23	D22	D21	D20	D19	D18	D17	D16
ADCR0	ADRST	ADSLP	ADOFF	ADOR2	ADOR1	ADOR0	—	VREFS
ADCR1	FLMS2	FLMS1	FLMS0	VRBUFN	VRBUFP	ADCDL	EOC	—
ADCS	—	—	—	ADCK4	ADCK3	ADCK2	ADCK1	ADCK0

A/D Converter Register List

Programmable Gain Amplifier – PGA

There are three registers related to the programmable gain control, PGAC0, PGAC1 and PGACS. The PGAC0 resister is used to select the PGA gain, ADC gain and the ADC reference gain. As well, the PGAC1 register is used to define the input connection, differential input offset voltage adjustment control and the VCM voltage selection. In addition, The PGACS register is used to select the input ends for the PGA. Therefore, the input channels have to be determined by the CHSP2~0 and CHSN2~0 bits to determine which analog channel input pins, temperature detector inputs or internal power supply are actually connected to the internal differential A/D converter.

- **PGAC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	VGS1	VGS0	AGS1	AGS0	PGS2	PGS1	PGS0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7   Unimplemented, read as "0"

Bit 6~5   **VGS1~VGS0**: REFP/REFN Differential reference voltage gain selection
               00: VREFGN=1
               01: VREFGN=1/2
               10: VREFGN=1/4
               11: Reserved

Bit 4~3   **AGS1~AGS0**: A/D converter PGAOP/PGAON Differential input signal gain selection
               00: ADGN=1
               01: ADGN=2
               10: ADGN=4
               11: ADGN=8

Bit 2~0   **PGS2~PGS0**: PGA D+/D- Differential channel input gain selection
               000: PGAGN=1
               001: PGAGN=2
               010: PGAGN=4
               011: PGAGN=8
               100: PGAGN=16
               101: PGAGN=32
               110: PGAGN=64
               111: PGAGN=128

- **PGAC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	VCMS	INIS	—	—	DCSET2	DCSET1	DCSET0	—
R/W	R/W	R/W	R/W	—	—	R/W	R/W	R/W	—
POR	1	0	—	—	0	0	0	—

Bit 7　　　**VCMS**: Analog Common mode voltage selection
　　　　　　0: 1.05V
　　　　　　1: 1.25V

Bit 6　　　**INIS**: Selected input ends DI+/DI- internal connection control bit
　　　　　　0: Not connected
　　　　　　1: Connected

Bit 5~4　　Unimplemented, read as "0"

Bit 3~1　　**DCSET2~DCSET0**: DI+/DI- differential channel input offset selection
　　　　　　000: DCSET = +0V
　　　　　　001: DCSET = +0.25 × ΔVR_I
　　　　　　010: DCSET = +0.5 × ΔVR_I
　　　　　　011: DCSET = +0.75 × ΔVR_I
　　　　　　100: DCSET = +0V
　　　　　　101: DCSET = -0.25 × ΔVR_I
　　　　　　110: DCSET = -0.5 × ΔVR_I
　　　　　　111: DCSET = -0.75 × ΔVR_I
　　　　　　The voltage, ΔVR_I, is the differential reference voltage which is amplified by specific gain selection based on the selected inputs.

Bit 0　　　Unimplemented, read as "0"

- **PGACS Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	CHSN2	CHSN1	CHSN0	CHSP2	CHSP1	CHSP0
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 5~3　　**CHSN2~CHSN0**: PGA negative input end selection
　　　　　　000: AN1
　　　　　　001: AN3
　　　　　　010~100: Reserved
　　　　　　101: $V_{DD}/5$
　　　　　　110: V_{CM}
　　　　　　111: Temperature sensor output – V_{TSON}
　　　　　　These bits are used to select the negative input for the PGA. If the DI- input is selected as a single end input, the The V_{CM} voltage must be selected as the positive input on DI+ for single end input PGA applications. It is recommended that when the V_{TSON} signal is selected as the negative input, the V_{TSOP} signal should be selected as the positive input for proper operations.

Bit 2~0   **CHSP2~CHSP0**: PGA positive input end selection
        000: AN0
        001: AN2
        010~101: Reserved
        110: V_{CM}
        111: Temperature sensor output – V_{TSOP}

These bits are used to select the positive input for the PGA. If the DI+ input is selected as a single end input, the The V_{CM} voltage must be selected as the negative input on DI- for single end input PGA applications. It is recommended that when the V_{TSOP} signal is selected as the positive input, the V_{TSON} signal should be selected as the negative input for proper operations.

A/D Converter Data Registers – ADRL, ADRM, ADRH

As the device contains a 24-bit A/D converter, it requires three data registers to store the converted value. These are a high byte register, known as ADRH, a middle byte byte register, known as ADRM and a low byte register, known as ADRL. After the conversion process takes place, these registers can be directly read by the microcontroller to obtain the digitised conversion value. D0~D23 are the A/D conversion result data bits.

- **ADRH Register**

Bit	7	6	5	4	3	2	1	0
Name	D23	D22	D21	D20	D19	D18	D17	D16
R/W	R	R	R	R	R	R	R	R
POR	×	×	×	×	×	×	×	×

"x" unknown

Bit 7~0   A/D conversion data Register bit 23~bit 16

- **ADRM 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	×	×	×	×	×	×	×	×

"x" unknown

Bit 7~0   A/D conversion data Register bit 15~bit 8

- **ADRL 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	×	×	×	×	×	×	×	×

"x" unknown

Bit 7~0   A/D conversion data Register bit 7~bit 0

A/D Converter Control Registers – ADCR0, ADCR1, ADCS

To control the function and operation of the A/D converter, three control registers known as ADCR0, ADCR1 and ADCS are provided. These 8-bit registers define functions such as the selection of which reference source is used to the internal ADC, the ADC clock source , the ADC output data rate as well as controlling the power-up function and monitoring the ADC end of conversion status.

- **ADCR0 Register**

Bit	7	6	5	4	3	2	1	0
Name	ADRST	ADSLP	ADOFF	ADOR2	ADOR1	ADOR0	—	VREFS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	—	R/W
POR	0	0	1	0	0	0	—	0

Bit 7   **ADRST**: A/D converter software reset enable control

     0: Disable

     1: Enable

This bit is used to reset the A/D converter internal digital SINC filter. This bit is set low for A/D normal operations. However, if set high, the internal digital SINC filter will be reset and the current A/D converted data will be aborted. A new A/D data conversion process will not be initiated until this bit is set low again.

Bit 6   **ADSLP**: A/D converter sleep mode enable control

     0: Normal mode

     1: Sleep mode

This bit is used to determine whether the A/D converter enters the sleep mode or not when the A/D converter is powered on by setting the ADOFF bit low. When the A/D converter is powered on and the ADSLP bit is low, the A/D converter will operate normally. However, the A/D converter will enter the sleep mode if the ADSLP bit is set high as the A/D converter has been powered on. The whole A/D converter circuit will be switched off except the PGA and VCM generator to reduce the power consumption and VCM start-up stable time.

Bit 5   **ADOFF**: A/D converter module power on/off control

     0: Power on

     1: Power off

This bit controls the ADC module power on/off function. This bit should be cleared to zero to enable the A/D converter. If the bit is set high then the ADC will be switched off reducing the device power consumption. As the ADC will consume a limited amount of power, even when not executing a conversion, this may be an important consideration in power sensitive battery powered applications.

It is recommended to set the ADOFF bit high before the device enters the IDLE/SLEEP mode for saving power. Setting the ADOFF bit high will power down the A/D converter module regardless of the ADSLP and ADRST bit settings.

Bit 4 ~ 2   **ADOR2~ADOR0**: A/D conversion oversampling rate selection

     000: Oversampling rate OSR=16384

     001: Oversampling rate OSR=8192

     010: Oversampling rate OSR=4096

     011: Oversampling rate OSR=2048

     100: Oversampling rate OSR=1024

     101: Oversampling rate OSR=512

     110: Oversampling rate OSR=256

     111: Oversampling rate OSR=128

Bit 1   Unimplemented, read as "0"

Bit 0   **VREFS**: A/D converter reference voltage pair selection

     0: Internal reference pair – V_{CM} & AV_{SS}

     1: External reference pair – VREFP & VREFN

- **ADCR1 Register**

Bit	7	6	5	4	3	2	1	0
Name	FLMS2	FLMS1	FLMS0	VRBUFN	VRBUFP	ADCDL	EOC	—
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 ~ 5   **FLMS2~FLMS0**: A/D converter clock divided ratio selection and sampled data doubling function (CHOP) enable control
    000: CHOP=2, $f_{ADCK}=f_{MCLK}/30$
    010: CHOP=2, $f_{ADCK}=f_{MCLK}/12$
    100: CHOP=1, $f_{ADCK}=f_{MCLK}/30$
    110: CHOP=1, $f_{ADCK}=f_{MCLK}/12$
    Others: Reserved

When the CHOP bit is equal to 2, it means that the sampled data amount will be doubled for the normal conversion mode. However, it can be regarded as the low latency conversion mode if the CHOP bit is equal to 1, which means that the sampled data doubling function is disabled.

Bit 4   **VRBUFN**: A/D converter negative reference voltage input buffer control
    0: Disable input buffer and enable bypass function
    1: Enable input buffer and disable bypass function

Bit 3   **VRBUFP**: A/D converter positive reference voltage input buffer control
    0: Disable input buffer and enable bypass function
    1: Enable input buffer and disable bypass function

Bit 2   **ADCDL**: A/D converted data latch function enable control
    0: Disable data latch function
    1: Enable data latch function

If the ADC converted data latch function is enabled, the latest converted data value will be latched and not be updated by any subsequent converted results until this function is disabled. Although the converted data is latched into the data registers, the ADC circuits remain operational, but will not generate interrupt and EOC will not change. It is recommended that this bit should be set high before reading the converted data in the ADRL, ADRM and ADRH registers. After the converted data has been read out, the bit can then be cleared to low to disable the ADC data latch function and allow further conversion values to be stored. In this way, the possibility of obtaining undesired data during ADC conversions can be prevented.

Bit 1   **EOC**: End of A/D conversion flag
    0: A/D conversion in progress
    1: A/D conversion ended
This bit must be cleared by software.

