Personal Care Flash MCU

HT45F3520/HT45F3530

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 21 bidirectional I/O lines
- Programmable I/O port source current for LED driving applications – only for HT45F3530
- Software controlled 4-SCOM LCD driver with 1/2 bias – only for HT45F3530
- Up to two pin-shared external interrupt
- Multiple Timer Modules 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
- Multi-channel 12-bit resolution A/D converter
- Internal DC-to-DC Converter
  - DC-to-DC converter input voltage range: 0.9V~4.4V
  - DC-to-DC converter output voltage range: 2.85V~5.0V
- 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: 16-pin NSOP, 24-pin SSOP

General Description

These devices are dedicated for use in Battery booster and power control applications. 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. Multiple and extremely flexible Timer Modules provide 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.

A high speed HIRC oscillator and a low speed LIRC oscillator are provided, they are fully integrated system oscillators and 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 personal care applications although the devices will also lend themselves for use in a range of other related applications.

Block Diagram
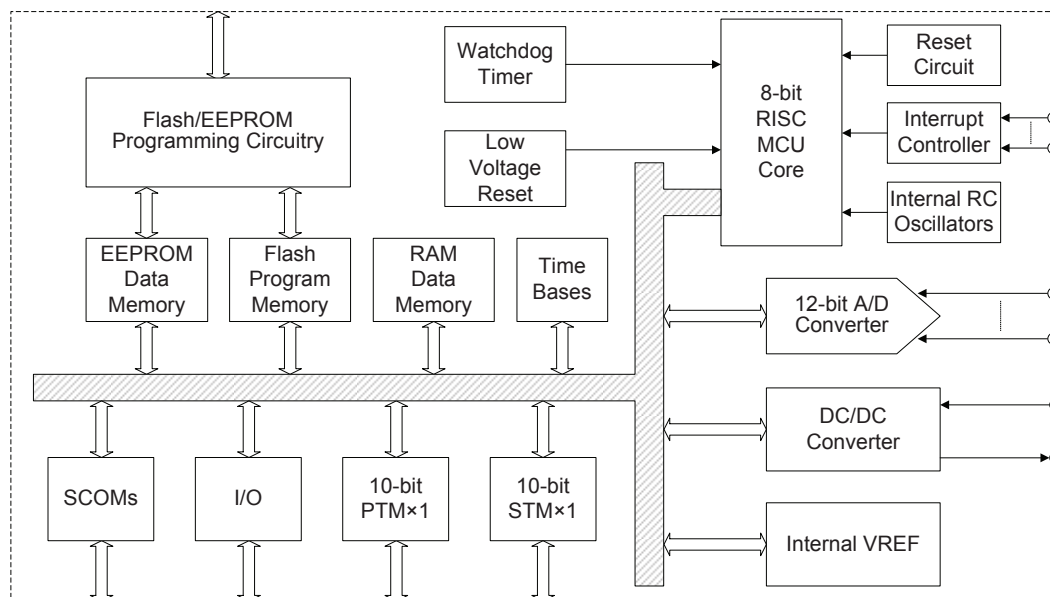
Note: The 4-SCOM LCD driving function is only for the HT45F3530.

Selection Table

Most features are common to these devices, the main features distinguishing them are Memory capacity, I/O count, external interrupt count, A/D converter channel count, LCD 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
HT45F3520	2.2V~5.5V	1K×14	64×8	32×8	13	1	12-bit×4
HT45F3530	2.2V~5.5V	2K×15	128×8	64×8	21	2	12-bit×8

Part No.	DC/DC	LCD Driver	Timer Module	Time Base	Stack	Package
HT45F3520	√	—	10-bit PTM×1 10-bit STM×1	2	4	16NSOP
HT45F3530	√	4-SCOM	10-bit PTM×1 10-bit STM×1	2	4	24SSOP

Pin Assignment

```
         PA6/INT0 ▢ 1        16 ▢ PA7
     PA5/STPI/STP ▢ 2        15 ▢ PB0
     PA4/STCK/PTCK ▢ 3       14 ▢ PB1
  PA3/AN3/VREF/VREFI ▢ 4     13 ▢ PB2
 PA2/AN2/ICPCK/OCDSCK ▢ 5    12 ▢ PB3
          PA1/AN1 ▢ 6        11 ▢ PB4/PTPI/PTP
 PA0/AN0/ICPDA/OCDSDA ▢ 7    10 ▢ VDD/VOUT
              VSS ▢ 8         9 ▢ LX
```
HT45F3520/HT45V3520
16 NSOP-A

```
        PA6/INT0/AN6 ▢ 1        24 ▢ PA7/AN7
     PA5/STPI/STP/AN5 ▢ 2       23 ▢ PB0/INT1
 PA4/STCK/PTCK/[INT1]/AN4 ▢ 3   22 ▢ PB1
          PC3/SCOM3 ▢ 4         21 ▢ PB2
          PC2/SCOM2 ▢ 5         20 ▢ PB3
          PC1/SCOM1 ▢ 6         19 ▢ PB4
          PC0/SCOM0 ▢ 7         18 ▢ PB5
   PA3/AN3/VREF/VREFI ▢ 8       17 ▢ PB6
 PA2/AN2/ICPCK/OCDSCK ▢ 9       16 ▢ PB7
             PA1/AN1 ▢ 10       15 ▢ PC4/PTPI/PTP
 PA0/AN0/ICPDA/OCDSDA ▢ 11      14 ▢ VDD/VOUT
                 VSS ▢ 12       13 ▢ LX
```
HT45F3530/HT45V3530
24 SSOP-A

Note: 1. If the pin-shared pin functions have multiple outputs simultaneously, the desired pin-shared function is determined by the corresponding software control bits.

2. The OCDSDA and OCDSCK pins are supplied for the OCDS dedicated pins and as such are only available for the HT45V3520 and HT45V3530 EV devices.

Pin Description

With the exception of the power pins and some relevant transformer control pins, all pins on the device can be referenced by their Port name, e.g. PA0, PA1 etc, which refer to the digital I/O function of the pins. However these Port pins are also shared with other function such as the Analog to Digital Converter, Timer Module pins etc. The function of each pin is listed in the following table, however the details behind how each pin is configured is contained in other sections of the datasheet.

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

HT45F3520

Pin Name	Function	OPT	I/T	O/T	Description
PA0/AN0/ ICPDA/ OCDSDA	PA0	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	AN0	PAS0	AN	—	ADC input channel 0
	ICPDA	—	ST	CMOS	ICP Data/Adress Line
	OCDSDA	—	ST	CMOS	OCDS Data/Adress Line for EV device only
PA1/AN1	PA1	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	AN1	PAS0	AN	—	ADC input channel 1
PA2/AN2/ ICPCK/ OCDSCK	PA2	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	AN2	PAS0	AN	—	ADC input channel 2
	ICPCK	—	ST	CMOS	ICP Clock Line
	OCDSCK	—	ST	—	OCDS Clock Line for EV device only
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/STCK/ PTCK	PA4	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	STCK	STMC0	ST	—	STM clock input
	PTCK	PTMC0	ST	—	PTM clock input
PA5/STPI/STP	PA5	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	STPI	PAS1	ST	—	STM capture input
	STP	PAS1	—	CMOS	STM output
PA6/INT0	PA6	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	INT0	INTEG	ST	—	External interrupt input
PA7	PA7	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
PB0~PB3	PB0~PB3	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
PB4/PTPI/PTP	PB4	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	PTPI	PBS0	ST	—	PTM capture input
	PTP	PBS0	—	CMOS	PTM output

Pin Name	Function	OPT	I/T	O/T	Description
LX	LX	—	PWR	—	External inductor connection pin for Boost
VDD/VOUT	VDD	—	PWR	—	Positive power supply
	VOUT	—	PWR	—	DC-DC (Boost) output
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

HT45F3530

Pin Name	Function	OPT	I/T	O/T	Description
PA0/AN0/ ICPDA/ OCDSDA	PA0	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	AN0	PAS0	AN	—	ADC input channel 0
	ICPDA	—	ST	CMOS	ICP Data Line
	OCDSDA	—	ST	CMOS	OCDS Data Line for EV device only
PA1/AN1	PA1	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	AN1	PAS0	AN	—	ADC input channel 1
PA2/AN2/ ICPCK/ OCDSCK	PA2	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	AN2	PAS0	AN	—	ADC input channel 2
	ICPCK	—	ST	CMOS	ICP Clock Line
	OCDSCK	—	ST	—	OCDS Clock Line for EV device only
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/STCK/ PTCK/ [INT1]/AN4	PA4	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	STCK	STMC0	ST	—	STM clock input
	PTCK	PTMC0	ST	—	PTM clock input
	INT1	IFS0 INTEG	ST	—	External interrupt 1 input
	AN4	PAS1	AN	—	ADC input channel 4
PA5/STPI/ STP/AN5	PA5	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	STPI	PAS1	ST	—	STM capture input
	STP	PAS1	—	CMOS	STM output
	AN5	PAS1	AN	—	ADC input channel 5
PA6/INT0/AN6	PA6	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	INT0	INTEG	ST	—	External interrupt 0 input
	AN6	PAS1	AN	—	ADC input channel 6

Pin Name	Function	OPT	I/T	O/T	Description
PA7/AN7	PA7	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	AN7	PAS1	AN	—	ADC input channel 7
PB0/INT1	PB0	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	INT1	IFS0 INTEG	ST	—	External interrupt 1 input
PB1~PB7	PB1~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	—	SCOM	Software controlled LCD COM output
PC1/SCOM1	PC1	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	SCOM1	PCS0	—	SCOM	Software controlled LCD COM output
PC2/SCOM2	PC2	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	SCOM2	PCS0	—	SCOM	Software controlled LCD COM output
PC3/SCOM3	PC3	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	SCOM3	PCS0	—	SCOM	Software controlled LCD COM output
PC4/PTPI/PTP	PC4	PCPU PCS1	ST	CMOS	General purpose I/O. Register enabled pull-up (80mA driving capacity)
	PTPI	PCS1	ST	—	PTM capture input
	PTP	PCS1	—	CMOS	PTM output
LX	LX	—	PWR	—	External inductor connection pin for Boost
VDD/VOUT	VDD	—	PWR	—	Positive power supply
	VOUT	—	PWR	—	DC-DC (Boost) output
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
SCOM: Software controlled LCD COM

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

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_{LIRC} on	—	3	5	µA
		5V		—	5	10	µA
	Standby Current (IDLE1 mode, HIRC)	3V	No load, all peripherals off, f_{LIRC} on, $f_{SYS}=f_{HIRC}/2=4MHz$	—	180	250	µA
		5V		—	400	600	µA
		3V	No load, all peripherals off, f_{LIRC} 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}	I/O Port Sink Current except PB4 (HT45F3520)	3V	$V_{OL}=0.1V_{DD}$	18	36	—	mA
		5V		33	66	—	mA
	PB4 Port Sink Current (HT45F3520)	3V	$V_{OL}=0.1V_{DD}$	16	32	—	mA
		5V		40	80	—	mA
	I/O Port Sink Current except PC4 (HT45F3530)	3V	$V_{OL}=0.1V_{DD}$	18	36	—	mA
		5V		33	66	—	mA
	PC4 Port Sink Current (HT45F3530)	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}	I/O Port Source Current except PB4 (HT45F3520)	3V	$V_{OH}=0.9V_{DD}$	-3	-6	—	mA
		5V		-7	-14	—	mA
	PB4 Port Source Current (HT45F3520)	3V	$V_{OH}=0.9V_{DD}$	-18	-38	—	mA
		5V		-40	-80	—	mA
	I/O Port Source Current except PC4 (HT45F3530)	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 Port Source Current (HT45F3530)	3V	$V_{OH}=0.9V_{DD}$	-18	-38	—	mA
		5V		-40	-80	—	mA
R_{PH}	Pull-high Resistance for I/O Ports	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ

A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	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°C~70°C	-7%	8	+7%	MHz
		2.2V~5.5V	Ta= -40°C~85°C	-11%	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}	xTCK 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}
		—	$f_{SYS}=f_{LIRC}$	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, $f_H=f_{HIRC}$	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

ADC Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	A/D Converter Operating Voltage	—	—	2.7	—	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	2.2V	SAINS[3:0] = 0000B, SAVRS[1:0] = 01B, $V_{REF}=V_{DD}$, t_{ADCK} =0.5µs	-3	—	+3	LSB
		3V					
		5V					
		2.2V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=10µs	-3	—	+3	LSB
		3V					
		5V					
INL	Integral Non-linearity	2.2V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=0.5µs	-4	—	+4	LSB
		3V					
		5V					
		2.2V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=10µs	-4	—	+4	LSB
		3V					
		5V					
I_{ADC}	Additional Power Consumption if A/D Converter is used	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	2.2V	No load	—	180	220	µA
		3V	No load	—	200	275	µA
		5V	No load	—	250	330	µA
V_{CM}	PGA Common Mode Voltage Range	2.2V	—	V_{SS} -0.3	—	V_{DD} -1.4	V
		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	2.2V	—	V_{SS} +0.1	—	V_{DD} -0.1	V
		3V	—	V_{SS} +0.1	—	V_{DD} -0.1	V
		5V	—	V_{SS} +0.1	—	V_{DD} -0.1	V
Ga	PGA Gain Accuracy	—	V_{DD}=1.9V~5.5V for $V_{RI}=V_{BG}$, V_{DD}=1.8V~5.5V for $V_{RI}=V_{REFI}$	-5	—	+5	%

LVR Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{LVR}	Low Voltage Reset Voltage	—	LVR Enable	-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

DC/DC (Boost) Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{BAT}	Battery Input Voltage Range	—	V_{OUT} = 2.85V/3.0V/3.15V/3.3V	0.9	—	2.8	V
		—	V_{OUT} = 5V	0.9	—	4.4	V
I_{DCSTB}	Standby Supply Current	—	V_{BAT} = 0.9V, MCU HALT (the most power saving mode), DCDIS=1, WDT disable, No load	—	—	2	µA
		—	V_{BAT} = 1.8V, MCU HALT (the most power saving mode), DCDIS = 1, WDT disable, No load	—	—	2	µA
L_{IN}	Input Inductor Value	—	—	2.2	4.7	10	µH
I_L	Input Inductor Current Rating	—	—	250	—	—	mA
R_{IL}	Inductor DC Resistance	—	—	—	—	0.5	Ω
C_{IN}	Input Capacitor Value	—	—	2.2	10	—	µF
C_{OUT}	Capacitance Connect to Output	—	—	2.2	10	—	µF
V_{OUT}	Output Voltage Range	—	V_{OUT}=2.85V, C_{OUT}=10µF, C_{IN}=10µF, L_{IN}=3.3µH, I_{LOAD}=1mA	- 5%	2.85	+ 5%	V
		—	V_{OUT}=3.0V, C_{OUT}=10µF, C_{IN}=10µF, L_{IN}=3.3µH, I_{LOAD}=1mA		3.0		
		—	V_{OUT}=3.15V, C_{OUT}=10µF, C_{IN}=10µF, L_{IN}=3.3µH, I_{LOAD}=1mA		3.15		
		—	V_{OUT}=3.3V, C_{OUT}=10µF, C_{IN}=10µF, L_{IN}=3.3µH, I_{LOAD}=1mA		3.3		
		—	V_{OUT}=5.0V, C_{OUT}=10µF, C_{IN}=10µF, L_{IN}=3.3µH, I_{LOAD}=1mA		5.0		
ΔV_{LOAD}	Output Load Regulation	—	V_{OUT}=3.0V, C_{OUT}=10µF, C_{IN}=10µF, L_{IN}=3.3µH, I_{LOAD}=1 to 20mA	-2	—	+2	%
		—	V_{OUT}=3.3V, C_{OUT}=10µF, C_{IN}=10µF, L_{IN}=3.3µH I_{LOAD}=1 to 18mA				
		—	V_{OUT}=5.0V, C_{OUT}=10µF, C_{IN}=10µF, L_{IN}=3.3µH I_{LOAD}=1 to 12mA				
P_{OUT}	Output Power	—	—	—	—	60	mW

