Personal Care Flash MCU

HT45F3420/HT45F3430

Revision: V1.31 Date: April 11, 2017

www.holtek.com

Table of Contents

Features

CPU Features

- Operating Voltage:
  - f_{SYS}=8MHz: 2.2V~5.5V
- Up to 0.5μs instruction cycle with 8MHz system clock at V_{DD}=5V
- Power down and wake-up functions to reduce power consumption
- Two Internal Oscillators – require no external components
  - 8MHz High Speed Oscillator - HIRC
  - 32 kHz Low Speed Oscillator - LIRC;
- Multi-mode operation: NORMAL, SLOW, IDLE and SLEEP
- All instructions executed in one or two instruction cycles
- Table read instructions
- 63 powerful instructions
- 4-level subroutine nesting
- Bit manipulation instruction

Peripheral Features

- Flash Program Memory: 1K×14~2K×15
- RAM Data Memory: 64×8~128×8
- True EEPROM Memory: 32×8~64×8
- Watchdog Timer function
- Up to 22 bidirectional I/O lines
- Programmable I/O port source current for LED driving applications - only for HT45F3430
- Software controlled 4-SCOM LCD driver with 1/2 bias - only for HT45F3430
- Up to two pin-shared external interrupt
- Two pins with integrated voltage divider circuit
- Timer Module for time measure, compare match output, capture input, PWM output and single pulse output functions
- Dual Time-Base functions for generation of fixed time interrupt signals
- High Resolution PWM with Complementary output (H.R. PWM)
- Multi-channel 12-bit resolution A/D converter
- Over current protection function with interrupt
- Over voltage protection function with interrupt
- Low voltage reset function
- Flash program memory can be re-programmed up to 100,000 times
- Flash program memory data retention > 10 years
- True EEPROM data memory can be re-programmed up to 1,000,000 times
- True EEPROM data memory data retention > 10 years
- Package types: 8-pin SOP/10-pin MSOP for HT45F3420
  16-pin NSOP/24-pin SSOP for HT45F3430

General Description

The series of devices are dedicated for use in battery power control. They are Flash Memory type 8-bit high performance RISC architecture microcontrollers. Offering users the convenience of Flash Memory multi-programming features, the devices also include a wide range of functions and features. Other memory includes an area of RAM Data Memory as well as an area of true EEPROM memory for storage of non-volatile data such as serial numbers, calibration data etc.

Analog features include a multi-channel 12-bit A/D converter function. A Timer Module provides timing, pulse generation, capture input, compare match output and PWM generation functions. Protective features such as an internal Watchdog Timer and Low Voltage Reset coupled with excellent noise immunity and ESD protection ensure that reliable operation is maintained in hostile electrical environments. The protection functions also include Over Current Protection and Over Voltage Protection.

A high speed HIRC and low speed LIRC oscillator are provided which are fully integrated system oscillators and which require no external components for their implementation.

The inclusion of flexible I/O programming features, Time-Base functions along with many other features ensure that the devices will find excellent use in applications such as electronic personal care products in addition to many others.

Selection Table

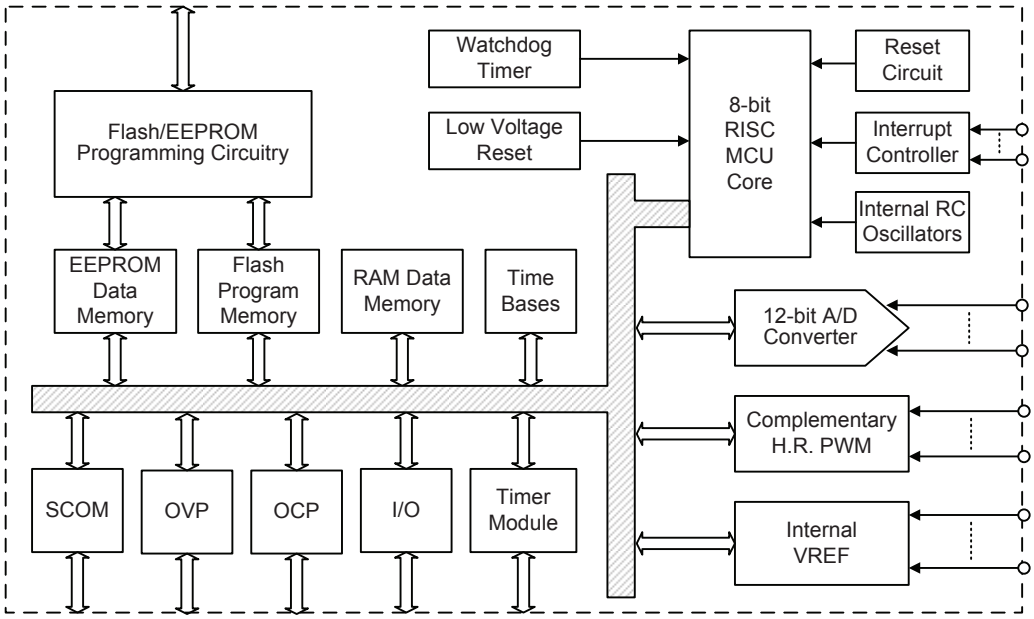
Most features are common to both devices. The main features distinguishing them are Memory capacity, I/O count, external interrupt count, A/D converter channel count, LCD/LED driving function and the package types. The following table summarises the main features of each device.

Part No.	V_{DD}	ROM	RAM	EEPROM	I/O	Ext. Int.	A/D Converter
HT45F3420	2.2V~5.5V	1K×14	64×8	32×8	8	1	12-bit×4
HT45F3430	2.2V~5.5V	2K×15	128×8	64×8	22	2	12-bit×8

Part No.	LCD Driver	LED Driver	Timer Module	Time Base	Stack	Package
HT45F3420	—	—	10-bit STM×1	2	4	8SOP 10MSOP
HT45F3430	4-SCOM	√	10-bit STM×1	2	4	16NSOP 24SSOP

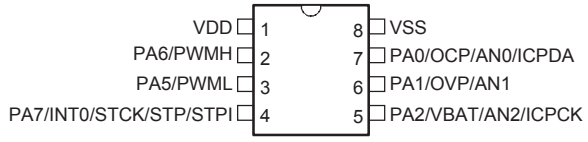
Note: As devices exist in more than one package format, the table reflects the situation for the package with the most pins.

Block Diagram

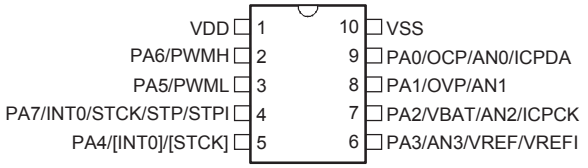


Note: The 4-SCOM LCD driving function only exists in the HT45F3430.

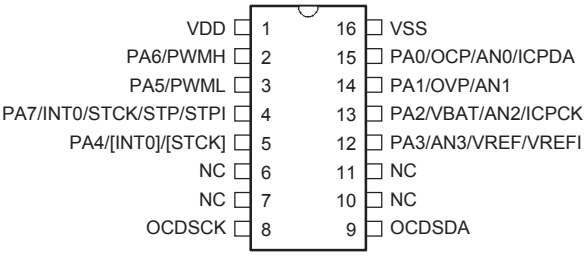
Pin Assignment



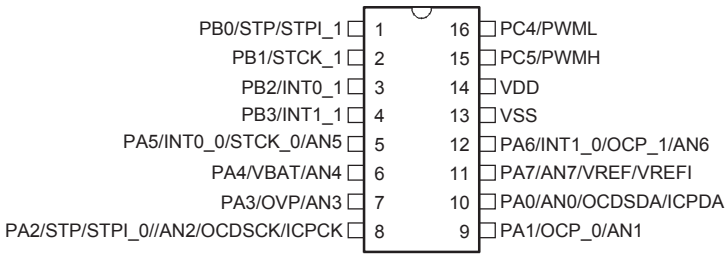
HT45F3420
8 SOP-A



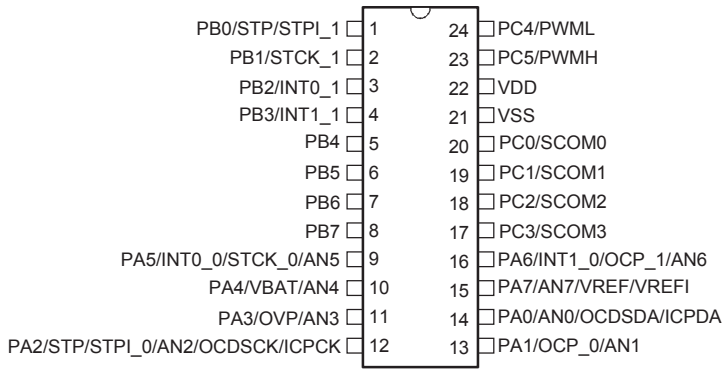
HT45F3420
10 MSOP-A



HT45V3420
16 NSOP-A



HT45F3430/HT45V3430
16 NSOP-A



HT45F3430/HT45V3430
24 SSOP-A

Note: 1. Pin names suffixed by "_0" or "_1" indicate non-fixed pinout remapping locations which should be selected by the IFS0 register.

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

3. The OCDSDA and OCDSCK pins are supplied for the OCDS dedicated pins and as such are only available for the HT45V3420 and HT45V3430 EV devices.

Pin Description

With the exception of the power pins and some relevant transformer control pins, all pins on the devices 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.

Note that the pin description refers to the largest package size, as a result some pins may not exist on smaller package types.

HT45F3420

Pin Name	Function	OPT	I/T	O/T	Description
PA0/OCP/AN0/ ICPDA	PA0	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	OCP	PAS0	AN	—	OCP input
	AN0	PAS0	AN	—	ADC input channel 0
	ICPDA	—	ST	CMOS	ICP Data Line

Pin Name	Function	OPT	I/T	O/T	Description
PA1/OVP/AN1	PA1	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	OVP	PAS0	AN	—	OVP input
	AN1	PAS0	AN	—	ADC input channel 1
PA2/VBAT/AN2/ ICPCK	PA2	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	VBAT	PAS0	AN	—	VBAT input
	AN2	PAS0	AN	—	ADC input channel 2
	ICPCK	—	ST	CMOS	ICP clock line
PA3/AN3/VREF/ VREFI	PA3	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	AN3	PAS0	AN	—	ADC input channel 3
	VREF	PAS0	AN	—	ADC reference input
	VREFI	PAS0	AN	—	PGA input for ADC reference
PA4/INT0/STCK	PA4	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	INT0	IFS0	ST	—	Interrupt input
	STCK	IFS0	ST	—	STM clock input
PA5/PWML	PA5	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up (always with 30K poly pull-low resistor)
	PWML	PAS1	—	CMOS	Complementary PWM output (always with 30K poly pull-low resistor)
PA6/PWMH	PA6	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	PWMH	PAS1	—	CMOS	Complementary PWM output
PA7/INT0/STP/ STCK/STPI	PA7	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	INT0	IFS0	ST	—	External interrupt input
	STP	PAS1	—	CMOS	STM output
	STCK	IFS0	ST	—	STM clock input
	STPI	PAS1	ST	—	STM capture input
VDD	VDD	—	PWR	—	Digital positive power supply
VSS	VSS	—	PWR	—	Digital negative power supply
OCDS EV					
OCDSCK	OCDSCK	—	ST	—	On Chip Debug System Clock Line (OCDS EV only)
OCDSDA	OCDSDA	—	ST	CMOS	On Chip Debug System Data Line (OCDS EV only)

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

HT45F3430

Pin Name	Function	OPT	I/T	O/T	Description
PA0/AN0/ OCDSDA/ICPDA	PA0	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	AN0	PAS0	AN	—	ADC input channel 0
	OCDSDA	—	ST	CMOS	OCDS Data Line for EV device only
	ICPDA	—	ST	CMOS	ICP Data Line
PA1/OCP_0/AN1	PA1	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	OCP_0	PAS0 IFS0	AN	—	OCP input
	AN1	PAS0	AN	—	ADC input channel 1
PA2/STP/STPI_0/ AN2/OCDSCK/ ICPCK	PA2	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	STP	PAS0	—	CMOS	STM output
	STPI_0	PAS0 IFS0	ST	—	STM capture input
	AN2	PAS0	AN	—	ADC input channel 2
	OCDSCK	—	ST	—	OCDS Clock Line for EV device only
	ICPCK	—	ST	CMOS	ICP Clock Line
PA3/OVP/AN3	PA3	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	OVP	PAS0 IFS0	AN	—	OVP input
	AN3	PAS0	AN	—	ADC input channel 3
PA4/VBAT/AN4	PA4	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	VBAT	PAS1	AN	—	VBAT input
	AN4	PAS1	AN	—	ADC input channel 4
PA5/INT0_0/ STCK_0/AN5	PA5	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	INT0_0	PAS1 IFS0	ST	—	External interrupt input
	STCK_0	PAS1 IFS0	ST	—	STM clock input
	AN5	PAS1	AN	—	ADC input channel 5
PA6/INT1_0/ OCP_1/AN6	PA6	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	INT1_0	PAS1 IFS0	ST	—	External interrupt input
	OCP_1	PAS1 IFS0	AN	—	OCP input
	AN6	PAS1	AN	—	ADC input channel 6

Pin Name	Function	OPT	I/T	O/T	Description
PA7/AN7/VREF/VREFI	PA7	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	VREF	PAS1	AN	—	ADC reference input
	VREFI	PAS1	AN	—	PGA input for ADC reference
	AN7	PAS1	AN	—	ADC input channel 7
PB0/STP/STPI_1	PB0	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	STP	PBS0	—	CMOS	STM output
	STPI_1	PBS0 IFS0	ST	—	STM capture input
PB1/STCK_1	PB1	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	STCK_1	IFS0	ST	—	STM clock input
PB2/INT0_1	PB2	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	INT0_1	IFS0	ST	—	External interrupt input
PB3/INT1_1	PB3	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	INT1_1	IFS0	ST	—	External interrupt input
PB4	PB4	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
PB5	PB5	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
PB6	PB6	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
PB7	PB7	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
PC0/SCOM0	PC0	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	SCOM0	PCS0	—	AN	SCOM output
PC1/SCOM1	PC1	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	SCOM1	PCS0	—	AN	SCOM output
PC2/SCOM2	PC2	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	SCOM2	PCS0	—	AN	SCOM output
PC3/SCOM3	PC3	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	SCOM3	PCS0	—	AN	SCOM output
PC4/PWML	PC4	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up (always with 30K poly pull-low resistor)
	PWML	PCS0	—	CMOS	Complementary PWM output (always with 30K poly pull-low resistor)
PC5/PWMH	PC5	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	PWMH	PCS0	—	CMOS	Complementary PWM output
VDD	VDD	—	PWR	—	Positive power supply.
VSS	VSS	—	PWR	—	Negative power supply.

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

Absolute Maximum Ratings

Supply Voltage ..... V_{SS}−0.3V to V_{SS}+6.0V
Input Voltage ..... V_{SS}−0.3V to V_{DD}+0.3V
Storage Temperature ..... -50°C to 125°C
Operating Temperature ..... -40°C to 85°C
I_{OL} Total 80mA
I_{OH} Total ..... -80mA
Total Power Dissipation 500mW

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

D.C. Characteristics

Ta=25°C

Symbol	Parameter		Test Conditions		Min.	Typ.	Max.	Unit
			V_{DD}	Conditions				
V_{DD}	Operating Voltage (HIRC)		—	$f_{SYS}=f_{HIRC}$=8MHz	V_{LVR}	—	5.5	V
			—	$f_{SYS}=f_{HIRC}/2$=4MHz	V_{LVR}	—	5.5	V
			—	$f_{SYS}=f_{HIRC}/4$=2MHz	V_{LVR}	—	5.5	V
			—	$f_{SYS}=f_{HIRC}/8$=1MHz	V_{LVR}	—	5.5	V
I_{DD}	Operating Current (HIRC)		3V	No load, all peripherals off, $f_{SYS}=f_{HIRC}/2$=4MHz	—	0.4	0.6	mA
			5V		—	0.8	1.2	mA
			3V	No load, all peripherals off, $f_{SYS}=f_{HIRC}$=8MHz	—	0.8	1.2	mA
			5V		—	1.6	2.4	mA
	Operating Current (LIRC)		3V	No load, all peripherals off, $f_{SYS}=f_{LIRC}$=32kHz	—	10	20	μA
			5V		—	30	50	μA
I_{STB}	Standby Current (SLEEP mode)		3V	No load, all peripherals off, WDT on	—	1.5	3	μA
			5V		—	3	5	μA
	Standby Current (IDLE0 mode)		3V	No load, all peripherals off, f_{SUB} on	—	3	5	μA
			5V		—	5	10	μA
	Standby Current (IDLE1 mode, HIRC)		3V	No load, all peripherals off, f_{SUB} on, $f_{SYS}=f_{HIRC}/2$=4MHz	—	180	250	μA
			5V		—	400	600	μA
			3V	No load, all peripherals off, f_{SUB} on, $f_{SYS}=f_{HIRC}$=8MHz	—	360	500	μA
			5V		—	600	800	μA
V_{IL}	Input Low Voltage for I/O Ports or Input Pins		5V	—	0	—	1.5	V
			—	—	0	—	0.2V_{DD}	V
V_{IH}	Input High Voltage for I/O Ports or Input Pins		5V	—	3.5	—	5.0	V
			—	—	0.8V_{DD}	—	V_{DD}	V
I_{OL}	HT45F3420	I/O Port Sink Current except PA5,PA6	3V	$V_{OL}=0.1V_{DD}$	18	36	—	mA
			5V		33	66	—	mA
		PA5,PA6 Port Sink Current	3V	$V_{OL}=0.1V_{DD}$	16	32	—	mA
			5V		40	80	—	mA
	HT45F3430	I/O Port Sink Current except PC4, PC5	3V	$V_{OL}=0.1V_{DD}$	16	32	—	mA
			5V		32	64	—	mA
		PC4, PC5 Port Sink Current	3V	$V_{OL}=0.1V_{DD}$	16	32	—	mA
			5V		40	80	—	mA

Symbol	Parameter		Test Conditions		Min.	Typ.	Max.	Unit
			V_{DD}	Conditions				
I_{OH}	HT45F3420	I/O Port Source Current except PA5,PA6	3V	V_{OH}=0.9V_{DD}	-3	-6	—	mA
			5V		-7	-14	—	mA
		PA5,PA6 Port Source Current	3V	V_{OH}=0.9V_{DD}	-18	-38	—	mA
			5V		-40	-80	—	mA
	HT45F3430	I/O Port Source Current except PC4, PC5	3V	V_{OH}=0.9V_{DD}, SLEDCn[m+1, m]=00B (n=0,1..., m=0 or 2 or 4 or 6)	-1.0	-2.0	—	mA
			5V		-2.0	-4.0	—	mA
			3V	V_{OH}=0.9V_{DD}, SLEDCn[m+1, m]=01B (n=0,1..., m=0 or 2 or 4 or 6)	-1.75	-3.5	—	mA
			5V		-3.5	-7.0	—	mA
			3V	V_{OH}=0.9V_{DD}, SLEDCn[m+1, m]=10B (n=0,1..., m=0 or 2 or 4 or 6)	-2.5	-5.0	—	mA
			5V		-5.0	-10	—	mA
			3V	V_{OH}=0.9V_{DD}, SLEDCn[m+1, m]=11B (n=0,1..., m=0 or 2 or 4 or 6)	-5.5	-11	—	mA
			5V		-11	-22	—	mA
		PC4, PC5 Port Source Current	3V	V_{OH}=0.9V_{DD}	-18	-38	—	mA
			5V	V_{OH}=0.9V_{DD}	-40	-80	—	mA
R_{PH}	Pull-high Resistance for I/O Ports		3V	—	20	60	100	kΩ
			5V	—	10	30	50	kΩ
R_{PL}	HT45F3420	Pull-low Resistance for PA5 Port	3V	—	-50%	30	50%	kΩ
			5V	—	-50%	30	50%	kΩ
	HT45F3430	Pull-low Resistance for PC4 Port	3V	—	-50%	30	50%	kΩ
			5V	—	-50%	30	50%	kΩ

A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Condition				
f_{SYS}	System Clock (HIRC)	V_{LVR}~5.5V	$f_{SYS}=f_{HIRC}$=8MHz	—	8	—	MHz
	System Clock (LIRC)	V_{LVR}~5.5V	$f_{SYS}=f_{LIRC}$=32kHz	—	32	—	kHz
f_{HIRC}	High Speed Internal RC Oscillator (HIRC)	5V	Ta= 25°C	-2%	8	+2%	MHz
		5V±0.5V	Ta= 0°C~70°C	-5%	8	+5%	MHz
		5V±0.5V	Ta= -40°C~85°C	-7%	8	+7%	MHz
		2.2V~5.5V	Ta=0 ~70°C	-7%	8	+7%	MHz
		2.2V~5.5V	Ta= -40°C~ 85°C	-10%	8	+10%	MHz
f_{LIRC}	Low Speed Internal RC Oscillator (LIRC)	5V	Ta= 25°C	-10%	32	+10%	kHz
		5V±0.5V	Ta= -40°C~ 85°C	-40%	32	+40%	kHz
		2.2V~5.5V	Ta= -40°C~ 85°C	- 50%	32	+60%	kHz
t_{TC}	STCK Input Pulse Width	—	—	0.3	—	—	μs
t_{INT}	External Interrupt Minimum Pulse Width	—	—	10	—	—	μs
t_{RSTD}	System Reset Delay Time (POR Reset, LVR Hardware Reset, WDT Software Reset)	—	—	25	50	100	ms
	System Reset Delay Time (WDT Time-out Hardware Cold Reset)	—	—	8.3	16.7	33.3	ms
t_{SST}	System Start-up Timer Period (Wake-up from Halt, f_{SYS} off at Halt)	—	$f_{SYS}=f_H$~f_H/64, $f_H=f_{HIRC}$	16	—	—	t_{HIRC}
		—	$f_{SYS}=f_{SUB}=f_{LIRC}$	2	—	—	t_{LIRC}
	System Start-up Timer Period (Slow Mode ↔ Normal Mode)	—	f_{HIRC} off → on (HTO=1)	16	—	—	t_{HIRC}
	System Start-up Timer Period (Wake-up from Halt, f_{SYS} on at Halt State)	—	$f_{SYS}=f_H$~f_H/64, $f_{SYS}=f_{HIRC}$	2	—	—	t_H
		—	$f_{SYS}=f_{LIRC}$	2	—	—	t_{SUB}
	System Start-up Timer Period (WDT Time-out Hardware Cold Reset)	—	—	0	—	—	t_H
t_{EERD}	EEPROM Read Time	—	—	—	2	4	t_{SYS}
t_{EEWR}	EEPROM Write Time	—	—	—	2	4	ms

A/D Converter Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	A/D Converter Operating Voltage	—	—	2.2	—	5.5	V
V_{ADI}	A/D Converter Input Voltage	—	—	0	—	V_{REF}	V
V_{REF}	A/D Converter Reference Voltage	—	—	2.2	—	V_{DD}	V
DNL	Differential Non-linearity	3V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, V_{REF}=V_{DD}, t_{ADCK} =0.5μs	-3	—	+3	LSB
		5V					
		3V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, V_{REF}=V_{DD}, t_{ADCK} =10μs	-3	—	+3	LSB
		5V					
INL	Integral Non-linearity	3V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, V_{REF}=V_{DD}, t_{ADCK} =0.5μs	-4	—	+4	LSB
		5V					
		3V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, V_{REF}=V_{DD}, t_{ADCK} =10μs	-4	—	+4	LSB
		5V					
I_{ADC}	Additional Current for A/D Converter Enable	3V	No load, t_{ADCK} =0.5μs	—	0.2	0.4	mA
		5V	No load, t_{ADCK} =0.5μs	—	0.3	0.6	mA
t_{ADCK}	A/D Converter Clock Period	—	—	0.5	—	10	μs
t_{ADC}	A/D Conversion Time (Include Sample and Hold Time)	—	—	—	16	—	t_{ADCK}
t_{ON2ST}	A/D Converter On-to-Start Time	—	—	4	—	—	μs
I_{PGA}	Additional Current for PGA Enable	3V	No load	—	420	500	μA
		5V	No load	—	460	550	μA
V_{CM}	PGA Common Mode Voltage Range	3V	—	V_{SS}-0.3	—	V_{DD}-1.4	V
		5V	—	V_{SS}-0.3	—	V_{DD}-1.4	V
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
Ga	PGA Gain Accuracy	3V	—	-5	—	+5	%
		5V	—				

OCP Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{OCP}	OCP Operating Current	3V	OCPEN [1:0]=01B DAC V_{REF}=2.5V	—	260	350	μA
		5V	OCPEN [1:0]=01B DAC V_{REF}=2.5V	—	350	420	μA
V_{OS_CMP}	Comparator Input Offset Voltage	3V	Without calibration (COF[4:0]=10000B)	-15	—	15	mV
		5V	Without calibration (COF[4:0]=10000B)	-15	—	15	mV
		3V	With calibration	-4	—	4	mV
		5V	With calibration	-4	—	4	mV
V_{HYS}	Hysteresis	3V	—	20	40	60	mV
		5V	—	20	40	60	mV
V_{CM_CMP}	Comparator Common Mode Voltage Range	3V	—	V_{SS}	—	V_{DD}-1.4	V
		5V	—	V_{SS}	—	V_{DD}-1.4	V
V_{OS_OPA}	OPA Input Offset Voltage	3V	Without calibration (OOF[5:0]=100000B)	-15	—	15	mV
		5V	Without calibration (OOF[5:0]=100000B)	-15	—	15	mV
		3V	With calibration	-4	—	4	mV
		5V	With calibration	-4	—	4	mV
V_{CM_OPA}	OPA Common Mode Voltage Range	3V	—	V_{SS}+0.2	—	V_{DD}-1.4	V
		5V	—	V_{SS}+0.2	—	V_{DD}-1.4	V
Gain	OPA Gain Error	3V	All gain (V_{CM_OPA} > 0.2V)	-5	—	5	%
		5V	All gain (V_{CM_OPA} > 0.2V)	-5	—	5	%
DNL	Differential Nonlinearity	3V	DAC V_{REF}=V_{DD}	-1	—	+1	LSB
		5V	DAC V_{REF}=V_{DD}	-1	—	+1	LSB
INL	Integral Nonlinearity	3V	DAC V_{REF}=V_{DD}	-1.5	—	+1.5	LSB
		5V	DAC V_{REF}=V_{DD}	-1.5	—	+1.5	LSB

OVP Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{OVP}	OVP Operating Current	3V	OVPEN=1, DAC V_{REF}=2.5V	—	45	54	μA
		5V	OVPEN=1, DAC V_{REF}=2.5V	—	56	72	μA
V_{OS}	Input Offset Voltage	3V	With calibration	-4	—	4	mV
		5V	With calibration	-4	—	4	mV
V_{HYS}	Hysteresis	5V	—	20	40	60	mV
V_{CM}	Common Mode Voltage Range	3V	—	V_{SS}	—	V_{DD} - 1.4	V
		5V	—	V_{SS}	—	V_{DD} - 1.4	V
DNL	Differential Nonlinearity	3V	DAC V_{REF}=V_{DD}	-1	—	+1	LSB
		5V	DAC V_{REF}=V_{DD}	-1	—	+1	LSB
INL	Integral Nonlinearity	3V	DAC V_{REF}=V_{DD}	-1.5	—	+1.5	LSB
		5V	DAC V_{REF}=V_{DD}	-1.5	—	+1.5	LSB

DLL 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	1.2	mA
		5V	DLLEN=1	—	1.5	2	mA
f_{DLL}	Operating Frequency	2.2V~ 5.5V	f_{HIRC} =8MHz	-10%	8	+10%	MHz

LVR Electrical Characteristics

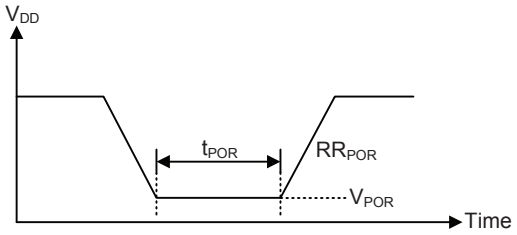
Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	1.9	—	5.5	V
V_{LVR}	Low Voltage Reset Voltage	—	LVR Enable, 2.1V option	-5%	2.1	+5%	V
V_{BG}	Bandgap Reference Voltage	—	—	-5%	1.04	+5%	V
t_{LVR}	Minimum Low Voltage Width to Reset	—	—	120	240	480	μs

Power on Reset Electrical Characteristics

Ta=25°C

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

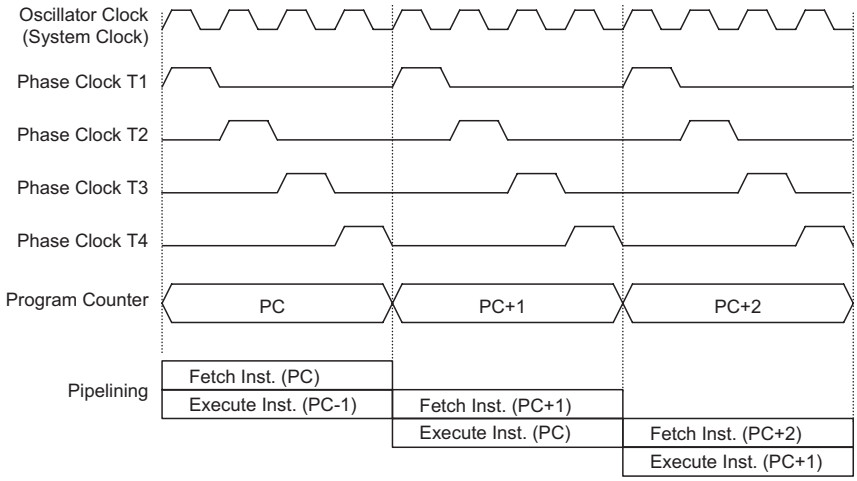


System Architecture

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

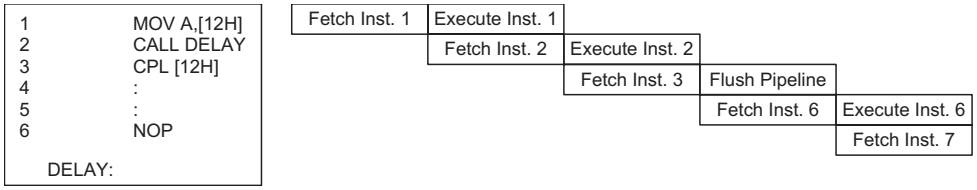
Clocking and Pipelining

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



System Clock and Pipelining

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



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.

