Glucose Meter Flash MCU

HT45F65/HT45F66/HT45F67

Revision: V2.10 Date: September 18, 2023

www.holtek.com

Table of Contents

Features

CPU Features

- Operating voltage
  - f_{SYS}=4MHz: 2.2V~5.5V
  - f_{SYS}=8MHz: 2.4V~5.5V
  - f_{SYS}=12MHz: 2.7V~5.5V
  - f_{SYS}=16MHz: 4.5V~5.5V
- Up to 0.33μs instruction cycle with 12MHz system clock at V_{DD}=5V
- Power down and wake-up functions to reduce power consumption
- Five oscillators
  - External Crystal – HXT
  - External 32.768kHz Crystal – LXT
  - External RC – ERC (HT45F66/HT45F67)
  - Internal RC – HIRC
  - Internal 32kHz RC – LIRC
- Multi-mode operation: NORMAL, SLOW, IDLE and SLEEP
- Fully integrated internal 4MHz, 8MHz and 12MHz oscillator requires no external components
- All instructions executed in one or two instruction cycles
- Table read instructions
- 61 or 63 powerful instructions
- Up to 12-level subroutine nesting
- Bit manipulation instruction

Peripheral Features

- Flash Program Memory: 8K×16~32K×16
- RAM Data Memory: 256×8~512×8
- In Application Programming (IAP)
- Watchdog Timer function
- Up to 59 bidirectional I/O lines
- LCD driver function
- Multiple pin-shared external interrupts
- Multiple Timer Module for time measure, input capture, compare match output, PWM output or single pulse output function
- Serial Interfaces Module with Dual SPI and I^2C interfaces
- Single Serial SPI Interface
- UART Interface for fully duplex asynchronous comminucation
- Dual Time-Base functions for generation of fixed time interrupt signals
- Multi-channel 12-bit resolution A/D converter
- Low voltage reset function
- Low voltage detect function
- Temperature sensor

- 10-bit DAC
- 16-bit Audio DAC
- Bandgap
- Operational Amplifier
- Flash program memory can be re-programmed up to 10,000 times
- Flash program memory data retention > 10 years
- Package types: 48-pin/64-pin and 80-pin LQFP

Development Tools

For rapid product development and to simplify device parameter setting, Holtek has provided relevant development tools which users can download from the following link:

https://www.holtek.com/esk-fv160-200 (HT45F65/67 only)

General Description

The devices are a Flash Memory A/D with LCD type 8-bit high performance RISC architecture microcontroller designed for applications that interface directly to analog signals and which require an LCD interface. Offering users the convenience of Flash Memory multi-programming features, these devices also include a wide range of functions and features. Other memory includes an area of RAM Data Memory as well as IAP for storage of non-volatile data such as serial numbers, calibration data etc.

Analog features include a multi-channel 12-bit A/D converter. Multiple and extremely flexible Timer Modules provide timing, pulse generation and PWM generation functions. Communication with the outside world is catered for by including fully integrated SPI or I^2C interface functions, two popular interfaces which provide designers with a means of easy communication with external peripheral hardware. Protective features such as an internal Watchdog Timer, Low Voltage Reset and Low Voltage Detector coupled with excellent noise immunity and ESD protection ensure that reliable operation is maintained in hostile electrical environments.

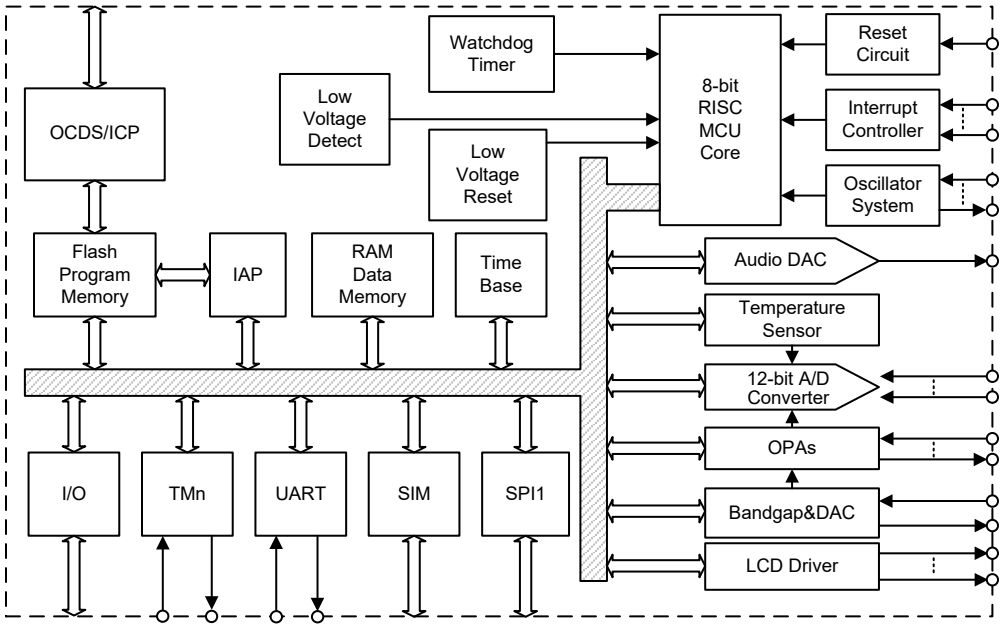
A full choice of HXT, LXT, ERC, HIRC and LIRC oscillator functions are provided including a fully integrated system oscillator which requires no external components for its implementation. The ability to operate and switch dynamically between a range of operating modes using different clock sources gives users the ability to optimise microcontroller operation and minimise power consumption.

Selection Table

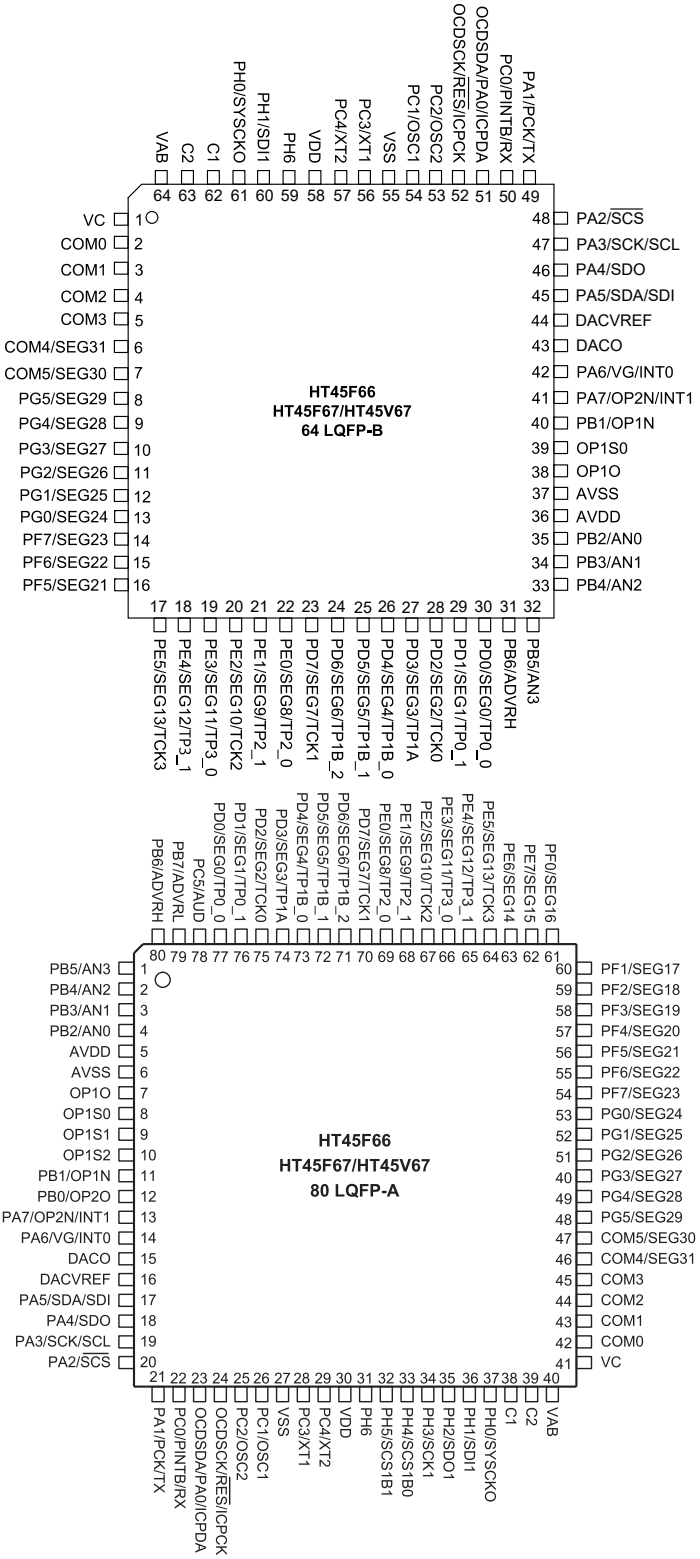
Most features are common to all devices, the main feature distinguishing them are Memory capacity, I/O count, TM features, stack capacity and package types. The following table summarises the main features of each device.

Part No.	V_{DD}	Program Memory	Data Memory	I/O	A/D	D/A	Audio DAC	Timer Module	Interface (SPI/I^2C)	SPI1	UART	Stack	package
HT45F65	2.2V~5.5V	8K×16	256×8	29	12-bit ×2	10-bit ×1	16-bit ×1	10-bit CTM×2 16-bit STM×1	√	√	√	12	48LQFP
HT45F66		16K×16	512×8	59	12-bit ×8	10-bit ×1	16-bit ×1	10-bit CTM×2 16-bit STM×1 10-bit ETM×1	√	√	√	12	64/80 LQFP
HT45F67		32K×16											

Block Diagram



Pin Assignment



Note: For less pin-count package types there will be unbonded pins which should be properly configured to avoid unwanted current consumption resulting from floating input conditions. Refer to the “Standby Current Considerations” and “Input/Output Ports” sections.

Pin Description

HT45F65/HT45V65

Pin Name	Function	OPT	I/T	O/T	Pin-Shared Mapping
PA0~PA7	Port A	PAWU PAPU	ST	CMOS	—
PB2~PB3	Port B	PBPU	ST	CMOS	—
PC2~PC4	Port C	PCPU	ST	CMOS	—
PD0~PD7	Port D	PDPU	ST	CMOS	—
PE0~PE7	Port E	PEPU	ST	CMOS	—
AN2~AN3	ADC input	PBFS	AN	—	PB2~PB3
DACVREF	10-bit DAC reference voltage	—	AN	—	—
DACO	10-bit DAC output	—	—	AO	—
OP1O	OPA1 output	—	—	AO	—
OP2O	OPA2 output	PAFS	—	AO	PA6
OP1N	OPA1 non-inverting input	PAFS	AN	—	PA5
OP2N	OPA2 non-inverting input	PAFS	AN	—	PA7
OP1S0	OPA1 output select 0	—	AN	—	—
OP1S1	OPA1 output select 1	—	AN	—	—
OP1S2	OPA1 output select 2	—	AN	—	—
TCK0~TCK2	TM0~TM2 external input clock	—	ST	—	PE2, PE4, PE6
TP0_0, TP0_1	TM0 output	—	ST	CMOS	PE1, PE1
TP1_0, TP1_1	TM1 output	—	ST	CMOS	PE3, PE3
TP2_0, TP2_1	TM2 output	—	ST	CMOS	PE5, PE5
INT0, INT1	External interrupt 0, 1	—	ST	—	PC4, PA7
PINTB	Peripheral Interrupt	—	ST	—	PD3
PCK	Peripheral Clock output	—	—	CMOS	PD4
SDI	SPI Data input	—	ST	—	PE0
SDO	SPI Data output	—	—	CMOS	PD7
$\overline{SCS}$	SPI Slave Select	—	ST	CMOS	PD5
SCS1B0	SPI1 Slave Select 0	—	ST	CMOS	PD1
SCS1B1	SPI1 Slave Select 1	—	ST	CMOS	PD2
SCK	SPI Serial Clock	—	ST	CMOS	PD6
SCK1	SPI1 Serial Clock	—	ST	CMOS	PD0
SCL	I^2C Clock	—	ST	NMOS	PD6
SDA	I^2C Data	—	ST	NMOS	PE0
OSC1	HXT/ERC pin	CO	HXT	—	PA2
OSC2	HXT pin	CO	—	HXT	PA1
XT1	LXT pin	CO	LXT	—	PA3
XT2	LXT pin	CO	—	LXT	PA4
OCDSCK	OCDS clock	—	ST	—	—
ICPCK	ICP clock	—	ST	—	—
$\overline{RES}$	Reset input	—	ST	—	—
VDD	Power supply	—	PWR	—	—
VSS	Ground	—	PWR	—	—
AVDD	Analog power supply	—	PWR	—	—
AVSS	Analog ground	—	PWR	—	—
VG	Virtual ground	—	AN	—	PC4
OCDSDA	OCDS address/data	—	ST	CMOS	PA0
ICPDA	ICP address/data	—	ST	CMOS	PA0
VAB	LCD voltage pump	—	—	—	—

Pin Name	Function	OPT	I/T	O/T	Pin-Shared Mapping
VC	LCD voltage pump	—	—	—	—
C1, C2	LCD voltage pump	—	—	—	—
SEG0~7	LCD segment output	—	—	CMOS	PD0~PD7
SEG8~15	LCD segment output	—	—	CMOS	PE0~PE7
COM0~COM3	LCD common output	—	—	CMOS	—
RX	UART serial data input pin	UCR1 UCR2	ST	—	PC2
TX	UART serial data output pin	UCR1 UCR2	—	CMOS	PC3

Note: I/T: Input type; O/T: Output type;
OP: Optional by configuration option (CO) or register option;
PWR: Power; CO: Configuration option;
ST: Schmitt Trigger input; CMOS: CMOS output;
NMOS: NMOS output; AN: Analog input pin;
AO: Analog output pin; HXT: High frequency crystal oscillator;
LXT: Low frequency crystal oscillator.
The OCDSDA and OCDSCK pins are only available for HT45V65 device.

HT45F66, HT45F67/HT45V67

This table refelects the situation for the larger package type.

Pin Name	Function	OPT	I/T	O/T	Pin-Shared Mapping
PA0~PA7	Port A	PAWU PAPU	ST	CMOS	—
PB0~PB7	Port B	PBPU	ST	CMOS	—
PC0~PC5	Port C	PCPU	ST	CMOS	—
PD0~PD7	Port D	PDPU	ST	CMOS	—
PE0~PE7	Port E	PEPU	ST	CMOS	—
PF0~PF7	Port F	PFPU	ST	CMOS	—
PG0~PG5	Port G	PGPU	ST	CMOS	—
PH0~PH6	Port H	PHPU	ST	CMOS	—
AN0~AN3	ADC Input	PBFS	AN	—	PB2~PB5
ADVRH	ADC Positive Reference Voltage	PBFS	AN	—	PB6
ADVRL	ADC Negative Reference Voltage	PBFS	AN	—	PB7
AUD	Audio DAC Output	PCFS	—	AO	PC5
DACVREF	10-bit DAC Reference Voltage	—	AN	—	—
DACO	10-bit DAC Output	—	—	AO	—
OP1O	OPA1 Output	—	—	AO	—
OP2O	OPA2 Output	PBFS	—	AO	PB0
OP1N	OPA1 Non-inverting Input	PBFS	AN	—	PB1
OP2N	OPA2 Non-inverting Input	PAFS SFS1	AN	—	PA7
OP1S0	OPA1 Output Select 0	—	AN	—	—
OP1S1	OPA1 Output Select 1	—	AN	—	—
OP1S2	OPA1 Output Select 2	—	AN	—	—
TCK0~TCK3	TM0~TM3 External Input Clock	—	ST	—	PD2, PD7, PE2, PE5
TP0_0, TP0_1	TM0 Output	—	ST	CMOS	PD0, PD1
TP1A	TM1 Output	—	ST	CMOS	PD3
TP1B_0~TP1B_2	TM1 Output	—	ST	CMOS	PD4, PD5, PD6
TP2_0, TP2_1	TM2 Output	—	ST	CMOS	PE0, PE1
TP3_0, TP3_1	TM3 Output	—	ST	CMOS	PE3, PE4

Pin Name	Function	OPT	I/T	O/T	Pin-Shared Mapping
INT0, INT1	External Interrupt 0, 1	—	ST	—	PA6, PA7
PINTB	Peripheral Interrupt	—	ST	—	PC0
PCK	Peripheral Clock Output	—	—	CMOS	PA1
SDI	SPI Data Input	—	ST	—	PA5
SDI1	SPI1 Data Input	—	ST	—	PH1
SDO	SPI Data Output	—	—	CMOS	PA4
SDO1	SPI1 Data Output	—	—	CMOS	PH2
$\overline{SCS}$	SPI Slave Select	—	ST	CMOS	PA2
SCS1B0	SPI1 Slave Select 0	—	ST	CMOS	PH4
SCS1B1	SPI1 Slave Select 1	—	ST	CMOS	PH5
SCK	SPI Serial Clock	—	ST	CMOS	PA3
SCK1	SPI1 Serial Clock	—	ST	CMOS	PH3
SCL	I^2C Clock	—	ST	NMOS	PA3
SDA	I^2C Data	—	ST	NMOS	PA5
OSC1	HXT/ERC Pin	CO	HXT	—	PC1
OSC2	HXT Pin	CO	—	HXT	PC2
XT1	LXT Pin	CO	LXT	—	PC3
XT2	LXT Pin	CO	—	LXT	PC4
OCDSCK	OCDS Clock	—	ST	—	—
ICPCK	ICP Clock	—	ST	—	—
$\overline{RES}$	Reset Input	—	ST	—	—
VDD	Power Supply	—	PWR	—	—
VSS	Ground	—	PWR	—	—
AVDD	Analog Power Supply	—	PWR	—	—
AVSS	Analog Ground	—	PWR	—	—
VG	Virtual Ground	—	AN	—	PA6
OCDSDA	OCDS Address/Data	—	ST	CMOS	PA0
ICPDA	ICP Address/Data	—	ST	CMOS	PA0
SYSCKO	System Cclock Output	—	—	CMOS	PH0
VAB	LCD Voltage Pump	—	—	—	—
VC	LCD Voltage Pump	—	—	—	—
C1, C2	LCD Voltage Pump	—	—	—	—
SEG0~7	LCD Segment Output	—	—	CMOS	PD0~PD7
SEG8~15	LCD Segment Output	—	—	CMOS	PE0~PE7
SEG16~23	LCD Segment Output	—	—	CMOS	PF0~PF7
SEG24~SEG29	LCD Segment Output	—	—	CMOS	PG0~PG5
SEG30, SEG31	LCD Segment Output	—	—	CMOS	—
COM0~COM5	LCD Common Output	—	—	CMOS	—
TX	UART serial data output pin	UCR1 UCR2	—	CMOS	PA1
RX	UART serial data input pin	UCR1 UCR2	ST	—	PC0

Note: I/T: Input type; O/T: Output type;
OP: Optional by configuration option (CO) or register option;
PWR: Power; CO: Configuration option;
ST: Schmitt Trigger input; CMOS: CMOS output;
NMOS: NMOS output; AN: Analog input pin;
AO: Analog output pin; HXT: High frequency crystal oscillator;
LXT: Low frequency crystal oscillator.
The OCDSDA and OCDSCK pins are only available for HT45V67 device.

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..... -60˚C to 150˚C

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

I_{OH} Total ..... -80mA

I_{OL} Total 80mA

Total Power Dissipation 500mW

Note: These are stress ratings only. Stresses exceeding the range specified under "Absolute Maximum Ratings" may cause substantial damage to 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_{DD1}	Operating Voltage (HXT/ERC)	—	f_{SYS}=4MHz	2.2	—	5.5	V
			f_{SYS}=8MHz	2.4	—	5.5	V
			f_{SYS}=12MHz	2.7	—	5.5	V
			f_{SYS}=16MHz	4.5	—	5.5	V
V_{DD2}	Operating Voltage (HIRC)	—	f_{SYS}=4MHz	2.2	—	5.5	V
			f_{SYS}=8MHz	2.4	—	5.5	V
			f_{SYS}=12MHz	2.7	—	5.5	V
I_{DD1}	Operating Current (f_{SYS}=f_H, f_S=f_{SUB}=f_{LXT} or f_{LIRC})	3V	No load, f_H=455kHz, ADC off, WDT enable	—	100	150	μA
		5V		—	280	450	μA
		3V	No load, f_H=1MHz, ADC off, WDT enable	—	250	400	μA
		5V		—	500	1000	μA
		3V	No load, f_H=4MHz, ADC off, WDT enable	—	420	630	μA
		5V		—	700	1000	μA
		3V	No load, f_H=8MHz, ADC off, WDT enable	—	0.8	1.5	mA
		5V		—	1.5	3.0	mA
		3V	No load, f_H=12MHz, ADC off, WDT enable	—	1.5	2.5	mA
		5V		—	3.0	5.0	mA
		5V	No load, f_H=16MHz, ADC off, WDT enable	—	5.0	8.0	mA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{DD2}	Operating Current ($f_S=f_{SUB}=f_{LXT}$ or f_{LIRC})	3V	No load, f_H=8MHz, $f_{SYS}=f_H/2$, ADC off, WDT enable	—	500	750	μA
		5V		—	800	1200	μA
		3V	No load, f_H=8MHz, $f_{SYS}=f_H/4$, ADC off, WDT enable	—	420	630	μA
		5V		—	700	1000	μA
		3V	No load, f_H=8MHz, $f_{SYS}=f_H/8$, ADC off, WDT enable	—	400	600	μA
		5V		—	600	800	μA
		3V	No load, f_H=8MHz, $f_{SYS}=f_H/16$, ADC off, WDT enable	—	360	540	μA
		5V		—	560	700	μA
		3V	No load, f_H=8MHz, $f_{SYS}=f_H/32$, ADC off, WDT enable	—	320	480	μA
		5V		—	520	650	μA
		3V	No load, f_H=8MHz, $f_{SYS}=f_H/64$, ADC off, WDT enable	—	280	420	μA
		5V		—	440	600	μA
I_{DD3}	Operating Current (LXT/LIRC, $f_{SYS}=f_L=f_{LXT}$, $f_S=f_{SUB}=f_{LXT}$)	3V	No load, ADC off, WDT enable, LVR enable	—	40	70	μA
		5V		—	60	100	μA
I_{STB1}	Standby Current (Idle) ($f_{SYS}=f_H$, $f_S=f_{SUB}=f_{LXT}$ or f_{LIRC})	3V	No load, system HALT, ADC off, WDT enable, f_{SYS}=8MHz, SPI or I²C on, PCK on, PCK=$f_{SYS}/8$	—	0.3	0.5	mA
		5V		—	0.5	0.8	mA
I_{STB2}	Standby Current (Idle) (f_{SYS}=off, $f_S=f_{SUB}=f_{LXT}$ or f_{LIRC})	3V	No load, system HALT, ADC off, WDT enable	—	1.5	3.0	μA
		5V		—	2.5	5.0	μA
I_{STB3}	Standby Current (Idle) (LXT, $f_{SYS}=f_L=f_{LXT}$, $f_S=f_{SUB}=f_{LXT}$)	3V	No load, system HALT, ADC off, WDT enable, f_{SYS}=32768Hz	—	1.9	4.0	μA
		5V		—	3.3	7.0	μA
I_{STB4}	Standby Current (Sleep) (f_{SYS}=off, $f_S=f_{SUB}=f_{LXT}$ or f_{LIRC})	3V	No load, system HALT, ADC off, WDT disable, f_{SYS}=12MHz	—	0.1	1.0	μA
		5V		—	0.3	2.0	μA
V_{IL1}	Input Low Voltage for I/O Ports	—	—	0	—	$0.3V_{DD}$	V
V_{IH1}	Input High Voltage for I/O Ports	—	—	$0.7V_{DD}$	—	V_{DD}	V
V_{IL2}	Input Low Voltage ($\overline{RES}$)	—	—	0	—	$0.4V_{DD}$	V
V_{IH2}	Input High Voltage ($\overline{RES}$)	—	—	$0.9V_{DD}$	—	V_{DD}	V
V_{LVR1}	Low Voltage Reset Voltage	—	LVR Enable, 2.1V option	-5%	2.1	+5%	V
V_{LVR2}			LVR Enable, 2.55V option		2.55		V
V_{LVR3}			LVR Enable, 3.15V option		3.15		V
V_{LVR4}			LVR Enable, 3.8V option		3.8		V
I_{LVR}	Low Voltage Reset Current	—	LVR Enable, LVDEN=0	—	30	50	μA
V_{LVD1}	Low Voltage Detector Voltage	—	LVDEN = 1, V_{LVD} = 2.0V	-5%	2.0	+5%	V
V_{LVD2}			LVDEN = 1, V_{LVD} = 2.2V		2.2		V
V_{LVD3}			LVDEN = 1, V_{LVD} = 2.4V		2.4		V
V_{LVD4}			LVDEN = 1, V_{LVD} = 2.7V		2.7		V
V_{LVD5}			LVDEN = 1, V_{LVD} = 3.0V		3.0		V
V_{LVD6}			LVDEN = 1, V_{LVD} = 3.3V		3.3		V
V_{LVD7}			LVDEN = 1, V_{LVD} = 3.6V		3.6		V
V_{LVD8}			LVDEN = 1, V_{LVD} = 4.0V		4.0		V
I_{LVD1}	Low Voltage Detector Current	—	LVR disable, LVDEN = 1	—	30	60	μA
I_{LVD2}			LVR enable, LVDEN = 1	—	1	1	μA
I_{OL}	I/O Port Sink Current (PA5,PA6,PA7 for HT45F65; PA7,PB0,PB1,PH6 for HT45F66/67)	3V	$V_{OL}=0.1V_{DD}$	6	12	—	mA
		5V	$V_{OL}=0.1V_{DD}$	10	20	—	mA
I_{OH}	I/O Port Source Current (PA5,PA6,PA7 for HT45F65; PA7,PB0,PB1,PH6 for HT45F66/67)	3V	$V_{OH}=0.9V_{DD}$	-6	-12	—	mA
		5V	$V_{OH}=0.9V_{DD}$	-10	-20	—	mA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{OL}	I/O Port Sink Current	3V	V_{OL}=0.1V_{DD}	4	8	—	mA
		5V	V_{OL}=0.1V_{DD}	10	20	—	mA
I_{OH}	I/O Port Source Current	3V	V_{OH}=0.9V_{DD}	-2	-4	—	mA
		5V	V_{OH}=0.9V_{DD}	-5	-10	—	mA
R_{PH}	Pull-high Resistance of 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_{CPU}	Operating Clock	2.2~5.5V	—	DC	—	4	MHz
		2.4~5.5V		DC	—	8	MHz
		2.7~5.5V		DC	—	12	MHz
		4.5~5.5V		DC	—	16	MHz
f_{SYS}	System Clock (HXT)	2.2~5.5V	—	0.4	—	4	MHz
		2.4~5.5V		0.4	—	8	MHz
		2.7~5.5V		0.4	—	12	MHz
		4.5~5.5V		0.4	—	16	MHz
f_{HIRC}	System Clock (HIRC)	3V/5V	Ta=25°C	-2%	4	+2%	MHz
		3V/5V	Ta=25°C	-2%	8	+2%	MHz
		5V	Ta=25°C	-2%	12	+2%	MHz
		2.2V~3.6V	Ta=0~70°C	-7%	4	+7%	MHz
		3.0V~5.5V	Ta=0~70°C	-7%	4	+7%	MHz
		2.4V~3.6V	Ta=0~70°C	-7%	8	+7%	MHz
		3.0V~5.5V	Ta=0~70°C	-7%	8	+7%	MHz
		3.0V~5.5V	Ta=0~70°C	-7%	12	+7%	MHz
f_{LXT}	System Clock (LXT)	—	—	—	32768	—	Hz
t_{RES}	External Reset Low Pulse Width	—	—	10	—	—	μs
t_{SST}	System Start-up Timer Period (Wake-up from HALT)	—	f_{SYS}=HXT or LXT	1024	—	—	t_{SYS}
			f_{SYS}=ERC or HIRC	16	—	—	
			f_{SYS}=LIRC	2	—	—	
t_{INT}	Interrupt Pulse Width	—	Turn on RC filter (Note)	10	—	—	μs
t_{LVR}	Low Voltage Width to Reset	—	—	120	240	480	μs
t_{LVD}	Low Voltage Width to Interrupt	—	—	20	45	90	μs
t_{LVDS}	LVDO Stable Time	—	LVR disable	—	—	150	μs
		—	LVR enable	—	—	15	μs

Note: The INTn RC filter is turned on/off via the configuration options.

ADC & Bandgap Electrical Characteristics

Ta= 25˚C

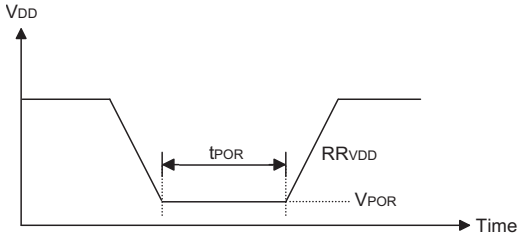
Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
AV_{DD}	Analog operating voltage	—	—	2.4	—	5.5	V
V_{ADI}	AD Input Voltage	—	—	0	—	V_{REF}	V
V_{REF}	ADC Input Reference Voltage Range	—	—	2.0	—	AV_{DD}	V
V_{BG}	Bandgap Reference With Buffer Voltage	—	AV_{DD}=3V	-3%	2.0	+3%	V
t_{BG}	Temperature coefficient of Build-in Band gap reference voltage	—	AV_{DD}=3V, T=10~40°C	-30	—	+170	ppm/°C
I_{BG}	Bandgap Reference With Buffer Driving Current	—	V_{BG} is used, LVR disable, LVDEN=0, BGOS[1:0]=10, BGEN=1	—	200	400	μA
I_{BG1}	Bandgap Reference With Buffer Driving Current	—	V_{BG} is used, LVR disable, LVDEN=0, BGOS[1:0]=00 or 01 BGEN=1	—	450	650	μA
DNL	Differential Non-linearity	3V	t_{AD}=0.5μs	-3	—	+3	LSB
INL	Integral Non-linearity	3V	t_{AD}=0.5μs	-4	—	+4	LSB
I_{ADC}	Additional Power Consumption if A/D Converter is used	3V	No load, t_{AD}=0.5μs	—	1.0	2.0	mA
		5V	No load, t_{AD}=0.5μs	—	1.5	3.0	mA
t_{BGS}	V_{BG} Turn on Stable Time	—	—	10	—	—	ms
t_{AD}	A/D Clock Period	2.7~5.5V	—	0.5	—	100	μs
t_{ADC}	A/D Conversion Time	2.7~5.5V	12 bit ADC	—	16	—	t_{AD}
t_{ON2ST}	ADC on to ADC start	2.7~5.5V	—	2	—	—	μs

Note: ADC conversion time (t_{ADC}) = n (bits ADC) + 4 (sampling time), the conversion for each bit needs one ADC clock (t_{AD}).

Power-on Reset Characteristics

Ta= 25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{POR}	V_{DD} Start Voltage to Ensure Power-on Reset	—	—	—	—	100	mV
RR_{VDD}	V_{DD} Raising 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



LCD D.C. Characteristics

Ta= 25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Condition				
I_{STB}	Standby Current (Idle) (f_{SYS}, f_{WDT} off, f_S=f_{SUB}=f_{LXT} or f_{LIRC})	3V	No load, system HALT, LCD on, WDT off, C type	—	3	6	μA
		5V		—	5	10	μA
I_{OL}	LCD Common and Segment Sink Current	3V	V_{OL}=0.1V_{LCD}	210	420	—	μA
		5V		350	700	—	μA
I_{OH}	LCD Common and Segment Source Current	3V	V_{OH}=0.9V_{LCD}	-80	-160	—	μA
		5V		-180	-360	—	μA

Audio DAC Electrical Characteristics

Ta= 25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DAC}	DAC operating voltage	—	—	2.4	—	5.5	V
I_{DAC}	DAC operating current	5V	1 kHz sin wave, full-scale	—	3	4.5	mA
THD	Total Harmonic Distortion	—	—	—	-54	-48	db
RES	Resolution	—	—	—	—	16	bit
V_O	Output Voltage Level	—	—	0.01	—	0.99	V_{DD}

10-bit DAC Electrical Characteristics

Ta= 25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DAC}	Operating Voltage	—	—	2.4	—	5.5	V
INL	Integral Non-linearity	3V	—	-4	—	+4	LSB
DNL	Differential Non-linearity	3V	—	-2	—	+2	LSB
R_O	Output Resistance	3V	—	—	5	—	KΩ
V_{OS}	DAC Output Voltage Stability	3V	BGOS[1:0]=10	-2	—	+2	mV
V_O	Output Voltage Level	—	—	0.01	—	0.99	V_{DD}

Operational Amplifier Electrical Characteristics

Ta= 25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
R_O	Output Resistance	2.2V~3.6V	Ta=0~50°C, Rload=50kΩ 0.2V<V_{OP}<V_{DD}-0.2V (Voltage Follower onfiguration)	—	—	10	Ω
I_{LOP1N}	OP1N Leakage Current	2.2V~3.6V	Ta=0~50°C	—	±5	—	nA
	Offset Voltage Temp. Drift	3V	—	-0.2	—	+0.2	μV/°C
f_{GB}	Gain Bandwidth Product	3V	R_L=1MΩ, C_L=60pF, V_{IN}=V_{CM}/2	—	100	—	kHz
V_{OS}	Input Offset Voltage	3.3V	—	-15	—	15	mV
V_{CM}	Common Mode Voltage Range	—	—	0	—	V_{DD}-1.4	V
A_V	Gain	5V	—	—	100	—	dB
V_{OP}	Operating Voltage	—	—	2.4	—	5.5	V
I_{OP}	Operating Current	5V	V_P=Vn=1/2V_{DD}	—	650	—	μA
CMRR	Common Mode Rejection Ratio	3V	—	—	80	—	dB
PSRR	Power Supply Rejection Ratio	5V	—	—	75	—	dB

Analog Switch Electrical Characteristics

Ta= 25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
R_{OP}	On Resistance (OP1Sx, OP2N and OP2O)	2.2V~3.6V	Ta=0~50°C, I_{SW}=200μA	—	—	500	Ω
R_{VG}	On Resistance (VG)	2.2V~3.6V	Ta=0~50°C, SW3=1	—	—	20	Ω
I_L	Leakage Current (VG)	2.2V~3.6V	Ta=0~50°C	—	±0.5	—	nA

Temperature Sensor Electrical Characteristics

Ta= 25°C

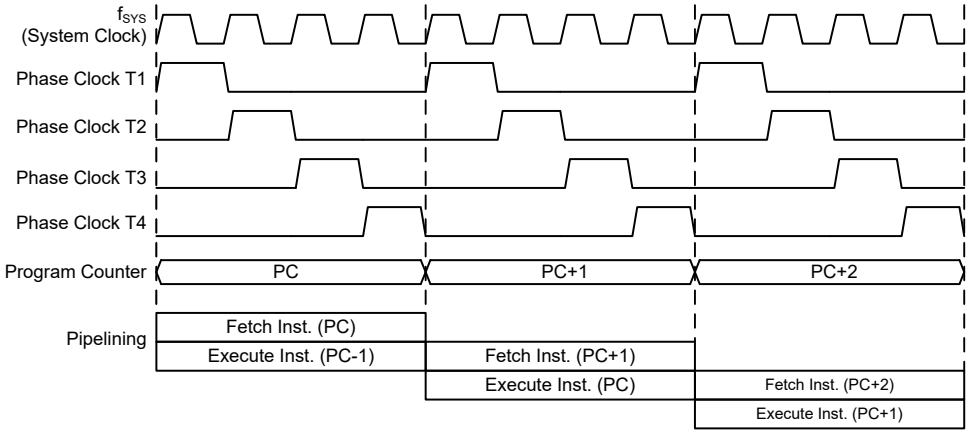
Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
T_{OP}	Operating Temperature	—	—	0	—	70	°C
T_{aa}	Absolute Accuracy	—	—	-3	—	+3	°C
T_g	Gain	2.2~3.6V	—	—	2.88	—	mV/°C
T_{OS}	Offset Temperature	2.2~3.6V	—	—	0.95	—	V
I_S	Standby Current	3.0	—	—	—	1	μA
Vtemp	Operating Voltage	—	—	2.4	—	5.5	V

System Architecture

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

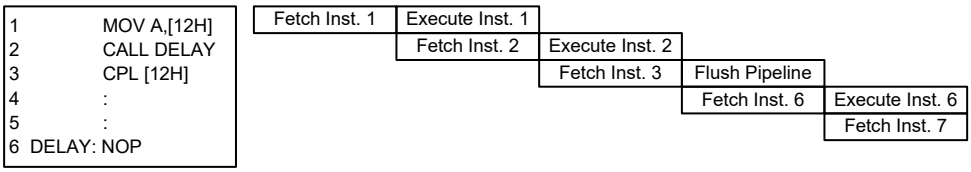
Clocking and Pipelining

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



System Clocking and Pipelining

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



Instruction Fetching

Program Counter

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

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

Part No.	Program Counter		
	High Byte		Low Byte
HT45F65	PC12~PC8		PCL7~PCL0
HT45F66	BP.5	PC12~PC8	
HT45F67	BP.6~BP.5		

Note: 1. BP.5: Bank Pointer Register bit 5 for HT45F66.
2. BP.6~BP.5: Bank Pointer Register bit 6~bit 5 for HT45F67.

Program Counter

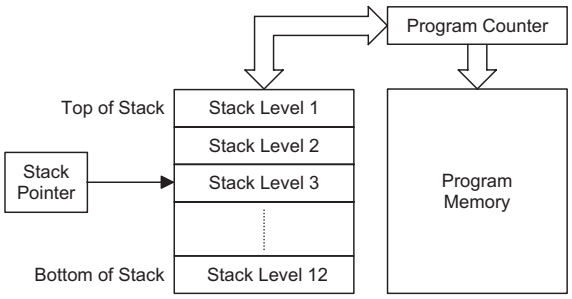
The lower byte of the Program Counter, known as the Program Counter Low register or PCL, is available for program control and is a readable and writeable register. By transferring data directly into this register, a short program jump can be executed directly, however, as only this low byte is available for manipulation, the jumps are limited to the present page of memory, that is 256 locations. When such program jumps are executed it should also be noted that a dummy cycle will be inserted. Manipulating the PCL register may cause program branching, so an extra cycle is needed to pre-fetch.

Stack

This is a special part of the memory which is used to save the contents of the Program Counter only. The stack is organized into 12 levels and neither part of the data nor part of the program space, and is neither readable nor writeable. The activated level is indexed by the Stack Pointer, and is neither readable nor writeable. At a subroutine call or interrupt acknowledge signal, the contents of the Program Counter are pushed onto the stack. At the end of a subroutine or an interrupt routine, signaled by a return instruction, RET or RETI, the Program Counter is restored to its previous value from the stack. After a device reset, the Stack Pointer will point to the top of the stack.

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

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



Part No.	Stack Levels
HT45F65/HT45F66/HT45F67	12

Arithmetic and Logic Unit – ALU

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

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

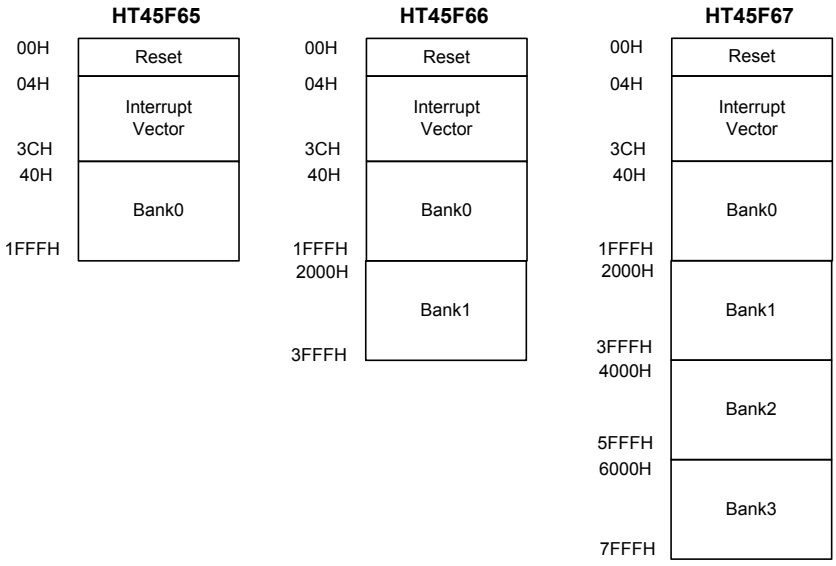
Flash Program Memory

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

Structure

The Program Memory has a capacity of 8K×16 bits to 32K×16 bits. The Program Memory is addressed by the Program Counter and also contains data, table information and interrupt entries. Table data, which can be setup in any location within the Program Memory, is addressed by a separate table pointer register.

The Program Memory divided into two Banks, Bank 0 and Bank1 for HT45F66. While the Program Memory divided into four Banks, Bank 0, Bank1, Bank2 and Bank 3 for HT45F67. The required Bank is selected using Bit 6 and Bit 5 of the BP Register.



Program Memory Structure

Special Vectors

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

Look-up Table

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

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

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

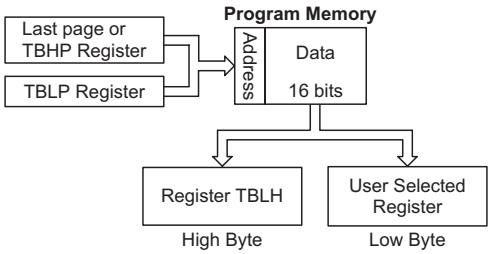


Table Program Example

The following example shows how the table pointer and table data is defined and retrieved from the microcontroller. This example uses raw table data located in the Program Memory which is stored there using the ORG statement. The value at this ORG statement is “1F00H” which is located in ROM Bank 3 and refers to the start address of the last page within the 32K Program Memory of the HT45F67. 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 "7F06H" 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 "TABRD [m]" instruction is being used. The high byte of the table data which in this case is equal to zero will be transferred to the TBLH register automatically when the "TABRD [m]" instruction is executed.

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

Table Read Program Example

```
rombank 3 code1
ds .section ‘data’
tempreg1 db?      ; temporary register #1
tempreg2 db?      ; temporary register #2
:
:
mov a, 06h        ; initialise low table pointer - note that this address
mov tblp, a       ; is referenced
mov a, 7Fh        ; initialise high table pointer
mov tbhp, a
:
:
tabrd tempreg1    ; transfers value in table referenced by table pointer data at
                  ; program memory address "7F06H" transferred to tempreg1 and TBLH
dec tblp          ; reduce value of table pointer by one
tabrd tempreg2    ; transfers value in table referenced by table pointer data at
                  ; program memory address "7F05H" transferred to tempreg2 and TBLH
                  ; in this example the data 1AH is transferred to tempreg1 and data
                  ; 0FH to register tempreg2
                  ; the value “00H” will be transferred to the high byte register TBLH
:
:
code1 .section ‘code’
org 7F00h         ; sets initial address of program memory
dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
:
:
```

In Circuit Programming – ICP

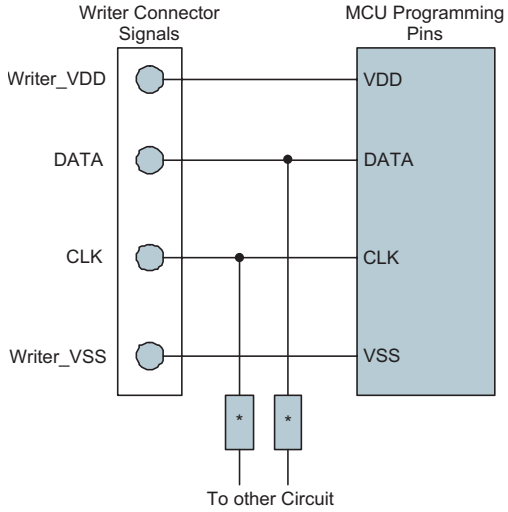
The provision of Flash type Program Memory provides the user with a means of convenient and easy upgrades and modifications to their programs on the same device. As an additional convenience, Holtek has provided a means of programming the microcontroller in-circuit using a 4-pin interface. This provides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

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

Holtek Writer Pins	MCU Programming Pins	Pin Description
ICPDA	PA0	Programming Serial Data/Address
ICPCK	$\overline{\text{RES}}$	Programming Clock and Device Reset
VDD	VDD	Power Supply
VSS	VSS	Ground

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

During the programming process, taking control of the PA0 and $\overline{\text{RES}}$ 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.

Programmer Pin	MCU Pins
DATA	PA0
CLK	$\overline{\text{RES}}$

In Application Programming – IAP

This device offers IAP function to update data or application program to flash ROM. Users can define any ROM location for IAP, but there are some features which user must notice in using IAP function.

- Erase page: 32 words/page (HT45F65)
  Erase page: 64 words/page (HT45F66/HT45F67)
- Writing: 32 words/time (HT45F65)
  Writing: 64 words/time (HT45F66/HT45F67)
- Reading: 1 wod/time

In Application Programming Control Register

The Address register, FARL/FARH, and the Data register, FD0L/FD0H, FD1L/FD1H, FD2L/FD2H and FD3L/FD3H, located in Data Memory Bank 0, together with the Control register, FC0, FC1 and FC2, located in Data Memory Bank 1 are the corresponding Flash access registers for IAP. As indirect addressing is the only way to access the FC0 and FC1 registers, all read and write operations to the registers must be performed using the Indirect Addressing Register, IAR1, and the Memory Pointer, MP1. Because the FC0 and FC1 control registers are located at the address of 40H and 41H in Data Memory Bank 1, the MP1 Memory Pointer must first be set to the value 40/41H and the Bank Pointer set to "1".

• FC0 Register

Bit	7	6	5	4	3	2	1	0
Name	CFWEN	FMOD2	FMOD1	FMOD0	FWPEN	FWT	FRDEN	FRD
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 **CFWEN**: Confirm the FWEN status
0: Flash ROM write disable
1: software write 1 no action
When this bit is clear by firmware, the IAP controller exits FWEN mode. The user can read this bit to confirm the FWEN status.

Bit 6~4 **FMOD2~FMOD0**: Mode selection
000: write program ROM
001: page erase program ROM
010: reserved
011: read program ROM
101: reserved
110: FWEN (flash ROM write enable) mode
111: reserved

Bit 3 **FWPEN**: Flash ROM write procedure enable
0: disabled
1: enabled
When this bit is set to "1" and FMOD2~FMOD0 is set to 110, the program can execute "Set flash write enable (FWEN)".
When this bit is set to "1" and FMOD2~FMOD0 is set to 000, the controller will move register (FD0L and FD0H) data to flash ROM internal page buffer.

Bit 2 **FWT**: Flash ROM write control bit
0: Do not initiate Flash ROM write or Flash ROM Write process is completed.
1: Initiate Flash ROM write process
This bit can be set by software only, when write process completed, hardware will clear "FWT" bit.

Bit 1 **FRDEN**: Flash ROM read enable bit
0: Flash ROM read disable
1: Flash ROM read enable

Bit 0 **FRD**: Flash ROM read control bit
0: Do not initiate Flash read or Flash read process is completed
1: Initiate Flash read process
This bit can be set by software only, when read process completed, hardware will clear "FRD" bit.

• **FC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	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 **55H**: whole chip reset
When user writes 55H to this register, it will generate a reset signal to reset whole chip.

• **FC2 Register**

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

Bit 7~1 Unimplemented, read as "0"

Bit 0 **CLWB**: Flash ROM Write Buffer clear control bit
0: Do not initiate clear Write Buffer or clear Write Buffer process is completed.
1: Initiate clear Write Buffer process.
Before page write action, user must be set CLWB bit to clear Write Buffer. This bit can be set by software only, when clear Write Buffer process completed, hardware will clear "CLWB" bit.

• **FARL Register**

Bit	7	6	5	4	3	2	1	0
Name	A7	A6	A5	A4	A3	A2	A1	A0
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 **A7~A0**: The flash address[7:0]

• **FARH Register (HT45F65)**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	A12	A11	A10	A9	A8
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 **A12~A8**: The flash address [12:8]

• **FARH Register (HT45F66)**

Bit	7	6	5	4	3	2	1	0
Name	—	—	A13	A12	A11	A10	A9	A8
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 **A13~A8**: The flash address [13:8]

• **FARH Register (HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	—	A14	A13	A12	A11	A10	A9	A8
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7 Unimplemented, read as "0"
Bit 6~0 **A14~A8**: The flash address[14:8]

• **FD0L Register**

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

Bit 7~0 **D7~D0**: The first flash ROM data[7:0]

• **FD0H Register**

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

Bit 7~0 **D15~D8**: The first flash ROM data[15:8]

• **FD1L Register**

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

Bit 7~0 **D7~D0**: The second flash ROM data[7:0]

• **FD1H Register**

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

Bit 7~0 **D15~D8**: The second flash ROM data[15:8]

• **FD2L Register**

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

Bit 7~0 **D7~D0**: The third flash ROM data[7:0]

• **FD2H Register**

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

Bit 7~0 **D15~D8**: The third flash ROM data[15:8]

• **FD3L Register**

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

Bit 7~0 **D7~D0**: The fourth flash ROM data[7:0]

• **FD3H Register**

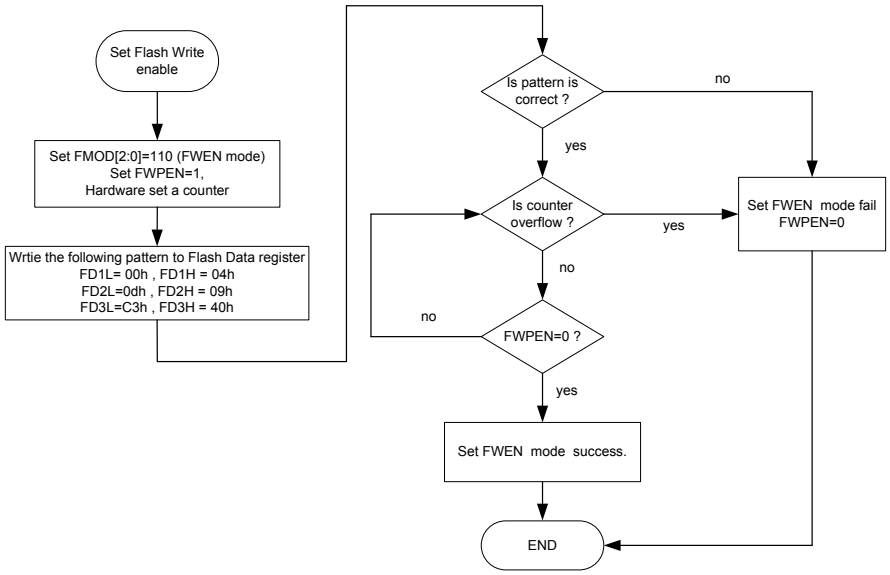
Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **D15~D8**: The fourth flash ROM data[15:8]

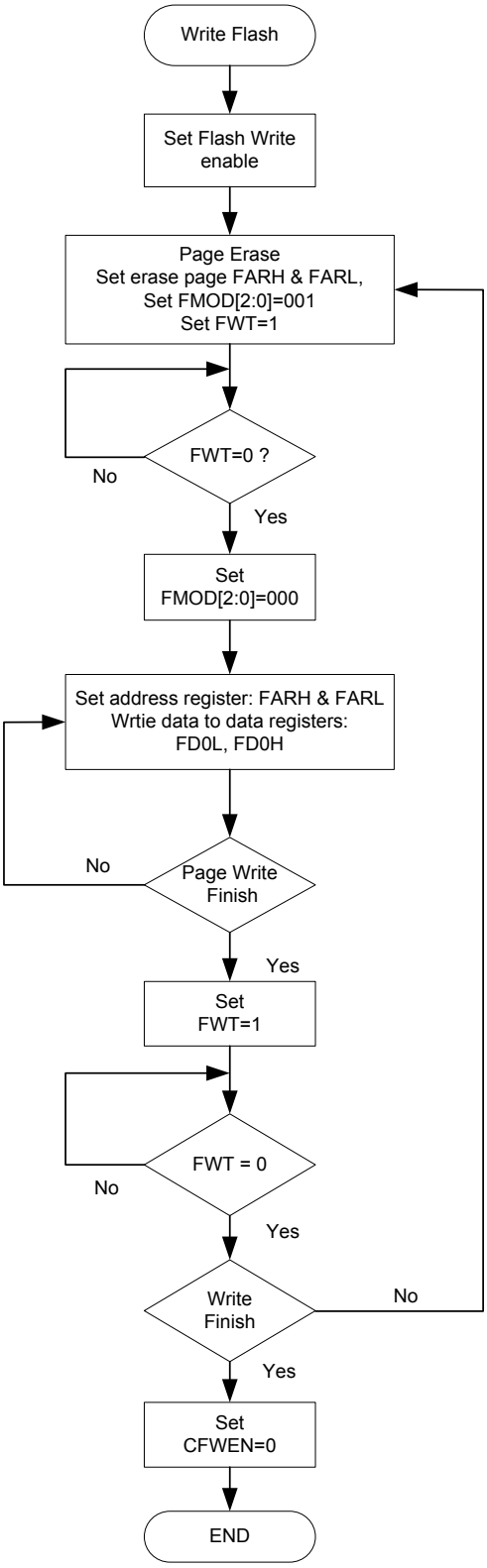
Set Flash Write Enable (FWEN)

In order to allow user to change Flash ROM data through Flash control register, the user must first enable the Flash write operation by the following procedure:

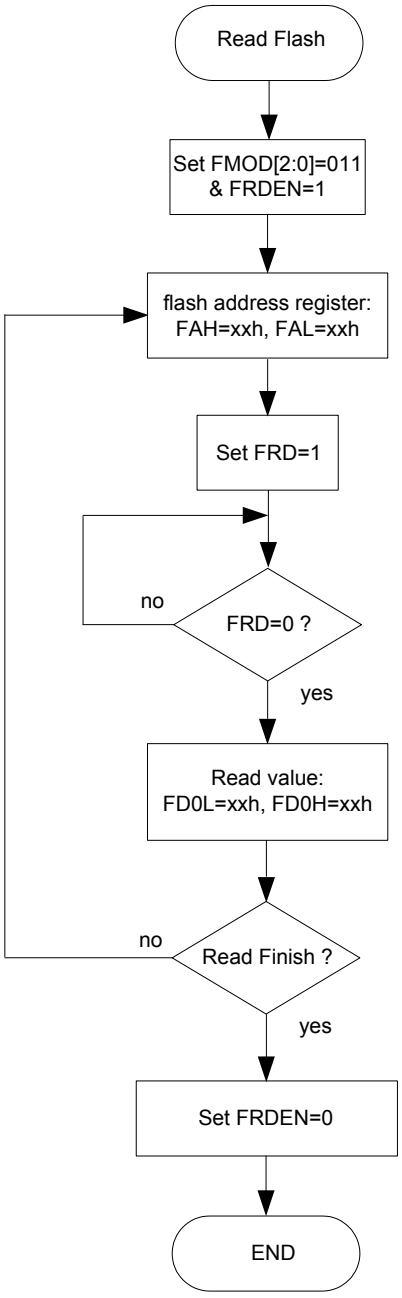
- Write 110 to the FMOD2~FMOD0 for setting FWEN mode
- Set FWPEN bit to 1. The step 1 and step 2 can set simultaneously
- The Hardware will start a 300μs counter to allow the user writing the correct pattern data to FD1L/FD1H~FD3L/FD3H. (The clock of 300μs counter is from LIRC.)
- Once 300μs counter is overflow or the pattern is uncorrect. The enable Flash write operation is fail and the user must repeat the above procedure one more
- No matter, the operation is success or fail. The hardware will clear FWPEN bit automatically
- The pattern data is (00H 04H 0DH 09H C3H 40H) to FD1L/FD1H~FD3L/FD3H
- Once the Flash write operation is enable, the user can change the Flash ROM data through the Flash control register
- For disable the Flash write operation, the user only clear CFWEN, it is no need to write the above procedure



Set Flash Write Enable Procedure



Write Flash Program Procedure



Read Flash Program Procedure

• **HT45F65**

ERASE PAGE	FARH	FARL[7:5]	Note
0	0000 0000	000	FARL[4:0]: don't care.
1	0000 0000	001	
2	0000 0000	010	
3	0000 0000	011	
4	0000 0000	100	
5	0000 0000	101	
6	0000 0000	110	
7	0000 0000	111	
8	0000 0001	000	
9	0000 0010	001	
:	:	:	

• **HT45F66 / HT45F67**

ERASE PAGE	FARH	FARL[7:6]	Note
0	0000 0000	00	FARL[5:0]: don't care.
1	0000 0000	01	
2	0000 0000	10	
3	0000 0000	11	
4	0000 0001	00	
5	0000 0001	01	
6	0000 0001	10	
7	0000 0001	11	
8	0000 0010	00	
9	0000 0010	01	
:	:	:	

On Chip Debug Support – OCDS

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

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

RAM Data Memory

The Data Memory is a volatile area of 8-bit wide RAM internal memory and is the location where temporary information is stored. Divided into three sections, the first of these is an area of RAM where special function registers are located. These registers have fixed locations and are necessary for correct operation of the devices.