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_Q	Quiescent Current	—	V_{OUT}=3.3V	—	33	50	μA
		—	V_{BAT}=1.8V, V_{OUT}=3.3V L_{IN}=3.3μH	—	80	120	μA
f_{DC-DC}	Clock Frequency	—	—	0.5	1.0	1.5	MHz
I_{DCLOAD}	Maximum DC Load Current During Startup	—	C_{OUT}=2.2μF/10μF, C_{IN}=10μF, L_{IN}=3.3μH	—	—	1	mA
t_{START}	DC to DC Startup Time	—	V_{BAT}=0.9V, V_{OUT}≥3.3V×0.95@V_{OUT}=3.3V, L_{IN}=3.3μH, C_{OUT}=10μF, C_{IN}=2.2μF/10μF, I_{LOAD}=1mA	—	—	2.5	ms
η	Efficiency	—	V_{BAT}=1.8V, V_{OUT}=3.3V, I_{LOAD}=18mA, L_{IN}=3.3μH, C_{OUT}=10μF, C_{IN}=10μF	—	80	—	%
I_{DCIL}	Minimum Inductor Current Limit	—	DCIL[1:0]=11B, V_{OUT}≤3.0V×0.9@V_{OUT}=3.0V, L_{IN}=3.3μH, C_{OUT}=10μF, C_{IN}=10μF	273	420	567	mA
			DCIL[1:0]=10B, V_{OUT}≤3.0V×0.9@V_{OUT}=3.0V, L_{IN}=3.3μH, C_{OUT}=10μF, C_{IN}=10μF	364	560	756	mA
			DCIL[1:0]=01B, V_{OUT}≤3.0V×0.9@V_{OUT}=3.0V, L_{IN}=3.3μH, C_{OUT}=10μF, C_{IN}=10μF	455	700	945	mA
			DCIL[1:0]=00B, V_{OUT}≤3.0V×0.9@V_{OUT}=3.0V, L_{IN}=3.3μH, C_{OUT}=10μF, C_{IN}=10μF	637	980	1323	mA

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_{VDD}	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
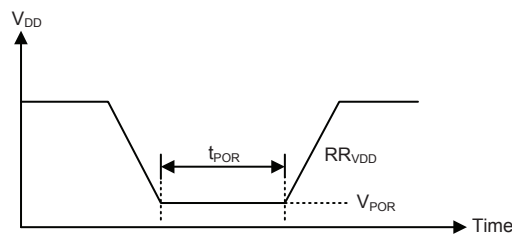
System Architecture

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 device 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.
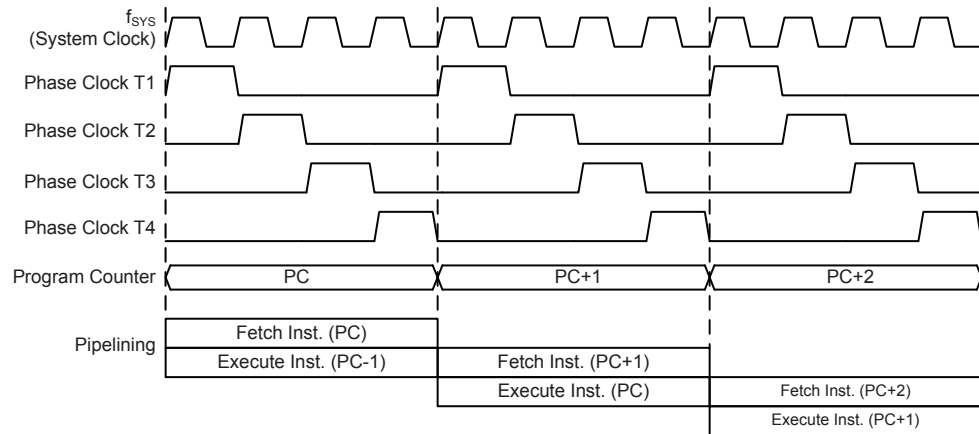
System Clock and Pipelining

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.

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

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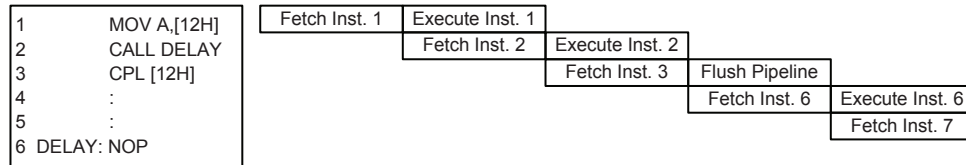
Instruction Fetching

Program Counter

During program execution, the Program Counter is used to keep track of the address of the next instruction to be executed. It is automatically incremented by one each time an instruction is executed except for instructions, such as "JMP" or "CALL" that 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
HT45F3520	PC9~PC8	PCL7~PCL0
HT45F3530	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.
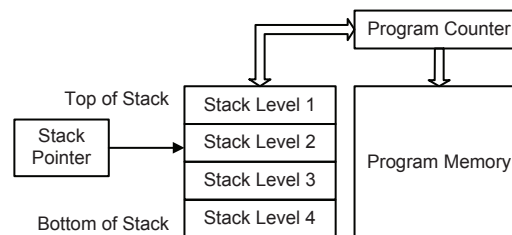
Arithmetic and Logic Unit – ALU

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 device offers users the flexibility to conveniently debug and develop their applications while also offering a means of field programming and updating.

Structure

The Program Memory has a capacity of 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
HT45F3520	1K×14
HT45F3530	2K×15
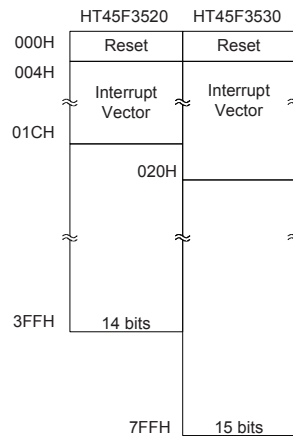
Program Memory Structure

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.
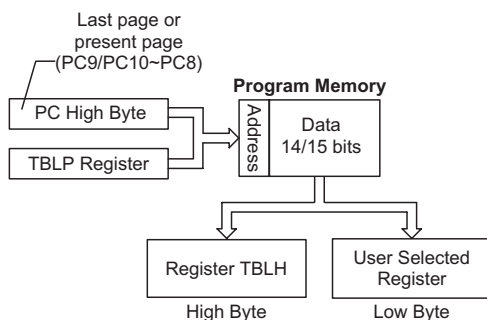
Table Program Example