Bit 0   Unimplemented, read as "0"

- **ADCS Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	ADCK4	ADCK3	ADCK2	ADCK1	ADCK0
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   **ADCK4~ADCK0**: A/D converter clock source f_{MCLK} divided ratio selection
     00000~11110: $f_{MCLK}=f_{SYS}$ / [2 × (ADCK field value+1)]
     11111: $f_{MCLK}=f_{SYS}$

The A/D converter clock source, f_{MCLK}, is typically designed as 4.9152MHz. However, the MCU might be selected to work with a different system clock. Therefore, the designer should use the ADCK4~ADCK0 bits to obtain a fixed 4.9152MHz A/D converter clock source. For example, if the system clock is 9.8304MHz, the ADCK field must be set to 0 to obtain an f_{MCLK} = 4.9152MHz.

A/D Operation

The ADC provides four operatiing modes, which are the Normal mode, Power down mode, Sleep mode and Reset mode, controlled respectively by the ADOFF, ADSLP and ADRST bits in the ADCR0 register. The following table illustrates the operating mode selection.

ENLDO	ADOFF	ADSLP	ADRST	Operating Mode	Description
0	1	x	x	Power down mode	Bandgap off, LDO off, V_{CM} generator off, PGA off, ADC off, V_{IN}/5 bias generator off, Temperature sensor off, VRN/VRP buffer off, SINC filter off
1	1	x	x	Power down mode	Bandgap on, LDO on, V_{CM} generator off, PGA off, ADC off, V_{IN}/5 bias generator off, Temperature sensor off, VRN/VRP buffer off, SINC filter off
0	0	1	x	Sleep mode (External voltage must be supplied on LDO output pin)	Bandgap on, LDO off, V_{CM} generator off, PGA on, ADC off, V_{IN}/5 bias generator off, Temperature sensor off, VRN/VRP buffer off, SINC filter on
0	0	0	0	Normal mode (External voltage must be supplied on LDO output pin)	Bandgap on, LDO off, V_{CM} generator on/off [1], PGA on, ADC on, V_{IN}/5 bias generator on/off [2], Temperature sensor on/off [3], VRN/VRP buffer on/off [4], SINC filter on
0	0	0	1	Reset mode (External voltage must be supplied on LDO output pin)	Bandgap on, LDO off, V_{CM} generator on/off [1], PGA on, ADC on, V_{IN}/5 bias generator on/off [2], Temperature sensor on/off [3], VRN/VRP buffer on/off [4], SINC filter Reset
1	0	1	x	Sleep mode	Bandgap on, LDO on, V_{CM} generator off, PGA on, ADC off, V_{IN}/5 bias generator off, Temperature sensor off, VRN/VRP buffer off, SINC filter on
1	0	0	0	Normal mode	Bandgap on, LDO on, V_{CM} generator on/off [1], PGA on, ADC on, V_{IN}/5 bias generator on/off [2], Temperature sensor on/off [3], VRN/VRP buffer on/off [4], SINC filter on
1	0	0	1	Reset mode	Bandgap on, LDO on, V_{CM} generator on/off [1], PGA on, ADC on, V_{IN}/5 bias generator on/off [2], Temperature sensor on/off [3], VRN/VRP buffer on/off [4], SINC filter Reset

A/D operation mode selection

Note: 1. The V_{CM} generator can be switched on or off by configuring the ENVCM bit.

2. The $V_{IN}/5$ bias generator can be switched on or off by configuring the CHSN[2:0] bits.

3. The temperature sensor can be switched on or off by configuring the CHSN[2:0] or CHSP[2:0] bits.

4. The VRN buffer can be switched on or off by the configuring VRBUFN bit while the VRP buffer can be switched on or off by the configuring VRBUFP bit.

5. "x": unknown.

To enable the ADC, the first step is to disable the ADC power down and sleep mode by clearing the ADOFF and ADSLP bits to make sure the ADC is powered up. The ADRST bit in the ADCR0 register is used to start and reset the A/D converter after power on. When the microcontroller sets this bit from low to high and then low again, an analog to digital converted data in SINC filter will be initiated. After this setup is complete, the ADC is ready for operation. These three bits are used to control the overall start operation of the internal analog to digital converter.

The EOC bit in the ADCR1 register is used to indicate when the analog to digital conversion process is complete. This bit will be automatically set high by the microcontroller after a conversion cycle has ended. 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 EOC bit in the ADCR1 register to check whether it has been set "1" as an alternative method of detecting the end of an A/D conversion cycle. The ADC converted data will be updated continuously by the new converted data. If the ADC converted data latch function is enabled, the latest converted data will be latched and the following new converted data will be discarded until this data latch function is disabled.

The clock source for the A/D converter should be typically fixed at a value of 4MHz, which originates from the system clock f_{SYS}, and can be chosen to be either f_{SYS} or a subdivided version of f_{SYS}. The division ratio value is determined by the ADCK4~ADCK0 bits in the ADCS register to obtain a 4MHz clock source for the ADC.

The differential reference voltage supply to the A/D Converter can be supplied from either the internal power supply, V_{CM} and AV_{SS}, or from an external reference source supplied on pins, VREFP and VREFN. The desired selection is made using the VREFS bit in the ADCR0 register.

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

    Enable the power LDO, VCM for PGA and ADC.

- Step 2

    Select the PGA, ADC and reference voltage gains by PGAC0 register.

- Step 3

    Select the PGA setting for input pins connection, V_{CM} voltage level and buffer option by PGAC1 register.

- Step 4

    Select the required A/D conversion clock source 4.9152MHz by correctly programming bits ADCK4~ADCK0 in the ADCS register.

- Step 5

    Select output data rate by cofiguring the ADOR[2:0] bits in the ADCR0 register and FLMS[2:0] bits in the ADCR1 register.

- Step 6

    Select which channel is to be connected to the internal PGA by correctly programming the CHSP2~CHSP0 and CHSN2~CHSN0 bits which are also contained in the PGACS register.

- Step 7

    Release the power down mode and sleep mode by clearing the ADOFF and ADSLP bits in ADCR0 register.

- Step 8

    Reset the A/D by setting the ADRST to high in the ADCR0 register and clearing this bit to zero to release reset status.

- Step 9

    If the interrupts are to be used, the interrupt control registers must be correctly configured to ensure the A/D converter interrupt function is active. The master interrupt control bit, EMI, and the A/D converter interrupt bit, ADE, must both be set high to do this.

- Step 10

    To check when the analog to digital conversion process is complete, the EOC bit in the ADCR1 register can be polled. The conversion process is complete when this bit goes high. When this occurs the A/D data registers ADRL, ADRM and ADRH can be read to obtain the conversion value. As an alternative method, if the interrupts are enabled and the stack is not full, the program can wait for an A/D interrupt to occur.

Note: When checking for the end of the conversion process, if the method of polling the EOC bit in the ADCR1 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 setting bit ADOFF high in the ADCR0 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.

A/D Transfer Function

This device contains a 24-bit $\Sigma\Delta$A/D converter and its full-scale converted digitised value is from 8388607 to -8388608 in decimal value. The converted data format is formed by a two's complement binary value. The MSB of the converted data is the signed bit. Since the full-scale analog input value is equal to the amplified value of the V_{CM} or differential reference input voltage, ΔVR_I, selected by the VREFS bit in ADCR0 register, this gives a single bit analog input value of V_{CM} or differential reference input voltage divided by 8388608.

$$1\ LSB = \Delta VR\_I / 8388608$$

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

$$\Delta SI\_I = (PGAGN \times ADGN \times \Delta DI\pm) + DCSET$$

$$\Delta VR\_I = VREGN \times \Delta VR\pm$$

$$ADC\_Conversion\_Data = (\Delta SI\_I \div \Delta VR\_I) \times K$$

Where K is equal to 2^{23}

Note: 1. The PGAGN, ADGN, VREGN values are decided by the PGS, AGS, VGS control bits.

2. ΔSI_I: Differential Input Signal after amplification and offset adjustment

3. PGAGN: Programmable Gain Amplifier gain

4. ADGN: A/D Converter gain

5. VREGN: Reference voltage gain

6. $\Delta DI\pm$: Differential channel input signal

7. DCSET: Offset voltage

8. $\Delta VR\pm$: Differential Reference voltage

9. ΔVR_I: Differential Reference input voltage after amplification

10. VREGN: Reference voltage gain

Due to the digital system design of the $\Sigma\Delta$A/D Converter, the maximum number of the A/D converted value is 8388607 and the minimum value is -8388608, therefore, we can have the middle number 0. The ADC_Conversion_Data equation illustrates this range of converted data variation.

A/D conversion data (2's compliment, Hexadecimal)	Decimal Value
0x7FFFFF	8388607
0x800000	-8388608

The following diagram shows the relationship between the DC input value and the ADC converted data which is presented by the Two's Complement.
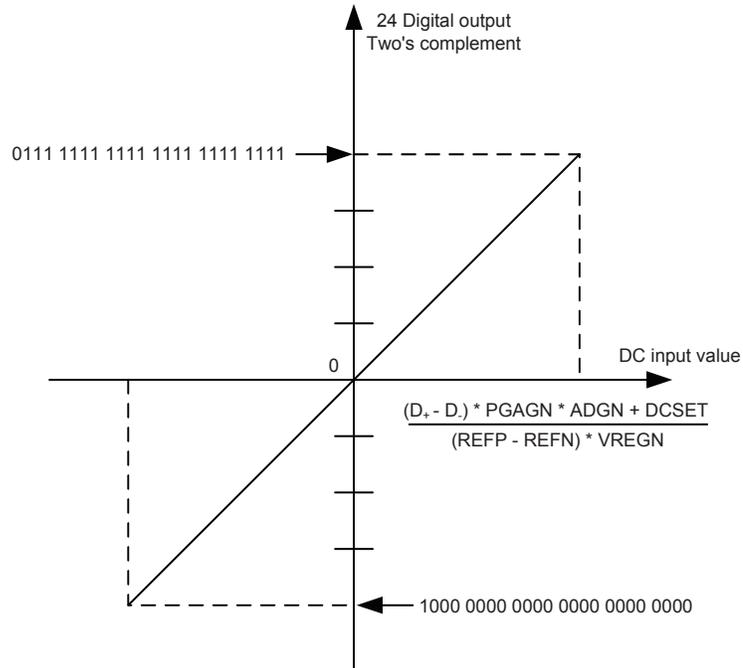
A/D Converted Data

The A/D converted data is related to the input voltage and the PGA selections. The format of the ADC output is a two's complement binary code. The length of this output code is 24 bits and the MSB is a signed bit. When the MSB is "0", which represents the input is "positive", on the other hand, as the MSB is "1", it represents the input is "negative". The maximum value is 8388607 and the minimum value is -8388608. If the input signal is over the maximum value, the converted data is limited by the 8388607, and if the input signal is less than the minimum value, the converted data is limited by -8388608.