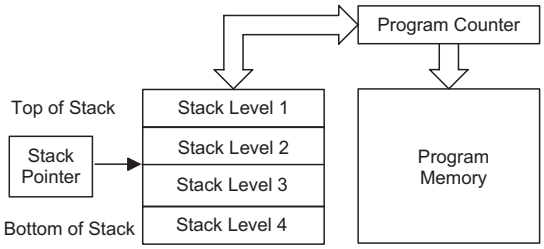
Device	Program Counter	
	Program Counter High byte	PCL Register
HT45F3420	PC9~PC8	PCL7~PCL0
HT45F3430	PC10~PC8	PCL7~PCL0

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

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



Arithmetic and Logic Unit – ALU

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

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

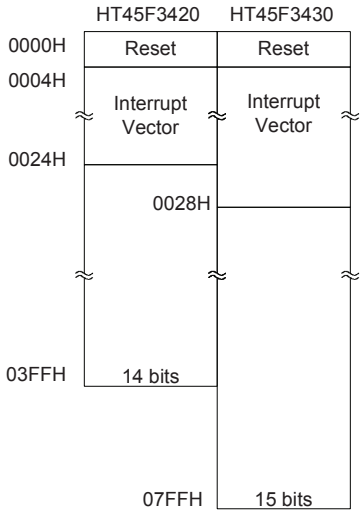
Flash Program Memory

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

Structure

The Program Memory has a capacity of 1K×14~2K×15 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.

Device	Capacity
HT45F3420	1K×14
HT45F3430	2K×15



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. This register defines the total address of the look-up table.

After setting up the table pointer, the table data can be retrieved from the Program Memory using the "TABRDC[m]" or "TABRDL[m]" instructions, respectively. When the instruction is executed, the lower order table byte from the Program Memory will be transferred to the user defined Data Memory register [m] as specified in the instruction. The higher order table data byte from the Program Memory will be transferred to the TBLH special register. Any unused bits in this transferred higher order byte will be read as "0".

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

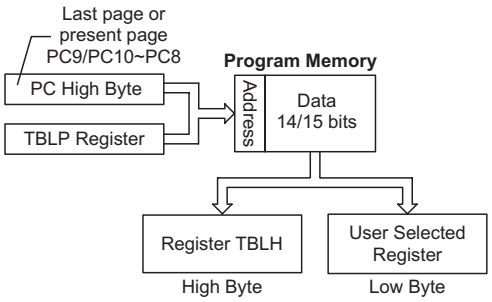


Table Program Example

The following example shows how the table pointer and table data is defined and retrieved from the microcontroller. This example uses raw table data located in the Program Memory which is stored there using the ORG statement. The value at this ORG statement is "700H" which refers to the start address of the last page within the 2K words Program Memory of the HT45F3430. The table pointer is setup here to have an initial value of "06H". This will ensure that the first data read from the data table will be at the Program Memory address "706H" or 6 locations after the start of the last page. Note that the value for the table pointer is referenced to the first address of the present page if the "TABRDC [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 "TABRDC [m]" instruction is executed.

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

Table Read Program Example

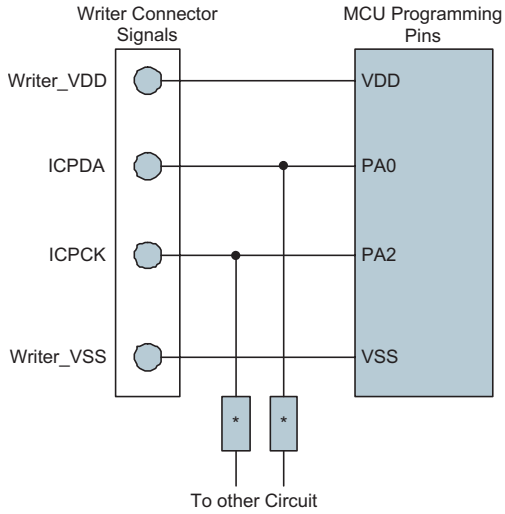
```
tempreg1 db ?      ; temporary register #1
tempreg2 db ?      ; temporary register #2
:
:
mov a,06h          ; initialise low table pointer - note that this address is referenced
mov tblp,a         ; to the last page or present page
:
:
tabrdl tempreg1    ; transfers value in table referenced by table pointer data at program
                   ; memory address "706H" transferred to tempreg1 and TBLH
dec tblp           ; reduce value of table pointer by one
tabrdl tempreg2    ; transfers value in table referenced by table pointer data at program
                   ; memory address "705H" transferred to tempreg2 and TBLH in this
                   ; example the data "1AH" is transferred to tempreg1 and data "0FH" to
                   ; register tempreg2
:
:
org 700h           ; sets initial address of program memory
dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
:
:
```

In Circuit Programming

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

Holtek Write Pins	MCU Programming Pins	Function
ICPDA	PA0	Programming Serial Data/Address
ICPCK	PA2	Programming Serial Clock
VDD	VDD	Power Supply
VSS	VSS	Ground

The Program Memory and EEPROM data memory can both 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 ground. The technical details regarding the in-circuit programming of the device are beyond the scope of this document and will be supplied in supplementary literature.



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

On-Chip Debug Support – OCDS

Holtek supplies EV devices HT45V3420 and HT45V3430 which are used to emulate the HT45F3420 and HT45F3430 devices. These EV devices provide an "On-Chip Debug" function to debug the device during the development process. The EV chips and the actual MCU devices are almost functionally compatible except for the "On-Chip Debug" function. Users can use the EV chip devices 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 using the EV device during debug, if there are other pin functions which are shared with the OCDSDA and OCDSCK pins, then these functions will have no effect in the EV chips. The two OCDS pins, which are pin-shared with the ICP programming pins, are still used as Flash Memory programming pins for ICP. For a more detailed OCDS description, 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
GND	VSS	Ground

RAM Data Memory

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

Structure

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

The overall Data Memory is subdivided into two banks, the structure of which depends upon the device chose.The Special Purpose Data Memory registers are accessible in all banks, with the exception of the EEC register at address 40H, which is only accessible in Bank 1. Switching between the different Data Memory banks is achieved by setting the Bank Pointer to the correct value. The start address of the Data Memory is the address 00H.

Device	RAM	Address
HT45F3420	Special Purpose	Bank0: 00H~3FH Bank1: 00H~40H (EEC in 40H)
	General Purpose: 64×8	Bank0: 40H~7FH
HT45F3430	Special Purpose	Bank0: 00H~7FH Bank1: 00H~7FH (EEC in 40H)
	General Purpose: 128×8	Bank0: 80H~FFH

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

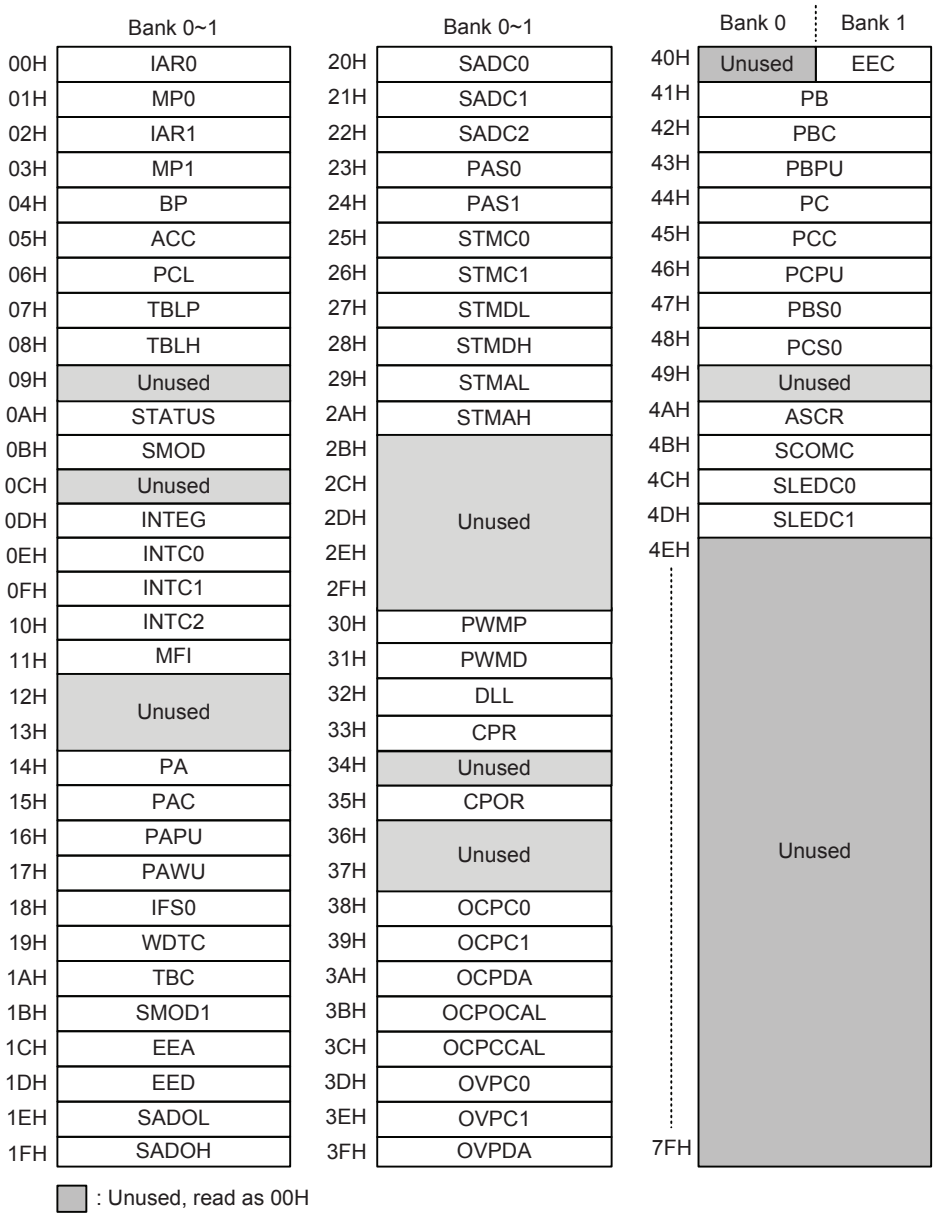
This area of Data Memory is where registers, necessary for the correct operation of the microcontroller, are stored. Most of the registers are both readable and writeable but some are protected and are readable only, the details of which are located under the relevant Special Function Register section.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

	Bank0 / Bank1
00H	IAR0
01H	MP0
02H	IAR1
03H	MP1
04H	BP
05H	ACC
06H	PCL
07H	TBLP
08H	TBLH
09H	Unused
0AH	STATUS
0BH	SMOD
0CH	Unused
0DH	INTEG
0EH	INTC0
0FH	INTC1
10H	INTC2
11H	MFI
12H	Unused
13H	Unused
14H	PA
15H	PAC
16H	PAPU
17H	PAWU
18H	IFS0
19H	WDTC
1AH	TBC
1BH	SMOD1
1CH	EEA
1DH	EED
1EH	SADOL
1FH	SADOH

	Bank0 / Bank1
20H	SADC0
21H	SADC1
22H	SADC2
23H	PAS0
24H	PAS1
25H	STMC0
26H	STMC1
27H	STMDL
28H	STMDH
29H	STMAL
2AH	STMAH
2BH ~ 2FH	Unused
30H	PWMP
31H	PWMD
32H	DLL
33H	CPR
34H	Unused
35H	CPOR
36H	Unused
37H	Unused
38H	OCPC0
39H	OCPC1
3AH	OCPDA
3BH	OCPOCAL
3CH	OCPCCAL
3DH	OVPC0
3EH	OVPC1
3FH	OVPDA
40H	Unused / EEC

: Unused, read as "00"

Special Purpose Data Memory Structure – HT45F3420



Special Purpose Data Memory Structure – HT45F3430

Special Function Register Description

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

Indirect Addressing Registers – IAR0, IAR1

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

Memory Pointers – MP0, MP1

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

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

Indirect Addressing Program Example

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

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

Bank Pointer – BP

For these devices, the Data Memory is divided into two banks, Bank0 and Bank1. Selecting the required Data Memory area is achieved using the Bank Pointer. Bit 0 of the Bank Pointer is used to select Data Memory Bank 0 or Bank 1.

The Data Memory is initialised to Bank 0 after a reset, except for a WDT time-out reset in the Power down Mode, in which case, the Data Memory bank remains unaffected. It should be noted that the Special Function Data Memory is not affected by the bank selection, which means that the Special Function Registers can be accessed from within any bank. Directly addressing the Data Memory will always result in Bank 0 being accessed irrespective of the value of the Bank Pointer. Accessing data from Bank1 must be implemented using Indirect Addressing.

BP Register

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

Bit 7 ~ 1 Unimplemented, read as "0"

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

Accumulator – ACC

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

Program Counter Low Register – PCL

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

Look-up Table Registers – TBLP, TBLH

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

Status Register – STATUS

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

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

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

- C is set if an operation results in a carry during an addition operation or if a borrow does not take place during a subtraction operation; otherwise C is cleared. C is also affected by a rotate through carry instruction.
- AC is set if an operation results in a carry out of the low nibbles in addition, or no borrow from the high nibble into the low nibble in subtraction; otherwise AC is cleared.
- Z is set if the result of an arithmetic or logical operation is zero; otherwise Z is cleared.
- OV is set if an operation results in a carry into the highest-order bit but not a carry out of the highest-order bit, or vice versa; otherwise OV is cleared.
- PDF is cleared by a system power-up or executing the "CLR WDT" instruction. PDF is set by executing the "HALT" instruction.
- TO is cleared by a system power-up or executing the "CLR WDT" or "HALT" instruction. TO is set by a WDT time-out.

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

STATUS Register

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

"x" unknown

Bit 7~6 Unimplemented, read as "0"

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

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

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

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

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

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

EEPROM Data Memory

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

EEPROM Data Memory Structure

The EEPROM Data Memory capacity is 32×8 to 64×8 bits for these series devices. Unlike the Program Memory and RAM Data Memory, the EEPROM Data Memory is not directly mapped and is therefore not directly accessible 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 two address registers and one data register in Bank 0 and a single control register in Bank 1.

Device	Capacity
HT45F3420	32×8
HT45F3430	64×8

EEPROM Registers

Three registers control the overall operation of the internal EEPROM Data Memory. These are the address registers, EEA, the data register, EED and a single control register, EEC. As both the EEA and EED registers are located in Bank 0 and Bank 1, they can be directly accessed in the same way as any other Special Function Register. The EEC register however, being located in Bank1 only, cannot be directly addressed directly and can only be read from or written to indirectly using the MP1 Memory Pointer and Indirect Addressing Register, IAR1. Because the EEC control register is located at address 40H in Bank 1, the MP1 Memory Pointer must first be set to the value 40H and the Bank Pointer register, BP, set to the value, 01H, before any operations on the EEC register are executed.

Register Name	Bit							
	7	6	5	4	3	2	1	0
EEA(HT45F3420)	—	—	—	D4	D3	D2	D1	D0
EEA(HT45F3430)	—	—	D5	D4	D3	D2	D1	D0
EED	D7	D6	D5	D4	D3	D2	D1	D0
EEC	—	—	—	—	WREN	WR	RDEN	RD

EEPROM Control Register List

EEA Register – HT45F3420

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

Bit 7 ~ 5 Unimplemented, read as "0"

Bit 4 ~ 0 Data EEPROM address
Data EEPROM address bit 4~ bit 0

EEA Register – HT45F3430

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 Data EEPROM address
Data EEPROM address bit 5~ bit 0

EED Register

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

Bit 7 ~ 0 Data EEPROM data
Data EEPROM data bit 7 ~ bit 0

EEC Register

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

Bit 7 ~ 4 Unimplemented, read as "0"

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

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

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

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

Note: The WREN, WR, RDEN and RD can not be set to "1" at the same time in one instruction. The WR and RD can not be set to "1" at the same time.

Reading Data from the EEPROM

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

Writing Data to the EEPROM

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

Write Protection

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

EEPROM Interrupt

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

Programming Considerations

Care must be taken that data is not inadvertently written to the EEPROM. Protection can be Periodic by ensuring that the Write Enable bit is normally cleared to zero when not writing. Also the Bank Pointer could be normally cleared to zero as this would inhibit access to Bank 1where the EEPROM control register exist. Although certainly not necessary, consideration might be given in the application program to the checking of the validity of new write data by a simple read back process. When writing data the WR bit must be set high immediately after the WREN bit has been set high, to ensure the write cycle executes correctly. The global interrupt bit EMI should also be cleared before a write cycle is executed and then re-enabled after the write cycle starts. Note that the devices should not enter the IDLE or SLEEP mode until the EEPROM read or write operation is totally complete. Otherwise, the EEPROM read or write operation will fail.

Programming Examples

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

```
MOV A, EEPROM_ADRES             ; user defined address
MOV EEA, A
MOV A, 040H                     ; setup memory pointer MP1
MOV MP1, A                      ; MP1 points to EEC register
MOV A, 01H                      ; setup Bank Pointer
MOV BP, A
SET IAR1.1                      ; set RDEN bit, enable read operations
SET IAR1.0                      ; start Read Cycle - set RD bit
BACK:
SZ  IAR1.0                      ; check for read cycle end
JMP BACK
CLR IAR1                        ; disable EEPROM write
CLR BP
MOV A, EED                      ; move read data to register
MOV READ_DATA, A
```

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

```
MOV A, EEPROM_ADRES             ; user defined address
MOV EEA, A
MOV A, EEPROM_DATA              ; user defined data
MOV EED, A
MOV A, 040H                     ; setup memory pointer MP1
MOV MP1, A                      ; MP1 points to EEC register
MOV A, 01H                      ; setup Bank Pointer
MOV BP, A
CLR EMI
SET IAR1.3                      ; set WREN bit, enable write operations
SET IAR1.2                      ; start Write Cycle - set WR bit- executed immediately
after set WREN bit
SET EMI
BACK:
SZ  IAR1.2                      ; check for write cycle end
JMP BACK
CLR IAR1                        ; disable EEPROM write
CLR BP
```

Oscillators

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

Oscillator Overview

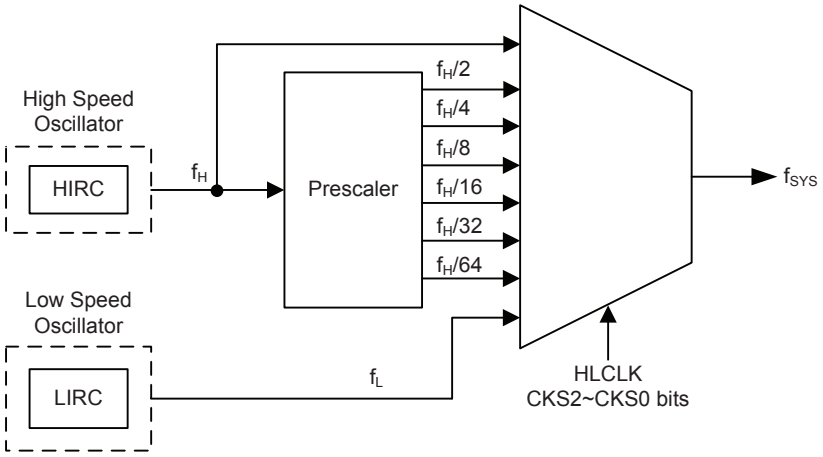
In addition to being the source of the main system clock the oscillators also provide clock sources for the Watchdog Timer and Time Base Interrupts. Two fully integrated internal oscillators, requiring no external components, are provided to form a range of both fast and slow system oscillators. The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

Type	Name	Freq.
Internal High Speed RC	HIRC	8MHz
Internal Low Speed RC	LIRC	32kHz

Oscillator Types

System Clock Configurations

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



System Clock Configurations

High Speed Internal RC Oscillator – HIRC

The high speed internal RC oscillator is a fully integrated system oscillator requiring no external components. The internal RC oscillator has a fixed frequency of 8MHz. Device trimming during the manufacturing process and the inclusion of internal frequency compensation circuits are used to ensure that the influence of the power supply voltage, temperature and process variations on the oscillation frequency are minimised. As a result, at a power supply of 5V and at temperature of 25°C degrees, the fixed oscillation frequency of the HIRC will have a tolerance within 2%.

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. 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.

Supplementary Oscillator

The low speed oscillator, in addition to providing a system clock source is also used to provide a clock source to other device functions, such as Watchdog Timer and the Time Base Interrupts.

Operating Modes and System Clocks

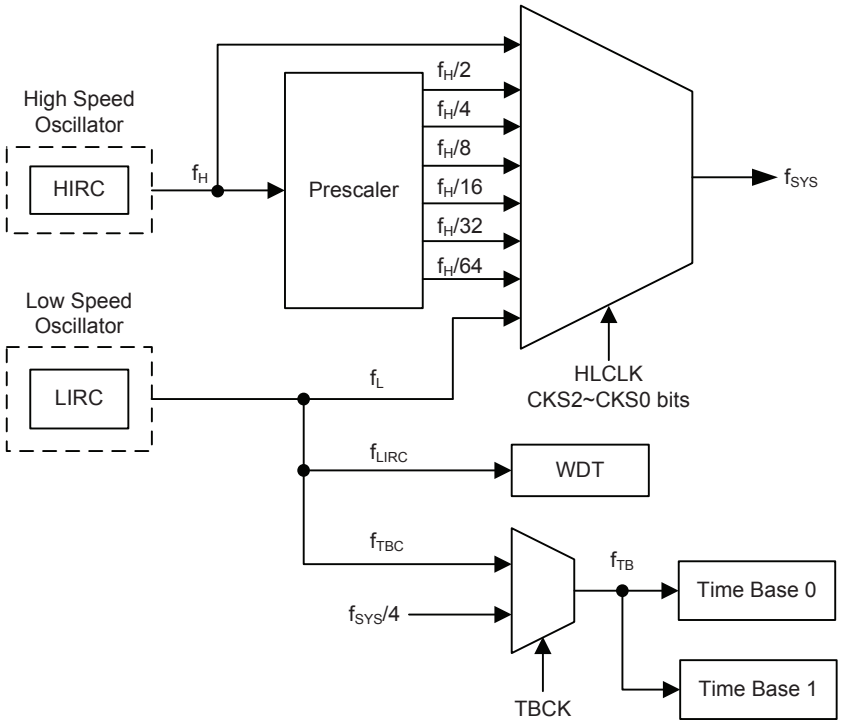
Present day applications require that their microcontrollers have high performance but often still demand that they consume as little power as possible, conflicting requirements that are especially true in battery powered portable applications. The fast clocks required for high performance will by their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

System Clocks

The devices have two different clock sources for both the CPU and peripheral function operation. By providing the user with clock options using register programming, a clock system can be configured to obtain maximum application performance.

The main system clock, can come from either a high frequency, f_H, or a low frequency, f_L, and is selected using the HLCLK bit and CKS2~CKS0 bits in the SMOD register. The high speed system clock can be sourced from HIRC oscillator. The low speed system clock source can be sourced from the internal LIRC oscillator. The other choice, which is a divided version of the high speed system oscillator has a range of $f_H/2$~$f_H/64$.

There is one additional internal clock for the peripheral circuits, the f_{TBC}. f_{TBC} is sourced from the LIRC oscillator. The f_{TBC} clock is used as a source for the Time Base interrupt function and for the STM.



System Clock Configurations

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

System Operating 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 NORMAL Mode and SLOW Mode. The remaining three modes, the SLEEP, IDLE0 and IDLE1 modes are used when the microcontroller CPU is switched off to conserve power.

Operating Mode	Description			
	CPU	f_{SYS}	f_{LIRC}	f_{TBC}
NORMAL mode	On	f_H~f_H/64	On	On
SLOW mode	On	f_L	On	On
IDLE0 mode	Off	Off	On	On
IDLE1 mode	Off	On	On	On
SLEEP mode	Off	Off	On	Off

NORMAL Mode

As the name suggests this is one of the main operating modes where the microcontroller has all of its functions operational and where the system clock is provided 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 64, the actual ratio being selected by the CKS2~CKS0 and HLCLK bits in the SMOD register. Although a high speed oscillator is used, running the microcontroller at a divided clock ratio reduces the operating current.

SLOW Mode

This is also a mode where the microcontroller operates normally although now with a slower speed clock source. The clock source used will be from the low speed oscillator LIRC. Running the microcontroller in this mode allows it to run with much lower operating currents. In the SLOW Mode, the f_H is off.

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} clocks will continue to operate and the Watchdog Timer function is enabled.

IDLE0 Mode

The IDLE0 Mode is entered when a HALT instruction is executed and when the IDLEN bit in the SMOD register is high and the FSYSON bit in the SMOD1 register is low. In the IDLE0 Mode the system oscillator will be inhibited from driving the CPU but some peripheral functions will remain operational such as the Watchdog Timer and TMs. In the IDLE0 Mode, the system oscillator will be stopped.