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 "300H" which refers to the start address of the last page within the 1K words Program Memory of the HT45F3520. 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 "306H" 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 "306H" 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 "305H" transferred to tempreg2 and TBLH in this
                     ; example the data "1AH" is transferred to tempreg1 and data "0FH" to
                     ; register tempreg2
        :
        :
org 300h             ; sets initial address of program memory
dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
        :
        :
```

In Circuit Programming – ICP

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

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

Holtek Write Pins	MCU Programming Pins	Function
ICPDA	PA0	Programming Serial Data/Adress
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.

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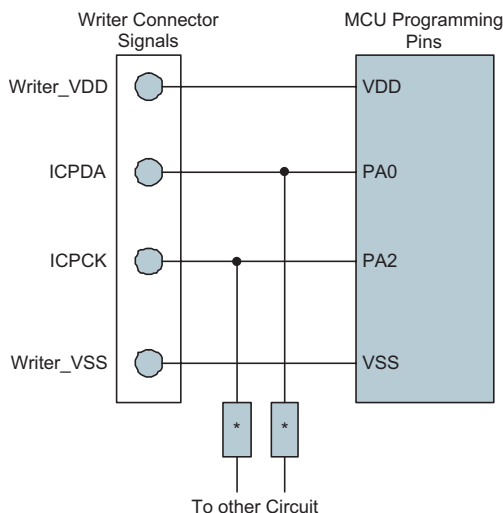
Note: * may be resistor or capacitor. The resistance of * must be greater than 1k or the capacitance of * must be less than 1nF.

On-Chip Debug Support – OCDS

There are two EV chips named HT45V3520 and HT45V3530 which are used to emulate the HT45F3520 and HT45F3530 repectively. Each EV chip device provides an "On-Chip Debug" function to debug the corresponding MCU device during the development process. The EV chip and the actual MCU device are almost functionally compatible except for the "On-Chip Debug" function. Users can use the EV chip device to emulate the real chip device behavior by connecting the OCDSDA and OCDSCK pins to the Holtek HT-IDE development tools. The OCDSDA pin is the OCDS Data/Address input/output pin while the OCDSCK pin is the OCDS clock input pin. When 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 chip. 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 chosen. 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
HT45F3520	Special Purpose	Bank0: 00H~3FH Bank1: 00H~40H (EEC in 40H)
	General Purpose: 64×8	Bank0: 40H~7FH
HT45F3530	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			Bank0 ┊ Bank1
00H	IAR0		20H	SADC0
01H	MP0		21H	SADC1
02H	IAR1		22H	SADC2
03H	MP1		23H	PAS0
04H	BP		24H	PAS1
05H	ACC		25H	PBS0
06H	PCL		26H	Unused
07H	TBLP		27H	Unused
08H	TBLH		28H	PB
09H	Unused		29H	PBC
0AH	STATUS		2AH	PBPU
0BH	SMOD		2BH	
0CH	Unused		┊	Unused
0DH	INTEG			
0EH	INTC0		2FH	
0FH	INTC1		30H	STMC0
10H	Unused		31H	STMC1
11H	MFI0		32H	STMDL
12H	MFI1		33H	STMDH
13H	Unused		34H	STMAL
14H	PA		35H	STMAH
15H	PAC		36H	PTMC0
16H	PAPU		37H	PTMC1
17H	PAWU		38H	PTMDL
18H	Unused		39H	PTMDH
19H	WDTC		3AH	PTMAL
1AH	TBC		3BH	PTMAH
1BH	SMOD1		3CH	PTMRPL
1CH	EEA		3DH	PTMRPH
1DH	EED		3EH	Unused
1EH	SADOL		3FH	DCC
1FH	SADOH		40H	Unused ┊ EEC

▨ : Unused, read as "00"

Special Purpose Data Memory Structure – HT45F3520

	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	MFI0	
12H	MFI1	
13H	Unused	
14H	PA	
15H	PAC	
16H	PAPU	
17H	PAWU	
18H	Unused	
19H	WDTC	
1AH	TBC	
1BH	SMOD1	
1CH	EEA	
1DH	EED	
1EH	SADOL	
1FH	SADOH	
20H	SADC0	
21H	SADC1	
22H	SADC2	

	Bank0	Bank1
23H	PAS0	
24H	PAS1	
25H	PCS0	
26H	PCS1	
27H	IFS0	
28H	PB	
29H	PBC	
2AH	PBPU	
2BH	PC	
2CH	PCC	
2DH	PCPU	
2EH	Unused	
2FH	Unused	
30H	STMC0	
31H	STMC1	
32H	STMDL	
33H	STMDH	
34H	STMAL	
35H	STMAH	
36H	PTMC0	
37H	PTMC1	
38H	PTMDL	
39H	PTMDH	
3AH	PTMAL	
3BH	PTMAH	
3CH	PTMRPL	
3DH	PTMRPH	
3EH	Unused	
3FH	DCC	
40H	Unused	EEC
41H	SCOMC	
42H	SLEDC0	
43H	SLEDC1	
44H ⋮ 7FH	Unused	

▒ : Unused, read as "00"

Special Purpose Data Memory Structure – HT45F3530

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 with Bank 0, all other Banks must be addressed indirectly using MP1 and IAR1. Note that for the HT45F3520 device, bit 7 of the Memory Pointers is not required to address the full memory space. When bit 7 of the Memory Pointers for the HT45F3520 is read, a value of "1" will be returned.

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 0~1.

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 indicate 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

The devices contain 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 varies from 32×8 to 64×8 bits, according to the device selected. 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	Address
HT45F3520	32×8	00H~1FH
HT45F3530	64×8	00H~3FH

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.

EEPROM Register List

- **HT45F3520**

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

- **HT45F3530**

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

EEA Register

- **HT45F3520**

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

- **HT45F3530**

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	Frequency
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.
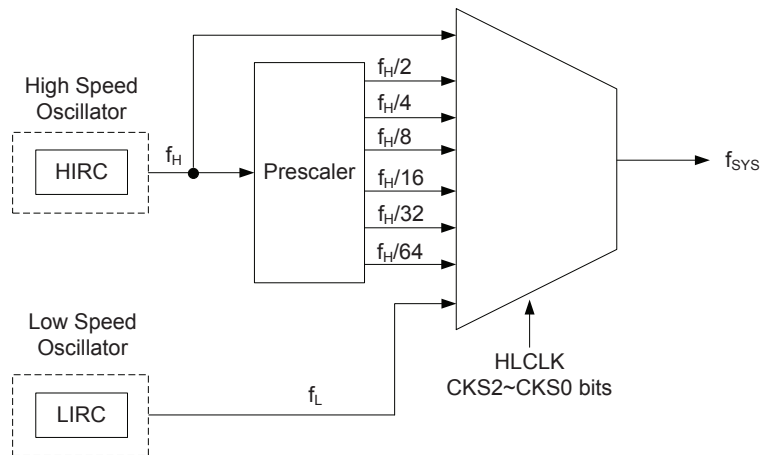
System Clock Configurations

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 $f_H/2 \sim f_H/64$.

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 TMs.
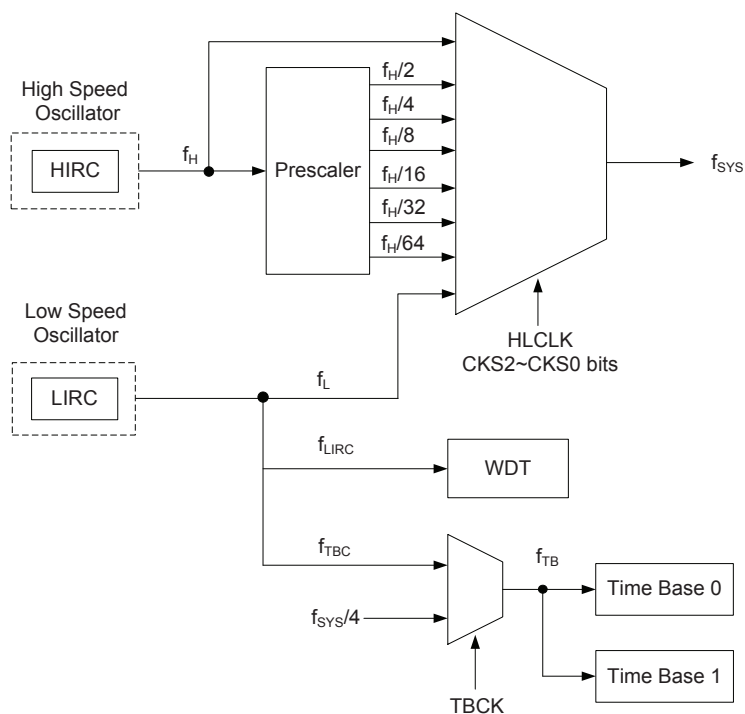
System Clock Configurations

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 $f_H \sim f_H/64$ 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	$f_H \sim f_H/64$	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 Register

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	1	1	0	—	0	0	1	0

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: $f_H/2 \sim f_H/64$ or f_L
    1: f_H

This bit is used to select if the f_H clock or the $f_H/2 \sim f_H/64$ 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 $f_H/2 \sim f_H/64$ 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
       Describe elsewhere

Bit 1   Unimplemented, read as "0"

Bit 0   **WRF**: WDT Control register software reset flag
       Describe 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 device enters the IDLE Mode or the SLEEP Mode is determined by the condition of the IDLEN bit in the SMOD register and FSYSON in the 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, $f_H/2 \sim f_H/64$ 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 TMs.
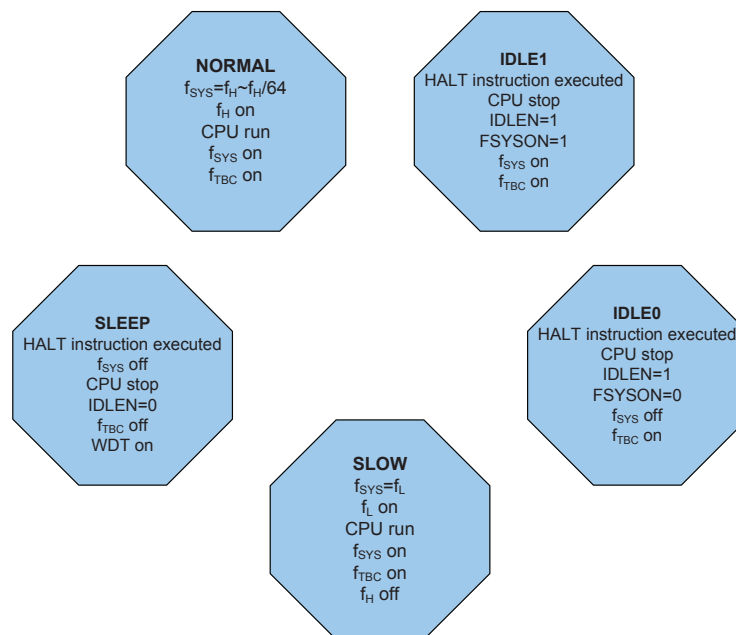
NORMAL Mode to SLOW Mode Switching

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.
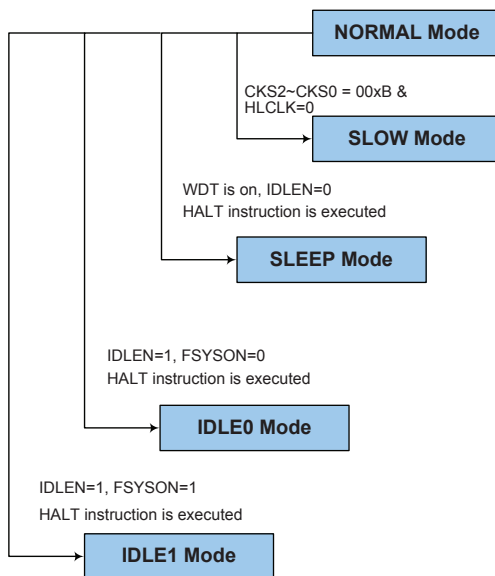
SLOW Mode to NORMAL Mode Switching

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.

SLOW Mode

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

NORMAL Mode

WDT is on, IDLEN=0
HALT instruction is executed

SLEEP Mode

IDLEN=1, FSYSON=0
HALT instruction is executed

IDLE0 Mode

IDLEN=1, FSYSON=1
HALT instruction is executed

IDLE1 Mode
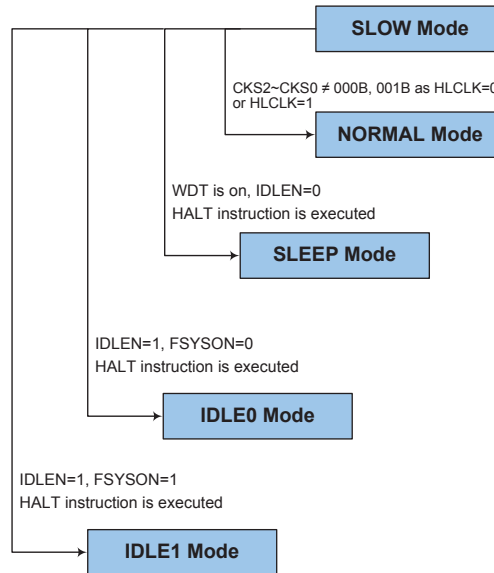
Entering the SLEEP Mode

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 device will not be immediately serviced, but will rather be serviced later when the related interrupt is finally enabled or when a stack level becomes free. The other situation is where the related interrupt is enabled and the stack is not full, in which case the regular interrupt response takes place. If an interrupt request flag is set high before entering the SLEEP or IDLE Mode, the wake-up function of the related interrupt will be disabled.

Watchdog Timer

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

Watchdog Timer Clock Source