A/D Converted data to voltage

The designer can recover the converted data by the following equations:

If MSB=0 (Positive Converted data):

$$\text{Input Voltage} = (\text{Converted data-0}) \times (\text{LSB/ PGA})$$

If the MSB=1(Negative Converted data):

$$\text{Input voltage} = (\text{Two's complement of converted data-0}) \times (\text{LSB/PGA})$$

Note: Two's complement=One's complement +1

A/D Programming Example

Example: Using an EOC polling method to detect the end of conversion

```
#include HT67F5630.inc
data .section    'data'
     adc_result_data_l db ?
     adc_result_data_m db ?
     adc_result_data_h db ?
code .section    'code'
start:
     clr     ADE           ; disable ADC interrupt
     mov     a, 0C3H       ; Power control for PGA, ADC
     mov     PWRC, a       ; PWRC=11000011, LDO enable, VCM enable, LDO output voltage:3.3V
     mov     a, 000H
     mov     PGAC0, a      ; PGA gain=1, ADC gain=1, Vref gain=1
     mov     a, 080H
     mov     PGAC1, a      ; Vcm=1.25V, INIS, INX, DCSET in default value
     set     VRBUFP        ; enable buffer for Vref+
     set     VRBUFN        ; enable buffer for Vref-
     set     VREFS         ; for using external reference
     clr     ADOR2         ; for 10Hz output data rate, ADOR[2:0]=001, FLMS[2:0]=000
     clr     ADOR1
     set     ADOR0
     clr     FLMS2
     clr     FLMS1
     clr     FLMS0
     clr     ADOFF         ; ADC exit power down mode.
     set     ADRST         ; ADC in reset mode
     clr     ADRST         ; ADC in convertsion (continuos mode)
     clr     EOC           ; Clear "EOC" flag
loop:
     snz     EOC           ; Polling "EOC" flag
     jmp     loop          ; Wait for read data
     clr     adc_result_data_h
     clr     adc_result_data_m
     clr     adc_result_data_l
     mov     a, ADRL
     mov     adc_result_data_l, a    ; Get Low byte ADC value
     mov     a, ADRM
     mov     adc_result_data_m, a    ; Get Middle byte ADC value
     mov     a, ADRH
     mov     adc_result_data_h, a    ; Get High byte ADC value
get_adc_value_ok:
     clr     EOC           ; Clearing read flag
     jmp     loop          ; for next data read
     end
```

Temperature sensor

This device provides an internal temperature sensor to compensate the device performance. By selecting the PGA input channels to V_{TSOP} and V_{TSON} signals, the A/D Converter can get the temperature information and the designer can do some compensation to the A/D converted data. The following block diagram illustrates the functional operation for the temperature sensor.
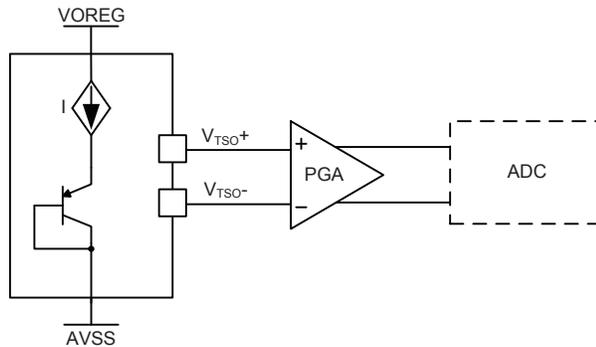
Comparator

The independent analog comparator is contained within the device. The function offers flexibility via its register controlled features such as power-down, polarity select, hysteresis etc. In sharing its pins with normal I/O pins the comparator does not waste precious I/O pins if there functions are otherwise unused.
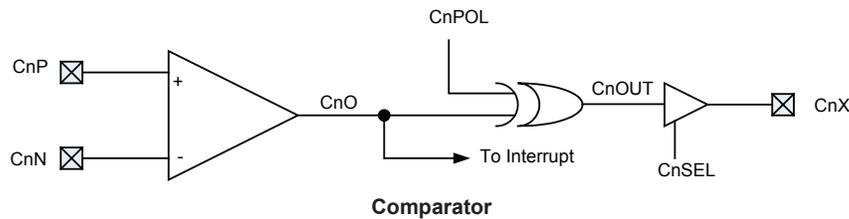
Comparator

Comparator Operation

The comparator is used to compare two analog voltages and provide an output based on their difference. Full control over the comparator is provided via one control register, CP0C. The comparator output is recorded via a bit in their respective control register, but can also be transferred out onto a shared I/O pin. Additional comparator functions include, output polarity, hysteresis functions and power down control.

Any pull-high resistors connected to the shared comparator input pins will be automatically disconnected when the comparator is enabled. As the comparator inputs approach their switching level, some spurious output signals may be generated on the comparator output due to the slow rising or falling nature of the input signals. This can be minimised by selecting the hysteresis function will apply a small amount of positive feedback to the comparator. Ideally the comparator should switch at the point where the positive and negative inputs signals are at the same voltage level, however, unavoidable input offsets introduce some uncertainties here. The hysteresis function, if enabled, also increases the switching offset value.

Comparator Register

There is one register for overall comparator operation. As corresponding bits in the register have idendical functions, they following register table applies to the register.

Register Name	Bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CP0C	C0SEL	C0EN	C0POL	C0OUT	C0OS	—	—	C0HYEN

Comparator Registers List

CP0C Register

Bit	7	6	5	4	3	2	1	0
Name	C0SEL	C0EN	C0POL	C0OUT	C0OS	—	—	C0HYEN
R/W	R/W	R/W	R/W	R	R/W	—	—	R/W
POR	1	0	0	0	0	—	—	1

Bit 7 **C0SEL**: Select Comparator pins or I/O pins
  0: I/O pin select
  1: Comparator pin select
This is the Comparator pin or I/O pin select bit. If the bit is high the comparator will be selected and the two comparator input pins will be enabled. As a result, these two pins will lose their I/O pin functions. Any pull-high configuration options associated with the comparator shared pins will also be automatically disconnected.

Bit 6 **C0EN**: Comparator On/Off control
  0: Off
  1: On
This is the Comparator on/off control bit. If the bit is zero the comparator will be switched off and no power consumed even if analog voltages are applied to its inputs. For power sensitive applications this bit should be cleared to zero if the comparator is not used or before the device enters the SLEEP or IDLE mode.

Bit 5 **C0POL**: Comparator output polarity
  0: Output not inverted
  1: Output inverted
This is the comparator polarity bit. If the bit is zero then the C0OUT bit will reflect the non-inverted output condition of the comparator. If the bit is high the comparator C0OUT bit will be inverted.

Bit 4 **C0OUT**: Comparator output bit
C0POL=0
  0: C0P < C0N
  1: C0P > C0N
C0POL=1
  0: C0P > C0N
  1: C0P < C0N
This bit stores the comparator output bit. The polarity of the bit is determined by the voltages on the comparator inputs and by the condition of the C0POL bit.

Bit 3 **C0OS**: Output path select
  0: C0OUT pin
  1: Internal use
This is the comparator output path select control bit. If the bit is cleared to zero and the C0SEL bit is 1 the comparator output is connected to an external C0OUT pin. If the bit is set high or the C0SEL bit is 0 the comparator output signal is only used internally by the device allowing the shared comparator output pin to retain its normal I/O operation.

Bit 2~1   Unimplemented, read as "0"

Bit 0   **C0HYEN**: Hysteresis Control
          0: Off
          1: On
          This is the hysteresis control bit and if set high will apply a limited amount of hysteresis to the comparator, as specified in the Comparator Electrical Characteristics table. The positive feedback induced by hysteresis reduces the effect of spurious switching near the comparator threshold.

Comparator Interrupt

The comparator also possesses its own interrupt function. When any one of the changes state, its relevant interrupt flag will be set, and if the corresponding interrupt enable bit is set, then a jump to its relevant interrupt vector will be executed. Note that it is the changing state of the C0OUT signal and generates an interrupt. If the microcontroller is in the SLEEP or IDLE Mode and the Comparator is enabled, then if the external input lines cause the Comparator output to change state, the resulting generated interrupt flag will also generate a wake-up. If it is required to disable a wake-up from occurring, then the interrupt flag should be first set high before entering the SLEEP or IDLE Mode.

Programming Considerations

If the comparator is enabled, it will remain active when the microcontroller enters the SLEEP or IDLE Mode, however as it will consume a certain amount of power, the user may wish to consider disabling it before the SLEEP or IDLE Mode is entered.

As comparator pins are shared with normal I/O pins the I/O registers for these pins will be read as zero (port control register is "1") or read as port data register value (port control register is "0") if the comparator function is enabled.

Interrupts

Interrupts are an important part of any microcontroller system. When an external event or an internal function such as a Timer Module or an A/D converter requires microcontroller attention, their corresponding interrupt will enforce a temporary suspension of the main program allowing the microcontroller to direct attention to their respective needs. The device contains several external interrupt and internal interrupts functions. The external interrupts are generated by the action of the external INT0~INT1 pins, while the internal interrupts are generated by various internal functions such as the TMs, Time Base, LVD, EEPROM and the A/D converter.

