Enhanced A/D Flash Type 8-Bit MCU with EEPROM

HT66F60A
HT66F70A

Revision: V1.00 Date: March 20, 2013

www.holtek.com

Table of Contents

Features

CPU Features

- Operating Voltage:
  - f_{SYS}=8MHz: 2.2V~5.5V
  - f_{SYS}=12MHz: 2.7V~5.5V
  - f_{SYS}=16MHz: 4.5V~5.5V
- Up to 0.25μs instruction cycle with 16MHz system clock at V_{DD}=5V
- Power down and wake-up functions to reduce power consumption
- Five oscillators:
  - External Crystal -- HXT
  - External 32.768kHz Crystal -- LXT
  - External RC -- ERC
  - Internal RC -- HIRC
  - Internal 32kHz RC -- LIRC
- Multi-mode operation: NORMAL, SLOW, IDLE and SLEEP
- Fully integrated internal 8MHz ocilllator requires no external components
- All instructions executed in 1~3 instruction cycles
- Table read instructions
- 114 powerful instructions
- Up to 16-level subroutine nesting
- Bit manipulation instruction

Peripheral Features

- Flash Program Memory: 16k×16~32k×16
- Data Memory: 1024×8~2048×8
- EEPROM Memory: 128×8
- In Application Programming function
- Watchdog Timer function
- Up to 61 bidirectional I/O lines
- Software controlled 4-SCOM lines LCD driver with 1/2 bias
- 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 – SIM for SPI or I^2C
- Single sefial SPI interface – SPIA
- Dual Comparator functions
- 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
- Wide range of available package types
- Flash program memory can be re-programmed up to 100,000 times
- Flash program memory data retention > 10 years
- EEPROM data memory can be re-programmed up to 1,000,000 times
- EEPROM data memory data retention > 10 years

General Description

The HT66Fx0A series of devices are Flash Memory A/D type 8-bit high performance RISC architecture microcontrollers, designed for a wide range of applications. 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 an area of EEPROM memory for storage of non-volatile data such as serial numbers, calibration data etc.

Analog features include a multi-channel 12-bit A/D converter and dual comparator functions. 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.

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

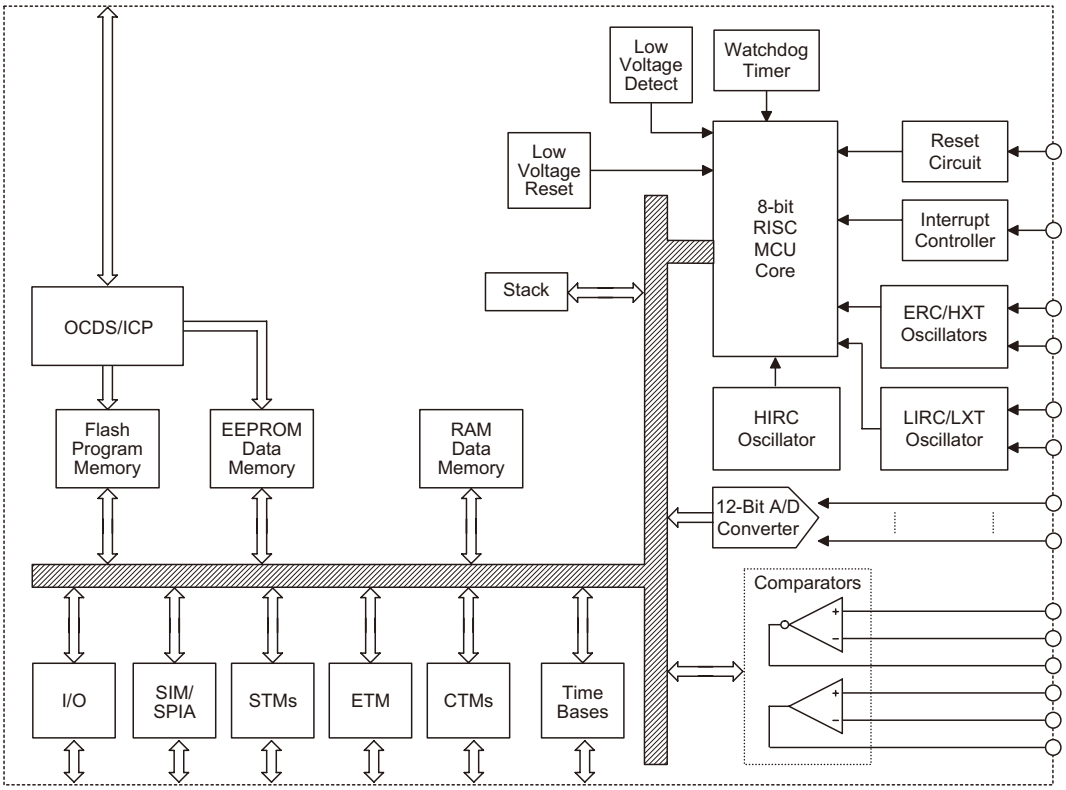
Selection Table

Most features are common to all devices. The main features distinguishing them are Program Memory and Data Memory capacity. The following table summarises the main features of each device.

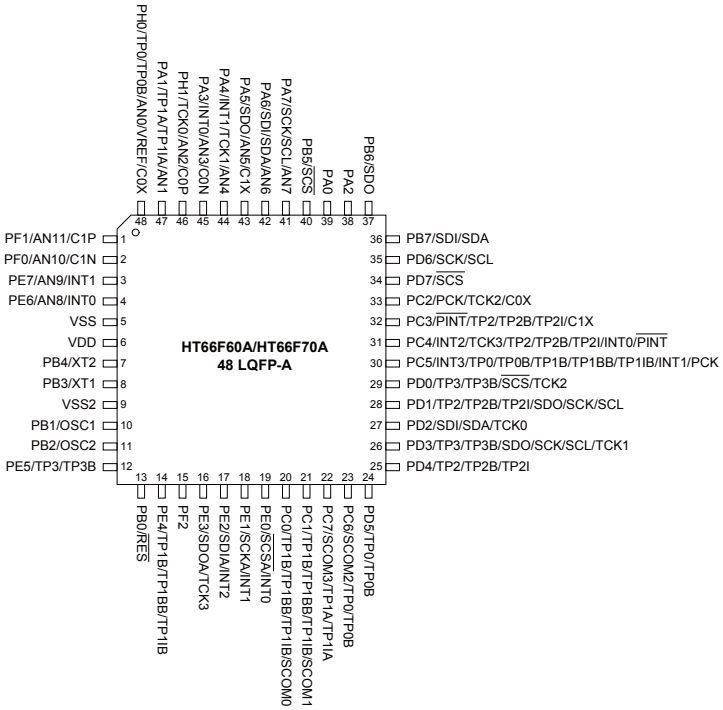
Part No.	Program Memory	Data Memory	Data EEPROM	I/O	External Interrupt	A/D Converter	Timer Module	SIM	SPIA	Time Base	Comparators	Stacks	package
HT66F60A	16k × 16	1024 × 8	128 × 8	61	4	12-bit × 12	10-bit CTM × 2 16-bit STM × 3 10-bit ETM × 1	√	√	2	2	16	48/64 LQFP
HT66F70A	32k × 16	2048 × 8	128 × 8	61	4	12-bit × 12	10-bit CTM × 2 16-bit STM × 3 10-bit ETM × 1	√	√	2	2	16	48/64 LQFP

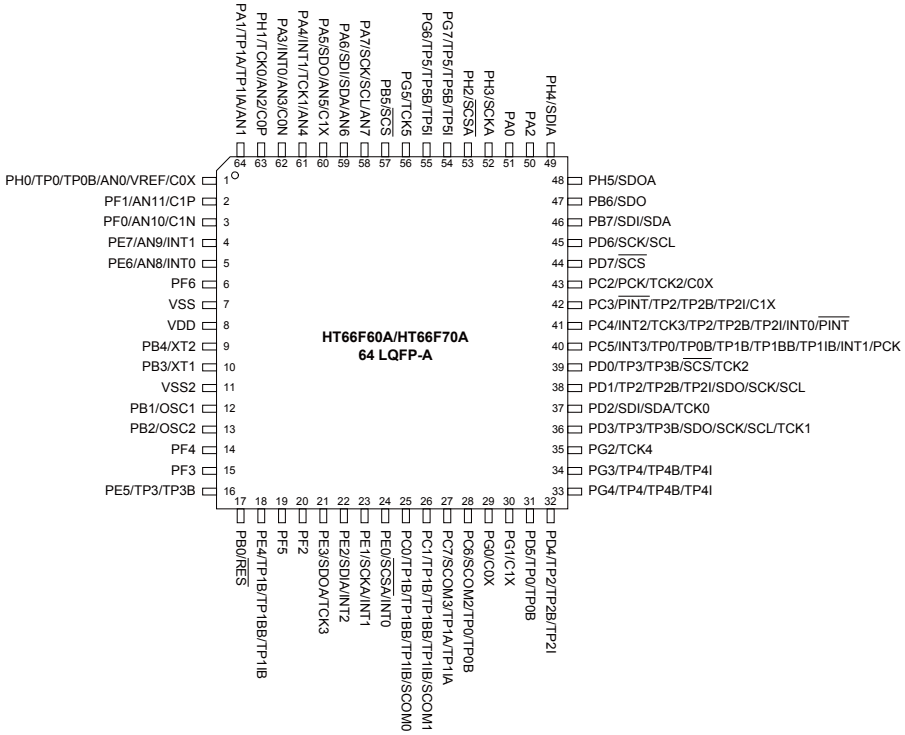
Note: As devices exist in more than one package format, the table reflects the situation for the package with the most pins.

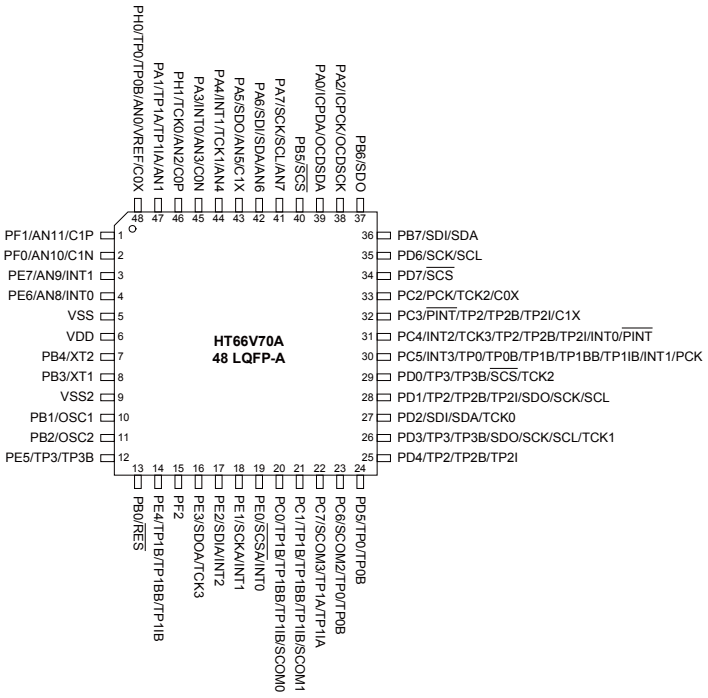
Block Diagram

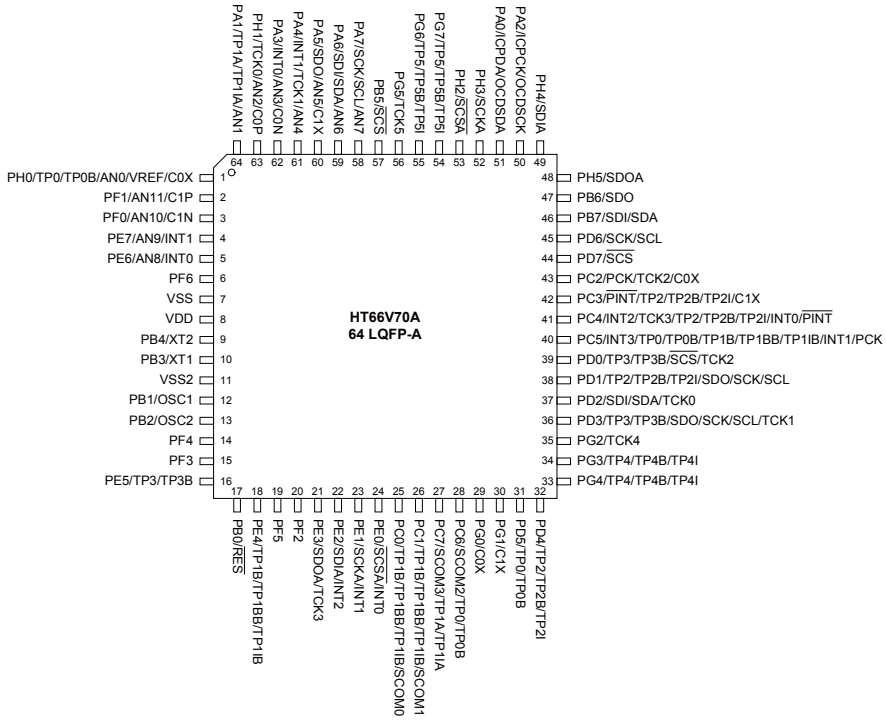


Pin Assignment









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

2. The HT66Vx0A device is the EV chip of the HT66Fx0A series of devices. It supports the "On-Chip Debug" function for debugging during development using the OCDSDA and OCDSCK pins connected to the Holtek HT-IDE development tools.

Pin Description

Pad Name	Function	OPT	I/T	O/T	Description
PA0/ICPDA/ OCDSDA	PA0	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	ICPDA	—	ST	CMOS	ICP Data/Address
	OCDSDA	—	ST	CMOS	OCDS Data/Address, for EV chip only
PA1/TP1A/ TP1IA/AN1	PA1	PAWU PAPU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	TP1A	PAS0	—	CMOS	TM1 A output
	TP1IA	IFS2	ST	—	TM1 A input
	AN1	PAS0	AN	—	A/D Converter analog input
PA2/ICPCK/ OCDSCK	PA2	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	ICPCK	—	ST	CMOS	ICP Clock pin
	OCDSCK	—	ST	—	OCDS Clock pin, for EV chip only
PA3/INT0/ AN3/C0N	PA3	PAWU PAPU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	INT0	INTEG INTC0 IFS0	ST	—	External Interrupt 0
	AN3	PAS1	AN	—	A/D Converter analog input
	C0N	PAS1	AN	—	Comparator 0 inverting input
PA4/INT1/ TCK1/AN4	PA4	PAPU PAWU PAS2	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	INT1	INTEG INTC0 IFS0	ST	—	External Interrupt 1
	TCK1	IFS1	ST	—	TM1 input
	AN4	PAS1	AN	—	A/D Converter analog input
PA5/SDO/ AN5/C1X	PA5	PAWU PAPU PAS2	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	SDO	PAS2	—	CMOS	SPI data output
	AN5	PAS2	AN	—	A/D Converter analog input
	C1X	PAS2	—	CMOS	Comparator 1 output
PA6/SDI/ SDA/AN6	PA6	PAWU PAPU PAS3	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	SDI	PAS3 IFS4	ST	—	SPI data input
	SDA	PAS3 IFS4	ST	NMOS	I^2C data line
	AN6	PAS3	AN	—	A/D Converter analog input
PA7/SCK/ SCL/AN7	PA7	PAWU PAPU PAS3	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	SCK	PAS3 IFS4	ST	CMOS	SPI serial clock
	SCL	PAS3 IFS4	ST	NMOS	I^2C clock line
	AN7	PAS3	AN	—	A/D Converter analog input

Pad Name	Function	OPT	I/T	O/T	Description
PB0/$\overline{RES}$	PB0	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	$\overline{RES}$	CO	ST	—	Reset pin
PB1/OSC1	PB1	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	OSC1	CO	HXT	—	HXT/ERC oscillator pin & EC mode input pin
PB2/OSC2	PB2	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	OSC2	CO	—	HXT	HXT oscillator pin
PB3/XT1	PB3	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	XT1	CO	LXT	—	LXT oscillator pin
PB4/XT2	PB4	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	XT2	CO	—	LXT	LXT oscillator pin
PB5/$\overline{SCS}$	PB5	PBPU PBS2	ST	CMOS	General purpose I/O. Register enabled pull-up
	$\overline{SCS}$	PBS2 IFS4	ST	CMOS	SPI slave select
PB6/SDO	PB6	PBPU PBS3	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	SDO	PBS3	—	CMOS	SPI data output
PB7/SDI/SDA	PB7	PBPU PBS3	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	SDI	PBS3 IFS4	ST	—	SPI data input
	SDA	PBS3 IFS4	ST	NMOS	I^2C data line
PC0/TP1B/ TP1BB/TP1IB/ SCOM0	PC0	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	TP1B	PCS0	—	CMOS	TM1 B output
	TP1BB	PCS0	—	CMOS	TM1 inverted B output
	TP1IB	IFS2	ST	—	TM1 B input
	SCOM0	PCS0	—	SCOM	LCD common output
PC1/TP1B/ TP1BB/TP1IB/ SCOM1	PC1	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	TP1B	PCS0	—	CMOS	TM1 B output
	TP1BB	PCS0	—	CMOS	TM1 inverted B output
	TP1IB	IFS2	ST	—	TM1 B input
	SCOM1	PCS0	—	SCOM	LCD common output
PC2/PCK/ TCK2/C0X	PC2	PCPU PCS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	PCK	PCS1	—	CMOS	Peripheral clock output
	TCK2	IFS1	ST	—	TM2 input
	C0X	PCS1	—	CMOS	Comparator 0 output
PC3/$\overline{PINT}$/TP2/ TP2B/TP2I/C1X	PC3	PCPU PCS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	$\overline{PINT}$	IFS0	ST	—	Peripheral interrupt
	TP2	PCS1	—	CMOS	TM2 output
	TP2B	PCS1	—	CMOS	TM2 inverted output
	TP2I	IFS2	ST	—	TM2 input
	C1X	PCS1	—	CMOS	Comparator 1 output

Pad Name	Function	OPT	I/T	O/T	Description
PC4/INT2/TCK3/ TP2/TP2B/TP2I/ INT0/$\overline{\text{PINT}}$	PC4	PCPU PCS2	ST	CMOS	General purpose I/O. Register enabled pull-up
	INT2	INTEG INTC3 IFS0	ST	—	External Interrupt 2
	TCK3	IFS1	ST	—	TM3 input
	TP2	PCS1	—	CMOS	TM2 output
	TP2B	PCS1	—	CMOS	TM2 inverted output
	TP2I	IFS2	ST	—	TM2 input
	INT0	INTEG INTC0 IFS0	ST	—	External Interrupt 0
	$\overline{\text{PINT}}$	IFS0	ST	—	Peripheral interrupt
PC5/INT3/TP0/ TP0B/TP1B/ TP1BB/TP1IB/ INT1/PCK	PC5	PCPU PCS2	ST	CMOS	General purpose I/O. Register enabled pull-up
	INT3	INTEG INTC3	ST	—	External Interrupt 3
	TP0	PCS2	—	CMOS	TM0 output
	TP0B	PCS2	—	CMOS	TM0 inverted output
	TP1B	PCS2	—	CMOS	TM1 B output
	TP1BB	PCS2	—	CMOS	TM1 inverted B output
	TP1IB	IFS2	ST	—	TM1 B input
	INT1	INTEG INTC0 IFS0	ST	—	External Interrupt 1
	PCK	PCS2	—	CMOS	Peripheral clock output
PC6/SCOM2/ TP0/TP0B	PC6	PCPU PCS3	ST	CMOS	General purpose I/O. Register enabled pull-up
	SCOM2	PCS3	—	SCOM	LCD common output
	TP0	PCS3	—	CMOS	TM0 output
	TP0B	PCS3	—	CMOS	TM0 inverted output
PC7/SCOM3/ TP1A/TP1IA	PC7	PCPU PCS3	ST	CMOS	General purpose I/O. Register enabled pull-up
	SCOM3	PCS3	—	SCOM	LCD common output
	TP1A	PCS3	—	CMOS	TM1 A output
	TP1IA	IFS2	ST	—	TM1 A input
PD0/TP3/TP3B/ $\overline{\text{SCS}}$/TCK2	PD0	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	TP3	PDS0	—	CMOS	TM3 output
	TP3B	PDS0	—	CMOS	TM3 inverted output
	$\overline{\text{SCS}}$	PDS0 IFS4	ST	CMOS	SPI slave select
	TCK2	IFS1	ST	—	TM2 input

Pad Name	Function	OPT	I/T	O/T	Description
PD1/TP2/TP2B/TP2I/SDO/SCK/SCL	PD1	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	TP2	PDS0	—	CMOS	TM2 output
	TP2B	PDS0	—	CMOS	TM2 inverted output
	TP2I	IFS2	ST	—	TM2 input
	SDO	PDS0	—	CMOS	SPI slave select
	SCK	PDS0 IFS4	ST	CMOS	SPI serial clock
	SCL	PDS0 IFS4	ST	NMOS	I²C clock line
PD2/SDI/SDA/TCK0	PD2	PDPU PDS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	SDI	PDS1 IFS4	ST	—	SPI data input
	SDA	PDS1 IFS4	ST	NMOS	I²C data line
	TCK0	IFS1	ST	—	TM0 input
PD3/TP3/TP3B/SDO/SCK/SCL/TCK1	PD3	PDPU PDS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	TP3	PDS1	—	CMOS	TM3 output
	TP3B	PDS1	—	CMOS	TM2 inverted output
	SDO	PDS1	—	CMOS	SPI slave select
	SCK	PDS1 IFS4	ST	CMOS	SPI serial clock
	SCL	PDS1 IFS4	ST	NMOS	I²C clock line
	TCK1	IFS1	ST	—	TM1 input
PD4/TP2/TP2B/TP2I	PD4	PDPU PDS2	ST	CMOS	General purpose I/O. Register enabled pull-up
	TP2	PDS2	—	CMOS	TM2 output
	TP2B	PDS2	—	CMOS	TM2 inverted output
	TP2I	IFS2	ST	—	TM2 input
PD5/TP0/TP0B	PD5	PDPU PDS2	ST	CMOS	General purpose I/O. Register enabled pull-up
	TP0	PDS2	—	CMOS	TM0 output
	TP0B	PDS2	—	CMOS	TM0 inverted output
PD6/SCK/SCL	PD6	PDPU PDS3	ST	CMOS	General purpose I/O. Register enabled pull-up
	SCK	PDS3 IFS4	ST	CMOS	SPI serial clock
	SCL	PDS3 IFS4	ST	NMOS	I²C clock line
PD7/$\overline{SCS}$	PD7	PDPU PDS3	ST	CMOS	General purpose I/O. Register enabled pull-up
	$\overline{SCS}$	PDS3 IFS4	ST	CMOS	SPI slave select
PE0/$\overline{SCSA}$/INT0	PE0	PEPU PES0	ST	CMOS	General purpose I/O. Register enabled pull-up
	$\overline{SCSA}$	PES0 IFS5	ST	CMOS	SPIA slave select
	INT0	INTEG INTC0 IFS0	ST	—	External Interrupt 0

Pad Name	Function	OPT	I/T	O/T	Description
PE1/SCKA/INT1	PE1	PEPU PES0	ST	CMOS	General purpose I/O. Register enabled pull-up
	SCKA	PES0 IFS5	ST	CMOS	SPIA serial clock
	INT1	INTEG INTC0 IFS0	ST	—	External Interrupt 1
PE2/SDIA/INT2	PE2	PEPU PES1	ST	CMOS	General purpose I/O. Register enabled pull-up
	SDIA	IFS5	ST	CMOS	SPI serial clock
	INT2	INTEG INTC3 IFS0	ST	—	External Interrupt 2
PE3/SDOA/TCK3	PE2	PEPU PES1	ST	CMOS	General purpose I/O. Register enabled pull-up
	SDOA	PES1	ST	CMOS	SPIA serial clock
	TCK3	IFS1	ST	—	TM3 input
PE4/TP1B/ TP1BB/TP1IB	PE4	PEPU PES2	ST	CMOS	General purpose I/O. Register enabled pull-up
	TP1B	PES2	—	CMOS	TM1 B output
	TP1BB	PES2	—	CMOS	TM1 inverted B output
	TP1IB	IFS2	ST	—	TM1 B input
PE5/TP3/TP3B	PE5	PEPU PES2	ST	CMOS	General purpose I/O. Register enabled pull-up
	TP3	PES2	—	CMOS	TM3 output
	TP3B	PES2	—	CMOS	TM3 inverted output
PE6/AN8/INT0	PE6	PEPU PES3	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN8	PES3	AN	—	A/D Converter analog input
	INT0	INTEG INTC0 IFS0	ST	—	External Interrupt 0
PE7/AN9/INT1	PE7	PEPU PES3	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN9	PES3	AN	—	A/D Converter analog input
	INT1	INTEG INTC0 IFS0	ST	—	External Interrupt 1
PF0/AN10/C1N	PF0	PFPU PFS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN10	PFS0	AN	—	A/D Converter analog input
	C1N	PFS0	AN	—	Comparator 1 interting input
PF1/AN11/C1P	PF1	PFPU PFS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN11	PFS0	AN	—	A/D Converter analog input
	C1P	PFS0	AN	—	Comparator 1 non-interting input
PF2~PF6	PFn	PFPU	ST	CMOS	General purpose I/O. Register enabled pull-up
PG0/C0X	PG0	PGPU PGS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	C0X	PGS0	—	CMOS	Comparator 0 output
PG1/C1X	PG1	PGPU PGS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	C1X	PGS0	—	CMOS	Comparator 1 output

Pad Name	Function	OPT	I/T	O/T	Description
PG2/TCK4	PG2	PGPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	TCK4	—	ST	—	TM4 input
PG3/TP4/ TP4B/TP4I	PG3	PGPU PGS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	TP4	PGS1	—	CMOS	TM4 output
	TP4B	PGS1	—	CMOS	TM4 inverted output
	TP4I	IFS3	ST	—	TM4 input
PG4/TP4/ TP4B/TP4I	PG4	PGPU PGS2	ST	CMOS	General purpose I/O. Register enabled pull-up
	TP4	PGS2	—	CMOS	TM4 output
	TP4B	PGS2	—	CMOS	TM4 inverted output
	TP4I	IFS3	ST	—	TM4 input
PG5/TCK5	PG5	PGPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	TCK5	—	ST	—	TM5 input
PG6/TP5/ TP5B/TP5I	PG6	PGPU PGS3	ST	CMOS	General purpose I/O. Register enabled pull-up
	TP5	PGS3	—	CMOS	TM5 output
	TP5B	PGS3	—	CMOS	TM5 inverted output
	TP5I	IFS3	ST	—	TM5 input
PG7/TP5/ TP5B/TP5I	PG7	PGPU PGS3	ST	CMOS	General purpose I/O. Register enabled pull-up
	TP5	PGS3	—	CMOS	TM5 output
	TP5B	PGS3	—	CMOS	TM5 inverted output
	TP5I	IFS3	ST	—	TM5 input
PH0/TP0/TP0B/ AN0/VREF/C0X	PH0	PHPU PHS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	TP0	PHS0	—	CMOS	TM0 output
	TP0B	PHS0	—	CMOS	TM0 inverted output
	AN0	PHS0	AN	—	A/D Converter analog input
	VREF	PHS0	AN	—	A/D Converter reference input
	C0X	PHS0	—	CMOS	Comparator 0 output
PH1/TCK0/ AN2/C0P	PH0	PHPU PHS0	ST	CMOS	General purpose I/O. Register enabled pull-up
	TCK0	IFS1	ST	—	TM0 input
	AN2	PHS0	AN	—	A/D Converter analog input
	C0P	PHS0	AN	—	Comparator 0 non-inverting input
PH2/$\overline{\text{SCSA}}$	PH2	PHPU PHS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	$\overline{\text{SCSA}}$	PHS1 IFS5	ST	CMOS	SPIA slave select
PH3/SCKA	PH3	PHPU PHS1	ST	CMOS	General purpose I/O. Register enabled pull-up
	SCKA	PHS1 IFS5	ST	CMOS	SPIA serial clock
PH4/SDIA	PH4	PHPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	SDIA	IFS5	ST	CMOS	SPIA serial data input
PH5/SDOA	PH5	PHPU PHS2	ST	CMOS	General purpose I/O. Register enabled pull-up
	SDOA	PHS2	ST	CMOS	SPIA serial data output
VDD	VDD	—	PWR	—	Positive Power supply
VSS	VSS	—	PWR	—	Negative Power supply. Ground

Pad Name	Function	OPT	I/T	O/T	Description
VSS2	VSS2	—	PWR	—	I/O Pad Power supply. Ground

Note: I/T: Input type; O/T: Output type
OPT: Optional by configuration option (CO) or register option
PWR: Power; CO: Configuration option
ST: Schmitt Trigger input;
CMOS: CMOS output; NMOS: NMOS output
HXT: High frequency crystal oscillator
LXT: Low frequency crystal oscillator

Absolute Maximum Ratings

Supply Voltage ..... V_{SS}−0.3V to V_{SS}+6.0V
Input Voltage ..... V_{SS}−0.3V to V_{DD}+0.3V
Storage Temperature..... -50˚C to 125˚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)	—	f_{SYS}=8MHz	2.2	—	5.5	V
			f_{SYS}=12MHz	2.7	—	5.5	V
			f_{SYS}=16MHz	4.5	—	5.5	V
V_{DD2}	Operating Voltage (ERC)	—	f_{SYS}=6MHz	2.2	—	5.5	V
			f_{SYS}=8MHz	2.7	—	5.5	V
			f_{SYS}=12MHz	4.5	—	5.5	V
V_{DD3}	Operating Voltage (HIRC)	—	f_{SYS}=4/8 MHz	2.2	—	5.5	V
I_{DD1}	Operating Current (HXT, f_{SYS}=f_H, f_S=f_{SUB}=f_{RTC} or f_{LIRC})	3V	No load, f_H=8MHz, ADC off, WDT enable	—	1.0	1.5	mA
		5V		—	2.5	4.0	mA
		3V	No load, f_H=10MHz, ADC off, WDT enable	—	1.2	2.0	mA
		5V		—	2.8	4.5	mA
		3V	No load, f_H=12MHz, ADC off, WDT enable	—	1.5	2.5	mA
		5V		—	3.5	5.5	mA
		5V	No load, f_H=16MHz, ADC off, WDT enable	—	4.5	7.0	mA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{DD2}	Operating Current (ERC, $f_{SYS}=f_H$, $f_S=f_{SUB}=f_{RTC}$ or f_{LIRC})	3V	No load, f_H=6MHz, ADC off, WDT enable	—	0.9	1.5	mA
		5V		—	2.0	3.0	mA
		3V	No load, f_H=8MHz, ADC off, WDT enable	—	1.2	2.0	mA
		5V		—	2.8	4.5	mA
		5V	No load, f_H=12MHz, ADC off, WDT enable	—	4.0	6.0	mA
I_{DD3}	Operating Current (HIRC OSC, $f_{SYS}=f_H$, $f_S=f_{SUB}=f_{RTC}$ or f_{LIRC})	3V	No load, f_H=4MHz, ADC off, WDT enable	—	0.7	1.2	mA
		5V		—	1.5	2.5	mA
		3V	No load, f_H=8MHz, ADC off, WDT enable	—	1.2	2.0	mA
		5V		—	2.8	4.5	mA
I_{DD4}	Operating Current (HXT, $f_{SYS}=f_L$, $f_S=f_{SUB}=f_{RTC}$ or f_{LIRC})	3V	No load, f_H=12MHz, $f_L=f_H/2$, ADC off, WDT enable	—	0.9	1.5	mA
		5V		—	2.1	3.3	mA
		3V	No load, f_H=12MHz, $f_L=f_H/4$, ADC off, WDT enable	—	0.6	1.0	mA
		5V		—	1.6	2.5	mA
		3V	No load, f_H=12MHz, $f_L=f_H/8$, ADC off, WDT enable	—	0.48	0.8	mA
		5V		—	1.2	2.0	mA
		3V	No load, f_H=12MHz, $f_L=f_H/16$, ADC off, WDT enable	—	0.42	0.7	mA
		5V		—	1.1	1.7	mA
		3V	No load, f_H=12MHz, $f_L=f_H/32$, ADC off, WDT enable	—	0.38	0.6	mA
		5V		—	1.0	1.5	mA
		3V	No load, f_H=12MHz, $f_L=f_H/64$, ADC off, WDT enable	—	0.36	0.55	mA
		5V		—	1.0	1.5	mA
I_{DD5}	Operating Current (LXT, $f_{SYS}=f_L=f_{RTC}$, $f_S=f_{SUB}=f_{RTC}$)	3V	No load, ADC off, WDT enable, LXTLP=0, LVD&LVR disable	—	10	20	μA
		5V		—	30	50	μA
		3V	No load, ADC off, WDT enable, LXTLP=1, LVD & LVR disable	—	10	20	μA
		5V		—	30	50	μA
I_{DD6}	Operating Current (LIRC, $f_{SYS}=f_L=f_{LIRC}$, $f_S=f_{SUB}=f_{LIRC}$)	3V	No load, ADC off, WDT enable, LVD&LVR disable	—	10	20	μA
		5V		—	30	50	μA
I_{STB1}	Standby Current (IDLE1) (HXT, $f_{SYS}=f_H$, $f_S=f_{SUB}=f_{RTC}$ or f_{LIRC})	3V	No load, system HALT, ADC off, WDT enable, f_{SYS}=12MHz	—	0.6	1.0	mA
		5V		—	1.2	2.0	mA
I_{STB2}	Standby Current (IDLE0) (HXT, f_{SYS}=off, $f_S=f_{SUB}=f_{RTC}$ or f_{LIRC})	3V	No load, system HALT, ADC off, WDT enable, f_{SYS}=12MHz	—	1.3	3.0	μA
		5V		—	2.2	5.0	μA
I_{STB3}	Standby Current (IDLE0) (ERC, f_{SYS}=off, $f_S=f_{SUB}=f_{RTC}$)	3V	No load, system HALT, ADC off, WDT enable, f_{SYS}=12MHz	—	1.3	3.0	μA
		5V		—	2.2	5.0	μA
I_{STB4}	Standby Current (IDLE0) (HIRC, f_{SYS}=off, $f_S=f_{SUB}=f_{LIRC}$)	3V	No load, system HALT, ADC off, WDT enable, f_{SYS}=8MHz	—	1.3	3.0	μA
		5V		—	2.2	5.0	μA
I_{STB5}	Standby Current (IDLE1) (HXT, $f_{SYS}=f_L$, $f_S=f_{SUB}=f_{RTC}$ or f_{LIRC})	3V	No load, system HALT, ADC off, WDT enable, f_{SYS}=12MHz/64	—	0.34	0.6	mA
		5V		—	0.85	1.2	mA
I_{STB6}	Standby Current (IDLE0) (HXT, f_{SYS}=off, $f_S=f_{SUB}=f_{RTC}$ or f_{LIRC})	3V	No load, system HALT, ADC off, WDT enable, f_{SYS}=12MHz/64	—	1.3	3.0	μA
		5V		—	2.2	5.0	μA
I_{STB7}	Standby Current (IDLE1) (LXT, $f_{SYS}=f_L=f_{RTC}$, $f_S=f_{SUB}=f_{RTC}$)	3V	No load, system HALT, ADC off, WDT enable, f_{SYS}=32768Hz, LXTLP=1	—	1.9	4.0	μA
		5V		—	3.3	7.0	μA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{STB8}	Standby Current (IDLE0) (LXT, f_{SYS}=off, f_S=f_{SUB}=f_{RTC})	3V	No load, system HALT, ADC off, WDT enable, f_{SYS}=32768Hz, LXTLP=1	—	1.3	3.0	μA
		5V		—	2.2	5.0	μA
I_{STB9}	Standby Current (IDLE0) (LIRC, f_{SYS}=off, f_S=f_{SUB}=f_{LIRC})	3V	No load, system HALT, ADC off, WDT enable, f_{SYS}=32kHz	—	1.3	3.0	μA
		5V		—	2.2	5.0	μA
I_{STB10}	Standby Current (SLEEP0) (HXT, f_{SYS}=off, f_S=f_{SUB}=f_{RTC} or f_{LIRC})	3V	No load, system HALT, ADC off, WDT disable, f_{SYS}=12MHz	—	0.1	1	μA
		5V		—	0.3	2	μA
I_{STB11}	Standby Current (SLEEP1) (HXT, f_{SYS}=off, f_S=f_{SUB}=f_{RTC})	3V	No load, system HALT, ADC off, WDT enable, f_{SYS}=12MHz	—	1.3	5.0	μA
		5V		—	2.2	10.0	μA
I_{STB12}	Standby Current (SLEEP1) (HXT, f_{SYS}=off, f_S=f_{SUB}=f_{LIRC})	3V	No load, system HALT, ADC off, WDT enable, f_{SYS}=12MHz	—	1.3	5.0	μA
		5V		—	2.2	10.0	μA
I_{STB13}	Standby Current (SLEEP0) (LXT, f_{SYS}=off, f_S=f_{SUB}=f_{LIRC} or f_{RTC})	3V	No load, system HALT, ADC off, WDT disable, f_{SYS}=32768Hz	—	0.1	1	μA
		5V		—	0.3	2	μA
I_{STB14}	Standby Current (SLEEP1) (LXT, f_{SYS}=off, f_S=f_{SUB}=f_{RTC})	3V	No load, system HALT, ADC off, WDT enable, f_{SYS}=32768Hz	—	1.3	5.0	μA
		5V		—	2.2	10.0	μA
I_{STB15}	Stanby Current (SLEEP) (HXT, f_{SYS}=off, f_S=f_{SUB}=f_{RTC} or f_{LIRC})	—	No load, system HALT, ADC off, WDT disable, f_{SYS}=12MHz, LVR enable and LVDEN=1	—	60	90	μA
V_{IL1}	Input Low Voltage for I/O port except $\overline{RES}$ pin	5	—	0	—	1.5	V
		—		0	—	0.2V_{DD}	V
V_{IH1}	Input High Voltage for I/O port except $\overline{RES}$ pin	5	—	3.5	—	5	V
		—		0.8V_{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_{SCOM}	V_{DD}/2 voltage for LCD COMn	2.5V~5.5V	No load	0.475	0.500	0.525	V_{DD}
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_{SYS1}	System clock (HXT)	—	2.2~5.5V	0.4	—	8	MHz
			2.7~5.5V	0.4	—	12	MHz
			4.5~5.5V	0.4	—	16	MHz
f_{SYS2}	System clock (ERC)	5V	Ta=25°C, External R_{ERC}=120kΩ	-2%	8	+2%	MHz
f_{SYS3}	System clock (HIRC)	5V	Ta=25°C	-2%	8	+2%	MHz
f_{SYS4}	System Clock (LXT)	—	—	—	32768	—	Hz
f_{SYS5}	System Clock (LIRC)	5V	Ta=25°C	-3%	32	+3%	kHz
t_{TIMER}	TCKn and timer capture Input Pulse Width	—	—	0.3	—	—	μs
t_{RES}	External Reset Low Pulse Width	—	—	10	—	—	μs
t_{INT}	Interrupt Pulse Width	—	—	10	—	—	μs
t_{SST}	System Start-up Timer Period (Wake-up from HALT, f_{SYS} off at HALT state, Slow Mode → Normal Mode, Normal Mode → Slow Mode)	—	f_{SYS}=HXT or LXT (Slow Mode→Normal Mode(HXT), Normal Mode→Slow Mode(LXT))	1024	—	—	t_{SYS}
			f_{SYS}=HXT or LXT (Wake-up from HALT, f_{SYS} off at HALT state)	1024	—	—	t_{SYS}
		—	f_{SYS}=ERC or HIRC	16	—	—	t_{SYS}
		—	f_{SYS}=LIRC	2	—	—	t_{SYS}
	System Start-up Timer Period (Wake-up from HALT, f_{SYS} on at HALT state)	—	—	2	—	—	t_{SYS}
	System Start-up Timer Period (Reset)	—	—	1024	—	—	t_{SYS}
t_{RSTD}	System Reset Delay Time (Power On Reset, LVR reset, LVRC software reset, WDTC software reset)	—	—	25	50	100	ms
	System Reset Delay Time ($\overline{RES}$ reset, WDT normal reset)	—	—	8.3	16.7	33.3	ms
t_{EERD}	EEPROM Read Time	—	—	1	2	4	t_{SYS}
t_{EEWR}	EEPROM Write Timet	—	—	1	2	4	ms

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

A/D Converter Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Condition				
V_{ADI}	A/D Converter Input Voltage	—	—	0	—	V_{REF}	V
V_{REF}	A/D Converter Reference Voltage	—	—	2	—	AV_{DD}	V
DNL	Differential non-linearity	5V	t_{ADCK}=0.5μs	-3	—	+3	LSB
INL	Integral non-linearity	5V	t_{ADCK}=0.5μs	-4	—	+4	LSB
I_{ADC}	Additional Power Consumption if A/D Converter is used	3V	No load (t_{ADCK}=0.5μs)	—	1.0	2.0	mA
		5V	No load (t_{ADCK}=0.5μs)	—	1.5	3.0	mA
t_{ADCK}	A/D Converter Clock Period	—	—	0.5	—	100	μs
t_{ADC}	A/D Conversion Time (Include Sample and Hold Time)	—	12 bit A/D Converter	—	16	—	t_{ADCK}
t_{ADS}	A/D Converter Sampling Time	—	—	—	4	—	t_{ADCK}
t_{ON2ST}	A/D Converter On-to-Start Time	—	—	2	—	—	μs

LVD & LVR Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
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_{LVR3}			LVR Enable, 3.15V option		3.15		
V_{LVR4}			LVR Enable, 3.8V option		3.8		
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
V_{BG}	Bandgap reference with buffer voltage	—	—	-3%	1.25	+3%	V
I_{BG}	Additional Power Consumption if bandgap reference with buffer is used	–	–	—	200	300	μA
I_{LVR}	Additional Power Consumption if LVR is used	3V	LVR disable→LVR enable	—	30	45	μA
		5V		—	60	90	μA
I_{LVD}	Additional Power Consumption if LVD is used	3V	LVD disable→LVD enable (LVR disable)	—	40	60	μA
		5V		—	75	115	μA
		3V	LVD disable→LVD enable (LVR enable)	—	30	45	μA
		5V		—	60	90	μA
t_{BGS}	V_{BG} turn on stable time	–	–	10	–	–	ms
t_{LVR}	Low Voltage Width to Reset	—	—	120	—	480	μs
t_{LVD}	Low Voltage Width to Interrupt	—	—	20	45	90	μs
t_{LVDS}	LVDO stable time	—	For LVR enable, LVD off→on	15	—	—	μs
		—	For LVR disable, LVD off→on	15	—	—	μs
t_{SRESET}	Software Reset Width to Reset	—	—	45	90	120	μs

Comparator Electrical Characteristics

Ta=25°C

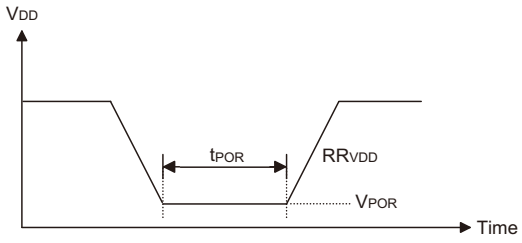
Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Condition				
V_{CMP}	Comparator operating voltage	—	—	2.2	—	5.5	V
I_{CMP}	Comparator operating current	3V	—	—	50	75	μA
		5V	—	—	85	130	μA
V_{CMPOS}	Comparator input offset voltage	—	—	-10	—	+10	mV
V_{HYS}	Hysteresis width	—		20	40	60	mV
V_{CM}	Comparator common mode voltage range	—	—	V_{SS}	—	V_{DD}-1.4V	V
A_{OL}	Comparator open loop gain	—	—	60	80	—	dB
t_{PD}	Comparator response time	3V	With 100mV overdrive(Note)	—	200	400	ns
		5V					

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

Power on Reset Electrical Characteristics

Ta=25°C

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



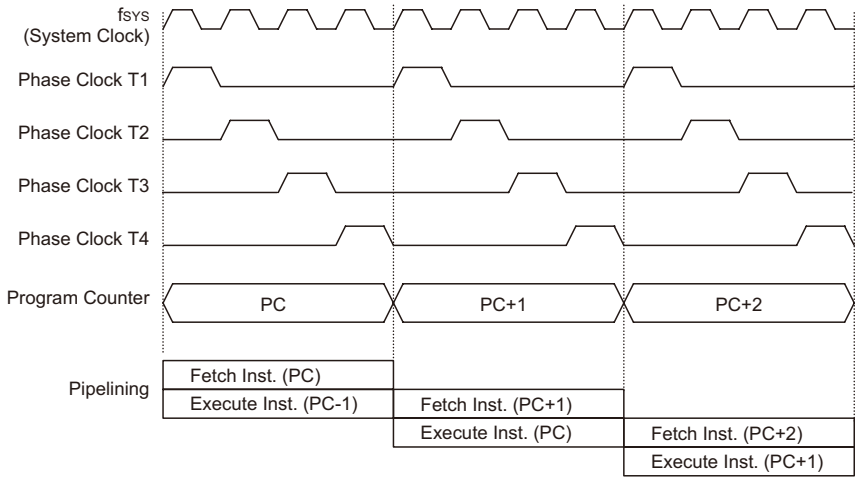
System Architecture

A key factor in the high-performance features of the Holtek range of microcontrollers is attributed to their internal system architecture. The range of 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 these devices suitable for low-cost, high-volume production for controller applications.

