A/D Flash MCU with Dual Operational Amplifiers

HT66F4530/HT66F4540/HT66F4550

Revision: V1.00 Date: December 21, 2016

www.holtek.com

Table of Contents

Features 7
CPU Features 7
Peripheral Features 7
General Description 8
Selection Table 8
Block Diagram 9
Pin Assignment 9
Pin Description 11
Absolute Maximum Ratings 20
D.C. Characteristics 20
Operating Voltage Characteristics 20
Standby Current Characteristics 21
Operating Current Characteristics 22
A.C. Characteristics 23
High Speed Internal Oscillator – HIRC – Frequency Accuracy 23
Low Speed Internal Oscillator Characteristics – LIRC 24
Operating Frequency Characteristic Curves 24
System Start Up Time Characteristics 24
Input/Output Characteristics 25
Memory Characteristics 26
LVD & LVR Electrical Characteristics 26
A/D Converter Electrical Characteristics 27
Internal Reference Voltage Characteristics 27
Operational Amplifier Electrical Characteristics 28
Comparator Electrical Characteristics 29
8-Bit D/A Converter Electrical Characteristics 29
16-Bit Voice D/A Converter Electrical Characteristics 29
LCD Electrical Characteristics 30
Power-on Reset Characteristics 30
System Architecture 31
Clocking and Pipelining 31
Program Counter 32
Stack 32
Arithmetic and Logic Unit – ALU 33
Flash Program Memory 34
Structure 34
Special Vectors 34
Look-up Table 35

Table Program Example 35
In Circuit Programming – ICP 36
On Chip Debug Support – OCDS 37

Data Memory 38
Structure 38
Data Memory Addressing 39
General Purpose Data Memory 39
Special Purpose Data Memory 39

Special Function Register Description 43
Indirect Addressing Registers – IAR0, IAR1, IAR2 43
Memory Pointers – MP0, MP1L, MP1H, MP2L, MP2H 43
Accumulator – ACC 45
Program Counter Low Register – PCL 45
Look-up Table Registers – TBLP, TBHP, TBLH 45
Status Register – STATUS 45

EEPROM Data Memory 47
EEPROM Data Memory Structure 47
EEPROM Registers 47
Reading Data from the EEPROM 49
Writing Data to the EEPROM 49
Write Protection 49
EEPROM Interrupt 49
Programming Considerations 50

Oscillators 51
Oscillator Overview 51
System Clock Configurations 51
External Crystal/Ceramic Oscillator – HXT 52
Internal High Speed RC Oscillator – HIRC 53
External 32.768kHz Crystal Oscillator – LXT 53
Internal 32kHz Oscillator – LIRC 54

Operating Modes and System Clocks 54
System Clocks 54
System Operation Modes 55
Control Registers 56
Operating Mode Switching 59
Standby Current Considerations 63
Wake-up 63

Watchdog Timer 64
Watchdog Timer Clock Source 64
Watchdog Timer Control Register 64
Watchdog Timer Operation 65

Reset and Initialisation 66
Reset Functions 66
Reset Initial Conditions 70

Input/Output Ports 75
Pull-high Resistors 76
Port A Wake-up 76
I/O Port Control Registers 77
I/O Port Sink Current Control 77
Pin-shared Functions 78
I/O Pin Structures 87
Programming Considerations 87
Timer Modules – TM 88
Introduction 88
TM Operation 88
TM Clock Source 89
TM Interrupts 89
TM External Pins 89
TM Input/Output Pin Selection 90
Programming Considerations 91
Standard Type TM – STM 92
Standard TM Operation 92
Standard Type TM Register Description 93
Standard Type TM Operation Modes 97
Periodic Type TM – PTM 107
Periodic TM Operation 107
Periodic Type TM Register Description 107
Periodic Type TM Operating Modes 112
Analog to Digital Converter 121
A/D Converter Overview 121
A/D Converter Register Description 122
A/D Converter Operation 125
A/D Converter Reference Voltage 126
A/D Converter Input Signals 126
Conversion Rate and Timing Diagram 127
Summary of A/D Conversion Steps 128
Programming Considerations 129
A/D Conversion Function 129
A/D Conversion Programming Examples 130
Serial Interface Module – SIM 132
SPI Interface 132
I^2C Interface 140

UART Interface – HT66F4540/HT66F4550 150
UART External Pins 151
UART Data Transfer Scheme 151
UART Status and Control Registers 151
Baud Rate Generator 157
UART Setup and Control 157
UART Transmitter 159
UART Receiver 160
Managing Receiver Errors 162
UART Interrupt Structure 163
UART Power Down and Wake-up 164
Operational Amplifiers and Comparators 165
Analog Function Operation 165
Control Registers 166
Offset Calibration Procedure 174
SCOM Function for LCD – HT66F4540/HT66F4550 175
LCD Operation 175
LCD Bias Current Control 175
Voice D/A Converter – HT66F4550 176
Voice D/A Converter Control Registers 176
Low Voltage Detector – LVD 177
LVD Register 177
LVD Operation 178
Interrupts 179
Interrupt Registers 179
Interrupt Operation 186
External Interrupts 190
SD Comparator Interrupts 190
A/D Converter Interrupt 190
Time Base Interrupts 191
SIM Interrupt 192
LVD Interrupt 192
EEPROM Interrupt 193
UART Interrupt 193
Multi-function Interrupts 193
TM Interrupts 193
Interrupt Wake-up Function 194
Programming Considerations 194
Configuration Option 195
Application Circuits 196

Instruction Set 197
Introduction 197
Instruction Timing 197
Moving and Transferring Data 197
Arithmetic Operations 197
Logical and Rotate Operation 198
Branches and Control Transfer 198
Bit Operations 198
Table Read Operations 198
Other Operations 198
Instruction Set Summary 199
Table Conventions 199
Extended Instruction Set 201
Instruction Definition 203
Extended Instruction Definition 212
Package Information 219
16-pin NSOP (150mil) Outline Dimensions 220
20-pin SSOP (150mil) Outline Dimensions 221
24-pin SSOP (150mil) Outline Dimensions 222
28-pin SSOP (150mil) Outline Dimensions 223

Features

CPU Features

- Operating Voltage:
  - f_{SYS}=4MHz: 2.2V~5.5V
  - f_{SYS}=8MHz: 2.2V~5.5V
  - f_{SYS}=12MHz: 2.7V~5.5V
- Up to 0.33μs instruction cycle with 12MHz system clock at V_{DD}=5V
- Power down and wake-up functions to reduce power consumption
- Oscillator types:
  - External High Speed Crystal – HXT
  - Internal High Speed RC – HIRC
  - External Low Speed 32.768kHz Crystal – LXT
  - Internal Low Speed 32kHz RC – LIRC
- Multi-mode operation: FAST, SLOW, IDLE and SLEEP
- Fully integrated internal 2MHz, 4MHz and 8MHz oscillator requires no external components
- All instructions executed in 1~3 instruction cycles
- Table read instructions
- 115 powerful instructions
- Up to 8-level subroutine nesting
- Bit manipulation instruction

Peripheral Features

- Program Memory: 2K×16~8K×16
- Data Memory: 128×8~384×8
- True EEPROM Memory: 32×8~64×8
- Watchdog Timer function
- Up to 26 bidirectional I/O lines
- Two external interrupt lines shared with I/O pins
- Two operational amplifiers, two comparators and three 8-bit D/A converters
- Software controlled 4-SCOM lines LCD driver with 1/2 bias – HT66F4540/HT66F4550
- Multiple Timer Modules for time measure, input capture, compare match output, PWM output or single pulse output function
- Dual Time-Base functions for generation of fixed time interrupt signals
- Serial Interface Module - SPI or I^2C
- UART Interface for fully duplex asynchronous communication – HT66F4540/HT66F4550
- Up to 8 channel 12-bit resolution A/D converter
- High performance 16-bit Voice D/A converter – HT66F4550
- Low voltage reset function
- Low voltage detect function
- Wide range of available package types

General Description

This series of devices are Flash Memory A/D type 8-bit high performance RISC architecture microcontrollers. Offering users the convenience of Flash Memory multi-programming features, these devices also include a wide range of flexible functions and features. Other memory includes an area of RAM Data Memory as well as an area of true EEPROM memory for storage of non-volatile data such as serial numbers, calibration data etc.

Analog features include a multi-channel 12-bit A/D converter, a 16-bit Voice D/A converter, dual comparators, dual operational amplifiers and three 8-bit multi-channel D/A converter. With regard to internal timers, these devices include multiple and extremely flexible Timer Modules providing functions for timing, pulse generation and PWM generation operations. Communication with the outside world is catered for by the inclusion of fully integrated SPI, I^2C and UART interfaces functions, 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 various internal and external high and low oscillators 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, a fully integrated LCD driver and Time-Base functions along with many other features ensure that these devices will find excellent use in applications such as smoke detectors, electronic metering, environmental monitoring, handheld instruments, household appliances, electronically controlled tools, motor driving in addition to many others.

Selection Table

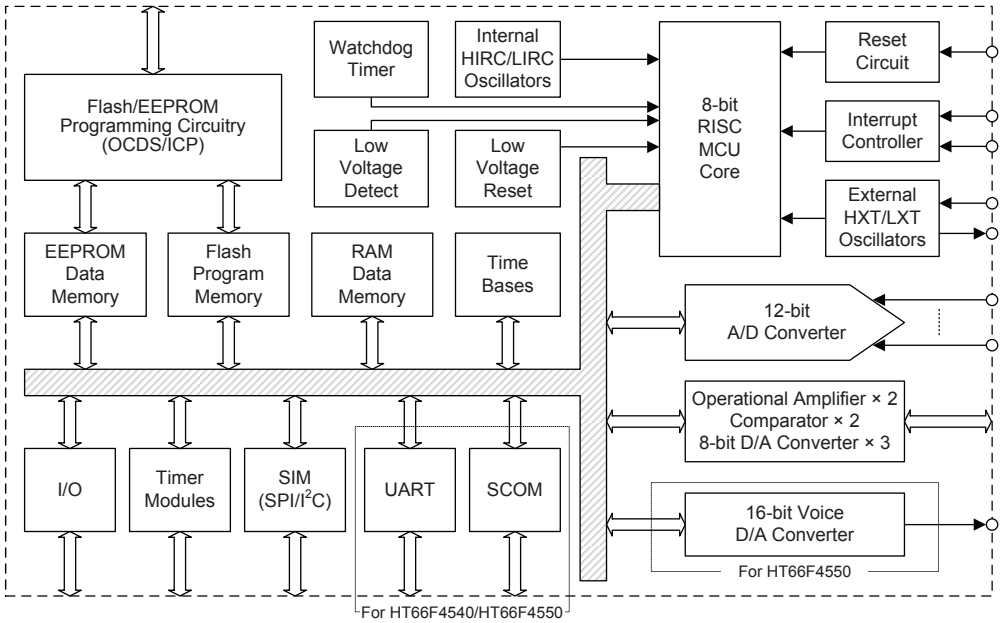
Most features are common to all devices. The main features distinguishing them are Memory capacity I/O count, A/D converter inputs, Timer Module features, SCOM, UART, Voice D/A converter, stack and package types. The following table summarises the main features of each device.

Part No.	Program Memory	Data Memory	Data EEPROM	I/O	External Interrupt	A/D	Timer Module	Time Base
HT66F4530	2K×16	128×8	32×8	18	2	12-bit×5	10-bit STM×1 10-bit PTM×1	2
HT66F4540	4K×16	256×8	64×8	26	2	12-bit×8	10-bit STM×1 10-bit PTM×2	2
HT66F4550	8K×16	384×8	64×8	26	2	12-bit×8	10-bit STM×2 10-bit PTM×2	2

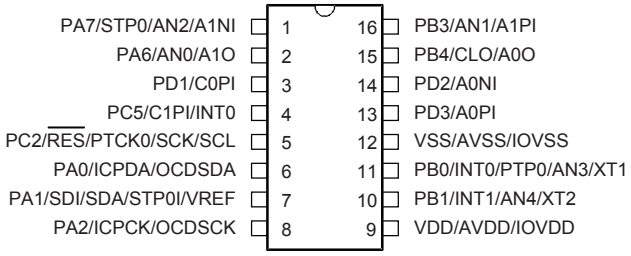
Part No.	SCOM	SIM	UART	Comp.	Op. Amp.	D/A	Voice D/A	Stack	Package
HT66F4530	×	√	×	2	2	8-bit×3	×	6	16NSOP 20SSOP
HT66F4540	4	√	√	2	2	8-bit×3	×	8	24/28SSOP
HT66F4550	4	√	√	2	2	8-bit×3	√	8	24/28SSOP

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

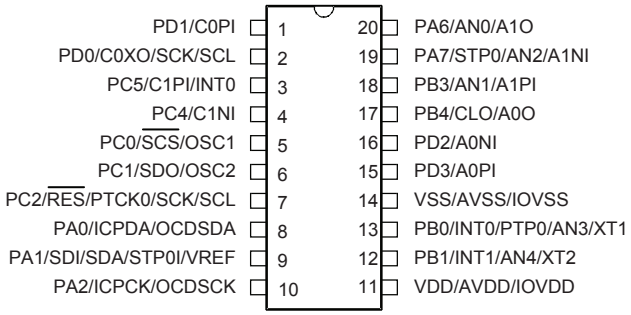
Block Diagram



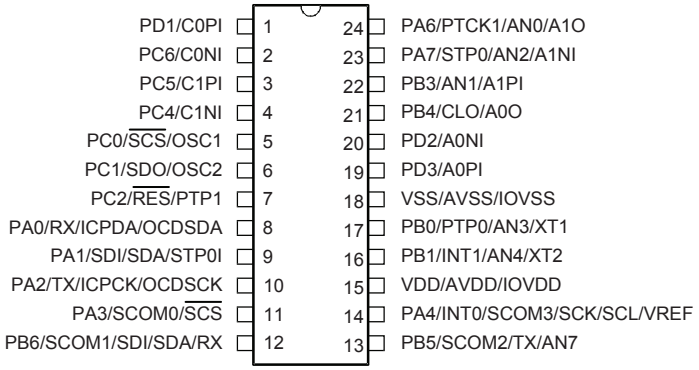
Pin Assignment



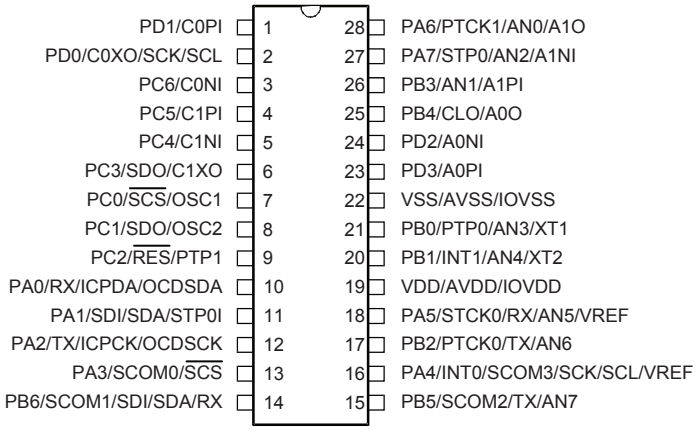
HT66F4530/HT66V4530
16 NSOP-A



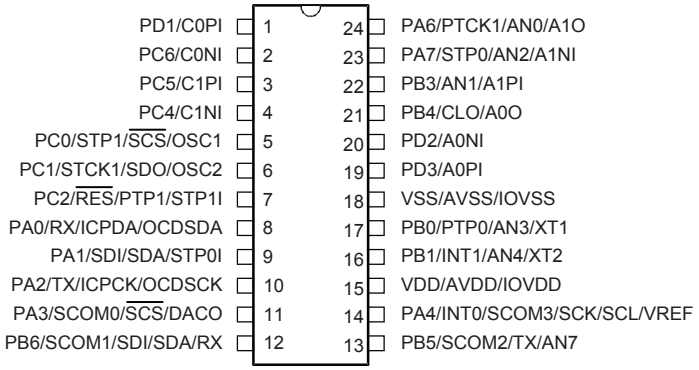
HT66F4530/HT66V4530
20 SSOP-A



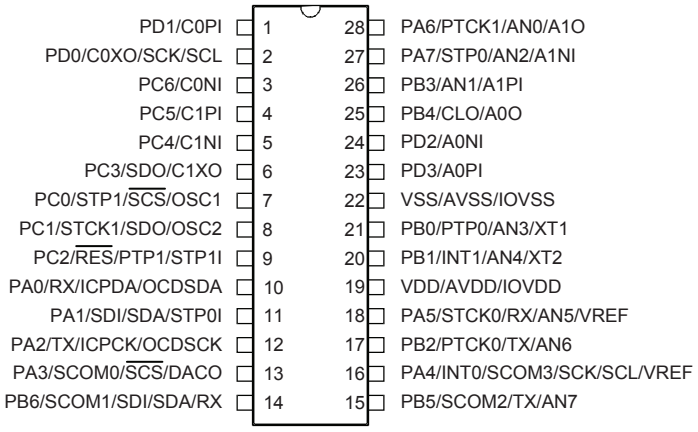
HT66F4540/HT66V4540
24 SSOP-A



HT66F4540/HT66V4540
28 SSOP-A



HT66F4550/HT66V4550
24 SSOP-A



HT66F4550/HT66V4550
28 SSOP-A

Note: 1. If the pin-shared pin functions have multiple outputs simultaneously, the desired pin-shared function is determined by the corresponding software control bits.
2. The OCDSDA and OCDSCK pins are used for the OCDS function and thus only exist on the HT66V45x0 device which is the OCDS EV chip for the HT66F45x0 device.

Pin Description

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

As the Pin Description Summary table applies to the package type with the most pins, not all of the above listed pins may be present on package types with smaller numbers of pins.

HT66F4530

Pin Name	Function	OPT	I/T	O/T	Description
PA0/ICPDA/ OCDSDA	PA0	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	ICPDA	—	ST	CMOS	ICP data/address
	OCDSDA	—	ST	CMOS	OCDS data/address, for EV chip only
PA1/SDI/SDA/ STP0I/VREF	PA1	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	SDI	PAS0 IFS	ST	—	SPI serial data input
	SDA	PAS0 IFS	ST	NMOS	I^2C data line
	STP0I	PAS0	ST	—	STM0 capture Input
	VREF	PAS0	AN	—	A/D Converter reference voltage input
PA2/ICPCK/ OCDSCK	PA2	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	ICPCK	—	ST	—	ICP clock pin
	OCDSCK	—	ST	—	OCDS clock pin, for EV chip only

Pin Name	Function	OPT	I/T	O/T	Description
PA6/AN0/A1O	PA6	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	AN0	PAS1	AN	—	A/D Converter input channel 0
	A1O	PAS1	—	AN	Operational Amplifier 1 output
PA7/STP0/AN2/ A1NI	PA7	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	STP0	PAS1	—	CMOS	STM0 output
	AN2	PAS1	AN	—	A/D Converter input channel 2
	A1NI	PAS1	AN	—	Operational Amplifier 1 inverting input pin
PB0/INT0/PTP0/ AN3/XT1	PB0	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	INT0	PBS0 IFS	ST	—	External Interrupt 0
	PTP0	PBS0	—	CMOS	PTM0 output
	AN3	PBS0	AN	—	A/D Converter input channel 3
	XT1	PBS0	LXT	—	LXT pin
PB1/INT1/AN4/XT2	PB1	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	INT1	PBS0	ST	—	External Interrupt 1
	AN4	PBS0	AN	—	A/D Converter input channel 4
	XT2	PBS0	—	LXT	LXT pin
PB3/AN1/A1PI	PB3	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	AN1	PBS0	AN	—	A/D Converter input channel 1
	A1PI	PBS0	AN	—	Operational Amplifier 1 non-inverting input pin
PB4/CLO/A0O	PB4	PBPU PBS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	CLO	PBS1	—	CMOS	System clock output
	A0O	PBS1	—	AN	Operational Amplifier 0 output
PC0/$\overline{\text{SCS}}$/OSC1	PC0	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	$\overline{\text{SCS}}$	PCS0 IFS	ST	CMOS	SPI slave select pin
	OSC1	PCS0	HXT	—	HXT pin
PC1/SDO/OSC2	PC1	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	SDO	PCS0	—	CMOS	SPI serial data output
	OSC2	PCS0	—	HXT	HXT pin
PC2/$\overline{\text{RES}}$/PTCK0/ SCK/SCL	PC2	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	$\overline{\text{RES}}$	PCS0	ST	—	External reset input
	PTCK0	PCS0	ST	—	PTM0 clock input
	SCK	PCS0 IFS	ST	CMOS	SPI serial clock
	SCL	PCS0 IFS	ST	NMOS	I^2C clock line
PC4/C1NI	PC4	PCPU PCS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	C1NI	PCS1	AN	—	Comparator 1 inverting input pin

Pin Name	Function	OPT	I/T	O/T	Description
PC5/C1PI/INT0	PC5	PCPU PCS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	C1PI	PCS1	AN	—	Comparator 1 non-inverting input pin
	INT0	PCS1 IFS	ST	—	External Interrupt 0
PD0/C0XO/SCK/ SCL	PD0	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	C0XO	PDS0	—	AN	Comparator 0 output
	SCK	PDS0 IFS	ST	CMOS	SPI serial clock
	SCL	PDS0 IFS	ST	NMOS	I^2C clock line
PD1/C0PI	PD1	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	C0PI	PDS0	AN	—	Comparator 0 non-inverting input pin
PD2/A0NI	PD2	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	A0NI	PDS0	AN	—	Operational Amplifier 0 inverting input pin
PD3/A0PI	PD3	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	A0PI	PDS0	AN	—	Operational Amplifier 0 non-inverting input pin
VDD/AVDD/ IOVDD*	VDD	—	PWR	—	Power Supply
	AVDD	—	PWR	—	A/D Converter Power Supply
	IOVDD	—	PWR	—	I/O Port Power Supply
VSS/AVSS/ IOVSS**	VSS	—	PWR	—	Ground
	AVSS	—	PWR	—	A/D Converter Ground
	IOVSS	—	PWR	—	I/O Port Ground

HT66F4540

Pin Name	Function	OPT	I/T	O/T	Description
PA0/RX/ICPDA/ OCDSDA	PA0	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	RX	PAS0 IFS	ST	—	UART receiver pin
	ICPDA	—	ST	CMOS	ICP data/address
	OCDSDA	—	ST	CMOS	OCDS data/address, for EV chip only
PA1/SDI/SDA/ STP0I	PA1	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	SDI	PAS0 IFS	ST	—	SPI serial data input
	SDA	PAS0 IFS	ST	NMOS	I^2C data line
	STP0I	PAS0	ST	—	STM0 capture Input
PA2/TX/ICPCK/ OCDSCK	PA2	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	TX	PAS0	—	CMOS	UART transmitter pin
	ICPCK	—	ST	—	ICP clock pin
	OCDSCK	—	ST	—	OCDS clock pin, for EV chip only

Pin Name	Function	OPT	I/T	O/T	Description
PA3/SCOM0/$\overline{SCS}$	PA3	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	SCOM0	PAS0	—	AN	Software controlled 1/2 bias LCD COM
	$\overline{SCS}$	PAS0 IFS	ST	CMOS	SPI slave select pin
PA4/INT0/SCOM3/ SCK/SCL/VREF	PA4	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	INT0	PAS1	ST	—	External Interrupt 0
	SCOM3	PAS1	—	AN	Software controlled 1/2 bias LCD COM
	SCK	PAS1 IFS	ST	CMOS	SPI serial clock
	SCL	PAS1 IFS	ST	NMOS	I^2C clock line
	VREF	PAS1	AN	—	A/D Converter reference voltage input
PA5/STCK0/ RX/AN5/VREF	PA5	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	STCK0	PAS1	ST	—	STM0 clock input
	RX	PAS1 IFS	ST	—	UART receiver pin
	AN5	PAS1	AN	—	A/D Converter input channel 5
	VREF	PAS1	AN	—	A/D Converter reference voltage input
PA6/PTCK1/AN0/ A1O	PA6	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	PTCK1	PAS1	ST	—	PTM1 clock input
	AN0	PAS1	AN	—	A/D Converter input channel 0
	A1O	PAS1	—	AN	Operational Amplifier 1 output
PA7/STP0/AN2/ A1NI	PA7	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	STP0	PAS1	—	CMOS	STM0 output
	AN2	PAS1	AN	—	A/D Converter input channel 2
	A1NI	PAS1	AN	—	Operational Amplifier 1 inverting input pin
PB0/PTP0/AN3/ XT1	PB0	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	PTP0	PBS0	—	CMOS	PTM0 output
	AN3	PBS0	AN	—	A/D Converter input channel 3
	XT1	PBS0	LXT	—	LXT pin
PB1/INT1/AN4/XT2	PB1	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	INT1	PBS0	ST	—	External Interrupt 1
	AN4	PBS0	AN	—	A/D Converter input channel 4
	XT2	PBS0	—	LXT	LXT pin
PB2/PTCK0/TX/ AN6	PB2	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	PTCK0	PBS0	ST	—	PTM0 clock input
	TX	PBS0	—	CMOS	UART transmitter pin
	AN6	PBS0	AN	—	A/D Converter input channel 6

Pin Name	Function	OPT	I/T	O/T	Description
PB3/AN1/A1PI	PB3	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	AN1	PBS0	AN	—	A/D Converter input channel 1
	A1PI	PBS0	AN	—	Operational Amplifier 1 non-inverting input pin
PB4/CLO/A0O	PB4	PBPU PBS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	CLO	PBS1	—	CMOS	System clock output
	A0O	PBS1	—	AN	Operational Amplifier 0 output
PB5/SCOM2/TX/ AN7	PB5	PBPU PBS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	SCOM2	PBS1	—	AN	Software controlled 1/2 bias LCD COM
	TX	PBS1	—	CMOS	UART transmitter pin
	AN7	PBS1	AN	—	A/D Converter input channel 7
PB6/SCOM1/ SDI/SDA/RX	PB6	PBPU PBS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	SCOM1	PBS1	—	AN	Software controlled 1/2 bias LCD COM
	SDI	PBS1 IFS	ST	—	SPI serial data input
	SDA	PBS1 IFS	ST	NMOS	I^2C data line
	RX	PBS1 IFS	ST	—	UART receiver pin
PC0/$\overline{\text{SCS}}$/OSC1	PC0	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	$\overline{\text{SCS}}$	PCS0 IFS	ST	CMOS	SPI slave select pin
	OSC1	PCS0	HXT	—	HXT pin
PC1/SDO/OSC2	PC1	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	SDO	PCS0	—	CMOS	SPI serial data output
	OSC2	PCS0	—	HXT	HXT pin
PC2/$\overline{\text{RES}}$/PTP1	PC2	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	$\overline{\text{RES}}$	PCS0	ST	—	External reset input
	PTP1	PCS0	—	CMOS	PTM1 output
PC3/SDO/C1XO	PC3	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	SDO	PCS0	—	CMOS	SPI serial data output
	C1XO	PCS0	—	AN	Comparator 1 output
PC4/C1NI	PC4	PCPU PCS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	C1NI	PCS1	AN	—	Comparator 1 inverting input pin
PC5/C1PI	PC5	PCPU PCS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	C1PI	PCS1	AN	—	Comparator 1 non-inverting input pin
PC6/C0NI	PC6	PCPU PCS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	C0NI	PCS1	AN	—	Comparator 0 inverting input pin