Interrupt Registers

Overall interrupt control, which basically means the setting of request flags when certain microcontroller conditions occur and the setting of interrupt enable bits by the application program, is controlled by a series of registers, located in the Special Purpose Data Memory. The first is the INTC0~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	—	—
INTn Pin	INTnE	INTnF	n=0~1
Multi-function	MFnE	MFnF	n=0~2
A/D Converter	ADE	ADF	—
Time Base	TBnE	TBnF	n=0~1
LVD	LVE	LVF	—
EEPROM	DEE	DEF	—
TM	TnPE	TnPF	n=0~1
	TnAE	TnAF	

Interrupt Register Bit Naming Conventions

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	MF0F	INT1F	INT0F	MF0E	INT1E	INT0E	EMI
INTC1	MF1F	TB1F	TB0F	ADF	MF1E	TB1E	TB0E	ADE
INTC2	—	CP0F	—	MF2F	—	CP0E	—	MF2E
MFI0	—	—	T0AF	T0PF	—	—	T0AE	T0PE
MFI1	—	—	DEF	LVF	—	—	DEE	LVE
MFI2	—	—	T1AF	T1PF	—	—	T1AE	T1PE

Interrupt Register List

INTEG Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4   Unimplemented, read as "0"

Bit 3~2   **INT1S1~INT1S0**: interrupt edge control for INT1 pin
           00: Disable
           01: Rising edge
           10: Falling edge
           11: Rising and falling edges

Bit 1~0   **INT0S1~INT0S0**: interrupt edge control for INT0 pin
           00: Disable
           01: Rising edge
           10: Falling edge
           11: Rising and falling edges

INTC0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	MF0F	INT1F	INT0F	MF0E	INT1E	INT0E	EMI
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7   Unimplemented, read as "0"

Bit 6   **MF0F**: Multi-function Interrupt 0 Request Flag
           0: No request
           1: Interrupt request

Bit 5   **INT1F**: INT1 interrupt request flag
           0: No request
           1: Interrupt request

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

Bit 3   **MF0E**: Multi-function Interrupt 0 Control
           0: Disable
           1: Enable

Bit 2   **INT1E**: INT1 interrupt control
           0: Disable
           1: Enable

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

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

INTC1 Register

Bit	7	6	5	4	3	2	1	0
Name	MF1F	TB1F	TB0F	ADF	MF1E	TB1E	TB0E	ADE
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   **MF1F**: Multi-function Interrupt 1 Request Flag
        0: No request
        1: Interrupt request

Bit 6   **TB1F**: Time Base 1 Interrupt Request Flag
        0: No request
        1: Interrupt request

Bit 5   **TB0F**: Time Base 0 Interrupt Request Flag
        0: No request
        1: Interrupt request

Bit 4   **ADF**: A/D Converter Interrupt Request Flag
        0: No request
        1: Interrupt request

Bit 3   **MF1E**: Multi-function Interrupt 1 Control
        0: Disable
        1: Enable

Bit 2   **TB1E**: Time Base 1 Interrupt Control
        0: Disable
        1: Enable

Bit 1   **TB0E**: Time Base 0 Interrupt Control
        0: Disable
        1: Enable

Bit 0   **ADE**: A/D Converter Interrupt Control
        0: Disable
        1: Enable

INTC2 Register

Bit	7	6	5	4	3	2	1	0
Name	—	CP0F	—	MF2F	—	CP0E	—	MF2E
R/W	—	R/W	—	R/W	—	R/W	—	R/W
POR	—	0	—	0	—	0	—	0

Bit 7   Unimplemented, read as "0"

Bit 6   **CP0F**: Comparator 0 Interrupt Control
        0: No request
        1: Interrupt request

Bit 5   Unimplemented, read as "0"

Bit 4   **MF2F**: Multi-function Interrupt 2 Request Flag
        0: No request
        1: Interrupt request

Bit 3   Unimplemented, read as "0"

Bit 2   **CP0E**: Comparator 0 Interrupt Control
        0: Disable
        1: Enable

Bit 1   Unimplemented, read as "0"

Bit 0   **MF2E**: Multi-function Interrupt 2 Control
        0: Disable
        1: Enable

MFI0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	T0AF	T0PF	—	—	T0AE	T0PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6   Unimplemented, read as "0"

Bit 5   **T0AF**: TM0 Comparator A match interrupt request flag
            0: No request
            1: Interrupt request

Bit 4   **T0PF**: TM0 Comparator P match interrupt request flag
            0: No request
            1: Interrupt request

Bit 3~2   Unimplemented, read as "0"

Bit 1   **T0AE**: TM0 Comparator A match interrupt control
            0: Disable
            1: Enable

Bit 0   **T0PE**: TM0 Comparator P match interrupt control
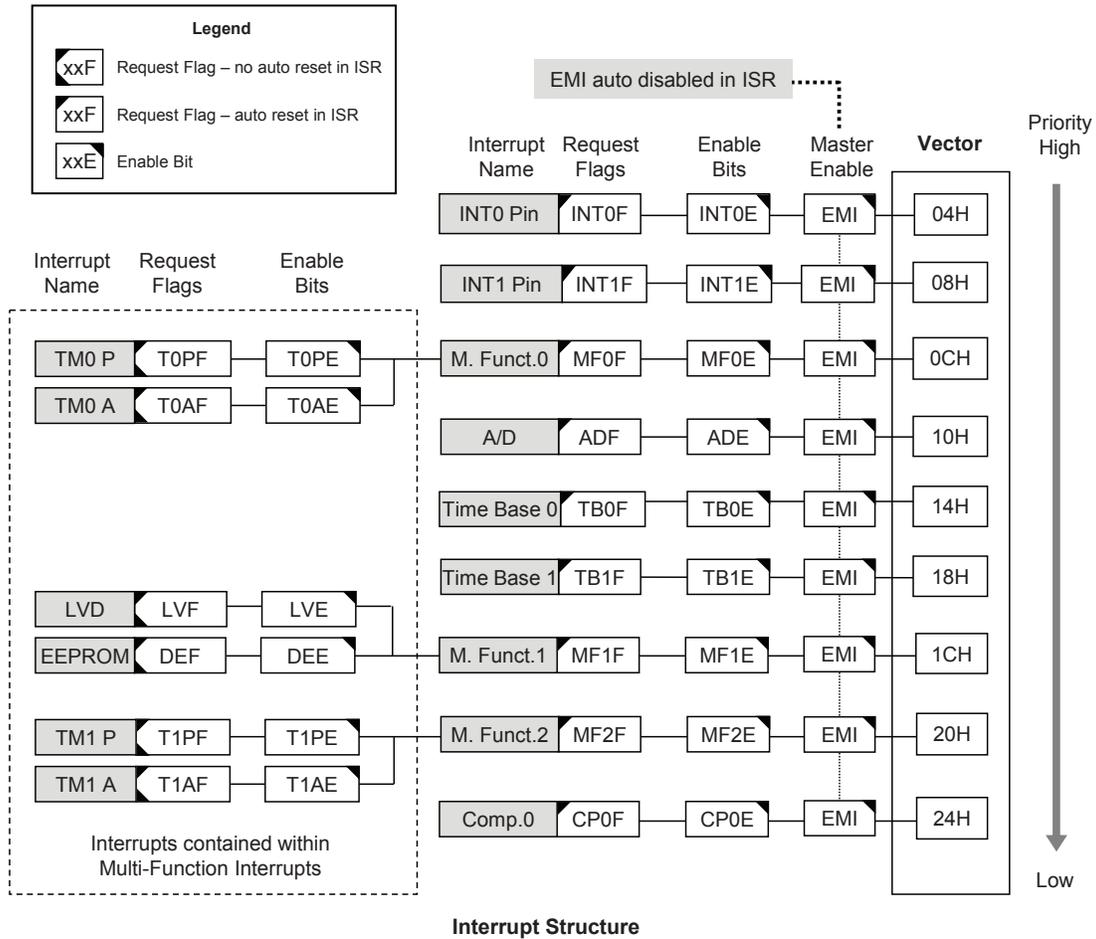   0: Disable
            1: Enable

MFI1 Register

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

Bit 7~6   Unimplemented, read as "0"

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 3~2   Unimplemented, read as "0"

Bit 1   **DEE**: Data EEPROM interrupt control
            0: Disable
            1: Enable

Bit 0   **LVE**: LVD Interrupt Control
            0: Disable
            1: Enable

MFI2 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	T1AF	T1PF	—	—	T1AE	T1PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6   Unimplemented, read as "0"

Bit 5   **T1AF**: TM1 Comparator A match interrupt request flag
        0: No request
        1: Interrupt request

Bit 4   **T1PF**: TM1 Comparator P match interrupt request flag
        0: No request
        1: Interrupt request

Bit 3~2   Unimplemented, read as "0"

Bit 1   **T1AE**: TM1 Comparator A match interrupt control
        0: Disable
        1: Enable

Bit 0   **T1PE**: TM1 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, 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.

Legend

xxF	Request Flag – no auto reset in ISR
xxF	Request Flag – auto reset in ISR
xxE	Enable Bit

EMI auto disabled in ISR

Interrupt Name	Request Flags	Enable Bits	Master Enable	Vector	Priority
INT0 Pin	INT0F	INT0E	EMI	04H	High
INT1 Pin	INT1F	INT1E	EMI	08H	
M. Funct.0	MF0F	MF0E	EMI	0CH	
A/D	ADF	ADE	EMI	10H	
Time Base 0	TB0F	TB0E	EMI	14H	
Time Base 1	TB1F	TB1E	EMI	18H	
M. Funct.1	MF1F	MF1E	EMI	1CH	
M. Funct.2	MF2F	MF2E	EMI	20H	
Comp.0	CP0F	CP0E	EMI	24H	Low

Interrupt Name	Request Flags	Enable Bits
TM0 P	T0PF	T0PE
TM0 A	T0AF	T0AE
LVD	LVF	LVE
EEPROM	DEF	DEE
TM1 P	T1PF	T1PE
TM1 A	T1AF	T1AE

Interrupts contained within Multi-Function Interrupts

Interrupt Structure

External Interrupt

The external interrupts are controlled by signal transitions on the pins INT0~INT1. An external interrupt request will take place when the external interrupt request flags, INT0F~INT1F, are set, which will occur when a transition, whose type is chosen by the edge select bits, appears on the external interrupt pins. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and respective external interrupt enable bit, INT0E~INT1E, must first be set. Additionally the correct interrupt edge type must be selected using the INTEG register to enable the external interrupt function and to choose the trigger edge type. As the external interrupt pins are pin-shared with I/O pins, they can only be configured as external interrupt pins if their external interrupt enable bit in the corresponding interrupt register has been set. The pin must also be setup as an input by setting the corresponding bit in the port control register. When the interrupt is enabled, the stack is not full and the correct transition type appears on the external interrupt pin, a subroutine call to the external interrupt vector, will take place. When the interrupt is serviced, the external interrupt request flags, INT0F~INT1F, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts. Note that any pull-high resistor selections on the external interrupt pins will remain valid even if the pin is used as an external interrupt input. The INTEG register is used to select the type of active edge that will trigger the external interrupt. A choice of either rising or falling or both edge types can be chosen to trigger an external interrupt. Note that the INTEG register can also be used to disable the external interrupt function.