IDLE1 Mode

The IDLE1 Mode is entered when a HALT instruction is executed and when the IDLEN bit in the SMOD register is high and the FSYSON bit in the SMOD1 register is high. In the IDLE1 Mode the system oscillator will be inhibited from driving the CPU but may continue to provide a clock source to keep some peripheral functions operational such as the Watchdog Timer and TMs. In the IDLE1 Mode, the system oscillator will continue to run, and this system oscillator may be high speed or low speed system oscillator. In the IDLE1Mode, the Watchdog Timer clock, f_{LIRC}, will be on.

Control Registers

The registers, SMOD and SMOD1, are used for overall control of the internal clocks within the devices.

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	0	0	0	—	0	0	1	1

Bit 7 ~ 5 **CKS2 ~ CKS0**: The system clock selection when HLCLK is "0"
000: f_L (f_{LIRC})
001: f_L (f_{LIRC})
010: $f_H/64$
011: $f_H/32$
100: $f_H/16$
101: $f_H/8$
110: $f_H/4$
111: $f_H/2$
These three bits are used to select which clock is used as the system clock source. In addition to the system clock source, which can be the LIRC, a divided version of the high speed system oscillator can also be chosen as the system clock source.

Bit 4 Unimplemented, read as "0"

Bit 3 **LTO**: Low speed system oscillator ready flag
0: Not ready
1: Ready
This is the low speed system oscillator ready flag which indicates when the low speed system oscillator is stable after power on reset or a wake-up has occurred.

Bit 2 **HTO**: High speed system oscillator ready flag
0: Not ready
1: Ready
This is the high speed system oscillator ready flag which indicates when the high speed system oscillator is stable. This flag is cleared to "0" by hardware when the device is powered on and then changes to a high level after the high speed system oscillator is stable. Therefore this flag will always be read as "1" by the application program after device power-on.

Bit 1 **IDLEN**: IDLE Mode Control
0: Disable
1: Enable
This is the IDLE Mode Control bit and determines what happens when the HALT instruction is executed. If this bit is high, when a HALT instruction is executed the device will enter the IDLE Mode. In the IDLE1 Mode the CPU will stop running but the system clock will continue to keep the peripheral functions operational, if FSYSON bit is high. If FSYSON bit is low, the CPU and the system clock will all stop in IDLE0 mode. If the bit is low the device will enter the SLEEP Mode when a HALT instruction is executed.

Bit 0 **HLCLK**: System Clock Selection
0: $f_H/2$ ~ $f_H/64$ or f_L
1: f_H
This bit is used to select if the f_H clock or the $f_H/2$ ~ $f_H/64$ or f_L clock is used as the system clock. When the bit is high the f_H clock will be selected and if low the $f_H/2$ ~ $f_H/64$ or f_L clock will be selected. When system clock switches from the f_H clock to the f_L clock and the f_H clock will be automatically switched off to conserve power.

SMOD1 Register

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

"x" unknown

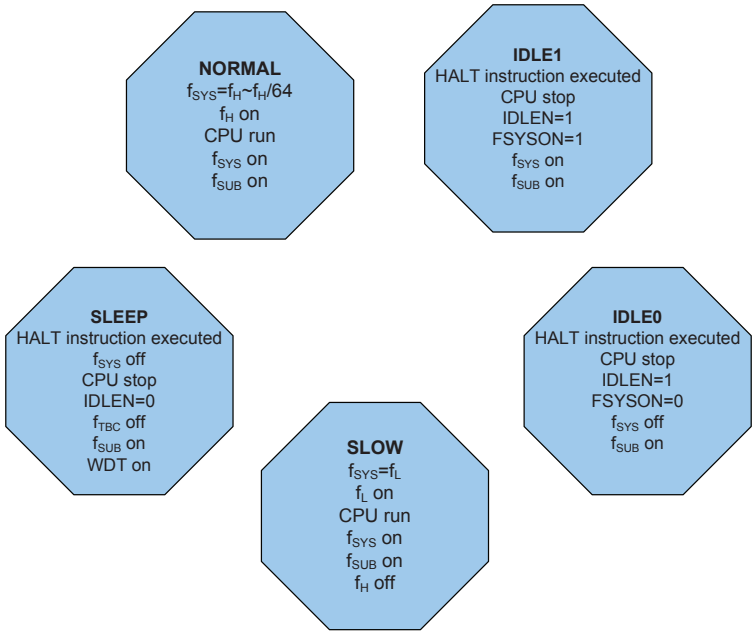
Bit 7 **FSYSON**: f_{SYS} Control in IDLE Mode
0: Disable
1: Enable
Bit 6~3 Unimplemented, read as 0
Bit 2 **LVRF**: LVR function reset flag
Described elsewhere
Bit 1 Unimplemented, read as 0
Bit 0 **WRF**: WDT Control register software reset flag
Described elsewhere

Operating Mode Switching

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

In simple terms, Mode Switching between the NORMAL Mode and SLOW Mode is executed using the HLCLK bit and CKS2~CKS0 bits in the SMOD register while Mode Switching from the NORMAL/SLOW Modes to the SLEEP/IDLE Modes is executed via the HALT instruction. When a HALT instruction is executed, whether the devices enter the IDLE Mode or the SLEEP Mode is determined by the condition of the IDLEN bit in the SMOD register and FSYSON in the SMOD1 register.

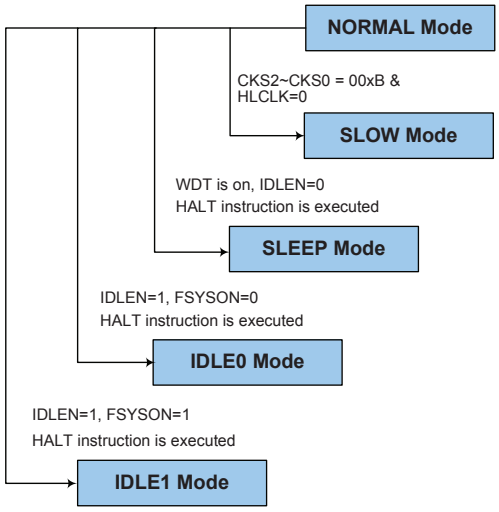
When the HLCLK bit switches to a low level, which implies that clock source is switched from the high speed clock source, f_H, to the clock source, $f_H/2$~$f_H/64$ or f_L. If the clock is from the f_L, the high speed clock source will stop running to conserve power. When this happens it must be noted that the $f_H/16$ and $f_H/64$ internal clock sources will also stop running, which may affect the operation of other internal functions such as the TM.



NORMAL Mode to SLOW Mode Switching

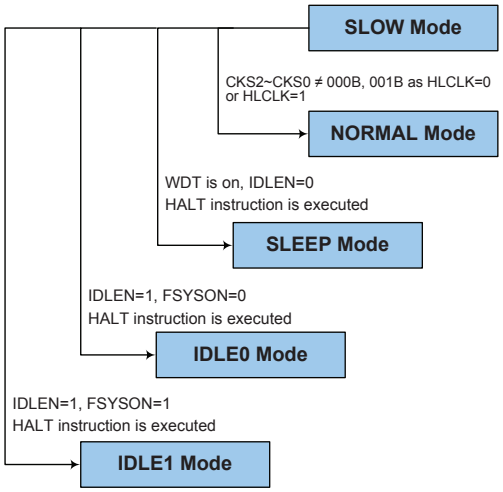
When running in the NORMAL Mode, which uses the high speed system oscillator, and therefore consumes more power, the system clock can switch to run in the SLOW Mode by setting the HLCLK bit to "0" and setting the CKS2~CKS0 bits to "000" or "001" in the SMOD register.This will then use the low speed system 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 NORMAL Mode Switching

In SLOW Mode the system uses LIRC low speed system oscillator. To switch back to the NORMAL Mode, where the high speed system oscillator is used, the HLCLK bit should be set to "1" or HLCLK bit is "0", but CKS2~CKS0 is set to "010", "011", "100", "101", "110" or "111". As a certain amount of time will be required for the high frequency clock to stabilise, the status of the HTO bit is checked.



Entering the SLEEP Mode

There is only one way for the devices 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". 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 will remain with the clock source coming from the f_{LIRC} clock.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Entering the IDLE0 Mode

There is only one way for the devices 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 SMOD1 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 will be on.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Entering the IDLE1 Mode

There is only one way for the devices 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 SMOD1 register equal to "1". When this instruction is executed under the conditions described above, the following will occur:

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

Standby Current Considerations

As the main reason for entering the SLEEP or IDLE Mode is to keep the current consumption of the devices to as low a value as possible, perhaps only in the order of several micro-amps except in the IDLE1 Mode, there are other considerations which must also be taken into account by the circuit designer if the power consumption is to be minimised. Special attention must be made to the I/O pins on the devices. 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 devices which have different package types, as there may be unbonbed 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. In the IDLE1 Mode the system oscillator is on, if the system oscillator is from the high speed system oscillator, the additional standby current will also be perhaps in the order of several hundred micro-amps.

Wake-up

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

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

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

Each pin on Port A can be setup using the PAWU register to permit a negative transition on the pin to wake-up the system. When a Port A pin wake-up occurs, the program will resume execution at the instruction following the "HALT" instruction. If the system is woken up by an interrupt, then two possible situations may occur. The first is where the related interrupt is disabled or the interrupt is enabled but the stack is full, in which case the program will resume execution at the instruction following the "HALT" instruction. In this situation, the interrupt which woke-up the devices 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 f_{LIRC} clock which is supplied by the LIRC oscillator. The Watchdog Timer source clock is then subdivided by a ratio of 2^8 to 2^{15} to give longer timeouts, the actual value being chosen using the WS2~WS0 bits in the WDTC register. The LIRC internal oscillator has an approximate period of 32kHz at a supply voltage of 5V. However, it should be noted that this specified internal clock period can vary with V_{DD}, temperature and process variations.

Watchdog Timer Control Register

A single register, WDTC, controls the required timeout period as well as the enable/reset operation. When a WDTC register reset occurs, the WRF software reset flag in the SMOD1 register will be set high.

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 control
01010/10101: WDT enable
Other values: Reset MCU
When these bits are changed to any other values by the environmental noise to reset the microcontroller, the reset operation will be activated after 2~3 LIRC clock cycles and the WRF bit will be set to 1to indicate the reset source.

Bit 2~ 0 **WS2 ~ WS0**: WDT Time-out period selection
000: $2^8/f_{LIRC}$
001: $2^9/f_{LIRC}$
010: $2^{10}/f_{LIRC}$
011: $2^{11}/f_{LIRC}$(default)
100: $2^{12}/f_{LIRC}$
101: $2^{13}/f_{LIRC}$
110: $2^{14}/f_{LIRC}$
111: $2^{15}/f_{LIRC}$
These three bits determine the division ratio of the Watchdog Timer sourece clock, which in turn determines the timeout period.

SMOD1 Register

Bit	7	6	5	4	3	2	1	0
Name	FSYSON	—	—	—	—	LVRF	—	WRF
R/W	R/W	—	—	—	—	R/W	—	R/W
POR	0	—	—	—	—	x	—	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 Unimplemented, read as 0

Bit 0 **WRF**: WDT Control register software reset flag
0: Not occur
1: Occurred
This bit is set to 1 by the WDT Control register software reset and only can be cleared by the application program.

Watchdog Timer Operation

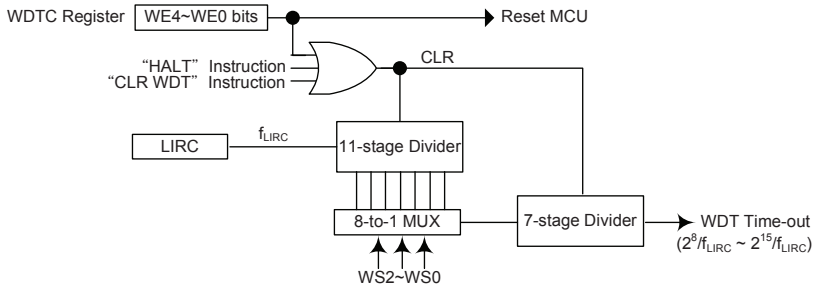
The Watchdog Timer operates by providing a device reset when its timer overflows. This means that in the application program and during normal operation the user has to strategically clear the Watchdog Timer before it overflows to prevent the Watchdog Timer from executing a reset. This is done using the clear watchdog instructions. If the program malfunctions for whatever reason, jumps to an unknown location, or enters an endless loop, the clear WDT instructions will not be executed in the correct manner, in which case the Watchdog Timer will overflow and reset the devices. The Watchdog Timer is always enabled, but there are five bits, WE4~WE0, in the WDTC register to give additional control of the Watchdog Timer.

WE4 ~ WE0 Bits	WDT Function
01010B or 10101B	Enable
Any other value	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 high and only the Program Counter and Stack Pointer will be reset. Three methods can be adopted to clear the contents of the Watchdog Timer. The first is a WDT reset, which means a value other than 01010B and 10101B is written into the WE4~WE0 bit locations, the second is using the Watchdog Timer software clear instructions and the third is via a HALT instruction. There is only one method of using software instruction to clear the Watchdog Timer. That is to use the single "CLR WDT" instruction to clear the WDT.

The maximum time-out period is when the 2^{15} division ratio is selected. As an example, with a 32 kHz LIRC oscillator as its source clock, this will give a maximum watchdog period of around 1 second for the 2^{15} division ratio, and a minimum timeout of 7.8ms for the 2^8 division ration.



Watchdog Timer

Reset and Initialisation

A reset function is a fundamental part of any microcontroller ensuring that the devices 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 microcontrollers to begin program execution from the lowest Program Memory address.

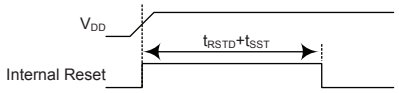
Another type of reset is when the Watchdog Timer overflows and resets the microcontrollers. All types of reset operations result in different register conditions being setup. Another reset exists in the form of a Low Voltage Reset, LVR, where a full reset is implemented in situations where the power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

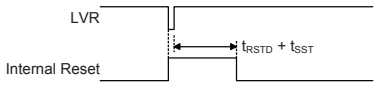
The most fundamental and unavoidable reset is the one that occurs after power is first applied to the microcontroller. As well as ensuring that the Program Memory begins execution from the first memory address, a power-on reset also ensures that certain other registers are preset to known conditions. 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.



Power-On Reset Timing Chart

Low Voltage Reset – LVR

The microcontrollers contain a low voltage reset circuit in order to monitor the supply voltage of the devices and provide an MCU reset should the value fall below a certain predefined level. The LVR function is always enabled during the normal and slow modes 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 SMOD1 register will also be set to1. 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 LVR 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} is 2.1V, the LVR will reset the device after 2~3 LIRC clock cycles. Note that the LVR function will be automatically disabled when the devices enter the SLEEP/IDLE mode.



Low Voltage Reset Timing Chart

- **SMOD1 Register**

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

"x" unknown

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

Bit 6~3 Unimplemented, read as 0

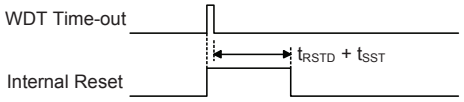
Bit 2 **LVRF**: LVR function reset flag
0: Not occur
1: Occurred
This bit can be clear to "0", but can not be set to "1".

Bit 1 Unimplemented, read as 0

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

Watchdog Time-out Reset during Normal Operation

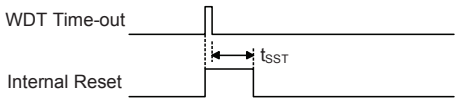
The Watchdog time-out Reset during normal operation is the same as an LVR reset except that the Watchdog time-out flag TO will be set to "1".



WDT Time-out Reset during Normal Operation Timing Chart

Watchdog Time-out Reset during SLEEP or IDLE Mode

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



WDT Time-out Reset during SLEEP or IDLE Timing Chart

Reset Initial Conditions

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

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

Note: "u" stands for unchanged

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

Item	Condition After RESET
Program Counter	Reset to zero
Interrupts	All interrupts will be disabled
WDT	Clear after reset, WDT begins counting
Timer Module	Timer Module 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 microcontrollers in different ways. To ensure reliable continuation of normal program execution after a reset occurs, it is important to know what conditions the microcontrollers are in after a particular reset occurs. The following table describes how each type of reset affects each of the microcontroller internal registers. Note that where more than one package type exists the table will reflect the situation for the larger package type.

HT45F3420

Register	Reset (Power On)	WDT Time-out (Normal Operation)	LVR Reset (Normal Operation)	WDT Time-out (HALT)*
MP0	1xxx xxxx	1xxx xxxx	1xxx xxxx	1uuu uuuu
MP1	1xxx xxxx	1xxx xxxx	1xxx xxxx	1uuu uuuu
BP	---- ---0	---- ---0	---- ---0	---- ---u
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	--xx xxxx	--uu uuuu	--uu uuuu	--uu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--11 uuuu
SMOD	000- 0011	000- 0011	000- 0011	uuu- uuuu
INTEG	---- --00	---- --00	---- --00	---- --uu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	00-0 00-0	00-0 00-0	00-0 00-0	uu-u uu-u
INTC2	--00 --00	--00 --00	--00 --00	--uu --uu
MFI	--00 --00	--00 --00	--00 --00	--uu --uu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS0	---- -000	---- -000	---- -000	---- -uuu
WDTC	0101 0011	0101 0011	0101 0011	uuuu uuuu
TBC	0011 -111	0011 -111	0011 -111	uuuu –uuu
SMOD1	0--- -x-0	0--- -x-0	0--- -x-0	u--- -u-u
EEA	---0 0000	---0 0000	---0 0000	---u uuuu
EED	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADOL (ADRFS=0)	xxxx ----	xxxx ----	xxxx ----	uuuu ----
SADOL (ADRFS=1)	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SADOH (ADRFS=0)	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu

Register	Reset (Power On)	WDT Time-out (Normal Operation)	LVR Reset (Normal Operation)	WDT Time-out (HALT)*
SADOH (ADRFS=1)	---- xxxx	---- xxxx	---- xxxx	---- uuuu
SADC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADC1	0000 -000	0000 -000	0000 -000	uuuu -uuu
SADC2	0--0 0000	0--0 0000	0--0 0000	0--0 0000
PAS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS1	0000 00--	0000 00--	0000 00--	uuuu uu--
STMC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDH	---- --00	---- --00	---- --00	---- --uu
STMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAH	---- --00	---- --00	---- --00	---- --uu
PWMP	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWMD	0000 0000	0000 0000	0000 0000	uuuu uuuu
DLL	0000 ---0	0000 ---0	0000 ---0	uuuu ---u
CPR	1000 0000	1000 0000	1000 0000	uuuu uuuu
CPOR	0010 0000	0010 0000	0010 0000	uuuu uuuu
OCPC0	0000 0--0	0000 0--0	0000 0--0	uuuu u--u
OCPC1	--00 0000	--00 0000	--00 0000	--uu uuuu
OCPDA	0000 0000	0000 0000	0000 0000	uuuu uuuu
OCPOCAL	0010 0000	0010 0000	0010 0000	uuuu uuuu
OCPCCAL	0001 0000	0001 0000	0001 0000	uuuu uuuu
OVPC0	--00 0000	--00 0000	--00 0000	--uu uuuu
OVPC1	0001 0000	0001 0000	0001 0000	uuuu uuuu
OVPDA	0000 0000	0000 0000	0000 0000	uuuu uuuu
EEC	---- 0000	---- 0000	---- 0000	---- uuuu

Note: "*" stands for warm reset

"-" not implemented

"u" stands for "unchanged"

"x" stands for "unknown"

HT45F3430

Register	Reset (Power On)	WDT Time-out (Normal Operation)	LVR Reset (Normal Operation)	WDT Time-out (HALT)*
MP0	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
MP1	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
BP	---- ---0	---- ---0	---- ---0	---- ---u
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	-xxx xxxx	-uuu uuuu	-uuu uuuu	-uuu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--11 uuuu
SMOD	000- 0011	000- 0011	000- 0011	uuu- uuuu
INTEG	---- 0000	---- 0000	---- 0000	---- uuuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	00-0 00-0	00-0 00-0	00-0 00-0	uu-u uu-u
INTC2	-000 -000	-000 -000	-000 -000	-uuu -uuu
MFI	--00 --00	--00 --00	--00 --00	--uu --uu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS0	--00 0000	--00 0000	--00 0000	--uu uuuu
WDTC	0101 0011	0101 0011	0101 0011	uuuu uuuu
TBC	0011 -111	0011 -111	0011 -111	uuuu –uuu
SMOD1	0--- -x-0	0--- -x-0	0--- -x-0	u--- -u-u
EEA	--00 0000	--00 0000	--00 0000	--uu uuuu
EED	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADOL (ADRFS=0)	xxxx ----	xxxx ----	xxxx ----	uuuu ----
SADOL (ADRFS=1)	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SADOH (ADRFS=0)	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SADOH (ADRFS=1)	---- xxxx	---- xxxx	---- xxxx	---- uuuu
SADC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADC1	0000 -000	0000 -000	0000 -000	uuuu -uuu
SADC2	0--0 0000	0--0 0000	0--0 0000	u--u uuuu
PAS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS1	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDH	---- --00	---- --00	---- --00	---- --uu
STMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAH	---- --00	---- --00	---- --00	---- --uu
PWMP	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWMD	0000 0000	0000 0000	0000 0000	uuuu uuuu
DLL	0000 ---0	0000 ---0	0000 ---0	uuuu ---u
CPR	1000 0000	1000 0000	1000 0000	uuuu uuuu

Register	Reset (Power On)	WDT Time-out (Normal Operation)	LVR Reset (Normal Operation)	WDT Time-out (HALT)*
CPOR	0010 0000	0010 0000	0010 0000	uuuu uuuu
OCPC0	0000 0--0	0000 0--0	0000 0--0	uuuu u--u
OCPC1	--00 0000	--00 0000	--00 0000	--uu uuuu
OCPDA	0000 0000	0000 0000	0000 0000	uuuu uuuu
OCPOCAL	0010 0000	0010 0000	0010 0000	uuuu uuuu
OCPCCAL	0001 0000	0001 0000	0001 0000	uuuu uuuu
OVPC0	--00 0000	--00 0000	--00 0000	--uu uuuu
OVPC1	0001 0000	0001 0000	0001 0000	uuuu uuuu
OVPDA	0000 0000	0000 0000	0000 0000	uuuu uuuu
EEC	---- 0000	---- 0000	---- 0000	---- uuuu
PB	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PC	--11 1111	--11 1111	--11 1111	--uu uuuu
PCC	--11 1111	--11 1111	--11 1111	--uu uuuu
PCPU	--00 0000	--00 0000	--00 0000	--uu uuuu
PBS0	---- ---0	---- ---0	---- ---0	---- ---u
PCS0	--00 0000	--00 0000	--00 0000	--uu uuuu
ASCR	---- 0000	---- 0000	---- 0000	---- uuuu
SCOMC	-000 ----	-000 ----	-000 ----	-uuu ----
SLEDC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SLEDC1	---- --00	---- --00	---- --00	---- --00

Note: "*" stands for warm reset

"-" not implemented

"u" stands for "unchanged"

"x" stands for "unknown"

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 devices provide bidirectional input/output lines labeled with port names PA~PC. 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.

Device	Register Name	Bit							
		7	6	5	4	3	2	1	0
HT45F3420/ HT45F3430	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
HT45F3430	PB	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
	PBC	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
	PBPU	PBPU7	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
	PC	—	—	PC5	PC4	PC3	PC2	PC1	PC0
	PCC	—	—	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
	PCPU	—	—	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0

I/O Control 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 registers PAPU~PCPU, and are implemented using weak PMOS transistors.

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

PAPU Register

Bit	7	6	5	4	3	2	1	0
Name	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

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

PBPU Register – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	PBPU7	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

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

PCPU Register – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
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 Port C bit 5 ~ bit 0 Pull-high Control
0: Disable
1: Enable

Port A Wake-up

The HALT instruction forces the microcontroller into the SLEEP or IDLE Mode which preserves power, a feature that is important for battery and other low-power applications. Various methods exist to wake-up the microcontroller, one of which is to change the logic condition on one of the Port A pins from high to low. This function is especially suitable for applications that can be woken up via external switches. Each pin on Port A can be selected individually to have this wake-up feature using the PAWU register. Note that the wake-up function can be controlled by the wake-up control registers only when the pin-shared functional pin is selected as general purpose input/output and the MCU enters the Power down 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 Port A bit 7 ~ bit 0 Wake-up Control
0: Disable
1: Enable

I/O Port Control Registers

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

PAC Register

Bit	7	6	5	4	3	2	1	0
Name	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

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

PBC Register – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

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

PCC Register – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	1	1	1	1	1	1

Bit 7~6 Unimplemented, read as "0"

Bit 5~0 Port C bit 5 ~ bit 0 Input/Output Control
0: Output
1: Input

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 some pins, the chosen function of the multi-function I/O pins is set by 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 devices include Port A~C function selection register PxS0 and PAS1, which can select the desired functions of the multi-function pin-shared pins and there is another register IFS0, which can select the input function source pin.

The most important point to note is to make sure that the desired pin-shared function is properly selected and also deselected. 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. To correctly deselect the pin-shared function, the peripheral function should first be disabled and then the corresponding pin-shared function control register can be modified to select other pin-shared functions.

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	PAS17	PAS16	PAS15	PAS14	PAS13	PAS12	—	—
IFS0	—	—	—	—	—	IFS02	IFS01	IFS00

Pin-shared Function Selection Register List – HT45F3420

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	PAS17	PAS16	PAS15	PAS14	PAS13	PAS12	PAS11	PAS10
PBS0	—	—	—	—	—	—	—	PBS00
PCS0	—	—	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
IFS0	—	—	IFS05	IFS04	IFS03	IFS02	IFS01	IFS00

Pin-shared Function Selection Register List – HT45F3430

- **PAS0 Register – HT45F3420**

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

Bit 7~6 **PAS07~PAS06**: PA3 Pin-shared function selection
00: PA3
01: PA3
10: VREF
11: VREF/AN3

Bit 5~4 **PAS05~PAS04**: PA2 Pin-shared function selection
00: PA2
01: PA2
10: PA2
11: VBAT/AN2

Bit 3~2 **PAS03~PAS02**: PA1 Pin-shared function selection
00: PA1
01: PA1
10: PA1
11: OVP/AN1

Bit 1~0 **PAS01~PAS00**: PA0 Pin-shared function selection
00: PA0
01: PA0
10: PA0
11: OCP/AN0

- **PAS0 Register – HT45F3430**

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

Bit 7~6 **PAS07~PAS06**: PA3 Pin-shared function selection
00: PA3
01: PA3
10: PA3
11: OVP/AN3

Bit 5~4 **PAS05~PAS04**: PA2 Pin-shared function selection
00: PA2/STPI_0
01: PA2/STPI_0
10: STP
11: AN2

Bit 3~2 **PAS03~PAS02**: PA1 Pin-shared function selection
00: PA1
01: PA1
10: PA1
11: OCP_0/AN1

Bit 1~0 **PAS01~PAS00**: PA0 Pin-shared function selection
00: PA0
01: PA0
10: PA0
11: AN0

- **PAS1 Register – HT45F3420**

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

Bit 7~6 **PAS17~PAS16**: PA7 Pin-shared function selection
00: PA7/STPI
01: PA7/STPI
10: PA7/STPI
11: STP

Bit 5~4 **PAS15~PAS14**: PA6 Pin-shared function selection
00: PA6
01: PA6
10: PA6
11: PWMH

Bit 3~2 **PAS13~PAS12**: PA5 Pin-shared function selection
00: PA5
01: PA5
10: PA5
11: PWML

Bit 1~0 Unimplemented, read as "0"

- **PAS1 Register – HT45F3430**

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

Bit 7~6 **PAS17~PAS16**: PA7 Pin-shared function selection
00: PA7
01: PA7
10: VREF
11: VREFI/AN7

Bit 5~4 **PAS15~PAS14**: PA6 Pin-shared function selection
00: PA6
01: PA6
10: PA6
11: OCP_1/AN6

Bit 3~2 **PAS13~PAS12**: PA5 Pin-shared function selection
00: PA5
01: PA5
10: PA5
11: AN5

Bit 1~0 **PAS11~PAS10**: PA4 Pin-shared function selection
00: PA4
01: PA4
10: PA4
11: VBAT/AN4

- **PBS0 Register – HT45F3430**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	PBS00
R/W	—	—	—	—	—	—	—	R/W
POR	—	—	—	—	—	—	—	0

Bit 7~1 Unimplemented, read as 0

Bit 0 **PBS00**: PB0 Pin-shared function selection
0: PB0/STPI_1
1: STP

- **PCS0 Register – HT45F3430**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
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 **PCS05**: PC5 Pin-shared function selection
0: PC5
1: PWMH

Bit 4 **PCS04**: PC4 Pin-shared function selection
0: PC4
1: PWML

Bit 3 **PCS03**: PC3 Pin-shared function selection
0: PC3
1: SCOM3

Bit 2 **PCS02**: PC2 Pin-shared function selection
0: PC2
1: SCOM2

Bit 1 **PCS01**: PC1 Pin-shared function selection
0: PC1
1: SCOM1

Bit 0 **PCS00**: PC0 Pin-shared function selection
0: PC0
1: SCOM0

- **IFS0 Register – HT45F3420**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	IFS02	IFS01	IFS00
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 Unimplemented, read as "0"

Bit 2 **IFS02**: OVP input source selection
0: OVP_IN1
1: OVP_IN2
The IFS02 bit is used to select the OVP input signal. The OVP_IN1 signal is from the external OVP pin directly while the OVP_IN2 signal is from the OVP pin integrated voltage divider circuit. Refer to the "Integrated Voltage Divider Circuits" section for details.