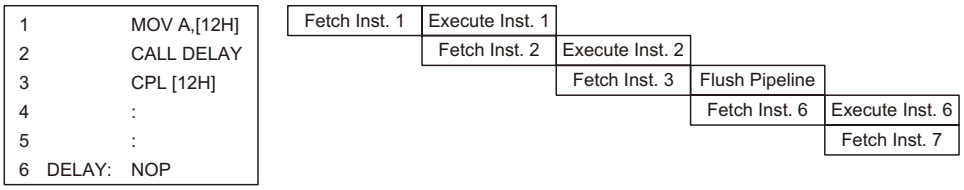
Clocking and Pipelining

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.

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



System Clocking and Pipelining



Instruction Fetching

Program Counter

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

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

Device	Program Counter	
	Porgram Counter High Byte	Porgram Counter Low Byte
HT66F60A	PC13~PC8	PCL7~PCL0
HT66F70A	PC14~PC8	

Program Counter

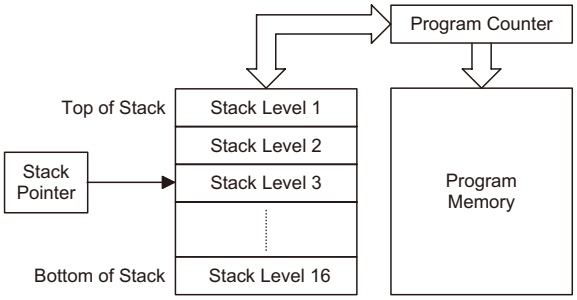
The lower byte of the Program Counter, known as the Program Counter Low register or PCL, is available for program control and is a readable and writeable register. By transferring data directly into this register, a short program jump can be executed directly; however, as only this low byte is available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

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

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



Arithmetic and Logic Unit – ALU

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

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

Flash Program Memory

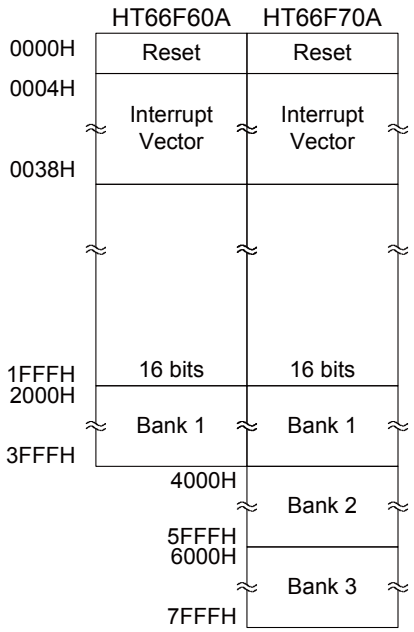
The Program Memory is the location where the user code or program is stored. For these devices series the Program Memory are Flash type, which means it can be programmed and re-programmed a large number of times, allowing the user the convenience of code modification on the same device. By using the 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 16K×16 bits to 32K×16 bits. The Program Memory is addressed by the Program Counter and also contains data, table information and interrupt entries information. Table data, which can be setup in any location within the Program Memory, is addressed by separate table pointer registers.

Device	Capacity	Banks
HT66F60A	16K × 16	0~1
HT66F70A	32K × 16	0~3

The series of devices has its Program Memory divided into two or four Banks, Bank 0~Bank 1 or Bank 0~Bank 3 respectively. The required Bank is selected using Bit 0 or Bit 0~1 of the BP Register dependent upon which device is selected.



Program Memory Structure

Special Vectors

Within the Program Memory, certain locations are reserved for the reset and interrupts. The location 0000H is reserved for use by these devices 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 memory [m] is located in current page. If the memory [m] is located in other pages, the data can be retrieved from the program memory using the "LTABRD[m]" or "LTABRDL[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.

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

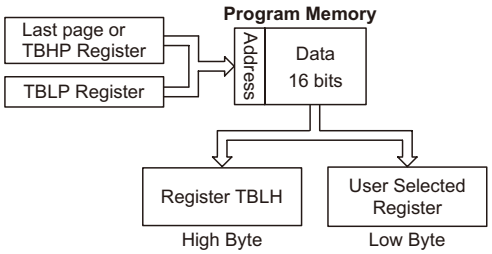


Table Program Example

The accompanying example shows how the table pointer and table data is defined and retrieved from the device. This example uses raw table data located in the last page which is stored there using the ORG statement. The value at this ORG statement is "3F00H" which refers to the start address of the last page within the 16K Program Memory of the HT66F60A device. 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 "3F06H" 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:

```
tempreg1 db ?          ; temporary register #1 in current page
tempreg2 db ?          ; temporary register #2 in current page
        :
        :
mov  a,06h             ; initialise low byte table pointer - note that this address
                       ; is referenced
mov  tblp, a           ; to the last page or present page
        :
        :
tabrdl   tempreg1      ; transfers value in table referenced by table pointer to tempreg1
                       ; Data at program memory address "3F06H" transferred to tempreg1
                       ; and TBLH
dec  tblp              ; reduce value of table pointer by one
tabrdl   tempreg2      ; transfers value in table referenced by table pointer to tempreg2
                       ; Data at program memory address "3F05H" transferred to tempreg2
                       ; and TBLH. In this example the data "1AH" is transferred to
                       ; tempreg1 and data "0FH" to register tempreg2 while the value "00H"
                       ; will be transferred to the high byte register TBLH
        :
        :
org 3F00h              ; sets initial address of last page

dc 000Ah, 000Bh, 000Ch, 000Dh, 000Eh, 000Fh, 001Ah, 001Bh
        :
        :
```

In Circuit Programming – ICP

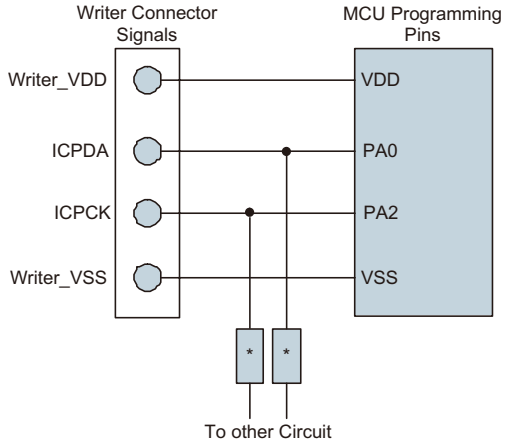
The provision of Flash type Program Memory provides the user with a means of convenient and easy upgrades and modifications to their programs on the same device.

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.

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

The Program Memory can be programmed serially in-circuit using this 4-wire interface. Data is downloaded and uploaded serially on a single pin with an additional line for the clock. Two additional lines are required for the power supply and one line for the reset. 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, the user must take care of the ICPDA and ICPCK pins for data and clock programming purposes to ensure that no other outputs are connected to these two pins.



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

On-Chip Debug Support – OCDS

There is an EV chip named HT66V70A which is used to emulate the HT66Fx0A series of devices. The HT66V70A device also provides the "On-Chip Debug" function to debug the HT66Fx0A series of devices during development process. The devices, HT66Fx0A and HT66V70A, are almost functional compatible except the "On-Chip Debug" function and package types. Users can use the HT66V70A device to emulate the HT66Fx0A series of devices behaviors 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 HT66V70A EV chip for debugging, the corresponding pin functions shared with the OCDSDA and OCDSCK pins in the HT66Fx0A series of devices will have no effect in the HT66V70A 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
GND	VSS	Ground

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: 64 words/page
- Writing: 64 words/time
- Reading: 1 word/time

In Application Programming Control Register

The Address register, FARL and FARH, and the Data registers, FD0L/FD0H, FD1L/FD1H, FD2L/FD2H and FD3L/FD3H, located in Data Memory section0, together with the Control registersr, FC0, FC1 and FC2, located in Data Memory section1 are the corresponding Flash access registers for IAP. As indirect addressing is the only way to access the FC0, FC1 and FC2 registers, all read and write operations to the registers must be performed using the Indirect Addressing Register, IAR1, and the Memory Pointer pair, MP1L and MP1H. Because the FC0, FC1 and FC2 control registers are located at the address of 43H~45H in Data Memory section1, the desired value ranged from 43H to 45H must first be written into the MP1L Memory Pointer low byte and the value "01H" must also be written into the MP1H Memory Pointer high byte.

• **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	1	1	1	0	0	0	0

Bit 7 **CFWEN:** Flash Memory Write enable control
0: Flash memory write function is disabled
1: Flash memory write function has been successfully enabled
When this bit is cleared to 0 by application program, the Flash memory write function is disabled. Note that writing a "1" into this bit results in no action. This bit is used to indicate that the Flash memory write function status. When this bit is set to 1 by hardware, it means that the Flash memory write function is enabled successfully. Otherwise, the Flash memory write function is disabled as the bit content is zero.

Bit 6~4 **FMOD2~FMOD0:** Mode selection
000: Write program memory
001: Page erase program memory
010: Reserved
011: Read program memory
101: Reserved
110: FWEN mode – Flash memory Write function Enable mode
111: Reserved

Bit 3 **FWPEN:** Flash memory write procedure enable control
0: Disable
1: Enable
When this bit is set to 1 and the FMOD field is set to "110", the IAP controller will execute the "Flash memory write function enable" procedure. Once the Flash memory write function is successfully enabled, it is not necessary to set the FWPEN bit any more.

Bit 2 **FWT:** Flash ROM write control bit
0: Do not initiate Flash memory write or Flash memory Write process is completed
1: Initiate Flash memory write process
This bit is set by software and cleared by hardware when the Flash memory write process is completed.

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

Bit 0 **FRD:** Flash memory read control bit
0: Do not initiate Flash memory read or Flash memory read process is completed
1: Initiate Flash memory read process
This bit is set by software and cleared by hardware when the Flash memory read process is completed.

- **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 Memory Write buffer clear control
0: Do not initiate Wirte Buffer Clear or Wirte Buffer Clear process is completed
1: Initiate Wirte Buffer Clear process
This bit is set by software and cleared by hardware when the Wirte Buffer Clear process is completed.

- **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 Flash Memory Address [7:0]

- **FARH Register**

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 Flash Memory 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 The first Flash Memory 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 The first Flash Memory 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 The second Flash Memory 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 The second Flash Memory 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 The third Flash Memory 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 The third Flash Memory 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 The fourth Flash Memory data [7:0]

- **FD3H Register**

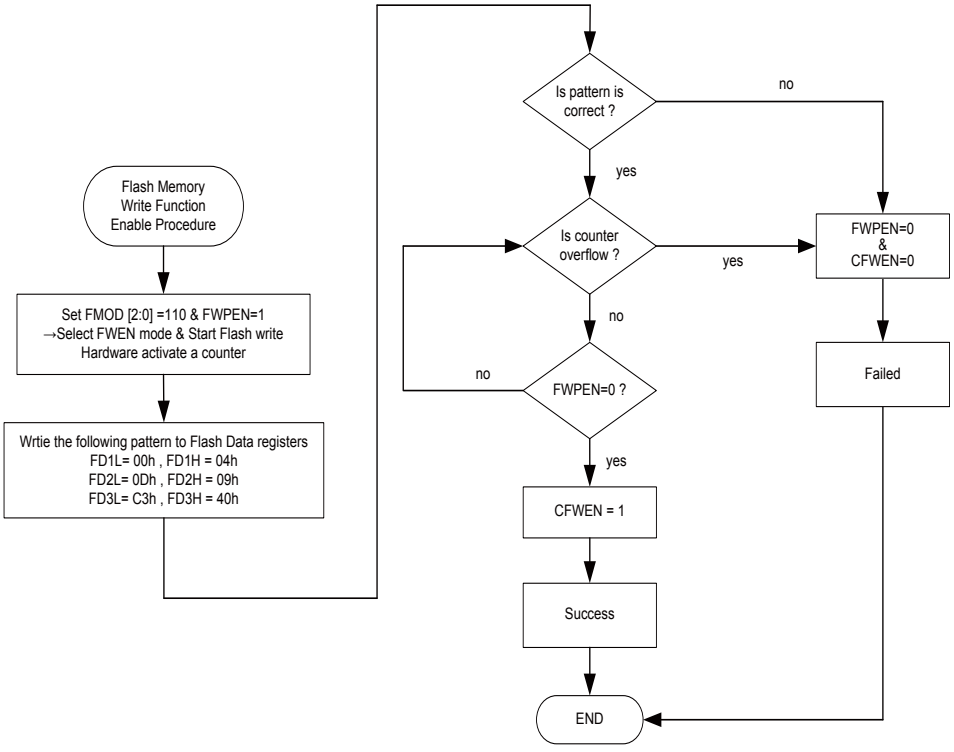
Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 The fourth Flash Memory data [15:8]

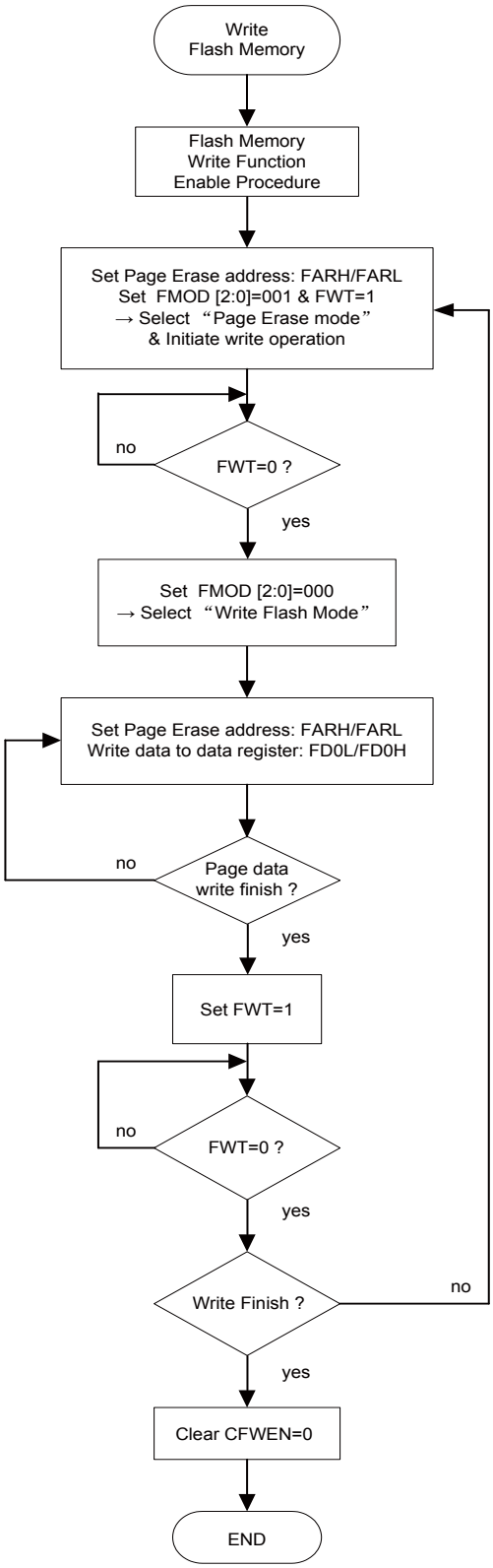
Flash Memory Write Function Enable Procedure

In order to allow users to change the Flash memory data through the IAP control registers, users must first enable the Flash memory write operation by the following procedurce:

- Write "110" into the FMOD2~FMOD0 bits to select the FWEN mode.
- Set the FWPEN bit to "1". The step 1 and step 2 can be executed simultaneously.
- The pattern data with a sequence of 00H, 04H, 0DH, 09H, C3H and 40H must be written into the FD1L, FD1H, FD2L, FD2H, FD3L and FD3H registers respectively.
- A counter with a time-out period of 300μs will be activated to allow users writing the correct pattern data into the FD1L/FD1H~FD3L/FD3H register pairs. The counter clock is derived from LIRC oscillator.
- If the counter overflows or the pattern data is incorrect, the Flash memory write operation will not be enabled and users must again repeat the above procedure. Then the FWPEN bit will automatically be cleared to 0 by hardware.
- If the pattern data is correct before the counter overflows, the Flash memory write operation will be enabled and the FPWEN bit will automatically be cleared to 0 by hardware. The CFWEN bit will also be set to 1 by hardware to indicate that the Flash memory write operation is successfully enabled.
- Once the Flash memory write operation is enabled, the user can change the Flash ROM data through the Flash control register.
- To disable the Flash mermoy write operation, the user can clear the CFWEN bit to 0.



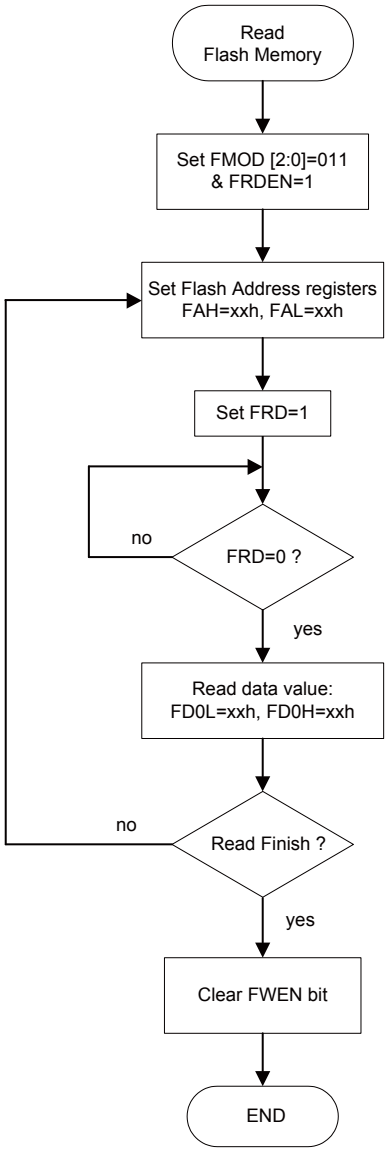
Flash Memory Write Function Enable Procedure



Write Flash Memory Procedure

ERASE PAGE	FARH	FARL[7:6]	Note
0	0000 0000	00	
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	
: :	: :	: :	FARL [5:0]: don't care
252	0011 1111	00	
253	0011 1111	01	
254	0011 1111	10	
255	0011 1111	11	
: :	: :	: :	
508	0111 1111	00	
509	0111 1111	01	
510	0111 1111	10	
511	0111 1111	11	

Note: There are 256 IAP erase pages in the HT66F60A device while there are 512 IAP erase pages in the HT66F70A device.



Read Flash Memory Procedure

Note: When the FWT or FRD bit is set to 1, the MCU is stopped.

Data Memory

The Data Memory is an 8-bit wide RAM internal memory and is the location where temporary information is stored.

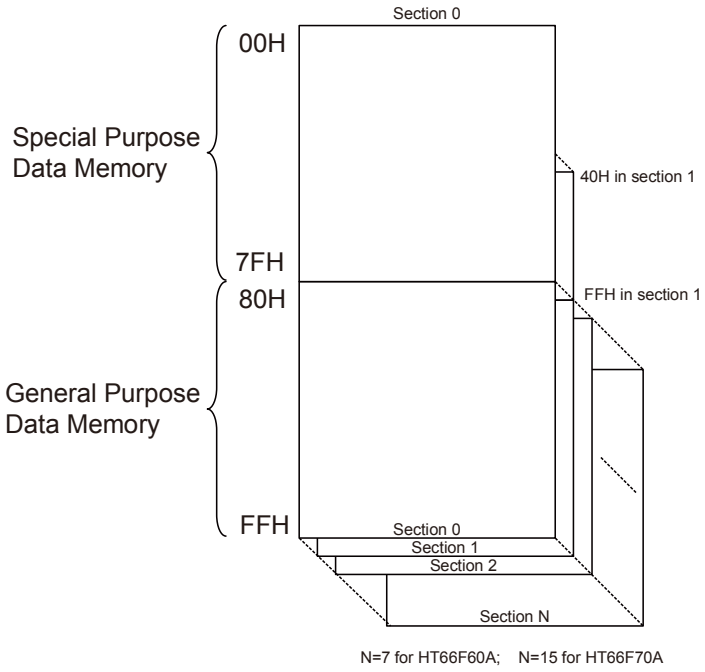
Divided into two types, the first of Data Memory is an area of RAM where special function registers are located. These registers have fixed locations and are necessary for correct operation of the device. Many of these registers can be read from and written to directly under program control, however,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

Structure

The Data Memory is divided into several sections, all of which are implemented in 8-bit wide Memory. Each of the Data Memory sections is categorized into two types, the Special Purpose Data Memory and the General Purpose Data Memory.

The start address of the Special Purpose Data Memory for all devices is the address 00H while the start address of the General Purpose Data Memory is the address 80H. The special purpose registers which are addressed from 00H to 3FH in Data Memory are common to all sections and are accessible in all sections. However, the special purpose registers located in the section 1 can only be accessed with the address from 40H to FFH.

Device	Capacity	Sections
HT66F60A	General Purpose: 1024×8	0: 80H~FFH 1: 80H~FFH : : 7: 80H~FFH
HT66F70A	General Purpose: 2048×8	0: 80H~FFH 1: 80H~FFH : : 15: 80H~FFH



Data Memory Structure

General Purpose Data Memory

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

Special Purpose Data Memory

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

	Section 0~7		Section 0, 2~7	Section 1
00H	IAR0	40H	Unused	EEC
01H	MP0	41H	EEA	Unused
02H	IAR1	42H	EED	Unused
03H	MP1L	43H	Unused	FC0
04H	MP1H	44H	Unused	FC1
05H	ACC	45H	Unused	FC2
06H	PCL	46H	CP0C	Unused
07H	TBLP	47H	CP1C	Unused
08H	TBLH	48H	TM1C0	IFS0
09H	TBHP	49H	TM1C1	IFS1
0AH	STATUS	4AH	TM1C2	IFS2
0BH	BP	4BH	TM1DL	IFS3
0CH	IAR2	4CH	TM1DH	IFS4
0DH	MP2L	4DH	TM1AL	IFS5
0EH	MP2H	4EH	TM1AH	Unused
0FH	Unused	4FH	TM1BL	Unused
10H	PAWU	50H	TM1BH	TM4C0
11H	PAPU	51H	TM2C0	TM4C1
12H	PA	52H	TM2C1	TM4DL
13H	PAC	53H	TM2DL	TM4DH
14H	Unused	54H	TM2DH	TM4AL
15H	PBPU	55H	TM2AL	TM4AH
16H	PB	56H	TM2AH	TM4RP
17H	PBC	57H	TM2RP	TM5C0
18H	Unused	58H	TM3C0	TM5C1
19H	PCPU	59H	TM3C1	TM5DL
1AH	PC	5AH	TM3DL	TM5DH
1BH	PCC	5BH	TM3DH	TM5AL
1CH	Unused	5CH	TM3AL	TM5AH
1DH	PDPU	5DH	TM3AH	TM5RP
1EH	PD	5EH	TM0C0	Unused
1FH	PDC	5FH	TM0C1	Unused
20H	Unused	60H	TM0DL	PAS0
21H	PEPU	61H	TM0DH	PAS1
22H	PE	62H	TM0AL	PAS2
23H	PEC	63H	TM0AH	PAS3
24H	Unused	64H	PSC0	Unused
25H	PFPU	65H	TBC0	Unused
26H	PF	66H	TBC1	PBS2
27H	PFC	67H	PSC1	PBS3
28H	Unused	68H	ADCR0	PCS0
29H	PGPU	69H	ADCR1	PCS1
2AH	PG	6AH	ADRL	PCS2
2BH	PGC	6BH	ADRH	PCS3
2CH	Unused	6CH	SIMC0	PDS0
2DH	PHPU	6DH	SIMC1	PDS1
2EH	PH	6EH	SIMD	PDS2
2FH	PHC	6FH	SIMC2/SIMA	PDS3
30H	INTC0	70H	I2CTOC	PES0
31H	INTC1	71H	SPIAC0	PES1
32H	INTC2	72H	SPIAC1	PES2
33H	INTC3	73H	SPIAD	PES3
34H	MFI0	74H	FARL	PFS0
35H	MFI1	75H	FARH	Unused
36H	MFI2	76H	FD0L	Unused
37H	MFI3	77H	FD0H	Unused
38H	MFI4	78H	FD1L	PGS0
39H	INTEG	79H	FD1H	PGS1
3AH	SMOD	7AH	FD2L	PGS2
3BH	SMOD1	7BH	FD2H	PGS3
3CH	LVRC	7CH	FD3L	PHS0
3DH	LVDC	7DH	FD3H	PHS1
3EH	WDTC	7EH	TBC2	PHS2
3FH	SMOD2	7FH	SCOMC	Unused

: Unused, read as 00H

HT66F60A Special Purpose Data Memory

	Section 0~15		Section 0, 2~15	Section 1
00H	IAR0	40H	Unused	EEC
01H	MP0	41H	EEA	Unused
02H	IAR1	42H	EED	Unused
03H	MP1L	43H	Unused	FC0
04H	MP1H	44H	Unused	FC1
05H	ACC	45H	Unused	FC2
06H	PCL	46H	CP0C	Unused
07H	TBLP	47H	CP1C	Unused
08H	TBLH	48H	TM1C0	IFS0
09H	TBHP	49H	TM1C1	IFS1
0AH	STATUS	4AH	TM1C2	IFS2
0BH	BP	4BH	TM1DL	IFS3
0CH	IAR2	4CH	TM1DH	IFS4
0DH	MP2L	4DH	TM1AL	IFS5
0EH	MP2H	4EH	TM1AH	Unused
0FH	Unused	4FH	TM1BL	Unused
10H	PAWU	50H	TM1BH	TM4C0
11H	PAPU	51H	TM2C0	TM4C1
12H	PA	52H	TM2C1	TM4DL
13H	PAC	53H	TM2DL	TM4DH
14H	Unused	54H	TM2DH	TM4AL
15H	PBPU	55H	TM2AL	TM4AH
16H	PB	56H	TM2AH	TM4RP
17H	PBC	57H	TM2RP	TM5C0
18H	Unused	58H	TM3C0	TM5C1
19H	PCPU	59H	TM3C1	TM5DL
1AH	PC	5AH	TM3DL	TM5DH
1BH	PCC	5BH	TM3DH	TM5AL
1CH	Unused	5CH	TM3AL	TM5AH
1DH	PDPU	5DH	TM3AH	TM5RP
1EH	PD	5EH	TM0C0	Unused
1FH	PDC	5FH	TM0C1	Unused
20H	Unused	60H	TM0DL	PAS0
21H	PEPU	61H	TM0DH	PAS1
22H	PE	62H	TM0AL	PAS2
23H	PEC	63H	TM0AH	PAS3
24H	Unused	64H	PSC0	Unused
25H	PFPU	65H	TBC0	Unused
26H	PF	66H	TBC1	PBS2
27H	PFC	67H	PSC1	PBS3
28H	Unused	68H	ADCR0	PCS0
29H	PGPU	69H	ADCR1	PCS1
2AH	PG	6AH	ADRL	PCS2
2BH	PGC	6BH	ADRH	PCS3
2CH	Unused	6CH	SIMC0	PDS0
2DH	PHPU	6DH	SIMC1	PDS1
2EH	PH	6EH	SIMD	PDS2
2FH	PHC	6FH	SIMC2/SIMA	PDS3
30H	INTC0	70H	I2CTOC	PES0
31H	INTC1	71H	SPIAC0	PES1
32H	INTC2	72H	SPIAC1	PES2
33H	INTC3	73H	SPIAD	PES3
34H	MFI0	74H	FARL	PFS0
35H	MFI1	75H	FARH	Unused
36H	MFI2	76H	FD0L	Unused
37H	MFI3	77H	FD0H	Unused
38H	MFI4	78H	FD1L	PGS0
39H	INTEG	79H	FD1H	PGS1
3AH	SMOD	7AH	FD2L	PGS2
3BH	SMOD1	7BH	FD2H	PGS3
3CH	LVRC	7CH	FD3L	PHS0
3DH	LVDC	7DH	FD3H	PHS1
3EH	WDTC	7EH	TBC2	PHS2
3FH	SMOD2	7FH	SCOMC	Unused

: Unused, read as 00H

HT66F70A 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, IAR1 and IAR2, although having their locations in normal RAM register space, do not actually physically exist as normal registers. The method of indirect addressing for RAM data manipulation uses these Indirect Addressing Registers and Memory Pointers, in contrast to direct memory addressing, where the actual memory address is specified. Actions on the IAR0, IAR1 and IAR2 registers will result in no actual read or write operation to these registers but rather to the memory location specified by their corresponding Memory Pointers, MP0, MP1L/MP1H or MP2L/MP2H. Acting as a pair, IAR0 and MP0 can together access data only from Section 0 while the IAR1 register together with MP1L/MP1H register pair and IAR2 register together with MP2L/MP2H register pair can access data from any Data Memory section. 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

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

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 RAM addresses.

Bank Pointer – BP

Depending upon which device is used, the Program Memory is divided into several banks. Selecting the required Program Memory area is achieved using the Bank Pointer.

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. 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.

Device	Bit							
	7	6	5	4	3	2	1	0
HT66F60A	—	—	—	—	—	—	—	BP0
HT66F70A	—	—	—	—	—	—	BP1	BP0

BP Register List

BP Register – HT66F60A

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	BP0
R/W	—	—	—	—	—	—	—	R/W
POR	—	—	—	—	—	—	—	0

Bit 7~1 Unimplemented, read as "0"

Bit 0 **BP0:** Program memory bank point
0: Bank 0
1: Bank 1

BP Register – HT66F70A

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	BP1	BP0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **BP1~BP0:** Program memory bank point
00: Bank 0
01: Bank 1
10: Bank 2
11: Bank 3

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), SC flag, CZ flag, power down flag (PDF), and watchdog time-out flag (TO). These arithmetic/logical operation and system management flags are used to record the status and operation of the microcontroller.

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

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

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

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

STATUS Register

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

"x": unknown

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

Bit 6 **CZ:** The the operational result of different flags for different instuctions.
For SUB/SUBM/LSUB/LSUBM instructions, the CZ flag is equal to the Z flag. For SBC/ SBCM/ LSBC/ LSBCM instructions, the CZ flag is the "AND" operation result which is performed by the previous operation CZ flag and current operation zero flag. For other instructions, the CZ flag willl not be affected.

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

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

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

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

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

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

EEPROM Data Memory

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

Device	Capacity	Address
HT66F60A	128×8	00H~7FH
HT66F70A		

EEPROM Data Memory Structure

The EEPROM Data Memory capacity is 128×8 bits for this series of devices. Unlike the Program Memory and RAM Data Memory, the EEPROM Data Memory is not directly mapped into memory space and is therefore not directly addressable in the same way as the other types of memory. Read andWrite operations to the EEPROM are carried out in single byte operations using an address and data register in Section 0 and a single control register in Section 1.

EEPROM Registers

Three registers control the overall operation of the internal EEPROM Data Memory. These are the address register, EEA, the data register, EED and a single control register, EEC. As both the EEA and EED registers are located in Section 0, they can be directly accessed in the same was as any other Special Function Register. The EEC register however, being located in Bank1, cannot be addressed directly and can only be read from or written to indirectly using the MP1 Memory Pointer and Indirect Addressing Register, IAR1. Because the EEC control register is located at address 40H in Section 1, the MP1L Memory Pointer low byte must first be set to the value 40H and the MP1H Memory Pointer high byte set to the value 01H before any operations on the EEC register are executed.

EEA Register

Bit	7	6	5	4	3	2	1	0
Name	—	EEA6	EEA5	EEA4	EEA3	EEA2	EEA1	EEA0
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": unknown

Bit 7 Unimplemented, read as "0"

Bit 6~0 **EEA6~EEA0:** Data EEPROM address bit 6~bit 0

EED Register

Bit	7	6	5	4	3	2	1	0
Name	EED7	EED6	EED5	EED4	EED3	EED2	EED1	EED0
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 **EED7~EED0:** Data EEPROM data bit 7~bit 0

EEC Register

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

Bit 7~4 Unimplemented, read as "0"

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

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

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

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

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

Reading Data from the EEPROM

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

Writing Data to the EEPROM

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

Write Protection

Protection against inadvertent write operation is provided in several ways. After the device is powered-on the Write Enable bit in the control register will be cleared preventing any write operations. Also at power-on the Memory Pointer pairs, MP1L/MP1H and MP2L/MP2H, will be reset to zero, which means that Data Memory Section 0 will be selected. As the EEPROM control register is located in Section 1, this adds a further measure of protection against spurious write operations. During normal program operation, ensuring that the Write Enable bit in the control register is cleared will safeguard against incorrect write operations.

EEPROM Interrupt

The EEPROM write interrupt is generated when an EEPROM write cycle has ended. The EEPROM interrupt must first be enabled by setting the DEE bit in the relevant interrupt register. However as the EEPROM is contained within a Multi-function Interrupt, the associated multi-function interrupt enable bit must also be set. When an EEPROM write cycle ends, the DEF request flag and its associated multi-function interrupt request flag will both be set. If the global, EEPROM and Multi-function interrupts are enabled and the stack is not full, a jump to the associated Multi-function Interrupt vector will take place. When the interrupt is serviced only the Multi-function interrupt flag will be automatically reset, the EEPROM interrupt flag must be manually reset by the application program. More details can be obtained in the Interrupt section.

Programming Considerations

Care must be taken that data is not inadvertently written to the EEPROM. Protection can be enhanced by ensuring that the Write Enable bit is normally cleared to zero when not writing. Also the Memory Pointer high byte, MP1H or MP2H, could be normally cleared to zero as this would inhibit access to Data Memory Section 1 where the EEPROM control register exist. Although certainly not necessary, consideration might be given in the application program to the checking of the validity of new write data by a simple read back process.

When writing data the WR bit must be set high immediately after the WREN bit has been set high, to ensure the write cycle executes correctly. The global interrupt bit EMI should also be cleared before a write cycle is executed and then re-enabled after the write cycle starts.

Programming Examples

Reading Data from the EEPROM – Polling Mothod

```
MOV  A, EEPROM_ADRES      ; user defined address
MOV  EEA, A
MOV  A, 040H              ; setup memory pointer MP1L
MOV  MP1L, A              ; MP1 points to EEC register
MOV  A, 01H               ; setup memory pointer MP1H
MOV  MP1H, A
SET  IAR1.1               ; set RDEN bit, enable read operations
SET  IAR1.0               ; start Read Cycle - set RD bit
BACK:
SZ       IAR1.0           ; check for read cycle end
JMP  BACK
CLR  IAR1                 ; disable EEPROM read/write
CLR  MP1H
MOV  A, EED               ; move read data to register
MOV  READ_DATA, A
```

Writing Data to the EEPROM – Polling Mothod

```
CLR  EMI
MOV  A, EEPROM_ADRES      ; user defined address
MOV  EEA, A
MOV  A, EEPROM_DATA       ; user defined data
MOV  EED, A
MOV  A, 040H              ; setup memory pointer MP1L
MOV  MP1L, A              ; MP1 points to EEC register
MOV  A, 01H               ; setup memory pointer MP1H
MOV  MP1H, A
SET  IAR1.3               ; set WREN bit, enable write operations
SET  IAR1.2               ; Start Write Cycle - set WR bit - executed immediately
                          ; after set WREN bit
SET  EMI
BACK:
SZ       IAR1.2           ; check for write cycle end
JMP  BACK
CLR  IAR1                 ; disable EEPROM read/write
CLR  MP1H
```

Oscillator

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, these devices have the flexibility to optimize the performance/power ratio, a feature especially important in power sensitive portable applications.

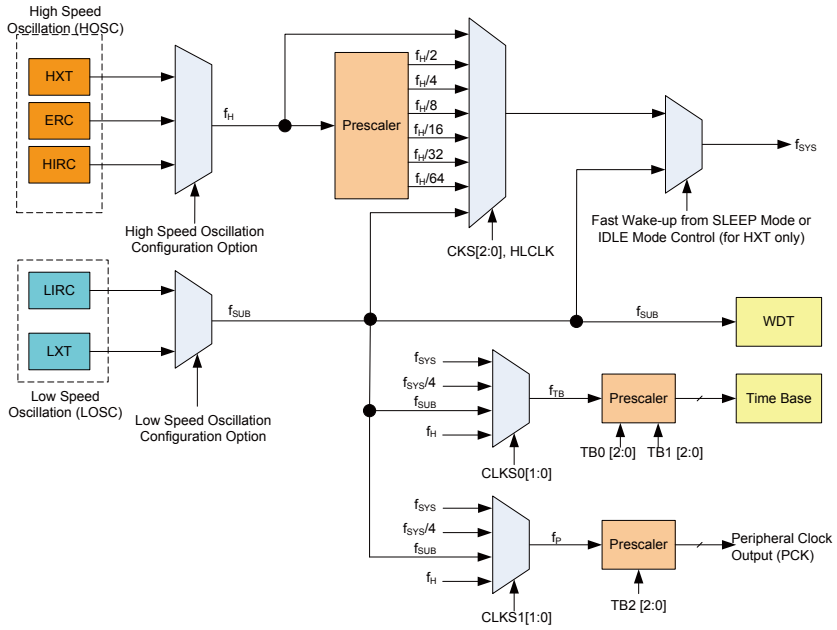
Type	Name	Frequency	Pins
External Crystal	HXT	400kHz~16MHz	OSC1/OSC2
External RC	ERC	400kHz~16MHz	OSC1
Internal High Speed RC	HIRC	8MHz	—
External Low Speed Crystal	LXT	32.768kHz	XT1/XT2
Internal Low Speed RC	LIRC	32kHz	—

Oscillator Types

System Clock Configurations

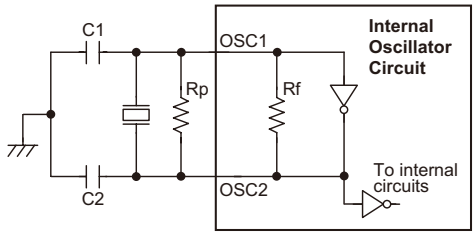
There are five methods of generating the system clock, three high speed oscillators and two low speed oscillators. The high speed oscillators are is the external crystal/ceramic oscillator, external RC network oscillator and the internal 8MHz RC oscillator. The two low speed oscillators are the internal 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 SMOD 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 SMOD register. Note that two oscillator selections must be made namely one high speed and one low speed system oscillators. It is not possible to choose a no-oscillator selection for either the high or low speed oscillator.



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.



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

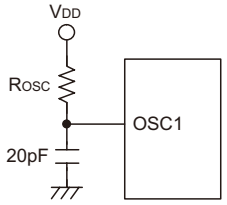
Crystal/Resonator Oscillator – HXT

Crystal Oscillator C1 and C2 Values		
Crystal Frequency	**C1**	**C2**
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

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 PB1, leaving pin PB2 free for use as a normal I/O pin.



External RC Oscillator – ERC

Internal High Speed RC Oscillator – HIRC

The internal RC oscillator is a fully integrated system oscillator requiring no external components. The internal RC oscillator has a fixed frequency of 8MHz. Device trimming during the manufacturing process and the inclusion of internal frequency compensation circuits are used to ensure that the influence of the power supply voltage, temperature and process variations on the oscillation frequency are minimised. As a result, at a power supply of 5V and at a temperature of 25°C degrees, the fixed oscillation frequency of 8MHz will have a tolerance within 2%. Note that if this internal system clock option is selected, as it requires no external pins for its operation, I/O pins PB1 and PB2 are free for use as normal I/O pins.

External 32.768kHz Crystal Oscillator – LXT

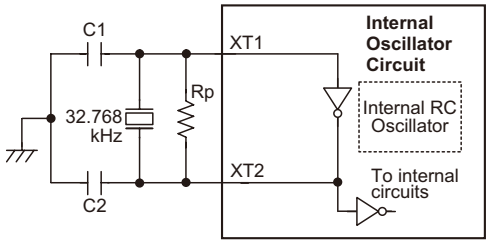
The External 32.768kHz Crystal System Oscillator is one of the low frequency oscillator choices, which is selected via configuration option. This clock source has a fixed frequency of 32.768kHz and requires a 32.768kHz crystal to be connected between pins XT1 and XT2. The external resistor and capacitor 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's specification. The external parallel feedback resistor, Rp, 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.



Note: 1. Rp, C1 and C2 are required.
2. Although not shown pins have aparasitic 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=5MΩ~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 SMOD2 register.

- **SMOD2 Register**

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

Bit 7~1 Unimplemented, read as "0"

Bit 0 **LXPLP:** LXT Low Power Control
0: Quick Start mode
1: Low Power mode

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 Low Speed Oscillator – LIRC

The Internal 32kHz System Oscillator is one of the low frequency oscillator choices, which is selected via configuration option. It is a fully integrated RC oscillator with a typical frequency of 32kHz at 5V, requiring no external components for its implementation. Device trimming during the manufacturing process and the inclusion of internal frequency compensation circuits are used to ensure that the influence of the power supply voltage, temperature and process variations on the oscillation frequency are minimised. As a result, at a power supply of 5V and at a temperature of 25°C degrees, the fixed oscillation frequency of 32kHz will have a tolerance within 3%.

Supplementary Oscillators

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

Operating Modes and System Clocks

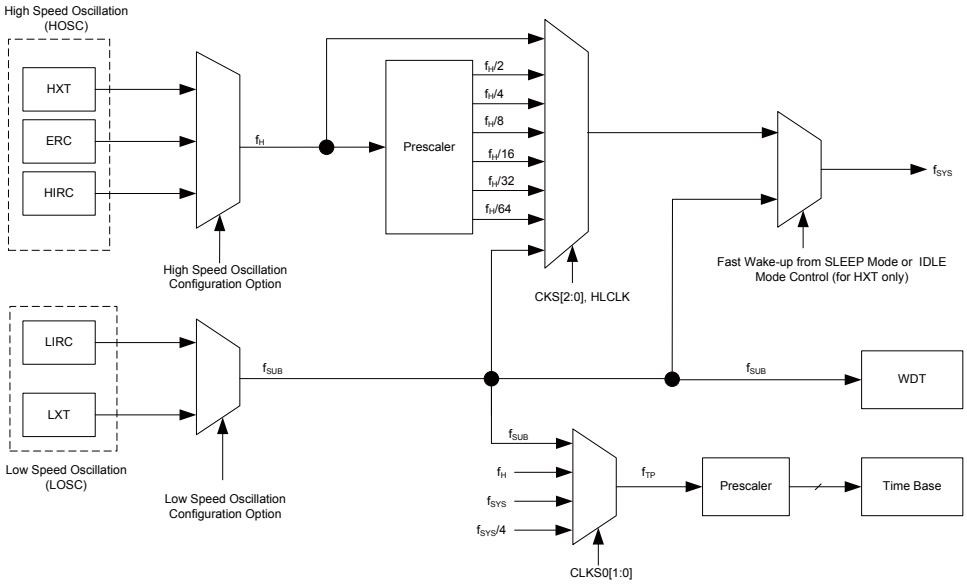
Present day applications require that their microcontrollers have high performance but often still demand that they consume as little power as possible, conflicting requirements that are especially true in battery powered portable applications. The fast clocks required for high performance will by their nature increase current consumption and of course vice versa, lower speed clocks reduce current consumption. As Holtek has provided 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 Clock

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

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

The f_{SUB} clock is used to provide a substitute clock for the microcontroller just after a wake-up has occurred to enable faster wake-up times. The f_{SUB} clock is also used to provide the clock source for time base and watchdog timer functions.