The Watchdog Timer clock source is provided by the internal 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
           Describe elsewhere

Bit 6~3   Unimplemented, read as "0"

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

Describe 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 cleared by the application program. Note that this bit can only be cleared to 0 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 device. 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 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 32kHz LIRC oscillator as its source clock, this will give a maximum watchdog period of around 1 second for the 2^{15} division ratio, and a minimum timeout of 8ms for the 2^8 division ration.

Watchdog Timer
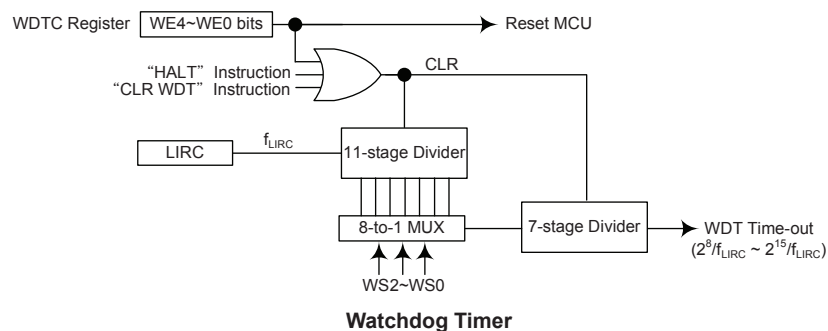
Reset and Initialisation

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

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

Reset Functions

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

Power-on Reset

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

Power-On Reset Timing Chart

Low Voltage Reset – LVR

The devices contain a low voltage reset circuit in order to monitor the supply voltage of the device and provide an MCU reset should the value fall below a certain predefined level. The LVR function 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 $0.9V \sim V_{LVR}$ such as might occur when changing the battery, the LVR will automatically reset the device internally and the LVRF bit in the SMOD1 register will also be set to 1. For a valid LVR signal, a low voltage, i.e., a voltage in the range between $0.9V \sim V_{LVR}$ must exist for greater than the value t_{LVR} specified in the LVR 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 devices after 2~3 LIRC clock cycles. Note that the LVR function will be automatically disabled when the devices enter the SLEEP/IDLE mode.
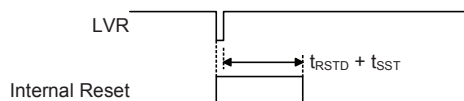
Low Voltage Reset Timing Chart

- **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
                       Describe 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
                       Describe 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".
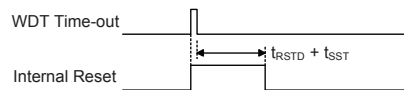
WDT Time-out Reset during Normal Operation Timing Chart

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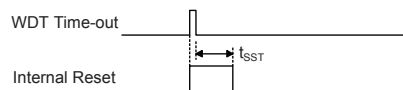
WDT Time-out Reset during SLEEP or IDLE Timing Chart

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 microcontroller are affected after a power-on reset occurs.

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

The different kinds of resets all affect the internal registers of the microcontroller in different ways. To ensure reliable continuation of normal program execution after a reset occurs, it is important to know what condition the microcontroller is in after a particular reset occurs. The following table describes how each type of reset affects each of the microcontroller internal registers. Note that where more than one package type exists the table will reflect the situation for the larger package type.

- **HT45F3520**

Register	Reset (Power On)	LVR Reset (Normal Operation)	WDT Time-out (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	--uu uuuu	--1u uuuu	--11 uuuu
SMOD	110- 0010	110- 0010	110- 0010	uuu- uuuu
INTEG	---- --00	---- --00	---- --00	---- --uu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI0	--00 --00	--00 --00	--00 --00	--uu --uu
MFI1	--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

Register	Reset (Power On)	LVR Reset (Normal Operation)	WDT Time-out (Normal Operation)	WDT Time-out (HALT)*
PAWU	0000 0000	0000 0000	0000 0000	uuuu 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	---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
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	---- 00--	---- 00--	---- 00--	---- uu--
PBS0	---- --00	---- --00	---- --00	---- --uu
PB	---1 1111	---1 1111	---1 1111	---u uuuu
PBC	---1 1111	---1 1111	---1 1111	---u uuuu
PBPU	---0 0000	---0 0000	---0 0000	---u 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
PTMC0	0000 0---	0000 0---	0000 0---	uuuu u---
PTMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDH	---- --00	---- --00	---- --00	---- --uu
PTMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMAH	---- --00	---- --00	---- --00	---- --uu
PTMRPL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMRPH	---- --00	---- --00	---- --00	---- --uu
DCC	--00 0000	--00 0000	--00 0000	--uu uuuu
EEC	---- 0000	---- 0000	---- 0000	---- uuuu

• **HT45F3530**

Register	Reset (Power On)	LVR Reset (Normal Operation)	WDT Time-out (Normal Operation)	WDT Time-out (HALT)*
MP0	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
MP1	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
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	--uu uuuu	--1u uuuu	--11 uuuu
SMOD	110- 0010	110- 0010	110- 0010	uuu- uuuu
INTEG	---- 0000	---- 0000	---- 0000	---- uuuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC2	---0 ---0	---0 ---0	---0 ---0	---u ---u
MFI0	--00 --00	--00 --00	--00 --00	--uu --uu
MFI1	--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
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
PCS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCS1	---- --00	---- --00	---- --00	---- --uu
IFS0	---- ---0	---- ---0	---- ---0	---- ---u
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	---1 1111	---1 1111	---1 1111	---u uuuu
PCC	---1 1111	---1 1111	---1 1111	---u uuuu
PCPU	---0 0000	---0 0000	---0 0000	---u uuuu
STMC0	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	Reset (Power On)	LVR Reset (Normal Operation)	WDT Time-out (Normal Operation)	WDT Time-out (HALT)*
STMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDH	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
STMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAH	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
PTMC0	0000 0 - - -	0000 0 - - -	0000 0 - - -	uuuu u - - -
PTMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDH	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
PTMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMAH	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
PTMRPL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMRPH	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - u u
DCC	- - 00 0000	- - 00 0000	- - 00 0000	- - uu uuuu
EEC	- - - - 0000	- - - - 0000	- - - - 0000	- - - - uuuu
SCOMC	- 000 - - - -	- 000 - - - -	- 000 - - - -	- uuu - - - -
SLEDC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SLEDC1	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0	- - - - - - 0 0

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.

I/O Control Register List

- **HT45F3520**

Register Name	Bit							
	7	6	5	4	3	2	1	0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PB	—	—	—	PB4	PB3	PB2	PB1	PB0
PBC	—	—	—	PBC4	PBC3	PBC2	PBC1	PBC0
PBPU	—	—	—	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0

- **HT45F3530**

Register Name	Bit							
	7	6	5	4	3	2	1	0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PB	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PBC	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PBPU	PBPU7	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PC	—	—	—	PC4	PC3	PC2	PC1	PC0
PCC	—	—	—	PCC4	PCC3	PCC2	PCC1	PCC0
PCPU	—	—	—	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0