Bit 1 **IFS01**: STCK source selection
0: PA7
1: PA4

Bit 0 **IFS00**: INT0 source selection
0: PA7
1: PA4

- **IFS0 Register – HT45F3430**

Bit	7	6	5	4	3	2	1	0
Name	—	—	IFS05	IFS04	IFS03	IFS02	IFS01	IFS00
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 **IFS05**: STPI input source selection
0: PA2
1: PB0

Bit 4 **IFS04**: INT1 input source selection
0: PA6
1: PB3

Bit 3 **IFS03**: OCP input source selection
0: PA1
1: PA6

Bit 2 **IFS02**: OVP input source selection
0: OVP_IN1
1: OVP_IN2
The IFS02 bit is used to select the OVP input signal. The OVP_IN1 signal is from the external OVP pin directly while the OVP_IN2 signal is from the OVP pin integrated voltage divider circuit. Refer to the "Integrated Voltage Divider Circuits" section for details.

Bit 1 **IFS01**: STCK source selection
0: PA5
1: PB1

Bit 0 **IFS00**: INT0 input source selection
0: PA5
1: PB2

I/O Port Source Current Control – only for HT45F3430

The HT45F3430 supports different source current driving capability for each I/O port. With the corresponding selection register, SLEDC0 and SLEDC1, each I/O port can support four levels of the source current driving capability. It can be used in LED driving applications. Users should refer to the D.C. characteristics section to select the desired source current for different applications.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SLEDC0	PBPS3	PBPS2	PBPS1	PBPS0	PAPS3	PAPS2	PAPS1	PAPS0
SLEDC1	—	—	—	—	—	—	PCPS1	PCPS0

I/O Port Source Current Control Register List

SLEDC0 Register

Bit	7	6	5	4	3	2	1	0
Name	PBPS3	PBPS2	PBPS1	PBPS0	PAPS3	PAPS2	PAPS1	PAPS0
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 **PBPS3~PBP2**: PB7~PB4 source current selection
00: Source current=Level 0 (min.)
01: Source current=Level 1
10: Source current=Level 2
11: Source current=Level 3 (max.)

Bit 5~4 **PBPS1~PBP0**: PB3~PB0 source current selection
00: Source current=Level 0 (min.)
01: Source current=Level 1
10: Source current=Level 2
11: Source current=Level 3 (max.)

Bit 3~2 **PAPS3~PAP2**: PA7~PA4 source current selection (PMOS adjust)
00: Source current=Level 0 (min.)
01: Source current=Level 1
10: Source current=Level 2
11: Source current=Level 3 (max.)

Bit 1~0 **PAPS1~PAP0**: PA3~PA0 source current selection (PMOS adjust)
00: Source current=Level 0 (min.)
01: Source current=Level 1
10: Source current=Level 2
11: Source current=Level 3 (max.)

Note: Users should refer to the D.C. Characteristics section to obtain the exact value for different applications.

SLEDC1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PCPS1	PCPS0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

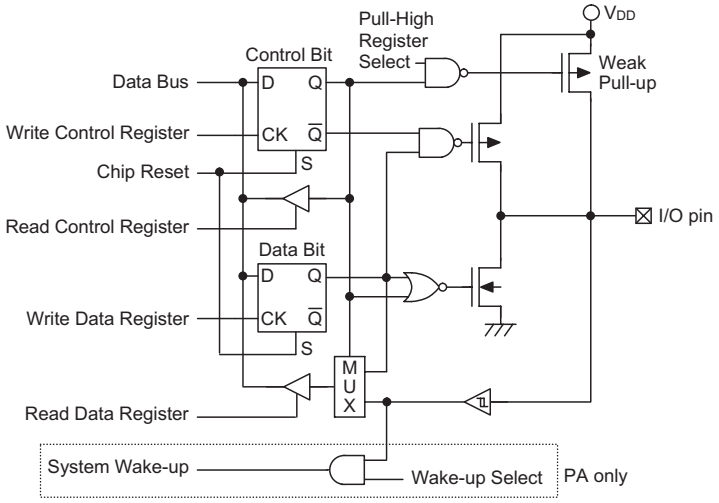
Bit 7~2 Reserved bits, read/writable

Bit 1~0 **PCPS1~PCPS0**: PC3~PC0 source current selection (PMOS adjust)
00: Source current=Level 0 (min.)
01: Source current=Level 1
10: Source current=Level 2
11: Source current=Level 3 (max.)

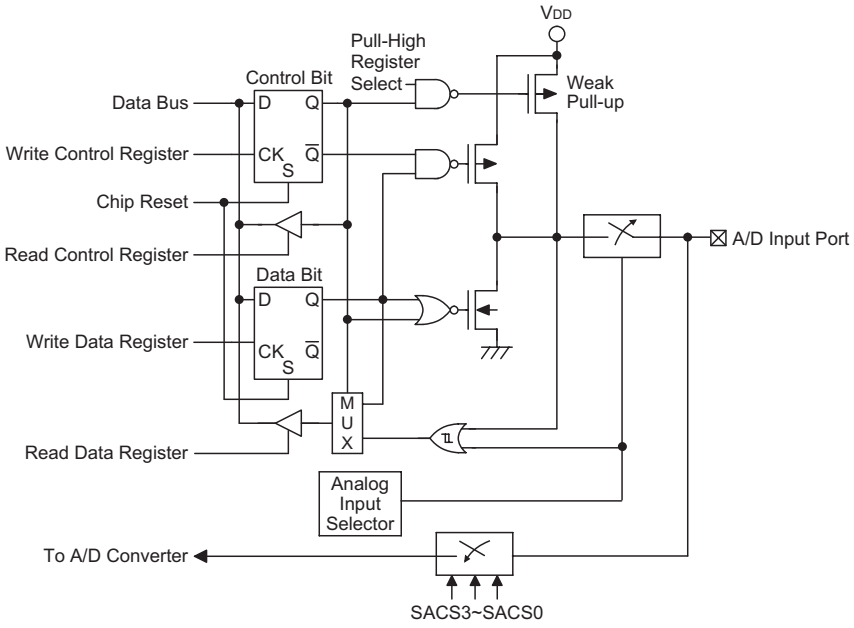
Note: Users should refer to the D.C. Characteristics section to obtain the exact value for different applications.

I/O Pin Structures

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



Generic Input/Output Structure



A/D 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 register, PAC~PCC, are then programmed to setup some pins as outputs, these output pins will have an initial high output value unless the associated port data register, PA~PC, are first programmed. Selecting which pins are inputs and which are outputs can be achieved byte-wide by loading the correct values into the appropriate port control register or by programming individual bits in the port control register using the "SET [m].i" and "CLR [m].i" instructions. Note that when using these bit control instructions, a read-modify-write operation takes place. The microcontroller must first read in the data on the entire port, modify it to the required new bit values and then rewrite this data back to the output ports.

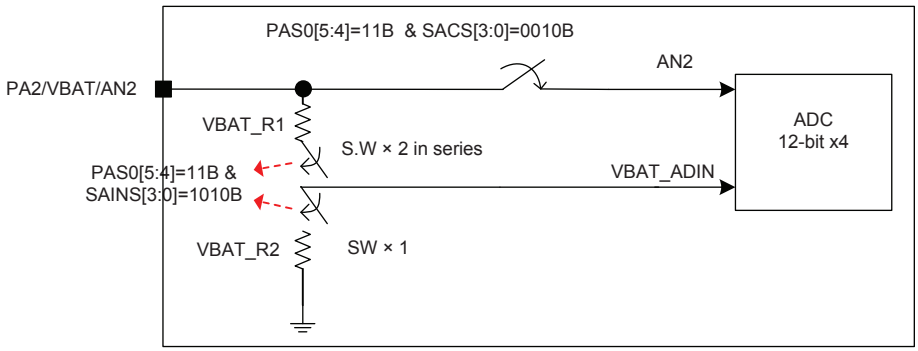
Port A has the additional capability of providing wake-up functions. When the device is in the SLEEP or IDLE Mode, various methods are available to wake the device up. One of these is a high to low transition of any of the Port A pins. Single or multiple pins on Port A can be setup to have this function.

Integrated Voltage Divider Circuits

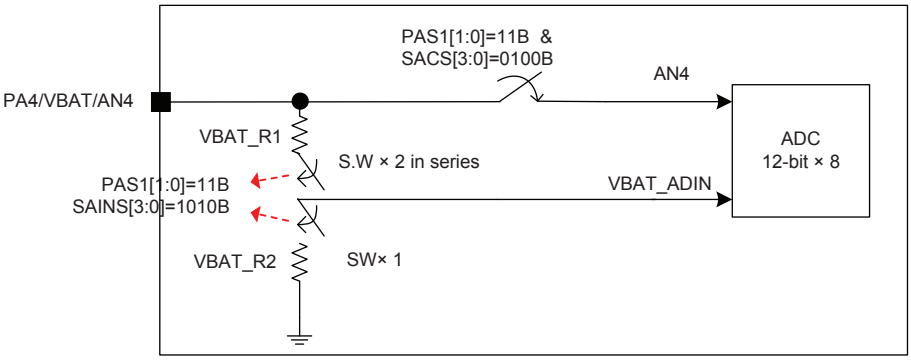
To measure the external battery voltage and to reduce the need for external components, the devices include a fully internal resistor divider circuit.

Functional Description

These resistor divider internal resistors can be connected or disconnected using register controlled internal analog switches.



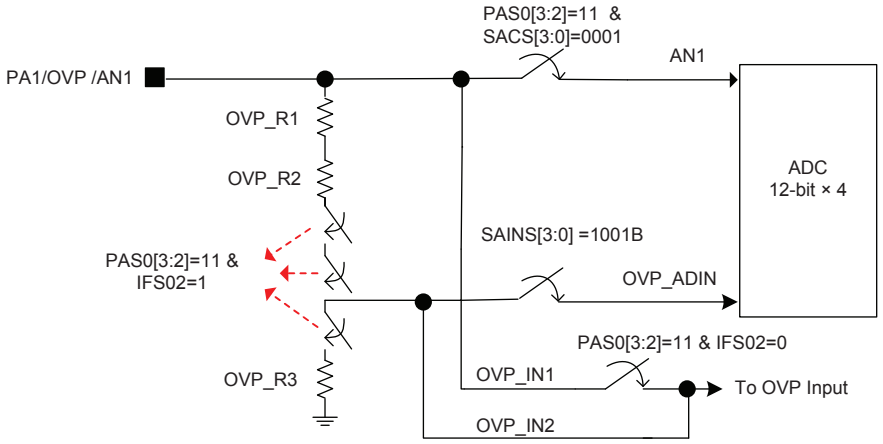
VBAT Pin Voltage Divider Circuit – HT45F3420



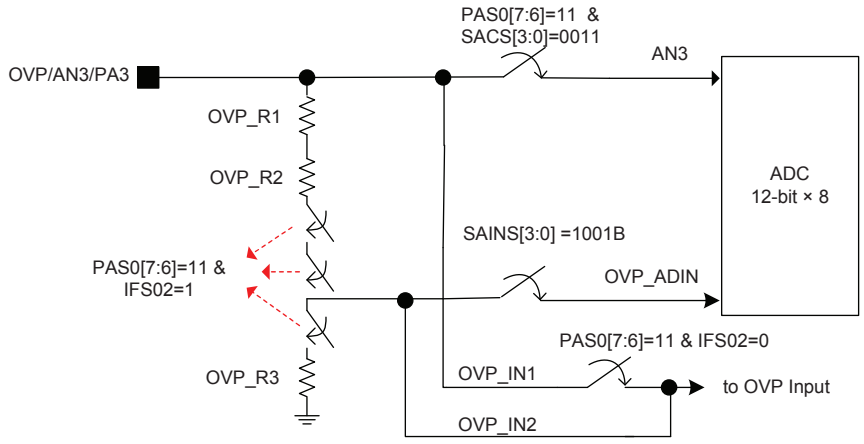
VBAT Pin Voltage Divider Circuit – HT45F3430

The battery voltage input pin, VBAT, pin is connected to an internal resistor divider circuit, VBAT_R1with a value of 2K and VBAT_R2 with a value of 1K. This gives a division ratio of 2:1, the accuracy of which must be equal to 1%. An input voltage of 4.2V on VBAT will generate a 1.4V voltage using this voltage divider.

When the SADC1 register bits SAINS[3:0] =1010B, then the switches between the VBAT_R1 and VBAT_R2 resistors will all be on. Now VBAT_R1 and VBAT_R2 will be connected directly and the divided voltage will be supplied to VBAT_ADIN. When SAINS[3:0] ≠1010B, then the switches between the VBAT_R1 and VBAT_R2 resistors will all be off.



OVP Pin Voltage Divider Circuit – HT45F3420



OVP Pin Voltage Divider Circuit – HT45F3430

For the over voltage protection function, the OVP pin is connected to an internal voltage divider. This is composed of resistors OVP_R1+OVP_R2, with a combined value of 2K, and OVP_R3 with a value of 1K. This gives a division ratio of 2:1, the accuracy of which must be equal to 1%. The total summed resistance of OVP_R1+ OVP_R2+ OVP_R3 should be 3k however this total resistance value can have a tolerance of 50%.

When the IFS0 register bit IFS02 =1, then the switches between the OVP_R2 and OVP_R3 resistors will all be on. Now the OVP_R2 and OVP_R3 resistors will be connected and the obtained divided voltage will be supplied to OVP_IN2. When IFS02=0, the switches between the OVP_R2 and OVP_R3 resistors will all be off.

Timer Module – TM

One of the most fundamental functions in any microcontroller device is the ability to control and measure time. To implement time related functions the devices include a Standard Timer Module, abbreviated to the name STM. The STM is multi-purpose timing unit and serves 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.The TM has two individual interrupts. The addition of input and output pins for the TM ensures that users are provided with timing units with a wide and flexible range of features.

The general features of the Standard type TM are described here with more detailed information provided in the Standard TM section.

Introduction

The devices contain one TM, namely Standard Type TM. The common features to the Standard TM will be described in this section and the detailed operation will be described in the Standard Type TM section. The main features of the STM are summarised in the accompanying table.

Function	STM
Timer/Counter	√
I/P Capture	√
Compare Match Output	√
PWM Channels	1
Single Pulse Output	1
PWM Alignment	Edge
PWM Adjustment Period & Duty	Duty or Period

TM Function Summary

TM Operation

The standard type TM offers 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 Sources

The clock source which drives the main counter in the TM can originate from various sources. The selection of the required clock source is implemented using the STCK2~STCK0 bits in the STMC0 control register. The clock source can be a ratio of the system clock f_{SYS} or the internal high clock f_H, the f_{TBC} clock source or the external STCK pin. The STCK 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 type TM has two internal interrupts, the internal comparator A or comparator P, which generate a TM interrupt when a compare match condition occurs. When a TM interrupt is generated, it can be used to clear the counter and also to change the state of the TM output pin.

TM External Pins

The TM has two TM input pin, with the label STCK and STPI respectivety. The TM input pin STCK, is essentially a clock source for the TM and is selected using the STCK2~STCK0 bits in the STMC0 register. This external TM input pin allows an external clock source to drive the internal TM. The TM input pin can be chosen to have either a rising or falling active edge. The STCK pin is also used as the external trigger input pin in single pulse mode. Another TM input pin STPI, is the capture input pin whose active edge can be a rising edge, a falling edge or both rising and falling edges and the active edge transition type is selected using the STIO1~STIO0 bits in the STMC1 register.

The TM also has an output pin, STP. When the TM is in the Compare Match Output Mode, the STP pin can be used as output pin and controlled by the TM to switch to a high or low level or to toggle when a compare match situation occurs. The external STP pin is also the output pin where the TM generates the PWM output waveform.

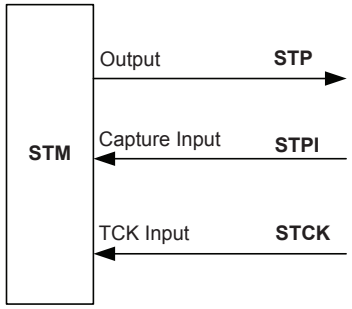
As the STM pins are pin-shared with other functions, the TM function must first be setup using registers. A single bit in one of the registers determines if its associated pin is to be used as an external TM pin or if it is to have another function.

Device	STM	
	Input Pins	Output Pin
HT45F3420/HT45F3430	STCK STPI	STP

TM External Pins

TM Input/Output Pin Control Register

Selecting to have a TM input/output or whether to retain its other shared function is implemented using one register, with a single bit in each register corresponding to a TM input/output pin. Configuring the selection bits correctly will setup the corresponding pin as a TM input/output. The details of the pin-shared function selection are described in the pin-shared function section.

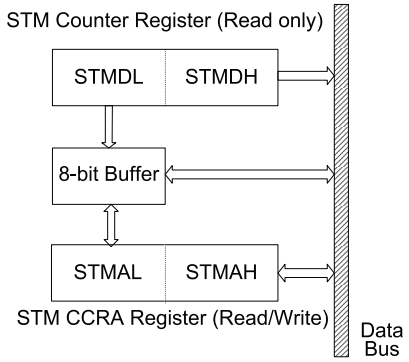


STM Function Pin Control Block Diagram

Programming Considerations

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

As the CCRA register is implemented in the way shown in the following diagram and accessing the register is carried out using the following access procedures. Accessing the CCRA low byte register without following these access procedures will result in unpredictable values.

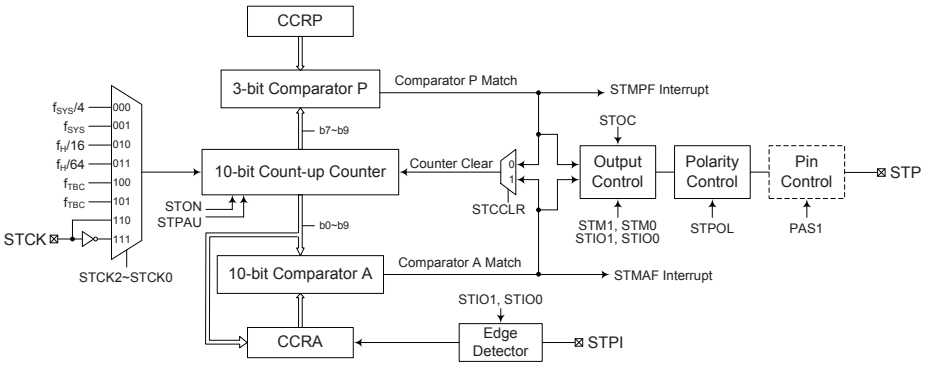


The following steps show the read and write procedures:

- Writing Data to CCRA
  - Step 1. Write data to Low Byte STMAL
    – note that here data is only written to the 8-bit buffer.
  - Step 2. Write data to High Byte STMAH
    – here data is written directly to the high byte registers and simultaneously data is latched from the 8-bit buffer to the Low Byte registers.
- Reading Data from the Counter Registers and CCRA
  - Step 1. Read data from the High Byte STMDH or STMAH
    – 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 STMDL or STMAL
    – 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 one external output pin.



Note: For HT45F3430, the STCK and STPI input source pins are selectable which are selected by the IFS0 register bits.

Standard Type TM Block Diagram

Standard Type TM Operation

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

The only way of changing the value of the 10-bit counter using the application program, is to clear the counter by changing the STON bit from low to high. The counter will also be cleared automatically by a counter overflow or a compare match with one of its associated comparators. When these conditions occur, a TM 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 and can also control an output pin. All operating setup conditions are selected using relevant internal registers.

Standard Type TM Registers

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

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

10-bit Standard Type TM Register List

STMC0 Register

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

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

Bit 6~4 **STCK2~STCK0**: Select STM Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{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_H and f_{TBC} are other internal clocks, the details of which can be found in the oscillator section.

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

Bit 2~0 **STRP2~ STRP0**: STM CCRP 3-bit register, compared with the STM Counter bit 9~bit 7 Comparator P Match Period
000: 1024 STM clocks
001: 128 STM clocks
010: 256 STM clocks
011: 384 STM clocks
100: 512 STM clocks
101: 640 STM clocks
110: 768 STM clocks
111: 896 STM clocks
These three bits are used to setup the value on the internal CCRP 3-bit register, which are then compared with the internal counter's highest three bits. The result of this comparison can be selected to clear the internal counter if the STCCLR bit is set to zero. Setting the STCCLR bit to zero ensures that a compare match with the CCRP values will reset the internal counter. As the CCRP bits are only compared with the highest three counter bits, the compare values exist in 128 clock cycle multiples. Clearing all three bits to zero is in effect allowing the counter to overflow at its maximum value.

STMC1 Register

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

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

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

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

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

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

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

Timer/counter Mode:
Unused

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

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

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

Bit 3 **STOC**: STM 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 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 Output polarity Control
0: Non-invert
1: Invert
This bit controls the polarity of the STM output pin. When the bit is set high the STM output pin will be inverted and not inverted when the bit is zero. It has no effect if the STM is in the Timer/Counter Mode.

Bit 1 **STDPX**: STM PWM period/duty Control
0: CCRP - period; CCRA - duty
1: CCRP - duty; CCRA - period
This bit, determines which of the CCRA and CCRP registers are used for period and duty control of the PWM waveform.

Bit 0 **STCCLR**: Select STM Counter clear condition
0: STM Comparator P match
1: STM Comparator A match
This bit is used to select the method which clears the counter. Remember that the Standard STM 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 or Input Capture 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 STM Counter Low Byte Register bit 7 ~ bit 0
STM 10-bit Counter bit 7 ~ bit 0

STMDH Register

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

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 STM Counter High Byte Register bit 1 ~ bit 0
STM 10-bit Counter bit 9 ~ bit 8

STMAL Register

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

Bit 7~0 STM CCRA Low Byte Register bit 7 ~ bit 0
STM 10-bit Counter bit 7 ~ bit 0

STMAH Register

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

Bit 7~2 Unimplemented, read as "0"
Bit 1~0 STM CCRA High Byte Register bit 1 ~ bit 0
STM 10-bit Counter bit 9 ~ bit 8

Standard Type TM Operating Modes

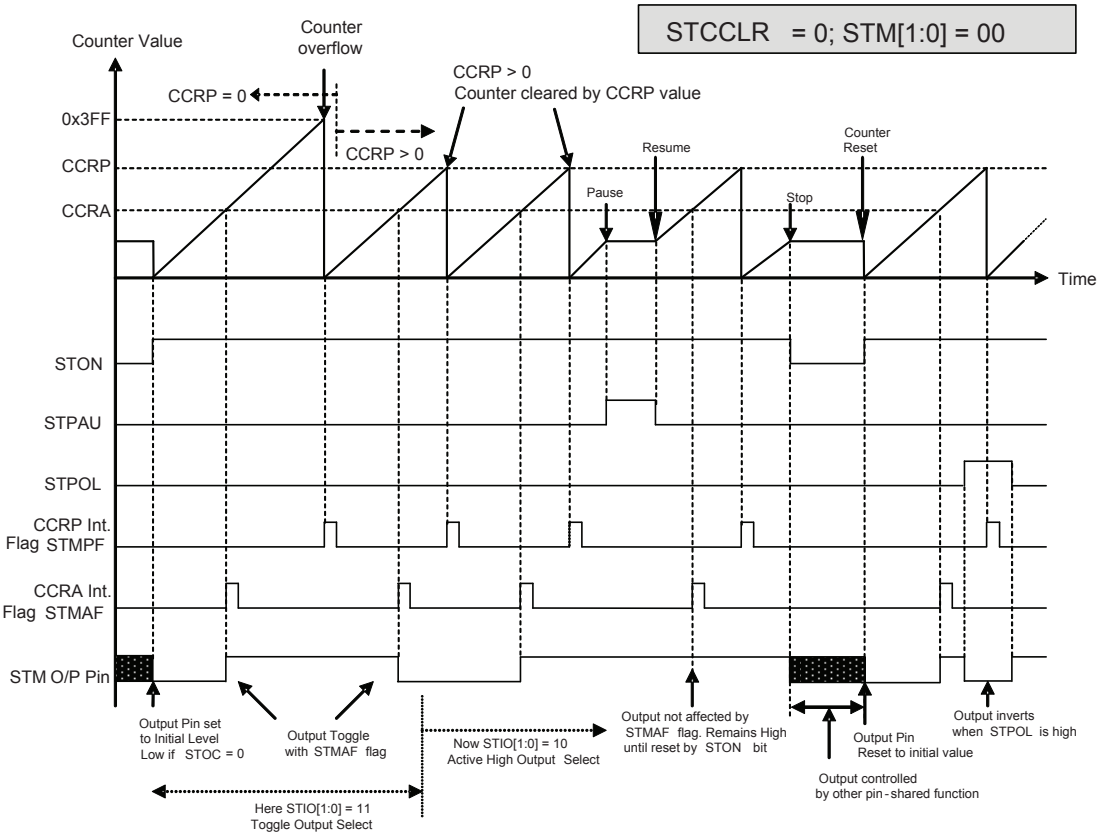
The Standard Type TM can operate in one of five operating modes, Compare Match Output Mode, PWM Output Mode, Single Pulse Output Mode, Capture Input Mode or Timer/Counter Mode. The operating mode is selected using the STM1 and STM0 bits in the STMC1 register.

Compare 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 cannot be set to "0". If the CCRA bits are all zero, the counter will overflow when it reaches its maximum 10-bit, 3FF Hex, value, however here the STMAF interrupt request flag will not be generated.