Comparator Interrupt

The comparator interrupt is controlled by an internal comparator. A comparator interrupt request will take place when the comparator interrupt request flag, CP0F, is set, a situation that will occur when the comparator output changes state. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and comparator interrupt enable bit, CP0E, must first be set. When the interrupt is enabled, the stack is not full and the comparator inputs generate a comparator output transition, a subroutine call to the comparator interrupt vector, will take place. When the interrupt is serviced, the comparator interrupt request flags will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts.

A/D Converter Interrupt

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

Multi-function Interrupt

Within this device 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 TM Interrupts, LVD interrupt and EEPROM Interrupt.

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, namely the TM Interrupts, LVD interrupt and EEPROM Interrupt will not be automatically reset and must be manually reset by the application program.

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. Their clock sources originate from the internal clock source f_{TB}. This f_{TB} input clock passes through a divider, the division ratio of which is selected by programming the appropriate bits in the TBC register to obtain longer interrupt periods whose value ranges. The clock source that generates f_{TB}, which in turn controls the Time Base interrupt period, can originate from several different sources, as shown in the System Operating Mode section.

Time Base Interrupt
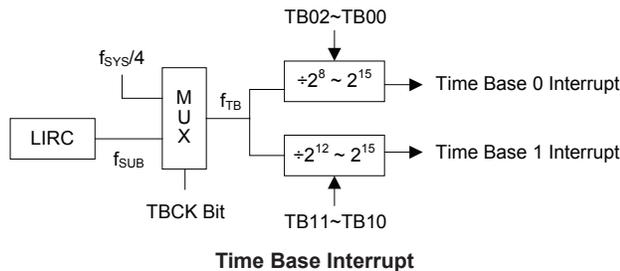
TBC Register

Bit	7	6	5	4	3	2	1	0
Name	TBON	TBCK	TB11	TB10	—	TB02	TB01	TB00
R/W	R/W	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	0	0	1	1	—	1	1	1

Bit 7   **TBON**: TB0 and TB1 Control
         0: Disable
         1: Enable

Bit 6   **TBCK**: Select f_{TB} Clock
         0: f_{SUB}
         1: $f_{SYS}/4$

Bit 5~4   **TB11~TB10**: Select Time Base 1 Time-out Period
         00: $4096/f_{TB}$
         01: $8192/f_{TB}$
         10: $16384/f_{TB}$
         11: $32768/f_{TB}$

Bit 3   Unimplemented, read as "0"

Bit 2~0   **TB02~TB00**: Select Time Base 0 Time-out Period
         000: $256/f_{TB}$
         001: $512/f_{TB}$
         010: $1024/f_{TB}$
         011: $2048/f_{TB}$
         100: $4096/f_{TB}$
         101: $8192/f_{TB}$
         110: $16384/f_{TB}$
         111: $32768/f_{TB}$

EEPROM Interrupt

The EEPROM interrupt is contained within the Multi-function 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, and associated Multi-function interrupt enable bit, 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 respective EEPROM Interrupt vector will take place. When the EEPROM 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 DEF flag will not be automatically cleared, it has to be cleared by the application program.

LVD Interrupt

The Low Voltage Detector Interrupt is contained within the Multi-function 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, Low Voltage Interrupt enable bit, LVE, and associated Multi-function interrupt enable bit, 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 Multi-function Interrupt vector, will take place. When the Low Voltage 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 LVF flag will not be automatically cleared, it has to be cleared by the application program.

TM Interrupts

The Compact and Periodic Type TMs have two interrupts each. All of the TM interrupts are contained within the Multi-function Interrupts. For each of the Compact and Periodic Type TMs there are two interrupt request flags TnPF and TnAF and two enable bits TnPE and TnAE. 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 transitions on the external interrupt pins, a low power supply voltage or comparator input change 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 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.

Low Voltage Detector – LVD

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

LVD Register

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

LVDC Register

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

Bit 7~6   Unimplemented, read as "0"

Bit 5   **LVDO**: LVD Output Flag
            0: No Low Voltage Detect
            1: Low Voltage Detect

Bit 4   **LVDEN**: Low Voltage Detector Control
            0: Disable
            1: Enable

Bit 3   Unimplemented, read as "0"

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

LVD Operation

The Low Voltage Detector function operates by comparing the power supply voltage, V_{DD}, with a pre-specified voltage level stored in the LVDC register. This has a range of between 2.0V and 4.0V. When the power supply voltage, V_{DD}, falls below this pre-determined value, the LVDO bit will be set high indicating a low power supply voltage condition. The Low Voltage Detector function is supplied by a reference voltagewhich will be automatically enabled.When the device is powered down the low voltage detector will remain active 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.
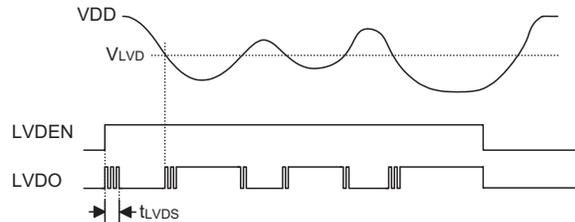
LVD Operation

The Low Voltage Detector also has its own interrupt which is contained within one of the Multi-function interrupts, 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_{LVDS} after the LVDO bit has been set high by a low voltage condition. When the device is powered down the Low Voltage Detector will remain active if the LVDEN bit is high. 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 SLEEP or 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 SLEEP or IDLE Mode.

When LVD function is enabled, it is recommenced to clear LVD flag first, and then enables interrupt function to avoid mistake action.

LCD Driver

For large volume applications, which incorporate an LCD in their design, the use of a custom display rather than a more expensive character based display reduces costs significantly. However, the corresponding COM and SEG signals required, which vary in both amplitude and time, to drive such a custom display require many special considerations for proper LCD operation to occur. The device contains an LCD Driver function, which with their internal LCD signal generating circuitry and various options, will automatically generate these time and amplitude varying signals to provide a means of direct driving and easy interfacing to a range of custom LCDs.

This device includes a wide range of options to enable LCD displays of various types to be driven. The table shows the range of options available across the device range.

| Driver No.	Bias	Duty	Bias Type
15×4	1/3	1/4	R

LCD Selections
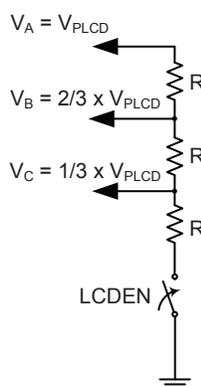
R Type 1/3 Bias Voltage Levels

LCD Display Memory

The device provides an area of embedded data memory for the LCD display. This area is located at 80H to 8FH in Bank 1 of the Data Memory. The bank pointer BP enables either the General Purpose Data Memory or LCD Memory to be chosen. When BP is set to "01H", any data written into location range 80H~8EH will affect the LCD display. When the BP is cleared to "00H", any data written into 80H~8EH will access the General Purpose Data Memory.

The LCD display memory can be read and written to only indirectly using MP1. When data is written into the display data area, it is automatically read by the LCD driver which then generates the corresponding LCD driving signals. To turn the display on or off, a "1" or a "0" is written to the corresponding bit of the display memory, respectively. The figure illustrates the mapping between the display memory and LCD pattern for the device.

LCD Memory Map
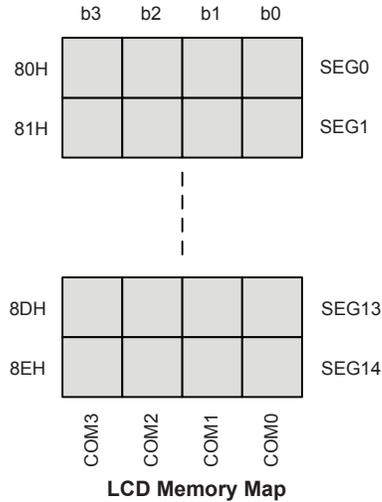
Clock Source

The LCD clock source is the internal clock signal, f_{SUB}, divided by 8, using an internal divider circuit. The f_{SUB} internal clock is supplied by the LIRC oscillator. For proper LCD operation, this arrangement is provided to generate an ideal LCD clock source frequency of 4kHz.

f_{SUB} Clock Source	LCD Clock Frequency
LIRC	4kHz

LCD Clock Source

LCD Registers

Control Registers in the Data Memory, are used to control the various setup features of the LCD Driver. There are four control registers for the LCD function, LCDC, LCD1, LCD2 and LCD3.

Various bits in the LCDC register control functions such as bias current selection, overall LCD enable and disable. The LCDEN bit in the LCDC register, which provides the overall LCD enable/ disable function, will only be effective when the device is in the Normal, Slow or Idle Mode. If the device is in the Sleep Mode then the display will always be disabled.

Bits LCDIS0 and LCDIS1 in the LCDC register are used to select the internal bias current to supply the LCD panel with the correct bias voltages. A choice to best match the LCD panel used in the application can be selected also to minimise bias current.

Three registers, LCD1~LCD3, are used to determine if the output function of display pins SEG0~SEG14 are used as segment drivers or I/O functions, COM0~COM3 are used as common drivers or I/O functions.

LCDC Register

Bit	7	6	5	4	3	2	1	0
Name	LCDEN	—	—	—	—	—	LCDIS1	LCDIS0
R/W	R/W	—	—	—	—	—	R/W	R/W
POR	0	—	—	—	—	—	0	0

Bit 7   **LCDEN**: LCD Enable Control
    0: Disable
    1: Enable
    In the Normal, Slow or Idle mode, the LCD on/off function can be controlled by this bit. In the Sleep mode, the LCD is always off.