System Clock Configurations

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

System Operation Modes

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

Operating Mode	Description		
	CPU	f_{SYS}	f_{SUB}
NORMAL Mode	On	f_H~f_H/64	On
SLOW Mode	On	f_{SUB}	On
IDLE0 Mode	Off	Off	On
IDLE1 Mode	Off	On	On
SLEEP0 Mode	Off	Off	Off
SLEEP1 Mode	Off	Off	On

NORMAL Mode

As the name suggests this is one of the main operating modes where the microcontroller has all of its functions operational and where the system clock is provided by one of the high speed oscillators. This mode operates allowing the microcontroller to operate normally with a clock source will come from one of the high speed oscillators, either the HXT, 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 SMOD register. Although a high speed oscillator is used, running the microcontroller at a divided clock ratio reduces the operating current.

SLOW Mode

This is also a mode where the microcontroller operates normally although now with a slower speed clock source. The clock source used will be from 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 SLEEP0 Mode is entered when an HALT instruction is executed and the IDLEN bit in the SMOD register is low. In the SLEEP0 mode the CPU will be stopped and the f_{SUB} clock 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 SLEEP1 Mode is entered when an HALT instruction is executed and the IDLEN bit in the SMOD register is low. In the SLEEP1 mode the CPU will be stopped. However the f_{SUB} will continue to operate if the LVDEN is “1” or the Watchdog Timer function is enabled.

IDLE0 Mode

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

IDLE1 Mode

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

Control Register

A register pair, SMOD and SMOD1, is used for overall control of the internal clocks within the device.

SMOD Register

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

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

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

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

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

This is the low speed system oscillator ready flag which indicates when the low speed system oscillator is stable after power on reset or a wake-up has occurred. The flag will be low when in the SLEEP0 Mode but after a wake-up has occurred, the flag will change to a high level after 1024 clock cycles if the LXT oscillator is used or 1~2 clock cycles is 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 or 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: $f_H/2$~$f_H/64$ or f_{SUB}
1: f_H
This bit is used to select if the f_H clock or the $f_H/2$~$f_H/64$ or f_{SUB} clock is used as the system clock. When the bit is high the f_H clock will be selected and if low the $f_H/2$~$f_H/64$ or f_{SUB} clock will be selected. When system clock switches from the f_H clock to the f_{SUB} clock and the f_H clock will be automatically switched off to conserve power.

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

Bit 1 **LRF:** LVR Control register software reset flag
0: Not occurred
1: Occurred
This bit is set to 1 if the LVRC register contains any non defined LVR voltage register values. This in effect acts like a software-reset function. This bit can only be cleared to 0 by the application program.

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

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, stabilize 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 FastWake-up function has no effect because the f_{SUB} clock is stopped. The FastWake-up enable/disable function is controlled using the FSTEN bit in the SMOD1 register.

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

If the ERC/HIRC or LIRC oscillator is used as the system oscillator, then it will take15~16 clock cycles of the ERC/HIRC oscillator or 1~2 clock cycles of the LIRC osrillator respectively 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	x	15~16 ERC cycles	15~16 ERC cycles		1~2 ERC cycles
HIRC	x	15~16 HIRC cycles	15~16 HIRC cycles		1~2 HIRC cycles
LIRC	x	1~2 LIRC cycles	1~2 LIRC cycles		1~2 LIRC cycles
LXT	x	1024 LXT cycles	1024 LXT cycles		1~2 LXT cycles

Wake-up Times

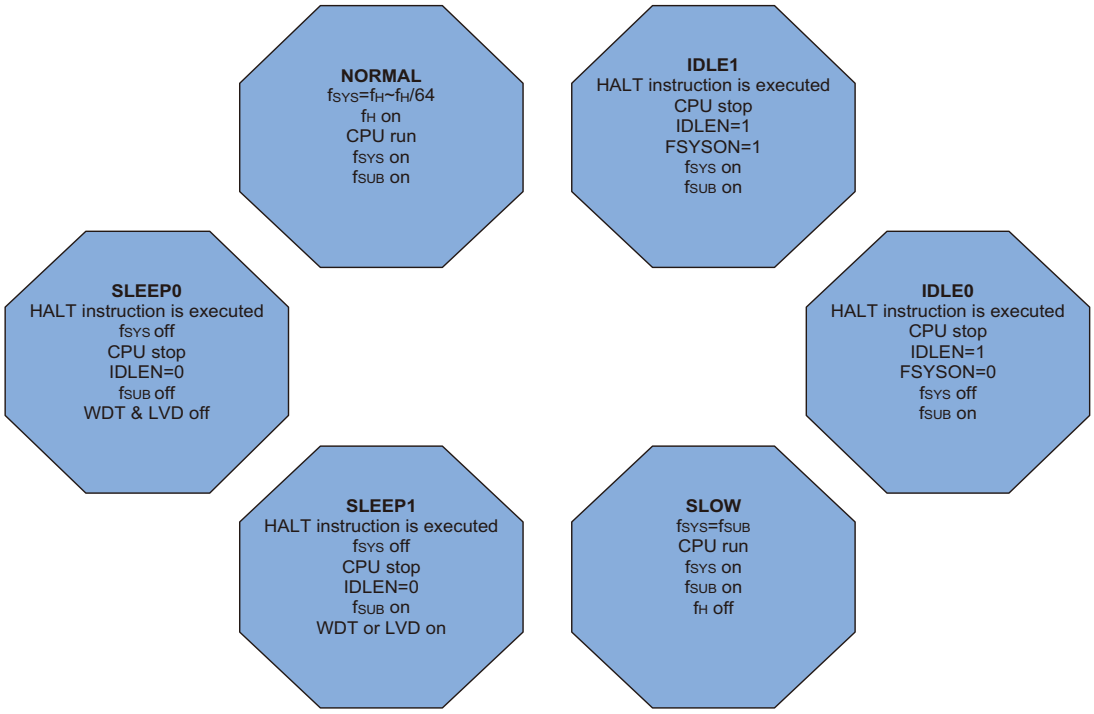
Note that if the Watchdog Timer is disabled, which means that the f_{SUB} clock derived from the LXT or LIRC oscillator is off, then there will be no Fast Wake-up function available when the device wakes up from the SLEEP0 Mode.

Operating Mode Switching

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

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

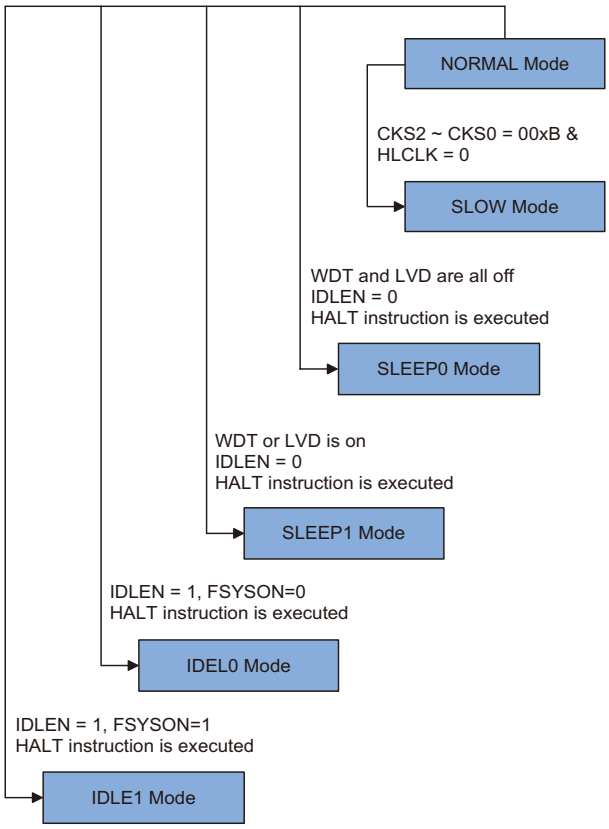
When the HLCLK bit switches to a low level, which implies that clock source is switched from the high speed clock source, f_H, to the clock source, $f_H/2$~$f_H/64$ or f_{SUB}. If the clock is from the f_{SUB}, the high speed clock source will stop running to conserve power. When this happens it must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



NORMAL Mode to SLOW Mode Switching

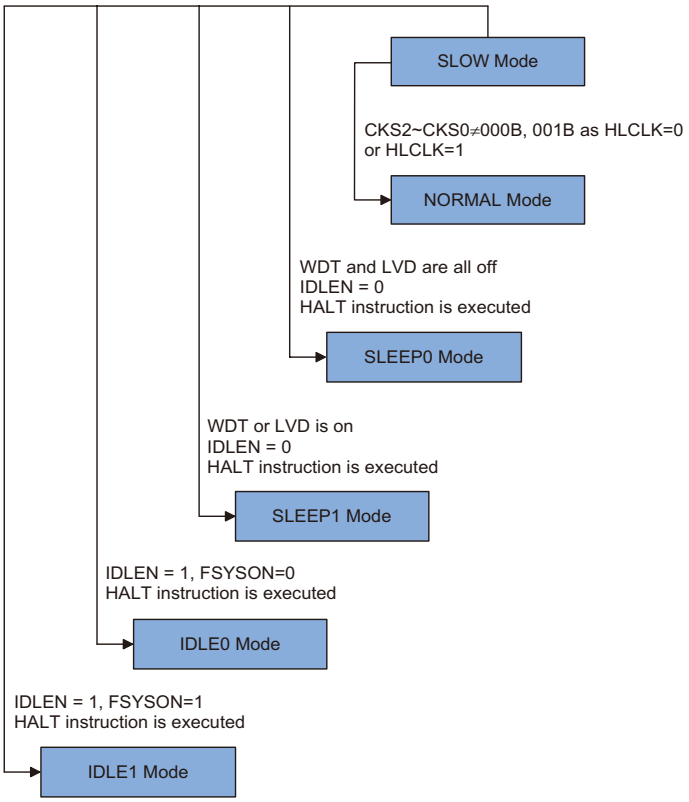
When running in the NORMAL Mode, which uses the high speed system oscillator, and therefore consumes more power, the system clock can switch to run in the SLOW Mode by set the HLCLK bit to "0" and set the CKS2~CKS0 bits to "000" or "001" in the SMOD register. This will then use the low speed system oscillator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

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



SLOW Mode to NORMAL Mode Switching

In SLOW Mode the system uses 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 SMOD register equal to "0" and the WDT and LVD both off. When this instruction is executed under the conditions described above, the following will occur:

- The system clock and the f_{SUB} clock will be stopped and the application program will stop at the "HALT" instruction.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and stopped as the WDT is disabled.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Entering the SLEEP1 Mode

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

- The system clock will be stopped and the application program will stop at the "HALT" instruction, but the WDT or LVD will remain with the clock source coming from the f_{SUB} clock.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting as the WDT is enabled and its clock source is derived from the f_{SUB} 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 IDLE0 Mode

There is only one way for the device to enter the IDLE0 Mode and that is to execute the "HALT" instruction in the application program with the IDLEN bit in SMOD register equal to "1" and the FSYSON bit in 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 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 as the WDT clock source is derived from the f_{SUB} clock.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Entering the IDLE1 Mode

There is only one way for the device to enter the IDLE1 Mode and that is to execute the "HALT" instruction in the application program with the IDLEN bit in SMOD register equal to "1" and the FSYSON bit in SMOD1 register equal to "1". When this instruction is executed under the conditions described above, the following will occur:

- The system 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 as the WDT clock source is derived from the f_{SUB} 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 these devices to as low a value as possible, perhaps only in the order of several micro-amps except in the IDLE1 Mode, there are other considerations which must also be taken into account by the circuit designer if the power consumption is to be minimised. Special attention must be made to the I/O pins on these devices. All high-impedance input pins must be connected to either a fixed high or low level as any floating input pins could create internal oscillations and result in increased current consumption. This also applies to devices which have different package types, as there may be unbonded pins. These must either be setup as outputs or if setup as inputs must have pull-high resistors connected.

Care must also be taken with the loads, which are connected to I/O pins, which are setup as outputs. These should be placed in a condition in which minimum current is drawn or connected only to external circuits that do not draw current, such as other CMOS inputs. Also note that additional standby current will also be required if the configuration options have enabled the 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, these devices will experience a full system reset, however, if these devices are woken up by a WDT overflow, a Watchdog Timer reset will be initiated. Although both of these wake-up methods will initiate a reset operation, the actual source of the wake-up can be determined by examining the TO and PDF flags. The PDF flag is cleared by a system power-up or executing the clear Watchdog Timer instructions and is set when executing the "HALT" instruction. The TO flag is set if a WDT time-out occurs, and causes a wake-up that only resets the Program Counter and Stack Pointer, the other flags remain in their original status.

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

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 stable 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 and TMs, for which the f_{SYS} is used. If the system clock source is switched from f_H to f_{SUB}, the clock source to the peripheral functions mentioned above will change accordingly.
- The on/off condition of f_{SUB} depends upon whether the WDT is enabled or disabled as the WDT clock source is selected from f_{SUB}.

Watchdog Timer

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

Watchdog Timer Clock Source

The Watchdog Timer clock source is provided by the f_{SUB} clock. The f_{SUB} clock can be sourced from either the LXT or LIRC oscillator selected by a configuration option. The LIRC internal oscillator has an approximate frequency of 32kHz and this specified internal clock period can vary with V_{DD}, temperature and process variations. The LXT oscillator is supplied by an external 32.768kHz crystal. 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.

Watchdog Timer Control Register

A single register, WDTC, controls the required timeout period as well as the enable/disable operation. This register controls the overall operation of the Watchdog Timer.

WDTC Register

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

Bit 7~3 **WE4~WE0:** WDT function enable control
10101: Disabled
01010: Enabled
Other Values: Reset MCU
If these bits are changed due to adverse environmental conditions, the microcontroller will be reset. The reset operation will be activated after 2~3 LIRC clock cycles and the WRF bit in the SMOD1 register will be set to 1.

Bit 2~0 **WS2~WS0:** Select WDT Timeout Period
000: $2^8/f_{SUB}$
001: $2^{10}/f_{SUB}$
010: $2^{12}/f_{SUB}$
011: $2^{14}/f_{SUB}$
100: $2^{15}/f_{SUB}$
101: $2^{16}/f_{SUB}$
110: $2^{17}/f_{SUB}$
111: $2^{18}/f_{SUB}$
These three bits determine the division ratio of the Watchdog Timer source clock, which in turn determines the timeout period.

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:** LVR function reset flag
Described elsewhere

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

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

Watchdog Timer Operation

The Watchdog Timer operates by providing a device reset when its timer overflows. This means that in the application program and during normal operation the user has to strategically clear the Watchdog Timer before it overflows to prevent the Watchdog Timer from executing a reset. This is done using the clear watchdog instructions. If the program malfunctions for whatever reason, jumps to an unknown location, or enters an endless loop, these clear instructions will not be executed in the correct manner, in which case the Watchdog Timer will overflow and reset the device. With regard to the Watchdog Timer enable/disable function, there are five bits, WE4~WE0, in the WDTC register to offer the enable/disable control and reset control of the Watchdog Timer. The WDT function will be disabled when the WE4~WE0 bits are set to a value of 10101B while the WDT function will be enabled if the WE4~WE0 bits are equal to 01010B. If the WE4~WE0 bits are set to any other values, other than 01010B and 10101B, it will reset the device after 2~3 f_{SUB} clock cycles. After power on these bits will have a value of 01010B.

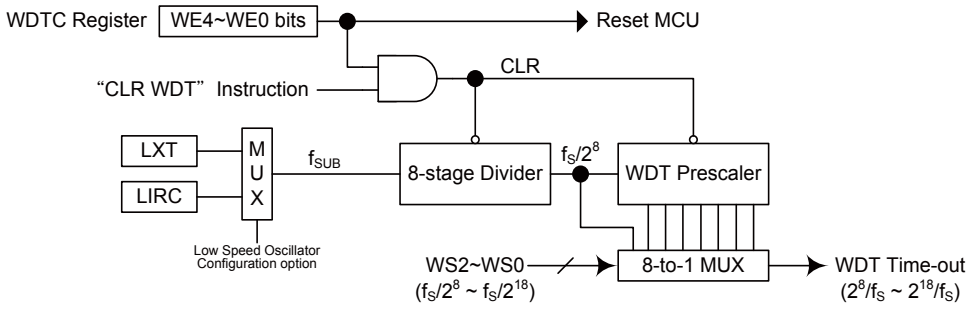
WDT Function Control	WE4~WE0 Bits	WDT Function
Application Program Enabled	10101B	Disable
	01010B	Enable
	Any other value	Reset MCU

Watchdog Timer Enable/Disable Control

Under normal program operation, a Watchdog Timer time-out will initialise a device reset and set the status bit TO. However, if the system is in the SLEEP or IDLE Mode, when a Watchdog Timer time-out occurs, the TO bit in the status register will be set and only the Program Counter and Stack Pointer will be reset. Three methods can be adopted to clear the contents of the Watchdog Timer. The first is a WDT reset, which means a certain value except 01010B and 10101B written into the WE4~WE0 field, the second is using the Watchdog Timer software clear instruction and the third is via a HALT instruction.

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

The maximum time out period is when the 2^{18} division ratio is selected. As an example, with a 32kHz LIRC oscillator as its source clock, this will give a maximum watchdog period of around 8 second for the 2^{18} division ratio, and a minimum timeout of 7.8ms for the 2^8 division ration.



Watchdog Timer

Reset and Initialisation

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

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

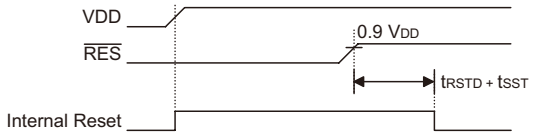
Another type of reset is when the Watchdog Timer overflows and resets the microcontroller. All types of reset operations result in different register conditions being setup. Another reset exists in the form of a Low Voltage Reset, LVR, where a full reset, similar to the $\overline{RES}$ reset is implemented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

Reset Functions

There are five ways in which a microcontroller 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 pins will be first set to inputs.



Note: t_{RSTD} is power-on delay, typical time=50ms

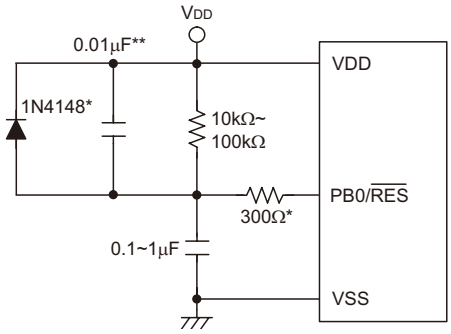
Power-on Reset Timing Chart

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

As the reset pin is shared with PB.0, the reset function must be selected using a configuration option. Although the microcontroller 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 microcontroller 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 microcontroller 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.



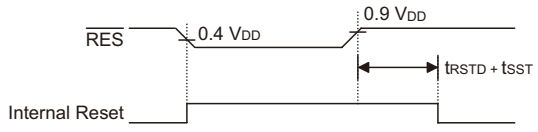
Note: * It is recommended that this component is added for added ESD protection.

** It is recommended that this component is added in environments where power line noise is significant.

Extern $\overline{\text{RES}}$ Circuit

More information regarding external reset circuits is located in Application Note HA0075E on the Holtek website.

Pulling the $\overline{\text{RES}}$ 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.

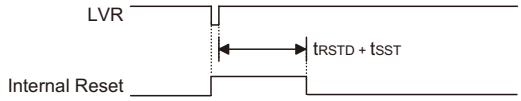


Note: t_{RSTD} is power-on delay, typical time=16.7ms

$\overline{\text{RES}}$ 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. The LVR function is always enabled with a specific LVR voltage, V_{LVR}. If the supply voltage of the device drops to within a range of 0.9V~V_{LVR} such as might occur when changing the battery, the LVR will automatically reset the device internally and the LVRF bit in the SMOD1 register will also be set to 1. For a valid LVR signal, a low supply voltage, i.e., a voltage in the range between 0.9V~V_{LVR} must exist for a time greater than that specified by t_{LVR} in the A.C. characteristics. If the low supply voltage state does not exceed this value, the LVR will ignore the low supply voltage and will not perform a reset function. The actual V_{LVR} value can be selected by the LVS bits in the LVRC register. If the LVS7~LVS0 bits have any other value, which may perhaps occur due to adverse environmental conditions such as noise, the LVR will reset the device after 2~3 f_{SUB} clock cycles. When this happens, the LRF bit in the SMOD1 register will be set to 1. After power on the register will have the value of 01010101B. Note that the LVR function will be automatically disabled when the device enters the power down mode.



Note: t_{RSTD} is power-on delay, typical time=50ms

Low Voltage Reset Timing Chart

- **LVRC Register**

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

Bit 7~0 **LVS7~LVS0:** LVR voltage select
01010101: 2.1V
00110011: 2.55V
10011001: 3.15V
10101010: 3.8V
Any other values: Generates MCU reset – register is reset to POR value

When an actual low voltage condition occurs, as specified by one of the four defined LVR voltage values above, an MCU reset will be generated. The reset operation will be activated after 2~3 f_{SUB} clock cycles. In this situation the register contents will remain the same after such a reset occurs.

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

• **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"

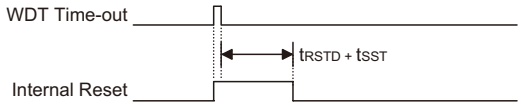
Bit 2 **LVRF:** LVR function reset flag
0: Not occurred
1: Occurred
This bit is set to 1 when a specific Low Voltage Reset situation condition occurs. This bit can only be cleared to 0 by the application program.

Bit 1 **LRF:** LVR Control register software reset flag
0: Not occurred
1: Occurred
This bit is set to 1 if the LVRC register contains any non defined LVR voltage register values. This in effect acts like a software-reset function. This bit can only be cleared to 0 by the application program.

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

Watchdog Time-out Reset during Normal Operation

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

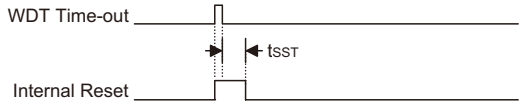


Note: t_{RSTD} is power-on delay, typical time=16.7ms

WDT Time-out Reset during Normal Operation Timing Chart

Watchdog Time-out Reset during SLEEP or IDLE Mode

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



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.

WDT Time-out Reset during SLEEP or IDLE Timing Chart

Reset Initial Conditions

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

TO	PDF	RESET Conditions
0	0	Power-on reset
u	u	$\overline{\text{RES}}$ 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

"u" stands for unchanged

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

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

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

Register Reset Status Table

Register	HT66F60A	HT66F70A	Power-on Reset	$\overline{\text{RES}}$ or LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
IAR0	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP0	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
IAR1	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1L	●	●	0000 0000	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1H	●	●	0000 0000	uuuu uuuu	uuuu uuuu	uuuu uuuu
IAR2	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP2L	●	●	0000 0000	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP2H	●	●	0000 0000	uuuu uuuu	uuuu uuuu	uuuu uuuu
ACC	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	●	●	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	●		--xx xxxx	--uu uuuu	--uu uuuu	--uu uuuu
TBHP		●	-xxx xxxx	-uuu uuuu	-uuu uuuu	-uuu uuuu
STATUS	●	●	xx00 xxxx	uuuu uuuu	uu1u uuuu	uu11 uuuu
BP	●		---- ---0	---- ---0	---- ---0	---- ---u
BP		●	---- --00	---- --00	---- --00	---- --uu

Register	HT66F60A	HT66F70A	Power-on Reset	$\overline{RES}$ or LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
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	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PC	•	•	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCC	•	•	1111 1111	1111 1111	1111 1111	uuuu 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	•	•	-000 0000	-000 0000	-000 0000	-uuu uuuu
PF	•	•	-111 1111	-111 1111	-111 1111	-uuu uuuu
PFC	•	•	-111 1111	-111 1111	-111 1111	-uuu uuuu
PGPU	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PG	•	•	1111 1111	1111 1111	1111 1111	uuuu uuuu
PGC	•	•	1111 1111	1111 1111	1111 1111	uuuu uuuu
PHPU	•	•	--00 0000	--00 0000	--00 0000	--uu uuuu
PH	•	•	--11 1111	--11 1111	--11 1111	--uu uuuu
PHC	•	•	--11 1111	--11 1111	--11 1111	--uu uuuu
INTEG	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC0	•	•	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC2	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC3	•	•	-000 -000	-000 -000	-000 -000	-uuu -uuu
MFI0	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI1	•	•	-000 -000	-000 -000	-000 -000	-uuu -uuu
MFI2	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI3	•	•	-000 -000	-000 -000	-000 -000	-uuu -uuu
MFI4	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
SMOD	•	•	0000 0011	0000 0011	0000 0011	uuuu uuuu
SMOD1	•	•	0--- -x00	0--- -1uu	0--- -uuu	u--- -uuu
SMOD2	•	•	---- ---0	---- ---0	---- ---0	---- ---u
LVRC	•	•	0101 0101	uuuu uuuu	0101 0101	uuuu uuuu
LVDC	•	•	--00 -000	--00 -000	--00 -000	--uu -uuu
WDTC	•	•	0101 0011	0101 0011	0101 0011	uuuu uuuu
EEA	•	•	-000 0000	-000 0000	-000 0000	-uuu uuuu
EED	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
CP0C	•	•	-000 ---1	-000 ---1	-000 ---1	-uuu ---u
CP1C	•	•	-000 ---1	-000 ---1	-000 ---1	-uuu ---u

Register	HT66F60A	HT66F70A	Power-on Reset	$\overline{RES}$ or LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
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
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
TM0C0	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TMnC1	●	●	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
PSC0	●	●	---- --00	---- --00	---- --00	---- --uu
TBC0	●	●	0--- -000	0--- -000	0--- -000	u--- -uuu
TBC1	●	●	0--- -000	0--- -000	0--- -000	u--- -uuu
PSC1	●	●	---- --00	---- --00	---- --00	---- --uu
ADCR0	●	●	0110 0000	0110 0000	0110 0000	uuuu uuuu
ADCR1	●	●	-000 -000	-000 -000	-000 -000	-uuu -uuu
ADRL (ADRFS=0)	●	●	xxxx ----	xxxx ----	xxxx ----	uuuu ----
ADRL (ADRFS=1)	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRH (ADRFS=0)	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRH (ADRFS=1)	●	●	---- xxxx	---- xxxx	---- xxxx	---- uuuu
SIMC0	●	●	111- 000-	111- 000-	111- 000-	uuu- uuu-
SIMC1	●	●	1000 0001	1000 0001	1000 0001	uuuu uuuu
SIMD	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu

Register	HT66F60A	HT66F70A	Power-on Reset	$\overline{RES}$ or LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
SIMC2/ SIMA	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
I2CTOC	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
SPIAC0	●	●	111- --0-	111- --0-	111- --0-	uuu- --u-
SPIAC1	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
SPIAD	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARL	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARH	●		--00 0000	--00 0000	--00 0000	--uu uuuu
FARH		●	-000 0000	-000 0000	-000 0000	-uuu 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
TBC2	●	●	0--- -000	0--- -000	0--- -000	u--- -uuu
SCOMC	●	●	0000 ----	0000 ----	0000 ----	uuuu ----
EEC	●	●	---- 0000	---- 0000	---- 0000	---- uuuu
FC0	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC1	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC2	●	●	---- ---0	---- ---0	---- ---0	---- ---u
IFS0	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS1	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS2	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS3	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS4	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS5	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM4C0	●	●	0000 0---	0000 0---	0000 0---	uuuu u---
TM4C1	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM4DL	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM4DH	●	●	---- --00	---- --00	---- --00	---- --uu
TM4AL	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM4AH	●	●	---- --00	---- --00	---- --00	---- --uu
TM4RP	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM5C0	●	●	0000 0---	0000 0---	0000 0---	uuuu u---
TM5C1	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM5DL	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM5DH	●	●	---- --00	---- --00	---- --00	---- --uu
TM5AL	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM5AH	●	●	---- --00	---- --00	---- --00	---- --uu
TM5RP	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS0	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS1	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	HT66F60A	HT66F70A	Power-on Reset	$\overline{RES}$ or LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
PAS2	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS3	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBS2	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBS3	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCS0	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCS1	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCS2	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCS3	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDS0	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDS1	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDS2	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDS3	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PES0	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PES1	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PES2	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PES3	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PFS0	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PGS0	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PGS1	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PGS2	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PGS3	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PHS0	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PHS1	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu
PHS2	•	•	0000 0000	0000 0000	0000 0000	uuuu uuuu

Note: "-" not implement
"u" means "unchanged"
"x" means "unknown"

Input/Output Ports

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

These 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.

I/O Port Register List

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	PCPU7	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PC	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	PCC7	PCC6	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
PE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
PEC	PEC7	PEC6	PEC5	PEC4	PEC3	PEC2	PEC1	PEC0
PFPU	—	PFPU6	PFPU5	PFPU4	PFPU3	PFPU2	PFPU1	PFPU0
PF	—	PF6	PF5	PF4	PF3	PF2	PF1	PF0
PFC	—	PFC6	PFC5	PFC4	PFC3	PFC2	PFC1	PFC0
PGPU	PGPU7	PGPU6	PGPU5	PGPU4	PGPU3	PGPU2	PGPU1	PGPU0
PG	PG7	PG6	PG5	PG4	PG3	PG2	PG1	PG0
PGC	PGC7	PGC6	PGC5	PGC4	PGC3	PGC2	PGC1	PGC0
PHPU	—	—	PHPU5	PHPU4	PHPU3	PHPU2	PHPU1	PHPU0
PH	—	—	PH5	PH4	PH3	PH2	PH1	PH0
PHC	—	—	PHC5	PHC4	PHC3	PHC2	PHC1	PHC0

"—": Unimplemented, read as "0"

PAWUn: PA wake-up function control
0: Disable
1: Enable

PAn/PBn/PCn/PDn/PEn/PFn/PGn/PHn: I/O Data bit
0: data 0
1: data 1

PACn/PBCn/PCCn/PDCn/PECn/PFCn/PGCn/PHCn: I/O type selection
0: Output
1: input

PAPUn/PBPUn/PCPUn/PDPUn/PEPUn/PFPUn/PGPUn/PHPUn: Pull-high function control
0: Disable
1: Enable

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.

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

I/O Port Control Registers

Each Port has its own control register, known as PAC~PHC, which controls the input/output configuration. With this control register, each I/O pin with or without pull-high resistors can be reconfigured dynamically under software control. 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.

Pin-shared Functions

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

Pin-shared Function Selection Register

The limited number of supplied pins in a package can impose restrictions on the amount of functions a certain device can contain. However by allowing the same pins to share several different functions and providing a means of function selection, a wide range of different functions can be incorporated into even relatively small package sizes. The device includes Port "x" output function slsection register "n", labeled as PxSn, and input function selection register "i", labeled as IFSi, which can

select the functions of the pin-shared function pins.

When the pin-shared input function is selected to be used, the corresponding input and output functions selection should be properly managed. For example, if the I²C SDA line is used, the corresponding output pin-shared function should be configured as the SDI/SDA function by configuring the PxSn register and the SDA signal intput should be properly selected using the IFSi register. However, if the external interrupt function is selected to be used, the relevant output pin-shared function should be selected as an I/O function and the interrupt input signal should be selected.

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAS0	PA1S3	PA1S2	PA1S1	PA1S0	D3	D2	D1	D0
PAS1	PA3S3	PA3S2	PA3S1	PA3S0	D3	D2	D1	D0
PAS2	PA5S3	PA5S2	PA5S1	PA5S0	PA4S3	PA4S2	PA4S1	PA4S0
PAS3	PA7S3	PA7S2	PA7S1	PA7S0	PA6S3	PA6S2	PA6S1	PA6S0
PBS2	PB5S3	PB5S2	PB5S1	PB5S0	D3	D2	D1	D0
PBS3	PB7S3	PB7S2	PB7S1	PB7S0	PB6S3	PB6S2	PB6S1	PB6S0
PCS0	PC1S3	PC1S2	PC1S1	PC1S0	PC0S3	PC0S2	PC0S1	PC0S0
PCS1	PC3S3	PC3S2	PC3S1	PC3S0	PC2S3	PC2S2	PC2S1	PC2S0
PCS2	PC5S3	PC5S2	PC5S1	PC5S0	PC4S3	PC4S2	PC4S1	PC4S0
PCS3	PC7S3	PC7S2	PC7S1	PC7S0	PC6S3	PC6S2	PC6S1	PC6S0
PDS0	PD1S3	PD1S2	PD1S1	PD1S0	PD0S3	PD0S2	PD0S1	PD0S0
PDS1	PD3S3	PD3S2	PD3S1	PD3S0	PD2S3	PD2S2	PD2S1	PD2S0
PDS2	PD5S3	PD5S2	PD5S1	PD5S0	PD4S3	PD4S2	PD4S1	PD4S0
PDS3	PD7S3	PD7S2	PD7S1	PD7S0	PD6S3	PD6S2	PD6S1	PD6S0
PES0	PE1S3	PE1S2	PE1S1	PE1S0	PE0S3	PE0S2	PE0S1	PE0S0
PES1	PE3S3	PE3S2	PE3S1	PE3S0	D3	D2	D1	D0
PES2	PE5S3	PE5S2	PE5S1	PE5S0	PE4S3	PE4S2	PE4S1	PE4S0
PES3	PE7S3	PD7S2	PE7S1	PE7S0	PE6S3	PE6S2	PE6S1	PE6S0
PFS0	PF1S3	PF1S2	PF1S1	PF1S0	PF0S3	PF0S2	PF0S1	PF0S0
PGS0	PG1S3	PG1S2	PG1S1	PG1S0	PG0S3	PG0S2	PG0S1	PG0S0
PGS1	PG3S3	PG3S2	PG3S1	PG3S0	D3	D2	D1	D0
PGS2	D7	D6	D5	D4	PG4S3	PG4S2	PG4S1	PG4S0
PGS3	PG7S3	PG7S2	PG7S1	PG7S0	PG6S3	PG6S2	PG6S1	PG6S0
PHS0	PH1S3	PH1S2	PH1S1	PH1S0	PH0S3	PH0S2	PH0S1	PH0S0
PHS1	PH3S3	PH3S2	PH3S1	PH3S0	PH2S3	PH2S2	PH2S1	PH2S0
PHS2	PH5S3	PH5S2	PH5S1	PH5S0	D3	D2	D1	D0
IFS0	PINTBS1	PINTBS0	INT2S1	INT2S0	INT1S1	INT1S0	INT0S1	INT0S0
IFS1	TCK3S1	TCK3S0	TCK2S1	TCK2S0	TCK1S1	TCK1S0	TCK0S1	TCK0S0
IFS2	TP2IS1	TP2IS0	TP1IBS1	TP1IBS0	TP1IAS1	TP1IAS0	D1	D0
IFS3	D7	D6	TP5IS1	TP5IS0	TP4IS1	TP4IS0	D1	D0
IFS4	D7	D6	SDIS1	SDIS0	SCKS1	SCKS0	SCSBS1	SCSBS0
IFS5	D7	D6	SDIAS1	SDIAS0	SCKAS1	SCKAS0	SCSABS1	SCSABS0

• **PAS0**

Bit	7	6	5	4	3	2	1	0
Name	PA1S3	PA1S2	PA1S1	PA1S0	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 7~4 **PA1S3~PA1S0:** Port A1 Function Selection
0000: I/O
0001: TP1A
0011: AN1
Others: Reserved

Bit 3~0 Reserved bits, can be read and written

• **PAS1**

Bit	7	6	5	4	3	2	1	0
Name	PA3S3	PA3S2	PA3S1	PA3S0	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 7~4 **PA3S3~PA3S0:** Port A3 Function Selection
0000: I/O
0011: AN3
0111: C0N
1111: AN3 and C0N
Others: Reserved

Bit 3~0 Reserved bits, can be read and written

• **PAS2**

Bit	7	6	5	4	3	2	1	0
Name	PA5S3	PA5S2	PA5S1	PA5S0	PA4S3	PA4S2	PA4S1	PA4S0
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 **PA5S3~PA5S0:** Port A5 Function Selection
0000: I/O
0001: SDO
0010: C1X
0011: AN5
Others: Reserved

Bit 3~0 **PA4S3~PA4S0:** Port A4 Function Selection
0000: I/O
0011: AN4
Others: Reserved

• **PAS3**

Bit	7	6	5	4	3	2	1	0
Name	PA7S3	PA7S2	PA7S1	PA7S0	PA6S3	PA6S2	PA6S1	PA6S0
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 **PA7S3~PA7S0:** Port A7 Function Selection
0000: I/O
0010: SCK/SCL
0011: AN7
Others: Reserved

Bit 3~0 **PA6S3~PA6S0:** Port A6 Function Selection
0000: I/O
0010: SDI/SDA
0011: AN6
Others: Reserved

• **PBS2**

Bit	7	6	5	4	3	2	1	0
Name	PB5S3	PB5S2	PB5S1	PB5S0	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 7~4 **PB5S3~PB5S0:** Port B5 Function Selection
0000: I/O
0001: $\overline{SCS}$
Others: Reserved

Bit 3~0 Reserved bits, can be read and written

• **PBS3**

Bit	7	6	5	4	3	2	1	0
Name	PB7S3	PB7S2	PB7S1	PB7S0	PB6S3	PB6S2	PB6S1	PB6S0
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 **PB7S3~PB7S0:** Port B7 Function Selection
0000: I/O
0010: SDI/SDA
Others: Reserved

Bit 3~0 **PB6S3~PB6S0:** Port B6 Function Selection
0000: I/O
0001: SDO
Others: Reserved

• **PCS0**

Bit	7	6	5	4	3	2	1	0
Name	PC1S3	PC1S2	PC1S1	PC1S0	PC0S3	PC0S2	PC0S1	PC0S0
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 **PC1S3~PC1S0:** Port C1 Function Selection
0000: I/O
0001: TP1B
0010: TP1BB
0011: SCOM1
Others: Reserved

Bit 3~0 **PC0S3~PC0S0:** Port C0 Function Selection
0000: I/O
0001: TP1B
0010: TP1BB
0011: SCOM0
Others: Reserved

• **PCS1**

Bit	7	6	5	4	3	2	1	0
Name	PC3S3	PC3S2	PC3S1	PC3S0	PC2S3	PC2S2	PC2S1	PC2S0
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 **PC3S3~PC3S0:** Port C3 Function Selection
0000: I/O
0001: TP2
0010: C1X
0100: TP1B
Others: Reserved

Bit 3~0 **PC2S3~PC2S0:** Port C2 Function Selection
0000: I/O
0001: PCK
0010: C0X
Others: Reserved

• **PCS2**

Bit	7	6	5	4	3	2	1	0
Name	PC5S3	PC5S2	PC5S1	PC5S0	PC4S3	PC4S2	PC4S1	PC4S0
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 **PC5S3~PC5S0:** Port C5 Function Selection
0000: I/O
0001: PCK
0010: TP0
0100: TP1B
0101: TP0B
0110: TP1BB
Others: Reserved

Bit 3~0 **PC4S3~PC4S0:** Port C4 Function Selection
0000: I/O
0001: TP2
0010: TP2B
Others: Reserved

• **PCS3**

Bit	7	6	5	4	3	2	1	0
Name	PC7S3	PC7S2	PC7S1	PC7S0	PC6S3	PC6S2	PC6S1	PC6S0
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 **PC7S3~PC7S0:** Port C7 Function Selection
0000: I/O
0001: TP1A
0011: SCOM3
Others: Reserved

Bit 3~0 **PC6S3~PC6S0:** Port C6 Function Selection
0000: I/O
0001: TP0
0010: TP0B
0011: SCOM2
Others: Reserved

• **PDS0**

Bit	7	6	5	4	3	2	1	0
Name	PD1S3	PD1S2	PD1S1	PD1S0	PD0S3	PD0S2	PD0S1	PD0S0
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 **PD1S3~PD1S0:** Port D1 Function Selection
0000: I/O
0001: TP2
0010: SCK/SCL
0100: SDO
0101: TP2B
Others: Reserved

Bit 3~0 **PD0S3~PD0S0:** Port D0 Function Selection
0000: I/O
0001: TP3
0010: $\overline{SCS}$
0100: TP3B
Others: Reserved

• **PDS1**

Bit	7	6	5	4	3	2	1	0
Name	PD3S3	PD3S2	PD3S1	PD3S0	PD2S3	PD2S2	PD2S1	PD2S0
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 **PD3S3~PD3S0:** Port D3 Function Selection
0000: I/O
0001: TP3
0010: SCK/SCL
0100: SDO
0101: TP3B
Others: Reserved

Bit 3~0 **PD2S3~PD2S0:** Port D2 Function Selection
0000: I/O
0010: SDI/SDA
Others: Reserved

• **PDS2**

Bit	7	6	5	4	3	2	1	0
Name	PD5S3	PD5S2	PD5S1	PD5S0	PD4S3	PD4S2	PD4S1	PD4S0
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 **PD5S3~PD5S0:** Port D5 Function Selection
0000: I/O
0001: TP0
0010: TP0B
Others: Reserved

Bit 3~0 **PD4S3~PD4S0:** Port D4 Function Selection
0000: I/O
0001: TP2
0010: TP2B
Others: Reserved

• **PDS3**

Bit	7	6	5	4	3	2	1	0
Name	PD7S3	PD7S2	PD7S1	PD7S0	PD6S3	PD6S2	PD6S1	PD6S0
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 **PD7S3~PD7S0:** Port D7 Function Selection
0000: I/O
0001: $\overline{\text{SCS}}$
Others: Reserved

Bit 3~0 **PD6S3~PD6S0:** Port D6 Function Selection
0000: I/O
0010: SCK/SCL
Others: Reserved

• **PES0**

Bit	7	6	5	4	3	2	1	0
Name	PE1S3	PE1S2	PE1S1	PE1S0	PE0S3	PE0S2	PE0S1	PE0S0
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 **PE1S3~PE1S0:** Port E1 Function Selection
0000: I/O
0001: SCKA
Others: Reserved

Bit 3~0 **PE0S3~PE0S0:** Port E0 Function Selection
0000: I/O
0001: $\overline{\text{SCSA}}$
Others: Reserved

- **PES1**

Bit	7	6	5	4	3	2	1	0
Name	PE3S3	PE3S2	PE3S1	PE3S0	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 7~4 **PE3S3~PE3S0:** Port E3 Function Selection
0000: I/O
0001: SDOA
Others: Reserved

Bit 3~0 Reserved bits, can be read and written.