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

• **HT45F3520**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
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   Port B bit 4 ~ bit 0 Pull-high Control
    0: Disable
    1: Enable

• **HT45F3530**

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

• **HT45F3530**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
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   Port C bit 4 ~ 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

- **HT45F3520**

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

Bit 7~5   Unimplemented, read as "0"

Bit 4~0   Port B bit 4 ~ bit 0 Input/Output Control
            0: Output
            1: Input

- **HT45F3530**

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

- **HT45F3530**

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

Bit 7~5   Unimplemented, read as "0"

Bit 4~0   Port C bit 4 ~ 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 includes Port A~C function selection register PxSn, which can select the desired functions of the multi-function pin-shared pins and another register IFS0, which can select the INT1 input function source pin for the HT45F3530.

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.

Pin-shared Function Selection Register List

- **HT45F3520**

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	—	—	—	—	PAS13	PAS12	—	—
PBS0	—	—	—	—	—	—	PBS01	PBS00

- **HT45F3530**

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
PCS0	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
PCS1	—	—	—	—	—	—	PCS11	PCS10
IFS0	—	—	—	—	—	—	—	IFS00

PAS0 Register

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: VREFI/AN3

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

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

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

PAS1 Register

- **HT45F3520**

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

Bit 7~4   Unimplemented, read as "0"

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

Bit 1~0   Unimplemented, read as "0"

- **HT45F3530**

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: PA7
    11: AN7

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

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

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

PBS0 Register

- **HT45F3520**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PBS01	PBS00
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2   Unimplemented, read as "0"

Bit 1~0   **PBS01~PBS00**: PB4 Pin-shared function selection
    00: PB4/PTPI
    01: PB4/PTPI
    10: PTP
    11: PB4/PTPI

PCS0 Register

• **HT45F3530**

Bit	7	6	5	4	3	2	1	0
Name	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6   **PCS07~PCS06**: PC3 Pin-shared function selection
     00: PC3
     01: PC3
     10: PC3
     11: SCOM3

Bit 5~4   **PCS05~PCS04**: PC2 Pin-shared function selection
     00: PC2
     01: PC2
     10: PC2
     11: SCOM2

Bit 3~2   **PCS03~PCS02**: PC1 Pin-shared function selection
     00: PC1
     01: PC1
     10: PC1
     11: SCOM1

Bit 1~0   **PCS01~PCS00**: PC0 Pin-shared function selection
     00: PC0
     01: PC0
     10: PC0
     11: SCOM0

PCS1 Register

• **HT45F3530**

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

Bit 7~2   Unimplemented, read as "0"

Bit 1~0   **PCS11~PCS10**: PC4 Pin-shared function selection
     00: PC4/PTPI
     01: PC4/PTPI
     10: PTP
     11: PC4/PTPI

IFS0 Register

• **HT45F3530**

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

Bit 7~1   Unimplemented, read as "0"

Bit 0   **IFS00**: INT1 input source selection
     0: PB0
     1: PA4

I/O Port Source Current Control – only for HT45F3530

The HT45F3530 device supports different source current driving capability for each I/O port except for PC4. 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~PBPS2**: 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~PBPS0**: 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~PAPS2**: 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~PAPS0**: 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.)

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.)

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
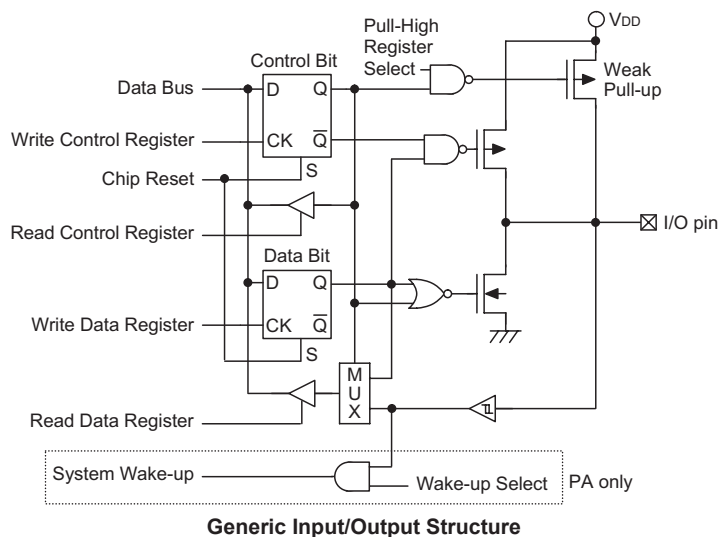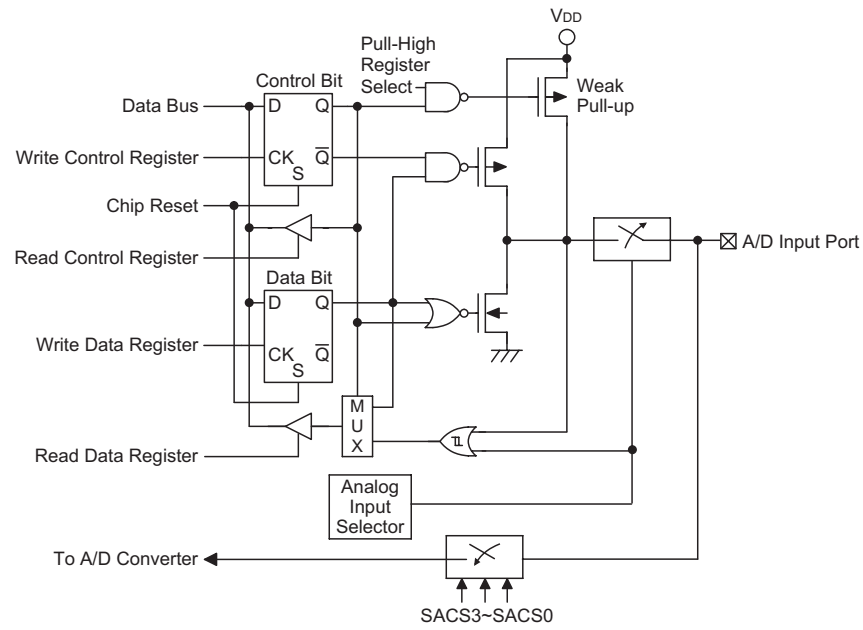
A/D Input/Output Structure

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.

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 and a Periodic Timer Module, abbreviated to the name STM and PTM respectively. The TM 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 common features of the different TM types are described here with more detailed information provided in the individual Standard and Periodic TM sections.

Introduction

Each device contains two TMs, one 10-bit Standard Type TM and one 10-bit Periodic Type TM. The main features of the STM and PTM are summarised in the accompanying table.

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

TM Function Summary

TM Operation

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

TM Clock Source

The clock source which drives the main counter in the TM can originate from various sources. The selection of the required clock source is implemented using the xTCK2~xTCK0 bits in the xTMC0 control registers, where "x" stands for S or P. 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 xTCK pin. The xTCK pin clock source is used to allow an external signal to drive the TM as an external clock source or for event counting.

TM Interrupts

The Standard and Periodic type TMs each 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

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

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

Each TM has one output pin with the label xTP. When the TM is in the Compare Match Output Mode, the xTP 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 xTP pin is also the output pin where the TM generates the PWM output waveform.

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

| STM		PTM	
Input	**Output**	**Input**	**Output**
STCK, STPI	STP	PTCK, PTPI	PTP

TM External Pins

TM Input/Output Pin Selection

Selecting to have a TM input/output or whether to retain its other shared function is implemented using the relevant pin-shared function selection registers, with the corresponding selection bits in each pin-shared function 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.
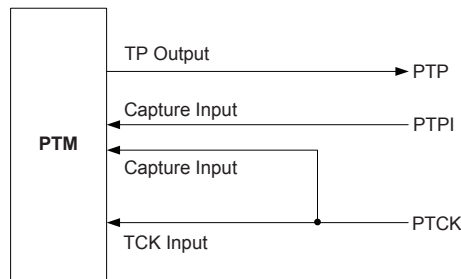

PTM Function Pin Control Block Diagram
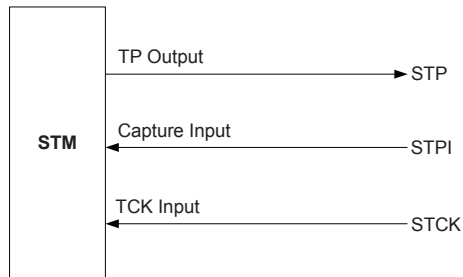
STM Function Pin Control Block Diagram

Programming Considerations

The TM Counter Registers, the Capture/Compare CCRA and the CCRP registers, being 10-bit, all have a low and high byte structure. The high bytes can be directly accessed, but as the low bytes can only be accessed via an internal 8-bit buffer, reading or writing to these register pairs must be carried out in a specific way. The important point to note is that data transfer to and from the 8-bit buffer and its related low byte only takes place when a write or read operation to its corresponding high byte is executed.

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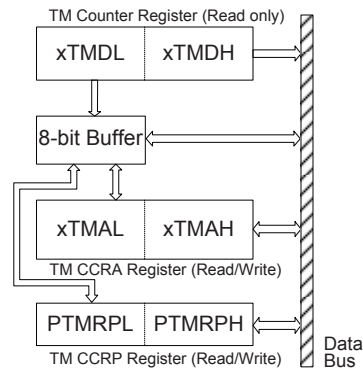

The following steps show the read and write procedures:

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