Pin Name	Function	OPT	I/T	O/T	Description
PD0/C0XO/SCK/SCL	PD0	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	C0XO	PDS0	—	AN	Comparator 0 output
	SCK	PDS0 IFS	ST	CMOS	SPI serial clock
	SCL	PDS0 IFS	ST	NMOS	I^2C clock line
PD1/C0PI	PD1	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	C0PI	PDS0	AN	—	Comparator 0 non-inverting input pin
PD2/A0NI	PD2	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	A0NI	PDS0	AN	—	Operational Amplifier 0 inverting input pin
PD3/A0PI	PD3	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	A0PI	PDS0	AN	—	Operational Amplifier 0 non-inverting input pin
VDD/AVDD/IOVDD*	VDD	—	PWR	—	Power Supply
	AVDD	—	PWR	—	A/D Converter Power Supply
	IOVDD	—	PWR	—	I/O Port Power Supply
VSS/AVSS/IOVSS**	VSS	—	PWR	—	Ground
	AVSS	—	PWR	—	A/D Converter Ground
	IOVSS	—	PWR	—	I/O Port Ground

HT66F4550

Pin Name	Function	OPT	I/T	O/T	Description
PA0/RX/ICPDA/OCDSDA	PA0	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	RX	PAS0 IFS	ST	—	UART receiver pin
	ICPDA	—	ST	CMOS	ICP data/address
	OCDSDA	—	ST	CMOS	OCDS data/address, for EV chip only
PA1/SDI/SDA/STP0I	PA1	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	SDI	PAS0 IFS	ST	—	SPI serial data input
	SDA	PAS0 IFS	ST	NMOS	I^2C data line
	STP0I	PAS0	ST	—	STM0 capture Input
PA2/TX/ICPCK/OCDSCK	PA2	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	TX	PAS0	—	CMOS	UART transmitter pin
	ICPCK	—	ST	—	ICP clock pin
	OCDSCK	—	ST	—	OCDS clock pin, for EV chip only

Pin Name	Function	OPT	I/T	O/T	Description
PA3/SCOM0/$\overline{SCS}$/DACO	PA3	PAPU PAWU PAS0	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	SCOM0	PAS0	—	AN	Software controlled 1/2 bias LCD COM
	$\overline{SCS}$	PAS0 IFS	ST	CMOS	SPI slave select pin
	DACO	PAS0	—	AN	Voice D/A Converter output
PA4/INT0/SCOM3/SCK/SCL/VREF	PA4	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	INT0	PAS1	ST	—	External Interrupt 0
	SCOM3	PAS1	—	AN	Software controlled 1/2 bias LCD COM
	SCK	PAS1 IFS	ST	CMOS	SPI serial clock
	SCL	PAS1 IFS	ST	NMOS	I^2C clock line
	VREF	PAS1	AN	—	A/D Converter reference voltage input
PA5/STCK0/RX/AN5/VREF	PA5	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	STCK0	PAS1	ST	—	STM0 clock input
	RX	PAS1 IFS	ST	—	UART receiver pin
	AN5	PAS1	AN	—	A/D Converter input channel 5
	VREF	PAS1	AN	—	A/D Converter reference voltage input
PA6/PTCK1/AN0/A1O	PA6	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	PTCK1	PAS1	ST	—	PTM1 clock input
	AN0	PAS1	AN	—	A/D Converter input channel 0
	A1O	PAS1	—	AN	Operational Amplifier 1 output
PA7/STP0/AN2/A1NI	PA7	PAPU PAWU PAS1	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	STP0	PAS1	—	CMOS	STM0 output
	AN2	PAS1	AN	—	A/D Converter input channel 2
	A1NI	PAS1	AN	—	Operational Amplifier 1 inverting input pin
PB0/PTP0/AN3/XT1	PB0	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	PTP0	PBS0	—	CMOS	PTM0 output
	AN3	PBS0	AN	—	A/D Converter input channel 3
	XT1	PBS0	LXT	—	LXT pin
PB1/INT1/AN4/XT2	PB1	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	INT1	PBS0	ST	—	External Interrupt 1
	AN4	PBS0	AN	—	A/D Converter input channel 4
	XT2	PBS0	—	LXT	LXT pin
PB2/PTCK0/TX/AN6	PB2	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	PTCK0	PBS0	ST	—	PTM0 clock input
	TX	PBS0	—	CMOS	UART transmitter pin
	AN6	PBS0	AN	—	A/D Converter input channel 6

Pin Name	Function	OPT	I/T	O/T	Description
PB3/AN1/A1PI	PB3	PBPU PBS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	AN1	PBS0	AN	—	A/D Converter input channel 1
	A1PI	PBS0	AN	—	Operational Amplifier 1 non-inverting input pin
PB4/CLO/A0O	PB4	PBPU PBS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	CLO	PBS1	—	CMOS	System clock output
	A0O	PBS1	—	AN	Operational Amplifier 0 output
PB5/SCOM2/TX/ AN7	PB5	PBPU PBS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	SCOM2	PBS1	—	AN	Software controlled 1/2 bias LCD COM
	TX	PBS1	—	CMOS	UART transmitter pin
	AN7	PBS1	AN	—	A/D Converter input channel 7
PB6/SCOM1/ SDI/SDA/RX	PB6	PBPU PBS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	SCOM1	PBS1	—	AN	Software controlled 1/2 bias LCD COM
	SDI	PBS1 IFS	ST	—	SPI serial data input
	SDA	PBS1 IFS	ST	NMOS	I^2C data line
	RX	PBS1 IFS	ST	—	UART receiver pin
PC0/STP1/$\overline{SCS}$/ OSC1	PC0	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	STP1	PCS0	—	CMOS	STM1 output
	$\overline{SCS}$	PCS0 IFS	ST	CMOS	SPI slave select pin
	OSC1	PCS0	HXT	—	HXT pin
PC1/STCK1/SDO/ OSC2	PC1	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	STCK1	PCS0	ST	—	STM1 clock input
	SDO	PCS0	—	CMOS	SPI serial data output
	OSC2	PCS0	—	HXT	HXT pin
PC2/$\overline{RES}$/PTP1/ STP1I	PC2	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	$\overline{RES}$	PCS0	ST	—	External reset input
	PTP1	PCS0	—	CMOS	PTM1 output
	STP1I	PCS0	ST	—	STM1 capture input
PC3/SDO/C1XO	PC3	PCPU PCS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	SDO	PCS0	—	CMOS	SPI serial data output
	C1XO	PCS0	—	AN	Comparator 1 output
PC4/C1NI	PC4	PCPU PCS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	C1NI	PCS1	AN	—	Comparator 1 inverting input pin
PC5/C1PI	PC5	PCPU PCS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	C1PI	PCS1	AN	—	Comparator 1 non-inverting input pin
PC6/C0NI	PC6	PCPU PCS1	ST	CMOS	General purpose I/O. Register enabled pull-high.
	C0NI	PCS1	AN	—	Comparator 0 inverting input pin

Pin Name	Function	OPT	I/T	O/T	Description
PD0/C0XO/SCK/SCL	PD0	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	C0XO	PDS0	—	AN	Comparator 0 output
	SCK	PDS0 IFS	ST	CMOS	SPI serial clock
	SCL	PDS0 IFS	ST	NMOS	I^2C clock line
PD1/C0PI	PD1	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	C0PI	PDS0	AN	—	Comparator 0 non-inverting input pin
PD2/A0NI	PD2	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	A0NI	PDS0	AN	—	Operational Amplifier 0 inverting input pin
PD3/A0PI	PD3	PDPU PDS0	ST	CMOS	General purpose I/O. Register enabled pull-high.
	A0PI	PDS0	AN	—	Operational Amplifier 0 non-inverting input pin
VDD/AVDD/IOVDD*	VDD	—	PWR	—	Power Supply
	AVDD	—	PWR	—	A/D Converter Power Supply
	IOVDD	—	PWR	—	I/O Port Power Supply
VSS/AVSS/IOVSS**	VSS	—	PWR	—	Ground
	AVSS	—	PWR	—	A/D Converter Ground
	IOVSS	—	PWR	—	I/O Port Ground

Legend: I/T: Input type;
O/T: Output type;
OPT: Optional by register option;
PWR: Power;
ST: Schmitt Trigger input;
CMOS: CMOS output;
NMOS: NMOS output;
AN: Analog signal;
HXT: High frequency crystal oscillator;
LXT: Low frequency crystal oscillator;
*: The AVDD and IOVDD pins are bonded together internally with VDD.
**: The AVSS and IOVSS pins are bonded together internally with VSS.

Absolute Maximum Ratings

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

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

D.C. Characteristics

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

Operating Voltage Characteristics

Ta= -40°C~85°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage – HXT	—	$f_{SYS}=f_{HXT}$=4MHz	2.2	—	5.5	V
			$f_{SYS}=f_{HXT}$=8MHz	2.2	—	5.5	
			$f_{SYS}=f_{HXT}$=12MHz	2.7	—	5.5	
	Operating Voltage – HIRC	—	$f_{SYS}=f_{HIRC}$=2MHz	2.2	—	5.5	V
			$f_{SYS}=f_{HIRC}$=4MHz	2.2	—	5.5	
			$f_{SYS}=f_{HIRC}$=8MHz	2.2	—	5.5	
	Operating Voltage – LXT	—	$f_{SYS}=f_{LXT}$=32.768kHz	2.2	—	5.5	V
	Operating Voltage – LIRC	—	$f_{SYS}=f_{LIRC}$=32kHz	2.2	—	5.5	V

Standby Current Characteristics

Ta=25°C

Symbol	Standby Mode	Test Conditions		Min.	Typ.	Max.	Max.	Unit
		V_{DD}	Conditions				85°C	
I_{STB}	SLEEP Mode	2.2V	WDT off	—	0.2	0.6	0.7	μA
		3V		—	0.2	0.8	1.0	
		5V		—	0.5	1.0	1.2	
		2.2V	WDT on	—	1.2	2.4	2.9	
		3V		—	1.5	3.0	3.6	
		5V		—	3.0	5.0	6.0	
	IDLE0 Mode – LIRC	2.2V	f_{SUB} on	—	2.4	4.0	4.8	μA
		3V		—	3.0	5.0	6.0	
		5V		—	5.0	10	12	
	IDLE0 Mode – LXT	2.2V	f_{SUB} on	—	2.4	4.0	4.8	μA
		3V		—	3.0	5.0	6.0	
		5V		—	5.0	10	12	
	IDLE1 Mode – HIRC	2.2V	f_{SUB} on, f_{SYS}=2MHz	—	60	90	90	μA
		3V		—	90	135	135	
		5V		—	180	270	270	
		2.2V	f_{SUB} on, f_{SYS}=4MHz	—	144	200	240	μA
		3V		—	180	250	300	
		5V		—	400	600	720	
		2.2V	f_{SUB} on, f_{SYS}=8MHz	—	288	400	480	μA
		3V		—	360	500	600	
		5V		—	600	800	960	
	IDLE1 Mode – HXT	2.2V	f_{SUB} on, f_{SYS}=4MHz	—	144	200	240	μA
		3V		—	180	250	300	
		5V		—	400	600	720	
		2.2V	f_{SUB} on, f_{SYS}=8MHz	—	288	400	480	μA
		3V		—	360	500	600	
		5V		—	600	800	960	
		2.7V	f_{SUB} on, f_{SYS}=12MHz	—	432	600	720	μA
		3V		—	540	750	900	
		5V		—	800	1200	1440	

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

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

Operating Current Characteristics

Ta=25°C

Symbol	Operating Mode	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{DD}	SLOW Mode – LIRC	2.2V	f_{SYS}=32kHz	—	8	16	μA
		3V		—	10	20	
		5V		—	30	50	
	SLOW Mode – LXT	2.2V	f_{SYS}=32.768kHz	—	8	16	μA
		3V		—	10	20	
		5V		—	30	50	
	FAST Mode – HIRC	2.2V	f_{SYS}=2MHz	—	0.14	0.21	mA
		3V		—	0.20	0.30	
		5V		—	0.40	0.60	
		2.2V	f_{SYS}=4MHz	—	0.30	0.50	mA
		3V		—	0.40	0.60	
		5V		—	0.80	1.20	
		2.2V	f_{SYS}=8MHz	—	0.60	1.00	mA
		3V		—	0.80	1.20	
		5V		—	1.60	2.4	
	FAST Mode – HXT	2.2V	f_{SYS}=4MHz	—	0.40	0.60	mA
		3V		—	0.50	0.75	
		5V		—	1.00	1.50	
		2.2V	f_{SYS}=8MHz	—	0.80	1.20	mA
		3V		—	1.00	1.50	
		5V		—	2.00	3.00	
		2.7V	f_{SYS}=12MHz	—	1.20	2.20	mA
		3V		—	1.50	2.75	
		5V		—	3.00	4.50	

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

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

A.C. Characteristics

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

High Speed Internal Oscillator – HIRC – Frequency Accuracy

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

3V Trim

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Temperature				
f_{HIRC}	2MHz Writer Trimmed HIRC Frequency	3V	Ta=25°C	-1%	2	+1%	MHz
			Ta= -40°C~85°C	-2%	2	+2%	
		2.2V~5.5V	Ta=25°C	-4%	2	+4%	
			Ta= -40°C~85°C	-6%	2	+6%	
	4MHz Writer Trimmed HIRC Frequency	3V	Ta=25°C	-1%	4	+1%	MHz
			Ta= -40°C~85°C	-2%	4	+2%	
		2.2V~5.5V	Ta=25°C	-3%	4	+3%	
			Ta= -40°C~85°C	-4%	4	+4%	
	8MHz Writer Trimmed HIRC Frequency	3V	Ta=25°C	-1%	8	+1%	MHz
			Ta= -40°C~85°C	-2%	8	+2%	
		2.2V~5.5V	Ta=25°C	-2.5%	8	+2.5%	
			Ta=-40°C~85°C	-3%	8	+3%	

5V Trim

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Temperature				
f_{HIRC}	2MHz Writer Trimmed HIRC Frequency	5V	Ta=25°C	-1%	2	+1%	MHz
			Ta= -40°C~85°C	-2%	2	+2%	
		2.2V~5.5V	Ta=25°C	-7%	2	+7%	
			Ta= -40°C~85°C	-8%	2	+8%	
	4MHz Writer Trimmed HIRC Frequency	5V	Ta=25°C	-1%	4	+1%	MHz
			Ta= -40°C~85°C	-2%	4	+2%	
		2.2V~5.5V	Ta=25°C	-5%	4	+5%	
			Ta= -40°C~85°C	-6%	4	+6%	
	8MHz Writer Trimmed HIRC Frequency	5V	Ta=25°C	-1%	8	+1%	MHz
			Ta= -40°C~85°C	-2%	8	+2%	
		2.2V~5.5V	Ta=25°C	-2.5%	8	+2.5%	
			Ta=-40°C~85°C	-4%	8	+4%	

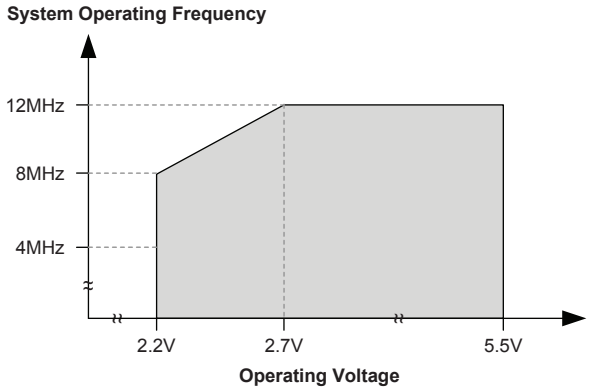
Notes: 1. The parameter values for 3V/5V trim are provided as these are the two selectable fixed voltages at which the HIRC frequency is trimmed by the writer.
2. The minimum and maximum tolerance values provided in the table are only for the frequency at which the writer trims the HIRC oscillator. After trimming at this chosen specific frequency any change in HIRC oscillator frequency using the oscillator register control bits by the application program will give a frequency tolerance to within ±20%.
3. It is recommended that the trim voltage is fixed at 3V for application voltage ranges from 2.2V to 5.5V, as the HIRC frequency tolerance is lower than that trimmed at 5V.

Low Speed Internal Oscillator Characteristics – LIRC

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Temp.				
f_{LIRC}	LIRC Frequency	2.2V~5.5V	Ta=25°C	-5%	32	+5%	kHz
			Ta= -40°C~85°C	-10%	32	+10%	
t_{START}	LIRC Start Up Time	3V	Ta= -40°C~85°C	—	—	500	μs
		5V		—	—	500	
Duty Cycle	Duty Cycle	—	Ta= -40°C~85°C	10	—	90	%

Note: Trim frequency at V_{DD}=3V, and only verify these parameters at V_{DD}=3V.

Operating Frequency Characteristic Curves



System Start Up Time Characteristics

Ta= -40°C~85°C

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
t_{SST}	System Start-up Time Wake-up from condition where f_{SYS} is off	$f_{SYS}=f_H \sim f_H/64$, $f_H=f_{HXT}$	—	128	—	t_{HXT}
		$f_{SYS}=f_H \sim f_H/64$, $f_H=f_{HIRC}$	—	16	—	t_{HIRC}
		$f_{SYS}=f_{SUB}=f_{LXT}$	—	1024	—	t_{LXT}
		$f_{SYS}=f_{SUB}=f_{LIRC}$	—	2	—	t_{LIRC}
	System Start-up Time Wake-up from condition where f_{SYS} is on	$f_{SYS}=f_H \sim f_H/64$, $f_H=f_{HXT}$ or f_{HIRC}	—	2	—	t_H
		$f_{SYS}=f_{SUB}=f_{LXT}$ or f_{LIRC}	—	2	—	t_{SUB}
	System Speed Switch Time FAST to SLOW Mode or SLOW to FAST Mode	f_{HXT} switches from off → on	—	1024	—	t_{HXT}
		f_{HIRC} switches from off → on	—	16	—	t_{HIRC}
		f_{LXT} switches from off → on	—	1024	—	t_{LXT}
t_{RSTD}	System Reset Delay Time Reset source from Power-on reset or LVR hardware reset	RR_{POR}=5V/ms	42	48	54	ms
	System Reset Delay Time LVRC/WDTC/RSTC software reset	—				
	System Reset Delay Time Reset source from WDT overflow or RES pin reset	—	14	16	18	ms
t_{SRESET}	Minimum Software Reset Width to Reset	—	45	90	123	μs

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

- The time units, shown by the symbols t_{HXT}, t_{HIRC} etc. are the inverse of the corresponding frequency values as provided in the frequency tables. For example $t_{HIRC}=1/f_{HIRC}$, $t_{SYS}=1/f_{SYS}$ etc.
- If the LIRC is used as the system clock and if it is off when in the SLEEP Mode, then an additional LIRC start up time, t_{START}, as provided in the LIRC frequency table, must be added to the t_{SST} time in the table above.
- The System Speed Switch Time is effectively the time taken for the newly activated oscillator to start up.

Input/Output Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{IL}	Input Low Voltage for I/O Ports except $\overline{RES}$ Pin	5V	—	0	—	1.5	V
		—		0	—	$0.2V_{DD}$	
	Input Low Voltage for $\overline{RES}$ Pin	—	—	0	—	$0.4V_{DD}$	V
V_{IH}	Input High Voltage for I/O Ports except $\overline{RES}$ Pin	5V	—	3.5	—	5.0	V
		—		$0.8V_{DD}$	—	V_{DD}	
	Input High Voltage for $\overline{RES}$ Pin	—	—	$0.9V_{DD}$	—	V_{DD}	V
I_{OL}	Sink Current for I/O Ports except PB0/PB1 Pin	3V	$V_{OL}=0.1V_{DD}$	16	32	—	mA
		5V		32	65	—	
	Sink Current for PB0/PB1 Pin	3V	$V_{OL}=0.3V_{DD}$, SKLEDC[m+1, m]=00B (m=0 or 2 or 4 or 6)	37.5	75	—	mA
		5V		75	150	—	
		3V	$V_{OL}=0.3V_{DD}$, SKLEDC[m+1, m]=01B (m=0 or 2 or 4 or 6)	50	100	—	mA
		5V		100	200	—	
		3V	$V_{OL}=0.3V_{DD}$, SKLEDC[m+1, m]=10B (m=0 or 2 or 4 or 6)	62.5	125	—	mA
		5V		125	250	—	
		3V	$V_{OL}=0.3V_{DD}$, SKLEDC[m+1, m]=11B (m=0 or 2 or 4 or 6)	75	150	—	mA
		5V		150	300	—	
I_{OH}	Port Source Current for I/O Ports	3V	$V_{OH}=0.9V_{DD}$	-4	-8	—	mA
		5V		-8	-16	—	
R_{PH}	Pull-High Resistance for I/O Ports (Note)	3V	—	20	60	100	kΩ
		5V		10	30	50	
I_{LEAK}	Input Leakage Current	5V	$V_{IN}=V_{DD}$ or $V_{IN}=V_{SS}$	—	—	±1	μA
t_{TPI}	TM Capture Input Pin Minimum Pulse Width	—	—	0.3	—	—	μs
t_{TCK}	TM Clock Input Pin Minimum Pulse Width	—	—	0.3	—	—	μs
t_{INT}	Interrupt Pin Minimum Pulse Width	—	—	10	—	—	μs
t_{RES}	External Reset Minimum Low Pulse Width	—	—	10	—	—	μs

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

Memory Characteristics

Ta= -40°C~85°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{RW}	V_{DD} for Read / Write	—	—	V_{DDmin}	—	V_{DDmax}	V
Program Flash / Data EEPROM Memory							
t_{DEW}	Erase / Write Cycle Time	—	—	2.2	2.5	2.8	ms
I_{DDPGM}	Programming / Erase Current on V_{DD}	—	—	—	—	5.0	mA
E_P	Cell Endurance	—	—	100K	—	—	E/W
t_{RETD}	ROM Data Retention Time	—	Ta=25°C	—	40	—	Year
RAM Data Memory							
V_{DR}	RAM Data Retention Voltage	—	Device in SLEEP Mode	1.0	—	—	V

LVD & LVR Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{LVR}	Low Voltage Reset Voltage	—	LVR enable, voltage select 2.1V	-5%	2.1	+5%	V
			LVR enable, voltage select 2.55V		2.55		
			LVR enable, voltage select 3.15V		3.15		
			LVR enable, voltage select 3.8V		3.8		
V_{LVD}	Low Voltage Detection Voltage	—	LVD enable, voltage select 2.0V	-5%	2.0	+5%	V
			LVD enable, voltage select 2.2V		2.2		
			LVD enable, voltage select 2.4V		2.4		
			LVD enable, voltage select 2.7V		2.7		
			LVD enable, voltage select 3.0V		3.0		
			LVD enable, voltage select 3.3V		3.3		
			LVD enable, voltage select 3.6V		3.6		
			LVD enable, voltage select 4.0V		4.0		
$I_{LVRLVDBG}$	Operating Current	3V	LVD enable, LVR enable, VBGEN=0	—	13	20	μA
		5V		—	16	24	
		3V	LVD enable, LVR enable, VBGEN=1	—	90	135	μA
		5V		—	130	185	
t_{LVDS}	LVDO Stable Time	—	For LVR enable, VBGEN=0, LVD off → on	—	—	15	μs
			For LVR disable, VBGEN=0, LVD off → on	—	—	150	
t_{LVR}	Minimum Low Voltage Width to Reset	—	—	120	240	480	μs
t_{LVD}	Minimum Low Voltage Width to Interrupt	—	—	60	120	240	μs
I_{LVR}	Additional Current for LVR Enable	—	LVD disable, VBGEN=0	—	30	45	μA
I_{LVD}	Additional Current for LVD Enable	—	LVR disable, VBGEN=0	—	10	20	μA

A/D Converter Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.7	—	5.5	V
V_{ADI}	Input Voltage	—	—	0	—	V_{REF}	V
V_{REF}	Reference Voltage	—	—	2	—	V_{DD}	V
DNL	Differential Non-linearity	3V	$V_{REF}=V_{DD}$, $t_{ADCK}=0.5\mu s$	—	—	±3	LSB
		5V					
		3V	$V_{REF}=V_{DD}$, $t_{ADCK}=10\mu s$				
		5V					
INL	Integral Non-linearity	3V	$V_{REF}=V_{DD}$, $t_{ADCK}=0.5\mu s$	—	—	±4	LSB
		5V					
		3V	$V_{REF}=V_{DD}$, $t_{ADCK}=10\mu s$				
		5V					
I_{ADC}	Additional Current for A/D Converter Enable	3V	No load, $t_{ADCK}=0.5\mu s$	—	1	2	mA
		5V		—	1.5	3	
t_{ADCK}	Clock Period	—	—	0.5	—	10	μs
t_{ON2ST}	A/D Converter On-to-Start Time	—	—	4	—	—	μs
t_{ADC}	Conversion Time(Include A/D Sample and Hold Time)	—	—	16	—	20	t_{ADCK}

Internal Reference Voltage Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.2	—	5.5	V
V_{BGREF}	Bandgap Reference Voltage	5.0V	—	-1%	1.2	+1%	V
		2.2V~5.5V	Ta= -40°C~85°C	-2%	1.2	+2%	V
I_{BGREF}	Operating Current	5.5V	Ta= -40°C~85°C	—	35	55	μA
I_{SD}	Shutdown Current	—	VBGREN=0	—	—	0.1	μA
t_{START}	Startup Time	2.2V~5.5V	—	—	—	400	μs

Note: 1. A 0.1μF ceramic capacitor should be connected between VDD and GND.
2. The V_{BGREF} voltage is used as the A/D converter internal signal input.