- **PES2**

Bit	7	6	5	4	3	2	1	0
Name	PE5S3	PE5S2	PE5S1	PE5S0	PE4S3	PE4S2	PE4S1	PE4S0
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 **PE5S3~PE5S0:** Port E5 Function Selection
0000: I/O
0001: TP3
0010: TP3B
Others: Reserved

Bit 3~0 **PE4S3~PE4S0:** Port E4 Function Selection
0000: I/O
0001: TP1B
0010: TP1BB
Others: Reserved

- **PES3**

Bit	7	6	5	4	3	2	1	0
Name	PE7S3	PE7S2	PE7S1	PE7S0	PE6S3	PE6S2	PE6S1	PE6S0
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 **PE7S3~PE7S0:** Port E7 Function Selection
0000: I/O
0011: AN9
Others: Reserved

Bit 3~0 **PE6S3~PE6S0:** Port E6 Function Selection
0000: I/O
0011: AN8
Others: Reserved

• **PFS0**

Bit	7	6	5	4	3	2	1	0
Name	PF1S3	PF1S2	PF1S1	PF1S0	PF0S3	PF0S2	PF0S1	PF0S0
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 **PF1S3~PF1S0:** Port F1 Function Selection
0000: I/O
0011: AN11
0111: C1P
1111: AN11 and C1P
Others: Reserved

Bit 3~0 **PF0S3~PF0S0:** Port F0 Function Selection
0000: I/O
0011: AN10
0111:C1N
1111: AN10 and C1N
Others: Reserved

• **PGS0**

Bit	7	6	5	4	3	2	1	0
Name	PG1S3	PG1S2	PG1S1	PG1S0	PG0S3	PG0S2	PG0S1	PG0S0
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 **PG1S3~PG1S0:** Port G1 Function Selection
0000: I/O
0001: C1X
Others: Reserved

Bit 3~0 **PG0S3~PG0S0:** Port G0 Function Selection
0000: I/O
0001:C0X
Others: Reserved

• **PGS1**

Bit	7	6	5	4	3	2	1	0
Name	PG3S3	PG3S2	PG3S1	PG3S0	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 7~4 **PG3S3~PG3S0:** Port G3 Function Selection
0000: I/O
0001: TP4
0010: TP4B
Others: Reserved

Bit 3~0 Reserved bits, can be read and written.

• **PGS2**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	PG4S3	PG4S2	PG4S1	PG4S0
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 Reserved bits, can be read and written.

Bit 3~0 **PG4S3~PG4S0:** Port G4 Function Selection
0000: I/O
0001: TP4
0010: TP4B
Others: Reserved

• **PGS3**

Bit	7	6	5	4	3	2	1	0
Name	PG7S3	PG7S2	PG7S1	PG7S0	PG6S3	PG6S2	PG6S1	PG6S0
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 **PG7S3~PG7S0:** Port G7 Function Selection
0000: I/O
0001: TP5
0010: TP5B
Others: Reserved

Bit 3~0 **PG6S3~PG6S0:** Port G6 Function Selection
0000: I/O
0001: TP5
0010: TP5B
Others: Reserved

• **PHS0**

Bit	7	6	5	4	3	2	1	0
Name	PH1S3	PH1S2	PH1S1	PH1S0	PH0S3	PH0S2	PH0S1	PH0S0
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 **PH1S3~PH1S0:** Port H1 Function Selection
0000: I/O
0011: AN2
0111: C0P
1111: AN2 and C0P
Others: Reserved

Bit 3~0 **PH0S3~PH0S0:** Port H0 Function Selection
0000: I/O
0001: TP0
0010: C0X
0011: AN0/VREF
0100: TP0B
Others: Reserved

• **PHS1**

Bit	7	6	5	4	3	2	1	0
Name	PH3S3	PH3S2	PH3S1	PH3S0	PH2S3	PH2S2	PH2S1	PH2S0
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 **PH3S3~PH3S0:** Port H3 Function Selection
0000: I/O
0001: SCKA
Others: Reserved

Bit 3~0 **PH2S3~PH2S0:** Port H2 Function Selection
0000: I/O
0001: $\overline{\text{SCSA}}$
Others: Reserved

• **PHS2**

Bit	7	6	5	4	3	2	1	0
Name	PH5S3	PH5S2	PH5S1	PH5S0	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 7~4 **PH5S3~PH5S0:** Port H5 Function Selection
0000: I/O
0001: SDOA
Others: Reserved

Bit 3~0 Reserved bits, can be read and written.

• **IFS0**

Bit	7	6	5	4	3	2	1	0
Name	PINTBS1	PINTBS0	INT2S1	INT2S0	INT1S1	INT1S0	INT0S1	INT0S0
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 **PINTBS1~PINTBS0:** $\overline{\text{PINT}}$ input source pin selection
00: PC3
Others: PC4

Bit 5~4 **INT2S1~INT2S0:** INT2 input source pin selection
00: PC4
Others: PE2

Bit 3~2 **INT1S1~INT1S0:** INT1 input source pin selection
00: PA4
01: PC5
10: PE1
11: PE7

Bit 1~0 **INT0S1~INT0S0:** INT0 input source pin selection
00: PA3
01: PC4
10: PE0
11: PE6

• **IFS1**

Bit	7	6	5	4	3	2	1	0
Name	TCK3S1	TCK3S0	TCK2S1	TCK2S0	TCK1S1	TCK1S0	TCK0S1	TCK0S0
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 **TCK3S1~TCK3S0:** TCK3 input source pin selection
00: PC4
Others: PE3

Bit 5~4 **TCK2S1~TCK2S0:** TCK2 input source pin selection
00: PC2
Others: PD0

Bit 3~2 **TCK1S1~TCK1S0:** TCK1 input source pin selection
00: PA4
Others: PD3

Bit 1~0 **TCK0S1~TCK0S0:** TCK0 input source pin selection
00: PH1
Others: PD2

• **IFS2**

Bit	7	6	5	4	3	2	1	0
Name	TP2IS1	TP2IS0	TP1IBS1	TP1IBS0	TP1IAS1	TP1IAS0	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 7~6 **TP2IS1~TP2IS0:** TP2I input source pin selection
00: PC3
01: PC4
10: PD1
11: PD4

Bit 5~4 **TP1IBS1~TP1IBS0:** TP1IB input source pin selection
00: PC0
01: PC1
10: PC5
11: PE4

Bit 3~2 **TP1IAS1~TP1IAS0:** TP1IA input source pin selection
00: PA1
Others: PC7

Bit 1~0 Reserved bits, can be read and written.

• **IFS3**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	TP5IS1	TP5IS0	TP4IS1	TP4IS0	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 7~6 Reserved bits, can be read and written.

Bit 5~4 **TP5IS1~TP5IS0:** TP5I input source pin selection
00: PG6
Others: PG7

Bit 3~2 **TP4IS1~TP4IS0:** TP4I input source pin selection
00: PG3
Others: PG4

Bit 1~0 Reserved bits, can be read and written.

• **IFS4**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	SDIS1	SDIS0	SCKS1	SCKS0	SCSBS1	SCSBS0
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 Reserved bits, can be read and written.

Bit 5~4 **SDIS1~SDIS0:** SDI/SDA input source pin selection
00: PA6
01: PB7
Others: PD2

Bit 3~2 **SCKS1~SCKS0:** SCK/SCL input source pin selection
00: PA7
01: PD3
10: PD1
11: PD6

Bit 1~0 **SCSBS1~SCSBS0:** $\overline{\text{SCS}}$ input source pin selection
00: PB5
01: PD0
Others: PD7

• **IFS5**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	SDIAS1	SDIAS0	SCKAS1	SCKAS0	SCSABS1	SCSABS0
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 Reserved bits, can be read and written.

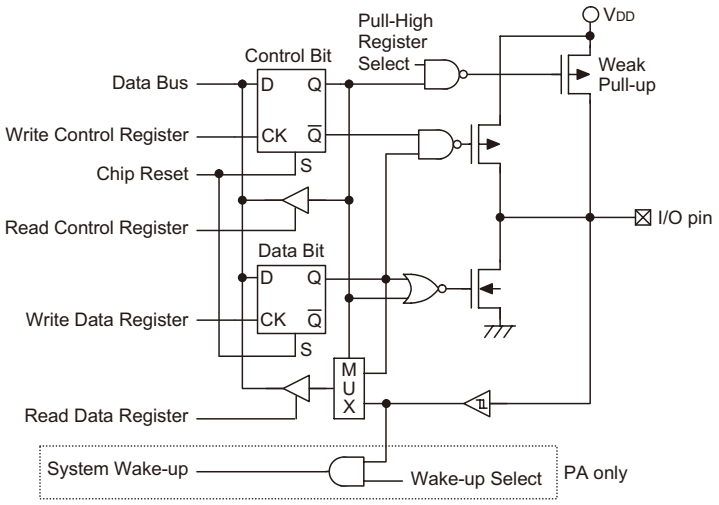
Bit 5~4 **SDIAS1~SDIAS0:** SDIA input source pin selection
00: PE2
Others: PH4

Bit 3~2 **SCKAS1~SCKAS0:** SCKA input source pin selection
00: PE1
Others: PH3

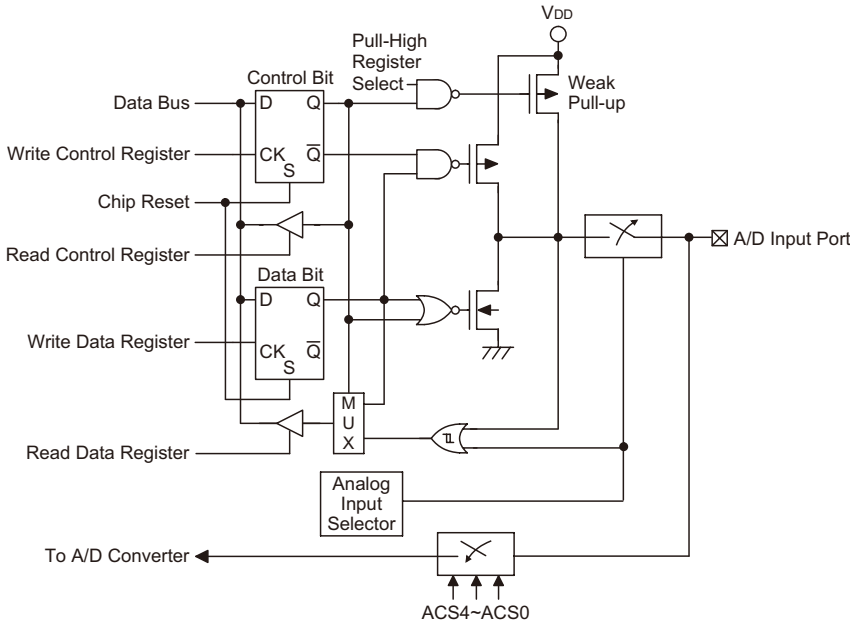
Bit 1~0 **SCSABS1~SCSABS0:** $\overline{\text{SCSA}}$ input source pin selection
00: PE0
Others: PH2

I/O Pin Structures

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



Generic Input/Output Structure



A/D Input/Output Structure

Programming Considerations

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

Port A has the additional capability of providing wake-up 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 devices 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 multiple interrupts. The addition of input and output pins for each TM ensures that users are provided with timing units with a wide and flexible range of features.

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

Introduction

The devices contain from two to six TMs depending upon which device is selected with each TM having a reference name of TM0~TM5. 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.

TM 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

Each device in the series contains a specific number of either Compact Type, Standard Type and Enhanced Type TM unit which are shown in the table together with their individual reference name, TM0~TM5.

Device	TM0	TM1	TM2	TM3	TM4	TM5
HT66F60A	10-bit CTM	10-bit ETM	16-bit STM	10-bit CTM	16-bit STM	16-bit STM
HT66F70A						

TM Name/Type Reference

TM Operation

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

TM Clock Source

The clock source which drives the main counter in each TM can originate from various sources. The selection of the required clock source is implemented using the TnCK2~TnCK0 bits in the TMn control registers. The clock source can be a ratio of either the system clock f_{SYS} or the internal high clock f_H, the f_{SUB} clock source or the external TCKn pin. 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 type TM has 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 two TM input pins, with the label TCKn and TPnI respectively. The TM input pin, TCKn, 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 TCKn pin is also used as the external trigger input pin in single pulse mode for the STM and ETM.

The other TM input pin, TPnI, is the capture input whose active edge can be a rising edge, a falling edge or both rising and falling edges and the active edge transition type is selected using the TnIO1 and TnIO0 bits in the TMnC1 register.

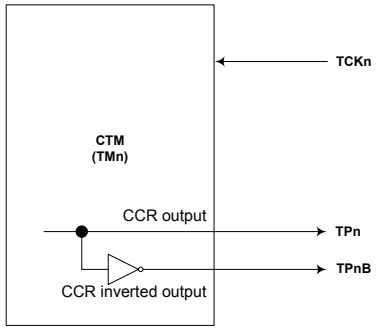
The TMs each have one or more output pins with the label TPn and TPnB respectively. When the TM is in the Compare Match Output Mode, these pins can be controlled by the TM to switch to a high or low level or to toggle when a compare match situation occurs. The external TPn output pin is also the pin 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 registers. The corresponding selection bits in the pin-shared function registers determines if its associated pin is to be used as an external TM output pin or if it is to have another function. The number of output pins for each TM type and device is different, the details are provided in the accompanying table.

Device	CTM	ETM	STM	Registers
HT66F60A HT66F70A	TP0, TP0B, TCK0 TP3, TP3B, TCK3	TP1A, TP1IA, TCK1 TP1B, TP1BB, TP1IB	TP2, TP2B, TP2I, TCK2 TP4, TP4B, TP4I, TCK4 TP5, TP5B, TP5I, TCK5	IFSi

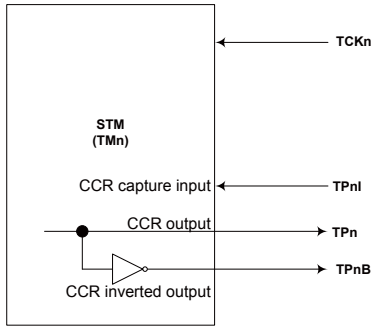
TM Input/Output Pins

TM Input/Output Pin Control

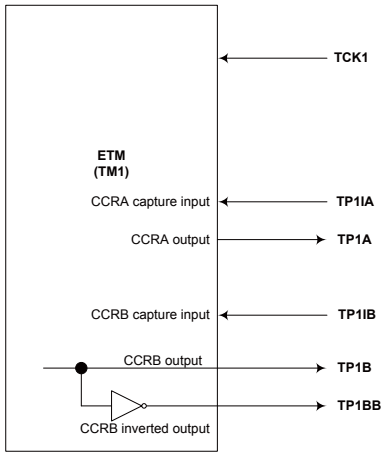
Selecting to have a TM input/output or whether to retain its other shared function is implemented using one or two registers, with the corresponding selection bits in each pin-shared function register corresponding to a TM input/output pin. Configuring the selection bits correctly will setup the corresponding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



CTM Function Pin Control Block Diagram (n=0 or 3)



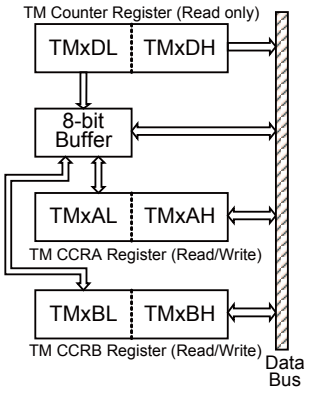
STM Function Pin Control Block Diagram (n=2, 4, 5)



ETM Function Pin Control Block Diagram

Programming Considerations

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



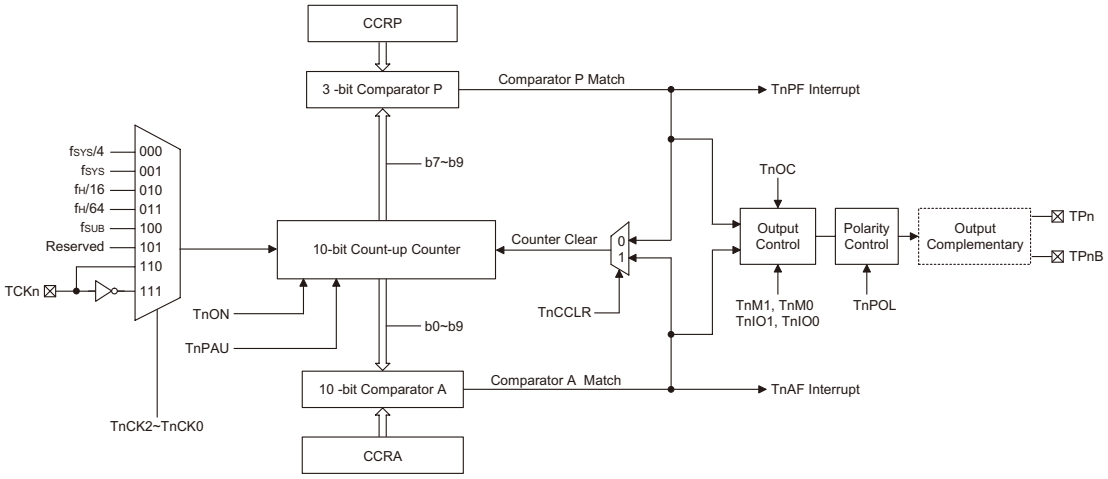
As the CCRA and CCRB registers are implemented in the way shown in the following diagram and accessing these register pairs is carried out in a specific way as described above, it is recommended to use the "MOV" instruction to access the CCRA and CCRB low byte registers, named TMxAL and TMxBL, using 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 two TM types, the Compact TM type still contains three operating modes, which are Compare Match Output, Timer/Event Counter and PWM Output modes. The Compact TM can 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	TM Type	TM Name	TM Input Pin	TM Output Pin
HT66F60A HT66F70A	10-bit CTM	TM0, TM3	TCK0, TP0I; TCK3, TP3I	TP0, TP0B; TP3, TP3B



Compact Type TM Block Diagram (n=0 or 3)

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 Register Description

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

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

Compact TM Register List (n=0 or 3)

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 **TMnDL**: 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 **TMnDH**: 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 **TMnAL**: 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 **TMnAH**: TMn CCRA High Byte Register bit 1 ~ bit 0
TMn 10-bit CCRA bit 9 ~ bit 8

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 TM0 Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: Reserved
110: TCK0 rising edge clock
111: TCK0 falling edge clock

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

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

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

If the TM is in the Compare Match Output Mode then the TM output pin will be reset to its initial condition, as specified by the TnOC bit, when the TnON bit changes from low to high.

Bit 2~0 **TnRP2~TnRP0:** TMn CCRP 3-bit register, compared with the TMn Counter bit 9~bit 7 Comparator P Match Period
000: 1024 TMn clocks
001: 128 TMn clocks
010: 256 TMn clocks
011: 384 TMn clocks
100: 512 TMn clocks
101: 640 TMn clocks
110: 768 TMn clocks
111: 896 TMn clocks

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

TMnC1 Register

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, TPnB 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, TPnB 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 he 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, TPnB Output polarity Control
0: Non-invert
1: Invert

This bit controls the polarity of the TPn or TPnB 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 Comparator P match
1: TMn Comparator 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.

Compact Type TM Operating Modes

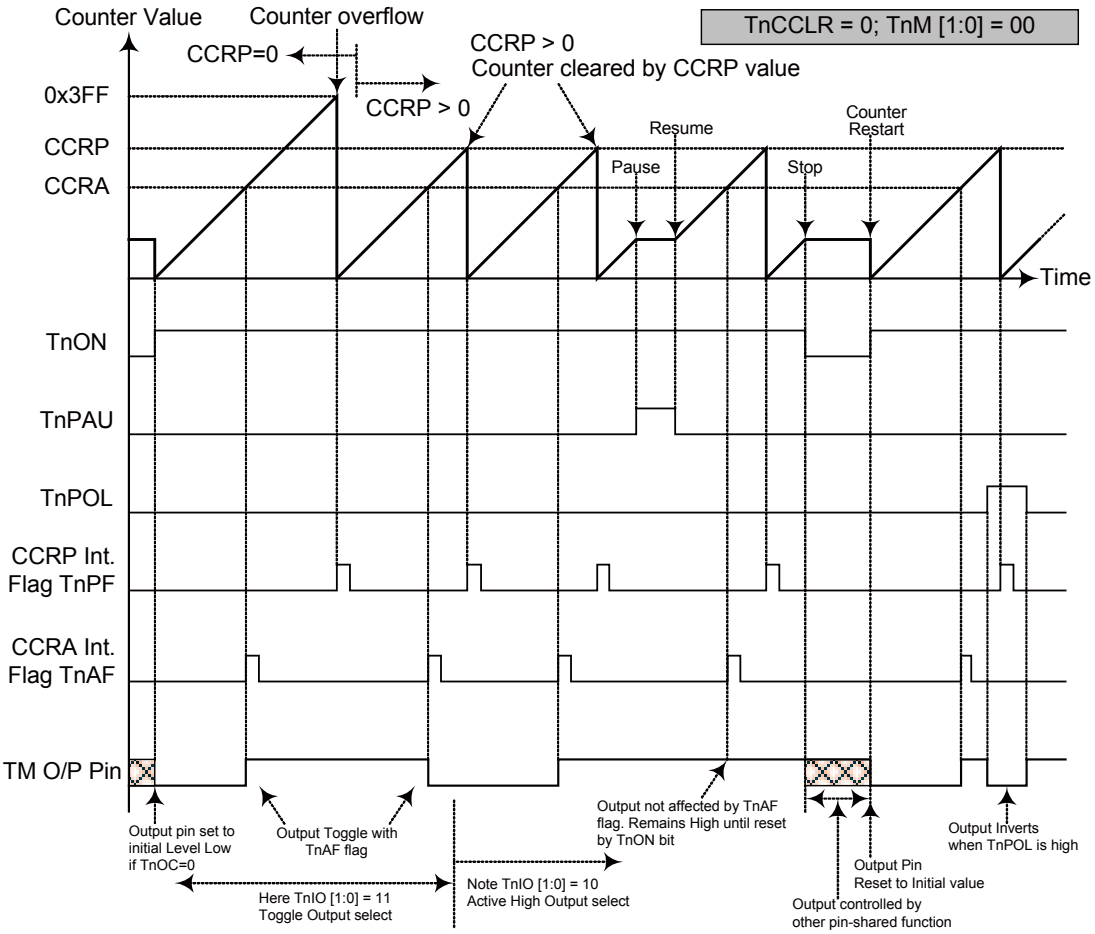
The Compact Type TM can operate in one of three operating modes, Compare Match Output Mode, PWM Mode or Timer/Counter Mode. The operating mode is selected using the TnM1 and TnM0 bits in the TMnC1 register.

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 a TnAF interrupt request flag is generated after a compare match occurs from Comparator A. The TnPF interrupt request flag, generated from a compare match 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.



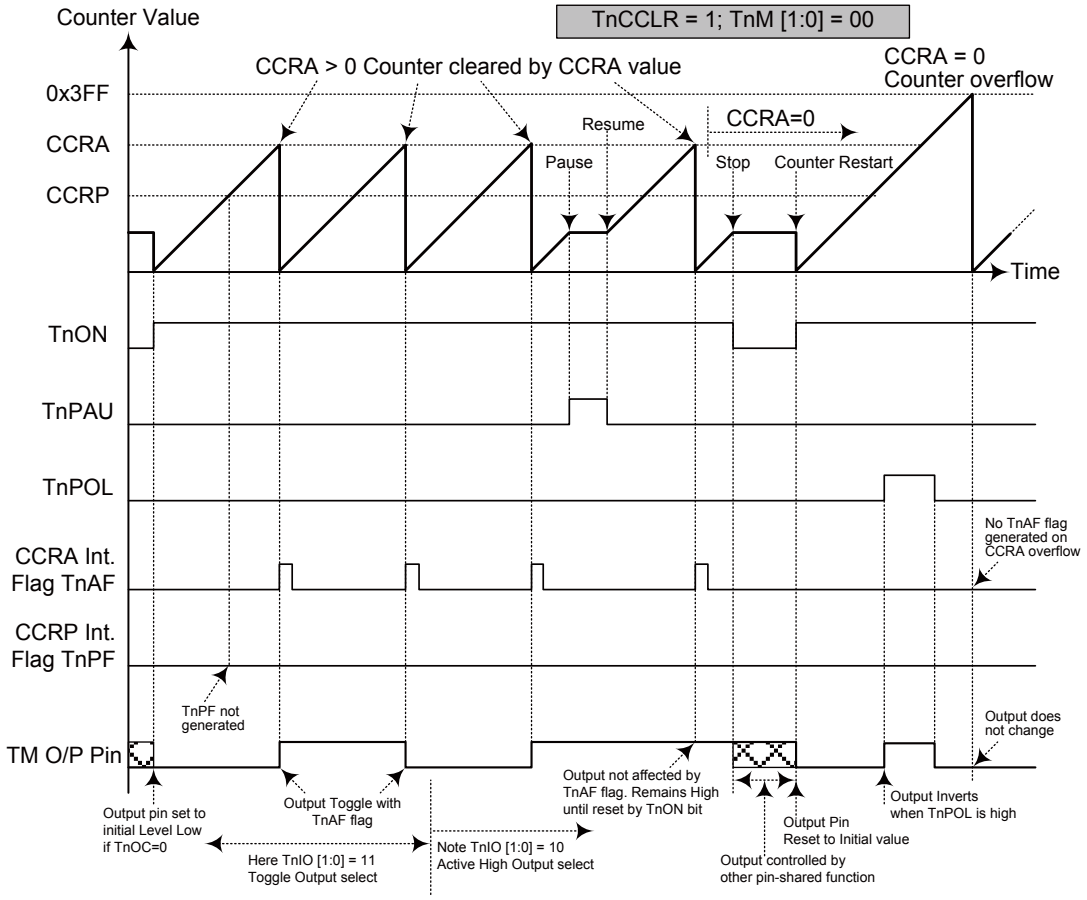
Compare Match Output Mode – TnCCLR=0

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

4. n=0 or 3



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
5. n=0 or 3

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,

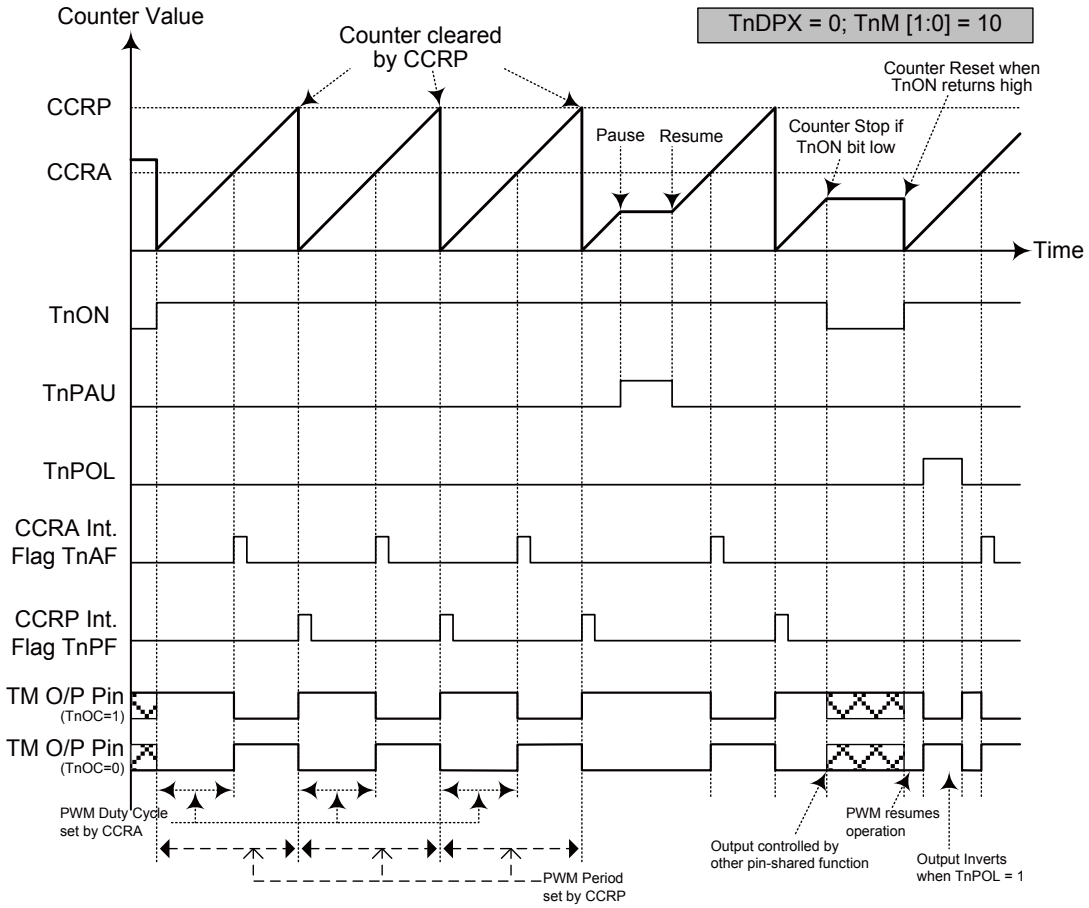
The CTM PWM output frequency=($f_{SYS}/4$)/512=$f_{SYS}/2048$=7.8125kHz, duty=128/512=25%.

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

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

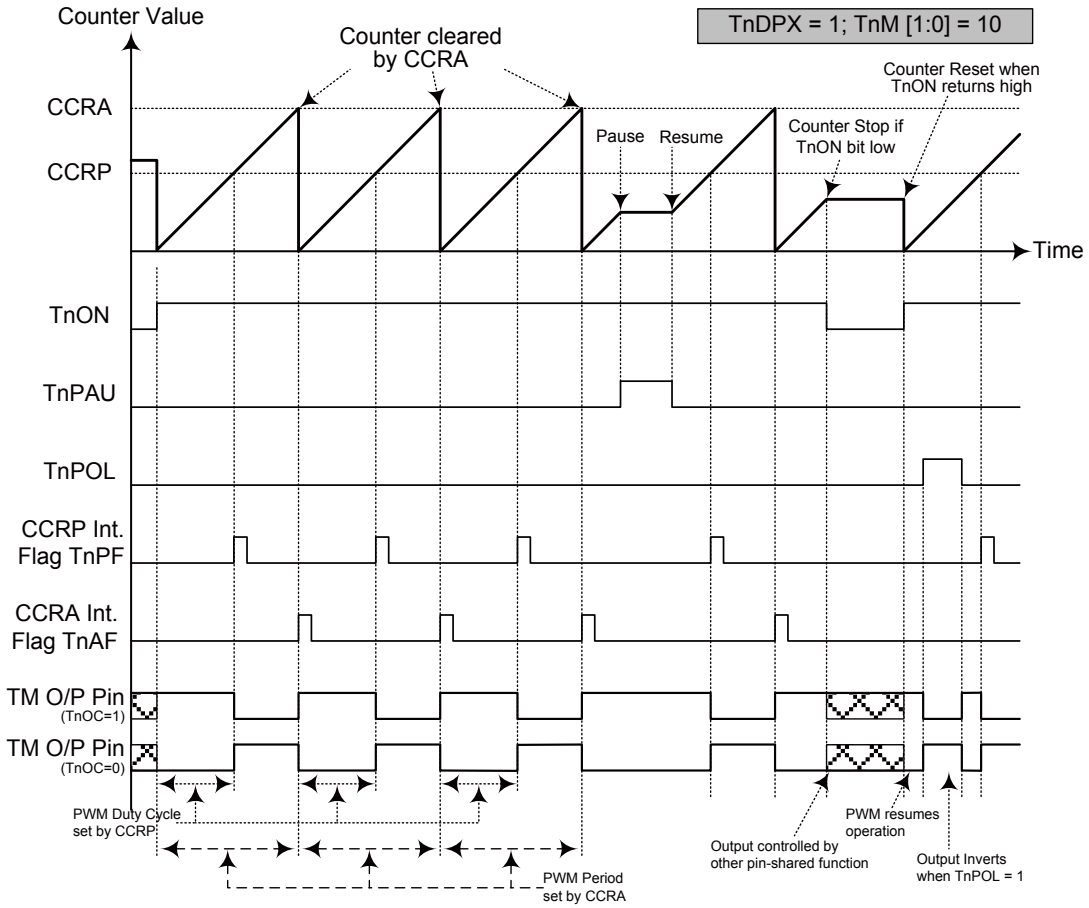
Note: 1. Here TnDPX=0 – Counter cleared by CCRP

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

5. n=0 or 3



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

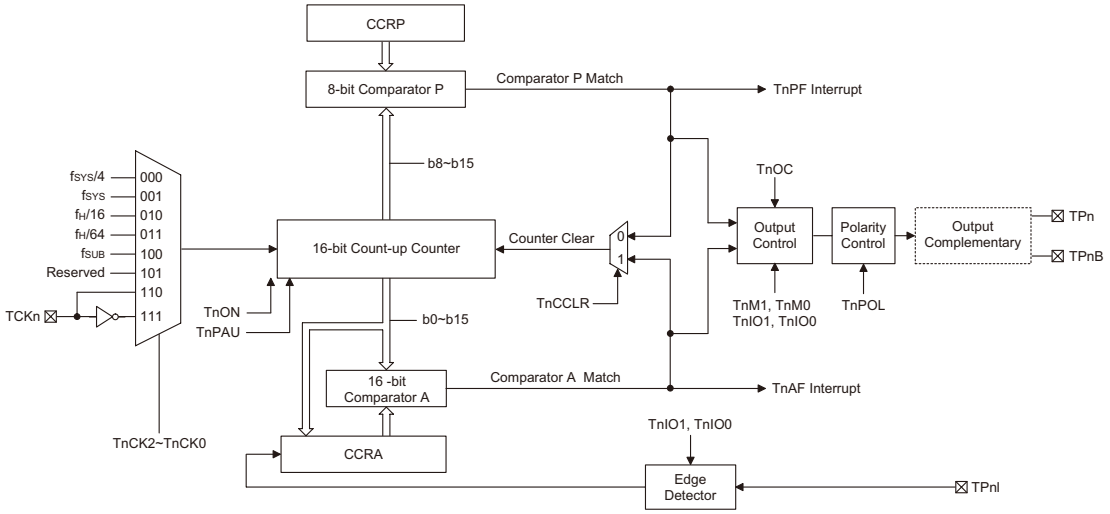
4. The TnCCLR bit has no influence on PWM operation

5. n=0 or 3

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 one or two external output pins.

Device	TM Type	TM Name	TM Input Pin	TM Output Pin
HT66F60A HT66F70A	16-bit STM	TM2, TM4, TM5	TCK2, TP2I; TCK4, TP4I; TCK5, TP5I	TP2, TP2B; TP4, TP4B; TP5, TP5B



Standard Type TM Block Diagram (n=2, 4 or 5)

Standard TM Operation

The size of Standard TM is 16-bit wide. 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 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 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 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 three or eight CCRP bits.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TMnC0	TnPAU	TnCK2	TnCK1	TnCK0	TnON	—	—	—
TMnC1	TnM1	TnM0	TnIO1	TnIO0	TnOC	TnPOL	TnDPX	TnCCLR
TMnDL	D7	D6	D5	D4	D3	D2	D1	D0
TMnDH	D15	D14	D13	D12	D11	D10	D9	D8
TMnAL	D7	D6	D5	D4	D3	D2	D1	D0
TMnAH	D15	D14	D13	D12	D11	D10	D9	D8
TMnRP	D7	D6	D5	D4	D3	D2	D1	D0

16-bit Standard TM Register List (n=2, 4 or 5)

TMnC0 Register

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

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

Bit 6~4 **TnCK2, TnCK1, TnCK0:** Select TMn Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: Reserved
110: TCKn rising edge clock
111: TCKn falling edge clock
These three bits are used to select the clock source for the TM. Selecting the Reserved clock input will effectively disable the internal counter. The external pin clock source can be chosen to be active on the rising or falling edge. The clock source f_{SYS} is the system clock, while f_H and f_{SUB} are other internal clocks, the details of which can be found in the oscillator section.

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

Bit 2~0 Unimplemented, read as "0"

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: Capture Input Mode
10: PWM Mode or Single Pulse Output Mode
11: Timer/Counter Mode
These bits setup the required operating mode for the TM. To ensure reliable operation the TM should be switched off before any changes are made to the TnM1 and TnM0 bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4 **TnIO1~TnIO0:** Select TPn, TPnB 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 TPnI
01: Input capture at falling edge of TPnI
10: Input capture at falling/rising edge of TPnI
11: Input capture disabled
Timer/counter Mode:
Unused
These two bits are used to determine how the TM output pin changes state when a certain condition is reached. The function that these bits select depends upon in which mode the TM is running.

In the Compare Match Output Mode, the TnIO1 and TnIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When 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 change the values of the TnIO1 and TnIO0 bits only after the TM has been switched off. Unpredictable PWM outputs will occur if the TnIO1 and TnIO0 bits are changed when the TM is running.

Bit 3 **TnOC:** TPn, TPnB Output control bit
Compare Match Output Mode
0: Initial low
1: Initial high
PWM Mode/Single Pulse Output Mode
0: Active low
1: Active high

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

Bit 2 **TnPOL:** TPn, TPnB Output polarity Control
0: Non-invert
1: Invert

This bit controls the polarity of the TPn or TPnB 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 Comparator P match
1: TMn Comparator A match

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

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 **TMnDL:** TMn Counter Low Byte Register bit 7~bit 0
TMn 16-bit Counter bit 7~bit 0

TMnDH 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 **TMnDH:** TMn Counter High Byte Register bit 7~bit 0
TMn 16-bit Counter bit 15~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 **TMnAL:** TMn CCRA Low Byte Register bit 7~bit 0
TMn 16-bit CCRA bit 7~bit 0

TMnAH 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 **TMnAH:** TMn CCRA High Byte Register bit 7~bit 0
TMn 16-bit CCRA bit 15~bit 8

TMnRP 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 **TMnRP:** TMn CCRP Register bit 7~bit 0
TMn CCRP 8-bit register, compared with the TMn Counter bit 15~bit 8.
Comparator P Match Period
0: 65536 TMn clocks
1~255: 256×(1~255) TMn 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 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 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

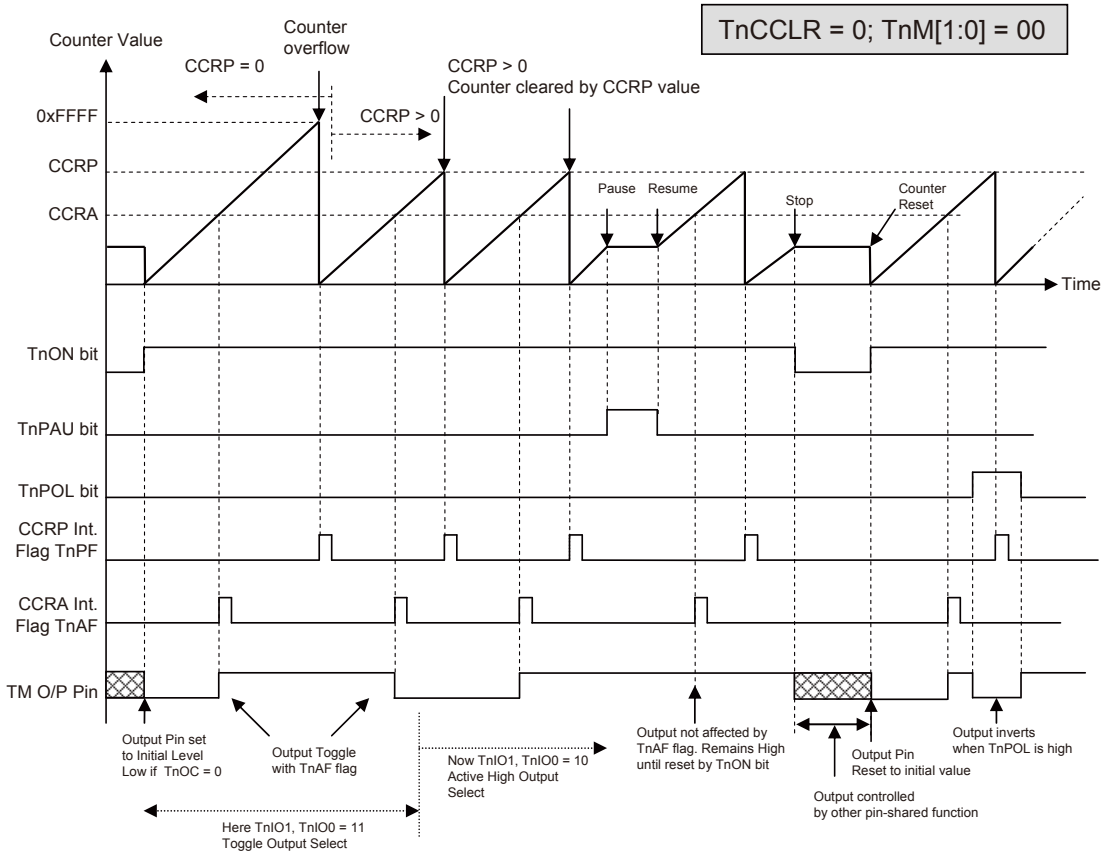
The Standard Type TM can operate in one of five operating modes, Compare Match Output Mode, PWM Output Mode, Single Pulse Output Mode, Capture Input Mode or Timer/Counter Mode. The operating mode is selected using the TnM1 and TnM0 bits in the TMnC1 register.

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 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 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. 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 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.



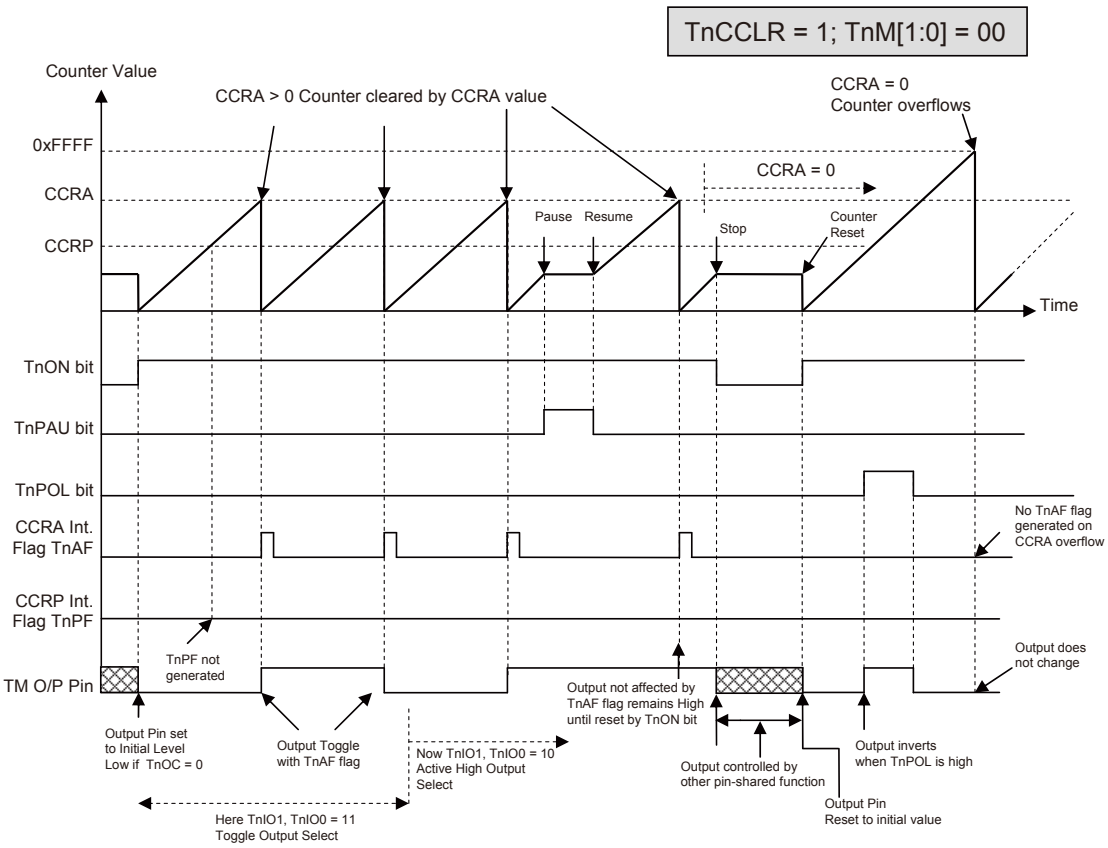
Compare Match Output Mode – TnCCLR=0

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

4. n=2, 4 or 5



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. A TnPF flag is not generated when TnCCLR=1

5. n=2, 4 or 5

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 and also the TnIO1 and TnIO0 bits should be set to 10 respectively. The PWM function within the TM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the TM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM mode, the TnCCLR bit has no effect as the PWM period. Both of the 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.