Many of these registers can be read from and written to directly under program control, however, some remain protected from user manipulation. The second area of Data Memory is reserved for general purpose use. All locations within this area are read and write accessible under program control. The third area is reserved for the LCD Memory. This special area of Data Memory is mapped directly to the LCD display so data written into this memory area will directly affect the displayed data.

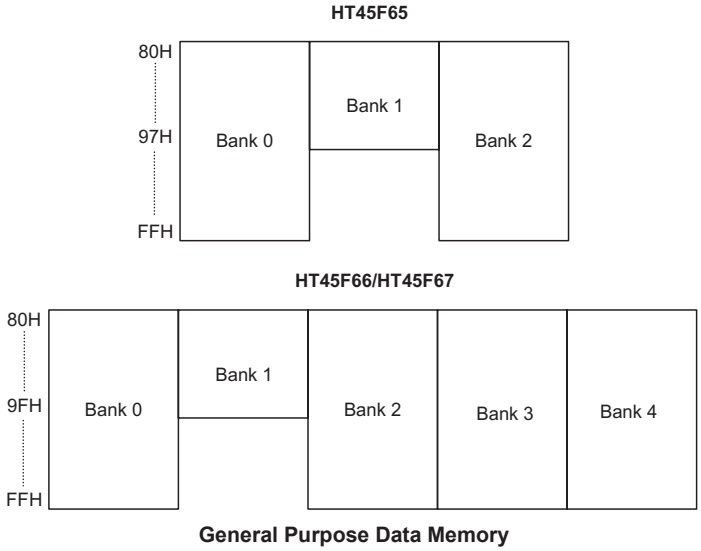
The addresses of the LCD Memory area overlap those in the General Purpose Data Memory area. Switching between the different Data Memory banks is achieved by setting the Bank Pointer to the correct value.

Device	Capacity	Banks
HT45F65	256×8	0: 80H~FFH 1: 80H~97F (LCD Memory) 2: 80H~FFH
HT45F66 HT45F67	512×8	0: 80H~FFH 1: 80H~9FH (LCD Memory) 2: 80H~FFH 3: 80H~FFH 4: 80H~FFH

Structure

The Data Memory is subdivided into several banks, all of which are implemented in 8-bit wide RAM. The Data Memory located in Bank 0 is subdivided into two sections, the Special Purpose Data Memory and the General Purpose Data Memory.

The start address of the Data Memory for the device is the address 00H. Registers which are common to all microcontrollers, such as ACC, PCL, etc., have the same Data Memory address. The LCD Memory is mapped into Bank 1. The other banks contain only General Purpose Data Memory for the device. As the Special Purpose Data Memory registers are mapped into all bank areas, they can subsequently be accessed from any bank location.



General Purpose Data Memory

HT45F65

	Bank 0~4	Bank 0, 2~4	Bank 1
00H	IAR0	40H	UCR2	FC0
01H	MP0	41H	BRG	FC1
02H	IAR1	42H	TxR/RXR	FC2
03H	MP1	43H	ADRH	
04H	BP	44H	ADCR0	
05H	ACC	45H	ADCR1	
06H	PCL	46H	ADCR2	
07H	TBLP	47H	SIMC0	
08H	TBLH	48H	SIMC1	
09H	TBHP	49H	SIMD	
0AH	STATUS	4AH	SIMA/SIMC2	
0BH	SMOD0	4BH	TM0C0	
0CH	LVDC	4CH	TM0C1	
0DH	INTEG	4DH	TM0DL	
0EH	WDTC	4EH	TM0DH	
0FH	TBC	4FH	TM0AL	
10H	INTC0	50H	TM0AH	
11H	INTC1	51H	Unused	
12H	INTC2	52H	Unused	
13H	SCKC	53H	Unused	
14H	MFI0	54H	Unused	
15H	MFI1	55H	Unused	
16H	MFI2	56H	Unused	
17H	Unused	57H	Unused	
18H	PAWU	58H	Unused	
19H	PAPU	59H	Unused	
1AH	PA	5AH	TM2C0	
1BH	PAC	5BH	TM2C1	
1CH	PBPU	5CH	TM2DL	
1DH	PB	5DH	TM2DH	
1EH	PBC	5EH	TM2AL	
1FH	PCPU	5FH	TM2AH	
20H	PC	60H	TM2RP	
21H	PCC	61H	TM1C0	
22H	PDPU	62H	TM1C1	
23H	PD	63H	TM1DL	
24H	PDC	64H	TM1DH	
25H	PEPU	65H	TM1AL	
26H	PE	66H	TM1AH	
27H	PEC	67H	LCDC	
28H	Unused	68H	SPI1C0	
29H	Unused	69H	SPI1C1	
2AH	Unused	6AH	SPI1D	
2BH	Unused	6BH	ADAC	
2CH	Unused	6CH	ADAL	
2DH	Unused	6DH	ADAH	
2EH	Unused	6EH	BGC	
2FH	Unused	6FH	DAC	
30H	Unused	70H	DAL	
31H	PAFS	71H	DAH	
32H	PBFS	72H	TSC	
33H	Unused	73H	PVREF	
34H	PCFS	74H	FARL	
35H	PDFS0	75H	FARH	
36H	PFFS1	76H	FD0L	
37H	PEFS0	77H	FD0H	
38H	PEFS1	78H	FD1L	
39H	Unused	79H	FD1H	
3AH	Unused	7AH	FD2L	
3BH	SMOD1	7BH	FD2H	
3CH	LVRC	7CH	FD3L	
3DH	ADRL	7DH	FD3H	
3EH	USR	7EH	OPC1	
3FH	UCR1	7FH	OPC2	

HT45F66/HT45F67

	Bank 0~4	Bank 0, 2~4	Bank 1
00H	IAR0	40H	UCR2	FC0
01H	MP0	41H	BRG	FC1
02H	IAR1	42H	TxR/RXR	FC2
03H	MP1	43H	ADRH	
04H	BP	44H	ADCR0	
05H	ACC	45H	ADCR1	
06H	PCL	46H	ADCR2	
07H	TBLP	47H	SIMC0	
08H	TBLH	48H	SIMC1	
09H	TBHP	49H	SIMD	
0AH	STATUS	4AH	SIMA/SIMC2	
0BH	SMOD0	4BH	TM0C0	
0CH	LVDC	4CH	TM0C1	
0DH	INTEG	4DH	TM0DL	
0EH	WDTC	4EH	TM0DH	
0FH	TBC	4FH	TM0AL	
10H	INTC0	50H	TM0AH	
11H	INTC1	51H	TM1C0	
12H	INTC2	52H	TM1C1	
13H	SCKC	53H	TM1C2	
14H	MFI0	54H	TM1DL	
15H	MFI1	55H	TM1DH	
16H	MFI2	56H	TM1AL	
17H	MFI3	57H	TM1AH	
18H	PAWU	58H	TM1BL	
19H	PAPU	59H	TM1BH	
1AH	PA	5AH	TM2C0	
1BH	PAC	5BH	TM2C1	
1CH	PBPU	5CH	TM2DL	
1DH	PB	5DH	TM2DH	
1EH	PBC	5EH	TM2AL	
1FH	PCPU	5FH	TM2AH	
20H	PC	60H	TM2RP	
21H	PCC	61H	TM3C0	
22H	PDPU	62H	TM3C1	
23H	PD	63H	TM3DL	
24H	PDC	64H	TM3DH	
25H	PEPU	65H	TM3AL	
26H	PE	66H	TM3AH	
27H	PEC	67H	LCDC	
28H	PFPU	68H	SPI1C0	
29H	PF	69H	SPI1C1	
2AH	PFC	6AH	SPI1D	
2BH	PGPU	6BH	ADAC	
2CH	PG	6CH	ADAL	
2DH	PGC	6DH	ADAH	
2EH	PHPU	6EH	BGC	
2FH	PH	6FH	DAC	
30H	PHC	70H	DAL	
31H	PAFS	71H	DAH	
32H	PBFS	72H	TSC	
33H	PCFS	73H	PVREF	
34H	PDFS	74H	FARL	
35H	PEFS	75H	FARH	
36H	PFFS	76H	FD0L	
37H	PGFS	77H	FD0H	
38H	PHFS	78H	FD1L	
39H	SFS0	79H	FD1H	
3AH	SFS1	7AH	FD2L	
3BH	SMOD1	7BH	FD2H	
3CH	LVRC	7CH	FD3L	
3DH	ADRL	7DH	FD3H	
3EH	USR	7EH	OPC1	
3FH	UCR1	7FH	OPC2	

: Unused, read as "00"

Special Purpose Data Memory

Special Function Register Description

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

Indirect Addressing Registers – IAR0, IAR1

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

Memory Pointers – MP0, MP1

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

Indirect Addressing Program Example

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

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

Bank Pointer – BP

The Program and Data Memory are divided into several banks. Selecting the required Program and Data Memory area is achieved using the Bank Pointer. For HT45F65, bits 1~0 of Bank Pointer is used to select Data Memory Banks 0~2. For HT45F66, bits 5 of the Bank Pointer is used to select Program Memory Banks 0~1, while bits 0~2 are used to select Data Memory Banks 0~4. For HT45F67, bits 5~6 of the Bank Pointer is used to select Program Memory Banks 0~3, while bits 0~2 are used to select Data Memory Banks 0~4.

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

As both the Program Memory and Data Memory share the same Bank Pointer Register, care must be taken during programming.

• **BP Register (HT45F65)**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	DMBP1	DMBP0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **DMBP1~DMBP0**: select Data Memory Banks
00: Bank0
01: Bank1
10: Bank2
11: undefined

• **BP Register (HT45F66)**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PMBP0	—	—	DMBP2	DMBP1	DMBP0
R/W	—	—	R/W	—	—	R/W	R/W	R/W
POR	—	—	0	—	—	0	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5 **PMBP0**: Select Program Memory Banks
0: Bank 0, Program Memory Address is from 0000H ~ 1FFFH
1: Bank 1, Program Memory Address is from 2000H ~ 3FFFH
To successfully jump to a non-consecutive program memory address located in different program memory bank using the branch instructions shcu as “JMP” or “CALL”, the program memory bank pointer bit, PMBP0, should first be properly setup before the branch instructions are executed.

Bit 4~3 Unimplemented, read as "0"

Bit 2~0 **DMBP2~DMBP0**: select Data Memory Banks
000: Bank0
001: Bank1
010: Bank2
011: Bank3
100: Bank4
110~111: Undefined

• **BP Register (HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	—	PMBP1	PMBP0	—	—	DMBP2	DMBP1	DMBP0
R/W	—	R/W	R/W	—	—	R/W	R/W	R/W
POR	—	0	0	—	—	0	0	0

Bit 7 Unimplemented, read as "0"

Bit 6~5 **PMBP1, PMBP0**: Select Program Memory Banks
00: Bank 0, Program Memory Address is from 0000H ~ 1FFFH
01: Bank 1, Program Memory Address is from 2000H ~ 3FFFH
10: Bank 2, Program Memory Address is from 4000H ~ 5FFFH
11: Bank 3, Program Memory Address is from 6000H ~ 7FFFH
To successfully jump to a non-consecutive program memory address located in different program memory bank using the branch instructions shcu as “JMP” or “CALL”, the corresponding program memory bank pointer bits, PMBP1~PMBP0, should first be properly setup before the branch instructions are executed.

Bit 4~3 Unimplemented, read as "0"

Bit 2~0 **DMBP2~DMBP0**: Select Data Memory Banks
000: Bank 0
001: Bank 1
010: Bank 2
011: Bank 3
100: Bank 4
110~111: Undefined

Accumulator – ACC

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

Program Counter Low Register – PCL

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

Look-up Table Registers – TBLP, TBHP, TBLH

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

Status Register – STATUS

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

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

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

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

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

• STATUS Register

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

"x" unknown

Bit 7~6 Unimplemented, read as "0"

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

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

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

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

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

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

C is also affected by a rotate through carry instruction.

Oscillators

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

Oscillator Overview

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

Type	Name	Freq.	Pins
External Crystal	HXT	400kHz~16MHz	OSC1/OSC2
External RC	ERC	8MHz	OSC1
Internal High Speed RC	HIRC	4, 8 or 12MHz	—
External Low Speed Crystal	LXT	32.768kHz	XT1/XT2
Internal Low Speed RC	LIRC	32kHz	—

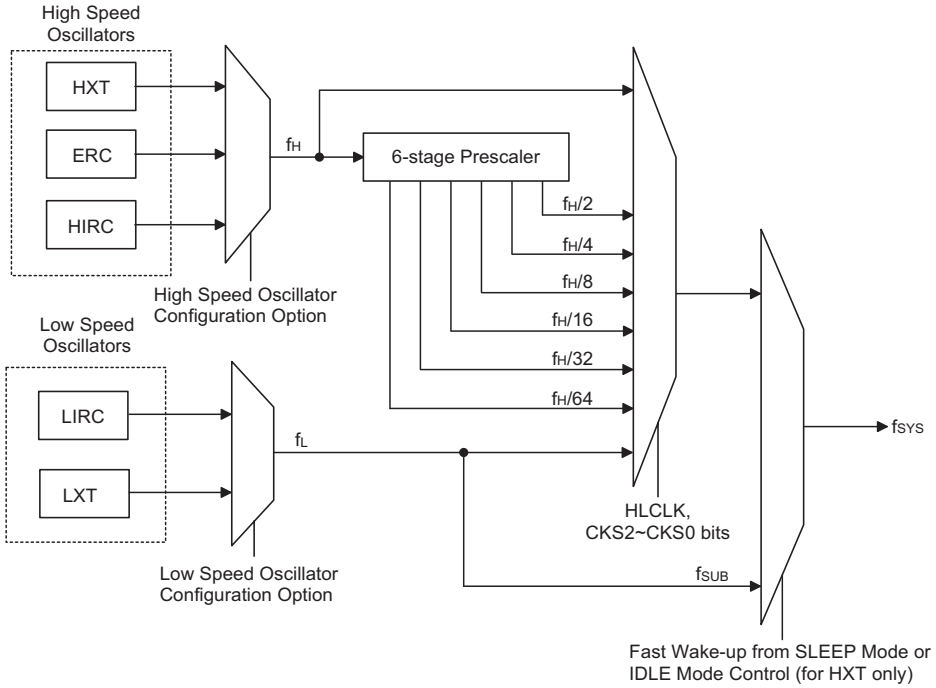
Note: The external RC oscillator only exists in the HT45F66/HT45F67.

Oscillator Types

System Clock Configurations

There are five methods of generating the system clock, three high speed oscillators and two low speed oscillators. The high speed oscillators are the external crystal/ceramic oscillator, external RC oscillator and the internal 4MHz, 8MHz or 12MHz RC oscillator. The two low speed oscillators are the internal 32kHz RC oscillator and the external 32.768kHz crystal 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 SMOD0 register and as the system clock can be dynamically selected.

The actual source clock used for each of the high speed and low speed oscillators is chosen via configuration options. The frequency of the slow speed or high speed system clock is also determined using the HLCLK bit and CKS2~CKS0 bits in the SMOD0 register. Note that two oscillator selections must be made namely one high speed and one low speed system oscillators. It is not possible to choose a no-oscillator selection for either the high or low speed oscillator.



Note: The external RC oscillator only exists in the HT45F66/HT45F67.

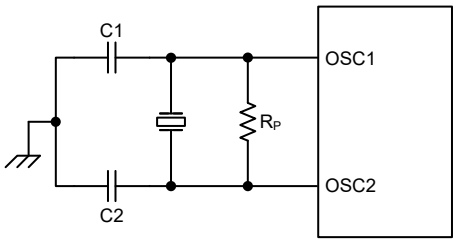
System Clock Configurations

External Crystal/Ceramic Oscillator – HXT

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

For oscillator stability and to minimise the effects of noise and crosstalk, it is important to ensure thatthe crystal and any associated resistors andcapacitors along with interconnectinglines are all located as close to the MCUas possible.

In order to avoid unexpected situations, select the correct HXT mode in the configuration option according to the externally connected crystal osicllator.



Note: 1. R_P is normally not required. C1 and C2 are required.
2. Although not shown OSC1/OSC2 pins have a parasitic capacitance of around 7pF.

Crystal/Resonator Oscillator – HXT

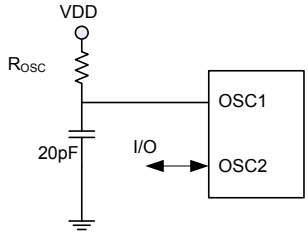
Crystal Oscillator C1 and C2 Values		
Crystal Frequency	C1	C2
16MHz	0pF	0pF
12MHz	0pF	0pF
8MHz	0pF	0pF
4MHz	0pF	0pF
1MHz	100pF	100pF
Note: C1 and C2 values are for guidance only.		

Crystal Recommended Capacitor Values

External RC Oscillator – ERC (HT45F66/HT45F67)

The External RC Oscillator is one of the high frequency oscillator choices for these two devices, which is selected via configuration option. Using the ERC oscillator only requires that a resistor, with a value between 56kΩ and 2.4MΩ, is connected between OSC1 and VDD, and a capacitor is connected between OSC1 and ground, providing a low cost oscillator configuration. It is only the external resistor that determines the oscillation frequency; the external capacitor has no influence over the frequency and is connected for stability purposes only. 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 resistance/frequency reference point, it can be noted that with an external 120kΩ resistor connected and with a 5V voltage power supply and temperature of 25°C degrees, the oscillator will have a frequency of 8MHz within a tolerance of 2%. Here only the OSC1 pin is used, which is shared with I/O pin PC1, leaving pin PC2 free for use as a normal I/O pin.

For oscillator stability and to minimise the effects of noise and crosstalk, it is important to locate the capacitor and resistoras close to the MCU as possible. Note that if there are external resistor and capacitor connected, the ERC mode (filter on) should be selected in the configuration option. However, if there are no external resistor and capacitor connected, the EC mode should be selected and the desired frequency is input through the OSC1 pin.



External RC Oscillator – ERC

Internal RC Oscillator – HIRC

The internal RC oscillator is a fully integrated system oscillator requiring no external components, which is selected via configuration option. The internal RC oscillator has three fixed frequencies of either 4MHz, 8MHz or 12MHz. Device trimming during the manufacturing process and the inclusion of internal frequency compensation circuits are used to ensure that the influence of the power supply voltage, temperature and process variations on the oscillation frequency are minimised. As a result, at a power supply of either 3V or 5V and at a temperature of 25°C degrees, the fixed oscillation frequency of 4MHz, 8MHz or 12MHz will have a tolerance within 2%. Note that if this internal system clock option is selected, it requires no external pins for its operation.

External 32.768kHz Crystal Oscillator – LXT

The External 32.768kHz Crystal System Oscillator is one of the low frequency oscillator choices, which is selected via configuration option. This clock source has a fixed frequency of 32.768kHz and requires a 32.768kHz crystal to be connected between pins XT1 and XT2. The external resistor and capacitor components connected to the 32.768kHz crystal are necessary to provide oscillation. For applications where precise frequencies are essential, these components may be required to provide frequency compensation due to different crystal manufacturing tolerances. During power-up there is a time delay associated with the LXT oscillator waiting for it to start-up.

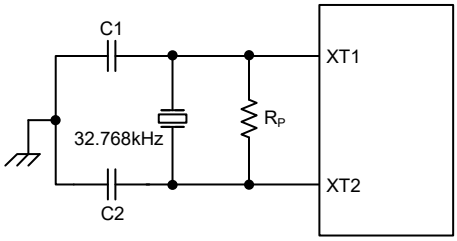
When the microcontroller enters the SLEEP or IDLE Mode, the system clock is switched off to stop microcontroller activity and to conserve power. However, in many microcontroller applications it may be necessary to keep the internal timers operational even when the microcontroller is in the SLEEP or IDLE Mode. To do this, another clock, independent of the system clock, must be provided.

However, for some crystals, to ensure oscillation and accurate frequency generation, it is necessary to add two small value external capacitors, C1 and C2. The exact values of C1 and C2 should be selected in consultation with the crystal or resonator manufacturer specification. The external parallel feedback resistor, R_P, is required.

Some configuration options determine if the XT1/XT2 pins are used for the LXT oscillator or as I/O pins.

- If the LXT oscillator is not used for any clock source, the XT1/XT2 pins can be used as normal I/O pins.
- If the LXT oscillator is used for any clock source, the 32.768kHz crystal should be connected to the XT1/XT2 pins.

For oscillator stability and to minimise the effects of noise and crosstalk, it is important to ensure thatthe crystal and any associated resistors andcapacitors along with interconnectinglines are all located as close to the MCUas possible.



Note: 1. R_P, C1 and C2 are required.
2. Although not shown XT1/XT2 pins have a parasitic capacitance of around 7pF.

External LXT Oscillator

LXT Oscillator C1 and C2 Values		
Crystal Frequency	**C1**	**C2**
32.768kHz	10pF	10pF
Note: 1. C1 and C2 values are for guidance only. 2. R_P=5M~10MΩ is recommended.		

32.768kHz Crystal Recommended Capacitor Values

LXT Oscillator Low Power Function

The LXT oscillator can function in one of two modes, the Quick Start Mode and the Low Power Mode. The mode selection is executed using the LXTLP bit in the TBC register.

LXTLP Bit	LXT Mode
0	Quick Start
1	Low-power

After power on, the LXTLP bit will be automatically cleared to zero ensuring that the LXT oscillator is in the Quick Start operating mode. In the Quick Start Mode the LXT oscillator will power up and stabilise quickly. However, after the LXT oscillator has fully powered up it can be placed into the Low-power mode by setting the LXTLP bit high. The oscillator will continue to run but with reduced current consumption, as the higher current consumption is only required during the LXT oscillator start-up. In power sensitive applications, such as battery applications, where power consumption must be kept to a minimum, it is therefore recommended that the application program sets the LXTLP bit high about 2 seconds after power-on.

It should be noted that, no matter what condition the LXTLP bit is set to, the LXT oscillator will always function normally, the only difference is that it will take more time to start up if in the Low-power mode.

Internal 32kHz Oscillator – LIRC

The Internal 32kHz System Oscillator is one of the low frequency oscillator choices, which is selected via configuration option. It is a fully integrated RC oscillator with a typical frequency of 32kHz at 5V, requiring no external components for its implementation. Device trimming during the manufacturing process and the inclusion of internal frequency compensation circuits are used to ensure that the influence of the power supply voltage, temperature and process variations on the oscillation frequency are minimised.

Supplementary Oscillators

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

Operating Modes and System Clocks

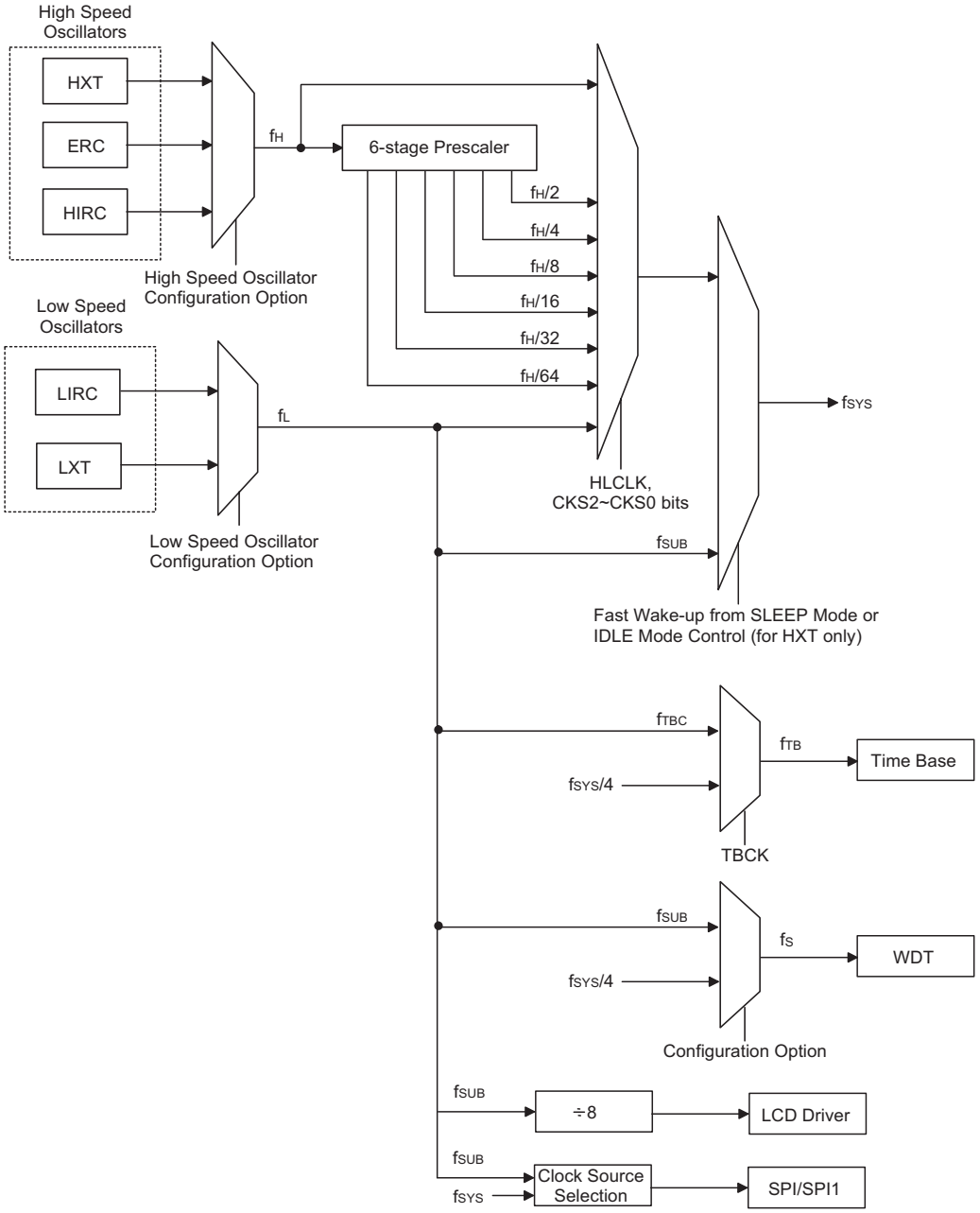
Present day applications require that their microcontrollers have high performance but often still demand that they consume as little power as possible, conflicting requirements that are especially true in battery powered portable applications. The fast clocks required for high performance will by their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

System Clocks

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

The main system clock, can come from either a high frequency f_H or low frequency f_L source, and is selected using the HLCLK bit and CKS2~CKS0 bits in the SMOD0 register. The high speed system clock can be sourced from either an HXT, ERC or HIRC oscillator, selected via a configuration option. The low speed system clock source can be sourced from internal clock f_L. If f_L is selected then it can be sourced by either the LXT or LIRC oscillator, selected via a configuration option. The other choice, which is a divided version of the high speed system oscillator has a range of $f_H/2$~$f_H/64$.

There are two additional internal clocks for the peripheral circuits, the substitute clock, f_{SUB}, and the Time Base clock, f_{TBC}. Each of these internal clocks is sourced by either the LXT or LIRC oscillators, selected via configuration options. The f_{SUB} clock is used to provide a substitute clock for the microcontroller just after a wake-up has occurred to enable faster wake-up times.



Device Clock Configurations

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

Together with $f_{SYS}/4$ which is also used as one of the clock sources for the Watchdog timer, the f_{TBC} clock is used as a source for the Time Base interrupt functions and for the TMs. The f_{SUB} is used as the LCD source.

The external RC oscillator only exists in the HT45F66/HT45F67.

System Operation Modes

There are six different modes of operation for the microcontroller, each one with its own special characteristics and which can be chosen according to the specific performance and power requirements of the application. There are two modes allowing normal operation of the microcontroller, the NORMAL Mode and SLOW Mode. The remaining four modes, the SLEEP0, SLEEP1, IDLE0 and IDLE1 Mode are used when the microcontroller CPU is switched off to conserve power.

Operating Mode	Description				
	CPU	f_{SYS}	f_{SUB}	f_S	f_{TBC}
NORMAL Mode	on	f_H~f_H/64	on	on	on
SLOW Mode	on	f_L	on	on	on
IDLE0 Mode	off	off	on	on/off	on
IDLE1 Mode	off	on	on	on	on
SLEEP0 Mode	off	off	off	off	off
SLEEP1 Mode	off	off	on	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 by one of the high speed oscillators. This mode allows the microcontroller to operate normally with a clock source from one of the high speed oscillators, either the HXT, ERC or HIRC oscillators. The high speed oscillator will however first be divided by a ratio ranging from 1 to 64, the actual ratio being selected by the CKS2~CKS0 and HLCLK bits in the SMOD0 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 one of the low speed oscillators, either the LXT or the LIRC. Running the microcontroller in this mode allows it to run with much lower operating currents. In the SLOW Mode, the f_H is off.

SLEEP0 Mode

The SLEEP Mode is entered when a HALT instruction is executed and when the IDLEN bit in the SMOD0 register is low. In the SLEEP0 mode the CPU will be stopped, and the f_{SUB} and f_S clocks will be stopped too, and the Watchdog Timer function is disabled. In this mode, the LVDEN is must set to "0". If the LVDEN is set to "1", it won't enter the SLEEP0 Mode.

SLEEP1 Mode

The SLEEP Mode is entered when a HALT instruction is executed and when the IDLEN bit in the SMOD0 register is low. In the SLEEP1 mode the CPU will be stopped. However the f_{SUB} and f_S clocks will continue to operate if the LVDEN is "1" or the Watchdog Timer function is enabled and if its clock source is chosen via configuration option to come from the f_{SUB}.

IDLE0 Mode

The IDLE0 Mode is entered when a HALT instruction is executed and when the IDLEN bit in the SMOD0 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, TMs, LCD driver, SPI1 and SIM. In the IDLE0 Mode, the system oscillator will be stopped. In the IDLE0 Mode the Watchdog Timer clock, f_S, will either be on or off depending upon the f_S clock source. If the source is f_{SYS}/4 then the f_S clock will be off, and if the source comes from f_{SUB} then f_S will be on.

IDLE1 Mode

The IDLE1 Mode is entered when a HALT instruction is executed and when the IDLEN bit in the SMOD0 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, TMs, LCD driver, SPI1 and SIM. In the IDLE1 Mode, the system oscillator will continue to run, and this system oscillator may be high speed or low speed system oscillator. In the IDLE1 Mode the Watchdog Timer clock, f_S, either from $f_{SYS}/4$ or f_{SUB}, will be on.

Control Register

A single register, SMOD0, is used for overall control of the internal clocks within the device.

• SMOD0 Register

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

Bit 7~5 **CKS2~CKS0**: The system clock selection when HLCLK is "0"
000: f_L (f_{LXT} or f_{LIRC})
001: f_L (f_{LXT} or 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 either the LXT or LIRC, a divided version of the high speed system oscillator can also be chosen as the system clock source.

Bit 4 **FSTEN**: Fast Wake-up Control (only for HXT)
0: Disable
1: Enable

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

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

This is the low speed system oscillator ready flag which indicates when the low speed system oscillator is stable after power on reset or a wake-up has occurred. The flag will be low when in the SLEEP0 Mode but after a wake-up has occurred, the flag will change to a high level after 1024 clock cycles if the LXT oscillator is used and 1~2 clock cycles if the LIRC oscillator is used.

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

This is the high speed system oscillator ready flag which indicates when the high speed system oscillator is stable. This flag is cleared to "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. The flag will be low when in the SLEEP or IDLE0 Mode but after a wake-up has occurred, the flag will change to a high level after 1024 clock cycles if the HXT oscillator is used and after 15~16 clock cycles if the ERC or HIRC oscillator is used.

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

Bit 0 **HLCLK**: system clock selection
0: $f_H/2 \sim f_H/64$ or f_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.

Fast Wake-up

To minimise power consumption the device can enter the SLEEP or IDLE0 Mode, where the system clock source to the device will be stopped. However when the device is woken up again, it can take a considerable time for the original system oscillator to restart, stabilise and allow normal operation to resume. To ensure the device is up and running as fast as possible a Fast Wake-up function is provided, which allows f_{SUB}, namely either the LXT or LIRC oscillator, to act as a temporary clock to first drive the system until the original system oscillator has stabilised. As the clock source for the Fast Wake-up function is f_{SUB}, the Fast Wake-up function is only available in the SLEEP1 and IDLE0 modes. When the device is woken up from the SLEEP0 mode, the Fast Wake-up function has no effect because the f_{SUB} clock is stopped. The Fast Wake-up enable/disable function is controlled using the FSTEN bit in the SMOD0 register.

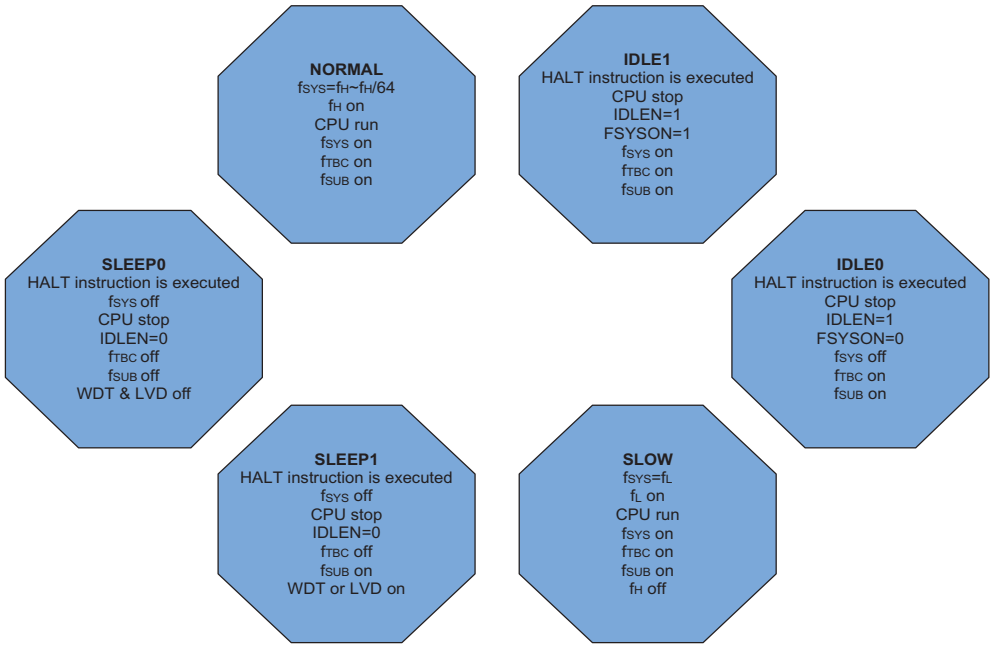
If the HXT oscillator is selected as the NORMAL Mode system clock, and if the Fast Wake-up function is enabled, then it will take one to two t_{SUB} clock cycles for the system to wake-up. The system will then initially run under the f_{SUB} clock source until 1024 HXT clock cycles have elapsed, at which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

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

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

Wake-Up Times

Note that in SLEEP0 mode the Watchdog Timer is disabled, which means that the LXT and LIRC are all both off, then there will be no Fast Wake-up function available when the device wake-up from the SLEEP0 Mode.



Operating Mode Switching

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

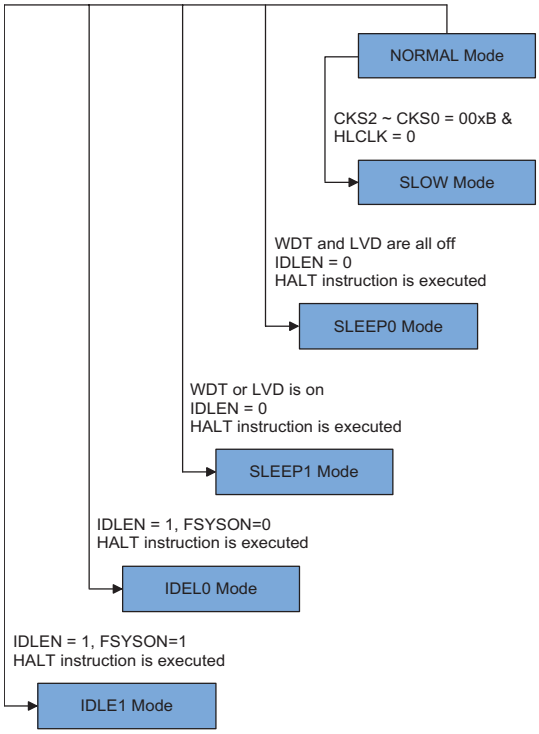
In simple terms, Mode Switching between the NORMAL Mode and SLOW Mode is executed using the HLCLK bit and CKS2~CKS0 bits in the SMOD0 register while Mode Switching from the NORMAL/SLOW Modes to the SLEEP/IDLE Modes is executed via the HALT instruction. When a HALT instruction is executed, whether the device enters the IDLE Mode or the SLEEP Mode is determined by the condition of the IDLEN bit in the SMOD0 register and FSYSON in the SMOD1 register.

When the HLCLK bit switches to a low level, which implies that clock source is switched from the high speed clock source, f_H, to the clock source, $f_H/2$~$f_H/64$ or f_L. If the clock is from the f_L, the high speed clock source will stop running to conserve power. When this happens it must be noted that the $f_H/16$ and $f_H/64$ internal clock sources will also stop running, which may affect the operation of other internal functions such as the TMs and the SIM. The accompanying flowchart shows what happens when the device moves between the various operating modes.

NORMAL Mode to SLOW Mode Switching

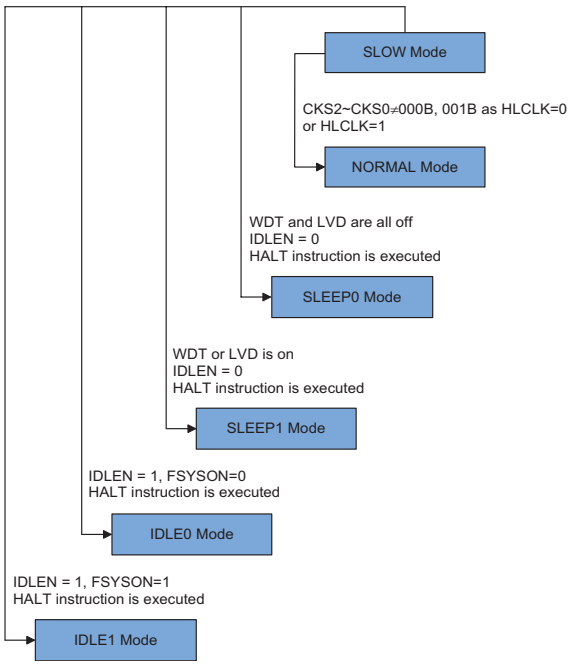
When running in the NORMAL Mode, which uses the high speed system oscillator, and therefore consumes more power, the system clock can switch to run in the SLOW Mode by set the HLCLK bit to "0" and set the CKS2~CKS0 bits to "000" or "001" in the SMOD0 register. This will then use the low speed system 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 LXT or the LIRC oscillator and therefore requires the selected oscillator to be stable before full mode switching occurs. This is monitored using the LTO bit in the SMOD0 register.



SLOW Mode to NORMAL Mode Switching

In SLOW Mode the system uses either the LXT or 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. The amount of time required for high speed system oscillator stabilization depends upon which high speed system oscillator type is used.



Entering the SLEEP0 Mode

There is only one way for the device to enter the SLEEP0 Mode and that is to execute the "HALT" instruction in the application program with the IDLEN bit in SMOD0 register equal to "0" and the WDT and LVD both off. When this instruction is executed under the conditions described above, the following will occur:

- The system clock, WDT clock and Time Base clock will be stopped and the application program will stop at the "HALT" instruction.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and stopped no matter if the WDT clock source originates from the f_{SUB} clock or from the system clock.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Entering the SLEEP1 Mode

There is only one way for the device to enter the SLEEP1 Mode and that is to execute the "HALT" instruction in the application program with the IDLEN bit in SMOD0 register equal to "0" and the WDT or LVD on. When this instruction is executed under the conditions described above, the following will occur:

- The system clock and Time Base clock will be stopped and the application program will stop at the "HALT" instruction, but the WDT or LVD will remain with the clock source coming from the f_{SUB} clock.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting if the WDT clock source is selected to come from the f_{SUB} clock as the WDT is enabled.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Entering the IDLE0 Mode

There is only one way for the device to enter the IDLE0 Mode and that is to execute the "HALT" instruction in the application program with the IDLEN bit in SMOD0 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 and f_{SUB} clock will be on.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting if the WDT clock source is selected to come from the f_{SUB} clock and the WDT is enabled. The WDT will stop if its clock source originates from the system clock.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Entering the IDLE1 Mode

There is only one way for the device to enter the IDLE1 Mode and that is to execute the "HALT" instruction in the application program with the IDLEN bit in SMOD0 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 and f_{SUB} clock will be on and the application program will stop at the "HALT" instruction.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting if the WDT is enabled regardless of the WDT clock source which originates from the f_{SUB} clock or from the system clock.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Standby Current Considerations

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

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

Wake-up

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

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

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

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

Programming Considerations

The HXT and LXT oscillators both use the same SST counter. For example, if the system is woken up from the SLEEP0 Mode and both the HXT and LXT oscillators need to start-up from an off state. The LXT oscillator uses the SST counter after HXT oscillator has finished its SST period.

- If the device is woken up from the SLEEP0 Mode to the NORMAL Mode, the high speed system oscillator needs an SST period. The device will execute first instruction after HTO is "1". At this time, the LXT oscillator may not be stability if f_{SUB} is from LXT oscillator. The same situation occurs in the power-on state. The LXT oscillator is not ready yet when the first instruction is executed.
- If the device is woken up from the SLEEP1 Mode to NORMAL Mode, and the system clock source is from HXT oscillator and FSTEN is "1", the system clock can be switched to the LXT or LIRC oscillator after wake up.
- There are peripheral functions, such as WDT, TMs and SPI1, LCD driver and SIM, for which the f_{SYS} is used. If the system clock source is switched from f_H to f_L, the clock source to the peripheral functions mentioned above will change accordingly.
- The on/off condition of f_{SUB} and f_S depends upon whether the WDT is enabled or disabled as the WDT clock source is selected from f_{SUB}.

System Clock Output (HT45F66/HT45F67)

There is a system clock output for peripheral application, please refer to control register for detail.

• SCKC Register

Bit	7	6	5	4	3	2	1	0
Name	SCKEN	—	—	—	—	—	SCKS1	SCKS0
R/W	R/W	—	—	—	—	—	R/W	R/W
POR	0	—	—	—	—	—	0	0

Bit 7 **SCKEN**: System clock output control
0: disabled
1: enabled

Bit 6~2 Unimplemented, read as "0"

Bit 1~0 **SCKS1, SCKS0**: System clock output selection
00: f_{SYS}
01: $f_{SYS}/2$
10: $f_{SYS}/4$
11: $f_{SYS}/8$

Watchdog Timer

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

Watchdog Timer Clock Source

The Watchdog Timer clock source is provided by the internal clock, f_S, which is in turn supplied by one of two sources selected by configuration option: f_{SUB} or $f_{SYS}/4$. The f_{SUB} clock can be sourced from either the LXT or LIRC oscillator, again chosen via a configuration option. The Watchdog Timer source clock is then subdivided by a ratio of 2^8 to 2^{18} to give longer timeouts, the actual value being chosen using the WS2~WS0 bits in the WDTC register. The LIRC internal oscillator has an approximate period of 32kHz at a supply voltage of 5V. However, it should be noted that this specified internal clock period can vary with V_{DD}, temperature and process variations. The LXT oscillator is supplied by an external 32.768kHz crystal.

Watchdog Timer Control Register

The WDT function can be always enabled or controlled by the WDTC register, which is selected by configuration option. A single register, WDTC, controls the required timeout period as well as the enable/disable operation. This register together with several configuration options control the overall operation of the Watchdog Timer.

- **WDTC Register**

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

Bit 7~3 **WE4~WE0**: WDT enable
10101: disable
01010: enable (default)
Other values: MCU reset (reset will be active after 2~3 LIRC clock for debounce time)
If the MCU reset is caused by the WE4~WE0 bits in the WDTC register, the WRF flag of SMOD1 register will be set.

Bit 2~0 **WS2~WS0**: select WDT timeout period
000: $2^8/f_S$
001: $2^{10}/f_S$
010: $2^{12}/f_S$
011: $2^{14}/f_S$ (default)
100: $2^{15}/f_S$
101: $2^{16}/f_S$
110: $2^{17}/f_S$
111: $2^{18}/f_S$

• **SMOD1 Register**

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

"x" unknown

Bit 7 **FSYSON**: f_{SYS} Control in IDLE Mode
0: disable
1: enable

Bit 6~3 Unimplemented, read as "0"

Bit 2 **LVRF**: reset caused by LVR function activation
Described elsewhere.

Bit 1 **LRF**: reset caused by LVRC setting
Described elsewhere.

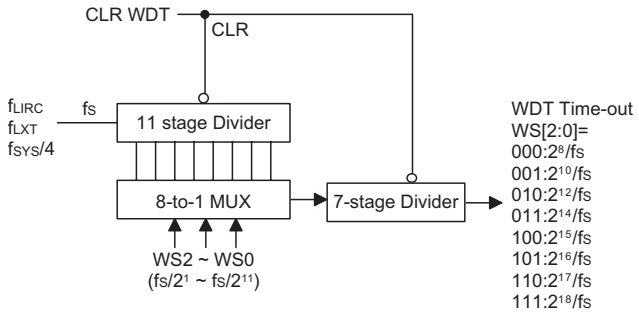
Bit 0 **WRF**: reset caused by WE[4:0] setting
0: not active
1: active
This bit can be clear to "0", but can not set to "1".

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.

Under normal program operation, a Watchdog Timer time-out will initialise a device reset and set the status bit TO. However, if the system is in the SLEEP or IDLE Mode, when a Watchdog Timer time-out occurs, the TO bit in the status register will be set and only the Program Counter and Stack Pointer will be reset. Three methods can be adopted to clear the contents of the Watchdog Timer. The second is using the Watchdog Timer software clear instructions and the third is via a HALT instruction.

To clear the Watchdog Timer is to use the single "CLR WDT" instruction. A simple execution of "CLR WDT" will clear the WDT.



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.

In addition to the power-on reset, situations may arise where it is necessary to forcefully apply a reset condition when the microcontroller is running. One example of this is where after power has been applied and the microcontroller is already running, the $\overline{\text{RES}}$ line is forcefully pulled low. In such a case, known as a normal operation reset, some of the registers remain unchanged allowing the microcontroller to proceed with normal operation after the reset line is allowed to return high.

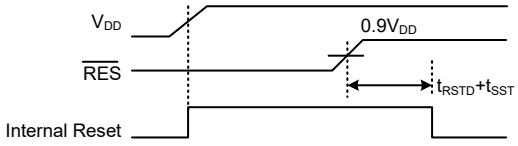
Another type of reset is when the Watchdog Timer overflows and resets. All types of reset operations result in different register conditions being setup. Another reset exists in the form of a Low Voltage Reset, LVR, where a full reset, similar to the $\overline{\text{RES}}$ reset is implemented in situations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

Reset Functions

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

Power-on Reset

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



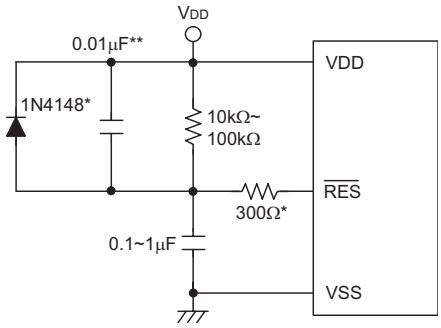
Power-On Reset Timing Chart

$\overline{\text{RES}}$ Pin Reset

Although the has an internal RC reset function, if the V_{DD} power supply rise time is not fast enough or does not stabilise quickly at power-on, the internal reset function may be incapable of providing proper reset operation. For this reason it is recommended that an external RC network is connected to the $\overline{\text{RES}}$ pin, whose additional time delay will ensure that the $\overline{\text{RES}}$ pin remains low for an extended period to allow the power supply to stabilise. During this time delay, normal operation of the will be inhibited. After the $\overline{\text{RES}}$ line reaches a certain voltage value, the reset delay time t_{RSTD} is invoked to provide an extra delay time after which the will begin normal operation. The abbreviation SST in the figures stands for System Start-up Timer.

For most applications a resistor connected between VDD and the $\overline{\text{RES}}$ pin and a capacitor connected between VSS and the $\overline{\text{RES}}$ pin will provide a suitable external reset circuit. Any wiring connected to the $\overline{\text{RES}}$ pin should be kept as short as possible to minimise any stray noise interference.

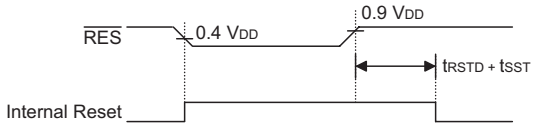
For applications that operate within an environment where more noise is present the Enhanced Reset Circuit shown is recommended.



External $\overline{RES}$ Circuit

Note: * It is recommended that this component is added for added ESD protection.
** It is recommended that this component is added in environments where power line noise is significant.

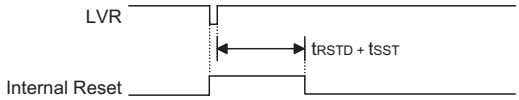
Pulling the $\overline{RES}$ Pin low using external hardware will also execute a device reset. In this case, as in the case of other resets, the Program Counter will reset to zero and program execution initiated from this point.



$\overline{RES}$ Reset Timing Chart

Low Voltage Reset – LVR

The microcontroller contains a low voltage reset circuit in order to monitor the supply voltage of the device. If the supply voltage of the device drops to within a range of 0.9V~V_{LVR} such as might occur when changing the battery, the LVR will automatically reset the device internally and the LVRF bit in the SMOD1 register will also be set to 1. The LVR includes the following specifications: For a valid LVR signal, a low voltage, i.e., a voltage in the range between 0.9V~V_{LVR} must exist for greater than the value t_{LVR} specified in the A.C. characteristics. If the low voltage state does not exceed t_{LVR}, the LVR will ignore it and will not perform a reset function.



Low Voltage Reset Timing Chart

- **LVRC Register**

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

Bit 7~0 **LVS7~LVS0**: LVR voltage select
01010101: 2.1V (default)
00110011: 2.55V
10011001: 3.15V
10101010: 3.8V
Other values: MCU reset (reset will be active after 2~3 LIRC clock for debounce time).
When an actual low voltage condition occurs, as specified by one of the four defined

LVR voltage values above, an MCU reset will be generated. In this situation the register contents will remain the same after such a reset occurs.

Any register value, other than the four defined LVR values above, will also result in the generation of an MCU reset. The reset operation will be activated after 2~3 LIRC clock cycles. However in this situation the register contents will be reset to the POR value.

- **SMOD1 Register**

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

"x" unknown

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

Bit 6~3 Unimplemented, read as "0"

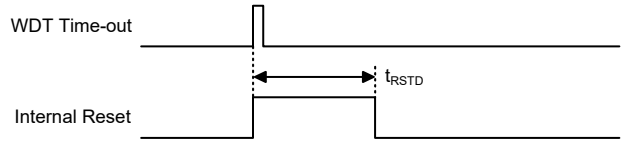
Bit 2 **LVRF**: reset caused by LVR function activation
0: not active
1: active
This bit can be cleared to "0", but can not set to "1".by software

Bit 1 **LRF**: reset caused by LVRC setting
0: not active
1: active
This bit can be cleared to "0", but can not set to "1".by software

Bit 0 **WRF**: reset caused by WE[4:0] setting
Described elsewhere.

Watchdog Time-out Reset during Normal Operation

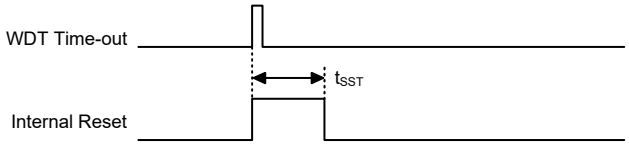
The Watchdog time-out Reset during normal operation is the same as a hardware $\overline{RES}$ pin reset except that the Watchdog time-out flag TO will be set to "1".



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 Timing Chart

Note: The t_{SST} is 15~16 clock cycles if the system clock source is provided by ERC or HIRC. The t_{SST} is 1024 clock for HXT or LXT. The t_{SST} is 1~2 clock for LIRC.

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	$\overline{\text{RES}}$ or LVR reset during Normal or SLOW Mode operation
1	u	WDT time-out reset during Normal or SLOW Mode operation
1	1	WDT time-out reset during IDLE or SLEEP Mode operation

Note: "u" stands for unchanged

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

Item	Condition after Reset
Program Counter	Reset to zero
Interrupts	All interrupts will be disabled
WDT	Clear after reset, WDT begins counting
Timer/Event Counter	Timer Counter will be turned off
Input/Output Ports	I/O ports will be setup as inputs
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.

