HOLTEK

Power Bank Flash MCU

BP45FH6N

Revision: V1.10 Date: November 18, 2019

www.holtek.com

Table of Contents

Features

CPU Features

- Operating voltage
  - f_{SYS}=8MHz: 2.2V~5.5V
  - f_{SYS}=12MHz: 2.7V~5.5V
  - f_{SYS}=16MHz: 3.3V~5.5V
- Up to 0.25μs instruction cycle with 16MHz system clock at V_{DD}=5V
- Power down and wake-up functions to reduce power consumption
- Oscillator types
  - Internal High Speed 8/12/16MHz RC – HIRC
  - Internal Low Speed 32kHz RC – LIRC
- Multi-mode operation: FAST, SLOW, IDLE and SLEEP
- Fully integrated internal oscillators require no external components
- All instructions executed in 1~3 instruction cycles
- Table read instructions
- 115 powerful instructions
- 8-level subroutine nesting
- Bit manipulation instruction

Peripheral Features

- Flash Program Memory: 6K×16
- Data Memory: 256×8
- True EEPROM Memory: 64×8
- Watchdog Timer function
- 28 bidirectional I/O lines
- 8 High Voltage output lines with short-circuit protection function
- Three pin-shared external interrupts
- Multiple Timer Modules for time measurement, input capture, compare match output or PWM output or single pulse output function
  - Single standard type 16-bit Timer Module – STM
  - Single periodic type 10-bit Timer Module – PTM
- Auto Adjust High Resolution PWM with Delay Lock Loop/Dead-Time
- Two over current protection (OCP) with interrupts
- Two sets of Over/Under voltage protection (OUVP) with interrupts
- USB auto detection function
- Dual Time-Base functions for generation of fixed time interrupt signals
- Multi-channel 12-bit resolution A/D converter with a Programmable Reference Voltage V_R
- Low voltage reset function
- Low voltage detect function
- Integrated Low Dropout Voltage Regulator – LDO
- Package type: 46-pin QFN

General Description

The device is a Flash Memory A/D type 8-bit high performance RISC architecture microcontroller, specifically designed for Power Bank applications. Offering users the convenience of Flash Memory multi-programming features, the 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.

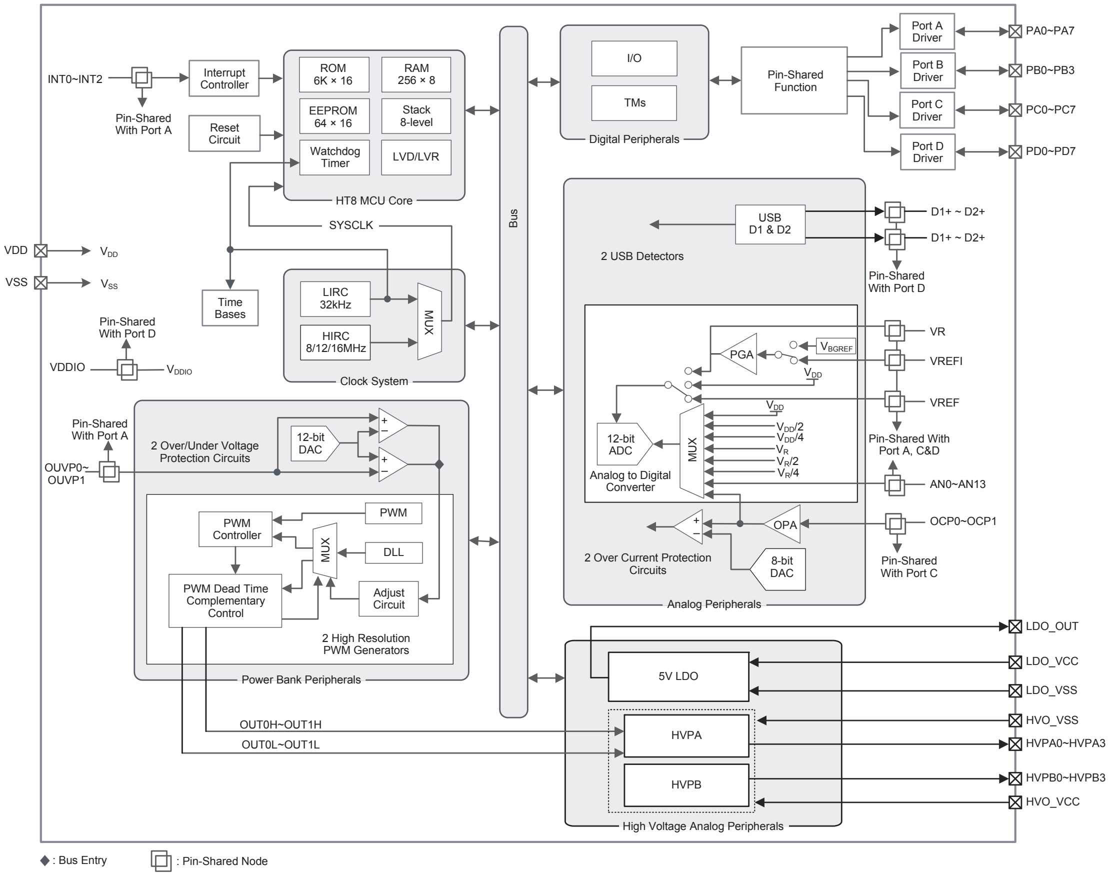
Analog features include a multi-channel 12-bit A/D converter and an internal 5V LDO (Low Dropout Regulator) for voltage regulator, two over current protection functions, two sets of over/ under voltage protection functions, Auto Adjust High Resolution PWM with Delay Lock Loop/ Dead-Time function and a USB device auto detection function. Multiple and extremely flexible Timer Modules provide timing, pulse generation and PWM generation functions. Protective features such as an internal Watchdog Timer, Low Voltage Reset and Low Voltage Detector coupled with excellent noise immunity and ESD protection ensure that reliable operation is maintained in hostile electrical environments.

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

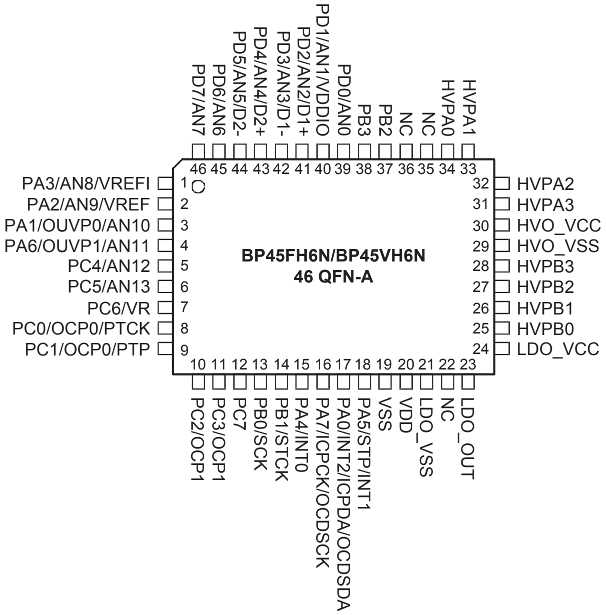
The High Voltage Output function specific to high voltage and high current applications is also fully integrated within the device. The inclusion of flexible I/O programming features, Time-Base functions along with many other features ensure that the device will find excellent use in different power bank applications.

Circuitry specific to Power Bank applications is also fully integrated within the device. These include functions such as over and under voltage protection, over current protection and auto detect. These features combine to ensure that a minimum of external components is required to implement Power Bank applications, providing the benefits of reduced component count and reduced circuit board areas.

Block Diagram



Pin Assignment



Note: 1. If the pin-shared pin functions have multiple outputs simultaneously, the desired pin-shared function is determined by the corresponding software control bits.
2. The OCDSDA and OCDSCK pins are supplied for the OCDS dedicated pins and as such only available for the BP45VH6N device which is the OCDS EV chip for the BP45FH6N device.

Pin Description

With the exception of the power pins and some relevant transformer control pins, all pins on the device can be referenced by their Port name, e.g. PA0, PA1 etc., which refer to the digital I/O function of the pins. However these Port pins are also shared with other function such as the Analog to Digital Converter, Timer Module pins etc. 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/INT2/ICPDA/OCDSDA	PA0	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	INT2	INTEG INTC0	ST	—	External interrupt 2 input
	ICPDA	—	ST	CMOS	In-circuit programming data/address pin
	OCDSDA	—	ST	CMOS	On-chip debug support data/address pin – for EV chip only
PA1/OUVP0/AN10	PA1	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	OUVP0	PAS0	AN	—	OVP/UVP 0 input
	AN10	PAS0	AN	—	A/D converter external signal input channel 10
PA2/AN9/VREF	PA2	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	AN9	PAS0	AN	—	A/D converter external signal input channel 9
	VREF	PAS0	AN	—	A/D converter and OVPn/OUVPn D/A converter external reference input

Pin Name	Function	OPT	I/T	O/T	Descriptions
PA3/AN8/VREFI	PA3	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	AN8	PAS0	AN	—	A/D converter external signal input channel 8
	VREFI	PAS0	AN	—	A/D converter external reference voltage input
PA4/INT0	PA4	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	INT0	PAS1 INTEG INTC0	ST	—	External interrupt 0 input
PA5/STP/INT1	PA5	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	STP	PAS1	ST	CMOS	STM output or STM capture input
	INT1	PAS1 INTEG INTC0	ST	—	External interrupt 1 input
PA6/OUVP1/AN11	PA6	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	OUVP1	PAS1	AN	—	OVP/UVP 1 input
	AN11	PAS1	AN	—	A/D converter external signal input channel 11
PA7/ICPCK/OCDSCK	PA7	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	ICPCK	—	ST	—	ICP clock input
	OCDSCK	—	ST	—	OCDS clock input – for EV chip only.
PB0	PB0	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
PB1/STCK	PB1	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	STCK	—	ST	—	STM input
PB2~PB3	PB2~PB3	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
PC0/OCP0/PTCK	PC0	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	OCP0	PCS0	AN	—	OCP0 input
	PTCK	PCS0	ST	—	PTM input
PC1/OCP0/PTP	PC1	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	OCP0	PCS0	AN	—	OCP0 input
	PTP	PCS0	ST	CMOS	PTM output or PTM capture input
PC2/OCP1	PC2	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	OCP1	PCS0	AN	—	OCP1 input
PC3/OCP1	PC3	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	OCP1	PCS0	AN	—	OCP1 input
PC4/AN12	PC4	PCPU PCS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN12	PCS1	AN	—	A/D converter external signal input channel 12
PC5/AN13	PC5	PCPU PCS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN13	PCS1	AN	—	A/D converter external signal input channel 13

Pin Name	Function	OPT	I/T	O/T	Descriptions
PC6/VR	PC6	PCPU PCS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	VR	PCS1	—	AN	A/D converter external reference voltage output
PC7	PC7	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up
PD0/AN0	PD0	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN0	PDS0	AN	—	A/D converter external signal input channel 0
PD1/AN1/VDDIO	PD1	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN1	PDS0	AN	—	A/D converter external signal input channel 1
	VDDIO	PDS0	PWR	—	Power supply for PD2~PD5 input/output pins
PD2/AN2/D1+	PD2	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN2	PDS0	AN	—	A/D converter external signal input channel 2
	D1+	PDS0	—	AN	USB DAC0 output pin
PD3/AN3/D1-	PD3	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN3	PDS0	AN	—	A/D converter external signal input channel 3
	D1-	PDS0	—	AN	USB DAC1 output pin
PD4/AN4/D2+	PD4	PDPU PDS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN4	PDS1	AN	—	A/D converter external signal input channel 4
	D2+	PDS1	—	AN	USB DAC2 output pin
PD5/AN5/D2-	PD5	PDPU PDS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN5	PDS1	AN	—	A/D converter external signal input channel 5
	D2-	PDS1	—	AN	USB DAC3 output pin
PD6/AN6	PD6	PDPU PDS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN6	PDS1	AN	—	A/D converter external signal input channel 6
PD7/AN7	PD7	PDPU PDS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN7	PDS1	AN	—	A/D converter external signal input channel 7
HVPA0~HVPA3	HVPA0~ HVPA3	HVPAEN	—	AN	High voltage Output
HVPB0~HVPB3	HVPB0~ HVPB3	HVPBEN	—	AN	High voltage Output
VDD	VDD	—	PWR	—	Digital positive power supply
VSS	VSS	—	PWR	—	Digital negative power supply, ground
HVO_VCC	HVO_ VCC	—	PWR	—	HVO and Level Shifter High voltage power supply
HVO_VSS	HVO_VSS	—	PWR	—	High voltage negative power supply, ground
LDO_OUT	LDO_OUT	—	—	PWR	LDO output voltage
LDO_VCC	LDO_VCC	—	PWR	—	LDO High voltage power supply
LDO_VSS	LDO_VSS	—	PWR	—	LDO negative power supply, ground

Legend: I/T: Input type;
OPT: Optional by register option;
ST: Schmitt Trigger input;
AN: Analog signal.

O/T: Output type;
PWR: Power;
CMOS: CMOS output;

Absolute Maximum Ratings

Supply Voltage (V_{LDO_VCC}, V_{HVO_VCC})	V_{SS}-0.3V to 13.5V
Supply Voltage (V_{DD})	V_{SS}-0.3V to 6.0V
High Voltage Input Voltage	V_{SS}-0.3V to V_{HVO_VCC}+0.3V
Input Voltage	V_{SS}-0.3V to V_{DD}+0.3V
Storage Temperature	-50˚C to 125˚C
Operating Temperature	-40˚C to 85˚C
I_{OH} Total	-120mA
I_{OL} Total	120mA
Total Power Dissipation	600mW

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

D.C. Characteristics

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

Operating Voltage Characteristics

Ta=-40˚C~85˚C

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_{DD}	Operating Voltage – HIRC	f_{SYS}=f_{HIRC}=8MHz	2.2	—	5.5	V
		f_{SYS}=f_{HIRC}=12MHz	2.7	—	5.5	
		f_{SYS}=f_{HIRC}=16MHz	3.3	—	5.5	
	Operating Voltage – LIRC	f_{SYS}=f_{LIRC}=32kHz	V_{LVR}	—	5.5	V

Standby Current Characteristics

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

Symbol	Standby Mode	Test Conditions		Min.	Typ.	Max.	Max.@ 85˚C	Unit
		V_{DD}	Conditions					
I_{STB}	SLEEP Mode	3V	WDT on	—	1.5	4.0	TBD	μA
		5V		—	3.0	5.0	TBD	
	IDLE0 Mode	3V	f_{SUB} on	—	3.0	5.0	TBD	μA
		5V		—	5.0	10	TBD	
	IDLE1 Mode – HIRC	2.2V	f_{SUB} on, f_{SYS}=8MHz	—	TBD	TBD	TBD	μA
		3V		—	360	500	TBD	
		5V		—	600	800	TBD	
		2.7V	f_{SUB} on, f_{SYS}=12MHz	—	TBD	TBD	TBD	μA
		3V		—	540	750	TBD	
		5V		—	800	1200	TBD	
		3.3V	f_{SUB} on, f_{SYS}=16MHz	—	TBD	TBD	TBD	mA
		5V		—	2.0	2.4	TBD	

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

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

Operating Current Characteristics

Ta=-40°C~85°C

Symbol	Operating Mode	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{DD}	Operating Current – LIRC	3V	f_{SYS}=32kHz	—	10	20	μA
		5V		—	30	50	
	Operating Current – HIRC	2.2V	f_{SYS}=8MHz	—	TBD	TBD	mA
		3V		—	0.8	1.2	
		5V		—	1.6	2.4	
		3V	f_{SYS}=12MHz	—	1.2	1.8	mA
		5V		—	2.4	3.6	
		3.3V	f_{SYS}=16MHz	—	TBD	TBD	mA
		5V		—	3.6	4.8	

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

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

A.C. Characteristics

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

High Speed Internal Oscillator – HIRC – Frequency Accuracy

During the program writing operation the writer will trim the HIRC oscillator at a user selected HIRC frequency and user selected voltage of either 3V or 5V.

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Temp.				
f_{HIRC}	8MHz Writer Trimmed HIRC Frequency	3V/5V	25˚C	-1%	8	+1%	MHz
			-40˚C~85˚C	-2%	8	+2%	
		2.2V~5.5V	25˚C	-2.5%	8	+2.5%	
			-40˚C~85˚C	-3%	8	+3%	
	12MHz Writer Trimmed HIRC Frequency	3V/5V	25˚C	-1%	12	+1%	MHz
			-40˚C~85˚C	-2%	12	+2%	
		2.7V~5.5V	25˚C	-2.5%	12	+2.5%	
			-40˚C~85˚C	-3%	12	+3%	
	16MHz Writer Trimmed HIRC Frequency	5V	25˚C	-1%	16	+1%	MHz
			-40˚C~85˚C	-2%	16	+2%	
		3.3V~5.5V	25˚C	-2.5%	16	+2.5%	
			-40˚C~85˚C	-3%	16	+3%	

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

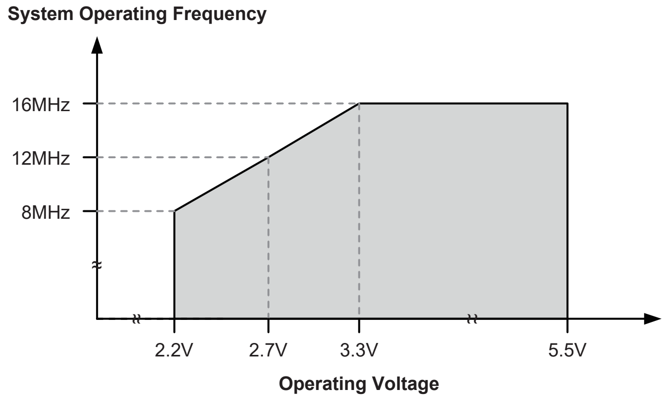
2. The row below the 3V/5V trim voltage row is provided to show the values for the full V_{DD} range operating voltage. It is recommended that the trim voltage is fixed at 3V for application voltage ranges from 2.2 to 3.6V and fixed at 5V for application voltage ranges from 3.3V to 5.5V.

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

Low Speed Internal Oscillator Characteristics – LIRC

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Temp.				
f_{LIRC}	Oscillator Frequency	5V	25˚C	-10%	32	+10%	kHz
		5V±0.5V	-40˚C~85˚C	-40%	32	+40%	
		V_{LVR}~5.5V	-40˚C~85˚C	-50%	32	+60%	

Operating Frequency Characteristic Curves



System Start Up Time Characteristics

Ta=-40˚C~85˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
t_{SST}	System Start-up Time Wake-up from condition where f_{SYS} is off	—	$f_{SYS}=f_H$~$f_H/64$, $f_H=f_{HIRC}$	16	—	—	t_{HIRC}
		—	$f_{SYS}=f_{SUB}=f_{LIRC}$	2	—	—	t_{LIRC}
	System Start-up Time Wake-up from condition where f_{SYS} is on	—	$f_{SYS}=f_H$~$f_H/64$, $f_H=f_{HIRC}$	2	—	—	t_H
		—	$f_{SYS}=f_{SUB}=f_{LIRC}$	2	—	—	t_{SUB}
	System Start-up Time Wake-up from condition where WDT time-out hardware cold reset	—	—	0	—	—	t_H
	System Speed Switch Time FAST to SLOW Mode or SLOW to FAST Mode	—	f_{HIRC} switches from off → on	16	—	—	t_{HIRC}
t_{RSTD}	System Reset Delay Time Reset source from Power-on Reset or LVR Hardware Reset	—	—	8.3	16.7	33.3	ms
	System Reset Delay Time LVRC/WDTC Software Reset	—	—				
	System Reset Delay Time Reset source from WDT Overflow hardware cold reset	—	—	8.3	16.7	33.3	ms
t_{SRESET}	Minimum Software Reset Width to Reset	—	—	45	90	120	μs

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

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

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

Input/Output Characteristics

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

Ta=25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{IL}	Input Low Voltage for I/O Ports except PD2~PD5 Pins	5V	—	0	—	1.5	V
		—	—	0	—	$0.2V_{DD}$	
V_{IH}	Input High Voltage for I/O Ports except PD2~PD5 Pins	5V	—	3.5	—	5.0	V
		—	—	$0.8V_{DD}$	—	V_{DD}	
I_{OL}	Sink Current for I/O Ports except PD2~PD5 Pins	3V	$V_{OL}=0.1V_{DD}$	16	32	—	mA
		5V		32	65	—	
I_{OH}	Source Current for I/O Ports except PD2~PD5 Pins	3V	$V_{OH}=0.9V_{DD}$	-4	-8	—	mA
		5V		-8	-16	—	
R_{PH}	Pull-high Resistance for I/O Ports except PD2~PD5 Pins (Note)	3V	—	20	60	100	kΩ
		5V	—	10	30	50	
t_{TCK}	PTCK and STCK Clock Input Minimum Pulse Width	—	—	0.3	—	—	μs
t_{INT}	Interrupt Input Pin Minimum Pulse Width	—	—	10	—	—	μs

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

Ta=25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	V_{DD} Power Supply for PD2~PD5 Pins	—	—	2.2	5.0	5.5	V
V_{DDIO}	V_{DDIO} Power Supply for PD2~PD5 Pins	—	—	1.8	—	V_{DD}	V
V_{IL}	Input Low Voltage for PD2~PD5 Pins	5V	—	0	—	1.5	V
		—	Pin power=V_{DD} or V_{DDIO}	0	—	0.2 (V_{DD} or V_{DDIO})	
V_{IH}	Input High Voltage for PD2~PD5 Pins	5V	—	3.5	—	5.0	V
		—	Pin power=V_{DD} or V_{DDIO}	$0.8V_{DD}$	—	V_{DD} or V_{DDIO}	
I_{OL}	Sink Current for PD2~PD5 Pins	3V	$V_{OL}=0.1(V_{DD}$ or $V_{DDIO})$, $V_{DDIO}=V_{DD}$	16	32	—	mA
		5V	$V_{OL}=0.1(V_{DD}$ or $V_{DDIO})$, $V_{DDIO}=V_{DD}$	32	65	—	
			$V_{OL}=0.1(V_{DD}$ or $V_{DDIO})$, V_{DDIO}=3V	20	40	—	
I_{OH}	Source Current for PD2~PD5 Pins	3V	$V_{OH}=0.9(V_{DD}$ or $V_{DDIO})$, $V_{DDIO}=V_{DD}$	-4	-8	—	mA
		5V	$V_{OH}=0.9(V_{DD}$ or $V_{DDIO})$, $V_{DDIO}=V_{DD}$	-8	-16	—	
			$V_{OH}=0.9(V_{DD}$ or $V_{DDIO})$, V_{DDIO}=3V	-2.5	-5	—	
R_{PH}	Pull-high Resistance for PD2~PD5 Pins (Note)	3V	$V_{DDIO}=V_{DD}$, LVPU=0, PDPU[5:2]=1111B	20	60	100	kΩ
		5V	$V_{DDIO}=V_{DD}$, LVPU=0, PDPU[5:2]=1111B	10	30	50	
		3V	$V_{DDIO}=V_{DD}$, LVPU=1, PDPU[5:2]=1111B	6.67	15	23	
		5V	$V_{DDIO}=V_{DD}$, LVPU=1, PDPU[5:2]=1111B	3.5	7.5	12	
I_{LEAK}	Input Leakage Current for PD2~PD5 Pins	5V	$V_{IN}=V_{DD}$ or $V_{IN}=V_{DDIO}$ or $V_{IN}=V_{SS}$	—	—	±1	μA

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

Memory Characteristics

Ta=-40˚C~85˚C

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

LVD/LVR Electrical Characteristics

Ta=-40˚C~85˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{LVR}	Low Voltage Reset Voltage	—	LVR enable	-5%	2.55	+5%	V
V_{LVD}	Low Voltage Detection Voltage	—	LVD enable, voltage select 2.7V	-5%	2.7	+5%	V
		—	LVD enable, voltage select 3.0V		3.0		
		—	LVD enable, voltage select 3.3V		3.3		
		—	LVD enable, voltage select 3.6V		3.6		
		—	LVD enable, voltage select 4.0V		4.0		
I_{LVR}	Additional Current for LVR Enable	—	LVD disable	—	—	10	μA
I_{LVD}	Additional Current for LVD Enable	—	LVR disable	—	60	90	μA
t_{LVDS}	LVDO Stable Time	—	For LVR enable, LVD off → on	—	—	15	μs
t_{LVR}	Minimum Low Voltage Width to Reset	—	—	120	240	480	μs
t_{LVD}	Minimum Low Voltage Width to Interrupt	—	—	60	120	240	μs

A/D Converter Electrical Characteristics

Ta=-40˚C~85˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.2	—	5.5	V
V_{ADI}	Input Voltage	—	—	0	—	V_{REF}	V
V_{REF}	Reference Voltage	—	—	2	—	V_{DD}	V
DNL	Differential Non-linearity	3V	SAINS[3:0]=0000B, SAVRS[1:0]=00B, V_{REF}=V_{DD}, t_{ADCK}=0.5μs	-3	—	+3	LSB
		5V					
		3V	SAINS[3:0]=0000B, SAVRS[1:0]=00B, V_{REF}=V_{DD}, t_{ADCK}=10μs				
		5V					

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
INL	Integral Non-linearity	3V	SAINS[3:0]=0000B, SAVRS[1:0]=00B, V_{REF}=V_{DD}, t_{ADCK}=0.5μs	-4	—	+4	LSB
		5V					
		3V	SAINS[3:0]=0000B, SAVRS[1:0]=00B, V_{REF}=V_{DD}, t_{ADCK}=10μs				
		5V					
I_{ADC}	Additional Current Consumption for A/D Converter Enable	3V	No load, t_{ADCK}=0.5μs	—	0.2	0.4	mA
		5V		—	0.3	0.6	
t_{ADCK}	Clock Period	—	—	0.5	—	10	μs
t_{ON2ST}	A/D Converter On-to-Start Time	—	—	4	—	—	μs
t_{ADS}	Sampling Time	—	—	—	4	—	t_{ADCK}
t_{ADC}	Conversion Time (Including A/D Sample and Hold Time)	—	—	—	16	—	t_{ADCK}

Reference Voltage Characteristics

Ta=25˚C, unless otherwise specified.

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.2	—	5.5	V
V_{BGREF}	Bandgap Reference Voltage	—	—	-1%	1.2	+1%	V
		—	Ta=-40˚C~85˚C	-2%	1.2	+2%	
I_{BGREF}	Operating Current	5.5V	Ta=-40˚C~85˚C	—	25	40	μA
PSRR	Power Supply Rejection Ratio	—	V_{RIPPLE}=1V_{P-P}, f_{RIPPLE}=100Hz	75	—	—	dB
En	Output Noise	—	f=0.1Hz~10Hz	—	300	—	$μV_{RMS}$
I_{DRV}	Buffer Driving Capability	—	ΔV_{BGREF}=-1%	1	—	—	mA
I_{SD}	Shutdown Current	—	VBGREN=0	—	—	0.1	μA
t_{START}	Startup Time	2.2V~5.5V	—	—	—	400	μs

Note: 1. All the above parameters are measured under conditions of no load condition unless otherwise described.
2. A 0.1μF ceramic capacitor should be connected between VDD and GND.
3. The V_{BGREF} voltage is used as the A/D converter reference voltage input.

Programmable Gain Amplifier Electrical Characteristics

Ta=25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{PGA}	Additional Current for PGA Enable	3V	No load	—	390	550	μA
		5V		—	500	650	
V_{OR}	PGA Maximum Output Voltage Range	3V	—	V_{SS}+0.1	–	V_{DD}-0.1	V
		5V	—	V_{SS}+0.1	–	V_{DD}-0.1	
V_{VR}	PGA Fix Voltage Range	2.2V~5.5V	V_{RI}=V_{BGREF}, (PGAIS=1)	-1%	2	+1%	V
		3.2V~5.5V	V_{RI}=V_{BGREF}, (PGAIS=1)	-1%	3	+1%	
		4.2V~5.5V	V_{RI}=V_{BGREF}, (PGAIS=1)	-1%	4	+1%	
I_{VR}	V_R Maximum Output Current	5V	V_{VR}=4V, ΔV_{VR}=-1%	1.2	2.4	—	mA
V_{IR}	PGA Input Voltage Range	3V	Gain=1, PGAIS=0, Relative gain, Gain error<±5%	V_{SS}+0.1	—	V_{DD}-1.4	V
		5V		V_{SS}+0.1	—	V_{DD}-1.4	

Over Current Protection Electrical Characteristics

Ta=25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{OCP}	Operating Current	3V	OCPnEN[1:0]=01B, DAC V_{REF}=2.5V	—	—	1000	μA
		5V		—	730	1250	
V_{OS_CMP}	Comparator Input Offset Voltage	3V	Without calibration (OCPnCOF[4:0]=10000B)	-15	—	15	mV
		5V		-15	—	15	
		3V	With calibration	-4	—	4	mV
		5V		-4	—	4	
V_{HYS}	Hysteresis	3V	—	10	40	60	mV
		5V		10	40	60	
V_{CM_CMP}	Comparator Common Mode Voltage Range	3V	—	V_{SS}	–	V_{DD}-1.4	V
		5V		V_{SS}	–	V_{DD}-1.4	
V_{OS_OPA}	OPA Input Offset Voltage	3V	Without calibration (OCPnOOF[5:0]=100000B)	-15	—	15	mV
		5V		-15	—	15	
		3V	With calibration	-4	—	4	mV
		5V		-4	—	4	
V_{CM_OPA}	OPA Common Mode Voltage Range	3V	—	V_{SS}	–	V_{DD}-1.4	V
		5V		V_{SS}	–	V_{DD}-1.4	
V_{OR}	OPA Maximum Output Voltage Range	3V	—	V_{SS}+0.1	–	V_{DD}-0.1	V
		5V		V_{SS}+0.1	–	V_{DD}-0.1	
Ga	PGA Gain Accuracy	3V	All gain	-5	–	5	%
		5V		-5	–	5	
Ro	R2R Output Resistance	3V	—	—	10	—	kΩ
		5V		—	10	—	
V_{REF}	DAC Reference voltage	3V/5V	OCPnVRS[1:0]=01B	2	—	V_{DD}	V
DNL	Differential Non-linearity	3V	DAC V_{REF}=V_{DD}	-1.5	—	+1.5	LSB
		5V		-1	—	+1	
INL	Integral Non-linearity	3V	DAC V_{REF}=V_{DD}	-2	—	+2	LSB
		5V		-1.5	—	+1.5	

Over/Under Voltage Protection Electrical Characteristics

Ta=25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{OUVP}	Operating Current	3V	UVPnEN=1, OVPnEN=1, DAC V_{REF}=2.5V	—	—	420	μA
		5V		—	300	500	
V_{OS}	Input Offset Voltage	3V	With calibration	-4	—	4	mV
		5V		-4	—	4	
V_{HYS}	Hysteresis	3V	—	10	20	30	mV
		5V	—	10	20	30	
V_{CM}	Common Mode Voltage Range	3V	—	V_{SS}	—	V_{DD}-1.4	V
		5V	—	V_{SS}	—	V_{DD}-1.4	
R_O	R2R output resistance	5V	—	—	10	—	kΩ
DNL	Differential Non-linearity	3V	DAC V_{REF}=V_{DD}	-1.5	—	+1.5	LSB
		5V		-1	—	+1	

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
INL	Integral Non-linearity	3V	DAC V_{REF}=V_{DD}	-2.0	—	+2.0	LSB
		5V		-1.5	—	+1.5	
t_{RP}	Response time	3V	With 100mV overdrive (Note)	—	200	400	ns
		5V		—	200	400	

Note: Load Condition: C_{LOAD}=50pF.

Delay Lock Loop Electrical Characteristics

Ta=25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{DLL}	Operating Current	3V	DLLEN=1	—	0.9	1.3	mA
		5V		—	1.5	2.1	
f_{DLL}	Operating Frequency	2.2V~5.5V	f_{HIRC}=8MHz	-10%	8	+10%	MHz
		2.7V~5.5V	f_{HIRC}=12MHz	-10%	12	+10%	
		3.0V~5.5V	f_{HIRC}=16MHz	-10%	16	+10%	
t_{DLLS}	DLL Stable Time	2.2V~5.5V	DLLEN from 0 to 1	—	20	30	μs
t_{PD}	Propagation Delay Time	2.2V~5.5V	DLLEN=1	1.5	2.5	3.3	ns

USB Auto Detection Electrical Characteristics

Ta=25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DAC}	DAC Operating Voltage	—	—	2.2	—	5.5	V
I_{DAC}	DAC Operating Current	3V	No Load	—	0.6	0.9	mA
		5V		—	1.0	1.5	
I_{DACSD}	DAC Shutdown Current	—	No Load	—	—	0.1	μA
N_R	DAC Resolution	—	—	—	8	—	bits
DNL	DAC Differential Non-linearity	—	No Load, DAC reference=V_{DD}	-1	—	+1	LSB
INL	DAC Integral Non-linearity	—	No Load, DAC reference=V_{DD}	-2	—	+2	LSB
V_{DACO}	Output Voltage Range	—	Code=0000H	V_{SS}	—	V_{SS}+0.2	V
		—	Code=0FFFH	V_{REF}-0.2	—	V_{REF}	
V_{REF}	Reference Voltage	—	—	2	—	V_{DD}	V
t_{ST}	Settling Time	3V	C_{LOAD}=50pF	—	—	5	μs
		5V		—	—	5	
R_O	R2R Output Resistance	3V	—	—	3	—	kΩ
		5V		—	5	—	
OSRR	Offset Error	3V	V_{REF}=V_{DD}=3V, Data word=128	—	—	50	mV
		5V	V_{REF}=V_{DD}=5V, Data word=128	—	—	80	
GERR	Gain Error	3V	V_{REF}=V_{DD}=3V, Data word=128	—	—	50	mV
		5V	V_{REF}=V_{DD}=5V, Data word=128	—	—	80	
I_{DACOL}	Output Sink Current	3V	Data word=0000H, V_{DACO}=0.1V_{REF}	20	—	—	mA
		5V		40	—	—	
I_{DACOH}	Output Source Current	3V	Data word=0FFFH, V_{DACO}=0.9V_{REF}	20	—	—	mA
		5V		40	—	—	

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{SC}	Output Short-circuit Current	3V	Data word=0FFFH	0.25	—	—	mA
		5V		0.4	—	—	
R_{ON}	Analog Switch on Resistance Between D1+/D2+ and D1-/D2-	5V	—	—	20	35	Ω
R_{PL}	Pull-low Resistance for D1+, D2+. D1-, D2-	5V	—	15	20	30	kΩ
ERR	The Error for D1+, D1-, D2+, D2- Output voltage	5V	DAC reference=V_{DD}, ADUDAn[7:0]=10010100B, D1+, D1-, D2+ or D2- connect 150kΩ to ground	2.57	2.7	2.84	V
		5V	DAC reference=V_{DD}, ADUDAn[7:0]=01101110B, D1+, D1-, D2+, D2- connect 150kΩ to ground	1.9	2.0	2.1	V

Low Dropout Regulator Electrical Characteristics

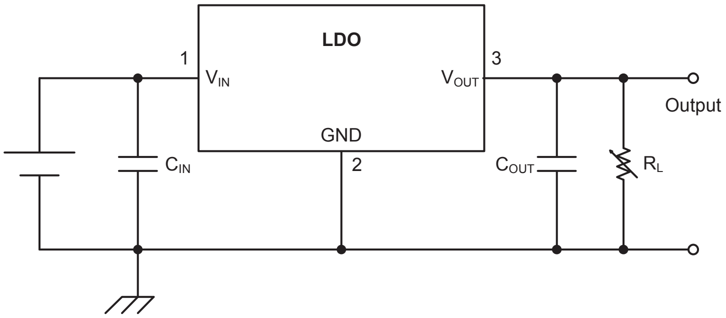
V_{IN}=V_{OUT}+1V, C_{LOAD}=10μF+0.1μF, Ta=-40˚C~85˚C, unless otherwise specified

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{IN}	Conditions				
V_{IN}	Input Voltage	—	—	V_{OUT}+0.1	6	13.5	V
V_{OUT}	Output Voltage	—	Ta=25˚C, I_{LOAD}=1mA, V_{OUT}=5.0V	-2%	5.0	2%	V
		—	I_{LOAD}=1mA, V_{OUT}=5.0V	-3%	5.0	3%	V
ΔV_{LOAD}	Load Regulation[(1)]	—	1mA≤I_{LOAD}≤70mA	—	0.02	0.05	%/mA
V_{DROP}	Dropout Voltage[(2)]	—	V_{OUT}=5V, ΔV_{OUT}=-2%, I_{LOAD}=50mA	—	150	300	mV
	Bypass Mode Dropout Voltage	—	3V<V_{IN}<5.1V, I_{LOAD}=50mA	—	300	600	mV
I_{OUT}	Output Current	—	ΔV_{OUT}=-3%	70	—	—	mA
		—	V_{IN}=V_{OUT}+2V, ΔV_{OUT}=-5%	100	—	—	mA
I_Q	Quiescent Current	12V	No load	—	3	8	μA
ΔV_{LINE}	Line Regulation	—	(V_{OUT}+1V)≤V_{IN}≤12V, I_{LOAD}=1mA	—	—	0.5	%/V
TC	Temperature Coefficient	—	I_{LOAD}=10mA	—	±1.5	±2	mV/˚C
ΔV_{OUT_RIPPLE}	Output Voltage Ripple	6V	I_{LOAD}=10mA	—	—	10	mV
RR	Ripple Rejection[(3)]	—	V_{IN}=10V_{DC}+2$V_{P-P(AC)}$, I_{LOAD}≤50mA, f=120Hz	35	—	—	dB
I_{LIMIT}	Current Limit	6V	ΔV_{OUT}=-10%	150	—	—	mA
t_{START}	LDO Startup Time	6V	I_{LOAD}=1mA, V_{OUT}=5V±5%	—	—	10	ms

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

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

3. Ripple rejection ratio measurement circuit. RR=20×log($\Delta V_{IN}/\Delta V_{OUT}$), C_{IN}=10μF/C_{OUT}=10μF+0.1μF.