16-bit STM, PWM Mode, Edge-aligned Mode, TnDPX=0

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

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

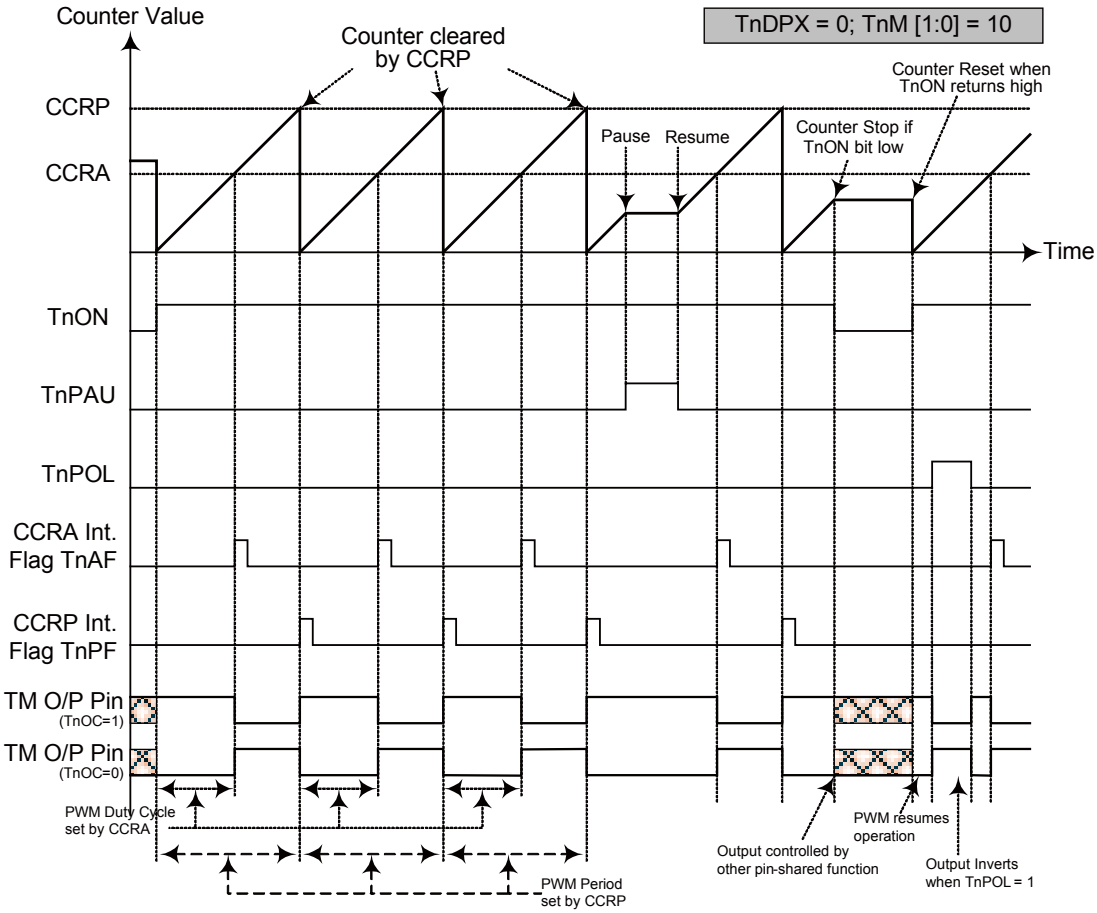
The STM PWM output frequency=$(f_{SYS}/4)/(2\times256)=f_{SYS}/2048$=7.8125kHz, duty=128/(2×256)=25%.

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, TnDPX=1

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

The PWM output period is determined by the CCRA register value together with the TM clock while the PWM duty cycle is defined by the (CCRP×256) except when the CCRP value is equal to 0.



PWM Mode – TnDPX=0

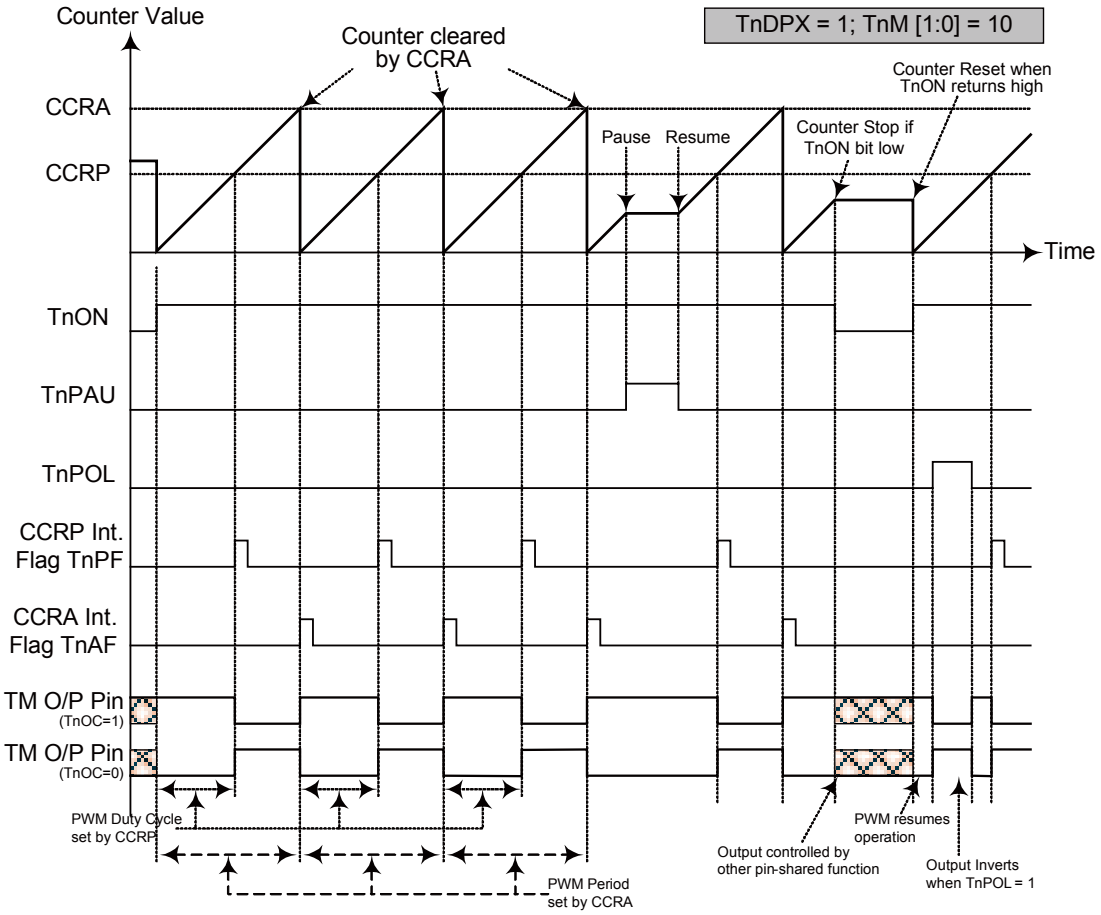
Note: 1. Here TnDPX=0, Counter cleared by CCRP

2. A counter clear sets the PWM Period

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

4. The TnCCLR bit has no influence on PWM operation

5. n=2, 4 or 5



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 running even when TnIO [1:0]=00 or 01

4. The TnCCLR bit has no influence on PWM operation

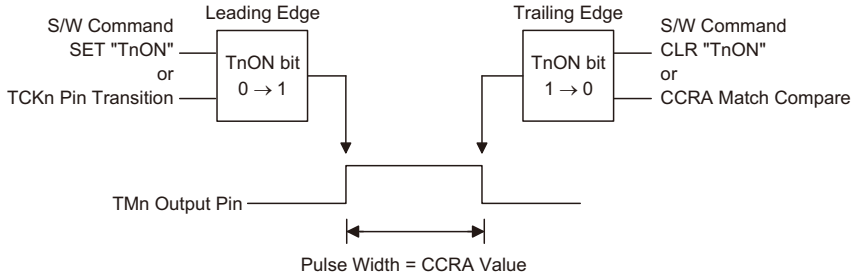
5. n=2, 4 or 5

Single Pulse Mode

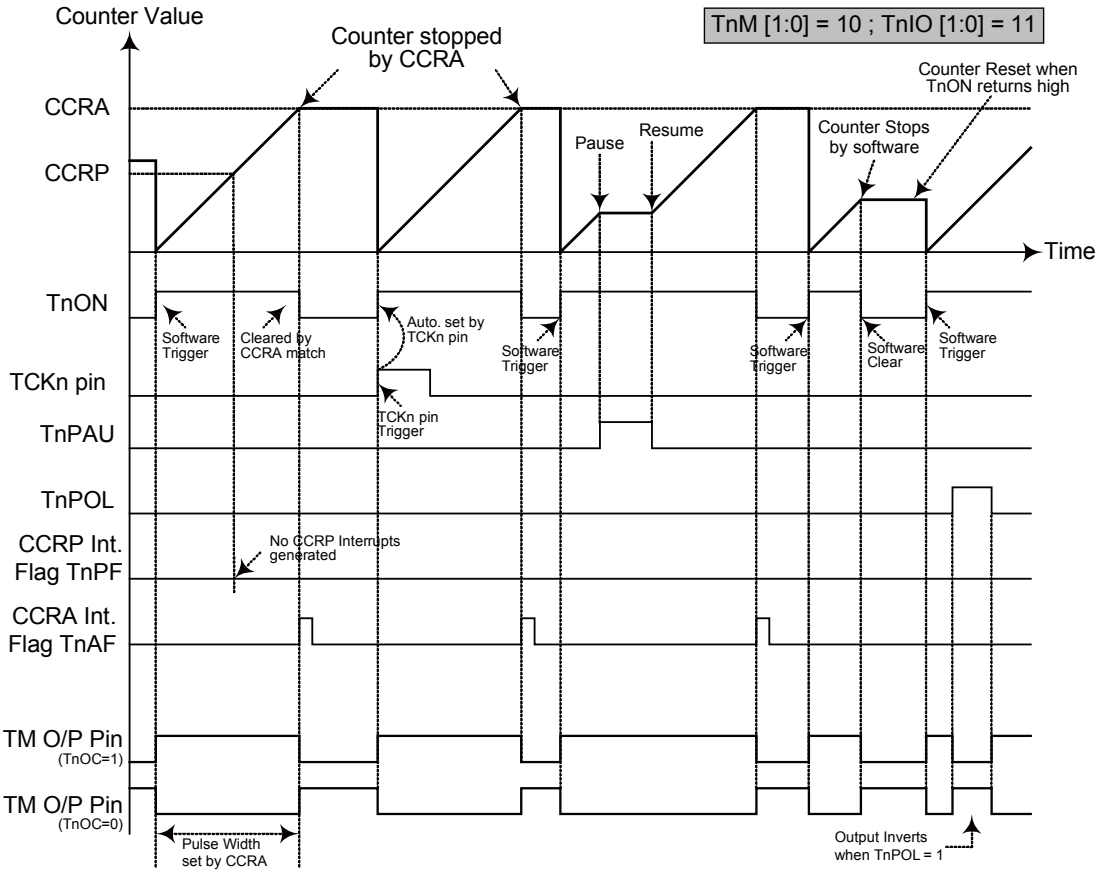
To select this mode, bits TnM1 and TnM0 in the TMnC1 register should be set to 10 respectively and also the TnIO1 and TnIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the TM output pin.

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

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



Single Pulse Generation



Single Pulse Mode

Note: 1. Counter stopped by CCRA

2. CCRP is not used

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

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

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

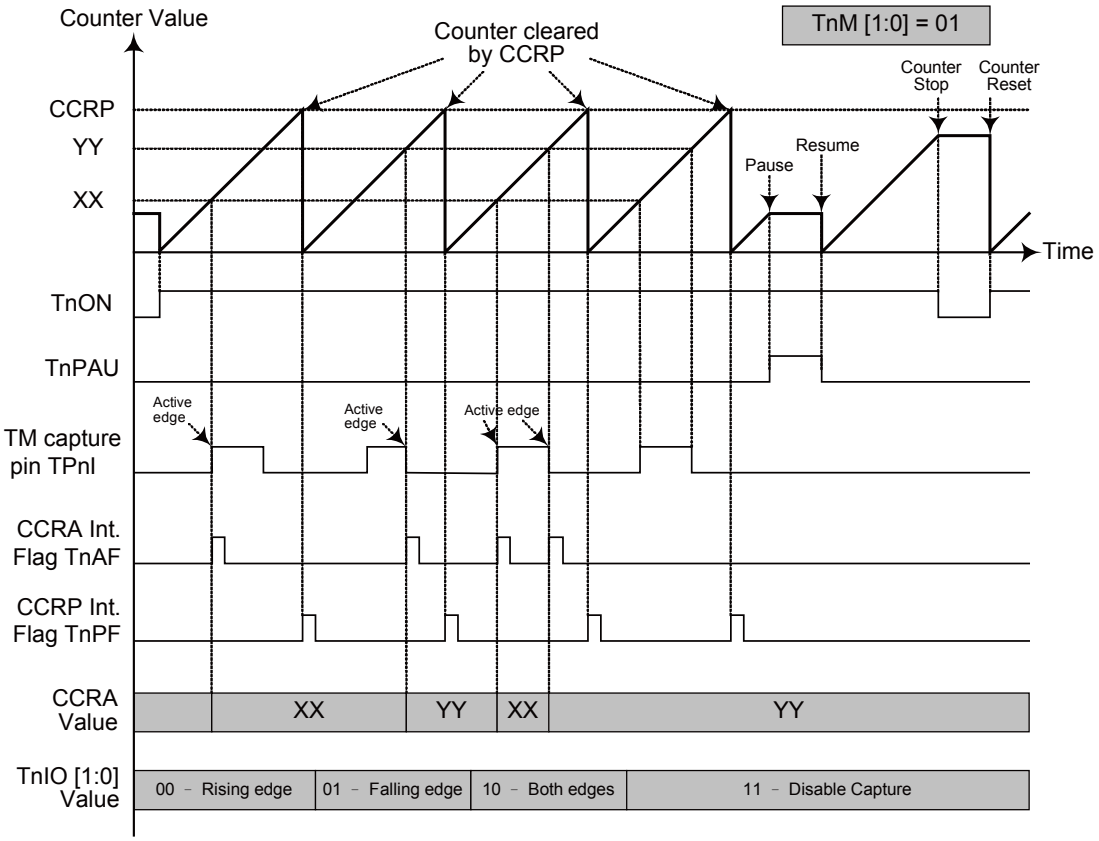
6. n=2, 4 or 5

Capture Input Mode

To select this mode bits TnM1 and TnM0 in the TMnC1 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 TPnI pin, whose active edge can be a rising edge, a falling edge or both rising and falling edges; the active edge transition type is selected using the TnIO1 and TnIO0 bits in the TMnC1 register. The counter is started when the TnON bit changes from low to high which is initiated using the application program.

When the required edge transition appears on the TPnI 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 TPnI pin the counter will continue to free run until the TnON bit changes from high to low. When a CCRP compare match occurs the counter will reset back to zero; in this way the CCRP value can be used to control the maximum counter value. When a CCRP compare match occurs from Comparator P, a TM interrupt will also be generated. Counting the number of overflow interrupt signals from the CCRP can be a useful method in measuring long pulse widths. The TnIO1 and TnIO0 bits can select the active trigger edge on the TPnI pin to be a rising edge, falling edge or both edge types. If the TnIO1 and TnIO0 bits are both set high, then no capture operation will take place irrespective of what happens on the TPnI pin, however it must be noted that the counter will continue to run.

The TnCCLR and TnDPX bits are not used in this Mode.



Capture Input Mode

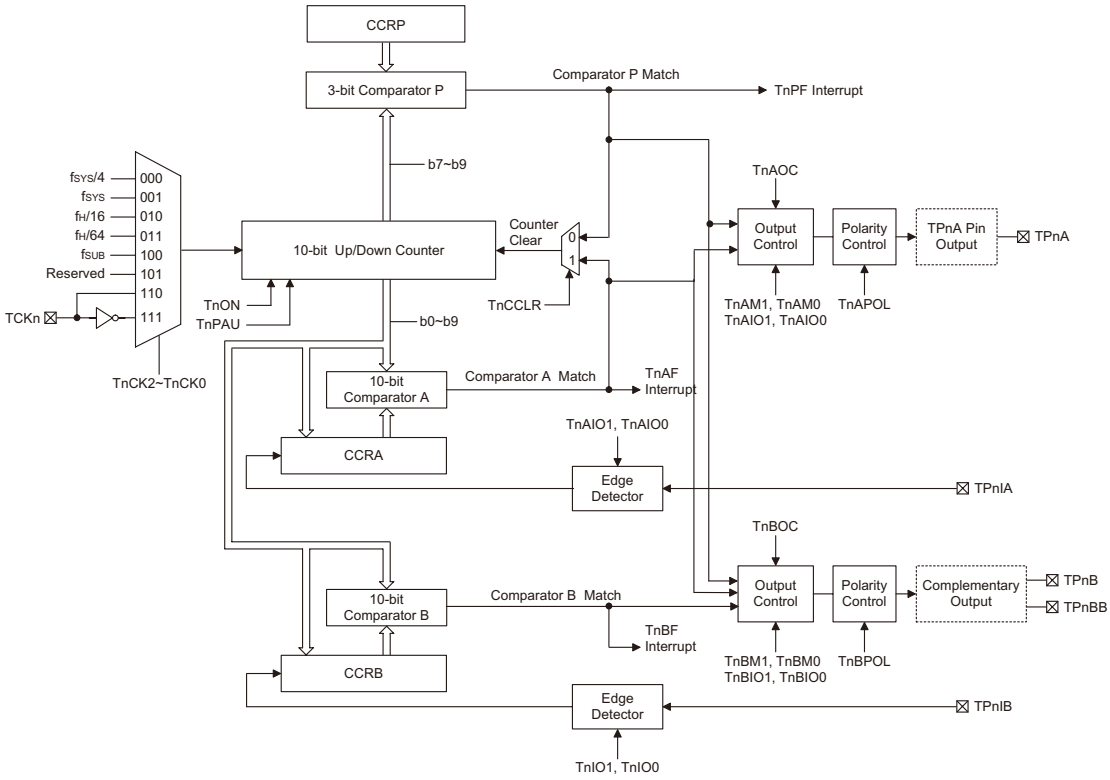
Note: 1. TnM [1:0]=01 and active edge set by the TnIO [1:0] bits

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.
6. n=2, 4 or 5

Enhanced Type TM – ETM

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	TM Type	TM Name.	TM Input Pin	TM Output Pin
HT66F60A HT66F70A	10-bit ETM	TM1	TCK1; TP1IA, TP1IB	TP1A; TP1B, TP1BB



Enhanced Type TM Block Diagram (n=1)

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 T1ON 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 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	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1C0	T1PAU	TnCK2	TnCK1	TnCK0	TnON	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	—	—	—	—	—	D10	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_{SUB}
101: Reserved
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_{SUB} 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 and clearing the bit disables the TM. Clearing this bit to zero will stop the counter from counting and turn off the TM which will reduce its power consumption. When the bit changes state from low to high the internal counter value will be reset to zero, however when the bit changes from high to low, the internal counter will retain its residual value until the bit returns high again.

If the TM is in the Compare Match Output Mode then the TM output pin will be reset to its initial condition, as specified by the 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 TM1clocks
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 TP1IA
01: Input capture at falling edge of TP1IA
10: Input capture at falling/rising edge of TP1IA
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 Count up or down flag
0: Count up
1: Count down

Bit 0 **T1CCLR:** Select TM1 Counter clear condition
0: TM1 Comparatror P match
1: TM1 Comparatror 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 PWM, 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/W	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, TP1BB 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 TP1IB
01: Input capture at falling edge of TP1IB
10: Input capture at falling/rising edge of TP1IB
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 change the values 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, TP1BB 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, TP1BB Output polarity Control
0: Non-invert
1: Invert
This bit controls the polarity of the TP1B, TP1BB 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 **TM1DL:** 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 **TM1DH:** 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 **TM1AL:** 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 **TM1AH:** 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 **TM1BL:** 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 **TM1BH:** 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 TnAM1 and TnAM0 bits in the TMnC1, and the TnBM1 and TnBM0 bits in the TMnC2 register.

ETM Operation Mode	CCRA Compare Match Output Mode	CCRA Timer/ Counter Mode	CCRB PWM Output Mode	CCRB Single Pulse Output Mode	CCRB 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	—	—	—	—	√

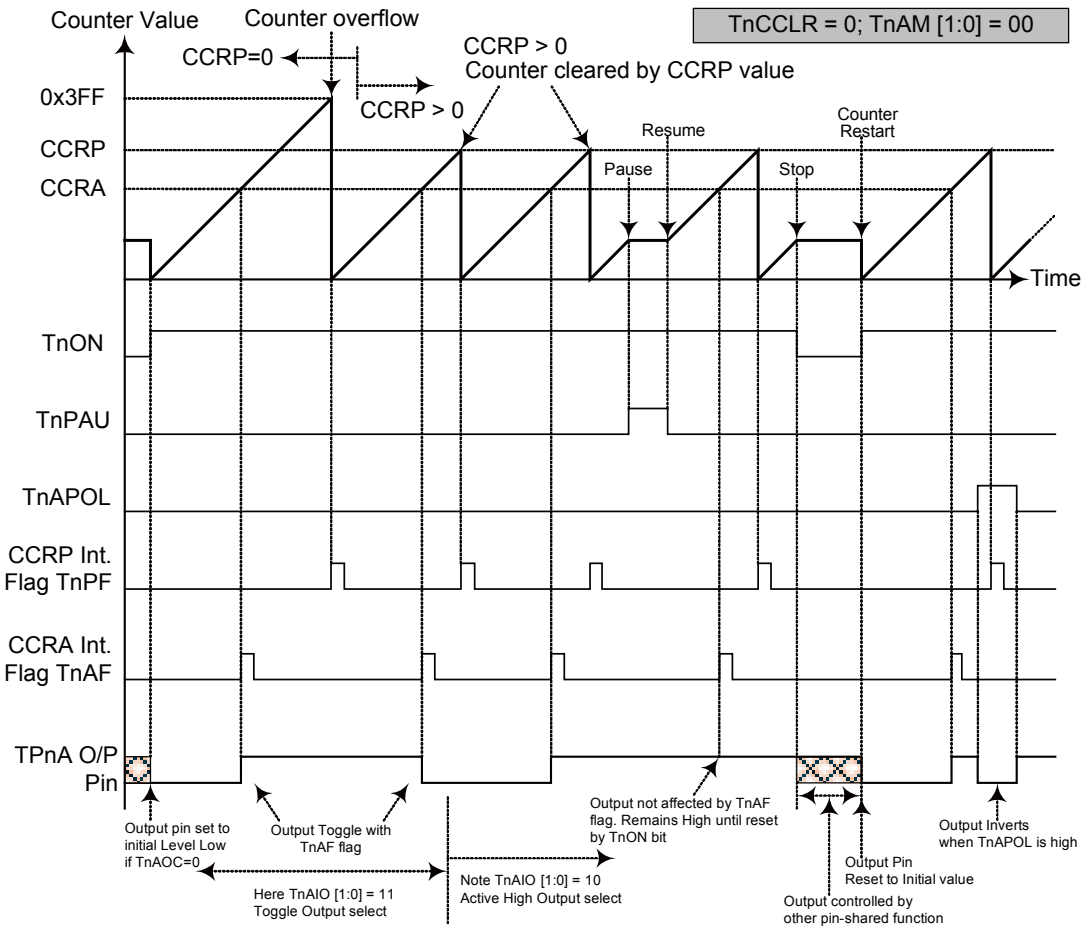
"√": permitted; "—": not permitted

Compare Output Mode

To select this mode, bits TnAM1, TnAM0 and TnBM1, TnBM0 in the TMnC1/TMnC2 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 TnCCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match occurs from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both the TnAF and TnPF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

If the TnCCLR bit in the 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.

As the name of the mode suggests, after a comparison is made, the TM output pin, will change state. The TM output pin condition however only changes state when a TnAF or TnBF interrupt request flag is generated after a compare match occurs from Comparator A or Comparator B. The TnPF interrupt request flag, generated from a compare match from Comparator P, will have no effect on the TM output pin. The way in which the TM output pin changes state is determined by the condition of the TnAIO1 and TnAIO0 bits in the TMnC1 register for ETM CCRA, and the TnBIO1 and TnBIO0 bits in the TMnC2 register for ETM CCRB. The TM output pin can be selected using the TnAIO1, TnAIO0 bits (for the TPnA pin) and TnBIO1, TnBIO0 bits (for the TP1B, TP1BB 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 TnON bit changes from low to high, is setup using the TnAOC or TnBOC bit for TPnA or TP1B, TP1BB output pins. Note that if the TnAIO1, TnAIO0 and TnBIO1, TnBIO0 bits are zero then no pin change will take place.



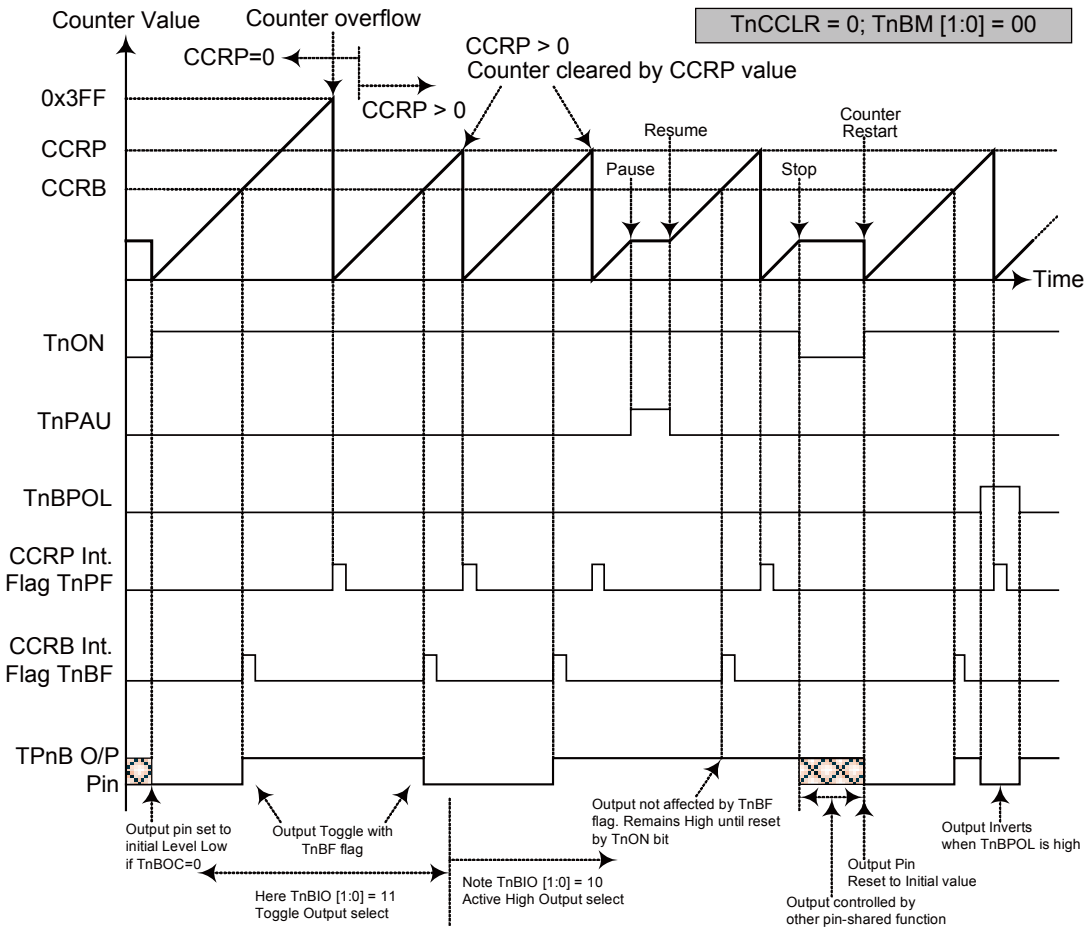
ETM CCRA Compare Match Output Mode – TnCCLR=0

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

4. n=1



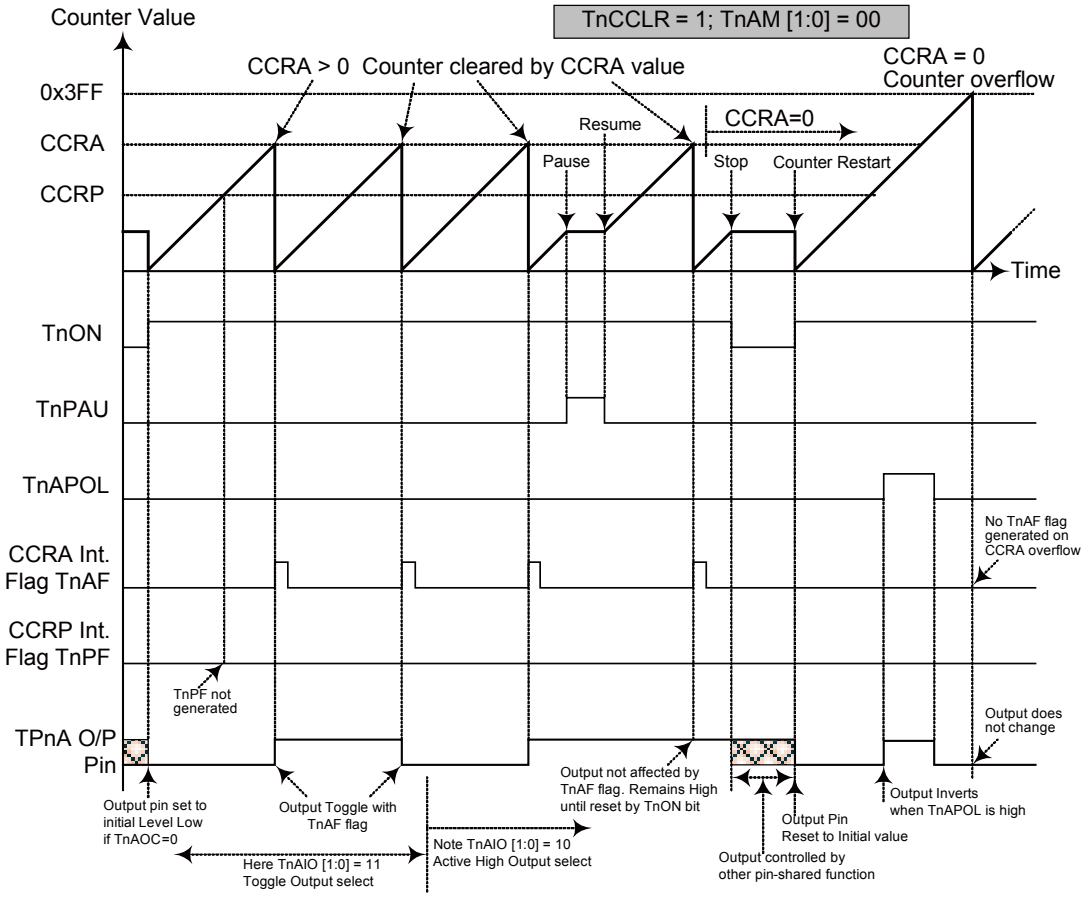
ETM CCRB Compare Match Output Mode – TnCCLR=0

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

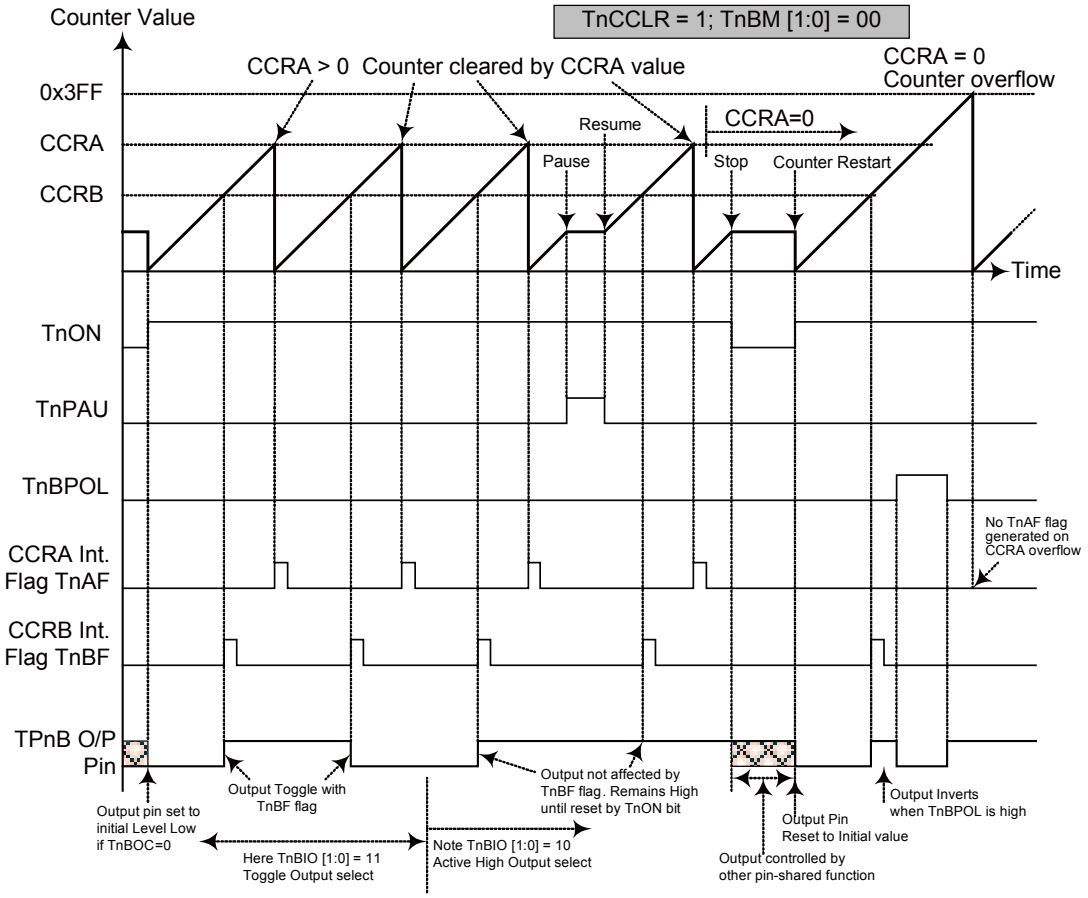
4. n=1



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
5. n=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
5. n=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, the required bit pairs, TnAM1, TnAM0 and TnBM1, TnBM0 should be set to 10 respectively and also the TnAIO1, TnAIO0 and TnBIO1, TnBIO0 bits should be set to 10 respectively. The PWM function within the TM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the TM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM mode, the TnCCLR bit is used to determine in which way the PWM period is controlled. With the TnCCLR bit set high, the PWM period can be finely controlled using the CCRA registers. In this case the CCRB registers are used to set the PWM duty value (for TPnB and TPnBB output pins). The CCRP bits are not used and TPnA output pin is not used. The PWM output can only be generated on the TPnB and TPnBB output pins. With the TnCCLR bit cleared to zero, the PWM period is set using one of the eight values of the three CCRP bits, in multiples of 128. Now both CCRA and CCRB registers can be used to setup different duty cycle values to provide dual PWM outputs on their relative TPnA, TPnB and TPnBB pins.

The TnPWM1 and TnPWM0 bits determine the PWM alignment type, which can be either edge or centre type. In edge alignment, the leading edge of the PWM signals will all be generated concurrently when the counter is reset to zero. With all power currents switching on at the same time, this may give rise to problems in higher power applications. In centre alignment the centre of the PWM active signals will occur sequentially, thus reducing the level of simultaneous power switching currents.

Interrupt flags, one for each of the CCRA, CCRB and CCRP, will be generated when a compare match occurs from either the Comparator A, Comparator B or Comparator P. The TnAOC and TnBOC bits in the TMnC1 and TMnC2 register are used to select the required polarity of the PWM waveform while the two TnAIO1, TnAIO0 and TnBIO1, TnBIO0 bits pairs are used to enable the PWM output or to force the TM output pin to a fixed high or low level. The TnAPOL and TnBPOL bit are used to reverse the polarity of the PWM output waveform.

ETM, PWM Mode, Edge-aligned Mode, TnCCLR=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}=12MHz, TM clock source select f_{SYS}/4, CCRP=100b, CCRA=128 and CCRB=256,

The TPnA PWM output frequency=(f_{SYS}/4)/512=f_{SYS}/2048=5.8594kHz, duty=128/512=25%.

The TPnB or TPnBB PWM output frequency=(f_{SYS}/4)/512=f_{SYS}/2048=5.8594kHz, 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, TnCCLR=1

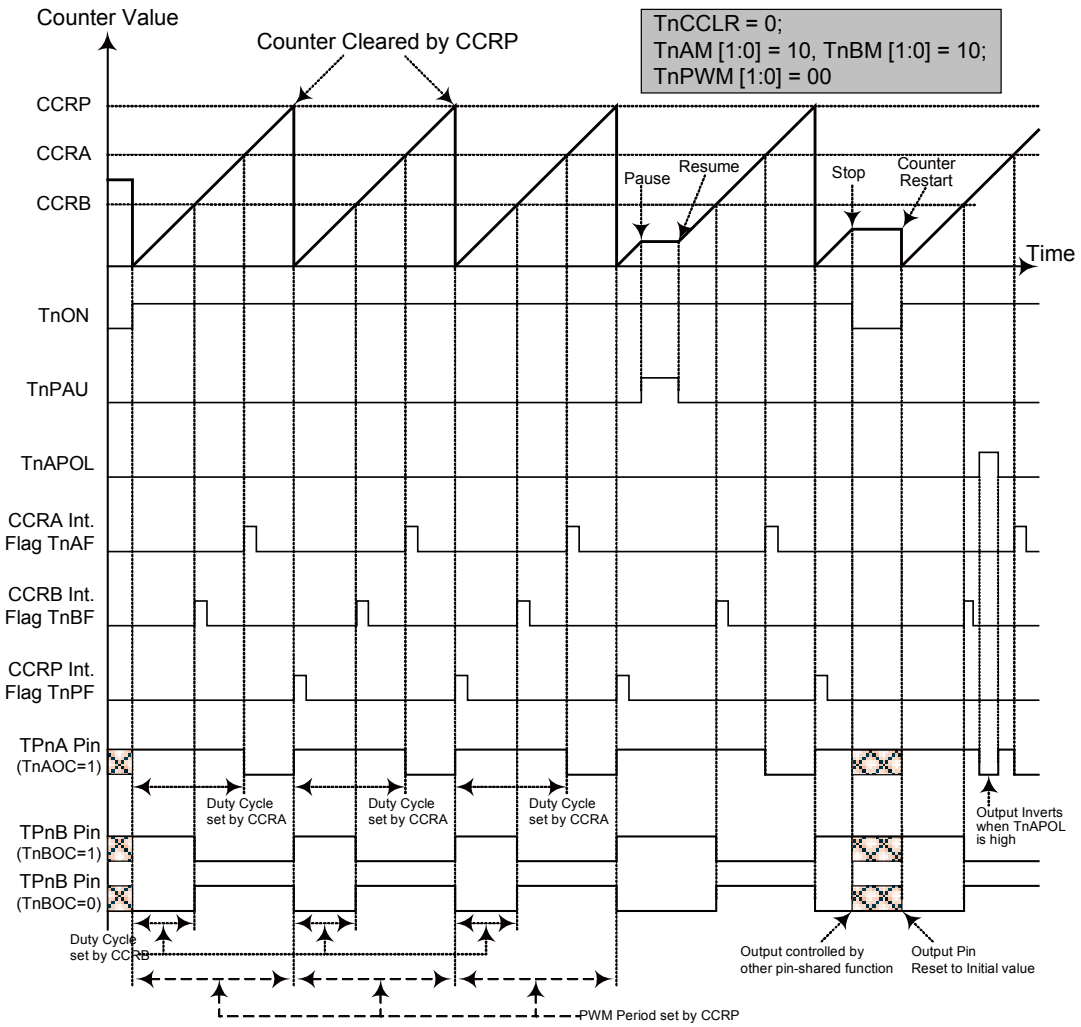
CCRA	1	2	3	511	512	1021	1022	1023
Period	1	2	3	511	512	1021	1022	1023
B Duty	CCRB							

ETM, PWM Mode, Center-aligned Mode, TnCCLR=0

CCRA	001b	010b	011b	100b	101b	110b	111b	000b
Period	256	512	768	1024	1280	1536	1792	2046
A Duty	(CCRA×2)-1							
B Duty	(CCRB×2)-1							

ETM, PWM Mode, Center-aligned Mode, TnCCLR=1

CCRA	1	2	3	511	512	1021	1022	1023
Period	2	4	6	1022	1024	2042	2044	2046
B Duty	(CCRB×2)-1							



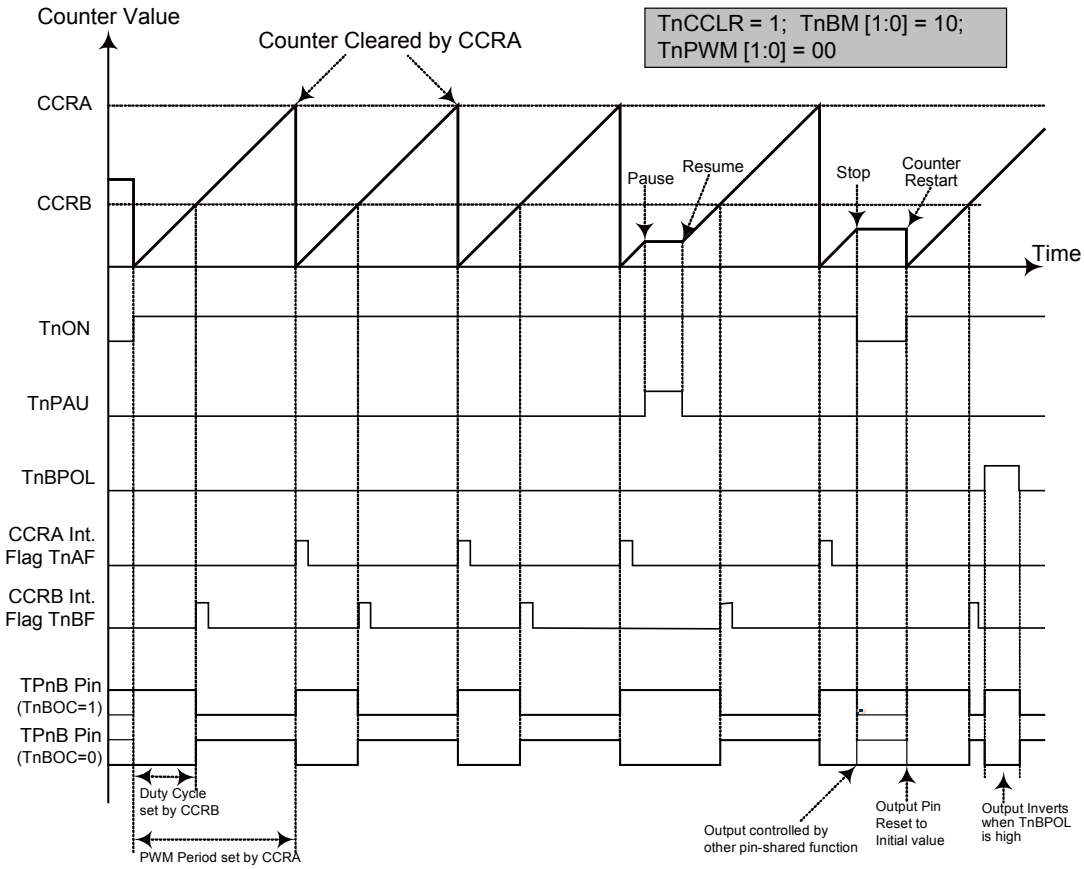
ETM PWM Mode – Edge Aligned

Note: 1. Here TnCCLR=0 therefore CCRP clears the 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