HT45F65

Register Name	Power On Reset	External Reset	WDT Reset from Normal Operation State	WDT Reset from Halt State
IAR0	---- ----	---- ----	---- ----	---- ----
MP0	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
IAR1	---- ----	---- ----	---- ----	---- ----
MP1	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
BP	---- --00	---- --00	---- --00	---- --uu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	---x xxxx	---u uuuu	---u uuuu	---u uuuu
STATUS	--00 xxxx	--uu uuuu	--1u uuuu	--11 uuuu
SMOD0	0000 0011	0000 0011	0000 0011	uuuu uuuu
LVDC	--00 -000	--00 -000	--00 -000	--uu -uuu
INTEG	---- 0000	---- 0000	---- 0000	---- uuuu
WDTC	0101 0011	0101 0011	0101 0011	Uuuu uuuu
TBC	0011 0111	0011 0111	0011 0111	uuuu uuuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	-000 -000	-000 -000	-000 -000	-uuu -uuu

Register Name	Power On Reset	External Reset	WDT Reset from Normal Operation State	WDT Reset from Halt State
INTC2	--00 --00	--00 --00	--00 --00	--uu --uu
SCKC	0--- --00	0--- --00	0--- --00	u--- --00
MFI0	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI1	-000 0000	-000 0000	-000 0000	-uuu uuuu
MFI2	0000 0000	0000 0000	0000 0000	Uuuu uuuu
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCPU	---0 0000	---0 0000	---0 0000	---u uuuu
PC	---1 1111	---1 1111	---1 1111	---u uuuu
PCC	---1 1111	---1 1111	---1 1111	---u uuuu
PDPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PE	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAFS	000- ----	000- ----	000- ----	uuu- ----
PBFS	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCFS	---0 0000	---0 0000	---0 0000	---u uuuu
PDFS0	-000 0000	-000 0000	-000 0000	-uuu uuuu
PDFS1	---0 0000	---0 0000	---0 0000	---u uuuu
PEFS0	-000 0000	-000 0000	-000 0000	-uuu uuuu
PEFS1	---0 0000	---0 0000	---0 0000	---u uuuu
PRES	---- ---0	---- ---0	---- ---0	---- ---u
SMOD1	0--- -x00	0--- -x00	0--- -x00	u--- -uuu
LVRC	0101 0101	0101 0101	0101 0101	uuuu uuuu
ADRL	xxxx ----	xxxx ----	xxxx ----	uuuu ----
USR	0000 1011	0000 1011	0000 1011	uuuu uuuu
UCR1	0000 00x0	0000 00x0	0000 00x0	uuuu uuuu
UCR2	0000 0000	0000 0000	0000 0000	uuuu uuuu
BRG	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TXRRXR	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRH	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCR0	011- -000	011- -000	011- -000	uuu- -uuu
ADCR1	---- -000	---- -000	---- -000	---- -uuu
ADCR2	---- 0000	---- 0000	---- 0000	---- uuuu
SIMC0	1110 0000	1110 0000	1110 0000	uuuu uuuu
SIMC1	1000 0001	1000 0001	1000 0001	uuuu uuuu
SIMD	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SIMA	xxxx xxx-	xxxx xxx-	xxxx xxx-	uuuu uuu-
SIMC2	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0C0	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0C1	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register Name	Power On Reset	External Reset	WDT Reset from Normal Operation State	WDT Reset from Halt State
TM0DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0DH	---- --00	---- --00	---- --00	---- --uu
TM0AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0AH	---- --00	---- --00	---- --00	---- --uu
TM2C0	0000 0---	0000 0---	0000 0---	uuuu u---
TM2C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2DH	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2AH	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2RP	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1C0	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1DH	---- --00	---- --00	---- --00	---- --uu
TM1AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1AH	---- --00	---- --00	---- --00	---- --uu
LCDC	000- ---0	000- ---0	000- ---0	uuu- ---u
SPI1C0	111- 0-00	111- 0-00	111- 0-00	uuu- u-uu
SPI1C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
SPI1D	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADAC	000- ---0	000- ---0	000- ---0	uuu- ---u
ADAL	0000 ----	0000 ----	0000 ----	uuuu ----
ADAH	0000 0000	0000 0000	0000 0000	uuuu uuuu
BGC	---0 --00	---0 --00	---0 --00	---u --uu
DAC	---- ---0	---- ---0	---- ---0	---- ---u
DAL	00-- ----	00-- ----	00-- ----	uu-- ----
DAH	0000 0000	0000 0000	0000 0000	0000 0000
TSC	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
PVREF	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARL	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARH	---0 0000	---0 0000	---0 0000	---u uuuu
FD0L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD0H	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD1L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD1H	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2H	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3H	0000 0000	0000 0000	0000 0000	uuuu uuuu
OPC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
OPC2	0000 0000	0000 0000	0000 0000	uuuu uuuu

Note: "u" stands for unchanged
"x" stands for unknown
"-" stands for unimplemented

HT45F66

Register Name	Power On Reset	$\overline{RES}$ Reset	WDT Time-out (Normal Operation)	WDT Time-out (Halt)
IAR0	---- ----	---- ----	---- ----	---- ----
MP0	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
IAR1	---- ----	---- ----	---- ----	---- ----
MP1	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
BP	--0- --00	--0- --00	--0- --00	--u- --uu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	--xx xxxx	--uu uuuu	--uu uuuu	--uu uuuu
STATUS	--00 xxxx	--uu uuuu	--1u uuuu	--11 uuuu
SMOD0	0000 0011	0000 0011	0000 0011	uuuu uuuu
LVDC	--00 -000	--00 -000	--00 -000	--uu -uuu
INTEG	---- 0000	---- 0000	---- 0000	---- uuuu
WDTC	0101 0011	0101 0011	0101 0011	uuuu uuuu
TBC	0011 0111	0011 0111	0011 0111	uuuu uuuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC2	--00 --00	--00 --00	--00 --00	--uu --uu
SCKC	0--- --00	0--- --00	0--- --00	u--- --00
MFI0	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI1	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI2	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI3	--00 --00	--00 --00	--00 --00	--uu --uu
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCPU	--00 0000	--00 0000	--00 0000	--uu uuuu
PC	--11 1111	--11 1111	--11 1111	--uu uuuu
PCC	--11 1111	--11 1111	--11 1111	--uu uuuu
PDPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PE	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PF	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PGPU	--00 0000	--00 0000	--00 0000	--uu uuuu
PG	--11 1111	--11 1111	--11 1111	--uu uuuu
PGC	--11 1111	--11 1111	--11 1111	--uu uuuu

Register Name	Power On Reset	$\overline{\text{RES}}$ Reset	WDT Time-out (Normal Operation)	WDT Time-out (Halt)
PHPU	-000 0000	-000 0000	-000 0000	-uuu uuuu
PH	-111 1111	-111 1111	-111 1111	-uuu uuuu
PHC	-111 1111	-111 1111	-111 1111	-uuu uuuu
PAFS	00-- ----	00-- ----	00-- ----	uu-- ----
PBFS	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCFS	--00 0000	--00 0000	--00 0000	--uu uuuu
PDFS	0000 0000	0000 0000	0000 0000	uuuu uuuu
PEFS	0000 0000	0000 0000	0000 0000	uuuu uuuu
PFFS	0000 0000	0000 0000	0000 0000	uuuu uuuu
PGFS	--00 0000	--00 0000	--00 0000	--uu uuuu
PHFS	---- ---0	---- ---0	---- ---0	---- ---u
SFS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SFS1	0000 0000	0000 0000	0000 0000	uuuu uuuu
SMOD1	0--- -x00	0--- -x00	0--- -x00	u--- -uuu
LVRC	0101 0101	0101 0101	0101 0101	uuuu uuuu
ADRL	xxxx ----	xxxx ----	xxxx ----	uuuu ----
USR	0000 1011	0000 1011	0000 1011	uuuu uuuu
UCR1	0000 00x0	0000 00x0	0000 00x0	uuuu uuuu
UCR2	0000 0000	0000 0000	0000 0000	uuuu uuuu
BRG	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TXR/RXR	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRH	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCR0	011- -000	011- -000	011- -000	uuu- -uuu
ADCR1	---- -000	---- -000	---- -000	---- -uuu
ADCR2	---- 0000	---- 0000	---- 0000	---- uuuu
SIMC0	1110 0000	1110 0000	1110 0000	uuuu uuuu
SIMC1	1000 0001	1000 0001	1000 0001	uuuu uuuu
SIMD	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SIMA	xxxx xxx-	xxxx xxx-	xxxx xxx-	uuuu uuu-
SIMC2	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0C0	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0DH	---- --00	---- --00	---- --00	---- --uu
TM0AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0AH	---- --00	---- --00	---- --00	---- --uu
TM1C0	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1C2	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1DH	---- --00	---- --00	---- --00	---- --uu
TM1AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1AH	---- --00	---- --00	---- --00	---- --uu
TM1BL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1BH	---- --00	---- --00	---- --00	---- --uu
TM2C0	0000 0---	0000 0---	0000 0---	uuuu u---
TM2C1	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register Name	Power On Reset	$\overline{\text{RES}}$ Reset	WDT Time-out (Normal Operation)	WDT Time-out (Halt)
TM2DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2DH	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2AH	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2RP	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3C0	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3DH	---- --00	---- --00	---- --00	---- --uu
TM3AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3AH	---- --00	---- --00	---- --00	---- --uu
LCDC	000- ---0	000- ---0	000- ---0	uuu- ---u
SPI1C0	1110 0000	1110 0000	1110 0000	uuuu uuuu
SPI1C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
SPI1D	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADAC	000- ---0	000- ---0	000- ---0	uuu- ---u
ADAL	0000 ----	0000 ----	0000 ----	uuuu ----
ADAH	0000 0000	0000 0000	0000 0000	uuuu uuuu
BGC	---0 --00	---0 --00	---0 --00	---u --uu
DAC	---- ---0	---- ---0	---- ---0	---- ---u
DAL	00-- ----	00-- ----	00-- ----	uu-- ----
DAH	0000 0000	0000 0000	0000 0000	0000 0000
TSC	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
FARL	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARH	--00 0000	--00 0000	--00 0000	--uu uuuu
FD0L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD0H	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD1L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD1H	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2H	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3H	0000 0000	0000 0000	0000 0000	uuuu uuuu
PVREF	0000 0000	0000 0000	0000 0000	uuuu uuuu
OPC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
OPC2	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC2	---- ---0	---- ---0	---- ---0	---- ---u

Note: "u" stands for unchanged
"x" stands for unknown
"-" stands for unimplemented

HT45F67

Register Name	Power On Reset	$\overline{\text{RES}}$ Reset	WDT Time-out (Normal Operation)	WDT Time-out (Halt)
IAR0	---- ----	---- ----	---- ----	---- ----
MP0	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
IAR1	---- ----	---- ----	---- ----	---- ----
MP1	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
BP	-00- -000	-00- -000	-00- -000	-uu- -uuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	-xxx xxxx	-uuu uuuu	-uuu uuuu	-uuu uuuu
STATUS	--00 xxxx	--uu uuuu	--1u uuuu	--11 uuuu
SMOD0	0000 0011	0000 0011	0000 0011	uuuu uuuu
LVDC	--00 -000	--00 -000	--00 -000	--uu -uuu
INTEG	---- 0000	---- 0000	---- 0000	---- uuuu
WDTC	0101 0011	0101 0011	0101 0011	uuuu uuuu
TBC	0011 0111	0011 0111	0011 0111	uuuu uuuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC2	--00 --00	--00 --00	--00 --00	--uu --uu
SCKC	0--- --00	0--- --00	0--- --00	u--- --00
MFI0	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI1	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI2	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI3	--00 --00	--00 --00	--00 --00	--uu --uu
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCPU	--00 0000	--00 0000	--00 0000	--uu uuuu
PC	--11 1111	--11 1111	--11 1111	--uu uuuu
PCC	--11 1111	--11 1111	--11 1111	--uu uuuu
PDPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PE	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PF	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PGPU	--00 0000	--00 0000	--00 0000	--uu uuuu
PG	--11 1111	--11 1111	--11 1111	--uu uuuu
PGC	--11 1111	--11 1111	--11 1111	--uu uuuu

Register Name	Power On Reset	$\overline{\text{RES}}$ Reset	WDT Time-out (Normal Operation)	WDT Time-out (Halt)
PHPU	-000 0000	-000 0000	-000 0000	-uuu uuuu
PH	-111 1111	-111 1111	-111 1111	-uuu uuuu
PHC	-111 1111	-111 1111	-111 1111	-uuu uuuu
PAFS	00-- ----	00-- ----	00-- ----	uu-- ----
PBFS	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCFS	--00 0000	--00 0000	--00 0000	--uu uuuu
PDFS	0000 0000	0000 0000	0000 0000	uuuu uuuu
PEFS	0000 0000	0000 0000	0000 0000	uuuu uuuu
PFFS	0000 0000	0000 0000	0000 0000	uuuu uuuu
PGFS	--00 0000	--00 0000	--00 0000	--uu uuuu
PHFS	---- ---0	---- ---0	---- ---0	---- ---u
SFS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SFS1	0000 0000	0000 0000	0000 0000	uuuu uuuu
SMOD1	0--- -x00	0--- -x00	0--- -x00	u--- -uuu
LVRC	0101 0101	0101 0101	0101 0101	uuuu uuuu
ADRL	xxxx ----	xxxx ----	xxxx ----	uuuu ----
USR	0000 1011	0000 1011	0000 1011	uuuu uuuu
UCR1	0000 00x0	0000 00x0	0000 00x0	uuuu uuuu
UCR2	0000 0000	0000 0000	0000 0000	uuuu uuuu
BRG	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TXR/RXR	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRH	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCR0	011- -000	011- -000	011- -000	uuu- -uuu
ADCR1	---- -000	---- -000	---- -000	---- -uuu
ADCR2	---- 0000	---- 0000	---- 0000	---- uuuu
SIMC0	1110 0000	1110 0000	1110 0000	uuuu uuuu
SIMC1	1000 0001	1000 0001	1000 0001	uuuu uuuu
SIMD	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SIMA	xxxx xxx-	xxxx xxx-	xxxx xxx-	uuuu uuu-
SIMC2	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0C0	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0DH	---- --00	---- --00	---- --00	---- --uu
TM0AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0AH	---- --00	---- --00	---- --00	---- --uu
TM1C0	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1C2	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1DH	---- --00	---- --00	---- --00	---- --uu
TM1AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1AH	---- --00	---- --00	---- --00	---- --uu
TM1BL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1BH	---- --00	---- --00	---- --00	---- --uu
TM2C0	0000 0---	0000 0---	0000 0---	uuuu u---
TM2C1	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register Name	Power On Reset	$\overline{\text{RES}}$ Reset	WDT Time-out (Normal Operation)	WDT Time-out (Halt)
TM2DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2DH	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2AH	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2RP	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3C0	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3DH	---- --00	---- --00	---- --00	---- --uu
TM3AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3AH	---- --00	---- --00	---- --00	---- --uu
LCDC	000- ---0	000- ---0	000- ---0	uuu- ---u
SPI1C0	1110 0000	1110 0000	1110 0000	uuuu uuuu
SPI1C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
SPI1D	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADAC	000- ---0	000- ---0	000- ---0	uuu- ---u
ADAL	0000 ----	0000 ----	0000 ----	uuuu ----
ADAH	0000 0000	0000 0000	0000 0000	uuuu uuuu
BGC	---0 --00	---0 --00	---0 --00	---u --uu
DAC	---- ---0	---- ---0	---- ---0	---- ---u
DAL	00-- ----	00-- ----	00-- ----	uu-- ----
DAH	0000 0000	0000 0000	0000 0000	0000 0000
TSC	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
FARL	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARH	-000 0000	-000 0000	-000 0000	-0uu uuuu
FD0L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD0H	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD1L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD1H	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2H	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3H	0000 0000	0000 0000	0000 0000	uuuu uuuu
PVREF	0000 0000	0000 0000	0000 0000	uuuu uuuu
OPC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
OPC2	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC2	---- ---0	---- ---0	---- ---0	---- ---u

Note: "u" stands for unchanged
"x" stands for unknown
"-" stands for unimplemented

Input/Output Ports

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

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

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PBPU	D7	D6	D5	D4	PBPU3	PBPU2	D1	D0
PB	D7	D6	D5	D4	PB3	PB2	D1	D0
PBC	D7	D6	D5	D4	PBC3	PBC2	D1	D0
PCPU	—	—	—	PCPU4	PCPU3	PCPU2	D1	D0
PC	—	—	—	PC4	PC3	PC2	D1	D0
PCC	—	—	—	PCC4	PCC3	PCC2	D1	D0
PDPU	PDPU7	PDPU6	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
PD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
PDC	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
PEPU	PEPU7	PEPU6	PEPU5	PEPU4	PEPU3	PEPU2	PEPU1	PEPU0
PE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
PEC	PEC7	PEC6	PEC5	PEC4	PEC3	PEC2	PEC1	PEC0

"—": Unimplemented, read as "0"

I/O Logic Function Register List – HT45F65

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PBPU	PBPU7	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PB	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PBC	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PCPU	—	—	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PC	—	—	PC5	PC4	PC3	PC2	PC1	PC0
PCC	—	—	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PDPU	PDPU7	PDPU6	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
PD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
PDC	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
PEPU	PEPU7	PEPU6	PEPU5	PEPU4	PEPU3	PEPU2	PEPU1	PEPU0

Register Name	Bit							
	7	6	5	4	3	2	1	0
PE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
PEC	PEC7	PEC6	PEC5	PEC4	PEC3	PEC2	PEC1	PEC0
PFPU	PFPU7	PFPU6	PFPU5	PFPU4	PFPU3	PFPU2	PFPU1	PFPU0
PF	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
PFC	PFC7	PFC6	PFC5	PFC4	PFC3	PFC2	PFC1	PFC0
PGPU	—	—	PGPU5	PGPU4	PGPU3	PGPU2	PGPU1	PGPU0
PG	—	—	PG5	PG4	PG3	PG2	PG1	PG0
PGC	—	—	PGC5	PGC4	PGC3	PGC2	PGC1	PGC0
PHPU	—	PHPU6	PHPU5	PHPU4	PHPU3	PHPU2	PHPU1	PHPU0
PH	—	PH6	PH5	PH4	PH3	PH2	PH1	PH0
PHC	—	PHC6	PHC5	PHC4	PHC3	PHC2	PHC1	PHC0

"—": Unimplemented, read as "0"

I/O Logic Function Register List – HT45F66/HT45F67

Pull-high Resistors

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

- **PxPU Register**

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

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

The PxPUn bit is used to control the pin pull-high function. Here the "x" is the Port name which can be A, B, C, D, E or F. However, the actual available bits for each I/O Port may be different.

For the HT45F65, the pull-high control bit denoted as "Dn" in the ports B and C should remain the POR value after power-on reset. This is also applied to the unbonded I/O pins in the smaller package of the HT45F66/HT45F67.

Port A Wake-up

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

• **PAWU Register**

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

Bit 7~0 **PAWU7~PAWU0**: 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~PHC, to control the input/output configuration. With this control register, each CMOS output or input can be reconfigured dynamically under software control. Each pin of the I/O ports is directly mapped to a bit in its associated port control register. For the I/O pin to function as an input, the corresponding bit of the control register must be written as a "1". This will then allow the logic state of the input pin to be directly read by instructions. When the corresponding bit of the control register is written as a "0", the I/O pin will be setup as a CMOS output. If the pin is currently setup as an output, instructions can still be used to read the output register. However, it should be noted that the program will in fact only read the status of the output data latch and not the actual logic status of the output pin.

• **PxC Register**

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

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

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

For the HT45F65, the port control bit denoted as “Dn” in the ports B and C should be cleared to 0 to set the corresponding pin as an output after power-on reset, which is also applied to the unbonded I/O pins in the smaller package of the HT45F66/HT45F67. This can prevent the device from consuming power due to input floating states for any unbonded pins.

Pin-remapping 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. The way in which the pin function of each pin is selected is different for each function and a priority order is established where more than one pin function is selected simultaneously. Additionally there are a series of PAFS, PBFS, PCFS, PDFS, PEFS, PFFS, PGFS, PHFS, SFS0 and SFS1 registers to establish certain pin functions.

Pin-remapping Registers

The limited number of supplied pins in a package can impose restrictions on the amount of functions a certain device can contain. However by allowing the same pins to share several different functions and providing a means of function selection, a wide range of different functions can be incorporated into even relatively small package sizes. The device includes PAFS, PBFS, PCFS, PDFS, PEFS, PFFS, PGFS, PHFS, SFS0 and SFS1 registers which can select the functions of certain pins.

- **Pin-remapping Register List (HT45F65)**

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAFS	PAFS7	PAFS6	PAFS5	—	—	—	—	—
PBFS	D7	D6	D5	D4	PBFS3	PBFS2	D1	D0
PCFS	—	—	—	PCFS4	PCFS3	PCFS2	D1	D0
PDFS0	—	PDFS4	PDFS3	PDFS22	PDFS21	PDFS12	PDFS11	PDFS00
PDFS1	—	—	—	PDFS7	PDFS62	PDFS61	PDFS52	PDFS51
PEFS0	—	PEFS32	PEFS31	PEFS2	PEFS12	PEFS11	PEFS02	PEFS01
PEFS1	—	—	—	PEFS7	PEFS6	PEFS52	PEFS51	PEFS4

- **Pin-remapping Register List (HT45F66/HT45F67)**

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAFS	PAFS7	PAFS6	—	—	—	—	—	—
PBFS	PBFS7	PBFS6	PBFS5	PBFS4	PBFS3	PBFS2	PBFS1	PBFS0
PCFS	—	—	PCFS5	PCFS4	PCFS3	PCFS2	PCFS1	PCFS0
PDFS	PDFS7	PDFS6	PDFS5	PDFS4	PDFS3	PDFS2	PDFS1	PDFS0
PEFS	PEFS7	PEFS6	PEFS5	PEFS4	PEFS3	PEFS2	PEFS1	PEFS0
PFFS	PFFS7	PFFS6	PFFS5	PFFS	PFFS3	PFFS2	PFFS1	PFFS0
PGFS	—	—	PGFS5	PGFS4	PGFS3	PGFS2	PGFS1	PGFS0
PHFS	—	—	—	—	—	—	—	PHFS0
SFS0	SFS07	SFS06	SFS05	SFS04	SFS03	SFS02	SFS01	SFS00
SFS1	SFS17	SFS16	SFS15	SFS14	SFS13	SFS12	SFS11	SFS10

• **PAFS Register (HT45F65)**

Bit	7	6	5	4	3	2	1	0
Name	PAFS7	PAFS6	PAFS5	—	—	—	—	—
R/W	R/W	R/W	R/W	—	—	—	—	—
POR	0	0	0	—	—	—	—	—

Bit 7 **PAFS7**: Port A7 function selection
0: I/O
1: OP2N

Bit 6 **PAFS6**: Port A6 function selection
0: I/O
1: OP2O

Bit 5 **PAFS5**: Port A5 function selection
0: I/O
1: OP1N

Bit 4~0 Unimplemented, read as "0"

• **PAFS Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	PAFS7	PAFS6	—	—	—	—	—	—
R/W	R/W	R/W	—	—	—	—	—	—
POR	0	0	—	—	—	—	—	—

Bit 7 **PAFS7**: Port A7 function selection
0: I/O
1: special function (OP2N or INT1)

Bit 6 **PAFS6**: Port A6 function selection
0: I/O
1: special function (VG or INT0)

Bit 5~0 Unimplemented, read as "0"

• **PBFS Register (HT45F65)**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	PBFS3	PBFS2	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 7~4 **D7~D4**: Reserved bits, should remain unchanged after power-on reset

Bit 3 **PBFS3**: Port B3 function selection
0: I/O
1: AN3

Bit 2 **PBFS2**: Port B2 function selection
0: I/O
1: AN2

Bit 1~0 **D1~D0**: Reserved bits, should remain unchanged after power-on reset

• **PBFS Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	PBFS7	PBFS6	PBFS5	PBFS4	PBFS3	PBFS2	PBFS1	PBFS0
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 **PBFS7**: Port B7 function selection
0: I/O
1: ADVRL

Bit 6 **PBFS6**: Port B6 function selection
0: I/O
1: ADVRH

Bit 5 **PBFS5**: Port B5 function selection
0: I/O
1: AN3

Bit 4 **PBFS4**: Port B4 function selection
0: I/O
1: AN2

Bit 3 **PBFS3**: Port B3 function selection
0: I/O
1: AN1

Bit 2 **PBFS2**: Port B2 function selection
0: I/O
1: AN0

Bit 1 **PBFS1**: Port B1 function selection
0: I/O
1: OP1N

Bit 0 **PBFS0**: Port B0 function selection
0: I/O
1: OP2O

• **PCFS Register (HT45F65)**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	PCFS4	PCFS3	PCFS2	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 **PCFS4**: Port C4 function selection
0: I/O
1: VG

Bit 3 **PCFS3**: Port C3 function selection
0: I/O
1: TX

Bit 2 **PCFS2**: Port C2 function selection
0: I/O
1: RX

Bit 1~0 **D1~D0**: Reserved bits, should remain unchanged after power-on reset

• **PCFS Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCFS5	PCFS4	PCFS3	PCFS2	PCFS1	PCFS0
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 **PCFS5**: Port C5 function selection
0: I/O
1: AUD
Bit 4 **PCFS4**: Port C4 function selection
0: I/O
1: by configuration option
Bit 3 **PCFS3**: Port C3 function selection
0: I/O
1: by configuration option
Bit 2 **PCFS2**: Port C2 function selection
0: I/O
1: by configuration option
Bit 1 **PCFS1**: Port C1 function selection
0: I/O
1: by configuration option
Bit 0 **PCFS0**: Port C0 function selection
0: I/O
1: by configuration option

• **PDFS0 Register (HT45F65)**

Bit	7	6	5	4	3	2	1	0
Name	—	PDFS4	PDFS3	PDFS22	PDFS21	PDFS12	PDFS11	PDFS00
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 **PDFS4**: Port D4 function selection
0: I/O
1: SEG4
Bit 5 **PDFS3**: Port D3 function selection
0: I/O
1: SEG3
Bit 4~3 **PDFS22~ PDFS21**: Port D2 function selection
00: I/O
01: SEG2
Others: I/O
Bit 2~1 **PDFS12~ PDFS11**: Port D1 function selection
00: I/O
01: SEG1
Others: I/O
Bit 0 **PDFS00**: Port D0 function selection
0: I/O
1: SEG0

• **PDFS1 Register (HT45F65)**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	PDFS7	PDFS62	PDFS61	PDFS52	PDFS51
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 **PDFS7**: Port D7 function selection
0: I/O
1: SEG7

Bit 3~2 **PDFS62~ PDFS61**: Port D6 function selection
00: I/O
01: SEG6
Others: I/O

Bit 1~0 **PDFS52~ PDFS51**: Port D5 function selection
00: I/O
01: SEG5
Others: I/O

• **PDFS Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	PDFS7	PDFS6	PDFS5	PDFS4	PDFS3	PDFS2	PDFS1	PDFS0
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 **PDFS7**: Port D7 function selection
0: I/O
1: special function (SEG7 or TCK1)
This bit setting does not affect the TCK1 function.

Bit 6 **PDFS6**: Port D6 function selection
0: I/O
1: special function (SEG6 or TP1B_2)

Bit 5 **PDFS5**: Port D5 function selection
0: I/O
1: special function (SEG5 or TP1B_1)

Bit 4 **PDFS4**: Port D4 function selection
0: I/O
1: special function (SEG4 or TP1B_0)

Bit 3 **PDFS3**: Port D3 function selection
0: I/O
1: special function (SEG3 or TP1A)

Bit 2 **PDFS2**: Port D2 function selection
0: I/O
1: special function (SEG2 or TCK0)

Bit 1 **PDFS1**: Port D1 function selection
0: I/O
1: special function (SEG1 or TP0_1)

Bit 0 **PDFS0**: Port D0 function selection
0: I/O
1: special function (SEG0 or TP0_0)

• **PEFS0 Register (HT45F65)**

Bit	7	6	5	4	3	2	1	0
Name	—	PEFS32	PEFS31	PEFS2	PEFS12	PEFS11	PEFS02	PEFS01
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7 Unimplemented, read as "0"
Bit 6~5 **PEFS32~ PEFS31**: Port E3 function selection
00: I/O
01: SEG11
Others: TP1_0 or TP1_1
Bit 4 **PEFS2**: Port E2 function selection
0: I/O
1: SEG10
Bit 3~2 **PEFS12~ PEFS11**: Port E1 function selection
00: I/O
01: SEG9
Others: TP0_0 or TP0_1
Bit 1~0 **PEFS02~ PEFS01**: Port E0 function selection
00: I/O
01: SEG8
Others: I/O

• **PEFS1 Register (HT45F65)**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	PEFS7	PEFS6	PEFS52	PEFS51	PEFS4
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 **PEFS7**: Port E7 function selection
0: I/O
1: SEG15
Bit 3 **PEFS6**: Port E6 function selection
0: I/O
1: SEG14
Bit 2~1 **PEFS52~ PEFS51**: Port E5 function selection
00: I/O
01: SEG13
Others: TP2_0 or TP2_1
Bit 0 **PEFS4**: Port E4 function selection
0: I/O
1: SEG12

• **PEFS Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	PEFS7	PEFS6	PEFS5	PEFS4	PEFS3	PEFS2	PEFS1	PEFS0
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 **PEFS7**: Port E7 function selection
0: I/O
1: SEG15

Bit 6 **PEFS6**: Port E6 function selection
0: I/O
1: SEG14

Bit 5 **PEFS5**: Port E5 function selection
0: I/O
1: special function (SEG13 or TCK3)

Bit 4 **PEFS4**: Port E4 function selection
0: I/O
1: special function (SEG12 or TP3_1)

Bit 3 **PEFS3**: Port E3 function selection
0: I/O
1: special function (SEG11 or TP3_0)

Bit 2 **PEFS2**: Port E2 function selection
0: I/O
1: special function (SEG10 or TCK2)

Bit 1 **PEFS1**: Port E1 function selection
0: I/O
1: special function (SEG9 or TP2_1)

Bit 0 **PEFS0**: Port E0 function selection
0: I/O
1: special function (SEG8 or TP2_0)

• **PFFS Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	PFFS7	PFFS6	PFFS5	PFFS4	PFFS3	PFFS2	PFFS1	PFFS0
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 **PFFS7**: Port F7 function selection
0: I/O
1: SEG23

Bit 6 **PFFS6**: Port F6 function selection
0: I/O
1: SEG22

Bit 5 **PFFS5**: Port F5 function selection
0: I/O
1: SEG21

Bit 4 **PFFS4**: Port F4 function selection
0: I/O
1: SEG20

Bit 3 **PFFS3**: Port F3 function selection
0: I/O
1: SEG19

Bit 2 **PFFS2**: Port F2 function selection
0: I/O
1: SEG18
Bit 1 **PFFS1**: Port F1 function selection
0: I/O
1: SEG17
Bit 0 **PFFS0**: Port F0 function selection
0: I/O
1: SEG16

• **PGFS Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PGFS5	PGFS4	PGFS3	PGFS2	PGFS1	PGFS0
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 **PGFS5**: Port G5 function selection
0: I/O
1: SEG29
Bit 4 **PGFS4**: Port G4 function selection
0: I/O
1: SEG28
Bit 3 **PGFS3**: Port G3 function selection
0: I/O
1: SEG27
Bit 2 **PGFS2**: Port G2 function selection
0: I/O
1: SEG26
Bit 1 **PGFS1**: Port G1 function selection
0: I/O
1: SEG25
Bit 0 **PGFS0**: Port G0 function selection
0: I/O
1: SEG24

• **PHFS Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	PHFS0
R/W	—	—	—	—	—	—	—	R/W
POR	—	—	—	—	—	—	—	0

Bit 7~1 Unimplemented, read as "0"
Bit 0 **PHFS0**: Port G0 function selection
0: I/O
1: SYSCKO

• **SFS0 Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	SFS07	SFS06	SFS05	SFS04	SFS03	SFS02	SFS01	SFS00
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 **SFS07**: PD7 special function selection
0: SEG7
1: TCK1 (input)
When PD7 is set to normal I/O input mode (PDC7=1 & PDFS7=0), it is recommended to disable TM1 to avoid to obtain wrong clock source.

Bit 6 **SFS06**: PD6 special function selection
0: SEG6
1: TP1B_2

Bit 5 **SFS05**: PD5 special function selection
0: SEG5
1: TP1B_1

Bit 4 **SFS04**: PD4 special function selection
0: SEG4
1: TP1B_0

Bit 3 **SFS03**: PD3 special function selection
0: SEG3
1: TP1A

Bit 2 **SFS02**: PD2 special function selection
0: SEG2
1: TCK0 (input)
When PD2 is set to normal I/O input mode (PDC2=1 & PDFS2=0), it is recommended to disable TM0 to avoid to obtain wrong clock source.

Bit 1 **SFS01**: PD1 special function selection
0: SEG1
1: TP0_1

Bit 0 **SFS00**: PD0 special function selection
0: SEG0
1: TP0_0

• **SFS1 Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	SFS17	SFS16	SFS15	SFS14	SFS13	SFS12	SFS11	SFS10
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 **SFS17**: PA7 special function selection
0: OP2N
1: INT1 (input)
When PA7 is set to normal I/O input mode, it is recommended to disable INT1 to avoid to obtain wrong clock source.

Bit 6 **SFS16**: PA6 special function selection
0: VG
1: INT0 (input)
When PA6 is set to normal I/O input mode, it is recommended to disable INT0 to avoid to obtain wrong clock source.

Bit 5 **SFS15**: PE5 special function selection
0: SEG13
1: TCK3 (input)
When PE5 is set to normal I/O input mode (PEC5=1 & PEFS5=0), it is recommended to disable TM3 to avoid to obtain wrong clock source.

Bit 4 **SFS14**: PE4 special function selection
0: SEG12
1: TP3_1

Bit 3 **SFS13**: PE3 special function selection
0: SEG11
1: TP3_0

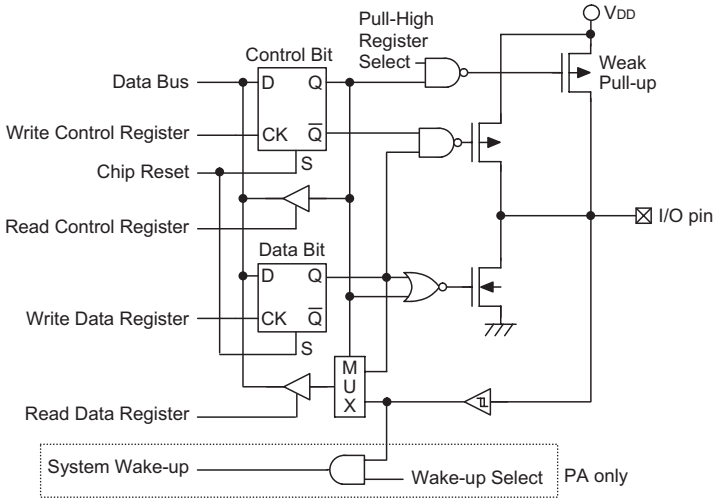
Bit 2 **SFS12**: PE2 special function selection
0: SEG10
1: TCK2 (input)
When PE2 is set to normal I/O input mode (PEC2=1 & PEFS2=0), it is recommended to disable TM2 to avoid to obtain wrong clock source.

Bit 1 **SFS11**: PE1 special function selection
0: SEG9
1: TP2_1

Bit 0 **SFS10**: PE0 special function selection
0: SEG8
1: TP2_0

I/O Pin Structures

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



Generic Input/Output Structure

Programming Considerations

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

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

Timer Modules – TM

One of the most fundamental functions in any microcontroller device is the ability to control and measure time. To implement time related functions each device includes several Timer Modules, abbreviated to the name TM. The TMs are multi-purpose timing units and serve to provide operations such as Timer/Counter, Input Capture, Compare Match Output and Single Pulse Output as well as being the functional unit for the generation of PWM signals. Each of the TMs has either two or three individual interrupts. The addition of input and output pins for each TM ensures that users are provided with timing units with a wide and flexible range of features.

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

Introduction

The devices contain four TMs with each TM having a reference name of TM0, TM1, TM2 and TM3. Each individual TM can be categorised as a certain type, namely Compact Type TM, Standard Type TM or Enhanced Type TM. Although similar in nature, the different TM types vary in their feature complexity. The common features to all of the Compact, Standard and Enhanced TMs will be described in this section, the detailed operation regarding each of the TM types will be described in separate sections. The main features and differences between the three types of TMs are summarised in the accompanying table.

Function	CTM	STM	ETM
Timer/Counter	√	√	√
I/P Capture	—	√	√
Compare Match Output	—	√	√
PWM Channels	1	1	2
Single Pulse Output	—	1	2
PWM Alignment	Edge	Edge	Edge & Centre
PWM Adjustment Period & Duty	Duty or Period	Duty or Period	Duty or Period

TM Function Summary

This chip contains a specific number of either Compact Type, Standard Type and Enhanced Type TM units which are shown in the table together with their individual reference name, TM0~TM3.

Device	TM0	TM1	TM2	TM3
HT45F65	10-bit CTM	10-bit CTM	16-bit STM	—
HT45F66/HT45F67	10-bit CTM	10-bit ETM	16-bit STM	10-bit CTM

TM Name/Type Reference

TM Operation

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

TM Clock Source

The clock source which drives the main counter in each TM can originate from various sources. The selection of the required clock source is implemented using the TnCK2~TnCK0 bits in the TM control registers. The clock source can be a ratio of either the system clock f_{SYS} or the internal high clock f_H, the f_{TBC} clock source or the external TCKn pin. Note that setting these bits to the value 101 will select a reserved clock input, in effect disconnecting the TM clock source. The TCKn pin clock source is used to allow an external signal to drive the TM as an external clock source or for event counting.

TM Interrupts

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

TM External Pins

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

The TMs each have more output pins with the label TPn. When the TM is in the Compare Match Output Mode, these pins can be controlled by the TM to switch to a high or low level or to toggle when a compare match situation occurs. The external TPn output pin is also the pin where the TM generates the PWM output waveform. As the TM output pins are pin-shared with other function, the TM output function must first be setup using relevant pin-remapping control registers.

All TM output pin names have a "_n" suffix. Pin names that include a "_1" or "_2" suffix indicate that they are from a TM with multiple output pins. This allows the TM to generate a complimentary output pair, selected using the I/O register data bits.

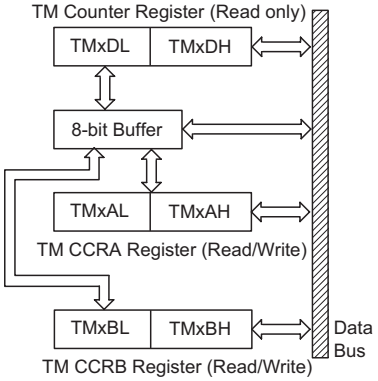
Device	CTM	STM	ETM
HT45F65	TP0_0, TP0_1 TP1_0, TP1_1	TP2_0, TP2_1	—
HT45F66/HT45F67	TP0_0, TP0_1 TP3_0, TP3_1	TP2_0, TP2_1	TP1A, TP1B_0, TP1B_1, TP1B_2

TM Output Pins

Programming Considerations

The TM Counter Registers and the Capture/Compare CCRA and CCRB registers, being either 10-bit or 16-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 CCRB registers are implemented in the way shown in the following diagram and accessing these register pairs is carried out in a specific way described above, it is recommended to use the "MOV" instruction to access the CCRA and CCRB low byte registers, named TMxAL and TMxBL, using the following access procedures. Accessing the CCRA or CCRB low byte registers without following these access procedures will result in unpredictable values.



The following steps show the read and write procedures:

- Writing Data to CCRB or CCRA
  - ♦ Step 1. Write data to Low Byte TMxAL or TMxBL
  - – Note that here data is only written to the 8-bit buffer.
  - ♦ Step 2. Write data to High Byte TMxAH or TMxBH
  - – 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 CCRB or CCRA
  - ♦ Step 1. Read data from the High Byte TMxDH, TMxAH or TMxBH
  - – Here data is read directly from the High Byte registers and simultaneously data is latched from the Low Byte register into the 8-bit buffer.
  - ♦ Step 2. Read data from the Low Byte TMxDL, TMxAL or TMxBL
  - – This step reads data from the 8-bit buffer.

Compact Type TM – CTM

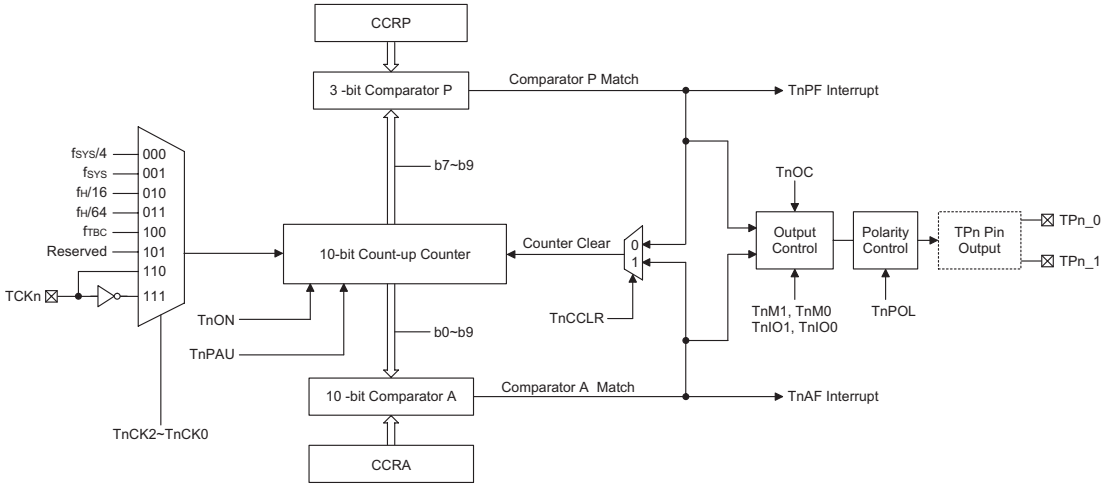
Although the simplest form of the three TM types, the Compact TM type still contains three operating modes, which are Compare Match Output, Timer/Event Counter and PWM Output modes. The Compact TM can also be controlled with an external input pin and can drive two external output pins. These two external output pins can be the same signal or the inverse signal.

Device	Name	TM No.	TM Input Pin	TM Output Pin
HT45F65	10-bit CTM	0, 1	TCK0, TCK1	TP0_0, TP0_1; TP1_0, TP1_1
HT45F66/HT45F67	10-bit CTM	0, 3	TCK0, TCK3	TP0_0, TP0_1; TP3_0, TP3_1

Compact TM Operation

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

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



Compact Type TM Block Diagram
(for HT45F65, n=0 or 1; for HT45F66/HT45F67, n=0 or 3)

Compact Type TM Register Description

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

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

Compact TM Register List

- **TMnC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	TnPAU	TnCK2	TnCK1	TnCK0	TnON	TnRP2	TnRP1	TnRP0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **TnPAU**: TMn Counter Pause Control
0: Run
1: Pause

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

Bit 6~4 **TnCK2~TnCK0**: Select TMn Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{TBC}
101: Undefined
110: TCKn rising edge clock
111: TCKn falling edge clock

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

Bit 3 **TnON**: TMn Counter On/Off Control
0: Off
1: On

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

Bit 2~0 **TnRP2~TnRP0**: TMn CCRP 3-bit register, compared with the TMn Counter bit 9 ~ bit 7 Comparator P Match Period
000: 1024 TMn clocks
001: 128 TMn clocks
010: 256 TMn clocks
011: 384 TMn clocks
100: 512 TMn clocks

101: 640 TMn clocks
110: 768 TMn clocks
111: 896 TMn clocks

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

- **TMnC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	TnM1	TnM0	TnIO1	TnIO0	TnOC	TnPOL	TnDPX	TnCCLR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **TnM1, TnM0**: Select TMn Operating Mode
00: Compare Match Output Mode
01: Undefined
10: PWM Mode
11: Timer/Counter Mode

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

Bit 5~4 **TnIO1, TnIO0**: Select TPn_0, TPn_1 output function
Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output
PWM Mode
00: PWM Output inactive state
01: PWM Output active state
10: PWM output
11: Undefined
Timer/Counter Mode
Unused

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

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

In the PWM Mode, the TnIO1 and TnIO0 bits determine how the TM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to only change the values of the TnIO1 and TnIO0 bits only after the TMn has been switched off.

Unpredictable PWM outputs will occur if the TnIO1 and TnIO0 bits are changed when the TM is running.

Bit 3 **TnOC**: TPn_0, TPn_1 Output control bit
Compare Match Output Mode
0: Initial low
1: Initial high
PWM Mode
0: Active low
1: Active high
This is the output control bit for the TM output pin. Its operation depends upon whether TM is being used in the Compare Match Output Mode or in the PWM Mode. It has no effect if the TM is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the TM output pin before a compare match occurs. In the PWM Mode it determines if the PWM signal is active high or active low.

Bit 2 **TnPOL**: TPn_0, TPn_1 Output polarity Control
0: Non-invert
1: Invert
This bit controls the polarity of the TPn_0 or TPn_1 output pin. When the bit is set high the TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in the Timer/Counter Mode.

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

Bit 0 **TnCCLR**: Select TMn Counter clear condition
0: TMn Comparatror P match
1: TMn Comparatror A match
This bit is used to select the method which clears the counter. Remember that the Compact TM contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the TnCCLR bit set high, the counter will be cleared when a compare match occurs from the Comparator A. When the bit is low, the counter will be cleared when a compare match occurs from the Comparator P or with a counter overflow. A counter overflow clearing method can only be implemented if the CCRP bits are all cleared to zero. The TnCCLR bit is not used in the PWM Mode.

• TMnDL Register

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

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

• TMnDH Register

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

Bit 7~2 Unimplemented, read as "0"

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

• **TMnAL Register**

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

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

• **TMnAH Register**

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

Bit 7~2 Unimplemented, read as "0"
Bit 1~0 **D9~D8**: TMn CCRA High Byte Register bit 1 ~ bit 0
TMn 10-bit CCRA bit 9 ~ bit 8

Compact Type TM Operating Modes

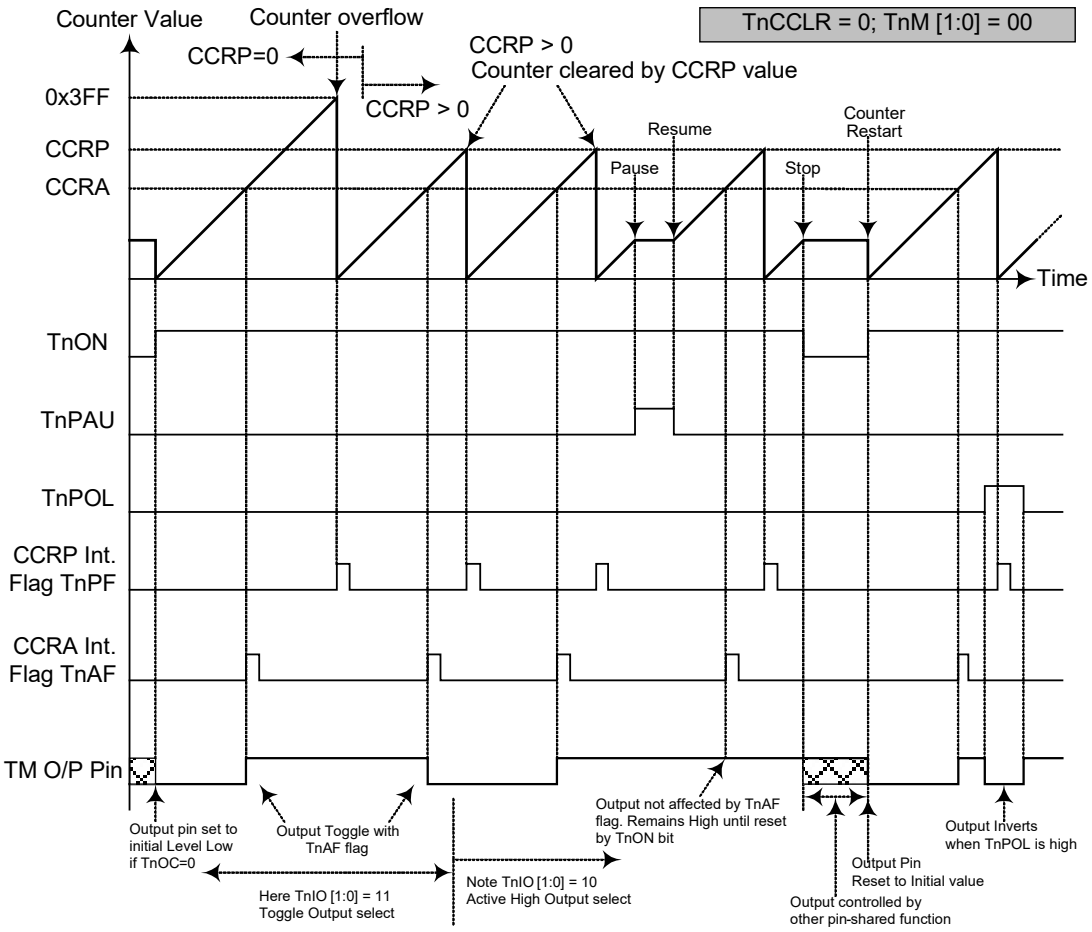
The Compact Type TM can operate in one of three operating modes, Compare Match Output Mode, PWM Mode or Timer/Counter Mode. The operating mode is selected using the TnM1 and TnM0 bits in the TMnC1 register.

Compare Match Output Mode

To select this mode, bits TnM1 and TnM0 in the TMnC1 register, should be set to 00 respectively. In this mode once the counter is enabled and running it can be cleared by three methods. These are a counter overflow, a compare match from Comparator A and a compare match from Comparator P. When the TnCCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match occurs from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both TnAF and TnPF interrupt request flags for the Comparator A and Comparator P respectively, will both be generated.

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

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

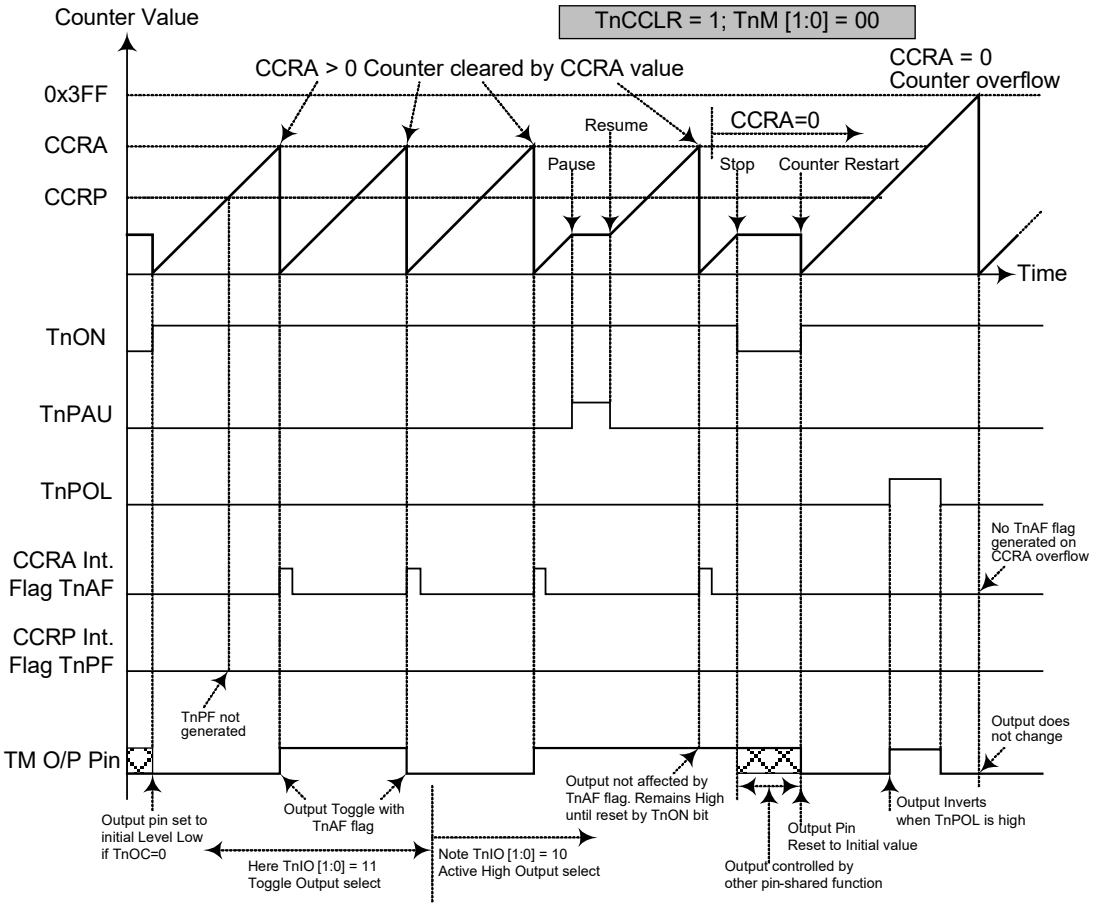


Compare Match Output Mode – TnCCLR= 0

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

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

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



Compare Match Output Mode – TnCCLR = 1

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

Timer/Counter Mode

To select this mode, bits TnM1 and TnM0 in the TMnC1 register should be set to 11 respectively. The Timer/Counter Mode operates in an identical way to the Compare Match Output Mode generating the same interrupt flags. The exception is that in the Timer/Counter Mode the TM output pin is not used. Therefore the above description and Timing Diagrams for the Compare Match Output Mode can be used to understand its function. As the TM output pin is not used in this mode, the pin can be used as a normal I/O pin or other pin-shared function.

PWM Output Mode

To select this mode, bits TnM1 and TnM0 in the TMnC1 register should be set to 10 respectively. The PWM function within the TM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the TM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM mode, the TnCCLR bit has no effect on the PWM operation. Both of the CCRA and CCRP registers are used to generate the PWM waveform, one register is used to clear the internal counter and thus control the PWM waveform frequency, while the other one is used to control the duty cycle. Which register is used to control either frequency or duty cycle is determined using the TnDPX bit in the TMnC1 register. The PWM waveform frequency and duty cycle can therefore be controlled by the values in the CCRA and CCRP registers.

An interrupt flag, one for each of the CCRA and CCRP, will be generated when a compare match occurs from either Comparator A or Comparator P. The TnOC bit in the TMnC1 register is used to select the required polarity of the PWM waveform while the two TnIO1 and TnIO0 bits are used to enable the PWM output or to force the TM output pin to a fixed high or low level. The TnPOL bit is used to reverse the polarity of the PWM output waveform.

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

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

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

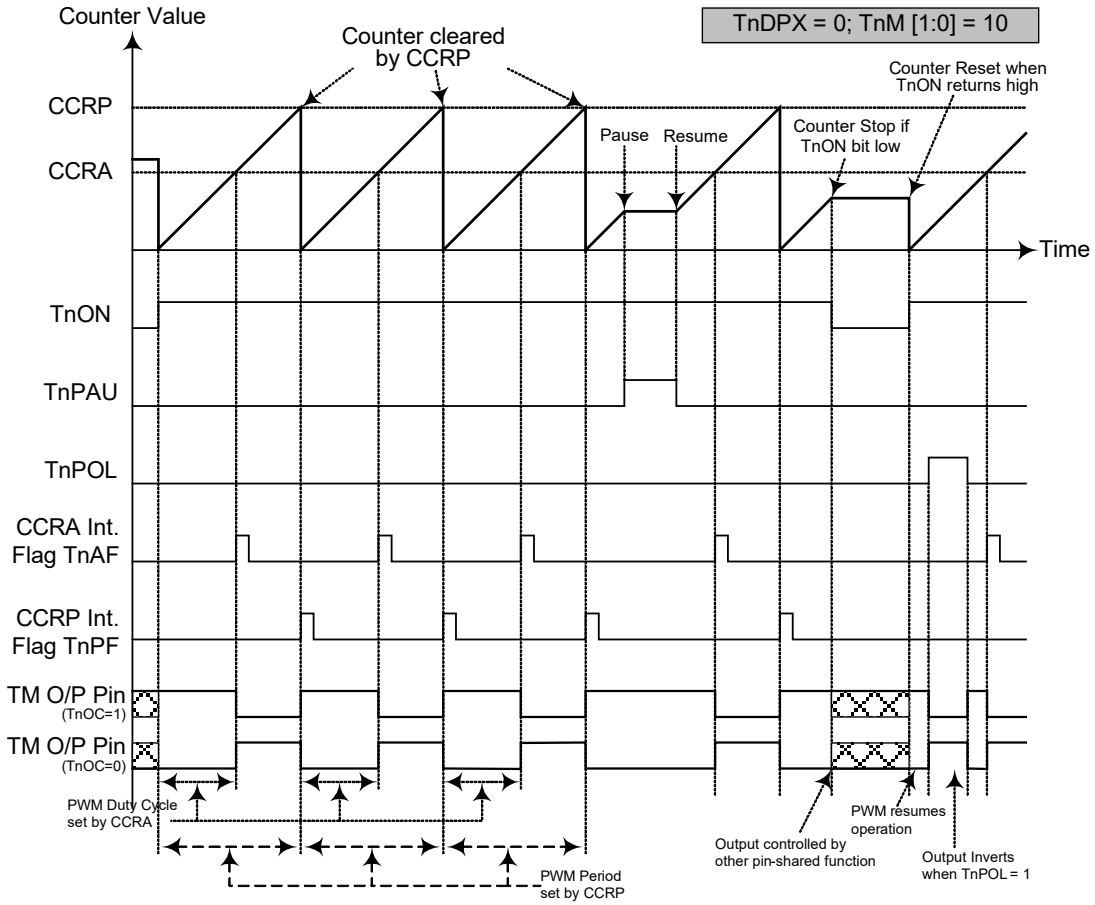
The CTM PWM output frequency = $(f_{SYS}/4)/512 = f_{SYS}/2048$ = 7.8125 kHz, duty = 128/512 = 25%.