4. Application information for LDO load capacitor selection for stability:

Recommended Output Capacitor

Ta=25˚C, unless otherwise specified

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
C_{LOAD}	Output Load Capacitor	—	Capacitor	4.7	10	—	μF

In common with most regulators, the LDO requires external capacitors between V_{OUT} and ground for regulator stability. Capacitor values of 4.7μF or large are acceptable, provided the smaller ESR is less than 10Ω. Aluminum electrolytic capacitor is suitable, provided they meet the requirements described above.

For better load transient response purposes, use a combination of a C_{LOAD} 10μF and extra 0.1μF capacitor on V_{OUT}. Note that the 0.1μF capacitor is always required on V_{OUT} and strong recommended be a multi-layer ceramic capacitor. The internal regulator is designed to be stable with an output filter capacitor C_{LOAD} and ESR as recommended.

High Voltage Output Electrical Characteristics

Ta=25˚C

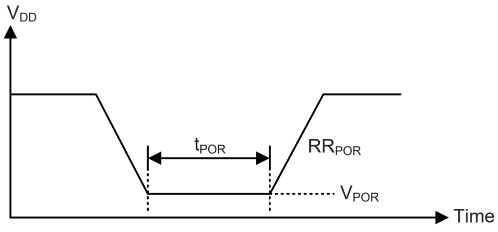
Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
High Voltage Output Electrical Characteristics for HVPA0~HVPA3 Pins							
V_{IN}	Input Voltage	—	See note	3.0	—	13.5	V
I_{OH}	Source Current for High Voltage Output Pins	—	V_{OH}=0.9×V_{IN}, V_{IN}=12V	-300	-360	—	mA
I_{OL}	Sink Current for High Voltage Output Pins	—	V_{OL}=0.1×V_{IN}, V_{IN}=12V	300	360	—	mA
High Voltage Output Electrical Characteristics for HVPB0~HVPB3 Pins							
V_{IL}	Input Low Voltage for High Voltage Output Pins	—	See note	3.0	—	13.5	V
I_{OH}	Source Current for High Voltage Output Pins	—	V_{OH}=0.9×V_{IN}, V_{IN}=12V	-50	-90	—	mA
I_{OL}	Sink Current for High Voltage Output Pins	—	V_{OL}=0.1×V_{IN}, V_{IN}=12V	50	90	—	mA

Note: When the voltage is below 3V, the HVPAn and HVPBn pins can only be used for general output high or low control, the switching speed between high and low can not exceeds 25Hz, otherwise the leakage current and output waveform distortion problems will occur.

Power-on Reset Characteristics

Ta=-40˚C~85˚C

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



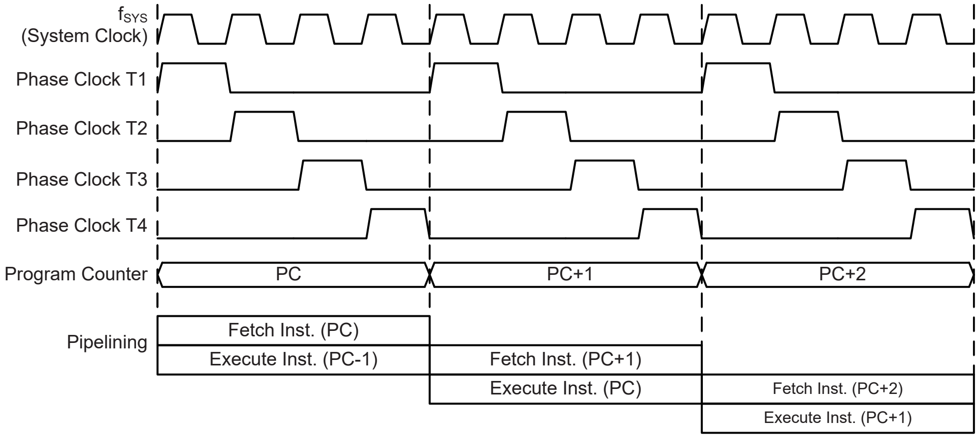
System Architecture

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

Clocking and Pipelining

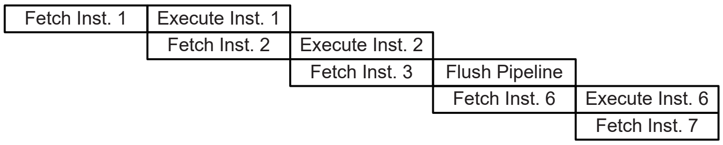
The main system clock, derived from either the HIRC or LIRC oscillator is subdivided into four internally generated non-overlapping clocks, T1~T4. The Program Counter is incremented at the beginning of the T1 clock during which time a new instruction is fetched. The remaining T2~T4 clocks carry out the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

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



System Clocking and Pipelining

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



Instruction Fetching

Program Counter

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

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

Program Counter	
Program Counter High Byte	PCL Register
PC12~PC8	PCL7~PCL0

Program Counter

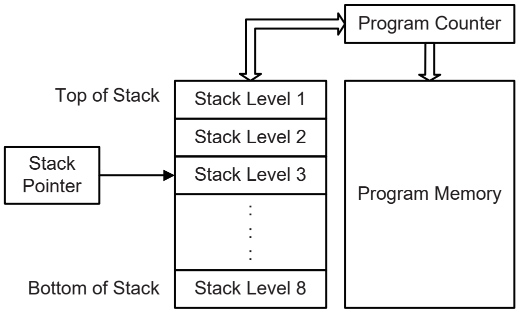
The lower byte of the Program Counter, known as the Program Counter Low register or PCL, is available for program control and is a readable and writeable register. By transferring data directly into this register, a short program jump can be executed directly; however, as only this low byte is available 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 multiple levels and neither part of the data nor part of the program space, and is neither readable nor writeable. The activated level is indexed by the Stack Pointer, and is neither readable nor writeable. At a subroutine call or interrupt acknowledge signal, the contents of the Program Counter are pushed onto the stack. At the end of a subroutine or an interrupt routine, signaled by a return instruction, RET or RETI, the Program Counter is restored to its previous value from the stack. After a device reset, the Stack Pointer will point to the top of the stack.

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

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



Arithmetic and Logic Unit – ALU

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

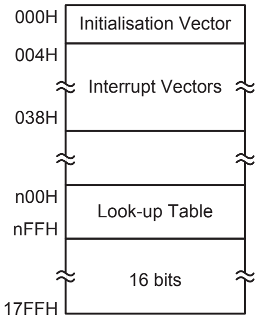
- Arithmetic operations:
  ADD, ADDM, ADC, ADCM, SUB, SUBM, SBC, SBCM, DAA,
  LADD, LADDM, LADC, LADCM, LSUB, LSUBM, LSBC, LSBCM, LDAA
- Logic operations:
  AND, OR, XOR, ANDM, ORM, XORM, CPL, CPLA,
  LAND, LANDM, LOR, LORM, LXOR, LXORM, LCPL, LCPLA
- Rotation:
  RRA, RR, RRCA, RRC, RLA, RL, RLCA, RLC,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

Flash Program Memory

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

Structure

The Program Memory has a capacity of 6K×16 bits. The Program Memory is addressed by the Program Counter and also contains data, table information and interrupt entries. Table data, which can be setup in any location within the Program Memory, is addressed by a separate table pointer register.



Program Memory Structure

Special Vectors

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

Look-up Table

Any location within the Program Memory can be defined as a look-up table where programmers can store fixed data. To use the look-up table, the table pointer must first be setup by placing the address of the look up data to be retrieved in the table pointer 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 corresponding table read instruction such as "TABRD [m]" or "TABRDL [m]" respectively when the memory [m] is located in Sector 0. If the memory [m] is located in other sectors except Sector 0, the data can be retrieved from the program memory using the corresponding extended table read instruction such as "LTABRD [m]" or "LTABRDL [m]" respectively. When the instruction is executed, the lower order table byte from the Program Memory will be transferred to the user defined Data Memory register [m] as specified in the instruction. The higher order table data byte from the Program Memory will be transferred to the TBLH special register. Any unused bits in this transferred higher order byte will be read as "0".

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

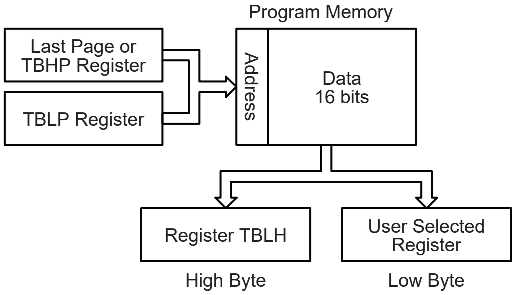


Table Program Example

The following example shows how the table pointer and table data is defined and retrieved from the microcontroller. This example uses raw table data located in the Program Memory which is stored there using the ORG statement. The value at this ORG statement is "1700H" which refers to the start address of the last page within the 6K Program Memory of the microcontroller. The table pointer low byte register is setup here to have an initial value of "06H". This will ensure that the first data read from the data table will be at the Program Memory address "1706H" or 6 locations after the start of the last page. Note that the value for the table pointer is referenced to the first address of the specific page pointed by TBLP and TBHP register if the "TABRD [m]" or "LTABRD [m]" instruction is being used. The high byte of the table data which in this case is equal to zero will be transferred to the TBLH register automatically when the "TABRD [m]" or "LTABRD [m]" instruction is executed.

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

Table Read Program Example

```
tempreg1 db?          ; temporary register #1
tempreg2 db?          ; temporary register #2
:
:
mov a,06h             ; initialise table pointer - note that this address is referenced
mov tblp,a            ; to the last page or the page that tbhp pointed
mov a,17h             ; initialise high table pointer
mov tbhp,a
:
:
tabrd tempreg1        ; transfers value in table referenced by table pointer
                      ; data at program memory address "1706H" transferred to tempreg1
                      ; and TBLH
dec tblp              ; reduce value of table pointer by one
tabrd tempreg2        ; transfers value in table referenced by table pointer
                      ; data at program memory address "1705H" transferred to tempreg2
                      ; and TBLH
                      ; in this example the data "1AH" is transferred to tempreg1 and
                      ; data "0FH" to tempreg2
```

```
:
:
org 1700h             ; 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.

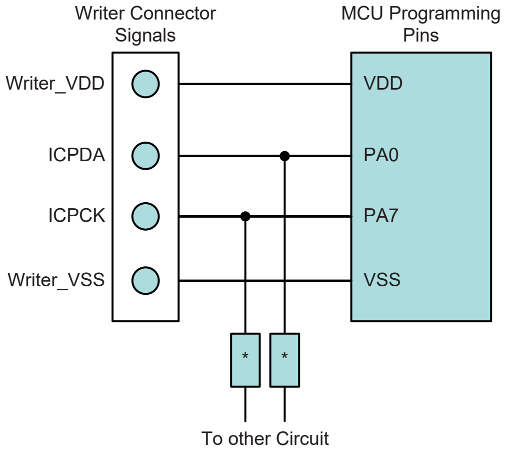
As an additional convenience, Holtek has provided a means of programming the microcontroller in-circuit using a 4-pin interface. This provides manufacturers with the possibility of manufacturing their circuit boards complete with a programmed or un-programmed microcontroller, and then programming or 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	PA7	Programming Clock
VDD	VDD	Power Supply (5V)
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 and one line for the reset. The technical details regarding the in-circuit programming of the device is beyond the scope of this document and will be supplied in supplementary literature.

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



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

On-Chip Debug Support – OCDS

There is an EV chip named BP45VH6N which is used to emulate the BP45FH6N 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 (5V)
VSS	VSS	Ground

Data Memory

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

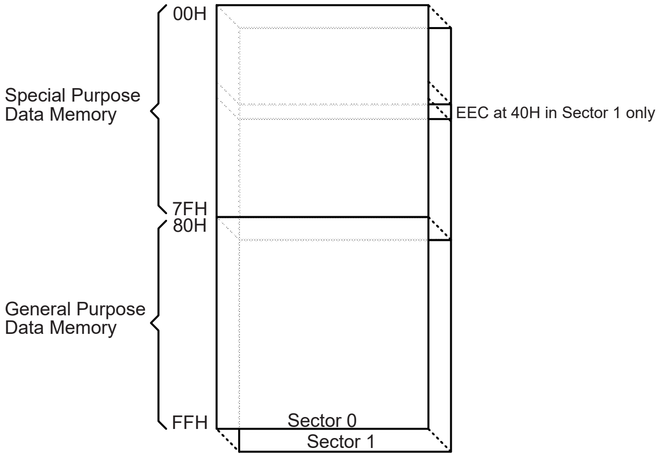
Categorized into two types, the first of these is an area of RAM, known as the Special Function Data Memory. These registers have fixed locations and are necessary for correct operation of the device. Many of these registers can be read from and written to directly under program control, however, some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

Structure

The Data Memory is subdivided into two sectors, all of which are implemented in 8-bit wide RAM. Each of the Data Memory Sector is categorized into two types, the special Purpose Data Memory and the General Purpose Data Memory. Switching between the different Data Memory sectors is achieved by setting the Memory Pointers to the correct value. The start address of the Data Memory is the address 00H.

Special Purpose Data Memory Address	General Purpose Data Memory	
	Capacity	Sector: Address
Sector 0: 00H~7FH Sector 1: 00H~7FH	256×8	0: 80H~FFH 1: 80H~FFH

Data Memory Summary



Data Memory Structure

Data Memory Addressing

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

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

General Purpose Data Memory

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

Special Purpose Data Memory

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

	Sector 0	Sector 1		Sector 0	Sector 1
00H	IAR0	▒	40H	▒	EEC
01H	MP0		41H	EEA	▒
02H	IAR1		42H	EED	
03H	MP1L		43H	▨	
04H	MP1H		44H	PWM1P	
05H	ACC		45H	PWM1D	
06H	PCL		46H	DLL1	
07H	TBLP		47H	PWM1C	
08H	TBLH		48H	DLLC	
09H	TBHP		49H	INTEG	
0AH	STATUS		4AH	INTC0	
0BH	▒		4BH	INTC1	
0CH	IAR2		4CH	INTC2	
0DH	MP2L		4DH	INTC3	
0EH	MP2H		4EH	MFI0	
0FH	LVDC		4FH	MFI1	
10H	SMOD		50H	OCPPC	
11H	CTRL		51H	OUVP0PC	
12H	▒	HVPAEN	52H	OUVP1PC	
13H	LVRC	▒	53H	OUTPC	
14H	PA	HVPCA	54H	PMPS	
15H	PAC	HVPSA	55H	LVPUC	
16H	PAPU	HVPBEN	56H	OUVP0C0	
17H	PAWU	▒	57H	OUVP0C1	
18H	PAS0	HVPCB	58H	OUVP0C2	
19H	PAS1	HVPSB	59H	OUVP0C3	
1AH	▒	▒	5AH	OVP0DAH	
1BH	▒		5BH	OVP0DAL	
1CH	PCS0		5CH	UVP0DAH	
1DH	PCS1		5DH	UVP0DAL	
1EH	PDS0		5EH	OUVP1C0	
1FH	PDS1		5FH	OUVP1C1	
20H	PWM0P	TBC	60H	OUVP1C2	
21H	PWM0D	▒	61H	OUVP1C3	
22H	DLL0		62H	OVP1DAH	
23H	PWM0C		63H	OVP1DAL	
24H	ADJ0DT	STMC0	64H	UVP1DAH	
25H	ADJ0S	STMC1	65H	UVP1DAL	
26H	ADJ0C	STMDL	66H	VBGRC	
27H	ADJ0MAXH	STMDH	67H	▒	
28H	ADJ0MAXL	STMAL	68H		
29H	ADJ0MINH	STMAH	69H		
2AH	ADJ0MINL	STMRP	6AH	OCP0C0	
2BH	ADJ0BH	PTMC0	6BH	OCP0C1	
2CH	ADJ0BL	PTMC1	6CH	OCP0DA	
2DH	ADJ1DT	PTMDL	6DH	OCP0OCAL	
2EH	ADJ1S	PTMDH	6EH	OCP0CCAL	
2FH	ADJ1C	PTMAL	6FH	OCP1C0	
30H	ADJ1MAXH	PTMAH	70H	OCP1C1	
31H	ADJ1MAXL	PTMPRL	71H	OCP1DA	
32H	ADJ1MINH	PTMPRH	72H	OCP1OCAL	
33H	ADJ1MINL	▒	73H	OCP1CCAL	
34H	ADJ1BH		74H	ADUDA0	
35H	ADJ1BL		75H	ADUDA1	
36H	WDTC		76H	ADUDA2	
37H	PC		77H	ADUDA3	
38H	PCC		78H	ADUC0	
39H	PCPU		79H	▒	
3AH	PD		7AH	ADUC1	
3BH	PDC		7BH	SADC0	
3CH	PDPU		7CH	SADC1	
3DH	PB		7DH	SADC2	
3EH	PBC		7EH	SADOH	
3FH	PBPU		7FH	SADOL	

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

Special Purpose Data Memory

Special Function Register Description

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

Indirect Addressing Registers – IAR0, IAR1, IAR2

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

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

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

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

Indirect Addressing Program Example 1

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

Indirect Addressing Program Example 2

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

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

Direct Addressing Program Example using extended instructions

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

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

Accumulator – ACC

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

Program Counter Low Register – PCL

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

Look-up Table Registers – TBLP, TBHP, TBLH

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

Status Register – STATUS

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

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

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

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

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

• STATUS Register

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

"x": unknown

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

Bit 6 **CZ**: The operational result of different flags for different instructions

For SUB/SUBM/LSUB/LSUBM instructions, the CZ flag is equal to the Z flag.

For SBC/SBCM/LSBC/LSBCM instructions, the CZ flag is the "AND" operation result which is performed by the previous operation CZ flag and current operation zero flag.

For other instructions, the CZ flag will not be affected.

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

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

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

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

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

Bit 0 **C**: Carry flag
0: No carry-out
1: An operation results in a carry during an addition operation or if a borrow does not take place during a subtraction operation

The "C" flag is also affected by a rotate through carry instruction.

EEPROM Data Memory

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

EEPROM Data Memory Structure

The EEPROM Data Memory capacity is 64×8 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 a data register in Sector 0 and a single control register in Sector 1.

EEPROM Registers

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

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

EEPROM Register List

- **EEA Register**

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

Bit 7~6 Unimplemented, read as "0"

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

- **EED Register**

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

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

- **EEC Register**

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

Bit 7~4 Unimplemented, read as “0”

Bit 3 **WREN**: Data EEPROM Write Enable
0: Disable
1: Enable

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

Bit 2 **WR**: EEPROM Write Control
0: Write cycle has finished
1: Activate a write cycle

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

Bit 1 **RDEN**: Data EEPROM Read Enable
0: Disable
1: Enable

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

Bit 0 **RD**: EEPROM Read Control
0: Read cycle has finished
1: Activate a read cycle

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

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

Reading Data from the EEPROM

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

Writing Data to the EEPROM

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

Write Protection

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

EEPROM Interrupt

The EEPROM write interrupt is generated when an EEPROM write cycle has ended. The EEPROM interrupt must first be enabled by setting the DEE bit in the relevant interrupt register. 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 Memory Pointer high byte register, MP1H or MP2H, could be normally cleared to zero as this would inhibit access to Sector 1 where the EEPROM control register exists. Although certainly not necessary, consideration might be given in the application program to the checking of the validity of new write data by a simple read back process.

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

Programming Examples

Reading Data from the EEPROM – Polling Method

```
MOV  A, EEPROM_ADRES      ; user defined address
MOV  EEA, A
MOV  A, 40H               ; setup memory pointer MP1L
MOV  MP1L, A              ; MP1 points to EEC register
MOV  A, 01H               ; setup memory pointer MP1H
MOV  MP1H, A
SET  IAR1.1               ; set RDEN bit, enable read operations
SET  IAR1.0               ; start Read Cycle - set RD bit
BACK:
SZ   IAR1.0               ; check for read cycle end
JMP  BACK
CLR  IAR1                 ; disable EEPROM read if no more read operations are required
CLR  MP1H
MOV  A, EED               ; move read data to register
MOV  READ_DATA, A
```

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

Writing Data to the EEPROM – Polling Method