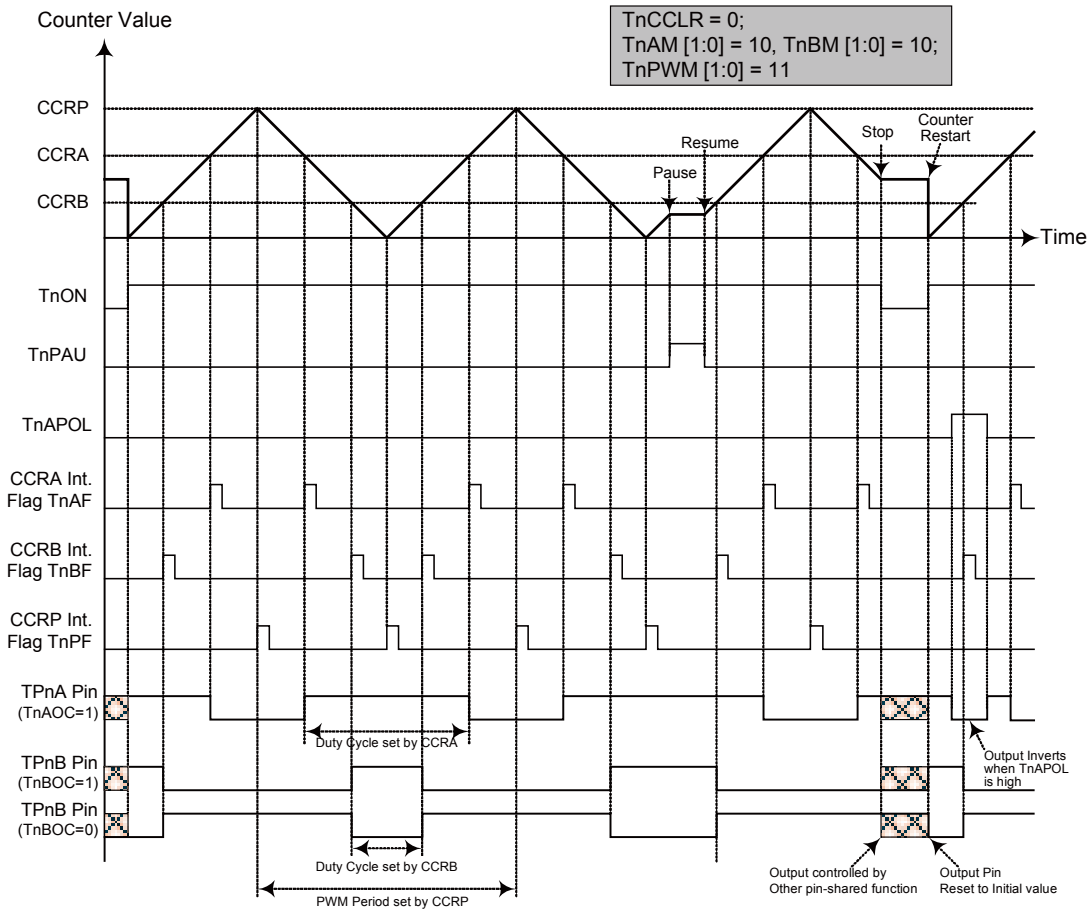
3. CCRA controls the TPnA PWM duty and CCRB controls the TPnB PWM duty

4. n=1



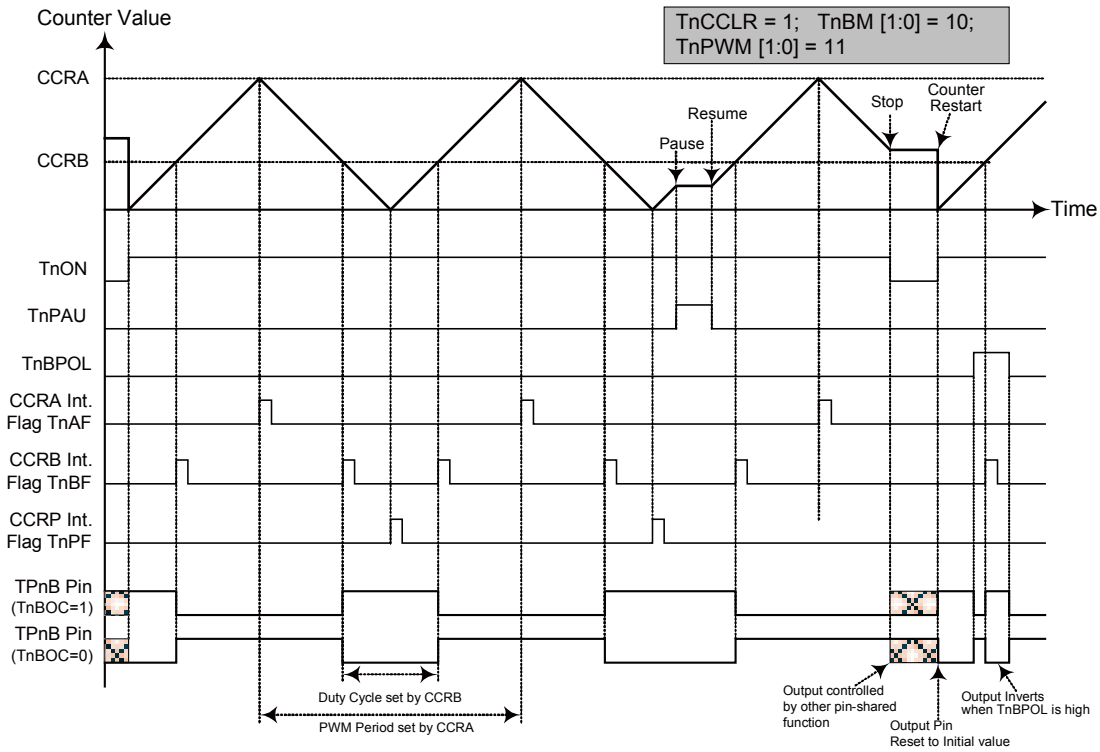
ETM PWM Mode – Edge Aligned

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
5. n=1



ETM PWM Mode – Centre Aligned

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
6. n=1



ETM PWM Mode – Centre Aligned

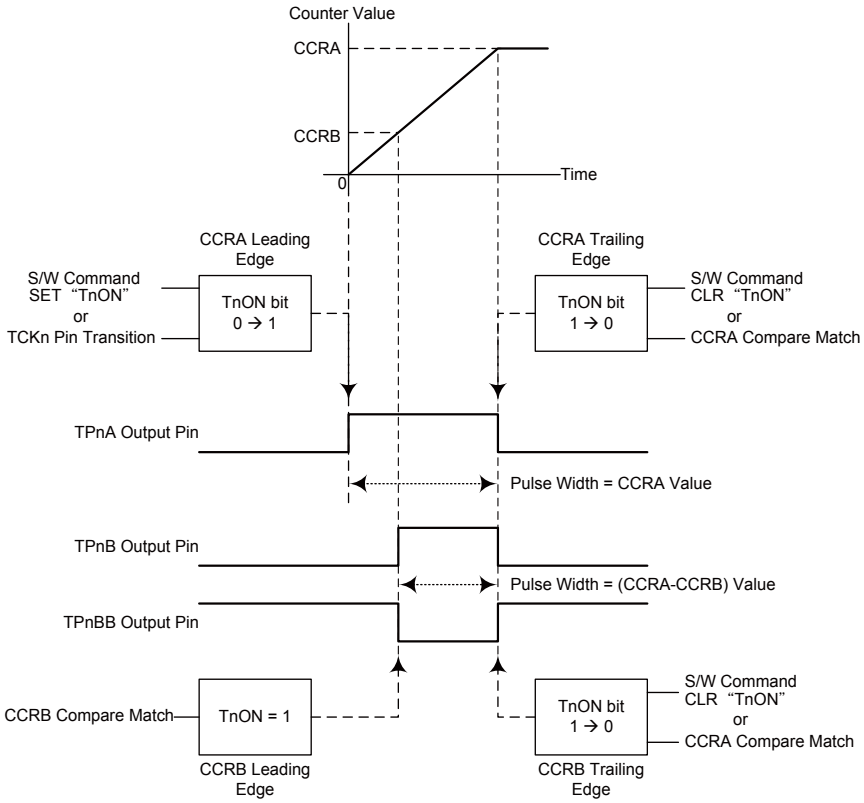
Note: 1. Here TnCCLR=1 therefore CCRA clears the counter and determines the PWM period
2. TnPWM [1:0]=11 therefore the PWM is centre aligned
3. The internal PWM function continues running even when TnBIO [1:0]=00 or 01
4. CCRA controls the TPnB PWM period and CCRB controls the TPnB PWM duty
5. CCRP will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

Single Pulse Mode

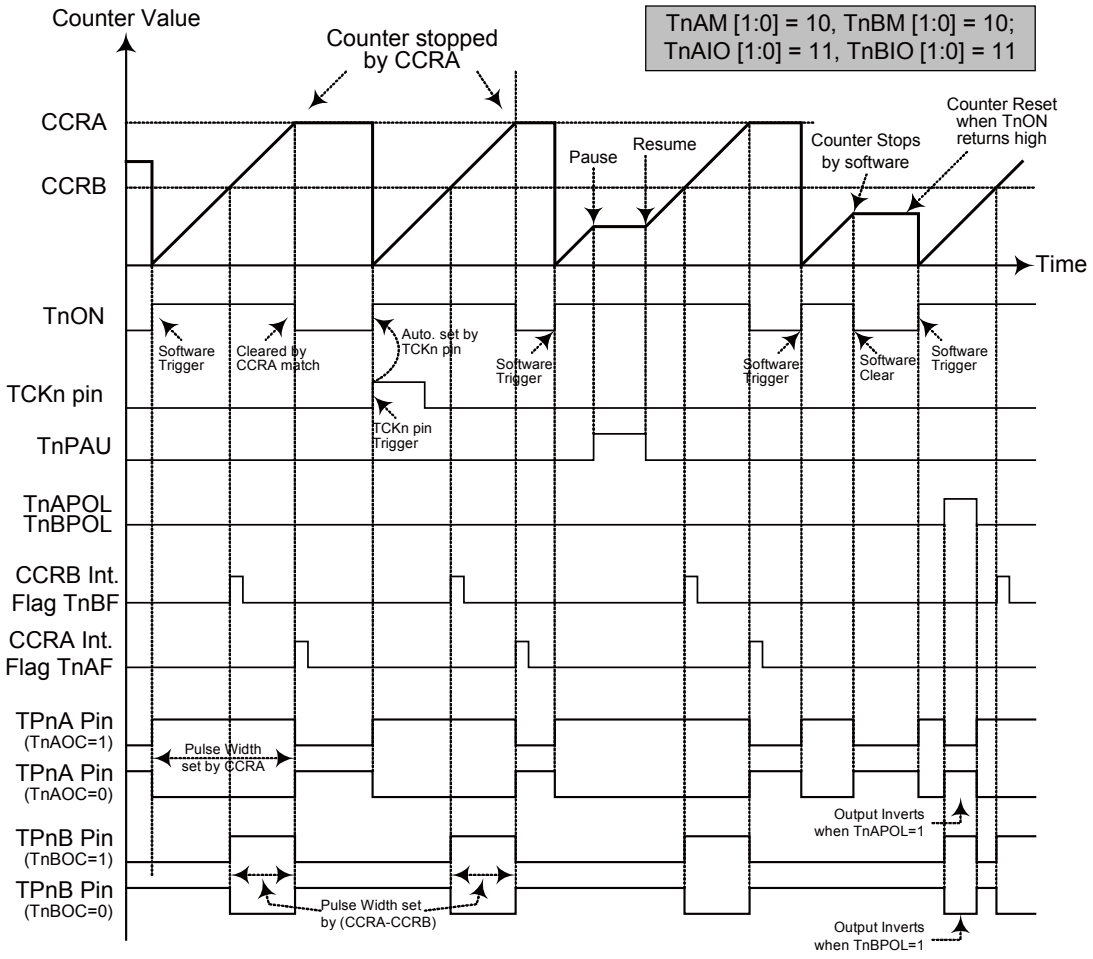
To select this mode, the required bit pairs, TnAM1, TnAM0 and TnBM1, TnBM0 should be set to 10 respectively and also the corresponding TnAIO1, TnAIO0 and TnBIO1, TnBIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the TM output pin.

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

However a compare match from Comparator A will also automatically clear the TnON bit and thus generate the Single Pulse output trailing edge of TPnA and TPnB or TPnBB. In this way the CCRA value can be used to control the pulse width of TPnA. The (CCRA-CCRB) value can be used to control the pulse width of TPnB and TPnBB. 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 TnON bit changes from low to high when the counter restarts. In the Single Pulse Mode CCRP is not used. The TnCCLR bit is also not used.



Single Pulse Generation



ETM – Single Pulse Mode

Note: 1. Counter stopped by CCRA

2. CCRP is not used

3. The pulse 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

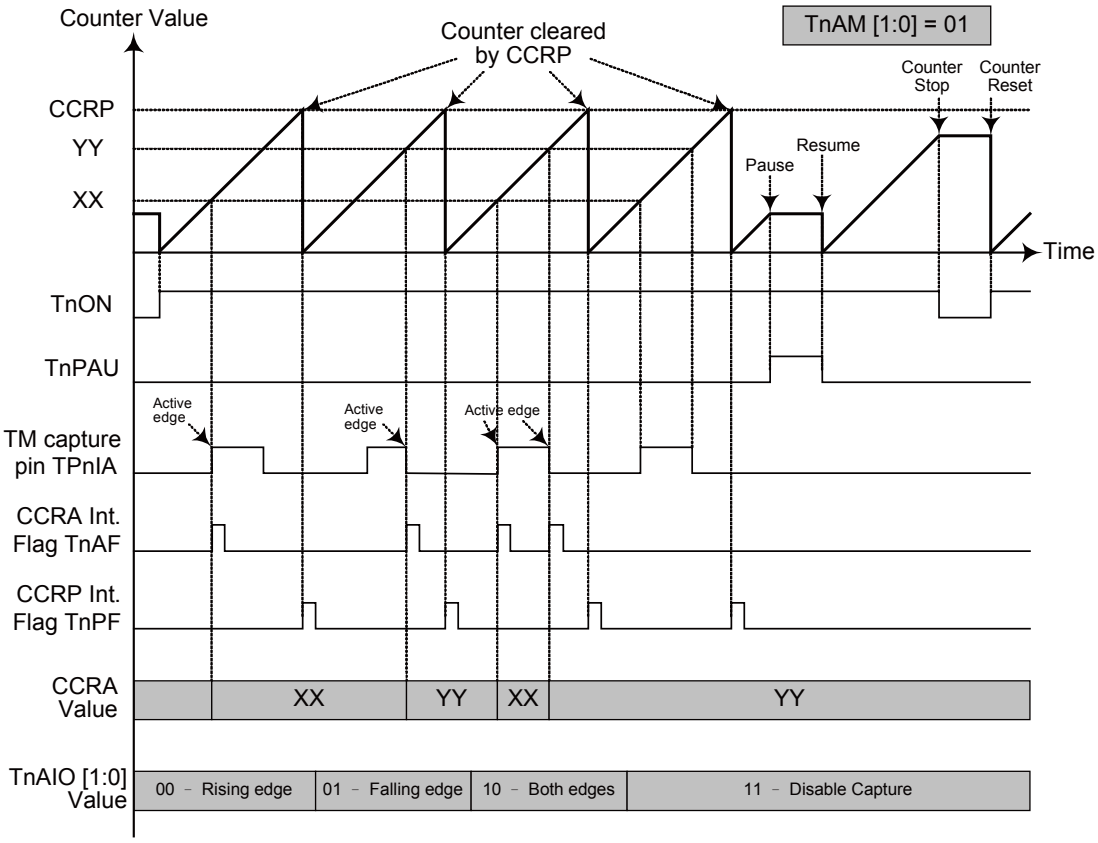
6. n=1

Capture Input Mode

To select this mode bits TnAM1, TnAM0 and TnBM1, TnBM0 in the TMnC1 and TMnC2 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 TPnIA and TPnIB pins, whose active edge can be a rising edge, a falling edge or both rising and falling edges; the active edge transition type is selected using the TnAIO1, TnAIO0 and TnBIO1, TnBIO0 bits in the TMnC1 and TMnC2 registers. The counter is started when the TnON bit changes from low to high which is initiated using the application program.

When the required edge transition appears on the TPnIA and TPnIB 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 TPnIA and TPnIB pins the counter will continue to free run until the TnON bit changes from high to low. When a CCRP compare match occurs the counter will reset back to zero; in this way the CCRP value can be used to control the maximum counter value. When a CCRP compare match occurs from Comparator P, a TM interrupt will also be generated. Counting the number of overflow interrupt signals from the CCRP can be a useful method in measuring long pulse widths. The TnAIO1, TnAIO0 and TnBIO1, TnBIO0 bits can select the active trigger edge on the TPnIA and TPnIB pins to be a rising edge, falling edge or both edge types. If the TnAIO1, TnAIO0 and TnBIO1, TnBIO0 bits are both set high, then no capture operation will take place irrespective of what happens on the TPnIA and TPnIB pins, however it must be noted that the counter will continue to run.

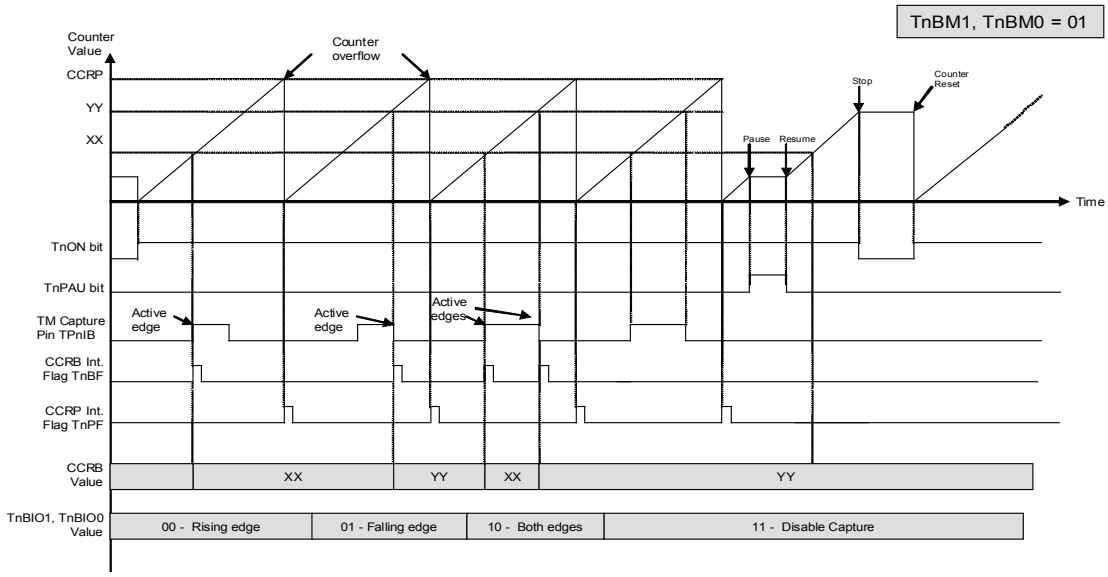
As the TPnIA and TPnIB pins are pin shared with other functions, care must be taken if the TM is in the Capture Input Mode. This is because if the pin is setup as an output, then any transitions on this pin may cause an input capture operation to be executed. The TnCCLR, TnAOC, TnBOC, TnAPOL and TnBPOL bits are not used in this mode.



ETM CCRA Capture Input Mode

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
6. n=1



ETM CCRB Capture Input Mode

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
6. n=1

Aanlog to Digital Converter

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

A/D Overview

The devices contains a multi-channel analog to digital converter which can directly interface to external analog signals, such as that from sensors or other control signals and convert these signals directly into either a 12-bit digital value.

Part No.	Input Channels	A/D Channel Select Bits	Input Pins
HT66F60A HT66F70A	12	ACS4~ACS0	AN0~AN11

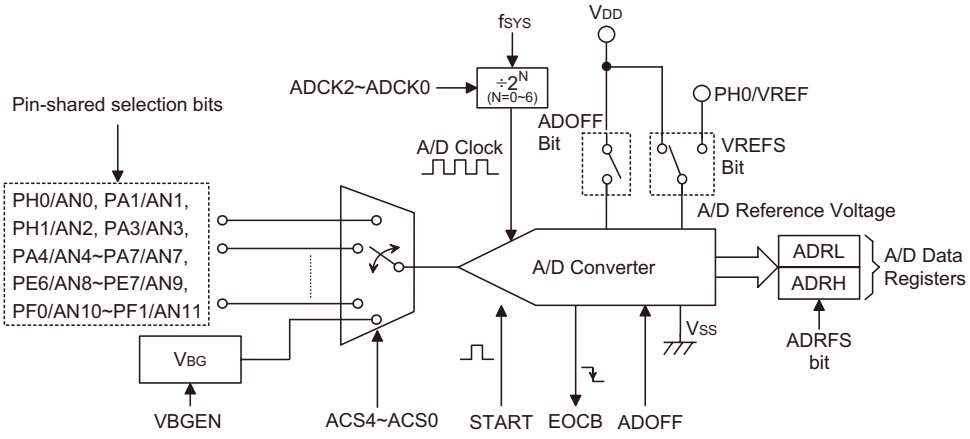
The accompanying block diagram shows the overall internal structure of the A/D converter, together with its associated registers.

A/D Converter Register Description

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

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADRL (ADRFS=0)	D3	D2	D1	D0	—	—	—	—
ADRL (ADRFS=1)	D7	D6	D5	D4	D3	D2	D1	D0
ADRH (ADRFS=0)	D11	D10	D9	D8	D7	D6	D5	D4
ADRH (ADRFS=1)	—	—	—	—	D11	D10	D9	D8
ADCR0	START	EOCB	ADOFF	ACS4	ACS3	ACS2	ACS1	ACS0
ADCR1	—	VBGEN	ADRFS	VREFS	—	ADCK2	ADCK1	ADCK0

A/D Converter Register List



A/D Converter Structure

A/D Converter Data Registers – ADRL, ADRH

As the devices contain an internal 12-bit A/D converter, it requires two data registers to store the converted value. These are a high byte register, known as 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. As only 12 bits of the 16-bit register space is utilised, the format in which the data is stored is controlled by the ADRFS bit in the ADCR0 register as shown in the accompanying table. D0~D11 are the A/D conversion result data bits. Any unused bits will be read as zero.

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

A/D Data Registers

A/D Converter Control Registers – ADCR0, ADCR1, PAS0~PAS3, PES3, PFS0, PHS0

To control the function and operation of the A/D converter, two control registers known as ADCR0 and ADCR1 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 ACS4~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 12 analog inputs must be routed to the converter. It is the function of the ACS4~ACS0 bits to determine which analog channel input pins or internal 1.25V is actually connected to the internal A/D converter.

The pin-shared function conntrol registers, named PAS0~PAS3, PES3, PFS0 and PHS0, contain the corresponding pin-shared function selection bits which determine which pins on Port A, Port E, Port F and Port H are used as analog inputs for the A/D converter input and which pins are not to be used as the A/D converter input. Configuring the corresponding bit will select the A/D input function or either the I/O or other pin-shared function. When the pin is selected to be an A/D input, its original function whether it is an I/O or other pin-shared function will be removed. In addition, any internal pull-high resistors connected to these pins will be automatically removed if the pin is selected to be an A/D input.

- **ADCR0 Register**

Bit	7	6	5	4	3	2	1	0
Name	START	EOCB	ADOFF	ACS4	ACS3	ACS2	ACS1	ACS0
R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	1	1	0	0	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 4~0 **ACS4~ACS0:** Select A/D channel
00000: AN0
00001: AN1
00010: AN2
00011: AN3
00100: AN4
00101: AN5
00110: AN6
00111: AN7
01000: AN8
01001: AN9
01010: AN10
01011: AN11
011xx: Undefined
1xxxx: Internal Bandgap voltage
These are the A/D channel select control bits. As there is only one internal hardware A/D converter each of the twelve A/D inputs must be routed to the internal converter using these bits. If the ACS4 bit is set high, then the internal Bandgap 1.25V will be routed to the A/D Converter.

• **ADCR1 Register**

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

Bit 7 Unimplemented, read as "0"

Bit 6 **VBGEN:** Internal Bandgap voltage control
0: Disable
1: Enable
This bit controls the internal Bandgap circuit on/off function to the A/D converter. When the bit is set high the bandgap voltage 1.25V can be used by the A/D converter. If 1.25V is not used by the A/D converter and the LVR/LVD function is disabled then the bandgap reference circuit will be automatically switched off to conserve power. When 1.25V is switched on for use by the A/D converter, a time tBG should be allowed for the bandgap circuit to stabilise before implementing an A/D conversion.

Bit 5 **ADRFS:** ADC Data Format Control
0: ADC Data MSB is ADRH bit 7, LSB is ADRL bit 4
1: ADC Data MSB is ADRH bit 3, LSB is ADRL bit 0
This bit controls the format of the 12-bit converted A/D value in the two A/D data registers. Details are provided in the A/D data register section.

Bit 4 **VREFS:** Select ADC reference voltage
0: Internal ADC power
1: VREF pin
This bit is used to select the reference voltage for the A/D converter. If the bit is high, then the A/D converter reference voltage is supplied on the external VREF pin. If the pin is low, then the internal reference is used which is taken from the power supply pin VDD.

Bit 3 Unimplemented, read as "0"

Bit 2~0 **ADCK2, ADCK1, 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.

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.

Although the A/D clock source is determined by the system clocky, f_{SYS}, and by bits ADCK2~ADCK0, there are some limitations on the maximum A/D clock source speed that can be selected. As the minimum value of permissible A/D clock period, t_{ADCK}, is 0.5μs, care must be taken for system clock frequencies equal to or greater than 4MHz. For example, if the system clock operates at a frequency of 4MHz, the ADCK2~ADCK0 bits should not be set to "000". Doing so will give A/D clock periods that are less than the minimum A/D clock period which may result in inaccurate A/D conversion values. Refer to the following table for examples, where values marked with an asterisk * show where, depending upon the device, special care must be taken, as the values may be less than the specified minimum A/D Clock Period.

f_{SYS}	A/D Clock Period (t_{ADCK})							
	ADCK2, ADCK1, ADCK0 =000 (f_{SYS})	ADCK2, ADCK1, ADCK0 =001 (f_{SYS}/2)	ADCK2, ADCK1, ADCK0 =010 (f_{SYS}/4)	ADCK2, ADCK1, ADCK0 =011 (f_{SYS}/8)	ADCK2, ADCK1, ADCK0 =100 (f_{SYS}/16)	ADCK2, ADCK1, ADCK0 =101 (f_{SYS}/32)	ADCK2, ADCK1, ADCK0 =110 (f_{SYS}/64)	ADCK2, ADCK1, ADCK0 =111
1MHz	1μs	2μs	4μs	8μs	16μs	32μs	64μs	Undefined
2MHz	500ns	1μs	2μs	4μs	8μs	16μs	32μs	Undefined
4MHz	250ns*	500ns	1μs	2μs	4μs	8μs	16μs	Undefined
8MHz	125ns*	250ns*	500ns	1μs	2μs	4μs	8μs	Undefined
12MHz	83ns*	167ns*	333ns*	667ns	1.33μs	2.67μs	5.33μs	Undefined

A/D Clock Period Examples

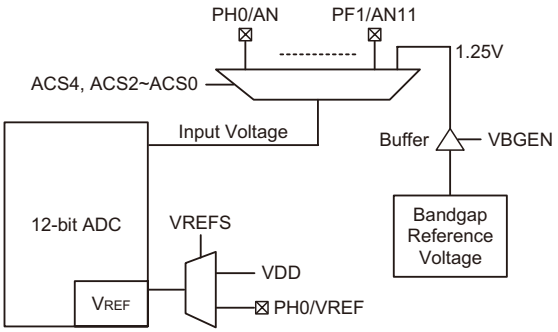
Controlling the power on/off function of the A/D converter circuitry is implemented using the ADOFF bit in the ADCR0 register. This bit must be zero to power on the A/D converter. When the ADOFF bit is cleared to zero to power on the A/D converter internal circuitry a certain delay, as indicated in the timing diagram, must be allowed before an A/D conversion is initiated. Even if no pins are selected for use as A/D inputs, 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 reference voltage supply to the A/D Converter can be supplied from either the positive power supply pin, VDD, or from an external reference sources supplied on pin VREF. The desired selection is made using the VREFS and PH0S3~PH0S0 bits. As the VREF pin is pin-shared with other functions, when the VREFS bit is set high and the PH0S3~PH0S0 bits are set to "0011", the VREF pin function will be selected and the other pin functions will be disabled automatically.

A/D Input Pins

All of the A/D analog input pins are pin-shared with the I/O pins on Port A, Port E, Port F and Port H as well as other functions. The corresponding selection bits in the PAS0~PAS3, PES3, PFS0 and PHS0 registers, determine whether the input pins are setup as A/D converter analog inputs or whether they have other functions. If the corresponding pin is setup to be an A/D converter input, the original pin functions disabled. In this way, pins can be changed under program control to change their function between A/D inputs and other functions. All pull-high resistors, which are setup through register programming, will be automatically disconnected if the pins are setup as A/D inputs. Note that it is not necessary to first setup the A/D pin as an input in the PAC, PEC, PFC or PHC port control register to enable the A/D input as when the relevant A/D input function selection bits enable an A/D input, the status of the port control register will be overridden.

The A/D converter has its own reference voltage pin, VREF, however the reference voltage can also be supplied from the power supply pin, a choice which is made through the VREFS bit in the ADCR1 register. The analog input values must not be allowed to exceed the value of V_{REF}.



A/D Input Structure

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 ACS4~ACS0 bits which are also contained in the ADCR1 and ADCR0 register.
- Step 4
  Select which pins are to be used as A/D inputs and configure them by correctly programming the corresponding pin-shared function selection registers.
- Step 5
  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 6

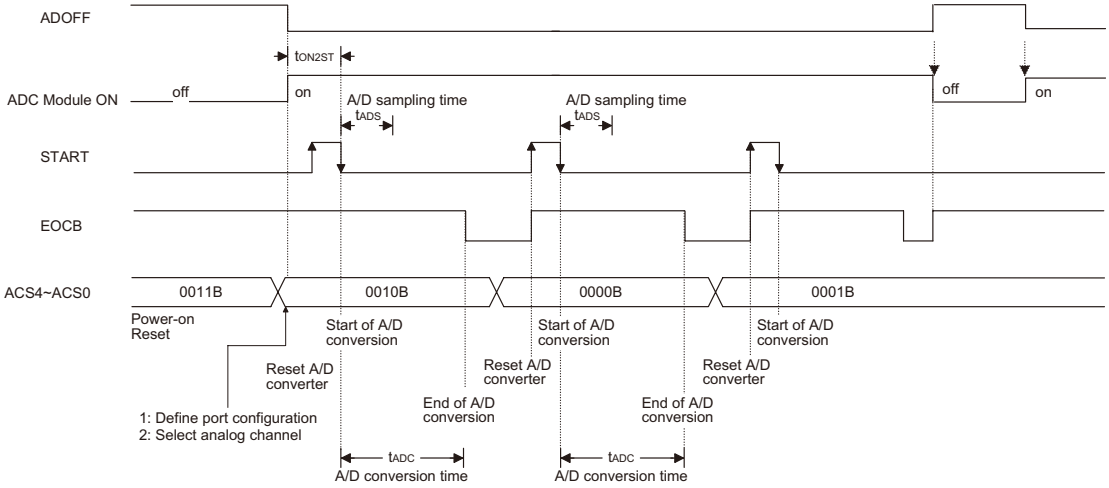
  The analog to digital conversion process can now be initialised by setting the START bit in the ADCR0 register from low to high and then low again. Note that this bit should have been originally cleared to zero.

- Step 7

  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 register 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 $16t_{ADCK}$ where t_{ADCK} 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

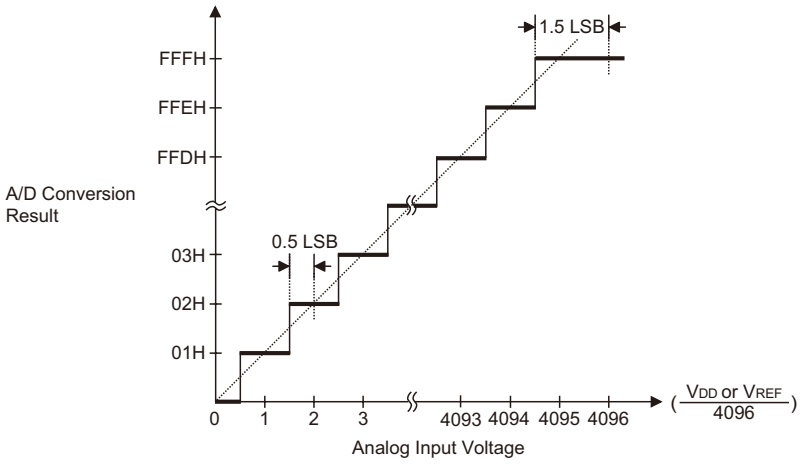
As the devices contain a 12-bit A/D converter, its full-scale converted digitised value is equal to FFFH. Since the full-scale analog input value is equal to the V_{DD} or V_{REF} voltage, this gives a single bit analog input value of V_{DD} or V_{REF} divided by 4096.

$$1\ LSB=(V_{DD}\ or\ V_{REF})\div 4096$$

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

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

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_{DD} or V_{REF} level.



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

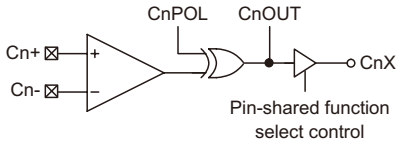
```
clr  ADE              ; disable ADC interrupt
mov  a,03H
mov  ADCR1,a          ; select fSYS/8 as A/D clock and switch off 1.25V
clr  ADOFF
mov  a,03h            ; setup PHS0 to configure pin AN0
mov  PHS0,a
mov  a,00h
mov  ADCR0,a          ; enable and connect AN0 channel to A/D converter
:
start_conversion:
clr  START            ; high pulse on start bit to initiate conversion
set  START            ; reset A/D
clr  START            ; start A/D
polling_EOC:
sz   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
```