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

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

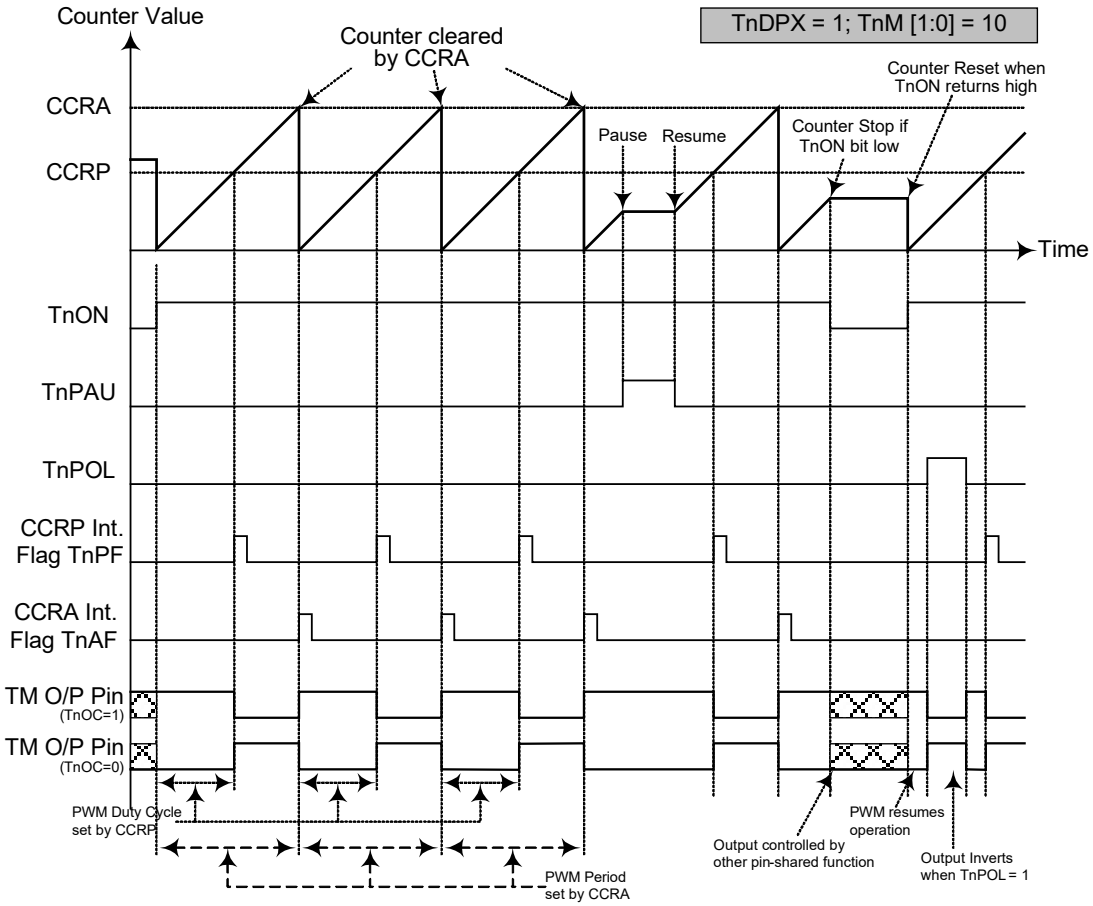
CCRP	001b	010b	011b	100b	101b	110b	111b	000b
Period	CCRA							
Duty	128	256	384	512	640	768	896	1024

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



PWM Mode – TnDPX = 0

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



PWM Mode – TnDPX = 1

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

Standard Type TM – STM

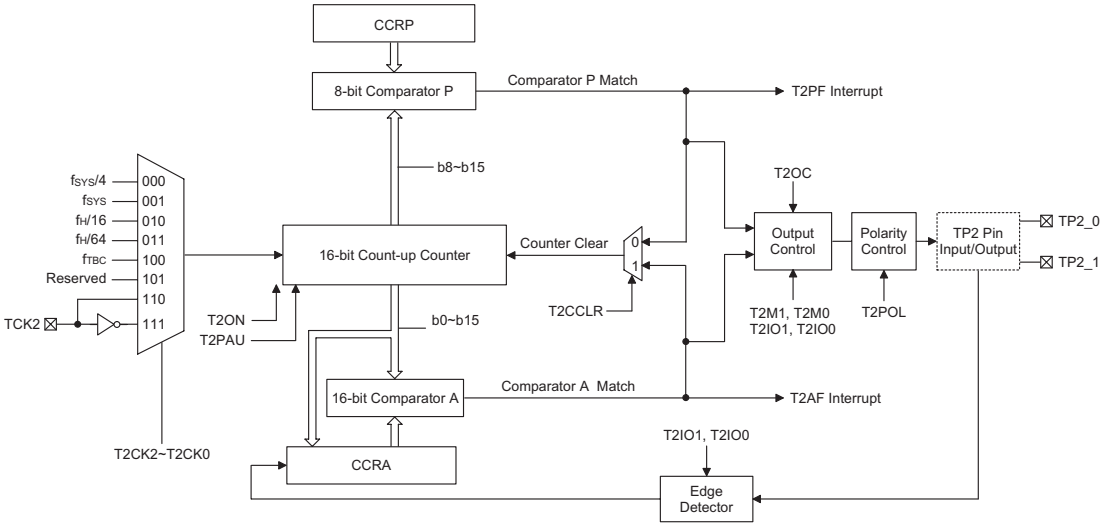
The Standard Type TM contains five operating modes, which are Compare Match Output, Timer/ Event Counter, Capture Input, Single Pulse Output and PWM Output modes. The Standard TM can also be controlled with an external input pin and can drive two external output pins.

Device	Name	TM No.	TM Input Pin	TM Output Pin
HT45F65	16-bit STM	2	TCK2	TP2_0, TP2_1
HT45F66/HT45F67	16-bit STM	2	TCK2	TP2_0, TP2_1

Standard TM Operation

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

The only way of changing the value of the 16-bit counter using the application program, is to clear the counter by changing the T2ON 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 including an input pin and can also control an output pin. All operating setup conditions are selected using relevant internal registers.



Standard Type TM Block Digram

Standard Type TM Register Description

Overall operation of the Standard TM is controlled using a series of registers. A read only register pair exists to store the internal counter 16-bit value, while a read/write register pair exists to store the internal 16-bit CCRA value. The remaining two registers are control registers which setup the different operating and control modes as well as the eight CCRP bits.

Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TM2C0	T2PAU	T2CK2	T2CK1	T2CK0	T2ON	—	—	—
TM2C1	T2M1	T2M0	T2IO1	T2IO0	T2OC	T2POL	T2DPX	T2CCLR
TM2DL	D7	D6	D5	D4	D3	D2	D1	D0
TM2DH	D15	D14	D13	D12	D11	D10	D9	D8
TM2AL	D7	D6	D5	D4	D3	D2	D1	D0
TM2AH	D15	D14	D13	D12	D11	D10	D9	D8
TM2RP	D7	D6	D5	D4	D3	D2	D1	D0

16-bit Standard TM Register List

- **TM2C0 Register**

Bit	7	6	5	4	3	2	1	0
Name	T2PAU	T2CK2	T2CK1	T2CK0	T2ON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

Bit 7 **T2PAU**: TM2 Counter Pause Control
0: Run
1: Pause
The counter can be paused by setting this bit high. Clearing the bit to zero restores normal counter operation. When in a Pause condition the TM will remain powered up and continue to consume power. The counter will retain its residual value when this bit changes from low to high and resume counting from this value when the bit changes to a low value again.

Bit 6~4 **T2CK2~T2CK0**: Select TM2 Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{TBC}
101: Undefined
110: TCK2 rising edge clock
111: TCK2 falling edge clock
These three bits are used to select the clock source for the TM. Selecting the Reserved clock input will effectively disable the internal counter. The external pin clock source can be chosen to be active on the rising or falling edge. The clock source f_{SYS} is the system clock, while f_H and f_{TBC} are other internal clocks, the details of which can be found in the oscillator section.

Bit 3 **T2ON**: TM2 Counter On/Off Control
0: Off
1: On
This bit controls the overall on/off function of the TM. Setting the bit high enables the counter to run, clearing the bit disables the TM. Clearing this bit to zero will stop the counter from counting and turn off the TM which will reduce its power consumption. When the bit changes state from low to high the internal counter value will be reset to zero, however when the bit changes from high to low, the internal counter will retain its residual value. If the TM is in the Compare Match Output Mode then the TM output pin will be reset to its initial condition, as specified by the T2OC bit, when the T2ON bit changes from low to high.

Bit 2~0 Unimplemented, read as "0"

• **TM2C1 Register**

Bit	7	6	5	4	3	2	1	0
Name	T2M1	T2M0	T2IO1	T2IO0	T2OC	T2POL	T2DPX	T2CCLR
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 **T2M1, T2M0**: Select TM2 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 TM. To ensure reliable operation the TM should be switched off before any changes are made to the T2M1 and T2M0 bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4 **T2IO1, T2IO0**: Select TP2_0, TP2_1 output function
Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output
PWM Mode/Single Pulse Output Mode
00: PWM Output inactive state
01: PWM Output active state
10: PWM output
11: Single pulse output
Capture Input Mode
00: Input capture at rising edge of TP2_0, TP2_1
01: Input capture at falling edge of TP2_0, TP2_1
10: Input capture at falling/rising edge of TP2_0, TP2_1
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 T2IO1 and T2IO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the T2OC bit in the TM2C1 register. Note that the output level requested by the T2IO1 and T2IO0 bits must be different from the initial value setup using the T2OC 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 T2ON bit from low to high.

In the PWM Mode, the T2IO1 and T2IO0 bits determine how the TM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to only change the values of the T2IO1 and T2IO0 bits only after the TM has been switched off. Unpredictable PWM outputs will occur if the T2IO1 and T2IO0 bits are changed when the TM is running.

Bit 3 **T2OC**: TP2_0, TP2_1 Output control bit
Compare Match Output Mode
0: Initial low
1: Initial high
PWM Mode/Single Pulse Output Mode
0: Active low
1: Active high
This is the output control bit for the TM output pin. Its operation depends upon whether TM is being used in the Compare Match Output Mode or in the PWM Mode/ Single Pulse Output Mode. It has no effect if the TM is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the TM output pin before a compare match occurs. In the PWM Mode it determines if the PWM signal is active high or active low.

Bit 2 **T2POL**: TP2_0, TP2_1 Output polarity Control
0: Non-invert
1: Invert
This bit controls the polarity of the TP2_0 or TP2_1 output pin. When the bit is set high the TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in the Timer/Counter Mode.

Bit 1 **T2DPX**: TM2 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 **T2CCLR**: Select TM2 Counter clear condition
0: TM2 Comparatror P match
1: TM2 Comparatror A match
This bit is used to select the method which clears the counter. Remember that the Compact TM contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the T2CCLR 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 T2CCLR bit is not used in the PWM Mode, Single Pulse or Input Capture Mode.

• TM2DL Register

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

Bit 7~0 **D7~D0**: TM2 Counter Low Byte Register bit 7 ~ bit 0
TM2 16-bit Counter bit 7 ~ bit 0

• TM2DH Register

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

Bit 7~0 **D15~D8**: TM2 Counter High Byte Register bit 7 ~ bit 0
TM2 16-bit Counter bit 15 ~ bit 8

• **TM2AL Register**

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

Bit 7~0 **D7~D0**: TM2 CCRA Low Byte Register bit 7 ~ bit 0
TM2 16-bit CCRA bit 7 ~ bit 0

• **TM2AH Register**

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

Bit 7~~0 **D15~D8**: TM2 CCRA High Byte Register bit 7 ~ bit 0
TM2 16-bit CCRA bit 15 ~ bit 8

• **TM2RP Register**

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

Bit 7~0 **D7~D0**: TM2 CCRP Register bit 7 ~ bit 0
TM2 CCRP 8-bit register, compared with the TM2 Counter bit 15 ~ bit 8. Comparator P Match Period
0: 65536 TM2 clocks
1~255: 256×(1~255) TM2 clocks
These eight bits are used to setup the value on the internal CCRP 8-bit register, which are then compared with the internal counter's highest eight bits. The result of this comparison can be selected to clear the internal counter if the T2CCLR bit is set to zero. Setting the T2CCLR bit to zero ensures that a compare match with the CCRP values will reset the internal counter. As the CCRP bits are only compared with the highest eight counter bits, the compare values exist in 256 clock cycle multiples. Clearing all eight bits to zero is in effect allowing the counter to overflow at its maximum value.

Standard Type TM Operating Modes

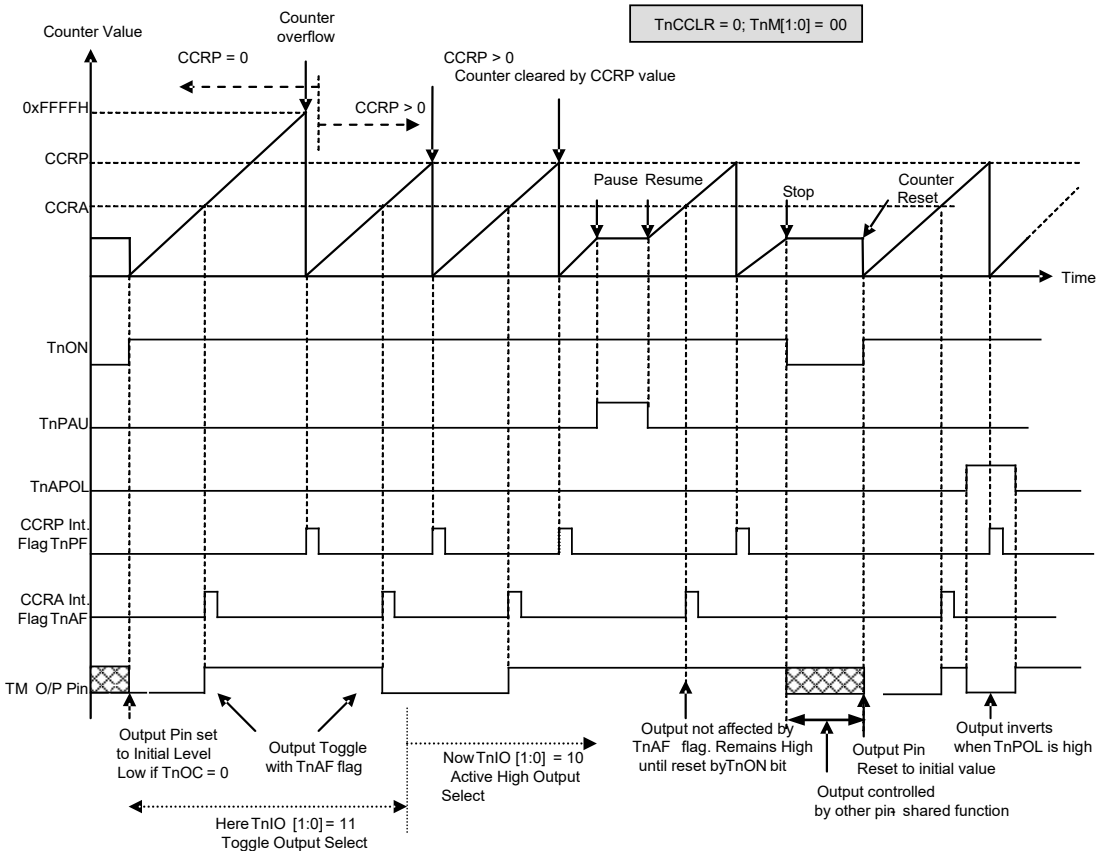
The Standard Type TM can operate in one of five operating modes, Compare Match Output Mode, PWM Output Mode, Single Pulse Output Mode, Capture Input Mode or Timer/Counter Mode. The operating mode is selected using the T2M1 and T2M0 bits in the TM2C1 register.

Compare Output Mode

To select this mode, bits T2M1 and T2M0 in the TM2C1 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 T2CCLR 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 T2AF and T2PF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

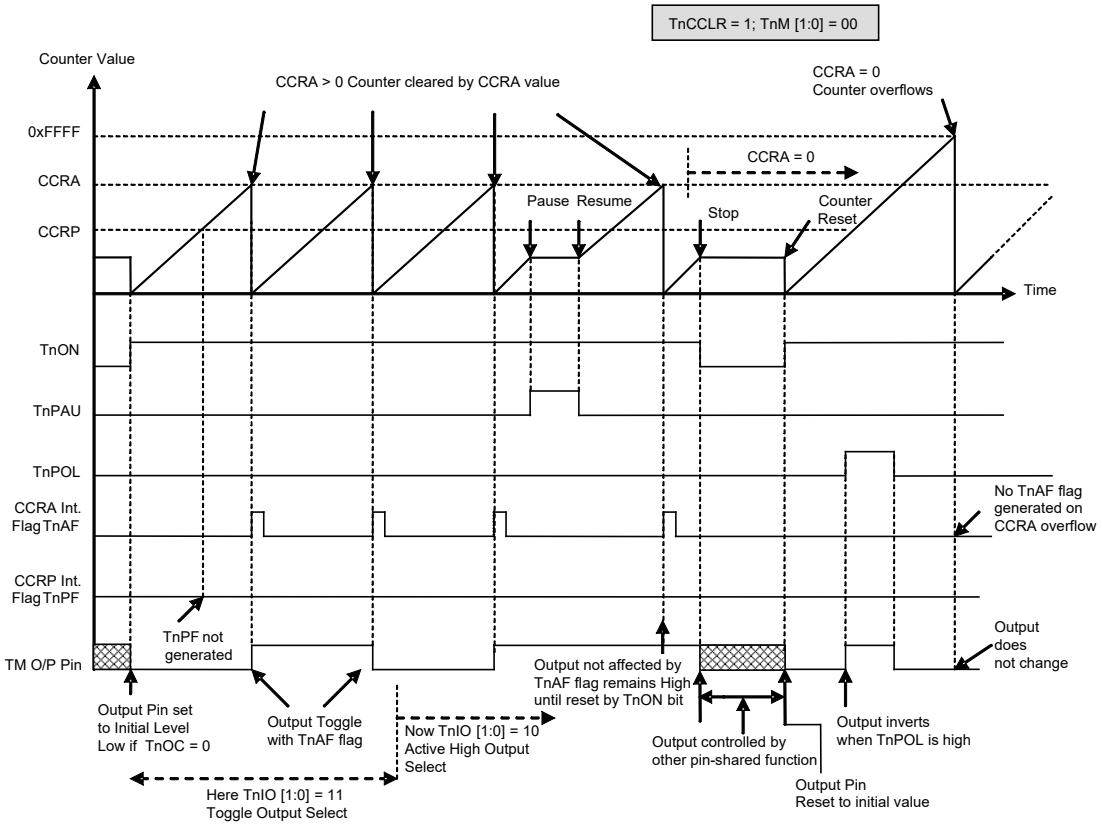
If the T2CCLR bit in the TM2C1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the T2AF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when T2CCLR is high no T2PF interrupt request flag will be generated. In the Compare Match Output Mode, the CCRA can not be set to "0".

As the name of the mode suggests, after a comparison is made, the TM output pin, will change state. The TM output pin condition however only changes state when an T2AF interrupt request flag is generated after a compare match occurs from Comparator A. The T2PF interrupt request flag, generated from a compare match occurs from Comparator P, will have no effect on the TM output pin. The way in which the TM output pin changes state are determined by the condition of the T2IO1 and T2IO0 bits in the TM2C1 register. The TM output pin can be selected using the T2IO1 and T2IO0 bits to go high, to go low or to toggle from its present condition when a compare match occurs from Comparator A. The initial condition of the TM output pin, which is setup after the T2ON bit changes from low to high, is setup using the T2OC bit. Note that if the T2IO1 and T2IO0 bits are zero then no pin change will take place.



Compare Match Output Mode – TnCCLR = 0 (n=2)

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



Compare Match Output Mode – TnCCLR = 1 (n=2)

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

Timer/Counter Mode

To select this mode, bits T2M1 and T2M0 in the TM2C1 register should be set to 11 respectively. The Timer/Counter Mode operates in an identical way to the Compare Match Output Mode generating the same interrupt flags. The exception is that in the Timer/Counter Mode the TM output pin is not used. Therefore the above description and Timing Diagrams for the Compare Match Output Mode can be used to understand its function. As the TM output pin is not used in this mode, the pin can be used as a normal I/O pin or other pin-shared function.

PWM Output Mode

To select this mode, bits T2M1 and T2M0 in the TM2C1 register should be set to 10 respectively and also the T2IO1 and T2IO0 bits should be set to 10 respectively. The PWM function within the TM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the TM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM mode, the T2CCLR 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 T2DPX bit in the TM2C1 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 T2OC bit in the TM2C1 register is used to select the required polarity of the PWM waveform while the two T2IO1 and T2IO0 bits are used to enable the PWM output or to force the TM output pin to a fixed high or low level. The T2POL bit is used to reverse the polarity of the PWM output waveform.

- **16-bit STM, PWM Mode, Edge-aligned Mode, T2DPX=0**

CCRP	1~255	000b
Period	CCRP×256	65536
Duty	CCRA	

If f_{SYS} = 16MHz, TM clock source select $f_{SYS}/4$, CCRP = 2 and CCRA = 128,

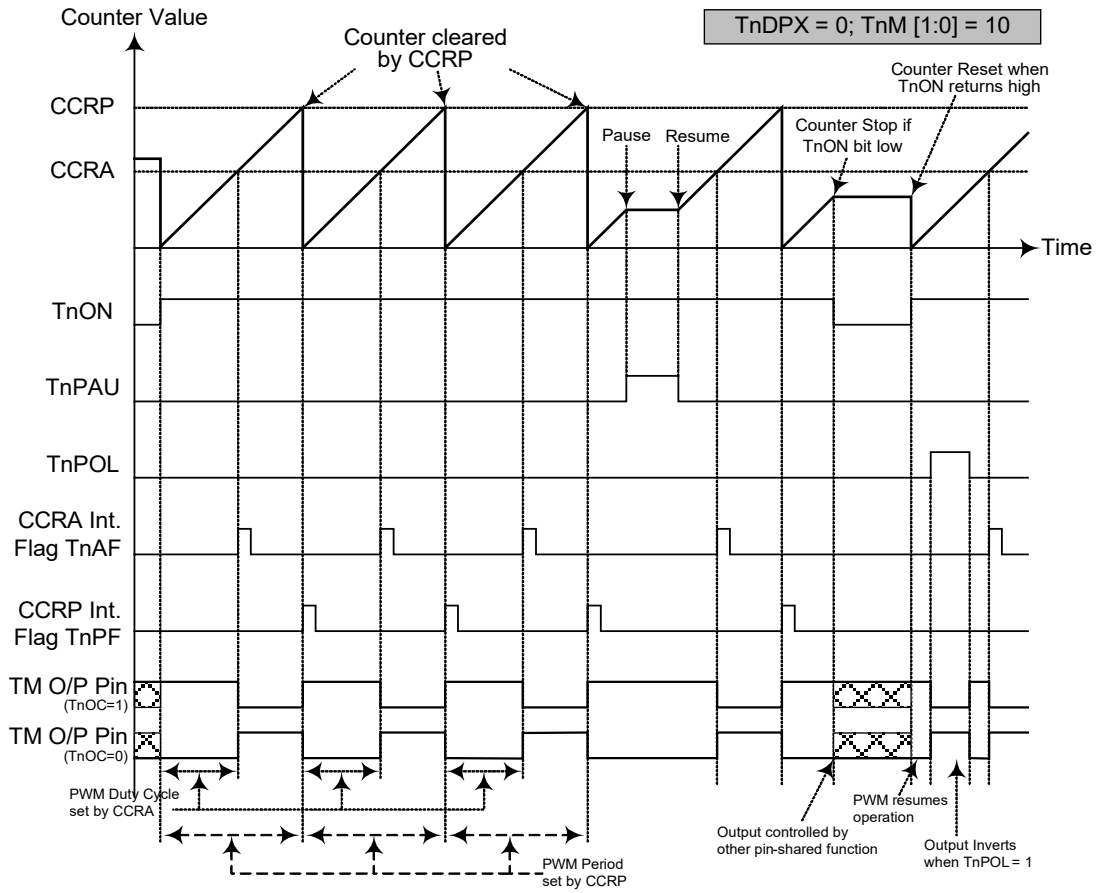
The STM PWM output frequency = $(f_{SYS}/4)/(2\times256) = f_{SYS}/2048$ = 7.8125kHz, duty = 128/(2×256) = 25%.

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

- **16-bit STM, PWM Mode, Edge-aligned Mode, T2DPX=1**

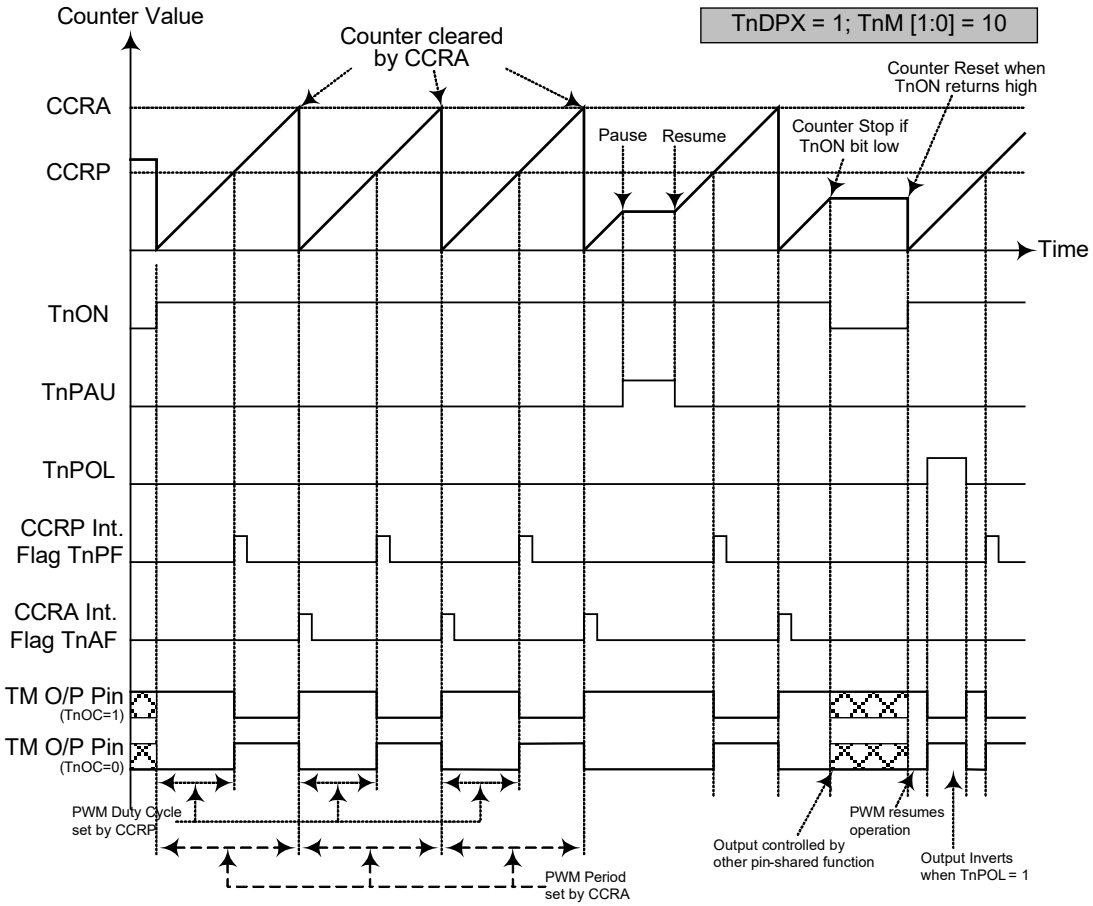
CCRP	1~255	000b
Period	CCRA	
Duty	CCRP×256	65536

The PWM output period is determined by the CCRA register value together with the TM clock while the PWM duty cycle is defined by the (CCRP×256) except when the CCRP value is equal to 000b.



PWM Mode – TnDPX = 0 (n=2)

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



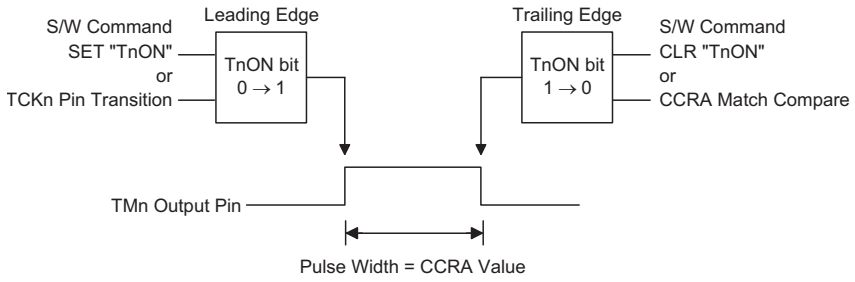
PWM Mode – TnDPX = 1 (n=2)

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

Single Pulse Mode

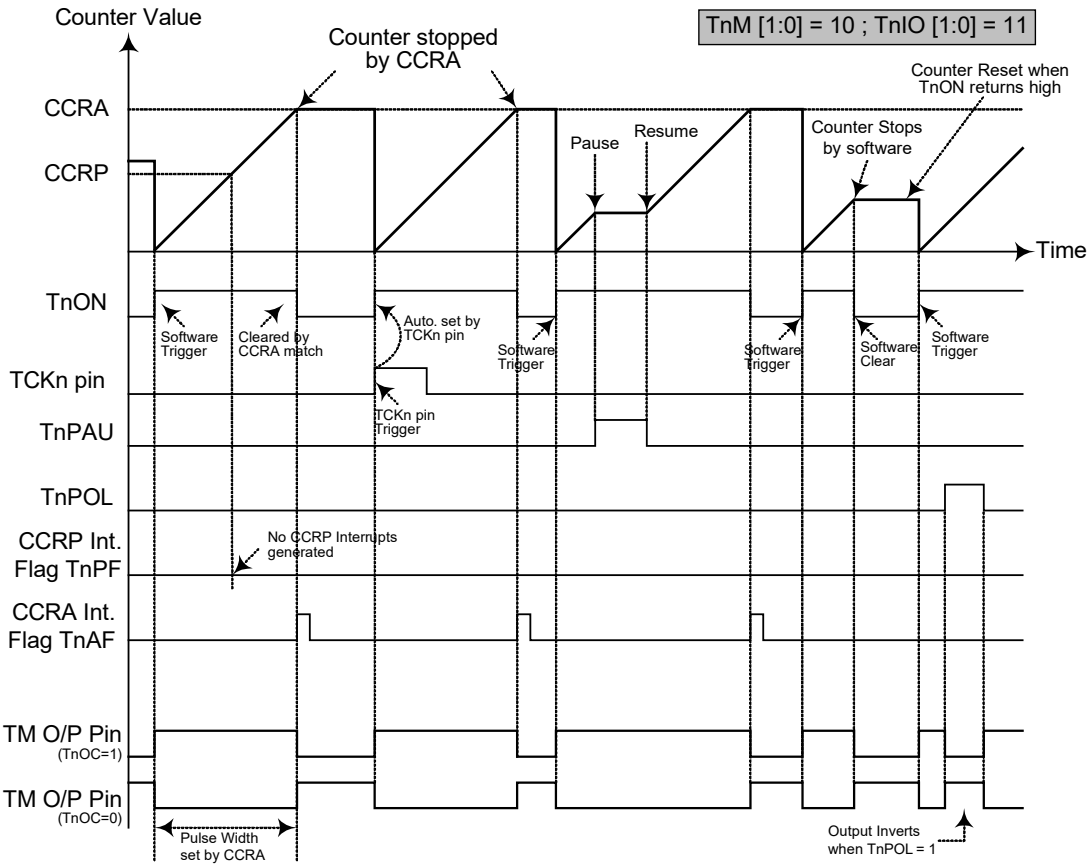
To select this mode, bits T2M1 and T2M0 in the TM2C1 register should be set to 10 respectively and also the T2IO1 and T2IO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the TM output pin.

The trigger for the pulse output leading edge is a low to high transition of the T2ON bit, which can be implemented using the application program. However in the Single Pulse Mode, the T2ON bit can also be made to automatically change from low to high using the external TCK2 pin, which will in turn initiate the Single Pulse output. When the T2ON bit transitions to a high level, the counter will start running and the pulse leading edge will be generated. The T2ON bit should remain high when the pulse is in its active state. The generated pulse trailing edge will be generated when the T2ON bit is cleared to zero, which can be implemented using the application program or when a compare match occurs from Comparator A.



Single Pulse Generation (n=2)

However a compare match from Comparator A will also automatically clear the T2ON 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 TM interrupt. The counter can only be reset back to zero when the T2ON bit changes from low to high when the counter restarts. In the Single Pulse Mode CCRP is not used. The T2CCLR and T2DPX bits are not used in this Mode.



Single Pulse Mode (n=2)

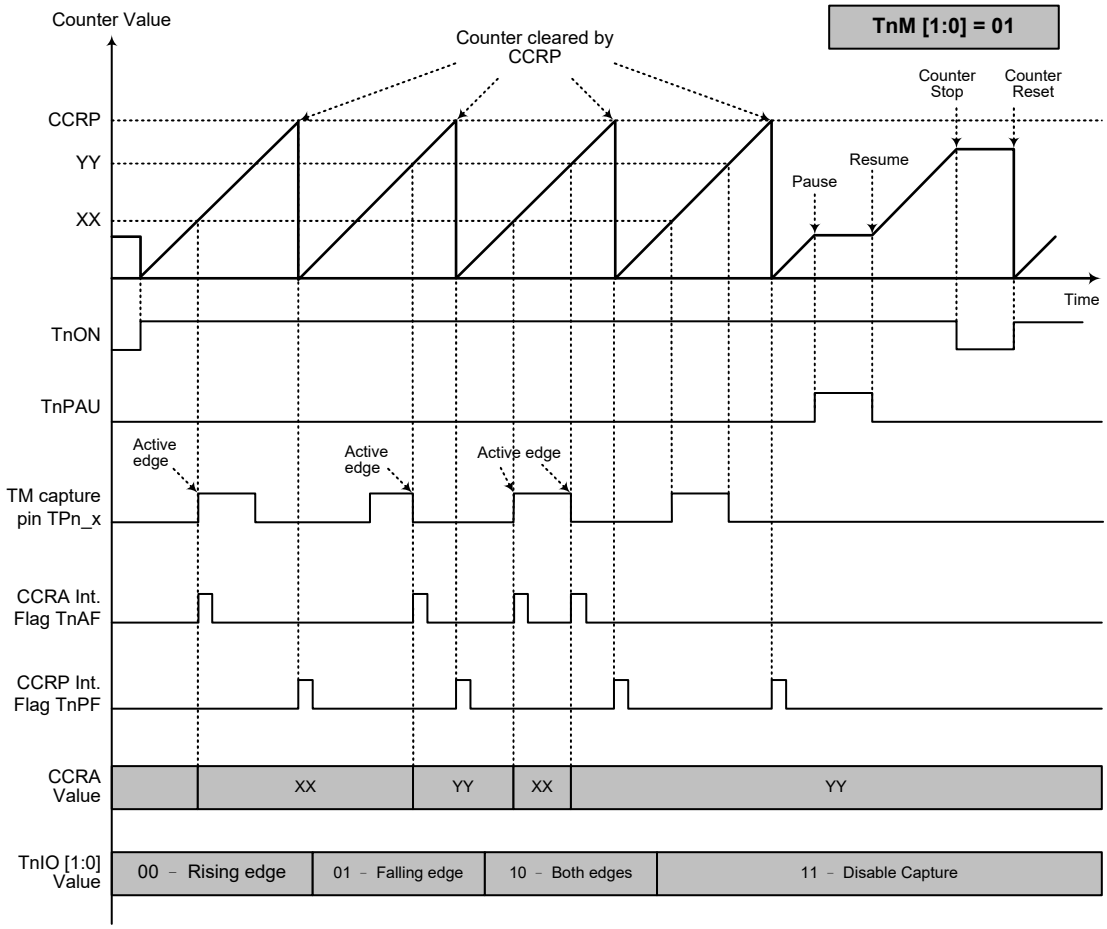
Note: 1. Counter stopped by CCRA
2. CCRP is not used
3. The pulse is triggered by the TCKn pin or by setting the TnON bit high
4. A TCKn pin active edge will automatically set the TnON bit hight
5. In the Single Pulse Mode, TnIO [1:0] must be set to "11" and can not be changed.

Capture Input Mode

To select this mode bits T2M1 and T2M0 in the TM2C1 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 TP2_0 or TP2_1 pin, 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 T2IO1 and T2IO0 bits in the TM2C1 register. The counter is started when the T2ON bit changes from low to high which is initiated using the application program.

When the required edge transition appears on the TP2_0 or TP2_1 pin the present value in the counter will be latched into the CCRA registers and a TM interrupt generated. Irrespective of what events occur on the TP2_0 or TP2_1 pin the counter will continue to free run until the T2ON bit changes from high to low. When a CCRP compare match occurs the counter will reset back to zero; in this way the CCRP value can be used to control the maximum counter value. When a CCRP compare match occurs from Comparator P, a TM interrupt will also be generated. Counting the number of overflow interrupt signals from the CCRP can be a useful method in measuring long pulse widths. The T2IO1 and T2IO0 bits can select the active trigger edge on the TP2_0 or TP2_1 pin to be a rising edge, falling edge or both edge types. If the T2IO1 and T2IO0 bits are both set high, then no capture operation will take place irrespective of what happens on the TP2_0 or TP2_1 pin, however it must be noted that the counter will continue to run.

As the TP2_0 or TP2_1 pin is pin shared with other functions, care must be taken if the TM is in the Input Capture 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 T2CCLR and T2DPX bits are not used in this Mode.



Capture Input Mode (n=2)

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

Enhanced Type TM – ETM (HT45F66/HT45F67)

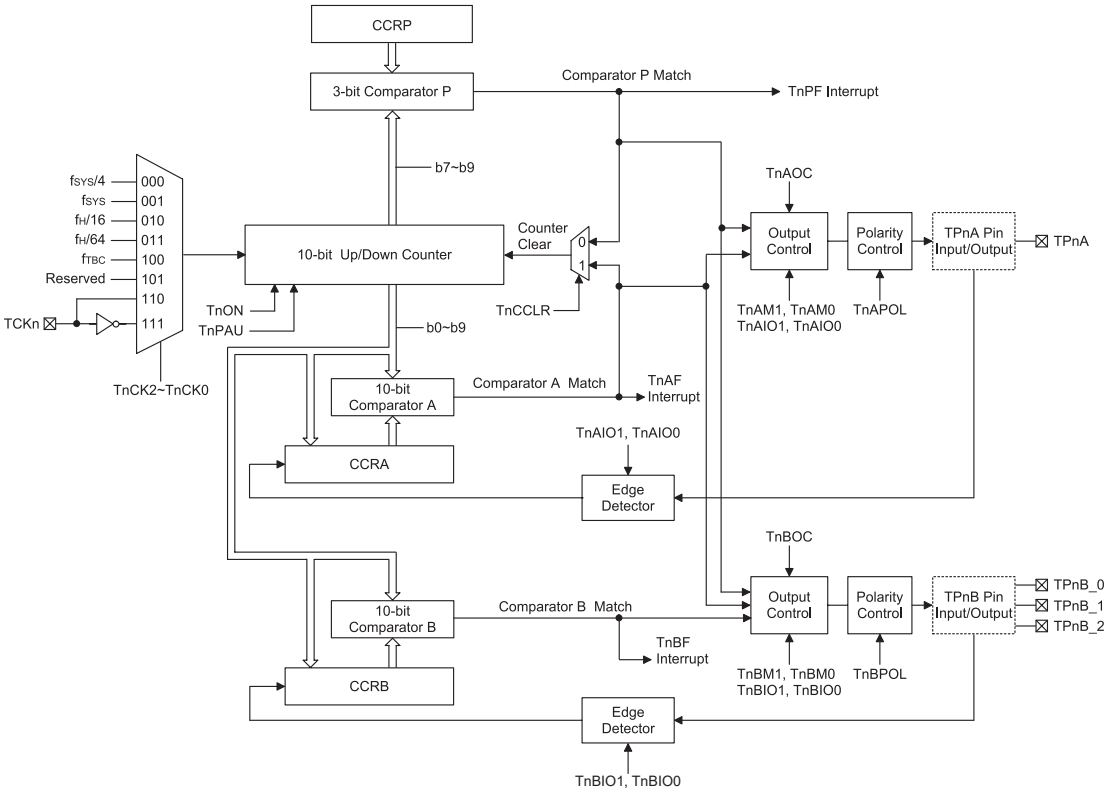
The Enhanced Type TM contains five operating modes, which are Compare Match Output, Timer/ Event Counter, Capture Input, Single Pulse Output and PWM Output modes. The Enhanced TM can also be controlled with an external input pin and can drive three or four external output pins.

Device	Name	TM No.	TM Input Pin	TM Output Pin
HT45F66/HT45F67	10-bit ETM	1	TCK1	TP1A, TP1B_0, TP1B_1, TP1B_2

Enhanced TM Operation

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

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



Enhanced Type TM Block Diagram (n=1)

Enhanced Type TM Register Description

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

Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TM1C0	T1PAU	T1CK2	T1CK1	T1CK0	T1ON	T1RP2	T1RP1	T1RP0
TM1C1	T1AM1	T1AM0	T1AIO1	T1AIO0	T1AOC	T1APOL	T1CDN	T1CCLR
TM1C2	T1BM1	T1BM0	T1BIO1	T1BIO0	T1BOC	T1BPOL	T1PWM1	T1PWM0
TM1DL	D7	D6	D5	D4	D3	D2	D1	D0
TM1DH	—	—	—	—	—	—	D9	D8
TM1AL	D7	D6	D5	D4	D3	D2	D1	D0
TM1AH	—	—	—	—	—	—	D9	D8
TM1BL	D7	D6	D5	D4	D3	D2	D1	D0
TM1BH	—	—	—	—	—	—	D9	D8

10-bit Enhanced TM Register List

- **TM1C0 Register**

Bit	7	6	5	4	3	2	1	0
Name	T1PAU	T1CK2	T1CK1	T1CK0	T1ON	T1RP2	T1RP1	T1RP0
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 **T1PAU**: TM1 Counter Pause Control
0: Run
1: Pause

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

Bit 6~4 **T1CK2~T1CK0**: Select TM1 Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{TBC}
101: Undefined
110: TCK1 rising edge clock
111: TCK1 falling edge clock

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

Bit 3 **T1ON**: TM1 Counter On/Off Control
0: Off
1: On

This bit controls the overall on/off function of the TM. Setting the bit high enables the counter to run, clearing the bit disables the TM. Clearing this bit to zero will stop the counter from counting and turn off the TM which will reduce its power consumption.

When the bit changes state from low to high the internal counter value will be reset to zero, however when the bit changes from high to low, the internal counter will retain its residual value. If the TM is in the Compare Match Output Mode then the TM output pin will be reset to its initial condition, as specified by the T1OC bit, when the T1ON bit changes from low to high.

Bit 2~0 **T1RP2~T1RP0**: TM1 CCRP 3-bit register, compared with the TM1 Counter bit 9~bit 7 Comparator P Match Period

000: 1024 TM1 clocks
001: 128 TM1 clocks
010: 256 TM1 clocks
011: 384 TM1 clocks
100: 512 TM1 clocks
101: 640 TM1 clocks
110: 768 TM1 clocks
111: 896 TM1 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 T1CCLR bit is set to zero. Setting the T1CCLR 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.

• TM1C1 Register

Bit	7	6	5	4	3	2	1	0
Name	T1AM1	T1AM0	T1AIO1	T1AIO0	T1AOC	T1APOL	T1CDN	T1CCLR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **T1AM1, T1AM0**: Select TM1 CCRA 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 TM. To ensure reliable operation the TM should be switched off before any changes are made to the T1AM1 and T1AM0 bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4 **T1AIO1, T1AIO0**: Select TP1A 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 TP1A
01: Input capture at falling edge of TP1A
10: Input capture at falling/rising edge of TP1A
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 T1AIO1 and T1AIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the T1AOC bit in the TM1C1 register. Note that the output level requested by the T1AIO1 and T1AIO0 bits must be different from the initial value setup using the T1AOC 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 T1ON bit from low to high.

In the PWM Mode, the T1AIO1 and T1AIO0 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 T1AIO1 and T1AIO0 bits only after the TM has been switched off. Unpredictable PWM outputs will occur if the T1AIO1 and T1AIO0 bits are changed when the TM is running.

Bit 3 **T1AOC**: TP1A Output control bit
Compare Match Output Mode
0: Initial low
1: Initial high
PWM Mode/Single Pulse Output Mode
0: Active low
1: Active high

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

Bit 2 **T1APOL**: TP1A Output polarity Control
0: Non-invert
1: Invert

This bit controls the polarity of the TP1A output pin. When the bit is set high the TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in the Timer/Counter Mode.

Bit 1 **T1CDN**: TM1 Counter count up or down flag
0: Count up
1: Count down

Bit 0 **T1CCLR**: Select TM1 Counter clear condition
0: TM1 Comparator P match
1: TM1 Comparator A match

This bit is used to select the method which clears the counter. Remember that the Enhanced TM contains three comparators, Comparator A, Comparator B and Comparator P, but only Comparator A or Comparator P can be selected to clear the internal counter. With the T1CCLR 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 T1CCLR bit is not used in the Single Pulse or Input Capture Mode.

• **TM1C2 Register**

Bit	7	6	5	4	3	2	1	0
Name	T1BM1	T1BM0	T1BIO1	T1BIO0	T1BOC	T1BPOL	T1PWM1	T1PWM0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **T1BM1, T1BM0**: Select TM1 CCRB 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 TM. To ensure reliable operation the TM should be switched off before any changes are made to the T1BM1 and T1BM0 bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4 **T1BIO1, T1BIO0**: Select TP1B_0, TP1B_1, TP1B_2 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 TP1B_0, TP1B_1, TP1B_2
01: Input capture at falling edge of TP1B_0, TP1B_1, TP1B_2
10: Input capture at falling/rising edge of TP1B_0, TP1B_1, TP1B_2
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 T1BIO1 and T1BIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator B. 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 B. When the bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the T1BOC bit in the TM1C2 register. Note that the output level requested by the T1BIO1 and T1BIO0 bits must be different from the initial value setup using the T1BOC 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 T1ON bit from low to high.

In the PWM Mode, the T1BIO1 and T1BIO0 bits determine how the TM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to only change the value of the T1BIO1 and T1BIO0 bits only after the TM has been switched off. Unpredictable PWM outputs will occur if the T1BIO1 and T1BIO0 bits are changed when the TM is running.

Bit 3 **T1BOC**: TP1B_0, TP1B_1, TP1B_2 Output control bit
Compare Match Output Mode
0: Initial low
1: Initial high
PWM Mode/Single Pulse Output Mode
0: Active low
1: Active high
This is the output control bit for the TM output pin. Its operation depends upon whether TM is being used in the Compare Match Output Mode or in the PWM Mode/ Single Pulse Output Mode. It has no effect if the TM is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the TM output pin before a compare match occurs. In the PWM Mode it determines if the PWM signal is active high or active low.

Bit 2 **T1BPOL**: TP1B_0, TP1B_1, TB1B_2 Output polarity Control
0: Non-invert
1: Invert
This bit controls the polarity of the TP1B_0, TP1B_1, TP1B_2 output pin. When the bit is set high the TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in the Timer/Counter Mode.

Bit 1~0 **T1PWM1, T1PWM0**: Select PWM Mode
00: Edge aligned
01: Centre aligned, compare match on count up
10: Centre aligned, compare match on count down
11: Centre aligned, compare match on count up or down

• TM1DL Register

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

Bit 7~0 **D7~D0**: TM1 Counter Low Byte Register bit 7 ~ bit 0
TM1 10-bit Counter bit 7 ~ bit 0

• TM1DH Register

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

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **D9~D8**: TM1 Counter High Byte Register bit 1 ~ bit 0
TM1 10-bit Counter bit 9 ~ bit 8

• TM1AL Register

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

Bit 7~0 **D7~D0**: TM1 CCRA Low Byte Register bit 7 ~ bit 0
TM1 10-bit CCRA bit 7 ~ bit 0

• **TM1AH Register**

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

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **D9~D8**: TM1 CCRA High Byte Register bit 1 ~ bit 0
TM1 10-bit CCRA bit 9 ~ bit 8

• **TM1BL Register**

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

Bit 7~0 **D7~D0**: TM1 CCRB Low Byte Register bit 7 ~ bit 0
TM1 10-bit CCRB bit 7 ~ bit 0

• **TM1BH Register**

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

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **D9~D8**: TM1 CCRB High Byte Register bit 1 ~ bit 0
TM1 10-bit CCRB bit 9 ~ bit 8

Enhanced Type TM Operating Modes

The Enhanced 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 T1AM1 and T1AM0 bits in the TM1C1, and the T1BM1 and T1BM0 bits in the TM1C2 register.

ETM Operating Mode	CCRA Compare Match Output Mode	CCRA Timer/Counter Mode	CCRA PWM Output Mode	CCRA Single Pulse Output Mode	CCRA Input Capture Mode
CCRB Compare Match Output Mode	√	—	—	—	—
CCRB Timer/Counter Mode	—	√	—	—	—
CCRB PWM Output Mode	—	—	√	—	—
CCRB Single Pulse Output Mode	—	—	—	√	—
CCRB Input Capture Mode	—	—	—	—	√

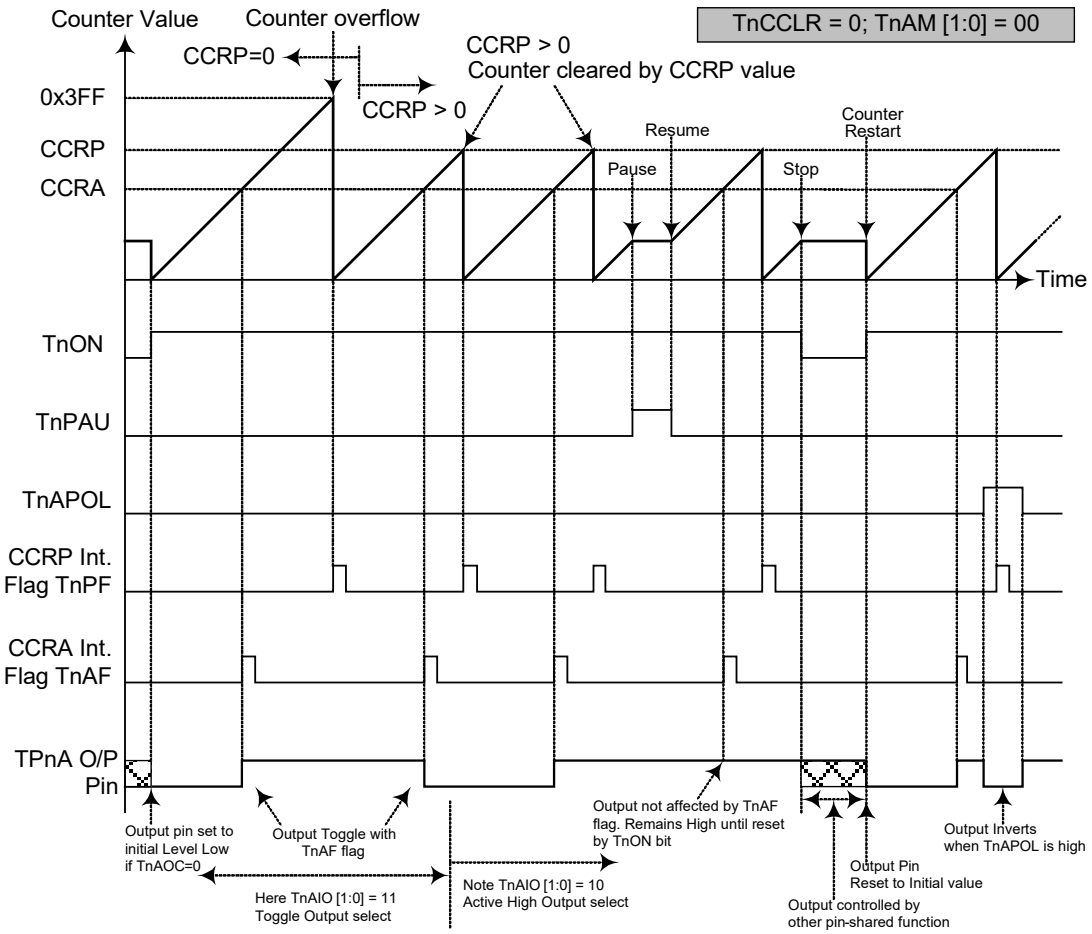
"√": permitted
"—": not permitted

Compare Output Mode

To select this mode, bits T1AM1, T1AM0 and T1BM1, T1BM0 in the TM1C1 and TM1C2 registers should be all cleared to zero. 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 T1CCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match occurs from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both the T1AF and T1PF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

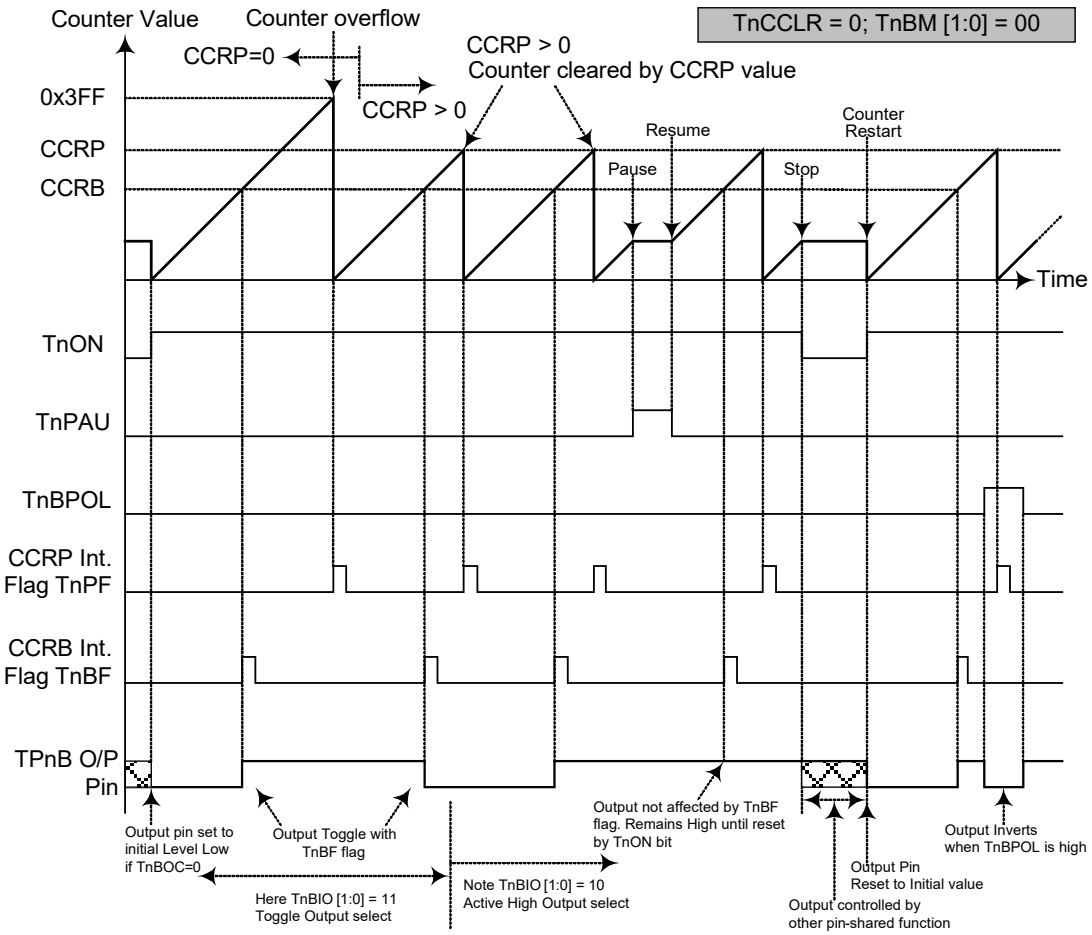
If the T1CCLR bit in the TM1C1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the T1AF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when T1CCLR is high no T1PF interrupt request flag will be generated.

As the name of the mode suggests, after a comparison is made, the TM output pin, will change state. The TM output pin condition however only changes state when a T1AF or T1BF interrupt request flag is generated after a compare match occurs from Comparator A or Comparator B. The T1PF interrupt request flag, generated from a compare match from Comparator P, will have no effect on the TM output pin. The way in which the TM output pin changes state is determined by the condition of the T1AIO1 and T1AIO0 bits in the TM1C1 register for ETM CCRA, and the T1BIO1 and T1BIO0 bits in the TM1C2 register for ETM CCRB. The TM output pin can be selected using the T1AIO1, T1AIO0 bits (for the TP1A pin) and T1BIO1, T1BIO0 bits (for the TP1B_0, TP1B_1 or TP1B_2 pins) to go high, to go low or to toggle from its present condition when a compare match occurs from Comparator A or a compare match occurs from Comparator B. The initial condition of the TM output pin, which is setup after the T1ON bit changes from low to high, is setup using the T1AOC or T1BOC bit for TP1A or TP1B_0, TP1B_1, TP1B_2 output pins. Note that if the T1AIO1, T1AIO0 and T1BIO1, T1BIO0 bits are zero then no pin change will take place.