```
MOV  A, EEPROM_ADRES      ; user defined address
MOV  EEA, A
MOV  A, EEPROM_DATA       ; user defined data
MOV  EED, A
MOV  A, 40H               ; setup memory pointer MP1L
MOV  MP1L, A              ; MP1 points to EEC register
MOV  A, 01H               ; setup memory pointer MP1H
MOV  MP1H, A
CLR  EMI
SET  IAR1.3               ; set WREN bit, enable write operations
SET  IAR1.2               ; start Write Cycle - set WR bit - executed immediately
                          ; after set WREN bit
SET  EMI
BACK:
SZ   IAR1.2               ; check for write cycle end
JMP  BACK
CLR  MP1H
```

Oscillators

Various oscillator options offer the user a wide range of functions according to their various application requirements. The flexible features of the oscillator functions ensure that the best optimisation can be achieved in terms of speed and power saving. Oscillator selections and operation are selected 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. Fully integrated internal oscillators, requiring no external components, are provided to form a wide range of both fast and slow system oscillators. The higher frequency oscillators provide higher performance but carry with it the disadvantage of higher power requirements, while the opposite is of course true for the lower frequency oscillators. With the capability of dynamically switching between fast and slow system clock, the device has the flexibility to optimize the performance/power ratio, a feature especially important in power sensitive portable applications.

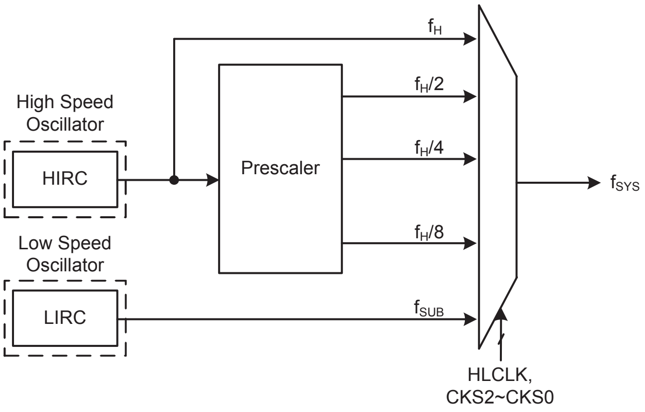
Type	Name	Frequency
Internal High Speed RC	HIRC	8/12/16MHz
Internal Low Speed RC	LIRC	32kHz

Oscillator Types

System Clock Configurations

There are two oscillator sources, a high speed oscillator and a low speed oscillator. The high speed oscillator is the internal 8/12/16MHz RC oscillator, HIRC. The low speed oscillator is the internal 32kHz RC oscillator, LIRC. Selecting whether the low or high speed oscillator is used as the system oscillator is implemented using the HLCLK and CKS2~CKS0 bits in the SMOD register and as the system clock can be dynamically selected.

The actual source clock used for the high speed oscillators is chosen via configuration options. The frequency of the slow speed or 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

Internal High Speed RC Oscillator – HIRC

The internal RC oscillator is a fully integrated system oscillator requiring no external components. The internal RC oscillator has three fixed frequencies of 8MHz, 12MHz and 16MHz, which is selected using a configuration option. 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 either 3V or 5V and at a temperature of 25˚C degrees, the fixed oscillation frequency of 8MHz, 12MHz or 16MHz will have a tolerance within 1%. Note that if this internal system clock option is selected, it requires no external pins for its operation.

Internal 32kHz Oscillator – LIRC

The Internal 32kHz System Oscillator is the low frequency oscillator. It is a fully integrated RC oscillator with a typical frequency of 32kHz at 5V, requiring no external components for its implementation. 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%.

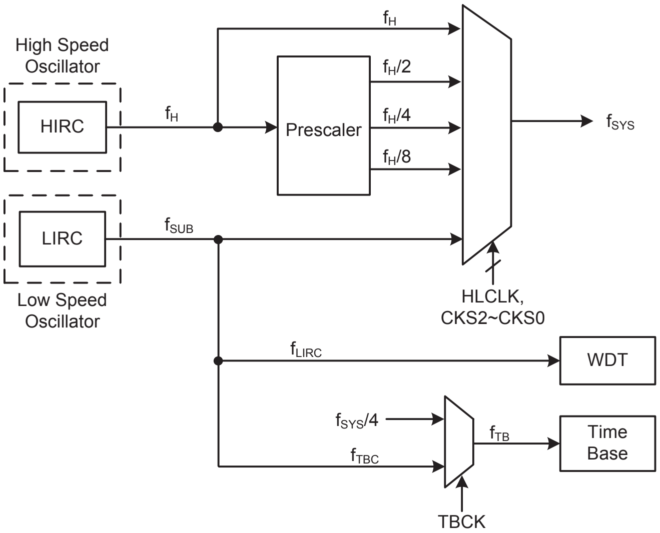
Operating Modes and System Clocks

Present day applications require that their microcontrollers have high performance but often still demand that they consume as little power as possible, conflicting requirements that are especially true in battery powered portable applications. The fast clocks required for high performance will by their nature increase current consumption and of course vice versa, lower speed clocks reduce current consumption. As Holtek has provided the device with both high and low speed clock sources and the means to switch between them dynamically, users 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 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 the HIRC oscillator. The low speed system clock source can be sourced from the LIRC oscillator. The other choice, which is a divided version of the high speed system oscillator has a range of $f_H/2$~$f_H/8$.



Device Clock Configurations

Note: When the system clock source f_{SYS} is switched to f_{SUB} from f_H, the high speed oscillator will stop to conserve the power. Thus there are also no the divided frequencies of f_H for peripheral circuit to use.

System Operation Modes

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

Operating Mode	Description			
	CPU	f_{SYS}	f_{LIRC}	f_{TBC}
FAST mode	On	f_H~$f_H/8$	On	On
SLOW mode	On	f_{SUB}	On	On
ILDE0 mode	Off	Off	On	On
IDLE1 mode	Off	On	On	On
SLEEP mode	Off	Off	On	Off

Note: The f_{LIRC} clock is always on as the WDT function is always enabled.

FAST Mode

This is one of the main operating modes where the microcontroller has all of its functions operational and where the system clock is provided by the high speed oscillator. This mode operates allowing the microcontroller to operate normally with a clock source will come from the high speed oscillator HIRC. The high speed oscillator will however first be divided by a ratio ranging from 1 to 8, 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 f_{SUB}. The f_{SUB} clock is derived from the LIRC oscillator. Running the microcontroller in this mode allows it to run with much lower operating currents. In the SLOW Mode, the f_H is off.

SLEEP 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 SLEEP mode the CPU will be stopped. However the f_{LIRC} clock will continue to operate to keep the Watchdog Timer function.

IDLE0 Mode

The IDLE0 Mode is entered when an HALT instruction is executed and when 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 and the system oscillator will be stopped, but the low speed oscillator will be on.

IDLE1 Mode

The IDLE1 Mode is entered when an HALT instruction is executed and when the 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, 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 low frequency clock will be on.

Note: If LVDEN=1 and the SLEEP or IDLE mode is entered, the LVD and bandgap functions will not be disabled, and the f_{SUB} clock will be forced to open.

Control Registers

The registers, SMOD and CTRL, are used to control the system clock and the corresponding oscillator configurations.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SMOD	CKS2	CKS1	CKS0	—	LTO	HTO	IDLEN	HLCLK
CTRL	FSYSON	—	—	—	—	LVRF	LRF	WRF

System Operating Mode Control Register List

- **SMOD Register**

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

Bit 7~5 **CKS2~CKS0**: System clock selection
000: f_{SUB}
001: f_{SUB}
010: Undefined
011: Undefined
100: Undefined
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 directly derived from f_H or f_{SUB}, a divided version of the high speed system oscillator can also be chosen as the system clock source.

Bit 4 Unimplemented, read as “0”

Bit 3 **LTO**: LIRC System OSC SST ready flag
0: Not ready
1: Ready

This is the low speed system oscillator SST 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 change to a high level after 1~2 cycles.

Bit 2 **HTO**: HIRC System OSC SST ready flag
0: Not ready
1: Ready

This is the high speed system oscillator SST ready flag which indicates when the high speed system oscillator is stable after a wake-up has occurred. The flag will change to a high level 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 an 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 an HALT instruction is executed.

Bit 0 **HLCLK**: System Clock Selection
0: $f_H/2$~$f_H/8$ or f_{SUB}
1: f_H

This bit is used to select if the f_H clock or the $f_H/2$~$f_H/8$ 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 $f_H/2$~$f_H/8$ 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**: System Clock 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**: LVRC Control register software reset flag
Described elsewhere.

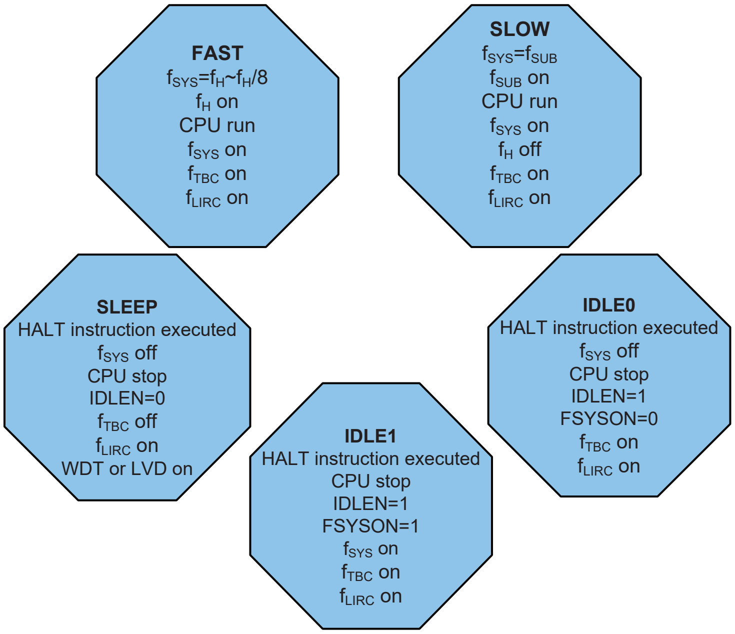
Bit 0 **WRF**: WDTC Control register software reset flag
Described elsewhere.

Operating Mode Switching

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

In simple terms, Mode Switching between the FAST Mode and SLOW Mode is executed using the HLCLK bit and CKS2~CKS0 bits in the SMOD register while Mode Switching from the FAST/SLOW Modes to the SLEEP/IDLE Modes is executed via the HALT instruction. When a HALT instruction is executed, whether the device enters the IDLE0 Mode, IDLE1 Mode or the SLEEP Mode is determined by the condition of the IDLEN bit in the SMOD register and FSYSON in the CTRL register.

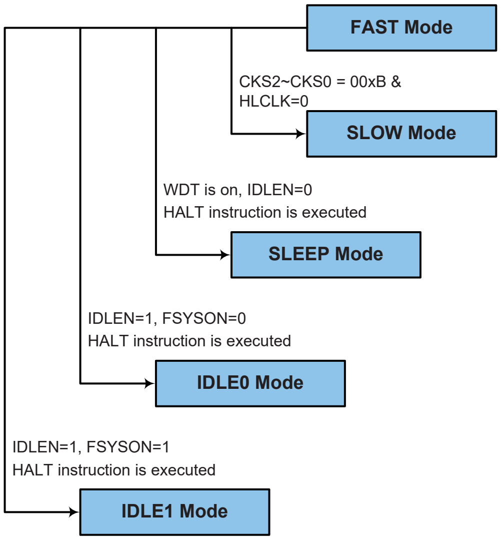
When the HLCLK bit switches to a low level, which implies that clock source is switched from the high speed clock source, f_H, to the clock source, $f_H/2$~$f_H/8$ or f_{SUB}. If the clock is from the f_{SUB}, the high speed clock source will stop running to conserve power. The accompanying flowchart shows what happens when the device moves between the various operating modes.



FAST Mode to SLOW Mode Switching

When running in the FAST Mode, which uses the high speed system oscillator, and therefore consumes more power, the system clock can switch to run in the SLOW Mode by HLCLK bit to "0" and setting 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.

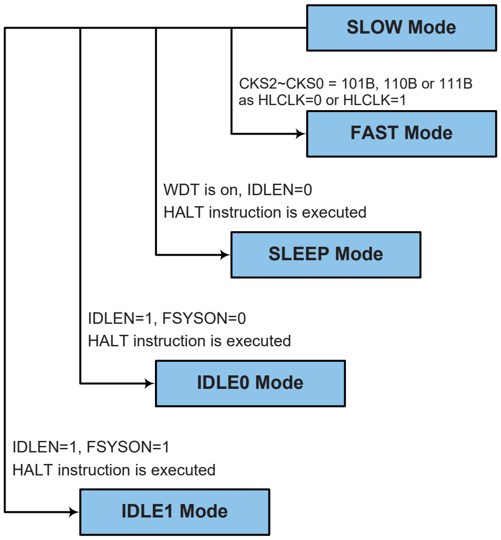
The SLOW Mode is sourced from the LIRC oscillator and therefore requires this oscillator to be stable before full mode switching occurs. This is monitored using the LTO bit in the SMOD register.



SLOW Mode to FAST Mode Switching

In SLOW mode the system clock is derived from f_{SUB}. When system clock is switched back to the FAST mode from f_{SUB}, the HLCLK bit should be set to "1" or HLCLK bit is "0", but CKS2~CKS0 is set to "101", "110" or "111".

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



Entering the SLEEP Mode

There is only one way for the device to enter the SLEEP Mode and that is to execute the "HALT" instruction in the application program with 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 and Time Base clock will be stopped and the application program will stop at the "HALT" instruction, but the WDT and LVD will remain on with the clock source coming from the f_{LIRC} clock.
- The Data Memory contents and registers will maintain their present condition.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.
- The WDT will be cleared and resume counting since the WDT function is always enabled.

Entering the IDLE0 Mode

There is only one way for the device to enter the IDLE0 Mode and that is to execute the "HALT" instruction in the application program with the 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 Time Base clock and the f_{LIRC} clock will be on.
- The Data Memory contents and registers will maintain their present condition.

- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.
- The WDT will be cleared and resume counting since the WDT function is always enabled.

Entering the IDLE1 Mode

There is only one way for the device to enter the IDLE1 Mode and that is to execute the “HALT” instruction in the application program with 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, Time Base clock and the f_{LIRC} 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 I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.
- The WDT will be cleared and resume counting since the WDT function is always enabled.

Standby Current Considerations

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

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

Wake-up

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

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

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

When the device executes the “HALT” instruction, the PDF flag will be set to 1. The PDF flag will be cleared to 0 if the device experiences a system power-up or executes the clear Watchdog Timer instruction. If the system is woken up by a WDT overflow, a Watchdog Timer reset will be initiated and the TO flag will be set to 1. The TO flag is set if a WDT time-out occurs and causes a wake-up that only resets the Program Counter and Stack Pointer, other flags remain in their original status.

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

Watchdog Timer

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

Watchdog Timer Clock Source

The Watchdog Timer clock source is provided by the internal clock, f_{LIRC} which is sourced from the LIRC oscillator. The 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 frequency of 32kHz and this specified internal clock period can vary with V_{DD}, temperature and process variations.

Watchdog Timer Control Register

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

- **WDTC Register**

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

Bit 7~3 **WE4~WE0**: WDT function software control
01010B or 10101B: Enable
Other values: Reset MCU

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

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

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

- **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 instruction. If the program malfunctions for whatever reason, jumps to an unknown location, or enters an endless loop, the clear instruction will not be executed in the correct manner, in which case the Watchdog Timer will overflow and reset the device. There are five bits, WE4~WE0, in the WDTC register to offer the enable control and reset control of the Watchdog Timer. The WDT function will be enabled when the WE4~WE0 bits are set to a value of 10101B or 01010B. If the WE4~WE0 bits are set to any other values, other than 01010B and 10101B, it will reset the device after a delay time, t_{SRESET}. After power on these bits will have a value of 01010B.

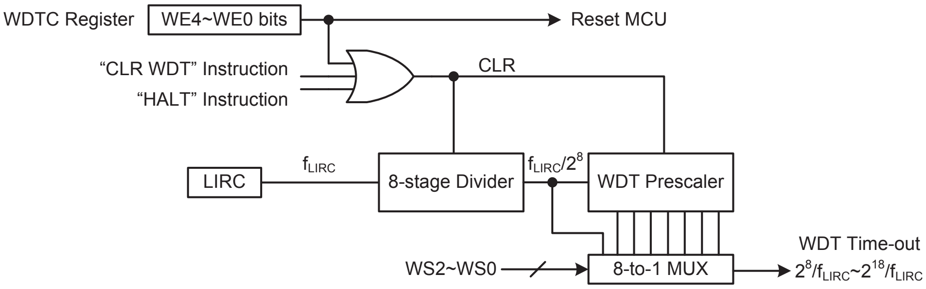
WE4~WE0 Bits	WDT Function
10101B/01010B	Enable
Any other values	Reset MCU

Watchdog Timer Enable/Reset Control

Under normal program operation, a Watchdog Timer time-out will initialise a device reset and set the status bit TO. 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 WDTC software reset, which means a certain value except 01010B and 10101B written into the WE4~WE0 bit filed, the second is using the Watchdog Timer software clear instruction and the third is via an HALT instruction.

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

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



Watchdog Timer

Reset and Initialisation

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

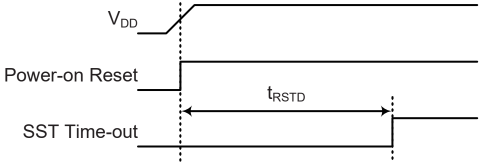
Another type of reset is when the Watchdog Timer overflows and resets the microcontroller. All types of reset operations result in different register conditions being setup. Another reset exists in the form of a Low Voltage Reset, LVR, where a full reset, is implemented in situations where the power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

Reset Functions

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

Power-on Reset

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



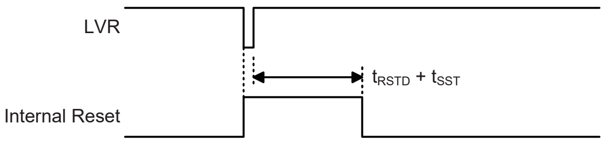
Note: t_{RSTD} is power-on delay, typical time=16.7ms

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 in the FAST and SLOW Mode with a specific LVR voltage, V_{LVR}. If the supply voltage of the device drops to within a range of 0.9V~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 voltage, i.e., a voltage in the range between 0.9V~V_{LVR} must exist for greater than the value t_{LVR} specified in the LVD/LVR Electrical Characteristics. If the low voltage state does not exceed this value, the LVR will ignore the low supply voltage and will not perform a reset function.

The actual V_{LVR} value is fixed at a value of 2.55V. However the LVS7~LVS0 bits still have effects on the LVR function. If these bits are changed to any other value except some certain values defined in the LVRC register by the environmental noise, the LVR will reset the device after t_{SRESET} time. 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 SLEEP or IDLE mode.



Note: t_{RSTD} is power-on delay, typical time=16.7ms

Low Voltage Reset Timing Chart

• LVRC Register

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

Bit 7~0 **LVS7~LVS0**: LVR Voltage Select control
01010101: 2.55V
00110011: 2.55V
10011001: 2.55V
10101010: 2.55V
Any other value: Generates MCU reset – register is reset to POR value

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

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

• 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"

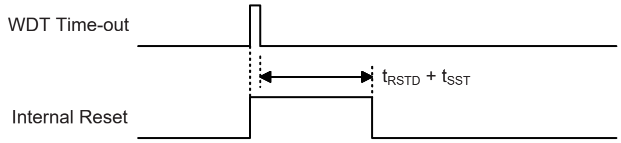
Bit 2 **LVRF**: LVR function reset flag
0: Not occur
1: Occurred
This bit is set high when a specific Low Voltage Reset situation condition occurs. This bit can only be cleared to zero by the application program.

Bit 1 **LRF**: LVR control register software reset flag
0: Not occur
1: Occurred
This bit is set to 1 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 0 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 in the FAST or SLOW mode, the Watchdog time-out flag TO will be set high.

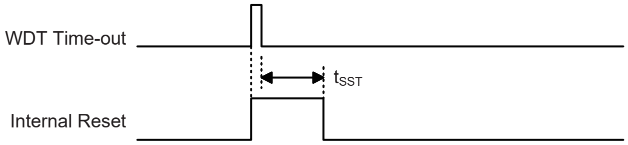


Note: t_{RSTD} is power-on delay, typical time=16.7ms

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 System Start Up Time Characteristics for t_{SST} details.



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

Reset Initial Conditions

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

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

"u" stands for unchanged

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

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

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

Register	Power On Reset	LVR Reset (Normal Operation)	WDT Time-out (Normal Operation)	WDT Time-out (SLEEP or IDLE)
IAR0	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP0	0000 0000	0000 0000	0000 0000	uuuu uuuu
IAR1	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP1L	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP1H	0000 0000	0000 0000	0000 0000	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	---x xxxx	---u uuuu	--- uuuu	---u uuuu
STATUS	xx00 xxxx	xxuu uuuu	uu1u uuuu	uu11 uuuu
IAR2	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2L	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2H	0000 0000	0000 0000	0000 0000	uuuu uuuu
LVDC	--00 -000	--00 -000	--00 -000	--uu -uuu
SMOD	111- 0010	111- 0010	111- 0010	uuu- uuuu
CTRL	0--- -x00	0--- -1uu	0--- -uuu	u--- -uuu
LVRC	0101 0101	uuuu uuuu	0101 0101	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS0	0000 00--	0000 00--	0000 00--	uuuu uu--
PAS1	--00 0000	--00 0000	--00 0000	--uu uuuu
PCS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCS1	--00 0000	--00 0000	--00 0000	--uu uuuu
PDS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDS1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWM0P	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWM0D	0000 0000	0000 0000	0000 0000	uuuu uuuu
DLL0	0000 ----	0000 ----	0000 ----	uuuu ----
PWM0C	0000 0000	0000 0000	0000 0000	uuuu uuuu
AD0JDT	--00 0000	--00 0000	--00 0000	--uu uuuu
ADJ0S	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADJ0C	000- xxx-	000- xxx-	000- xxx-	uuu- uuu-
ADJ0MAXH	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	Power On Reset	LVR Reset (Normal Operation)	WDT Time-out (Normal Operation)	WDT Time-out (SLEEP or IDLE)
ADJ0MAXL	0000 ----	0000 ----	0000 ----	uuuu ----
ADJ0MINH	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADJ0MINL	0000 ----	0000 ----	0000 ----	uuuu ----
ADJ0BH	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADJ0BL	0000 ----	0000 ----	0000 ----	uuuu ----
AD1JDT	--00 0000	--00 0000	--00 0000	--uu uuuu
ADJ1S	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADJ1C	000- xxx-	000- xxx-	000- xxx-	uuu- uuu-
ADJ1MAXH	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADJ1MAXL	0000 ----	0000 ----	0000 ----	uuuu ----
ADJ1MINH	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADJ1MINL	0000 ----	0000 ----	0000 ----	uuuu ----
ADJ1BH	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADJ1BL	0000 ----	0000 ----	0000 ----	uuuu ----
WDTC	0101 0011	0101 0011	0101 0011	uuuu uuuu
PC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PB	---- 1111	---- 1111	---- 1111	---- uuuu
PBC	---- 1111	---- 1111	---- 1111	---- uuuu
PBPU	---- 0000	---- 0000	---- 0000	---- uuuu
EEA	--00 0000	--00 0000	--00 0000	--uu uuuu
EED	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWM1P	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWM1D	0000 0000	0000 0000	0000 0000	uuuu uuuu
DLL1	0000 ----	0000 ----	0000 ----	uuuu ----
PWM1C	0000 0000	0000 0000	0000 0000	uuuu uuuu
DLLC	10-- ---0	10-- ---0	10-- ---0	uu-- ---u
INTEG	--00 0000	--00 0000	--00 0000	--uu uuuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC2	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC3	-000 0000	-000 0000	-000 0000	-uuu uuuu
MFI0	-000 0000	-000 0000	-000 0000	-uuu uuuu
MFI1	-000 0000	-000 0000	-000 0000	-uuu uuuu
OCPPC	0000 0000	0000 0000	0000 0000	uuuu uuuu
OUVP0PC	0000 0000	0000 0000	0000 0000	uuuu uuuu
OUVP1PC	0000 0000	0000 0000	0000 0000	uuuu uuuu
OUTPC	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	Power On Reset	LVR Reset (Normal Operation)	WDT Time-out (Normal Operation)	WDT Time-out (SLEEP or IDLE)
PMPS	---- --00	---- --00	---- --00	---- --uu
LVPUC	---- ---0	---- ---0	---- ---0	---- ---u
OUVP0C0	--00 0000	--00 0000	--00 0000	--uu uuuu
OUVP0C1	--00 0000	--00 0000	--00 0000	--uu uuuu
OUVP0C2	0001 0000	0001 0000	0001 0000	uuuu uuuu
OUVP0C3	0001 0000	0001 0000	0001 0000	uuuu uuuu
OVP0DAH	---- 0000	---- 0000	---- 0000	---- uuuu
OVP0DAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
UVP0DAH	---- 0000	---- 0000	---- 0000	---- uuuu
UVP0DAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
OUVP1C0	--00 0000	--00 0000	--00 0000	--uu uuuu
OUVP1C1	--00 0000	--00 0000	--00 0000	--uu uuuu
OUVP1C2	0001 0000	0001 0000	0001 0000	uuuu uuuu
OUVP1C3	0001 0000	0001 0000	0001 0000	uuuu uuuu
OVP1DAH	---- 0000	---- 0000	---- 0000	---- uuuu
OVP1DAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
UVP1DAH	---- 0000	---- 0000	---- 0000	---- uuuu
UVP1DAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
VBGRC	---- ---0	---- ---0	---- ---0	---- ---u
OCP0C0	0000 0--0	0000 0--0	0000 0--0	uuuu u--u
OCP0C1	--00 0000	--00 0000	--00 0000	--uu uuuu
OCP0DA	0000 0000	0000 0000	0000 0000	uuuu uuuu
OCP0OCAL	0010 0000	0010 0000	0010 0000	uuuu uuuu
OCP0CCAL	0001 0000	0001 0000	0001 0000	uuuu uuuu
OCP1C0	0000 0--0	0000 0--0	0000 0--0	uuuu u--u
OCP1C1	--00 0000	--00 0000	--00 0000	--uu uuuu
OCP1DA	0000 0000	0000 0000	0000 0000	uuuu uuuu
OCP1OCAL	0010 0000	0010 0000	0010 0000	uuuu uuuu
OCP1CCAL	0001 0000	0001 0000	0001 0000	uuuu uuuu
ADUDA0	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADUDA1	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADUDA2	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADUDA3	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADUC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADUC1	--00 0000	--00 0000	--00 0000	--uu uuuu
SADC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADC1	0000 -000	0000 -000	0000 -000	uuuu -uuu
SADC2	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
SADOH	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu (ADRFS=0)
				---- uuuu (ADRFS=1)
SADOL	xxxx ----	xxxx ----	xxxx ----	uuuu ---- (ADRFS=0)
				uuuu uuuu (ADRFS=1)

Register	Power On Reset	LVR Reset (Normal Operation)	WDT Time-out (Normal Operation)	WDT Time-out (SLEEP or IDLE)
HVPAEN	---- 0000	---- 0000	---- 0000	---- uuuu
HVPCA	---- 0000	---- 0000	---- 0000	---- uuuu
HVPSA	---- 0000	---- 0000	---- 0000	---- uuuu
HVPBEN	---- 0000	---- 0000	---- 0000	---- uuuu
HVPCB	---- 0000	---- 0000	---- 0000	---- uuuu
HVPSB	---- 0000	---- 0000	---- 0000	---- uuuu
TBC	0011 -111	0011 -111	0011 -111	uuuu -uuu
STMC0	0000 0---	0000 0---	0000 0---	uuuu u---
STMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDH	---- --00	---- --00	---- --00	---- --uu
STMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAH	---- --00	---- --00	---- --00	---- --uu
STMRP	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMC0	---- --00	---- --00	---- --00	---- --uu
PTMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDH	---- --00	---- --00	---- --00	---- --uu
PTMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMAH	---- --00	---- --00	---- --00	---- --uu
PTMPRL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMPRH	---- --00	---- --00	---- --00	---- --uu
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~PD. These I/O ports are mapped to the RAM Data Memory with specific addresses as shown in the Special Purpose Data Memory table. All of these I/O ports can be used for input and output operations. For input operation, these ports are non-latching, which means the inputs must be ready at the T2 rising edge of instruction "MOV A, [m]", where m denotes the port address. For output operation, all the data is latched and remains unchanged until the output latch is rewritten.

Register Name	Bit							
	7	6	5	4	3	2	1	0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PB	—	—	—	—	PB3	PB2	PB1	PB0
PBC	—	—	—	—	PBC3	PBC2	PBC1	PBC0
PBPU	—	—	—	—	PBPU3	PBPU2	PBPU1	PBPU0
PC	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	PCC7	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PCPU	PCPU7	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
PDC	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
PDPU	PDPU7	PDPU6	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
LVPUC	—	—	—	—	—	—	—	LVPU

"—": Unimplemented, read as "0"

I/O Logic Function Register List

Pull-high Resistors

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

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

- **PxPU Register**

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

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

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

- **LVPUC Register**

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

Bit 7~1 Unimplemented, read as “0”

Bit 0 **LVPU**: Pull-high resistor select when low voltage power supply
0: All pin pull high resistor is 60kΩ @ 3V
1: All pin pull high resistor is 15kΩ @ 3V

This bit is used to select the pull-high resistor value for low voltage power supply applications. The LVPU bit is only available when the corresponding pin pull-high function is enabled by setting the relevant pull-high control bit high. This bit will have no effect when the pull-high function is disabled.

Port A Wake-up

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

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

- **PAWU Register**

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

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

I/O Port Control Registers

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

• PxC Register

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

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

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

I/O Port Power Source Control

This device supports different I/O port power source selections for PD2~PD5. The port power can come from either the power pin VDD or VDDIO which is determined using the PMPS[1:0] bits in the PMPS register. The VDDIO power pin function should first be selected using the corresponding pin-shared function selection bits if the port power is supposed to come from the VDDIO pin. An important point to know is that the input power voltage on the VDDIO pin should be equal to or less than the device supply power voltage when the VDDIO pin is selected as the port power supply pin.

• PMPS Register

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

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **PMPS1~PMPS0**: PD2~PD5 pin power source selection
0x: VDD
1x: VDDIO

Pin-shared Functions

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

Pin-shared Function Selection Registers

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

The most important point to note is to make sure that the desired pin-shared function is properly selected and also deselected. For most pin-shared functions, to select the desired pin-shared function, the pin-shared function should first be correctly selected using the corresponding pin-shared control register. After that the corresponding peripheral functional setting should be configured and then the peripheral function can be enabled. However, a special point must be noted for some digital input pins, such as INTn, xTCK etc, which share the same pin-shared control configuration with their corresponding general purpose I/O functions when setting the relevant pin-shared control bit fields. To select these pin functions, in addition to the necessary pin-shared control and peripheral functional setup aforementioned, they must also be setup as an input by setting the corresponding bit in the I/O port control register. To correctly deselect the pin-shared function, the peripheral function should first be disabled and then the corresponding pin-shared function control register can be modified to select other pin-shared functions.

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	—	—
PAS1	—	—	PAS15	PAS14	PAS13	PAS12	PAS11	PAS10
PCS0	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
PCS1	—	—	PCS15	PCS14	PCS13	PCS12	PCS11	PCS10
PDS0	PDS07	PDS06	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
PDS1	PDS17	PDS16	PDS15	PDS14	PDS13	PDS12	PDS11	PDS10

Pin-shared Function Selection Register List

• PAS0 Register

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

Bit 7~6 **PAS07~PAS06**: PA3 Pin-Shared function selection
00/11: PA3
01: AN8
10: VREFI

Bit 5~4 **PAS05~PAS04**: PA2 Pin-Shared function selection
00/11: PA2
01: AN9
10: VREF

Bit 3~2 **PAS03~PAS02**: PA1 Pin-Shared function selection
00/10/11: PA1
01: OUVP0/AN10

Bit 1~0 Unimplemented, read as "0"

• PAS1 Register

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

Bit 7~6 Unimplemented, read as "0"

Bit 5~4 **PAS15~PAS14**: PA6 Pin-Shared function selection
00/10/11: PA6
01: OUVP1/AN11

Bit 3~2 **PAS13~PAS12**: PA5 Pin-Shared function selection
00/10/11: PA5/INT1
01: STP

When the PAS13~PAS12 bits are set to 01, the PAC5 bit in the PAC register should also be cleared to zero.

Bit 1~0 **PAS11~PAS10**: PA4 Pin-Shared function selection
00/01/10: PA4/INT0
11: Undefined

• PCS0 Register

Bit	7	6	5	4	3	2	1	0
Name	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
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 **PCS07~PCS06**: PC3 Pin-Shared function selection
00/10/11: PC3
01: OCP1

Note: If PCS0[7:4]=0101B, both the pins can be used for OCP1 input at the same time.

Bit 5~4 **PCS05~PCS04**: PC2 Pin-Shared function selection
00/10/11: PC2
01: OCP1

Note: If PCS0[7:4]=0101B, both the pins can be used for OCP1 input at the same time.

Bit 3~2 **PCS03~PCS02**: PC1 Pin-Shared function selection
00/11: PC1
01: OCP0
10: PTP

Note: If PCS0[3:0]=0101B, both the pins can be used for OCP0 input at the same time.

Bit 1~0 **PCS01~PCS00**: PC0 Pin-Shared function selection
00/10/11: PC0/PTCK
01: OCP0

Note: If PCS0[3:0]=0101B, both the pins can be used for OCP0 input at the same time.

• PCS1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCS15	PCS14	PCS13	PCS12	PCS11	PCS10
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~4 Unimplemented, read as "0"

Bit 5~4 **PCS15~PCS14**: PC6 Pin-Shared function selection
00/10/11: PC6
01: VR

Bit 3~2 **PCS13~PCS12**: PC5 Pin-Shared function selection
00/10/11: PC5
01: AN13

Bit 1~0 **PCS11~PCS10**: PC4 Pin-Shared function selection
00/10/11: PC4
01: AN12

• PDS0 Register

Bit	7	6	5	4	3	2	1	0
Name	PDS07	PDS06	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PDS07~PDS06**: PD3 Pin-Shared function selection
00: PD3
01: AN3
10: D1-
11: AN3 output plus DAC1 output signal

Bit 5~4 **PDS05~PDS04**: PD2 Pin-Shared function selection
00: PD2
01: AN2
10: D1+
11: AN2 output plus DAC0 output signal

Bit 3~2 **PDS03~PDS02**: PD1 Pin-Shared function selection
00/11: PD1
01: AN1
10: VDDIO

Bit 1~0 **PDS01~PDS00**: PD0 Pin-Shared function selection
00/10/11: PD0
01: AN0

• PDS1 Register

Bit	7	6	5	4	3	2	1	0
Name	PDS17	PDS16	PDS15	PDS14	PDS13	PDS12	PDS11	PDS10
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 **PDS17~PDS16**: PD7 Pin-Shared function selection
00/10/11: PD7
01: AN7

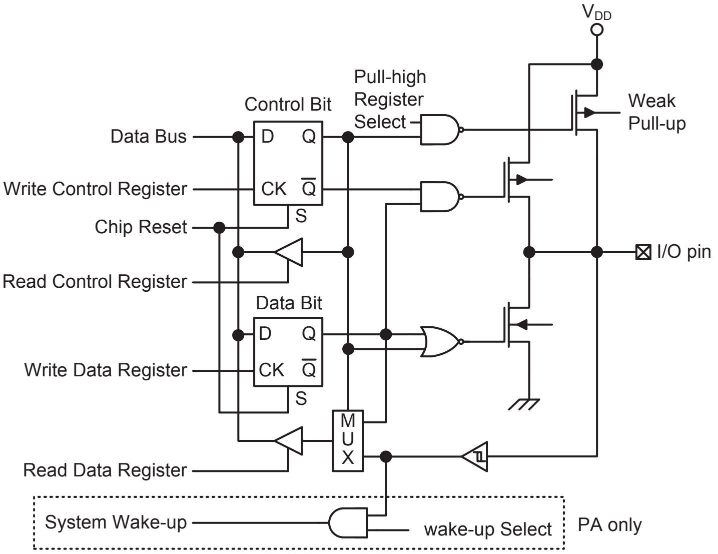
Bit 5~4 **PDS15~PDS14**: PD6 Pin-Shared function selection
00/10/11: PD6
01: AN6

Bit 3~2 **PDS13~PDS12**: PD5 Pin-Shared function selection
00: PD5
01: AN5
10: D2-
11: AN5 output plus DAC3 output signal

Bit 1~0 **PDS11~PDS10**: PD4 Pin-Shared function selection
00: PD4
01: AN4
10: D2+
11: AN4 output plus DAC2 output signal

I/O Pin Structures

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



Logic Function Input/Output Structure

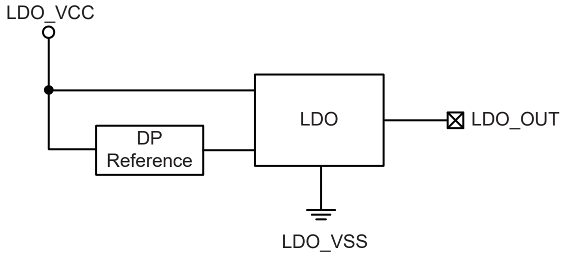
Programming Considerations

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

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

Low Dropout Regulator – LDO

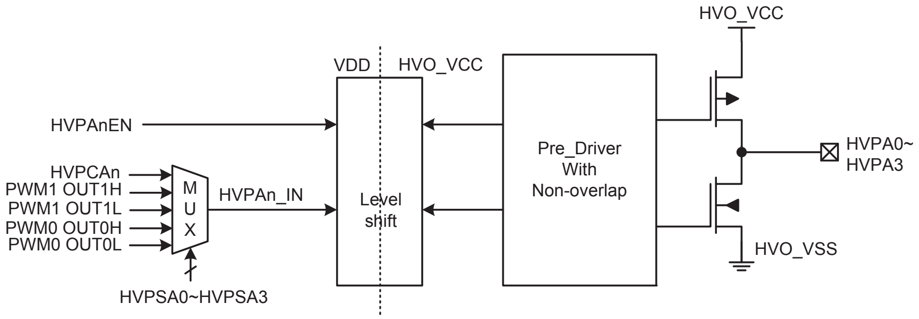
The device contains an internal Low Dropout Regulator, known as LDO. The LDO regulator can reduce a higher voltage approximately 5.1V to 13.5V on input pin LDO_VCC to a 5V level supplied on output pin LDO_OUT. When the LDO input voltage, V_{IN}, is ranged from 3V to 5.1V, the LDO output voltage, V_{OUT}, follows V_{IN}. This lower voltage level can provide a fixed power supply for internal or external circuits.



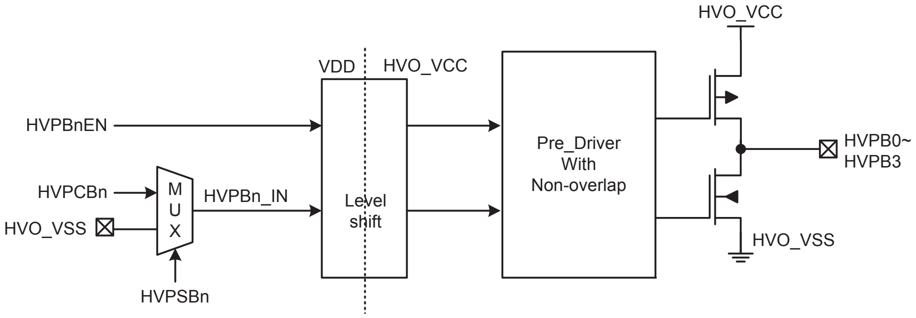
LDO Block Diagram

High Voltage Output Ports

The device provides several 12V high voltage output lines, known as HVPxn, where "x" stands for A or B port and "n" stands for the specific High Voltage Output number. These high voltage ports can convert 5V logic output signals to 12V voltage outputs.



High Voltage HVPA Block Diagram (n=0~3)



High Voltage HVPB Block Diagram (n=0~3)

Note: HVPAn_IN/HVPBn_IN: The driven signal of HVPAn/HVPBn input.

High Voltage Output Functional Description

The HVPxEN register is used to control the HVPxn pin output/floating status. The HVPSx register is used to select the HVPSxn output function. The HVPSAn output function can be chosen to be controlled by the HVPCx register or sourced from the internal PWM1 OUT1H output, PWM1 OUT1L output, PWM0 OUT0H output or PWM0 OUT0L output, while The HVPSBn output function can be chosen to be controlled by the HVPCx register or connected to the HVO_VSS pin. If the HVPxEN register is currently setup as output mode, it can check whether the external circuit is abnormal.

High Voltage Output Registers Description

Overall operation of High Voltage Output ports is controlled using a series of registers. Each high voltage HVPx port has a set of identical registers, HVPxEN, HVPCx and HVPSx.

Register Name	Bit							
	7	6	5	4	3	2	1	0
HVPAEN	—	—	—	—	HVPA3EN	HVPA2EN	HVPA1EN	HVPA0EN
HVPCA	—	—	—	—	HVPCA3	HVPCA2	HVPCA1	HVPCA0
HVPSA	—	—	—	—	HVPSA3	HVPSA2	HVPSA1	HVPSA0
HVPBEN	—	—	—	—	HVPB3EN	HVPB2EN	HVPB1EN	HVPB0EN
HVPCB	—	—	—	—	HVPCB3	HVPCB2	HVPCB1	HVPCB0
HVPSB	—	—	—	—	HVPSB3	HVPSB2	HVPSB1	HVPSB0

High Voltage Output Register List

• HVPxEN Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	HVPx3EN	HVPx2EN	HVPx1EN	HVPx0EN
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

HVPxnEN: HVO Port x pin type selection
0: Floating
1: Output
The HVPxnEN bit is used to control the pin type selection. Here the "x" can be A or B.

• HVPCx Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	HVPCx3	HVPCx2	HVPCx1	HVPCx0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

HVPCxn: HVO Port x pin output control bit
0: Output HVO_VSS
1: Output HVO_VCC
The HVPCxn bit is used to control the pin output state. Here the "x" can be A or B.

- **HVPSA Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	HVPSA3	HVPSA2	HVPSA1	HVPSA0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4 Unimplemented, read as "0"

Bit 3 **HVPSA3**: HVO HVPSA3 output function selection
0: Control by HVPCA3
1: From PWM1 OUT1L output

Bit 2 **HVPSA2**: HVO HVPSA2 output function selection
0: Control by HVPCA2
1: From PWM1 OUT1H output

Bit 1 **HVPSA1**: HVO HVPSA1 output function selection
0: Control by HVPCA1
1: From PWM0 OUT0L output

Bit 0 **HVPSA0**: HVO HVPSA0 output function selection
0: Control by HVPCA0
1: From PWM0 OUT0H output

- **HVPSB Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	HVPSB3	HVPSB2	HVPSB1	HVPSB0
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 **HVPSB3~HVPSB0**: HVO HVPSB3~HVPSB0 output function selection
0: Control by HVPCBn
1: Connect to the HVO_VSS pin

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 Standard and Periodic TM sections.

Introduction

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

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

TM Function Summary

STM	PTM
16-bit STM	10-bit PTM

TM Name/Type Reference

TM Operation

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

TM Clock Source

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

TM Interrupts

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

TM External Pins

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

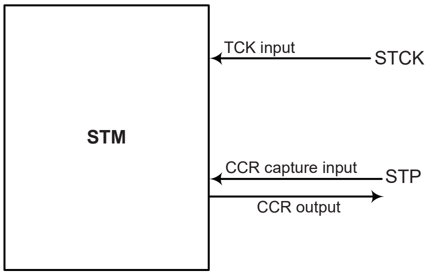
Another pin xTP, which also can be the xTM input pin, is the capture input pin whose active edge can be a rising edge, a falling edge or both rising and falling edges. The active edge transition type is selected using the xTIO1~xTIO0 bits in the xTMC1 register. For the PTM, there is another capture input,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

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

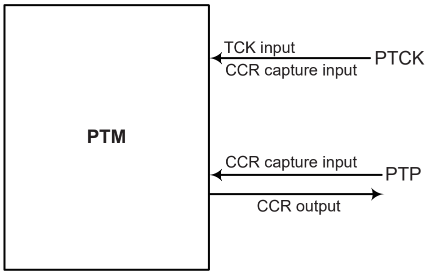
As the TM input and output pins are pin-shared with other functions, the TM input and output function must first be setup using relevant pin-shared function selection register described in the Pin-shared Function section. The details of the pin-shared function selection are described in the pin-shared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

STM		PTM	
Input	Output	Input	Output
STCK, STP	STP	PTCK, PTP	PTP

TM External Pins



STM Function Pin Block Diagram

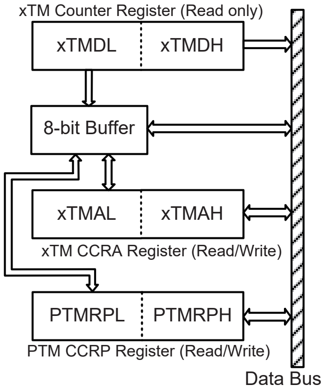


PTM Function Pin Block Diagram

Programming Considerations

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

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

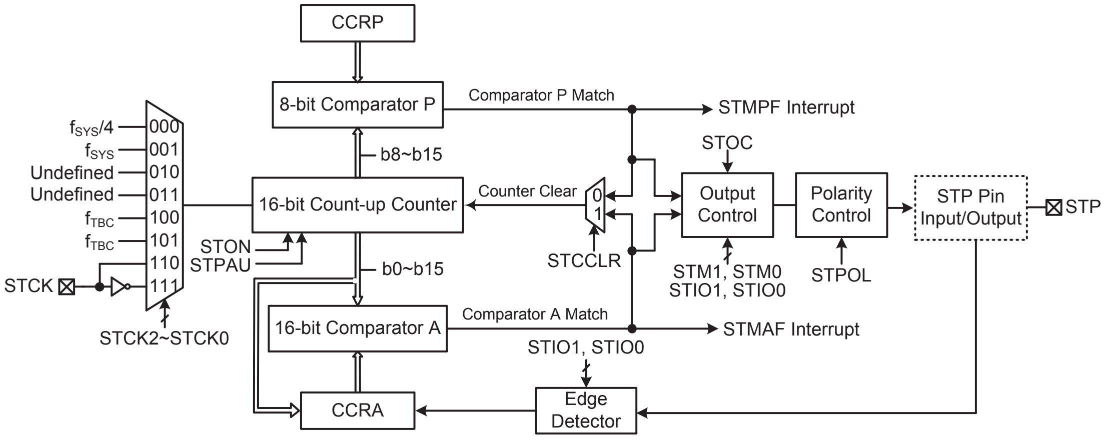

xTM Counter Register (Read only)
xTMDL
xTMDH
8-bit Buffer
xTMAL
xTMAH
xTM CCRA Register (Read/Write)
PTMRPL
PTMRPH
PTM CCRP Register (Read/Write)
Data Bus

The following steps show the read and write procedures:

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

Standard Type TM – STM

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



Standard Type TM Block Diagram

Standard TM Operation

The size of Standard TM is 16-bit wide and its core is a 16-bit count-up counter which is driven by a user selectable internal or external clock source. There are also two internal comparators with the names, Comparator A and Comparator P. These comparators will compare the value in the counter with CCRP and CCRA registers. The CCRP comparator is 8-bit wide whose value is compared the with highest 8 bits in the counter while the CCRA is the sixteen bits and therefore compares all counter bits.

The only way of changing the value of the 16-bit counter using the application program, is to clear the counter by changing the STON bit from low to high. The counter will also be cleared automatically by a counter overflow or a compare match with one of its associated comparators. When these conditions occur, a STM interrupt signal will also usually be generated. The Standard Type TM can operate in a number of different operational modes, can be driven by different clock sources including an input pin and can also control the output pins. All operating setup conditions are selected using relevant internal registers.

Standard Type TM Register Description

Overall operation of the Standard TM is controlled using a series of registers. A read only register pair exists to store the internal counter 16-bit value, while a read/write register pair exists to store the internal 16-bit CCRA value. The STMRP register is used to store the 8-bit 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
STMC0	STPAU	STCK2	STCK1	STCK0	STON	—	—	—
STMC1	STM1	STM0	STIO1	STIO0	STOC	STPOL	STDPX	STCCLR
STMDL	D7	D6	D5	D4	D3	D2	D1	D0
STMDH	D15	D14	D13	D12	D11	D10	D9	D8
STMAL	D7	D6	D5	D4	D3	D2	D1	D0
STMAH	D15	D14	D13	D12	D11	D10	D9	D8
STMRP	D7	D6	D5	D4	D3	D2	D1	D0

16-bit Standard TM Register List

• STMC0 Register

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

Bit 7 **STPAU**: STM Counter Pause control
0: Run
1: Pause

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

Bit 6~4 **STCK2~STCK0**: Select STM Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: Undefined
011: Undefined
100: f_{TBC}
101: f_{TBC}
110: STCK rising edge clock
111: STCK falling edge clock

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

Bit 3 **STON**: STM Counter On/Off control
0: Off
1: On

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

Bit 2~0 Unimplemented, read as “0”

- **STMC1 Register**

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

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

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

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

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

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

Capture Input Mode
00: Input capture at rising edge of STP
01: Input capture at falling edge of STP
10: Input capture at rising/falling edge of STP
11: Input capture disabled

Timer/Counter Mode
Unused

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

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

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

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

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

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

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

Bit 2 **STPOL**: STM STP Output polarity control
0: Non-invert
1: Invert

This bit controls the polarity of the STP output pin. When the bit is set high the STM output pin will be inverted and non-inverted when the bit is zero. It has no effect if the STM is in the Timer/Counter Mode.

Bit 1 **STDPX**: STM PWM duty/period control
0: CCRP – period; CCRA – duty
1: CCRP – duty; CCRA – period

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

Bit 0 **STCCLR**: STM Counter Clear condition selection
0: Comparator P match
1: Comparator A match

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

- **STMDL Register**

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

Bit 7~0 **D7~D0**: STM Counter Low Byte Register bit 7~bit 0
STM 16-bit Counter bit 7~bit 0

- **STMDH Register**

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

Bit 7~0 **D15~D8**: STM Counter High Byte Register bit 7~bit 0
STM 16-bit Counter bit 15~bit 8

• STMAL Register

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

Bit 7~0 **D7~D0**: STM CCRA Low Byte Register bit 7~bit 0
STM 16-bit CCRA bit 7~bit 0

• STMAH Register

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

Bit 7~0 **D15~D8**: STM CCRA High Byte Register bit 7~bit 0
STM 16-bit CCRA bit 15~bit 8

• STMRP Register

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

Bit 7~0 **D7~D0**: STM CCRP 8-bit register, compared with the STM counter bit 15~bit 8
Comparator P Match Period=
0: 65536 STM clocks
1~255: (1~255)×256 STM clocks

These eight bits are used to setup the value on the internal CCRP 8-bit register, which are then compared with the internal counter's highest eight bits. The result of this comparison can be selected to clear the internal counter if the STCCLR bit is set to zero. Setting the STCCLR bit to zero ensures that a compare match with the CCRP values will reset the internal counter. As the CCRP bits are only compared with the highest eight counter bits, the compare values exist in 256 clock cycle multiples. Clearing all eight bits to zero is in effect allowing the counter to overflow at its maximum value.

Standard Type TM Operation Modes

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

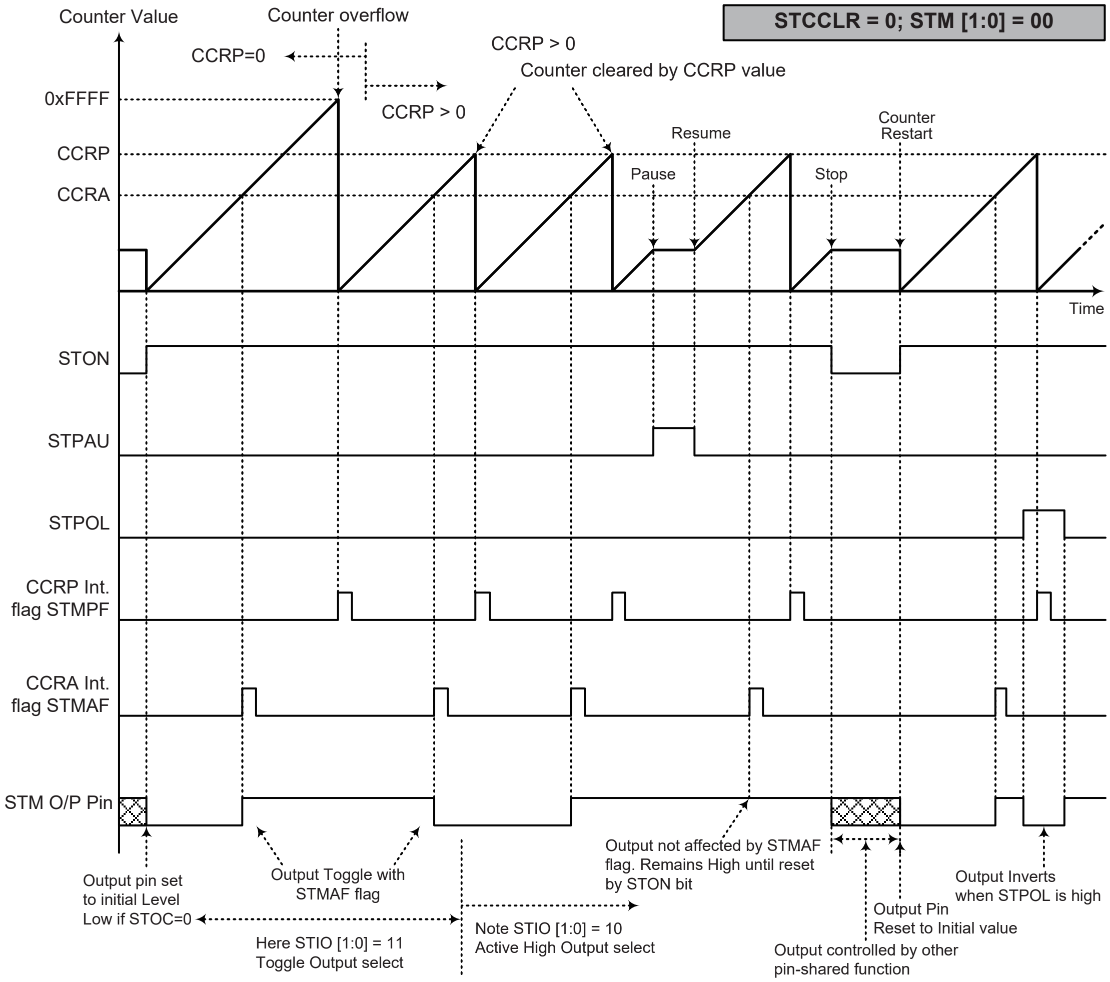
Compare Match Output Mode

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

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

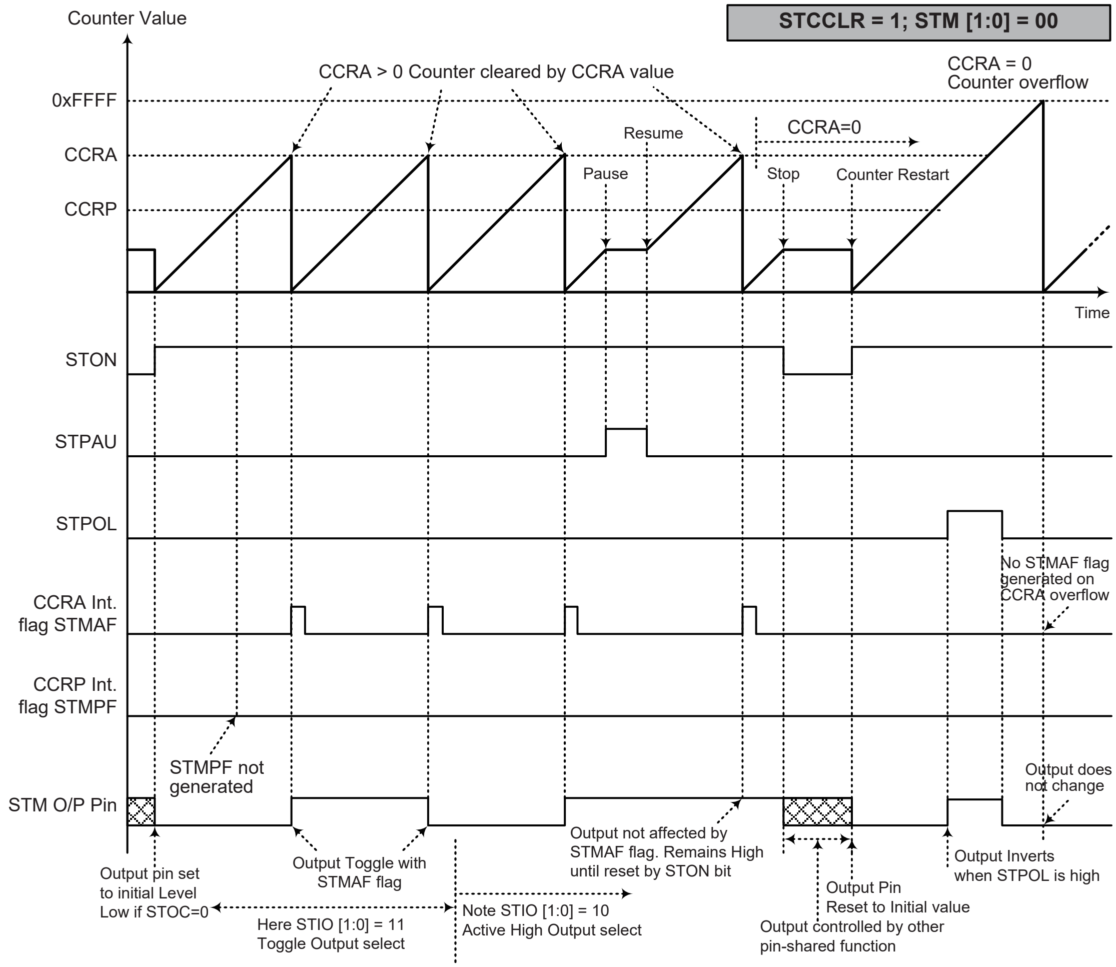
If the CCRA bits are all zero, the counter will overflow when its reaches its maximum 16-bit, FFFF Hex, value, however here the STMAF interrupt request flag will not be generated.

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


STCCLR = 0; STM [1:0] = 00
Counter Value
Counter overflow
CCRP=0
CCRP > 0
CCRP > 0
Counter cleared by CCRP value
0xFFFF
CCRP
CCRA
Resume
Counter Restart
Pause
Stop
Time
STON
STPAU
STPOL
CCRP Int. flag STMPF
CCRA Int. flag STMAF
STM O/P Pin
Output pin set to initial Level Low if STOC=0
Output Toggle with STMAF flag
Here STIO [1:0] = 11 Toggle Output select
Note STIO [1:0] = 10 Active High Output select
Output not affected by STMAF flag. Remains High until reset by STON bit
Output Pin Reset to Initial value
Output controlled by other pin-shared function
Output Inverts when STPOL is high

Compare Match Output Mode – STCCLR=0

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


STCCLR = 1; STM [1:0] = 00
Counter Value
CCRA > 0 Counter cleared by CCRA value
CCRA = 0
Counter overflow
0xFFFF
CCRA
CCRP
Resume
CCRA=0
Pause
Stop
Counter Restart
Time
STON
STPAU
STPOL
No STMAF flag generated on CCRA overflow
CCRA Int. flag STMAF
CCRP Int. flag STMPF
STMPF not generated
Output does not change
STM O/P Pin
Output not affected by STMAF flag. Remains High until reset by STON bit
Output pin set to initial Level Low if STOC=0
Output Toggle with STMAF flag
Here STIO [1:0] = 11 Toggle Output select
Note STIO [1:0] = 10 Active High Output select
Output Pin Reset to Initial value
Output controlled by other pin-shared function
Output Inverts when STPOL is high

Compare Match Output Mode – STCCLR=1

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

Timer/Counter Mode

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

PWM Output Mode

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

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

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

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

CCRP	1~255	0
Period	CCRP×256	65536
Duty	CCRA	

If f_{SYS}=16MHz, STM clock source is f_{SYS}/4, CCRP=2 and CCRA=128,

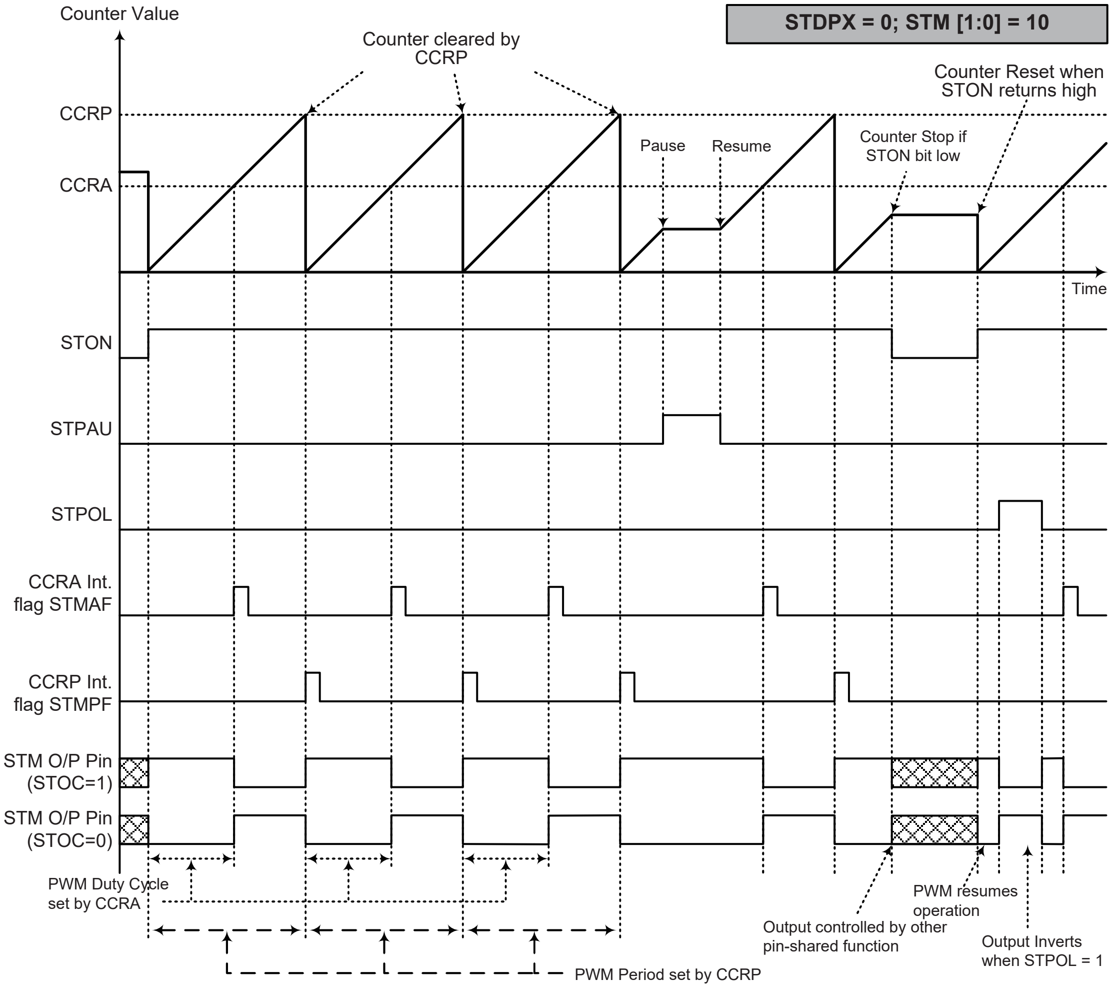
The STM PWM output frequency=(f_{SYS}/4)/(2×256)=f_{SYS}/2048=7.8125kHz, duty=128/(2×256)=25%.

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%.

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

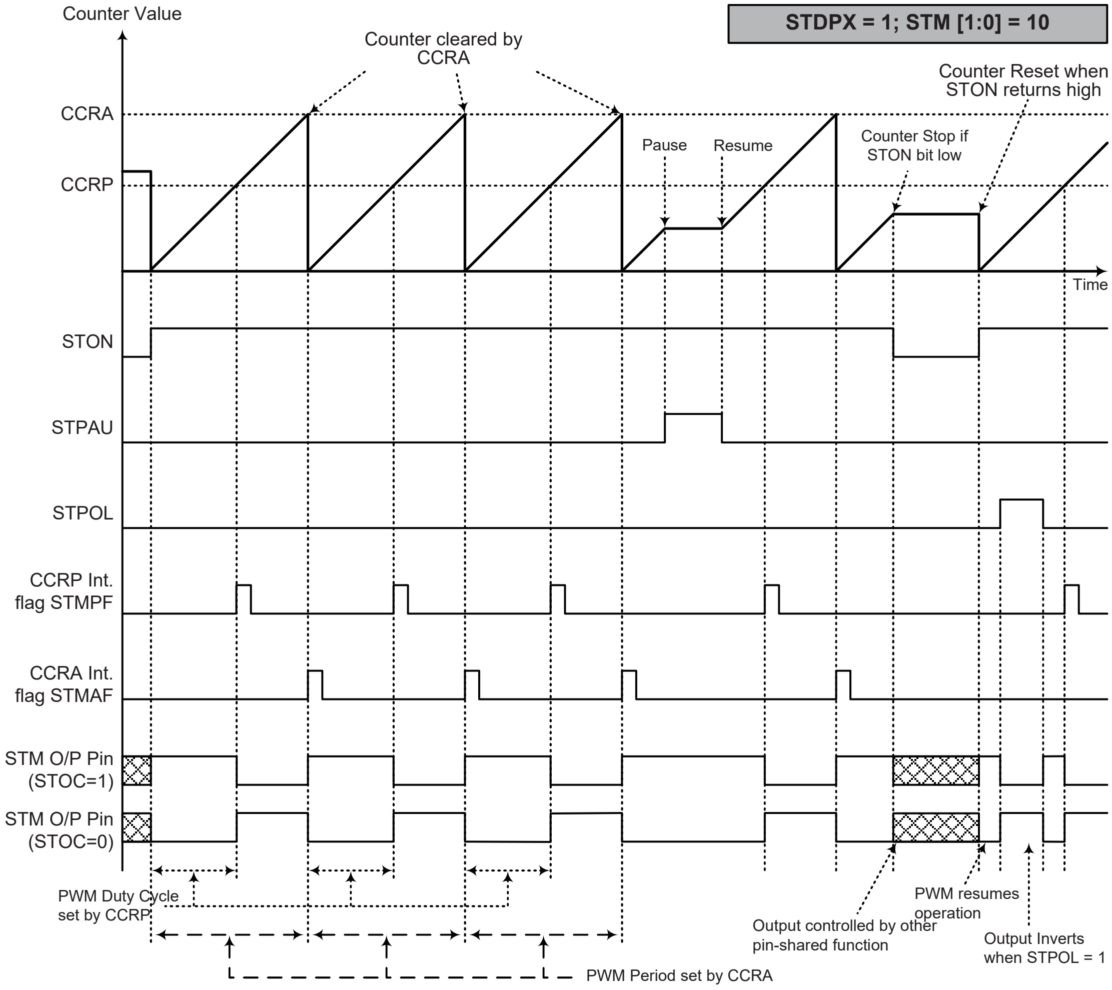
CCRP	1~255	0
Period	CCRA	
Duty	CCRP×256	65536

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



PWM Output Mode – STDPX=0

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



PWM Output Mode – STDPX=1

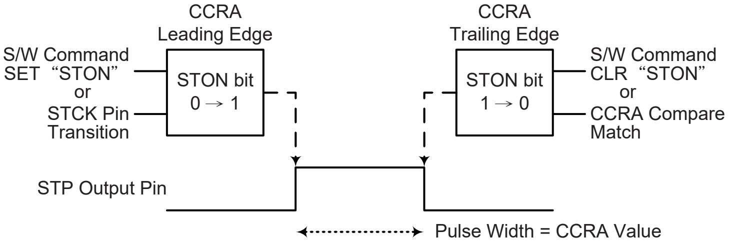
Note: 1. Here STDPX=1 – Counter cleared by CCRA
2. A counter clear sets the PWM Period
3. The internal PWM function continues even when STIO[1:0]=00 or 01
4. The STCCLR bit has no influence on PWM operation

Single Pulse Output Mode

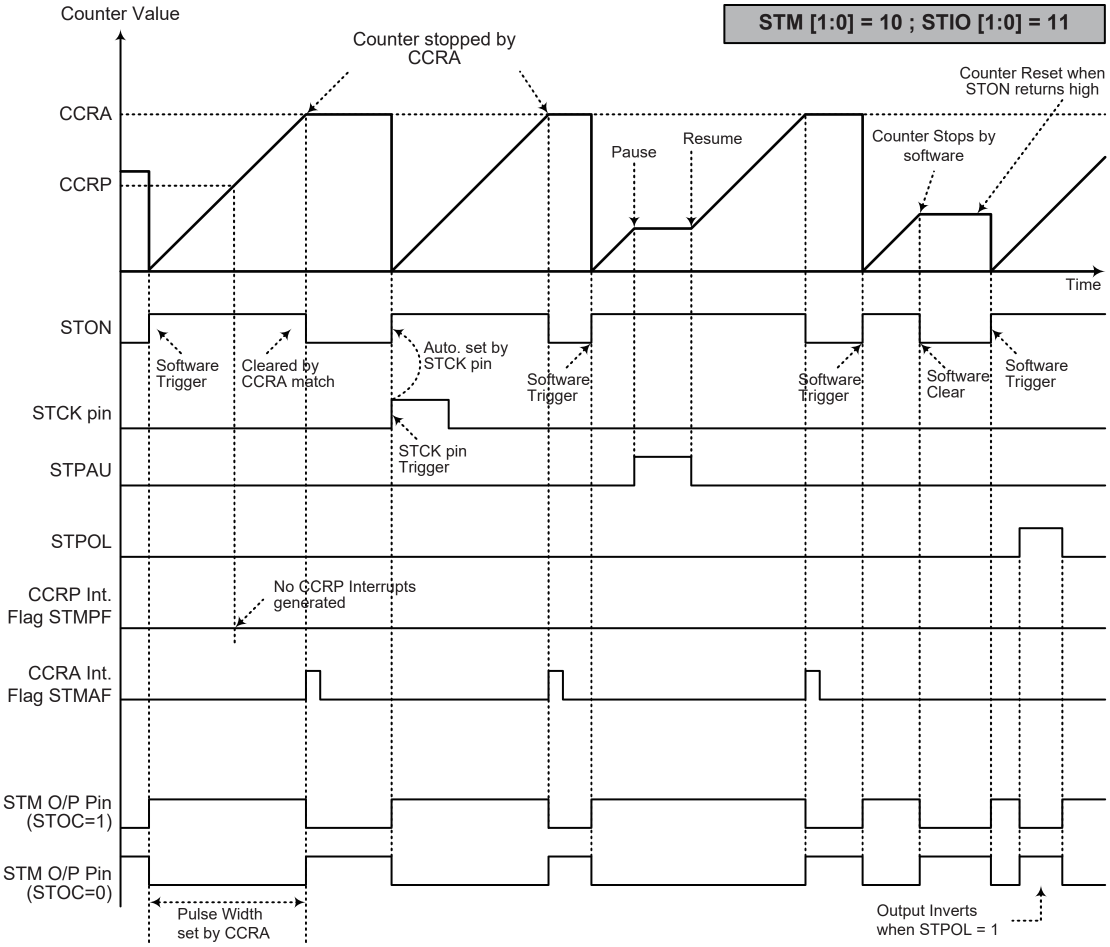
To select this mode, bits STM1 and STM0 in the STMC1 register should be set to 10 respectively and also the STIO1 and STIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the STM output pin.

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

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



Single Pulse Generation



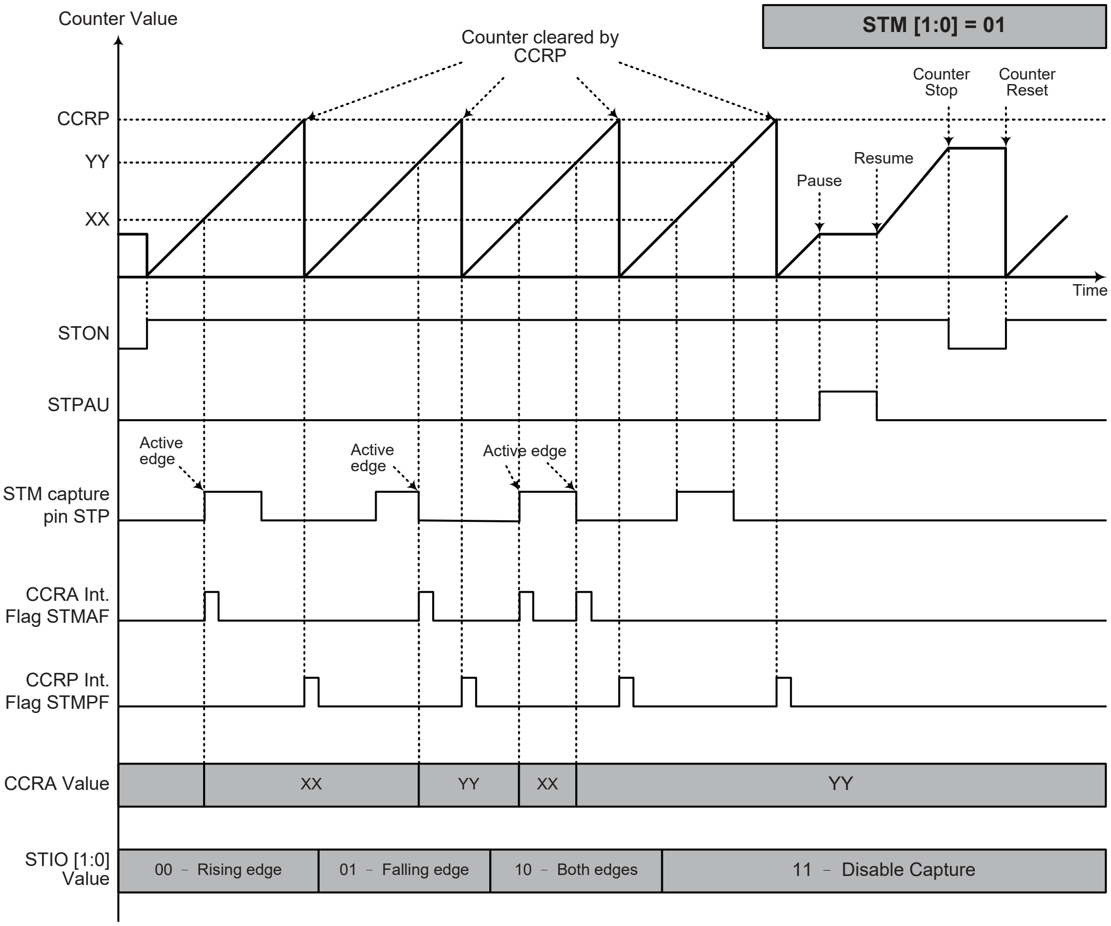
Single Pulse Output Mode

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

Capture Input Mode

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

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

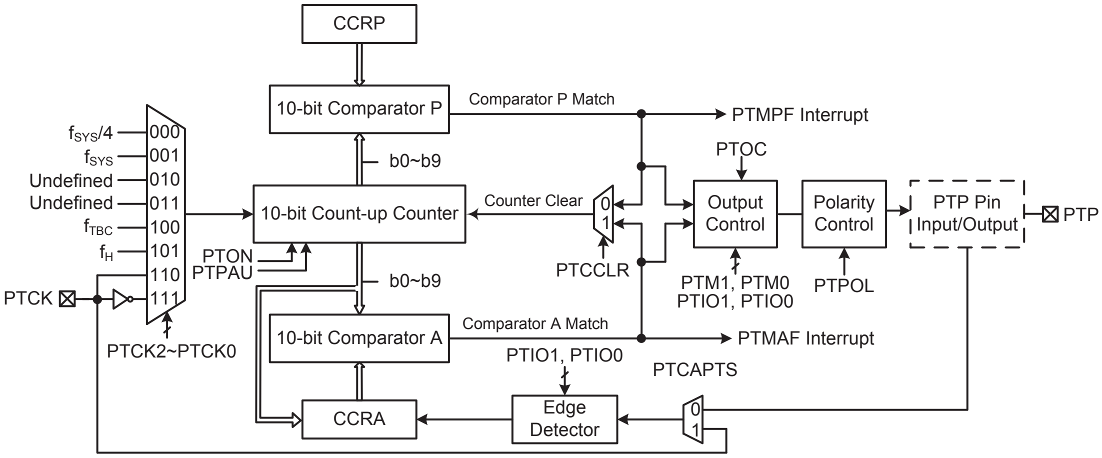


Capture Input Mode

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

Periodic Type TM – PTM

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



Periodic Type TM Block Diagram

Periodic TM Operation

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

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

Periodic Type TM Register Description

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

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

10-bit Periodic TM Register List

- **PTMC0 Register**

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