Standard Type TM – STM

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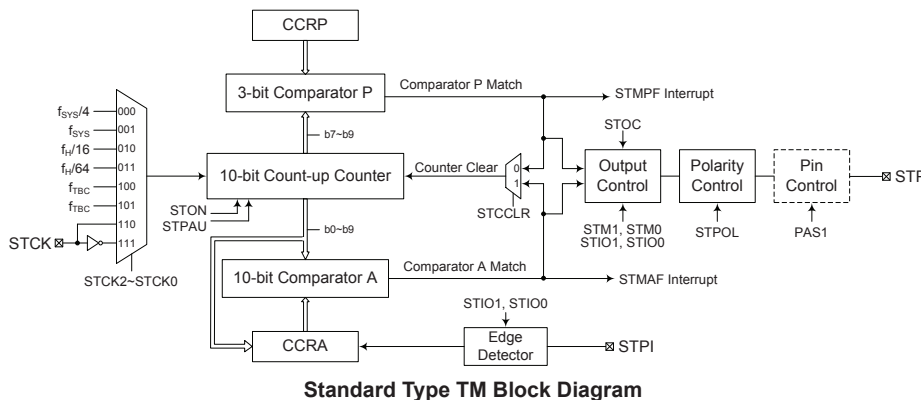
Standard Type TM Block Diagram

Standard 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 Register Description

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 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
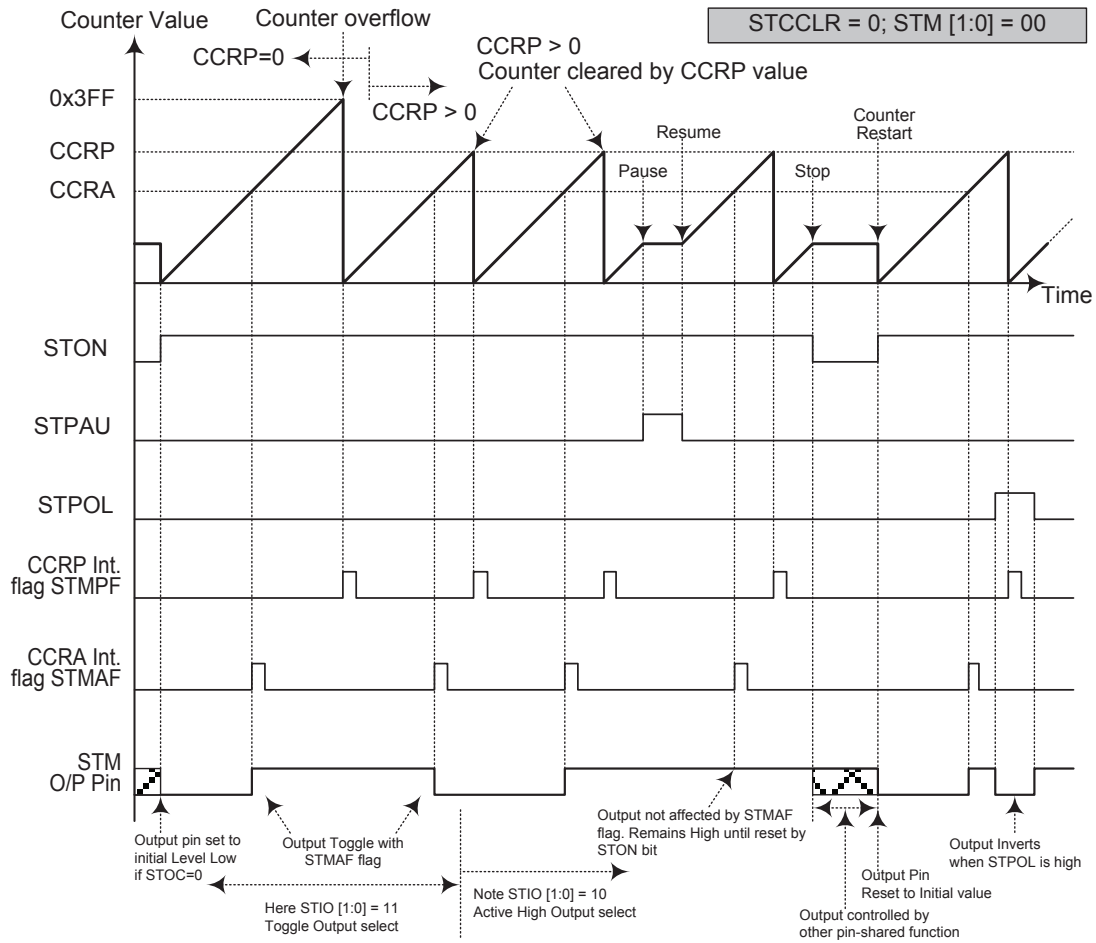
Standard Type TM Operating Modes

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.

STCCLR = 0; STM [1:0] = 00

Counter Value — Counter overflow

CCRP=0

CCRP > 0

CCRP > 0
Counter cleared by CCRP value

0x3FF

CCRP

CCRA

Resume

Counter Restart

Pause

Stop

Time

STON

STPAU

STPOL

CCRP Int. flag STMPF

CCRA Int. flag STMAF

STM O/P Pin

Output pin set to initial Level Low if STOC=0

Output Toggle with STMAF flag

Here STIO [1:0] = 11 Toggle Output select

Note STIO [1:0] = 10 Active High Output select

Output not affected by STMAF flag. Remains High until reset by STON bit

Output controlled by other pin-shared function

Output Pin Reset to Initial value

Output Inverts when STPOL is high

STM Compare Match Output – 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
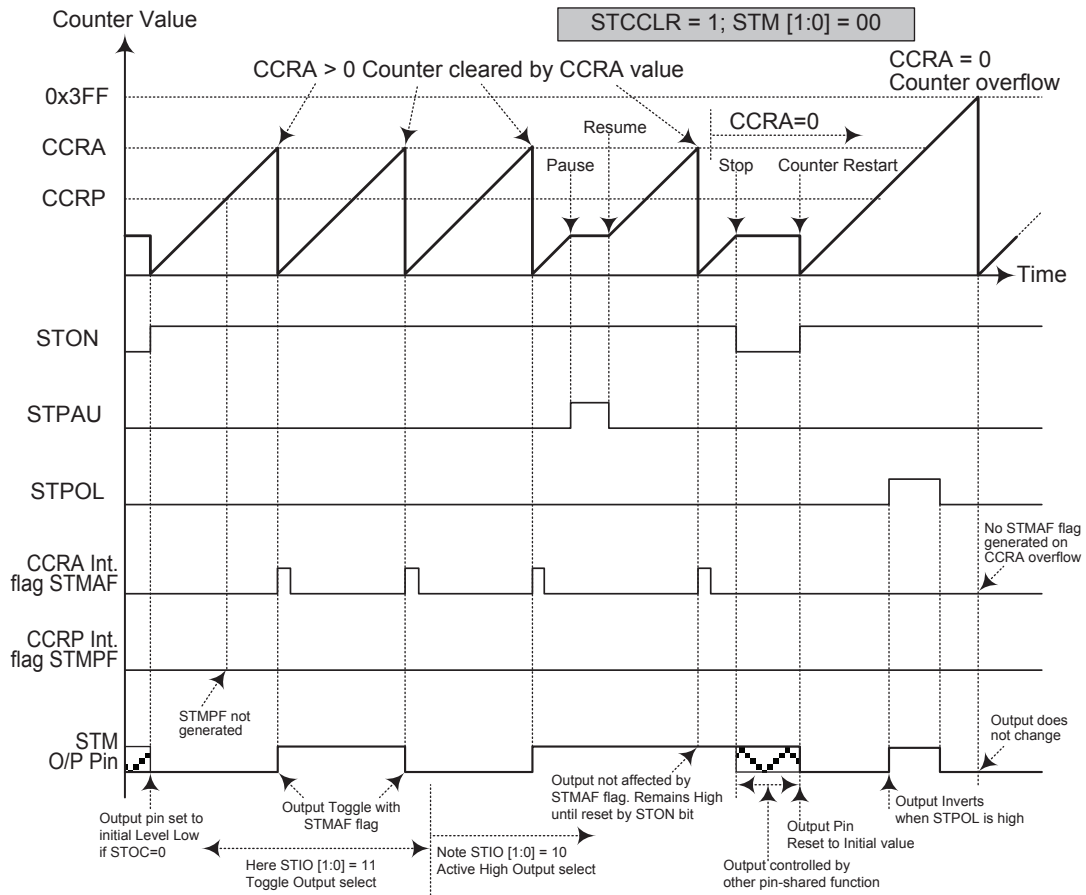
STM Compare Match Output – STCCLR=1

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.
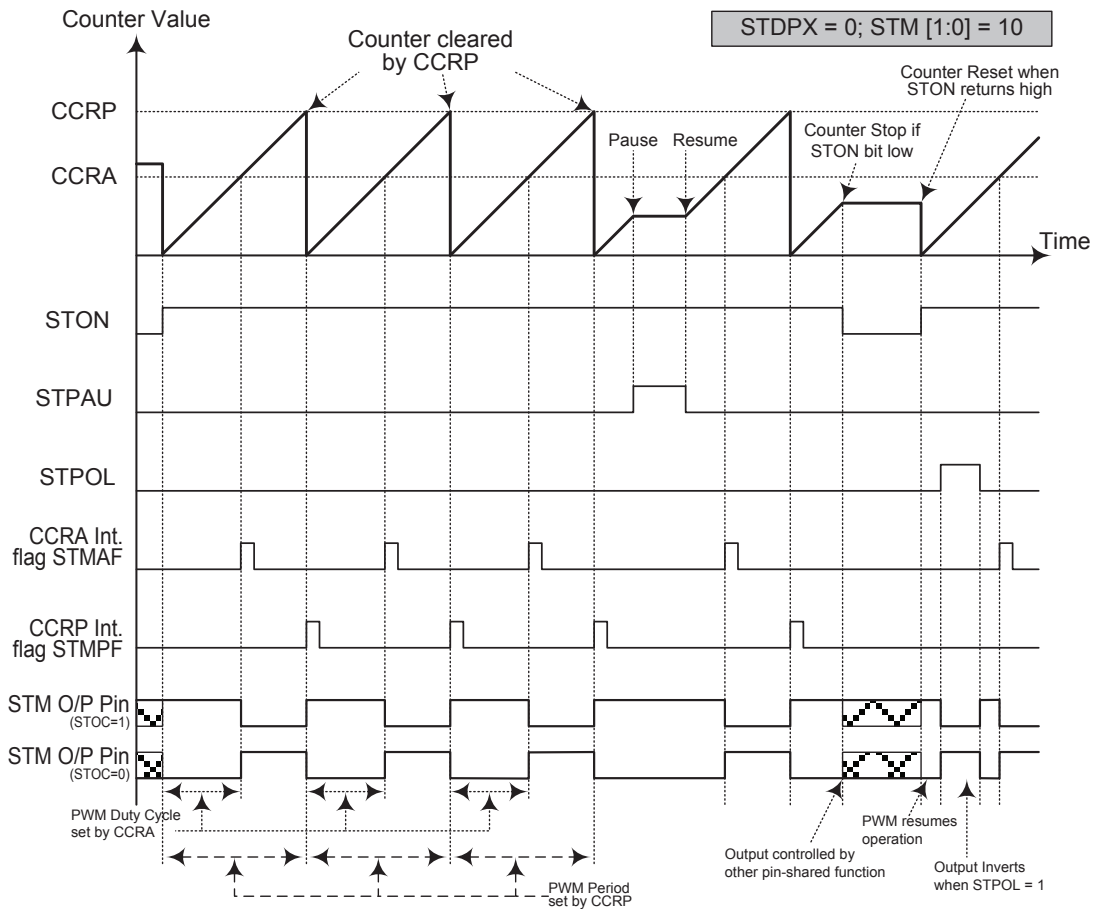
PWM Output Mode – STDPX=0

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
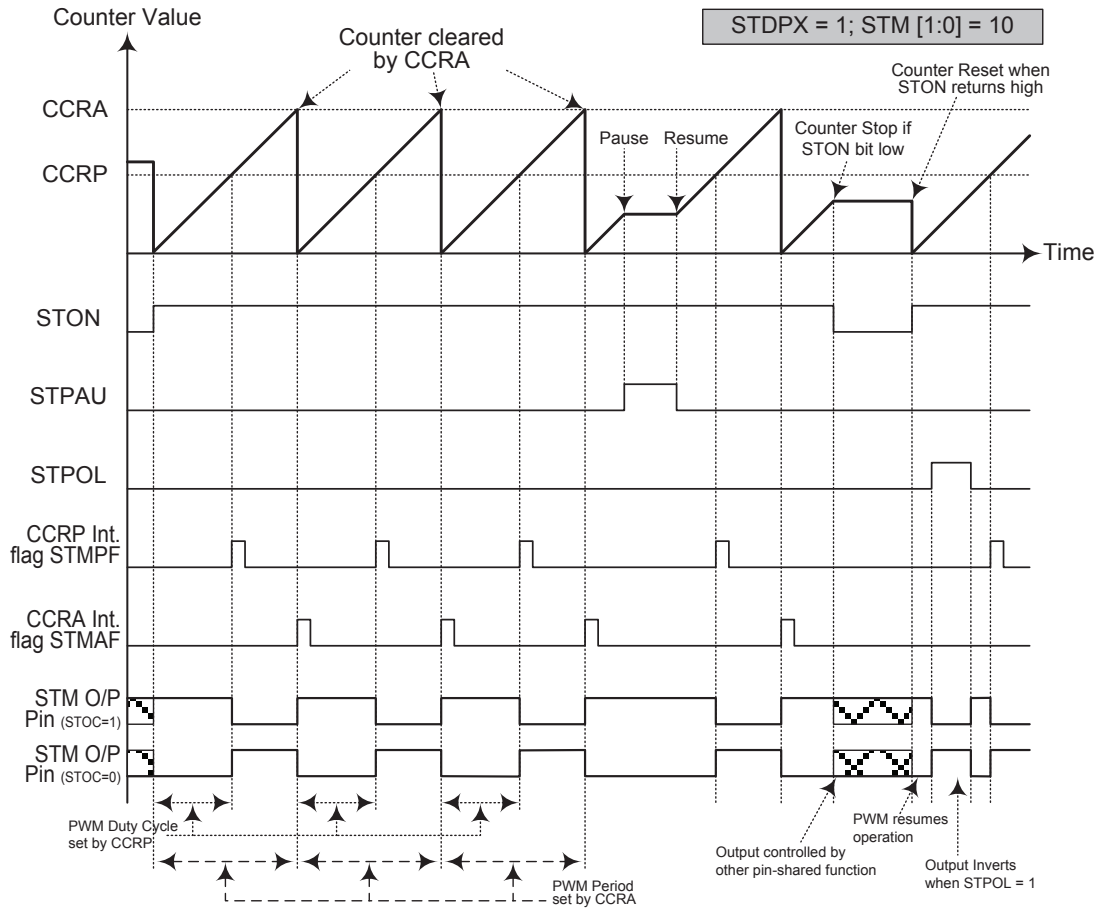
PWM Output Mode – STDPX=1

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.

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.


Single Pulse Generation
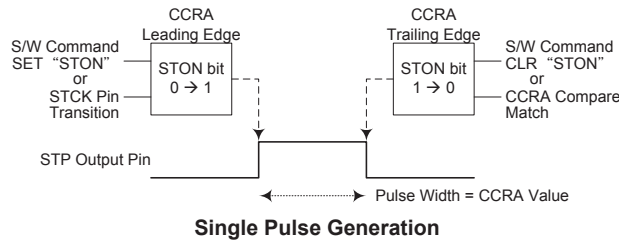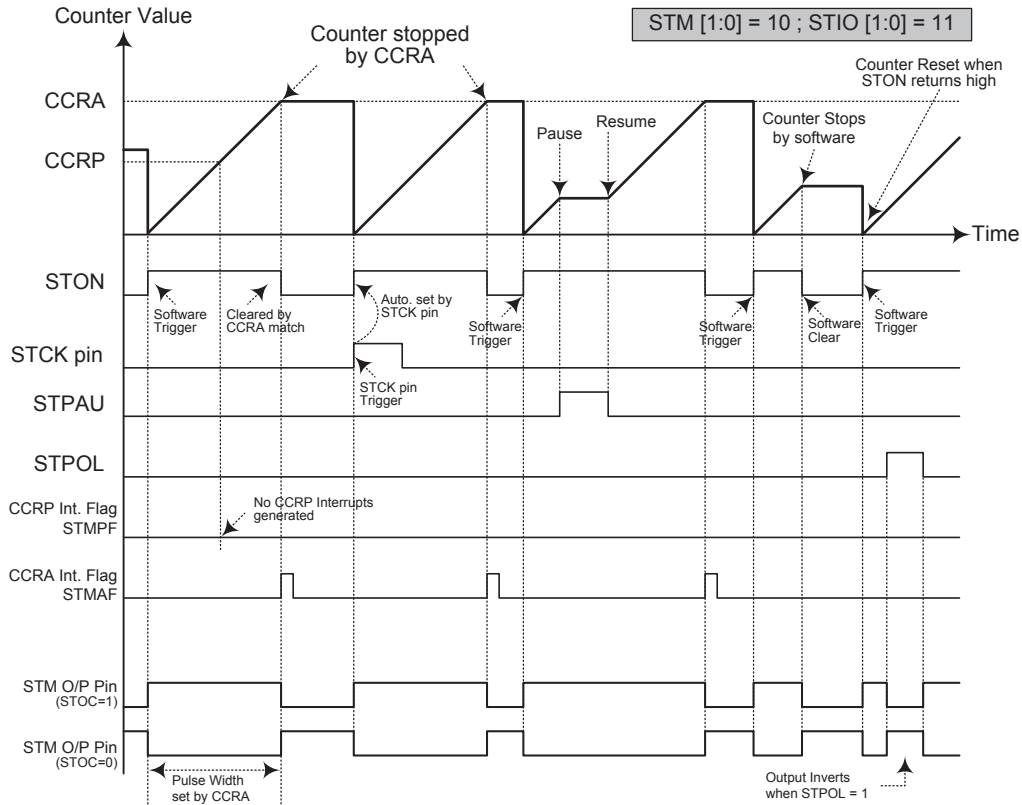
Single Pulse Mode

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.

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.
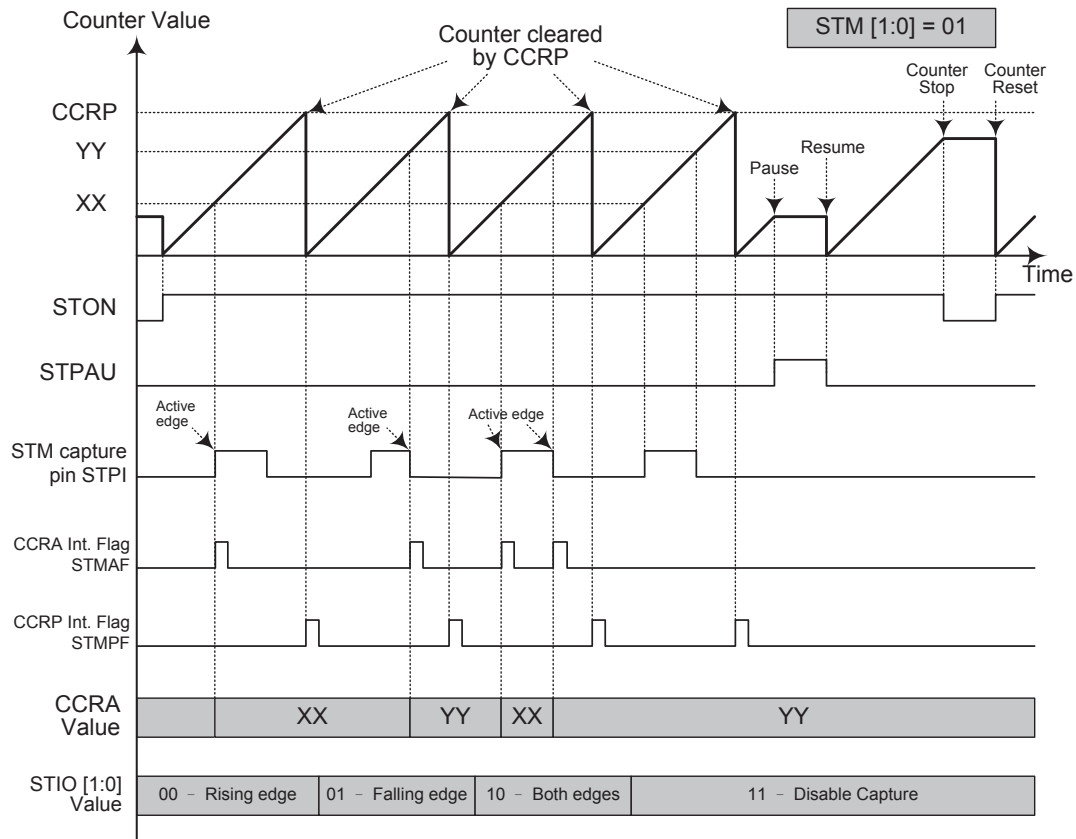
Capture Input Mode

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.

Periodic Type TM – PTM

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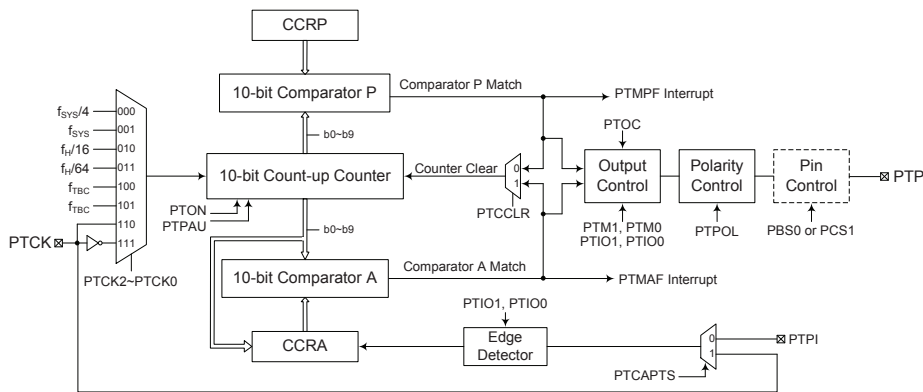
Periodic Type TM Block Diagram

Periodic TM Operation

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

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

Periodic Type TM Register Description

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

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

10-bit Periodic TM Register List

PTMC0 Register

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

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

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

Bit 6 ~ 4   **PTCK2~PTCK0**: Select PTM Counter clock
     000: $f_{SYS}/4$
     001: f_{SYS}
     010: $f_H/16$
     011: $f_H/64$
     100: f_{TBC}
     101: f_{TBC}
     110: PTCK rising edge clock
     111: PTCK falling edge clock

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

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

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

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

Bit 2 ~ 0   Unimplemented, read as "0"

PTMC1 Register

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

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

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

Bit 5 ~ 4   **PTIO1~PTIO0**: Select PTP output function

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

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

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

Timer/counter Mode:
  Unused

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

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

In the PWM Mode, the PTIO1 and PTIO0 bits determine how the PTM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to change the values of the PTIO1 and PTIO0 bits only after the PTM has been switched off. Unpredictable PWM outputs will occur if the PTIO1 and PTIO0 bits are changed when the PTM is running.

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

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

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

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

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

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

Bit 1   **PTCAPTS**: PTM capture trigger source select
         0: From PTPI
         1: From PTCK

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

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

PTMDL Register

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

Bit 7 ~ 0   PTM Counter Low Byte Register bit 7 ~ bit 0
                 PTM 10-bit Counter bit 7 ~ bit 0

PTMDH Register

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

Bit 7 ~ 2   Unimplemented, read as "0"

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

PTMAL Register

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

Bit 7 ~ 0   PTM CCRA Low Byte Register bit 7 ~ bit 0
                 PTM 10-bit CCRA bit 7 ~ bit 0

PTMAH Register

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

Bit 7 ~ 2   Unimplemented, read as "0"

Bit 1 ~ 0   PTM CCRA High Byte Register bit 1 ~ bit 0
                 PTM 10-bit CCRA bit 9 ~ bit 8

PTMRPL Register

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

Bit 7 ~ 0   PTM CCRP Low Byte Register bit 7 ~ bit 0
             PTM 10-bit CCRP bit 7 ~ bit 0

PTMRPH Register

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

Bit 7 ~ 2   Unimplemented, read as "0"
Bit 1 ~ 0   PTM CCRP High Byte Register bit 1 ~ bit 0
             PTM 10-bit CCRP bit 9 ~ bit 8

Periodic Type TM Operating Modes

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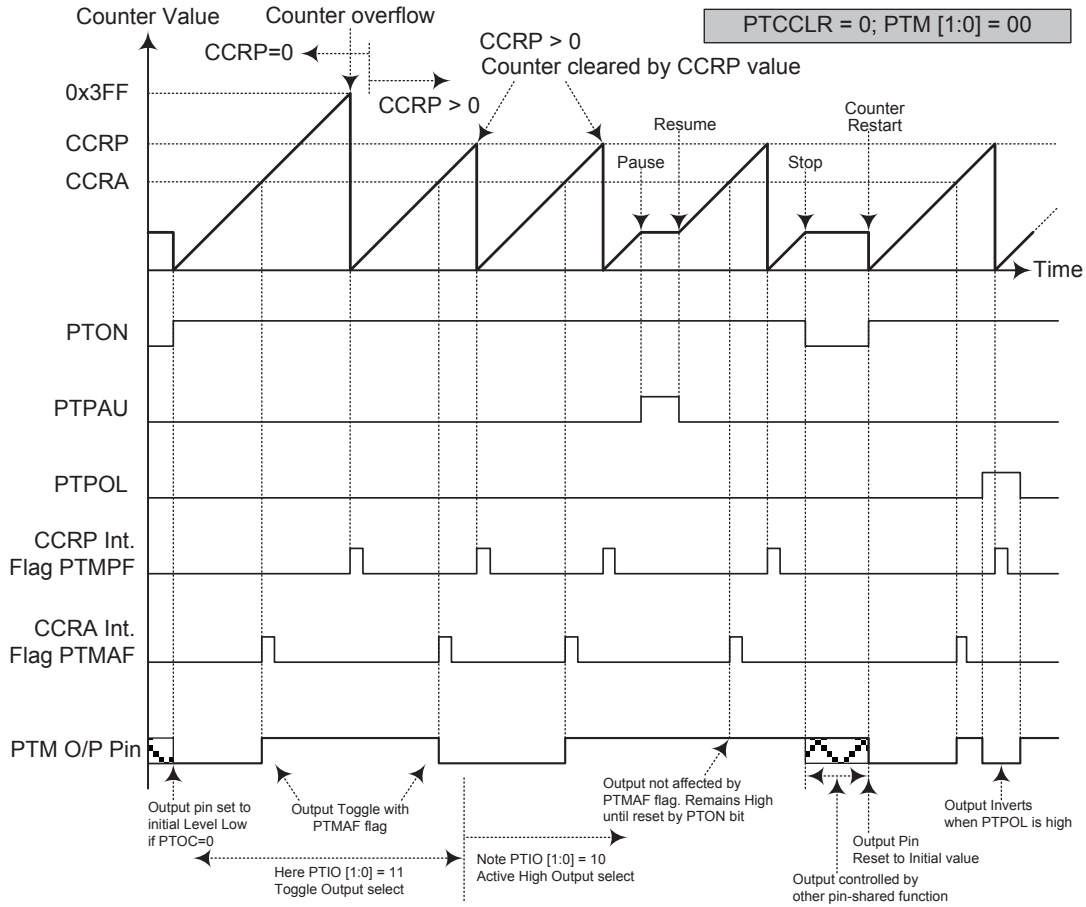
Compare Match Output Mode

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

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

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

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

Compare Match Output Mode – PTCCLR=0

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

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

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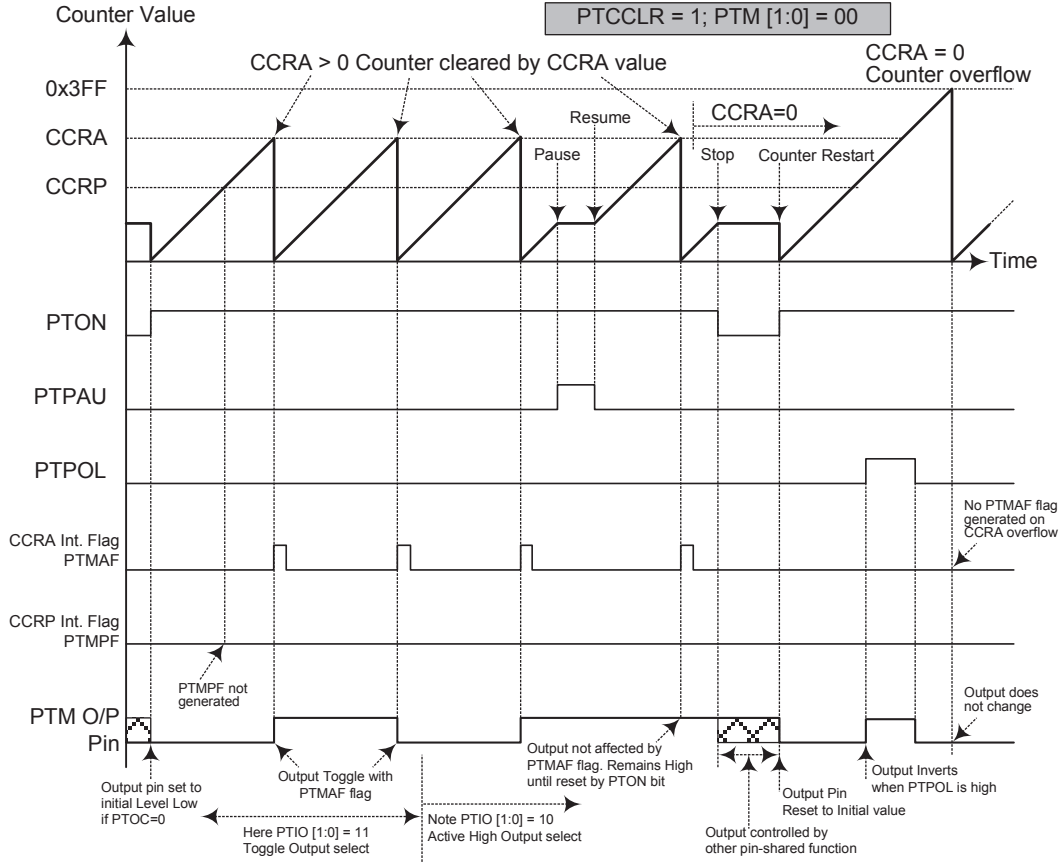
Compare Match Output Mode – PTCCLR=1

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

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

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

4. A PTMPF flag is not generated when PTCCLR=1

Timer/Counter Mode

To select this mode, bits PTM1 and PTM0 in the PTMC1 register should be set to 11 respectively. The Timer/Counter Mode operates in an identical way to the Compare Match Output Mode generating the same interrupt flags. The exception is that in the Timer/Counter Mode the PTM output pin is not used. Therefore the above description and Timing Diagrams for the Compare Match Output Mode can be used to understand its function.

PWM Output Mode

To select this mode, bits PTM1 and PTM0 in the PTMC1 register should be set to 10 respectively. The PWM function within the PTM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the PTM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

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

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

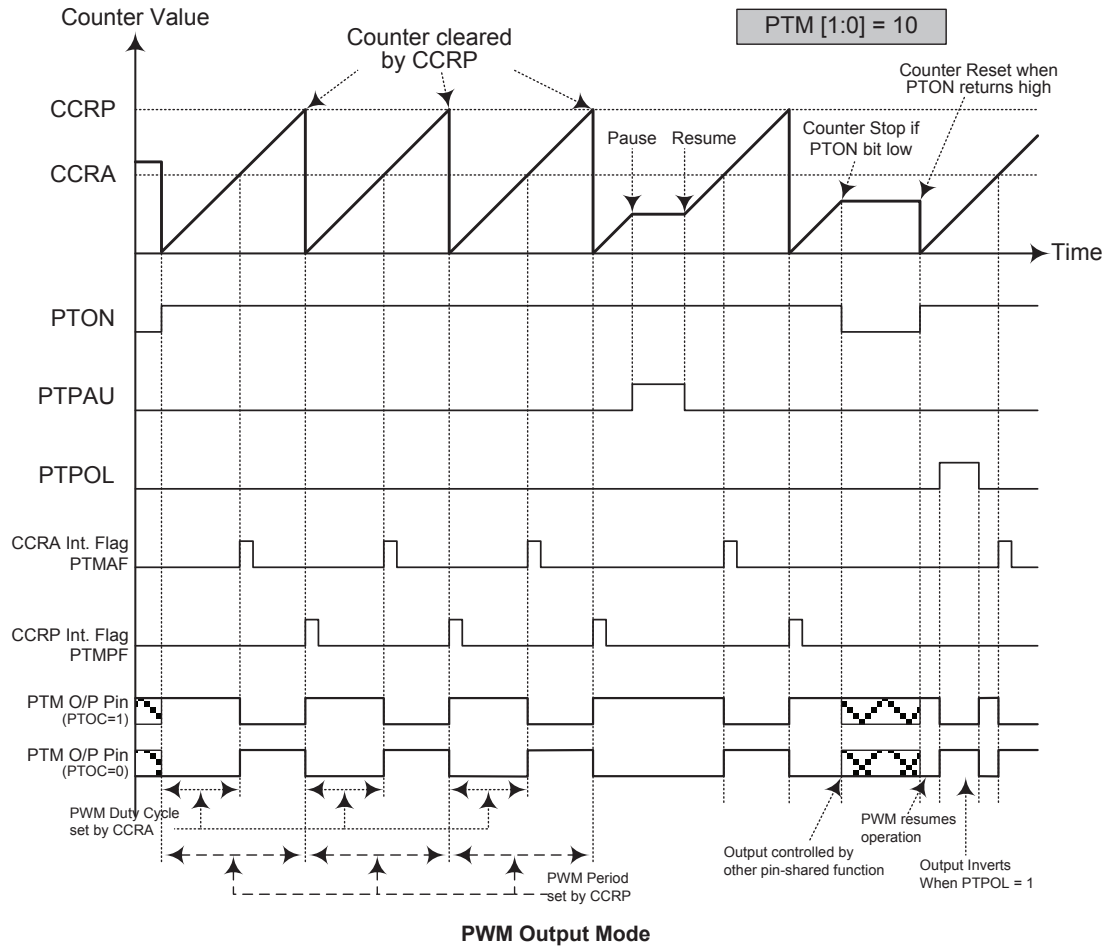- **10-bit PTM, PWM Mode, Edge-aligned Mode**