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

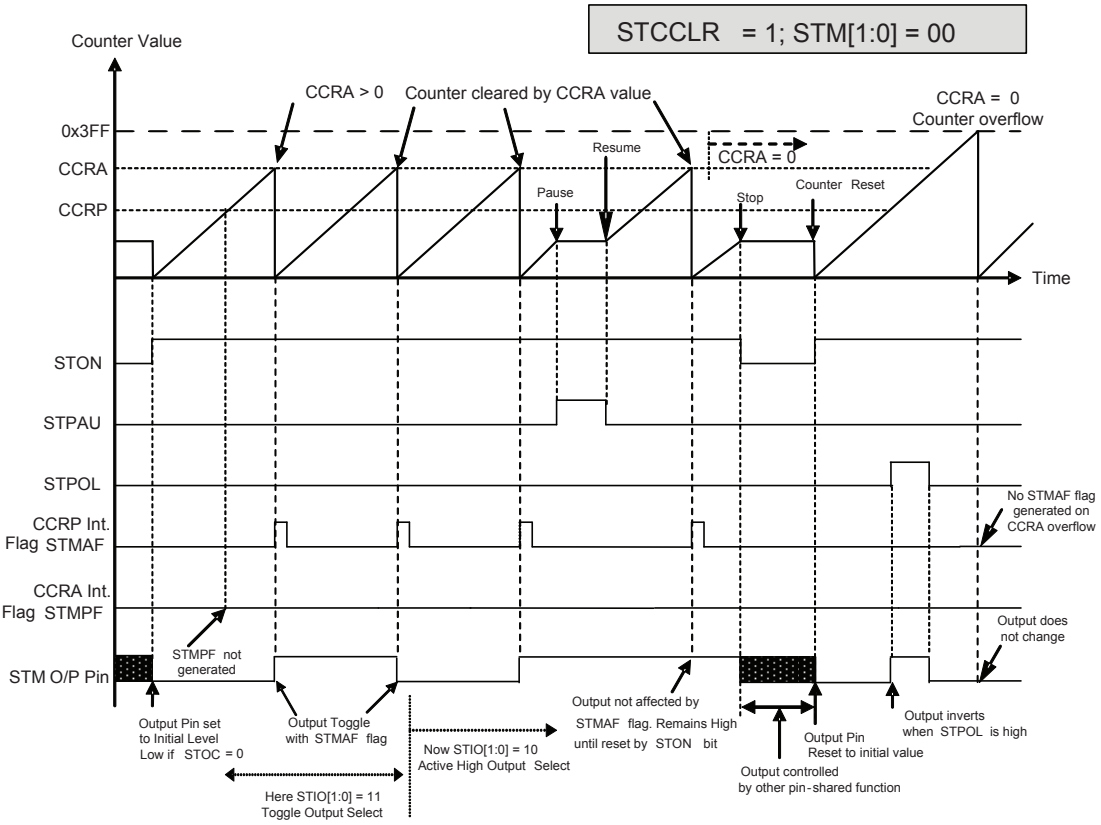


Compare Match Output Mode – STCCLR=0

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

2. The TM output pin controlled only by the STMAF flag

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



Compare Match Output Mode – STCCLR=1

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

2. The TM output pin controlled only by the STMAF flag

3. The output pin reset to initial state by a STON 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 function by setting pin-share function register.

PWM Output Mode

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

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

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

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

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

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

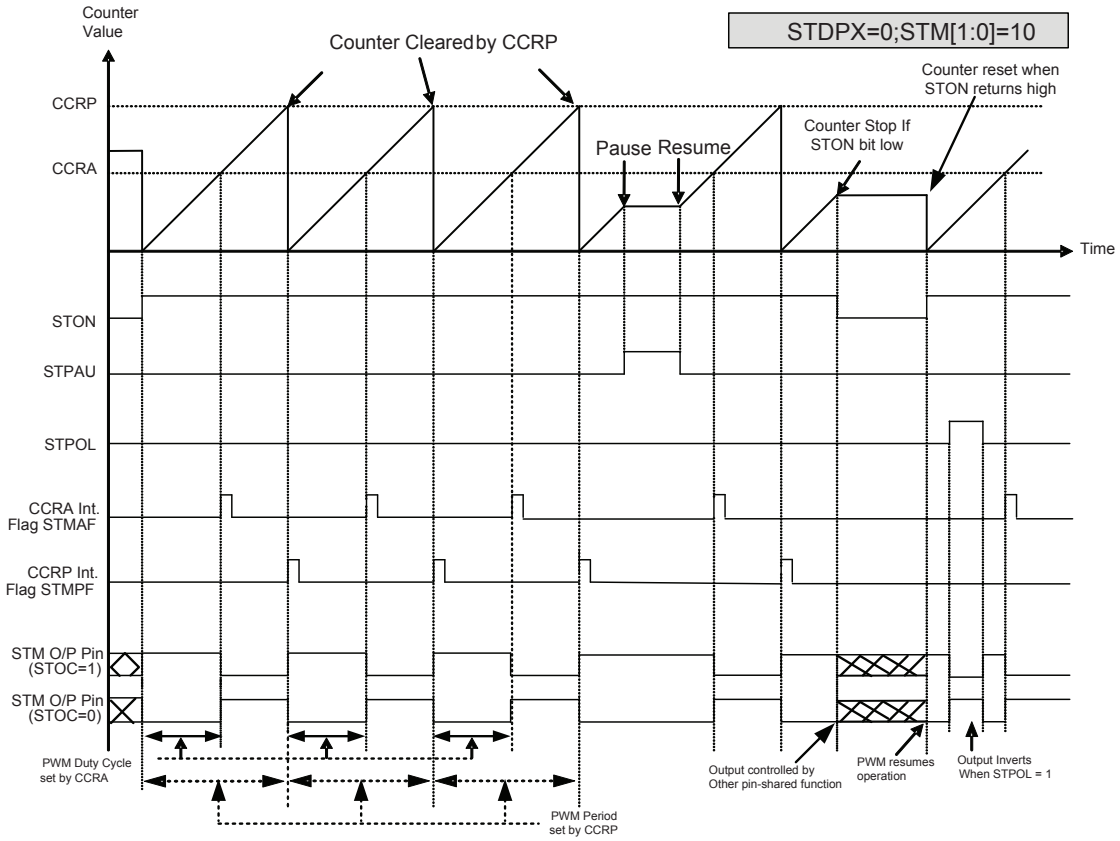
The STM PWM output frequency=(f_{SYS}/4) / 512=f_{SYS}/2048=1.9531kHz, duty=128/512=25%.

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%.

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

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

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



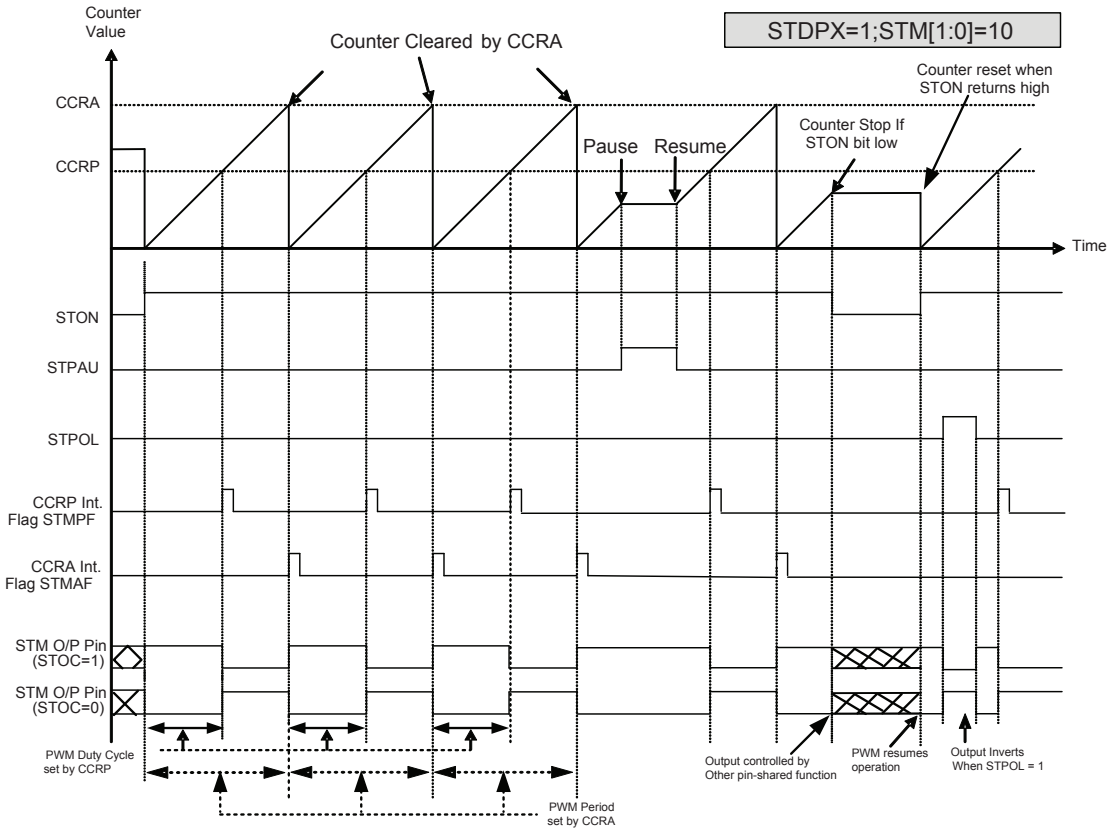
PWM Output Mode – STDPX=0

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

2. A counter clear sets 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 PWM Period

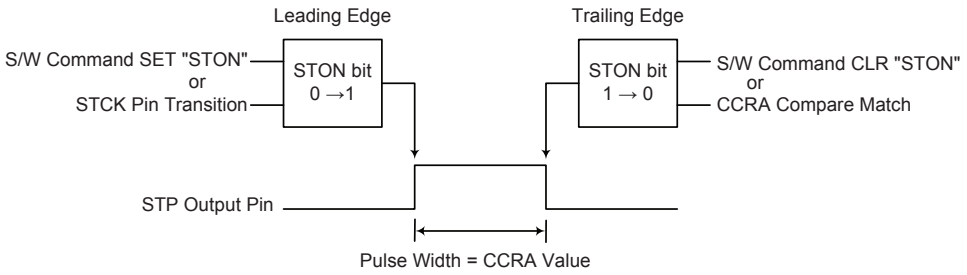
3. The internal PWM function continues even when STIO[1:0]=00 or 01

4. The STCCLR bit has no influence on PWM operation

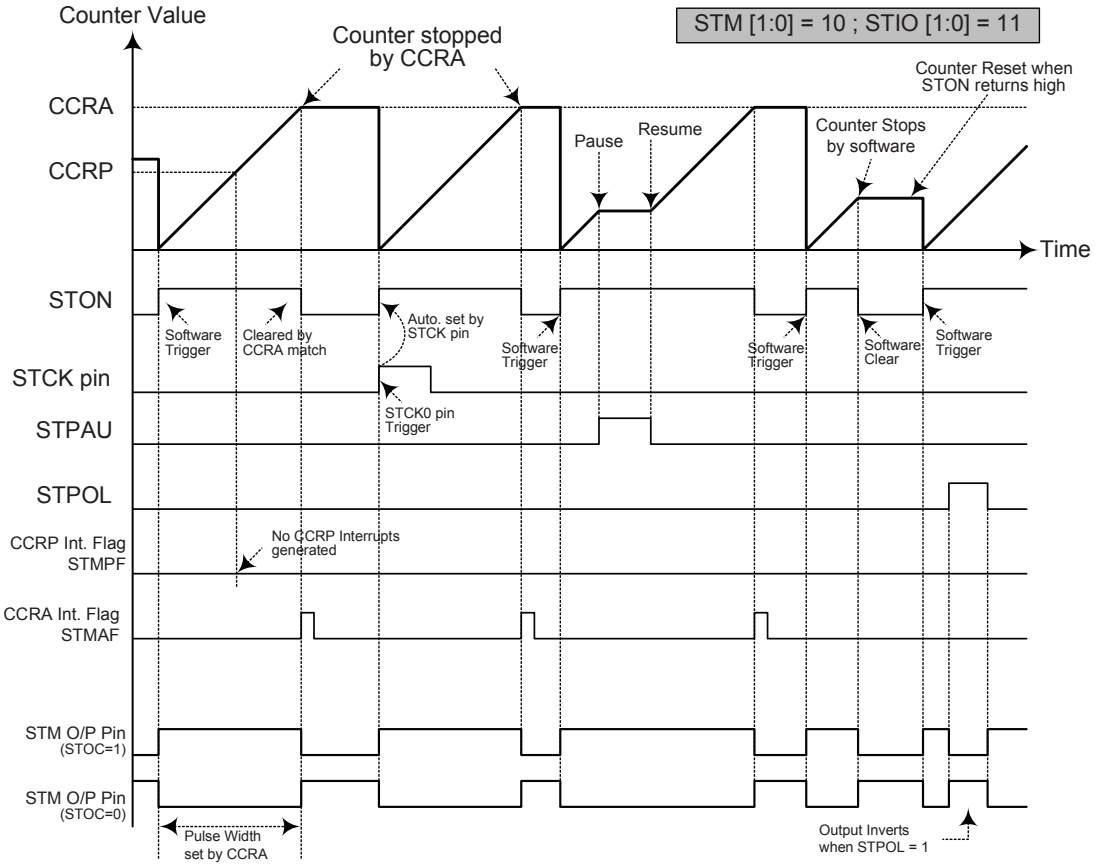
Single Pulse Mode

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

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



Single Pulse Generation



Single Pulse Mode

Note: 1. Counter stopped by CCRA match

2. CCRP is not used

3. The pulse is triggered by setting the STON bit high

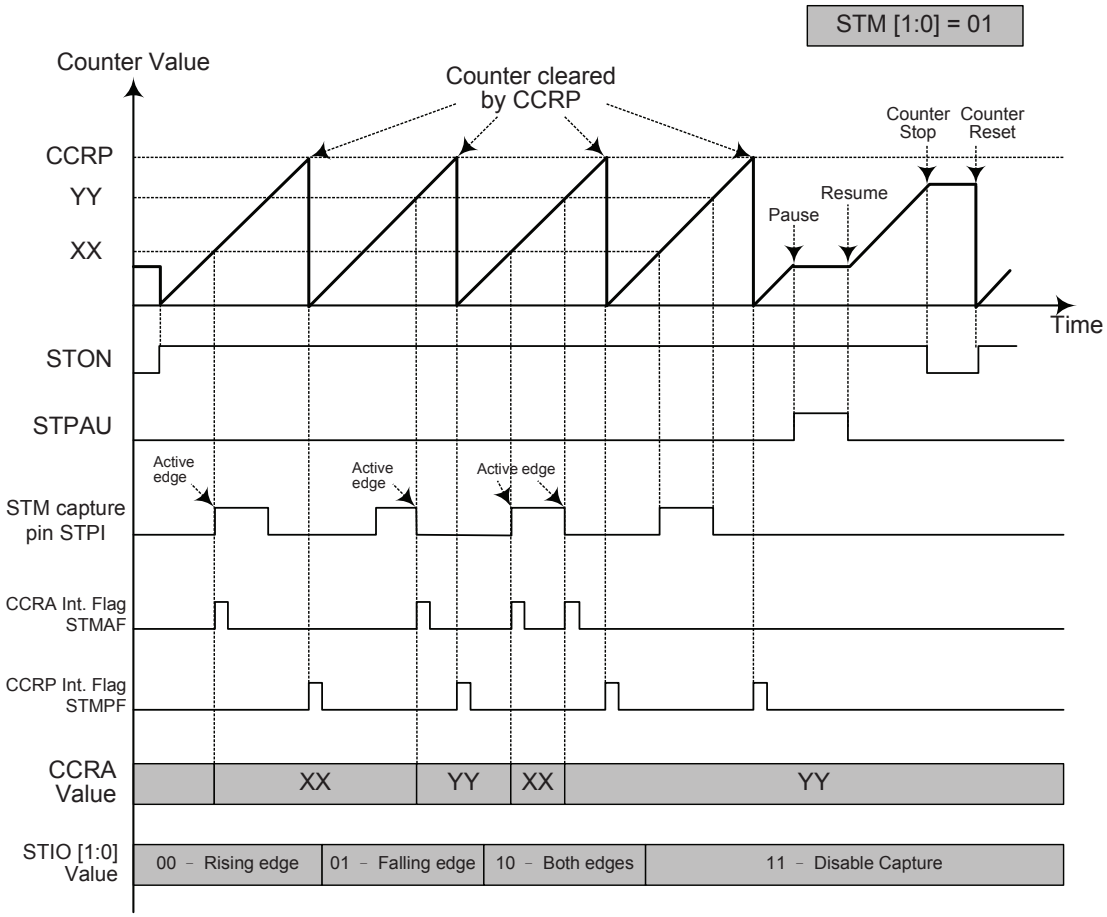
4. In the Single Pulse Mode, STIO [1:0] must be set to "11" and cannot be changed.

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

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 measurement. The external signal is supplied on the STPI, whose 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 STIO1 and STIO0 bits in the STMC1 register. The counter is started when the STON bit changes from low to high which is initiated using the application program.

When the required edge transition appears on the STPI the present value in the counter will be latched into the CCRA registers and a STM interrupt generated. Irrespective of what events occur on the STPI the counter will continue to free run until the STON bit changes from high to low. When a CCRP compare match occurs the counter will reset back to zero; in this way the CCRP value can be used to control the maximum counter value. When a CCRP compare match occurs from Comparator P, a STM interrupt will also be generated. Counting the number of overflow interrupt signals from the CCRP can be a useful method in measuring long pulse widths. The STIO1 and STIO0 bits can select the active trigger edge on the STPI to be a rising edge, falling edge or both edge types. If the STIO1 and STIO0 bits are both set high, then no capture operation will take place irrespective of what happens on the STPI, 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 TM Capture input pin active edge transfers the counter value to CCRA

3. The STCCLR and STDPX bits are 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.

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.

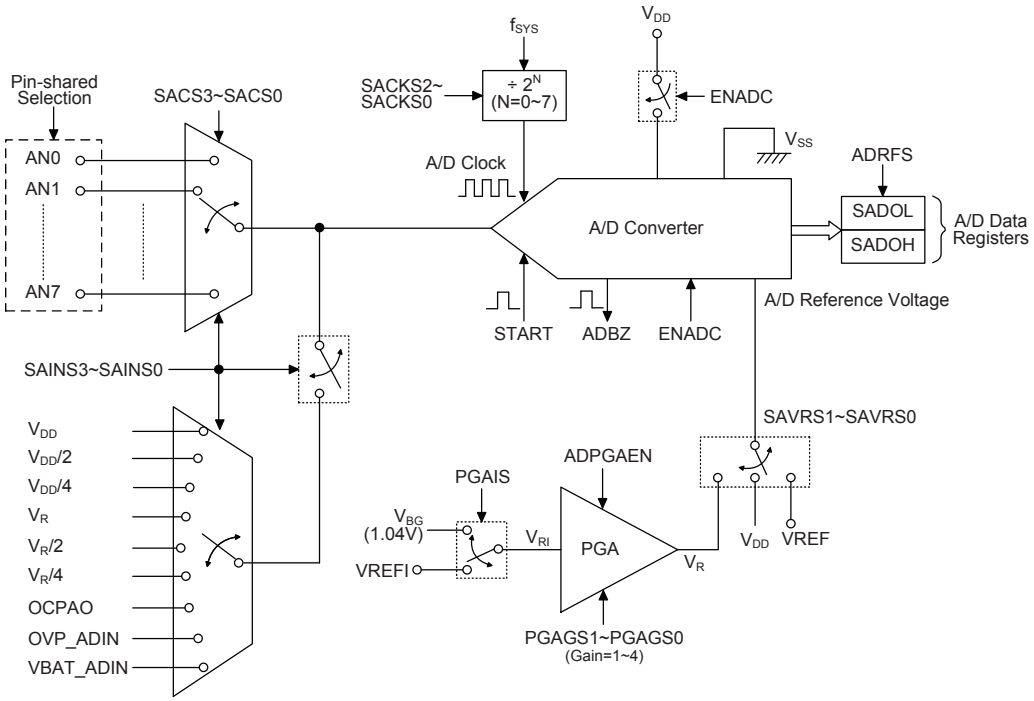
A/D Overview

Each 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. The external or internal analog signal to be converted is determined by the SAINS3~SAINS0 bits together with the SACS3~SACS0 bits. Note that when the internal analog signal is to be converted, the selected external input channel will be automatically disconnected to advoid malfunction.

More information about the A/D input signal is described in the "A/D Converter Control Registers" and "A/D Converter Input Pins" sections respectively.

Device	External Input Channels	Internal Signals	A/D Channel Select Bits
HT45F3420	AN0~AN3	V_{DD}, $V_{DD}/2$, $V_{DD}/4$ V_R, $V_R/2$, $V_R/4$ OCPAO, OVP_ADIN, VBAT_ADIN	SAINS[3:0] SACS[3:0]
HT45F3430	AN0~AN7		

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



Note: For HT45F3420, the external input channels are AN0~AN3
For HT45F3430, the external input channels are AN0~AN7

A/D Converter Structure

A/D Converter Registers

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

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

A/D Converter Register List

A/D Converter Data Registers – SADOL, SADOH

As the devices contain 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 the A/D converter data register contents will be unchanged if the A/D converter is disabled.

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

A/D Data Registers

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

To control the function and operation of the A/D converter, several control registers known as SADC0, SADC1 and 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. 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 if the analog signal to be converted comes from an internal analog signal or from an external analog channel input.

The pin-shared function control registers, named PAS0 and PAS1, contain the corresponding pin-shared selection bits which determine which pins on Port A 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 resistors connected to these pins will be automatically removed if the pin is selected to be an A/D input.

SADC0 Register

Bit	7	6	5	4	3	2	1	0
Name	START	ADBZ	ENADC	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**: ADC busy flag
0: A/D conversion ended or no conversion
1: A/D is busy
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 as 0 after the A/D conversion is complete.

Bit 5 **ENADC**: ADC enable/disable control
0: ADC disable
1: ADC 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 output data format selection bit
0: ADC output data format → SADOH=D[11:4]; SADOL=D[3:0]
1: ADC output data format → SADOH=D[11:8]; SADOL=D[7:0]
This bit controls the format of the 12-bit converted A/D value in the two A/D data registers. Details are provided in the A/D converter data register section.

Bit3~0 **SACS3~SACS0**: ADC input channels selection
HT45F3420:
0000: ADC input channel comes from AN0
0001: ADC input channel comes from AN1
0010: ADC input channel comes from AN2
0011: ADC input channel comes from AN3
Other values: ADC input is floating
HT45F3430:
0000: ADC input channel comes from AN0
0001: ADC input channel comes from AN1
0010: ADC input channel comes from AN2
0011: ADC input channel comes from AN3
0100: ADC input channel comes from AN4
0101: ADC input channel comes from AN5
0110: ADC input channel comes from AN6
0111: ADC input channel comes from AN7
Other values: ADC input is floating

SADC1 Register

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

Bit 7~4 **SAINS3~SAINS0**: Internal ADC input channel selection bits
0000: ADC input only comes from external pin
0001: ADC input comes from V_{DD}
0010: ADC input comes from $V_{DD}/2$
0011: ADC input comes from $V_{DD}/4$
0100: ADC input only comes from external pin
0101: ADC input comes from V_R
0110: ADC input comes from $V_R/2$
0111: ADC input comes from $V_R/4$
1000: ADC input comes from OCPAO
1001: ADC input comes from OVP_ADIN
1010: ADC input comes from VBAT_ADIN
1011: ADC input grounding (unused)
Other Values: ADC input only comes from external pin

Note: V_R is PGA output voltage. OCPAO is the OCP circuit OPA output. VBAT_ADIN and OVP_ADIN come from the integrated voltage divider circuits which can refer to the "Integrated Voltage Divider Circuits" section.

Bit 3 Unimplemented, read as "0"

Bit 2~0 **SACKS2~SACKS0**: ADC clock rate selection bit
000: f_{SYS}
001: $f_{SYS}/2$
010: $f_{SYS}/4$
011: $f_{SYS}/8$
100: $f_{SYS}/16$
101: $f_{SYS}/32$
110: $f_{SYS}/64$
111: $f_{SYS}/128$

SADC2 Register

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

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

Note: When the PGA output V_R is selected as ADC input or ADC reference voltage, the PGA need to be enabled by setting this bit high. Otherwise the PGA need to disable by clearing the ADPGAEN bit to conserve the power. As the PGA output also can be used for the OCP and OVP DAC reference input, it should be enabled first if it is selected by these functions.

Bit 6~5 Unimplemented, read as "0"

Bit 4 **PGAIS**: PGA input (V_{RI}) selection
0: V_{RI} only comes from V_{REFI} pin
1: V_{RI} only comes from V_{BG}

Bit 3~2 **SAVRS1~SAVRS0**: ADC reference voltage selection
00: ADC reference voltage comes from V_{DD}
01: ADC reference voltage comes from V_{REF} pin
1x: ADC reference voltage comes from V_R (PGA output)

Bit 1~0 **PGAGS1~PGAGS0**: PGA gain selection bits
00: Gain =1
01: Gain =2
10: Gain =3
11: Gain =4

A/D Converter Operation

The START bit is used to start and reset the A/D converter. When the microcontroller sets this bit from low to high and then low again, an analog to digital conversion cycle will be initiated. When the START bit is brought from low to high but not low again, the ADBZ bit in the SADC0 register will be cleared to zero and the analog to digital converter will be reset. It is the START bit that is used to control the overall start operation of the internal analog to digital converter.

The ADBZ bit in the SADC0 register is used to indicate whether the analog to digital conversion process is in process or not. When the A/D converter is reset by setting the START bit from low to high, the ADBZ flag will be cleared to 0. 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 be used to 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.

Although the A/D clock source is determined by the system clock f_{SYS}, and by bits SACKS2~SACKS0, there are some limitations on the maximum A/D clock source speed that can be selected. As the recommended value of permissible A/D clock period, t_{ADCK}, is from 0.5μs to 10μs, care must be taken for system clock frequencies. For example, if the system clock operates at a frequency of 4MHz, the SACKS2~SACKS0 bits should not be set to 000B or 11xB. Doing so will give A/D clock periods that are less than the minimum A/D clock period or greater than the maximum A/D clock period which may result in inaccurate A/D conversion values.

f_{SYS}	A/D Clock Period (t_{ADCK})							
	SACKS2, SACKS1, SACKS0 =000 (f_{SYS})	SACKS2, SACKS1, SACKS0 =001 (f_{SYS}/2)	SACKS2, SACKS1, SACKS0 =010 (f_{SYS}/4)	SACKS2, SACKS1, SACKS0 =011 (f_{SYS}/8)	SACKS2, SACKS1, SACKS0 =100 (f_{SYS}/16)	SACKS2, SACKS1, SACKS0 =101 (f_{SYS}/32)	SACKS2, SACKS1, SACKS0 =110 (f_{SYS}/64)	SACKS2, SACKS1, SACKS0 =111 (f_{SYS}/128)
1MHz	1μs	2μs	4μs	8μs	16μs*	32μs*	64μs*	128μs*
2MHz	500ns	1μs	2μs	4μs	8μs	16μs*	32μs*	64μs*
4MHz	250ns*	500ns	1μs	2μs	4μs	8μs	16μs*	32μs*
8MHz	125ns*	250ns*	500ns	1μs	2μs	4μs	8μs	16μs*

A/D Clock Period Examples

Controlling the power on/off function of the A/D converter circuitry is implemented using the ENADC bit in the SADC0 register. This bit must be set high to power on the A/D converter. When the ENADC 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 by configuring the corresponding pin-shared control bits, if the ENADC bit is high then some power will still be consumed. In power conscious applications it is therefore recommended that the ENADC is set low to reduce power consumption when the A/D converter function is not being used.