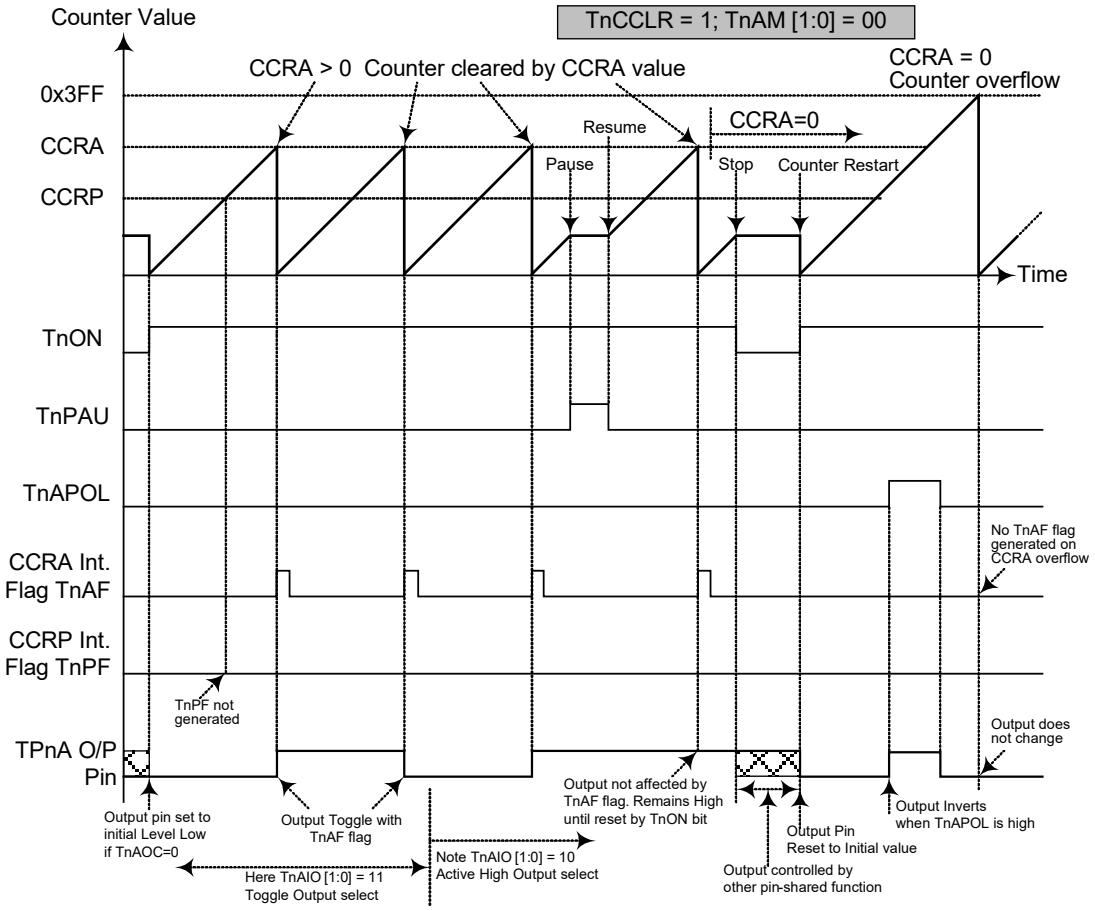
ETM CCRA Compare Match Output Mode – TnCCLR = 0

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



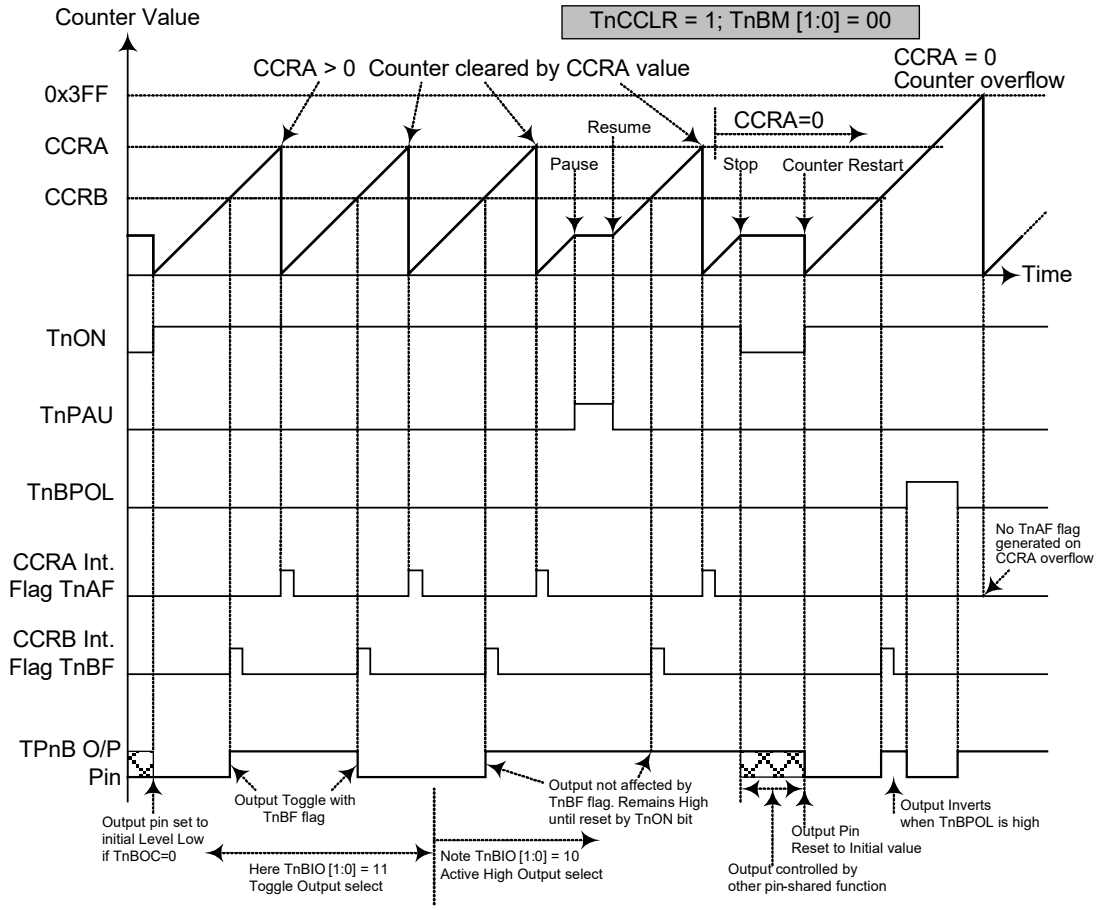
ETM CCRB Compare Match Output Mode – TnCCLR = 0

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



ETM CCRA Compare Match Output Mode – TnCCLR = 1

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



ETM CCRB Compare Match Output Mode – TnCCLR = 1

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

Timer/Counter Mode

To select this mode, bits T1AM1, T1AM0 and T1BM1, T1BM0 in the TM1C1 and TM1C2 register should be set to 11 respectively. The Timer/Counter Mode operates in an identical way to the Compare Match Output Mode generating the same interrupt flags. The exception is that in the Timer/Counter Mode the TM output pin is not used. Therefore the above description and Timing Diagrams for the Compare Match Output Mode can be used to understand its function. As the TM output pin is not used in this mode, the pin can be used as a normal I/O pin or other pin-shared function.

PWM Output Mode

To select this mode, bits T1AM1, T1AM0 and T1BM1, T1BM0 in the TM1C1 and TM1C2 register should be set to 10 respectively and also the corresponding T1AIO1, T1AIO0 and T1BIO1, T1BIO0 bits should be set to 10 respectively. The PWM function within the TM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the TM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM mode, the T1CCLR 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 T1DPX bit in the TM1C1 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 T1OC bit in the TM1C1 register is used to select the required polarity of the PWM waveform while the two T1IO1 and T1IO0 bits are used to enable the PWM output or to force the TM output pin to a fixed high or low level. The T1POL bit is used to reverse the polarity of the PWM output waveform.

- **ETM, PWM Mode, Edge - aligned Mode, T1CCLR=0**

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

If f_{SYS} = 16MHz, TM clock source select $f_{SYS}/4$, CCRP = 100b, CCRA = 128 and CCRB = 256,

The TP1A PWM output frequency = $(f_{SYS}/4)/512 = f_{SYS}/2048$ = 7.8125kHz, duty = 128/512 = 25%.

The TP1B_n PWM output frequency = $(f_{SYS}/4)/512 = f_{SYS}/2048$ = 7.8125kHz, duty = 256/512 = 50%.

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

- **ETM, PWM Mode, Edge - aligned Mode, T1CCLR=1**

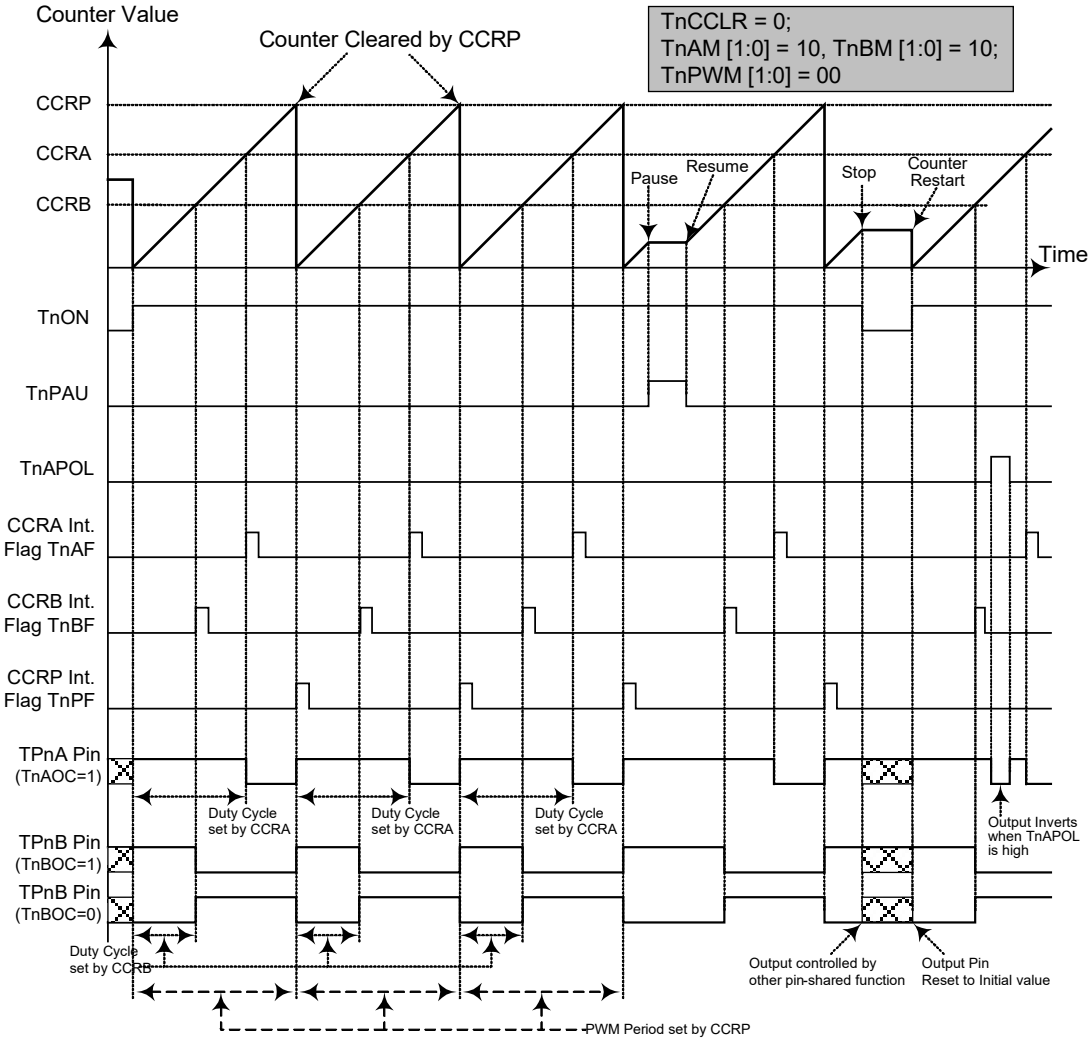
CCRP	1	2	3	511	512	1021	1022	1023
Period	1	2	3	511	512	1021	1022	1023
B Duty	CCRB							

• **ETM, PWM Mode, Center - aligned Mode, T1CCLR=0**

CCRP	001b	010b	011b	100b	101b	110b	111b	000b
Period	256	384	512	640	768	896	1024	2046
A Duty	(CCRA×2)-1							
B Duty	(CCRB×2)-1							

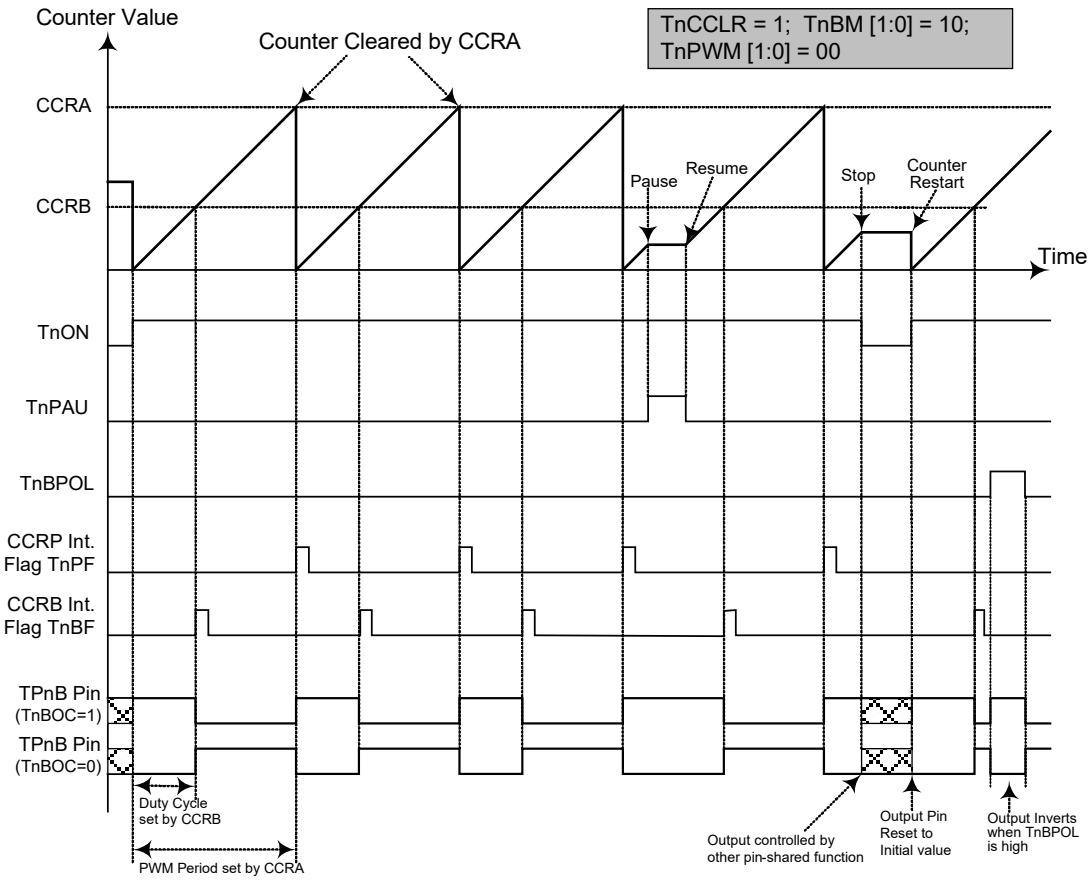
• **ETM, PWM Mode, Center - aligned Mode, T1CCLR=1**

CCRP	1	2	3	511	512	1021	1022	1023
Period	2	3	511	512	1021	1022	1023	2046
B Duty	(CCRB×2)-1							



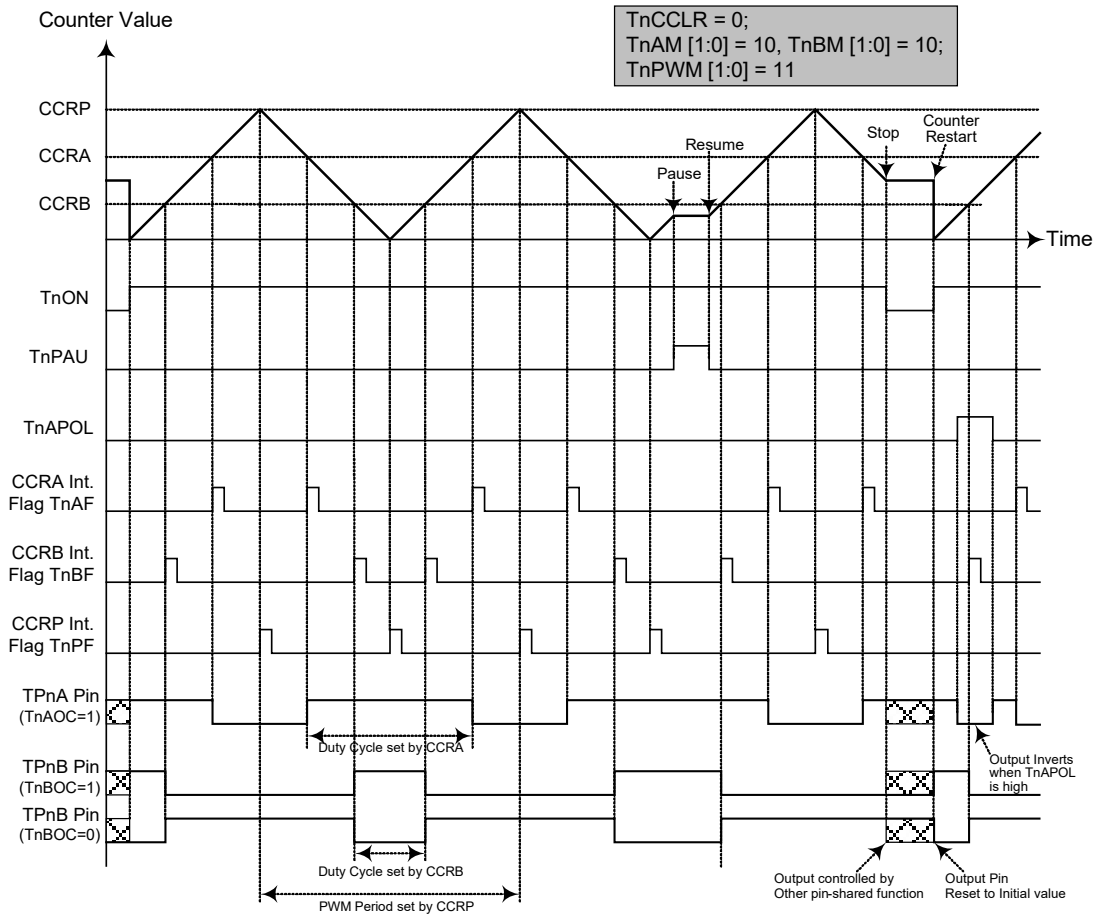
ETM PWM Mode – Edge Aligned (n=1)

Note: 1. Here TnCCLR=0 therefore CCRP clears counter and determines the PWM period
2. The internal PWM function continues running even when TnAIO [1:0] (or TnBIO [1:0]) = 00 or 01
3. CCRA controls the TPnA PWM duty and CCRB controls the TPnB PWM duty



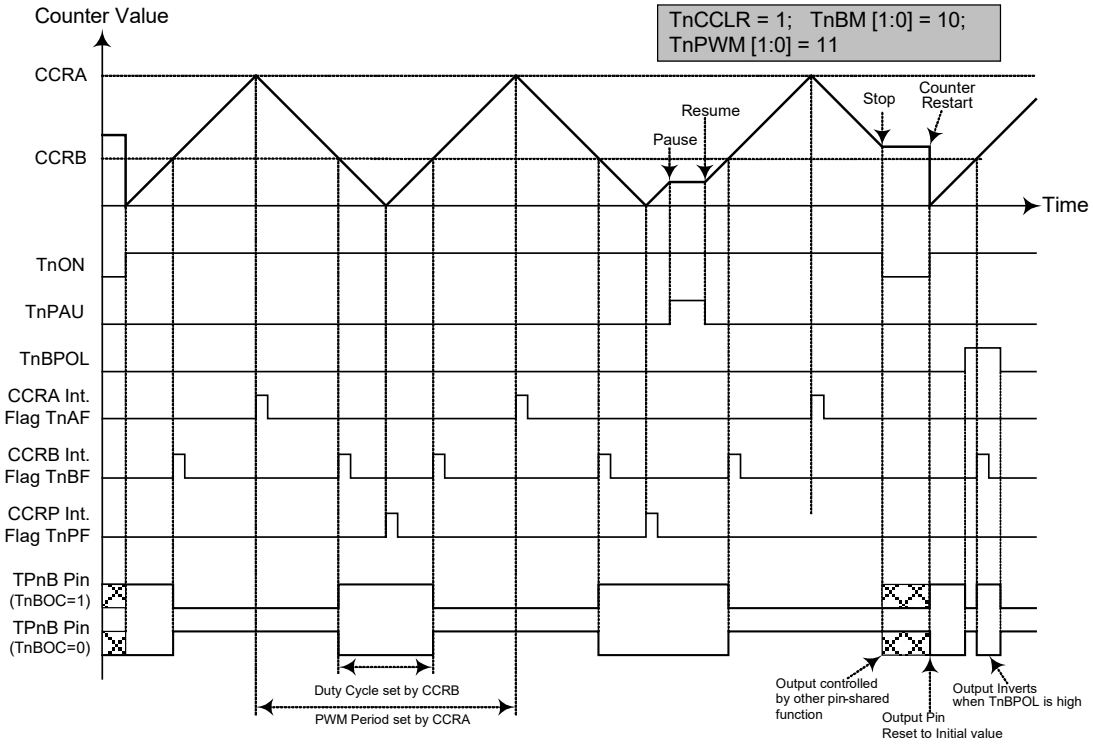
ETM PWM Mode – Edge Aligned (n=1)

Note: 1. Here TnCCLR=1, therefore CCRA clears the counter and determines the PWM period
2. The internal PWM function continues running even when TnBIO [1:0] = 00 or 01
3. The CCRA controls the TPnB PWM period and CCRB controls the TPnB PWM duty
4. Here the TM pin control register should not enable the TPnA pin as a TM output pin.



ETM PWM Mode - Centre Aligned (n=1)

Note: 1. Here TnCCLR=0 therefore CCRP clears the counter and determines the PWM period
2. TnPWM [1:0] =11 therefore the PWM is centre aligned
3. The internal PWM function continues running even when TnAIO [1:0] (or TnBIO [1:0]) = 00 or 01
4. CCRA controls the TPnA PWM duty and CCRB controls the TPnB PWM duty
5. CCRP will generate an interrupt request when the counter decrements to its zero value



ETM PWM Mode – Centre Aligned (n=1)

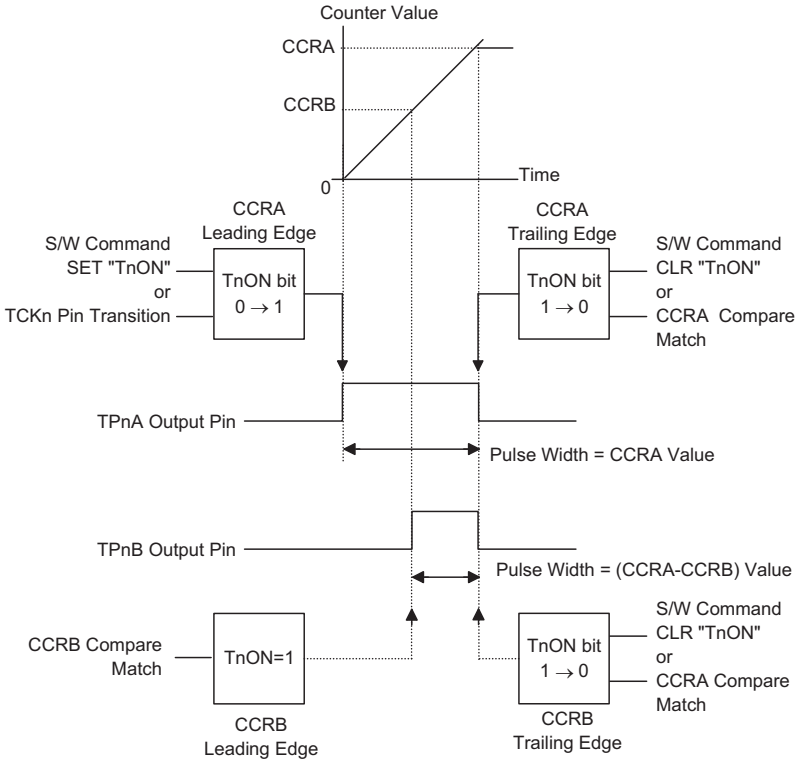
Note: 1. Here TnCCLR=1 therefore CCRA clears the counter and determines the PWM period
2. TnPWM [1:0] =11 therefore the PWM is centre aligned
3. The internal PWM function continues running even when TnBIO [1:0] = 00 or 01
4. CCRA controls the TPnB PWM period and CCRB controls the TPnB PWM duty
5. CCRP will generate an interrupt request when the counter decrements to its zero value

Single Pulse Mode

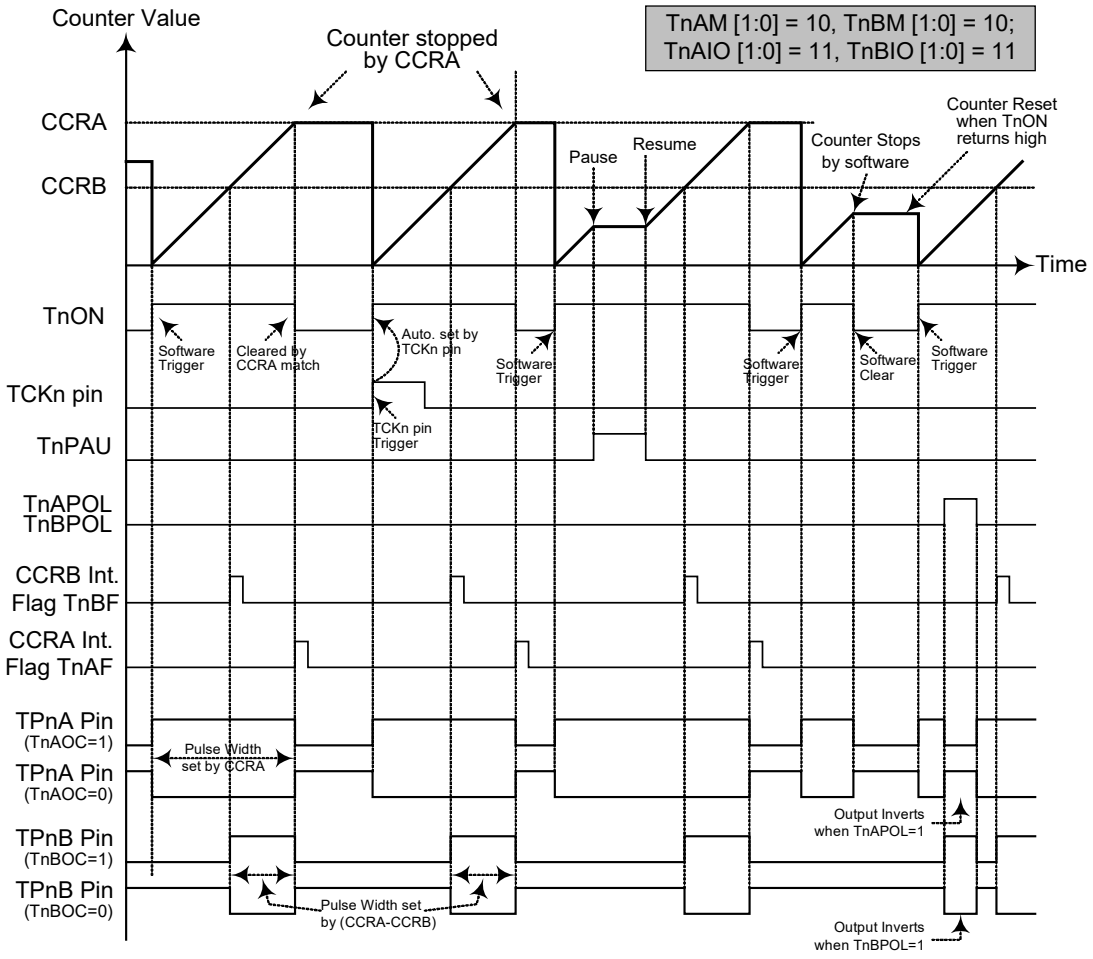
To select this mode, the required bit pairs, T1AM1, T1AM0 and T1BM1, T1BM0 should be set to 10 respectively and also the corresponding T1AIO1, T1AIO0 and T1BIO1, T1BIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the TM output pin.

The trigger for the pulse TP1A output leading edge is a low to high transition of the T1ON bit, which can be implemented using the application program. The trigger for the pulse TP1B output leading edge is a compare match from Comparator B, which can be implemented using the application program. However in the Single Pulse Mode, the T1ON bit can also be made to automatically change from low to high using the external TCK1 pin, which will in turn initiate the Single Pulse output of TP1A. When the T1ON bit transitions to a high level, the counter will start running and the pulse leading edge of TP1A will be generated. The T1ON bit should remain high when the pulse is in its active state. The generated pulse trailing edge of TP1A and TP1B will be generated when the T1ON 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 T1ON bit and thus generate the Single Pulse output trailing edge of TP1A and TP1B. In this way the CCRA value can be used to control the pulse width of TP1A. The CCRA-CCRB value can be used to control the pulse width of TP1B. A compare match from Comparator A and Comparator B will also generate TM interrupts. The counter can only be reset back to zero when the T1ON bit changes from low to high when the counter restarts. In the Single Pulse Mode CCRP is not used. The T1CCLR bit is also not used.



Single Pulse Generation (n=1)



Single Pulse Mode (n=1)

Note: 1. Counter stopped by CCRA

2. CCRP is not used

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

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

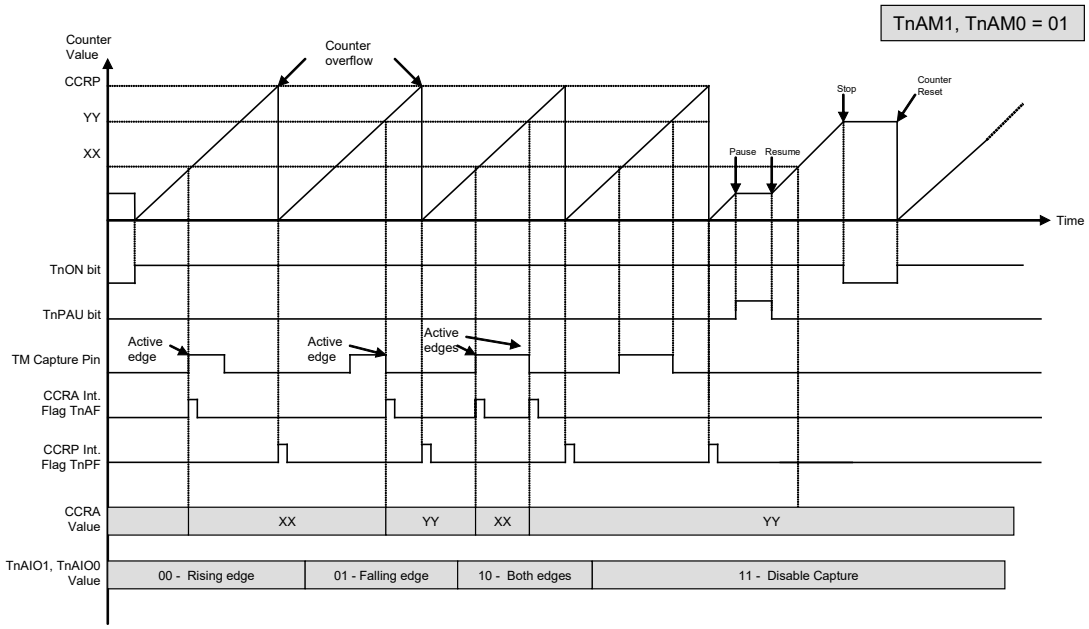
5. In the Single Pulse Mode, TnAIO [1:0] and TnBIO [1:0] must be set to "11" and can not be changed.

Capture Input Mode

To select this mode, bits T1AM1, T1AM0 and T1BM1, T1BM0 in the TM1C1 and TM1C2 registers 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 TP1A and TP1B_0, TP1B_1, TP1B_2 pins, 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 T1AIO1, T1AIO0 and T1BIO1, T1BIO0 bits in the TM1C1 and TM1C2 registers. The counter is started when the T1ON bit changes from low to high which is initiated using the application program.

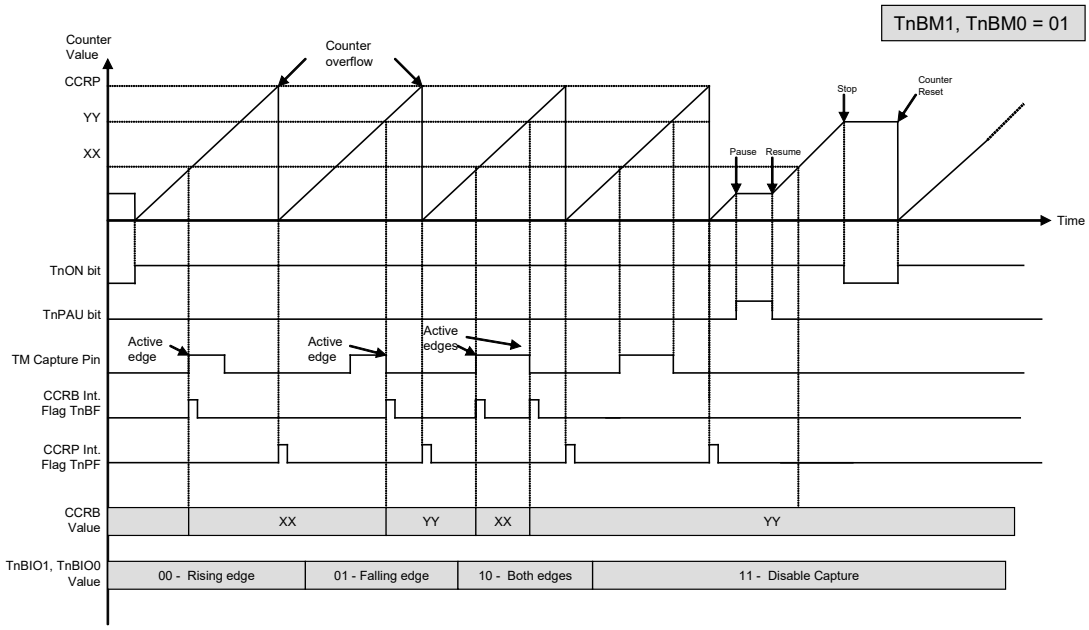
When the required edge transition appears on the TP1A and TP1B_0, TP1B_1, TP1B_2 pins the present value in the counter will be latched into the CCRA and CCRB registers and a TM interrupt generated. Irrespective of what events occur on the TP1A and TP1B_0, TP1B_1, TP1B_2 pins the counter will continue to free run until the T1ON bit changes from high to low. When a CCRP compare match occurs the counter will reset back to zero; in this way the CCRP value can be used to control the maximum counter value. When a CCRP compare match occurs from Comparator P, a TM interrupt will also be generated. Counting the number of overflow interrupt signals from the CCRP can be a useful method in measuring long pulse widths. The T1AIO1, T1AIO0 and T1BIO1, T1BIO0 bits can select the active trigger edge on the TP1A and TP1B_0, TP1B_1, TP1B_2 pins to be a rising edge, falling edge or both edge types. If the T1AIO1, T1AIO0 and T1BIO1, T1BIO0 bits are both set high, then no capture operation will take place irrespective of what happens on the TP1A and TP1B_0, TP1B_1, TP1B_2 pins, however it must be noted that the counter will continue to run.

As the TP1A and TP1B_0, TP1B_1, TP1B_2 pins are pin shared with other functions, care must be taken if the TM 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 T1CCLR, T1AOC, T1BOC, T1APOL and T1BPOL bits are not used in this mode.



ETM CCRA Capture Input Mode (n=1)

Note: 1. TnAM [1:0] = 01 and active edge set by the TnAIO [1:0] bits
2. The TM Capture input pin active edge transfers the counter value to CCRA
3. TnCCLR bit not used
4. No output function – TnAOC and TnAPOL bits not used
5. CCRP determines the counter value and the counter has a maximum count value when CCRP is equal to zero.



ETM CCRB Capture Input Mode (n=1)

Note: 1. TnBM [1:0] = 01 and active edge set by the TnBIO [1:0] bits
2. The TM Capture input pin active edge transfers the counter value to CCRB
3. TnCCLR bit not used
4. No output function – TnBOC and TnBPOL bits 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 – ADC

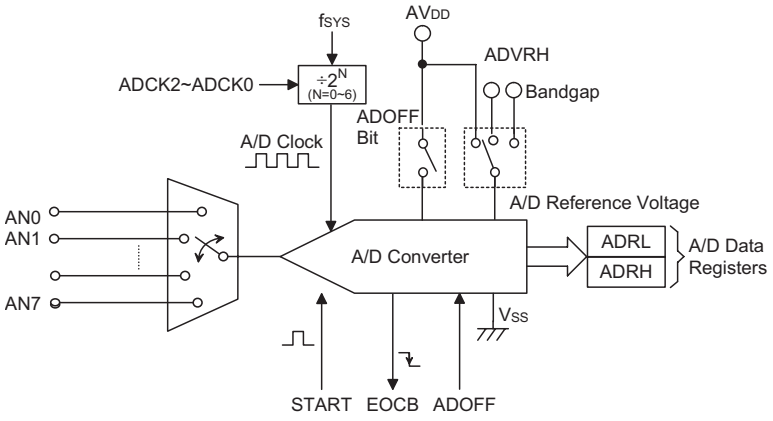
The need to interface to real world analog signals is a common requirement for many electronic systems. However, to properly process these signals by a microcontroller, they must first be converted into digital signals by A/D converters. By integrating the A/D conversion electronic circuitry into the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

A/D Overview

The devices contain a multi-channel analog to digital converter, channel 0~3 are for external signal input, channel 4~6 are for OPA1 application and channel 7 is for OPA2 application. The positive reference voltage is selected by VRPS[1:0] option, and the negative reference voltage is selected by VRNS[1:0].

Device	Input Channels	A/D Channel Select Bits	Input Pins
HT45F65	2	ACS2~ACS0	AN2~NA3
HT45F66/HT45F67	8	ACS2~ACS0	AN0~NA7

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



A/D Converter Structure

A/D Converter Data Registers – ADRL, ADRH

The device, which has an internal 12-bit A/D converter, requires two data registers, a high byte register, known as ADRH, and a low byte register, known as ADRL. After the conversion process takes place, these registers can be directly read by the microcontroller to obtain the digitised conversion value. Only the high byte register, ADRH, utilises its full 8-bit contents. The low byte register utilises only 4 bit of its 8-bit contents as it contains only the lowest bits of the 12-bit converted value.

In the following table, D0~D11 is the A/D conversion data result bits.

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADRL	D3	D2	D1	D0	—	—	—	—
ADRH	D11	D10	D9	D8	D7	D6	D5	D4

A/D Data Registers

• **ADRL, ADRH Register**

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

"x" unknown
"—" unimplemented, read as "0"

D11~D0: ADC conversion data

A/D Converter Control Registers – ADCR0, ADCR1, ADCR2

To control the function and operation of the A/D converter, three control registers known as ADCR0, ADCR1 and ADCR2 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 end of conversion status. The ACS2~ACS0 bits in the ADCR0 register define the ADC input channel number. As the device contains only one actual analog to digital converter hardware circuit, each of the individual 8 analog inputs must be routed to the converter. It is the function of the ACS2~ACS0 bits to determine which analog channel input pin is actually connected to the internal A/D converter.

• **ADCR0 Register**

Bit	7	6	5	4	3	2	1	0
Name	START	EOCB	ADOFF	—	—	ACS2	ACS1	ACS0
R/W	R/W	R	R/W	—	—	R/W	R/W	R/W
POR	0	1	1	—	—	0	0	0

Bit 7 **START**: Start the A/D conversion
0→1→0: Start
0→1 : Reset the A/D converter and set EOCB to "1"
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. When the bit is set high the A/D converter will be reset.

Bit 6 **EOCB**: End of A/D conversion flag
0: A/D conversion ended
1: A/D conversion in progress
This read only flag is used to indicate when an A/D conversion process has completed. When the conversion process is running, the bit will be high.

Bit 5 **ADOFF**: ADC module power on/off control bit
0: ADC module power on
1: ADC module power off
This bit controls the power to the A/D internal function. This bit should be cleared to zero to enable the A/D converter. If the bit is set high then the A/D converter will be switched off reducing the device power consumption. As the A/D converter will consume a limited amount of power, even when not executing a conversion, this may be an important consideration in power sensitive battery powered applications.
Note: 1. It is recommended to set ADOFF=1 before entering IDLE/SLEEP Mode for saving power.
2. ADOFF=1 will power down the ADC module.

Bit 4~3 Unimplemented, read as "0"

Bit 2~0 **ACS2~ACS0**: Select A/D channel
000: AN0
001: AN1
010: AN2
011: AN3
100: AN4, it is connected to pin OP1S0.
101: AN5, it is connected to pin OP1S1.
110: AN6, it is connected to pin OP1S2.
111: AN7, it is connedted to OP2O.
Note that only some channels are available in smaller packages.

• **ADCR1 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	ADCK2	ADCK1	ADCK0
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 Unimplemented, read as "0"

Bit 2~0 **ADCK2~ADCK0**: Select ADC clock source
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: Undefined
These three bits are used to select the clock source for the A/D converter.

• **ADCR2 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	VRPS1	VRPS0	VRNS1	VRNS0
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 **VRPS1, VRPS0**: ADC postive reference voltage selection
00: from AVDD
01: from ADVRH pin
1x: from bandgap
For smaller packages that the ADVRH is not externally bounded, this field cannot be set to "01".

Bit 1~0 **VRNS1, VRNS0**: ADC negative reference voltage selection
00: from AVSS
01: from ADVRL pin
1x: from DACO pin
For smaller packages that the ADVRL is not externally bounded, this field cannot be set to "01".

A/D Operation

The START bit in the ADCR0 register 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 EOCB bit in the ADCR0 register will be set high 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 EOCB bit in the ADCR0 register is used to indicate when the analog to digital conversion process is complete. This bit will be automatically set to "0" by the microcontroller after a conversion cycle has ended. In addition, the corresponding A/D interrupt request flag will be set in the interrupt control register, and if the interrupts are enabled, an appropriate internal interrupt signal will be generated. This A/D internal interrupt signal will direct the program flow to the associated A/D internal interrupt address for processing. If the A/D internal interrupt is disabled, the microcontroller can be used to poll the EOCB bit in the ADCR0 register to check whether it has been cleared as an alternative method of detecting the end of an A/D conversion cycle.

The clock source for the A/D converter, which originates from the system clock f_{SYS}, can be chosen to be either f_{SYS} or a subdivided version of f_{SYS} The division ratio value is determined by the ADCK2~ADCK0 bits in the ADCR1 register.

Controlling the power on/off function of the A/D converter circuitry is implemented using the ADOFF bit in the ADCR0 register. This bit must be zero to power on the A/D converter. When the ADOFF bit is cleared to zero 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, if the ADOFF bit is zero then some power will still be consumed. In power conscious applications it is therefore recommended that the ADOFF is set high to reduce power consumption when the A/D converter function is not being used.

The positive reference voltage supply to the A/D Converter can be supplied from either the positive power supply pin, AVDD, or from an external reference sources supplied on pin ADVRH or from bandgap output. The desired selection is made using the VPRS[1:0] bit.

The negative reference voltage supply to the A/D Converter can be supplied from either the ground power supply pin, AVSS, or from an external reference sources supplied on pin ADVRL or from DAC output. The desired selection is made using the VNRS[1:0] bit.

The ADVRH and ADVRL pins are pin-shared with other functions.

Summary of A/D Conversion Steps

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

- Step 1

  Select the required A/D conversion clock by correctly programming bits ADCK2~ADCK0 in the ADCR1 register.

- Step 2

  Enable the A/D by clearing the ADOFF bit in the ADCR0 register to zero.

- Step 3

  Select which channel is to be connected to the internal A/D converter by correctly programming the ACS2~ACS0 bits which are also contained in the ADCR0 register.

- Step 4

  Select which pins are to be used as A/D inputs and configure related I/O pin-remapping control register.

- Step 5

  Select positive and negative reference voltage at VRPS[1:0] and VRNS[1:0] bits for ADC reference voltage.

- Step 6

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

- Step 7

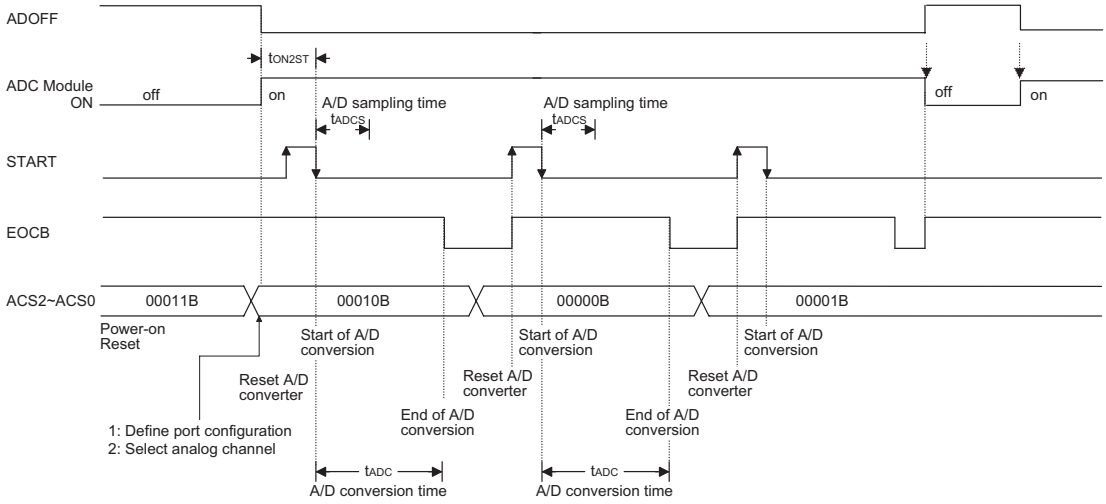
  The analog to digital conversion process can now be initialised by setting the START bit in the ADCR0 register from low to high and then low again. Note that this bit should have been originally cleared to zero.

- Step 8

  To check when the analog to digital conversion process is complete, the EOCB bit in the ADCR0 register can be polled. The conversion process is complete when this bit goes low. When this occurs the A/D data registers ADRL and ADRH can be read to obtain the conversion value. As an alternative method, if the interrupts are enabled and the stack is not full, the program can wait for an A/D interrupt to occur.

Note: When checking for the end of the conversion process, if the method of polling the EOCB bit in the ADCR0 register is used, the interrupt enable step above can be omitted.

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



A/D Conversion Timing

Programming Considerations

During microcontroller operations where the A/D converter is not being used, the A/D internal circuitry can be switched off to reduce power consumption, by setting bit ADOFF high in the ADCR0 register. When this happens, the internal A/D converter circuits will not consume power irrespective of what analog voltage is applied to their input lines. 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 device contains 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, this gives a single bit analog input value of V_{REF} divided by 4096.

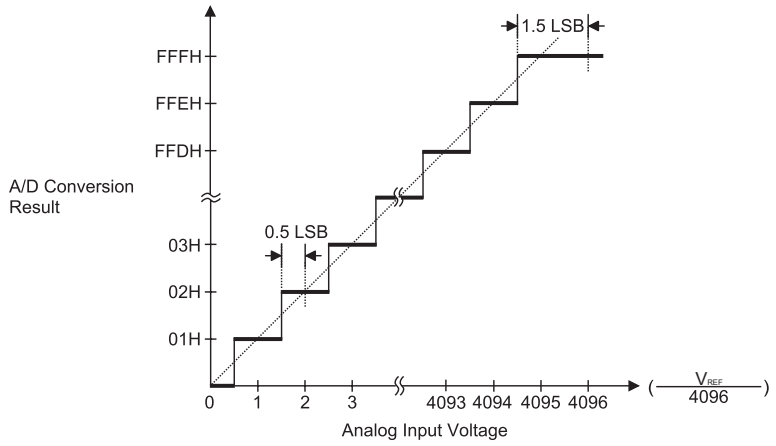
$$1\ LSB = V_{REF} \div 4096$$

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

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

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

Note that here the V_{REF} voltage is the actual A/D converter reference voltage source determined by the ADCR2 register.