Bit 6~2   Unimplemented, read as "0"

Bit 1~0   **LCDIS1~LCDIS0**: LCD Bias Current Select
    00: 25μA
    01: 50μA
    10: 100μA
    11: 200μA

LCD1 Register

Bit	7	6	5	4	3	2	1	0
Name	LCDS7	LCDS6	LCDS5	LCDS4	LCDS3	LCDS2	LCDS1	LCDS0
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   **LCDS7**: SEG7 Output Control
    0: Disable
    1: Enable

Bit 6   **LCDS6**: SEG6 Output Control
    0: Disable
    1: Enable

Bit 5   **LCDS5**: SEG5 Output Control
    0: Disable
    1: Enable

Bit 4   **LCDS4**: SEG4 Output Control
    0: Disable
    1: Enable

Bit 3   **LCDS3**: SEG3 Output Control
    0: Disable
    1: Enable

Bit 2   **LCDS2**: SEG2 Output Control
    0: Disable
    1: Enable

Bit 1   **LCDS1**: SEG1 Output Control
    0: Disable
    1: Enable

Bit 0   **LCDS0**: SEG0 Output Control
    0: Disable
    1: Enable

LCD2 Register

Bit	7	6	5	4	3	2	1	0
Name	D7	LCDS14	LCDS13	LCDS12	LCDS11	LCDS10	LCDS9	LCDS8
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   Undefined, used as general data bit

Bit 6   **LCDS14**: SEG14 Output Control
    0: Disable
    1: Enable

Bit 5   **LCDS13**: SEG13 Output Control
    0: Disable
    1: Enable

Bit 4   **LCDS12**: SEG12 Output Control
    0: Disable
    1: Enable

Bit 3   **LCDS11**: SEG11 Output Control
    0: Disable
    1: Enable

Bit 2   **LCDS10**: SEG10 Output Control
    0: Disable
    1: Enable

Bit 1   **LCDS9**: SEG9 Output Control
    0: Disable
    1: Enable

Bit 0   **LCDS8**: SEG8 Output Control
    0: Disable
    1: Enable

LCD3 Register

| Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	COM3	COM2	COM1	COM0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4   Unimplemented, read as "0"

Bit 3   **COM3**: COM3 Output Control
    0: Disable
    1: Enable

Bit 2   **COM2**: COM2 Output Control
    0: Disable
    1: Enable

Bit 1   **COM1**: COM1 Output Control
    0: Disable
    1: Enable

Bit 0   **COM0**: COM0 Output Control
    0: Disable
    1: Enable

LCD Driver Output

The output structure of the device LCD driver can be 15×4. The LCD driver bias type has R type only. The number of COM and SEG is selected by software option. The LCD driver has a fixed 1/3 value.

LCD Waveform Timing Diagrams

The device generates 1/4 duty and 1 /3 bias LCD signals, as shown below.
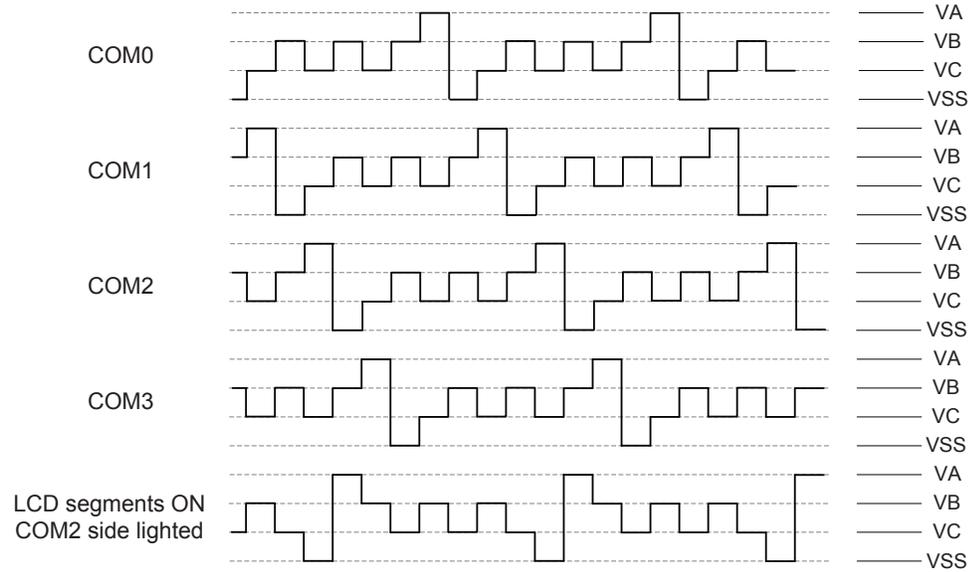
Note: For R type, $V_A=V_{PLCD}$, $V_B=V_{PLCD}\times2/3$ and $V_C=V_{PLCD}\times1/3$

LCD Driver Output – 1/4 Duty, 1/3 Bias

LCD Charge Pump

The COMs and SEGs pins can be powered up by the external PLCD pin or internal charge pump circuit which is determined by the LCDPR bit in the LCDCP register. When the LCDPR bit is set low, the LCD driver power is supplied by the external PLCD pin. The PLCD pin voltage should be equal to or less than V_{DD} if the PLCD pin is selected to be used. If the LCDPR bit is set high, the LCD driver power is supplied by the internal charge pump circuit. There are four charge pump output voltage levels which are selected by the CPVS1~CPVS0 bits in the LCDCP register. If the internal charge pump circuit is used, an external 4.7μF capacitor should be connected to the external PLCD pin for output voltage stability. Note that there is also a relationship between the selected charge pump output voltage and V_{DD} when the internal charge pump circuit is used.

LCDPR	LCD Driver Power Supply	Relationship
0	PLCD pin	$V_{DD} \geq V_{PLCD}$
1	Charge Pump Circuit	$V_{DD} \geq V_{PLCD} - 0.7V$

LCD Driver Power Supply Relationship
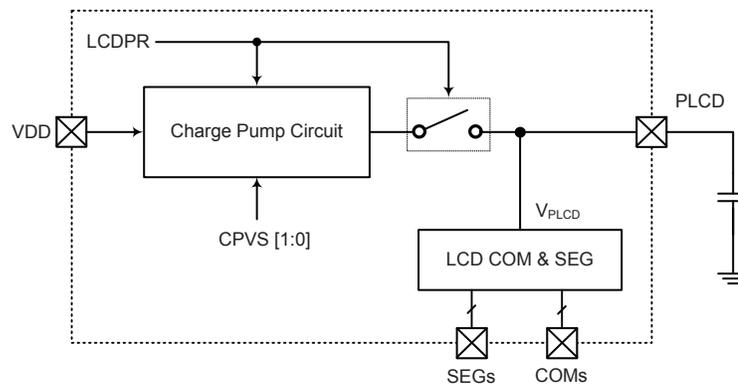
LCD Driver Charge Pump Circuit

LCDCP Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	LCDPR	—	CPVS1	CPVS0
R/W	—	—	—	—	R/W	—	R/W	R/W
POR	—	—	—	—	0	—	0	0

Bit 7~4　　Unimplemented, read as "0"

Bit 3　　**LCDPR**: LCD power supply selection
　　　　　0: From external PLCD pin
　　　　　1: From internal charge pump circuit

Bit 2　　Unimplemented, read as "0"

Bit 1~0　　**CPVS1~CPVS0**: LCD charge pump output voltage selection
　　　　　00: 3.3V
　　　　　01: 3.0V
　　　　　10: 2.7V
　　　　　11: 4.5V

Programming Considerations

Certain precautions must be taken when programming the LCD. One of these is to ensure that the LCD Memory is properly initialised after the microcontroller is powered on. Like the General Purpose Data Memory, the contents of the LCD Memory are in an unknown condition after power-on. As the contents of the LCD Memory will be mapped into the actual display, it is important to initialise this memory area into a known condition soon after applying power to obtain a proper display pattern.

Consideration must also be given to the capacitive load of the actual LCD used in the application. As the load presented to the microcontroller by LCD pixels can be generally modeled as mainly capacitive in nature, it is important that this is not excessive, a point that is particularly true in the case of the COM lines which may be connected to many LCD pixels. The accompanying diagram depicts the equivalent circuit of the LCD.

One additional consideration that must be taken into account is what happens when the microcontroller enters the Idle or Slow Mode. The LCDEN control bit in the LCDC register permits the display to be powered off to reduce power consumption. If this bit is zero, the driving signals to the display will cease, producing a blank display pattern but reducing any power consumption associated with the LCD.

After Power-on, note that as the LCDEN bit will be cleared to zero, the display function will be disabled.


LCD Panel Equivalent Circuit
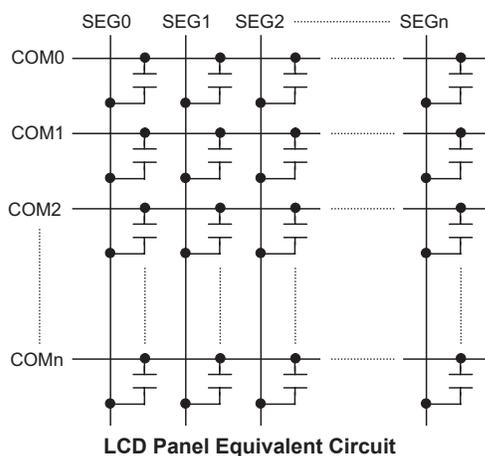
Configuration Option

Configuration options refer to certain options within the MCU that are programmed into the device during the programming process. During the development process, these options are selected using the HT-IDE software development tools. As these options are programmed into the device using the hardware programming tools, once they are selected they cannot be changed later using the application program. All options must be defined for proper system function, the details of which are shown in the table.