The reference voltage supply to the A/D Converter can be supplied from either the internal ADC power or from an external reference sources supplied on pin VREF or V_{BG} voltage. The desired selection is made using the SAVRS1~ SAVRS0 bits. 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. When V_R is selected by ADC reference voltage, the PGA needs to be enabled by setting the ADPGAEN bit to 1.

A/D Converter Input Signals

All of the A/D analog input pins are pin-shared with the I/O pins on Port A as well as other functions. The corredponding selection bits for each I/O pin in the PAS0 and PAS1 registers, determine whether the input pins are setup as A/D converter analog inputs or whether they have other functions. If the pin-shared function control bits configure its corresponding pin as an A/D analog channel input, the pin will be setup to be an A/D converter external channel input and the original pin functions 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 PAC port control register to enable the A/D input as when the pin-shared function control bits enable an A/D input, the status of the port control register will be overridden.

There are several internal analog signals, the Bandgap reference voltage V_{BG}, V_{DD}, PGA output V_R, OCP output OCPAO or the integrated voltage diving circuit generating VBAT_ADIN and OVP_ADIN signals, which can be connected to the A/D converter as the analog input signal by configuring the SAINS3~SAINS0 bits. If the SAINS3~SAINS0 bits are set to "0000", 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 is selected to be converted. If the SAINS3~SAINS0 bits are set to "0101~0111", the PGA output voltage is selected to be converted.

The A/D converter has its own reference voltage pin, VREF, however the reference voltage can also be supplied from the power supply pin, a choice which is made through the SAVRS[1:0] in the SADC2 register. The analog input values must not be allowed to exceed the value of the selected ADC reference voltage. The A/D converter also has a VREFI pin which is one of PGA inputs for ADC reference. To select this PGA input signal, the PGAIS bit must be cleared to zero and the revelent pin-shared control bits should be properly configured.

Note that when the programs select external signal (AN0~AN3 for HT45F3420, AN0~AN7 for HT45F3430) and internal signal (V_{DD}, $V_{DD}/2$, $V_{DD}/4$, V_R, $V_R/2$, $V_R/4$, OVPAO, VBAT_ADIN or OVP_ADIN) as an A/D Converter input signal simultaneously, then the hardware will only choose the internal signal as an ADC input. In addition, if the program selects an external reference voltage V_{REF} and the internal reference voltage V_{DD} or V_R as the ADC reference voltage, then the hardware will only choose the internal reference voltage as the ADC reference voltage input. The same hardware control method can be applied to the PGA input selection. If the application program selects the external voltage, V_{REFI}, and an internal voltage V_{BG} as PGA input simultaneously, then the hardware will only choose the internal voltage V_{BG} as the PGA input.

SAINS[3:0]	SACS[3:0]	Input Signals	Description
0000, 0100, 1100~1111	0000~0011	AN0~AN3	External channel analog input
	0000~0111	AN0~AN7	
0001	xxxx	V_{DD}	A/D converter power supply voltage
0010	xxxx	$V_{DD}/2$	A/D converter power supply voltage/2
0011	xxxx	$V_{DD}/4$	A/D converter power supply voltage/4
0101	xxxx	V_R	Internal reference voltage
0110	xxxx	$V_R/2$	Internal reference voltage/2
0111	xxxx	$V_R/4$	Internal reference voltage/4
1000	xxxx	OCPAO	OCP circuit OPA output
1001	xxxx	OVP_ADIN	From the integrated voltage diving circuit
1010	xxxx	VBAT_ADIN	From the integrated voltage diving circuit
1011	xxxx	Ground	Unused

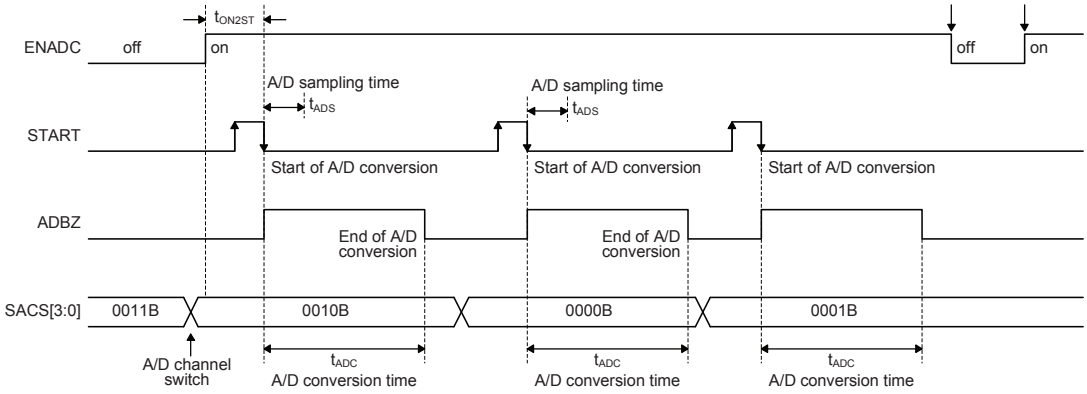
A/D Converter Input Signals Selection

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 A/D conversion which is defined as t_{ADC} are necessary.

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

However, there is a usage limitation on the next A/D conversion after the current conversion is complete. When the current A/D conversion is complete, the converted digital data will be stored in the A/D data register pair and then latched after half an A/D clock cycle. If the START bit is set to 1 in half an A/D clock cycle after the end of A/D conversion, the converted digital data stored in the A/D data register pair will be changed. Therefore, it is recommended to initiate the next A/D conversion after a certain period greater than half an A/D clock cycle at the end of current A/D conversion.



A/D Conversion Timing

Summary of A/D Conversion Steps

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

Step 1

Select the required A/D conversion frequency by SACKS2~ SACKS0

Step 2

Enable the ADC by set ENADC=1

Step 3

Select which pins will be configure as ADC analog inputs

Step 4

If input comes from I/O, set SAINS[3:0]=0000 and then set SACS[3:0] to corresponding PAD input

If input comes from internal input, set SAINS[3:0] to corresponding internal input source

Step 5

Select reference voltage comes from external V_{REF} or from V_{DD} or V_{BG} by SAVRS[1:0]

Step 6

Select ADC output data format by ADRFS

Step 7

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

Step 8

The A/D convert procedure can now be initialized by set START from low to high and then low again

Step 9

If ADC is under conversion, ADBZ=1. After A/D conversion process is completed, the ADBZ flag will go low, and then output data can be read from SADOH and SADOL registers. If the A/D interrupt is enabled and the stack is not full, data can be acquired by interrupt service program. Another way to get the A/D output data is polling the ADBZ flag.

Programming Considerations

During microcontroller operations where the A/D converter is not being used, the A/D internal circuitry can be switched off to reduce power consumption, by clearing the ENADC bit 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 Transfer Function

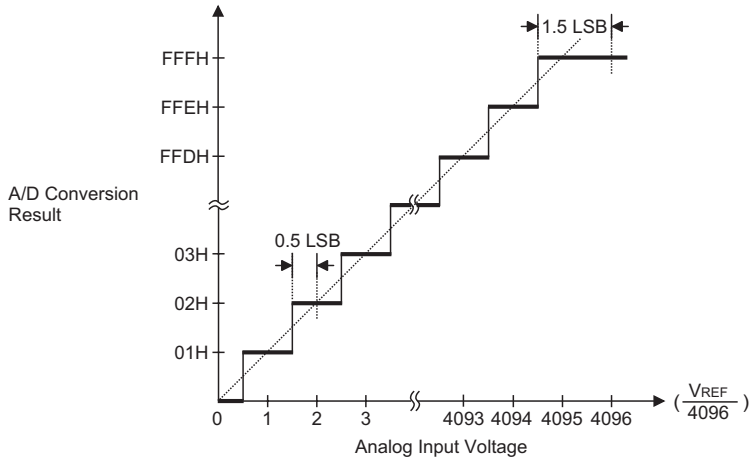
As the devices contain a 12-bit A/D converter, its full-scale converted digitised value is equal to FFFH. Since the full-scale analog input value is equal to the reference voltage, V_{REF}, which can be V_{DD}, VREF pin voltage or V_R, this gives a single bit analog input value of reference voltage value divided by 4096.

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

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

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

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



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 EOCB polling method to detect the end of conversion

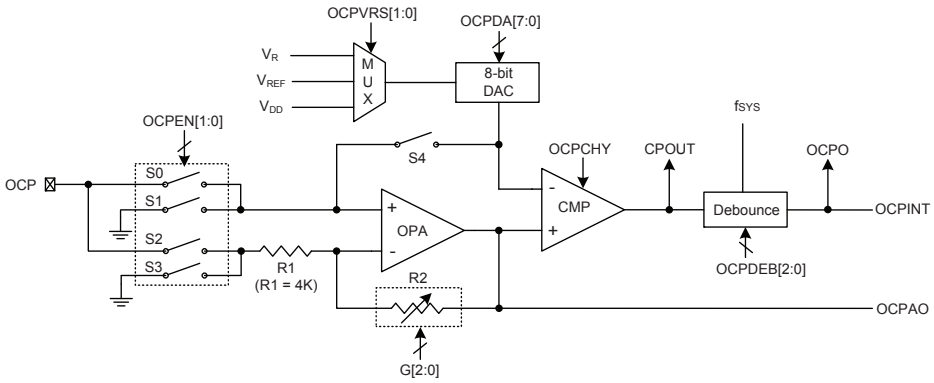
```
clr  ADE              ; disable ADC interrupt
mov  a,03H
mov  SADC1,a          ; select fSYS/8 as A/D clock
set  ENADC
mov  a,03h            ; setup PAS0 to configure pin AN0
mov  PAS0,a
mov  a,20h
mov  SADC0,a          ; enable ADC and connect AN0 channel to A/D converter
:
start_conversion:
clr  START            ; high pulse on start bit to initiate conversion
set  START            ; reset A/D
clr  START            ; start A/D
polling_EOC:
sz   ADBZ             ; poll the SADC0 register ADBZ bit to detect end of A/D conversion
jmp polling_EOC       ; continue polling
mov  a,SADOL          ; read low byte conversion result value
mov  SADOL_buffer,a   ; save result to user defined register
mov  a,SADOH          ; read high byte conversion result value
mov  SADOH_buffer,a   ; save result to user defined register
:
:
jmp  start_conversion ; start next a/d conversion
```

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

```
clr  ADE              ; disable ADC interrupt
mov  a,03H
mov  SADC1,a          ; select fSYS/8 as A/D clock
set  ENADC
mov  a,03h            ; setup PAS0 to configure pin AN0
mov  PAS0,a
mov  a,20h
mov  SADC0,a          ; enable ADC 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
```

Over Current Protection

The devices include an 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 OCP pin is converted to a relevant voltage level according to the current value using the OCP 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 OCP interrupt will be generated if the corresponding interrupt control is enabled.



Note: 1. V_R is the ADC PGA output signal

2. For HT45F3430 device, OCP input source pin can be PA1 or PA6 determined by IFS03 bit in the IFS0 register.

Over Current Protection Circuit

Over Current Protection Operation

The illustrated OCP circuit is used to prevent the input current from exceeding a reference level. The current on the OCP pin is converted to a voltage and then amplified by the OCP operational amplifier with a programmable gain from 1 to 50 selected by the G2~G0 bits in the OCPC1 register. This is known as a Programmable Gain Amplifier or PGA. This PGA can also be configured to operate in the non-inverting, inverting or input offset calibration mode determined by the OCPEN1 and OCPEN0 bits in the OCPC0 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_{REF} or V_R, selected by the OCPVRS[1:0] bits in the OCPC0 register. The comparator output, CPOUT, will first be filtered with a certain de-bounce time period selected by the OCPDEB2~OCPDEB0 bits in the OCPC1 register. Then a filtered OCP digital comparator output, OCPO, is obtained to indicate whether an over current condition occurs or not. The OCPO bit will be set to 1 if an over current condition occurs. Otherwise, the OCPO bit is zero. Once an over current event occurs, i.e., the converted voltage of the OCP input current is greater than the reference voltage, the corresponding interrupt will be generated if the relevant interrupt control bit is enabled.

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 remaining two registers are control registers which control the OCP function, D/A converter reference voltage select, PGA gain select, comparator de-bounce time together with the hysteresis function.

Register Name	Bit							
	7	6	5	4	3	2	1	0
OCPC0	OCPEN1	OCPEN0	OCPVRS1	OCPVRS0	OCPCHY	—	—	OCPO
OCPC1	—	—	G2	G1	G0	OCPDEB2	OCPDEB1	OCPDEB0
OCPDA	D7	D6	D5	D4	D3	D2	D1	D0
OCPOCAL	OOFM	ORSP	OOF5	OOF4	OOF3	OOF2	OOF1	OOF0
OCPCCAL	CPOUT	COFM	CRSP	COF4	COF3	COF2	COF1	COF0

OCP Control Register List

OCPC0 Register

Bit	7	6	5	4	3	2	1	0
Name	OCPEN1	OCPEN0	OCPVRS1	OCPVRS0	OCPCHY	—	—	OCPO
R/W	R/W	R/W	R/W	R/W	R/W	—	—	R
POR	0	0	0	0	0	—	—	0

Bit 7~6 **OCPEN1 ~ OCPEN0**: OCP function operating mode selection
00: OCP function is disabled, S1and S3 on, S0 and S2 off
01: Non-inverting mode, S0 and S3 on, S1 and S2 off
10: Inverting mode, S1 and S2 on, S0 and S3 off
11: Calibration mode, S1 and S3 on, S0 and S2 off

Bit 5~4 **OCPVRS1~OCPVRS0**: OCP DAC reference voltage selection
00: From V_{DD}
01: From V_{REF}
1x: From V_R (ADC PGA output)

Bit 3 **OCPCHY**: OCP Comparator hysteresis function control
0: Disable
1: Enable

Bit 2~1 Unimplemented, read as "0"

Bit 0 **OCPO**: OCP digital output bit
0: The monitored source current is not over
1: The monitored source current is over

OCPC1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	G2	G1	G0	OCPDEB2	OCPDEB1	OCPDEB0
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 **G2~G0**: PGA R2/R1 ratio selection
000: Unity gain buffer (non-inverting mode) or R2/R1=1(inverting 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 inverting and non-inverting mode. The calculating formula of the PGA gain for the inverting and non-inverting mode is described in the "Input Voltage Range" section.

Bit 2~0 **OCPDEB2~OCPDEB0**: OCP 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: $t_{DEB}=1/f_{SYS}$

OCPDA 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 OCP DAC output voltage control bits
OCP DAC Output=(DAC reference voltage/256) × D[7:0]

OCPOCAL Register

Bit	7	6	5	4	3	2	1	0
Name	OOFM	ORSP	OOF5	OOF4	OOF3	OOF2	OOF1	OOF0
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 **OOFM**: OCP Operational Amplifier Input Offset Calibration Mode Enable control
0: Input Offset Calibration Mode Disable
1: Input Offset Calibration Mode Enable
This bit is used to control the OCP operational amplifier input offset Calibration function. The OCPEN1 and OCPEN0 bits must first be set to "11" and then the OOFM bit must be set to 1 followed by the COFM bit being setting 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 **ORSP**: OCP 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 **OOF5~OOF0**: OCP 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 OCP 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.

OCPCCAL Register

Bit	7	6	5	4	3	2	1	0
Name	CPOUT	COFM	CRSP	COF4	COF3	COF2	COF1	COF0
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 **CPOUT**: OCP Comparator Output, 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 OCP operates in the input offset Calibration mode. If the CPOUT 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 **COFM**: OCP Comparator Input Offset Calibration Mode Enable control
0: Input Offset Calibration Mode Disabled
1: Input Offset Calibration Mode Enabled
This bit is used to control the OCP comparator input offset Calibration function. The OCPEN1 and OCPEN0 bits must first be set to "11" and then the COFM bit must be set to 1 followed by the OOFM bit being setting 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 **CRSP**: OCP Comparator Input Offset Calibration Reference Input select
0: Select negative input as the reference input
1: Select positive input as the reference input

Bit 4~0 **COF4~COF0**: OCP Comparator Input Offset Calibration value
This 5-bit field is used to perform the comparator input offset calibration operation and the value for the OCP comparator input offset calibration can be restored into this bit field. More detailed information is described in the "Comparator Input Offset Calibration" section.

Input Voltage Range

Together with different PGA operating modes, the input voltage on the OCP 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-inverting mode and the PGA output is obtained using the formula below:

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

- When the PGA operates in the non-inverting mode by setting the OCPEN[1:0] to "01" with unity gain select by setting the G[2:0] to "000", the PGA will act as an unit-gain buffer whose output is equal to V_{IN} .

$$\mathbf{V_{OUT} = V_{IN}}$$

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

$$\mathbf{V_{OUT} = -\frac{R_2}{R_1} \times V_{IN}}$$

Offset Calibration

The OCP circuit has 4 operating mode controlled by OCPEN[1:0], one of it is calibration mode. In calibration mode, OPAMP and comparator offset can be calibrated.

OPAMP Calibration

Step1: Set OCPEN[1:0] = 11, OOFM = 1, and COFM = 0, ORSP=1, OCP is now under OPAMP calibration status.

Step2: Set OOF[5:0] = 000000 then read CPOUT bit.

Step3: Let OOF = OOF + 1 then read CPOUT bit, if CPOUT is changed; record the data as VOS1.

Step4: Set OOF[5:0] = 111111 then read CPOUT bit.

Step5: Let OOF = OOF - 1 then read CPOUT bit, if CPOUT is changed; record the data as VOS2.

Step6: restore VOS = (VOS1 + VOS2) / 2 to OOF[5:0], the calibration is finished.

Note: S4 is OFF. In this mode operation, OPAMP output to CPOUT bypass comparator.

Comparator Calibration

Step1: Set OCPEN[1:0] = 11, COFM = 1, and OOFM = 0, OCP is now under comparator calibration status, S4 on.

Step2: Set COF[4:0] = 00000 then read CPOUT bit.

Step3: Let COF = COF + 1 then read CPOUT bit, if CPOUT is changed; record the data as VOS1.

Step4: Set COF[4:0] = 11111 then read CPOUT bit.

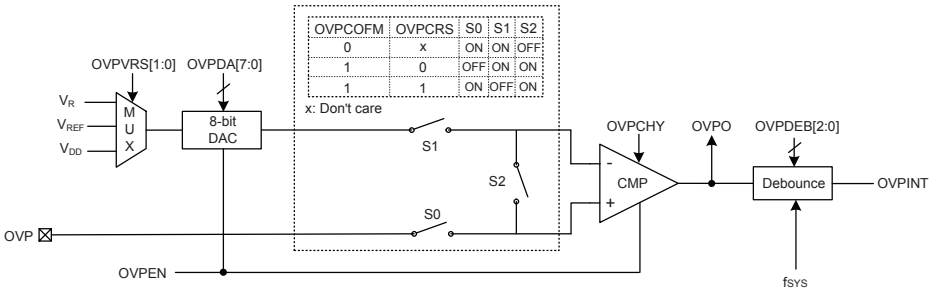
Step5: Let COF = COF - 1 then read CPOUT bit, if CPOUT data is changed; record the data as VOS2.

Step6: restore VOS = (VOS1 + VOS2) / 2 to COF[4:0], the calibration is finished.

Note: S4 is used ON and DAC is OFF, only for comparator calibration procedure. In the normal mode operation, it is off.

Over Voltage Protection

The devices include an over voltage protection function which provides a protection mechanism for applications. To prevent the operating voltage from exceeding a specific level, the voltage on the OVP pin is compared with a reference voltage generated by an 8-bit DAC. When an over voltage event occurs, an OVP interrupt will be generated if the corresponding interrupt control is enabled.



Note: 1. V_R is the ADC PGA output signal

2. The OVP pin signal can be OVP_IN1 or OVP_IN2 which is selected using the IFS02 bit in the IFS0 register.

Over Voltage Protection Circuit

Over Voltage Protection Operation

The source voltage is supplied on the OVP pin and then connected to one input of the comparator. A DAC is used to generate a reference voltage. The comparator compares the reference voltage with the input voltage to produce the OVPO signal.

Over Voltage Protection Control Registers

Overall operation of the over voltage protection is controlled using several registers. One register is used to provide the reference voltages for the over voltage protection circuit. The remaining two registers are control registers which are used to control the OVP function, DAC reference voltage selection, comparator de-bounce time, comparator hysteresis function together with the comparator input offset calibration.

Register Name	Bit							
	7	6	5	4	3	2	1	0
OVPC0	—	—	OVPEN	OVPCHY	OVPVRS1	OVPVRS0	OVPDEB1	OVPDEB0
OVPC1	OVPO	OVPCOFM	OVPCRS	OVPCOF4	OVPCOF3	OVPCOF2	OVPCOF1	OVPCOF0
OVPDA	D7	D6	D5	D4	D3	D2	D1	D0

OVP Control Register List

OVPC0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	OVPEN	OVPCHY	OVPVRS1	OVPVRS0	OVPDEB1	OVPDEB0
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 **OVPEN**: OVP function control bit
0: Disable
1: Enable
If the OVPEN bit is cleared to 0, the over voltage protection function is disabled and no power will be consumed. This results in the comparator and D/A converter of OVP all being switched off.

Bit 4 **OVPCHY**: OVP comparator hysteresis function control bit
0: Disable
1: Enable

Bit 3~2 **OVPVRS1~OVPVRS0**: OVP DAC reference voltage selection bit
00: DAC reference voltage comes from V_{DD}
01: DAC reference voltage comes from V_{REF}
1x: DAC reference voltage comes from V_R (ADC PGA output)

Bit 1~0 **OVPDEB1~OVPDEB0**: OVP comparator debounce time control bits
00: No debounce
01: (7~8) × 1/f_{SYS}
10: (15~16) × 1/ f_{SYS}
11: (31~32) × 1/ f_{SYS}

OVPC1 Register

Bit	7	6	5	4	3	2	1	0
Name	OVPO	OVPCOFM	OVPCRS	OVPCOF4	OVPCOF3	OVPCOF2	OVPCOF1	OVPCOF0
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 **OVPO**: OVP comparator ouptut bit
0: Positive input voltage < negative input voltage
1: Positive input voltage > negative input voltage

Bit 6 **OVPCOFM**: OVP comparator normal operation or input offset voltage Calibration mode selection
0: Normal operation
1: Input offset voltage calibration mode

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

Bit 4~0 **OVPCOF4 ~ OVPCOF0**: OVP comparator input offset voltage calibration control bits

OVPDA 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 OVP DAC output voltage control bits
DAC V_{OUT}=(DAC reference voltage /256) × D[7:0]

Offset Calibration

The OVPCOFM bit in the OVPC1 register is used to select the OVP comparator operating mode, normal operation or offset calibration mode. If set the bit high, the comparator will enter the offset voltage calibration mode. It is need to note that before offset calibration, the hysteresis voltage should be zero by set OVPCHY=0 and because the OVP pin are pin-shared with I/O, it should be configured as comparator input first.

Comparator Calibration Procedure:

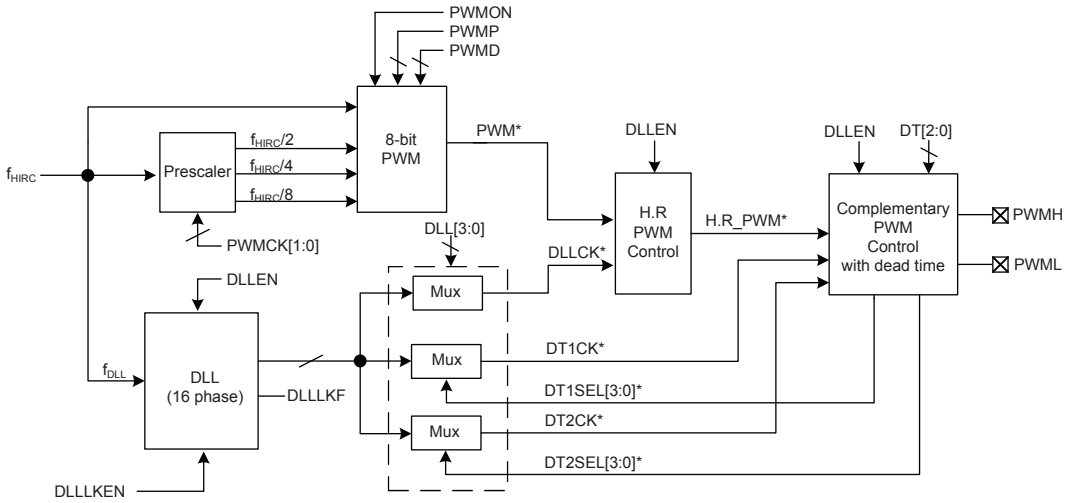
- Step1: Set OVPCOFM=1, OVPCRS=1, the OVP is now in the comparator calibration mode, S0 and S2 on. To make sure V_{OS} as minimize as possible after calibration, the input reference voltage in calibration should be the same as input DC operating voltage in normal mode operation.
- Step2: Set OVPCOF [4:0] =00000 then read OVPO bit
- Step3: Let OVPCOF[4:0]=OVPCOF[4:0]+1 then read the OVPO bit status; if OVPO is changed, record the OVPCOF[4:0] data as V_{OS1}
- Step4: Set OVPCOF [4:0] =11111 then read the OVPO bit status
- Step5: Let OVPCOF[4:0]=OVPCOF[4:0]-1 then read the OVPO bit status; if OVPO data is changed, record the OVPCOF[4:0] data as V_{OS2}.
- Step6: Restore $V_{OS}=(V_{OS1} + V_{OS2})/2$ to the OVPCOF[4:0] bits. The calibration is finished. If $(V_{OS1} + V_{OS2})/2$ is not integral, discard the decimal. Residue $V_{OS}=V_{OUT} – V_{IN}$

High Resolution Complementary Output PWM Generator

The devices contain a multi-feature fully integrated PWM Generator which has complimentary outputs for maximum applicatgion flexibility.

Functional Description

The High Resolution 8-bit PWM generator circuits include a delay lock loop circuit and PWM complementary outputs with dead time insertion.



Note: 1. "*" is the internal signal name and not the Special Function Register bit.
DT1SEL[3:0] and DT2SEL[3:0] are calculated and selected automatically based on the DT[2:0].
DT1CK is the PWMH DT reference signal
DT2CK is the PWML DT reference signal
DLLCK is the H.R_PWM reference signal
2. H.R=High Resolution

H.R PWM Output Block Diagram

High Resolution PWM Registers

Overall operation of the High Resolution PWM is controlled using five registers. A PWM period register, PWMP, exists to store the desired 8-bit PWM period value. The PWM duty value is stored in an 8-bit PWMD register. The PWM function control, PWM counter clock selection, DLL circuit and dead time duration is determined by the CPR register. The register DLL is used for the DLL circuit phase selection. The remaining register CPOR is used for PWM setup, complementary output control, protection and inverting control.