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

If f_{SYS} = 4MHz, PTMn clock source select $f_{SYS}/4$, CCRP = 512 and CCRA = 128,

The PTM 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%.

PWM Output Mode

Note: 1. A counter clear sets the PWM Period

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

3. The PTCCLR bit has no influence on PWM operation

Single Pulse Mode

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

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

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


Single Pulse Generation
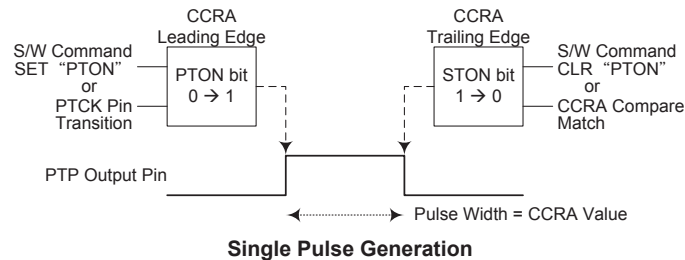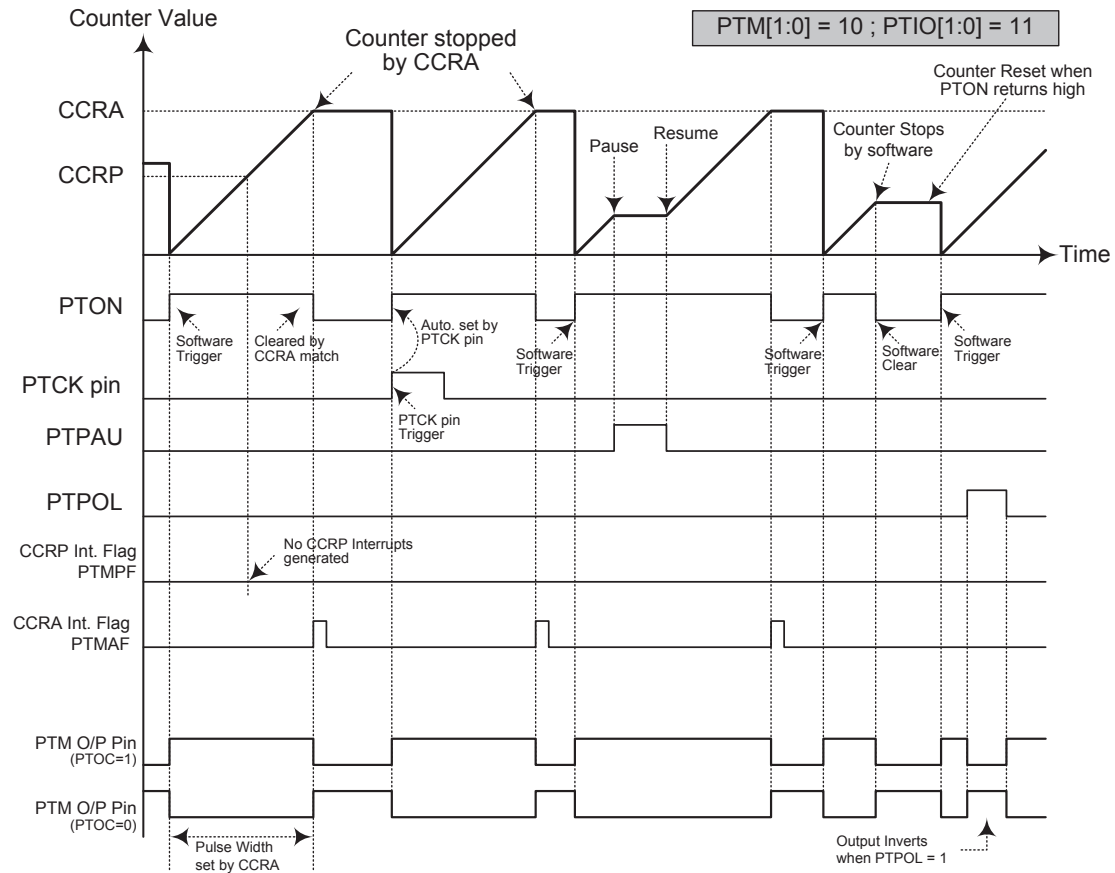
Single Pulse Mode

Note: 1. Counter stopped by CCRA

2. CCRP is not used

3. The pulse is triggered by the PTCK pin or by setting the PTON bit high

4. A PTCK pin active edge will automatically set the PTnON bit hight

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

Capture Input Mode

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

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

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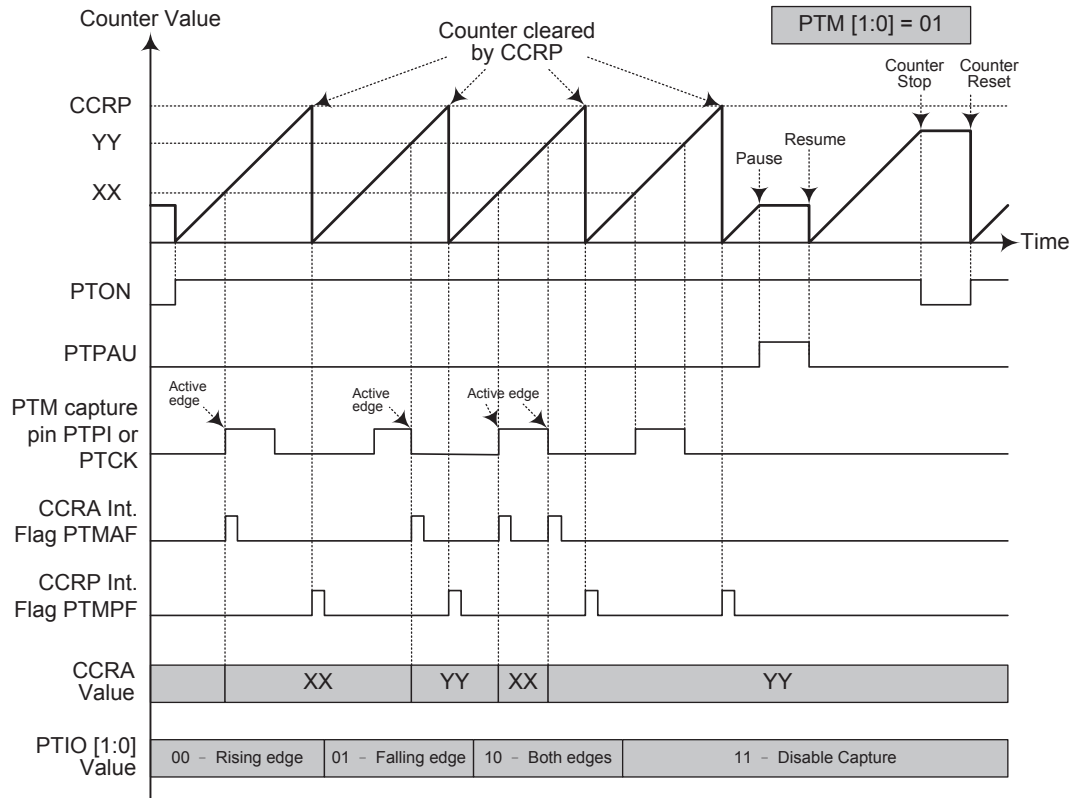
Capture Input Mode

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

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

3. PTCCLR bit not used

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

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

Analog to Digital Converter

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

A/D 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 SAINS and SACS bit fields. Note that when the internal analog signal is to be converted, the selected external input channel will be automatically disconnected to avoid malfunction. More detailed 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	A/D Channel Select Bits	Input Pins
HT45F3520	4	SAINS3~SAINS0, SACS3~SACS0	AN0~AN3
HT45F3530	8	SAINS3~SAINS0, SACS3~SACS0	AN0~AN7
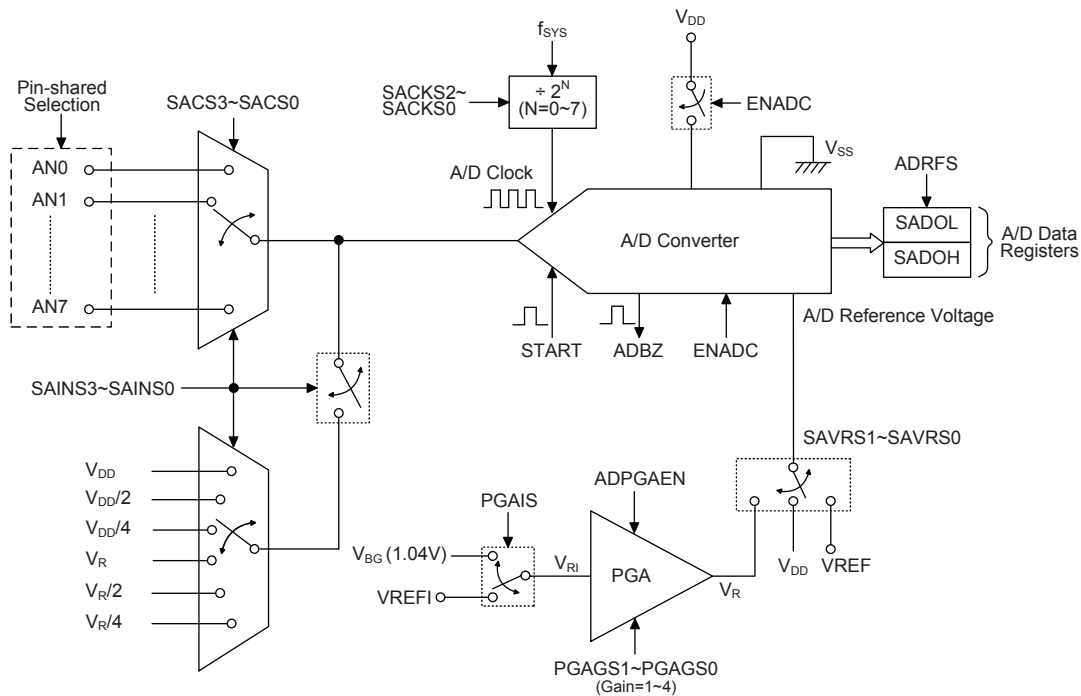
A/D Converter Structure

A/D Converter Register Description

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

Register Name	Bit							
	7	6	5	4	3	2	1	0
SADOL (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, PAS0, PAS1

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. 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.

Note that when the programs select external signal (AN0~AN7) and internal signal (V_{DD}, $V_{DD}/2$, $V_{DD}/4$, V_R, $V_R/2$ or $V_R/4$) as an ADC 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 on the VREF pin 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.

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.

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
1000~1011	xxxx	Ground	Unused

A/D Converter Input Signal Selection

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 register
    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 à SADOH=D[11:4]; SADOL=D[3:0]

       1: ADC output data format à 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

HT45F3520:

     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

HT45F3530:

     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 bit

     0000: ADC input only comes from external pin

     0001: ADC input only comes from V_{DD}

     0010: ADC input only comes from $V_{DD}/2$

     0011: ADC input only comes from $V_{DD}/4$

     0100: ADC input only comes from external pin

     0101: ADC input only comes from V_R

     0110: ADC input only comes from $V_R/2$

     0111: ADC input only comes from $V_R/4$

     1000: ADC input grounding (unused)

     1001: ADC input grounding (unused)

     1010: ADC input grounding (unused)

     1011: ADC input grounding (unused)

     Other Values: ADC input only comes from external pin

Note: V_R is PGA output voltage.

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 zero to conserve the power.

Bit 6~5   Unimplemented, read as "0"

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

Bit 3~2   **SAVRS1~SAVRS0**: ADC reference voltage selection
     00: ADC reference voltage only comes from V_{DD}
     01: ADC reference voltage only comes from VREF pin
     1x: ADC reference voltage only 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 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$_R$ 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 Pins

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.

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 SAVRS1~SAVRS0 bits 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.

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
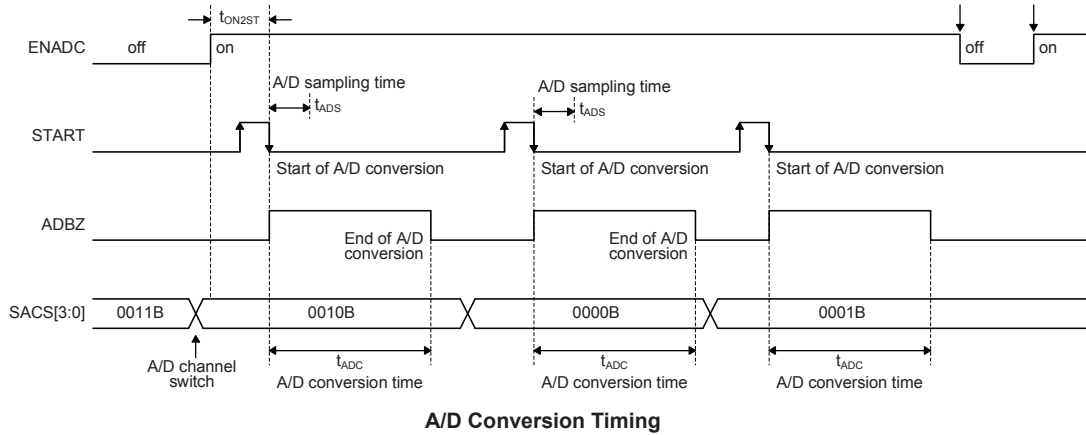
Summary of A/D Conversion Steps

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 clock by properly programming the SACKS2~SACKS0 bits in the SDAC1 register.

- Step 2

  Enable the A/D converter by setting the ENADC bit in the SADC0 register to 1.

- Step 3

  Select which signal is to be connected to the internal A/D converter by correctly configuring the SAINS3~SAINS0 bits.
  Select the external channel input to be coverted, go to Step 4.
  Select the internal analog signal to be converted, go to Step 5.

- Step 4

  If the A/D input signal comes from the external channel input selecting by configuring the SAINS bit field, the corresponding pin should first be configured as A/D input function by configuring the relevant pin-shared control bits. The desired analog channel then should be selected by configuring the SACS bit field. After this step, go to Step 6.

- Step 5

  If the A/D input signal comes from the internal analog signal, the SAINS bit field should be properly configured and then the external channel input will automatically be disconnected regardless of the SACS bit field value. After this step, go to Step 6.

- Step 6

  Select the reference voltage source by configuring the SAVRS1~SAVRS0 bits.

- Step 7

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

- Step 8

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

- Step 9

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

- Step 10

  If A/D conversion is in progress, the ADBZ flag will be set high. After A/D conversion process is completed, the ADBZ flag will go low and then output data can be read from the SADOH and SADOL registers. If the ADC interrupt is enabled and the stack is not full, data can be acquired by interrupt service program.

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

Programming Considerations

During microcontroller operations where the A/D converter is not being used, the A/D internal circuitry can be switched off to reduce power consumption, by clearing the 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 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 V_{REF} voltage, where V_{REF} stands for the selected A/D reference which can be V_{DD} or VREF pin voltage or V_R, this gives a single bit analog input value of V_{REF} divided by 4096.

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

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

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

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

Ideal A/D Transfer Function
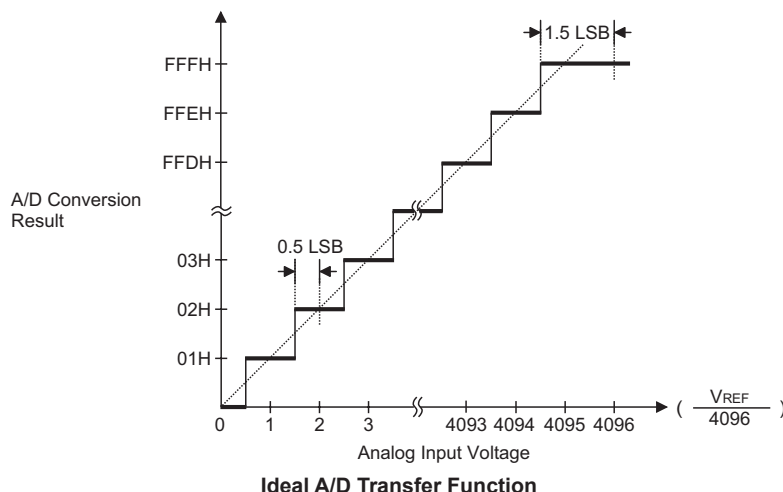
A/D Programming Examples

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

- **Example: using an EOCB polling method to detect the end of conversion**