Bit 7 **PTPAU**: PTM Counter Pause Control
0: Run
1: Pause

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

Bit 6~4 **PTCK2~PTCK0**: Select PTM Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: Undefined
011: Undefined
100: f_{TBC}
101: f_H
110: PTCK rising edge clock
111: PTCK falling edge clock

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

Bit 3 **PTON**: PTM Counter On/Off Control
0: Off
1: On

This bit controls the overall on/off function of the PTM. Setting the bit high enables the counter to run, clearing the bit disables the PTM. Clearing this bit to zero will stop the counter from counting and turn off the PTM which will reduce its power consumption. When the bit changes state from low to high the internal counter value will be reset to zero, however when the bit changes from high to low, the internal counter will retain its residual value until the bit returns high again.

If the PTM is in the Compare Match Output Mode, PWM Output Mode or Single Pulse Output Mode then the PTM output pin will be reset to its initial condition, as specified by the PTOC bit, when the PTON bit changes from low to high.

Bit 2~0 Unimplemented, read as “0”

- **PTMC1 Register**

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

Bit 7~6 **PTM1~PTM0**: Select PTM Operating Mode
00: Compare Match Output Mode
01: Capture Input Mode
10: PWM Output Mode or Single Pulse Output Mode
11: Timer/Counter Mode

These bits setup the required operating mode for the PTM. To ensure reliable operation the PTM should be switched off before any changes are made to the PTM1 and PTM0 bits. In the Timer/Counter Mode, the PTM output pin state is undefined.

Bit 5~4 **PTIO1~PTIO0**: Select PTM external pin (PTP or PTCK) function

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

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

Capture Input Mode
00: Input capture at rising edge of PTP or PTCK
01: Input capture at falling edge of PTP or PTCK
10: Input capture at falling/rising edge of PTP or PTCK
11: Input capture disabled

Timer/Counter Mode
Unused

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

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

In the PWM Output Mode, the PTIO1 and PTIO0 bits determine how the PTM 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 PTIO1 and PTIO0 bits only after the TM has been switched off. Unpredictable PWM outputs will occur if the PTIO1 and PTIO0 bits are changed when the PTM is running.

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

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

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

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

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

This bit controls the polarity of the PTP output pin. When the bit is set high the PTM output pin will be inverted and non-inverted when the bit is zero. It has no effect if the PTM is in the Timer/Counter Mode.

Bit 1 **PTCAPTS**: PTM Capture Trigger Source Selection
0: From PTP pin
1: From PTCK pin

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

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

• PTMDL Register

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

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

• PTMDH Register

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

Bit 7~2 Unimplemented, read as "0"

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

• PTMAL Register

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

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

• PTMAH Register

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

Bit 7~2 Unimplemented, read as “0”

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

• PTMRPL Register

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

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

• PTMRPH Register

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

Bit 7~2 Unimplemented, read as “0”

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

Periodic Type TM Operating Modes

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

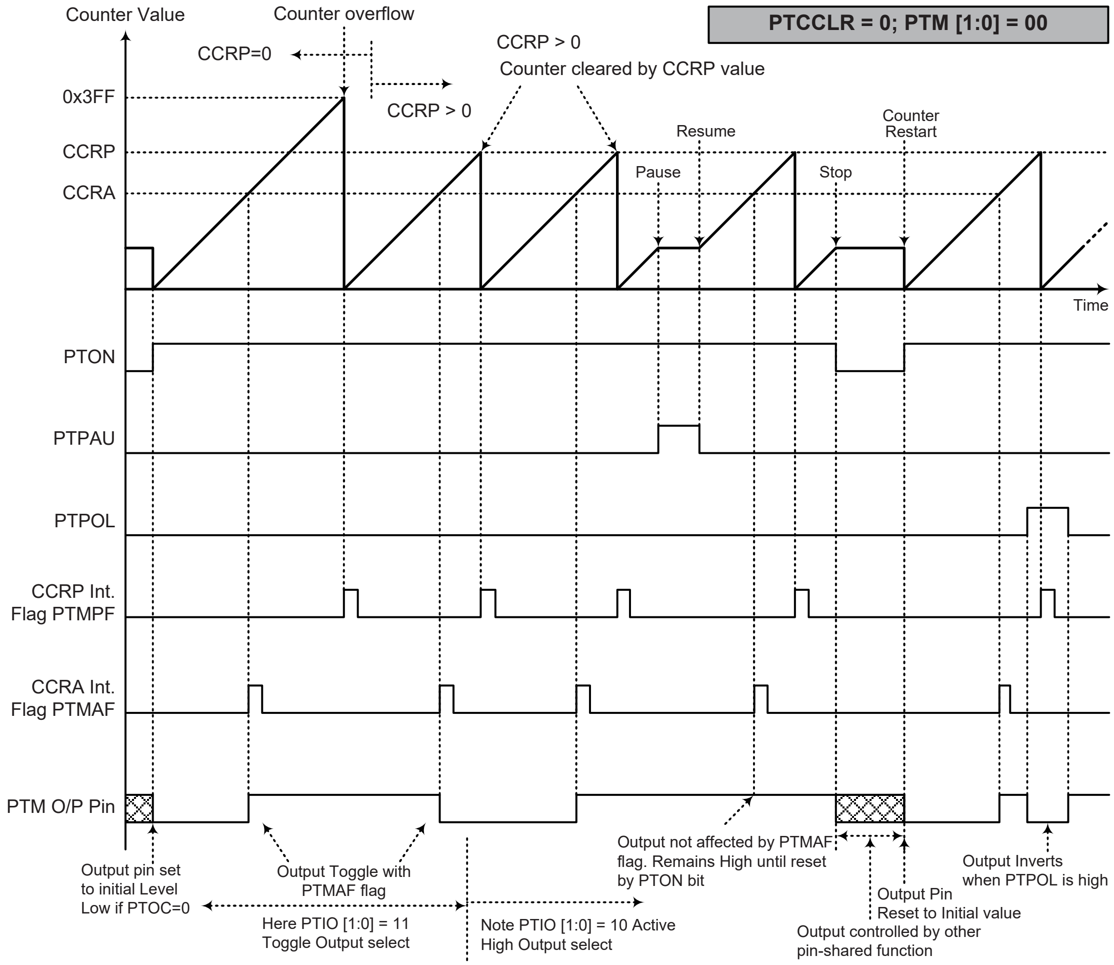
Compare Match Output Mode

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

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

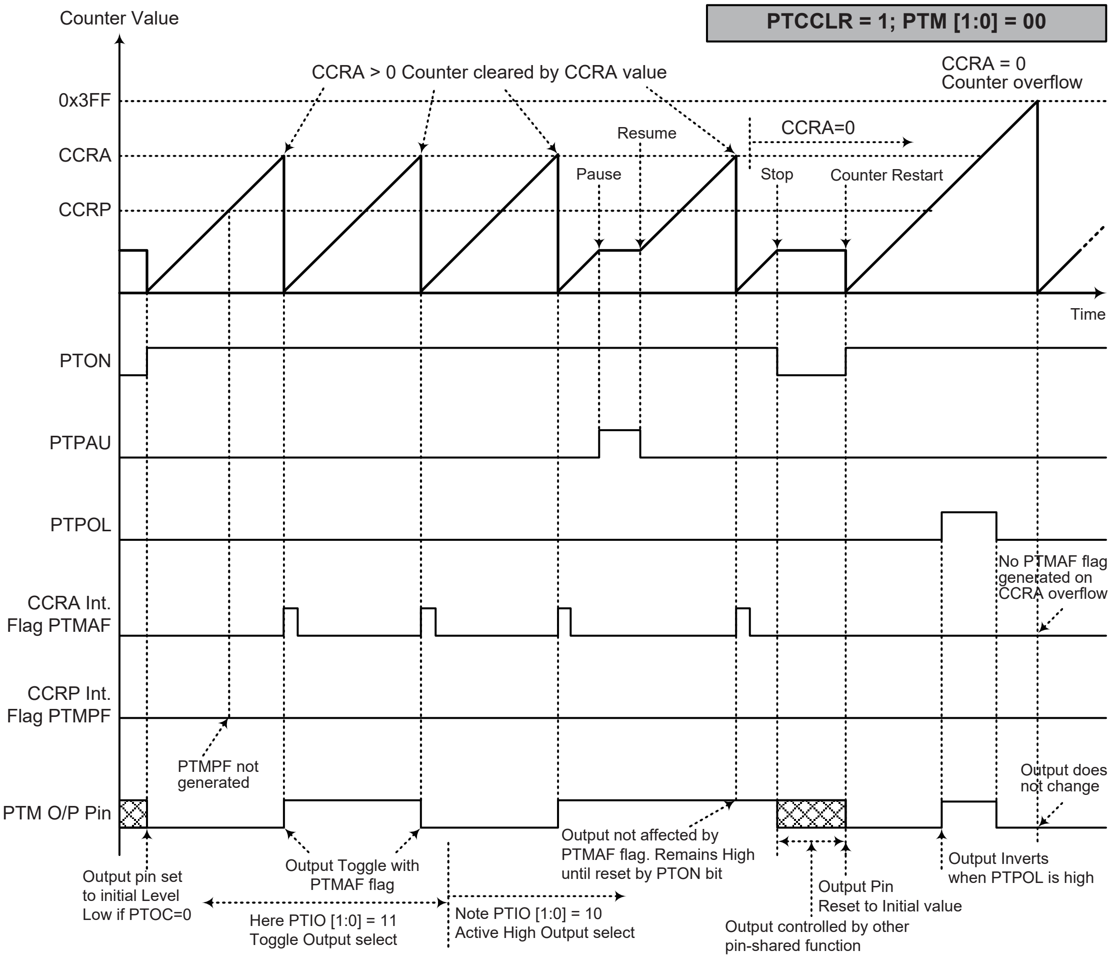
If the CCRA bits are all zero, the counter will overflow when it reaches its maximum 10-bit, 3FF Hex, value, however here the PTMAF interrupt request flag will not be generated.

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


PTCCLR = 0; PTM [1:0] = 00
Counter Value
Counter overflow
CCRP=0
CCRP > 0
CCRP > 0
Counter cleared by CCRP value
0x3FF
CCRP
CCRA
Resume
Counter Restart
Pause
Stop
Time
PTON
PTPAU
PTPOL
CCRP Int. Flag PTMPF
CCRA Int. Flag PTMAF
PTM O/P Pin
Output pin set to initial Level Low if PTOC=0
Output Toggle with PTMAF flag
Here PTIO [1:0] = 11 Toggle Output select
Note PTIO [1:0] = 10 Active High Output select
Output not affected by PTMAF flag. Remains High until reset by PTON bit
Output Pin Reset to Initial value
Output controlled by other pin-shared function
Output Inverts when PTPOL is high

Compare Match Output Mode – PTCCLR=0

Note: 1. With PTCCLR=0 a Comparator P match will clear the counter
2. The PTM output pin is controlled only by the PTMAF flag
3. The output pin is reset to its initial state by a PTON bit rising edge


PTCCLR = 1; PTM [1:0] = 00
Counter Value
CCRA > 0 Counter cleared by CCRA value
CCRA = 0
Counter overflow
0x3FF
CCRA
CCRP
Resume
Pause
CCRA=0
Stop
Counter Restart
Time
PTON
PTPAU
PTPOL
CCRA Int.
Flag PTMAF
CCRP Int.
Flag PTMPF
No PTMAF flag generated on CCRA overflow
PTMPF not generated
Output does not change
PTM O/P Pin
Output not affected by PTMAF flag. Remains High until reset by PTON bit
Output pin set to initial Level Low if PTOC=0
Output Toggle with PTMAF flag
Here PTIO [1:0] = 11 Toggle Output select
Note PTIO [1:0] = 10 Active High Output select
Output Pin Reset to Initial value
Output controlled by other pin-shared function
Output Inverts when PTPOL is high

Compare Match Output Mode – PTCCLR=1

Note: 1. With PTCCLR=1 a Comparator A match will clear the counter
2. The PTM output pin is controlled only by the PTMAF flag
3. The output pin is reset to its initial state by a PTON bit rising edge
4. A PTMPF flag is not generated when PTCCLR=1

Timer/Counter Mode

To select this mode, bits PTM1 and PTM0 in the PTMC1 register should be set to 11 respectively. The Timer/Counter Mode operates in an identical way to the Compare Match Output Mode generating the same interrupt flags. The exception is that in the Timer/Counter Mode the 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 PTM1 and PTM0 in the PTMC1 register should be set to 10 respectively and also the PTIO1 and PTIO0 bits should be set to 10 respectively. The PWM function within the PTM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the PTM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

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

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

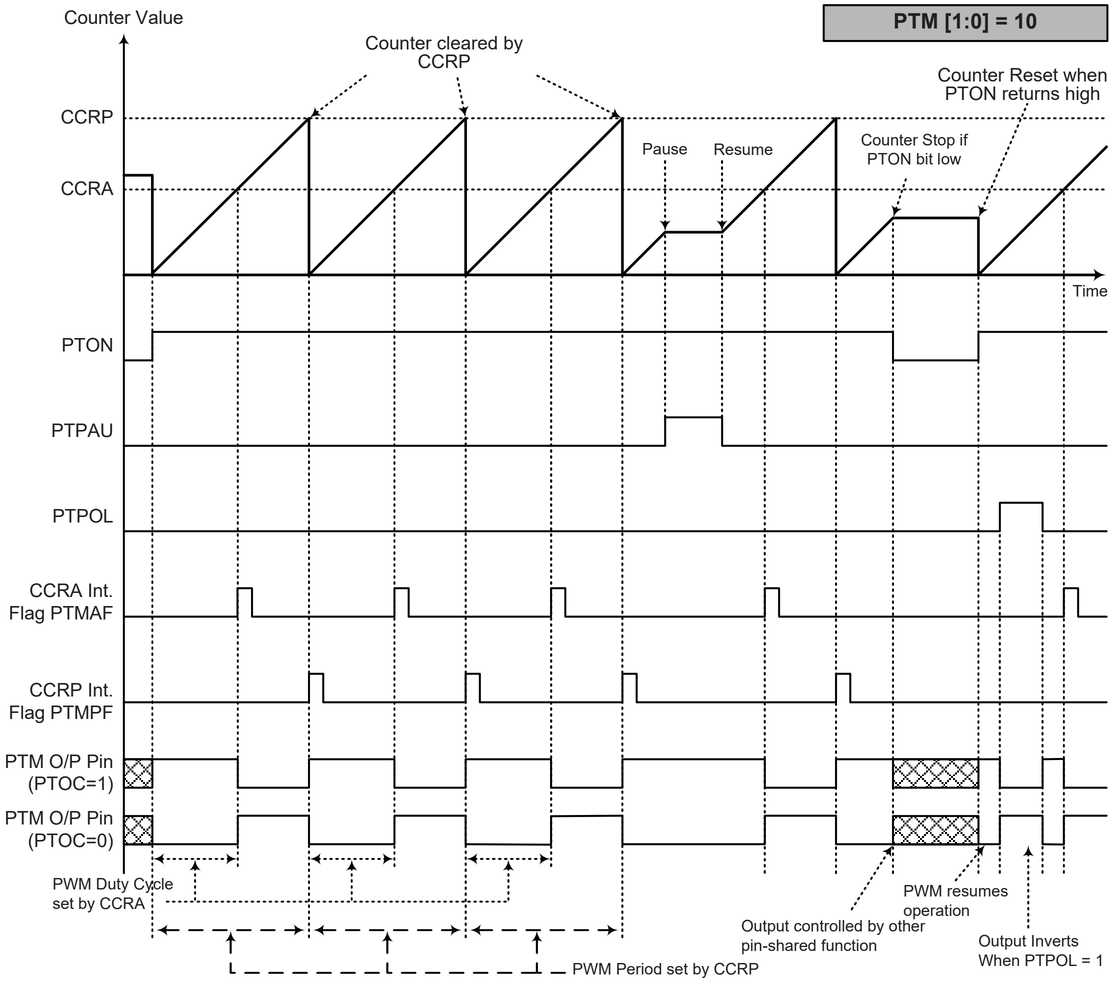
- **10-bit PTM, PWM Output Mode, Edge-aligned Mode**

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

If f_{SYS}=16MHz, PTM clock source select f_{SYS}/4, CCRP=512 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 Output Mode

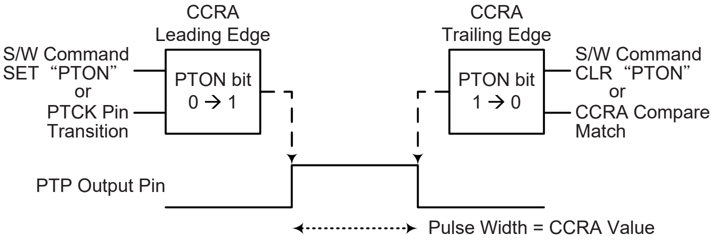
Note: 1. Counter cleared by CCRP
2. A counter clear sets the PWM Period
3. The internal PWM function continues running even when PTIO[1:0]=00 or 01
4. The PTCCLR bit has no influence on PWM operation

Single Pulse Output Mode

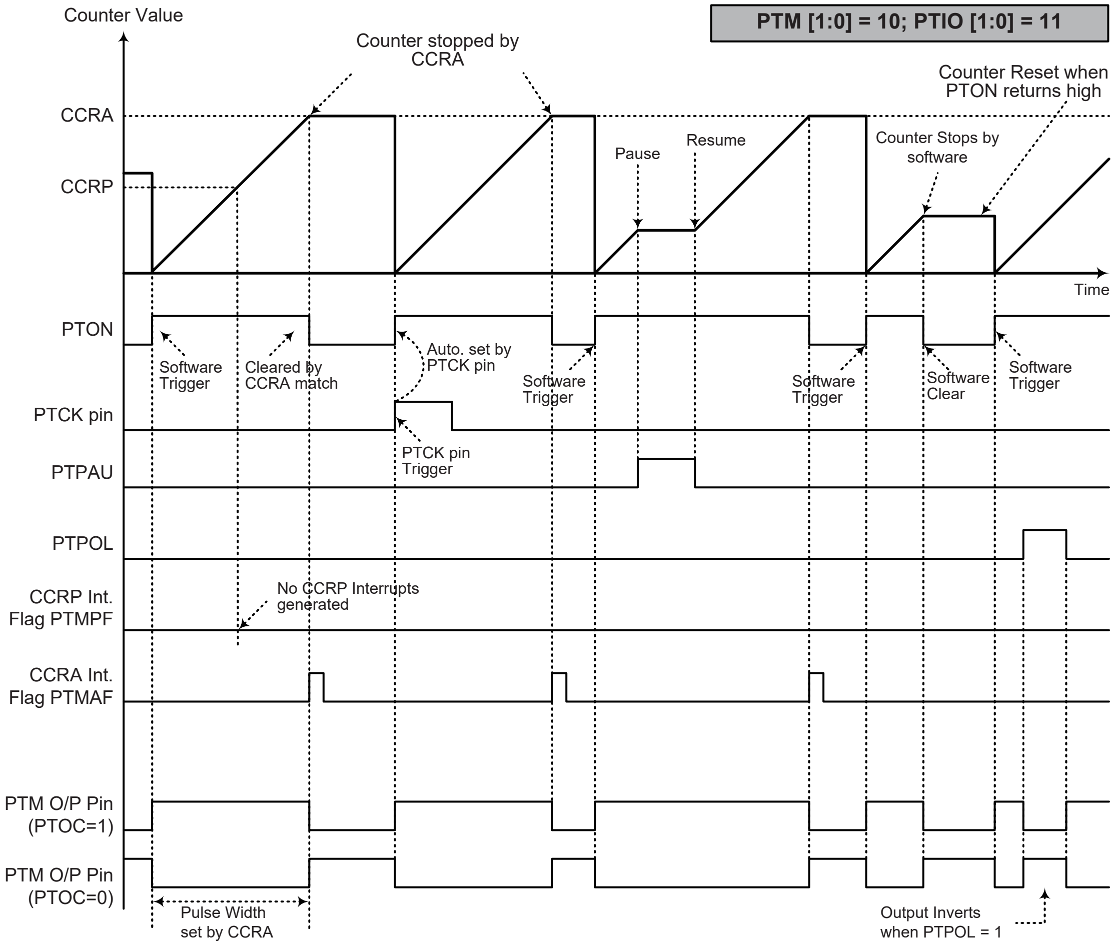
To select this mode, bits PTM1 and PTM0 in the PTMC1 register should be set to 10 respectively and also the PTIO1 and PTIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the PTM output pin.

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

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



Single Pulse Generation



Single Pulse Output Mode

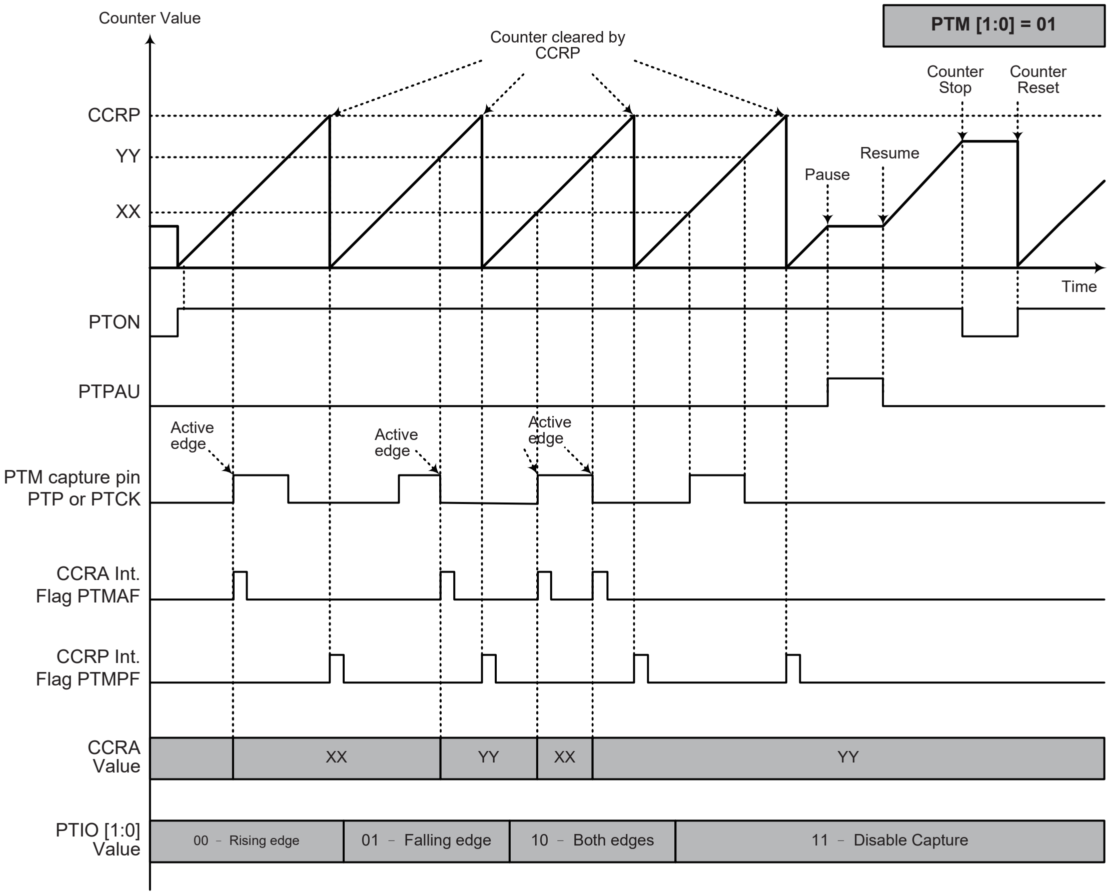
Note: 1. Counter stopped by CCRA
2. CCRP is not used
3. The pulse is triggered by the PTCK pin or by setting the PTON bit high
4. A PTCK pin active edge will automatically set the PTON bit high
5. In the Single Pulse Output Mode, PTIO[1:0] must be set to “11” and cannot be changed

Capture Input Mode

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

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

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



Capture Input Mode

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

Analog to Digital Converter

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

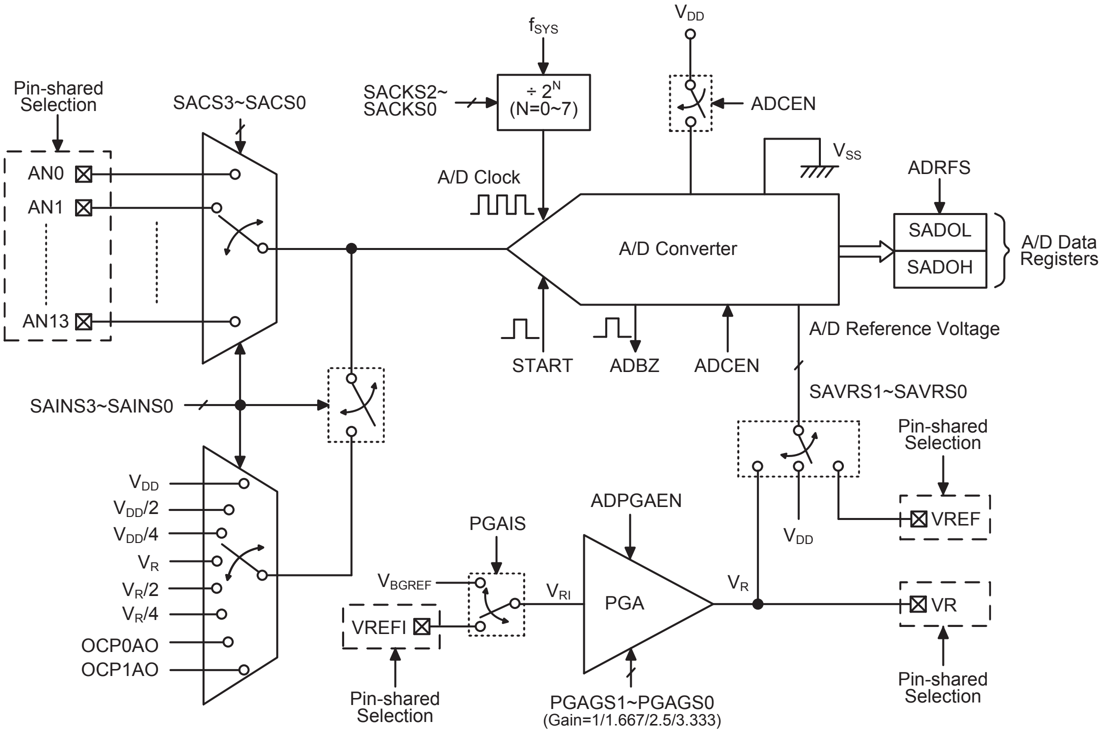
A/D Converter Overview

This device contains a multi-channel analog to digital converter which can directly interface to external analog signals, such as that from sensors or other control signals and convert these signals directly into a 12-bit digital value. It also can convert the internal signals of the Over Current Protection 0 or Over Current Protection 1 OPA output signal into a 12-bit digital value.

The external or internal analog signal to be converted is determined by the SAINS3~SAINS0 bits together with the SACS3~SACS0 bits. When the external analog signal is to be converted, the corresponding pin-shared control bits should first be properly configured and then desired external channel input should be selected using the SAINS3~SAINS0 and SACS3~SACS0 bits. Note that when the internal analog signal is to be converted using the SAINS bit field, the external channel analog input will be automatically be switched off. More detailed information about the A/D input signal is described in the "A/D Converter Control Registers" and "A/D Converter Input Signals" sections respectively.

External Input Channels	Internal Signal	A/D Channel Select Bits
14: AN0~AN13	8: V_{DD}, $V_{DD}/2$, $V_{DD}/4$, V_R, $V_R/2$, $V_R/4$, OCP0AO, OCP1AO	SAINS3~SAINS0, SACS3~SACS0

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



Note: The A/D Converter PGA output V_R can drive the OCPn, OVPn and UVPn DAC.

A/D Converter Structure

A/D Converter Register Description

Overall operation of the A/D converter is controlled using six registers. A read only register pair exists to store the A/D converter data 12-bit single value. The remaining four registers are control registers which setup the operating and control function of the A/D converter. The register VBGRC can be used to control the Bandgap reference voltage.

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

A/D Converter Register List

A/D Converter Data Registers – SADOL, SADOH

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

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

A/D Converter Data Registers

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

To control the function and operation of the A/D converter, three control registers known as SADC0~SADC2 are provided. These 8-bit registers define functions such as the selection of which analog channel is connected to the internal A/D converter, the digitised data format, the A/D clock source as well as controlling the start function and monitoring the A/D converter busy status. As the device contains only one actual analog to digital converter hardware circuit, each of the external or internal analog signal inputs must be routed to the converter. The SACS3~SACS0 bits in the SADC0 register are used to determine which external channel input is selected to be converted. The SAINS3~SAINS0 bits in the SADC1 register are used to determine that the analog signal to be converted comes from the internal analog signal or external analog channel input.

The relevant pin-shared function selection bits determine which pins on I/O Ports are used as analog inputs for the A/D converter input and which pins are not to be used as the A/D converter input. When the pin is selected to be an A/D input, its original function, whether it is an I/O or other pin-shared function will be removed. In addition, any internal pull-high resistor connected to the pin will be automatically removed if the pin is selected to be an A/D converter input. The A/D converter also contains programmable gain amplifier, PGA, to generate the A/D converter internal reference voltage. The overall operation of the PGA is controlled using the SADC2 register.

An additional register named VBGRC is provided. The VBGREN bit in the VBGRC register is used for Bandgap reference voltage control.

- **SADC0 Register**

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

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

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

Bit 5 **ADCEN**: A/D converter function enable control
0: Disable
1: Enable

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

Bit 4 **ADRFS**: A/D converter data format select
0: A/D converter data format → SADOH=D[11:4]; SADOL=D[3:0]
1: A/D converter data format → SADOH=D[11:8]; SADOL=D[7:0]

This bit controls the format of the 12-bit converted A/D value in the two A/D data registers. Details are provided in the A/D data register section.

Bit 3~0 **SACS3~SACS0**: A/D converter external analog channel input select
0000: AN0
0001: AN1
:
:
1100: AN12
1101: AN13
1110~1111: Non-existed channel, the input will be floating if selected

- **SADC1 Register**

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

Bit 7~4 **SAINS3~SAINS0**: A/D converter input signal select
0000: External input – External analog channel input
0001: Internal input – Internal A/D converter power supply voltage V_{DD}
0010: Internal input – Internal A/D converter power supply voltage $V_{DD}/2$
0011: Internal input – Internal A/D converter power supply voltage $V_{DD}/4$
0100: External input – External analog channel input
0101: Internal input – Internal A/D converter PGA output voltage V_R
0110: Internal input – Internal A/D converter PGA output voltage $V_R/2$
0111: Internal input – Internal A/D converter PGA output voltage $V_R/4$
1000: Internal input – OCP0 OPA output, OCP0AO
1001: Internal input – OCP1 OPA output, OCP1AO
1010~1011: Reserved, connected to ground
1100~1111: External input – External analog channel input

When the internal analog signal and the external signal are selected to be converted simultaneously, the external channel input signal will automatically be switched off regardless of the SACS3~SACS0 bit field value.

Bit 3 Unimplemented, read as “0”

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

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

- **SADC2 Register**

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

Bit 7 **ADPGAEN**: PGA enable/disable control
0: Disable
1: Enable

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

Bit 6~5 Unimplemented, read as "0"

Bit 4 **PGAIS**: PGA input (V_{RI}) selection
0: External VREFI pin
1: Internal independent reference voltage, V_{BGREF}

When the external voltage on VREFI pin and the internal independent reference voltage V_{BGREF} are selected as the PGA input simultaneously, the hardware will only choose the internal voltage V_{BGREF} as the PGA input.

Bit 3~2 **SAVRS1~SAVRS0**: A/D converter reference voltage select
00: Internal A/D converter power, V_{DD}
01: External VREF pin
1x: Internal PGA output voltage, V_R

These bits are used to select the A/D converter reference voltage. When the internal A/D converter power or the internal PGA output voltage and the external input voltage on VREF pin are selected as the reference voltage simultaneously, the hardware will only choose the internal reference voltage as the A/D converter reference voltage.

Bit 1~0 **PGAGS1~PGAGS0**: PGA gain select
00: Gain=1
01: Gain=1.667, V_R=2V for V_{RI}=V_{BGREF} (PGAIS=1)
10: Gain=2.5, V_R=3V for V_{RI}=V_{BGREF} (PGAIS=1)
11: Gain=3.333, V_R=4V for V_{RI}=V_{BGREF} (PGAIS=1)

- **VBGRC Register**

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

Bit 7~1 Unimplemented, read as "0"

Bit 0 **VBGREN**: Independent reference bandgap enable/disable control
0: Disable
1: Enable

When the VBGREN bit is cleared to zero, the V_{BGREF} is in a high impedance state.

A/D Converter Operation

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

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

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

f_{SYS}	A/D Clock Period (t_{ADCK})							
	SACKS[2:0] =000 (f_{SYS})	SACKS[2:0] =001 (f_{SYS}/2)	SACKS[2:0] =010 (f_{SYS}/4)	SACKS[2:0] =011 (f_{SYS}/8)	SACKS[2:0] =100 (f_{SYS}/16)	SACKS[2:0] =101 (f_{SYS}/32)	SACKS[2:0] =110 (f_{SYS}/64)	SACKS[2:0] =111 (f_{SYS}/128)
1MHz	1μs	2μs	4μs	8μs	16μs *	32μs *	64μs *	128μs *
2MHz	500ns	1μs	2μs	4μs	8μs	16μs *	32μs *	64μs *
4MHz	250ns *	500ns	1μs	2μs	4μs	8μs	16μs *	32μs *
8MHz	125ns *	250ns *	500ns	1μs	2μs	4μs	8μs	16μs *
12MHz	83ns *	167ns *	333ns *	667ns	1.33μs	2.67μs	5.33μs	10.67μs *
16MHz	62.5ns *	125ns *	250ns *	500ns	1μs	2μs	4μs	8μs

A/D Clock Period Examples

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

A/D Converter Reference Voltage

The reference voltage supply to the A/D converter can be supplied from the positive power supply, V_{DD}, or from an external reference source supplied on pin VREF, or from the internal PGA output voltage, V_R. The desired selection is made using the SAVRS1 and SAVRS0 bits. When the SAVRS bit field is set to “00”, the A/D converter reference voltage will come from the V_{DD}. If the SAVRS bit field is set to “01”, the A/D converter reference voltage will come from the VREF pin. Otherwise, the A/D converter reference voltage will come from the PGA output, V_R. As the VREF pin is pin-shared with other functions, when the VREF pin is selected as the reference voltage supply pin, the VREF pin-shared function control bits should be properly configured to disable other pin functions. In addition, if the program selects an external reference voltage on VREF pin and the internal reference voltage V_{DD} or V_R as the A/D converter reference voltage, then the hardware will only choose the internal reference voltage as the A/D converter reference voltage input. The analog input values must not be allowed to exceed the value of the selected reference voltage, V_{REF}.

The A/D converter also has a VREFI pin which is one of PGA inputs for A/D converter reference. To select this PGA input signal, the PGAIS bit in the SADC2 register must be cleared to zero and the revelent pin-shared control bits should be properly configured. However, the PGA input can be aslo supplied from the internal independent reference voltage, V_{BGREF}. If the application program selects the external voltage on the VREFI pin and an internal voltage V_{BGREF} as PGA input simultaneously, then the hardware will only choose the internal voltage V_{BGREF} as PGA input.

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

A/D Converter Reference Voltage Selection

A/D Converter Input Signals

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

If the SAINS3~SAINS0 bits are set to “0000”, “0100” or “1110~1111”, the external analog channel input is selected to be converted and the SACS3~SACS0 bits can determine which actual external channel is selected to be converted. If the SAINS3~SAINS0 bits are set to “0001~0011”, the V_{DD} voltage with a specific ratio of 1, 1/2 or 1/4 is selected to be converted. If the SAINS3~SAINS0 bits are set to “0101~0111”, the PGA output voltage with a specific ratio of 1, 1/2 or 1/4 is selected to be converted. If the SAINS3~SAINS0 bits are set to “1000”, the OCP0 OPA output, OCP0AO is selected to be converted. If the SAINS3~SAINS0 bits are set to “1001”, the OCP1 OPA output, OCP1AO is selected to be converted.

Note that when the programs select external signal (AN0~AN13) and internal signal (V_{DD}, $V_{DD}/2$, $V_{DD}/4$, V_R, $V_R/2$, $V_R/4$, OCP0AO or OCP1AO) as an A/D converter input signal simultaneously, then the hardware will only choose the internal signal as an A/D converter input, the external analog signal will be switched off automatically.

SAINS[3:0]	SACS[3:0]	Input Signals	Description
0000, 0100, 1100~1111	0000~1101	AN0~AN13	External pin analog input
	111x	—	Floating, no external channel is selected
0001	xxxx	V_{DD}	Internal A/D converter power supply voltage V_{DD}
0010	xxxx	$V_{DD}/2$	Internal A/D converter power supply voltage $V_{DD}/2$
0011	xxxx	$V_{DD}/4$	Internal A/D converter power supply voltage $V_{DD}/4$
0101	xxxx	V_R	Internal A/D converter PGA output voltage V_R
0110	xxxx	$V_R/2$	Internal A/D converter PGA output voltage $V_R/2$
0111	xxxx	$V_R/4$	Internal A/D converter PGA output voltage $V_R/4$
1000	xxxx	OCP0AO	OCP0 OPA output
1001	xxxx	OCP1AO	OCP1 OPA output
101x	xxxx	—	Reserved, connected to ground.

"x": Don't care

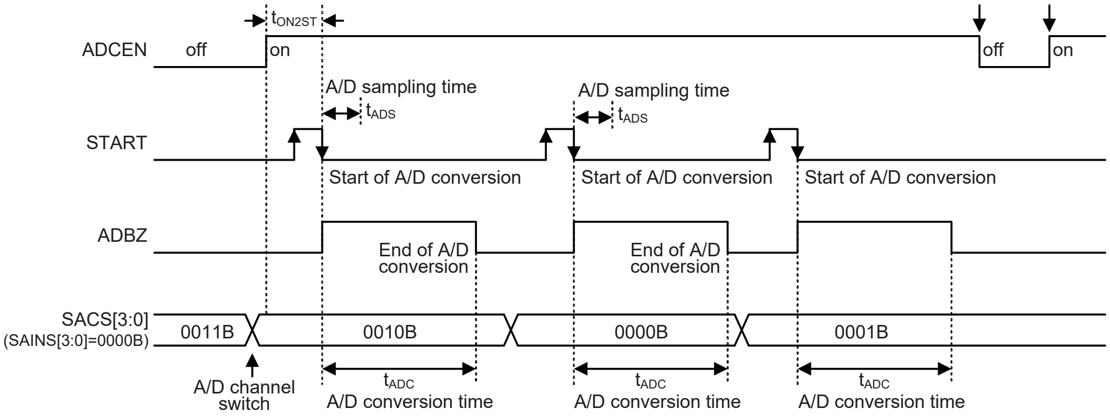
A/D Converter Input Signal Selection

Conversion Rate and Timing Diagram

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

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

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



A/D Conversion Timing – External Channel Input

Summary of A/D Conversion Steps

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

- Step 1
  Select the required A/D conversion clock by correctly programming bits SACKS2~SACKS0 in the SADC1 register.
- Step 2
  Enable the A/D converter by setting the ADCEN bit in the SADC0 register to one.
- Step 3
  Select which signal is to be connected to the internal A/D converter by correctly configuring the SAINS3~SAINS0 bits in the SADC1 register.
  Select the external channel input to be converted, go to Step 4.
  Select the internal analog signal to be converted, go to Step 5.
- Step 4
  If the A/D input signal comes from the external channel input selected by configuring the SAINS bit field, the corresponding pins should be configured as A/D input function by configuring the relevant pin-shared function control bits. The desired analog channel then should be selected by configuring the SACS bit field. After this step, go to Step 6.
- Step 5
  If the A/D input signal comes from the internal analog signal, the SAINS bit field should be properly configured and then the external channel input will automatically be disconnected regardless of the SACS bit field value. After this step, go to Step 6.
- Step 6
  Select the reference voltage source by configuring the SAVRS1~SAVRS0 bits in the SADC2 register. If the PGA output voltage is selected, the PGA must be enabled and then select the PGA input source by configuring the PGAIS bit in the SADC2 register.
- Step 7
  Select A/D converter output data format by setting the ADRFS bit in the SADC0 register.
- Step 8
  If A/D conversion interrupt is used, the interrupt control registers must be correctly configured to ensure the A/D interrupt function is active. As the A/D Converter interrupt is contained within a Multi-function interrupt, the associated multi-function interrupt enable bit, MFnE, the master interrupt control bit, EMI, and the A/D conversion interrupt control bit, ADE, must both be set high in advance.
- Step 9
  The A/D conversion procedure can now be initialized by setting the START bit from low to high and then low again.
- Step 10
  If A/D conversion is in progress, the ADBZ flag will be set high. After the A/D conversion process is complete, the ADBZ flag will go low and then the output data can be read from SADOH and SADOL registers.

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

Programming Considerations

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

A/D Conversion Function

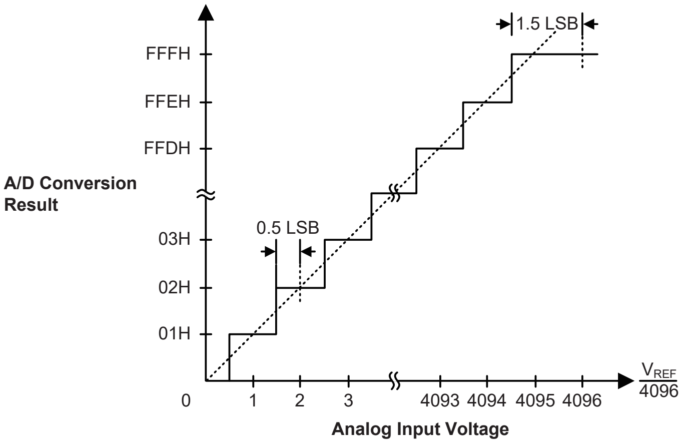
As the device contains a 12-bit A/D converter, its full-scale converted digitised value is equal to 0FFFH. Since the full-scale analog input value is equal to the actual A/D converter reference voltage, V_{REF}, this gives a single bit analog input value of V_{REF} divided by 4096.

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

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

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

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



Ideal A/D Transfer Function

A/D Conversion Programming Examples

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

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