Register Name	Bit							
	7	6	5	4	3	2	1	0
PWMP	D7	D6	D5	D4	D3	D2	D1	D0
PWMD	D7	D6	D5	D4	D3	D2	D1	D0
DLL	DLL3	DLL2	DLL1	DLL0	—	—	—	DLLLKF
CPR	DLLLKEN	DLLEN	PWMCK1	PWMCK0	PWMON	DT2	DT1	DT0
CPOR	PWMACT	CNVTYP	INVH	INVL	OCPHEN	OCPLEN	OVPHEN	OVPLEN

High Resolution PWM Generator Register List

PWMP 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 PWM period register
PWM period=PWMP[7:0] +1

PWMD 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 PWM duty register
These two registers, PWMP and PWMD, are used for 8-bit PWM Period and Duty control. The following should be noted during setup:
1. The PWMD value should meet the condition: $1 \leq$ PWMD $\leq$ (PWMP - 1)
2. PWM Duty (Min.)=1+ DLL[3:0] - DT[2:0] when DLL[3:0]=0000B, DT[2:0]=111B
3. PWM Duty (Max.)=PWMP-1 + DLL[3:0] - DT[2:0] when DLL [3:0]=1111B, DT[2:0]= 000B

DLL Register

Bit	7	6	5	4	3	2	1	0
Name	DLL3	DLL2	DLL1	DLL0	—	—	—	DLLLKF
R/W	R/W	R/W	R/W	R/W	—	—	—	R/W
POR	0	0	0	0	—	—	—	0

Bit 7~4 **DLL3~DLL0**: DLL phase selection
0000: H.R_PWM Duty Falling edge is fine-adjusted to be at the DLL phase#0 Rising edge
0001: H.R_PWM Duty Falling edge is fine-adjusted to be at the DLL phase#1 Rising edge
0010: H.R_PWM Duty Falling edge is fine-adjusted to be at the DLL phase#2 Rising edge
…
1110: H.R_PWM Duty Falling edge is fine-adjusted to be at the DLL phase#14 Rising edge
1111: H.R_PWM Duty Falling edge is fine-adjusted to be at the DLL phase#15 Rising edge

Bit 3~1 Unimplemented, read as "0"

Bit 0 **DLLLKF**: DLL circuit losing lock flag
0: No losing lock occurs
1: Losing lock occurs
This bit can be cleared to zero by software, but can not be set high.
Note: When DLLLKEN=1 and DLL function is enabled, if no losing lock conditions occur, 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.

CPR Register

Bit	7	6	5	4	3	2	1	0
Name	DLLLKEN	DLLEN	PWMCK1	PWMCK0	PWMON	DT2	DT1	DT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	0	0	0	0	0	0	0

Bit 7 **DLLLKEN**: DLL circuit Locking 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 sovled 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 conditions.

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 DT[2:0]
If this bit is cleared then the PWMCK[1:0] bits can be set to 00~11 by software and the H.R_PWM=PWM, no dead time is inserted. If set high, the hardware will set PWMCK[1:0] be 00 which cannot be changed, the PWM will be finely adjusted by the DLL and then output the High Resolution PWM output with dead time inserted.

Bit 5~4 **PWMCK1~PWMCK0**: PWM counter clock source selection
00: f_{HIRC}
01: $f_{HIRC}/2$
10: $f_{HIRC}/4$
11: $f_{HIRC}/8$

Bit 3 **PWMON**: PWM function control bit
0: Disable, PWM counter=0
1: Enable

Bit 2~0 **DT2~DT0**: Dead time selection
000: Dead time=$t_{DLL}\times0$~ $t_{DLL}\times1$
001: Dead time=$t_{DLL}\times2$~ $t_{DLL}\times3$
010: Dead time=$t_{DLL}\times4$~ $t_{DLL}\times5$
011: Dead time=$t_{DLL}\times6$~ $t_{DLL}\times7$
100: Dead time=$t_{DLL}\times8$~ $t_{DLL}\times9$
101: Dead time=$t_{DLL}\times10$~ $t_{DLL}\times11$
110: Dead time=$t_{DLL}\times12$~ $t_{DLL}\times13$
111: Dead time=$t_{DLL}\times14$~ $t_{DLL}\times15$
Note: $t_{DLL}=1/(f_{HIRC}\times16)$

CPOR Register

Bit	7	6	5	4	3	2	1	0
Name	PWMACT	CNVTYP	INVH	INVL	OCPHEN	OCPLEN	OVPHEN	OVPLEN
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 **PWMACT**: PWMH/PWML output active control bit
0: Force inactive
1: Active

Bit 6 **CNVTYP**: Converter type selection
0: Buck (duty @ PWMH)
1: Boost (duty @ PWML)
The CNVTYP bit is used to select either Buck or Boost converter type and to determine if the PWM output is PWMH or PWML. When using the Buck type, the duty control outputs to PWMH, when using the Boost type, the duty control outputs to PWML.

Bit 5 **INVH**: Inverting control before sending to PWMH_HV
0: Non-inverted
1: Inverted

Bit 4 **INVL**: Inverting control before sending to PWML_HV
0: Non-inverted
1: Inverted

Bit 3 **OCPHEN**: Over current protection enable control for PWMH
0: Disable
1: Enable

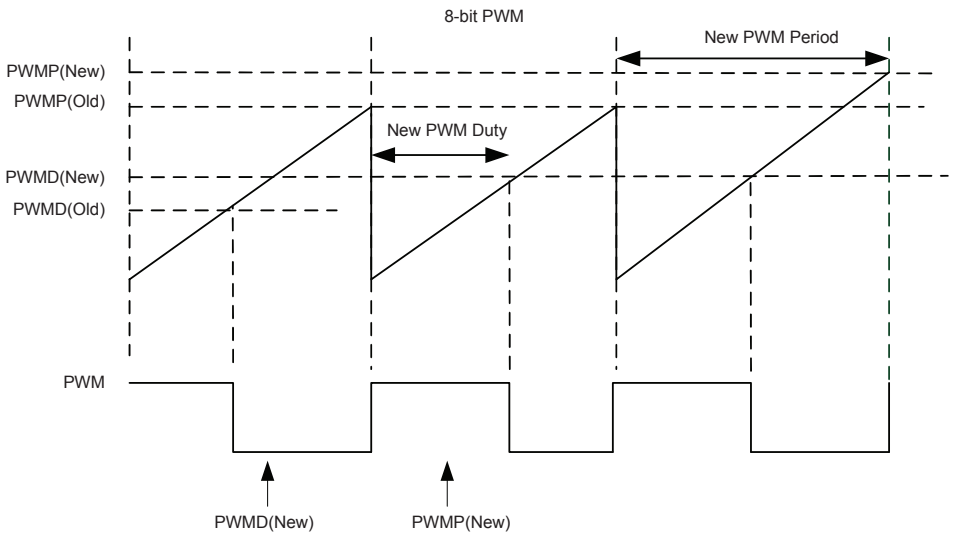
Bit 2 **OCPLEN**: Over current protection enable control for PWML
0: Disable
1: Enable

Bit 1 **OVPHEN**: Over voltage protection enable control for PWMH
0: Disable
1: Enable

Bit 0 **OVPLEN**: Over voltage protection enable control for PWML
0: Disable
1: Enable

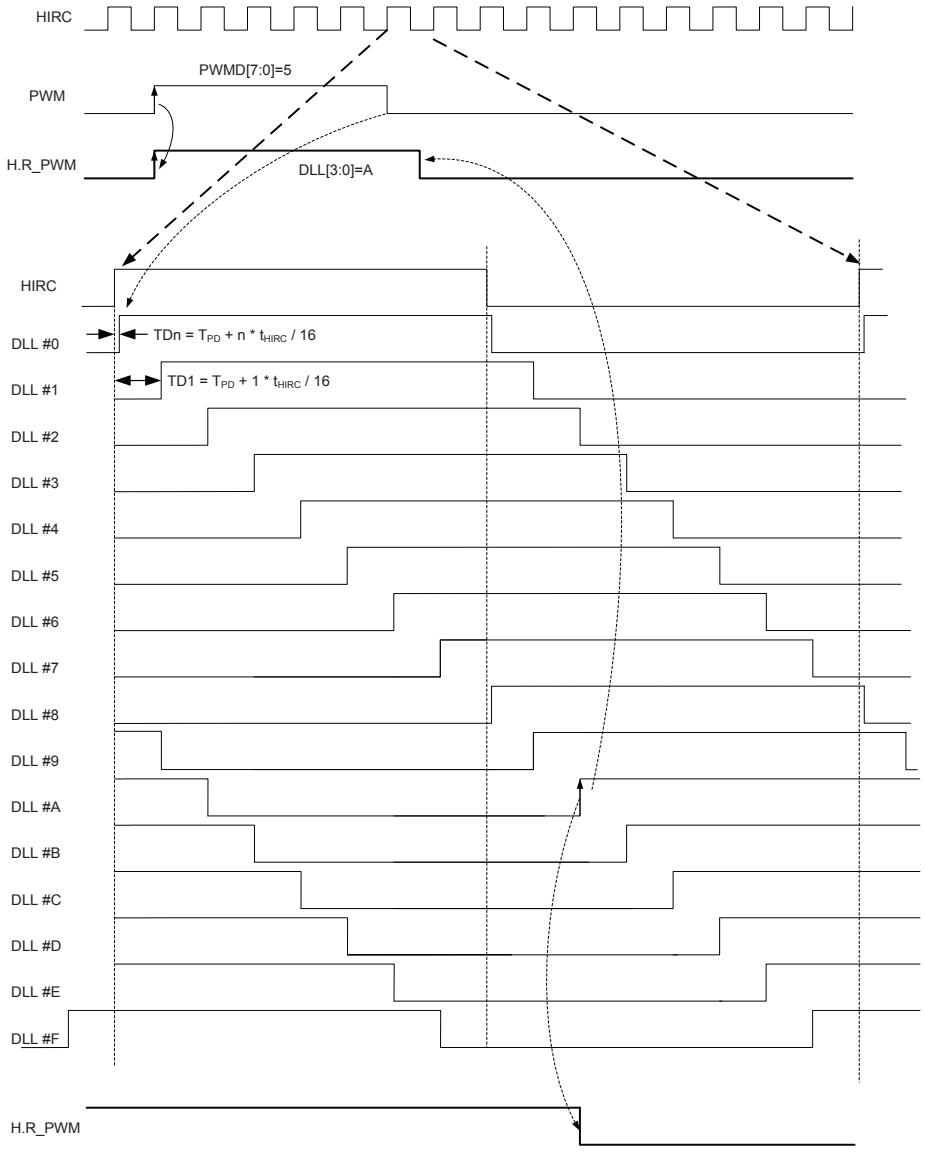
PWM Generator

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



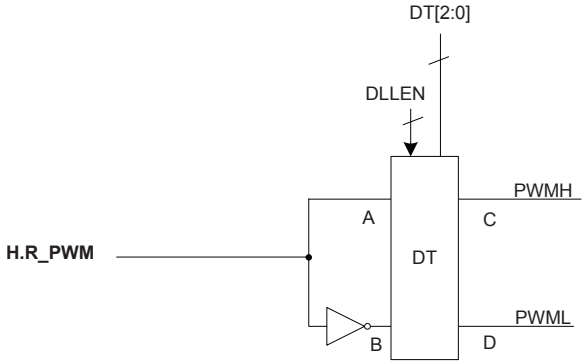
Delay Lock Loop

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 PWM signal output. If the PWM clock=f_{HIRC} , this means that the PWM output duty resolution is 1/ f_{HIRC}. The PWM signal passes through the DLL phase selection and H. R_PWM control which is set by the DLL[3:0] bits in the DLL register circuit to output a fine-tuned PWM signal, H.R_PWM with the PWM duty resolution increased by 4 bits.

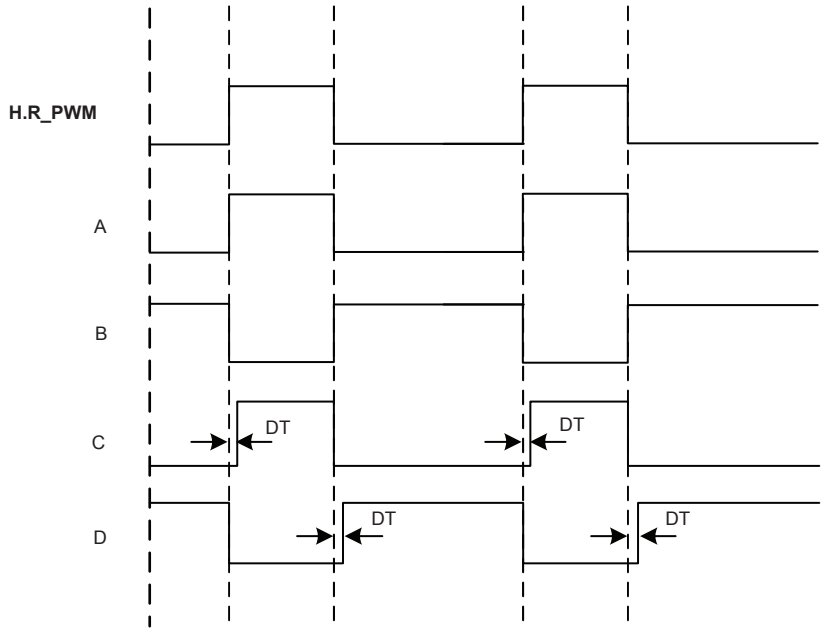


Dead-Time Insert

The devices provide a complementary output pair of signals which can be used as a PWM driver signal. The signal is sourced from the High Resolution PWM output signal, 8-bit PWM with DLL circuit. PWM output is an active high signal. By using the DLLEN bit, the dead time generator will be enabled and a dead time, which is programmable using the DT[2:0] bits in the CPR register, will be inserted to prevent excessive DC current. The dead time will be inserted whenever the rising edge of the dead time generator input signal occurs.



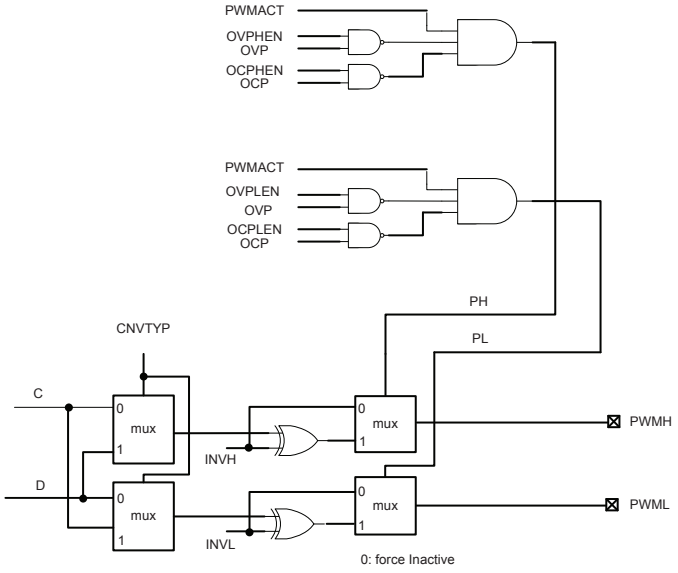
Complementary PWM output
(with dead time control)



Note: C and D are the complementary PWM 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 also may be in an inactive state resulting from some unpredictable reasons, such as malfunctions or electrical noise. The devices provide a protection function to force the two signals to output inverting signals when the PWMH or PWML signal is in an inactive state. The inverting control circuitry determines whether the signals are inverted or not using corresponding inverting control bits by setting up the INVH or INVL bit in the CPOR register. The devices also include over current protection and over voltage protection functions which are described in the OCP and OVP sections. By using the OCPHEN, OCPLEN, OVPHEN and OVPLEN bits, the protection circuit for the output signals will be enabled.



Programming Considerations

The following steps show the read and write procedures:

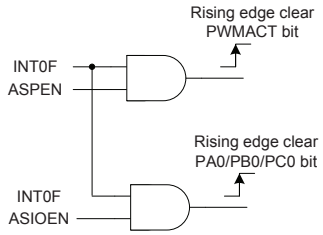
- Writing Data to DLL/PWMD
  - Step 1. Write data to DLL
    - Note that here data is only written to the 4-bit buffer.
  - Step 2. Write data to PWMD
    - Here data is written directly to PWMD and simultaneously data is latched from the 4-bit buffer to the DLL register.
- Reading Data from DLL/PWMD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

Auto Shut-down Control – only for HT45F3430

The HT45F3430 device provides an auto shut-down function to protect the internal circuits from damaged by the external conditons.

Functional Description

For the device, the PWM and PA0/PB0/PC0 port have the auto shut-down function. When certain external conditions occur such as USB Power injected, the hardware can immediately shut down the PWM or specified I/O port which is usually used for controlling the load On/Off.



ASCR Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	ASPEN	ASIOEN	ASIOS1	ASIOS0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4 Unimplemented, read as "0"

Bit 5 **ASPEN**: PWM Auto shut-down enable control
0: Disable
1: Enable
If ASPEN=1, PWMACT will be cleared as 0 if INT0 interrupt occurs.

Bit 2 **ASIOEN**: I/O Ports Auto shut-down enable control
0: Disable
1: Enable
If ASIOEN=1, PA0/PB0/PC0 bit will be cleared as 0 if INT0 interrupt occurs. PA0, PB0 or PC0 is selected by ASIOS[1:0] bits

Bit 1~0 **ASIOS1~ASIOS0**: Auto shut-down I/O selection
00: PA0
01: PB0
1x: PC0
Note: When the ASPEN or ASIOEN bit is "1", if set the INT0F bit high by software, it will not clear the PWMACT or PA0/PB0/PC0 bit. It will occur only when the INT0F flag is triggered by the external INT0 pin.

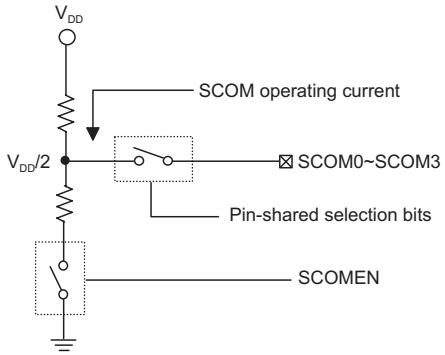
LCD SCOM Function – only for HT45F3430

The HT45F3430 device has the capability of driving external LCD panels. The common pins for LCD driving, SCOM0~SCOM3, are pin shared with certain pin on the I/O ports. The LCD signals are generated using the application program.

LCD Operation

An external LCD panel can be driven using this device by configuring the I/O pins as common pins. The LCD driver function is controlled using the SCOMC register which in addition to controlling the overall on/off function also controls the bias voltage setup function. This enables the LCD COM driver to generate the necessary $V_{DD}/2$ voltage levels for LCD 1/2 bias operation.

The SCOMEN bit in the SCOMC register is the overall master control for the LCD driver. The LCD SCOMn pin is selected to be used for LCD driving by the corresponding pin-shared function selection bits. Note that the Port Control register does not need to first setup the pins as outputs to enable the LCD driver operation.



LCD COM Bias

LCD Bias Current Control

The LCD COM driver enables a range of selections to be provided to suit the requirement of the LCD panel which are being used. The bias resistor choice is implemented using the ISEL1 and ISEL0 bits in the SCOMC register.

SCOMC Register

Bit	7	6	5	4	3	2	1	0
Name	—	ISEL1	ISEL0	SCOMEN	—	—	—	—
R/W	—	R/W	R/W	R/W	—	—	—	—
POR	—	0	0	0	—	—	—	—

Bit 7 Unimplemented, read as "0"

Bit 6~5 **ISEL1~ISEL0**: Select resistor for R type LCD bias current (V_{DD}=5V)
00: 2×100kΩ (1/2 Bias), I_{BIAS}=25μA
01: 2×50 kΩ (1/2 Bias), I_{BIAS}=50μA
10: 2×25 kΩ (1/2 Bias), I_{BIAS}=100μA
11: 2×12.5 kΩ (1/2 Bias), I_{BIAS}=200μA

Bit 4 **SCOMEN**: LCD function enable control bit
0: Disable
1: Enable
When SCOMEN is set, it will turn on the DC path of resistor to generate 1/2 V_{DD} bias voltage.

Bit 3~0 Unimplemented, read as "0"

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 devices contain external and internal interrupts functions. The external interrupt is generated by the action of the external INTn pin, while the internal interrupts are generated by various internal functions such as the TM, Time Base, EEPROM, OCP, OVP and A/D converter.

Interrupt Registers

Overall interrupt control, which basically means the setting of request flags when certain microcontroller conditions occur and the setting of interrupt enable bits by the application program, is controlled by a series of registers, located in the Special Purpose Data Memory, as shown in the accompanying table. The registers fall into three categories. The first is the INTC0~INTC2 registers which setup the primary interrupts, the second is the MFI register which setups the Multi-function interrupt. 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	—	—
OCP	OCPE	OCPF	—
OVP	OVPE	OVPF	—
INTn Pin	INTnE	INTnF	For HT45F3420, n=0 only For HT45F3430, n=0 or 1
Multi-function	MFE	MFF	—
A/D Converter	ADE	ADF	—
Time Base	TBnE	TBnF	n=0 or 1
EEPROM	DEE	DEF	—
TM	STMAE	STMAF	—
	STMPE	STMPF	—

Interrupt Register Bit Naming Conventions

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	—	—	INT0S1	INT0S0
INTC0	—	INT0F	OVPF	OCPF	INT0E	OVPE	OCPE	EMI
INTC1	DEF	ADF	—	MFF	DEE	ADE	—	MFE
INTC2	—	—	TB1F	TB0F	—	—	TB1E	TB0E
MFI	—	—	STMAF	STMPF	—	—	STMAE	STMPE

Interrupt Register List – HT45F3420

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	INT0F	OVPF	OCPF	INT0E	OVPE	OCPE	EMI
INTC1	DEF	ADF	—	MFF	DEE	ADE	—	MFE
INTC2	—	INT1F	TB1F	TB0F	—	INT1E	TB1E	TB0E
MFI	—	—	STMAF	STMPF	—	—	STMAE	STMPE

Interrupt Register List – HT45F3430

INTEG Register – HT45F3420

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

Bit 7 ~ 2 Unimplemented, read as "0"

Bit 1 ~ 0 **INT0S1~INT0S0**: INT0 pin interrupt active edge selection
00: Disable Interrupt
01: Rising Edge
10: Falling Edge
11: Both rising and falling edges

INTEG Register – HT45F3430

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

Bit 7 ~ 4 Unimplemented, read as "0"

Bit 3 ~ 2 **INT1S1~INT1S0**: INT1pin interrupt active edge selection
00: Disable Interrupt
01: Rising Edge
10: Falling Edge
11: Both rising and falling edges

Bit 1 ~ 0 **INT0S1~INT0S0**: INT0 pin interrupt active edge selection
00: Disable Interrupt
01: Rising Edge
10: Falling Edge
11: Both rising and falling edges

INTC0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	INT0F	OVPF	OCPF	INT0E	OVPE	OCPE	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 **INT0F**: External Interrupt 0 Request Flag
0: No request
1: Interrupt request

Bit 5 **OVPF**: OVP Interrupt Request Flag
0: No request
1: Interrupt request

Bit 4 **OCPF**: OCP Interrupt Request Flag
0: No request
1: Interrupt request

Bit 3 **INT0E**: External Interrupt 0 Control
0: Disable
1: Enable

Bit 2 **OVPE**: OVP Interrupt Control
0: Disable
1: Enable

Bit 1 **OCPE**: OCP Interrupt Control
0: Disable
1: Enable

Bit 0 **EMI**: Global Interrupt Control
0: Disable
1: Enable

INTC1 Register

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

Bit 7 **DEF**: Data EEPROM Interrupt Request Flag
0: No request
1: Interrupt request

Bit 6 **ADF**: A/D Converter Interrupt Request Flag
0: No request
1: Interrupt request

Bit 5 Unimplemented, read as "0"

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

Bit 3 **DEE**: Data EEPROM Interrupt Control
0: Disable
1: Enable

Bit 2 **ADE**: A/D Converter Interrupt Control
0: Disable
1: Enable

Bit 1 Unimplemented, read as "0"

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

INTC2 Register – HT45F3420

Bit	7	6	5	4	3	2	1	0
Name	—	—	TB1F	TB0F	—	—	TB1E	TB0E
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5 **TB1F**: Time Base 1 Interrupt Request Flag
0: No request
1: Interrupt request

Bit 4 **TB0F**: Time Base 0 Interrupt Request Flag
0: No request
1: Interrupt request

Bit 3~2 Unimplemented, read as "0"

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

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

INTC2 Register – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	—	INT1F	TB1F	TB0F	—	INT1E	TB1E	TB0E
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **INT1F**: External Interrupt 1 Request Flag
0: No request
1: Interrupt request

Bit 5 **TB1F**: Time Base 1 Interrupt Request Flag
0: No request
1: Interrupt request

Bit 4 **TB0F**: Time Base 0 Interrupt Request Flag
0: No request
1: Interrupt request

Bit 3 Unimplemented, read as "0"

Bit 2 **INT1E**: External Interrupt 1 Control
0: Disable
1: Enable

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

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

MFI Register

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

Bit 7 ~ 6 Unimplemented, read as "0"
Bit 5 **STMAF**: STM Comparator A Interrupt Request Flag
0: No request
1: Interrupt request
Bit 4 **STMPF**: STM Comparator P Interrupt Request Flag
0: No request
1: Interrupt request
Bit 3 ~2 Unimplemented, read as "0"
Bit 1 **STMAE**: STM Comparator A Interrupt Control
0: Disable
1: Enable
Bit 0 **STMPE**: STM Comparator P Interrupt Control
0: Disable
1: Enable

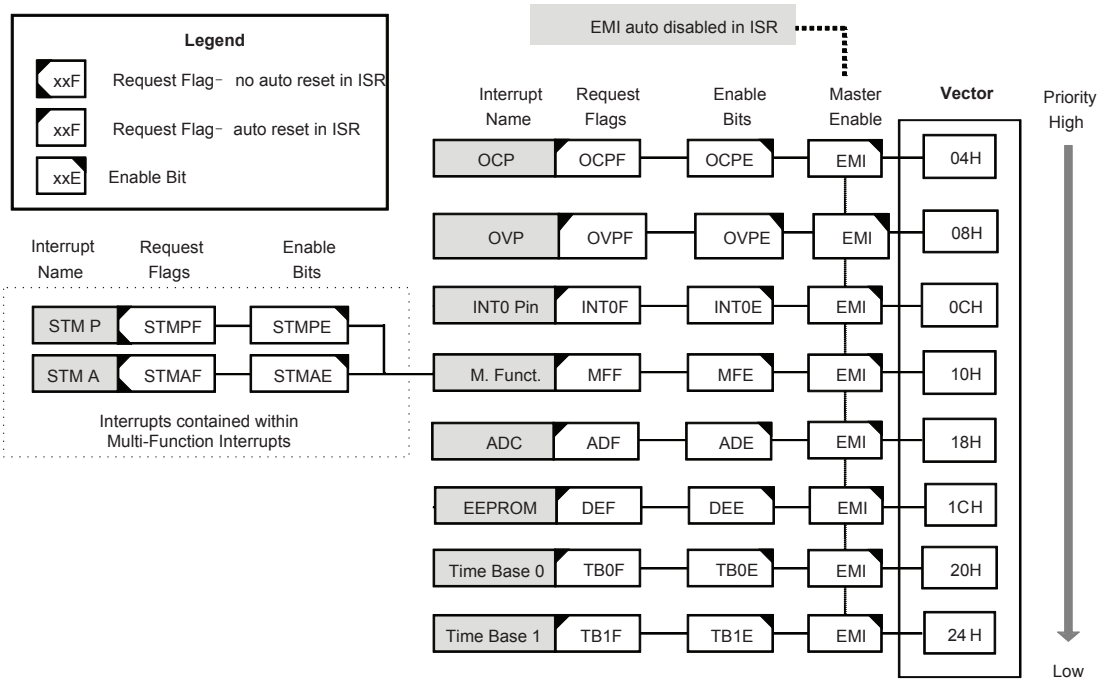
Interrupt Operation

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.