Operational Amplifier Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{OPA}	Additional Current for Operational Amplifier Enable	—	No load, SDAmBW[1:0]=00B (m=0, 1)	—	1.2	2.4	μA
			No load, SDAmBW[1:0]=01B (m=0, 1)	—	10	16	
			No load, SDAmBW[1:0]=10B (m=0, 1)	—	80	128	
			No load, SDAmBW[1:0]=11B (m=0, 1)	—	200	320	
V_{OS}	Input Offset Voltage	5V	Without calibration (SDAmOF[5:0]=100000B, m=0, 1)	-15	—	+15	mV
			With calibration	-2	—	+2	mV
V_{CM}	Common Mode Voltage Range	—	—	V_{SS}	—	V_{DD}-1.4	V
PSRR	Power Supply Rejection Ratio	5V	—	58	70	—	dB
CMRR	Common Mode Rejection Ratio	5V	—	58	80	—	dB
A_{OL}	Open Loop Gain	—	—	60	80	—	dB
SR	Slew Rate +, Slew Rate -	5V	R_L=1MΩ, C_L=60pF, SDA1BW[1:0]=00 or SDA0BW[1:0]=00	1.0	1.5	—	V/ms
			R_L=1MΩ, C_L=60pF, SDA1BW[1:0]=01 or SDA0BW[1:0]=01	10	15	—	
			R_L=1MΩ, C_L=60pF, SDA1BW[1:0]=10 or SDA0BW[1:0]=10	300	500	—	
			R_L=1MΩ, C_L=60pF, SDA1BW[1:0]=11 or SDA0BW[1:0]=11	1200	1800	—	
GBW	Operational Amplifier Gain Band Bandwidth (Each Operational Amplifier)	5V	R_L=1MΩ, C_L=100pF, SDA1BW[1:0]=00 or SDA0BW[1:0]=00	2.5	5	—	kHz
			R_L=1MΩ, C_L=100pF, SDA1BW[1:0]=01 or SDA0BW[1:0]=01	20	40	—	
			R_L=1MΩ, C_L=100pF, SDA1BW[1:0]=10 or SDA0BW[1:0]=10	400	600	—	
			R_L=1MΩ, C_L=100pF, SDA1BW[1:0]=11 or SDA0BW[1:0]=11	1300	2000	—	

Comparator Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DDC}	Comparator Operating Voltage	—	—	2.2	—	5.5	V
I_{DD}	Comparator Operating Current (Each Comparator)	—	No load, SDC1IS[1:0]=00 or SDC0IS[1:0]=00	—	1.7	2.7	μA
			No load, SDC1IS[1:0]=01 or SDC0IS[1:0]=01	—	14	22	
			No load, SDC1IS[1:0]=10 or SDC0IS[1:0]=10	—	36	57	
			No load, SDC1IS[1:0]=11 or SDC0IS[1:0]=11	—	58	92	
V_{OPOS}	Comparator Input Offset Voltage	5V	SDCnOF[4:0]=10000B V_{DDC} > 250mV	-10	—	+10	mV
			With calibration, V_{DDC} > 250mV	-4	—	+4	mV
V_{CM}	Comparator Common Mode Voltage Range	—	—	V_{SS}	—	V_{DD}-1.4V	V

8-Bit D/A Converter Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.2	—	5.5	V
V_{DACO}	Output Voltage Range	—	—	V_{SS}	—	V_{REF}	V
V_{REF}	Reference Voltage	—	—	2	—	V_{DD}	V
I_{DAC}	Additional Current for D/A Converter Enable (Each D/A Converter)	3V	—	—	—	200	μA
		5V	—	—	—	280	μA
t_{ST}	Settling Time	3V	C_{LOAD}=50pF	—	—	10	μs
		5V		—	—	10	μs
DNL	Differential Non-linearity	3V	V_{REF}=V_{DD}	—	—	±1	LSB
		5V		—	—	±1	LSB
INL	Integral Non-linearity	3V	V_{REF}=V_{DD}	—	—	±1.5	LSB
		5V		—	—	±1.5	LSB

16-Bit Voice D/A Converter Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.2	5.0	5.5	V
I_{DAC}	Additional Current for D/A Converter Enable With Buffer	3V	—	—	—	3	mA
		5V		—	—	3	
$I_{STB\ (DAC)}$	Standby Current	5V	DACEN=0	—	—	1	μA
THD+N	Total Harmonic Distortion + Noise(Note)	3V	10kΩ load	—	-55	—	dB
V_{OUT}	Output Voltage Range	5V	No load	0.01	—	0.99	V_{DD}
t_{DACS}	D/A Converter Circuit Turn On Stable Time	5V	—	—	—	1	ms

Note: Sin wave input @ 1kHz, -6dBFS

LCD Electrical Characteristics

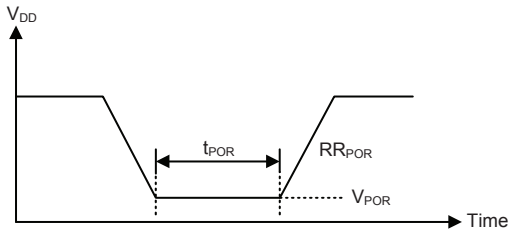
Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{BIAS}	V_{DD}/2 Bias Current for LCD	3V	ISEL[1:0]=00B	10.5	15	19.5	μA
		5V		17.5	25	32.5	
		3V	ISEL[1:0]=01B	21	30	39	μA
		5V		35	50	65	
		3V	ISEL[1:0]=10B	42	60	78	μA
		5V		70	100	130	
		3V	ISEL[1:0]=11B	82.6	118	153.4	μA
		5V		140	200	260	
V_{SCOM}	V_{DD}/2 Voltage for LCD COM Port	2.2V~5.5V	No load	0.475V_{DD}	0.5V_{DD}	0.525V_{DD}	V

Power-on Reset Characteristics

Ta=25°C

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

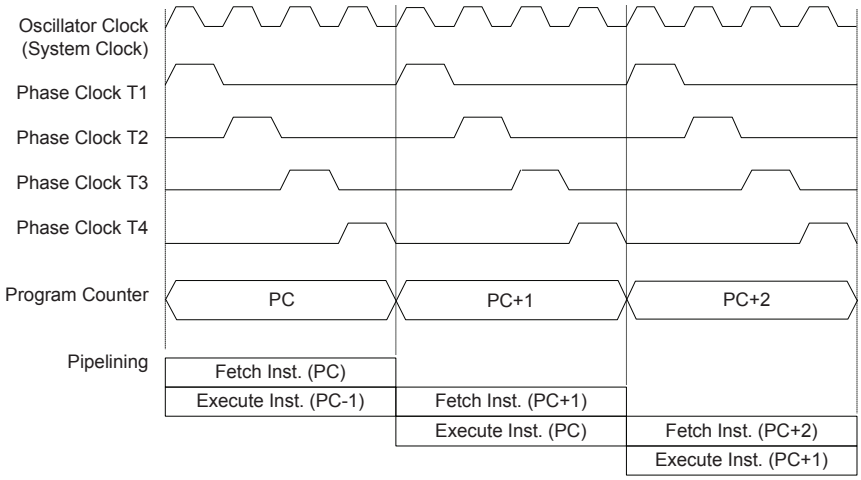


System Architecture

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

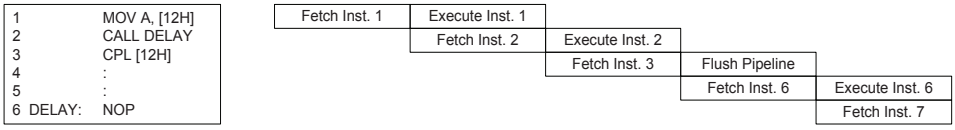
Clocking and Pipelining

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



System Clocking and Pipelining

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



Instruction Fetching

Program Counter

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

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

Device	Program Counter	
	High Byte	Low Byte (PCL)
HT66F4530	PC10~PC8	PCL7~PCL0
HT66F4540	PC11~PC8	PCL7~PCL0
HT66F4550	PC12~PC8	PCL7~PCL0

Program Counter

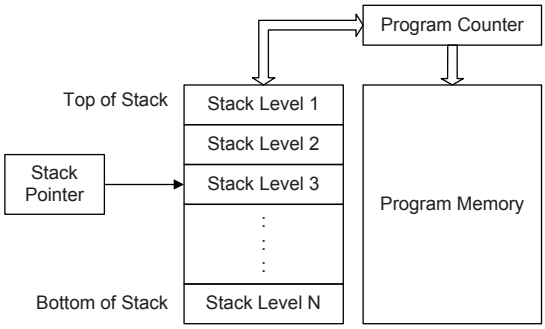
The lower byte of the Program Counter, known as the Program Counter Low register or PCL, is available for program control and is a readable and writeable register. By transferring data directly into this register, a short program jump can be executed directly; however, as only this low byte is available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

Stack

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

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

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



Note: N=6 for HT66F4530 while N=8 for HT66F4540/HT66F4550.

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,
  LRR, LRRA, LRRCA, LRRC, LRLA, LRL, LRLCA, LRLC
- Increment and Decrement:
  INCA, INC, DECA, DEC,
  LINCA, LINC, LDECA, LDEC
- Branch decision:
  JMP, SZ, SZA, SNZ, SIZ, SDZ, SIZA, SDZA, CALL, RET, RETI,
  LSZ, LSZA, LSNZ, LSIZ, LSDZ, LSIZA, LSDZA

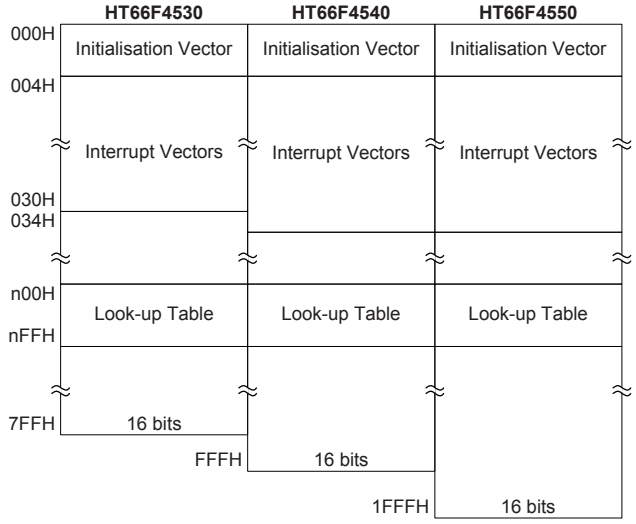
Flash Program Memory

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

Device	Capacity
HT66F4530	2K×16
HT66F4540	4K×16
HT66F4550	8K×16

Structure

The Program Memory has a capacity of 2K×16 to 8K×16 bits. The Program Memory is addressed by the Program Counter and also contains data, table information and interrupt entries. Table data, which can be setup in any location within the Program Memory, is addressed by a separate table pointer register.



Program Memory Structure

Special Vectors

Within the Program Memory, certain locations are reserved for the reset and interrupts. The location 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 registers, TBLP and TBHP. These registers define the total address of the look-up table.

After setting up the table pointer, the table data can be retrieved from the Program Memory using the corresponding table read instruction such as "TABRD [m]" or "TABRDL [m]" respectively when the memory [m] is located in sector 0. If the memory [m] is located in other sectors, the data can be retrieved from the program memory using the corresponding extended table read instruction such as "LTABRD [m]" or "LTABRDL [m]" respectively. When the instruction is executed, the lower order table byte from the Program Memory will be transferred to the user defined Data Memory register [m] as specified in the instruction. The higher order table data byte from the Program Memory will be transferred to the TBLH special register.

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

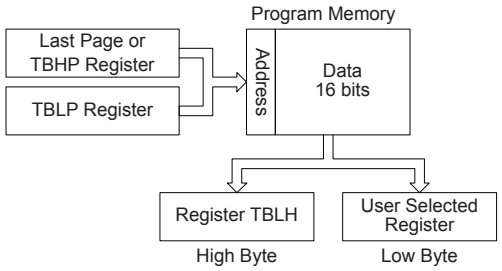


Table Program Example

The following example shows how the table pointer and table data is defined and retrieved from the microcontroller. This example uses raw table data located in the Program Memory which is stored there using the ORG statement. The value at this ORG statement is "0700H" which refers to the start address of the last page within the 2K Program Memory of the HT66F4530 device. The table pointer low byte register is setup here to have an initial value of "06H". This will ensure that the first data read from the data table will be at the Program Memory address "0706H" or 6 locations after the start of the last page. Note that the value for the table pointer is referenced to the first address specified by TBLP and TBHP if the "TABRD [m]" or "LTABRD [m]" instruction is being used. The high byte of the table data which in this case is equal to zero will be transferred to the TBLH register automatically when the "TABRD [m]" or "LTABRD [m]" instruction is executed.

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

Table Read Program Example

```
tempreg1 db ?   ; temporary register #1
tempreg2 db ?   ; temporary register #2
:
:
mov a,06h       ; initialise low table pointer - note that this address is referenced
mov tblp,a      ; to the last page or the page that tbhp pointed
mov a,07h       ; initialise high table pointer
mov tbhp,a
:
:
tabrd tempreg1  ; transfers value in table referenced by table pointer data at program
                ; memory address "0706H" transferred to tempreg1 and TBLH
dec tblp        ; reduce value of table pointer by one
tabrd tempreg2  ; transfers value in table referenced by table pointer data at program
                ; memory address "0705H" transferred to tempreg2 and TBLH in this example
                ; the data "1AH" is transferred to tempreg1 and data "0FH" to register
                ; tempreg2
:
:
org 0700h       ; sets initial address of program memory
dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
:
:
```

In Circuit Programming – ICP

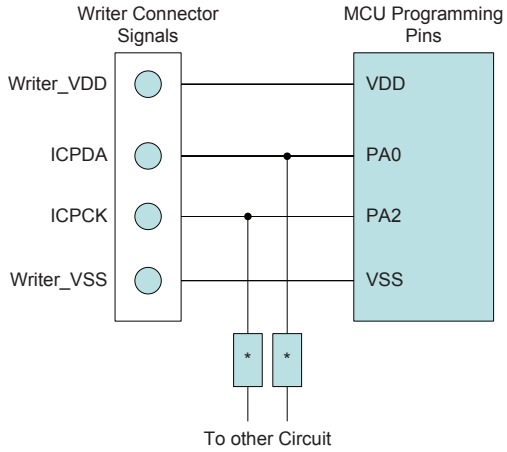
The provision of Flash type Program Memory provides the user with a means of convenient and easy upgrades and modifications to their programs on the same device. As an additional convenience, Holtek has provided a means of programming the microcontroller in-circuit using a 4-pin interface. This provides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

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

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

The Program Memory and EEPROM Data Memory can be programmed serially in-circuit using this 4-wire interface. Data is downloaded and uploaded serially on a single pin with an additional line for the clock. Two additional lines are required for the power supply. The technical details regarding the in-circuit programming of these 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 1kΩ or the capacitance of * must be less than 1nF.

On Chip Debug Support – OCDS

There is an EV chip named HT66V45x0 which is used to emulate the HT66F45x0 device. The EV chip device also provides an "On-Chip Debug" function to debug the real MCU device during the development process. The EV chip and the real MCU device are almost functionally compatible except for "On-Chip Debug" function. Users can use the EV chip device to emulate the real chip device behavior by connecting the OCDSDA and OCDSCK pins to the Holtek HT-IDE development tools. The OCDSDA pin is the OCDS Data/Address input/output pin while the OCDSCK pin is the OCDS clock input pin. When users use the EV chip for debugging, other functions which are shared with the OCDSDA and OCDSCK pins in these devices will have no effect in the EV chip. However, the two OCDS pins which are pin-shared with the ICP programming pins are still used as the Flash Memory programming pins for ICP. For more detailed OCDS information, refer to the corresponding document named "Holtek e-Link for 8-bit MCU OCDS User's Guide".

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

Data Memory

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

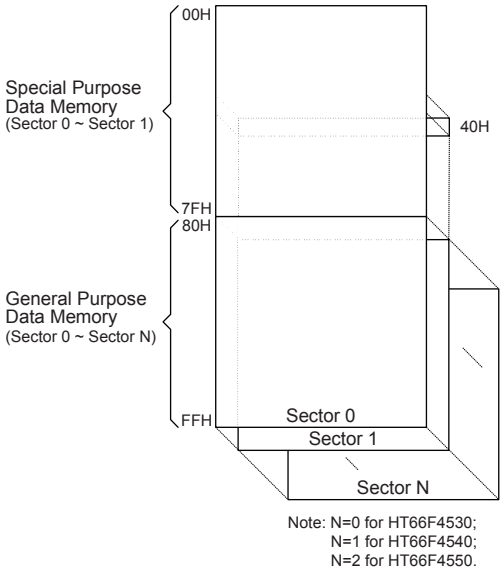
Categorized into two types, the first of these is an area of RAM where special function registers are located. These registers have fixed locations and are necessary for correct operation of these devices. Many of these registers can be read from and written to directly under program control, however,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

Switching between the different Data Memory sectors is achieved by properly setting the Memory Pointers to correct value.

Structure

The Data Memory is subdivided into several sectors, all of which are implemented in 8-bit wide RAM. Each of the Data Memory Sector is categorized into two types, the special Purpose Data Memory and the General Purpose Data Memory. The address range of the Special Purpose Data Memory for these devices is from 00H to 7FH while the General Purpose Data Memory address range is from 80H to FFH.

Device	Special Purpose Data Memory	General Purpose Data Memory	
	Located Sectors	Capacity	Sector: Address
HT66F4530	0: 00H~7FH 1: 40H (EEC)	128×8	0: 80H~FFH
HT66F4540	0: 00H~7FH 1: 40H (EEC)	256×8	0: 80H~FFH 1: 80H~FFH
HT66F4550	0: 00H~7FH 1: 40H (EEC)	384×8	0: 80H~FFH 1: 80H~FFH 2: 80H~FFH



Data Memory Structure

Data Memory Addressing

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

Direct Addressing can be used in all sectors using the corresponding instruction which can address all available data memory space. For the accessed data memory which is located in any data memory sectors except sector 0, the extended instructions can be used to access the data memory instead of using the indirect addressing access. The main difference between standard instructions and extended instructions is that the data memory address "m" in the extended instructions has up to 10 valid bits, the high byte indicates a sector and the low byte indicates a specific address.

General Purpose Data Memory

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

Special Purpose Data Memory

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

	Sector 0	Sector 1
00H	IAR0	
01H	MP0	
02H	IAR1	
03H	MP1L	
04H	MP1H	
05H	ACC	
06H	PCL	
07H	TBLP	
08H	TBLH	
09H	TBHP	
0AH	STATUS	
0BH		
0CH	IAR2	
0DH	MP2L	
0EH	MP2H	
0FH	RSTFC	
10H	SCC	
11H	HIRCC	
12H	HXTC	
13H	LXTC	
14H	PA	
15H	PAC	
16H	PAPU	
17H	PAWU	
18H	PSCR	
19H	LVRC	
1AH	WDTC	
1BH	TB0C	
1CH	TB1C	
1DH	LVDC	
1EH		
1FH	RSTC	
20H	SADOL	
21H	SADOH	
22H	SADC0	
23H	SADC1	
24H	VBGRC	
25H	PB	
26H	PBC	
27H	PBPU	
28H	PTM0C0	
29H	PTM0C1	
2AH	PTM0DL	
2BH	PTM0DH	
2CH	PTM0AL	
2DH	PTM0AH	
2EH	PTM0RPL	
2FH	PTM0RPH	
30H	STM0C0	
31H	STM0C1	
32H	STM0DL	
33H	STM0DH	
34H	STM0AL	
35H	STM0AH	
36H		
37H		
38H		
39H		
3AH		
3BH		
3CH		
3DH		
3EH		
3FH		

	Sector 0	Sector 1
40H		EEC
41H	EEA	
42H	EED	
43H		
44H		
45H	SIMC0	
46H	SIMC1	
47H	SIMD	
48H	SIMA/SIMC2	
49H	SIMTOC	
4AH	INTEG	
4BH	INTC0	
4CH	INTC1	
4DH	INTC2	
4EH	INTC3	
4FH	MFI0	
50H	MFI1	
51H	IFS	
52H	PD	
53H	PDC	
54H	PDPU	
55H		
56H		
57H	PC	
58H	PCC	
59H	PCPU	
5AH		
5BH		
5CH		
5DH		
5EH		
5FH		
60H	PAS0	
61H	PAS1	
62H	PBS0	
63H	PBS1	
64H	PCS0	
65H	PCS1	
66H	PDS0	
67H		
68H	SKLEDC	
69H	SDSW0	
6AH	SDSW1	
6BH	SDDAC0C	
6CH	SDDA0L	
6DH	SDDAC1C	
6EH	SDDA1L	
6FH	SDDAC2C	
70H	SDDA2L	
71H	SDPGAC	
72H	SDA0C	
73H	SDA0VOS	
74H	SDA1C	
75H	SDA1VOS	
76H	SDC0C	
77H	SDC0VOS	
78H	SDC1C	
79H	SDC1VOS	
7AH	SDCHYC	
7BH		
7CH		
7DH		
7EH		
7FH		

: Unused, read as 00H

Special Purpose Data Memory Structure – HT66F4530

	Sector 0	Sector 1
00H	IAR0	
01H	MP0	
02H	IAR1	
03H	MP1L	
04H	MP1H	
05H	ACC	
06H	PCL	
07H	TBLP	
08H	TBLH	
09H	TBHP	
0AH	STATUS	
0BH		
0CH	IAR2	
0DH	MP2L	
0EH	MP2H	
0FH	RSTFC	
10H	SCC	
11H	HIRCC	
12H	HXTC	
13H	LXTC	
14H	PA	
15H	PAC	
16H	PAPU	
17H	PAWU	
18H	PSCR	
19H	LVRC	
1AH	WDTC	
1BH	TB0C	
1CH	TB1C	
1DH	LVDC	
1EH	SCOMC	
1FH	RSTC	
20H	SADOL	
21H	SADOH	
22H	SADC0	
23H	SADC1	
24H	VBGRC	
25H	PB	
26H	PBC	
27H	PBPU	
28H	PTM0C0	
29H	PTM0C1	
2AH	PTM0DL	
2BH	PTM0DH	
2CH	PTM0AL	
2DH	PTM0AH	
2EH	PTM0RPL	
2FH	PTM0RPH	
30H	STM0C0	
31H	STM0C1	
32H	STM0DL	
33H	STM0DH	
34H	STM0AL	
35H	STM0AH	
36H		
37H	PTM1C0	
38H	PTM1C1	
39H	PTM1DL	
3AH	PTM1DH	
3BH	PTM1AL	
3CH	PTM1AH	
3DH	PTM1RPL	
3EH	PTM1RPH	
3FH		

	Sector 0	Sector 1
40H		EEC
41H	EEA	
42H	EED	
43H		
44H		
45H	SIMC0	
46H	SIMC1	
47H	SIMD	
48H	SIMA/SIMC2	
49H	SIMTOC	
4AH	INTEG	
4BH	INTC0	
4CH	INTC1	
4DH	INTC2	
4EH	INTC3	
4FH	MFI0	
50H	MFI1	
51H	IFS	
52H	PD	
53H	PDC	
54H	PDPU	
55H		
56H		
57H	PC	
58H	PCC	
59H	PCPU	
5AH		
5BH		
5CH		
5DH		
5EH		
5FH		
60H	PAS0	
61H	PAS1	
62H	PBS0	
63H	PBS1	
64H	PCS0	
65H	PCS1	
66H	PDS0	
67H		
68H	SKLEDC	
69H	SDSW0	
6AH	SDSW1	
6BH	SDDAC0C	
6CH	SDDA0L	
6DH	SDDAC1C	
6EH	SDDA1L	
6FH	SDDAC2C	
70H	SDDA2L	
71H	SDPGAC	
72H	SDA0C	
73H	SDA0VOS	
74H	SDA1C	
75H	SDA1VOS	
76H	SDC0C	
77H	SDC0VOS	
78H	SDC1C	
79H	SDC1VOS	
7AH	SDCHYC	
7BH	USR	
7CH	UCR1	
7DH	UCR2	
7EH	TXR_RXR	
7FH	BRG	

: Unused, read as 00H

Special Purpose Data Memory Structure – HT66F4540

	Sector 0	Sector 1
00H	IAR0	
01H	MP0	
02H	IAR1	
03H	MP1L	
04H	MP1H	
05H	ACC	
06H	PCL	
07H	TBLP	
08H	TBLH	
09H	TBHP	
0AH	STATUS	
0BH		
0CH	IAR2	
0DH	MP2L	
0EH	MP2H	
0FH	RSTFC	
10H	SCC	
11H	HIRCC	
12H	HXTC	
13H	LXTC	
14H	PA	
15H	PAC	
16H	PAPU	
17H	PAWU	
18H	PSCR	
19H	LVRC	
1AH	WDTC	
1BH	TB0C	
1CH	TB1C	
1DH	LVDC	
1EH	SCOMC	
1FH	RSTC	
20H	SADOL	
21H	SADOH	
22H	SADC0	
23H	SADC1	
24H	VBGRC	
25H	PB	
26H	PBC	
27H	PBPU	
28H	PTM0C0	
29H	PTM0C1	
2AH	PTM0DL	
2BH	PTM0DH	
2CH	PTM0AL	
2DH	PTM0AH	
2EH	PTM0RPL	
2FH	PTM0RPH	
30H	STM0C0	
31H	STM0C1	
32H	STM0DL	
33H	STM0DH	
34H	STM0AL	
35H	STM0AH	
36H		
37H	PTM1C0	
38H	PTM1C1	
39H	PTM1DL	
3AH	PTM1DH	
3BH	PTM1AL	
3CH	PTM1AH	
3DH	PTM1RPL	
3EH	PTM1RPH	
3FH		

	Sector 0	Sector 1
40H		EEC
41H	EEA	
42H	EED	
43H	DACC	
44H		
45H	SIMC0	
46H	SIMC1	
47H	SIMD	
48H	SIMA/SIMC2	
49H	SIMTOC	
4AH	INTEG	
4BH	INTC0	
4CH	INTC1	
4DH	INTC2	
4EH	INTC3	
4FH	MFI0	
50H	MFI1	
51H	IFS	
52H	PD	
53H	PDC	
54H	PDPU	
55H	DAL	
56H	DAH	
57H	PC	
58H	PCC	
59H	PCPU	
5AH	STM1C0	
5BH	STM1C1	
5CH	STM1DL	
5DH	STM1DH	
5EH	STM1AL	
5FH	STM1AH	
60H	PAS0	
61H	PAS1	
62H	PBS0	
63H	PBS1	
64H	PCS0	
65H	PCS1	
66H	PDS0	
67H		
68H	SKLEDC	
69H	SDSW0	
6AH	SDSW1	
6BH	SDDAC0C	
6CH	SDDA0L	
6DH	SDDAC1C	
6EH	SDDA1L	
6FH	SDDAC2C	
70H	SDDA2L	
71H	SDPGAC	
72H	SDA0C	
73H	SDA0VOS	
74H	SDA1C	
75H	SDA1VOS	
76H	SDC0C	
77H	SDC0VOS	
78H	SDC1C	
79H	SDC1VOS	
7AH	SDCHYC	
7BH	USR	
7CH	UCR1	
7DH	UCR2	
7EH	TXR_RXR	
7FH	BRG	

: Unused, read as 00H

Special Purpose Data Memory Structure – HT66F4550

Special Function Register Description

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

Indirect Addressing Registers – IAR0, IAR1, IAR2

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

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

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

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

Indirect Addressing Program Example 1

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

Indirect Addressing Program Example 2

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

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

Direct Addressing Program Example using extended instructions

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

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

Accumulator – ACC

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

Program Counter Low Register – PCL

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

Look-up Table Registers – TBLP, TBHP, TBLH

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

Status Register – STATUS

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

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

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

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

- PDF is cleared by a system power-up or executing the "CLR WDT" instruction. PDF is set by executing the "HALT" instruction.
- TO is cleared by a system power-up or executing the "CLR WDT" or "HALT" instruction. TO is set by a WDT time-out.
- CZ is the operational result of different flags for different instructions. Refer to register definitions for more details.
- SC is the result of the "XOR" operation which is performed by the OV flag and the MSB of the current instruction operation result.