```
clr ADE              ; disable ADC interrupt
mov a,03H
mov SADC1,a          ; select fSYS/8 as A/D clock
mov a,01h            ; setup PDS0 to configure pin AN0
mov PDS0,a
mov a,20h
mov SADC0,a
mov a,00h
mov SADC2,a            ; enable and connect AN0 channel to A/D converter
:
:
start_conversion:
clr START              ; high pulse on start bit to initiate conversion
set START              ; reset A/D
clr START              ; start A/D
polling_EOC:
sz ADBZ                ; poll the SADC0 register ADBZ bit to detect end of A/D conversion
jmp polling_EOC        ; continue polling
mov a,SADOL            ; read low byte conversion result value
mov SADOL_buffer,a     ; save result to user defined register
mov a,SADOH            ; read high byte conversion result value
mov SADOH_buffer,a     ; save result to user defined register
:
:
jmp start_conversion   ; start next A/D conversion
```

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

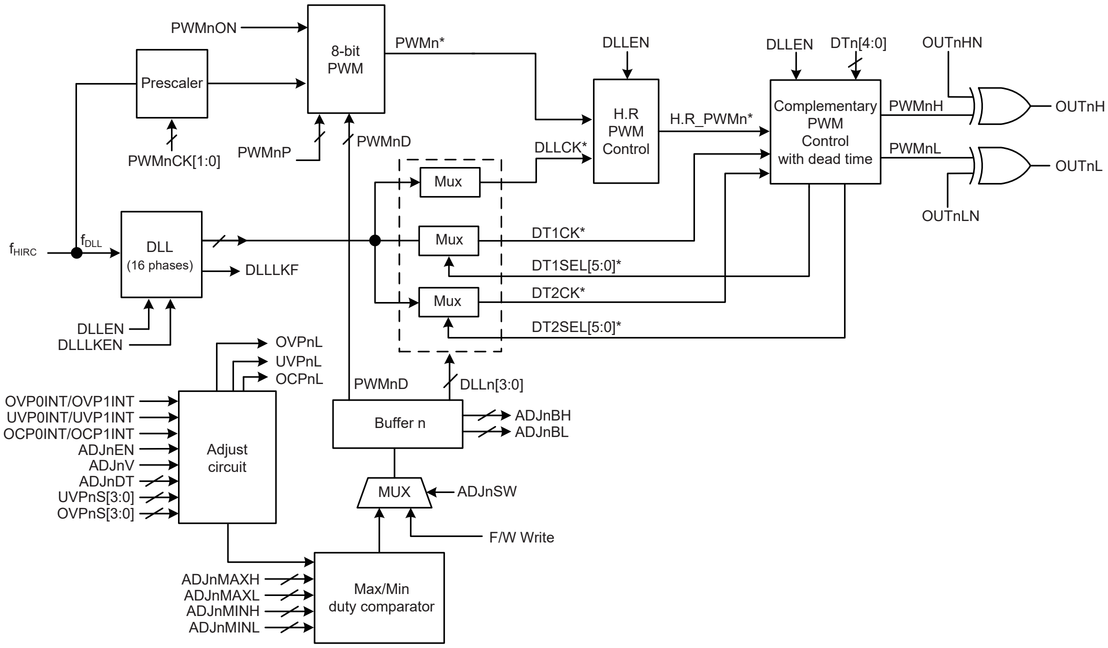
```
clr ADE                ; disable ADC interrupt
mov a,03H
mov SADC1,a            ; select fSYS/8 as A/D clock
mov a,01h              ; setup PDS0 to configure pin AN0
mov PDS0,a
mov a,20h
mov SADC0,a
mov a,00h
mov SADC2,a            ; enable and connect AN0 channel to A/D converter
:
:
start_conversion:
clr START              ; high pulse on START bit to initiate conversion
set START              ; reset A/D
clr START              ; start A/D
clr ADF                ; clear ADC interrupt request flag
set ADE                ; enable ADC interrupt
set EMI                ; enable global interrupt
:
:
; ADC interrupt service routine
ADC_ISR:
mov acc_stack,a        ; save ACC to user defined memory
mov a,STATUS
mov status_stack,a     ; save STATUS to user defined memory
:
:
mov a,SADOL            ; read low byte conversion result value
mov SADOL_buffer,a     ; save result to user defined register
mov a,SADOH            ; read high byte conversion result value
mov SADOH_buffer,a     ; save result to user defined register
:
:
EXIT_INT_ISR:
mov a,status_stack
mov STATUS,a           ; restore STATUS from user defined memory
mov a,acc_stack        ; restore ACC from user defined memory
reti
```

Auto Adjust High Resolution PWM with Delay Lock Loop/Dead-Time

The device contains a multi-feature fully integrated PWM Generator which has complimentary outputs for maximum application flexibility.

Functional Description

The High Resolution 8-bit PWM circuits include a PWM generator circuit, a delay lock loop circuit and PWM complementary outputs with dead time insertion. The device also provides the PWM duty adjusting control for high resolution PWM output.



H.R PWM Output Block Diagram (n=0~1)

Note: 1. "*" means it is the internal signal name and not the Special Function Register bit.

DT1SEL[5:0] and DT2SEL[5:0] are calculated and selected automatically based on the DTn[4:0].

DT1CK is the PWMnH DT reference signal

DT2CK is the PWMnL DT reference signal

DLLCK is the H.R_PWMn reference signal

2. H.R=High Resolution

High Resolution PWM Registers

The basic operation of the High Resolution PWM is controlled using several registers. A PWM period register, PWMnP, exists to store the desired 8-bit PWM period value. The PWM duty value is stored in an 8-bit PWMnD register. The PWMn function control, PWMn counter clock selection, DLL circuit and dead time duration is determined by the PWMnC register. The register DLLn is used for the DLL circuit phase selection. The DLLC register is used for the DLL circuit enable control and the losing lock protection control.

There are also some registers used for the auto adjust PWM function. The register ADJnC is to control the auto-adjust function enable/disable, the PWMn duty adjust operation control and store the OUVPn comparator output status. The ADJnS register is to select the adjust steps when over voltage or under voltage condition occurs while the ADJnDT is to select the auto adjust function delay time after being triggered. The two register pairs of ADJnMAXH & ADJnMAXL and

ADJnMINH & ADJnMINL are used to set the maximum and minimum duty data. The register pair of ADJnBH & ADJnBL is used to store the auto-adjust PWM buffer duty data. The remaining register of OUTPC is used for the PWMn output signals inverting control.

Register Name	Bit							
	7	6	5	4	3	2	1	0
PWMnP	D7	D6	D5	D4	D3	D2	D1	D0
PWMnD	D7	D6	D5	D4	D3	D2	D1	D0
PWMnC	PWMnCK1	PWMnCK0	PWMnON	DTn4	DTn3	DTn2	DTn1	DTn0
DLLn	DLLn3	DLLn2	DLLn1	DLLn0	—	—	—	—
DLLC	DLLLKEN	DLLEN	—	—	—	—	—	DLLLKF
ADJnC	ADJnEN	ADJnV	ADJnSW	—	OVPnL	UVPnL	OCPnL	—
ADJnS	OVPnS3	OVPnS2	OVPnS1	OVPnS0	UVPnS3	UVPnS2	UVPnS1	UVPnS0
ADJnDT	—	—	D5	D4	D3	D2	D1	D0
ADJnMAXH	D11	D10	D9	D8	D7	D6	D5	D4
ADJnMAXL	D3	D2	D1	D0	—	—	—	—
ADJnMINH	D11	D10	D9	D8	D7	D6	D5	D4
ADJnMINL	D3	D2	D1	D0	—	—	—	—
ADJnBH	D11	D10	D9	D8	D7	D6	D5	D4
ADJnBL	D3	D2	D1	D0	—	—	—	—
OUTPC	OUT1HS	OUT1LS	OUT0HS	OUT0LS	OUT1HN	OUT1LN	OUT0HN	OUT0LN

High Resolution PWM Generator & Auto-adjust Register List (n=0~1)

• PWMnP Register

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

Bit 7~0 **D7~D0**: 8-bit PWMn period register
PWMn period=PWMnP[7:0]+1

• PWMnD Register

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

Bit 7~0 **D7~D0**: 8-bit PWMn duty register
These registers, PWMnP and PWMnD, are used for 8-bit PWMn Period and Duty control. The following should be noted during setup:

1. The PWMnD value should meet the condition: 1≤PWMnD≤(PWMnP–1)
2. PWMnD (Min.)=1+DLLn[3:0]–DTn[4:0] where DLLn[3:0]=0000B, DTn[4:0]=11111B
3. PWMnD (Max.)=PWMnP–1+DLLn[3:0]–DTn[4:0] where DLLn[3:0]=1111B, DTn[4:0]=00000B

- **PWMnC Register**

Bit	7	6	5	4	3	2	1	0
Name	PWMnCK1	PWMnCK0	PWMnON	DTn4	DTn3	DTn2	DTn1	DTn0
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 **PWMnCK1~PWMnCK0**: PWMn counter clock source selection
00: f_{HIRC}
01: $f_{HIRC}/2$
10: $f_{HIRC}/4$
11: $f_{HIRC}/8$

Bit 5 **PWMnON**: PWMn function control bit
0: Disable, PWM counter=0
1: Enable

Bit 4~0 **DTn4~DTn0**: Dead time selection
00000: Dead time=t_{DLL}×0~t_{DLL}×1
00001: Dead time=t_{DLL}×2~t_{DLL}×3
00010: Dead time=t_{DLL}×4~t_{DLL}×5
00011: Dead time=t_{DLL}×6~t_{DLL}×7
00100: Dead time=t_{DLL}×8~t_{DLL}×9
00101: Dead time=t_{DLL}×10~t_{DLL}×11
00110: Dead time=t_{DLL}×12~t_{DLL}×13
00111: Dead time=t_{DLL}×14~t_{DLL}×15
01000: Dead time=t_{DLL}×16~t_{DLL}×17
…
11110: Dead time=t_{DLL}×60~t_{DLL}×61
11111: Dead time=t_{DLL}×62~t_{DLL}×63
Note: $t_{DLL}=1/(f_{HIRC}\times16)$

- **DLLn Register**

Bit	7	6	5	4	3	2	1	0
Name	DLLn3	DLLn2	DLLn1	DLLn0	—	—	—	—
R/W	R/W	R/W	R/W	R/W	—	—	—	—
POR	0	0	0	0	—	—	—	—

Bit 7~4 **DLLn3~DLLn0**: DLL phase selection
0000: H.R_PWMn Duty Falling edge is fine-adjusted to be at the DLL phase#0 Rising edge
0001: H.R_PWMn Duty Falling edge is fine-adjusted to be at the DLL phase#1 Rising edge
0010: H.R_PWMn Duty Falling edge is fine-adjusted to be at the DLL phase#2 Rising edge
…
1110: H.R_PWMn Duty Falling edge is fine-adjusted to be at the DLL phase#14 Rising edge
1111: H.R_PWMn Duty Falling edge is fine-adjusted to be at the DLL phase#15 Rising edge

Bit 3~0 Unimplemented, read as “0”

• DLLC Register

Bit	7	6	5	4	3	2	1	0
Name	DLLLKEN	DLLEN	—	—	—	—	—	DLLLKF
R/W	R/W	R/W	—	—	—	—	—	R/W
POR	1	0	—	—	—	—	—	0

Bit 7 **DLLLKEN**: DLL circuit Losing Lock protection function control
0: Disable
1: Enable

Note: When DLLLKEN=1 and DLL function is enabled, if a losing lock condition occurs, the DLLLKF bit is set high and the losing lock condition will be solved by DLL automatically. While when DLLLKEN=0, the DLLLKF bit is always zero even if a losing lock condition occurs, and the DLL will not solve the condition.

Bit 6 **DLLEN**: DLL and Dead Time function control bit
0: DLL disabled and no dead time inserted
1: DLL enabled and the dead time inserted which is decided by DTn[4:0]

If this bit is cleared then the PWMnCK[1:0] bits can be set to 00~11 by software and the H.R_PWMn=PWMn, no dead time is inserted. If set high, the hardware will set PWMnCK[1:0] be 00 which cannot be changed, the PWMn will be finely adjusted by the DLL and then output the High Resolution PWM output with dead time inserted.

Bit 5~1 Unimplemented, read as “0”

Bit 0 **DLLLKF**: DLL circuit losing lock flag
0: No losing lock condition occurs
1: Losing lock occurs

This bit can be cleared to zero by software, but can not be set high by software.

Note: When DLLLKEN=1 and DLL function is enabled, if no losing lock condition occurs, the DLLLKF bit is 0, if a losing lock condition occurs, the DLLLKF bit is set high which can only be cleared by software. If DLLLKEN=0, the DLLLKF bit is always zero.

• ADJnDT Register

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

Bit 7~6 Unimplemented, read as “0”

Bit 5~0 **D5~D0**: Auto Adjust PWMn delay time selection
000000: Delay time=PWMn Cycle×2
000001: Delay time=PWMn Cycle×4
000010: Delay time=PWMn Cycle×8
…
111111: Delay time=PWMn Cycle×128

Delay time=(ADJnDT[5:0]+1)×PWMn Cycle×2

- **ADJnS Register**

Bit	7	6	5	4	3	2	1	0
Name	OVPnS3	OVPnS2	OVPnS1	OVPnS0	UVPnS3	UVPnS2	UVPnS1	UVPnS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~4 **OVPnS3~OVPnS0**: OVP0/OVP1 auto adjust PWMn duty steps selection
0000: 0 step
0001: 1 step
…
1111: 15 steps

Bit 3~0 **UVPnS3~UVPnS0**: UVP0/UVP1 auto adjust PWMn duty steps selection
0000: 0 step
0001: 1 step
…
1111: 15 steps

- **ADJnC Register**

Bit	7	6	5	4	3	2	1	0
Name	ADJnEN	ADJnV	ADJnSW	—	OVPnL	UVPnL	OCPnL	—
R/W	R/W	R/W	R/W	—	R	R	R	—
POR	0	0	0	—	x	x	x	—

"x": unknown

Bit 7 **ADJnEN**: Auto Adjust PWMn Duty Function control
0: Disable
1: Enable

Bit 6 **ADJnV**: Auto Adjust PWMn Duty action selection
0: OVPnL increase duty, UVPnL decrease duty
1: OVPnL decrease duty, UVPnL increase duty

Bit 5 **ADJnSW**: PWMn Duty adjustment by S/W auto adjust control
0: Disable, write into Buffer from PWMnD+DLLn registers by F/W
1: Enable, write into Buffer by auto adjust system

This bit can be cleared to zero by software, but can not be set high by software.

When the ADJnEN is cleared to zero, the auto adjust circuit is off, so this bit is always 0. When the ADJnEN bit is set high, the auto adjust circuit is on.

Bit 4 Unimplemented, read as "0"

Bit 3 **OVPnL**: OVP0/OVP1 Comparator Output Status
0: Output low (no over voltage occurs)
1: Output high (over voltage occurs)

Bit 2 **UVPnL**: UVP0/UVP1 Comparator Output Status
0: Output low (no under voltage occurs)
1: Output high (under voltage occurs)

Bit 1 **OCPnL**: OCP0/OCP1 Comparator Output Status
0: Output low (no over current occurs)
1: Output high (over current occurs)

Bit 0 Unimplemented, read as "0"

The relationship between OVPnL/UVPnL/OCPnL and ADJnSW is shown in the following table.

OVPnL	UVPnL	OCPnL	Note	
			ADJnSW	Description
0	0	0	0	The auto adjust system is not triggered.
0	0	1	0	An over current occurs, the auto adjust system is not triggered.
0	1	0	1	The auto adjust system is triggered, no over current occurs.
0	1	1	0	A under voltage occurs, the auto adjust system is not triggered since an over current occurs.
1	0	0	1	The auto adjust system is triggered, no over current occurs.
1	0	1	1	An over voltage occurs, the auto adjust system is triggered regardless of an over current occurs.
1	1	0	0	An over voltage and a under voltage occur, the external circuit error will occur and the auto adjust system is not triggered.
1	1	1	0	An over current occurs, the auto adjust system is not triggered regardless of an over voltage and a under voltage occur.

- **ADJnMAXH & ADJnMAXL Registers**

Register	ADJnMAXH								ADJnMAXL							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	—	—	—	—
POR	0	0	0	0	0	0	0	0	0	0	0	0	—	—	—	—

"—": Unimplemented, read as "0"

D11~D4: Auto-adjust PWMn Maximum duty high byte

D3~D0: Auto-adjust PWMn Maximum duty low byte

D11~D4 is corresponding to the PWMnD[7:0], D3~D0 is corresponding to the DLLn[3:0] bits.

- **ADJnMINH & ADJnMINL Registers**

Register	ADJnMINH								ADJnMINL							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	—	—	—	—
POR	0	0	0	0	0	0	0	0	0	0	0	0	—	—	—	—

"—": Unimplemented, read as "0"

D11~D4: Auto-adjust PWMn Minimum duty high byte

D3~D0: Auto-adjust PWMn Minimum duty low byte

D11~D4 is corresponding to the PWMnD[7:0], D3~D0 is corresponding to the DLLn[3:0] bits.

- **ADJnBH & ADJnBL Registers**

Register	ADJnBH								ADJnBL							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	—	—	—	—
POR	0	0	0	0	0	0	0	0	0	0	0	0	—	—	—	—

"—": Unimplemented, read as "0"

D11~D4: Auto-adjust PWMn Buffer duty high byte

D3~D0: Auto-adjust PWMn Buffer duty low byte

D11~D4 is corresponding to the PWMnD[7:0], D3~D0 is corresponding to the DLLn[3:0] bits.

- **OUTPC Register**

Bit	7	6	5	4	3	2	1	0
Name	OUT1HS	OUT1LS	OUT0HS	OUT0LS	OUT1HN	OUT1LN	OUT0HN	OUT0LN
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 **OUT1HS**: OUT1H status when an OCP/OUVP occurs or when the PWMn is disabled
0: Output 0
1: Output 1

Bit 6 **OUT1LS**: OUT1L status when an OCP/OUVP occurs or when the PWMn is disabled
0: Output 0
1: Output 1

Bit 5 **OUT0HS**: OUT0H status when an OCP/OUVP occurs or when the PWMn is disabled
0: Output 0
1: Output 1

Bit 4 **OUT0LS**: OUT0L status when an OCP/OUVP occurs or when the PWMn is disabled
0: Output 0
1: Output 1

Bit 3 **OUT1HN**: OUT1H signal inverting control
0: Non-invert
1: Invert

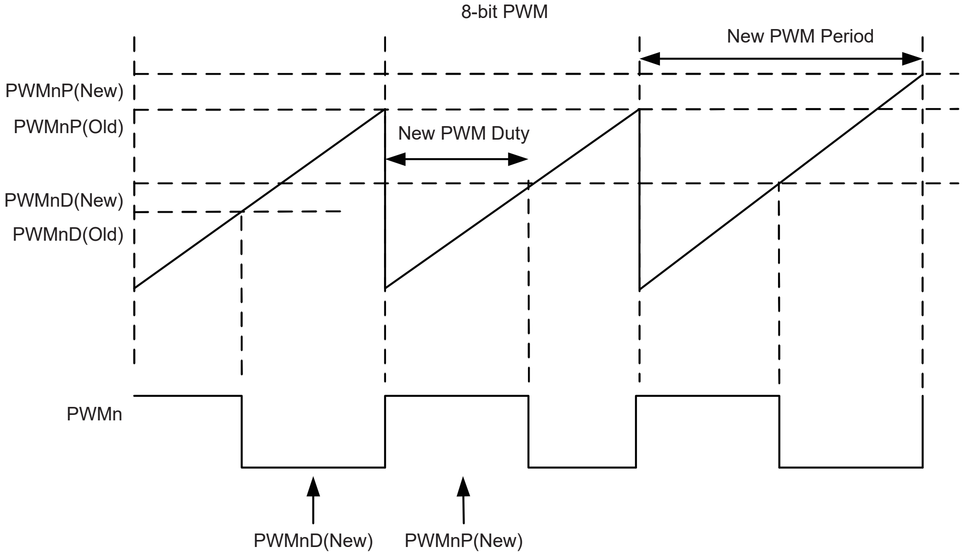
Bit 2 **OUT1LN**: OUT1L signal inverting control
0: Non-invert
1: Invert

Bit 1 **OUT0HN**: OUT0H signal inverting control
0: Non-invert
1: Invert

Bit 0 **OUT0LN**: OUT0L signal inverting control
0: Non-invert
1: Invert

PWM Generator

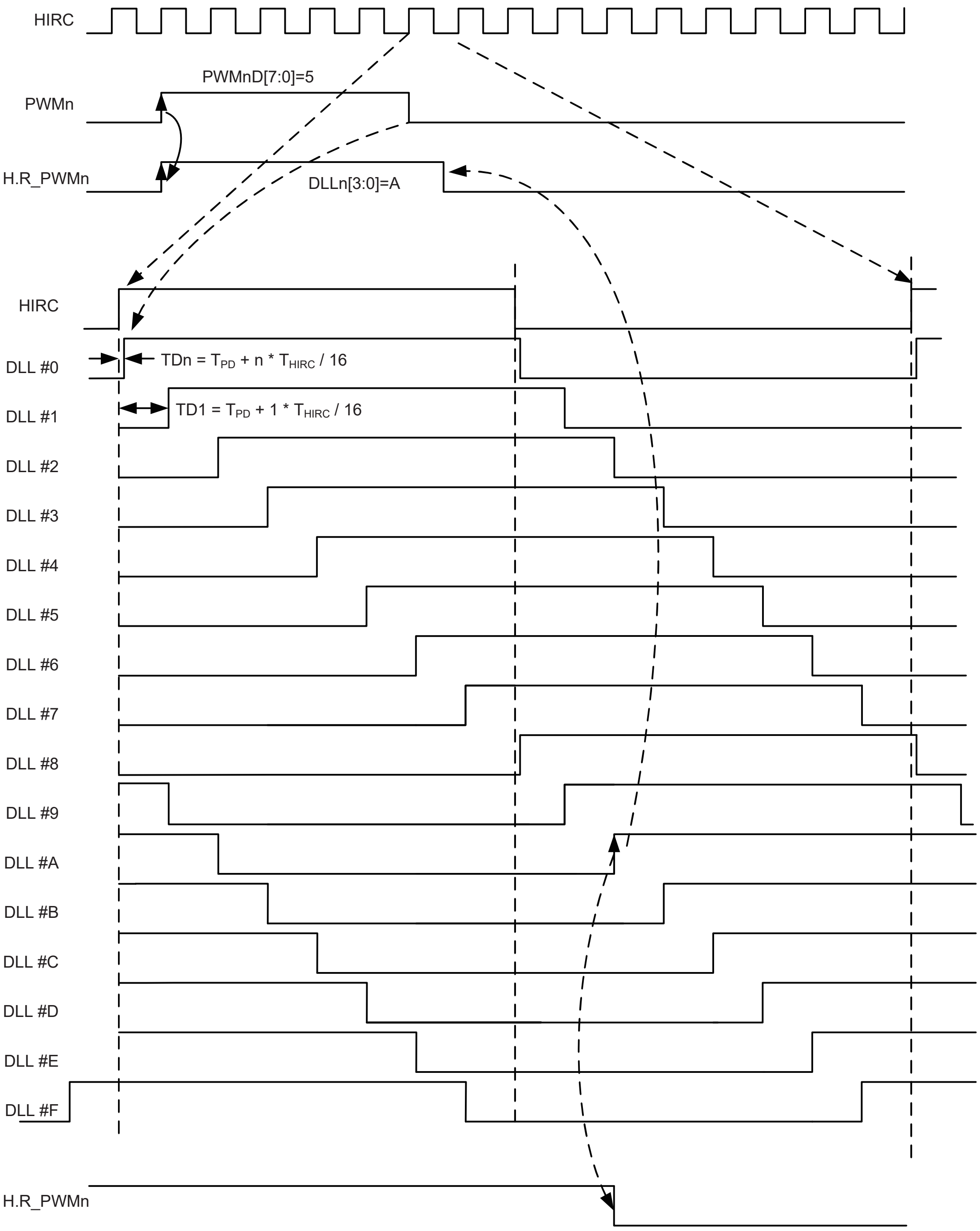
The PWM signal generator is driven by the HIRC clock and can generate PWMn signal, with a variable duty and period cycles by configuring the 8-bit PWMnP and PWMnD registers. The PWMn signal period is dependent upon the PWMn counter clock source which is set by the PWMnCK[1:0] bits in the PWMnC register and determined by the PWMnP register. The PWMn signal duty is determined by the PWMnD register content.



After the DLLn, DTn[4:0], PWMnD and PWMnP register values are changed by software, then the new data will be updated by the hardware when the PWMn Counter is cleared to zero.

Delay Lock Loop

DLL is an abbreviation for Delay Lock Loop. The DLL can generate 16 phase outputs within one HIRC clock period. The 16 phase outputs are used to fine tune the PWMn signal output. The PWMn clock is f_{HIRC}, which means that the PWMn output duty resolution is $1/f_{HIRC}$. The PWMn signal passes through the DLLn phase selection and PWMn control which is set by the DLLn[3:0] bits in the DLLn register circuit to output a fine-tuned PWMn signal, H.R_PWMn with the PWMn duty resolution increased by 4 bits.


HIRC
PWMnD[7:0]=5
PWMn
H.R_PWMn
DLLn[3:0]=A
HIRC
DLL #0
TDn = T_{PD} + n * T_{HIRC} / 16
DLL #1
TD1 = T_{PD} + 1 * T_{HIRC} / 16
DLL #2
DLL #3
DLL #4
DLL #5
DLL #6
DLL #7
DLL #8
DLL #9
DLL #A
DLL #B
DLL #C
DLL #D
DLL #E
DLL #F
H.R_PWMn

Losing Lock Protection

The device also provides the Losing lock protection circuit which can be enabled by the DLLLKEN bit.

If the MCU is disturbed, a losing lock error that the phase time generated by the DLL is 1.5 times of the normal phase time may occur. If the DLLLKEN bit is high, the losing lock circuit is enabled and then the phase time will return normal in 30μs. Additionally the flag bit DLLLKF which is 0 in normal operation will be set high to notify users that a losing lock error occurred. If the DLLLKEN bit is not set high, the Losing lock protection circuit is off. So after a losing lock condition occurs, the DLL phase time cannot return normal automatically and the DLLLKF flag is always 0.

Auto-adjust Circuit

In order to increase the DC-DC response speed, the device provides an auto-adjust circuit together with the PWM generator. The following summarises the steps to implement the auto adjust function.

Step 1. Clear the ADJnEN bit to zero to turn off the auto-adjust circuit for initialization:

- Set the Maximum/Minimum Duty by programming the12-bit ADJnMAXH & ADJnMAXL and ADJnMINH & ADJnMINL registers. Note: 1<Min<(PWMnD+DLLn)<Max
- Configure the 6-bit ADJnDT register to set the delay time after every trigger.
- Set the ADJnV bit in the ADJnC register to select the duty adjust action (increase or decrease).
- Select the duty adjusting step (0~15) when OUVP0/OUVP1 occurs by setting the ADJnS register.
- Set the OVP0/OVP1 and UVP0/UVP1 limited Voltage. Note: UVP0/UVP1<Target Voltage <OVP0/OVP1

Step 2. PWMn Start:

- PWMn initialization, including PWMnP and (PWMnD+DLLn) initialization
- Set the PWMnON bit high to enable the PWMn generator.
- Adjust the Duty after reading the OUVP0/OUVP1 output, make sure the output voltage equal to the target voltage.

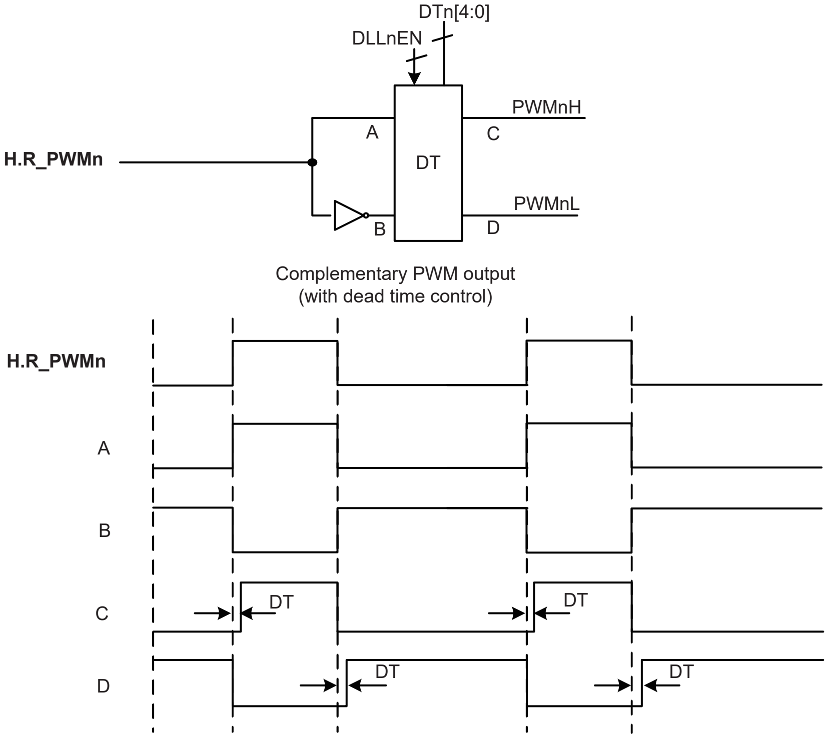
Step 3. Set the ADJnEN bit high to enable the auto-adjust circuit, then the OVPnL/UVPnL level can trigger the auto-adjustment function.

Step 4. If an UVP0/UVP1 interrupt occurs, based on the UVP0/UVP1 and OCP0/OCP1 comparator output, the firmware gives a delay time to determine the under-voltage condition is caused by increasing load instantaneously or by an external devices short circuit.

Step 5. If an OVP0/OVP1 interrupt occurs, based on the OVP0/OVP1 comparator output, the firmware gives a delay time to determine the over-voltage condition is caused by decreasing load instantaneously or by an error on the feedback circuit.

Dead-Time Insert

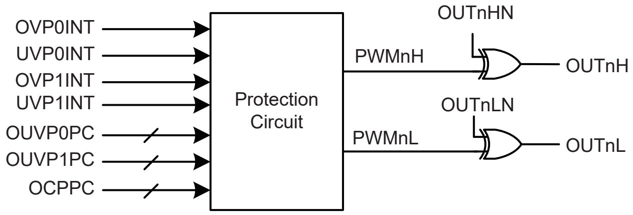
The device provides a complementary output pair of signals which can be used as a PWMn driver signal. The signal is sourced from the High Resolution PWM output signal, 8-bit PWMn with DLL circuit. PWM output is an active high signal. By using the DLLnEN bit, the dead time generator will be enabled and a dead time, which is programmable using the DTn[4:0] bits in the PWMnC register, will be inserted to prevent excessive DC currents. The dead time will be inserted whenever the rising edge of the dead time generator input signal occurs.



Note: C and D are the complementary PWMn control with dead time circuitry's output signals.

Protection and Inverting Control

Although a dead time has been inserted into the H.R_PWM complementary pair signals to prevent excessive DC current, these two signals may also be in an inactive state resulting from some unpredictable reasons, such as malfunctions or electrical noise. The device provides a protection function to force the two signals to output inverting signals when the PWMnH or PWMnL signal is in an inactive state. The inverting control circuitry determines whether the signals are inverted or not using corresponding inverting control bit, OUTnHN or OUTnLN bit, in the OUTPC register.



The device also includes over current protection, over voltage protection and under voltage protection functions for the PWMn output signals which are described in the Over Current Protection section OCPPC register and Over/Under Voltage Protection section OUVPnPC register. The PWM output OUT0H/OUT0L, OUT1H/OUT1L can be forced as inactive state controlled by OUT0HS/OUT0LS, OUT1HS/OUT1LS bits in the OUTPC register for one of OCP0/OCP1, OVP0/OVP1 or UVP0/UVP1 occurs. The OCP0/OCP1/OVP0/OVP1/UVP0/UVP1 also generates interrupt to inform MCU. Once OCP0/OCP1/OVP0/OVP1/UVP0/UVP1 disappears, the OUT0H/OUT0L, OUT1H/OUT1L will recover to send PWM output. Details about the current and voltage protection functions refer to the "Over Current Protection" and "Over/Under Voltage Protection" chapters.

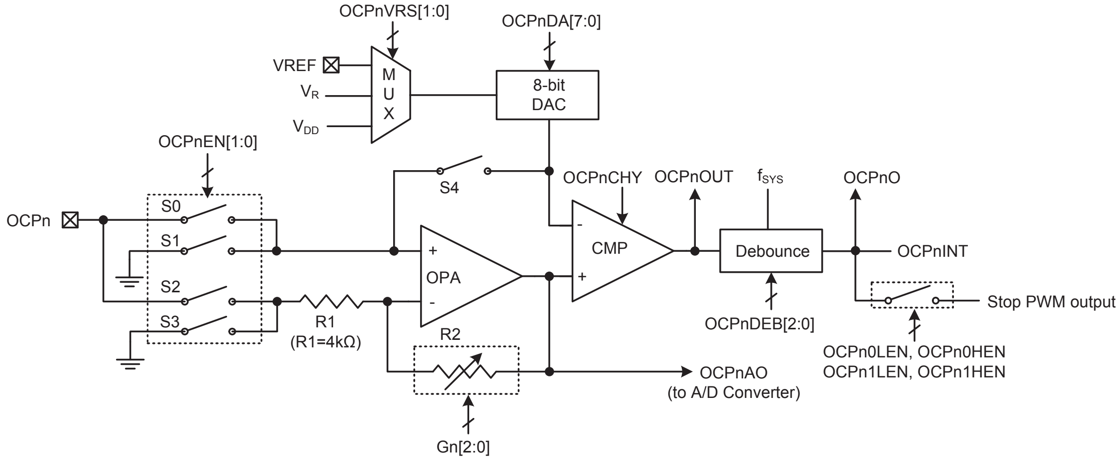
Programming Considerations

The following steps show the read and write procedures:

- Writing Data to DLLn/PWMnD
  - ♦ Step 1. Write data to DLLn
    – Note that here data is only written to the 4-bit buffer.
  - ♦ Step 2. Write data to PWMnD
    – Here data is written directly to PWMnD register and simultaneously data is latched from the 4-bit buffer to the DLLn register.
- Reading Data from DLLn/PWMnD
  - ♦ Step 1. Read data from PWMnD
    – Here data is read directly from the PWMnD register and simultaneously data is latched from the DLLn register into the 4-bit buffer.
  - ♦ Step 2. Read data from DLLn
    – This step reads data from the 4-bit buffer.

Over Current Protection

The device includes the over current protection function which provides a protection mechanism for applications. To prevent the battery charge or load current from exceeding a specific level, the current on the OCPn pin is converted to a relevant voltage level according to the current value using the OCPn operational amplifier. It is then compared with a reference voltage generated by an 8-bit D/A converter. When an over current event occurs, an OCPn interrupt will be generated if the corresponding interrupt control is enabled.



Note: V_R is from the A/D Converter PGA output and the OCPnAO can be selected as the A/D Converter input signals.

Over Current Protection Circuit (n=0~1)

Over Current Protection Operation

The illustrated OCPn circuit is used to prevent the input current from exceeding a reference level. The current on the OCPn pin is converted to a voltage and then amplified by the OCPn operational amplifier with a programmable gain from 1 to 50 selected by the Gn2~Gn0 bits in the OCPnC1 register. This is known as a Programmable Gain Amplifier or PGA. This PGA can also be configured to operate in the non-invert, invert or input offset calibration mode determined by the OCPnEN1 and OCPnEN0 bits in the OCPnC0 register. After the current is converted and amplified to a specific voltage level, it will be compared with a reference voltage provided by an 8-bit DAC. The 8-bit DAC power can be V_{DD}, V_R or V_{REF}, selected by the OCPnVRS[1:0] bits in the OCPnC0 register. The comparator output, OCPnCOUT, will first be filtered with a certain de-bounce time period selected by the OCPnDEB2~OCPnDEB0 bits in the OCPnC1 register. Then a filtered OCPn digital comparator output, OCPnO, is obtained to indicate whether an over current condition occurs or not. The OCPnO bit will be set to 1 if an over current condition occurs. Otherwise, the OCPnO bit is zero. Once an over current event occurs, i.e., the converted voltage of the OCPn input current is greater than the reference voltage, the corresponding interrupt will be generated if the relevant interrupt control bit is enabled.

The device provides over current protection control of the PWM output signals OUT0H/OUT0L, OUT1H/OUT1L which can be enabled by the corresponding OCPPC register bits. If the protection control is enabled, these signals can be forced to an inactive state when an over current protection condition occurs. The OCPn also generates an interrupt to inform the MCU. Once the over current condition is resolved the OUT0H/OUT0L, OUT1H/OUT1L outputs will recover and continue to generate PWM signals.

Details about the OUTnH and OUTnL signal polarity and output control when OCPn occurs is described in the OCPPC register description.

Over Current Protection Control Registers

Overall operation of the over current protection is controlled using several registers. One register is used to provide the reference voltages for the over current protection circuit. There are two registers used to cancel out the operational amplifier and comparator input offset. The two control registers are to control the OCPn function, D/A converter reference voltage select, PGA gain select, comparator de-bounce time together with the hysteresis function. The remaining register of OCPPC is used to control the whether the PWMn output signals OUT0H/OUT0L, OUT1H/OUT1L is forced into an inactive state when an OCPn condition occurs.

Register Name	Bit							
	7	6	5	4	3	2	1	0
OCPnC0	OCPnEN1	OCPnEN0	OCPnVRS1	OCPnVRS0	OCPnCHY	—	—	OCPnO
OCPnC1	—	—	Gn2	Gn1	Gn0	OCPnDEB2	OCPnDEB1	OCPnDEB0
OCPnDA	D7	D6	D5	D4	D3	D2	D1	D0
OCPnOCAL	OCPnOOFM	OCPnORSP	OCPnOOF5	OCPnOOF4	OCPnOOF3	OCPnOOF2	OCPnOOF1	OCPnOOF0
OCPnCCAL	OCPnCOUT	OCPnCOFM	OCPnCRSP	OCPnCOF4	OCPnCOF3	OCPnCOF2	OCPnCOF1	OCPnCOF0
OCPPC	OCP11LEN	OCP11HEN	OCP10LEN	OCP10HEN	OCP01LEN	OCP01HEN	OCP00LEN	OCP00HEN

OCPn Register List (n=0~1)

- **OCPnC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	OCPnEN1	OCPnEN0	OCPnVRS1	OCPnVRS0	OCPnCHY	—	—	OCPnO
R/W	R/W	R/W	R/W	R/W	R/W	—	—	R
POR	0	0	0	0	0	—	—	0

Bit 7~6 **OCPnEN1~OCPnEN0**: OCPn function operating mode selection
00: OCPn function is disabled, S1and S3 on, S0 and S2 off
01: Non-invert mode, S0 and S3 on, S1 and S2 off
10: Invert mode, S1 and S2 on, S0 and S3 off
11: Calibration mode, S1 and S3 on, S0 and S2 off

Bit 5~4 **OCPnVRS1~OCPnVRS0**: OCPn DAC reference voltage selection
00/11: From V_{DD}
01: From external VREF pin
10: From V_R

Note: when setting these bits to “10” to select the V_R as the OCPn DAC reference voltage, care must be taken that as the V_R signal is from the A/D Converter PGA output, so the PGA must first be enabled by setting the ADPGAEN bit high.

Bit 3 **OCPnCHY**: OCPn Comparator hysteresis function control
0: Disable
1: Enable

Bit 2~1 Unimplemented, read as “0”

Bit 0 **OCPnO**: OCPn digital output bit
0: No over current condition occurs in the monitored source current
1: Over current condition occurs in the monitored source current

- **OCPnC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	Gn2	Gn1	Gn0	OCPnDEB2	OCPnDEB1	OCPnDEB0
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 **Gn2~Gn0**: R2/R1 ratio selection
000: Unity gain buffer (non-invert mode) or R2/R1=1(invert mode)
001: R2/R1=5
010: R2/R1=10
011: R2/R1=15
100: R2/R1=20
101: R2/R1=30
110: R2/R1=40
111: R2/R1=50

These bits are used to select the R2/R1 ratio to obtain various gain values for invert and non-invert mode. The calculating formula of the OCPn PGA gain for the invert and non-invert mode is described in the “Input Voltage Range” section.

Bit 2~0 **OCPnDEB2~OCPnDEB0**: OCPn output filter debounce time selection
000: Bypass, without debounce
001: (1~2)×t_{DEB}
010: (3~4)×t_{DEB}
011: (7~8)×t_{DEB}
100: (15~16)×t_{DEB}
101: (31~32)×t_{DEB}
110: (63~64)×t_{DEB}
111: (127~128)×t_{DEB}

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

- **OCPnDA Register**

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

Bit 7~0 **D7~D0**: OCPn DAC output voltage control bits
OCP DAC Output=(DAC reference voltage/256)×D[7:0]

- **OCPnOCAL Register**

Bit	7	6	5	4	3	2	1	0
Name	OCPnOOFM	OCPnORSP	OCPnOOF5	OCPnOOF4	OCPnOOF3	OCPnOOF2	OCPnOOF1	OCPnOOF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	0	0	0	0	0

Bit 7 **OCPnOOFM**: OCPn Operational Amplifier normal operation or input offset voltage cancellation mode selection
0: Normal operation, input offset calibration disabled
1: Input Offset Calibration Mode

This bit is used to control the OCPn operational amplifier input offset Calibration function. The OCPnEN1 and OCPnEN0 bits must first be set to “11” and then the OCPnOOFM bit must be set to 1 followed by the OCPnCOFM bit being cleared to 0, then the operational amplifier input offset Calibration mode will be enabled. Refer to the “Operational Amplifier Input Offset Calibration” section for the detailed offset Calibration procedures.

Bit 6 **OCPnORSP**: OCPn Operational Amplifier Input Offset Voltage Calibration Reference selection
0: Select negative input as the reference input
1: Select positive input as the reference input

Bit 5~0 **OCPnOOF5~OCPnOOF0**: OCPn Operational Amplifier Input Offset Voltage Calibration value

This 6-bit field is used to perform the operational amplifier input offset calibration operation and the value for the OCPn operational amplifier input offset calibration can be restored into this bit field. More detailed information is described in the “Operational Amplifier Input Offset Calibration” section.

- **OCPnCCAL Register**

Bit	7	6	5	4	3	2	1	0
Name	OCPnCOUT	OCPnCOFM	OCPnCRSP	OCPnCOF4	OCPnCOF3	OCPnCOF2	OCPnCOF1	OCPnCOF0
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	1	0	0	0	0

Bit 7 **OCPnCOUT**: OCPn Comparator Output bit, positive logic (read only)
0: Positive input voltage < Negative input voltage
1: Positive input voltage > Negative input voltage

This bit is used to indicate whether the positive input voltage is greater than the negative input voltage when the OCPn operates in the input offset Calibration mode. If the OCPnCOUT is set to 1, the positive input voltage is greater than the negative input voltage. Otherwise, the positive input voltage is less than the negative input voltage.

Bit 6 **OCPnCOFM**: OCPn Comparator normal operation or input offset calibration mode selection
0: Normal operation, input offset calibration mode disabled
1: Input Offset Calibration Mode Enabled

This bit is used to control the OCPn comparator input offset Calibration function. The OCPnEN1 and OCPnEN0 bits must first be set to “11” and then the OCPnCOFM bit must be set to 1 followed by the OCPnOOFM bit being cleared to 0, then the comparator input offset calibration mode will be enabled. Refer to the "Comparator Input Offset Calibration" section for the detailed offset calibration procedures.

Bit 5 **OCPnCRSP**: OCPn Comparator Input Offset Calibration Reference Input selection
0: Select negative input as the reference input
1: Select positive input as the reference input

Bit 4~0 **OCPnCOF4~OCPnCOF0**: OCPn Comparator Input Offset Calibration value

This 5-bit field is used to perform the comparator input offset calibration operation and the value for the OCPn comparator input offset calibration can be restored into this bit field. More detailed information is described in the "Comparator Input Offset Calibration" section.

- **OCPPC Register**

Bit	7	6	5	4	3	2	1	0
Name	OCP11LEN	OCP11HEN	OCP10LEN	OCP10HEN	OCP01LEN	OCP01HEN	OCP00LEN	OCP00HEN
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 **OCP11LEN**: OUT1L Over Current Protection 1 Enable control
0: Disable
1: Enable

This bit is used to control the OUT1L signal output when an over current 1 condition occurs. If clear this bit to 0, the function is disabled. This means the OUT1L output will not be affected when OCP1 occurs. If set this bit high, the function is enabled. This means when an OCP1 condition occurs the OUT1L status is controlled by the OUT1LS bit in the OUTPC register.

Bit 6 **OCP11HEN**: OUT1H Over Current Protection 1 Enable control
0: Disable
1: Enable

This bit is used to control the OUT1H signal output when an over current 1 condition occurs. If clear this bit to 0, the function is disabled. This means the OUT1H output will not be affected when OCP1 occurs. If set this bit high, the function is enabled. This means when an OCP1 condition occurs the OUT1H status is controlled by the OUT1HS bit in the OUTPC register.

Bit 5 **OCP10LEN**: OUT0L Over Current Protection 1 Enable control
0: Disable
1: Enable

This bit is used to control the OUT0L signal output when an over current 1 condition occurs. If clear this bit to 0, the function is disabled. This means the OUT0L output will not be affected when OCP1 occurs. If set this bit high, the function is enabled. This means when an OCP1 condition occurs the OUT0L status is controlled by the OUT0LS bit in the OUTPC register.

Bit 4 **OCP10HEN**: OUT0H Over Current Protection 1 Enable control
0: Disable
1: Enable

This bit is used to control the OUT0H signal output when an over current 1 condition occurs. If clear this bit to 0, the function is disabled. This means the OUT0H output will not be affected when OCP1 occurs. If set this bit high, the function is enabled. This means when an OCP1 condition occurs the OUT0H status is controlled by the OUT0HS bit in the OUTPC register.

Bit 3 **OCP01LEN**: OUT1L Over Current Protection 0 Enable control
0: Disable
1: Enable

occurs. If clear this bit to 0, the function is disabled. This means the OUT1L output will not be affected when OCP0 occurs. If set this bit high, the function is enabled.

This means when an OCP0 condition occurs the OUT1L status is controlled by the OUT1LS bit in the OUTPC register.

Bit 2 **OCP01HEN**: OUT1H Over Current Protection 0 Enable control
0: Disable
1: Enable
This bit is used to control the OUT1H signal output when an over current 0 condition occurs. If clear this bit to 0, the function is disabled. This means the OUT1H output will not be affected when OCP0 occurs. If set this bit high, the function is enabled. This means when an OCP0 condition occurs the OUT1H status is controlled by the OUT1HS bit in the OUTPC register.

Bit 1 **OCP00LEN**: OUT0L Over Current Protection 0 Enable control
0: Disable
1: Enable
This bit is used to control the OUT0L signal output when an over current 0 condition occurs. If clear this bit to 0, the function is disabled. This means the OUT0L output will not be affected when OCP0 occurs. If set this bit high, the function is enabled. This means when an OCP0 condition occurs the OUT0L status is controlled by the OUT0LS bit in the OUTPC register.

Bit 0 **OCP00HEN**: OUT0H Over Current Protection 0 Enable control
0: Disable
1: Enable
This bit is used to control the OUT0H signal output when an over current 0 condition occurs. If clear this bit to 0, the function is disabled. This means the OUT0H output will not be affected when OCP0 occurs. If set this bit high, the function is enabled. This means when an OCP0 condition occurs the OUT0H status is controlled by the OUT0HS bit in the OUTPC register.

Input Voltage Range

Together with different PGA operating modes, the input voltage on the OCPn pin can be positive or negative for flexible operation. The PGA output for the positive or negative input voltage is calculated based on different formulas and described by the following.

- For input voltages V_{IN}>0, the PGA operates in the non-invert mode and the PGA output is obtained using the formula below:

$$V_{OUT} = (1+R_2/R_1)\times V_{IN}$$

- When the PGA operates in the non-invert mode by setting the OCPnEN[1:0] to “01” with unity gain select by setting the Gn[2:0] to “000”, the PGA will act as a unit-gain buffer whose output is equal to V_{IN}.

$$V_{OUT} = V_{IN}$$

- For input voltages 0>V_{IN}>-0.4V, the PGA operates in the invert mode and the PGA output is obtained using the formula below. Note that if the input voltage is negative, it cannot be lower than -0.4V which will result in current leakage.

$$V_{OUT} = -(R_2/R_1)\times V_{IN}$$

OCPn OPA and Comparator Offset Calibration

The OCPn circuit has four operating modes controlled by OCPnEN[1:0], one of them is calibration mode. In calibration mode, Operational amplifier and comparator offset can be calibrated.

Operational Amplifier Input Offset Calibration

Step 1. Set OCPnEN[1:0]=11, OCPnOOFM=1 and OCPnCOFM=0, the OCPn will operate in the operational amplifier input offset Calibration mode.

Step 2. Set OCPnOOF[5:0]=000000B and then read the OCPnCOUT bit.

Step 3. Increase the OCPnOOF[5:0] value by 1 and then read the OCPnCOUT bit.

If the OCPnCOUT bit state has not changed, then repeat Step 3 until the OCPnCOUT bit state has changed.

If the OCPnCOUT bit state has changed, record the OCPnOOF value as V_{OOS1} and then go to Step 4.