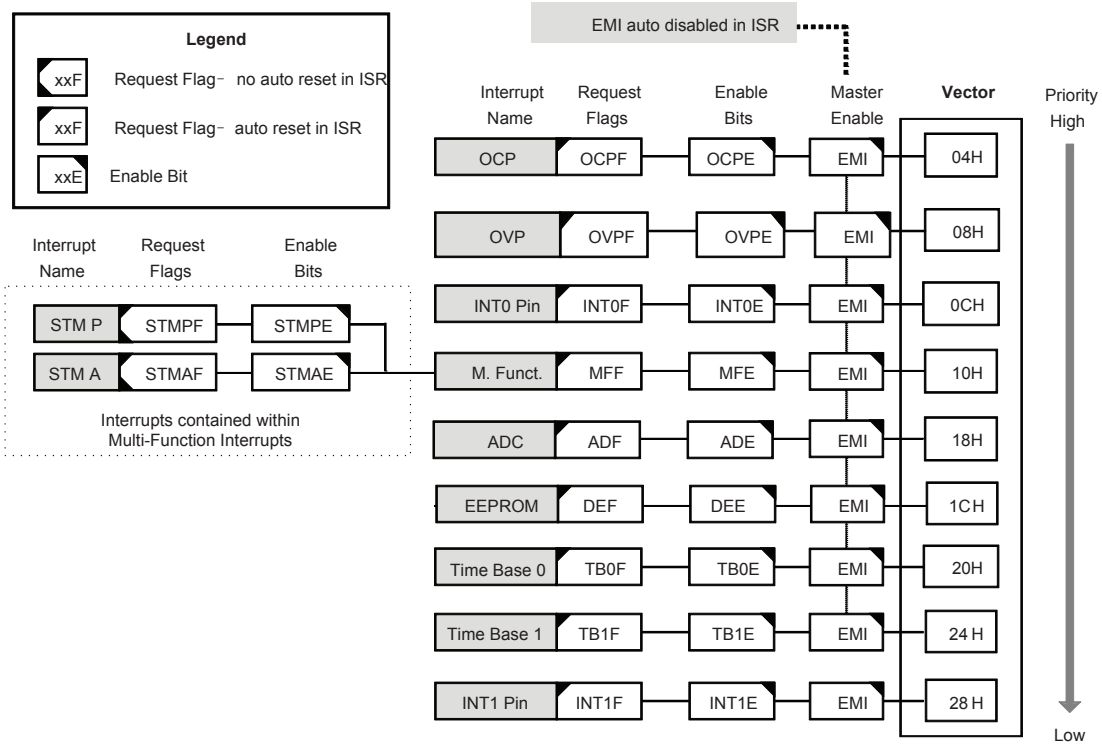
When an interrupt is generated, the Program Counter, which stores the address of the next instruction to be executed, will be transferred onto the stack. The Program Counter will then be loaded with a new address which will be the value of the corresponding interrupt vector. The microcontroller will 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 – HT45F3420



Interrupt Structure – HT45F3430

External Interrupt

The external interrupt is controlled by signal transitions on the pin INT0 or INT1. For the HT45F3430 device, the INT0 and INT1 input source is selectable which can be set by the IFS0 register bits. An external interrupt request will take place when the external interrupt request flag, INTnF, is set, which will occur when a transition, whose type is chosen by the edge select bits, appears on the external interrupt pins. To allow the program to branch to the interrupt vector address, the global interrupt enable bit, EMI, and the external interrupt enable bit, INTnE, must first be set. Additionally the correct interrupt edge type must be selected using the INTEG register to enable the external interrupt function and to choose the trigger edge type. As the external interrupt pins are pin-shared with I/O pins, it can only be configured as external interrupt pin if its external interrupt enable bit in the corresponding interrupt register has been set. The pin must also be setup as an input by setting the corresponding bit in the port control register. When the interrupt is enabled, the stack is not full and the correct transition type appears on the external interrupt pin, a subroutine call to the external interrupt vector, will take place. When the interrupt is serviced, the external interrupt request flag, INTnF, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts. Note that the pull-high resistor selection on the external interrupt pin will remain valid even if the pin is used as an external interrupt input.

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

Over Current Protection Interrupt

The OCP Interrupt is controlled by detecting the OCP input current. An OCP Interrupt request will take place when the OCP Interrupt request flag, OCPF, 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 OCP Interrupt enable bit, OCPE, 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 OCP Interrupt vector, will take place. When the interrupt is serviced, the OCP Interrupt flag, OCPF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

Over Voltage Protection Interrupt

The OVP Interrupt is controlled by detecting the input voltage. An OVP Interrupt request will take place when the OVP Interrupt request flag, OVPF, is set, which occurs when a large voltage is detected. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and OVP Interrupt enable bit, OVPE, must first be set. When the interrupt is enabled, the stack is not full and a large voltage is detected, a subroutine call to the OVP Interrupt vector, will take place. When the interrupt is serviced, the OVP Interrupt flag, OVPF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

Multi-function Interrupt

Within the devices there is only one Multi-function interrupts. Unlike the other independent interrupts, the interrupt has no independent source, but rather are formed from other existing interrupt sources, namely the STM CCRA and CCRP interrupts.

A Multi-function interrupt request will take place when the Multi-function interrupt request flag, MFF is set. The Multi-function interrupt flag will be set when any of 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 the Multi-function interrupt occurs, a subroutine call to the Multi-function interrupt vector will take place. When the interrupt is serviced, the related Multi-Function request flag, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts.

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

A/D Converter Interrupt

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

Time Base Interrupt

The function of the Time Base Interrupts is to provide regular time signal in the form of an internal interrupt. They are controlled by the overflow signals from their respective timer functions. When these happens their respective interrupt request flags, TB0F or TB1F will be set. To allow the program to branch to their respective interrupt vector addresses, the global interrupt enable bit, EMI and Time Base enable bits, TB0E or TB1E, must first be set. When the interrupt is enabled, the stack is not full and the Time Base overflows, a subroutine call to their respective vector locations will take place. When the interrupt is serviced, the respective interrupt request flag, TB0F or TB1F, will be automatically reset and the EMI bit will be cleared to disable other interrupts.

The purpose of the Time Base Interrupt is to provide an interrupt signal at fixed time periods. Their clock sources originate from the internal clock source f_{TB}. This f_{TB} input clock passes through a divider, the division ratio of which is selected by programming the appropriate bits in the TBC register to obtain longer interrupt periods whose value ranges. The clock source that generates f_{TB}, which in turn controls the Time Base interrupt period, can originate from several different sources which is selected using the TBCK bit in the TBC register.

TBC Register

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

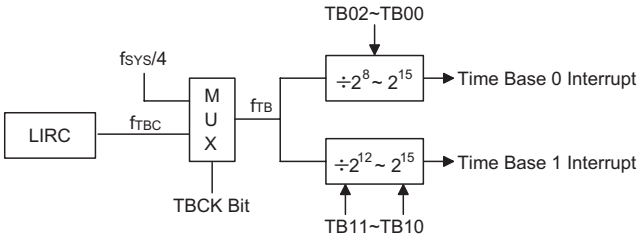
Bit 7 **TBON**: TB0 and TB1 Control bit
0: Disable
1: Enable

Bit 6 **TBCK**: Select f_{TB} Clock Source
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: $2^{15}/f_{TB}$



Time Base Interrupt

EEPROM Interrupt

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

TM Interrupts

The STM has two interrupts which are both contained within the Multi-function Interrupt. For the TM there are two interrupt request flags STMPF and STMAF and two enable bits STMPE and STMAE. A TM interrupt request will take place when any of the TM request flags is set, a situation which occurs when a TM comparator P or comparator A match situation happens.

To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the respective TM Interrupt enable bit, and associated Multi-function interrupt enable bit, MFF, 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 TM 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 MFF flag will be automatically cleared. As the TM interrupt request flags will not be automatically cleared, they have to be cleared by the application program.

Interrupt Wake-up Function

Each of the interrupt functions has the capability of waking up the microcontroller when in the SLEEP or IDLE Mode. A wake-up is generated when an interrupt request flag changes from low to high and is independent of whether the interrupt is enabled or not. Therefore, even though the device is in the SLEEP or IDLE Mode and its system oscillator stopped, situations such as external edge transitions on the external interrupt pin, a low power supply voltage or comparator input change may cause their respective interrupt flag to be set high and consequently generate an interrupt. Care must therefore be taken if spurious wake-up situations are to be avoided. If an interrupt wake-up function is to be disabled then the corresponding interrupt request flag should be set high before the device enters the SLEEP or IDLE Mode. The interrupt enable bits have no effect on the interrupt wake-up function.

Programming Considerations

By disabling the relevant interrupt enable bits, a requested interrupt can be prevented from being serviced, however, once an interrupt request flag is set, it will remain in this condition in the interrupt register until the corresponding interrupt is serviced or until the request flag is cleared by the application program.

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

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

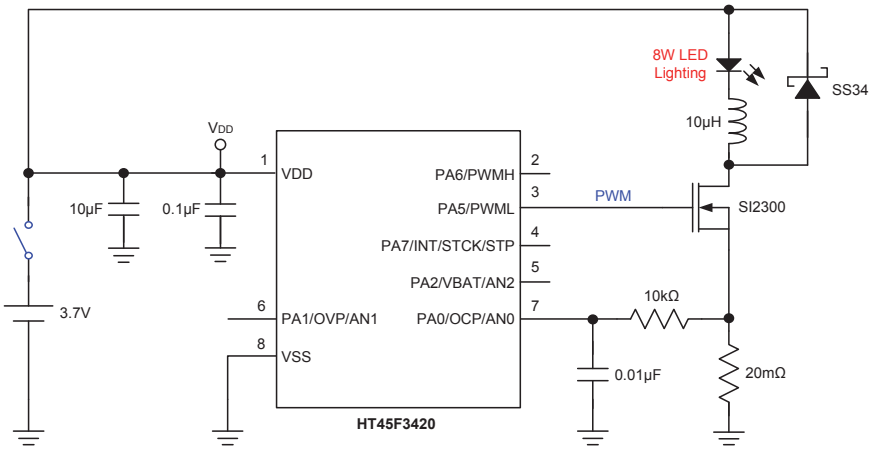
Every interrupt has the capability of waking up the microcontroller when it is in SLEEP or IDLE Mode, the wake up being generated when the interrupt request flag changes from low to high. If it is required to prevent a certain interrupt from waking up the microcontroller then its respective request flag should be first set high before enter SLEEP or IDLE Mode.

As only the Program Counter is pushed onto the stack, then when the interrupt is serviced, if the contents of the accumulator, status register or other registers are altered by the interrupt service program, their contents should be saved to the memory at the beginning of the interrupt service routine.

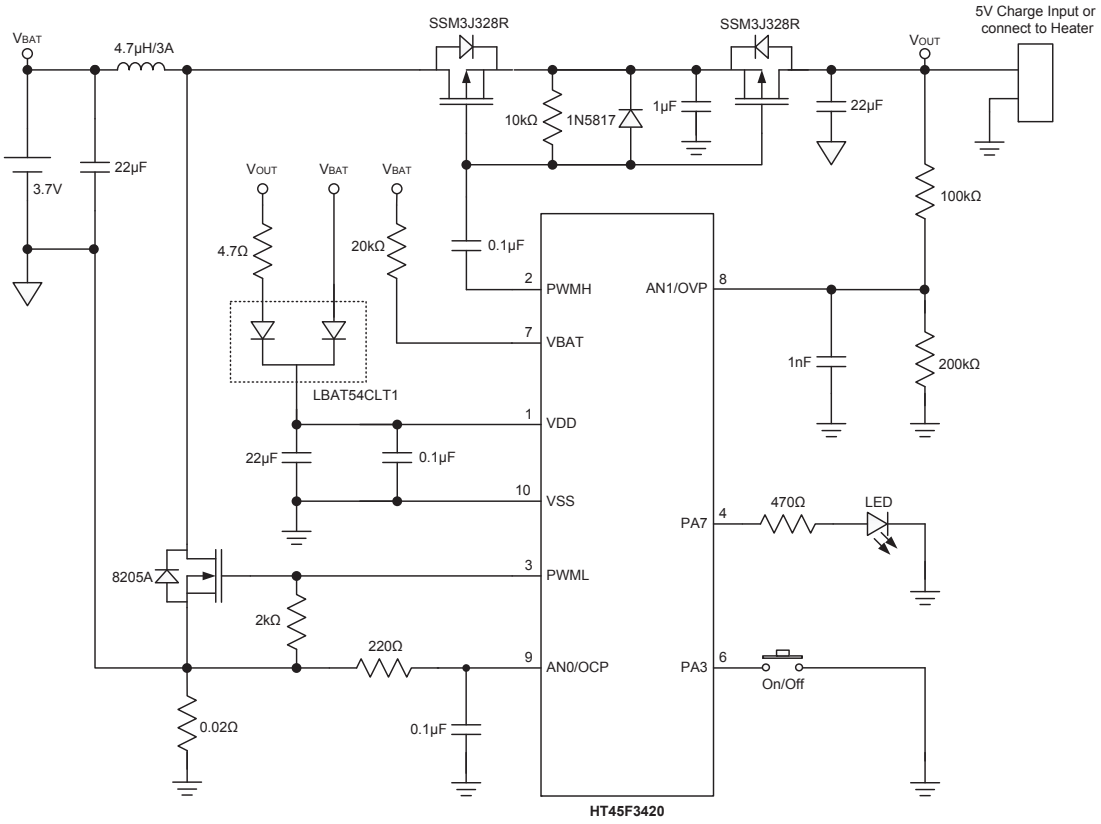
To return from an interrupt subroutine, either a RET or RETI instruction may be executed. The RETI instruction in addition to executing a return to the main program also automatically sets the EMI bit high to allow further interrupts. The RET instruction however only executes a return to the main program leaving the EMI bit in its present zero state and therefore disabling the execution of further interrupts.

Application Circuits

LED Flashlight



Electronic Cigarette



Instruction Set

Introduction

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

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

Instruction Timing

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

Moving and Transferring Data

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

Arithmetic Operations

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

Logical and Rotate Operation

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

Branches and Control Transfer

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

Bit Operations

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

Table Read Operations

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

Other Operations

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

Instruction Set Summary

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

Table Conventions

x: Bits immediate data
m: Data Memory address
A: Accumulator
i: 0~7 number of bits
addr: Program memory address

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

Mnemonic	Description	Cycles	Flag Affected
Data Move			
MOV A,[m]	Move Data Memory to ACC	1	None
MOV [m],A	Move ACC to Data Memory	1Note	None
MOV A,x	Move immediate data to ACC	1	None
Bit Operation			
CLR [m].i	Clear bit of Data Memory	1Note	None
SET [m].i	Set bit of Data Memory	1Note	None
Branch Operation			
JMP addr	Jump unconditionally	2	None
SZ [m]	Skip if Data Memory is zero	1Note	None
SZA [m]	Skip if Data Memory is zero with data movement to ACC	1Note	None
SZ [m].i	Skip if bit i of Data Memory is zero	1Note	None
SNZ [m].i	Skip if bit i of Data Memory is not zero	1Note	None
SIZ [m]	Skip if increment Data Memory is zero	1Note	None
SDZ [m]	Skip if decrement Data Memory is zero	1Note	None
SIZA [m]	Skip if increment Data Memory is zero with result in ACC	1Note	None
SDZA [m]	Skip if decrement Data Memory is zero with result in ACC	1Note	None
CALL addr	Subroutine call	2	None
RET	Return from subroutine	2	None
RET A,x	Return from subroutine and load immediate data to ACC	2	None
RETI	Return from interrupt	2	None
Table Read Operation			
TABRD [m]	Read table (specific page) to TBLH and Data Memory	2Note	None
TABRDC [m]	Read table (current page) to TBLH and Data Memory	2Note	None
TABRDL [m]	Read table (last page) to TBLH and Data Memory	2Note	None
Miscellaneous			
NOP	No operation	1	None
CLR [m]	Clear Data Memory	1Note	None
SET [m]	Set Data Memory	1Note	None
CLR WDT	Clear Watchdog Timer	1	TO, PDF
CLR WDT1	Pre-clear Watchdog Timer	1	TO, PDF
CLR WDT2	Pre-clear Watchdog Timer	1	TO, PDF
SWAP [m]	Swap nibbles of Data Memory	1Note	None
SWAPA [m]	Swap nibbles of Data Memory with result in ACC	1	None
HALT	Enter power down mode	1	TO, PDF

Note: 1. For skip instructions, if the result of the comparison involves a skip then two cycles are required, if no skip takes place only one cycle is required.
2. Any instruction which changes the contents of the PCL will also require 2 cycles for execution.
3. For the "CLR WDT1" and "CLR WDT2" instructions the TO and PDF flags may be affected by the execution status. The TO and PDF flags are cleared after both "CLR WDT1" and "CLR WDT2" instructions are consecutively executed. Otherwise the TO and PDF flags remain unchanged.

Instruction Definition

ADC A,[m] Add Data Memory to ACC with Carry

Description The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the Accumulator.

Operation ACC ← ACC + [m] + C

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

ADCM A,[m] Add ACC to Data Memory with Carry

Description The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the specified Data Memory.

Operation [m] ← ACC + [m] + C

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

ADD A,[m] Add Data Memory to ACC

Description The contents of the specified Data Memory and the Accumulator are added. The result is stored in the Accumulator.

Operation ACC ← ACC + [m]

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

ADD A,x Add immediate data to ACC

Description The contents of the Accumulator and the specified immediate data are added. The result is stored in the Accumulator.

Operation ACC ← ACC + x

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

ADDM A,[m] Add ACC to Data Memory

Description The contents of the specified Data Memory and the Accumulator are added. The result is stored in the specified Data Memory.

Operation [m] ← ACC + [m]

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

AND A,[m] Logical AND Data Memory to ACC

Description Data in the Accumulator and the specified Data Memory perform a bitwise logical AND operation. The result is stored in the Accumulator.

Operation ACC ← ACC "AND" [m]

Affected flag(s) Z

AND A,x Logical AND immediate data to ACC

Description Data in the Accumulator and the specified immediate data perform a bit wise logical AND operation. The result is stored in the Accumulator.

Operation ACC ← ACC "AND" x

Affected flag(s) Z

ANDM A,[m] Logical AND ACC to Data Memory

Description Data in the specified Data Memory and the Accumulator perform a bitwise logical AND operation. The result is stored in the Data Memory.

Operation [m] ← ACC "AND" [m]

Affected flag(s) Z

CALL addr Subroutine call

Description Unconditionally calls a subroutine at the specified address. The Program Counter then increments by 1 to obtain the address of the next instruction which is then pushed onto the stack. The specified address is then loaded and the program continues execution from this new address. As this instruction requires an additional operation, it is a two cycle instruction.

Operation Stack ← Program Counter + 1
Program Counter ← addr

Affected flag(s) None

CLR [m] Clear Data Memory

Description Each bit of the specified Data Memory is cleared to 0.

Operation [m] ← 00H

Affected flag(s) None

CLR [m].i Clear bit of Data Memory

Description Bit i of the specified Data Memory is cleared to 0.

Operation [m].i ← 0

Affected flag(s) None

CLR WDT Clear Watchdog Timer

Description The TO, PDF flags and the WDT are all cleared.

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CLR WDT1 Pre-clear Watchdog Timer

Description The TO, PDF flags and the WDT are all cleared. Note that this instruction works in conjunction with CLR WDT2 and must be executed alternately with CLR WDT2 to have effect. Repetitively executing this instruction without alternately executing CLR WDT2 will have no effect.

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CLR WDT2 Pre-clear Watchdog Timer

Description The TO, PDF flags and the WDT are all cleared. Note that this instruction works in conjunction with CLR WDT1 and must be executed alternately with CLR WDT1 to have effect. Repetitively executing this instruction without alternately executing CLR WDT1 will have no effect.

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CPL [m] Complement Data Memory

Description Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa.

Operation [m] ← $\overline{[m]}$

Affected flag(s) Z

CPLA [m] Complement Data Memory with result in ACC

Description Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa. The complemented result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation $ACC \leftarrow \overline{[m]}$

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] − C

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

SBCM A,[m] Subtract Data Memory from ACC with Carry and result in Data Memory

Description: The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation: [m] ← ACC − [m] − C

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

SDZ [m] Skip if decrement Data Memory is 0

Description: The contents of the specified Data Memory are first decremented by 1. If the result is 0 the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation:
[m] ← [m] − 1
Skip if [m]=0

Affected flag(s): None

SDZA [m] Skip if decrement Data Memory is zero with result in ACC

Description The contents of the specified Data Memory are first decremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.

Operation ACC ← [m] − 1
Skip if ACC=0

Affected flag(s) None

SET [m] Set Data Memory

Description Each bit of the specified Data Memory is set to 1.

Operation [m] ← FFH

Affected flag(s) None

SET [m].i Set bit of Data Memory

Description Bit i of the specified Data Memory is set to 1.

Operation [m].i ← 1

Affected flag(s) None

SIZ [m] Skip if increment Data Memory is 0

Description The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation [m] ← [m] + 1
Skip if [m]=0

Affected flag(s) None

SIZA [m] Skip if increment Data Memory is zero with result in ACC

Description The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation ACC ← [m] + 1
Skip if ACC=0

Affected flag(s) None

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

Description If bit i of the specified Data Memory is not 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is 0 the program proceeds with the following instruction.

Operation Skip if [m].i ≠ 0

Affected flag(s) None

SUB A,[m] Subtract Data Memory from ACC

Description The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC − [m]

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

SUBM A,[m] Subtract Data Memory from ACC with result in Data Memory

Description The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation [m] ← ACC − [m]

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

SUB A,x Subtract immediate data from ACC

Description The immediate data specified by the code is subtracted from the contents of the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC − x

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

SWAP [m] Swap nibbles of Data Memory

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

Operation [m].3~[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 (TBHP and TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

TABRDC [m] Read table (current page) to TBLH and Data Memory

Description The low byte of the program code (current page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

TABRDL [m] Read table (last page) to TBLH and Data Memory

Description The low byte of the program code (last page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

XOR A,[m] Logical XOR Data Memory to ACC

Description Data in the Accumulator and the specified Data Memory perform a bitwise logical XOR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″XOR″ [m]

Affected flag(s) Z

XORM A,[m] Logical XOR ACC to Data Memory

Description Data in the specified Data Memory and the Accumulator perform a bitwise logical XOR operation. The result is stored in the Data Memory.

Operation [m] ← ACC ″XOR″ [m]

Affected flag(s) Z

XOR A,x Logical XOR immediate data to ACC

Description Data in the Accumulator and the specified immediate data perform a bitwise logical XOR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″XOR″ x

Affected flag(s) Z

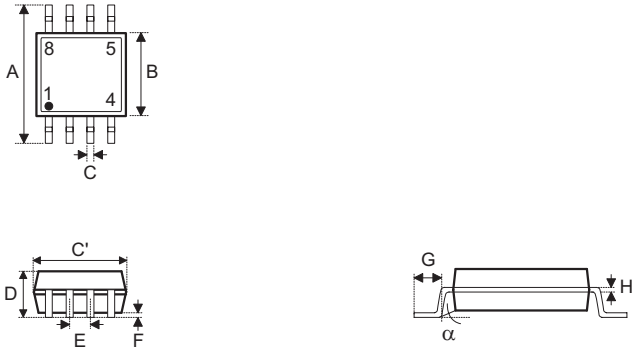
Package Information

Note that the package information provided here is for consultation purposes only. As this information may be updated at regular intervals users are reminded to consult the Holtek website for the latest version of the Package/Carton Information.

Additional supplementary information with regard to packaging is listed below. Click on the relevant section to be transferred to the relevant website page.

- Package Information (include Outline Dimensions, Product Tape and Reel Specifications)
- The Operation Instruction of Packing Materials
- Carton information

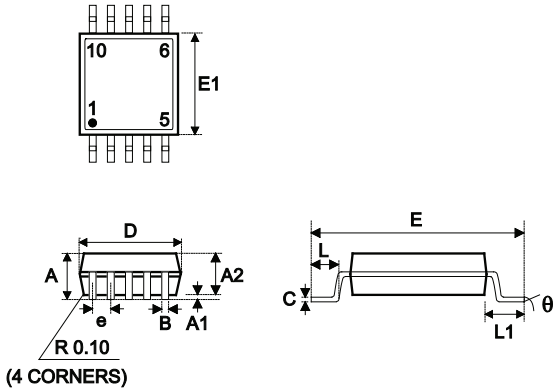
8-pin SOP (150mil) Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.012	—	0.020
C′	—	0.193 BSC	—
D	—	—	0.069
E	—	0.050 BSC	—
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	—	6.00 BSC	—
B	—	3.90 BSC	—
C	0.31	—	0.51
C′	—	4.90 BSC	—
D	—	—	1.75
E	—	1.27 BSC	—
F	0.10	—	0.25
G	0.40	—	1.27
H	0.10	—	0.25
α	0°	—	8°

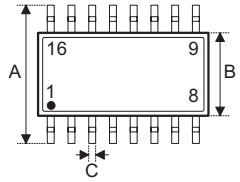
10-pin MSOP Outline Dimensions

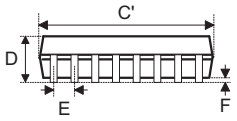


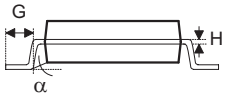
Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	—	—	0.043
A1	0.000	—	0.006
A2	0.030	0.033	0.037
B	0.007	—	0.013
C	0.003	—	0.009
D	—	0.118 BSC	—
E	—	0.193 BSC	—
E1	—	0.118 BSC	—
e	—	0.020 BSC	—
L	0.016	0.024	0.031
L1	—	0.037 BSC	—
y	—	0.004	—
θ	0°	—	8°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	—	—	1.10
A1	0.00	—	0.15
A2	0.75	0.85	0.95
B	0.17	—	0.33
C	0.08	—	0.23
D	—	3.00 BSC	—
E	—	4.90 BSC	—
E1	—	3.00 BSC	—
e	—	0.50 BSC	—
L	0.40	0.60	0.80
L1	—	0.95 BSC	—
y	—	0.10	—
θ	0°	—	8°

16-pin NSOP (150mil) Outline Dimensions



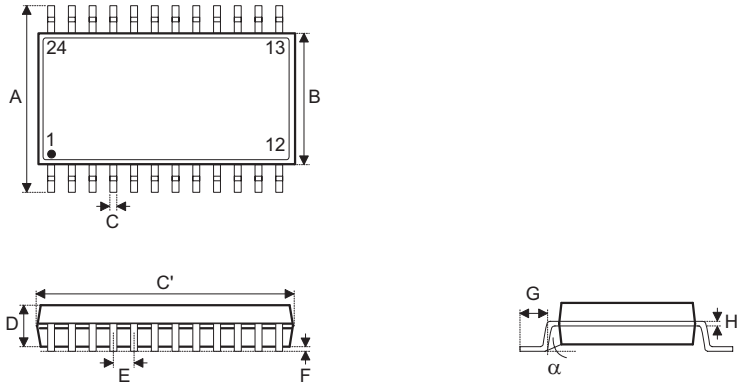




Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.012	—	0.020
C'	—	0.390 BSC	—
D	—	—	0.069
E	—	0.050 BSC	—
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	—	6.000 BSC	—
B	—	3.900 BSC	—
C	0.31	—	0.51
C'	—	9.900 BSC	—
D	—	—	1.75
E	—	1.270 BSC	—
F	0.10	—	0.25
G	0.40	—	1.27
H	0.10	—	0.25
α	0°	—	8°

24-pin SSOP (150mil) Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.008	—	0.012
C’	—	0.341 BSC	—
D	—	—	0.069
E	—	0.025 BSC	—
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	—	6.000 BSC	—
B	—	3.900 BSC	—
C	0.20	—	0.30
C’	—	8.660 BSC	—
D	—	—	1.75
E	—	0.635 BSC	—
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
α	0°	—	8°