In addition, on entering an interrupt sequence or executing a subroutine call, the status register will not be pushed onto the stack automatically. If the contents of the status registers are important and if the subroutine can corrupt the status register, precautions must be taken to correctly save it. Note that bits 0~3 of the STATUS register are both readable and writeable bits.

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

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

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

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

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

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

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

EEPROM Data Memory

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
HT66F4530	32×8	00H~1FH
HT66F4540	64×8	00H~3FH
HT66F4550	64×8	00H~3FH

EEPROM Data Memory Structure

The EEPROM Data Memory capacity is 32×8 to 64×8 bits for these 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 and Write operations to the EEPROM are carried out in single byte operations using an address and a data register in Sector 0 and a single control register in Sector 1.

EEPROM Registers

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

Register Name	Bit							
	7	6	5	4	3	2	1	0
EEA(HT66F4530)	—	—	—	EEA4	EEA3	EEA2	EEA1	EEA0
EEA(HT66F4540/ HT66F4550)	—	—	EEA5	EEA4	EEA3	EEA2	EEA1	EEA0
EED	D7	D6	D5	D4	D3	D2	D1	D0
EEC	—	—	—	—	WREN	WR	RDEN	RD

EEPROM Registers List

EEA Register – HT66F4530

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

Bit 7~5 Unimplemented, read as "0"

Bit 4~0 **EEA4~EEA0**: Data EEPROM address
Data EEPROM address bit 4 ~ bit 0

EEA Register – HT66F4540/HT66F4550

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

Bit 7~6 Unimplemented, read as "0"

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

EED Register

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

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

EEC Register

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

Bit 7~4 Unimplemented, read as "0"

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

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

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

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

Note: The WREN, WR, RDEN and RD cannot be set high at the same time in one instruction. The WR and RD cannot be set high 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 these devices are powered-on the Write Enable bit in the control register will be cleared preventing any write operations. Also at power-on the Memory Pointer high byte register, MP1H or MP2H, will be reset to zero, which means that Data Memory Sector 0 will be selected. As the EEPROM control register is located in Sector 1, this adds a further measure of protection against spurious write operations. During normal program operation, ensuring that the Write Enable bit in the control register is cleared will safeguard against incorrect write operations.