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 and switch off 1.25V
clr  ADOFF
mov  a,03h            ; setup PHS0 to configure pin AN0
mov  PHS0,a
mov  a,00h
mov  ADCR0,a          ; enable and connect AN0 channel to A/D converter
Start_conversion:
clr  START            ; high pulse on START bit to initiate conversion
set  START            ; reset A/D
clr  START            ; start A/D
clr  ADF              ; clear ADC interrupt request flag
set  ADE              ; enable ADC interrupt
set  EMI              ; enable global interrupt
:
:
                      ; ADC interrupt service routine
ADC_ISR:
mov  acc_stack,a      ; save ACC to user defined memory
mov  a,STATUS
mov  status_stack,a   ; save STATUS to user defined memory
:
:
mov  a,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_INT_ISR:
mov  a,status_stack
mov  STATUS,a         ; restore STATUS from user defined memory
mov  a,acc_stack      ; restore ACC from user defined memory
reti
```

Comparators

Two independent analog comparators are contained within these devices. These functions offer flexibility via their register controlled features such as power-down, polarity select, hysteresis etc. In sharing their pins with normal I/O pins the comparators do not waste precious I/O pins if there functions are otherwise unused.



Comparator

Comparator Operation

The devices contain two comparator functions which are used to compare two analog voltages and provide an output based on their difference. Full control over the two internal comparators is provided via two control registers, CP0C and CP1C, one assigned to each comparator. The comparator output is recorded via a bit in their respective control register, but can also be transferred out onto a shared I/O pin. Additional comparator functions include, output polarity, hysteresis functions and power down control.

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

Comparator Registers

There are two registers for overall comparator operation, one for each comparator. As corresponding bits in the two registers have identical functions, they following register table applies to both registers.

Register Name	Bit							
	7	6	5	4	3	2	1	0
CP0C	—	C0EN	C0POL	C0OUT	—	—	—	C0HYEN
CP1C	—	C1EN	C1POL	C1OUT	—	—	—	C1HYEN

Comparator Registers List

CP0C Register

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

Bit 7 Unimplemented, read as "0"

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

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

Bit 4 **C0OUT:** Comparator output bit
C0POL=0
0: C0+ < C0-
1: C0+ > C0-
C0POL=1
0: C0+ > C0-
1: C0+ < C0-
This bit stores the comparator output bit. The polarity of the bit is determined by the voltages on the comparator inputs and by the condition of the C0POL bit.

Bit 3~1 Unimplemented, read as "0"

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

CP1C Register

Bit	7	6	5	4	3	2	1	0
Name	—	C1EN	C1POL	C1OUT	—	—	—	C1HYEN
R/W	—	R/W	R/W	R	—	—	—	R/W
POR	—	0	0	0	—	—	—	1

Bit 7 Unimplemented, read as "0"

Bit 6 **C1EN:** Comparator On/Off control
0: Off
1: On
This is the Comparator on/off control bit. If the bit is zero the comparator will be switched off and no power consumed even if analog voltages are applied to its inputs. For power sensitive applications this bit should be cleared to zero if the comparator is not used or before the devices enter the SLEEP or IDLE mode.

Bit 5 **C1POL:** Comparator output polarity
0: Output not inverted
1: Output inverted
This is the comparator polarity bit. If the bit is zero then the C1OUT bit will reflect the non-inverted output condition of the comparator. If the bit is high the comparator C1OUT bit will be inverted.

Bit 4 **C1OUT:** Comparator output bit
C1POL=0
0: C1+ < C1-
1: C1+ > C1-
C1POL=1
0: C1+ > C1-
1: C1+ < C1-
This bit stores the comparator output bit. The polarity of the bit is determined by the voltages on the comparator inputs and by the condition of the C1POL bit.

Bit 3~1 Unimplemented, read as "0"

Bit 0 **C1HYEN:** Hysteresis Control
0: Off
1: On
This is the hysteresis control bit and if set high will apply a limited amount of hysteresis to the comparator, as specified in the Comparator Electrical Characteristics table. The positive feedback induced by hysteresis reduces the effect of spurious switching near the comparator threshold.

Comparator Interrupt

Each also possesses its own interrupt function. When any one of the changes state, its relevant interrupt flag will be set, and if the corresponding interrupt enable bit is set, then a jump to its relevant interrupt vector will be executed. Note that it is the changing state of the C0OUT or C1OUT bit and not the output pin which generates an interrupt. If the microcontroller is in the SLEEP or IDLE Mode and the Comparator is enabled, then if the external input lines cause the Comparator output to change state, the resulting generated interrupt flag will also generate a wake-up. If it is required to disable a wake-up from occurring, then the interrupt flag should be first set high before entering the SLEEP or IDLE Mode.

Programming Considerations

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

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

Serial Interface Module – SIM

These devices contain a Serial Interface Module, which includes both the four-line SPI interface or 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 or EEPROM memory, etc. The SIM interface pins are pin-shared with other I/O pins and therefore the SIM interface functional pins must first be selected using the corresponding pin-shared function selection bits. 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 pins are selected using pull-high control registers when the SIM function is enabled and the corresponding pins are used as SIM input pins.

SPI Interface

The SPI interface is often used to communicate with external peripheral devices such as sensors, Flash or EEPROM 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 devices can be either master or slave. Although the SPI interface specification can control multiple slave devices from a single master, but these devices 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.

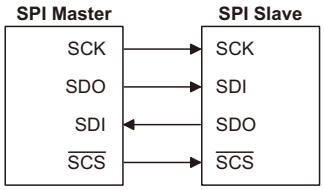
SPI Interface Operation

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 pins must first be selected by configuring the pin-shared function selection bits and setting the correct bits in the SIMC0 and SIMC2 registers. After the desired SPI configuration has been set it can be disabled or enabled using the SIMEN bit in the SIMC0 register. 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.

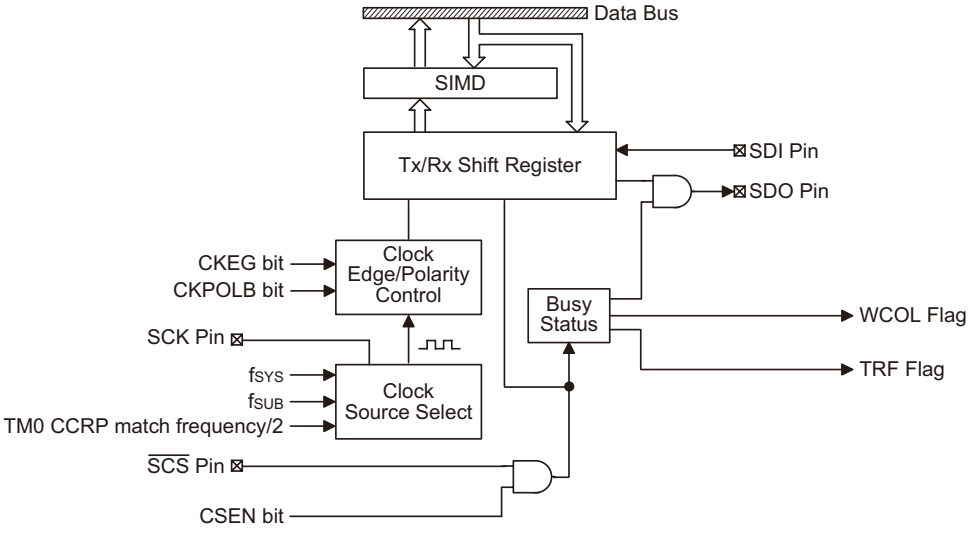
The SPI function in this device offers the following features:

- 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

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 Master/Slave Connection



SPI Bolck Diagram

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	—	SIMDEB1	SIMDEB0	SIMEN	—
SIMD	D7	D6	D5	D4	D3	D2	D1	D0
SIMC2	D7	D6	CKPOLB	CKEG	MLS	CSEN	WCOL	TRF

SIM Registers 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 devices write 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 devices 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	—	SIMDEB1	SIMDEB0	SIMEN	—
R/W	R/W	R/W	R/W	—	R/W	R/W	R/W	—
POR	1	1	1	—	0	0	0	—

Bit 7~5 **SIM2, SIM1, 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: Non SIM function

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 TM0. If the SPI Slave Mode is selected then the clock will be supplied by an external Master devices.

Bit 4 Unimplemented, read as "0"

Bit 3~2 **SIMDEB1~SIMDEB0:** I^2C Debounce Time Selection
00: No debounce
01: 2 system clock debounce
1x: 4 system clock debounce

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 lose their SPI or I^2C function and the SIM operating current will be reduced to a minimum value. When the bit is high the SIM interface is enabled. The SIM configuration option must have first enabled the SIM interface for this bit to be effective.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"

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 Undefined bit
This bit can be read or written by the application 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 I/O pin or the other functions. If the bit is high the $\overline{SCS}$ pin will be enabled and used as a select pin.

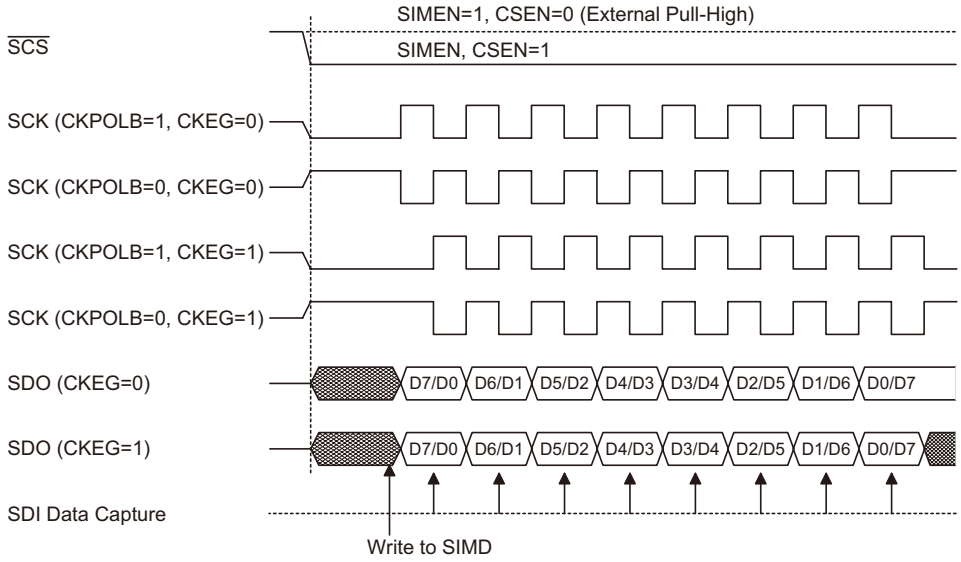
Bit 1 **WCOL:** SPI Write Collision flag
0: No collision
1: Collision
The WCOL flag is used to detect if a data collision has occurred. If this bit is high it means that data has been attempted to be written to the SIMD register during a data transfer operation. This writing operation will be ignored 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.

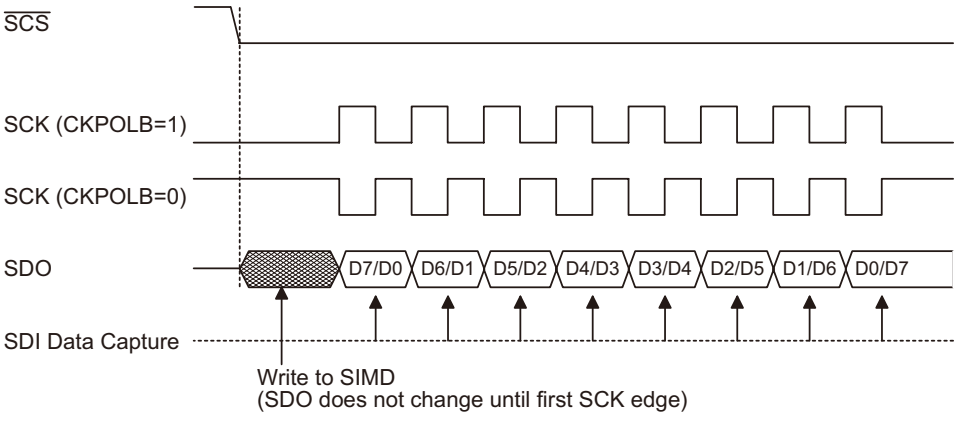
SPI Communication

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

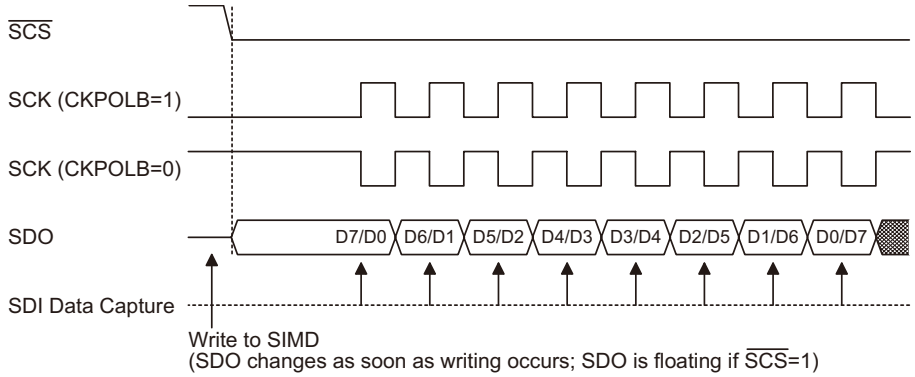
The SPI will continue to function even in the IDLE Mode.



SPI Master Mode Timing

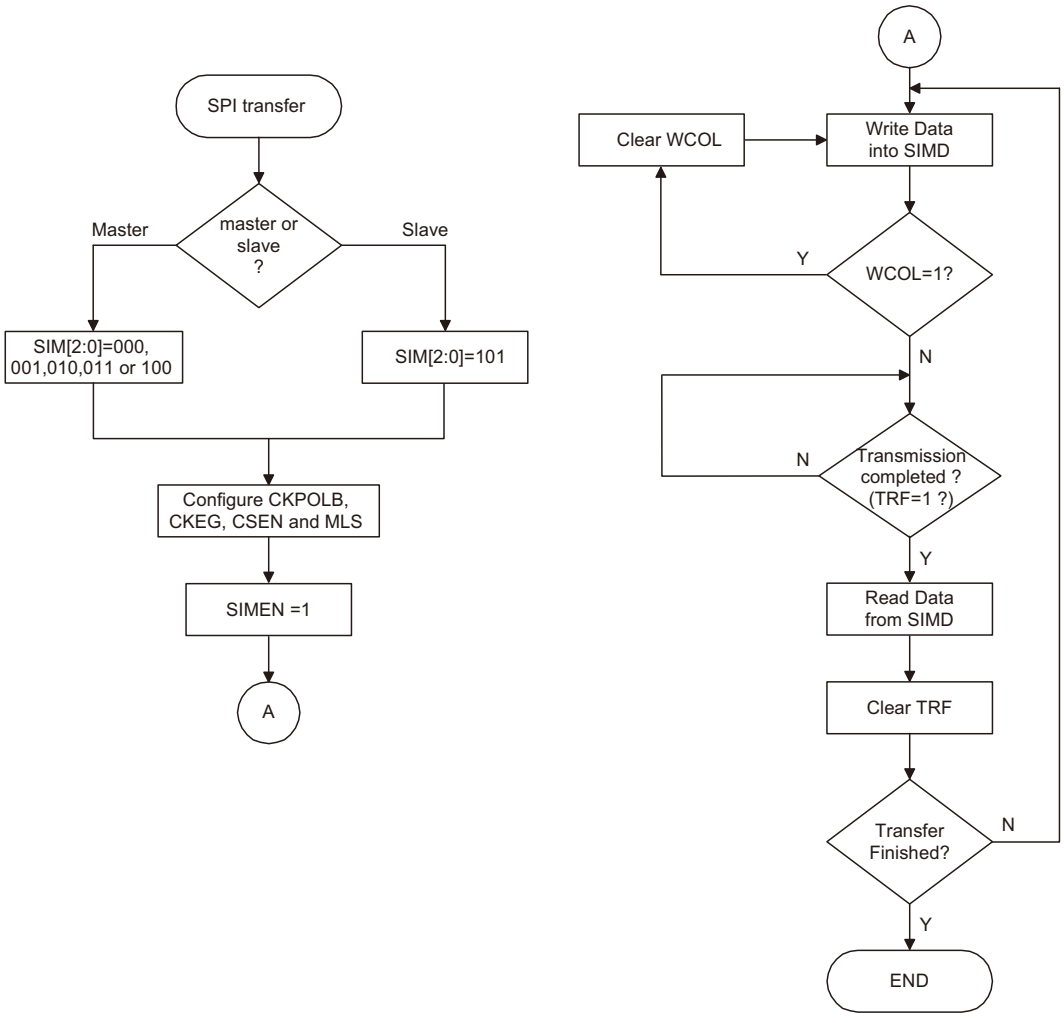


SPI Slave Mode Timing – CKEG=0



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

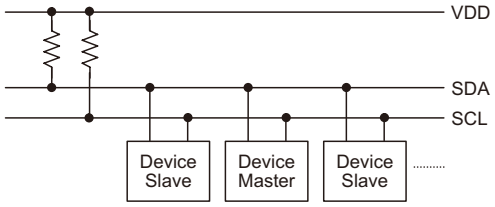
SPI Slave Mode Timing – CKEG=1



SPI Transfer Control Flowchart

I²C Interface

The I²C interface is used to communicate with external peripheral devices such as sensors, EEPROM memory 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

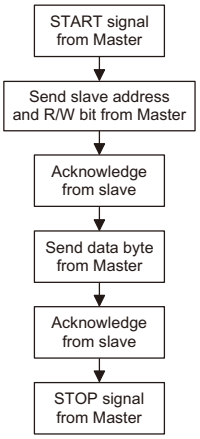
The I²C serial interface is a two line interface, a serial data line, SDA, and serial clock line, SCL. As many devices may be connected together on the same bus, their outputs are both open drain types. For this reason it is necessary that external pull-high resistors are connected to these outputs. 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 these devices, which only operate in slave mode, there are two methods of transferring data on the I²C bus, the slave transmit mode and the slave receive mode.

The SIMDEB1 and SIMDEB0 bits determine the debounce time of the I²C interface. This uses the system 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 2 or 4 system clocks. To achieve the required I²C data transfer speed, there exists a relationship between the system clock, f_{SYS}, and the I²C debounce time. For either the I²C Standard or Fast mode operation, users must take care of the selected system clock frequency and the configured debounce time to match the criterion shown in the following table.

I²C Debounce Time Selection	I²C Standard Mode (100kHz)	I²C Fast Mode (400kHz)
No debounce	f_{SYS} > 2MHz	f_{SYS} > 5MHz
2 system clock debounce	f_{SYS} > 4MHz	f_{SYS} > 10MHz
4 system clock debounce	f_{SYS} > 8MHz	f_{SYS} > 20MHz

I²C Minimum f_{SYS} Frequency



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	—	SIMDEB1	SIMDEB0	SIMEN	—
SIMC1	HCF	HANS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
SIMD	D7	D6	D5	D4	D3	D2	D1	D0
SIMA	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	D0

I²C Registers List

SIMC0 Register

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	—	SIMDEB1	SIMDEB0	SIMEN	—
R/W	R/W	R/W	R/W	—	R/W	R/W	R/W	—
POR	1	1	1	—	0	0	0	—

Bit 7~5 **SIM2, SIM1, 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 Unimplemented, read as "0"

Bit 3~2 **SIMDEB1~SIMDEB0:** I^2C Debounce Time Selection
00: No debounce
01: 2 system clock debounce
1x: 4 system clock debounce

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. The SIM configuration option must have first enabled the SIM interface for this bit to be effective. If the SIM is configured to operate as an SPI interface via 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"

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 receiver
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 - must be cleared by the application program after wake-up
This bit should be set to 1 to enable the I²C address match wake up from the SLEEP or IDLE Mode. If the IAMWU bit has been set before entering either the SLEEP or IDLE mode to enable the I²C address match wake up, then this bit must be cleared by the application program after wake-up to ensure correction device operation.

Bit 0 **RXAK:** I²C Bus Receive acknowledge flag
0: Slave receive acknowledge flag
1: Slave does not receive acknowledge flag
The RXAK flag is the receiver acknowledge flag. When the RXAK flag is "0", it means that a acknowledge signal has been received at the 9th clock, after 8 bits of data have been transmitted. When the slave device in the transmit mode, the slave device checks the RXAK flag to determine if the master receiver wishes to receive the next byte. The slave transmitter will therefore continue sending out data until the RXAK flag is "1". When this occurs, the slave transmitter will release the SDA line to allow the master to send a STOP signal to release the I²C Bus.

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 devices write 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 devices 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

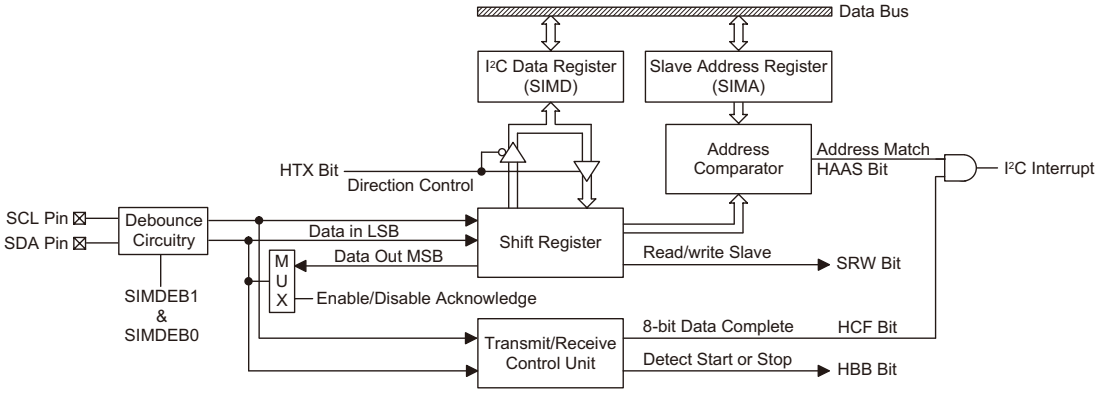
SIMA Register

Bit	7	6	5	4	3	2	1	0
Name	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	—
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": 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
This bit can be read or written by user software program.

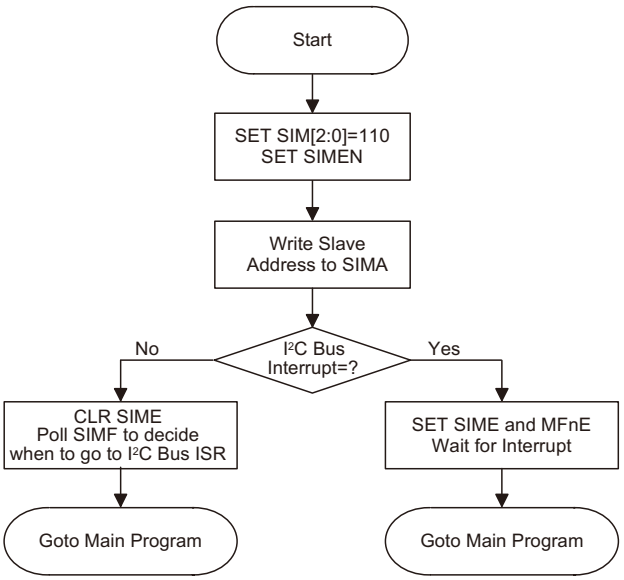


I²C Block Diagram

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 and SIMEN bits in the SIMC0 register to "1" 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 devices 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 slave 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.

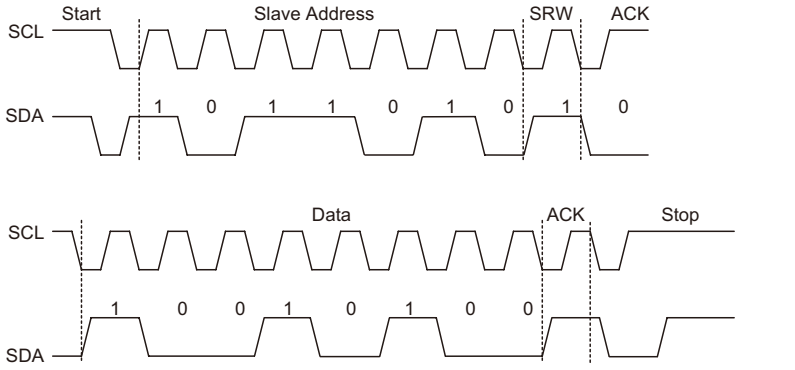
I²C Bus Slave Address Acknowledge Signal

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.

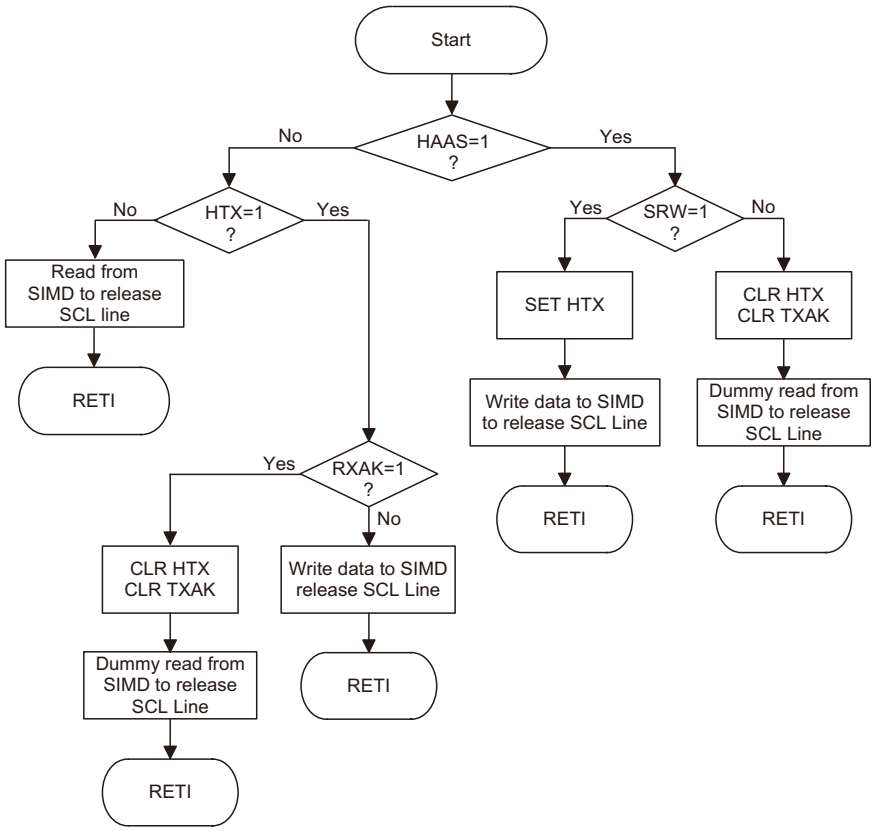


S=Start (1 bit)
SA=Slave Address (7 bits)
SR=SRW bit (1 bit)
M=Slave device send acknowledge bit (1 bit)
D=Data (8 bits)
A=ACK (RXAK bit for transmitter, TXAK bit for receiver 1 bit)
P=Stop (1 bit)

S	SA	SR	M	D	A	D	A	----------	S	SA	SR	M	D	A	D	A	----------	P

Note: *When a slave address is matched, the devices 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.

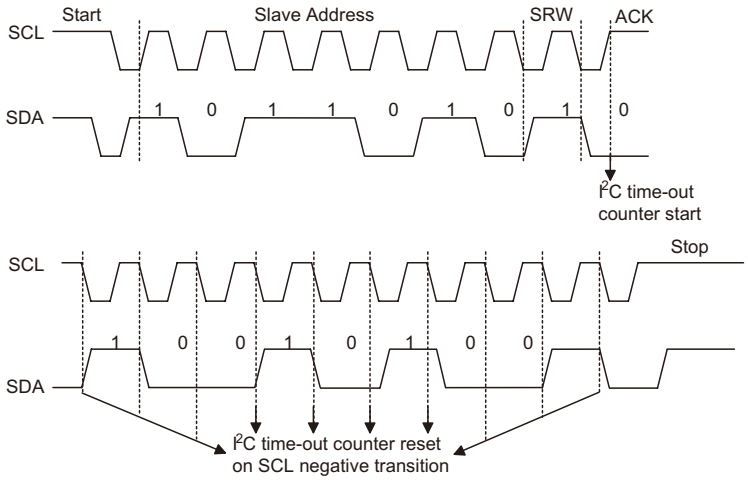
I²C Communication Timing Diagram



I²C Bus ISR flow Chart

I²C Time-out Control

In order to reduce the I²C lockup problem due to reception of erroneous clock sources, a time-out function is provided. If the clock source connected to the I²C bus is not received for a while, then the I²C circuitry and registers will be reset after a certain time-out period. The time-out counter starts to count on an I²C bus "START" & "address match"condition, and is cleared by an SCL falling edge. Before the next SCL falling edge arrives, if the time elapsed is greater than the time-out period specified by the I2CTOC register, then a time-out condition will occur. The time-out function will stop when an I²C "STOP" condition occurs.



I²C Time-out

When an I²C time-out counter overflow occurs, the counter will stop and the I2CTOEN bit will be cleared to zero and the I2CTF bit will be set high to indicate that a time-out condition has occurred. The time-out condition will also generate an interrupt which uses the I²C interrrupt vector. When an I²C time-out occurs, the I²C internal circuitry will be reset and the registers will be reset into the following condition:

Register	After I²C Time-out
SIMD, SIMA, SIMC0	No change
SIMC1	Reset to POR condition

I²C Registers after Time-out

The I2CTOF flag can be cleared by the application program. There are 64 time-out period selections which can be selected using the I2CTOS bits in the I2CTOC register. The time-out duration is calculated by the formula: $((1\sim64) \times (32/f_{SUB}))$. This gives a time-out period which ranges from about 1ms to 64ms.

I2CTOC Register

Bit	7	6	5	4	3	2	1	0
Name	I2CTOEN	I2CTOF	I2CTOS5	I2CTOS4	I2CTOS3	I2CTOS2	I2CTOS1	I2CTOS0
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 **I2CTOEN:** I²C Time-out Control
0: Disable
1: Enable

Bit 6 **I2CTOF:** I²C Time-out flag
0: No time-out occurred
1: Time-out occurred

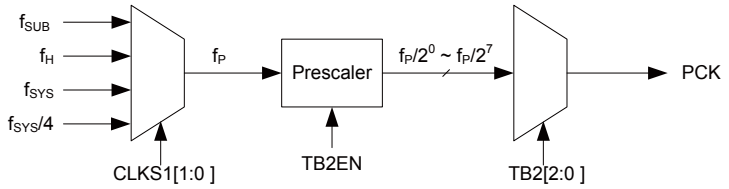
Bit 5~0 **I2CTOS5~I2CTOS0:** I²C Time-out Time Selection
I²C Time-out clock source is $f_{SUB}/32$
I²C Time-out time is given by: (I2CTOS [5:0] +1) × $(32/f_{SUB})$

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 using the relevant pin-shared function selection bit. The Peripheral Clock function is controlled using the TB2EN bit in the TBC2 register. The clock source for the Peripheral Clock Output can originate from the system clock f_{SYS}, the instruction clock, the high speed oscillator clock f_H or the f_{SUB} clock which can be selected by the CLKS11 and CLKS10 bits in the PSC1 register. The TB2EN bit in the TBC2 register is the overall on/off control, setting TB2EN bit to 1 enables the Peripheral Clock while setting TB2EN bit to 0 disables it. The required division ratio of the peripheral clock is selected using the TB22, TB21 and TB20 bits in the TBC2 register. If the peripheral clock source is switched off when the device enters the power down mode, this will disable the Peripheral Clock output.



Peripheral Clock Output

Peripheral Clock Registers

There are two internal registers which control the overall operation of the Peripheral Clock Output. These are the PSC1 and TBC2 registers.

Name	Bit							
	7	6	5	4	3	2	1	0
PSC1	—	—	—	—	—	—	CLKS11	CLKS10
TBC2	TB2EN	—	—	—	—	TB22	TB21	TB20

PCK Register List

PSC1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CLKS11	CLKS10
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **CLKS11, CLKS10:** Peripheral Clock Source selection
00: f_{SYS}
01: $f_{SYS}/4$
10: f_{SUB}
11: f_H

TBC2 Register

Bit	7	6	5	4	3	2	1	0
Name	TB2EN	—	—	—	—	TB22	TB21	TB20
R/W	R/W	—	—	—	—	R/W	R/W	R/W
POR	0	—	—	—	—	0	0	0

Bit 7 **TB2EN:** Peripheral Clock Function enable control
0: Disable
1: Enable

Bit 6~3 Unimplemented, read as "0"

Bit 1~0 **TB22, TB21, TB20:** Peripheral Clock output division selection
000: f_P
001: $f_P/2$
010: $f_P/4$
011: $f_P/8$
100: $f_P/16$
101: $f_P/32$
110: $f_P/64$
111: $f_P/128$

Serial Interface – SPIA

The device contains an independent SPI function. It is important not to confuse this independent SPI function with the additional one contained within the combined SIM function, which is described in another section of this datasheet. This independent SPI function will carry the name SPIA to distinguish it from the other one in the SIM.

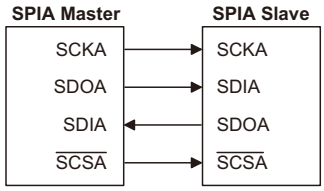
This SPIA interface is often used to communicate with external peripheral devices such as sensors, Flash or EEPROM memory devices, etc. Originally developed by Motorola, the four line SPI interface is a synchronous serial data interface that has a relatively simple communication protocol simplifying 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 SPIA interface specification can control multiple slave devices from a single master, this device is provided only one $\overline{\text{SCSA}}$ pin. If the master needs to control multiple slave devices from a single master, the master can use I/O pins to select the slave devices.

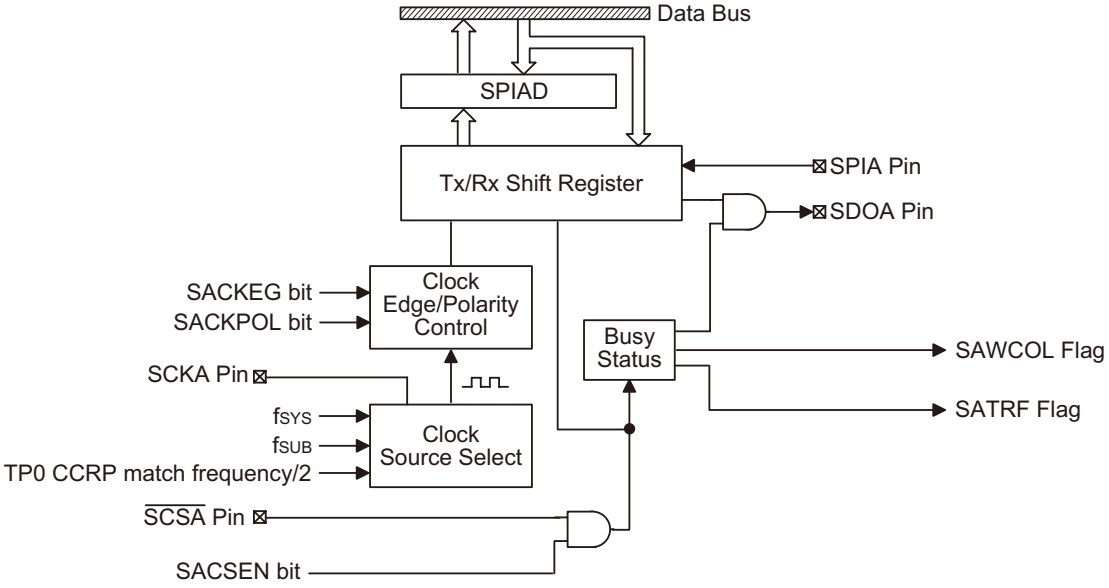
SPIA Interface Operation

The SPIA interface is a full duplex synchronous serial data link. It is a four line interface with pin names SDIA, SDOA, SCKA and $\overline{\text{SCSA}}$. Pins SDIA and SDOA are the Serial Data Input and Serial Data Output lines, SCKA is the Serial Clock line and $\overline{\text{SCSA}}$ is the Slave Select line. As the SPIA interface pins are pin-shared with other functions, the SPIA interface pins must first be selected by configuring the corresponding selection bits in the pin-shared function selection registers. The SPIA interface function is disabled or enabled using the SPIAEN bit in the SPIAC0 register. Communication between devices connected to the SPIA 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 $\overline{\text{SCSA}}$ pin only one slave device can be utilised.

The $\overline{\text{SCSA}}$ pin is controlled by the application program, set the the SACSEN bit to "1" to enable the $\overline{\text{SCSA}}$ pin function and clear the SACSEN bit to "0" to place the $\overline{\text{SCSA}}$ pin into an I/O function.



SPIA Master/Slave Connection



SPIA Block Diagram

The SPIA Serial Interface function includes the following features:

- Full-duplex synchronous data transfer
- Both Master and Slave mode
- LSB first or MSB first data transmission modes
- Transmission complete flag
- Rising or falling active clock edge

The status of the SPIA 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 SACSEN and SPIAEN.

SPIA registers

There are three registers which control the overall operation of the SPIA interface. These are the SPIAD data registers and two control registers SPIAC0 and SPIAC1.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SPIAC0	SASPI2	SASPI1	SASPI0	—	—	—	SPIAEN	—
SPIAC1	—	—	SACKPOL	SACKEG	SAMLS	SACSEN	SAWCOL	SATRF
SPIAD	D7	D6	D5	D4	D3	D2	D1	D0

SPIA Registers List

The SPIAD register is used to store the data being transmitted and received. Before the device writes data to this SPIA bus, the actual data to be transmitted must be placed in the SPIAD register. After the data is received from the SPIA bus, the device can read it from the SPIAD register. Any transmission or reception of data from the SPIA bus must be made via the SPIAD registers.

SPIAD 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	×	×	×	×	×	×	×	×

"×" unknown

There are also two control registers for the SPIA interface, SPIAC0 and SPIAC1. Register SPIAC0 is used to control the enable/disable function and to set the data transmission clock frequency. Register SPIAC1 is used for other control functions such as LSB/MSB selection, write collision flag, etc.

SPIAC0 Register

Bit	7	6	5	4	3	2	1	0
Name	SASPI2	SASPI1	SASPI0	—	—	—	SPIAEN	—
R/W	R/W	R/W	R/W	—	—	—	R/W	—
POR	1	1	1	—	—	—	0	—

Bit 7~5 **SASPI2~SASPI0:** SPIA Master/Slave Clock Select
000: SPIA master, $f_{SYS}/4$
001: SPIA master, $f_{SYS}/16$
010: SPIA master, $f_{SYS}/64$
011: SPIA master, f_{SUB}
100: SPIA master, TP0 CCRP match frequency/2 (PFD)
101: SPIA slave
110: Reserved
111: Reserved

Bit 4~2 Unimplemented, read as "0"

Bit 1 **SPIAEN:** SPIA enable or disable
0: Disable
1: Enable
The bit is the overall on/off control for the SPIA interface. When the SPIAEN bit is cleared to zero to disable the SPIA interface, the SDIA, SDOA, SCKA and $\overline{SCSA}$ lines will lose their SPI function and the SPIA operating current will be reduced to a minimum value. When the bit is high, the SPIA interface is enabled.

Bit 0 Unimplemented, read as "0"

SPIAC1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	SACKPOL	SACKEG	SAMLS	SACSEN	SAWCOL	SATRF
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 **SACKPOL:** Determines the base condition of the clock line
0: SCKA line will be high when the clock is inactive
1: SCKA line will be low when the clock is inactive
The SACKPOL bit determines the base condition of the clock line, if the bit is high, then the SCKA line will be low when the clock is inactive. When the SACKPOL bit is low, then the SCKA line will be high when the clock is inactive.

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

Bit 3 **SAMLS:** data shift order
0: The LSB of data is transmitted first
1: The MSB of data is transmitted first

Bit 2 **SACSEN:** SPIA $\overline{\text{SCSA}}$ pin Control
0: Disable
1: Enable
The SACSEN bit is used as an enable/disable for the $\overline{\text{SCSA}}$ pin. If this bit is low, then the $\overline{\text{SCSA}}$ pin function will be disabled and used as an I/O function. If the bit is high the $\overline{\text{SCSA}}$ pin will be enabled and used as a select pin.

Bit 1 **SAWCOL:** SPIA Write Collision flag
0: Collision free
1: Collision detected
The SAWCOL 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 SPIAD 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.

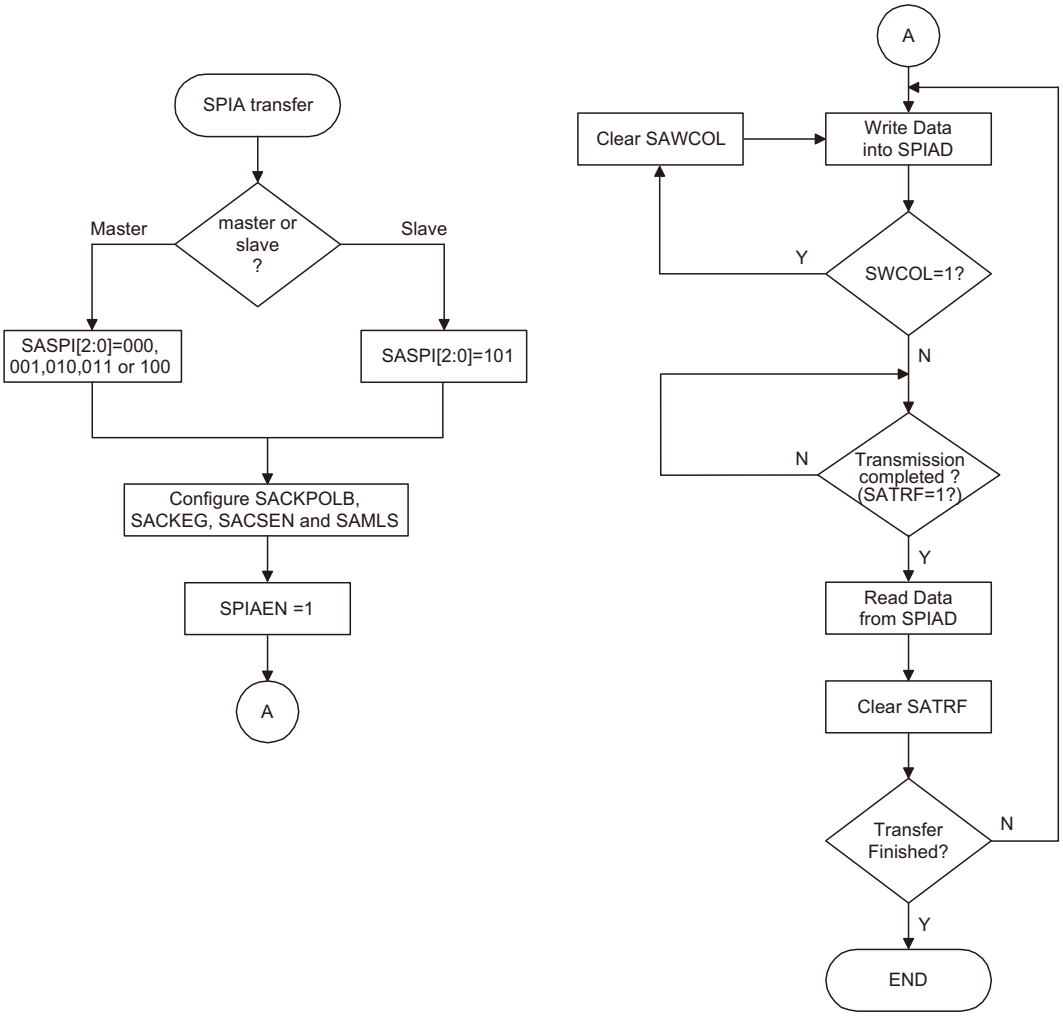
Bit 0 **SATRF:** SPIA Transmit/Receive Complete flag
0: Data is being transferred
1: SPIA data transmission is completed
The SATRF bit is the Transmit/Receive Complete flag and is set to "1" automatically when an SPIA data transmission is completed, but must cleared to "0" by the application program. It can be used to generate an interrupt.

SPIA Communication

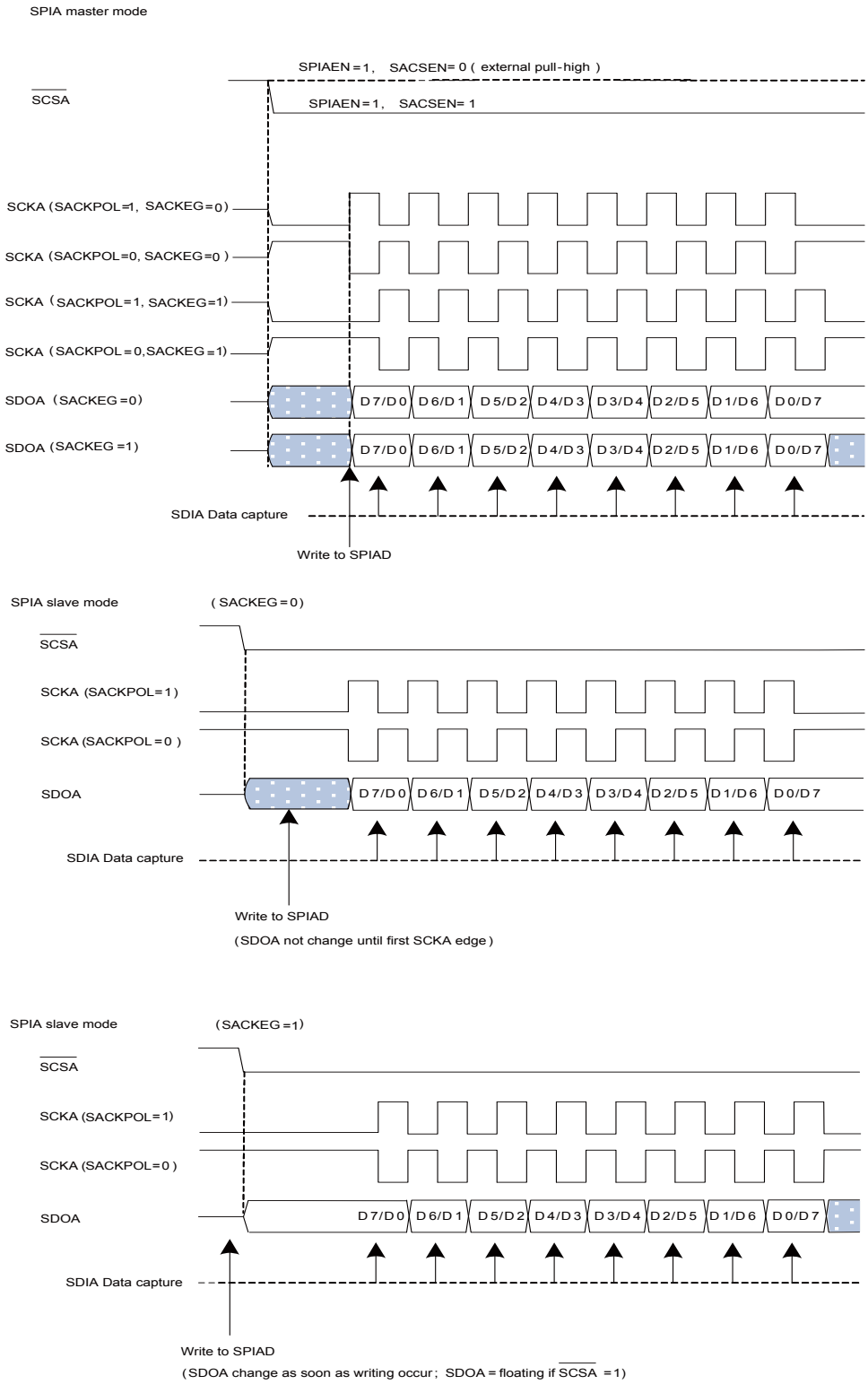
After the SPIA interface is enabled by setting the SPIAEN bit high, then in the Master Mode, when data is written to the SPIAD register, transmission/reception will begin simultaneously. When the data transfer is complete, the SATRF flag will be set automatically, but must be cleared using the application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

The master should output a $\overline{SCSA}$ 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 $\overline{SCSA}$ signal depending upon the configurations of the SACKPOL bit and SACKEG bit. The accompanying timing diagram shows the relationship between the slave data and $\overline{SCSA}$ signal for various configurations of the SACKPOL and SACKEG bits.

The SPIA will continue to function even in the IDLE Mode.



SPIA Transfer Control Flowchart



Note :
For SPIA slave mode, if SPIAEN=1 and SACSEN=0, SPIA is always enabled and ignore the $\overline{SCSA}$ level.

SPIA Master/Slave ModeTiming Diagram

SPIA Bus Enable/Disable

To enable the SPIA bus, set SACSEN=1 and $\overline{SCSA}$=0, then wait for data to be written into the SPIAD (TXRX buffer) register. For the Master Mode, after data has been written to the SPIAD (TXRX buffer) register, then transmission or reception will start automatically. When all the data has been transferred the SATRF bit should be set. For the Slave Mode, when clock pulses are received on SCKA, data in the TXRX buffer will be shifted out or data on SDIA will be shifted in.

To Disable the SPIA bus SCKA, SDIA, SDOA, $\overline{SCSA}$ will become I/O pins or other pin-shared functions.

SPIA Operation

All communication is carried out using the 4-line interface for either Master or Slave Mode.

The SACSEN bit in the SPIAC1 register controls the overall function of the SPIA interface. Setting this bit high will enable the SPIA interface by allowing the $\overline{SCSA}$ line to be active, which can then be used to control the SPIA interface. If the SACSEN bit is low, the SPIA interface will be disabled and the $\overline{SCSA}$ line will be an I/O pin or other pin-shared functions and can therefore not be used for control of the SPIA interface. If the SACSEN bit and the SPIAEN bit in the SPIAC0 register are set high, this will place the SDIA line in a floating condition and the SDOA line high. If in Master Mode the SCKA line will be either high or low depending upon the clock polarity selection bit SACKPOL in the SPIAC1 register. If in Slave Mode the SCKA line will be in a floating condition. If SPIAEN is low then the bus will be disabled and $\overline{SCSA}$, SDIA, SDOA and SCKA pins will all become I/O pins or other pin-shared functions. In the Master Mode the Master will always generate the clock signal. The clock and data transmission will be initiated after data has been written into the SPIAD register. In the Slave Mode, the clock signal will be received from an external master device for both data transmission and reception. The following sequences show the order to be followed for data transfer in both Master and Slave Mode.

Master Mode:

- Step 1
  Select the clock source and Master mode using the SASPI2~SASPI0 bits in the SPIAC0 control register
- Step 2
  Setup the SACSEN bit and setup the SAMLS bit to choose if the data is MSB or LSB shifted first, this must be same as the Slave device.
- Step 3
  Setup the SPIAEN bit in the SPIAC0 control register to enable the SPIA interface.
- Step 4
  For write operations: write the data to the SPIAD register, which will actually place the data into the TXRX buffer. Then use the SCKA and $\overline{SCSA}$ lines to output the data. After this go to step 5.
  For read operations: the data transferred in on the SDIA line will be stored in the TXRX buffer until all the data has been received at which point it will be latched into the SPIAD register.
- Step 5
  Check the SAWCOL bit if set high then a collision error has occurred so return to step 4. If equal to zero then go to the following step.
- Step 6
  Check the SATRF bit or wait for a SPIA serial bus interrupt.

- Step 7
  Read data from the SPIAD register.
- Step 8
  Clear SATRF.
- Step 9
  Go to step 4.

Slave Mode:

- Step 1
  Select the SPI Slave mode using the SASPI2~SASPI0 bits in the SPIAC0 control register
- Step 2
  Setup the SACSEN bit and setup the SAMLS bit to choose if the data is MSB or LSB shifted first, this setting must be the same with the Master device.
- Step 3
  Setup the SPIAEN bit in the SPIAC0 control register to enable the SPIA interface.
- Step 4
  For write operations: write the data to the SPIAD register, which will actually place the data into the TXRX buffer. Then wait for the master clock SCKA and $\overline{\text{SCSA}}$ signal. After this, go to step 5.
  For read operations: the data transferred in on the SDIA line will be stored in the TXRX buffer until all the data has been received at which point it will be latched into the SPIAD register.
- Step 5
  Check the SAWCOL bit if set high then a collision error has occurred so return to step 4. If equal to zero then go to the following step.
- Step 6
  Check the SATRF bit or wait for a SPIA serial bus interrupt.
- Step 7
  Read data from the SPIAD register.
- Step 8
  Clear SATRF.
- Step 9
  Go to step 4.

Error Detection

The SAWCOL bit in the SPIAC1 register is provided to indicate errors during data transfer. The bit is set by the SPIA serial Interface but must be cleared by the application program. This bit indicates a data collision has occurred which happens if a write to the SPIAD register takes place during a data transfer operation and will prevent the write operation from continuing.

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. These devices contain several external interrupt and internal interrupts functions. The external interrupts are generated by the action of the external INT0~INT3 and $\overline{\text{PINT}}$ pins, while the internal interrupts are generated by various internal functions such as the TMs, Comparators,Time Base, LVD, EEPROM, SIM 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, as shown in the accompanying table. The number of registers depends upon the device chosen but fall into three categories. The first is the INTC0~INTC3 registers which setup the primary interrupts, the second is the MFI0~MFI4 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=0~3
Comparator	CPnE	CPnF	n=0~1
A/D Converter	ADE	ADF	—
Time Base	TBnE	TBnF	n=0~1
Multi-function	MFnE	MFnF	n=0~4
SIM	SIME	SIMF	—
LVD	LVE	LVF	—
EEPROM	DEE	DEF	—
$\overline{\text{PINT}}$	XPE	XPF	—
SPIA	SPIAE	SPIAF	—
TM	TnPE	TnPF	n=0~5
	TnAE	TnAF	n=0~5
	TnBE	TnBF	n=1

Interrupt Register Bit Naming Conventions

Interrupt Register Contents

Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	INT3S1	INT3S0	INT2S1	INT2S0	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	CP0F	INT1F	INT0F	CP0E	INT1E	INT0E	EMI
INTC1	ADF	MF1F	MF0F	CP1F	ADE	MF1E	MF0E	CP1E
INTC2	MF3F	TB1F	TB0F	MF2F	MF3E	TB1E	TB0E	MF2E
INTC3	—	MF4F	INT3F	INT2F	—	MF4E	INT3E	INT2E
MFI0	T2AF	T2PF	TnAF	TnPF	T2AE	T2PE	T0AE	T0PE
MFI1	—	T1BF	T1AF	T1PF	—	T1BE	T1AE	T1PE
MFI2	SIMF	T3AF	T3PF	XPF	SIME	T3AE	T3PE	XPE
MFI3	—	SPIAF	DEF	LVF	—	SPIAE	DEE	LVE
MFI4	T5AF	T5PF	T4AF	T4PF	T5AE	T5PE	T4AE	T4PE

INTEG Register

Bit	7	6	5	4	3	2	1	0
Name	INT3S1	INT3S0	INT2S1	INT2S0	INT1S1	INT1S0	INT0S1	INT0S0
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 **INT3S1, INT3S0:** interrupt edge control for INT3 pin
00: Disable
01: Rising edge
10: Falling edge
11: Rising and falling edges

Bit 5~4 **INT2S1, INT2S0:** interrupt edge control for INT2 pin
00: Disable
01: Rising edge
10: Falling edge
11: Rising and falling edges

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

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

INTC0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	CP0F	INT1F	INT0F	CP0E	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 **CP0F:** Comparator 0 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 **CP0E:** Comparator 0 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

Bit	7	6	5	4	3	2	1	0
Name	ADF	MF1F	MF0F	CP1F	ADE	MF1E	MF0E	CP1E
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 **ADF:** A/D Converter interrupt request flag
0: No request
1: Interrupt request

Bit 6 **MF1F:** Multi-function interrupt 1 request flag
0: No request
1: Interrupt request

Bit 5 **MF0F:** Multi-function interrupt 0 request flag
0: No request
1: Interrupt request

Bit 4 **CP1F:** Comparator 1 interrupt request flag
0: No request
1: Interrupt request

Bit 3 **ADE:** A/D Converter interrupt control
0: Disable
1: Enable

Bit 2 **MF1E:** Multi-function interrupt 1 control
0: Disable
1: Enable

Bit 1 **MF0E:** Multi-function interrupt 0 control
0: Disable
1: Enable

Bit 0 **CP1E:** Comparator 1 interrupt control
0: Disable
1: Enable

INTC2 Register

Bit	7	6	5	4	3	2	1	0
Name	MF3F	TB1F	TB0F	MF2F	MF3E	TB1E	TB0E	MF2E
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 **TB1F:** Time Base 1 interrupt request flag
0: No request
1: Interrupt request

Bit 5 **TB0F:** Time Base 0 interrupt request flag
0: No request
1: Interrupt request

Bit 4 **MF2F:** Multi-function interrupt 2 request flag
0: No request
1: Interrupt request

Bit 3 **MF3E:** Multi-function interrupt 3 control
0: Disable
1: Enable

Bit 2 **TB1E:** Time Base 1 interrupt control
0: Disable
1: Enable

Bit 1 **TB0E:** Time Base 0 interrupt control
0: Disable
1: Enable

Bit 0 **MF2E:** Multi-function interrupt 2 control
0: Disable
1: Enable

INTC3 Register

Bit	7	6	5	4	3	2	1	0
Name	—	MF4F	INT3F	INT2F	—	MF4E	INT3E	INT2E
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 **MF4F:** Multi-function interrupt 4 request flag
0: No request
1: Interrupt request

Bit 5 **INT3F:** INT3 interrupt request flag
0: No request
1: Interrupt request

Bit 4 **INT2F:** INT2 interrupt request flag
0: No request
1: Interrupt request

Bit 3 Unimplemented, read as "0"

Bit 2 **MF4E:** Multi-function interrupt 4 control
0: Disable
1: Enable

Bit 1 **INT3E:** INT3 interrupt control
0: Disable
1: Enable

Bit 0 **INT2E:** INT2 interrupt control
0: Disable
1: Enable

MFI0 Register

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

Bit	7	6	5	4	3	2	1	0
Name	—	T1BF	T1AF	T1PF	—	T1BE	T1AE	T1PE
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 **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 Unimplemented, read as "0"

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

Bit	7	6	5	4	3	2	1	0
Name	SIMF	T3AF	T3PF	XPF	SIME	T3AE	T3PE	XPE
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 **SIMF:** SIM interrupt request flag
0: No request
1: Interrupt request

Bit 6 **T3AF:** TM3 Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 5 **T3PF:** TM3 Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **XPF:** External peripheral interrupt request flag
0: No request
1: Interrupt request

Bit 3 **SIME:** SIM Interrupt Control
0: Disable
1: Enable

Bit 2 **T3AE:** TM3 Comparator A match interrupt control
0: Disable
1: Enable

Bit 1 **T3PE:** TM3 Comparator P match interrupt control
0: Disable
1: Enable

Bit 0 **XPE:** External peripheral Interrupt Control
0: Disable
1: Enable

MFI3 Register

Bit	7	6	5	4	3	2	1	0
Name	—	SPIAF	DEF	LVF	—	SPIAE	DEE	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 **SPIAF:** SPIA interrupt request flag
0: No request
1: Interrupt request

Bit 5 **DEF:** Data EEPROM interrupt request flag
0: No request
1: Interrupt request

Bit 4 **LVF:** LVD interrupt request flag
0: No request
1: Interrupt request

Bit 3 Unimplemented, read as "0"

Bit 2 **SPIAE:** SPIA Interrupt Control
0: Disable
1: Enable

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

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