Ideal A/D Transfer Function

A/D Programming Example

The following two programming examples illustrate how to setup and implement an A/D conversion. In the first example, the method of polling the EOCB bit in the ADCR0 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  ADCR1, a         ; select fSYS/8 as A/D clock
clr  ADOFF
mov  a, 00H           ; setup ADCR0 register to configure Port as A/D inputs
mov  ADCR0, a         ; and select AN0 to be connected to the A/D converter
:
:
Start_conversion:
clr  START
set  START            ; reset A/D
clr  START            ; start A/D
Polling_EOC:
sz   EOCB             ; poll the ADCR0 register EOCB bit to detect end of A/D
conversion
jmp  polling_EOC      ; continue polling
mov  a, ADRL          ; read low byte conversion result value
mov  adrl_buffer, a   ; save result to user defined register
mov  a, ADRH          ; read high byte conversion result value
mov  adrh_buffer, a   ; save result to user defined register
:
jmp  start_conversion ; start next A/D conversion
```

Note: To power off the ADC, it is necessary to set ADOFF as "1".

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

```
clr  ADE              ; disable ADC interrupt
mov  a, 03H
mov  ADCR1, a         ; select fSYS/8 as A/D clock
clr  ADOFF
mov  a, 00H           ; setup ADCR0 register to configure Port as A/D inputs
mov  ADCR0, a         ; and select AN0 to be connected to the A/D
:
:
Start_conversion:
clr  START
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_:
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, ADRL          ; read low byte conversion result value
mov  adrl_buffer, a   ; save result to user defined register
mov  a, ADRH          ; read high byte conversion result value
mov  adrh_buffer, a   ; save result to user defined register
:
:
EXIT_ISR:
mov  a, status_stack
mov  STATUS, a        ; restore STATUS from user defined memory
mov  a, acc_stack     ; restore ACC from user defined memory
clr  ADF              ; clear ADC interrupt flag
reti
```

Note: To power off the ADC, it is necessary to set ADOFF as "1".

Audio DAC

The devices contain an internal 16-bit DAC function which can be used for audio signal generation.

Audio Output and Volume Control

The voice data must be written into register ADAL and ADAH, with the higher eight bits to ADAH and the lower four bits to the higher nibble of ADRL The audio signal is output on the AUD pin. There are 8 scales of volume controllable level that are provided for the voltage type DAC output. The programmer can change the volume by only writing the volume control data to the VOL[2:0].

Voice Control bit

The voice DAC control bit ADAEN controls DAC circuit enable/disable. If the DAC circuit is not enabled, any ADAH/ADAL output is in valid. Writing a "1" to ADAEN bit is to enable DAC circuit, and writing a "0" to ADAEN bit is to disable DAC circuit.

- **ADAC Register**

Bit	7	6	5	4	3	2	1	0
Name	VOL2	VOL1	VOL0	—	—	—	—	ADAEN
R/W	R/W	R/W	R/W	—	—	—	—	R/W
POR	0	0	0	—	—	—	—	0

Bit 7~5 **VOL2~VOL0**: Volume control
Bit 4~1 Unimplemented, read as "0"
Bit 0 **ADAEN**: Audio DAC control
0: disabled
1: enabled
When this bit is equal to "0", the audio DAC enter power down mode.

- **ADAL Register**

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

Bit 7~4 **D3~D0**: The bit[3:0] of data of audio DAC
Bit 3~0 Unimplemented, read as "0"

- **ADAH Register**

Bit	7	6	5	4	3	2	1	0
Name	D11	D10	D9	D8	D7	D6	D5	D4
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 **D11~D4**: The bit[11:4] of data of audio DAC

Digital to Analog Converter – DAC

This device includes a 10-bit DAC which provides 0 to 0.5 DACVREF to the non-inverting input of OPA, and is implemented by DAL and DAH setting. The DACVREF input reference voltage is decided by BGOS[1:0].

• DAC Register

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

Bit 7~1 Unimplemented, read as "0"
Bit 0 **DAEN**: 10-bit DAC control
0: disabled
1: enabled

• DAL Register

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

Bit 7~6 **D1, D0**: The bit[1:0] of data of 10-bit DAC
Bit 5~0 Unimplemented, read as "0"

• DAH Register

Bit	7	6	5	4	3	2	1	0
Name	D9	D8	D7	D6	D5	D4	D3	D2
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 **D9~D2**: The bit[9:2] of data of 10-bit DAC

Operational Amplifier

This chip builds in OPA1 and OPA2 for measure application. The 10-bit DAC offers a 0 to 0.5 DAC reference voltage to non-inverting input of OPA1 and OPA2. The OPA1 supports 3 different amplification by SW0~SW2 setting, and then outputs to channel 4~6 of ADC.

The OPA2 is for temperature measure and external signal measure and outputs to channel 7 of ADC.

The OPA1 and OPA2 are controlled by register OPC1 and OPC2 setting.

The user can control switch 0~5 for vary application. Please refer to register TSC for switch control.

• OPC1 Register

Bit	7	6	5	4	3	2	1	0
Name	OP1_AOFM	OP1_ARS	OP1EN	OP1AOF4	OP1AOF3	OP1AOF2	OP1AOF1	OP1AOF0
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 **OP1_AOFM**: Input offset voltage cancellation mode and operational amplifier mode selection
0: operational amplifier mode
1: input offset voltage cancellation mode

Bit 6 **OP1_ARS**: Operational amplifier input offset voltage cancellation reference selection bit
0: select OP1N as the reference input
1: select OP1P as the reference input
When OP1_AOFM (S3C) = 0, S2C and S1C will be closed at the same time.
When OP1_AOFM = 1, it will be set OP1_ARS to select OP1P or OP1N as a reference point.

Bit 5 **OP1EN**: Operational amplifier 1 control
0: disabled
1: enabled

Bit 4~0 **OP1AOF4~OP1AOF0**: Operational amplifier input offset voltage cancellation control bits

• OPC2 Register

Bit	7	6	5	4	3	2	1	0
Name	OP2_AOFM	OP2_ARS	OP2EN	OP2AOF4	OP2AOF3	OP2AOF2	OP2AOF1	OP2AOF0
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 **OP2_AOFM**: Input offset voltage cancellation mode and operational amplifier mode selection
0: operational amplifier mode
1: input offset voltage cancellation mode

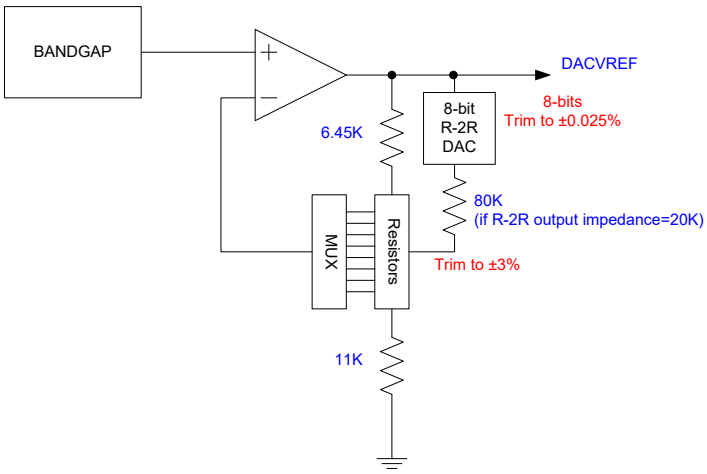
Bit 6 **OP2_ARS**: Operational amplifier input offset voltage cancellation reference selection bit
0: select OP2N as the reference input
1: select OP2P as the reference input
When OP2_AOFM (S3C) = 0, S2C and S1C will be closed at the same time.
When OP2_AOFM = 1, it will be set OP2_ARS to select OP2P or OP2N as a reference point.

Bit 5 **OP2EN**: Operational amplifier 2 control
0: disabled
1: enabled

Bit 4~0 **OP2AOF4~OP2AOF0**: Operational amplifier input offset voltage cancellation control bits

Bandgap

The bandgap circuit is as followring figure, which consists of 8-bit DAC, bandgap, OPA and resistors and provides accurate reference voltage to 12-bit ADC and 10-bit DAC, and output reference voltage is selected by BGOS[1:0]. The 8-bit DAC offers a more accurate reference voltage to other circuit.



• **BGC Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	BGEN	—	—	BGOS1	BGOS0
R/W	—	—	—	R/W	—	—	R/W	R/W
POR	—	—	—	0	—	—	0	0

Bit 7~5 Unimplemented, read as "0"

Bit 4 **BGEN**: Bandgap control
0: disable
1: enable
When this bit is equal to 0, the bandgap enters power down mode.

Bit 3~2 Unimplemented, read as "0"

Bit 1~0 **BGOS1, BGOS0**: Bandgap output selection.
0x: 2.0V
10: AV_{DD}
11: high impedance (floating)

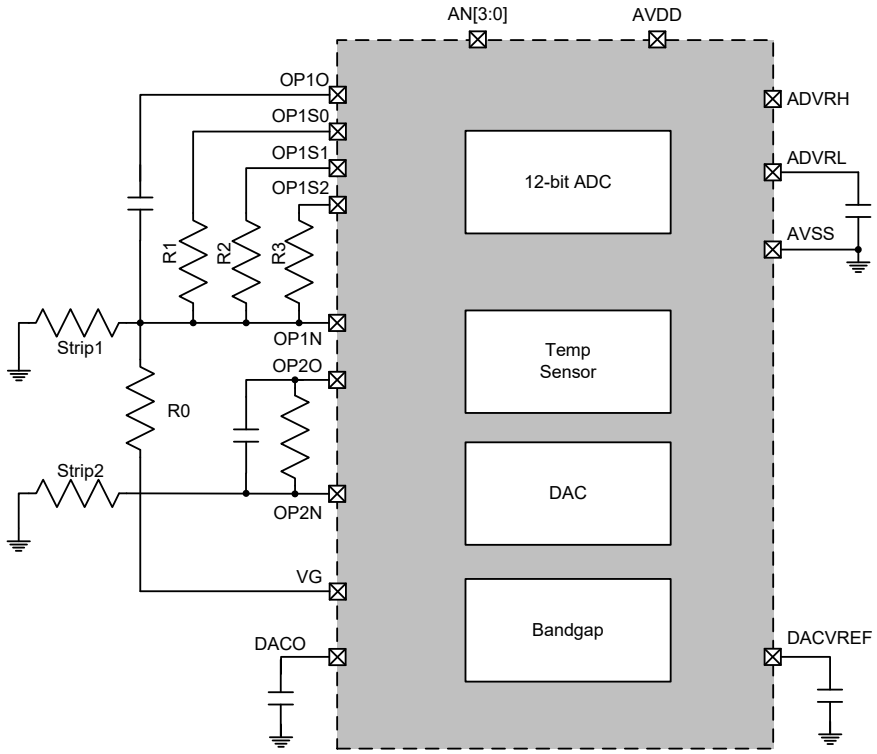
• **PVREF Register**

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

Bit 7~0 **D7~D0**: 8-bit DAC output for band gap fine turn

Analog Application Circuit

The analog application circuit is as followring figure, which consists of 12-bit ADC, bandgap, 10-bit DAC, temperature sensor and operational amplifier for special function application. Each function is described in the above section. When the analog application circuit is applied, please refer to this diagram.



Analog application diagram

Serial Interface Module – SIM

The devices contain a Serial Interface Module, which includes both the four line SPI interface and the two line I^2C interface types, to allow an easy method of communication with external peripheral hardware. Having relatively simple communication protocols, these serial interface types allow the microcontroller to interface to external SPI or I^2C based hardware such as sensors, Flash memory, etc. The SIM interface pins are pin-shared with other I/O pins therefore the SIM interface function must first be enabled using the SIMC0 register. As both interface types share the same pins and registers, the choice of whether the SPI or I^2C type is used is made using the SIM operating mode control bits, named SIM2~SIM0, in the SIMC0 register. These pull-high resistors of the SIM pin-shared I/O are selected using pull-high control registers, and also if the SIM function is enabled.

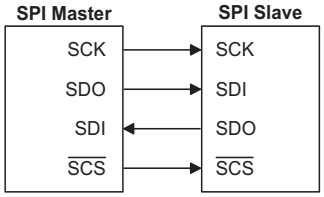
SPI Interface

This SPI interface function which is contained within the Serial Interface Module, should not be confused with the other independent SPI function, known as SPI1, which is described in another section of this datasheet. The SPI interface is often used to communicate with external peripheral devices such as sensors, Flash memory devices etc. Originally developed by Motorola, the four line SPI interface is a synchronous serial data interface that has a relatively simple communication protocol simplifying the programming requirements when communicating with external hardware devices.

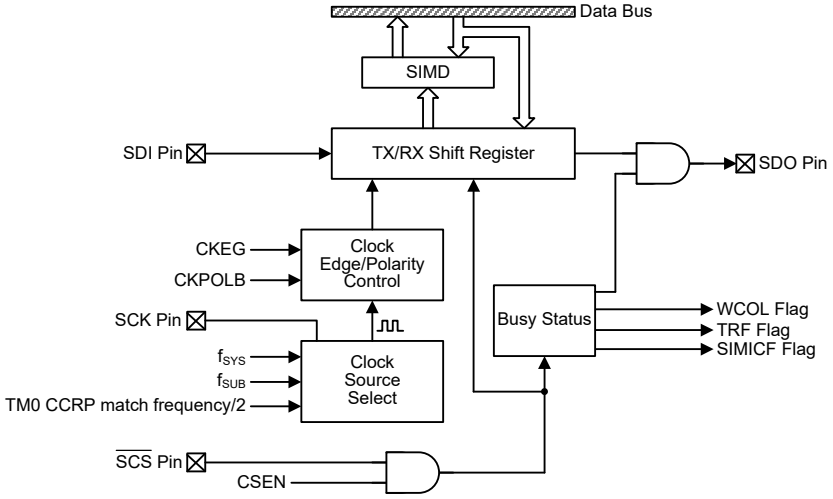
The communication is full duplex and operates as a slave/master type, where the device can be either master or slave. Although the SPI interface specification can control multiple slave devices from a single master, but this device provided only one $\overline{SCS}$ pin. If the master needs to control multiple slave devices from a single master, the master can use I/O pin to select the slave devices.

SPI Interface Operation

The SPI interface is a full duplex synchronous serial data link. It is a four line interface with pin names SDI, SDO, SCK and $\overline{SCS}$. Pins SDI and SDO are the Serial Data Input and Serial Data Output lines, SCK is the Serial Clock line and $\overline{SCS}$ is the Slave Select line. As the SPI interface pins are pin-shared with normal I/O pins and with the I^2C function pins, the SPI interface must first be enabled by setting the correct bits in the SIMC0 and SIMC2 registers. Communication between devices connected to the SPI interface is carried out in a slave/master mode with all data transfer initiations being implemented by the master. The Master also controls the clock signal. As the device only contains a single $\overline{SCS}$ pin only one slave device can be utilized. The $\overline{SCS}$ pin is controlled by software, set CSEN bit to "1" to enable $\overline{SCS}$ pin function, set CSEN bit to "0" the $\overline{SCS}$ pin will be floating state.



SPI Master/Slave Connection



SPI Block Diagram

The SPI function in this device offers the following features:

- Full duplex synchronous data transfer
- Both Master and Slave modes
- LSB first or MSB first data transmission modes
- Transmission complete flag
- Rising or falling active clock edge
- WCOL and CSEN bit enabled or disable select

The status of the SPI interface pins is determined by a number of factors such as whether the device is in the master or slave mode and upon the condition of certain control bits such as CSEN and SIMEN.

SPI Registers

There are three internal registers which control the overall operation of the SPI interface. These are the SIMD data register and two registers SIMC0 and SIMC2. Note that the SIMC1 register is only used by the I²C interface.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	PCKEN	PCKP1	PCKP0	SIMEN	SIMCF
SIMD	D7	D6	D5	D4	D3	D2	D1	D0
SIMC2	D7	D6	CKPOLB	CKEG	MLS	CSEN	WCOL	TRF

SPI Register List

The SIMD register is used to store the data being transmitted and received. The same register is used by both the SPI and I²C functions. Before the device writes data to the SPI bus, the actual data to be transmitted must be placed in the SIMD register. After the data is received from the SPI bus, the device can read it from the SIMD register. Any transmission or reception of data from the SPI bus must be made via the SIMD register.

- **SIMD Register**

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

"x" unknown

There are also two control registers for the SPI interface, SIMC0 and SIMC2. Note that the SIMC2 register also has the name SIMA which is used by the I²C function. The SIMC1 register is not used by the SPI function, only by the I²C function. Register SIMC0 is used to control the enable/disable function and to set the data transmission clock frequency. Although not connected with the SPI function, the SIMC0 register is also used to control the Peripheral Clock Prescaler. Register SIMC2 is used for other control functions such as LSB/MSB selection, write collision flag etc.

- **SIMC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	PCKEN	PCKP1	PCKP0	SIMEN	SIMCF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	0	0	0	0	0

Bit 7~5 **SIM2~SIM0**: SIM Operating Mode Control
000: SPI master mode; SPI clock is $f_{SYS}/4$
001: SPI master mode; SPI clock is $f_{SYS}/16$
010: SPI master mode; SPI clock is $f_{SYS}/64$
011: SPI master mode; SPI clock is f_{SUB}
100: SPI master mode; SPI clock is TM0 CCRP match frequency/2
101: SPI slave mode
110: I²C slave mode
111: Unused mode
These bits setup the overall operating mode of the SIM function. As well as selecting if the I²C or SPI function, they are used to control the SPI Master/Slave selection and the SPI Master clock frequency. The SPI clock is a function of the system clock but can also be chosen to be sourced from the TM0. If the SPI Slave Mode is selected then the clock will be supplied by an external Master device.

Bit 4 **PCKEN**: PCK Output Pin Control
0: Disable
1: Enable

Bit 3~2 **PCKP1, PCKP0**: Select PCK output pin frequency
00: f_{SYS}
01: $f_{SYS}/4$
10: $f_{SYS}/8$
11: TM0 CCRP match frequency/2

Bit 1 **SIMEN**: SIM Control
0: Disable
1: Enable
The bit is the overall on/off control for the SIM interface. When the SIMEN bit is cleared to zero to disable the SIM interface, the SDI, SDO, SCK and $\overline{SCS}$, or SDA and SCL lines will be in a floating condition and the SIM operating current will be reduced to a minimum value. When the bit is high the SIM interface is enabled. If the SIM is configured to operate as an SPI interface via the SIM2~SIM0 bits, the contents of the SPI control registers will remain at the previous settings when the SIMEN bit changes from low to high and should therefore be first initialised by the application program. If the SIM is configured to operate as an I^2C interface via the SIM2~SIM0 bits and the SIMEN bit changes from low to high, the contents of the I^2C control bits such as HTX and TXAK will remain at the previous settings and should therefore be first initialised by the application program while the relevant I^2C flags such as HCF, HAAS, HBB, SRW and RXAK will be set to their default states.

Bit 0 **SIMCF**: SIM transfer incomplete flag for SPI interface
0: No SPI incomplete transfer occurs
1: SPI incomplete transfer occurred
The SIMCF bit is only used for the SPI slave device to indicate whether the SPI transfer is complete or not. When the SPI transfer is in progress and the $\overline{SCS}$ pin is set to "1", the SPI transfer will be stopped and the SIMCF flag will be set to 1 by hareware.

- **SIMC2 Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	CKPOLB	CKEG	MLS	CSEN	WCOL	TRF
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 **D7, D6**: Undefined bit
This bit can be read or written by user software program.

Bit 5 **CKPOLB**: Determines the base condition of the clock line
0: the SCK line will be high when the clock is inactive
1: the SCK line will be low when the clock is inactive
The CKPOLB bit determines the base condition of the clock line, if the bit is high, then the SCK line will be low when the clock is inactive. When the CKPOLB bit is low, then the SCK line will be high when the clock is inactive.

Bit 4 **CKEG**: Determines SPI SCK active clock edge type
CKPOLB=0
0: SCK is high base level and data capture at SCK rising edge
1: SCK is high base level and data capture at SCK falling edge
CKPOLB=1
0: SCK is low base level and data capture at SCK falling edge
1: SCK is low base level and data capture at SCK rising edge
The CKEG and CKPOLB bits are used to setup the way that the clock signal outputs and inputs data on the SPI bus. These two bits must be configured before data transfer is executed otherwise an erroneous clock edge may be generated. The CKPOLB bit determines the base condition of the clock line, if the bit is high, then the SCK line will be low when the clock is inactive. When the CKPOLB bit is low, then the SCK line will be high when the clock is inactive. The CKEG bit determines active clock edge type which depends upon the condition of CKPOLB bit.

Bit 3 **MLS**: SPI Data shift order
0: LSB
1: MSB

This is the data shift select bit and is used to select how the data is transferred, either MSB or LSB first. Setting the bit high will select MSB first and low for LSB first.

Bit 2 **CSEN**: SPI $\overline{SCS}$ pin Control
0: Disable
1: Enable

The CSEN bit is used as an enable/disable for the $\overline{SCS}$ pin. If this bit is low, then the $\overline{SCS}$ pin will be disabled and placed into a floating condition. If the bit is high the $\overline{SCS}$ pin will be enabled and used as a select pin.

Bit 1 **WCOL**: SPI Write Collision flag
0: No collision
1: Collision

The WCOL flag is used to detect if a data collision has occurred. If this bit is high it means that data has been attempted to be written to the SIMD register during a data transfer operation. This writing operation will be ignored if data is being transferred. The bit can be cleared by the application program.

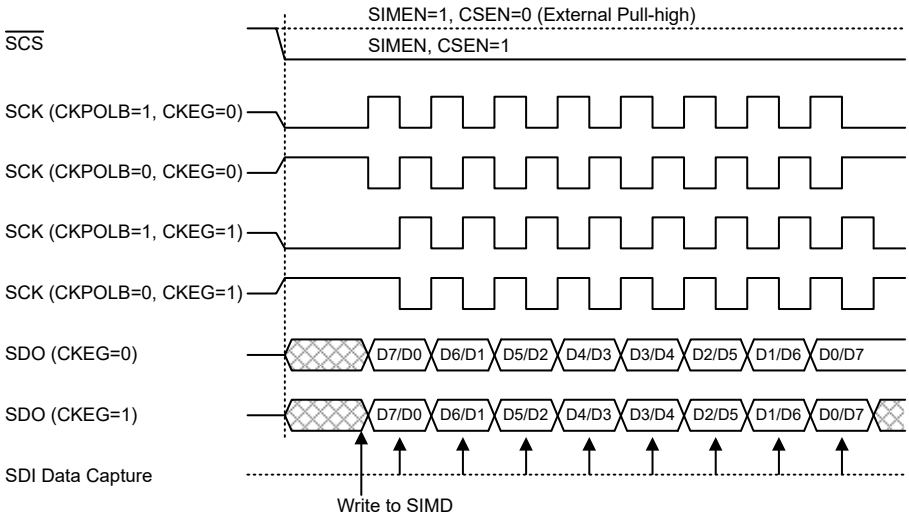
Bit 0 **TRF**: SPI Transmit/Receive Complete flag
0: Data is being transferred
1: SPI data transmission is completed

The TRF bit is the Transmit/Receive Complete flag and is set "1" automatically when an SPI data transmission is completed, but must set to "0" by the application program. It can be used to generate an interrupt.

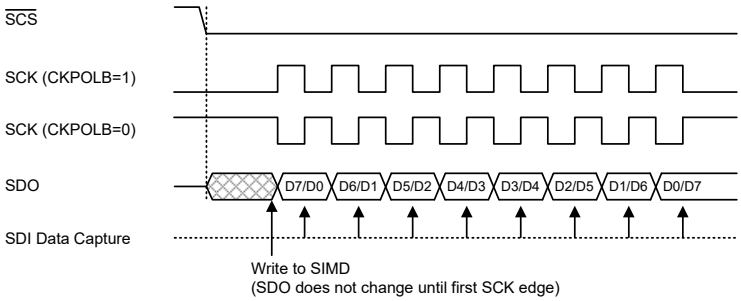
SPI Communication

After the SPI interface is enabled by setting the SIMEN bit high, then in the Master Mode, when data is written to the SIMD register, transmission/reception will begin simultaneously. When the data transfer is complete, the TRF flag will be set automatically, but must be cleared using the application program. In the Slave Mode, when the clock signal from the master has been received, any data in the SIMD register will be transmitted and any data on the SDI pin will be shifted into the SIMD register. The master should output an $\overline{SCS}$ signal to enable the slave device before a clock signal is provided. The slave data to be transferred should be well prepared at the appropriate moment relative to the SCK signal depending upon the configurations of the CKPOLB bit and CKEG bit. The accompanying timing diagram shows the relationship between the slave data and SCK signal for various configurations of the CKPOLB and CKEG bits.

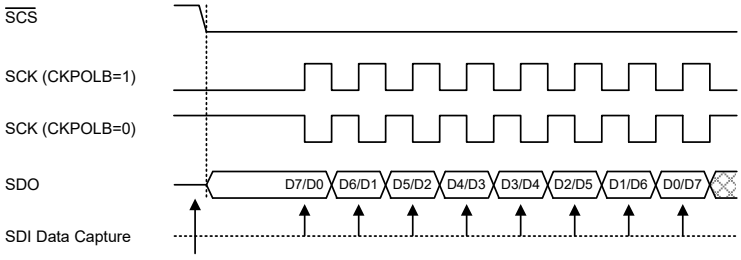
The SPI Master mode will continue to function if the SPI clock is running.



SPI Master Mode Timing

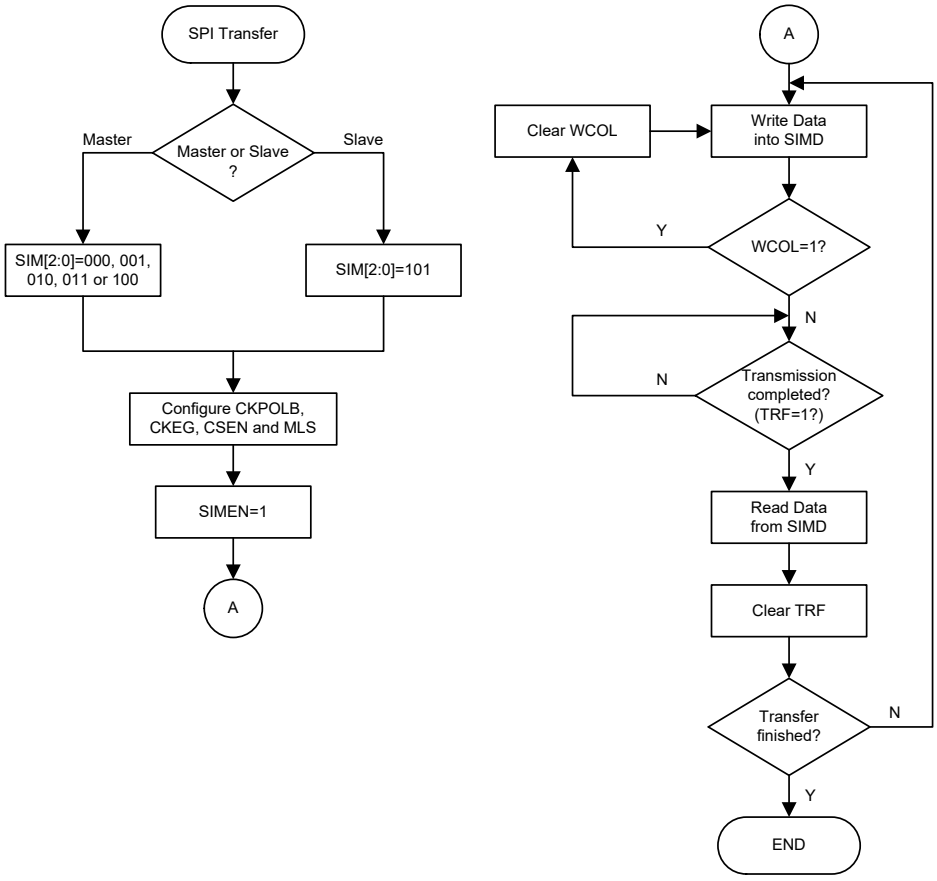


SPI Slave Mode Timing – CKEG=0



Note: For SPI slave mode, if SIMEN=1 and CSEN=0, SPI is always enabled and ignores the $\overline{SCS}$ level.

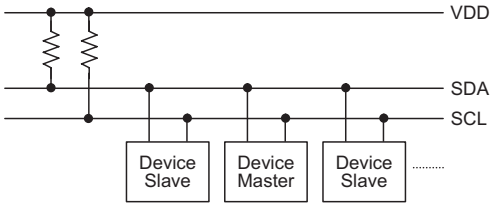
SPI Slave Mode Timing – CKEG=1



SPI Transfer Control Flowchart

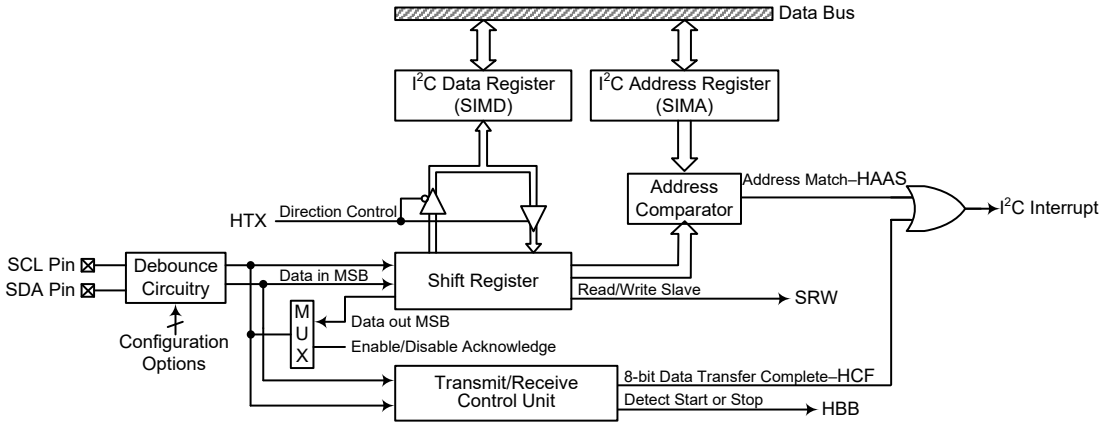
I²C Interface

The I²C interface is used to communicate with external peripheral devices such as sensors etc. Originally developed by Philips, it is a two line low speed serial interface for synchronous serial data transfer. The advantage of only two lines for communication, relatively simple communication protocol and the ability to accommodate multiple devices on the same bus has made it an extremely popular interface type for many applications.



I²C Master/Slave Bus Connection

I²C Interface Operation

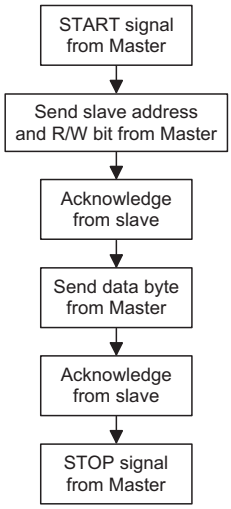


I²C Block Diagram

The I²C serial interface is a two line interface, a serial data line, SDA, and serial clock line, SCL. As many devices may be connected together on the same bus, their outputs are both open drain types. For this reason it is necessary that external pull-high resistors are connected to these outputs. Note that no chip select line exists, as each device on the I²C bus is identified by a unique address which will be transmitted and received on the I²C bus.

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

A configuration option exists to determine the debounce time of the I²C interface. This uses the internal clock to in effect add a debounce time to the external clock to reduce the possibility of glitches on the clock line causing erroneous operation. The debounce time, if selected, can be chosen to be either 1 or 2 system clocks.



I²C Interface Operation

I²C Registers

There are three control registers associated with the I²C bus, SIMC0, SIMC1 and SIMA and one data register, SIMD. The SIMD register, which is shown in the above SPI section, is used to store the data being transmitted and received on the I²C bus. Before the microcontroller writes data to the I²C bus, the actual data to be transmitted must be placed in the SIMD register. After the data is received from the I²C bus, the microcontroller can read it from the SIMD register. Any transmission or reception of data from the I²C bus must be made via the SIMD register.

Note that the SIMA register also has the name SIMC2 which is used by the SPI function. Bit SIMEN and bits SIM2~SIM0 in register SIMC0 are used by the I²C interface.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	PCKEN	PCKP1	PCKP0	SIMEN	D0
SIMC1	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
SIMD	D7	D6	D5	D4	D3	D2	D1	D0
SIMCA	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	D0

I²C Register List

• SIMC0 Register

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	PCKEN	PCKP1	PCKP0	SIMEN	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	0	0	0	0	0

Bit 7~5 **SIM2~SIM0**: SIM Operating Mode Control
000: SPI master mode; SPI clock is $f_{SYS}/4$
001: SPI master mode; SPI clock is $f_{SYS}/16$
010: SPI master mode; SPI clock is $f_{SYS}/64$
011: SPI master mode; SPI clock is f_{SUB}
100: SPI master mode; SPI clock is TM0 CCRP match frequency/2
101: SPI slave mode
110: I²C slave mode
111: Unused mode

These bits setup the overall operating mode of the SIM function. As well as selecting if the I²C or SPI function, they are used to control the SPI Master/Slave selection and the SPI Master clock frequency. The SPI clock is a function of the system clock but can also be chosen to be sourced from the TM0. If the SPI Slave Mode is selected then the clock will be supplied by an external Master device.

Bit 4 **PCKEN**: PCK Output Pin Control
0: Disable
1: Enable

Bit 3~2 **PCKP1~PCKP0**: Select PCK output pin frequency
00: f_{SYS}
01: $f_{SYS}/4$
10: $f_{SYS}/8$
11: TM0 CCRP match frequency/2

Bit 1 **SIMEN**: SIM Control
0: Disable
1: Enable

The bit is the overall on/off control for the SIM interface. When the SIMEN bit is cleared to zero to disable the SIM interface, the SDI, SDO, SCK and $\overline{SCS}$, or SDA and SCL lines will be in a floating condition and the SIM operating current will be reduced to a minimum value. When the bit is high the SIM interface is enabled. If the SIM is configured to operate as an SPI interface via the SIM2~SIM0 bits, the contents of the SPI control registers will remain at the previous settings when the SIMEN bit changes from low to high and should therefore be first initialised by the application program. If the SIM is configured to operate as an I²C interface via the SIM2~SIM0 bits and the SIMEN bit changes from low to high, the contents of the I²C control bits such as HTX and TXAK will remain at the previous settings and should therefore be first initialised by the application program while the relevant I²C flags such as HCF, HAAS, HBB, SRW and RXAK will be set to their default states.

Bit 0 Undefined bit, can be read or written by user software program.

- **SIMC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
R/W	R	R	R	R/W	R/W	R	R/W	R
POR	1	0	0	0	0	0	0	1

Bit 7 **HCF**: I²C Bus data transfer completion flag
0: Data is being transferred
1: Completion of an 8-bit data transfer

The HCF flag is the data transfer flag. This flag will be zero when data is being transferred. Upon completion of an 8-bit data transfer the flag will go high and an interrupt will be generated.

Bit 6 **HAAS**: I²C Bus address match flag
0: Not address match
1: Address match

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

Bit 5 **HBB**: I²C Bus busy flag
0: I²C Bus is not busy
1: I²C Bus is busy

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

Bit 4 **HTX**: Select I²C slave device is transmitter or receivera
0: Slave device is the receiver
1: Slave device is the transmitter

Bit 3 **TXAK**: I²C Bus transmit acknowledge flag
0: Slave send acknowledge flag
1: Slave do not send acknowledge flag
The TXAK bit is the transmit acknowledge flag. After the slave device receipt of 8-bits of data, this bit will be transmitted to the bus on the 9th clock from the slave device. The slave device must always set TXAK bit to "0" before further data is received.

Bit 2 **SRW**: I²C Slave Read/Write flag
0: Slave device should be in receive mode
1: Slave device should be in transmit mode
The SRW flag is the I²C Slave Read/Write flag. This flag determines whether the master device wishes to transmit or receive data from the I²C bus. When the transmitted address and slave address is match, that is when the HAAS flag is set high, the slave device will check the SRW flag to determine whether it should be in transmit mode or receive mode. If the SRW flag is high, the master is requesting to read data from the bus, so the slave device should be in transmit mode. When the SRW flag is zero, the master will write data to the bus, therefore the slave device should be in receive mode to read this data.

Bit 1 **IAMWU**: I²C Address Match Wake-up Control
0: Disable
1: Enable
This bit should be set to "1" to enable I²C address match wake up from SLEEP or IDLE Mode.

Bit 0 **RXAK**: I²C Bus Receive acknowledge flag
0: Slave receive acknowledge flag
1: Slave do not receive acknowledge flag

• SIMD Register

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

"x" unknown

• SIMA Register

Bit	7	6	5	4	3	2	1	0
Name	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x" unknown

Bit 7~1 **IICA6~IICA0**: I²C slave address
IICA6~ IICA0 is the I²C slave address bit 6 ~ bit 0.
The SIMA register is also used by the SPI interface but has the name SIMC2. The SIMA register is the location where the 7-bit slave address of the slave device is stored. Bits 7~ 1 of the SIMA register define the device slave address. Bit 0 is not defined.
When a master device, which is connected to the I²C bus, sends out an address, which matches the slave address in the SIMA register, the slave device will be selected. Note that the SIMA register is the same register address as SIMC2 which is used by the SPI interface.

Bit 0 Undefined bit, can be read or written by user software program.

I²C Bus Communication

Communication on the I²C bus requires four separate steps, a START signal, a slave device address transmission, a data transmission and finally a STOP signal. When a START signal is placed on the I²C bus, all devices on the bus will receive this signal and be notified of the imminent arrival of data on the bus. The first seven bits of the data will be the slave address with the first bit being the MSB. If the address of the slave device matches that of the transmitted address, the HAAS bit in the SIMC1 register will be set and an I²C interrupt will be generated. After entering the interrupt service routine, the slave device must first check the condition of the HAAS bit to determine whether the interrupt source originates from an address match or from the completion of an 8-bit data transfer. During a data transfer, note that after the 7-bit slave address has been transmitted, the following bit, which is the 8th bit, is the read/write bit whose value will be placed in the SRW bit. This bit will be checked by the slave device to determine whether to go into transmit or receive mode. Before any transfer of data to or from the I²C bus, the microcontroller must initialise the bus, the following are steps to achieve this:

- Step 1

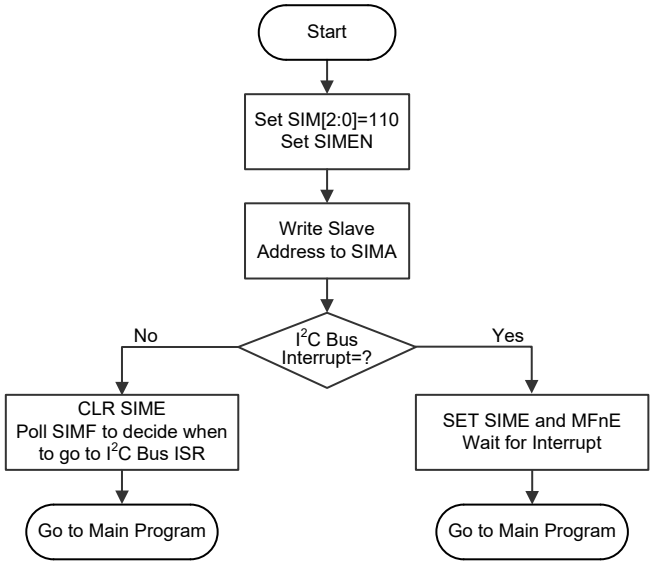
  Set the SIM2~SIM0 bits to "110" and SIMEN bit to "1" in the SIMC0 register to enable the I²C bus.

- Step 2

  Write the slave address of the device to the I²C bus address register SIMA.

- Step 3

  Set the SIME and SIM Muti-Function interrupt enable bit of the interrupt control register to enable the SIM interrupt and Multi-function interrupt.



I²C Bus Initialisation Flow Chart

I²C Bus Start Signal

The START signal can only be generated by the master device connected to the I²C bus and not by the slave device. This START signal will be detected by all devices connected to the I²C bus. When detected, this indicates that the I²C bus is busy and therefore the HBB bit will be set. A START condition occurs when a high to low transition on the SDA line takes place when the SCL line remains high.

Slave Address

The transmission of a START signal by the master will be detected by all devices on the I²C bus. To determine which slave device the master wishes to communicate with, the address of the slave device will be sent out immediately following the START signal. All slave devices, after receiving this 7-bit address data, will compare it with their own 7-bit slave address. If the address sent out by the master matches the internal address of the microcontroller slave device, then an internal I²C bus interrupt signal will be generated. The next bit following the address, which is the 8th bit, defines the read/write status and will be saved to the SRW bit of the SIMC1 register. The slave device will then transmit an acknowledge bit, which is a low level, as the 9th bit. The slave device will also set the status flag HAAS when the addresses match.

As an I²C bus interrupt can come from two sources, when the program enters the interrupt subroutine, the HAAS bit should be examined to see whether the interrupt source has come from a matching slave address or from the completion of a data byte transfer. When a slave address is matched, the device must be placed in either the transmit mode and then write data to the SIMD register, or in the receive mode where it must implement a dummy read from the SIMD register to release the SCL line.

I²C Bus Read/Write Signal

The SRW bit in the SIMC1 register defines whether the master device wishes to read data from the I²C bus or write data to the I²C bus. The slave device should examine this bit to determine if it is to be a transmitter or a receiver. If the SRW flag is "1" then this indicates that the master device wishes to read data from the I²C bus, therefore the slave device must be setup to send data to the I²C bus as a transmitter. If the SRW flag is "0" then this indicates that the master wishes to send data to the I²C bus, therefore the slave device must be setup to read data from the I²C bus as a receiver.

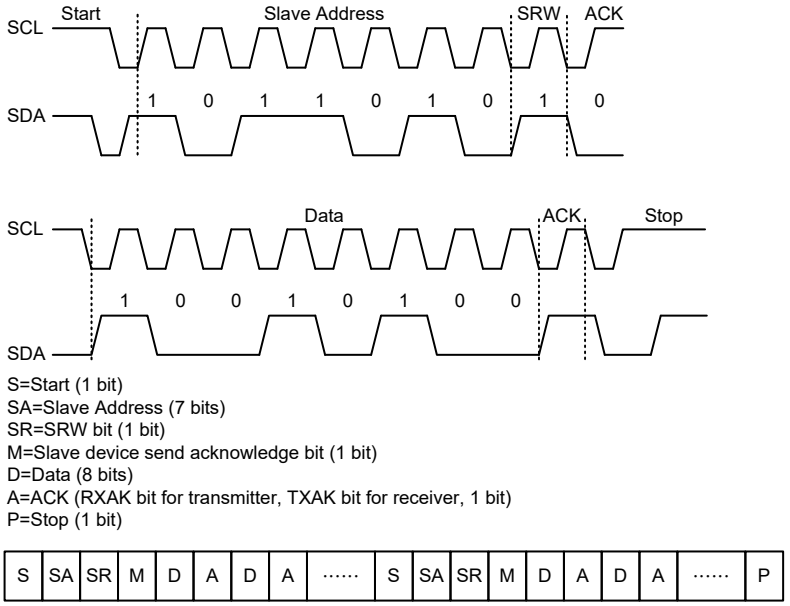
I²C Bus Slave Address Acknowledge Signal

After the master has transmitted a calling address, any slave device on the I²C bus, whose own internal address matches the calling address, must generate an acknowledge signal. The acknowledge signal will inform the master that a slave device has accepted its calling address. If no acknowledge signal is received by the master then a STOP signal must be transmitted by the master to end the communication. When the HAAS flag is high, the addresses have matched and the slave device must check the SRW flag to determine if it is to be a transmitter or a receiver. If the SRW flag is high, the slave device should be setup to be a transmitter so the HTX bit in the SIMC1 register should be set to "1". If the SRW flag is low, then the microcontroller slave device should be setup as a receiver and the HTX bit in the SIMC1 register should be set to "0".

I²C Bus Data and Acknowledge Signal

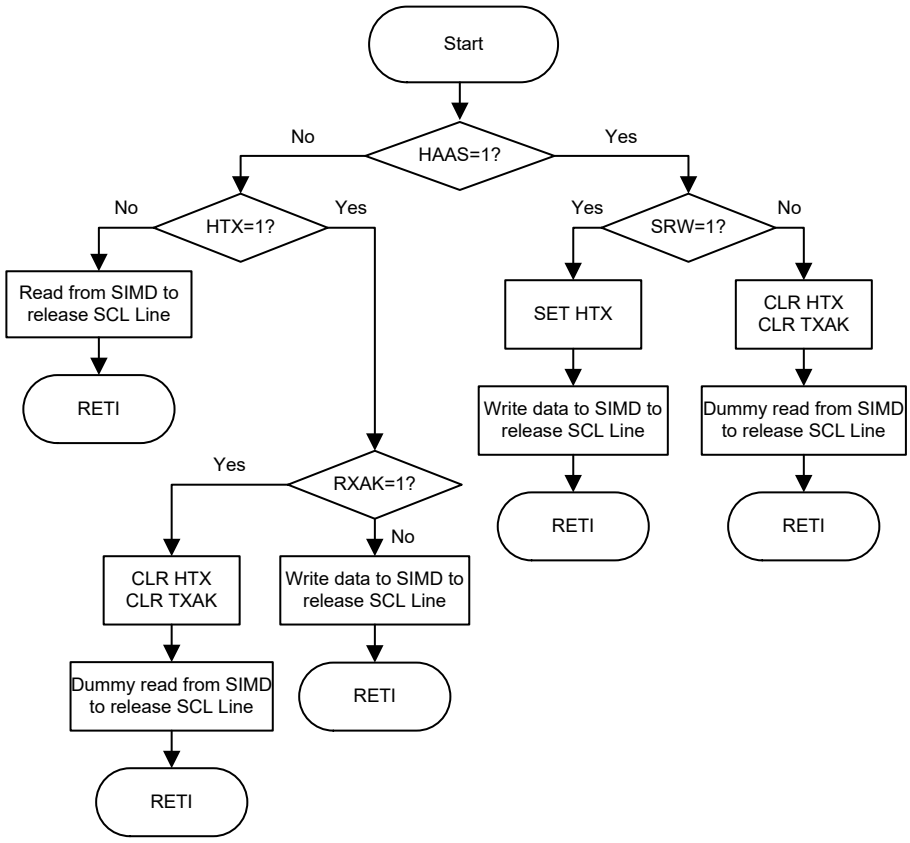
The transmitted data is 8-bits wide and is transmitted after the slave device has acknowledged receipt of its slave address. The order of serial bit transmission is the MSB first and the LSB last. After receipt of 8-bits of data, the receiver must transmit an acknowledge signal, level "0", before it can receive the next data byte. If the slave transmitter does not receive an acknowledge bit signal from the master receiver, then the slave transmitter will release the SDA line to allow the master to send a STOP signal to release the I²C Bus. The corresponding data will be stored in the SIMD register. If setup as a transmitter, the slave device must first write the data to be transmitted into the SIMD register. If setup as a receiver, the slave device must read the transmitted data from the SIMD register.

When the slave receiver receives the data byte, it must generate an acknowledge bit, known as TXAK, on the 9th clock. The slave device, which is setup as a transmitter will check the RXAK bit in the SIMC1 register to determine if it is to send another data byte, if not then it will release the SDA line and await the receipt of a STOP signal from the master.



I²C Communication Timing Diagram

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



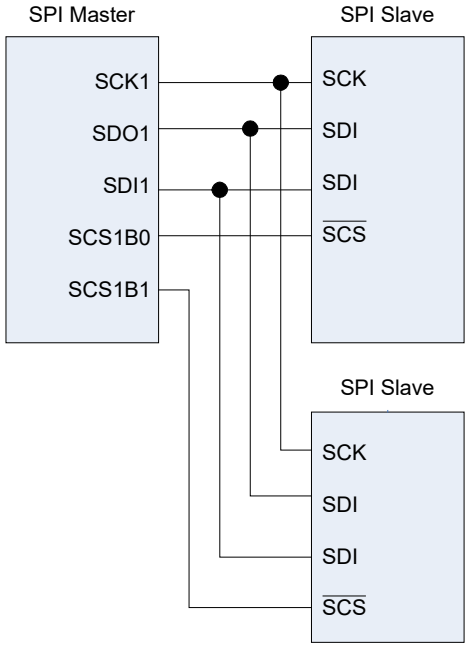
I²C Bus ISR Flow Chart

SPI1 Interface

The SPI1 interface is a full duplex synchronous serial data link. It is a four line interface with pin names SDI1, SDO1, SCK1, SCS1B0 and SCS1B1. Pins SDI1 and SDO1 are the Serial Data Input and Serial Data Output lines, SCK1 is the Serial Clock line and SCS1B0 and SCS1B1 are the Slave Select line. As the SPI1 interface pins are pin-shared with normal I/O pins, the SPI1 interface must first be enabled by setting the correct bits in the SPI1C0 and SPI1C1 registers. Communication between devices connected to the SPI1 interface is carried out in a slave/master mode with all data transfer initiations being implemented by the master. The Master also controls the clock signal.

The SCS1B0 and SCS1B1 pin are controlled by the application program, set the S1CSEN bit to "1" to enable the SCS1B0 and SCS1B1 pin function and clear the S1CSEN bit to "0" to place the SCS1B0 and SCS1B1 pin into a floating state.

For smaller packages that some of SPI1 functional pins are not externally bounded, their functions can be simulated using I/O pins if necessary.



SPI1 Master/Slave Bus Connection

The SPI1 function in this device offers the following features:

- Full duplex synchronous data transfer
- Both Master and Slave modes
- LSB first or MSB first data transmission modes
- Transmission complete flag
- Rising or falling active clock edge
- SAWCOL and S1CSEN bit enabled or disable select

The status of the SPI1 interface pins is determined by a number of factors such as whether the device is in the master or slave mode and upon the condition of certain control bits such as S1CSEN and SPI1EN.

SPI1 Registers

There are three internal registers which control the overall operation of the SPI1 interface. These are the SPI1D data register and two registers SPI1C0 and SPI1C1.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SPI1C0	S1MS2	S1MS1	S1MS0	—	SPICF	—	SPI1EN	S1CS
SPI1C1	—	—	S1CKPOL	S1CKEG	S1MLS	S1CSEN	S1WCOL	S1TRF
SPI1D	D7	D6	D5	D4	D3	D2	D1	D0

SPI1 Register List

• SPI1C0 Register

Bit	7	6	5	4	3	2	1	0
Name	S1MS2	S1MS1	S1MS0	—	SPICF	—	SPI1EN	S1CS
R/W	R/W	R/W	R/W	—	R/W	—	R/W	R/W
POR	1	1	1	—	0	—	0	0

Bit 7~5 **S1MS2~S1MS0**: Master/Slave Clock Select
000: SPI1 master, $f_{SYS}/4$
001: SPI1 master, $f_{SYS}/16$
010: SPI1 master, $f_{SYS}/64$
011: SPI1 master, f_{SUB}
100: SPI1 master, TP0 CCRP match frequency/2 (PFD)
101: SPI1 slave

Bit 4 Unimplemented, read as "0"

Bit 3 **SPICF**: SPI1 interface transfer incomplete flag
0: No SPI1 incomplete transfer occurs
1: SPI1 incomplete transfer occurred
The SPICF bit is only used for the SPI1 slave device to indicate whether the SPI1 transfer is complete or not. When the SPI1 transfer is in progress and the SCS1B0 or SCS1B1 pin is set to "1", the SPI1 transfer will be stopped and the SPICF flag will be set to 1 by hareware.

Bit 2 Unimplemented, read as "0"

Bit 1 **SPI1EN**: SPI1 enable or disable
0: Disable
1: Enable

Bit 0 **S1CS**: SPI1 chip select pin
0: SCS1B0
1: SCS1B1

• SPI1C1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	S1CKPOL	S1CKEG	S1MLS	S1CSEN	S1WCOL	S1TRF
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"
This bit can be read or written by user software program.

Bit 5 **S1CKPOL**: Determines the base condition of the clock line
0: SCK1 line will be high when the clock is inactive
1: SCK1 line will be low when the clock is inactive
The S1CKPOL bit determines the base condition of the clock line, if the bit is high, then the SCK1 line will be low when the clock is inactive. When the S1CKPOL bit is low, then the SCK1 line will be high when the clock is inactive.

Bit 4 **S1CKEG**: Determines the SPI1 SCK1 active clock edge type
S1CKPOL = 0:
0: SCK1 has high base level with data capture on SCK1 rising edge
1: SCK1 has high base level with data capture on SCK1 falling edge
S1CKPOL = 1:
0: SCK1 has low base level with data capture on SCK1 falling edge
1: SCK1 has low base level with data capture on SCK1 rising edge
The S1CKEG and S1CKPOL bits are used to setup the way that the clock signal outputs and inputs data on the SPI bus. These two bits must be configured before a data transfer is executed otherwise an erroneous clock edge may be generated. The S1CKPOL bit determines the base condition of the clock line, if the bit is high, then the SCK1 line will be low when the clock is inactive. When the S1CKPOL bit is low, then the SCK1 line will be high when the clock is inactive. The S1CKEG bit determines active clock edge type which depends upon the condition of the S1CKPOL bit.

Bit 3 **S1MLS**: MSB/LSB First Bit
0: LSB shift first
1: MSB shift first
This is the data shift select bit and is used to select how the data is transferred, either MSB or LSB first. Setting the bit high will select MSB first and low for LSB first.

Bit 2 **S1CSEN**: Select Signal Enable/Disable Bit
0: SCS1B0 and SCS1B1 floating
1: Enable
The S1CSEN bit is used as an enable/disable for the SCS1B0 and SCS1B1 pin. If this bit is low, then the SCS1B0 and SCS1B1 pin will be disabled and placed into a floating condition. If the bit is high the SCS1B0 and SCS1B1 pin will be enabled and used as a select pin.

Bit 1 **S1WCOL**: Write Collision Bit
0: Collision free
1: Collision detected
The S1WCOL flag is used to detect if a data collision has occurred. If this bit is high it means that data has been attempted to be written to the SPI1D register during a data transfer operation. This writing operation will be ignored if data is being transferred. The bit can be cleared by the application program.

Bit 0 **S1TRF**: Transmit/Receive Flag
0: Not complete
1: Transmission/reception complete
The S1TRF bit is the Transmit/Receive Complete flag and is set "1" automatically when an SPI1 data transmission is completed, but must set to zero by the application program. It can be used to generate an interrupt.

- **SPI1D 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

SPI1 Communication

After the SPI1 interface is enabled by setting the SPI1EN bit high, then in the Master Mode, when data is written to the SPI1D register, transmission/reception will begin simultaneously. When the data transfer is complete, the S1TRF flag will be set automatically, but must be cleared using the application program. In the Slave Mode, when the clock signal from the master has been received, any data in the SPI1D register will be transmitted and any data on the SDI1 pin will be shifted into the SPI1D register.

The master should output a SCS1Bn signal to enable the slave device before a clock signal is provided. The slave data to be transferred should be well prepared at the appropriate moment relative to the SCS1Bn signal depending upon the configurations of the S1CKPOL bit and S1CKEG bit. The accompanying timing diagram shows the relationship between the slave data and $\overline{SCS}$ signal for various configurations of the S1CKPOL and S1CKEG bits.

The SPI Master mode will continue to function if the SPI clock is running.

Peripheral Clock Output

The Peripheral Clock Output allows the device to supply external hardware with a clock signal synchronised to the microcontroller clock.

Peripheral Clock Operation

As the peripheral clock output pin, PCK, is shared with I/O line, the required pin function is chosen via PCKEN in the SIMC0 register. The Peripheral Clock function is controlled using the SIMC0 register. The clock source for the Peripheral Clock Output can originate from either the TM0 CCRP match frequency/2 or a divided ratio of the internal f_{SYS} clock. The PCKEN bit in the SIMC0 register is the overall on/off control, setting PCKEN bit to "1" enables the Peripheral Clock, and setting PCKEN bit to "0" disables it. The required division ratio of the system clock is selected using the PCKP1 and PCKP0 bits in the same register. If the device enters the SLEEP Mode, this will disable the Peripheral Clock output.

- **SIMC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	PCKEN	PCKP1	PCKP0	SIMEN	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	—
POR	1	1	1	0	0	0	0	—

Bit 7~5 **SIM2~SIM0**: SIM Operating Mode Control
000: SPI master mode; SPI clock is $f_{SYS}/4$
001: SPI master mode; SPI clock is $f_{SYS}/16$
010: SPI master mode; SPI clock is $f_{SYS}/64$
011: SPI master mode; SPI clock is f_{SUB}
100: SPI master mode; SPI clock is TM0 CCRP match frequency/2
101: SPI slave mode
110: I^2C slave mode
111: Unused mode
These bits setup the overall operating mode of the SIM function. As well as selecting if the I^2C or SPI function, they are used to control the SPI Master/Slave selection and the SPI Master clock frequency. The SPI clock is a function of the system clock but can also be chosen to be sourced from the TM0. If the SPI Slave Mode is selected then the clock will be supplied by an external Master device.

Bit 4 **PCKEN**: PCK Output Pin Control
0: Disable
1: Enable

Bit 3~2 **PCKP1~PCKP0**: Select PCK output pin frequency
00: f_{SYS}
01: $f_{SYS}/4$
10: $f_{SYS}/8$
11: TM0 CCRP match frequency/2

Bit 1 **SIMEN**: SIM Control
0: Disable
1: Enable
The bit is the overall on/off control for the SIM interface. When the SIMEN bit is cleared to zero to disable the SIM interface, the SDI, SDO, SCK and $\overline{SCS}$, or SDA and SCL lines will be in a floating condition and the SIM operating current will be reduced to a minimum value. When the bit is high the SIM interface is enabled. If the SIM is configured to operate as an SPI interface via the SIM2~SIM0 bits, the contents of the SPI control registers will remain at the previous settings when the SIMEN bit changes from low to high and should therefore be first initialised by the application program. If the SIM is configured to operate as an I^2C interface via the SIM2~SIM0 bits and the SIMEN bit changes from low to high, the contents of the I^2C control bits such as HTX and TXAK will remain at the previous settings and should therefore be first initialised by the application program while the relevant I^2C flags such as HCF, HAAS, HBB, SRW and RXAK will be set to their default states.

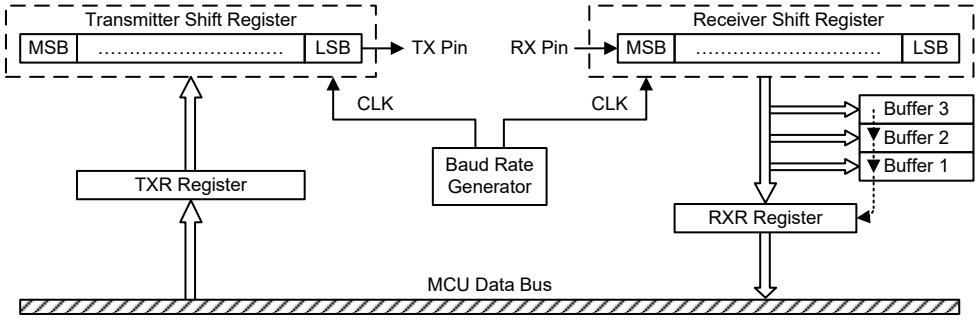
Bit 0 Unimplemented, read as "0"

UART Interface

The devices contain an integrated full-duplex asynchronous serial communications UART interface that enables communication with external devices that contain a serial interface. The UART function has many features and can transmit and receive data serially by transferring a frame of data with eight or nine data bits per transmission as well as being able to detect errors when the data is overwritten or incorrectly framed. The UART function possesses its own internal interrupt which can be used to indicate when a reception occurs or when a transmission terminates.

The integrated UART function contains the following features:

- Full-duplex, asynchronous communication
- 8 or 9 bits character length
- Even, odd or no parity options
- One or two stop bits
- Baud rate generator with 8-bit prescaler
- Parity, framing, noise and overrun error detection
- Support for interrupt on address detect (last character bit=1)
- Separately enabled transmitter and receiver
- 4-byte Deep FIFO Receive Data Buffer
- Transmit and receive interrupts
- Interrupts can be initialized by the following conditions:
  - Transmitter Empty
  - Transmitter Idle
  - Receiver Full
  - Receiver Overrun
  - Address Mode Detect



UART Data Transfer Scheme

UART External Interface

To communicate with an external serial interface, the internal UART has two external pins known as TX and RX. The TX pin is the UART transmitter pin, which can be used as a general purpose I/O or other pin-shared functional pin if the pin is not configured as a UART transmitter, which occurs when the TXEN bit in the UCR2 control register is equal to zero. Similarly, the RX pin is the UART receiver pin, which can also be used as a general purpose I/O pin, if the pin is not configured as a receiver, which occurs if the RXEN bit in the UCR2 register is equal to zero. Along with the UARTEN bit, the TXEN and RXEN bits, if set, will automatically setup these I/O pins or other pin-shared functional to their respective TX output and RX input conditions and disable any pull-high resistor option which may exist on the TX pin. However, the pull-high resistor related to the RX pin is controlled by the corresponding I/O pull-high function control bit.

UART Data Transfer Scheme

The block diagram shows the overall data transfer structure arrangement for the UART interface. The actual data to be transmitted from the MCU is first transferred to the TXR register by the application program. The data will then be transferred to the Transmit Shift Register from where it will be shifted out, LSB first, onto the TX pin at a rate controlled by the Baud Rate Generator. Only the TXR register is mapped onto the MCU Data Memory, the Transmit Shift Register is not mapped and is therefore inaccessible to the application program.

Data to be received by the UART is accepted on the external RX pin, from where it is shifted in, LSB first, to the Receiver Shift Register at a rate controlled by the Baud Rate Generator. When the shift register is full, the data will then be transferred from the shift register to the internal RXR register, where it is buffered and can be manipulated by the application program. Only the RXR register is mapped onto the MCU Data Memory, the Receiver Shift Register is not mapped and is therefore inaccessible to the application program.

It should be noted that the actual register for data transmission and reception, although referred to in the text, and in application programs, as separate TXR and RXR registers, only exists as a single shared register in the Data Memory. This shared register known as the TXR/RXR register is used for both data transmission and data reception.

UART Status and Control Registers

There are five control registers associated with the UART function. The USR, UCR1 and UCR2 registers control the overall function of the UART, while the BRG register controls the Baud rate. The actual data to be transmitted and received on the serial interface is managed through the TXR/RXR data registers.

• **TXR/RXR Register**

The TXR/RXR register is the data register which is used to store the data to be transmitted on the TX pin or being received from the RX pin.

Bit	7	6	5	4	3	2	1	0
Name	TXRX7	TXRX6	TXRX5	TXRX4	TXRX3	TXRX2	TXRX1	TXRX0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x": unknown

Bit 7~0 **TXRX7~TXRX0**: UART Transmit/Receive Data bits