EEPROM Interrupt

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

Programming Considerations

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

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

Programming Examples

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

```
MOV  A, EEPROM_ADRES          ; user defined address
MOV  EEA, A
MOV  A, 040H                  ; setup memory pointer MP1L
MOV  MP1L, A                  ; MP1L 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 method**

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

Oscillators

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

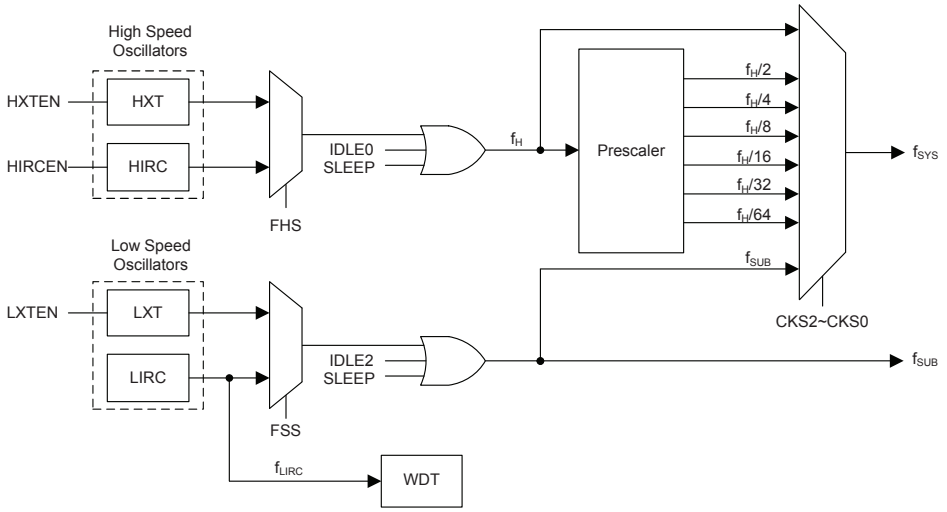
Type	Name	Frequency	Pins
External High Speed Crystal	HXT	400kHz~12MHz	OSC1/OSC2
Internal High Speed RC	HIRC	2/4/8MHz	—
External Low Speed Crystal	LXT	32.768kHz	XT1/XT2
Internal Low Speed RC	LIRC	32kHz	—

Oscillator Types

System Clock Configurations

There are several oscillator sources, two high speed oscillators and two low speed oscillators. The high speed system clocks are sourced from the external crystal/ceramic oscillator, HXT, and the internal 2/4/8MHz RC oscillator, HIRC. The low speed oscillators are the external 32.768kHz crystal oscillator, LXT, and the internal 32kHz RC oscillator, LIRC. Selecting whether the low or high speed oscillator is used as the system oscillator is implemented using the CKS2~CKS0 bits in the SCC register and as the system clock can be dynamically selected.

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

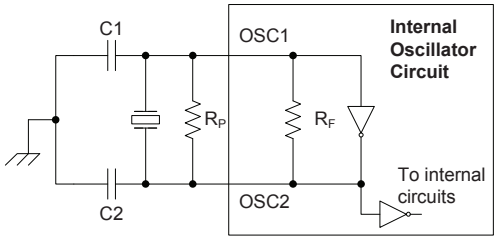


System Clock Configurations

External Crystal/Ceramic Oscillator – HXT

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

For oscillator stability and to minimise the effects of noise and crosstalk, it is important to ensure that the crystal and any associated resistors and capacitors along with interconnecting lines are all located as close to the MCU as possible.



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

Crystal/Resonator Oscillator – HXT

Crystal Oscillator C1 and C2 Values		
Crystal Frequency	**C1**	**C2**
12MHz	0pF	0pF
8MHz	0pF	0pF
6MHz	0pF	0pF
4MHz	0pF	0pF
1MHz	100pF	100pF
Note: C1 and C2 values are for guidance only.		

Crystal Recommended Capacitor Values

Internal High Speed RC Oscillator – HIRC

The internal RC oscillator is a fully integrated system oscillator requiring no external components. The internal RC oscillator has three fixed frequencies of 2MHz, 4MHz and 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. Note that if this internal system clock option is selected, as it requires no external pins for its operation, I/O pins PC0 and PC1 are free for use as normal I/O pins.

External 32.768kHz Crystal Oscillator – LXT

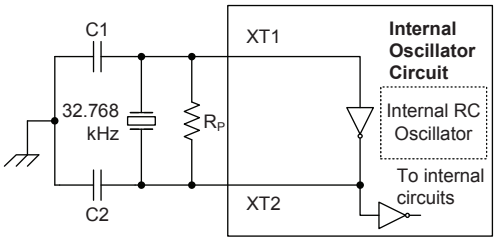
The External 32.768kHz Crystal System Oscillator is one of the low frequency oscillator choices, which is selected via a software control bit, FSS. This clock source has a fixed frequency of 32.768kHz and requires a 32.768kHz crystal to be connected between pins XT1 and XT2. The external resistor and 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. After the LXT oscillator is enabled by setting the LXTEN bit to 1, there is a time delay associated with the LXT oscillator waiting for it to start-up.

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, R_P, is required.

The pin-shared software control bits determine if the XT1/XT2 pins are used for the LXT oscillator or as I/O or other pin-shared functional pins.

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

For oscillator stability and to minimise the effects of noise and crosstalk, it is important to ensure that the crystal and any associated resistors and capacitors along with interconnecting lines are all located as close to the MCU as possible.



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

External LXT Oscillator

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

32.768kHz Crystal Recommended Capacitor Values

Internal 32kHz Oscillator – LIRC

The Internal 32kHz System Oscillator is one of the low frequency oscillator choices, which is selected via a software control bit, FSS. 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.

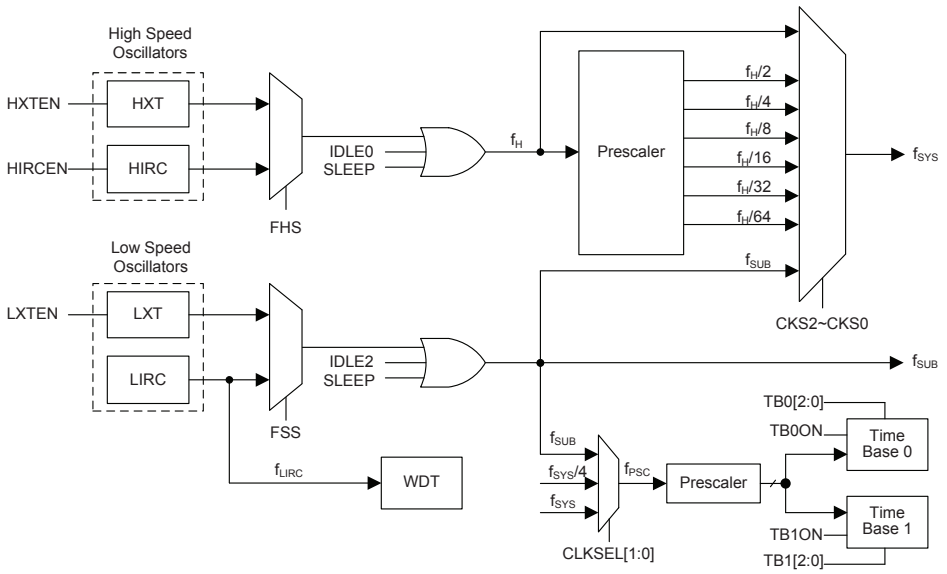
Operating Modes and System Clocks

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

System Clocks

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

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



Device Clock Configurations

Note: When the system clock source f_{SYS} is switched to f_{SUB} from f_H, the high speed oscillator will stop to conserve the power or continue to oscillate to provide the clock source, $f_H \sim f_H/64$, for peripheral circuit to use, which is determined by configuring the corresponding high speed oscillator enable control bit.

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 FAST Mode and SLOW Mode. The remaining four modes, the SLEEP, IDLE0, IDLE1 and IDLE2 Mode are used when the microcontroller CPU is switched off to conserve power.

Operation Mode	CPU	Register Setting			f_{SYS}	f_H	f_{SUB}	f_{LIRC}
		FHIDEN	FSIDEN	CKS2~CKS0				
FAST	On	x	x	000~110	$f_H \sim f_H/64$	On	On	On
SLOW	On	x	x	111	f_{SUB}	On/Off (1)	On	On
IDLE0	Off	0	1	000~110	Off	Off	On	On
				111	On			
IDLE1	Off	1	1	xxx	On	On	On	On
IDLE2	Off	1	0	000~110	On	On	Off	On
				111	Off			
SLEEP	Off	0	0	xxx	Off	Off	Off	On/Off (2)

"x": Don't care

Note: 1. The f_H clock will be switched on or off by configuring the corresponding oscillator enable bit in the SLOW mode.

2. The f_{LIRC} clock can be switched on or off which is controlled by the WDT function being enabled or disabled in the SLEEP mode.

FAST 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 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 bits in the SCC register. Although a high speed oscillator is used, running the microcontroller at a divided clock ratio reduces the operating current.

SLOW Mode

This is also a mode where the microcontroller operates normally although now with a slower speed clock source. The clock source used will be from f_{SUB}. The f_{SUB} clock is derived from either the LIRC or LXT oscillator determined by the FSS bit in the SCC register.

SLEEP Mode

The SLEEP Mode is entered when an HALT instruction is executed and when the FHIDEN and FSIDEN bit are low. In the SLEEP mode the CPU will be stopped. The f_{SUB} clock provided to the peripheral function will also be stopped, too. However the f_{LIRC} clock still continues to operate if the WDT function is enabled.

IDLE0 Mode

The IDLE0 Mode is entered when an HALT instruction is executed and when the FHIDEN bit in the SCC register is low and the FSIDEN bit in the SCC register is high. In the IDLE0 Mode the CPU will be switched off but the low speed oscillator will be turned on to drive some peripheral functions.

IDLE1 Mode

The IDLE1 Mode is entered when an HALT instruction is executed and when the FHIDEN bit in the SCC register is high and the FSIDEN bit in the SCC register is high. In the IDLE1 Mode the CPU will be switched off but both the high and low speed oscillators will be turned on to provide a clock source to keep some peripheral functions operational.

IDLE2 Mode

The IDLE2 Mode is entered when an HALT instruction is executed and when the FHIDEN bit in the SCC register is high and the FSIDEN bit in the SCC register is low. In the IDLE2 Mode the CPU will be switched off but the high speed oscillator will be turned on to provide a clock source to keep some peripheral functions operational.

Control Registers

The registers, SCC, HIRCC, HXTC and LXTC, are used to control the system clock and the corresponding oscillator configurations.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SCC	CKS2	CKS1	CKS0	—	FHS	FSS	FHIDEN	FSIDEN
HIRCC	—	—	—	—	HIRC1	HIRC0	HIRCF	HIRCEN
HXTC	—	—	—	—	—	HXTM	HXTF	HXTEN
LXTC	—	—	—	—	—	—	LXTF	LXTEN

System Operating Mode Control Registers List

SCC Register

Bit	7	6	5	4	3	2	1	0
Name	CKS2	CKS1	CKS0	—	FHS	FSS	FHIDEN	FSIDEN
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 7~5 **CKS2~CKS0**: System clock selection
000: f_H
001: $f_H/2$
010: $f_H/4$
011: $f_H/8$
100: $f_H/16$
101: $f_H/32$
110: $f_H/64$
111: f_{SUB}
These three bits are used to select which clock is used as the system clock source. In addition to the system clock source directly derived from f_H or f_{SUB}, a divided version of the high speed system oscillator can also be chosen as the system clock source.

Bit 4 Unimplemented, read as 0.

Bit 3 **FHS**: High Frequency clock selection
0: HIRC
1: HXT

Bit 2 **FSS**: Low Frequency clock selection
0: LIRC
1: LXT

Bit 1 **FHIDEN**: High Frequency oscillator control when CPU is switched off
0: Disable
1: Enable
This bit is used to control whether the high speed oscillator is activated or stopped when the CPU is switched off by executing an "HALT" instruction.

Bit 0 **FSIDEN**: Low Frequency oscillator control when CPU is switched off
0: Disable
1: Enable
This bit is used to control whether the low speed oscillator is activated or stopped when the CPU is switched off by executing an "HALT" instruction. The LIRC oscillator is controlled by this bit together with the WDT function enable control. If this bit is cleared to 0 but the WDT function is enabled, the f_{LIRC} clock will also be enabled.

HIRCC Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	HIRC1	HIRC0	HIRCF	HIRCEN
R/W	—	—	—	—	R/W	R/W	R	R/W
POR	—	—	—	—	0	0	0	1

Bit 7~4 Unimplemented, read as 0.

Bit 3~2 **HIRC1~HIRC0**: HIRC frequency selection
00: 2MHz
01: 4MHz
10: 8MHz
11: 2MHz
When the HIRC oscillator is enabled or the HIRC frequency selection is changed by application program, the clock frequency will automatically be changed after the HIRCF flag is set to 1.

Bit 1 **HIRCF**: HIRC oscillator stable flag
0: HIRC unstable
1: HIRC stable
This bit is used to indicate whether the HIRC oscillator is stable or not. When the HIRCEN bit is set to 1 to enable the HIRC oscillator or the HIRC frequency selection is changed by application program, the HIRCF bit will first be cleared to 0 and then set to 1 after the HIRC oscillator is stable.

Bit 0 **HIRCEN**: HIRC oscillator enable control
0: Disable
1: Enable

HXTC Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	HXTM	HXTF	HXTEN
R/W	—	—	—	—	—	R/W	R	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 Unimplemented, read as 0.

Bit 2 **HXTM**: HXT mode selection
0: HXT frequency ≤ 10MHz
1: HXT frequency > 10MHz
This bit is used to select the HXT oscillator operating mode. Note that this bit must be properly configured before the HXT is enabled. When the HXTEN bit is set to 1 to enable the HXT oscillator, it is invalid to change the value of this bit.

Bit 1 **HXTF**: HXT oscillator stable flag
0: HXT unstable
1: HXT stable
This bit is used to indicate whether the HXT oscillator is stable or not. When the HXTEN bit is set to 1 to enable the HXT oscillator, the HXTF bit will first be cleared to 0 and then set to 1 after the HXT oscillator is stable.

Bit 0 **HXTEN**: HXT oscillator enable control
0: Disable
1: Enable

LXTC Register

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

Bit 7~2 Unimplemented, read as 0.

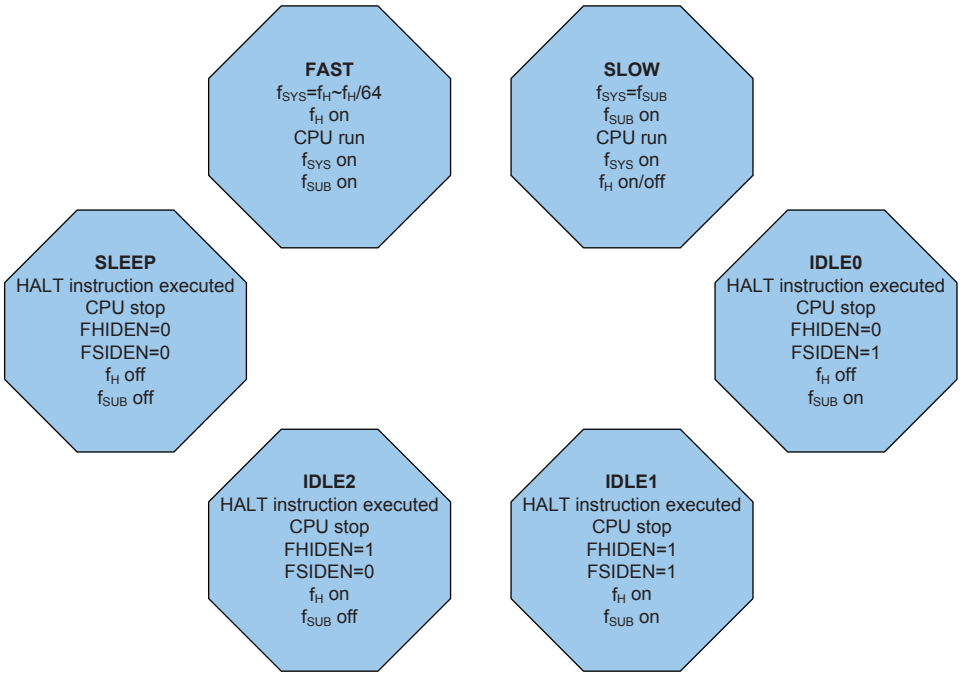
Bit 1 **LXTF**: LXT oscillator stable flag
0: LXT unstable
1: LXT stable
This bit is used to indicate whether the LXT oscillator is stable or not. When the LXTEN bit is set to 1 to enable the LXT oscillator, the LXTF bit will first be cleared to 0 and then set to 1 after the LXT oscillator is stable.

Bit 0 **LXTEN**: LXT oscillator enable control
0: Disable
1: Enable

Operating Mode Switching

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

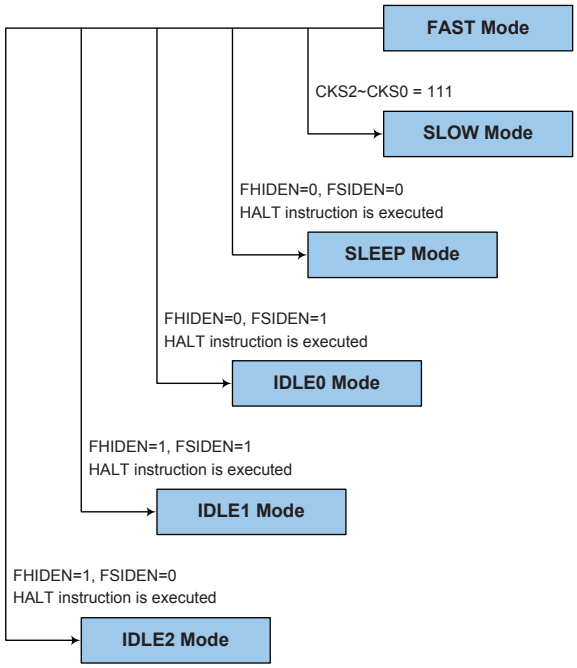
In simple terms, Mode Switching between the FAST Mode and SLOW Mode is executed using the CKS2~CKS0 bits in the SCC register while Mode Switching from the FAST/SLOW Modes to the SLEEP/IDLE Modes is executed via the HALT instruction. When an HALT instruction is executed, whether these devices enter the IDLE Mode or the SLEEP Mode is determined by the condition of the FHIDEN and FSIDEN bits in the SCC register.



FAST Mode to SLOW Mode Switching

When running in the FAST Mode, which uses the high speed system oscillator, and therefore consumes more power, the system clock can switch to run in the SLOW Mode by set the CKS2~CKS0 bits to "111" in the SCC 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.

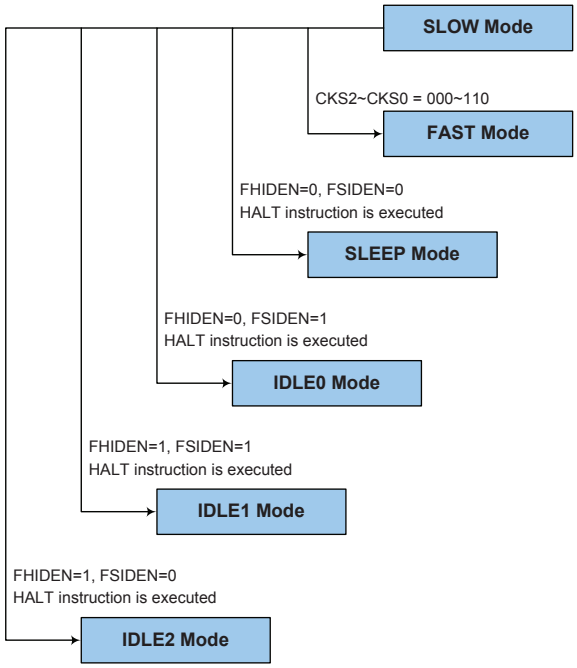
The SLOW Mode is sourced from the LXT or LIRC oscillator determined by the FSS bit in the SCC register and therefore requires this oscillator to be stable before full mode switching occurs.



SLOW Mode to FAST Mode Switching

In SLOW mode the system clock is derived from f_{SUB}. When system clock is switched back to the FAST mode from f_{SUB}, the CKS2~CKS0 bits should be set to "000"~"110" and then the system clock will respectively be switched to $f_H \sim f_H/64$.

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



Entering the SLEEP Mode

There is only one way for these devices to enter the SLEEP Mode and that is to execute the "HALT" instruction in the application program with both the FHIDEN and FSIDEN bits in the SCC register equal to "0". In this mode all the clocks and functions will be switched off except the WDT function. 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.
- The Data Memory contents and registers will maintain their present condition.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.
- The WDT will be cleared and resume counting if the WDT function is enabled. If the WDT function is disabled, the WDT will be cleared and then stopped.

Entering the IDLE0 Mode

There is only one way for these devices to enter the IDLE0 Mode and that is to execute the "HALT" instruction in the application program with the FHIDEN bit in the SCC register equal to "0" and the FSIDEN bit in the SCC register equal to "1". When this instruction is executed under the conditions described above, the following will occur:

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

Entering the IDLE1 Mode

There is only one way for these devices to enter the IDLE1 Mode and that is to execute the "HALT" instruction in the application program with both the FHIDEN and FSIDEN bits in the SCC register equal to "1". When this instruction is executed under the conditions described above, the following will occur:

- The f_H and f_{SUB} clocks will be on but the application program will stop at the "HALT" instruction.
- The Data Memory contents and registers will maintain their present condition.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.
- The WDT will be cleared and resume counting if the WDT function is enabled. If the WDT function is disabled, the WDT will be cleared and then stopped.

Entering the IDLE2 Mode

There is only one way for these devices to enter the IDLE2 Mode and that is to execute the "HALT" instruction in the application program with the FHIDEN bit in the SCC register equal to "1" and the FSIDEN bit in the SCC register equal to "0". When this instruction is executed under the conditions described above, the following will occur:

- The f_H clock will be on but the f_{SUB} clock will be off and the application program will stop at the "HALT" instruction.
- The Data Memory contents and registers will maintain their present condition.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.
- The WDT will be cleared and resume counting if the WDT function is enabled. If the WDT function is disabled, the WDT will be cleared and then stopped.

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 and IDLE2 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 these 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 LXT or LIRC oscillator has enabled.

In the IDLE1 and IDLE2 Mode the high speed oscillator is on, if the peripheral function clock source is derived from the high speed oscillator, the additional standby current will also be perhaps in the order of several hundred micro-amps.

Wake-up

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

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

- An external $\overline{\text{RES}}$ pin reset
- An external falling edge on Port A
- A system interrupt
- A WDT overflow

If the system is woken up by an external $\overline{\text{RES}}$ pin 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 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.

Watchdog Timer

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

Watchdog Timer Clock Source

The Watchdog Timer clock source is provided by the internal clock, f_{LIRC} which is sourced from the LIRC oscillator. The 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 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.

WDTC Register

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

Bit 7~3 **WE4~WE0**: WDT function software control
10101: Disable
01010: Enable
Others: Reset MCU
When these bits are changed by the environmental noise or software setting to reset the microcontroller, the reset operation will be activated after t_{SRESET} time and the WRF bit in the RSTFC register will be set high.

Bit 2~0 **WS2~WS0**: WDT time-out period selection
000: $2^8/f_{LIRC}$
001: $2^{10}/f_{LIRC}$
010: $2^{12}/f_{LIRC}$
011: $2^{14}/f_{LIRC}$
100: $2^{15}/f_{LIRC}$
101: $2^{16}/f_{LIRC}$
110: $2^{17}/f_{LIRC}$
111: $2^{18}/f_{LIRC}$
These three bits determine the division ratio of the watchdog timer source clock, which in turn determines the time-out period.

RSTFC Register

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

"x": unknown

Bit 7~4 Unimplemented, read as "0"

Bit 3 **RSTF**: Reset control register software reset flag
Describe elsewhere.

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 these devices. 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 these devices after t_{SRESET} time. After power on these bits will have a value of 01010B.

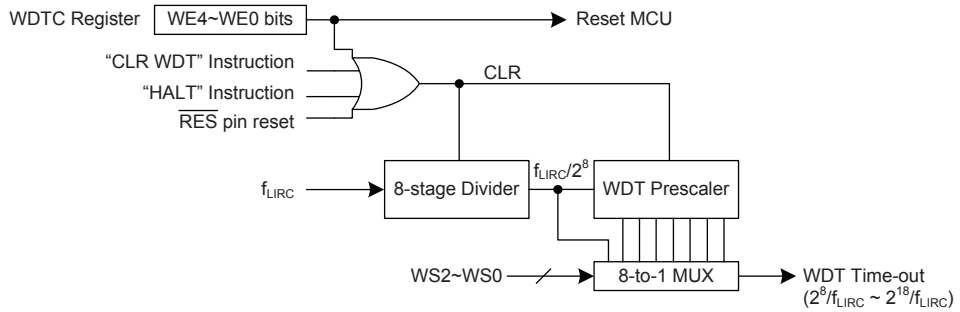
WE4~WE0 Bits	WDT Function
10101B	Disable
01010B	Enable
Any other values	Reset MCU

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. Four 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 bits, the second is using the Watchdog Timer software clear instruction, the third is via a HALT instruction and the fourth is an external hardware reset, which means a low level on the external $\overline{RES}$ pin.

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

The maximum time out period is when the 2^{18} division ratio is selected. As an example, with a 32kHz LIRC oscillator as its source clock, this will give a maximum watchdog period of around 8 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 these devices can be set to some predetermined condition irrespective of outside parameters. The most important reset condition is after power is first applied to the microcontroller. In this case, internal circuitry will ensure that the microcontroller, after a short delay, will be in a well-defined state and ready to execute the first program instruction. After this power-on reset, certain important internal registers will be set to defined states before the program commences. One of these registers is the Program Counter, which will be reset to zero forcing the 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 these devices are running. One example of this is where after power has been applied and these devices are already running, the $\overline{\text{RES}}$ line is forcefully pulled low. In such a case, known as a normal operation reset, some of the registers remain unchanged allowing these devices to proceed with normal operation after the reset line is allowed to return high.

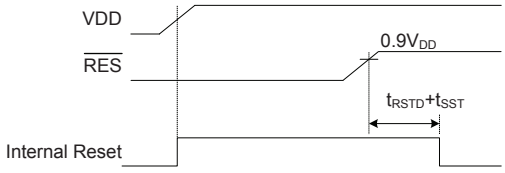
Another type of reset is when the Watchdog Timer overflows and resets the microcontroller. All types of reset operations result in different register conditions being setup. Another reset exists in the form of a Low Voltage Reset, LVR, where a full reset, similar to the $\overline{\text{RES}}$ reset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

Reset Functions

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



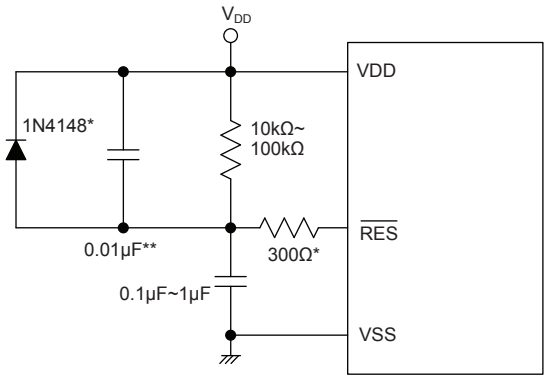
Power-on Reset Timing Chart

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

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.



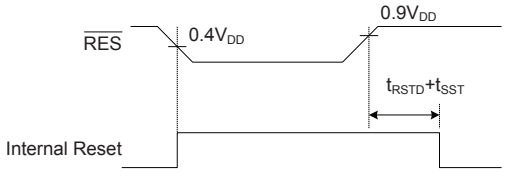
Note: * It is recommended that this component is added for added ESD protection.

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

External $\overline{\text{RES}}$ 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.



$\overline{\text{RES}}$ Reset Timing Chart

- **RSTC Register**

Bit	7	6	5	4	3	2	1	0
Name	RSTC7	RSTC6	RSTC5	RSTC4	RSTC3	RSTC2	RSTC1	RSTC0
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 **RSTC7~RSTC0**: PC2/$\overline{\text{RES}}$ selection
01010101: configured as PC2 or other pin-shared functions
10101010: configured as $\overline{\text{RES}}$ pin
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 t_{SRESET} time and the RSTF bit in the RSTFC register will be set to 1.
All resets will reset this register to POR value except the WDT time out hardware warm reset. Note that if the register is set to 10101010 to select the $\overline{\text{RES}}$ pin, this configuration has higher priority than other related pin-shared controls.

- **RSTFC Register**