MFI4 Register

Bit	7	6	5	4	3	2	1	0
Name	T5AF	T5PF	T4AF	T4PF	T5AE	T5PE	T4AE	T4PE
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 **T5AF:** TM5 Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 6 **T5PF:** TM5 Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 5 **T4AF:** TM4 Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **T4PF:** TM4 Comparator P match interrupt request flag
0: No request
1: Interrupt request

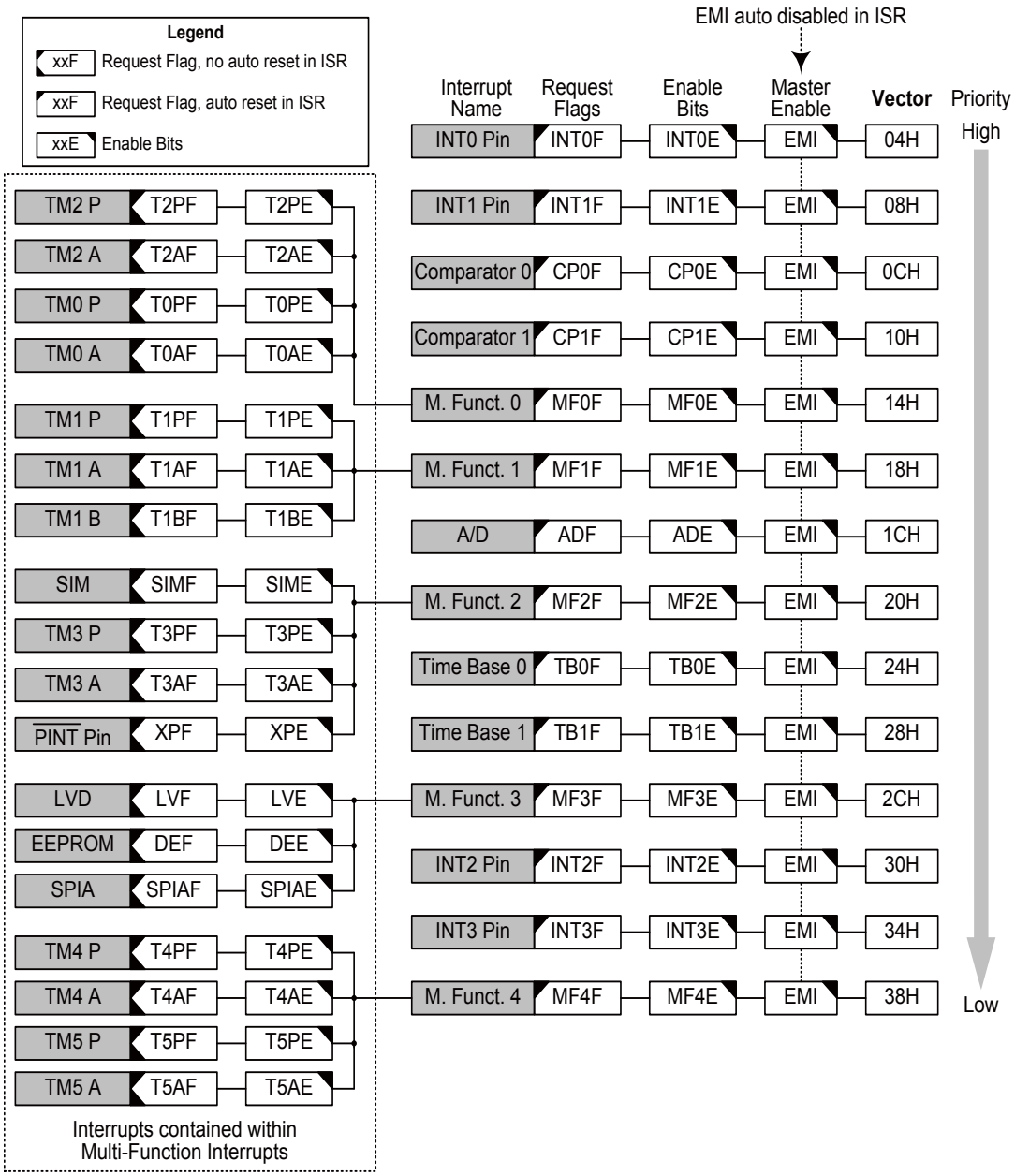
Bit 3 **T5AE:** TM5 Comparator A match interrupt control
0: Disable
1: Enable

Bit 2 **T5PE:** TM5 Comparator P match interrupt control
0: Disable
1: Enable

Bit 1 **T4AE:** TM4 Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **T4PE:** TM4 Comparator P match interrupt control
0: Disable
1: Enable

Interrupt Operation



Interrupt Structure

External Interrupt

The external interrupts are controlled by signal transitions on the pins INT0~INT3. An external interrupt request will take place when the external interrupt request flags, INT0F~INT3F are set, which will occur when a transition, whose type is chosen by the edge select bits, appears on the external interrupt pins. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and respective external interrupt enable bit, INT0E~INT3E, 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~INT3F, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts. Note that any pull-high resistor selections on the external interrupt pins will remain valid even if the pin is used as an external interrupt input.

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

Comparator Interrupt

The comparator interrupts are controlled by the two internal comparators. A comparator interrupt request will take place when the comparator interrupt request flags, CP0F or CP1F, are set, a situation that will occur when the comparator output changes state. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and comparator interrupt enable bits, CP0E and CP1E, must first be set. When the interrupt is enabled, the stack is not full and the comparator inputs generate a comparator output transition, a subroutine call to the comparator interrupt vector, will take place. When the interrupt is serviced, the external interrupt request flags, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts.

Multi-function Interrupt

Within these devices are five 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, SIM Interrupt, External Peripheral Interrupt, LVD interrupt and EEPROM Interrupt.

A Multi-function interrupt request will take place when any of the Multi-function interrupt request flags, MF0F~MF5F, are set. The Multi-function interrupt flags will be set when any of their included functions generate an interrupt request flag. To allow the program to branch to its respective interrupt vector address, when the Multi-function interrupt is enabled and the stack is not full, and either one of the interrupts contained within each of Multi-function interrupt occurs, a subroutine call to one of the Multi-function interrupt vectors will take place. When the interrupt is serviced, the related Multi-Function request flag, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts.

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

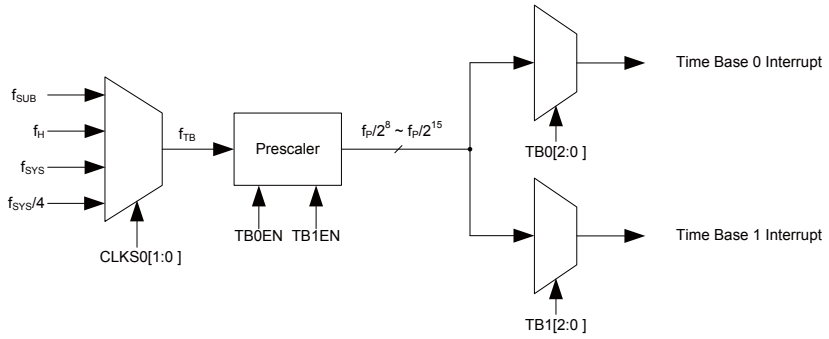
A/D Converter Interrupt

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 Interrupt

The function of the Time Base Interrupt is to provide regular time signal in the form of an internal interrupt. It is controlled by the overflow signal from its internal timer. When this happens its interrupt request flag, TBnF, will be set. To allow the program to branch to its respective interrupt vector addresses, the global interrupt enable bit, EMI and Time Base enable bit, TBnE, must first be set. When the interrupt is enabled, the stack is not full and the Time Base overflows, a subroutine call to its respective vector location will take place. When the interrupt is serviced, the interrupt request flag, TBnF, will be automatically reset and the EMI bit will be cleared to disable other interrupts.

The purpose of the Time Base Interrupt is to provide an interrupt signal at fixed time periods. Its clock source, f_{TB}, originates from the internal clock source f_{SUB}, $f_{SYS}/4$, f_{SYS} or f_H and then passes through a divider, the division ratio of which is selected by programming the appropriate bits in the TBC0 and TBC1 registers to obtain longer interrupt periods whose value ranges. The clock source which in turn controls the Time Base interrupt period is selected using the CLKS01 and CLKS00 bits in the PSC0 register.



Time Base Interrupt

PSC0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CLKS01	CLKS00
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **CLKS01~CLKS00:** Time Base clock source Selection
00: f_{SYS}
01: $f_{SYS}/4$
10: f_{TB}
11: f_H

TBC0 Register

Bit	7	6	5	4	3	2	1	0
Name	TB0ON	—	—	—	—	TB02	TB01	TB00
R/W	R/W	—	—	—	—	R/W	R/W	R/W
POR	0	—	—	—	—	0	0	0

Bit 7 **TB0ON:** Time Base 0 Enable/Disable Control
0: Disable
1: Enable

Bit 6~3 Unimplemented, read as "0"

Bit 2~0 **TB02~TB00:** Time Base 0 Time-out Period
000: $2^8/f_{TB}$
001: $2^9/f_{TB}$
010: $2^{10}/f_{TB}$
011: $2^{11}/f_{TB}$
100: $2^{12}/f_{TB}$
101: $2^{13}/f_{TB}$
110: $2^{14}/f_{TB}$
111: $2^{15}/f_{TB}$

TBC1 Register

Bit	7	6	5	4	3	2	1	0
Name	TB1ON	—	—	—	—	TB12	TB11	TB10
R/W	R/W	—	—	—	—	R/W	R/W	R/W
POR	0	—	—	—	—	0	0	0

Bit 7 **TB1ON:** Time Base 1 Enable/Disable Control
0: Disable
1: Enable

Bit 6~3 Unimplemented, read as "0"

Bit 2~0 **TB12~TB10:** Time Base 1 Time-out Period
000: $2^8/f_{TB}$
001: $2^9/f_{TB}$
010: $2^{10}/f_{TB}$
011: $2^{11}/f_{TB}$
100: $2^{12}/f_{TB}$
101: $2^{13}/f_{TB}$
110: $2^{14}/f_{TB}$
111: $2^{15}/f_{TB}$

Serial Interface Module Interrupts

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.

SPIA Interface Interrupt

The SPIA Interface Interrupt is contained within the Multi-function Interrupt. A SPIA Interrupt request will take place when the SPIA Interrupt request flag, SPIAF, is set, which occurs when a byte of data has been received or transmitted by the SPIA interface. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the SPIA Interface Interrupt enable bit, SPIAE, 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 SPIA interface, a subroutine call to the respective Multi-function Interrupt vector, will take place. When the SPIA 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 SPIAF 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 $\overline{\text{PINT}}$ 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.

EEPROM Interrupt

The EEPROM Interrupt, is contained within the Multi-function Interrupt. An EEPROM Interrupt request will take place when the EEPROM Interrupt request flag, DEF, is set, which occurs when an EEPROM Write cycle ends. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, EEPROM Interrupt enable bit, DEE, and associated Multi-function interrupt enable bit, must first be set. When the interrupt is enabled, the stack is not full and an EEPROM Write cycle ends, a subroutine call to the respective Multi-function Interrupt vector, will take place. When the EEPROM Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared. As the DEF flag will not be automatically cleared, it has to be cleared by the application program.

LVD Interrupt

The Low Voltage Detector Interrupt is contained within the Multi-function Interrupt. An LVD Interrupt request will take place when the LVD Interrupt request flag, LVF, is set, which occurs when the Low Voltage Detector function detects a low power supply voltage. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, Low Voltage Interrupt enable bit, LVE, and associated Multi-function interrupt enable bit, must first be set. When the interrupt is enabled, the stack is not full and a low voltage condition occurs, a subroutine call to the Multi-function Interrupt vector, will take place. When the Low Voltage Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared. As the LVF flag will not be automatically cleared, it has to be cleared by the application program.

TM Interrupts

The Compact TM has two interrupts, while the Enhanced Type TM has three interrupts. All of the TM interrupts are contained within the Multi-function Interrupts. For the Compact Type TM 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 its respective interrupt vector address, the global interrupt enable bit, EMI, respective TM Interrupt enable bit, and relevant Multi-function Interrupt enable bit, MFnE, must first be set. When the interrupt is enabled, the stack is not full and a TM comparator match situation occurs, a subroutine call to the relevant Multi-function Interrupt vector locations, will take place. When the TM interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the related MFnF flag will be automatically cleared. As the TM interrupt request flags will not be automatically cleared, they have to be cleared by the application program.

Interrupt Wake-up Function

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

Programming Considerations

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

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

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

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

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

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

Low Voltage Detector – LVD

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

LVD Register

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

LVDC Register

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

Bit 7~6 Unimplemented, read as "0"

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

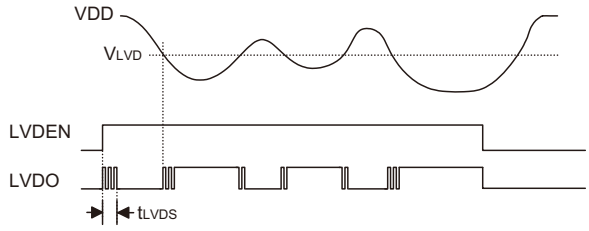
Bit 4 **LVDEN:** Low Voltage Detector Enable/Disable
0: Disable
1: Enable

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 voltage which 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. When the device is powered down the Low Voltage Detector will remain active if the LVDEN bit is high. In this case, the LVF interrupt request flag will be set, causing an interrupt to be generated if V_{DD} falls below the preset LVD voltage. This will cause the device to wake-up from the SLEEP or IDLE Mode, however if the Low Voltage Detector wake up function is not required then the LVF flag should be first set high before the device enters the SLEEP or IDLE Mode.

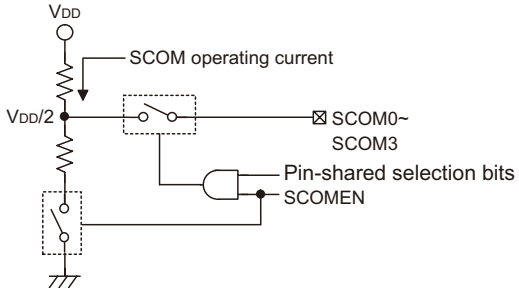
SCOM Function for LCD

The devices have the capability of driving external LCD panels. The common pins for LCD driving, SCOM0~SCOM3, are pin shared with the PC0~PC1, PC6~PC7 pins. The LCD signals (COM and SEG) are generated using the application program.

LCD Operation

An external LCD panel can be driven using this device by configuring the PC0~PC1, PC6~PC7 pins as common pins and using other output ports lines as segment pins. The LCD driver function is controlled using the SCOMC register which in addition to controlling the overall on/off function also controls the bias voltage setup function. This enables the LCD COM driver to generate the necessary $V_{DD}/2$ voltage levels for LCD 1/2 bias operation.

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



LCD COM Bias

LCD Bias Control

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

SCOMC Register

Bit	7	6	5	4	3	2	1	0
Name	D7	ISEL1	ISEL0	SCOMEN	—	—	—	—
R/W	R/W	R/W	R/W	R/W	—	—	—	—
POR	0	0	0	0	—	—	—	—

Bit 7 Reserved Bit
0: Correct level - bit must be reset to zero for correct operation
1: Unpredictable operation - bit must not be set high

Bit 6~5 **ISEL1, ISEL0:** Select SCOM typical bias current (V_{DD}=5V)
00: 25μA
01: 50μA
10: 100μA
11: 200μA

Bit 4 **SCOMEN:** SCOM module Control
0: Disable
1: Enable

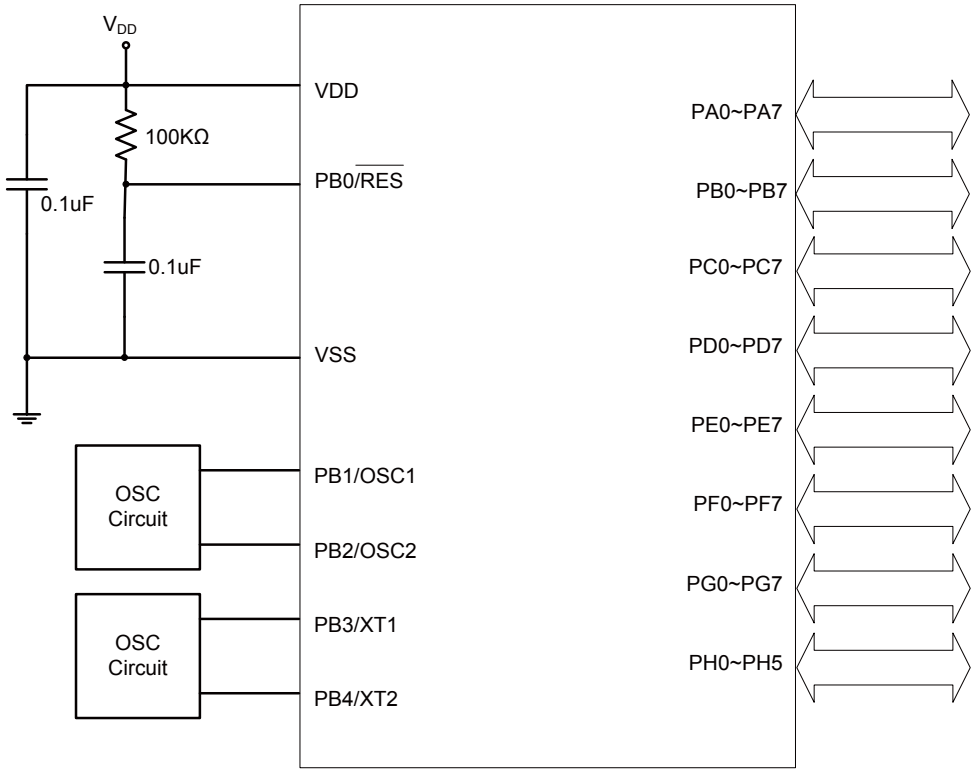
Bit 3~0 Unimplemented, read as "0"

Configuration Options

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

No.	Options
1	High Speed System Oscillator Selection – f_H HXT, ERC or HIRC
2	Low Speed System Oscillator Selection – f_{SUB} LXT or LIRC
3	I/O or Reset pin selection Reset pin or I/O pin

Application Circuits



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 section 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
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			
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
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			
TABRD [m]	Read table 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 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 up to three cycles are required, if no skip takes place only one cycle is required.

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

3. For the "CLR WDT" instruction the TO and PDF flags may be affected by the execution status. The TO and PDF flags are cleared after the "CLR WDT" instructions is executed. Otherwise the TO and PDF flags remain unchanged.

Extended Instruction Set

The extended instructions are used to support the full range address access for the data memory. When the accessed data memory is located in any data memory sections except section 0, the extended instruction can be used to access the data memory instead of using the indirect addressing access to improve the CPU firmware performance.

Mnemonic	Description	Cycles	Flag Affected
Arithmetic			
LADD A,[m]	Add Data Memory to ACC	2	Z, C, AC, OV, SC
LADDM A,[m]	Add ACC to Data Memory	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 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 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 up to four cycles are required, if no skip takes place two cycles is required.

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

Instruction Definition

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

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

Operation ACC ← ACC + [m] + C

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

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

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

Operation [m] ← ACC + [m] + C

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

ADD A,[m] Add Data Memory to ACC

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

Operation ACC ← ACC + [m]

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

ADD A,x Add immediate data to ACC

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

Operation ACC ← ACC + x

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

ADDM A,[m] Add ACC to Data Memory

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

Operation [m] ← ACC + [m]

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

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

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

Operation ACC ← ACC "AND" [m]

Affected flag(s) Z

AND A,x Logical AND immediate data to ACC

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

Operation ACC ← ACC "AND" x

Affected flag(s) Z

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

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

Operation [m] ← ACC "AND" [m]

Affected flag(s) Z

CALL addr Subroutine call

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

Operation Stack ← Program Counter + 1
Program Counter ← addr

Affected flag(s) None

CLR [m] Clear Data Memory

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

Operation [m] ← 00H

Affected flag(s) None

CLR [m].i Clear bit of Data Memory

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

Operation [m].i ← 0

Affected flag(s) None

CLR WDT Clear Watchdog Timer

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

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CLR WDT1 Pre-clear Watchdog Timer

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

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CLR WDT2 Pre-clear Watchdog Timer

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

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CPL [m] Complement Data Memory

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

Operation [m] ← $\overline{[m]}$

Affected flag(s) Z

CPLA [m] Complement Data Memory with result in ACC

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

Operation ACC ← $\overline{[m]}$

Affected flag(s) Z

DAA [m] Decimal-Adjust ACC for addition with result in Data Memory

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

Operation [m] ← ACC + 00H or
[m] ← ACC + 06H or
[m] ← ACC + 60H or
[m] ← ACC + 66H

Affected flag(s) C

DEC [m] Decrement Data Memory

Description Data in the specified Data Memory is decremented by 1.

Operation [m] ← [m] − 1

Affected flag(s) Z

DECA [m] Decrement Data Memory with result in ACC

Description Data in the specified Data Memory is decremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC ← [m] − 1

Affected flag(s) Z

HALT Enter power down mode

Description This instruction stops the program execution and turns off the system clock. The contents of the Data Memory and registers are retained. The WDT and prescaler are cleared. The power down flag PDF is set and the WDT time-out flag TO is cleared.

Operation TO ← 0
PDF ← 1

Affected flag(s) TO, PDF

INC [m] Increment Data Memory

Description Data in the specified Data Memory is incremented by 1.

Operation [m] ← [m] + 1

Affected flag(s) Z

INCA [m] Increment Data Memory with result in ACC

Description Data in the specified Data Memory is incremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC ← [m] + 1

Affected flag(s) Z

JMP addr Jump unconditionally

Description The contents of the Program Counter are replaced with the specified address. Program execution then continues from this new address. As this requires the insertion of a dummy instruction while the new address is loaded, it is a two cycle instruction.

Operation Program Counter ← addr

Affected flag(s) None

MOV A,[m] Move Data Memory to ACC

Description The contents of the specified Data Memory are copied to the Accumulator.

Operation ACC ← [m]

Affected flag(s) None

MOV A,x Move immediate data to ACC

Description The immediate data specified is loaded into the Accumulator.

Operation ACC ← x

Affected flag(s) None

MOV [m],A Move ACC to Data Memory

Description The contents of the Accumulator are copied to the specified Data Memory.

Operation [m] ← ACC

Affected flag(s) None

NOP No operation

Description No operation is performed. Execution continues with the next instruction.

Operation No operation

Affected flag(s) None

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

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

Operation ACC ← ACC ″OR″ [m]

Affected flag(s) Z

OR A,x Logical OR immediate data to ACC

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

Operation ACC ← ACC ″OR″ x

Affected flag(s) Z

ORM A,[m] Logical OR ACC to Data Memory

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

Operation [m] ← ACC ″OR″ [m]

Affected flag(s) Z

RET Return from subroutine

Description The Program Counter is restored from the stack. Program execution continues at the restored address.

Operation Program Counter ← Stack

Affected flag(s) None

RET A,x Return from subroutine and load immediate data to ACC

Description The Program Counter is restored from the stack and the Accumulator loaded with the specified immediate data. Program execution continues at the restored address.

Operation Program Counter ← Stack
ACC ← x

Affected flag(s) None

RETI Return from interrupt

Description The Program Counter is restored from the stack and the interrupts are re-enabled by setting the EMI bit. EMI is the master interrupt global enable bit. If an interrupt was pending when the RETI instruction is executed, the pending Interrupt routine will be processed before returning to the main program.

Operation Program Counter ← Stack
EMI ← 1

Affected flag(s) None

RL [m] Rotate Data Memory left

Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0.

Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← [m].7

Affected flag(s) None

RLA [m] Rotate Data Memory left with result in ACC

Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← [m].7

Affected flag(s) None

RLC [m] Rotate Data Memory left through Carry

Description The contents of the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into bit 0.

Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← C
C ← [m].7

Affected flag(s) C

RLCA [m] Rotate Data Memory left through Carry with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into the bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← C
C ← [m].7

Affected flag(s) C

RR [m] Rotate Data Memory right

Description The contents of the specified Data Memory are rotated right by 1 bit with bit 0 rotated into bit 7.

Operation [m].i ← [m].(i+1); (i=0~6)
[m].7 ← [m].0

Affected flag(s) None

RRA [m] Rotate Data Memory right with result in ACC

Description: Data in the specified Data Memory and the carry flag are rotated right by 1 bit with bit 0 rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation:
ACC.i ← [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] - $\overline{C}$

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

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

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

Operation: [m] ← ACC − [m] − C

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

SDZ [m] Skip if decrement Data Memory is 0

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

Operation [m] ← [m] − 1
Skip if [m]=0

Affected flag(s) None

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

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

Operation ACC ← [m] − 1
Skip if ACC=0

Affected flag(s) None

SET [m] Set Data Memory

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

Operation [m] ← FFH

Affected flag(s) None

SET [m].i Set bit of Data Memory

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

Operation [m].i ← 1

Affected flag(s) None

SIZ [m] Skip if increment Data Memory is 0

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

Operation [m] ← [m] + 1
Skip if [m]=0

Affected flag(s) None

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

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

Operation ACC ← [m] + 1
Skip if ACC=0

Affected flag(s) None

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

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

Operation Skip if [m].i ≠ 0

Affected flag(s) None

SNZ [m] Skip if Data Memory is not 0

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

Operation Skip if [m]≠ 0

Affected flag(s) None

SUB A,[m] Subtract Data Memory from ACC

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

Operation ACC ← ACC − [m]

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

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

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

Operation [m] ← ACC − [m]

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

SUB A,x Subtract immediate data from ACC

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

Operation ACC ← ACC − x

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

SWAP [m] Swap nibbles of Data Memory

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

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s) None

SWAPA [m] Swap nibbles of Data Memory with result in ACC

Description The low-order and high-order nibbles of the specified Data Memory are interchanged. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC.3~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0

Affected flag(s) None

SZ [m] Skip if Data Memory is 0

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

Operation Skip if [m]=0

Affected flag(s) None

SZA [m] Skip if Data Memory is 0 with data movement to ACC

Description The contents of the specified Data Memory are copied to the Accumulator. If the value is zero, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation ACC ← [m]
Skip if [m]=0

Affected flag(s) None

SZ [m].i Skip if bit i of Data Memory is 0

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

Operation Skip if [m].i=0

Affected flag(s) None

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

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

ITABRD [m] Increment table pointer low byte first and read table to TBLH and Data Memory

Description Increment table pointer low byte, TBLP, first and then the program code addressed by the table pointer (TBHP and TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

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

Affected flag(s) None

ITABRDL [m] Increment table pointer low byte first and read table (last page) to TBLH and Data Memory

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

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

Affected flag(s) None

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

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

Operation ACC ← ACC ″XOR″ [m]

Affected flag(s) Z

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

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

Extended Instruction Definition

The extended instructions are used to directly access the data stored in any data memory sections.

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

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

Operation ACC ← ACC + [m] + C

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

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

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

Operation [m] ← ACC + [m] + C

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

LADD A,[m] Add Data Memory to ACC

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

Operation ACC ← ACC + [m]

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

LADDM A,[m] Add ACC to Data Memory

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

Operation [m] ← ACC + [m]

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

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

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

Operation ACC ← ACC "AND" [m]

Affected flag(s) Z

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

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

Operation [m] ← ACC "AND" [m]

Affected flag(s) Z

LCLR [m] Clear Data Memory

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

Operation [m] ← 00H

Affected flag(s) None

LCLR [m].i Clear bit of Data Memory

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

Operation [m].i ← 0

Affected flag(s) None

LCPL [m] Complement Data Memory

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

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

Affected flag(s) Z

LCPLA [m] Complement Data Memory with result in ACC

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

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

Affected flag(s) Z

LDAA [m] Decimal-Adjust ACC for addition with result in Data Memory

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

Operation $[m] \leftarrow ACC + 00H$ or
$[m] \leftarrow ACC + 06H$ or
$[m] \leftarrow ACC + 60H$ or
$[m] \leftarrow ACC + 66H$

Affected flag(s) C

LDEC [m] Decrement Data Memory

Description Data in the specified Data Memory is decremented by 1.

Operation $[m] \leftarrow [m] - 1$

Affected flag(s) Z

LDECA [m] Decrement Data Memory with result in ACC

Description Data in the specified Data Memory is decremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation $ACC \leftarrow [m] - 1$

Affected flag(s) Z

LINC [m] Increment Data Memory

Description Data in the specified Data Memory is incremented by 1.

Operation $[m] \leftarrow [m] + 1$

Affected flag(s) Z

LINCA [m] Increment Data Memory with result in ACC

Description Data in the specified Data Memory is incremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation $ACC \leftarrow [m] + 1$

Affected flag(s) Z

LMOV A,[m] Move Data Memory to ACC

Description The contents of the specified Data Memory are copied to the Accumulator.

Operation ACC ← [m]

Affected flag(s) None

LMOV [m],A Move ACC to Data Memory

Description The contents of the Accumulator are copied to the specified Data Memory.

Operation [m] ← ACC

Affected flag(s) None

LOR A,[m] Logical OR Data Memory to ACC

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

Operation ACC ← ACC ″OR″ [m]

Affected flag(s) Z

LORM A,[m] Logical OR ACC to Data Memory

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

Operation [m] ← ACC ″OR″ [m]

Affected flag(s) Z

LRL [m] Rotate Data Memory left

Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0.

Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← [m].7

Affected flag(s) None

LRLA [m] Rotate Data Memory left with result in ACC

Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← [m].7

Affected flag(s) None

LRLC [m] Rotate Data Memory left through Carry

Description The contents of the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into bit 0.

Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← C
C ← [m].7

Affected flag(s) C

LRLCA [m] Rotate Data Memory left through Carry with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into the bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← C
C ← [m].7

Affected flag(s) C

LRR [m] Rotate Data Memory right

Description The contents of the specified Data Memory are rotated right by 1 bit with bit 0 rotated into bit 7.

Operation [m].i ← [m].(i+1); (i=0~6)
[m].7 ← [m].0

Affected flag(s) None

LRRA [m] Rotate Data Memory right with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated right by 1 bit with bit 0 rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.i ← [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 two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation [m] ← [m] − 1
Skip if [m]=0

Affected flag(s) None

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

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

Operation ACC ← [m] − 1
Skip if ACC=0

Affected flag(s) None

LSET [m] Set Data Memory

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

Operation [m] ← FFH

Affected flag(s) None

LSET [m].i Set bit of Data Memory

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

Operation [m].i ← 1

Affected flag(s) None

LSIZ [m] Skip if increment Data Memory is 0

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

Operation [m] ← [m] + 1
Skip if [m]=0

Affected flag(s) None

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

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

Operation ACC ← [m] + 1
Skip if ACC=0

Affected flag(s) None

LSNZ [m].i Skip if Data Memory is not 0

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

Operation Skip if [m].i ≠ 0

Affected flag(s) None

LSNZ [m] Skip if Data Memory is not 0

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

Operation Skip if [m] ≠ 0

Affected flag(s) None

LSUB A,[m] Subtract Data Memory from ACC

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

Operation ACC ← ACC − [m]

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

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

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

Operation [m] ← ACC − [m]

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

LSWAP [m] Swap nibbles of Data Memory

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

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s) None

LSWAPA [m] Swap nibbles of Data Memory with result in ACC

Description The low-order and high-order nibbles of the specified Data Memory are interchanged. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC.3~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0

Affected flag(s) None

LSZ [m] Skip if Data Memory is 0

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

Operation Skip if [m]=0

Affected flag(s) None

LSZA [m] Skip if Data Memory is 0 with data movement to ACC

Description The contents of the specified Data Memory are copied to the Accumulator. If the value is zero, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation ACC ← [m]
Skip if [m]=0

Affected flag(s) None

LSZ [m].i Skip if bit i of Data Memory is 0

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

Operation Skip if [m].i=0

Affected flag(s) None

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

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

LITABRD [m] Increment table pointer low byte first and read table to TBLH and Data Memory

Description Increment table pointer low byte, TBLP, first and then the program code addressed by the table pointer (TBHP and TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

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

Affected flag(s) None

LITABRDL [m] Increment table pointer low byte first and read table (last page) to TBLH and Data Memory

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

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

Affected flag(s) None

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

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

Operation ACC ← ACC ″XOR″ [m]

Affected flag(s) Z

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

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

Operation [m] ← ACC ″XOR″ [m]

Affected flag(s) Z

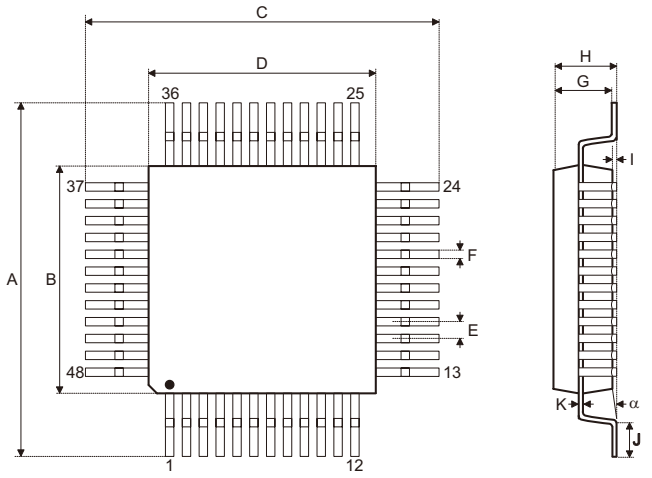
Package Information

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

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

- Further Package Information (include Outline Dimensions, Product Tape and Reel Specifications)
- Packing Meterials Information
- Carton information
- PB FREE Products
- Green Packages Products

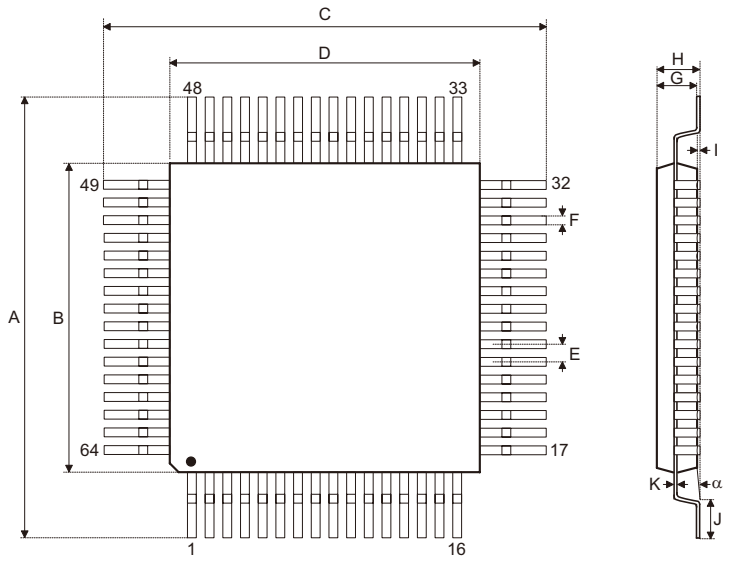
48-pin LQFP (7mm×7mm) Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	0.350	—	0.358
B	0.272	—	0.280
C	0.350	—	0.358
D	0.272	—	0.280
E	—	0.020	—
F	—	0.008	—
G	0.053	—	0.057
H	—	—	0.063
I	—	0.004	—
J	0.018	—	0.030
K	0.004	—	0.008
α	0°	—	7°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	8.90	—	9.10
B	6.90	—	7.10
C	8.90	—	9.10
D	6.90	—	7.10
E	—	0.50	—
F	—	0.20	—
G	1.35	—	1.45
H	—	—	1.60
I	—	0.10	—
J	0.45	—	0.75
K	0.10	—	0.20
α	0°	—	7°

64-pin LQFP (7mm×7mm) Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	0.350	—	0.358
B	0.272	—	0.280
C	0.350	—	0.358
D	0.272	—	0.280
E	—	0.016	—
F	0.005	—	0.009
G	0.053	—	0.057
H	—	—	0.063
I	0.002	—	0.006
J	0.018	—	0.030
K	0.004	—	0.008
α	0°	—	7°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	8.90	—	9.10
B	6.90	—	7.10
C	8.90	—	9.10
D	6.90	—	7.10
E	—	0.40	—
F	0.13	—	0.23
G	1.35	—	1.45
H	—	—	1.60
I	0.05	—	0.15
J	0.45	—	0.75
K	0.09	—	0.20
α	0°	—	7°

Holtek Semiconductor Inc. (Headquarters)
No.3, Creation Rd. II, Science Park, Hsinchu, Taiwan
Tel: 886-3-563-1999
Fax: 886-3-563-1189
http://www.holtek.com.tw

Holtek Semiconductor Inc. (Taipei Sales Office)
4F-2, No. 3-2, YuanQu St., Nankang Software Park, Taipei 115, Taiwan
Tel: 886-2-2655-7070
Fax: 886-2-2655-7373
Fax: 886-2-2655-7383 (International sales hotline)

Holtek Semiconductor (China) Inc. (Dongguan Sales Office)
Building No.10, Xinzhu Court, (No.1 Headquarters), 4 Cuizhu Road, Songshan Lake, Dongguan, China 523808
Tel: 86-769-2626-1300
Fax: 86-769-2626-1311, 86-769-2626-1322

Holtek Semiconductor (USA), Inc. (North America Sales Office)
46729 Fremont Blvd., Fremont, CA 94538, USA
Tel: 1-510-252-9880
Fax: 1-510-252-9885
http://www.holtek.com

Copyright© 2013 by HOLTEK SEMICONDUCTOR INC.
The information appearing in this Data Sheet is believed to be accurate at the time of publication. However, Holtek assumes no responsibility arising from the use of the specifications described. The applications mentioned herein are used solely for the purpose of illustration and Holtek makes no warranty or representation that such applications will be suitable without further modification, nor recommends the use of its products for application that may present a risk to human life due to malfunction or otherwise. Holtek's products are not authorized for use as critical components in life support devices or systems. Holtek reserves the right to alter its products without prior notification. For the most up-to-date information, please visit our web site at http://www.holtek.com.tw.