No.	Options
Oscillator Options	
1	High Speed System Oscillator Selection – f_H: 1. HXT 2. HIRC
2	HIRC Frequency Selection: 1. 4.9152MHz 2. 9.8304MHz 3. 14.7456MHz
Watchdog Timer Options	
4	WDT function: 1. Always enable 2. Controlled by WDT Control Register
5	WDT Clock Selection – f_S: 1. f_{SUB} 2. $f_{SYS}/4$

Application Circuit
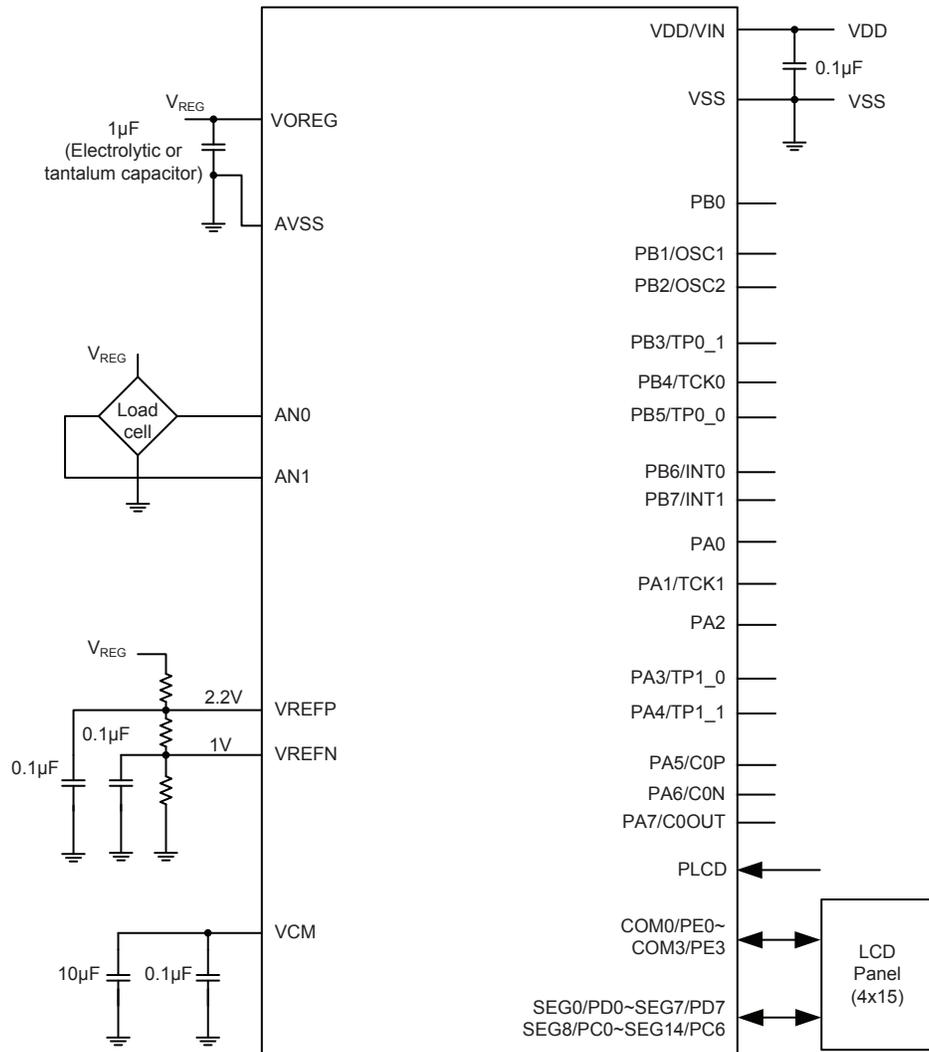
Instruction Set

Introduction

Central to the successful operation of any microcontroller is its instruction set, which is a set of program instruction codes that directs the microcontroller to perform certain operations. In the case of Holtek microcontroller, a comprehensive and flexible set of over 60 instructions is provided to enable programmers to implement their application with the minimum of programming overheads.

For easier understanding of the various instruction codes, they have been subdivided into several functional groupings.

Instruction Timing

Most instructions are implemented within one instruction cycle. The exceptions to this are branch, call, or table read instructions where two instruction cycles are required. One instruction cycle is equal to 4 system clock cycles, therefore in the case of an 8MHz system oscillator, most instructions would be implemented within 0.5μs and branch or call instructions would be implemented within 1μs. Although instructions which require one more cycle to implement are generally limited to the JMP, CALL, RET, RETI and table read instructions, it is important to realize that any other instructions which involve manipulation of the Program Counter Low register or PCL will also take one more cycle to implement. As instructions which change the contents of the PCL will imply a direct jump to that new address, one more cycle will be required. Examples of such instructions would be "CLR PCL" or "MOV PCL, A". For the case of skip instructions, it must be noted that if the result of the comparison involves a skip operation then this will also take one more cycle, if no skip is involved then only one cycle is required.

Moving and Transferring Data

The transfer of data within the microcontroller program is one of the most frequently used operations. Making use of three kinds of MOV instructions, data can be transferred from registers to the Accumulator and vice-versa as well as being able to move specific immediate data directly into the Accumulator. One of the most important data transfer applications is to receive data from the input ports and transfer data to the output ports.

Arithmetic Operations

The ability to perform certain arithmetic operations and data manipulation is a necessary feature of most microcontroller applications. Within the Holtek microcontroller instruction set are a range of add and subtract instruction mnemonics to enable the necessary arithmetic to be carried out. Care must be taken to ensure correct handling of carry and borrow data when results exceed 255 for addition and less than 0 for subtraction. The increment and decrement instructions INC, INCA, DEC and DECA provide a simple means of increasing or decreasing by a value of one of the values in the destination specified.

Logical and Rotate Operation

The standard logical operations such as AND, OR, XOR and CPL all have their own instruction within the Holtek microcontroller instruction set. As with the case of most instructions involving data manipulation, data must pass through the Accumulator which may involve additional programming steps. In all logical data operations, the zero flag may be set if the result of the operation is zero. Another form of logical data manipulation comes from the rotate instructions such as RR, RL, RRC and RLC which provide a simple means of rotating one bit right or left. Different rotate instructions exist depending on program requirements. Rotate instructions are useful for serial port programming applications where data can be rotated from an internal register into the Carry bit from where it can be examined and the necessary serial bit set high or low. Another application which rotate data operations are used is to implement multiplication and division calculations.

Branches and Control Transfer

Program branching takes the form of either jumps to specified locations using the JMP instruction or to a subroutine using the CALL instruction. They differ in the sense that in the case of a subroutine call, the program must return to the instruction immediately when the subroutine has been carried out. This is done by placing a return instruction "RET" in the subroutine which will cause the program to jump back to the address right after the CALL instruction. In the case of a JMP instruction, the program simply jumps to the desired location. There is no requirement to jump back to the original jumping off point as in the case of the CALL instruction. One special and extremely useful set of branch instructions are the conditional branches. Here a decision is first made regarding the condition of a certain data memory or individual bits. Depending upon the conditions, the program will continue with the next instruction or skip over it and jump to the following instruction. These instructions are the key to decision making and branching within the program perhaps determined by the condition of certain input switches or by the condition of internal data bits.

Bit Operations

The ability to provide single bit operations on Data Memory is an extremely flexible feature of all Holtek microcontrollers. This feature is especially useful for output port bit programming where individual bits or port pins can be directly set high or low using either the "SET [m].i" or "CLR [m].i" instructions respectively. The feature removes the need for programmers to first read the 8-bit output port, manipulate the input data to ensure that other bits are not changed and then output the port with the correct new data. This read-modify-write process is taken care of automatically when these bit operation instructions are used.

Table Read Operations

Data storage is normally implemented by using registers. However, when working with large amounts of fixed data, the volume involved often makes it inconvenient to store the fixed data in the Data Memory. To overcome this problem, Holtek microcontrollers allow an area of Program Memory to be set as a table where data can be directly stored. A set of easy to use instructions provides the means by which this fixed data can be referenced and retrieved from the Program Memory.

Other Operations

In addition to the above functional instructions, a range of other instructions also exist such as the "HALT" instruction for Power-down operations and instructions to control the operation of the Watchdog Timer for reliable program operations under extreme electric or electromagnetic environments. For their relevant operations, refer to the functional related sections.

Instruction Set Summary

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

Table Conventions

x: Bits immediate data

m: Data Memory address

A: Accumulator

i: 0~7 number of bits

addr: Program memory address

Mnemonic	Description	Cycles	Flag Affected
Arithmetic			
ADD A,[m]	Add Data Memory to ACC	1	Z, C, AC, OV
ADDM A,[m]	Add ACC to Data Memory	1[Note]	Z, C, AC, OV
ADD A,x	Add immediate data to ACC	1	Z, C, AC, OV
ADC A,[m]	Add Data Memory to ACC with Carry	1	Z, C, AC, OV
ADCM A,[m]	Add ACC to Data memory with Carry	1[Note]	Z, C, AC, OV
SUB A,x	Subtract immediate data from the ACC	1	Z, C, AC, OV
SUB A,[m]	Subtract Data Memory from ACC	1	Z, C, AC, OV
SUBM A,[m]	Subtract Data Memory from ACC with result in Data Memory	1[Note]	Z, C, AC, OV
SBC A,[m]	Subtract Data Memory from ACC with Carry	1	Z, C, AC, OV
SBCM A,[m]	Subtract Data Memory from ACC with Carry, result in Data Memory	1[Note]	Z, C, AC, OV
DAA [m]	Decimal adjust ACC for Addition with result in Data Memory	1[Note]	C
Logic Operation			
AND A,[m]	Logical AND Data Memory to ACC	1	Z
OR A,[m]	Logical OR Data Memory to ACC	1	Z
XOR A,[m]	Logical XOR Data Memory to ACC	1	Z
ANDM A,[m]	Logical AND ACC to Data Memory	1[Note]	Z
ORM A,[m]	Logical OR ACC to Data Memory	1[Note]	Z
XORM A,[m]	Logical XOR ACC to Data Memory	1[Note]	Z
AND A,x	Logical AND immediate Data to ACC	1	Z
OR A,x	Logical OR immediate Data to ACC	1	Z
XOR A,x	Logical XOR immediate Data to ACC	1	Z
CPL [m]	Complement Data Memory	1[Note]	Z
CPLA [m]	Complement Data Memory with result in ACC	1	Z
Increment & Decrement			
INCA [m]	Increment Data Memory with result in ACC	1	Z
INC [m]	Increment Data Memory	1[Note]	Z
DECA [m]	Decrement Data Memory with result in ACC	1	Z
DEC [m]	Decrement Data Memory	1[Note]	Z
Rotate			
RRA [m]	Rotate Data Memory right with result in ACC	1	None
RR [m]	Rotate Data Memory right	1[Note]	None
RRCA [m]	Rotate Data Memory right through Carry with result in ACC	1	C
RRC [m]	Rotate Data Memory right through Carry	1[Note]	C
RLA [m]	Rotate Data Memory left with result in ACC	1	None
RL [m]	Rotate Data Memory left	1[Note]	None
RLCA [m]	Rotate Data Memory left through Carry with result in ACC	1	C
RLC [m]	Rotate Data Memory left through Carry	1[Note]	C