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

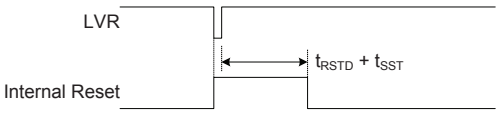
"x": unknown

Bit 7~4 Unimplemented, read as "0"
Bit 3 **RSTF**: Reset control register software reset flag
0: Not occurred
1: Occurred
This bit is set to 1 by the RSTC control register software reset and cleared by the application program. Note that this bit can only be cleared to 0 by the application program.
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
Described elsewhere.

Low Voltage Reset – LVR

The microcontrollers contain a low voltage reset circuit in order to monitor the supply voltage of these devices and provide an MCU reset should the value fall below a certain predefined level.

The LVR function is always enabled with a specific LVR voltage V_{LVR}. If the supply voltage of these devices drop to within a range of 0.9V~V_{LVR} such as might occur when changing the battery in battery powered applications, the LVR will automatically reset these devices internally and the LVRF bit in the RSTFC 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 LVD & LVR Electrical 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 are changed to some different values by environmental noise, the LVR will reset these devices after t_{SRESET} time. When this happens, the LRF bit in the RSTFC 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 these devices enter the power down mode.



Low Voltage Reset Timing Chart

- **LVRC Register**

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

Bit 7~0 **LVS7~LVS0**: LVR Voltage Select control
01010101: 2.1V
00110011: 2.55V
10011001: 3.15V
10101010: 3.8V
Any other value: 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. In this situation the register contents will remain the same after such a reset occurs.

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

- **RSTFC Register**

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

"x": unknown

Bit 7~4 Unimplemented, read as "0"

Bit 3 **RSTF**: Reset control register software reset flag
Describe elsewhere.

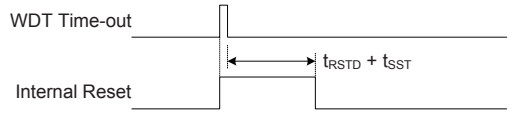
Bit 2 **LVRF**: LVR function reset flag
0: Not occur
1: Occurred
This bit is set to 1 when a specific Low Voltage Reset situation condition occurs. This bit can only be cleared to 0 by the application program.

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

Bit 0 **WRF**: WDT control register software reset flag
Describe elsewhere.

Watchdog Time-out Reset during Normal Operation

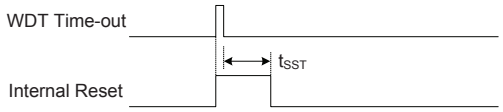
The Watchdog time-out Reset during normal operation in the FAST or SLOW mode is the same as a hardware $\overline{RES}$ pin reset except that the Watchdog time-out flag TO will be set to "1".



WDT Time-out Reset during Normal Operation Timing Chart

Watchdog Time-out Reset during SLEEP or IDLE Mode

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



WDT Time-out Reset during SLEEP or IDLE Timing Chart

Reset Initial Conditions

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

TO	PDF	RESET Conditions
0	0	Power-on reset
u	u	$\overline{RES}$ or LVR reset during FAST or SLOW Mode operation
1	u	WDT time-out reset during FAST or SLOW Mode operation
1	1	WDT time-out reset during IDLE or SLEEP Mode operation

"u" stands for unchanged

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

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

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

Register	HT66F4530	HT66F4540	HT66F4550	Power On Reset	$\overline{RES}$ Reset (Normal Operation)	$\overline{RES}$ Reset (IDLE/SLEEP)	WDT Time-out (Normal Operation)	WDT Time-out (IDLE/SLEEP)
IAR0	•	•	•	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP0	•	•	•	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
IAR1	•	•	•	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP1L	•	•	•	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP1H	•	•	•	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
ACC	•	•	•	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	•	•	•	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	•	•	•	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	•	•	•	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	•	—	—	---- -xxx	---- -uuu	---- -uuu	---- -uuu	---- -uuu
	—	•	—	---- xxxx	---- uuuu	---- uuuu	---- uuuu	---- uuuu
	—	—	•	---x xxxx	---u uuuu	---u uuuu	---u uuuu	---u uuuu
STATUS	•	•	•	xx00 xxxx	uuuu uuuu	uu01 uuuu	xx1u uuuu	uu11 uuuu
IAR2	•	•	•	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2L	•	•	•	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2H	•	•	•	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
RSTFC	•	•	•	---- 0x00	---- uuuu	---- uuuu	---- uuuu	---- uuuu
SCC	•	•	•	000- 0000	000- 0000	000- 0000	000- 0000	uuu- uuuu
HIRCC	•	•	•	---- 0001	---- 0001	---- 0001	---- 0001	---- uuuu
HXTC	•	•	•	---- -000	---- -000	---- -000	---- -000	---- -uuu
LXTC	•	•	•	---- --00	---- --00	---- --00	---- --00	---- --uu
PA	•	—	—	11-- -111	11-- -111	11-- -111	11-- -111	uu-- -uuu
	—	•	•	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	•	—	—	11-- -111	11-- -111	11-- -111	11-- -111	uu-- -uuu
	—	•	•	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAPU	•	—	—	00-- -000	00-- -000	00-- -000	00-- -000	uu-- -uuu
	—	•	•	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAWU	•	—	—	00-- -000	00-- -000	00-- -000	00-- -000	uu-- -uuu
	—	•	•	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PSCR	•	•	•	---- --00	---- --00	---- --00	---- --00	---- --uu
LVRC	•	•	•	0101 0101	0101 0101	0101 0101	0101 0101	uuuu uuuu
WDTC	•	•	•	0101 0011	0101 0011	0101 0011	0101 0011	uuuu uuuu
TB0C	•	•	•	0--- -000	0--- -000	0--- -000	0--- -000	u--- -uuu
TB1C	•	•	•	0--- -000	0--- -000	0--- -000	0--- -000	u--- -uuu
LVDC	•	•	•	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
SCOMC	—	•	•	-000 ----	-000 ----	-000 ----	-000 ----	-uuu ----
RSTC	•	•	•	01010101	01010101	01010101	01010101	uuuuuuuu
SADOL	•	•	•	xxxx ----	xxxx ----	xxxx ----	xxxx ----	uuuu ---- (ADRFS=0)
	•	•	•					uuuu uuuu (ADRFS=1)
SADOH	•	•	•	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu (ADRFS=0)
	•	•	•					---- uuuu (ADRFS=1)
SADC0	•	•	•	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	HT66F4530	HT66F4540	HT66F4550	Power On Reset	$\overline{RES}$ Reset (Normal Operation)	$\overline{RES}$ Reset (IDLE/SLEEP)	WDT Time-out (Normal Operation)	WDT Time-out (IDLE/SLEEP)
SADC1	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
VBGRC	●	●	●	---- ---0	---- ---0	---- ---0	---- ---0	---- ---u
PB	●	—	—	---1 1-11	---1 1-11	---1 1-11	---1 1-11	---u u-uu
	—	●	●	-111 1111	-111 1111	-111 1111	-111 1111	-uuu uuuu
PBC	●	—	—	---1 1-11	---1 1-11	---1 1-11	---1 1-11	---u u-uu
	—	●	●	-111 1111	-111 1111	-111 1111	-111 1111	-uuu uuuu
PBPU	●	—	—	---0 0-00	---0 0-00	---0 0-00	---0 0-00	---u u-uu
	—	●	●	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
PTM0C0	●	●	●	0000 0---	0000 0---	0000 0---	0000 0---	uuuu u---
PTM0C1	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0DL	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0DH	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
PTM0AL	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0AH	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
PTM0RPL	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0RPH	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
STM0C0	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM0C1	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM0DL	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM0DH	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
STM0AL	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM0AH	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
PTM1C0	—	●	●	0000 0---	0000 0---	0000 0---	0000 0---	uuuu u---
PTM1C1	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1DL	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1DH	—	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
PTM1AL	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1AH	—	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
PTM1RPL	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1RPH	—	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
EEC	●	●	●	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu
EEA	●	—	—	---0 0000	---0 0000	---0 0000	---0 0000	---u uuuu
	—	●	●	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
EED	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
DACC	—	—	●	---- ---0	---- ---0	---- ---0	---- ---0	---- ---u
SIMC0	●	●	●	111- 0000	111- 0000	111- 0000	111- 0000	uuu- uuuu
SIMC1	●	●	●	1000 0001	1000 0001	1000 0001	1000 0001	uuuu uuuu
SIMD	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SIMA	●	●	●	0000 000-	0000 000-	0000 000-	0000 000-	uuuu uuu-
SIMC2	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SIMTOC	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTEG	●	●	●	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu
INTC0	●	●	●	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC2	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	HT66F4530	HT66F4540	HT66F4550	Power On Reset	$\overline{RES}$ Reset (Normal Operation)	$\overline{RES}$ Reset (IDLE/SLEEP)	WDT Time-out (Normal Operation)	WDT Time-out (IDLE/SLEEP)
INTC3	●	—	—	---0 ---0	---0 ---0	---0 ---0	---0 ---0	---u ---u
	—	●	●	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
MFI0	●	—	—	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI1	●	●	—	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
	—	—	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PD	●	●	●	---- 1111	---- 1111	---- 1111	---- 1111	---- uuuu
PDC	●	●	●	---- 1111	---- 1111	---- 1111	---- 1111	---- uuuu
PDPU	●	●	●	---- 0000	--- 0000	--- 0000	--- 0000	---- uuuu
DAL	—	—	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
DAH	—	—	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PC	●	—	—	--11 -111	--11 -111	--11 -111	--11 -111	--uu -uuu
	—	●	●	-111 1111	-111 1111	-111 1111	-111 1111	-uuu uuuu
PCC	●	—	—	--11 -111	--11 -111	--11 -111	--11 -111	--uu -uuu
	—	●	●	-111 1111	-111 1111	-111 1111	-111 1111	-uuu uuuu
PCPU	●	—	—	--11 -111	--11 -111	--11 -111	--11 -111	--uu -uuu
	—	●	●	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
STM1C0	—	—	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM1C1	—	—	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM1DL	—	—	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM1DH	—	—	●	---- --00	---- --00	---- --00	---- --00	---- --uu
STM1AL	—	—	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM1AH	—	—	●	---- --00	---- --00	---- --00	---- --00	---- --uu
PAS0	●	—	—	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS1	●	—	—	0000 ----	0000 ----	0000 ----	0000 ----	uuuu ----
	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBS0	●	—	—	00-- 0000	00-- 0000	00-- 0000	00-- 0000	uu-- uuuu
	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBS1	●	—	—	---- --00	---- --00	------00	---- --00	---- --uu
	—	●	●	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
PCS0	●	—	—	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCS1	●	—	—	---- 0000	---- 0000	----0000	---- 0000	---- uuuu
	—	●	●	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
PDS0	●	—	—	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SKLEDC	●	●	●	---- --00	---- --00	------00	---- --00	---- --uu
SDSW0	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SDSW1	●	●	●	---0 0000	---0 0000	---0 0000	---0 0000	---u uuuu
SDDAC0C	●	●	●	0--- ----	0--- ----	0--- ----	0--- ----	u--- ----
SDDA0L	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SDDAC1C	●	●	●	0--- ----	0--- ----	0--- ----	0--- ----	u--- ----
SDDA1L	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	HT66F4530	HT66F4540	HT66F4550	Power On Reset	$\overline{RES}$ Reset (Normal Operation)	$\overline{RES}$ Reset (IDLE/SLEEP)	WDT Time-out (Normal Operation)	WDT Time-out (IDLE/SLEEP)
SDDAC2C	●	●	●	0--- ----	0--- ----	0--- ----	0--- ----	u--- ----
SDDA2L	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SDPGAC	●	●	●	00-- -000	00-- -000	00-- -000	00-- -000	uu-- -uuu
SDA0C	●	●	●	-00- --00	-00- --00	-00- --00	-00- --00	-uu- --uu
SDA0VOS	●	●	●	0010 0000	0010 0000	0010 0000	0010 0000	uuuu uuuu
SDA1C	●	●	●	-00- --00	-00- --00	-00- --00	-00- --00	-uu---uu
SDA1VOS	●	●	●	0010 0000	0010 0000	0010 0000	0010 0000	uuuu uuuu
SDC0C	●	●	●	000- 0000	000- 0000	000- 0000	000- 0000	uuu- uuuu
SDC0VOS	●	●	●	-001 0000	-001 0000	-001 0000	-001 0000	-uuu uuuu
SDC1C	●	●	●	000- 0000	000- 0000	000- 0000	000- 0000	uuu- uuuu
SDC1VOS	●	●	●	-001 0000	-001 0000	-001 0000	-001 0000	-uuu uuuu
SDCHYC	●	●	●	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu
USR	—	●	●	0000 1011	0000 1011	0000 1011	0000 1011	uuuu uuuu
UCR1	—	●	●	0000 00x0	0000 00x0	0000 00x0	0000 00x0	uuuu uuuu
UCR2	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
TXR_RXR	—	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
BRG	—	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu

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

Input/Output Ports

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

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

Register Name	Bit							
	7	6	5	4	3	2	1	0
PA	PA7	PA6	—	—	—	PA2	PA1	PA0
PAC	PAC7	PAC6	—	—	—	PAC2	PAC1	PAC0
PAPU	PAPU7	PAPU6	—	—	—	PAPU2	PAPU1	PAPU0
PAWU	PAWU7	PAWU6	—	—	—	PAWU2	PAWU1	PAWU0
PB	—	—	—	PB4	PB3	—	PB1	PB0
PBC	—	—	—	PBC4	PBC3	—	PBC1	PBC0
PBPU	—	—	—	PBPU4	PBPU3	—	PBPU1	PBPU0
PC	—	—	PC5	PC4	—	PC2	PC1	PC0
PCC	—	—	PCC5	PCC4	—	PCC2	PCC1	PCC0
PCPU	—	—	PCPU5	PCPU4	—	PCPU2	PCPU1	PCPU0
PD	—	—	—	—	PD3	PD2	PD1	PD0
PDC	—	—	—	—	PDC3	PDC2	PDC1	PDC0
PDPU	—	—	—	—	PDPU3	PDPU2	PDPU1	PDPU0

I/O Logic Function Registers List – HT66F4530

Register Name	Bit							
	7	6	5	4	3	2	1	0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PB	—	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PBC	—	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PBPU	—	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PC	—	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	—	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PCPU	—	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PD	—	—	—	—	PD3	PD2	PD1	PD0
PDC	—	—	—	—	PDC3	PDC2	PDC1	PDC0
PDPU	—	—	—	—	PDPU3	PDPU2	PDPU1	PDPU0

I/O Logic Function Registers List – HT66F4540/HT66F4550

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~PDPU, and are implemented using weak PMOS transistors.

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

PxPU Register

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

PxPUn: I/O Port x Pin pull-high function control

0: Disable

1: Enable

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

Port A Wake-up

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

Note that the wake-up function can be controlled by the wake-up control registers only when the pin-shared functional pin is selected as general purpose input/output and the MCU enters the Power down mode.

PAWU Register

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

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

0: Disable

1: Enable

I/O Port Control Registers

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

PxC Register

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

PxCn: I/O Port x Pin type selection

0: Output

1: Input

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

I/O Port Sink Current Control

These devices support different sink current driving capability only for PB0 and PB1 pins. With the corresponding selection register, SKLEDC, these pins can support four levels of the sink current driving capability. Users should refer to the Input/Output Characteristics section to select the desired sink current for different applications.

SKLEDC Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	SKLEDC1	SKLEDC0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **SKLEDC1~SKLEDC0**: PB1~PB0 sink current selection

00: Sink current=Level 0 (Min.)

01: Sink current=Level 1

10: Sink current=Level 2

11: Sink current=Level 3 (Max.)

Note: Users should refer to the Input/Output Characteristics section to obtain the exact value for different applications.

Pin-shared Functions

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

Pin-shared Function Selection Registers

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

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

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAS0	—	—	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	PAS17	PAS16	PAS15	PAS14	—	—	—	—
PBS0	PBS07	PBS06	—	—	PBS03	PBS02	PBS01	PBS00
PBS1	—	—	—	—	—	—	PBS11	PBS10
PCS0	—	—	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
PCS1	—	—	—	—	PCS13	PCS12	PCS11	PCS10
PDS0	PDS07	PDS06	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
IFS	IFS7	IFS6	IFS5	IFS4	IFS3	IFS2	IFS1	IFS0

Pin-shared Function Selection Registers List – HT66F4530

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	PAS17	PAS16	PAS15	PAS14	PAS13	PAS12	PAS11	PAS10
PBS0	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
PBS1	—	—	PBS15	PBS14	PBS13	PBS12	PBS11	PBS10
PCS0	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
PCS1	—	—	PCS15	PCS14	PCS13	PCS12	PCS11	PCS10
PDS0	PDS07	PDS06	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
IFS	IFS7	IFS6	IFS5	IFS4	IFS3	IFS2	IFS1	IFS0

Pin-shared Function Selection Registers List – HT66F4540/HT66F4550

- **PAS0 Register – HT66F4530**

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

Bit 7~6 Unimplemented, read as "0"

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

Bit 3~2 **PAS03~PAS02**: PA1 Pin-Shared function selection
00: PA1/STP0I
01: SDI/SDA
10: VREF
11: PA1/STP0I

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

- **PAS0 Register – HT66F4540**

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

Bit 7~6 **PAS07~PAS06**: PA3 Pin-Shared function selection
00: PA3
01: SCOM0
10: $\overline{SCS}$
11: PA3

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

Bit 3~2 **PAS03~PAS02**: PA1 Pin-Shared function selection
00: PA1/STP0I
01: SDI/SDA
10: PA1/STP0I
11: PA1/STP0I

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

- **PAS0 Register – HT66F4550**

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

Bit 7~6 **PAS07~PAS06**: PA3 Pin-Shared function selection
00: PA3
01: SCOM0
10: $\overline{SCS}$
11: DACO

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

Bit 3~2 **PAS03~PAS02**: PA1 Pin-Shared function selection
00: PA1/STP0I
01: SDI/SDA
10: PA1/STP0I
11: PA1/STP0I

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

- **PAS1 Register – HT66F4530**

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

Bit 7~6 **PAS17~PAS16**: PA7 Pin-Shared function selection
00: PA7
01: STP0
10: AN2
11: A1NI

Bit 5~4 **PAS15~PAS14**: PA6 Pin-Shared function selection
00: PA6
01: AN0
10: A1O
11: PA6

Bit 3~0 Unimplemented, read as "0"

- **PAS1 Register – HT66F4540/HT66F4550**

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

Bit 7~6 **PAS17~PAS16**: PA7 Pin-Shared function selection
00: PA7
01: STP0
10: AN2
11: A1NI

Bit 5~4 **PAS15~PAS14**: PA6 Pin-Shared function selection
00: PA6/PTCK1
01: AN0
10: A1O
11: PA6/PTCK1

Bit 3~2 **PAS13~PAS12**: PA5 Pin-Shared function selection
00: PA5/STCK0
01: RX
10: AN5
11: VREF

Bit 1~0 **PAS11~PAS10**: PA4 Pin-Shared function selection
00: PA4/INT0
01: SCK/SCL
10: VREF
11: SCOM3

- **PBS0 Register – HT66F4530**

Bit	7	6	5	4	3	2	1	0
Name	PBS07	PBS06	—	—	PBS03	PBS02	PBS01	PBS00
R/W	R/W	R/W	—	—	R/W	R/W	R/W	R/W
POR	0	0	—	—	0	0	0	0

Bit 7~6 **PBS07~PBS06**: PB3 Pin-Shared function selection
00: PB3
01: AN1
10: A1PI
11: PB3

Bit 5~4 Unimplemented, read as "0"

Bit 3~2 **PBS03~PBS02**: PB1 Pin-Shared function selection
00: PB1/INT1
01: AN4
10: PB1/INT1
11: XT2

Bit 1~0 **PBS01~PBS00**: PB0 Pin-Shared function selection
00: PB0/INT0
01: PTP0
10: AN3
11: XT1

- **PBS0 Register – HT66F4540/HT66F4550**

Bit	7	6	5	4	3	2	1	0
Name	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
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 **PBS07~PBS06**: PB3 Pin-Shared function selection
00: PB3
01: AN1
10: A1PI
11: PB3

Bit 5~4 **PBS05~PBS04**: PB2 Pin-Shared function selection
00: PB2/PTCK0
01: AN6
10: TX
11: PB2/PTCK0

Bit 3~2 **PBS03~PBS02**: PB1 Pin-Shared function selection
00: PB1/INT1
01: AN4
10: PB1/INT1
11: XT2

Bit 1~0 **PBS01~PBS00**: PB0 Pin-Shared function selection
00: PB0
01: PTP0
10: AN3
11: XT1

- **PBS1 Register – HT66F4530**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PBS11	PBS10
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **PBS11~PBS10**: PB4 Pin-Shared function selection
00: PB4
01: CLO (System Clock Output): When System Clock is disable, CLO is forced to high)
10: A0O
11: PB4

- **PBS1 Register – HT66F4540/HT66F4550**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PBS15	PBS14	PBS13	PBS12	PBS11	PBS10
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5~4 **PBS15~PBS14**: PB6 Pin-Shared function selection
00: PB6
01: SCOM1
10: SDI/SDA
11: RX

Bit 3~2 **PBS13~PBS12**: PB5 Pin-Shared function selection
00: PB5
01: SCOM2
10: TX
11: AN7

Bit 1~0 **PBS11~PBS10**: PB4 Pin-Shared function selection
00: PB4
01: CLO (System Clock Output): When System Clock is disable, CLO is forced to high)
10: A0O
11: PB4

- **PCS0 Register – HT66F4530**

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

Bit 7~6 Unimplemented, read as "0"

Bit 5~4 **PCS05~PCS04**: PC2 Pin-Shared function selection
00: PC2/$\overline{\text{RES}}$/PTCK0
01: SCK/SCL
10: PC2/$\overline{\text{RES}}$/PTCK0
11: PC2/$\overline{\text{RES}}$/PTCK0

Bit 3~2 **PCS03~PCS02**: PC1 Pin-Shared function selection
00: PC1
01: SDO
10: OSC2
11: PC1