• **USR Register**

The USR register is the status register for the UART, which can be read by the program to determine the present status of the UART. All flags within the USR register are read only and further explanations are given below:

Bit	7	6	5	4	3	2	1	0
Name	PERR	NF	FERR	OERR	RIDLE	RXIF	TIDLE	TXIF
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	1	0	1	1

Bit 7 **PERR**: Parity error flag
0: No parity error is detected
1: Parity error is detected

The PERR flag is the parity error flag. When this read only flag is "0", it indicates a parity error has not been detected. When the flag is "1", it indicates that the parity of the received word is incorrect. This error flag is applicable only if Parity mode (odd or even) is selected. The flag can also be cleared by a software sequence which involves a read to the status register USR followed by an access to the RXR data register.

Bit 6 **NF**: Noise flag
0: No noise is detected
1: Noise is detected

The NF flag is the noise flag. When this read only flag is "0", it indicates no noise condition. When the flag is "1", it indicates that the UART has detected noise on the receiver input. The NF flag is set during the same cycle as the RXIF flag but will not be set in the case of as overrun. The NF flag can be cleared by a software sequence which will involve a read to the status register USR followed by an access to the RXR data register.

Bit 5 **FERR**: Framing error flag
0: No framing error is detected
1: Framing error is detected

The FERR flag is the framing error flag. When this read only flag is "0", it indicates that there is no framing error. When the flag is "1", it indicates that a framing error has been detected for the current character. The flag can also be cleared by a software sequence which will involve a read to the status register USR followed by an access to the RXR data register.

Bit 4 **OERR**: Overrun error flag
0: No overrun error is detected
1: Overrun error is detected

The OERR flag is the overrun error flag which indicates when the receiver buffer has overflowed. When this read only flag is "0", it indicates that there is no overrun error. When the flag is "1", it indicates that an overrun error occurs which will inhibit further transfers to the RXR receive data register. The flag is cleared by a software sequence, which is a read to the status register USR followed by an access to the RXR data register.

Bit 3 **RIDLE**: Receiver status

0: data reception is in progress (data being received)
1: no data reception is in progress (receiver is idle)

The RIDLE flag is the receiver status flag. When this read only flag is "0", it indicates that the receiver is between the initial detection of the start bit and the completion of the stop bit. When the flag is "1", it indicates that the receiver is idle. Between the completion of the stop bit and the detection of the next start bit, the RIDLE bit is "1" indicating that the UART receiver is idle and the RX pin stays in logic high condition.

Bit 2 **RXIF**: Receive RXR data register status

0: RXR data register is empty
1: RXR data register has available data

The RXIF flag is the receive data register status flag. When this read only flag is "0", it indicates that the RXR read data register is empty. When the flag is "1", it indicates that the RXR read data register contains new data. When the contents of the shift register are transferred to the RXR register, an interrupt is generated if RIE=1 in the UCR2 register. If one or more errors are detected in the received word, the appropriate receive-related flags NF, FERR, and/or PERR are set within the same clock cycle. The RXIF flag is cleared when the USR register is read with RXIF set, followed by a read from the RXR register, and if the RXR register has no data available.

Bit 1 **TIDLE**: Transmission status

0: data transmission is in progress (data being transmitted)
1: no data transmission is in progress (transmitter is idle)

The TIDLE flag is known as the transmission complete flag. When this read only flag is "0", it indicates that a transmission is in progress. This flag will be set to "1" when the TXIF flag is "1" and when there is no transmit data or break character being transmitted. When TIDLE is equal to 1, the TX pin becomes idle with the pin state in logic high condition. The TIDLE flag is cleared by reading the USR register with TIDLE set and then writing to the TXR register. The flag is not generated when a data character or a break is queued and ready to be sent.

Bit 0 **TXIF**: Transmit TXR data register status

0: character is not transferred to the transmit shift register
1: character has transferred to the transmit shift register (TXR data register is empty)

The TXIF flag is the transmit data register empty flag. When this read only flag is "0", it indicates that the character is not transferred to the transmitter shift register. When the flag is "1", it indicates that the transmitter shift register has received a character from the TXR data register. The TXIF flag is cleared by reading the UART status register (USR) with TXIF set and then writing to the TXR data register. Note that when the TXEN bit is set, the TXIF flag bit will also be set since the transmit data register is not yet full.

• UCR1 Register

The UCR1 register together with the UCR2 register are the UART control registers that are used to set the various options for the UART function such as overall on/off control, parity control, data transfer bit length, etc. Further explanation on each of the bits is given below:

Bit	7	6	5	4	3	2	1	0
Name	UARTEN	BNO	PREN	PRT	STOPS	TXBRK	RX8	TX8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	W
POR	0	0	0	0	0	0	x	0

"x": unknown

Bit 7 **UARTEN**: UART function enable control

0: disable UART. TX and RX pins are I/O or other pin-shared functions
1: enable UART. TX and RX pins function as UART pins

The UARTEN bit is the UART enable bit. When this bit is equal to "0", the UART will be disabled and the RX pin as well as the TX pin will be set as I/O or other pin-shared functions. When the bit is equal to "1", the UART will be enabled and the TX and RX pins will function as defined by the TXEN and RXEN enable control bits. When the UART is disabled, it will empty the buffer so any character remaining in the buffer will be discarded. In addition, the value of the baud rate counter will be reset. If the UART is disabled, all error and status flags will be reset. Also the TXEN, RXEN, TXBRK, RXIF, OERR, FERR, PERR and NF bits will be cleared, while the TIDLE, TXIF and RIDLE bits will be set. Other control bits in UCR1, UCR2 and BRG registers will remain unaffected. If the UART is active and the UARTEN bit is cleared, all pending transmissions and receptions will be terminated and the module will be reset as defined above. When the UART is re-enabled, it will restart in the same configuration.

Bit 6 **BNO**: Number of data transfer bits selection

0: 8-bit data transfer

1: 9-bit data transfer

This bit is used to select the data length format, which can have a choice of either 8-bit or 9-bit format. When this bit is equal to "1", a 9-bit data length format will be selected. If the bit is equal to "0", then an 8-bit data length format will be selected. If 9-bit data length format is selected, then bits RX8 and TX8 will be used to store the 9th bit of the received and transmitted data respectively.

Bit 5 **PREN**: Parity function enable control

0: Parity function is disabled

1: Parity function is enabled

This bit is the parity function enable bit. When this bit is equal to 1, the parity function will be enabled. If the bit is equal to 0, then the parity function will be disabled.

Bit 4 **PRT**: Parity type selection bit

0: Even parity for parity generator

1: Odd parity for parity generator

This bit is the parity type selection bit. When this bit is equal to 1, odd parity type will be selected. If the bit is equal to 0, then even parity type will be selected.

Bit 3 **STOPS**: Number of stop bits selection

0: One stop bit format is used

1: Two stop bits format is used

This bit determines if one or two stop bits are to be used. When this bit is equal to "1", two stop bits format are used. If the bit is equal to "0", then only one stop bit format is used.

Bit 2 **TXBRK**: Transmit break character

0: No break character is transmitted

1: Break characters transmit

The TXBRK bit is the Transmit Break Character bit. When this bit is equal to "0", there are no break characters and the TX pin operats normally. When the bit is equal to "1", there are transmit break characters and the transmitter will send logic zeros. When this bit is equal to "1", after the buffered data has been transmitted, the transmitter output is held low for a minimum of a 13-bit length and until the TXBRK bit is reset.

Bit 1 **RX8**: Receive data bit 8 for 9-bit data transfer format (read only)

This bit is only used if 9-bit data transfers are used, in which case this bit location will store the 9th bit of the received data known as RX8. The BNO bit is used to determine whether data transfes are in 8-bit or 9-bit format.

Bit 0 **TX8**: Transmit data bit 8 for 9-bit data transfer format (write only)

This bit is only used if 9-bit data transfers are used, in which case this bit location will store the 9th bit of the transmitted data known as TX8. The BNO bit is used to determine whether data transfes are in 8-bit or 9-bit format.

• **UCR2 Register**

The UCR2 register is the second of the UART control registers and serves several purposes. One of its main functions is to control the basic enable/disable operation if the UART Transmitter and Receiver as well as enabling the various UART interrupt sources. The register also serves to control the baud rate speed, receiver wake-up function enable and the address detect function enable. Further explanation on each of the bits is given below:

Bit	7	6	5	4	3	2	1	0
Name	TXEN	RXEN	BRGH	ADDEN	WAKE	RIE	TIIE	TEIE
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 **TXEN**: UART Transmitter enable control
0: UART Transmitter is disabled
1: UART Transmitter is enabled

The bit named TXEN is the Transmitter Enable Bit. When this bit is equal to "0", the transmitter will be disabled with any pending data transmissions being aborted. In addition the buffers will be reset. In this situation the TX pin will be used as an I/O or other pin-shared functional pin. If the TXEN bit is equal to "1" and the UARTEN bit is also equal to 1, the transmitter will be enabled and the TX pin will be controlled by the UART. Clearing the TXEN bit during a transmission will cause the data transmission to be aborted and will reset the transmitter. If this situation occurs, the TX pin will be used as an I/O or other pin-shared functional pin.

Bit 6 **RXEN**: UART Receiver enable control
0: UART Receiver is disabled
1: UART Receiver is enabled

The bit named RXEN is the Receiver Enable Bit. When this bit is equal to "0", the receiver will be disabled with any pending data receptions being aborted. In addition the receiver buffers will be reset. In this situation the RX pin will be used as an I/O or other pin-shared functional pin. If the RXEN bit is equal to "1" and the UARTEN bit is also equal to 1, the receiver will be enabled and the RX pin will be controlled by the UART. Clearing the RXEN bit during a reception will cause the data reception to be aborted and will reset the receiver. If this situation occurs, the RX pin will be used as an I/O or other pin-shared functional pin.

Bit 5 **BRGH**: Baud Rate speed selection
0: Low speed baud rate
1: High speed baud rate

The bit named BRGH selects the high or low speed mode of the Baud Rate Generator. This bit, together with the value placed in the baud rate register, BRG, controls the baud rate of the UART. If the bit is equal to 0, the low speed mode is selected.

Bit 4 **ADDEN**: Address detect function enable control
0: Address detection function is disabled
1: Address detection function is enabled

The bit named ADDEN is the address detection function enable control bit. When this bit is equal to 1, the address detection function is enabled. When it occurs, if the 8th bit, which corresponds to RX7 if BNO=0, or the 9th bit, which corresponds to RX8 if BNO=1, has a value of "1", then the received word will be identified as an address, rather than data. If the corresponding interrupt is enabled, an interrupt request will be generated each time the received word has the address bit set, which is the 8th or 9th bit depending on the value of the BNO bit. If the address bit known as the 8th or 9th bit of the received word is "0" with the address detection function being enabled, an interrupt will not be generated and the received data will be discarded.

Bit 3 **WAKE**: RX pin falling edge wake-up function enable control
0: RX pin wake-up function is disabled
1: RX pin wake-up function is enabled
The bit enables or disables the receiver wake-up function. If this bit is equal to 1 and the device is in IDLE or SLEEP mode, a falling edge on the RX pin will wake up the device. If this bit is equal to 0 and the device is in IDLE or SLEEP mode, any edge transitions on the RX pin will not wake up the device.

Bit 2 **RIE**: Receiver interrupt enable control
0: Receiver related interrupt is disabled
1: Receiver related interrupt is enabled
The bit enables or disables the receiver interrupt. If this bit is equal to 1 and when the receiver overrun flag OERR or received data available flag RXIF is set, the UART interrupt request flag will be set. If this bit is equal to 0, the UART interrupt request flag will not be influenced by the condition of the OERR or RXIF flags.

Bit 1 **TIIE**: Transmitter Idle interrupt enable control
0: Transmitter idle interrupt is disabled
1: Transmitter idle interrupt is enabled
The bit enables or disables the transmitter idle interrupt. If this bit is equal to 1 and when the transmitter idle flag TIDLE is set, due to a transmitter idle condition, the UART interrupt request flag will be set. If this bit is equal to 0, the UART interrupt request flag will not be influenced by the condition of the TIDLE flag.

Bit 0 **TEIE**: Transmitter Empty interrupt enable control
0: Transmitter empty interrupt is disabled
1: Transmitter empty interrupt is enabled
The bit enables or disables the transmitter empty interrupt. If this bit is equal to 1 and when the transmitter empty flag TXIF is set, due to a transmitter empty condition, the UART interrupt request flag will be set. If this bit is equal to 0, the UART interrupt request flag will not be influenced by the condition of the TXIF flag.

Baud Rate Generator

To setup the speed of the serial data communication, the UART function contains its own dedicated baud rate generator. The baud rate is controlled by its own internal free running 8-bit timer, the period of which is determined by two factors. The first of these is the value placed in the BRG register and the second is the value of the BRGH bit within the UCR2 control register. The BRGH bit decides, if the baud rate generator is to be used in a high speed mode or low speed mode, which in turn determines the formula that is used to calculate the baud rate. The value in the BRG register, N, which is used in the following baud rate calculation formula determines the division factor. Note that N is the decimal value placed in the BRG register and has a range of between 0 and 255.

UCR2 BRGH Bit	0	1
Baud Rate (BR)	$\frac{f_{SYS}}{[64(N+1)]}$	$\frac{f_{SYS}}{[16(N+1)]}$

By programming the BRGH bit which allows selection of the related formula and programming the required value in the BRG register, the required baud rate can be setup. Note that because the actual baud rate is determined using a discrete value, N, placed in the BRG register, there will be an error associated between the actual and requested value. The following example shows how the BRG register value N and the error value can be calculated.

• **BRG Register**

Bit	7	6	5	4	3	2	1	0
Name	BRG7	BRG6	BRG5	BRG4	BRG3	BRG2	BRG1	BRG0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x": unknown

Bit 7~0 **BRG7~BRG0**: Baud Rate values

By programming the BRGH bit in the UCR2 register which allows selection of the related formula described above and programming the required value in the BRG register, the required baud rate can be setup.

Calculating the Baud Rate and Error Values

For a clock frequency of 4MHz, and with BRGH set to 0 determine the BRG register value N, the actual baud rate and the error value for a desired baud rate of 4800.

From the above table the desired baud rate $BR = \frac{f_{SYS}}{[64(N+1)]}$

Re-arranging this equation gives $N = \frac{f_{SYS}}{(BR \times 64)} - 1$

Giving a value for $N = \frac{4000000}{(4800 \times 64)} - 1 = 12.0208$

To obtain the closest value, a decimal value of 12 should be placed into the BRG register. This gives

an actual or calculated baud rate value of $BR = \frac{4000000}{[64(12+1)]} = 4808$

Therefore the error is equal to $\frac{4808 - 4800}{4800} = 0.16\%$

The following tables show the actual values of baud rate and error values for the two value of BRGH.

Baud Rate K/BPS	Baud Rates for BRGH=0								
	f_{SYS}=4MHz			f_{SYS}=3.579545MHz			f_{SYS}=7.159MHz		
	BRG	Kbaud	Error(%)	BRG	Kbaud	Error(%)	BRG	Kbaud	Error(%)
0.3	207	0.300	0.16	185	0.300	0.00	—	—	—
1.2	51	1.202	0.16	46	1.190	-0.83	92	1.203	0.23
2.4	25	2.404	0.16	22	2.432	1.32	46	2.380	-0.83
4.8	12	4.808	0.16	11	4.661	-2.90	22	4.863	1.32
9.6	6	8.929	-6.99	5	9.321	-2.90	11	9.322	-2.90
19.2	2	20.833	8.51	2	18.643	-2.90	5	18.643	-2.90
38.4	—	—	—	—	—	—	2	32.286	-2.90
57.6	0	62.500	8.51	0	55.930	-2.90	1	55.930	-2.90
115.2	—	—	—	—	—	—	0	111.859	-2.90

Baud Rates and Error Values for BRGH=0

Baud Rate K/BPS	Baud Rates for BRGH=1								
	f_{SYS}=4MHz			f_{SYS}=3.579545MHz			f_{SYS}=7.159MHz		
	BRG	Kbaud	Error(%)	BRG	Kbaud	Error(%)	BRG	Kbaud	Error(%)
0.3	—	—	—	—	—	—	—	—	—
1.2	207	1.202	0.16	185	1.203	0.23	—	—	—
2.4	103	2.404	0.16	92	2.406	0.23	185	2.406	0.23
4.8	51	4.808	0.16	46	4.76	-0.83	92	4.811	0.23
9.6	25	9.615	0.16	22	9.727	1.32	46	9.520	-0.83
19.2	12	19.231	0.16	11	18.643	-2.90	22	19.454	1.32
38.4	6	35.714	-6.99	5	37.286	-2.90	11	37.286	-2.90
57.6	3	62.5	8.51	3	55.930	-2.90	7	55.930	-2.90
115.2	1	125	8.51	1	111.86	-2.90	3	111.86	-2.90
250	0	250	0	—	—	—	—	—	—

Baud Rates and Error Values for BRGH=1

UART Setup and Control

For data transfer, the UART function utilizes a non-return-to-zero, more commonly known as NRZ, format. This is composed of one start bit, eight or nine data bits and one or two stop bits. Parity is supported by the UART hardware and can be setup to be even, odd or no parity. For the most common data format, 8 data bits along with no parity and one stop bit, denoted as 8, N, 1, is used as the default setting, which is the setting at power-on. The number of data bits and stop bits, along with the parity, are setup by programming the corresponding BNO, PRT, PREN and STOPS bits in the UCR1 register. The baud rate used to transmit and receive data is setup using the internal 8-bit baud rate generator, while the data is transmitted and received LSB first. Although the transmitter and receiver of the UART are functionally independent, they both use the same data format and baud rate. In all cases stop bits will be used for data transmission.

Enabling/Disabling the UART Interface

The basic on/off function of the internal UART function is controlled using the UARTEN bit in the UCR1 register. If the UARTEN, TXEN and RXEN bits are set, then these two UART pins will act as normal TX output pin and RX input pin respectively. If no data is being transmitted on the TX pin, then it will default to a logic high value.

Clearing the UARTEN bit will disable the TX and RX pins and these two pins will be used as an I/O or other pin-shared functional pin. When the UART function is disabled, the buffer will be reset to an empty condition, at the same time discarding any remaining residual data. Disabling the UART will also reset the enable control, the error and status flags with bits TXEN, RXEN, TXBRK, RXIF, OERR, FERR, PERR and NF being cleared while bits TIDLE, TXIF and RIDLE will be set. The remaining control bits in the UCR1, UCR2 and BRG registers will remain unaffected. If the UARTEN bit in the UCR1 register is cleared while the UART is active, then all pending transmissions and receptions will be immediately suspended and the UART will be reset to a condition as defined above. If the UART is then subsequently re-enabled, it will restart again in the same configuration.

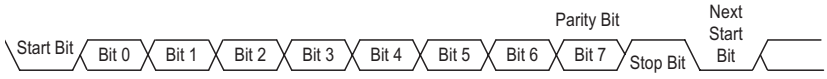
Data, Parity and Stop Bit Selection

The format of the data to be transferred is composed of various factors such as data bit length, parity on/off, parity type, address bits and the number of stop bits. These factors are determined by the setup of various bits within the UCR1 register. The BNO bit controls the number of data bits which can be set to either 8 or 9. The PRT bit controls the choice if odd or even parity. The PREN bit controls the parity on/off function. The STOPS bit decides whether one or two stop bits are to be used. The following table shows various formats for data transmission.The address bit, which is the MSB of the data byte, identifies the frame as an address character or data if the address detect function is enabled. The number of stop bits, which can be either one or two, is independent of the data length.

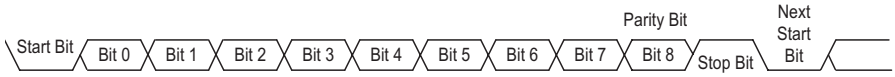
Start Bit	Data Bits	Address Bits	Parity Bits	Stop Bit
Example of 8-bit Data Formats				
1	8	0	0	1
1	7	0	1	1
1	7	1	1	1
Example of 8-bit Data Formats				
1	9	0	0	1
1	8	0	1	1
1	8	1	0	1

Trams,otter Receiver Data Format

The following diagram shows the transmit and receive waveforms for both 8-bit and 9-bit data formats.



8-Bit Data Format



9-Bit Data Format

UART Transmitter

Data word lengths of either 8 or 9 bits can be selected by programming the BNO bit in the UCR1 register. When BNO bit is set, the word length will be set to 9 bits. In this case the 9th bit, which is the MSB, needs to be stored in the TX8 bit in the UCR1 register. At the transmitter core lies the Transmitter Shift Register, more commonly known as the TSR, whose data is obtained from the transmit data register, which is known as the TXR register. The data to be transmitted is loaded into this TXR register by the application program. The TSR register is not written to with new data until the stop bit from the previous transmission has been sent out. As soon as this stop bit has been transmitted, the TSR can then be loaded with new data from the TXR register, if it is available. It should be noted that the TSR register, unlike many other registers, is not directly mapped into the Data Memory area and as such is not available to the application program for direct read/write operations. An actual transmission of data will normally be enabled when the TXEN bit is set, but the data will not be transmitted until the TXR register has been loaded with data and the baud rate generator has defined a shift clock source. However, the transmission can also be initiated by first loading data into the TXR register, after which the TXEN bit can be set. When a transmission of data begins, the TSR is normally empty, in which case a transfer to the TXR register will result in an immediate transfer to the TSR. If during a transmission the TXEN bit is cleared, the transmission will immediately cease and the transmitter will be reset. The TX output pin will then return to the I/O or other pin-shared function.

Transmitting Data

When the UART is transmitting data, the data is shifted on the TX pin from the shift register, with the least significant bit LSB first. In the transmit mode, the TXR register forms a buffer between the internal bus and the transmitter shift register. It should be noted that if 9-bit data format has been selected, then the MSB will be taken from the TX8 bit in the UCR1 register. The steps to initiate a data transfer can be summarized as follows:

- Make the correct selection of the BNO, PRT, PREN and STOPS bits to define the required word length, parity type and number of stop bits.
- Setup the BRG register to select the desired baud rate.
- Set the TXEN bit to ensure that the UART transmitter is enabled and the TX pin is used as a UART transmitter pin.
- Access the USR register and write the data that is to be transmitted into the TXR register. Note that this step will clear the TXIF bit.

This sequence of events can now be repeated to send additional data. It should be noted that when TXIF=0, data will be inhibited from being written to the TXR register. Clearing the TXIF flag is always achieved using the following software sequence:

1. A USR register access
2. A TXR register write execution

The read-only TXIF flag is set by the UART hardware and if set indicates that the TXR register is empty and that other data can now be written into the TXR register without overwriting the previous data. If the TEIE bit is set, then the TXIF flag will generate an interrupt. During a data transmission, a write instruction to the TXR register will place the data into the TXR register, which will be copied to the shift register at the end of the present transmission. When there is no data transmission in progress, a write instruction to the TXR register will place the data directly into the shift register, resulting in the commencement of data transmission, and the TXIF bit being immediately set. When a frame transmission is complete, which happens after stop bits are sent or after the break frame, the TIDLE bit will be set. To clear the TIDLE bit the following software sequence is used:

1. A USR register access
2. A TXR register write execution

Note that both the TXIF and TIDLE bits are cleared by the same software sequence.

Transmitting Break

If the TXBRK bit is set, then the break characters will be sent on the next transmission. Break character transmission consists of a start bit, followed by 13xN “0” bits, where N=1, 2, etc. if a break character is to be transmitted, then the TXBRK bit must be first set by the application program and then cleared to generate the stop bits. Transmitting a break character will not generate a transmit interrupt. Note that a break condition length is at least 13 bits long. If the TXBRK bit is continually kept at a logic high level, then the transmitter circuitry will transmit continuous break characters. After the application program has cleared the TXBRK bit, the transmitter will finish transmitting the last break character and subsequently send out one or two stop bits. The automatic logic high at the end of the last break character will ensure that the start bit of the next frame is recognized.

UART Receiver

The UART is capable of receiving word lengths of either 8 or 9 bits can be selected by programming the BNO bit in the UCR1 register. When BNO bit is set, the word length will be set to 9 bits. In this case the 9th bit, which is the MSB, will be stored in the RX8 bit in the UCR1 register. At the receiver core lies the Receiver Shift Register more commonly known as the RSR. The data which is received on the RX external input pin is sent to the data recovery block. The data recovery block operating speed is 16 times that of the baud rate, while the main receive serial shifter operates at the baud rate. After the RX pin is sampled for the stop bit, the received data in RSR is transferred to the receive data register, if the register is empty. The data which is received on the external RX input pin is sampled three times by a majority detect circuit to determine the logic level that has been placed onto the RX pin. It should be noted that the RSR register, unlike many other registers, is not directly mapped into the Data Memory area and as such is not available to the application program for direct read/write operations.

Receiving Data

When the UART receiver is receiving data, the data is serially shifted in on the external RX input pin to the shift register, with the least significant bit LSB first. The RXR register is a four byte deep FIFO data buffer, where four bytes can be held in the FIFO while the 5th byte can continue to be received. Note that the application program must ensure that the data is read from RXR before the 5th byte has been completely shifted in, otherwise the 5th byte will be discarded and an overrun error OERR will be subsequently indicated. The steps to initiate a data transfer can be summarized as follows:

- Make the correct selection of the BNO, PRT, PREN and STOPS bits to define the required word length, parity type and number of stop bits.
- Setup the BRG register to select the desired baud rate.
- Set the RXEN bit to ensure that the UART receiver is enabled and the RX pin is used as a UART receiver pin.

At this point the receiver will be enabled which will begin to look for a start bit.

When a character is received, the following sequence of events will occur:

- The RXIF bit in the USR register will be set when the RXR register has data available. There will be at most three more characters available before an overrun error occurs.
- When the contents of the shift register have been transferred to the RXR register and if the RIE bit is set, then an interrupt will be generated.
- If during reception, a frame error, noise error, parity error or an overrun error has been detected, then the error flags can be set.

The RXIF bit can be cleared using the following software sequence:

1. A USR register access
2. A RXR register read execution

Receiving Break

Any break character received by the UART will be managed as a framing error. The receiver will count and expect a certain number of bit times as specified by the values programmed into the BNO and STOPS bits. If the break is much longer than 13 bit times, the reception will be considered as complete after the number of bit times specified by BNO and STOPS. The RXIF bit is set, FERR is set, zeros are loaded into the receive data register, interrupts are generated if appropriate and the RIDLE bit is set. A break is regarded as a character that contains only zeros with the FERR flag

set. If a long break signal has been detected, the receiver will regard it as a data frame including a start bit, data bits and the invalid stop bit and the FERR flag will be set. The receiver must wait for a valid stop bit before looking for the next start bit. The receiver will not make the assumption that the break condition on the line is the next start bit. The break character will be loaded into the buffer and no further data will be received until stop bits are received. It should be noted that the RIDLE read only flag will go high when the stop bits have not yet been received. The reception of a break character on the UART registers will result in the following:

- The framing error flag, FERR, will be set.
- The receive data register, RXR, will be cleared.
- The OERR, NF, PERR, RIDLE or RXIF flags will possibly be set.

Idle Status

When the receiver is reading data, which means it will be in between the detection of a start bit and the reading of a stop bit, the receiver status flag in the USR register, otherwise known as the RIDLE flag, will have a zero value. In between the reception of a stop bit and the detection of the next start bit, the RIDLE flag will have a high value, which indicates the receiver is in an idle condition.

Receiver Interrupt

The read only receive interrupt flag RXIF in the USR register is set by an edge generated by the receiver. An interrupt is generated if RIE=1, when a word is transferred from the Receive Shift Register, RSR, to the Receive Data Register, RXR. An overrun error can also generate an interrupt if RIE=1.

Managing Receiver Errors

Several types of reception errors can occur within the UART module, the following section describes the various types and how they are managed by the UART.

Overrun Error – OERR

The RXR register is composed of a four byte deep FIFO data buffer, where four bytes can be held in the FIFO register, while a 5th byte can continue to be received. Before the 5th byte has been entirely shifted in, the data should be read from the RXR register. If this is not done, the overrun error flag OERR will be consequently indicated.

In the event of an overrun error occurring, the following will happen:

- The OERR flag in the USR register will be set.
- The RXR contents will not be lost.
- The shift register will be overwritten.
- An interrupt will be generated if the RIE bit is set.

The OERR flag can be cleared by an access to the USR register followed by a read to the RXR register.

Noise Error – NF

Over-sampling is used for data recovery to identify valid incoming data and noise. If noise is detected within a frame, the following will occur:

- The read only noise flag, NF, in the USR register will be set on the rising edge of the RXIF bit.
- Data will be transferred from the shift register to the RXR register.
- No interrupt will be generated. However this bit rises at the same time as the RXIF bit which itself generates an interrupt.

Note that the NF flag is reset by a USR register read operation followed by an RXR register read operation.

Framing Error – FERR

The read only framing error flag, FERR, in the USR register, is set if a zero is detected instead of stop bits. If two stop bits are selected, both stop bits must be high. Otherwise the FERR flag will be set. The FERR flag and the received data will be recorded in the USR and TXR_RXR registers respectively, and the flag is cleared in any reset.

Parity Error – PERR

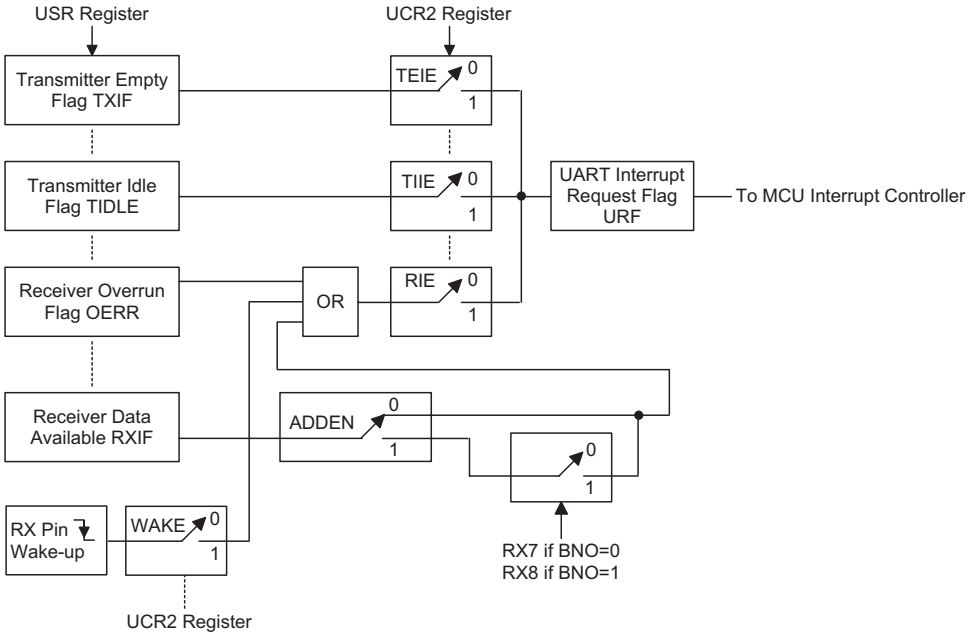
The read only parity error flag, PERR, in the USR register, is set if the parity of the received word is incorrect. This error flag is only applicable if the parity function is enabled, PREN=1, and if the parity type, odd or even, is selected. The read only PERR flag and the received data will be recorded in the USR and TXR_RXR registers respectively. It is cleared on any reset, it should be noted that the flags, FERR and PERR, in the USR register should first be read by the application program before reading the data word.

UART Interrupt Structure

Several individual UART conditions can generate a UART interrupt. When these conditions exist, a low pulse will be generated to get the attention of the microcontroller. These conditions are a transmitter data register empty, transmitter idle, receiver data available, receiver overrun, address detect and an RX pin wake-up. When any of these conditions are created, if its corresponding interrupt control is enabled and the stack is not full, the program will jump to its corresponding interrupt vector where it can be serviced before returning to the main program. Four of these conditions have the corresponding USR register flags which will generate a UART interrupt if its associated interrupt enable control bit in the UCR2 register is set. The two transmitter interrupt conditions have their own corresponding enable control bits, while the two receiver interrupt conditions have a shared enable control bit. These enable bits can be used to mask out individual UART interrupt sources.

The address detect condition, which is also a UART interrupt source, does not have an associated flag, but will generate a UART interrupt when an address detect condition occurs if its function is enabled by setting the ADDEN bit in the UCR2 register. An RX pin wake-up, which is also a UART interrupt source, does not have an associated flag, but will generate a UART interrupt if the microcontroller is woken up by a falling edge on the RX pin, if the WAKE and RIE bits in the UCR2 register are set. Note that in the event of an RX wake-up interrupt occurring, there will be a certain period of delay, commonly known as the System Start-up Time, for the oscillator to restart and stabilize before the system resumes normal operation.

Note that the USR register flags are read only and cannot be cleared or set by the application program, neither will they be cleared when the program jumps to the corresponding interrupt servicing routine, as is the case for some of the other interrupts. The flags will be cleared automatically when certain actions are taken by the UART, the details of which are given in the UART register section. The overall UART interrupt can be disabled or enabled by the related interrupt enable control bits in the interrupt control registers of the microcontroller to decide whether the interrupt requested by the UART module is masked out or allowed.



UART Interrupt Scheme

Address Detect Mode

Setting the Address Detect function enable control bit, ADDEN, in the UCR2 register, enables this special function. If this bit is set to 1, then an additional qualifier will be placed on the generation of a Receiver Data Available interrupt, which is requested by the RXIF flag. If the ADDEN bit is equal to 1, then when the data is available, an interrupt will only be generated, if the highest received bit has a high value. Note that the related interrupt enable control bits, URE and MF1E, and the EMI bit of the microcontroller must also be enabled for correct interrupt generation. The highest address bit is the 9th bit if the bit BNO=1 or the 8th bit if the bit BNO=0. If the highest bit is high, then the received word will be defined as an address rather than data. A Data Available interrupt will be generated every time the last bit of the received word is set. If the ADDEN bit is equal to 0, then a Receive Data Available interrupt will be generated each time the RXIF flag is set, irrespective of the data last bit status. The address detection and parity functions are mutually exclusive functions. Therefore, if the address detect function is enabled, then to ensure correct operation, the parity function should be disabled by resetting the parity function enable bit PREN to zero.

ADDEN	9th Bit if BNO=1 8th Bit if BNO=0	UART Interrupt Generated
0	0	√
	1	√
1	0	x
	1	√

ADDEN Bit Function

UART Power Down Mode and Wake-up

When the MCU is in the Power Down Mode, the UART will cease to function. When the device enters the IDLE or SLEEP Mode, all clock sources to the module are shutdown. If the MCU enters the Power Down Mode while a transmission is still in progress, then the transmission will be paused until the UART clock source derived from the microcontroller is activated. In a similar way, if the

MCU enters the IDLE or SLEEP Mode while receiving data, then the reception of data will likewise be paused. When the MCU enters the IDLE or SLEEP Mode, note that the USR, UCR1, UCR2, transmit and receive registers, as well as the BRG register will not be affected. It is recommended to make sure first that the UART data transmission or reception has been finished before the microcontroller enters the IDLE or SLEEP mode.

The UART function contains a receiver RX pin wake-up function, which is enabled or disabled by the WAKE bit in the UCR2 register. If this bit, along with the UART enable bit, UARTEN, the receiver enable bit, RXEN and the receiver interrupt bit, RIE, are all set before the MCU enters the IDLE or SLEEP Mode, then a falling edge on the RX pin will wake up the MCU from the IDLE or SLEEP Mode. Note that as it takes certain system clock cycles after a wake-up, before normal microcontroller operation resumes, any data received during this time on the RX pin will be ignored.

For a UART wake-up interrupt to occur, in addition to the bits for the wake-up being set, the global interrupt enable bit, EMI, the multi-function interrupt enable bit, MF1E, and the UART interrupt enable bit, URE, must also be set. If the EMI and MF1E bits are not set then only a wake up event will occur and no interrupt will be generated. Note also that as it takes certain system clock cycles after a wake-up before normal microcontroller resumes, the UART interrupt will not be generated until after this time has elapsed.

Low Voltage Detector – LVD

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

LVD Register

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

- **LVDC Register**

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

Bit 7~6 Unimplemented, read as "0"

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

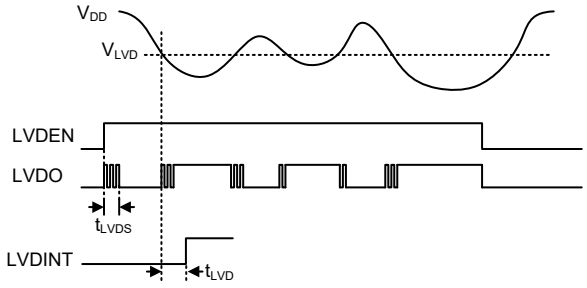
Bit 4 **LVDEN**: Low Voltage Detector Control
0: Disable
1: Enable

Bit 3 Unimplemented, read as "0"

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

LVD Operation

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



LVD Operation

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

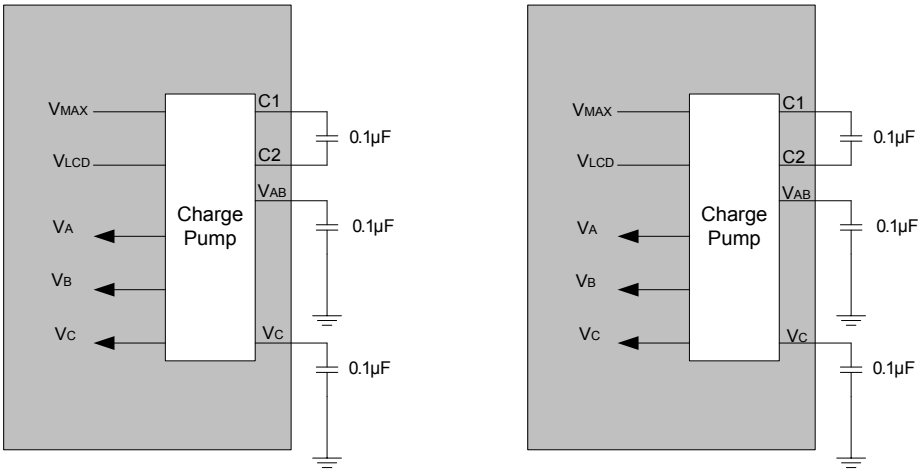
When LVD function is enabled, it is recommenced to clear LVD flag first, and then enables interrupt function to avoid mistake action.

LCD Driver

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

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

Part No.	Duty	Bias	Bias Type	Wave Type
HT45F65	1/4	1/3	C	A or B
HT45F66 HT45F67	1/4 1/6	1/3	C	A or B



C type and 1/3 Bias,
V_{AS}= "1"
V_{AB} = 3/2 xV_{LCD} = V_A
V_B = V_{LCD}
V_C = 1/2 x V_{LCD}

C type and 1/3 Bias,
V_{AS} = "0"
V_A = V_{LCD}
V_{AB} = 2/3 x V_{LCD} = V_B
V_C = 1/3 x V_{LCD}

C Type Bias Voltage Levels

Note: If V_{AS}= "1", V_{LCD}= V_{DD} and V_{MAX}= V_{AB}.
If V_{AS}= "0", V_{LCD}= V_{MAX}= V_{DD}.

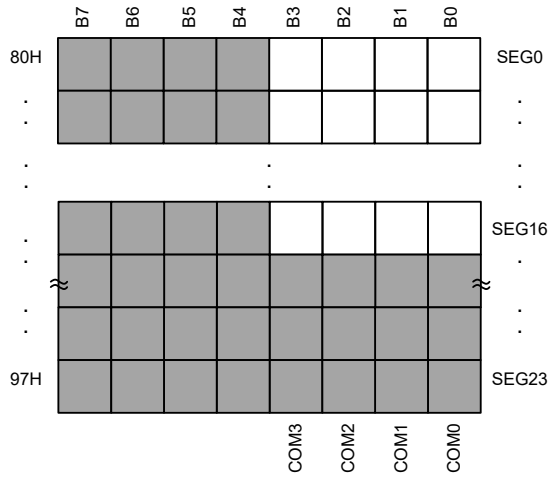
LCD Memory

An area of Data Memory is especially reserved for use for the LCD display data. This data area is known as the LCD Memory. Any data written here will be automatically read by the internal display driver circuits, which will in turn automatically generate the necessary LCD driving signals. Therefore any data written into this Memory will be immediately reflected into the actual display connected to the microcontroller.

As the LCD Memory addresses overlap those of the General Purpose Data Memory, it s stored in its own independent Bank 1 area. The Data Memory Bank to be used is chosen by using the Bank Pointer, which is a special function register in the Data Memory, with the name, BP. To access the LCD Memory therefore requires first that Bank 1 is selected by writing a value of 01H to the BP register.

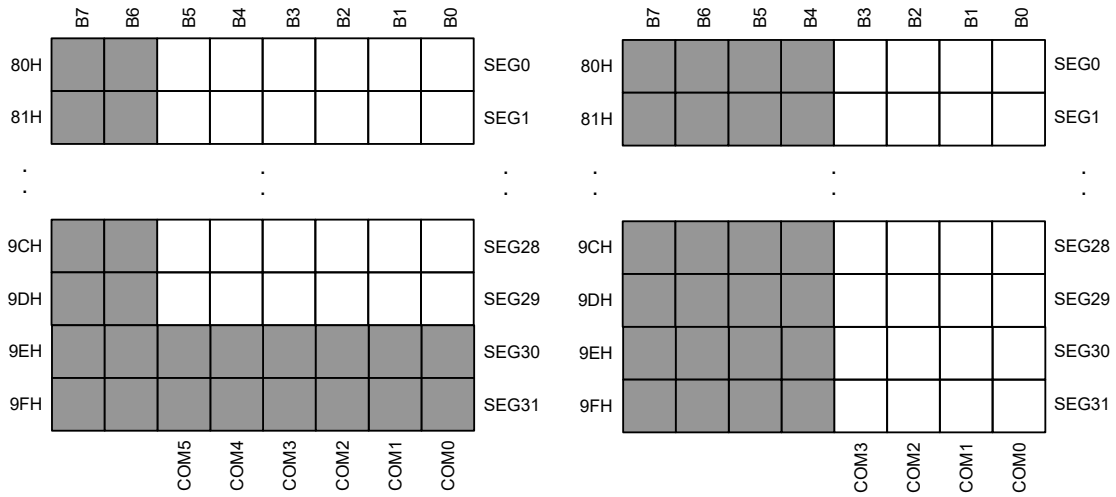
After this, the memory can then be accessed by using indirect addressing through the use of Memory Pointer MP1. With Bank 1 selected, then using MP1 to read or write to the memory area, starting with address 80H, will result in operations to the LCD Memory. Directly addressing the Display Memory is not applicable and will result in a data access to the Bank 0 General Purpose Data Memory.

The accompanying LCD Memory Map diagrams shows how the internal LCD Memory is mapped to the Segments and Commons of the display for the device. LCD Memory Maps for the device with smaller package type can be extrapolated from these diagrams.



LCD RAM (16 SEG × 4 COM)

LCD Memory Map − HT45F65



LCD RAM (30 SEG × 6 COM)

LCD RAM (32 SEG × 4 COM)

LCD Memory Map − HT45F66/HT45F67

LCD Registers

Control Registers in the Data Memory, are used to control the various setup features of the LCD Driver.

There is one control register for the LCD function, LCDC.

Various bits in this registers control functions such as duty type, overall LCD enable and disable. The LCDEN bit in the LCDC register, which provides the overall LCD enable/disable function, will only be effective when the device is in the Normal, Slow or Idle Mode. If the device is in the Sleep Mode then the display will always be disabled. The TYPE bit in the same register is used to select whether Type A or Type B LCD control signals are used.

• **LCDC Register – HT45F65**

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

Bit 7 **TYPE**: LCD Type Control
0: Type A
1: Type B

Bit 6 **VAS**: V_A voltage selection for C type LCD
0: V_A equals to V_{DD}
1: V_A equals $1.5 \times V_{DD}$

Bit 5~1 Unimplemented, read as "0"

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

• **LCDC Register – HT45F66/HT45F67**

Bit	7	6	5	4	3	2	1	0
Name	TYPE	VAS	DYT	—	—	—	—	LCDEN
R/W	R/W	R/W	R/W	—	—	—	—	R/W
POR	0	0	0	—	—	—	—	0

Bit 7 **TYPE**: LCD Type Control
0: Type A
1: Type B

Bit 6 **VAS**: V_A voltage selection for C type LCD
0: V_A equals to V_{DD}
1: V_A equals $1.5 \times V_{DD}$

Bit 5 **DYT**: LCD Duty Control
0: 1/4 duty, 32SEG × 4COM
1: 1/6 duty, 30SEG × 6COM

Bit 4~1 Unimplemented, read as "0"

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

Clock Source

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

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

LCD Clock Source

LCD Driver Output

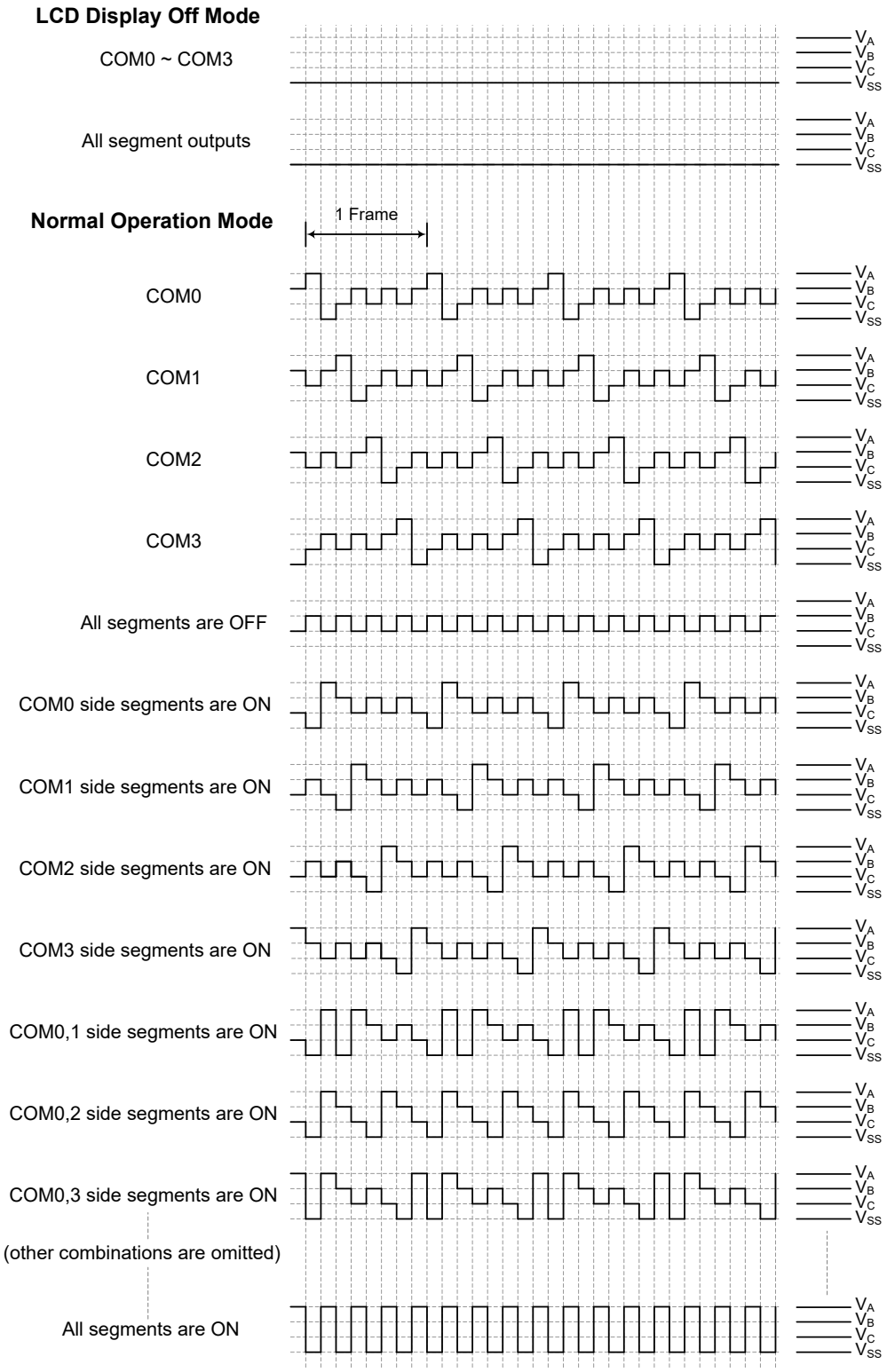
The number of COM and SEG outputs supplied by the LCD driver, as well as its duty selections, is dependent upon how the LCD control bits are programmed.

The nature of Liquid Crystal Displays require that only AC voltages can be applied to their pixels as the application of DC voltages to LCD pixels may cause permanent damage. For this reason the relative contrast of an LCD display is controlled by the actual RMS voltage applied to each pixel, which is equal to the RMS value of the voltage on the COM pin minus the voltage applied to the SEG pin. This differential RMS voltage must be greater than the LCD saturation voltage for the pixel to be on and less than the threshold voltage for the pixel to be off.

The requirement to limit the DC voltage to zero and to control as many pixels as possible with a minimum number of connections, requires that both a time and amplitude signal is generated and applied to the application LCD. These time and amplitude varying signals are automatically generated by the LCD driver circuits in the microcontroller. What is known as the duty determines the number of common lines used, which are also known as backplanes or COMs. The duty, which is chosen by a control bit to have a value of 1/3 and which equates to a COM number of 3, therefore defines the number of time divisions within each LCD signal frame. Two types of signal generation are also provided, known as Type A and Type B, the required type is selected via the TYPE bit in the LCDC register. Type B offers lower frequency signals, however lower frequencies may introduce flickering and influence display clarity.

LCD Waveform Timing Diagrams

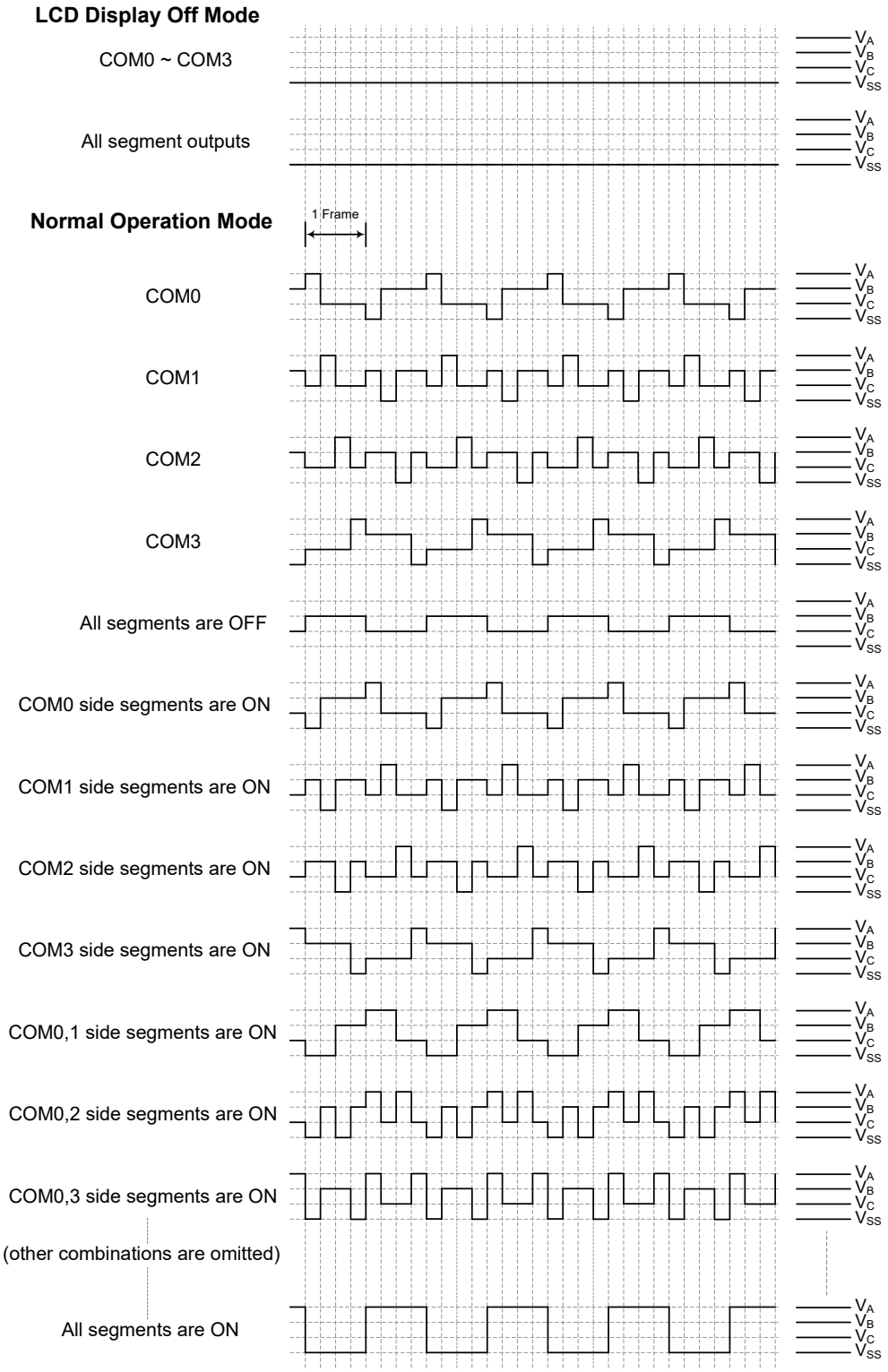
The accompanying timing diagrams depict the display driver signals generated by the microcontroller for various values of duty. The huge range of various permutations only permits a few types to be displayed here.



LCD Driver Output (1/4 Duty, 1/3 Bias, A-type waveform)

Note: 1. For 1/3 C type bias, $V_A=V_{LCD}\times1.5$, $V_B=V_{LCD}$ and $V_C=V_{LCD}\times1/2$.

2. The LCD function can be as on or off by software control.



LCD Driver Output (1/4 Duty, 1/3 Bias, B-type waveform)

Note: 1. For 1/3 C type bias, $V_A=V_{LCD}\times1.5$, $V_B=V_{LCD}$ and $V_C=V_{LCD}\times1/2$.

2. The LCD function can be as on or off by software control.

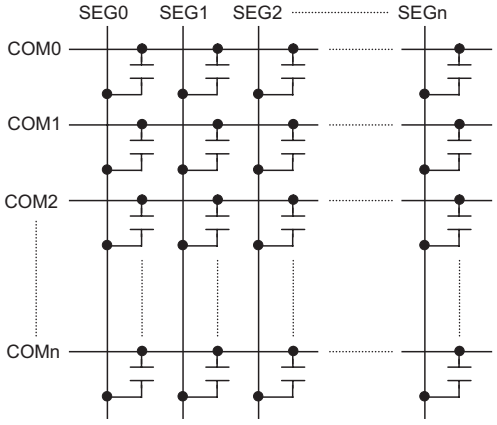
Programming Considerations

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

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

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

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



LCD Panel Equivalent Circuit

Temperature Sensor

This chip builds in temperature sensor and switchs for temperature measure application. When switch 5 is close (on) and switch 4 is open (off), the output signal of temperature sensor will output to channel 7 of ADC via OPA2.

All switchs is controlled in TSC register.

- **TSC Register**

Bit	7	6	5	4	3	2	1	0
Name	TSEN	—	SW5	SW4	SW3	SW2	SW1	SW0
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 **TSEN**: Temperature sensor control
0: disabled
1: enabled

Bit 6 Unimplemented, read as "0"

Bit 5 **SW5**: Switch 5 control
0: open
1: close
This bit is for strip2 measure.

Bit 4 **SW4**: Switch 4 control
0: open
1: close
This bit is for temperature sensor.

Bit 3 **SW3**: Switch 3 control
0: open
1: close
When this bit is close, the VG pin connects to ground.

Bit 2 **SW2**: Switch 2 control
0: open
1: close
When this bit is close, the OP1S2 pin connects to OP1O pin

Bit 1 **SW1**: Switch 1 control
0: open
1: close
When this bit is close, the OP1S1 pin connects to OP1O pin