```
clr  ADE              ; disable ADC interrupt
mov  a,03H
mov  SADC1,a          ; select f_SYS/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 f_SYS/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
```

SCOM Function for LCD – Only for HT45F3530

The HT45F3530 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
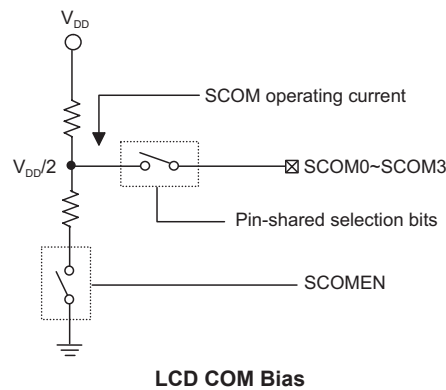
LCD Bias Current Control

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

Bit 3~0   Unimplemented, read as "0"

DC/DC Converter

The devices contain a DC/DC Converter to provide the power supply for the microcontroller. The DC/DC converter is a switching boost converter. It requires only two external components to provide an output voltage whose value is selected by configuring the DC/DC output voltage selection bits, DCVS2~DCVS0, in the DC/DC converter control register, named DCC. Since the DC/DC converter has a boost type architecture, then as long as $V_{BAT}<V_{OUT}+0.2V$, regulation will be maintained and V_{OUT} will remain at its selected voltage. However should $V_{BAT}>V_{OUT}+0.2V$, then regulation will cease and V_{OUT} will vary with the input voltage. The DC/DC converter voltage output is connected to an external VOUT pin as well as being connected to the VDD pin.
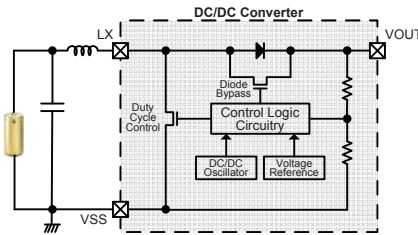
DC/DC Converter Block Diagram

DC/DC Converter Register

The DC/DC Converter function is controlled using a single register with the name DCC. Three bits in this register, DCVS2~DCVS0, are used to select one of eight DC/DC output voltages. The DCDIS bit is used to control the overall on/off function of the DC/DC Converter. Setting the bit high will disable the DC/DC Converter while clearing the bit to zero will enable DC/DC Converter. Two bits in this register, DCIL1~DCIL0, are used to select the inductor current limit.

DCC Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	DCIL1	DCIL0	DCDIS	DCVS2	DCVS1	DCVS0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6   Unimplemented, read as "0"

Bit 5~4   **DCIL1~DCIL0**: DC/DC converter inductor current limit select
    00: 980mA
    01: 700mA
    10: 560mA
    11: 420mA

Bit 3   **DCDIS**: DC/DC converter disable control
    0: Enable
    1: Disable

This bit is only vailable for use after the HALT instruction has been executed. The DC/DC converter will be always enabled regardless of the DCDIS bit value if the microcontroller is operating normally and the HALT instruction has not been executed.

If the DCDIS bit is set to 1 and the microcontroller executes the HALT instruction, then the DC/DC converter will be switched off in the power down mode. After the device is woken up, the DC/DC converter will be switched on.

Bit 2~0   **DCVS2~DCVS0**: Select DC/DC converter output voltage
      000: 3.3V
      001: 2.85V
      010: 3.0V
      011: 3.15V
      100: 3.3V
      101: 3.3V
      110: 3.3V
      111: 5.0V
Note: When the input voltage is greater than the output voltage, the output voltage maybe greater than the selected output voltage.

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 TMs, Time Base, EEPROM 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 MFI0~MFI1 registers which setup 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	—	—
INTn Pin	INTnE	INTnF	n=0 or 1, (n=1 for HT45F3530 only)
Multi-function	MFnE	MFnF	n=0 or 1
A/D Converter	ADE	ADF	—
Time Base	TBnE	TBnF	n=0 or 1
EEPROM	DEE	DEF	—
TM	STMAE	STMAF	—
	STMPE	STMPF	
	PTMAF	PTMAE	
	PTMPF	PTMPE	

Interrupt Register Bit Naming Conventions

Interrupt Register List

- **HT45F3520**

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	—	—	INT0S1	INT0S0
INTC0	—	MF1F	MF0F	INT0F	MF1E	MF0E	INT0E	EMI
INTC1	TB1F	TB0F	DEF	ADF	TB1E	TB0E	DEE	ADE
MFI0	—	—	STMAF	STMPF	—	—	STMAE	STMPE
MFI1	—	—	PTMAF	PTMPF	—	—	PTMAE	PTMPE

- **HT45F3530**

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	MF1F	MF0F	INT0F	MF1E	MF0E	INT0E	EMI
INTC1	TB1F	TB0F	DEF	ADF	TB1E	TB0E	DEE	ADE
INTC2	—	—	—	INT1F	—	—	—	INT1E
MFI0	—	—	STMAF	STMPF	—	—	STMAE	STMPE
MFI1	—	—	PTMAF	PTMPF	—	—	PTMAE	PTMPE

INTEG Register

- **HT45F3520**

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

- **HT45F3530**

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	—	MF1F	MF0F	INT0F	MF1E	MF0E	INT0E	EMI
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7   Unimplemented, read as "0"

Bit 6   **MF1F**: Multi-function Interrupt 1 Request Flag
        0: No request
        1: Interrupt request

Bit 5   **MF0F**: Multi-function Interrupt 0 Request Flag
        0: No request
        1: Interrupt request

Bit 4   **INT0F**: External Interrupt 0 Request Flag
        0: No request
        1: Interrupt request

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

Bit 2   **MF0E**: Multi-function Interrupt 0 Control
        0: Disable
        1: Enable

Bit 1   **INT0E**: External Interrupt 0 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	TB1F	TB0F	DEF	ADF	TB1E	TB0E	DEE	ADE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7   **TB1F**: Time Base 1 Interrupt Request Flag
        0: No request
        1: Interrupt request

Bit 6   **TB0F**: Time Base 0 Interrupt Request Flag
        0: No request
        1: Interrupt request

Bit 5   **DEF**: Data EEPROM Interrupt Request Flag
        0: No request
        1: Interrupt request

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

Bit 3   **TB1E**: Time Base 1 Interrupt Control
        0: Disable
        1: Enable

Bit 2   **TB0E**: Time Base 0 Interrupt Control
        0: Disable
        1: Enable

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

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

INTC2 Register

• **HT45F3530**

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

Bit 7~5   Unimplemented, read as "0"

Bit 4   **INT1F**: External Interrupt 1 Request Flag
    0: No request
    1: Interrupt request

Bit 3~1   Unimplemented, read as "0"

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

MFI0 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

MFI1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	PTMAF	PTMPF	—	—	PTMAE	PTMPE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7 ~ 6   Unimplemented, read as "0"
Bit 5   **PTMAF**: PTM Comparator A Interrupt Request Flag
    0: No request
    1: Interrupt request

Bit 4   **PTMPF**: PTM Comparator P Interrupt Request Flag
                  0: No request
                  1: Interrupt request

Bit 3 ~2   Unimplemented, read as "0"

Bit 1   **PTMAE**: PTM Comparator A Interrupt Control
                  0: Disable
                  1: Enable

Bit 0   **PTMPE**: PTM 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
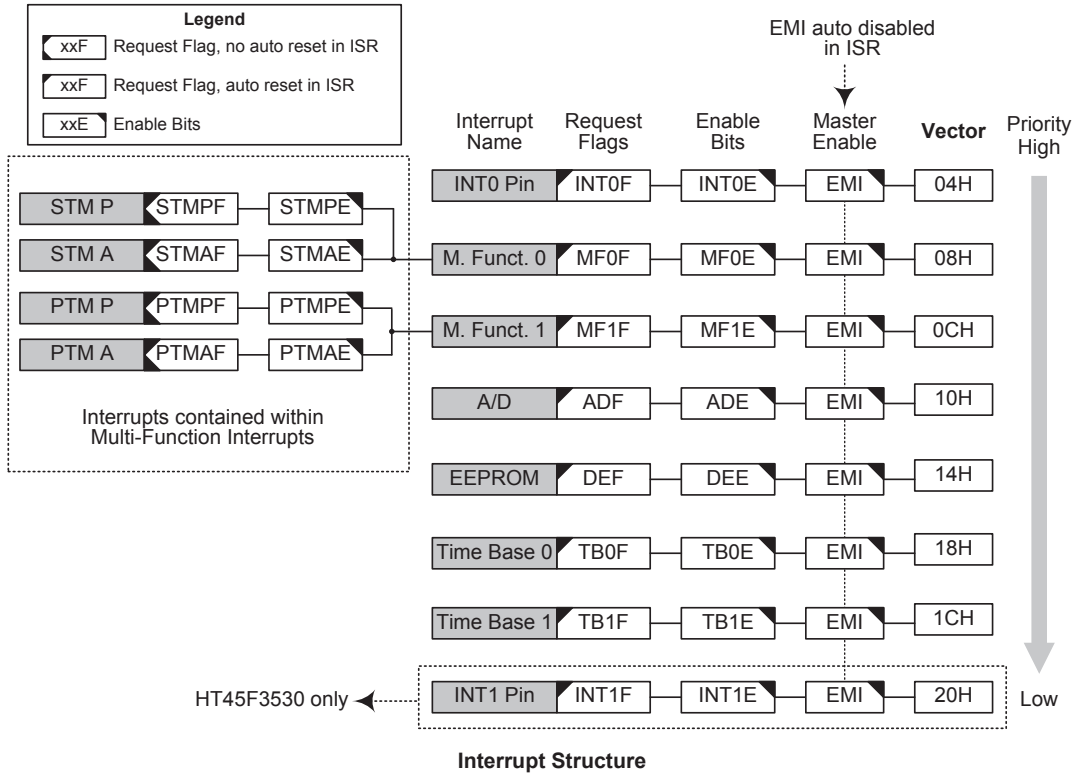
External Interrupt

The external interrupt is controlled by signal transitions on the pin INT0 or INT1. For the HT45F3530 device, the 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.

Multi-function Interrupt

Within the devices there are two Multi-function interrupts. Unlike the other independent interrupts, these interrupts have no independent source, but rather are formed from other existing interrupt sources, namely the STM/PTM CCRA and CCRP interrupts.

A Multi-function interrupt request will take place when the Multi-function interrupt request flag, MFnF is set. The Multi-function interrupt flag will be set when any of their included functions generate an interrupt request flag. 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/PTM 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: $4096/f_{TB}$
        01: $8192/f_{TB}$
        10: $16384/f_{TB}$
        11: $32768/f_{TB}$

Bit 3   Unimplemented, read as "0"

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

Time Base Interrupt
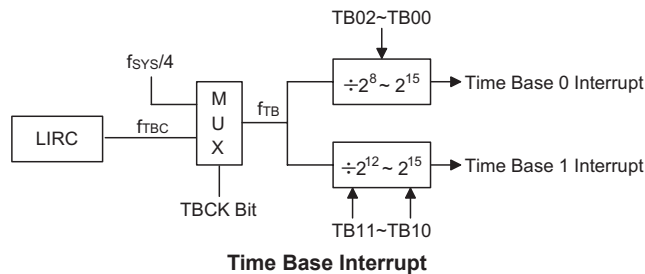
EEPROM Interrupt

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 and PTM have two interrupts which are both contained within the Multi-function Interrupt. For each TM there are two interrupt request flags xTMPF and xTMAF and two enable bits xTMPE and xTMAE, where "x" stands for S or P. 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 the associated Multi-function interrupt enable bit, MFnE, must first be set. When the interrupt is enabled, the stack is not full and a TM comparator match situation occurs, a subroutine call to the relevant 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 MFnF flag will be automatically cleared. As the TM interrupt request flags will not be automatically cleared, they have to be cleared by the application program.

Interrupt Wake-up Function

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

Programming Considerations

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

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

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

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

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

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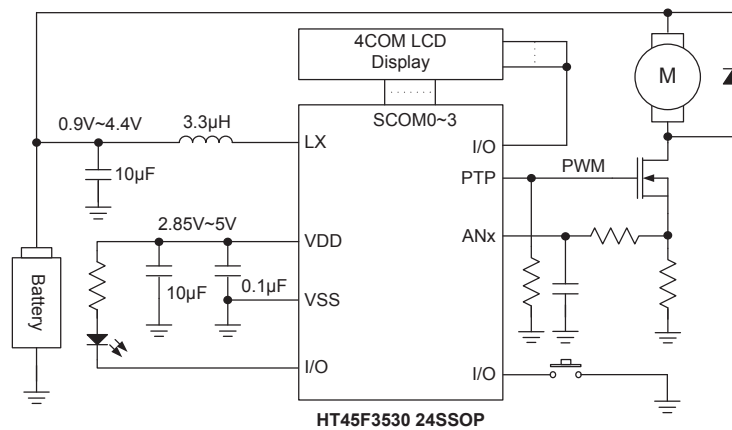
Application Circuits

HT45F3520/HT45F3530 without Charge Battery Application

HT45F3520/HT45F3530
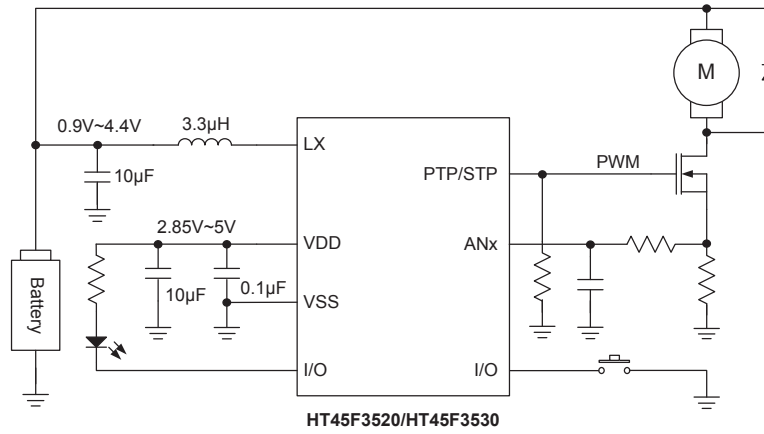
HT45F3520/HT45F3530 with Charge Battery Application

HT45F3520/HT45F3530
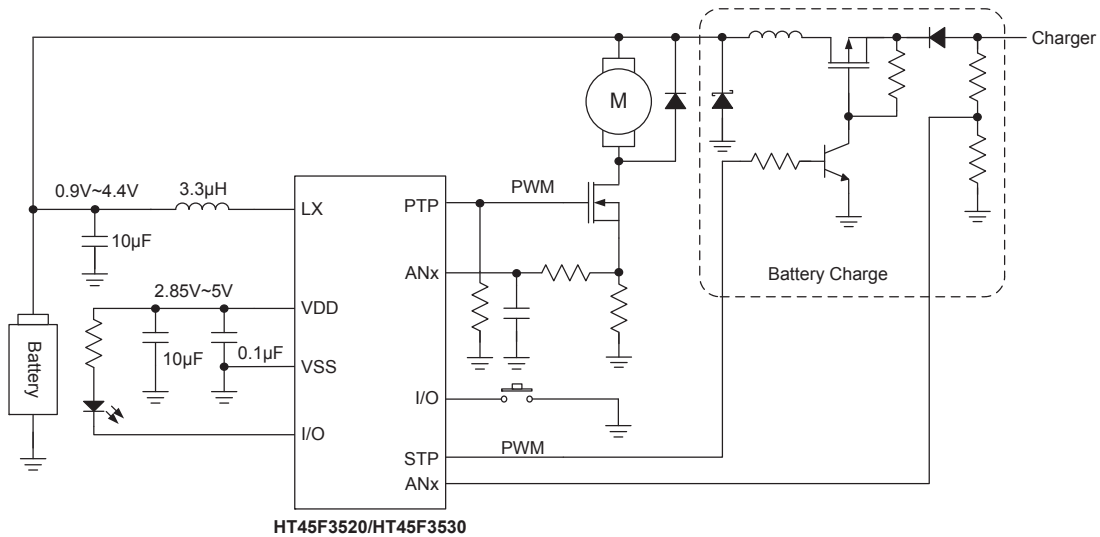
Software LCD Display Application for HT45F3530 Only

HT45F3530 24SSOP

Instruction Set

Introduction

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

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

Instruction Timing

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

Moving and Transferring Data

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

Arithmetic Operations

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

Logical and Rotate Operation

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

Branches and Control Transfer

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

Bit Operations

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

Table Read Operations

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

Other Operations

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

Instruction Set Summary

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

Table Conventions

x: Bits immediate data

m: Data Memory address

A: Accumulator

i: 0~7 number of bits

addr: Program memory address

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

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

Note: 1. For skip instructions, if the result of the comparison involves a skip then two cycles are required, if no skip takes place only one cycle is required.

2. Any instruction which changes the contents of the PCL will also require 2 cycles for execution.

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

Instruction Definition

ADC A,[m]	Add Data Memory to ACC with Carry
Description	The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the Accumulator.
Operation	ACC ← 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	$[m] \leftarrow \overline{[m]}$
Affected flag(s)	Z

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

RL [m]	Rotate Data Memory left
Description	The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0.
Operation	[m].(i+1) ← [m].i; (i=0~6) [m].0 ← [m].7
Affected flag(s)	None

RLA [m]	Rotate Data Memory left with result in ACC
Description	The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.
Operation	ACC.(i+1) ← [m].i; (i=0~6) ACC.0 ← [m].7
Affected flag(s)	None

RLC [m]	Rotate Data Memory left through Carry
Description	The contents of the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into bit 0.
Operation	[m].(i+1) ← [m].i; (i=0~6) [m].0 ← C C ← [m].7
Affected flag(s)	C

RLCA [m]	Rotate Data Memory left through Carry with result in ACC
Description	Data in the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into the bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.
Operation	ACC.(i+1) ← [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 and the carry flag are rotated right by 1 bit with bit 0 rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.
Operation	$ACC.i \leftarrow [m].(i+1)$; $(i=0\sim6)$ $ACC.7 \leftarrow [m].0$
Affected flag(s)	None

RRC [m]	Rotate Data Memory right through Carry
Description	The contents of the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7.
Operation	$[m].i \leftarrow [m].(i+1)$; $(i=0\sim6)$ $[m].7 \leftarrow C$ $C \leftarrow [m].0$
Affected flag(s)	C

RRCA [m]	Rotate Data Memory right through Carry with result in ACC
Description	Data in the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.
Operation	$ACC.i \leftarrow [m].(i+1)$; $(i=0\sim6)$ $ACC.7 \leftarrow C$ $C \leftarrow [m].0$
Affected flag(s)	C

SBC A,[m]	Subtract Data Memory from ACC with Carry
Description	The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.
Operation	$ACC \leftarrow ACC - [m] - C$
Affected flag(s)	OV, Z, AC, C

SBCM A,[m]	Subtract Data Memory from ACC with Carry and result in Data Memory
Description	The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.
Operation	$[m] \leftarrow ACC - [m] - C$
Affected flag(s)	OV, Z, AC, C

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

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

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

Operation　　$ACC \leftarrow [m] - 1$
Skip if ACC=0

Affected flag(s)　　None

SET [m]　　Set Data Memory

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

SET [m].i　　Set bit of Data Memory

Description　　Bit i of the specified Data Memory is set to 1.
Operation　　$[m].i \leftarrow 1$
Affected flag(s)　　None

SIZ [m]　　Skip if increment Data Memory is 0

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

Operation　　$[m] \leftarrow [m] + 1$
Skip if [m]=0

Affected flag(s)　　None

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

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

Operation　　$ACC \leftarrow [m] + 1$
Skip if ACC=0

Affected flag(s)　　None

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

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

Operation　　Skip if $[m].i \neq 0$
Affected flag(s)　　None

SUB A,[m]　　Subtract Data Memory from ACC

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

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

SUBM A,[m]	Subtract Data Memory from ACC with result in Data Memory
Description	The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.
Operation	[m] ← 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.

- Further Package Information (include Outline Dimensions, Product Tape and Reel Specifications)

- Packing Meterials Information

- Carton information

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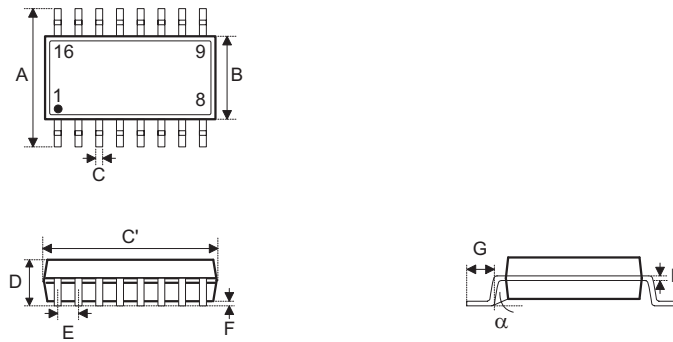
Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.012	—	0.020
C'	—	0.390 BSC	—
D	—	—	0.069
E	—	0.050 BSC	—
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	—	6 BSC	—
B	—	3.9 BSC	—
C	0.31	—	0.51
C'	—	9.9 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°

24-pin SSOP (150mil) Outline Dimensions
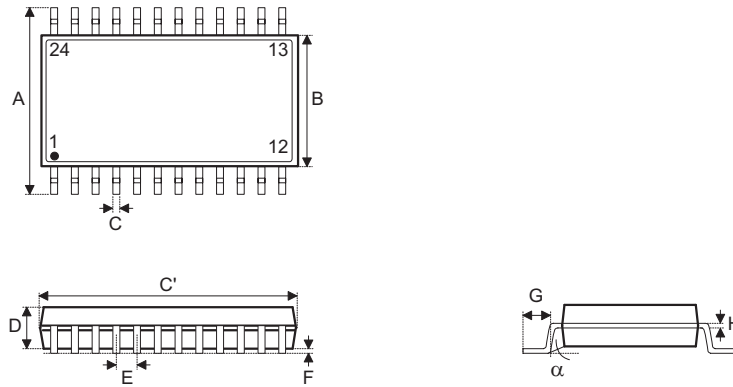
Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.008	—	0.012
C'	—	0.341 BSC	—
D	—	—	0.069
E	—	0.025 BSC	—
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	—	6.0 BSC	—
B	—	3.9 BSC	—
C	0.20	—	0.30
C'	—	8.66 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°