Bit 1~0 **PCS01~PCS00**: PC0 Pin-Shared function selection
00: PC0
01: $\overline{\text{SCS}}$
10: OSC1
11: PC0

- **PCS0 Register – HT66F4540**

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

Bit 7~6 **PCS07~PCS06**: PC3 Pin-Shared function selection
00: PC3
01: PC3
10: SDO
11: C1XO

Bit 5~4 **PCS05~PCS04**: PC2 Pin-Shared function selection
00: PC2/$\overline{\text{RES}}$
01: PTP1
10: PC2/$\overline{\text{RES}}$
11: PC2/$\overline{\text{RES}}$

Bit 3~2 **PCS03~PCS02**: PC1 Pin-Shared function selection
00: PC1
01: SDO
10: OSC2
11: PC1

Bit 1~0 **PCS01~PCS00**: PC0 Pin-Shared function selection
00: PC0
01: $\overline{SCS}$
10: OSC1
11: PC0

- **PCS0 Register – HT66F4550**

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

Bit 7~6 **PCS07~PCS06**: PC3 Pin-Shared function selection
00: PC3
01: PC3
10: SDO
11: C1XO

Bit 5~4 **PCS05~PCS04**: PC2 Pin-Shared function selection
00: PC2/$\overline{RES}$/STP1I
01: PTP1
10: PC2/$\overline{RES}$/STP1I
11: PC2/$\overline{RES}$/STP1I

Bit 3~2 **PCS03~PCS02**: PC1 Pin-Shared function selection
00: PC1/STCK1
01: SDO
10: OSC2
11: PC1/STCK1

Bit 1~0 **PCS01~PCS00**: PC0 Pin-Shared function selection
00: PC0
01: $\overline{SCS}$
10: OSC1
11: STP1

- **PCS1 Register – HT66F4530**

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

Bit 7~4 Unimplemented, read as "0"

Bit 3~2 **PCS13~PCS12**: PC5 Pin-Shared function selection
00: PC5
01: C1PI
10: PC5
11: PC5

Bit 1~0 **PCS11~PCS10**: PC4 Pin-Shared function selection
00: PC4
01: C1NI
10: PC4
11: PC4

- **PCS1 Register – HT66F4540/HT66F4550**

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

Bit 7~6 Unimplemented, read as "0"

Bit 5~4 **PCS15~PCS14**: PC6 Pin-Shared function selection
00: PC6
01: C0NI
10: PC6
11: PC6

Bit 3~2 **PCS13~PCS12**: PC5 Pin-Shared function selection
00: PC5
01: C1PI
10: PC5
11: PC5

Bit 1~0 **PCS11~PCS10**: PC4 Pin-Shared function selection
00: PC4
01: C1NI
10: PC4
11: PC4

- **PDS0 Register**

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

Bit 7~6 **PDS07~PDS06**: PD3 Pin-Shared function selection
00: PD3
01: A0PI
10: PD3
11: PD3

Bit 5~4 **PDS05~PDS04**: PD2 Pin-Shared function selection
00: PD2
01: A0NI
10: PD2
11: PD2

Bit 3~2 **PDS03~PDS02**: PD1 Pin-Shared function selection
00: PD1
01: C0PI
10: PD1
11: PD1

Bit 1~0 **PDS01~PDS00**: PD0 Pin-Shared function selection
00: PD0
01: C0XO
10: SCK/SCL
11: PD0

- **IFS Register – HT66F4530**

Bit	7	6	5	4	3	2	1	0
Name	IFS7	IFS6	IFS5	IFS4	IFS3	IFS2	IFS1	IFS0
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 **IFS7~IFS6**: SDI/SDA input source pin selection
00: PA1
01: PA1
10: PA1
11: PA1

Bit 5~4 **IFS5~IFS4**: $\overline{SCS}$ input source pin selection
00: PC0
01: PC0
10: PC0
11: PC0

Bit 3~2 **IFS3~IFS2**: SCK/SCL input source pin selection
00: PD0
01: PD0
10: PC2
11: PC2

Bit 1~0 **IFS1~IFS0**: INT0 input source pin selection
00: PB0
01: PB0
10: PC5
11: PC5

Note: In the SPI Master mode, the PC0 pin can be used as the $\overline{SCS}$ pin after SIMEN=1 and CSEN=1 and is independent of the IFS[5:4] bits, while the PC2 and PD0 pins each can be used as the SCK pin after SIMEN=1 and are independent of the IFS[3:2] bits.

- **IFS Register – HT66F4540/HT66F4550**

Bit	7	6	5	4	3	2	1	0
Name	IFS7	IFS6	IFS5	IFS4	IFS3	IFS2	IFS1	IFS0
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 **IFS7~IFS6**: SDI/SDA input source pin selection
00: PA1
01: PA1
10: PB6
11: PB6

Bit 5~4 **IFS5~IFS4**: $\overline{SCS}$ input source pin selection
00: PA3
01: PA3
10: PC0
11: PC0

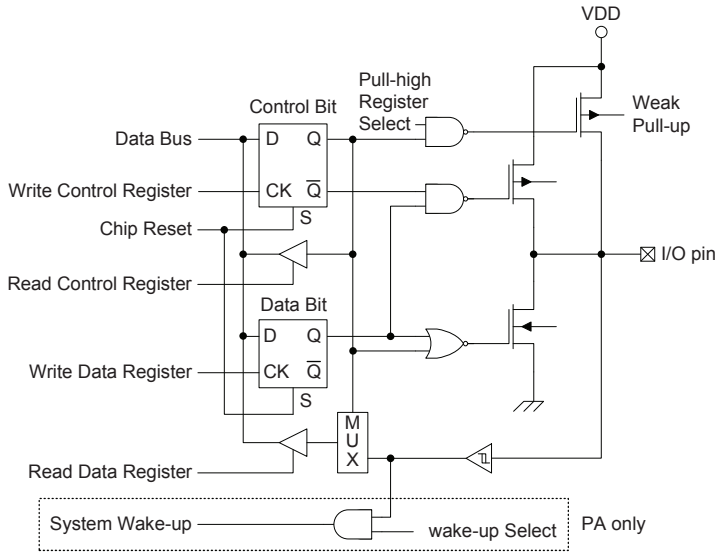
Bit 3~2 **IFS3~IFS2**: SCK/SCL input source pin selection
00: PD0
01: PD0
10: PA4
11: PA4

Bit 1~0 **IFS1~IFS0**: RX input source pin selection
00: PB6
01: PA0
10: PA5
11: PB6

Note: In the SPI Master mode, the PA3 and PC0 pins each can be used as the $\overline{SCS}$ pin after SIMEN=1 and CSEN=1 and are independent of the IFS[5:4] bits, while the PA4 and PD0 pins each can be used as the SCK pin after SIMEN=1 and are independent of the IFS[3:2] bits.

I/O Pin Structures

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



Logic Function Input/Output Structure

Programming Considerations

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

Port A has the additional capability of providing wake-up functions. When these devices are in the SLEEP or IDLE Mode, various methods are available to wake these devices 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 these 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 two individual interrupts. The addition of input and output pins for each TM ensures that users are provided with timing units with a wide and flexible range of features.

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

Introduction

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

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

TM Function Summary

Device	STM	PTM
HT66F4530	10-bit STM0	10-bit PTM0
HT66F4540	10-bit STM0	10-bit PTM0 10-bit PTM1
HT66F4550	10-bit STM0 10-bit STM1	10-bit PTM0 10-bit PTM1

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 xTnCK2~xTnCK0 bits in the xTMn control registers, where "x" stands for S or P type TM and "n" stands for the specific TM serial number. The clock source can be a ratio of the system clock f_{SYS} or the internal high clock f_H, the f_{SUB} clock source or the external xTCKn pin. The xTCKn pin clock source is used to allow an external signal to drive the TM as an external clock source or for event counting.

TM Interrupts

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

TM External Pins

Each of the TMs, irrespective of what type, has one or two TM input pins, with the label xTCKn and xTPnI respectively. The xTMn input pin, xTCKn, is essentially a clock source for the xTMn and is selected using the xTnCK2~xTnCK0 bits in the xTMnC0 register. This external TM input pin allows an external clock source to drive the internal TM. The xTCKn input pin can be chosen to have either a rising or falling active edge. The xTCKn pin is also used as the external trigger input pin in single pulse output mode.

The other xTMn input pin, xTPnI, 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 xTnIO1~xTnIO0 bits in the xTMnC1 register. There is another capture input, PTCKn, for PTMn capture input mode, which can be used as the external trigger input source.

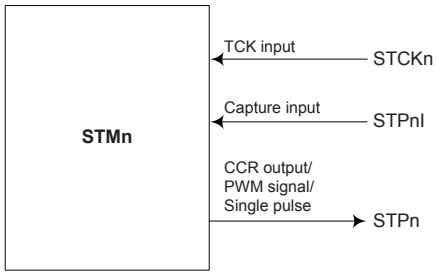
The TMs each have one output pin with the label xTPn. The TM output pins can be selected using the corresponding pin-shared function selection bits described in the Pin-shared Function section. When the TM is in the Compare Match Output Mode, these pins can be controlled by the TM to switch to a high or low level or to toggle when a compare match situation occurs. The external xTPn output pin is also the pin where the TM generates the PWM output waveform. As the TM output pins are pin-shared with other functions, the TM output function must first be setup using relevant pin-shared function selection register.

Device	STM		PTM	
	Input	Output	Input	Output
HT66F4530	STP0I	STP0	PTCK0	PTP0
HT66F4540	STCK0, STP0I	STP0	PTCK0, PTCK1	PTP0, PTP1
HT66F4550	STCK0, STP0I; STCK1, STP1I	STP0, STP1	PTCK0, PTCK1	PTP0, PTP1

TM External Pins

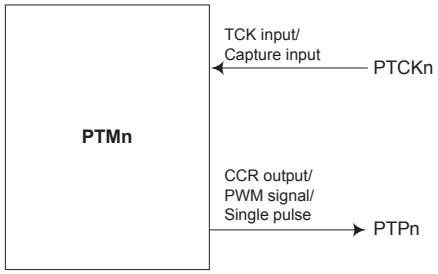
TM Input/Output Pin Selection

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



Note: STCKn is not available for HT66F4530.

STM Function Pin Control Block Diagram – n=0 or 1

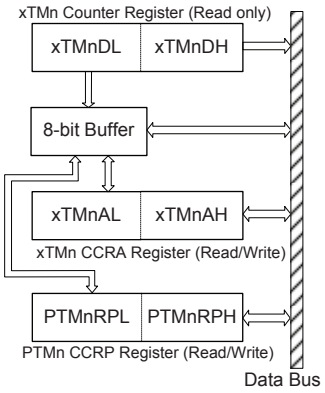


PTM Function Pin Control Block Diagram – n=0 or 1

Programming Considerations

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

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



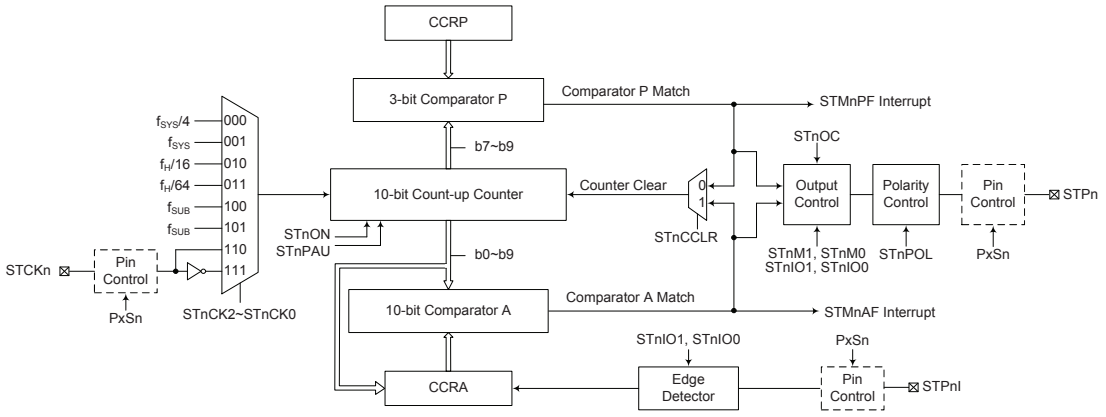
The following steps show the read and write procedures:

- Writing Data to CCRA or CCRP
  - Step 1. Write data to Low Byte xTMnAL or PTMnRPL
    - Note that here data is only written to the 8-bit buffer.
  - Step 2. Write data to High Byte xTMnAH or PTMnRPH
    - 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, CCRA or CCRP
  - Step 1. Read data from the High Byte xTMnDH, xTMnAH or PTMnRPH
    - 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 xTMnDL, xTMnAL or PTMnRPL
    - This step reads data from the 8-bit buffer.

Standard Type TM – STM

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

Device	STM Core	STM Input Pin	STM Output Pin
HT66F4530	10-bit STM (STM0)	STP0I	STP0
HT66F4540	10-bit STM (STM0)	STCK0, STP0I	STP0
HT66F4550	10-bit STM (STM0, STM1)	STCK0, STP0I; STCK1, STP1I	STP0; STP1



Note: 1. n=0 for HT66F4530/HT66F4540, n=0 or 1 for HT66F4550.
2. STCKn is not available for HT66F4530.

Standard Type TM Block Diagram

Standard TM Operation

The size of Standard TM is 10-bit wide and 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 comparator is 3-bit wide whose value is compared with the highest 8 bits in the counter while the CCRA is the 10 bits and therefore compares 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 STnON 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 STMn 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 10-bit value, while a read/write register pair exists to store the internal 10-bit CCRA value. The remaining two registers are control registers which setup the different operating and control modes as well as three CCRP bits.

Register Name	Bit							
	7	6	5	4	3	2	1	0
STMnC0	STnPAU	STnCK2	STnCK1	STnCK0	STnON	STnRP2	STnRP1	STnRP0
STMnC1	STnM1	STnM0	STnIO1	STnIO0	STnOC	STnPOL	STnDPX	STnCCLR
STMnDL	D7	D6	D5	D4	D3	D2	D1	D0
STMnDH	—	—	—	—	—	—	D9	D8
STMnAL	D7	D6	D5	D4	D3	D2	D1	D0
STMnAH	—	—	—	—	—	—	D9	D8

10-bit Standard TM Registers List – n=0 or 1

STMnC0 Register

Bit	7	6	5	4	3	2	1	0
Name	STnPAU	STnCK2	STnCK1	STnCK0	STnON	STnRP2	STnRP1	STnRP0
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 **STnPAU**: STMn 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 STMn 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 **STnCK2~STnCK0**: Select STMn Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: f_{SUB}
110: STCKn rising edge clock(unavailable for HT66F4530)
111: STCKn falling edge clock (unavailable for HT66F4530)
These three bits are used to select the clock source for the STMn. 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 **STnON**: STMn Counter On/Off control
0: Off
1: On
This bit controls the overall on/off function of the STMn. Setting the bit high enables the counter to run while clearing the bit disables the STMn. Clearing this bit to zero will stop the counter from counting and turn off the STMn 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 STMn is in the Compare Match Output Mode, PWM Output Mode or Single Pulse Output Mode then the STMn output pin will be reset to its initial condition, as specified by the STnOC bit, when the STnON bit changes from low to high.

Bit 2~0 **STnRP2~STnRP0**: STMn CCRP 3-bit register, compared with the STMn counter bit 9~bit 7

Comparator P Match Period =
000: 1024 STMn clocks
001: 128 STMn clocks
010: 256 STMn clocks
011: 384 STMn clocks
100: 512 STMn clocks
101: 640 STMn clocks
110: 768 STMn clocks
111: 896 STMn 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 STnCCLR bit is set to zero. Setting the STnCCLR 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.

STMnC1 Register

Bit	7	6	5	4	3	2	1	0
Name	STnM1	STnM0	STnIO1	STnIO0	STnOC	STnPOL	STnDPX	STnCCLR
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 **STnM1~STnM0**: Select STMn Operating Mode
00: Compare Match Output Mode
01: Capture Input Mode
10: PWM Output Mode or Single Pulse Output Mode
11: Timer/Counter Mode

These bits setup the required operating mode for the STMn. To ensure reliable operation the STMn should be switched off before any changes are made to the STnM1 and STnM0 bits. In the Timer/Counter Mode, the STMn output pin control will be disabled.

Bit 5~4 **STnIO1~STnIO0**: Select STMn external pin (STPn or STPnI) function

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

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

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

Timer/Counter Mode
Unused

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

In the Compare Match Output Mode, the STnIO1 and STnIO0 bits determine how the STMn 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 STMn output pin should be setup using the STnOC bit in the STMnC1 register. Note that the output level requested by the STnIO1 and STnIO0 bits must be different from the initial value setup using the STnOC bit otherwise no change will occur on the STMn output pin when a compare match occurs. After the STMn output pin changes state, it can be reset to its initial level by changing the level of the STnON bit from low to high.

In the PWM Output Mode, the STnIO1 and STnIO0 bits determine how the STMn 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 STnIO1 and STnIO0 bits only after the STMn has been switched off. Unpredictable PWM outputs will occur if the STnIO1 and STnIO0 bits are changed when the STMn is running.

Bit 3 **STnOC**: STMn STPn Output control

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

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

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

Bit 2 **STnPOL**: STMn STPn Output polarity control
0: Non-inverted
1: Inverted

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

Bit 1 **STnDPX**: STMn PWM duty/period control
0: CCRP – period; CCRA – duty
1: CCRP – duty; CCRA – period

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

Bit 0 **STnCCLR**: STMn Counter Clear condition selection
0: Comparator P match
1: Comparator A match

This bit is used to select the method which clears the counter. Remember that the Standard TM contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the STnCCLR 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 STnCCLR bit is not used in the PWM Output, Single Pulse Output or Capture Input Mode.

STMnDL Register

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

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

STMnDH Register

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

Bit 7~2 Unimplemented, read as "0"

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

STMnAL Register

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

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

STMnAH Register

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

Bit 7~2 Unimplemented, read as "0"

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

Standard Type TM Operation Modes

The Standard Type TM can operate in one of five operating modes, Compare Match Output Mode, PWM Output Mode, Single Pulse Output Mode, Capture Input Mode or Timer/Counter Mode. The operating mode is selected using the STnM1 and STnM0 bits in the STMnC1 register.

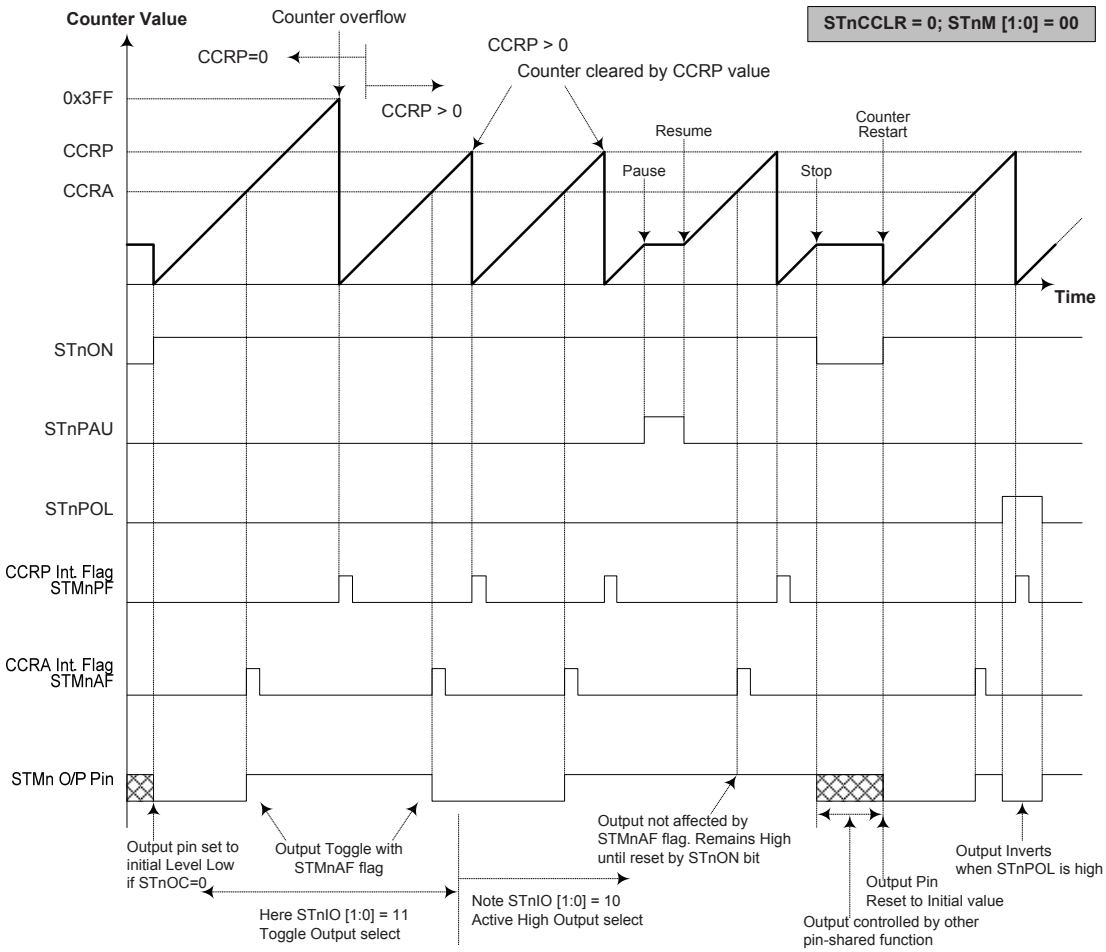
Compare Match Output Mode

To select this mode, bits STnM1 and STnM0 in the STMnC1 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 STnCCLR 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 STMnAF and STMnPF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

If the STnCCLR bit in the STMnC1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the STMnAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when STnCCLR is high no STMnPF interrupt request flag will be generated. In the Compare Match Output Mode, the CCRA cannot be set to "0".

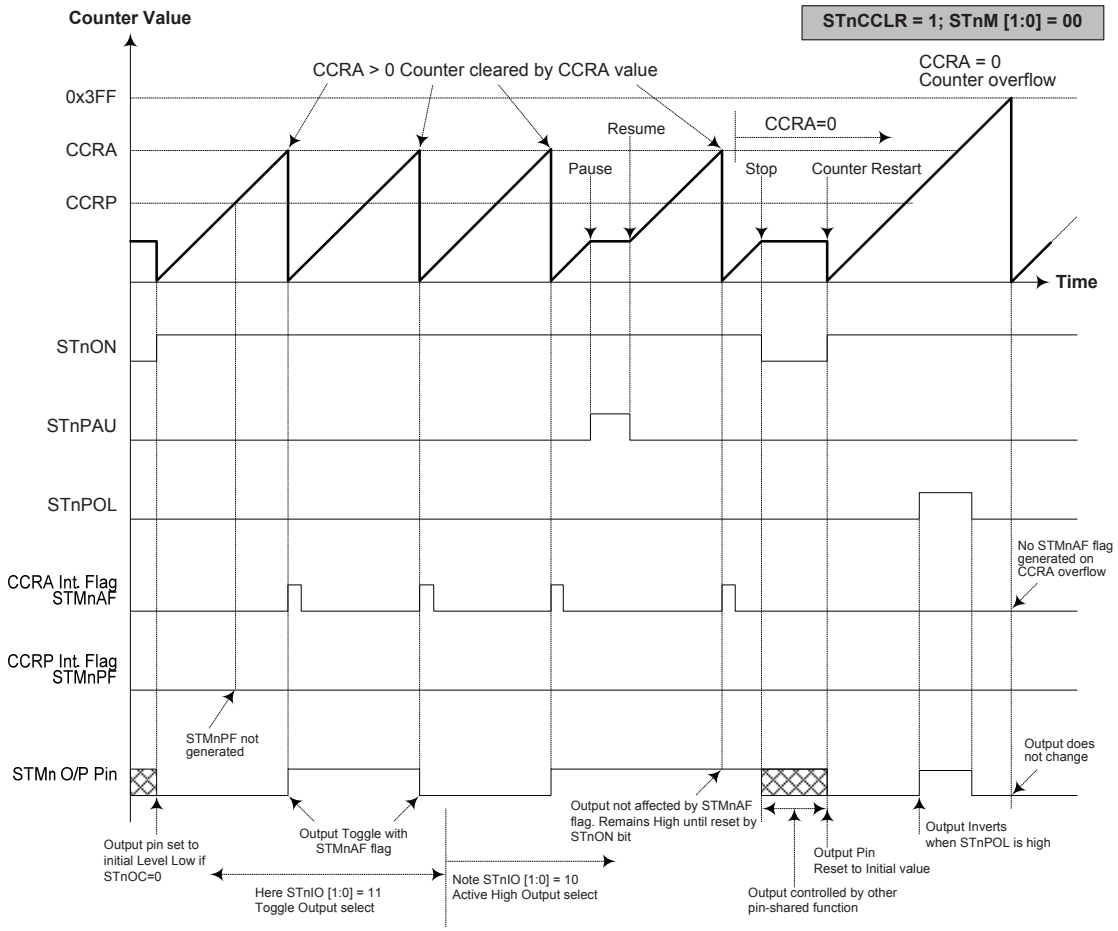
If the CCRA bits are all zero, the counter will overflow when it reaches its maximum 10-bit, 3FF Hex, value, however here the STMnAF interrupt request flag will not be generated.

As the name of the mode suggests, after a comparison is made, the STMn output pin, will change state. The STMn output pin condition however only changes state when a STMnAF interrupt request flag is generated after a compare match occurs from Comparator A. The STMnPF interrupt request flag, generated from a compare match occurs from Comparator P, will have no effect on the STMn output pin. The way in which the STMn output pin changes state are determined by the condition of the STnIO1 and STnIO0 bits in the STMnC1 register. The STMn output pin can be selected using the STnIO1 and STnIO0 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 STMn output pin, which is setup after the STnON bit changes from low to high, is setup using the STnOC bit. Note that if the STnIO1 and STnIO0 bits are zero then no pin change will take place.



Compare Match Output Mode – STnCCLR=0 (n=0 or 1)

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



Compare Match Output Mode – STnCCLR=1 (n=0 or 1)

Note: 1. With STnCCLR=1 a Comparator A match will clear the counter
2. The STMn output pin is controlled only by the STMnAF flag
3. The output pin is reset to its initial state by a STnON bit rising edge
4. A STMnPF flag is not generated when STnCCLR=1

Timer/Counter Mode

To select this mode, bits STnM1 and STnM0 in the STMnC1 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 STMn 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 STMn 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 STnM1 and STnM0 in the STMnC1 register should be set to 10 respectively and also the STnIO1 and STnIO0 bits should be set to 10 respectively. The PWM function within the STMn 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 STMn output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM Output Mode, the STnCCLR 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 STnDPX bit in the STMnC1 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 STnOC bit in the STMnC1 register is used to select the required polarity of the PWM waveform while the two STnIO1 and STnIO0 bits are used to enable the PWM output or to force the STMn output pin to a fixed high or low level. The STnPOL bit is used to reverse the polarity of the PWM output waveform.

- **10-bit STMn, PWM Output Mode, Edge-aligned Mode, STnDPX=0**

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

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

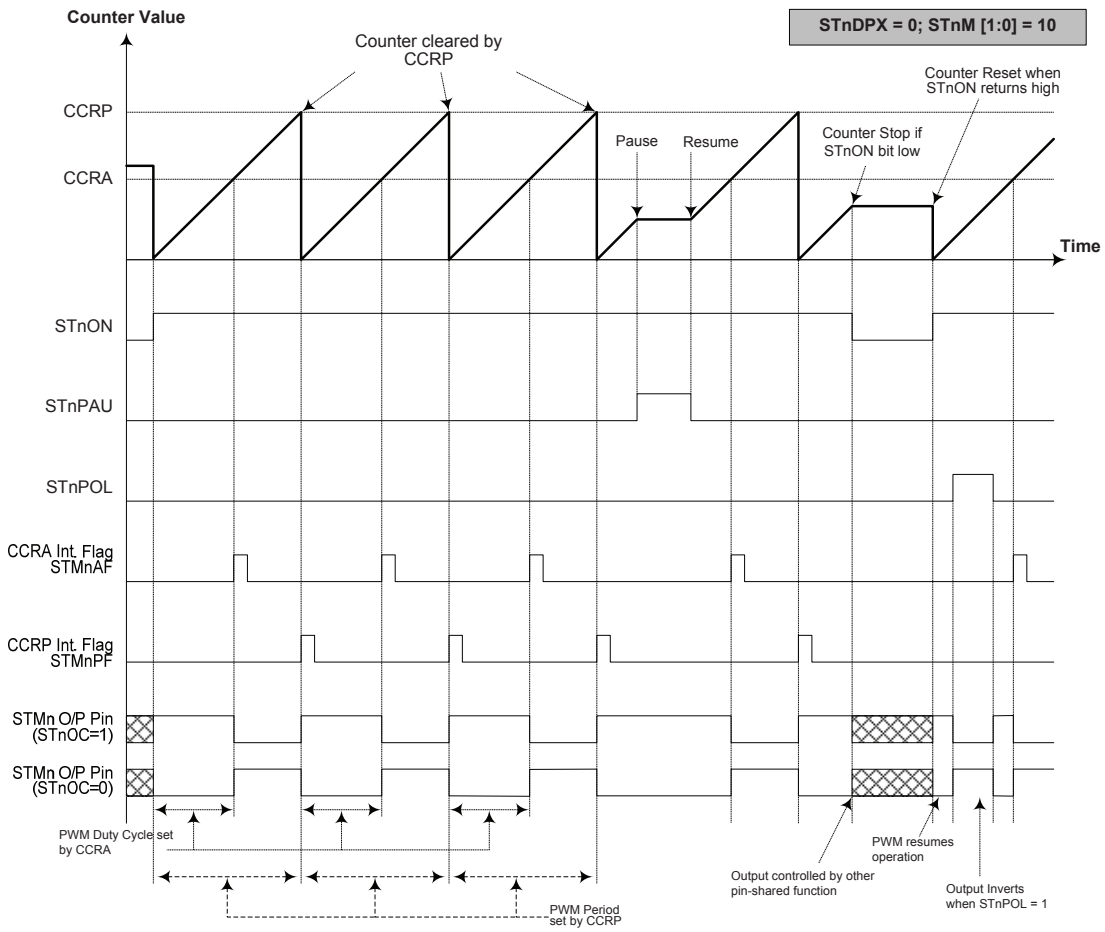
The STMn PWM output frequency=(f_{SYS}/4) / 512=f_{SYS}/2048=1.9531kHz, duty=128/512=25%.

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

- **10-bit STMn, PWM Output Mode, Edge-aligned Mode, STnDPX=1**

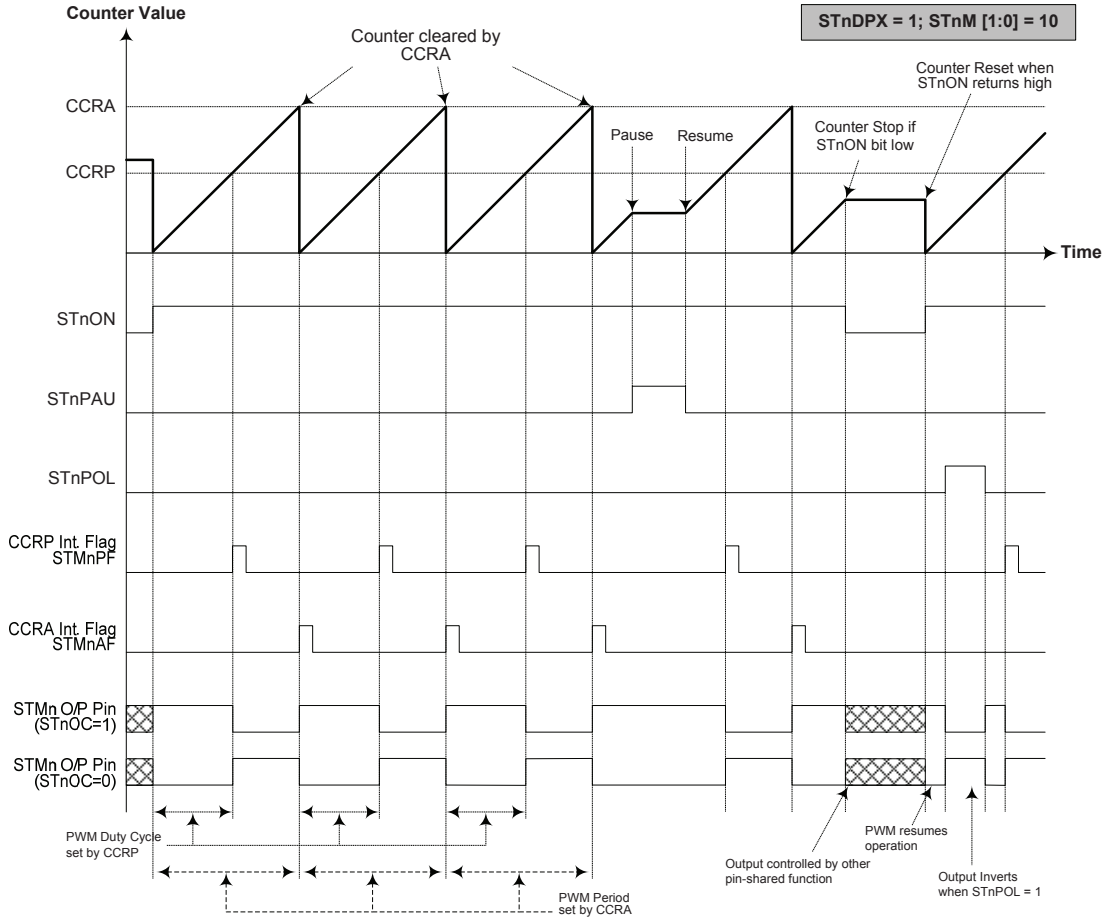
CCRP	001b	010b	011b	100b	101b	110b	111b	000b
Period	CCRA							
Duty	128	256	384	512	640	768	896	1024

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



PWM Output Mode – STnDPX=0 (n=0 or 1)

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



PWM Output Mode – STnDPX=1 (n=0 or 1)

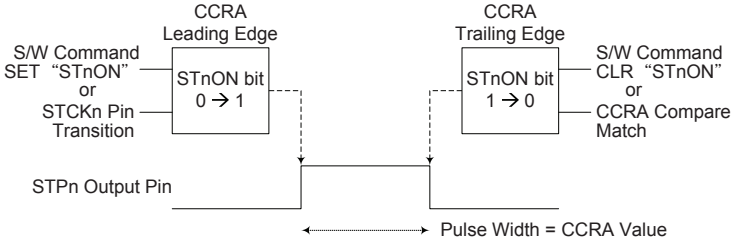
Note: 1. Here STnDPX=1 – Counter cleared by CCRA
2. A counter clear sets the PWM Period
3. The internal PWM function continues even when STnIO[1:0]=00 or 01
4. The STnCCLR bit has no influence on PWM operation

Single Pulse Output Mode

To select this mode, bits STnM1 and STnM0 in the STMnC1 register should be set to 10 respectively and also the STnIO1 and STnIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the STMn output pin.

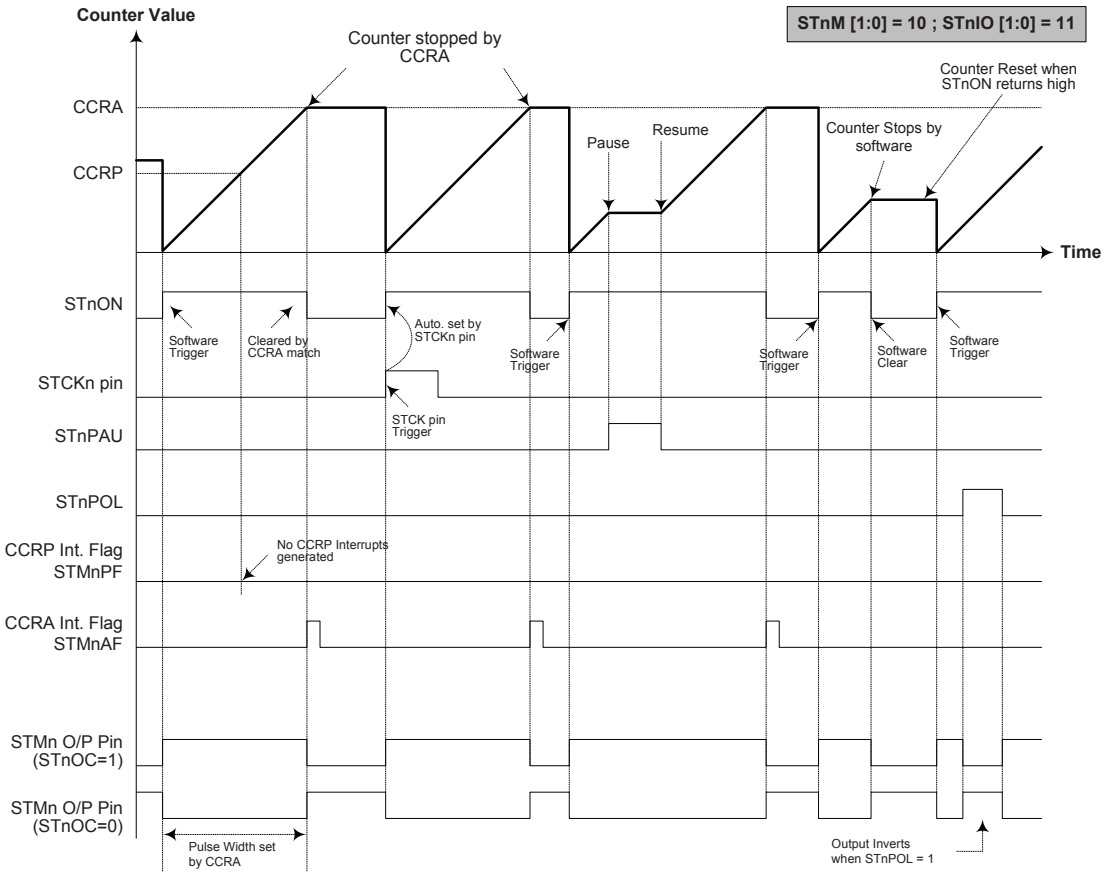
The trigger for the pulse output leading edge is a low to high transition of the STnON bit, which can be implemented using the application program. However in the Single Pulse Output Mode, the STnON bit can also be made to automatically change from low to high using the external STCKn pin, which will in turn initiate the Single Pulse output. When the STnON bit transitions to a high level, the counter will start running and the pulse leading edge will be generated. The STnON bit should remain high when the pulse is in its active state. The generated pulse trailing edge will be generated when the STnON 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 STnON 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 STMn interrupt. The counter can only be reset back to zero when the STnON bit changes from low to high when the counter restarts. In the Single Pulse Output Mode CCRP is not used. The STnCCLR and STnDPX bits are not used in this Mode.



Note: STCKn is not available for HT66F4530.

Single Pulse Generation (n=0 or 1)



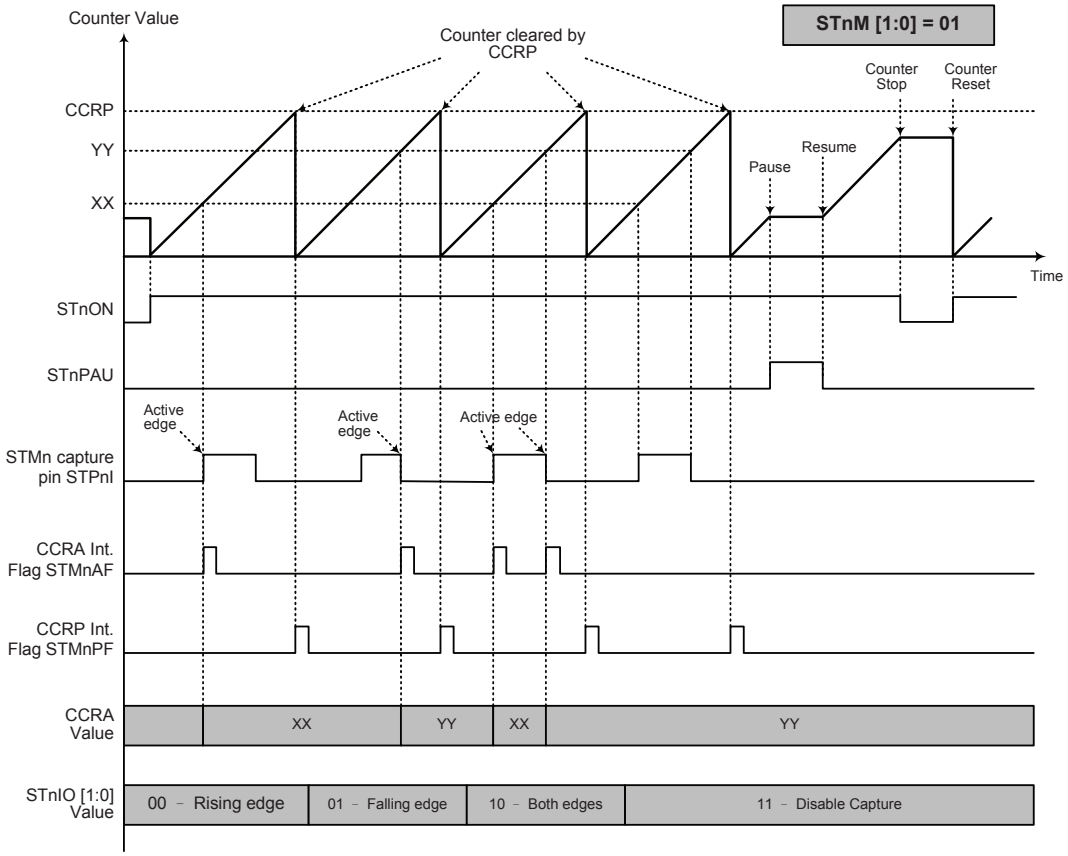
Single Pulse Output Mode (n=0 or 1)

Note: 1. Counter stopped by CCRA
2. CCRP is not used
3. The pulse triggered by the STCKn pin or by setting the STnON bit high
4. A STCKn pin active edge will automatically set the STnON bit high.
5. In the Single Pulse Output Mode, STnIO[1:0] must be set to "11" and cannot be changed.
6. STCKn is not available for HT66F4530.

Capture Input Mode

To select this mode bits STnM1 and STnM0 in the STMnC1 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 STPnI 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 STnIO1 and STnIO0 bits in the STMnC1 register. The counter is started when the STnON bit changes from low to high which is initiated using the application program.

When the required edge transition appears on the STPnI pin the present value in the counter will be latched into the CCRA registers and a STMn interrupt generated. Irrespective of what events occur on the STPnI pin the counter will continue to free run until the STnON 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 STMn 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 STnIO1 and STnIO0 bits can select the active trigger edge on the STPnI pin to be a rising edge, falling edge or both edge types. If the STnIO1 and STnIO0 bits are both set high, then no capture operation will take place irrespective of what happens on the STPnI pin, however it must be noted that the counter will continue to run. The STnCCLR and STnDPX bits are not used in this Mode.