Bit 0 **SW0**: Switch 0 control
0: open
1: close
When this bit is close, the OP1S0 pin connects to OP1O pin

Interrupts

Interrupts are an important part of any microcontroller system. When an external event or an internal function such as a Timer Module or an A/D converter requires microcontroller attention, their corresponding interrupt will enforce a temporary suspension of the main program allowing the microcontroller to direct attention to their respective needs. The device contains several external interrupt and internal interrupts functions. The external interrupts are generated by the action of the external INT0~INT1 and PINTB pins, while the internal interrupts are generated by various internal functions such as the TMs, Time Base, LVD, SIM, SPI1 and the A/D converter.

Interrupt Registers

Overall interrupt control, which basically means the setting of request flags when certain microcontroller conditions occur and the setting of interrupt enable bits by the application program, is controlled by a series of registers, located in the Special Purpose Data Memory. The number of registers depends upon the device chosen but fall into three categories. The first is the INTC0~INTC2 registers which setup the primary interrupts, the second is the MFI0~MFI3 registers which setup the Multi-function interrupts. Finally there is an INTEG register to setup the external interrupt trigger edge type.

Each register contains a number of enable bits to enable or disable individual registers as well as interrupt flags to indicate the presence of an interrupt request. The naming convention of these follows a specific pattern. First is listed an abbreviated interrupt type, then the (optional) number of that interrupt followed by either an "E" for enable/disable bit or "F" for request flag.

Function	Enable Bit	Request Flag	Notes
Global	EMI	—	—
INTn Pin	INTnE	INTnF	n=0 or 1
A/D Converter	ADE	ADF	—
Multi-function	MFnE	MFnF	n=0~3
Time Base	TBnE	TBnF	n=0 or 1
SIM	SIME	SIMF	—
SPI1	SPI1E	SPI1F	—
LVD	LVE	LVF	—
UART	URE	URF	—
PINTB Pin	XPE	XPF	—
TM	TnPE	TnPF	n=0~3
	TnAE	TnAF	n=0~3
	TnBE	TnBF	n=1

Interrupt Register Bit Naming Conventions

Interrupt Register Contents

HT45F65

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	ADF	INT1F	INT0F	ADE	INT1E	INT0E	EMI
INTC1	—	MF2F	MF1F	MF0F	—	MF2E	MF1E	MF0E
INTC2	—	—	TB1F	TB0F	—	—	TB1E	TB0E
MFI0	T1AF	T1PF	T0AF	T0PF	T1AE	T1PE	T0AE	T0PE
MFI1	—	URF	SPI1F	LVF	—	URE	SPI1E	LVE
MFI2	T2AF	T2PF	XPF	SIMF	T2AE	T2PE	XPE	SIME

HT45F66/HT45F67

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	ADF	INT1F	INT0F	ADE	INT1E	INT0E	EMI
INTC1	MF3F	MF2F	MF1F	MF0F	MF3E	MF2E	MF1E	MF0E
INTC2	—	—	TB1F	TB0F	—	—	TB1E	TB0E
MFI0	T2AF	T2PF	T0AF	T0PF	T2AE	T2PE	T0AE	T0PE
MFI1	URF	T1BF	T1AF	T1PF	URE	T1BE	T1AE	T1PE
MFI2	T3AF	T3PF	XPF	SIMF	T3AE	T3PE	XPE	SIME
MFI3	—	—	SPI1F	LVF	—	—	SPI1E	LVE

• **INTEG Register**

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

Bit 7~4 Unimplemented, read as "0"

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

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

• **INTC0 Register**

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

Bit 7 Unimplemented, read as "0"

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

Bit 5 **INT1F**: INT1 interrupt request flag
0: no request
1: interrupt request

Bit 4 **INT0F**: INT0 interrupt request flag
0: no request
1: interrupt request

Bit 3 **ADE**: A/D Converter Interrupt Control
0: disable
1: enable

Bit 2 **INT1E**: INT1 interrupt control
0: disable
1: enable

Bit 1 **INT0E**: INT0 interrupt control
0: disable
1: enable

Bit 0 **EMI**: Global interrupt control
0: disable
1: enable

• **INTC1 Register (HT45F65)**

Bit	7	6	5	4	3	2	1	0
Name	—	MF2F	MF1F	MF0F	—	MF2E	MF1E	MF0E
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

Bit 7 Unimplemented, read as "0"
Bit 6 **MF2F**: Multi-function Interrupt 2 Request Flag
0: no request
1: interrupt request
Bit 5 **MF1F**: Multi-function Interrupt 1 Request Flag
0: no request
1: interrupt request
Bit 4 **MF0F**: Multi-function Interrupt 0 Request Flag
0: no request
1: interrupt request
Bit 3 Unimplemented, read as "0"
Bit 2 **MF2E**: Multi-function Interrupt 2 Control
0: disable
1: enable
Bit 1 **MF1E**: Multi-function Interrupt 1 Control
0: disable
1: enable
Bit 0 **MF0E**: Multi-function Interrupt 0 Control
0: disable
1: enable

• **INTC1 Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	MF3F	MF2F	MF1F	MF0F	MF3E	MF2E	MF1E	MF0E
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 **MF3F**: Multi-function Interrupt 3 Request Flag
0: no request
1: interrupt request
Bit 6 **MF2F**: Multi-function Interrupt 2 Request Flag
0: no request
1: interrupt request
Bit 5 **MF1F**: Multi-function Interrupt 1 Request Flag
0: no request
1: interrupt request
Bit 4 **MF0F**: Multi-function Interrupt 0 Request Flag
0: no request
1: interrupt request
Bit 3 **MF3E**: Multi-function Interrupt 3 Control
0: disable
1: enable
Bit 2 **MF2E**: Multi-function Interrupt 2 Control
0: disable
1: enable
Bit 1 **MF1E**: Multi-function Interrupt 1 Control
0: disable
1: enable
Bit 0 **MF0E**: Multi-function Interrupt 0 Control
0: disable
1: enable

• **INTC2 Register**

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

Bit 7~6 Unimplemented, read as "0"

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

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

Bit 3~2 Unimplemented, read as "0"

Bit 1 **TB1E**: Time Base 1 Interrupt Control
0: disable
1: enable

Bit 0 **TB0E**: Time Base 0 Interrupt Control
0: disable
1: enable

• **MFI0 Register (HT45F65)**

Bit	7	6	5	4	3	2	1	0
Name	T1AF	T1PF	T0AF	T0PF	T1AE	T1PE	T0AE	T0PE
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 **T1AF**: TM1 Comparator A match interrupt request flag
0: no request
1: interrupt request

Bit 6 **T1PF**: TM1 Comparator P match interrupt request flag
0: no request
1: interrupt request

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

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

Bit 3 **T1AE**: TM1 Comparator A match interrupt control
0: disable
1: enable

Bit 2 **T1PE**: TM1 Comparator P match interrupt control
0: disable
1: enable

Bit 1 **T0AE**: TM0 Comparator A match interrupt control
0: disable
1: enable

Bit 0 **T0PE**: TM0 Comparator P match interrupt control
0: disable
1: enable

• **MFI0 Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	T2AF	T2PF	T0AF	T0PF	T2AE	T2PE	T0AE	T0PE
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 **T2AF**: TM2 Comparator A match interrupt request flag
0: no request
1: interrupt request

Bit 6 **T2PF**: TM2 Comparator P match interrupt request flag
0: no request
1: interrupt request

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

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

Bit 3 **T2AE**: TM2 Comparator A match interrupt control
0: disable
1: enable

Bit 2 **T2PE**: TM2 Comparator P match interrupt control
0: disable
1: enable

Bit 1 **T0AE**: TM0 Comparator A match interrupt control
0: disable
1: enable

Bit 0 **T0PE**: TM0 Comparator P match interrupt control
0: disable
1: enable

• **MFI1 Register (HT45F65)**

Bit	7	6	5	4	3	2	1	0
Name	—	URF	SPI1F	LVF	—	URE	SPI1E	LVE
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

Bit 7 Unimplemented, read as “0”

Bit 6 **URF**: UART interrupt request flag
0: no request
1: interrupt request

Bit 5 **SPI1F**: SPI1 interrupt request flag
0: no request
1: interrupt request

Bit 4 **LVF**: LVD interrupt request flag
0: no request
1: interrupt request

Bit 3 Unimplemented, read as “0”

Bit 2 **URE**: UART Interrupt Control
0: disable
1: enable

Bit 1 **SPI1E**: SPI1 Interrupt Control
0: disable
1: enable

Bit 0 **LVE**: LVD Interrupt Control
0: disable
1: enable

• **MFI1 Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	URF	T1BF	T1AF	T1PF	URE	T1BE	T1AE	T1PE
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 **URF**: UART interrupt request flag
0: no request
1: interrupt request

Bit 6 **T1BF**: TM1 Comparator B match interrupt request flag
0: no request
1: interrupt request

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

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

Bit 3 **URE**: UART interrupt control
0: disable
1: enable

Bit 2 **T1BE**: TM1 Comparator B match interrupt control
0: disable
1: enable

Bit 1 **T1AE**: TM1 Comparator A match interrupt control
0: disable
1: enable

Bit 0 **T1PE**: TM1 Comparator P match interrupt control
0: disable
1: enable

• **MFI2 Register (HT45F65)**

Bit	7	6	5	4	3	2	1	0
Name	T2AF	T2PF	XPF	SIMF	T2AE	T2PE	XPE	SIME
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 **T2AF**: TM2 Comparator A match interrupt request flag
0: no request
1: interrupt request

Bit 6 **T2PF**: TM2 Comparator P match interrupt request flag
0: no request
1: interrupt request

Bit 5 **XPF**: External peripheral interrupt request flag
0: no request
1: interrupt request

Bit 4 **SIMF**: SIM interrupt request flag
0: no request
1: interrupt request

Bit 3 **T2AE**: TM2 Comparator A match interrupt control
0: disable
1: enable

Bit 2 **T2PE**: TM2 Comparator P match interrupt control
0: disable
1: enable

Bit 1 **XPE**: External Peripheral Interrupt Control
0: disable
1: enable

Bit 0 **SIME**: SIM Interrupt Control
0: disable
1: enable

• **MFI2 Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	T3AF	T3PF	XPF	SIMF	T3AE	T3PE	XPE	SIME
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 **T3AF**: TM3 Comparator A match interrupt request flag
0: no request
1: interrupt request

Bit 6 **T3PF**: TM3 Comparator P match interrupt request flag
0: no request
1: interrupt request

Bit 5 **XPF**: External peripheral interrupt request flag
0: no request
1: interrupt request

Bit 4 **SIMF**: SIM interrupt request flag
0: no request
1: interrupt request

Bit 3 **T3AE**: TM3 Comparator A match interrupt control
0: disable
1: enable

Bit 2 **T3PE**: TM3 Comparator P match interrupt control
0: disable
1: enable

Bit 1 **XPE**: External Peripheral Interrupt Control
0: disable
1: enable

Bit 0 **SIME**: SIM Interrupt Control
0: disable
1: enable

• **MFI3 Register (HT45F66/HT45F67)**

Bit	7	6	5	4	3	2	1	0
Name	—	—	SPI1F	LVF	—	—	SPI1E	LVE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5 **SPI1F**: SPI1 interrupt request flag
0: no request
1: interrupt request

Bit 4 **LVF**: LVD interrupt request flag
0: no request
1: interrupt request

Bit 3~2 Unimplemented, read as "0"

Bit 1 **SPI1E**: SPI1 Interrupt Control
0: disable
1: enable

Bit 0 **LVE**: LVD Interrupt Control
0: disable
1: enable

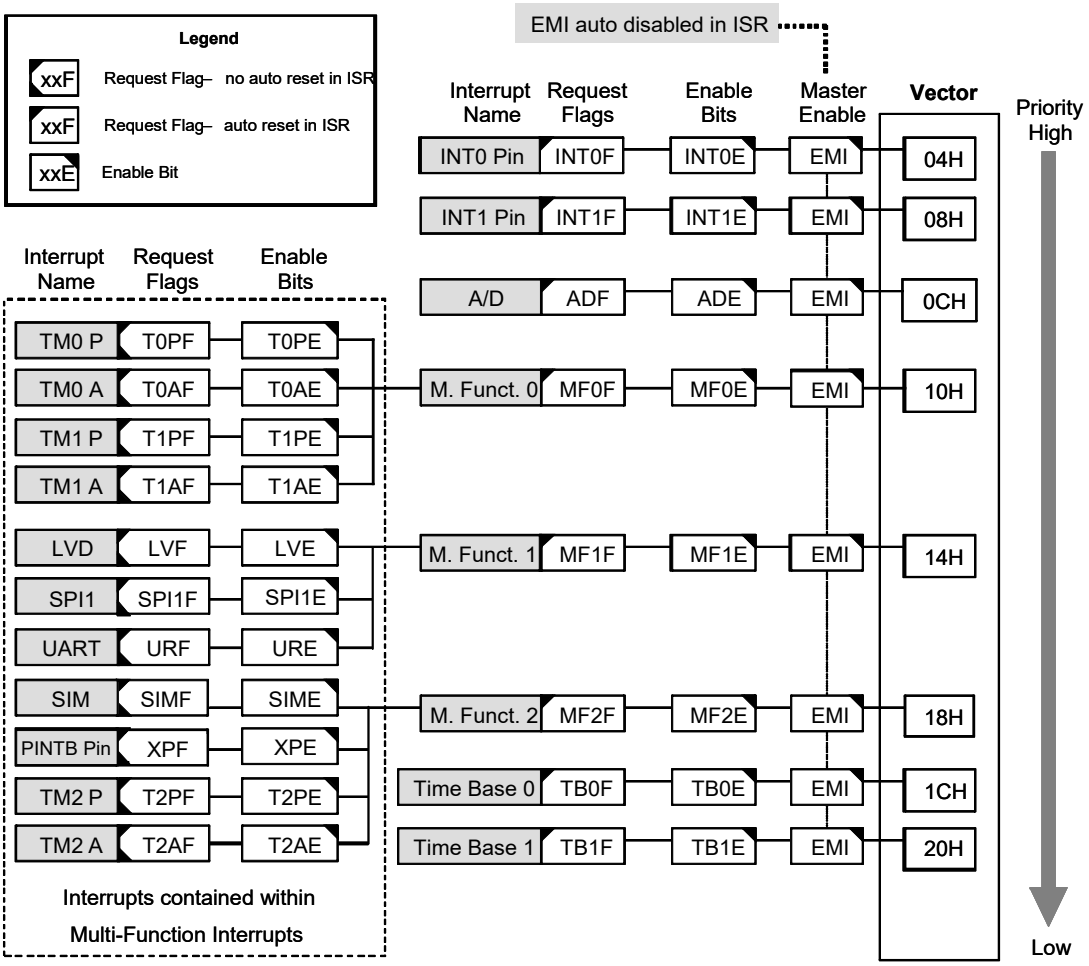
Interrupt Operation

When the conditions for an interrupt event occur, such as a TM Comparator P, Comparator A or Comparator B 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.

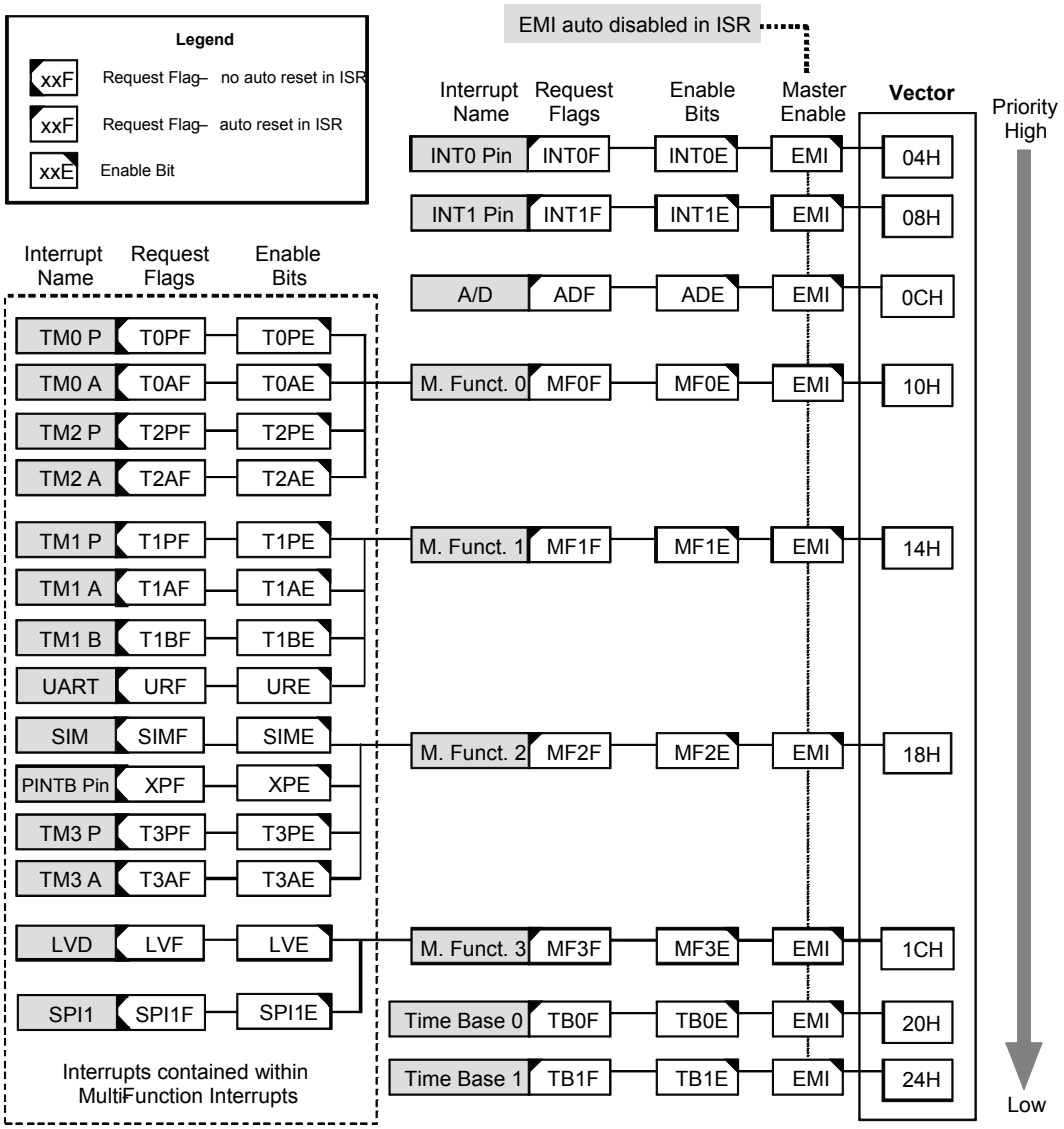
When an interrupt is generated, the Program Counter, which stores the address of the next instruction to be executed, will be transferred onto the stack. The Program Counter will then be loaded with a new address which will be the value of the corresponding interrupt vector. The microcontroller will 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 – HT45F65



Interrupt Structure – HT45F66/HT45F67

External Interrupts

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

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

Multi-function Interrupt

Within the devices there are three or four Multi-function interrupts. Unlike the other independent interrupts, these interrupts have no independent source, but rather are formed from other existing interrupt sources, namely the TM Interrupts, UART interrupt, SIM Interrupt, SPI1 Interrupt, External Peripheral Interrupt and LVD interrupt.

A Multi-function interrupt request will take place when any of the Multi-function interrupt request flags, MFnF are set. The Multi-function interrupt flags will be set when any of their included functions generate an interrupt request flag. To allow the program to branch to their respective interrupt vector addresses, when the Multi-function interrupt is enabled and the stack is not full, and either one of the interrupts contained within each of Multi-function interrupt occurs, a subroutine call to one of the Multi-function interrupt vectors will take place. When the interrupt is serviced, the related Multi-Function request flag will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts.

However, it must be noted that, although the Multi-function Interrupt flags will be automatically reset when the interrupt is serviced, the request flags from the original source of the Multi-function interrupts, namely the TM Interrupts, UART interrupt, SIM Interrupt, SPI1 Interrupt, External Peripheral Interrupt and LVD interrupt will not be automatically reset and must be manually reset by the application program.

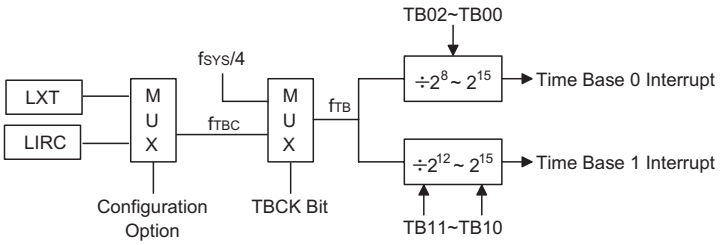
A/D Converter Interrupt

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

Time Base Interrupts

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

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



Time Base Interrupt

• **TBC Register**

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

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

Bit 6 **TBCK**: Select f_{TB} Clock
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 **LXTLP**: LXT Low Power Control
0: Disable (LXT quick start-up)
1: Enable (LXT slow start-up)

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: $32768/f_{TB}$

Serial Interface Module Interrupt

The Serial Interface Module Interrupt, also known as the SIM interrupt, is contained within the Multi-function Interrupt. A SIM Interrupt request will take place when the SIM Interrupt request flag, SIMF, is set, which occurs when a byte of data has been received or transmitted by the SIM interface. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the Serial Interface Interrupt enable bit, SIME, and Muti-function interrupt enable bits, must first be set. When the interrupt is enabled, the stack is not full and a byte of data has been transmitted or received by the SIM interface, a subroutine call to the respective Multi-function Interrupt vector, will take place. When the Serial Interface Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared. As the SIMF flag will not be automatically cleared, it has to be cleared by the application program.

External Peripheral Interrupt

The External Peripheral Interrupt operates in a similar way to the external interrupt and is contained within the Multi-function Interrupt. A Peripheral Interrupt request will take place when the External Peripheral Interrupt request flag, XPF, is set, which occurs when a negative edge transition appears on the PINTB pin. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, external peripheral interrupt enable bit, XPE, and associated Multi-function interrupt enable bit, must first be set. When the interrupt is enabled, the stack is not full and a negative transition appears on the External Peripheral Interrupt pin, a subroutine call to the respective Multi-function Interrupt, will take place. When the External Peripheral Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared.

As the XPF flag will not be automatically cleared, it has to be cleared by the application program. The external peripheral interrupt pin is pin-shared with several other pins with different functions. It must therefore be properly configured to enable it to operate as an External Peripheral Interrupt pin.

Serial Peripheral Interface Interrupt

The Serial Interface Interrupt, also known as the SPI1 interrupt, is contained within the Multi-function Interrupt. A SPI1 Interrupt request will take place when the SPI1 Interrupt request flag, SPI1F, is set, which occurs when a byte of data has been received or transmitted by the SPI1 interface. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the Serial Interface Interrupt enable bit, SPI1E, and Multi-function interrupt enable bits, must first be set. When the interrupt is enabled, the stack is not full and a byte of data has been transmitted or received by the SPI1 interface, a subroutine call to the respective Multi-function Interrupt vector, will take place. When the Serial Interface Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared. As the SPI1F flag will not be automatically cleared, it has to be cleared by the application program.

LVD Interrupt

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

the Multi-function Interrupt vector, will take place. When the Low Voltage Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared. As the LVF flag will not be automatically cleared, it has to be cleared by the application program.

TM Interrupts

The Compact and Standard Type TMs have two interrupts each, while the Enhanced Type TM has three interrupts. All of the TM interrupts are contained within the Multi-function Interrupts. For each of the Compact and Standard Type TMs there are two interrupt request flags TnPF and TnAF and two enable bits TnPE and TnAE. For the Enhanced Type TM there are three interrupt request flags TnPF, TnAF and TnBF and three enable bits TnPE, TnAE and TnBE. A TM interrupt request will take place when any of the TM request flags are set, a situation which occurs when a TM comparator P, A or B match situation happens.

To allow the program to branch to their respective interrupt vector addresses, the global interrupt enable bit, EMI, respective TM Interrupt enable bit, and relevant Multi-function Interrupt enable bit, MFnE, must first be set. When the interrupt is enabled, the stack is not full and a TM comparator match situation occurs, a subroutine call to the relevant Multi-function Interrupt vector locations, will take place. When the TM interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the related MFnF flag will be automatically cleared. As the TM interrupt request flags will not be automatically cleared, they have to be cleared by the application program.

UART Interrupt

The UART Interrupt is contained within the Multi-function Interrupt. A UART Interrupt request will take place when the UART Interrupt request flag, URF, is set, which occurs when a transmitter data register empty, transmitter idle, receiver data available, receiver overrun, address detect and an RX pin wake-up. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the UART Interrupt enable bit, URE, and Muti-function interrupt enable bits, must first be set. When the interrupt is enabled, the stack is not full and a UART interrupt condition occurs, a subroutine call to the respective Multi-function Interrupt vector, will take place. When the UART Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared. As the URF flag will not be automatically cleared, it has to be cleared by the application program.

Interrupt Wake-up Function

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

Programming Considerations

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

Where a certain interrupt is contained within a Multi-function interrupt, then when the interrupt service routine is executed, as only the Multi-function interrupt request flags, MF0F~MF3F, 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.

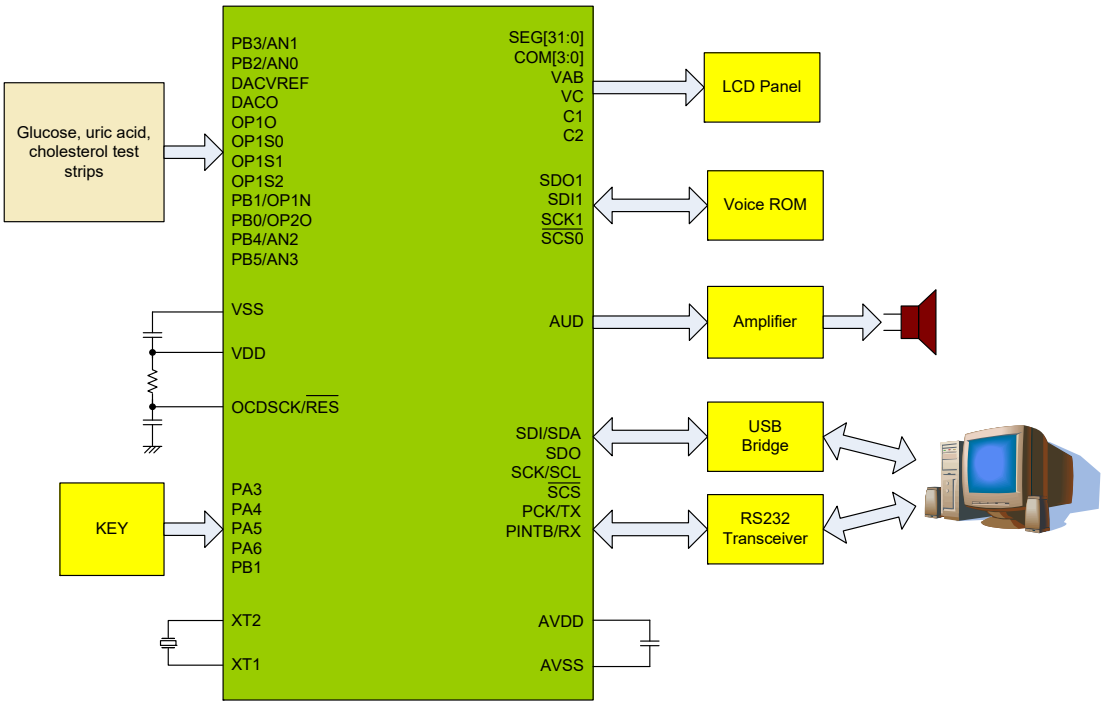
Configuration Options

Configuration options refer to certain options within the MCU that are programmed into the device during the programming process. During the development process, these options are selected using the HT-IDE software development tools. All options must be defined for proper system function, the details of which are shown in the table.

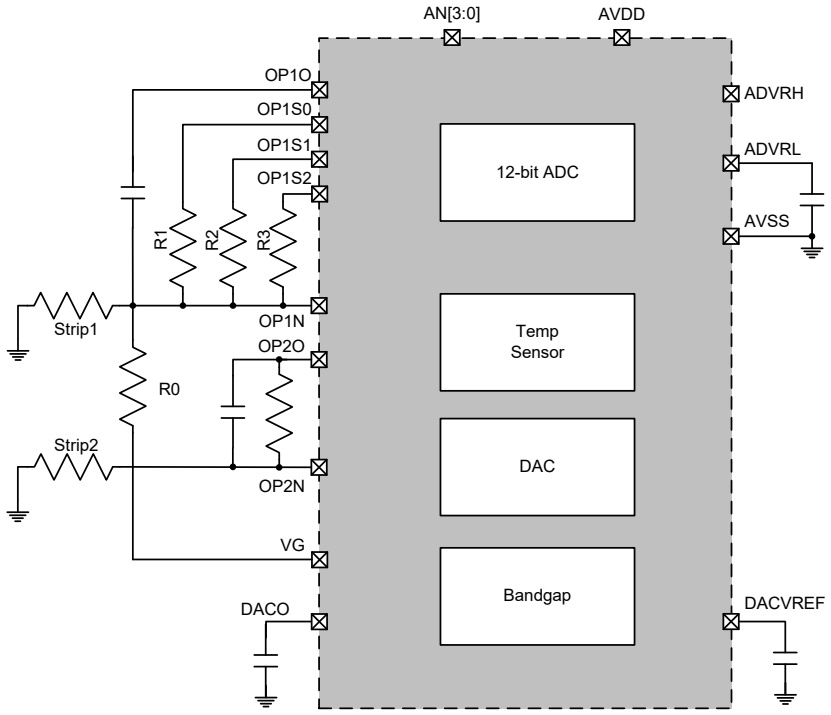
No.	Options
Oscillator Options	
1	High Speed Oscillator Selection – f_H: HXT ERC, Filter on (HT45F66/HT45F67 only) EC, Filter off (HT45F66/HT45F67 only) HIRC
2	Low Speed Oscillator Selection – f_L: LXT or LIRC
3	HXT Mode Selection: HT45F65: <10MHz or >10MHz HT45F66/HT45F67: 1MH~12MHz or 455kHz
4	HIRC Frequency Selection – f_{HIRC}: 4MHz, 8MHz or 12MHz
Watchdog Options	
5	Watchdog Timer Control: Always Enable or By S/W Control (WDTC)
6	Watchdog Timer Clock Selection – f_S: f_{SUB} or $f_{SYS}/4$
INTn Option	
7	INTn Pin RC Filter Control: Disable Enable
I²C Option	
8	I²C Debounce Time Selection: No debounce 1 system clock 2 system clocks

Application Circuits

Application Block Diagram



Glucose Measure Circuit



Note: For higher precision application requirements, it is recommended to use an external voltage reference source (connected to DACVREF pin).

Instruction Set

Introduction

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

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

Instruction Timing

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

Moving and Transferring Data

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

Arithmetic Operations

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

Logical and Rotate Operation

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

Branches and Control Transfer

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

Bit Operations

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

Table Read Operations

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

Other Operations

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

Instruction Set Summary

The instructions related to the data memory access in the following table can be used when the desired data memory is located in Data Memory sector 0.

Table Conventions

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

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

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

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

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

Extended Instruction Set

The extended instructions are used to support the full range address access for the data memory. When the accessed data memory is located in any data memory sector except sector 0, the extended instruction can be used to directly access the data memory instead of using the indirect addressing access. This can not only reduce the use of Flash memory space but also improve the CPU execution efficiency.

Mnemonic	Description	Cycles	Flag Affected
Arithmetic			
LADD A,[m]	Add Data Memory to ACC	2	Z, C, AC, OV, SC
LADDM A,[m]	Add ACC to Data Memory	2Note	Z, C, AC, OV, SC
LADC A,[m]	Add Data Memory to ACC with Carry	2	Z, C, AC, OV, SC
LADCM A,[m]	Add ACC to Data memory with Carry	2Note	Z, C, AC, OV, SC
LSUB A,[m]	Subtract Data Memory from ACC	2	Z, C, AC, OV, SC, CZ
LSUBM A,[m]	Subtract Data Memory from ACC with result in Data Memory	2Note	Z, C, AC, OV, SC, CZ
LSBC A,[m]	Subtract Data Memory from ACC with Carry	2	Z, C, AC, OV, SC, CZ
LSBCM A,[m]	Subtract Data Memory from ACC with Carry, result in Data Memory	2Note	Z, C, AC, OV, SC, CZ
LDAA [m]	Decimal adjust ACC for Addition with result in Data Memory	2Note	C
Logic Operation			
LAND A,[m]	Logical AND Data Memory to ACC	2	Z
LOR A,[m]	Logical OR Data Memory to ACC	2	Z
LXOR A,[m]	Logical XOR Data Memory to ACC	2	Z
LANDM A,[m]	Logical AND ACC to Data Memory	2Note	Z
LORM A,[m]	Logical OR ACC to Data Memory	2Note	Z
LXORM A,[m]	Logical XOR ACC to Data Memory	2Note	Z
LCPL [m]	Complement Data Memory	2Note	Z
LCPLA [m]	Complement Data Memory with result in ACC	2	Z
Increment & Decrement			
LINCA [m]	Increment Data Memory with result in ACC	2	Z
LINC [m]	Increment Data Memory	2Note	Z
LDECA [m]	Decrement Data Memory with result in ACC	2	Z
LDEC [m]	Decrement Data Memory	2Note	Z
Rotate			
LRRA [m]	Rotate Data Memory right with result in ACC	2	None
LRR [m]	Rotate Data Memory right	2Note	None
LRRCA [m]	Rotate Data Memory right through Carry with result in ACC	2	C
LRRC [m]	Rotate Data Memory right through Carry	2Note	C
LRLA [m]	Rotate Data Memory left with result in ACC	2	None
LRL [m]	Rotate Data Memory left	2Note	None
LRLCA [m]	Rotate Data Memory left through Carry with result in ACC	2	C
LRLC [m]	Rotate Data Memory left through Carry	2Note	C
Data Move			
LMOV A,[m]	Move Data Memory to ACC	2	None
LMOV [m],A	Move ACC to Data Memory	2Note	None
Bit Operation			
LCLR [m].i	Clear bit of Data Memory	2Note	None
LSET [m].i	Set bit of Data Memory	2Note	None

Mnemonic	Description	Cycles	Flag Affected
Branch			
LSZ [m]	Skip if Data Memory is zero	2Note	None
LSZA [m]	Skip if Data Memory is zero with data movement to ACC	2Note	None
LSNZ [m]	Skip if Data Memory is not zero	2Note	None
LSZ [m].i	Skip if bit i of Data Memory is zero	2Note	None
LSNZ [m].i	Skip if bit i of Data Memory is not zero	2Note	None
LSIZ [m]	Skip if increment Data Memory is zero	2Note	None
LSDZ [m]	Skip if decrement Data Memory is zero	2Note	None
LSIZA [m]	Skip if increment Data Memory is zero with result in ACC	2Note	None
LSDZA [m]	Skip if decrement Data Memory is zero with result in ACC	2Note	None
Table Read			
LTABRD [m]	Read table (specific page) to TBLH and Data Memory	3Note	None
LTABRDL [m]	Read table (last page) to TBLH and Data Memory	3Note	None
LITABRD [m]	Increment table pointer TBLP first and Read table (specific page) to TBLH and Data Memory	3Note	None
LITABRDL [m]	Increment table pointer TBLP first and Read table (last page) to TBLH and Data Memory	3Note	None
Miscellaneous			
LCLR [m]	Clear Data Memory	2Note	None
LSET [m]	Set Data Memory	2Note	None
LSWAP [m]	Swap nibbles of Data Memory	2Note	None
LSWAPA [m]	Swap nibbles of Data Memory with result in ACC	2	None

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

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

Instruction Definition

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

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

Operation ACC ← ACC + [m] + C

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

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

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

Operation [m] ← ACC + [m] + C

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

ADD A,[m] Add Data Memory to ACC

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

Operation ACC ← ACC + [m]

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

ADD A,x Add immediate data to ACC

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

Operation ACC ← ACC + x

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

ADDM A,[m] Add ACC to Data Memory

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

Operation [m] ← ACC + [m]

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

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

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

Operation ACC ← ACC "AND" [m]

Affected flag(s) Z

AND A,x Logical AND immediate data to ACC

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

Operation ACC ← ACC "AND" x

Affected flag(s) Z

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

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

Operation [m] ← ACC "AND" [m]

Affected flag(s) Z

CALL addr Subroutine call

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

Operation Stack ← Program Counter + 1
Program Counter ← addr

Affected flag(s) None

CLR [m] Clear Data Memory

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

Operation [m] ← 00H

Affected flag(s) None

CLR [m].i Clear bit of Data Memory

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

Operation [m].i ← 0

Affected flag(s) None

CLR WDT Clear Watchdog Timer

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

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CPL [m] Complement Data Memory

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

Operation [m] ← [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 ← [m]

Affected flag(s) Z

DAA [m] Decimal-Adjust ACC for addition with result in Data Memory

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

Operation [m] ← ACC + 00H or
[m] ← ACC + 06H or
[m] ← ACC + 60H or
[m] ← ACC + 66H

Affected flag(s) C

DEC [m] Decrement Data Memory

Description Data in the specified Data Memory is decremented by 1.

Operation [m] ← [m] − 1

Affected flag(s) Z

DECA [m] Decrement Data Memory with result in ACC

Description Data in the specified Data Memory is decremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC ← [m] − 1

Affected flag(s) Z

HALT Enter power down mode

Description This instruction stops the program execution and turns off the system clock. The contents of the Data Memory and registers are retained. The WDT and prescaler are cleared. The power down flag PDF is set and the WDT time-out flag TO is cleared.

Operation TO ← 0
PDF ← 1

Affected flag(s) TO, PDF

INC [m] Increment Data Memory

Description Data in the specified Data Memory is incremented by 1.

Operation [m] ← [m] + 1

Affected flag(s) Z

INCA [m] Increment Data Memory with result in ACC

Description Data in the specified Data Memory is incremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC ← [m] + 1

Affected flag(s) Z

JMP addr Jump unconditionally

Description The contents of the Program Counter are replaced with the specified address. Program execution then continues from this new address. As this requires the insertion of a dummy instruction while the new address is loaded, it is a two cycle instruction.

Operation Program Counter ← addr

Affected flag(s) None

MOV A,[m] Move Data Memory to ACC

Description The contents of the specified Data Memory are copied to the Accumulator.

Operation ACC ← [m]

Affected flag(s) None

MOV A,x Move immediate data to ACC

Description The immediate data specified is loaded into the Accumulator.

Operation ACC ← x

Affected flag(s) None

MOV [m],A Move ACC to Data Memory

Description The contents of the Accumulator are copied to the specified Data Memory.

Operation [m] ← ACC

Affected flag(s) None

NOP No operation

Description No operation is performed. Execution continues with the next instruction.

Operation No operation

Affected flag(s) None

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

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

Operation ACC ← ACC "OR" [m]

Affected flag(s) Z

OR A,x Logical OR immediate data to ACC

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

Operation ACC ← ACC "OR" x

Affected flag(s) Z

ORM A,[m] Logical OR ACC to Data Memory

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

Operation [m] ← ACC "OR" [m]

Affected flag(s) Z

RET Return from subroutine

Description The Program Counter is restored from the stack. Program execution continues at the restored address.

Operation Program Counter ← Stack

Affected flag(s) None

RET A,x Return from subroutine and load immediate data to ACC

Description The Program Counter is restored from the stack and the Accumulator loaded with the specified immediate data. Program execution continues at the restored address.

Operation Program Counter ← Stack
ACC ← x

Affected flag(s) None

RETI Return from interrupt

Description The Program Counter is restored from the stack and the interrupts are re-enabled by setting the EMI bit. EMI is the master interrupt global enable bit. If an interrupt was pending when the RETI instruction is executed, the pending Interrupt routine will be processed before returning to the main program.

Operation Program Counter ← Stack
EMI ← 1

Affected flag(s) None

RL [m] Rotate Data Memory left

Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0.

Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← [m].7

Affected flag(s) None

RLA [m] Rotate Data Memory left with result in ACC

Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← [m].7

Affected flag(s) None

RLC [m] Rotate Data Memory left through Carry

Description The contents of the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into bit 0.

Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← C
C ← [m].7

Affected flag(s) C

RLCA [m] Rotate Data Memory left through Carry with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into the bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← C
C ← [m].7

Affected flag(s) C

RR [m] Rotate Data Memory right

Description The contents of the specified Data Memory are rotated right by 1 bit with bit 0 rotated into bit 7.

Operation [m].i ← [m].(i+1); (i=0~6)
[m].7 ← [m].0

Affected flag(s) None

RRA [m] Rotate Data Memory right with result in ACC

Description Data in the specified Data Memory is rotated right by 1 bit with bit 0 rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.i ← [m].(i+1); (i=0~6)
ACC.7 ← [m].0

Affected flag(s) None

RRC [m] Rotate Data Memory right through Carry

Description The contents of the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7.

Operation [m].i ← [m].(i+1); (i=0~6)
[m].7 ← C
C ← [m].0

Affected flag(s) C

RRCA [m] Rotate Data Memory right through Carry with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.i ← [m].(i+1); (i=0~6)
ACC.7 ← C
C ← [m].0

Affected flag(s) C

SBC A,[m] Subtract Data Memory from ACC with Carry

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

Operation ACC ← ACC − [m] − C

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

SBC A, x Subtract immediate data from ACC with Carry

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

Operation ACC ← ACC – [m] – C

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

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

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

Operation [m] ← ACC − [m] − C

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

SDZ [m] Skip if decrement Data Memory is 0

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

Operation [m] ← [m] − 1
Skip if [m]=0

Affected flag(s) None

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

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

Operation ACC ← [m] − 1
Skip if ACC=0

Affected flag(s) None

SET [m] Set Data Memory

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

Operation [m] ← FFH

Affected flag(s) None

SET [m].i Set bit of Data Memory

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

Operation [m].i ← 1

Affected flag(s) None

SIZ [m] Skip if increment Data Memory is 0

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

Operation [m] ← [m] + 1
Skip if [m]=0

Affected flag(s) None

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

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

Operation ACC ← [m] + 1
Skip if ACC=0

Affected flag(s) None

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

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

Operation Skip if [m].i ≠ 0

Affected flag(s) None

SNZ [m] Skip if Data Memory is not 0

Description The contents of the specified Data Memory are read out and then written back to the specified Data Memory again. If the specified Data Memory is not 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is 0 the program proceeds with the following instruction.

Operation Skip if [m]≠ 0

Affected flag(s) None

SUB A,[m] Subtract Data Memory from ACC

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

Operation ACC ← ACC − [m]

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

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

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

Operation [m] ← ACC − [m]

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

SUB A,x Subtract immediate data from ACC

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

Operation ACC ← ACC − x

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

SWAP [m] Swap nibbles of Data Memory

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

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s) None

SWAPA [m] Swap nibbles of Data Memory with result in ACC

Description The low-order and high-order nibbles of the specified Data Memory are interchanged. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC.3~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0

Affected flag(s) None

SZ [m] Skip if Data Memory is 0

Description The contents of the specified Data Memory are read out and then written back to the specified Data Memory again. 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 (TBLP and TBHP) is moved to the specified Data Memory and the high byte moved to TBLH.

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

Affected flag(s) None

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

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

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

Affected flag(s) None

ITABRD [m] Increment table pointer low byte first and read table (specific page) to TBLH and Data Memory Description Increment table pointer low byte, TBLP, first and then the program code (specific page) addressed by the table pointer (TBHP and TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)

TBLH ← program code (high byte)

Affected flag(s) None

ITABRDL [m] Increment table pointer low byte first and read table (last page) to TBLH and Data Memory

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

Operation [m] ← program code (low byte)

TBLH ← program code (high byte)

Affected flag(s) None

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

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

Operation ACC ← ACC "XOR" [m]

Affected flag(s) Z

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

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

Operation [m] ← ACC "XOR" [m]

Affected flag(s) Z

XOR A,x Logical XOR immediate data to ACC

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

Operation ACC ← ACC "XOR" x

Affected flag(s) Z

Extended Instruction Definition

The extended instructions are used to directly access the data stored in any data memory sections.

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

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

Operation ACC ← ACC + [m] + C

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

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

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

Operation [m] ← ACC + [m] + C

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

LADD A,[m] Add Data Memory to ACC

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

Operation ACC ← ACC + [m]

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

LADDM A,[m] Add ACC to Data Memory

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

Operation [m] ← ACC + [m]

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

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

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

Operation ACC ← ACC "AND" [m]

Affected flag(s) Z

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

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

Operation [m] ← ACC "AND" [m]

Affected flag(s) Z

LCLR [m] Clear Data Memory

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

Operation [m] ← 00H

Affected flag(s) None

LCLR [m].i Clear bit of Data Memory

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

Operation [m].i ← 0

Affected flag(s) None

LCPL [m] Complement Data Memory

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

Operation [m] ← [m]

Affected flag(s) Z

LCPLA [m] Complement Data Memory with result in ACC

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

Operation ACC ← [m]

Affected flag(s) Z

LDAA [m] Decimal-Adjust ACC for addition with result in Data Memory

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

Operation [m] ← ACC + 00H or
[m] ← ACC + 06H or
[m] ← ACC + 60H or
[m] ← ACC + 66H

Affected flag(s) C

LDEC [m] Decrement Data Memory

Description Data in the specified Data Memory is decremented by 1.

Operation [m] ← [m] − 1

Affected flag(s) Z

LDECA [m] Decrement Data Memory with result in ACC

Description Data in the specified Data Memory is decremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC ← [m] − 1

Affected flag(s) Z

LINC [m] Increment Data Memory

Description Data in the specified Data Memory is incremented by 1.

Operation [m] ← [m] + 1

Affected flag(s) Z

LINCA [m] Increment Data Memory with result in ACC

Description Data in the specified Data Memory is incremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC ← [m] + 1

Affected flag(s) Z

LMOV A,[m] Move Data Memory to ACC

Description The contents of the specified Data Memory are copied to the Accumulator.

Operation ACC ← [m]

Affected flag(s) None

LMOV [m],A Move ACC to Data Memory

Description The contents of the Accumulator are copied to the specified Data Memory.

Operation [m] ← ACC

Affected flag(s) None

LOR A,[m] Logical OR Data Memory to ACC

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

Operation ACC ← ACC "OR" [m]

Affected flag(s) Z

LORM A,[m] Logical OR ACC to Data Memory

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

Operation [m] ← ACC "OR" [m]

Affected flag(s) Z

LRL [m] Rotate Data Memory left

Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0.

Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← [m].7

Affected flag(s) None

LRLA [m] Rotate Data Memory left with result in ACC

Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← [m].7

Affected flag(s) None

LRLC [m] Rotate Data Memory left through Carry

Description The contents of the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into bit 0.

Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← C
C ← [m].7

Affected flag(s) C

LRLCA [m] Rotate Data Memory left through Carry with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into the bit 0. The rotated result is stored in theAccumulator and the contents of the Data Memory remain unchanged.

Operation ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← C
C ← [m].7

Affected flag(s) C

LRR [m] Rotate Data Memory right

Description The contents of the specified Data Memory are rotated right by 1 bit with bit 0 rotated into bit 7.

Operation [m].i ← [m].(i+1); (i=0~6)
[m].7 ← [m].0

Affected flag(s) None

LRRA [m] Rotate Data Memory right with result in ACC

Description Data in the specified Data Memory is rotated right by 1 bit with bit 0 rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.i ← [m].(i+1); (i=0~6)
ACC.7 ← [m].0

Affected flag(s) None

LRRC [m] Rotate Data Memory right through Carry

Description The contents of the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7.

Operation [m].i ← [m].(i+1); (i=0~6)
[m].7 ← C
C ← [m].0

Affected flag(s) C

LRRCA [m] Rotate Data Memory right through Carry with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.i ← [m].(i+1); (i=0~6)
ACC.7 ← C
C ← [m].0

Affected flag(s) C

LSBC A,[m] Subtract Data Memory from ACC with Carry

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

Operation ACC ← ACC − [m] − C

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

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

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

Operation [m] ← ACC − [m] − C

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

LSDZ [m] Skip if decrement Data Memory is 0

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

Operation [m] ← [m] − 1
Skip if [m]=0

Affected flag(s) None

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

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

Operation ACC ← [m] − 1
Skip if ACC=0

Affected flag(s) None

LSET [m] Set Data Memory

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

Operation [m] ← FFH

Affected flag(s) None

LSET [m].i Set bit of Data Memory

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

Operation [m].i ← 1

Affected flag(s) None

LSIZ [m] Skip if increment Data Memory is 0

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

Operation [m] ← [m] + 1
Skip if [m]=0

Affected flag(s) None

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

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

Operation ACC ← [m] + 1
Skip if ACC=0

Affected flag(s) None

LSNZ [m].i Skip if 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 three cycle instruction. If the result is 0 the program proceeds with the following instruction.

Operation Skip if [m].i ≠ 0

Affected flag(s) None

LSNZ [m] Skip if Data Memory is not 0

Description The contents of the specified Data Memory are read out and then written to the specified Data Memory again. If the content of the specified Data Memory is not 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a three cycle instruction. If the result is 0 the program proceeds with the following instruction.

Operation Skip if [m] ≠ 0

Affected flag(s) None

LSUB A,[m] Subtract Data Memory from ACC

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

Operation ACC ← ACC − [m]

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

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

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

Operation [m] ← ACC − [m]

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

LSWAP [m] Swap nibbles of Data Memory

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

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s) None

LSWAPA [m] Swap nibbles of Data Memory with result in ACC

Description The low-order and high-order nibbles of the specified Data Memory are interchanged. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC.3~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0

Affected flag(s) None

LSZ [m] Skip if Data Memory is 0

Description The contents of the specified Data Memory are read out and then written to the specified Data Memory again. 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 three cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation Skip if [m]=0

Affected flag(s) None

LSZA [m] Skip if Data Memory is 0 with data movement to ACC

Description The contents of the specified Data Memory are copied to the Accumulator. If the value is zero, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a three cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation ACC ← [m]
Skip if [m]=0

Affected flag(s) None

LSZ [m].i Skip if bit i of Data Memory is 0

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

Operation Skip if [m].i=0

Affected flag(s) None

LTABRD [m] Read table (specific page) to TBLH and Data Memory

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

LITABRD [m] Increment table pointer low byte first and read table (specific page) to TBLH and Data Memory

Description Increment table pointer low byte, TBLP, first and then the program code (specific page) addressed by the table pointer (TBHP and TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)

TBLH ← program code (high byte)

Affected flag(s) None

LITABRDL [m] Increment table pointer low byte first and read table (last page) to TBLH and Data Memory

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

Operation [m] ← program code (low byte)

TBLH ← program code (high byte)

Affected flag(s) None

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

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

Operation ACC ← ACC "XOR" [m]

Affected flag(s) Z

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

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

Operation [m] ← ACC "XOR" [m]

Affected flag(s) Z

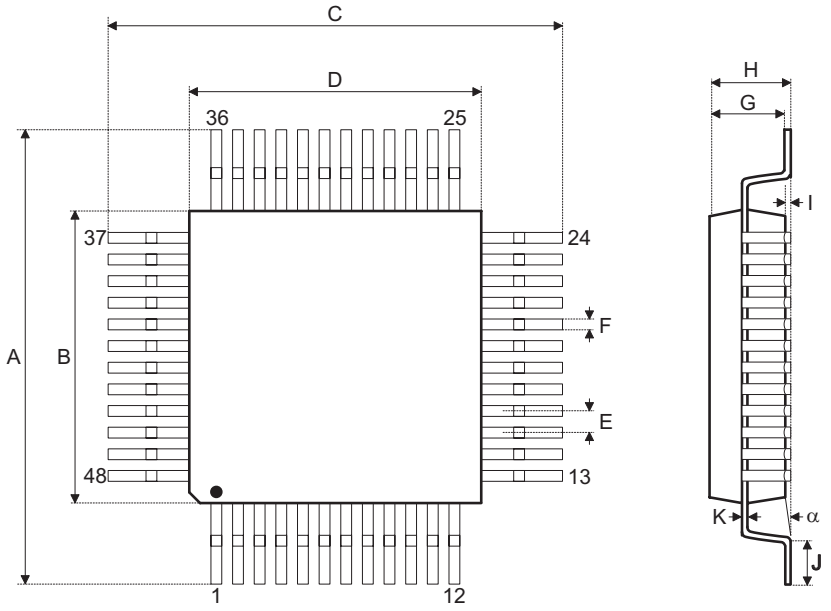
Package Information

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

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

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

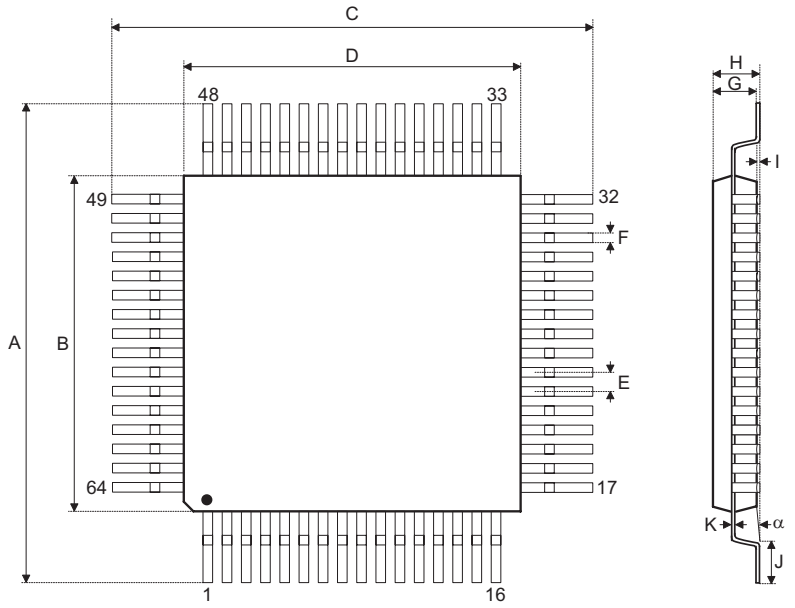
48-pin LQFP (7mm×7mm) Outline Dimensions



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

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

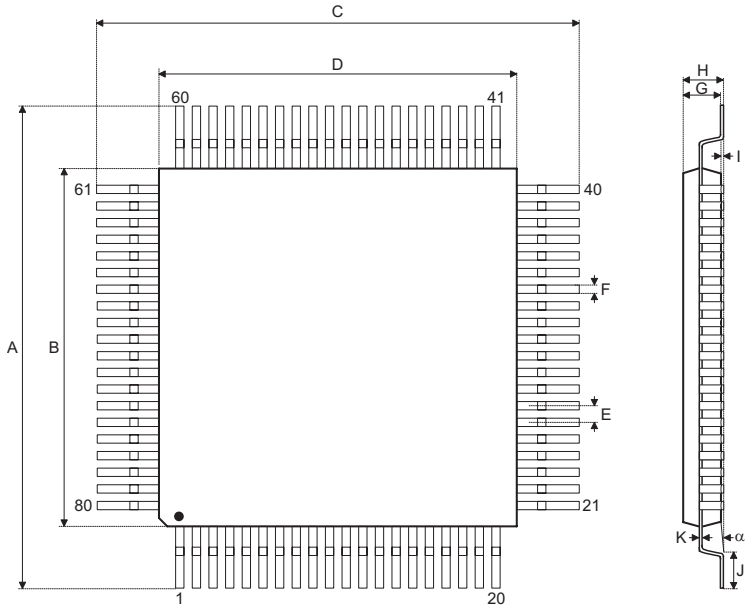
64-pin LQFP (7mm×7mm) Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A		0.354 BSC	
B		0.276 BSC	
C		0.354 BSC	
D		0.276 BSC	
E		0.016 BSC	
F	0.005	0.007	0.009
G	0.053	0.055	0.057
H	—	—	0.063
I	0.002	—	0.006
J	0.018	0.024	0.030
K	0.004	—	0.008
α	0°	—	7°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A		9.00 BSC	
B		7.00 BSC	
C		9.00 BSC	
D		7.00 BSC	
E		0.40 BSC	
F	0.13	0.18	0.23
G	1.35	1.40	1.45
H	—	—	1.60
I	0.05	—	0.15
J	0.45	0.60	0.75
K	0.09	—	0.20
α	0°	—	7°

80-pin LQFP (10mm×10mm) Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A		0.472 BSC	
B		0.394 BSC	
C		0.472 BSC	
D		0.394 BSC	
E		0.016 BSC	
F	0.005	0.007	0.009
G	0.053	0.055	0.057
H	—	—	0.063
I	0.002	—	0.006
J	0.018	0.024	0.030
K	0.004	—	0.008
α	0°	—	7°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A		12.00 BSC	
B		10.00 BSC	
C		12.00 BSC	
D		10.00 BSC	
E		0.40 BSC	
F	0.13	0.18	0.23
G	1.35	1.40	1.45
H	—	—	1.60
I	0.05	—	0.15
J	0.45	0.60	0.75
K	0.09	—	0.20
α	0°	—	7°