Mnemonic	Description	Cycles	Flag Affected
Data Move			
MOV A,[m]	Move Data Memory to ACC	1	None
MOV [m],A	Move ACC to Data Memory	1[Note]	None
MOV A,x	Move immediate data to ACC	1	None
Bit Operation			
CLR [m].i	Clear bit of Data Memory	1[Note]	None
SET [m].i	Set bit of Data Memory	1[Note]	None
Branch Operation			
JMP addr	Jump unconditionally	2	None
SZ [m]	Skip if Data Memory is zero	1[Note]	None
SZA [m]	Skip if Data Memory is zero with data movement to ACC	1[Note]	None
SZ [m].i	Skip if bit i of Data Memory is zero	1[Note]	None
SNZ [m].i	Skip if bit i of Data Memory is not zero	1[Note]	None
SIZ [m]	Skip if increment Data Memory is zero	1[Note]	None
SDZ [m]	Skip if decrement Data Memory is zero	1[Note]	None
SIZA [m]	Skip if increment Data Memory is zero with result in ACC	1[Note]	None
SDZA [m]	Skip if decrement Data Memory is zero with result in ACC	1[Note]	None
CALL addr	Subroutine call	2	None
RET	Return from subroutine	2	None
RET A,x	Return from subroutine and load immediate data to ACC	2	None
RETI	Return from interrupt	2	None
Table Read Operation			
TABRD [m]	Read table (specific page) 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 ← Program Counter + 1 Program Counter ← addr
Affected flag(s)	None

CLR [m]	Clear Data Memory
Description	Each bit of the specified Data Memory is cleared to 0.
Operation	[m] ← 00H
Affected flag(s)	None

CLR [m].i	Clear bit of Data Memory
Description	Bit i of the specified Data Memory is cleared to 0.
Operation	[m].i ← 0
Affected flag(s)	None

CLR WDT	Clear Watchdog Timer
Description	The TO, PDF flags and the WDT are all cleared.
Operation	WDT cleared TO ← 0 PDF ← 0
Affected flag(s)	TO, PDF

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 ← 0 PDF ← 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 ← 0 PDF ← 0
Affected flag(s)	TO, PDF

CPL [m]	Complement Data Memory
Description	Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa.
Operation	[m] ← $\overline{[m]}$
Affected flag(s)	Z

CPLA [m]	Complement Data Memory with result in ACC
Description	Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa. The complemented result is stored in the Accumulator and the contents of the Data Memory remain unchanged.
Operation	ACC ← $\overline{[m]}$
Affected flag(s)	Z

DAA [m]	Decimal-Adjust ACC for addition with result in Data Memory
Description	Convert the contents of the Accumulator value to a BCD (Binary Coded Decimal) value resulting from the previous addition of two BCD variables. If the low nibble is greater than 9 or if AC flag is set, then a value of 6 will be added to the low nibble. Otherwise the low nibble remains unchanged. If the high nibble is greater than 9 or if the C flag is set, then a value of 6 will be added to the high nibble. Essentially, the decimal conversion is performed by adding 00H, 06H, 60H or 66H depending on the Accumulator and flag conditions. Only the C flag may be affected by this instruction which indicates that if the original BCD sum is greater than 100, it allows multiple precision decimal addition.
Operation	[m] ← ACC + 00H or [m] ← ACC + 06H or [m] ← ACC + 60H or [m] ← ACC + 66H
Affected flag(s)	C

DEC [m]	Decrement Data Memory
Description	Data in the specified Data Memory is decremented by 1.
Operation	[m] ← [m] − 1
Affected flag(s)	Z

DECA [m]	Decrement Data Memory with result in ACC
Description	Data in the specified Data Memory is decremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.
Operation	ACC ← [m] − 1
Affected flag(s)	Z

HALT	Enter power down mode
Description	This instruction stops the program execution and turns off the system clock. The contents of the Data Memory and registers are retained. The WDT and prescaler are cleared. The power down flag PDF is set and the WDT time-out flag TO is cleared.
Operation	TO ← 0 PDF ← 1
Affected flag(s)	TO, PDF

INC [m]	Increment Data Memory
Description	Data in the specified Data Memory is incremented by 1.
Operation	[m] ← [m] + 1
Affected flag(s)	Z

INCA [m]	Increment Data Memory with result in ACC
Description	Data in the specified Data Memory is incremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.
Operation	ACC ← [m] + 1
Affected flag(s)	Z

JMP addr	Jump unconditionally
Description	The contents of the Program Counter are replaced with the specified address. Program execution then continues from this new address. As this requires the insertion of a dummy instruction while the new address is loaded, it is a two cycle instruction.
Operation	Program Counter ← addr
Affected flag(s)	None

MOV A,[m]	Move Data Memory to ACC
Description	The contents of the specified Data Memory are copied to the Accumulator.
Operation	ACC ← [m]
Affected flag(s)	None

MOV A,x	Move immediate data to ACC
Description	The immediate data specified is loaded into the Accumulator.
Operation	ACC ← x
Affected flag(s)	None

MOV [m],A	Move ACC to Data Memory
Description	The contents of the Accumulator are copied to the specified Data Memory.
Operation	[m] ← ACC
Affected flag(s)	None

NOP	No operation
Description	No operation is performed. Execution continues with the next instruction.
Operation	No operation
Affected flag(s)	None

OR A,[m]	Logical OR Data Memory to ACC
Description	Data in the Accumulator and the specified Data Memory perform a bitwise logical OR operation. The result is stored in the Accumulator.
Operation	ACC ← ACC ″OR″ [m]
Affected flag(s)	Z

OR A,x	Logical OR immediate data to ACC
Description	Data in the Accumulator and the specified immediate data perform a bitwise logical OR operation. The result is stored in the Accumulator.
Operation	ACC ← ACC ″OR″ x
Affected flag(s)	Z

ORM A,[m]	Logical OR ACC to Data Memory
Description	Data in the specified Data Memory and the Accumulator perform a bitwise logical OR operation. The result is stored in the Data Memory.
Operation	[m] ← ACC ″OR″ [m]
Affected flag(s)	Z

RET	Return from subroutine
Description	The Program Counter is restored from the stack. Program execution continues at the restored address.
Operation	Program Counter ← Stack
Affected flag(s)	None

RET A,x	Return from subroutine and load immediate data to ACC
Description	The Program Counter is restored from the stack and the Accumulator loaded with the specified immediate data. Program execution continues at the restored address.
Operation	Program Counter ← Stack ACC ← x
Affected flag(s)	None

RETI	Return from interrupt
Description	The Program Counter is restored from the stack and the interrupts are re-enabled by setting the EMI bit. EMI is the master interrupt global enable bit. If an interrupt was pending when the RETI instruction is executed, the pending Interrupt routine will be processed before returning to the main program.
Operation	Program Counter ← Stack EMI ← 1
Affected flag(s)	None

RL [m]	Rotate Data Memory left
Description	The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0.
Operation	$[m].(i+1) \leftarrow [m].i$; (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 and the carry flag are 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	$ACC \leftarrow ACC - [m] - C$
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	$[m] \leftarrow ACC - [m] - C$
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 ← [m] − 1 Skip if ACC=0
Affected flag(s)	None

SET [m]	Set Data Memory
Description	Each bit of the specified Data Memory is set to 1.
Operation	[m] ← FFH
Affected flag(s)	None

SET [m].i	Set bit of Data Memory
Description	Bit i of the specified Data Memory is set to 1.
Operation	[m].i ← 1
Affected flag(s)	None

SIZ [m]	Skip if increment Data Memory is 0
Description	The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.
Operation	[m] ← [m] + 1 Skip if [m]=0
Affected flag(s)	None

SIZA [m]	Skip if increment Data Memory is zero with result in ACC
Description	The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.
Operation	ACC ← [m] + 1 Skip if ACC=0
Affected flag(s)	None

SNZ [m].i	Skip if 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 ≠ 0
Affected flag(s)	None

SUB A,[m]	Subtract Data Memory from ACC
Description	The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.
Operation	ACC ← ACC − [m]
Affected flag(s)	OV, Z, AC, C

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

48-pin LQFP (7mm×7mm) Outline Dimensions
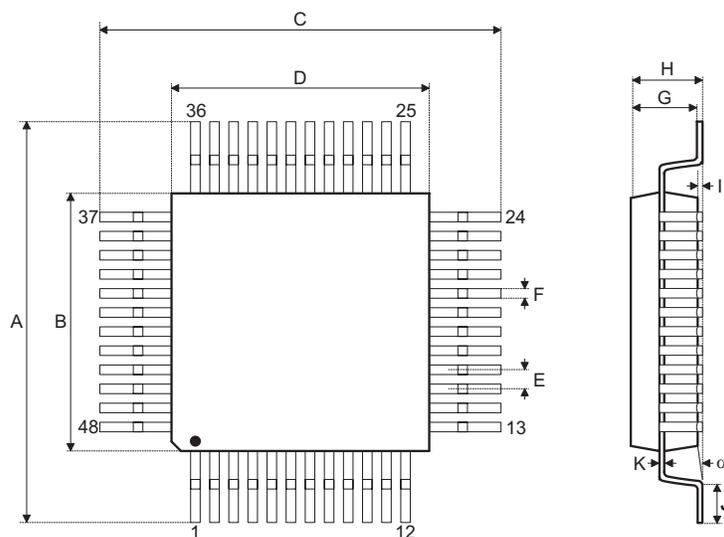
Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	—	0.354 BSC	—
B	—	0.276 BSC	—
C	—	0.354 BSC	—
D	—	0.276 BSC	—
E	—	0.020 BSC	—
F	0.007	0.009	0.011
G	0.053	0.055	0.057
H	—	—	0.063
I	0.002	—	0.006
J	0.018	0.024	0.030
K	0.004	—	0.008
α	0°	—	7°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	—	9.00 BSC	—
B	—	7.00 BSC	—
C	—	9.00 BSC	—
D	—	7.00 BSC	—
E	—	0.50 BSC	—
F	0.17	0.22	0.27
G	1.35	1.40	1.45
H	—	—	1.60
I	0.05	—	0.15
J	0.45	0.60	0.75
K	0.09	—	0.20
α	0°	—	7°