Step 4. Set OCPnOOF[5:0]=111111B and read the OCPnCOUT bit.

Step 5. Decrease the OCPnOOF[5:0] value by 1 and then read the OCPnCOUT bit.

If the OCPnCOUT bit state has not changed, then repeat Step 5 until the OCPnCOUT bit state has changed.

If the OCPnCOUT bit state has changed, record the OCPnOOF value as V_{OOS2} and then go to Step 6.

Step 6. Restore the operational amplifier input offset calibration value V_{OOS} into the OCPnOOF[5:0] bit field. The offset Calibration procedure is now finished.

Where $V_{OOS}=(V_{OOS1}+V_{OOS2})/2$

Comparator Input Offset Calibration

Step 1. Set OCPnEN[1:0]=11, OCPnCOFM=1 and OCPnOOFM=0, the OCPn is now in the comparator input offset calibration mode. S4 is on (S4 is used for calibration mode, in normal mode operation, it is off).

Step 2. Set OCPnCOF[4:0]=00000B and read the OCPnCOUT bit.

Step 3. Increase the OCPnCOF[4:0] value by 1 and then read the OCPnCOUT bit.

If the OCPnCOUT bit state has not changed, then repeat Step 3 until the OCPnCOUT bit state has changed.

If the OCPnCOUT bit state has changed, record the OCPnCOF value as V_{COS1} and then go to Step 4.

Step 4. Set OCPnCOF[4:0]=11111B and then read the OCPnCOUT bit.

Step 5. Decrease the OCPnCOF[4:0] value by 1 and then read the OCPnCOUT bit.

If the OCPnCOUT bit state has not changed, then repeat Step 5 until the OCPnCOUT bit state has changed.

If the OCPnCOUT bit state has changed, record the OCPnCOF value as V_{COS2} and then go to Step 6.

Step 6. Restore the comparator input offset calibration value V_{COS} into the OCPnCOF[4:0] bit field. The offset Calibration procedure is now finished.

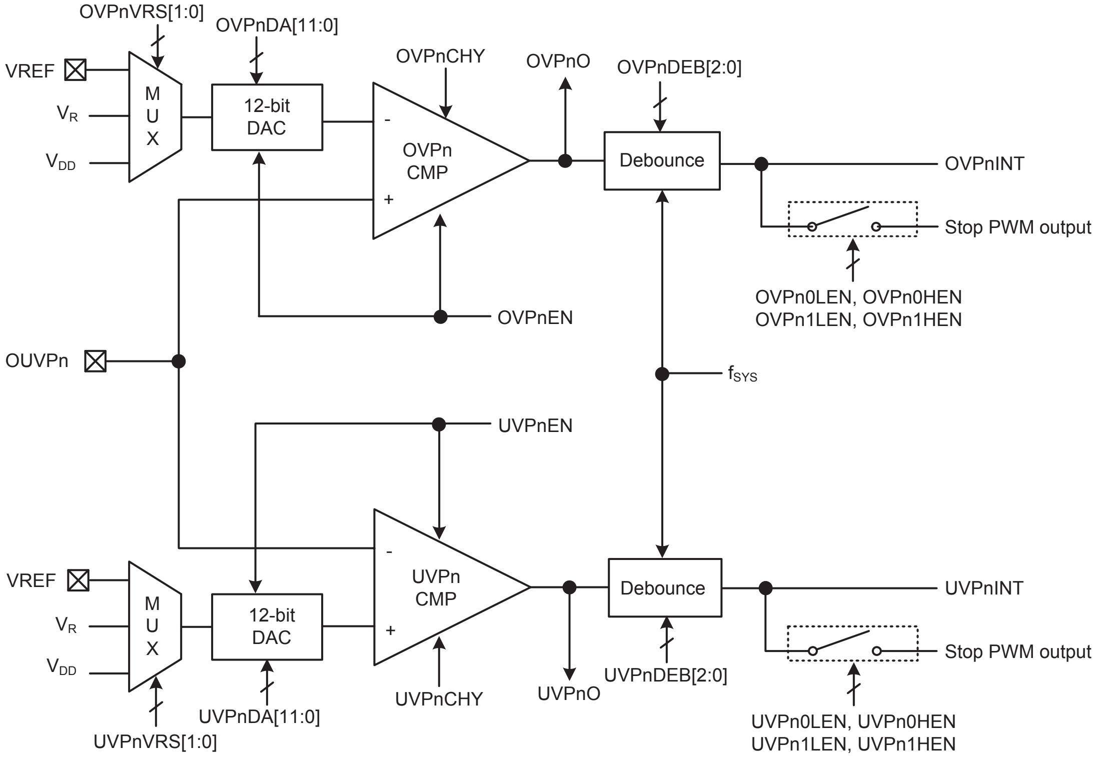
Where $V_{COS}=(V_{COS1}+V_{COS2})/2$

Over/Under Voltage Protection

The device includes the internal over/under voltage protection (OUVP) function which can be used for the application of battery charge/discharge.

OUVP Circuit Operation

The OUVP circuit is built-in with the Over Voltage Protection (OVPn) and the Under Voltage Protection (UVPn) functions.



Note: V_R is from the A/D Converter PGA output

Over/Under Voltage Protection Block Diagram (n=0~1)

Over Voltage Protection function

To prevent the output voltage from exceeding the specific voltage level, the OVPn input voltage is compared with a reference voltage generated by a 12-bit D/A converter. The 12-bit D/A converter reference input signal range can come from V_{DD}, V_R or external VREF pin which is selected by the OVPnVRS[1:0] bits. Once the OVPn input voltage is greater than the reference voltage, the OVPnO will change from “0” to “1”. The OVPnINT is the de-bounce version of OVPnO and used to indicate that the source voltage coming from OUVPn input is over the specification or not. OVPnO is defined as OVPn output and OVPnINT is OVPn interrupt trigger. The comparator of the OVPn also has a hysteresis function controlled by OVPnCHY bit.

Under Voltage Protection function

To prevent the output voltage from being less than the specific voltage, the UVPn input voltage is compared with a reference voltage generated by a 12-bit D/A converter. The 12-bit D/A converter reference input signal range can come from V_{DD}, V_R or external VREF pin which is selected by the UVPnVRS[1:0] bits. Once the UVPn input voltage is lower than the reference voltage, the UVPnO will change from “0” to “1”. The UVPnINT is the de-bounce version of UVPnO and used to indicate that the source voltage coming from the OUVPn input is under the specification or not. UVPnO is defined as UVPn output and UVPnINT is UVPn interrupt trigger. The comparator of the UVPn also has a hysteresis function controlled by UVPnCHY bit.

The device provides over and under voltage protection control of the PWM output signals OUT0H/OUT0L, OUT1H/OUT1L which can be enabled by the corresponding OUVPnPC register bits. If the protection control is enabled, these signals can be forced to an inactive state when an over voltage or under voltage conditon occurs. The OVPn/UVPn also generates an interrupt to inform MCU. Once the over voltage or under voltage condition is resolved the OUT0H/OUT0L, OUT1H/OUT1L outputs will continue to generate PWM signals.

Details about the OUTnH and OUTnL signal polarity and output control when OVPn or UVPn occurs is described in the OUVPnPC register description.

OUVPn Register Description

The overall operation of the voltage protection and under voltage protection is controlled using several registers.

Register Name	Bit							
	7	6	5	4	3	2	1	0
OUVPnC0	—	—	OVPnEN	OVPnCHY	OVPnVRS1	OVPnVRS0	OVPnDEB1	OVPnDEB0
OUVPnC1	—	—	UVPnEN	UVPnCHY	UVPnVRS1	UVPnVRS0	UVPnDEB1	UVPnDEB0
OUVPnC2	OVPnO	OVPnCOFM	OVPnCRS	OVPnCOF4	OVPnCOF3	OVPnCOF2	OVPnCOF1	OVPnCOF0
OUVPnC3	UVPnO	UVPnCOFM	UVPnCRS	UVPnCOF4	UVPnCOF3	UVPnCOF2	UVPnCOF1	UVPnCOF0
OVPnDAH	—	—	—	—	D11	D10	D9	D8
OVPnDAL	D7	D6	D5	D4	D3	D2	D1	D0
UVPnDAH	—	—	—	—	D11	D10	D9	D8
UVPnDAL	D7	D6	D5	D4	D3	D2	D1	D0
OUVPnPC	UVPn1LEN	UVPn1HEN	UVPn0LEN	UVPn0HEN	OVPn1LEN	OVPn1HEN	OVPn0LEN	OVPn0HEN

OUVP Register List (n=0~1)

- **OVPnDAH & OVPnDAL Registers**

Register	OVPnDAH								OVPnDAL							
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	—	—	—	—	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

"—": Unimplemented, read as "0"

D11~D8: OVPn DAC reference level selection high byte

D7~D0: OVPn DAC reference level selection low byte

OVPn DAC Output=(OVPn DAC V_{REF}/4096)×D[11:0]

Note: Write OVPnDAL register only write to shadow buffer, and until write OVPnDAH register will also copy the shadow buffer data to OVPnDAL register.

- **UVPnDAH & UVPnDAL Registers**

Register	UVPnDAH								UVPnDAL							
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	—	—	—	—	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

"—": Unimplemented, read as "0"

D11~D8: UVPn DAC reference level selection high byte

D7~D0: UVPn DAC reference level selection low byte

UVPn DAC Output=(UVPn DAC V_{REF}/4096)×D[11:0]

Note: Write UVPnDAL register only write to shadow buffer, and until write UVPnDAH register will also copy the shadow buffer data to UVPnDAL register.

- **OUVPnC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	OVPnEN	OVPnCHY	OVPnVRS1	OVPnVRS0	OVPnDEB1	OVPnDEB0
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 **OVPnEN**: Over Voltage Protection n function Enable control
0: Disable
1: Enable

If the OVPnEN bit is cleared to 0, the over voltage protection n function is disabled and no power will be consumed. This results in the comparator and D/A converter of OVPn all being switched off.

Bit 4 **OVPnCHY**: Over Voltage Protection n Comparator Hysteresis Enable control
0: Disable
1: Enable

Bit 3~2 **OVPnVRS1~OVPnVRS0**: OVPn DAC reference voltage selection
00: From V_{DD}
01: From external VREF pin
10: From V_R
11: From V_{DD}

Note: when seting these bits to "10" to select the V_R as the OVPn DAC reference voltage, care must be taken that as the V_R signal is from the A/D Converter PGA output, so the PGA must first be enabled by setting the ADPGAEN bit high.

Bit 1~0 **OVPnDEB1~OVPnDEB0**: Over Voltage Protection n comparator debounce time selection
00: No debounce
01: (7~8)×1/f_{SYS}
10: (15~16)×1/f_{SYS}
11: (31~32)×1/f_{SYS}

- **OUVPnC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	UVPnEN	UVPnCHY	UVPnVRS1	UVPnVRS0	UVPnDEB1	UVPnDEB0
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 **UVPnEN**: Under Voltage Protection n function Enable control
0: Disable
1: Enable

If the UVPnEN bit is cleared to 0, the under voltage protection n function is disabled and no power will be consumed. This results in the comparator and D/A converter of UVPn all being switched off.

Bit 4 **UVPnCHY**: Under Voltage Protection n Comparator Hysteresis Enable control
0: Disable
1: Enable

Bit 3~2 **UVPnVRS1~UVPnVRS0**: UVPn DAC reference voltage selection
00: From V_{DD}
01: From external VREF pin
10: From V_R
11: From V_{DD}

Note: when seting these bits to “10” to select the V_R as the UVPn DAC reference voltage, care must be taken that as the V_R signal is from the A/D Converter PGA output, so the PGA must first be enabled by setting the ADPGAEN bit high.

Bit 1~0 **UVPnDEB1~UVPnDEB0**: Under Voltage Protection n comparator debounce time selection
00: No debounce
01: (7~8)×1/f_{SYS}
10: (15~16)×1/f_{SYS}
11: (31~32)×1/f_{SYS}

- **OUVPnC2 Register**

Bit	7	6	5	4	3	2	1	0
Name	OVPnO	OVPnCOFM	OVPnCRS	OVPnCOF4	OVPnCOF3	OVPnCOF2	OVPnCOF1	OVPnCOF0
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	1	0	0	0	0

Bit 7 **OVPnO**: OVPn comparator output bit
0: Positive input voltage < negative input voltage
1: Positive input voltage > negative input voltage

Bit 6 **OVPnCOFM**: OVPn comparator normal operation or input offset voltage cancellation mode selection bit
0: Normal operation
1: Input offset voltage calibration mode

Bit 5 **OVPnCRS**: OVPn comparator input offset voltage calibration reference selection bit
0: Input reference voltage comes from negative input
1: Input reference voltage comes from positive input

This bit is used to select that the reference input voltage comes from the OVPn D/A converter or external input. Note that this bit is only available when the OVPn comparator input offset voltage calibration mode is selected by setting the OVPnCOFM bit to 1.

Bit 4~0 **OVPnCOF4~OVPnCOF0**: OVPn comparator input offset voltage calibration control bits

• OUVPnC3 Register

Bit	7	6	5	4	3	2	1	0
Name	UVPnO	UVPnCOFM	UVPnCRS	UVPnCOF4	UVPnCOF3	UVPnCOF2	UVPnCOF1	UVPnCOF0
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	1	0	0	0	0

Bit 7 **UVPnO**: UVPn comparator output bit
0: Positive input voltage < negative input voltage
1: Positive input voltage > negative input voltage

Bit 6 **UVPnCOFM**: UVPn comparator normal operation or input offset voltage cancellation mode selection bit
0: Normal operation
1: Input offset voltage calibration mode

Bit 5 **UVPnCRS**: UVPn comparator input offset voltage calibration reference selection bit
0: Input reference voltage comes from negative input
1: Input reference voltage comes from positive input

This bit is used to select that the reference input voltage comes from the UVPn D/A converter or external input. Note that this bit is only available when the UVPn comparator input offset voltage calibration mode is selected by setting the UVPnCOFM bit to 1.

Bit 4~0 **UVPnCOF4~UVPnCOF0**: UVPn comparator input offset voltage calibration control bits

• OUVPnPC Register

Bit	7	6	5	4	3	2	1	0
Name	UVPn1LEN	UVPn1HEN	UVPn0LEN	UVPn0HEN	OVPn1LEN	OVPn1HEN	OVPn0LEN	OVPn0HEN
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 **UVPn1LEN**: OUT1L Under Voltage Protection n Enable control
0: Disable
1: Enable

This bit is used to control the OUT1L signal output when the under voltage protection condition occurs. If this bit is cleared to 0 the function is disabled. This means the OUT1L output will not be affected when UVPn occurs. If set this bit high, the function is enabled. This means when an UVPn condition occurs the OUT1L status is controlled by the OUT1LS bit in the OUTPC register.

Bit 6 **UVPn1HEN**: OUT1H Under Voltage Protection n Enable control
0: Disable
1: Enable

This bit is used to control the OUT1H signal output when the under voltage protection condition occurs. If clear this bit to 0, the function is disabled. This means the OUT1H output will not be affected when UVPn occurs. If set this bit high, the function is enabled. This means when an UVPn condition occurs the OUT1H status is controlled by the OUT1HS bit in the OUTPC register.

Bit 5 **UVPn0LEN**: OUT0L Under Voltage Protection n Enable control
0: Disable
1: Enable

This bit is used to control the OUT0L signal output when the under voltage protection condition occurs. If clear this bit to 0, the function is disabled. This means the OUT0L output will not be affected when UVPn occurs. If set this bit high, the function is enabled. This means when an UVPn condition occurs the OUT0L status is controlled by the OUT0LS bit in the OUTPC register.

Bit 4 **UVPn0HEN**: OUT0H Under Voltage Protection n Enable control
0: Disable
1: Enable

This bit is used to control the OUT0H signal output when the under voltage protection condition occurs. If clear this bit to 0, the function is disabled. This means the OUT0H output will not be affected when UVPn occurs. If set this bit high, the function is enabled. This means when an UVPn condition occurs the OUT0H status is controlled by the OUT0HS bit in the OUTPC register.

Bit 3 **OVPn1LEN**: OUT1L Over Voltage Protection n Enable control
0: Disable
1: Enable

This bit is used to control the OUT1L signal output when the over voltage protection condition occurs. If clear this bit to 0, the function is disabled. This means the OUT1L output will not be affected when OVPn occurs. If set this bit high, the function is enabled. This means when an OVPn condition occurs the OUT1L status is controlled by the OUT1LS bit in the OUTPC register.

Bit 2 **OVPn1HEN**: OUT1H Over Voltage Protection n Enable control
0: Disable
1: Enable

This bit is used to control the OUT1H signal output when the over voltage protection condition occurs. If clear this bit to 0, the function is disabled. This means the OUT1H output will not be affected when OVPn occurs. If set this bit high, the function is enabled. This means when an OVPn condition occurs the OUT1H status is controlled by the OUT1HS bit in the OUTPC register.

Bit 1 **OVPn0LEN**: OUT0L Over Voltage Protection n Enable control
0: Disable
1: Enable

This bit is used to control the OUT0L signal output when the over voltage protection condition occurs. If clear this bit to 0, the function is disabled. This means the OUT0L output will not be affected when OVPn occurs. If set this bit high, the function is enabled. This means when an OVPn condition occurs the OUT0L status is controlled by the OUT0LS bit in the OUTPC register.

Bit 0 **OVPn0HEN**: OUT0H Over Voltage Protection n Enable control
0: Disable
1: Enable

This bit is used to control the OUT0H signal output when the over voltage protection condition occurs. If clear this bit to 0, the function is disabled. This means the OUT0H output will not be affected when OVPn occurs. If set this bit high, the function is enabled. This means when an OVPn condition occurs the OUT0H status is controlled by the OUT0HS bit in the OUTPC register.

OVPn and UVPn Comparator Offset calibration

The OVPn and UVPn circuits provide comparator offset calibration function. Before offset calibration, the hysteresis voltage should be zero by clearing the OVPnCHY or UVPnCHY bit to zero. As the OUVPn input pins are pin-shared with other functions, the OUVPn pin function must first be setup as comparator input using the corresponding pin-shared function selection register bits. The following content are the steps for the OVPn or UVPn comparator calibration.

OVPn Comparator calibration

Step 1. Set OVPnCOFM=1, OVPnCRS=1, comparator is now under offset calibration mode. To make sure V_{OS} as minimize as possible after calibration, the input reference voltage in calibration should be the same as input DC operating voltage in normal mode operation.

Step 2. Set OVPnCOF[4:0]=00000B and then read the OVPnO bit

Step 3. Increase the OVPnCOF[4:0] by 1 and then read the OVPnO bit

If the OVPnO bit state has not changed, then repeat Step 3 until the OVPnO bit state has changed.

If the OVPnO bit state has changed, record the OVPnCOF[4:0] value as V_{OS1} and then go to Step 4.

Step 4. Set OVPnCOF[4:0]=11111B and then read the OVPnO bit

Step 5. Decrease the OVPnCOF[4:0] value by 1 and then read the OVPnO bit

If the OVPnO bit state has not changed, then repeat Step 5 until the OVPnO bit state has changed.

If the OVPnO bit state has changed, record the OVPnCOF[4:0] value as V_{OS2} and then go to Step 6.

Step 6. Restore the $V_{OS}=(V_{OS1}+V_{OS2})/2$ to OVPnCOF[4:0] bit field, the calibration procedure is now finished.

If $(V_{OS1}+V_{OS2})/2$ is not integral, discard the decimal.

Residue $V_{OS}=V_{OUT}-V_{IN}$ (1)

UVPn Comparator calibration

Step 1. Set UVPnCOFM=1, UVPnCRS=1, comparator is now under offset calibration mode. To make sure V_{OS} as minimize as possible after calibration, the input reference voltage in calibration should be the same as input DC operating voltage in normal mode operation.

Step 2. Set UVPnCOF[4:0]=00000B and then read the UVPnO bit

Step 3. Increase the UVPnCOF[4:0] by 1 and then read the UVPnO bit

If the UVPnO bit state has not changed, then repeat Step 3 until the UVPnO bit state has changed.

If the UVPnO bit state has changed, record the UVPnCOF[4:0] value as V_{OS1} and then go to Step 4.

Step 4. Set UVPnCOF[4:0]=11111B and then read the UVPnO bit

Step 5. Decrease the UVPnCOF[4:0] value by 1 and then read the UVPnO bit

If the UVPnO bit state has not changed, then repeat Step 5 until the UVPnO bit state has changed.

If the UVPnO bit state has changed, record the UVPnCOF[4:0] value as V_{OS2} and then go to Step 6.

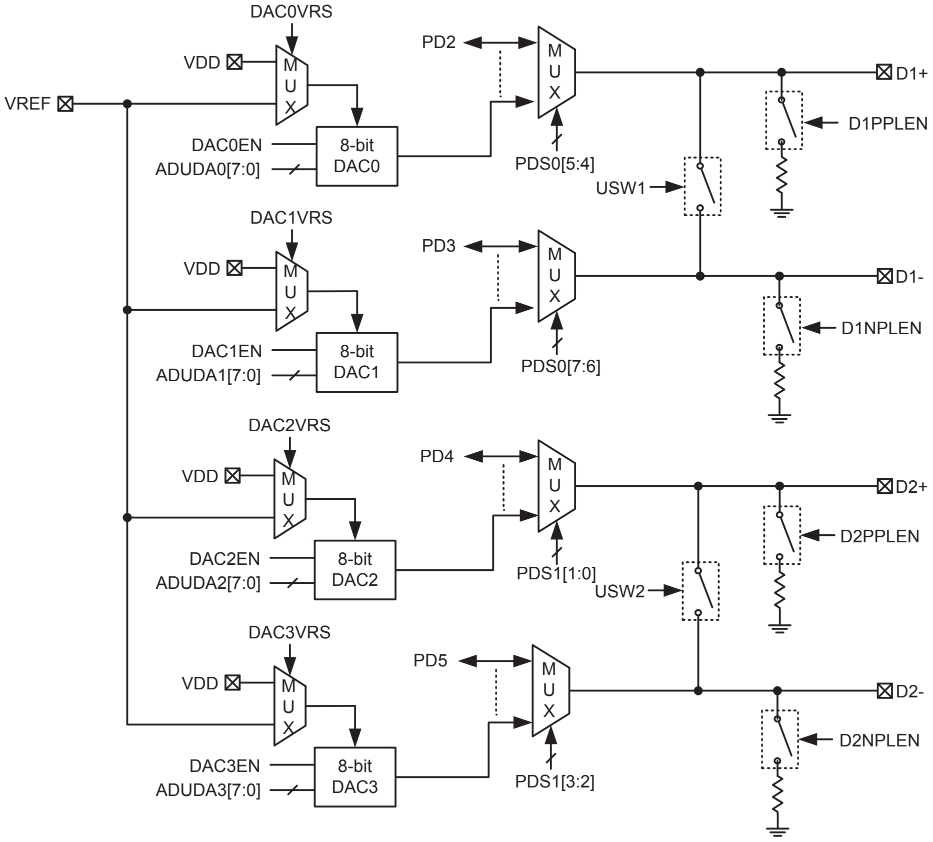
Step 6. Restore the $V_{OS}=(V_{OS1}+V_{OS2})/2$ to UVPnCOF[4:0] bit field, the calibration procedure is now finished.

If $(V_{OS1}+V_{OS2})/2$ is not integral, discard the decimal.

Residue $V_{OS}=V_{OUT}-V_{IN}$ (2)

USB Auto Detection

The device includes two USB ports named D1+/D1- and D2+/D2- to implement the Charge/Discharge Devices Auto Detection function. Users can distinguish the device connected to the USB ports is a dedicated charger, portable device, general USB interface or charging device with USB interface by monitoring the voltage and current of the connected USB lines.



USB Auto Detection Block Diagram

D1+/D1- and D2+/D2- for Auto Detection

There four 8-bit D/A Converters, DACn, which is enabled by the DACnEN bits in the ADUC0 register. The D/A Converter output signal is controlled by the ADUDAn register value and the reference voltage which is selected by the DACnVRS bit in the ADUC0 register. There is an analog switch connected between the D1+ and D1- lines, which is controlled by the USW1 bit. Similarly, there is an analog switch connected between the D2+ and D2- lines, which is controlled by the USW2 bit. But it needs to note that only when one of the D1+ and D1- or D2+ and D2- pins is in the analog or digital input type by setting the Pin-shared Function register, and then set the USW1 or USW2 bit high, can the switch be on. The D1+/D1- and D2+/D2- lines are individually connected a pull low resistor respectively to VSS which are controlled by the D1NPLEN/D1PPLEN and D2NPLEN/D2PPLEN bits in the ADUC1 register.

USB Auto Detection Registers

Overall operation of the USB auto detection function is controlled using several registers.

Register Name	Bit							
	7	6	5	4	3	2	1	0
ADUC0	DAC3VRS	DAC2VRS	DAC1VRS	DAC0VRS	DAC3EN	DAC2EN	DAC1EN	DAC0EN
ADUC1	—	—	D2NPLEN	D2PPLEN	D1NPLEN	D1PPLEN	USW2	USW1
ADUDA0	D7	D6	D5	D4	D3	D2	D1	D0
ADUDA1	D7	D6	D5	D4	D3	D2	D1	D0
ADUDA2	D7	D6	D5	D4	D3	D2	D1	D0
ADUDA3	D7	D6	D5	D4	D3	D2	D1	D0

USB Auto Detection Register List

• ADUC0 Register

Bit	7	6	5	4	3	2	1	0
Name	DAC3VRS	DAC2VRS	DAC1VRS	DAC0VRS	DAC3EN	DAC2EN	DAC1EN	DAC0EN
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 **DAC3VRS**: DAC3 reference voltage selection
0: From VDD pin
1: From VREF pin

Bit 6 **DAC2VRS**: DAC2 reference voltage selection
0: From VDD pin
1: From VREF pin

Bit 5 **DAC1VRS**: DAC1 reference voltage selection
0: From VDD pin
1: From VREF pin

Bit 4 **DAC0VRS**: DAC0 reference voltage selection
0: From VDD pin
1: From VREF pin

Bit 3 **DAC3EN**: DAC3 enable Control
0: Disable
1: Enable

Bit 2 **DAC2EN**: DAC2 enable Control
0: Disable
1: Enable

Bit 1 **DAC1EN**: DAC1 enable Control
0: Disable
1: Enable

Bit 0 **DAC0EN**: DAC0 enable Control
0: Disable
1: Enable

- **ADUC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	D2NPLEN	D2PPLEN	D1NPLEN	D1PPLEN	USW2	USW1
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 **D2NPLEN**: D2- pin Pull-Low Control
0: Disable
1: Enable

Bit 4 **D2PPLEN**: D2+ pin Pull-Low Control
0: Disable
1: Enable

Bit 3 **D1NPLEN**: D1- pin Pull-Low Control
0: Disable
1: Enable

Bit 2 **D1PPLEN**: D1+ pin Pull-Low Control
0: Disable
1: Enable

Bit 1 **USW2**: USW2 switch control
0: Switch off
1: Switch on

This bit controls the USW2 switch on/off. But only when one of the D2+ and D2- pins is used as the analog or digital input pin by setting the Pin-shared Function register, and then set the USW2 bit high, can the switch be on.

Bit 0 **USW1**: USW1 switch control
0: Switch off
1: Switch on

This bit controls the USW1 switch on/off. But only when one of the D1+ and D1- pins is used as the analog or digital input pin by setting the Pin-shared Function register, and then set the USW1 bit high, can the switch be on.

- **ADUDA0 Register**

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

Bit 7~0 **D7~D0**: 8-bit DAC0 Output Control Data Bits
DAC0 Output=(DAC0 Reference Voltage)×(ADUDA0[7:0])/256

- **ADUDA1 Register**

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

Bit 7~0 **D7~D0**: 8-bit DAC1 Output Control Data Bits
DAC1 Output=(DAC1 Reference Voltage)×(ADUDA1[7:0])/256

- **ADUDA2 Register**

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

Bit 7~0 **D7~D0**: 8-bit DAC2 Output Control Data Bits
DAC2 Output=(DAC2 Reference Voltage)×(ADUDA2[7:0])/256

- **ADUDA3 Register**

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

Bit 7~0 **D7~D0**: 8-bit DAC3 Output Control Data Bits
DAC3 Output=(DAC3 Reference Voltage)×(ADUDA3[7:0])/256

Low Voltage Detector – LVD

The 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 five 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 Detected
1: Low Voltage Detected

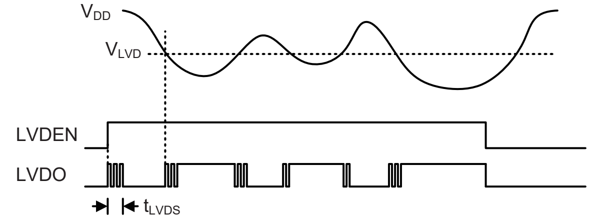
Bit 4 **LVDEN**: Low Voltage Detector Enable control
0: Disable
1: Enable

Bit 3 Unimplemented, read as “0”

Bit 2~0 **VLVD2~VLVD0**: LVD Voltage selection
000: Undefined
001: Undefined
010: Undefined
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 2.7V and 4.0V. When the power supply voltage, V_{DD}, falls below this pre-determined value, the LVDO bit will be set high indicating a low power supply voltage condition. When the device is in the IDLE/SLEEP mode 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, providing an alternative means of low voltage detection, in addition to polling the LVDO bit. The interrupt will only be generated after a delay of t_{LVD} after the LVDO bit has been set high by a low voltage condition. In this case, the LVF interrupt request flag will be set, causing an interrupt to be generated if V_{DD} falls below the preset LVD voltage. This will cause the device to wake-up from the 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.

Interrupts

Interrupts are an important part of any microcontroller system. When an external event or an internal function such as a Timer Module or an A/D converter requires microcontroller attention, their corresponding interrupt will enforce a temporary suspension of the main program allowing the microcontroller to direct attention to their respective needs. The device contains several external interrupt and internal interrupt functions. The external interrupts are generated by the action of the external INT0~INT2 pins, while the internal interrupts are generated by various internal functions such as the TMs, Time Bases, OCPs, OVPs, UVPs, LVD, EEPROM and the A/D converter, etc.

Interrupt Registers

Overall interrupt control, which basically means the setting of request flags when certain microcontroller conditions occur and the setting of interrupt enable bits by the application program, is controlled by a series of registers, located in the Special Purpose Data Memory, as shown in the accompanying table. The number of registers falls into three categories. The first is the INTC0~INTC3 registers which setup the primary interrupts, the second is the MFI0~MFI1 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=0~2
Over Current Protection	OCPnE	OCPnF	n=0~1
Under Voltage Protection	UVPnE	UVPnF	n=0~1
Over Voltage Protection	OVPnE	OVPnF	n=0~1
Multi-function	MFnE	MFnF	n=0~1
LVD	LVE	LVF	—
Time Base	TBnE	TBnF	n=0~1
A/D Converter	ADE	ADF	—
EEPROM	DEE	DEF	—
STM	STMPE	STMPF	—
	STMAE	STMAF	
PTM	PTMPE	PTMPF	—
	PTMAE	PTMAF	

Interrupt Register Bit Naming Conventions

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	INT2S1	INT2S0	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	OVP0F	OCP1F	OCP0F	OVP0E	OCP1E	OCP0E	EMI
INTC1	INT0F	UVP1F	UVP0F	OVP1F	INT0E	UVP1E	UVP0E	OVP1E
INTC2	MF1F	MF0F	INT2F	INT1F	MF1E	MF0E	INT2E	INT1E
INTC3	—	TB1F	TB0F	LVF	—	TB1E	TB0E	LVE
MFI0	—	DEF	STMAF	STMPF	—	DEE	STMAE	STMPE
MFI1	—	ADF	PTMAF	PTMPF	—	ADE	PTMAE	PTMPE

Interrupt Register List

- **INTEG Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	INT2S1	INT2S0	INT1S1	INT1S0	INT0S1	INT0S0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 Unimplemented, read as “0”

Bit 5~4 **INT2S1~INT2S0**: Interrupt edge control for INT2 pin
00: Disable
01: Rising edge
10: Falling edge
11: Rising and falling edges

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	—	OVP0F	OCP1F	OCP0F	OVP0E	OCP1E	OCP0E	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 **OVP0F**: Over Voltage Protection 0 interrupt request flag
0: No request
1: Interrupt request

Bit 5 **OCP1F**: Over Current Protection 1 interrupt request flag
0: No request
1: Interrupt request

Bit 4 **OCP0F**: Over Current Protection 0 interrupt request flag
0: No request
1: Interrupt request

Bit 3 **OVP0E**: Over Voltage Protection 0 interrupt control
0: Disable
1: Enable

Bit 2 **OCP1E**: Over Current Protection 1 interrupt control
0: Disable
1: Enable

Bit 1 **OCP0E**: Over Current Protection 0 interrupt control
0: Disable
1: Enable

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

- **INTC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	INT0F	UVP1F	UVP0F	OVP1F	INT0E	UVP1E	UVP0E	OVP1E
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 **INT0F**: INT0 interrupt request flag
0: No request
1: Interrupt request

Bit 6 **UVP1F**: Under Voltage Protection 1 interrupt request flag
0: No request
1: Interrupt request

Bit 5 **UVP0F**: Under Voltage Protection 0 interrupt request flag
0: No request
1: Interrupt request

Bit 4 **OVP1F**: Over Voltage Protection 1 interrupt request flag
0: No request
1: Interrupt request

Bit 3 **INT0E**: INT0 interrupt control
0: Disable
1: Enable

Bit 2 **UVP1E**: Under Voltage Protection 1 interrupt control
0: Disable
1: Enable

Bit 1 **UVP0E**: Under Voltage Protection 0 interrupt control
0: Disable
1: Enable

Bit 0 **OVP1E**: Over Voltage Protection 1 interrupt control
0: Disable
1: Enable

- **INTC2 Register**

Bit	7	6	5	4	3	2	1	0
Name	MF1F	MF0F	INT2F	INT1F	MF1E	MF0E	INT2E	INT1E
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **MF1F**: Multi-function interrupt 1 request flag
0: No request
1: Interrupt request

Bit 6 **MF0F**: Multi-function interrupt 0 request flag
0: No request
1: Interrupt request

Bit 5 **INT2F**: INT2 interrupt request flag
0: No request
1: Interrupt request

Bit 4 **INT1F**: INT1 interrupt request flag
0: No request
1: Interrupt request

Bit 3 **MF1E**: Multi-function interrupt 1 control
0: Disable
1: Enable

Bit 2 **MF0E**: Multi-function interrupt 0 control
0: Disable
1: Enable

Bit 1 **INT2E**: INT2 interrupt control
0: Disable
1: Enable

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

• INTC3 Register

Bit	7	6	5	4	3	2	1	0
Name	—	TB1F	TB0F	LVF	—	TB1E	TB0E	LVE
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

Bit 7 Unimplemented, read as “0”

Bit 6 **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 **LVF**: LVD interrupt request flag
0: No request
1: Interrupt request

Bit 3 Unimplemented, read as “0”

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 **LVE**: LVD interrupt control
0: Disable
1: Enable

• MFI0 Register

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

Bit 7 Unimplemented, read as “0”

Bit 6 **DEF**: Data EEPROM interrupt request flag
0: No request
1: Interrupt request

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

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

Bit 3 Unimplemented, read as "0"

Bit 2 **DEE**: Data EEPROM interrupt control
0: Disable
1: Enable

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

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

• MFI1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	ADF	PTMAF	PTMPF	—	ADE	PTMAE	PTMPE
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **ADF**: A/D Converter interrupt request flag
0: No request
1: Interrupt request

Bit 5 **PTMAF**: PTM Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **PTMPF**: PTM Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 3 Unimplemented, read as "0"

Bit 2 **ADE**: A/D Converter interrupt control
0: Disable
1: Enable

Bit 1 **PTMAE**: PTM Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **PTMPE**: PTM Comparator P match interrupt control
0: Disable
1: Enable

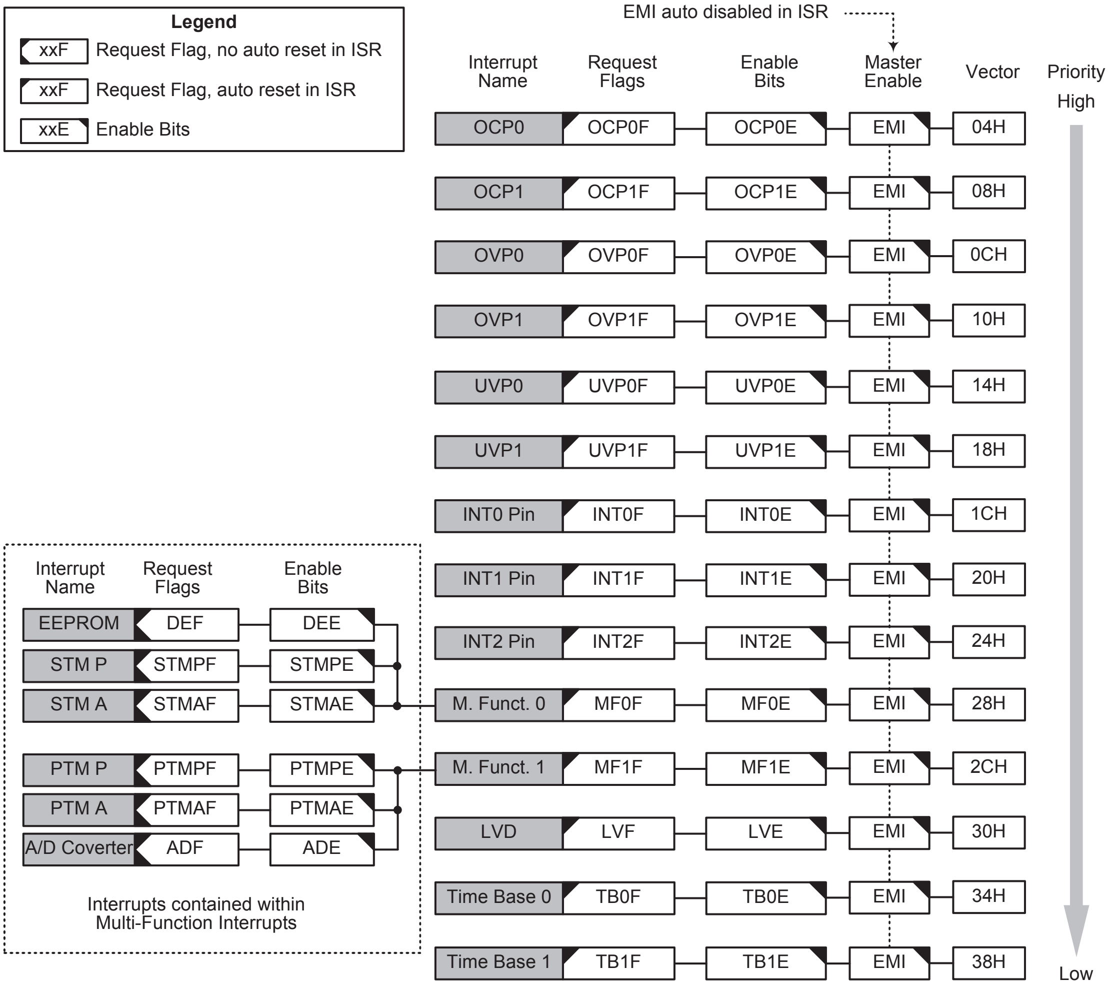
Interrupt Operation

When the conditions for an interrupt event occur, such as a TM Comparator P or Comparator A match or A/D conversion completion etc., the relevant interrupt request flag will be set. Whether the request flag actually generates a program jump to the relevant interrupt vector is determined by the condition of the interrupt enable bit. If the enable bit is set high then the program will jump to its relevant vector; if the enable bit is zero then although the interrupt request flag is set an actual interrupt will not be generated and the program will not jump to the relevant interrupt vector. The global interrupt enable bit, if cleared to zero, will disable all interrupts.

When an interrupt is generated, the Program Counter, which stores the address of the next instruction to be executed, will be transferred onto the stack. The Program Counter will then be loaded with a new address which will be the value of the corresponding interrupt vector. The microcontroller will then fetch its next instruction from this interrupt vector. The instruction at this vector will usually be a "JMP" which will jump to another section of program which is known as the interrupt service routine. Here is located the code to control the appropriate interrupt. The interrupt service routine must be terminated with a “RETI”, which retrieves the original Program Counter address from the stack and allows the microcontroller to continue with normal execution at the point where the interrupt occurred.

The various interrupt enable bits, together with their associated request flags, are shown in the accompanying diagrams with their order of priority. Some interrupt sources have their own individual vector while others share the same multi-function interrupt vector. Once an interrupt subroutine is serviced, all the other interrupts will be blocked, as the global interrupt enable bit, EMI bit will be cleared automatically. This will prevent any further interrupt nesting from occurring. However, if other interrupt requests occur during this interval, although the interrupt will not be immediately serviced, the request flag will still be recorded.

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



Interrupt Structure

External Interrupts

The external interrupts are controlled by signal transitions on the pins INT0~INT2. An external interrupt request will take place when the external interrupt request flags, INT0F~INT2F, 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~INT2E, must first be set. Additionally the correct interrupt edge type must be selected using the INTEG register to enable the external interrupt function and to choose the trigger edge type. As the external interrupt pins are pin-shared with I/O pins, they can only be configured as external interrupt pins if their external interrupt enable bit in the corresponding interrupt register has been set and the external interrupt pin is selected by the corresponding pin-shared function selection bits. The pin must also be setup as an input by setting the corresponding bit in the port control register. When the interrupt is enabled, the stack is not full and the correct transition type appears on the external interrupt pin, a subroutine call to the external interrupt vector, will take place. When the interrupt is serviced, the external interrupt request flags, INT0F~INT2F, 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.

Over Current Protection Interrupts

The OCPn Interrupt is controlled by detecting the OCPn input current. An OCPn Interrupt request will take place when the OCPn Interrupt request flag, OCPnF, is set, which occurs when a large current is detected. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and OCPn Interrupt enable bit, OCPnE, must first be set. When the interrupt is enabled, the stack is not full and an over current is detected, a subroutine call to the OCPn Interrupt vector, will take place. When the interrupt is serviced, the OCPn Interrupt flag, OCPnF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

Over Voltage Protection Interrupts

The OVPn Interrupt is controlled by detecting the OVPn input voltage. An OVPn Interrupt request will take place when the OVPn Interrupt request flag, OVPnF, is set, which occurs when the Over Voltage Protection circuit detects an over voltage condition. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and OVPn Interrupt enable bit, OVPnE, must first be set. When the interrupt is enabled, the stack is not full and an over voltage is detected, a subroutine call to the OVPn Interrupt vector, will take place. When the interrupt is serviced, the OVPn Interrupt flag, OVPnF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

Under Voltage Protection Interrupts

The UVPn Interrupt is controlled by detecting the UVPn input voltage. An UVPn Interrupt request will take place when the UVPn Interrupt request flag, UVPnF, is set, which occurs when the Under Voltage Protection circuit detects an under voltage condition. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and UVPn Interrupt enable bit, UVPnE, must first be set. When the interrupt is enabled, the stack is not full and an under voltage is detected, a subroutine call to the UVPn Interrupt vector, will take place. When the interrupt is serviced, the UVPn Interrupt flag, UVPnF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

Multi-function Interrupts

Within the device there are up to two 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, EEPROM Interrupt and A/D converter 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 will not be automatically reset and must be manually reset by the application program.

Timer Module Interrupts

Each of the Standard Type TM and Periodic Type TM has two interrupts. All of the TM interrupts are contained within the Multi-function Interrupts. For the Standard Type TM and the Periodic Type TM, each has two interrupt request flags of STMPF, STMAF and PTMPF, PTMAF and two enable bits of STMPE, STMAE and PTMPE, PTMAE. 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.

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.

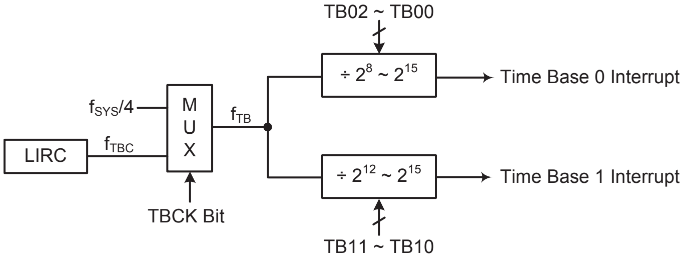
A/D Converter Interrupt

The A/D Converter Interrupt is also contained within the Multi-function 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, and associated Multi-function interrupt enable bit, 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 A/D Converter 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 A/D Converter Interrupt flag bit, ADF will not be automatically cleared, it has to be cleared 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. Its clock source, f_{TB}, originates from the internal clock source $f_{SYS}/4$ or f_{TBC} and then passes through a divider, the division ratio of which is selected by programming the appropriate bits in the TBC registers to obtain longer interrupt periods whose value ranges.



Time Base Interrupts

• 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**: Time Base 0 and Time Base 1 Control bit
0: Disable
1: Enable

Bit 6 **TBCK**: f_{TB} Clock Source Selection
0: f_{TBC}
1: $f_{SYS}/4$

Bit 5~4 **TB11~TB10**: Select Time Base 1 Time-out Period
00: $2^{12}/f_{TB}$
01: $2^{13}/f_{TB}$
10: $2^{14}/f_{TB}$
11: $2^{15}/f_{TB}$

Bit 3 Unimplemented, read as “0”

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

LVD Interrupt

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

Interrupt Wake-up Function

Each of the interrupt functions has the capability of waking up the microcontroller when in the SLEEP or IDLE Mode. A wake-up is generated when an interrupt request flag changes from low to high and is independent of whether the interrupt is enabled or not. Therefore, even though the device is in the SLEEP or IDLE Mode and its system oscillator stopped, situations such as external edge transitions on the external interrupt pins or a low power supply voltage may cause their respective interrupt flag to be set high and consequently generate an interrupt. Care must therefore be taken if spurious wake-up situations are to be avoided. If an interrupt wake-up function is to be disabled then the corresponding interrupt request flag should be set high before the device enters the SLEEP or IDLE Mode. The interrupt enable bits have no effect on the interrupt wake-up function.

Programming Considerations

By disabling the relevant interrupt enable bits, a requested interrupt can be prevented from being serviced, however, once an interrupt request flag is set, it will remain in this condition in the interrupt register until the corresponding interrupt is serviced or until the request flag is cleared by the application program.

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

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

Every interrupt has the capability of waking up the microcontroller when it is in the SLEEP or IDLE Mode, the wake up being generated when the interrupt request flag changes from low to high. If it is required to prevent a certain interrupt from waking up the microcontroller then its respective request flag should be first set high before enter SLEEP or IDLE Mode.

As only the Program Counter is pushed onto the stack, then when the interrupt is serviced, if the contents of the accumulator, status register or other registers are altered by the interrupt service program, their contents should be saved to the memory at the beginning of the interrupt service routine.

To return from an interrupt subroutine, either a RET or RETI instruction may be executed. The RETI instruction in addition to executing a return to the main program also automatically sets the EMI bit high to allow further interrupts. The RET instruction however only executes a return to the main program leaving the EMI bit in its present zero state and therefore disabling the execution of further interrupts.

Configuration Options

Configuration options refer to certain options within the MCU that are programmed into the device during the programming process. During the development process, these options are selected using the HT-IDE software development tools. 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 Option	
1	HIRC Frequency Selection – f_{HIRC}: 8MHz, 12MHz or 16MHz

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 several 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 such as 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 setup as a table where data can be directly stored. A set of easy to use instructions provides the means by which this fixed data can be referenced and retrieved from the Program Memory.

Other Operations

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

Instruction Set Summary

The instructions related to the data memory access in the following table can be used when the desired data memory is located in Data Memory sector 0.

Table Conventions

x: Bits immediate data
m: Data Memory address
A: Accumulator
i: 0~7 number of bits
addr: Program memory address

Mnemonic	Description	Cycles	Flag Affected
Arithmetic			
ADD A,[m]	Add Data Memory to ACC	1	Z, C, AC, OV, SC
ADDM A,[m]	Add ACC to Data Memory	1^{Note}	Z, C, AC, OV, SC
ADD A,x	Add immediate data to ACC	1	Z, C, AC, OV, SC
ADC A,[m]	Add Data Memory to ACC with Carry	1	Z, C, AC, OV, SC
ADCM A,[m]	Add ACC to Data memory with Carry	1^{Note}	Z, C, AC, OV, SC
SUB A,x	Subtract immediate data from the ACC	1	Z, C, AC, OV, SC, CZ
SUB A,[m]	Subtract Data Memory from ACC	1	Z, C, AC, OV, SC, CZ
SUBM A,[m]	Subtract Data Memory from ACC with result in Data Memory	1^{Note}	Z, C, AC, OV, SC, CZ
SBC A,x	Subtract immediate data from ACC with Carry	1	Z, C, AC, OV, SC, CZ
SBC A,[m]	Subtract Data Memory from ACC with Carry	1	Z, C, AC, OV, SC, CZ
SBCM A,[m]	Subtract Data Memory from ACC with Carry, result in Data Memory	1^{Note}	Z, C, AC, OV, SC, CZ
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]	Skip if Data Memory is not 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
TABRDL [m]	Read table (last page) to TBLH and Data Memory	2^{Note}	None
ITABRD [m]	Increment table pointer TBLP first and Read table to TBLH and Data Memory	2^{Note}	None
ITABRDL [m]	Increment table pointer TBLP first and Read table (last page) to TBLH and Data Memory	2^{Note}	None
Miscellaneous			
NOP	No operation	1	None
CLR [m]	Clear Data Memory	1^{Note}	None
SET [m]	Set Data Memory	1^{Note}	None
CLR WDT	Clear Watchdog Timer	1	TO, PDF
SWAP [m]	Swap nibbles of Data Memory	1^{Note}	None
SWAPA [m]	Swap nibbles of Data Memory with result in ACC	1	None
HALT	Enter power down mode	1	TO, PDF

Note: 1. For skip instructions, if the result of the comparison involves a skip then up to three cycles are required, if no skip takes place only one cycle is required.

2. Any instruction which changes the contents of the PCL will also require 2 cycles for execution.

3. For the “CLR WDT” instruction the TO and PDF flags may be affected by the execution status. The TO and PDF flags are cleared after the “CLR WDT” instructions is executed. Otherwise the TO and PDF flags remain unchanged.

Extended Instruction Set

The extended instructions are used to support the full range address access for the data memory. When the accessed data memory is located in any data memory sections except sector 0, the extended instruction can be used to access the data memory instead of using the indirect addressing access to improve the CPU firmware performance.

Mnemonic	Description	Cycles	Flag Affected
Arithmetic			
LADD A,[m]	Add Data Memory to ACC	2	Z, C, AC, OV, SC
LADDM A,[m]	Add ACC to Data Memory	2^{Note}	Z, C, AC, OV, SC
LADC A,[m]	Add Data Memory to ACC with Carry	2	Z, C, AC, OV, SC
LADCM A,[m]	Add ACC to Data memory with Carry	2^{Note}	Z, C, AC, OV, SC
LSUB A,[m]	Subtract Data Memory from ACC	2	Z, C, AC, OV, SC, CZ
LSUBM A,[m]	Subtract Data Memory from ACC with result in Data Memory	2^{Note}	Z, C, AC, OV, SC, CZ
LSBC A,[m]	Subtract Data Memory from ACC with Carry	2	Z, C, AC, OV, SC, CZ
LSBCM A,[m]	Subtract Data Memory from ACC with Carry, result in Data Memory	2^{Note}	Z, C, AC, OV, SC, CZ
LDAA [m]	Decimal adjust ACC for Addition with result in Data Memory	2^{Note}	C
Logic Operation			
LAND A,[m]	Logical AND Data Memory to ACC	2	Z
LOR A,[m]	Logical OR Data Memory to ACC	2	Z
LXOR A,[m]	Logical XOR Data Memory to ACC	2	Z
LANDM A,[m]	Logical AND ACC to Data Memory	2^{Note}	Z
LORM A,[m]	Logical OR ACC to Data Memory	2^{Note}	Z
LXORM A,[m]	Logical XOR ACC to Data Memory	2^{Note}	Z
LCPL [m]	Complement Data Memory	2^{Note}	Z
LCPLA [m]	Complement Data Memory with result in ACC	2	Z
Increment & Decrement			
LINCA [m]	Increment Data Memory with result in ACC	2	Z
LINC [m]	Increment Data Memory	2^{Note}	Z
LDECA [m]	Decrement Data Memory with result in ACC	2	Z
LDEC [m]	Decrement Data Memory	2^{Note}	Z
Rotate			
LRRA [m]	Rotate Data Memory right with result in ACC	2	None
LRR [m]	Rotate Data Memory right	2^{Note}	None
LRRCA [m]	Rotate Data Memory right through Carry with result in ACC	2	C
LRRC [m]	Rotate Data Memory right through Carry	2^{Note}	C
LRLA [m]	Rotate Data Memory left with result in ACC	2	None
LRL [m]	Rotate Data Memory left	2^{Note}	None
LRLCA [m]	Rotate Data Memory left through Carry with result in ACC	2	C
LRLC [m]	Rotate Data Memory left through Carry	2^{Note}	C
Data Move			
LMOV A,[m]	Move Data Memory to ACC	2	None
LMOV [m],A	Move ACC to Data Memory	2^{Note}	None
Bit Operation			
LCLR [m].i	Clear bit of Data Memory	2^{Note}	None
LSET [m].i	Set bit of Data Memory	2^{Note}	None

Mnemonic	Description	Cycles	Flag Affected
Branch			
LSZ [m]	Skip if Data Memory is zero	2[Note]	None
LSZA [m]	Skip if Data Memory is zero with data movement to ACC	2[Note]	None
LSNZ [m]	Skip if Data Memory is not zero	2[Note]	None
LSZ [m].i	Skip if bit i of Data Memory is zero	2[Note]	None
LSNZ [m].i	Skip if bit i of Data Memory is not zero	2[Note]	None
LSIZ [m]	Skip if increment Data Memory is zero	2[Note]	None
LSDZ [m]	Skip if decrement Data Memory is zero	2[Note]	None
LSIZA [m]	Skip if increment Data Memory is zero with result in ACC	2[Note]	None
LSDZA [m]	Skip if decrement Data Memory is zero with result in ACC	2[Note]	None
Table Read			
LTABRD [m]	Read table to TBLH and Data Memory	3[Note]	None
LTABRDL [m]	Read table (last page) to TBLH and Data Memory	3[Note]	None
LITABRD [m]	Increment table pointer TBLP first and Read table to TBLH and Data Memory	3[Note]	None
LITABRDL [m]	Increment table pointer TBLP first and Read table (last page) to TBLH and Data Memory	3[Note]	None
Miscellaneous			
LCLR [m]	Clear Data Memory	2[Note]	None
LSET [m]	Set Data Memory	2[Note]	None
LSWAP [m]	Swap nibbles of Data Memory	2[Note]	None
LSWAPA [m]	Swap nibbles of Data Memory with result in ACC	2	None

Note: 1. For these extended skip instructions, if the result of the comparison involves a skip then up to four cycles are required, if no skip takes place two cycles is required.

2. Any extended instruction which changes the contents of the PCL register will also require three cycles for execution.

Instruction Definition

ADC A,[m] Add Data Memory to ACC with Carry
Description The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the Accumulator.
Operation ACC ← ACC + [m] + C
Affected flag(s) OV, Z, AC, C, SC

ADCM A,[m] Add ACC to Data Memory with Carry
Description The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the specified Data Memory.
Operation [m] ← ACC + [m] + C
Affected flag(s) OV, Z, AC, C, SC

ADD A,[m] Add Data Memory to ACC
Description The contents of the specified Data Memory and the Accumulator are added. The result is stored in the Accumulator.
Operation ACC ← ACC + [m]
Affected flag(s) OV, Z, AC, C, SC

ADD A,x Add immediate data to ACC
Description The contents of the Accumulator and the specified immediate data are added. The result is stored in the Accumulator.
Operation ACC ← ACC + x
Affected flag(s) OV, Z, AC, C, SC

ADDM A,[m] Add ACC to Data Memory
Description The contents of the specified Data Memory and the Accumulator are added. The result is stored in the specified Data Memory.
Operation [m] ← ACC + [m]
Affected flag(s) OV, Z, AC, C, SC

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

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

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

CALL addr Subroutine call

Description Unconditionally calls a subroutine at the specified address. The Program Counter then increments by 1 to obtain the address of the next instruction which is then pushed onto the stack. The specified address is then loaded and the program continues execution from this new address. As this instruction requires an additional operation, it is a two cycle instruction.

Operation Stack ← Program Counter + 1
Program Counter ← addr

Affected flag(s) None

CLR [m] Clear Data Memory

Description Each bit of the specified Data Memory is cleared to 0.

Operation [m] ← 00H

Affected flag(s) None

CLR [m].i Clear bit of Data Memory

Description Bit i of the specified Data Memory is cleared to 0.

Operation [m].i ← 0

Affected flag(s) None

CLR WDT Clear Watchdog Timer

Description The TO, PDF flags and the WDT are all cleared.

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CPL [m] Complement Data Memory

Description Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa.

Operation $[m] \leftarrow \overline{[m]}$

Affected flag(s) Z

CPLA [m] Complement Data Memory with result in ACC

Description Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa. The complemented result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation $ACC \leftarrow \overline{[m]}$

Affected flag(s) Z

DAA [m] Decimal-Adjust ACC for addition with result in Data Memory

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

Operation [m] ← ACC + 00H or
[m] ← ACC + 06H or
[m] ← ACC + 60H or
[m] ← ACC + 66H

Affected flag(s) C

DEC [m] Decrement Data Memory

Description Data in the specified Data Memory is decremented by 1.

Operation [m] ← [m] − 1

Affected flag(s) Z

DECA [m] Decrement Data Memory with result in ACC

Description Data in the specified Data Memory is decremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC ← [m] − 1

Affected flag(s) Z

HALT Enter power down mode

Description This instruction stops the program execution and turns off the system clock. The contents of the Data Memory and registers are retained. The WDT and prescaler are cleared. The power down flag PDF is set and the WDT time-out flag TO is cleared.

Operation TO ← 0
PDF ← 1

Affected flag(s) TO, PDF

INC [m] Increment Data Memory

Description Data in the specified Data Memory is incremented by 1.

Operation [m] ← [m] + 1

Affected flag(s) Z

INCA [m] Increment Data Memory with result in ACC

Description Data in the specified Data Memory is incremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC ← [m] + 1

Affected flag(s) Z

JMP addr Jump unconditionally

Description The contents of the Program Counter are replaced with the specified address. Program execution then continues from this new address. As this requires the insertion of a dummy instruction while the new address is loaded, it is a two cycle instruction.

Operation Program Counter ← addr

Affected flag(s) None

MOV A,[m] Move Data Memory to ACC

Description The contents of the specified Data Memory are copied to the Accumulator.

Operation ACC ← [m]

Affected flag(s) None

MOV A,x Move immediate data to ACC

Description The immediate data specified is loaded into the Accumulator.

Operation ACC ← x

Affected flag(s) None

MOV [m],A Move ACC to Data Memory

Description The contents of the Accumulator are copied to the specified Data Memory.

Operation [m] ← ACC

Affected flag(s) None

NOP No operation

Description No operation is performed. Execution continues with the next instruction.

Operation No operation

Affected flag(s) None

OR A,[m] Logical OR Data Memory to ACC

Description Data in the Accumulator and the specified Data Memory perform a bitwise logical OR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″OR″ [m]

Affected flag(s) Z

OR A,x Logical OR immediate data to ACC

Description Data in the Accumulator and the specified immediate data perform a bitwise logical OR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″OR″ x

Affected flag(s) Z

ORM A,[m] Logical OR ACC to Data Memory

Description Data in the specified Data Memory and the Accumulator perform a bitwise logical OR operation. The result is stored in the Data Memory.

Operation [m] ← ACC ″OR″ [m]

Affected flag(s) Z

RET Return from subroutine

Description The Program Counter is restored from the stack. Program execution continues at the restored address.

Operation Program Counter ← Stack

Affected flag(s) None

RET A,x Return from subroutine and load immediate data to ACC

Description The Program Counter is restored from the stack and the Accumulator loaded with the specified immediate data. Program execution continues at the restored address.

Operation Program Counter ← Stack
ACC ← x

Affected flag(s) None

RETI Return from interrupt

Description The Program Counter is restored from the stack and the interrupts are re-enabled by setting the EMI bit. EMI is the master interrupt global enable bit. If an interrupt was pending when the RETI instruction is executed, the pending Interrupt routine will be processed before returning to the main program.

Operation Program Counter ← Stack
EMI ← 1

Affected flag(s) None

RL [m] Rotate Data Memory left

Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0.

Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← [m].7

Affected flag(s) None

RLA [m]

Rotate Data Memory left with result in ACC

Description
The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation
ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← [m].7

Affected flag(s)
None

RLC [m]

Rotate Data Memory left through Carry

Description
The contents of the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into bit 0.

Operation
[m].(i+1) ← [m].i; (i=0~6)
[m].0 ← C
C ← [m].7

Affected flag(s)
C

RLCA [m]

Rotate Data Memory left through Carry with result in ACC

Description
Data in the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into the bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation
ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← C
C ← [m].7

Affected flag(s)
C

RR [m]

Rotate Data Memory right

Description
The contents of the specified Data Memory are rotated right by 1 bit with bit 0 rotated into bit 7.

Operation
[m].i ← [m].(i+1); (i=0~6)
[m].7 ← [m].0

Affected flag(s)
None

RRA [m]

Rotate Data Memory right with result in ACC

Description
Data in the specified Data Memory is rotated right by 1 bit with bit 0 rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation
ACC.i ← [m].(i+1); (i=0~6)
ACC.7 ← [m].0

Affected flag(s)
None

RRC [m]

Rotate Data Memory right through Carry

Description
The contents of the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7.

Operation
[m].i ← [m].(i+1); (i=0~6)
[m].7 ← C
C ← [m].0

Affected flag(s)
C

RRCA [m] Rotate Data Memory right through Carry with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.i ← [m].(i+1); (i=0~6)
ACC.7 ← C
C ← [m].0

Affected flag(s) C

SBC A,[m] Subtract Data Memory from ACC with Carry

Description The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC − [m] − $\overline{C}$

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

SBC A, x Subtract immediate data from ACC with Carry

Description The immediate data and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC – [m] – $\overline{C}$

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

SBCM A,[m] Subtract Data Memory from ACC with Carry and result in Data Memory

Description The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation [m] ← ACC − [m] − $\overline{C}$

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

SDZ [m] Skip if decrement Data Memory is 0

Description The contents of the specified Data Memory are first decremented by 1. If the result is 0 the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation [m] ← [m] − 1
Skip if [m]=0

Affected flag(s) None

SDZA [m] Skip if decrement Data Memory is zero with result in ACC

Description The contents of the specified Data Memory are first decremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.

Operation ACC ← [m] − 1
Skip if ACC=0

Affected flag(s) None

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

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

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

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

SNZ [m].i Skip if Data Memory is not 0
Description If the specified Data Memory is not 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is 0 the program proceeds with the following instruction.
Operation Skip if [m].i ≠ 0
Affected flag(s) None

SNZ [m] Skip if Data Memory is not 0
Description If the specified Data Memory is not 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is 0 the program proceeds with the following instruction.
Operation Skip if [m]≠ 0
Affected flag(s) None

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

SUBM A,[m] Subtract Data Memory from ACC with result in Data Memory

Description The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation [m] ← ACC − [m]

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

SUB A,x Subtract immediate data from ACC

Description The immediate data specified by the code is subtracted from the contents of the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC − x

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

SWAP [m] Swap nibbles of Data Memory

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

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s) None

SWAPA [m] Swap nibbles of Data Memory with result in ACC

Description The low-order and high-order nibbles of the specified Data Memory are interchanged. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC.3~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0

Affected flag(s) None

SZ [m] Skip if Data Memory is 0

Description If the contents of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation Skip if [m]=0

Affected flag(s) None

SZA [m] Skip if Data Memory is 0 with data movement to ACC

Description The contents of the specified Data Memory are copied to the Accumulator. If the value is zero, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation ACC ← [m]
Skip if [m]=0

Affected flag(s) None

SZ [m].i Skip if bit i of Data Memory is 0

Description If bit i of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.

Operation Skip if [m].i=0

Affected flag(s) None

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

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

ITABRD [m]	Increment table pointer low byte first and read table to TBLH and Data Memory
Description	Increment table pointer low byte, TBLP, first and then the program code addressed by the table pointer (TBHP and TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.
Operation	[m] ← program code (low byte) TBLH ← program code (high byte)
Affected flag(s)	None

ITABRDL [m]	Increment table pointer low byte first and read table (last page) to TBLH and Data Memory
Description	Increment table pointer low byte, TBLP, first and then the low byte of the program code (last page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.
Operation	[m] ← program code (low byte) TBLH ← program code (high byte)
Affected flag(s)	None

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

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

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

Extended Instruction Definition

The extended instructions are used to directly access the data stored in any data memory sections.

LADC A,[m] Add Data Memory to ACC with Carry
Description The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the Accumulator.
Operation ACC ← ACC + [m] + C
Affected flag(s) OV, Z, AC, C, SC

LADCM A,[m] Add ACC to Data Memory with Carry
Description The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the specified Data Memory.
Operation [m] ← ACC + [m] + C
Affected flag(s) OV, Z, AC, C, SC

LADD A,[m] Add Data Memory to ACC
Description The contents of the specified Data Memory and the Accumulator are added. The result is stored in the Accumulator.
Operation ACC ← ACC + [m]
Affected flag(s) OV, Z, AC, C, SC

LADDM A,[m] Add ACC to Data Memory
Description The contents of the specified Data Memory and the Accumulator are added. The result is stored in the specified Data Memory.
Operation [m] ← ACC + [m]
Affected flag(s) OV, Z, AC, C, SC

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

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

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

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

LCPL [m] Complement Data Memory

Description: Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa.

Operation: $[m] \leftarrow \overline{[m]}$

Affected flag(s): Z

LCPLA [m] Complement Data Memory with result in ACC

Description: Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa. The complemented result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation: $ACC \leftarrow \overline{[m]}$

Affected flag(s): Z

LDAA [m] Decimal-Adjust ACC for addition with result in Data Memory

Description: Convert the contents of the Accumulator value to a BCD (Binary Coded Decimal) value resulting from the previous addition of two BCD variables. If the low nibble is greater than 9 or if AC flag is set, then a value of 6 will be added to the low nibble. Otherwise the low nibble remains unchanged. If the high nibble is greater than 9 or if the C flag is set, then a value of 6 will be added to the high nibble. Essentially, the decimal conversion is performed by adding 00H, 06H, 60H or 66H depending on the Accumulator and flag conditions. Only the C flag may be affected by this instruction which indicates that if the original BCD sum is greater than 100, it allows multiple precision decimal addition.

Operation: $[m] \leftarrow ACC + 00H$ or
$[m] \leftarrow ACC + 06H$ or
$[m] \leftarrow ACC + 60H$ or
$[m] \leftarrow ACC + 66H$

Affected flag(s): C

LDEC [m] Decrement Data Memory

Description: Data in the specified Data Memory is decremented by 1.

Operation: $[m] \leftarrow [m] - 1$

Affected flag(s): Z

LDECA [m] Decrement Data Memory with result in ACC

Description: Data in the specified Data Memory is decremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation: $ACC \leftarrow [m] - 1$

Affected flag(s): Z

LINC [m] Increment Data Memory

Description: Data in the specified Data Memory is incremented by 1.

Operation: $[m] \leftarrow [m] + 1$

Affected flag(s): Z

LINCA [m] Increment Data Memory with result in ACC

Description: Data in the specified Data Memory is incremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation: $ACC \leftarrow [m] + 1$

Affected flag(s): Z

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

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

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

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

LRL [m] Rotate Data Memory left
Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0.
Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← [m].7
Affected flag(s) None

LRLA [m] Rotate Data Memory left with result in ACC
Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.
Operation ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← [m].7
Affected flag(s) None

LRLC [m] Rotate Data Memory left through Carry
Description The contents of the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into bit 0.
Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← C
C ← [m].7
Affected flag(s) C

LRLCA [m] Rotate Data Memory left through Carry with result in ACC
Description Data in the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into the bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.
Operation ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← C
C ← [m].7
Affected flag(s) C

LRR [m] Rotate Data Memory right

Description The contents of the specified Data Memory are rotated right by 1 bit with bit 0 rotated into bit 7.

Operation [m].i ← [m].(i+1); (i=0~6)
[m].7 ← [m].0

Affected flag(s) None

LRRA [m] Rotate Data Memory right with result in ACC

Description Data in the specified Data Memory is rotated right by 1 bit with bit 0 rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.i ← [m].(i+1); (i=0~6)
ACC.7 ← [m].0

Affected flag(s) None

LRRC [m] Rotate Data Memory right through Carry

Description The contents of the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7.

Operation [m].i ← [m].(i+1); (i=0~6)
[m].7 ← C
C ← [m].0

Affected flag(s) C

LRRCA [m] Rotate Data Memory right through Carry with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.i ← [m].(i+1); (i=0~6)
ACC.7 ← C
C ← [m].0

Affected flag(s) C

LSBC A,[m] Subtract Data Memory from ACC with Carry

Description The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

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

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

LSBCM A,[m] Subtract Data Memory from ACC with Carry and result in Data Memory

Description The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

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

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

LSDZ [m] Skip if decrement Data Memory is 0

Description The contents of the specified Data Memory are first decremented by 1. If the result is 0 the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation [m] ← [m] − 1
Skip if [m]=0

Affected flag(s) None

LSDZA [m] Skip if decrement Data Memory is zero with result in ACC

Description The contents of the specified Data Memory are first decremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.

Operation ACC ← [m] − 1
Skip if ACC=0

Affected flag(s) None

LSET [m] Set Data Memory

Description Each bit of the specified Data Memory is set to 1.

Operation [m] ← FFH

Affected flag(s) None

LSET [m].i Set bit of Data Memory

Description Bit i of the specified Data Memory is set to 1.

Operation [m].i ← 1

Affected flag(s) None

LSIZ [m] Skip if increment Data Memory is 0

Description The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation [m] ← [m] + 1
Skip if [m]=0

Affected flag(s) None

LSIZA [m] Skip if increment Data Memory is zero with result in ACC

Description The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation ACC ← [m] + 1
Skip if ACC=0

Affected flag(s) None

LSNZ [m].i Skip if Data Memory is not 0

Description If the specified Data Memory is not 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is 0 the program proceeds with the following instruction.

Operation Skip if [m].i ≠ 0

Affected flag(s) None

LSNZ [m] Skip if Data Memory is not 0

Description If the content of the specified Data Memory is not 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is 0 the program proceeds with the following instruction.

Operation Skip if [m] ≠ 0

Affected flag(s) None

LSUB A,[m] Subtract Data Memory from ACC

Description The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC − [m]

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

LSUBM A,[m] Subtract Data Memory from ACC with result in Data Memory

Description The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation [m] ← ACC − [m]

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

LSWAP [m] Swap nibbles of Data Memory

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

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s) None

LSWAPA [m] Swap nibbles of Data Memory with result in ACC

Description The low-order and high-order nibbles of the specified Data Memory are interchanged. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC.3~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0

Affected flag(s) None

LSZ [m] Skip if Data Memory is 0

Description If the contents of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation Skip if [m]=0

Affected flag(s) None

LSZA [m] Skip if Data Memory is 0 with data movement to ACC

Description The contents of the specified Data Memory are copied to the Accumulator. If the value is zero, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation ACC ← [m]
Skip if [m]=0

Affected flag(s) None

LSZ [m].i Skip if bit i of Data Memory is 0

Description If bit i of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.

Operation Skip if [m].i=0

Affected flag(s) None

LTABRD [m] Read table (current page) to TBLH and Data Memory

Description The low byte of the program code (current page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

LTABRDL [m] Read table (last page) to TBLH and Data Memory

Description The low byte of the program code (last page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

LITABRD [m] Increment table pointer low byte first and read table to TBLH and Data Memory

Description Increment table pointer low byte, TBLP, first and then the program code addressed by the table pointer (TBHP and TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

LITABRDL [m] Increment table pointer low byte first and read table (last page) to TBLH and Data Memory

Description Increment table pointer low byte, TBLP, first and then the low byte of the program code (last page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

LXOR A,[m] Logical XOR Data Memory to ACC

Description Data in the Accumulator and the specified Data Memory perform a bitwise logical XOR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″XOR″ [m]

Affected flag(s) Z

LXORM A,[m] Logical XOR ACC to Data Memory

Description Data in the specified Data Memory and the Accumulator perform a bitwise logical XOR operation. The result is stored in the Data Memory.

Operation [m] ← ACC ″XOR″ [m]

Affected flag(s) Z

Package Information

Note that the package information provided here is for consultation purposes only. As this information may be updated at regular intervals users are reminded to consult the Holtek website for the latest version of the Package/Carton Information.

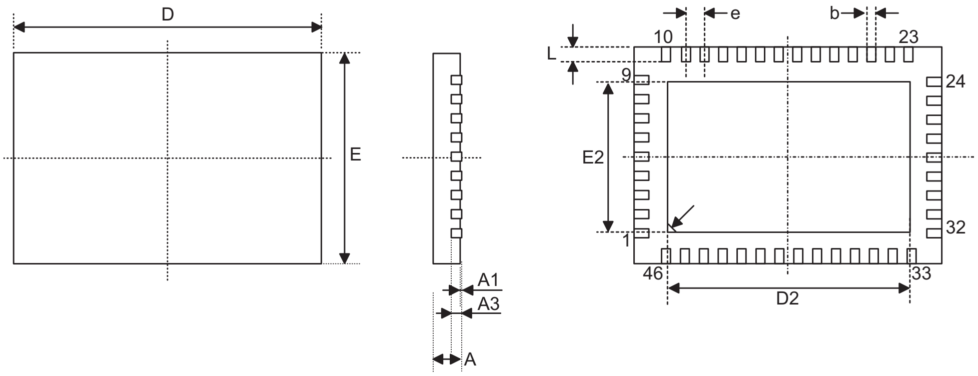
Additional supplementary information with regard to packaging is listed below. Click on the relevant section to be transferred to the relevant website page.

- Package Information (include Outline Dimensions, Product Tape and Reel Specifications)

- The Operation Instruction of Packing Materials

- Carton information

SAW Type 46-pin QFN (6.5mm×4.5mm) Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	0.031	0.033	0.035
A1	0.000	0.001	0.002
A3	—	0.008 BSC	—
b	0.006	0.008	0.010
D	0.254	0.256	0.258
E	0.175	0.177	0.179
e	—	0.016 BSC	—
D2	0.197	0.201	0.205
E2	0.118	0.122	0.126
L	0.012	0.016	0.020

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	0.80	0.85	0.90
A1	0.00	0.02	0.04
A3	—	0.20 BSC	—
b	0.15	0.20	0.25
D	6.45	6.50	6.55
E	4.45	4.50	4.55
e	—	0.40 BSC	—
D2	5.00	5.10	5.20
E2	3.00	3.10	3.20
L	0.30	0.40	0.50

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