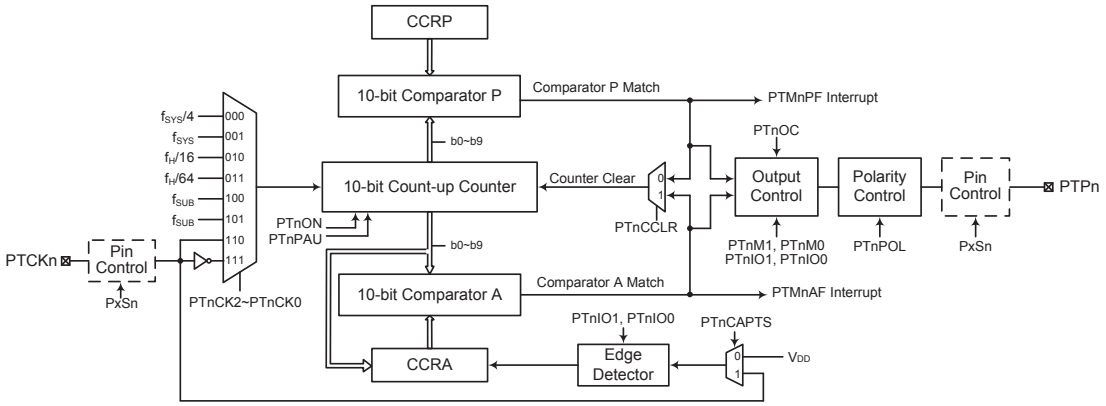
Capture Input Mode (n=0 or 1)

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

Periodic Type TM – PTM

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

Device	PTM Core	PTM Input Pin	PTM Output Pin
HT66F4530	10-bit PTM(PTM0)	PTCK0	PTP0
HT66F4540	10-bit PTM(PTM0, PTM1)	PTCK0; PTCK1	PTP0; PTP1
HT66F4550	10-bit PTM(PTM0, PTM1)	PTCK0; PTCK1	PTP0; PTP1



Note: 1. n=0 for HT66F4530, n=0 or 1 for HT66F4540/HT66F4550.
2. The Capture input can only come from the PTCKn pin for the HT66F45x0 devices.

Periodic Type TM Block Diagram

Periodic TM Operation

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

The only way of changing the value of the 10-bit counter using the application program, is to clear the counter by changing the PTnON 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 PTMn interrupt signal will also usually be generated. The Periodic Type TM can operate in a number of different operational modes, can be driven by different clock sources including an input pin and can also control more than one output pin. All operating setup conditions are selected using relevant internal registers.

Periodic Type TM Register Description

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

Register Name	Bit							
	7	6	5	4	3	2	1	0
PTMnC0	PTnPAU	PTnCK2	PTnCK1	PTnCK0	PTnON	—	—	—
PTMnC1	PTnM1	PTnM0	PTnIO1	PTnIO0	PTnOC	PTnPOL	PTnCAPTS	PTnCCLR
PTMnDL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnDH	—	—	—	—	—	—	D9	D8
PTMnAL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnAH	—	—	—	—	—	—	D9	D8
PTMnRPL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnRPH	—	—	—	—	—	—	D9	D8

10-bit Periodic TM Registers List – n=0 or 1

PTMnC0 Register

Bit	7	6	5	4	3	2	1	0
Name	PTnPAU	PTnCK2	PTnCK1	PTnCK0	PTnON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

Bit 7 **PTnPAU**: PTMn 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 PTMn 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 **PTnCK2~PTnCK0**: Select PTMn Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: f_{SUB}
110: PTCKn rising edge clock
111: PTCKn falling edge clock
These three bits are used to select the clock source for the PTMn. 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 **PTnON**: PTMn Counter On/Off Control
0: Off
1: On
This bit controls the overall on/off function of the PTMn. Setting the bit high enables the counter to run, clearing the bit disables the PTMn. Clearing this bit to zero will stop the counter from counting and turn off the PTMn 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 PTMn is in the Compare Match Output Mode, PWM Output Mode or Single Pulse Output Mode then the PTMn output pin will be reset to its initial condition, as specified by the PTnOC bit, when the PTnON bit changes from low to high.

Bit 2~0 Unimplemented, read as "0"

PTMnC1 Register

Bit	7	6	5	4	3	2	1	0
Name	PTnM1	PTnM0	PTnIO1	PTnIO0	PTnOC	PTnPOL	PTnCAPTS	PTnCCLR
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 **PTnM1~PTnM0**: Select PTMn Operating Mode
00: Compare Match Output Mode
01: Capture Input Mode
10: PWM Output Mode or Single Pulse Output Mode
11: Timer/Counter Mode

These bits setup the required operating mode for the PTMn. To ensure reliable operation the PTMn should be switched off before any changes are made to the PTnM1 and PTnM0 bits. In the Timer/Counter Mode, the PTMn output pin control must be disabled.

Bit 5~4 **PTnIO1~PTnIO0**: Select PTMn external pin (PTPn or PTCKn) function

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

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

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

Timer/Counter Mode
Unused

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

In the Compare Match Output Mode, the PTnIO1 and PTnIO0 bits determine how the PTMn output pin changes state when a compare match occurs from the Comparator A. The PTMn 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 PTMn output pin should be setup using the PTnOC bit in the PTMnC1 register. Note that the output level requested by the PTnIO1 and PTnIO0 bits must be different from the initial value setup using the PTnOC bit otherwise no change will occur on the PTMn output pin when a compare match occurs. After the PTMn output pin changes state, it can be reset to its initial level by changing the level of the PTnON bit from low to high.

In the PWM Output Mode, the PTnIO1 and PTnIO0 bits determine how the PTMn 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 PTnIO1 and PTnIO0 bits only after the TM has been switched off. Unpredictable PWM outputs will occur if the PTnIO1 and PTnIO0 bits are changed when the PTMn is running.

Bit 3 **PTnOC**: PTMn PTPn Output control bit

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

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

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

Bit 2 **PTnPOL**: PTMn PTPn Output polarity Control
0: Non-invert
1: Invert

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

Bit 1 **PTnCAPTS**: PTMn Capture Trigger Source Selection
0: Undefined
1: From PTCKn pin

Note: When the PTMn is being used in the capture input mode, this bit must be set high.

Bit 0 **PTnCCLR**: Select PTMn Counter clear condition
0: PTMn Comparator P match
1: PTMn Comparator A match

This bit is used to select the method which clears the counter. Remember that the Periodic TM contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the PTnCCLR 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 PTnCCLR bit is not used in the PWM Output Mode, Single Pulse or Capture Input Mode.

PTMnDL Register

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

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

PTMnDH Register

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

Bit 7~2 Unimplemented, read as "0"

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

PTMnAL Register

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

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

PTMnAH Register

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

Bit 7~2 Unimplemented, read as "0"

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

PTMnRPL Register

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

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

PTMnRPH Register

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

Bit 7~2 Unimplemented, read as "0"

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

Periodic Type TM Operating Modes

The Periodic Type TM can operate in one of five operating modes, Compare Match Output Mode, PWM Output Mode, Single Pulse Output Mode, Capture Input Mode or Timer/Counter Mode. The operating mode is selected using the PTnM1 and PTnM0 bits in the PTMnC1 register.

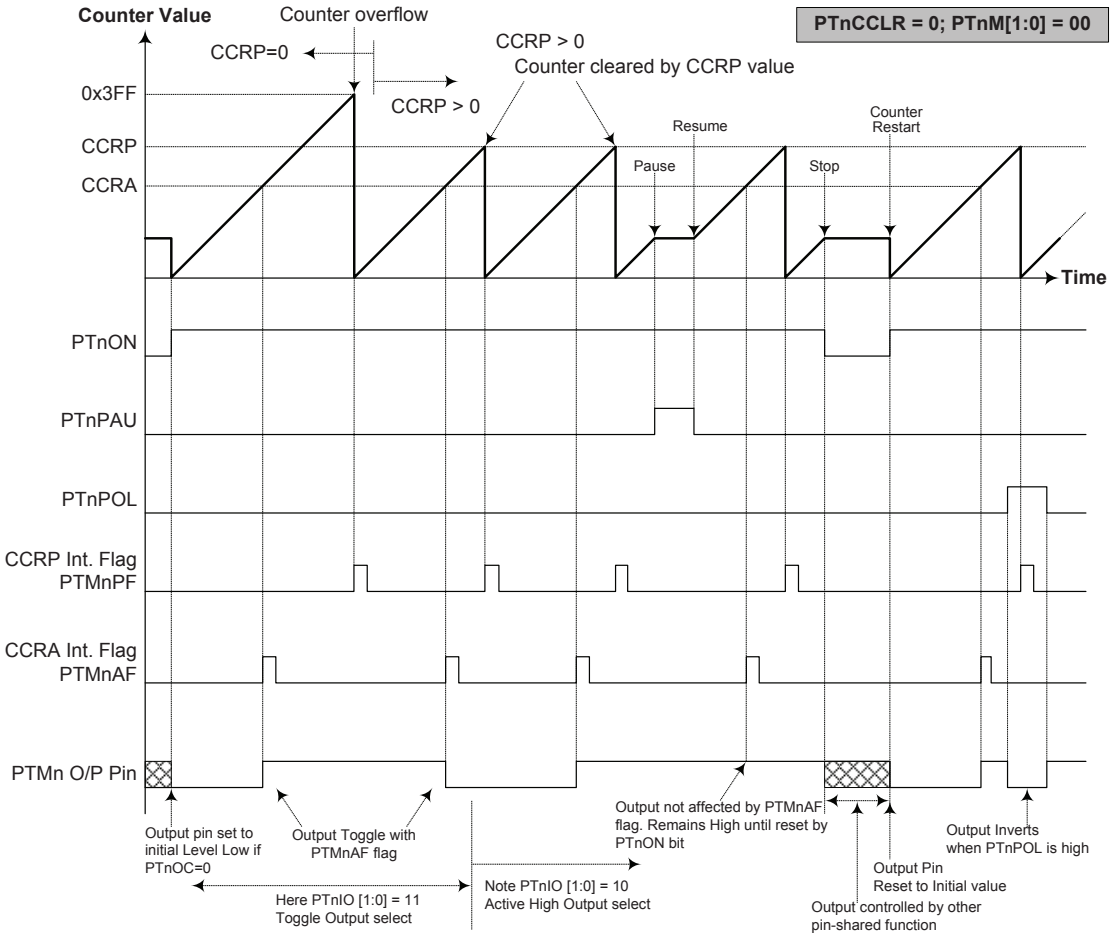
Compare Match Output Mode

To select this mode, bits PTnM1 and PTnM0 in the PTMnC1 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 PTnCCLR 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 PTMnAF and PTMnPF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

If the PTnCCLR bit in the PTMnC1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the PTMnAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when PTnCCLR is high no PTMnPF interrupt request flag will be generated. In the Compare Match Output Mode, the CCRA can not be cleared to zero.

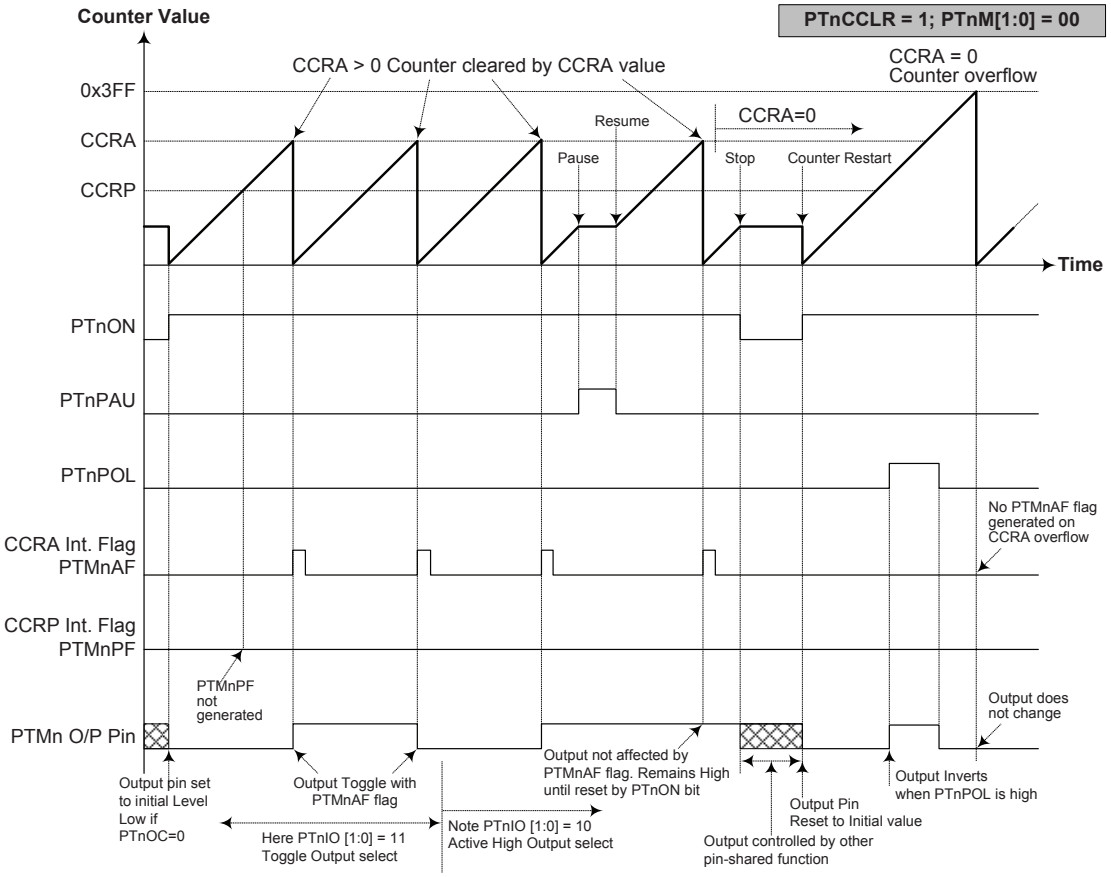
If the CCRA bits are all zero, the counter will overflow when its reaches its maximum 10-bit, 3FF Hex, value, however here the PTMnAF interrupt request flag will not be generated.

As the name of the mode suggests, after a comparison is made, the PTMn output pin, will change state. The PTMn output pin condition however only changes state when a PTMnAF interrupt request flag is generated after a compare match occurs from Comparator A. The PTMnPF interrupt request flag, generated from a compare match occurs from Comparator P, will have no effect on the PTMn output pin. The way in which the PTMn output pin changes state are determined by the condition of the PTnIO1 and PTnIO0 bits in the PTMnC1 register. The PTMn output pin can be selected using the PTnIO1 and PTnIO0 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 PTMn output pin, which is setup after the PTnON bit changes from low to high, is setup using the PTnOC bit. Note that if the PTnIO1 and PTnIO0 bits are zero then no pin change will take place.



Compare Match Output Mode – PTnCCLR=0 (n=0 or 1)

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



Compare Match Output Mode – PTnCCLR=1 (n=0 or 1)

Note: 1. With PTnCCLR=1 a Comparator A match will clear the counter
2. The PTMn output pin is controlled only by the PTMnAF flag
3. The output pin is reset to its initial state by a PTnON bit rising edge
4. A PTMnPF flag is not generated when PTnCCLR=1

Timer/Counter Mode

To select this mode, bits PTnM1 and PTnM0 in the PTMnC1 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 PTnM1 and PTnM0 in the PTMnC1 register should be set to 10 respectively. The PWM function within the PTMn 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 PTMn output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

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

An interrupt flag, one for each of the CCRA and CCRP, will be generated when a compare match occurs from either Comparator A or Comparator P. The PTnOC bit in the PTMnC1 register is used to select the required polarity of the PWM waveform while the two PTnIO1 and PTnIO0 bits are used to enable the PWM output or to force the PTMn output pin to a fixed high or low level. The PTnPOL bit is used to reverse the polarity of the PWM output waveform.

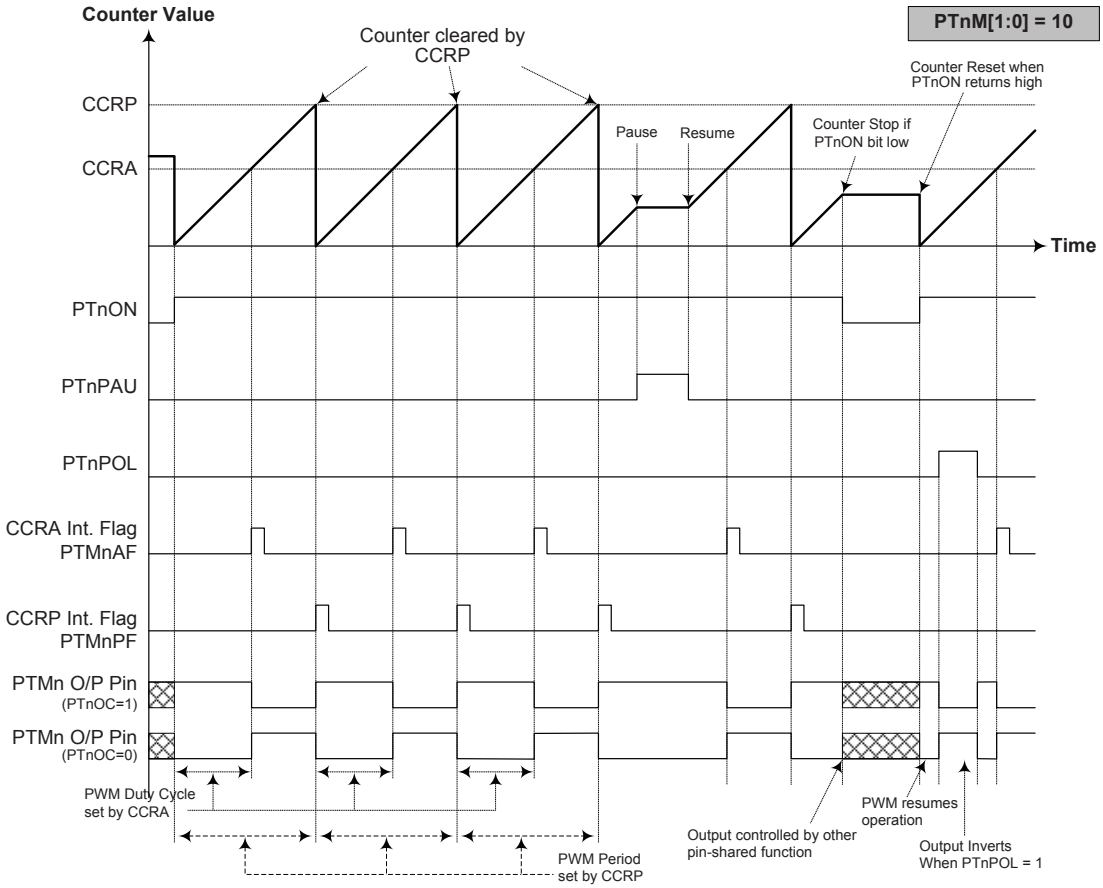
- **10-bit PTMn, PWM Output Mode, Edge-aligned Mode**

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

If f_{SYS}=12MHz, PTMn clock source select f_{SYS}/4, CCRP=512 and CCRA=128,

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



PWM Output Mode (n=0 or 1)

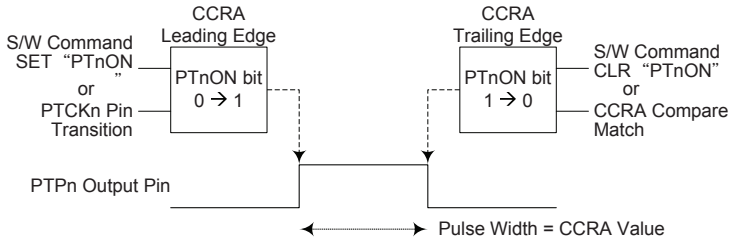
Note: 1. Counter cleared by CCRP
2. A counter clear sets the PWM Period
3. The internal PWM function continues running even when PTnIO[1:0]=00 or 01
4. The PTnCCLR bit has no influence on PWM operation

Single Pulse Output Mode

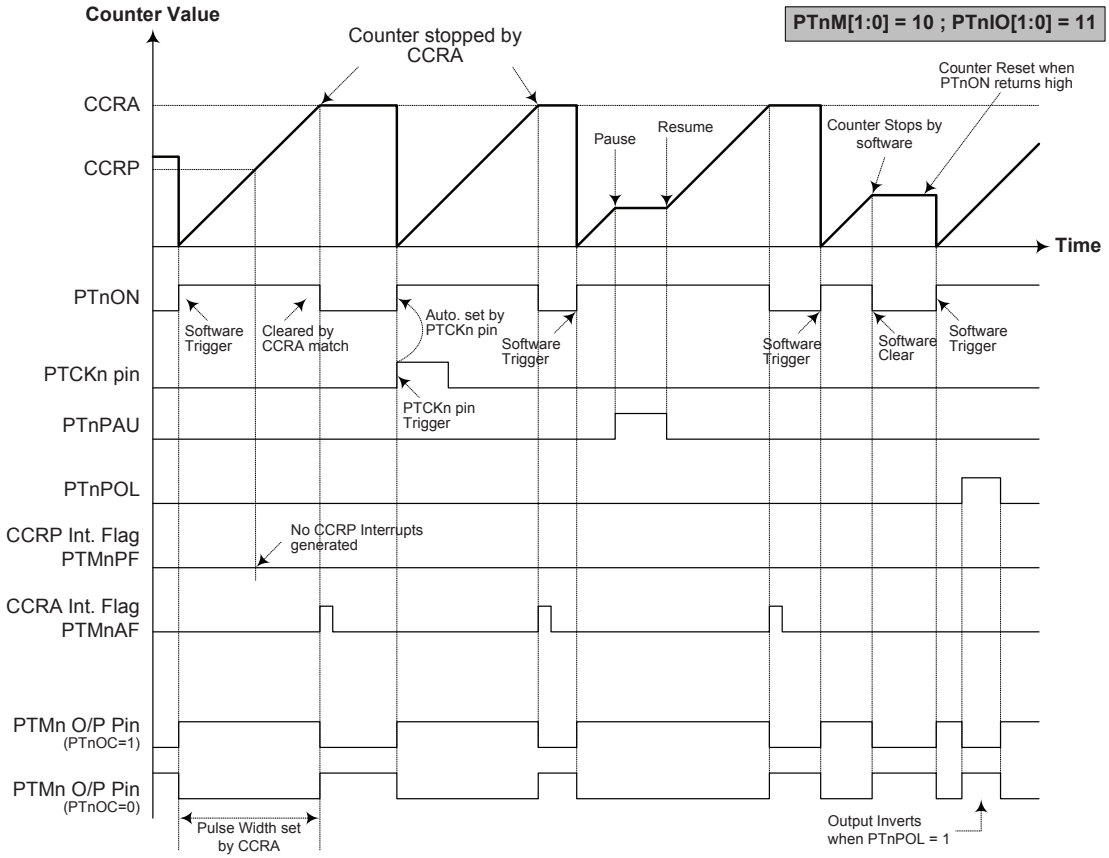
To select this mode, bits PTnM1 and PTnM0 in the PTMnC1 register should be set to 10 respectively and also the PTnIO1 and PTnIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the PTMn output pin.

The trigger for the pulse output leading edge is a low to high transition of the PTnON bit, which can be implemented using the application program. However in the Single Pulse Output Mode, the PTnON bit can also be made to automatically change from low to high using the external PTCKn pin, which will in turn initiate the Single Pulse output. When the PTnON bit transitions to a high level, the counter will start running and the pulse leading edge will be generated. The PTnON bit should remain high when the pulse is in its active state. The generated pulse trailing edge will be generated when the PTnON 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 PTnON 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 PTMn interrupt. The counter can only be reset back to zero when the PTnON bit changes from low to high when the counter restarts. In the Single Pulse Output Mode CCRP is not used. The PTnCCLR bit is not used in this Mode.



Single Pulse Generation (n=0 or 1)



Single Pulse Output Mode (n=0 or 1)

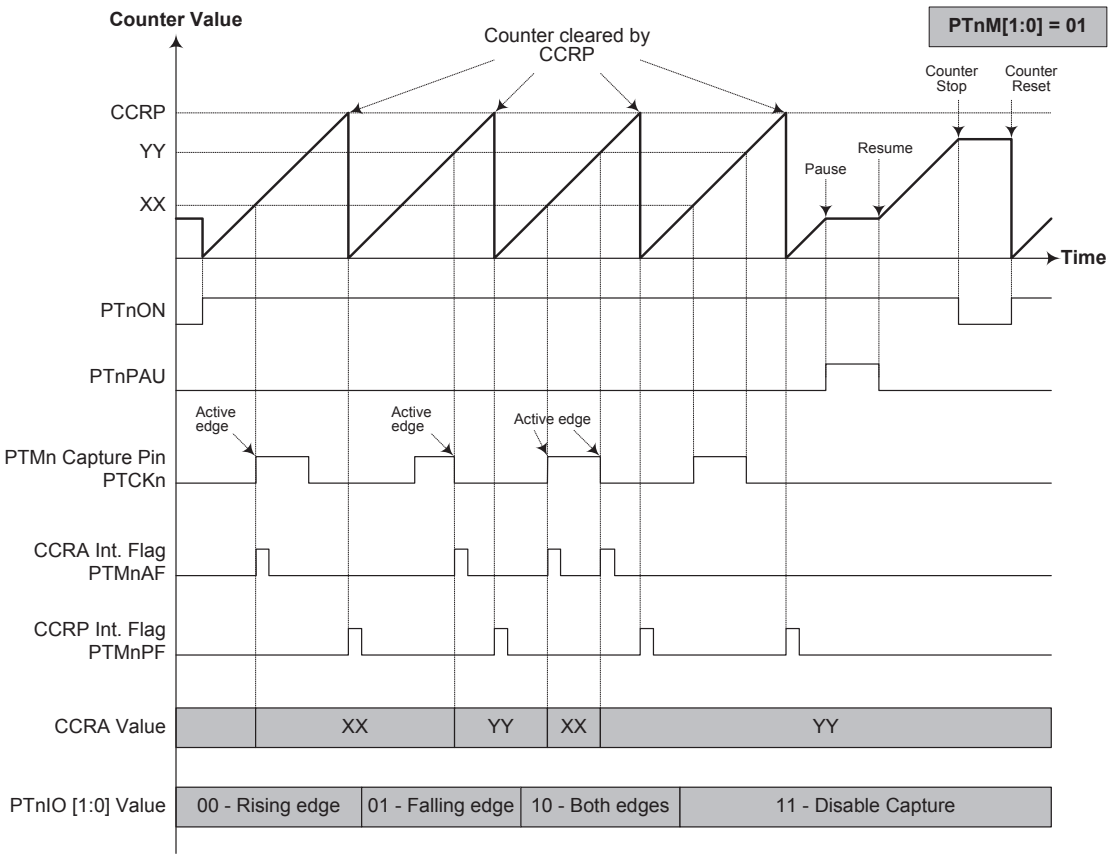
Note: 1. Counter stopped by CCRA
2. CCRP is not used
3. The pulse is triggered by the PTCKn pin or by setting the PTnON bit high
4. A PTCKn pin active edge will automatically set the PTnON bit high
5. In the Single Pulse Output Mode, PTnIO[1:0] must be set to "11" and cannot be changed.

Capture Input Mode

To select this mode bits PTnM1 and PTnM0 in the PTMnC1 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 PTCKn pin which is selected using the PTnCAPTS bit in the PTMnC1 register. The input pin active edge can be either a rising edge, a falling edge or both rising and falling edges; the active edge transition type is selected using the PTnIO1 and PTnIO0 bits in the PTMnC1 register. The counter is started when the PTnON bit changes from low to high which is initiated using the application program.

When the required edge transition appears on the PTCKn pin the present value in the counter will be latched into the CCRA registers and a PTMn interrupt generated. Irrespective of what events occur on the PTCKn pin, the counter will continue to free run until the PTnON 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 PTMn 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 PTnIO1 and PTnIO0 bits can select the active trigger edge on the PTCKn pin to be a rising edge, falling edge or both edge types. If the PTnIO1 and PTnIO0 bits are both set high, then no capture operation will take place irrespective of what happens on the PTCKn pin, however it must be noted that the counter will continue to run.

As the PTCKn pin is pin shared with other functions, care must be taken if the PTMn 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 PTnCCLR, PTnOC and PTnPOL bits are not used in this Mode.



Capture Input Mode (n=0 or 1)

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

Analog to Digital Converter

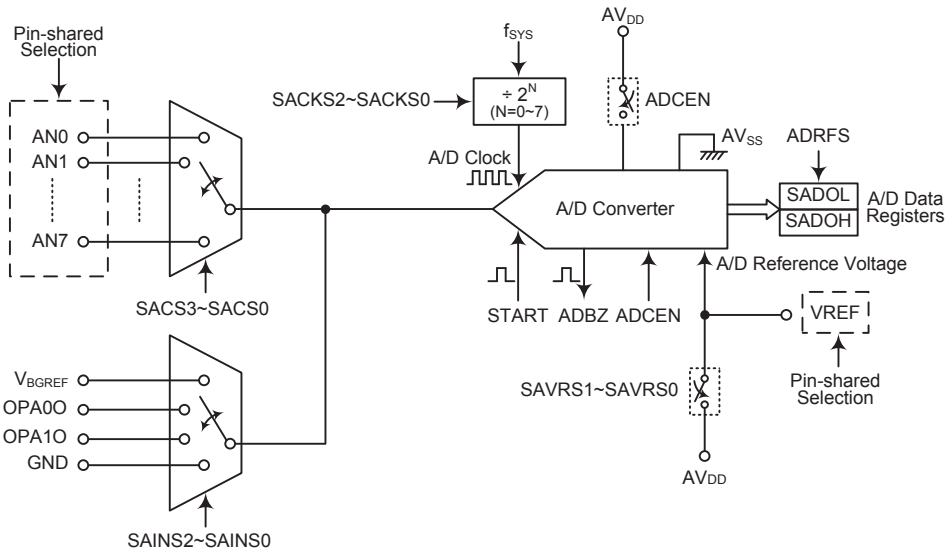
The need to interface to real world analog signals is a common requirement for many electronic systems. However, to properly process these signals by a microcontroller, they must first be converted into digital signals by A/D converters. By integrating the A/D conversion electronic circuitry into the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

A/D Converter Overview

These devices contain a multi-channel analog to digital converter which can directly interface to external analog signals, such as that from sensors or other control signals and convert these signals directly into a 12-bit digital value. It also can convert the internal signals, the high performance bandgap reference voltage V_{BGREF}, the SD operational amplifier 0 output signal OPA0O and the SD operational amplifier 1 output signal OPA1O, into a 12-bit digital value. The external or internal analog signal to be converted is determined by the SAINS2~SAINS0 bits together with the SACS3~SACS0 bits. More detailed information about the A/D input signal is described in the "A/D Converter Control Registers" and "A/D Converter Input Signals" sections.

Device	External Input Channels	Internal Signals	Channel Select Bits
HT66F4530	5: AN0~AN4	3: V_{BGREF}, OPA0O, OPA1O	SAINS2~SAINS0, SACS3~SACS0
HT66F4540	8: AN0~AN7		
HT66F4550	8: AN0~AN7		

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



A/D Converter Structure

A/D Converter Register Description

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

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

A/D Converter Registers List

A/D Converter Data Registers – SADOL, SADOH

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

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

A/D Data Registers

A/D Converter Control Registers – SADC0, SADC1

To control the function and operation of the A/D converter, two control registers known as SADC0 and SADC1 are provided. These 8-bit registers define functions such as the selection of which analog channel is connected to the internal A/D converter, the digitised data format, the A/D clock source as well as controlling the start function and monitoring the A/D converter busy status. As these devices contain only one actual analog to digital converter hardware circuit, each of the external or internal analog signal inputs must be routed to the converter. The SACS3~SACS0 bits in the SADC0 register are used to determine which external channel input is selected to be converted. The SAINS2~SAINS0 bits in the SADC1 register are used to determine that the analog signal to be converted comes from the internal analog signal or external analog channel input.

The relevant pin-shared function selection bits determine which pins on I/O Ports are used as analog inputs for the A/D converter input and which pins are not to be used as the A/D converter input. When the pin is selected to be an A/D input, its original function whether it is an I/O or other pin-shared function will be removed. In addition, any internal pull-high resistor connected to the pin will be automatically removed if the pin is selected to be an A/D converter input.

- **SADC0 Register – HT66F4530**

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

Bit 7 **START**: Start the A/D conversion
0→1→0: Start
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 **ADBZ**: A/D converter busy flag
0: No A/D conversion is in progress
1: A/D conversion is in progress
This read only flag is used to indicate whether the A/D conversion is in progress or not. When the START bit is set from low to high and then to low again, the ADBZ flag will be set to 1 to indicate that the A/D conversion is initiated. The ADBZ flag will be cleared to 0 after the A/D conversion is complete.

Bit 5 **ADCEN**: A/D converter function enable control
0: Disable
1: Enable
This bit controls the A/D internal function. This bit should be set to one to enable the A/D converter. If the bit is set low, then the A/D converter will be switched off reducing these devices power consumption. When the A/D converter function is disabled, the contents of the A/D data register pair known as SADOH and SADOL will be unchanged.

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

Bit 3~0 **SACS3~SACS0**: A/D converter external analog channel input select
0000: AN0
0001: AN1
0010: AN2
0011: AN3
0100: AN4
0101~1111: Non-existed channel, the input will be floating if selected

- **SADC0 Register – HT66F4540/HT66F4550**

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

Bit 7 **START**: Start the A/D conversion
0→1→0: Start
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 **ADBZ**: A/D converter busy flag
0: No A/D conversion is in progress
1: A/D conversion is in progress
This read only flag is used to indicate whether the A/D conversion is in progress or not. When the START bit is set from low to high and then to low again, the ADBZ flag will be set to 1 to indicate that the A/D conversion is initiated. The ADBZ flag will be cleared to 0 after the A/D conversion is complete.

Bit 5 **ADCEN**: A/D converter function enable control
0: Disable
1: Enable
This bit controls the A/D internal function. This bit should be set to one to enable the A/D converter. If the bit is set low, then the A/D converter will be switched off reducing these devices power consumption. When the A/D converter function is disabled, the contents of the A/D data register pair known as SADOH and SADOL will be unchanged.

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

Bit 3~0 **SACS3~SACS0**: A/D converter external analog channel input select
0000: AN0
0001: AN1
0010: AN2
0011: AN3
0100: AN4
0101: AN5
0110: AN6
0111: AN7
1000~1111: Non-existed channel, the input will be floating if selected

- **SADC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	SAINS2	SAINS1	SAINS0	SAVRS1	SAVRS0	SACKS2	SACKS1	SACKS0
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 7~5 **SAINS2~SAINS0**: A/D converter input signal select
000: External input – External analog channel input
001: Internal input – Internal high performance bandgap reference voltage, V_{BGREF}
010: Internal input – Internal SD operational amplifier 0 output signal, OPA0O
011: Internal input – Internal SD operational amplifier 1 output signal, OPA1O
100: Internal input – Unused, connected to ground
101~111: External input – External analog channel input
Care must be taken if the SAINS2~SAINS0 bits are set from "001" to "100" to select the internal analog signal to be converted. When the internal analog signal is selected to be converted, the external input pin must never be selected as the A/D input signal by properly setting the SACS3~SACS0 bits. Otherwise, the external channel input will be connected together with the internal analog signal. This will result in unpredictable situations such as an irreversible damage.

Bit 4~3 **SAVRS1~SAVRS0**: A/D converter reference voltage select
00: VREF pin
01: Internal A/D converter power, AV_{DD}
1x: VREF pin
These bits are used to select the A/D converter reference voltage. Care must be taken if the SAVRS1~SAVRS0 bits are set to "01" to select the internal A/D converter power as the reference voltage source. When the internal A/D converter power is selected as the reference voltage, the VREF pin cannot be configured as the reference voltage input by properly configuring the corresponding pin-shared function control bits. Otherwise, the external input voltage on VREF pin will be connected to the internal A/D converter power.

Bit 2~0 **SACKS2~SACKS0**: A/D conversion clock source select
000: f_{SYS}
001: $f_{SYS}/2$
010: $f_{SYS}/4$
011: $f_{SYS}/8$
100: $f_{SYS}/16$
101: $f_{SYS}/32$
110: $f_{SYS}/64$
111: $f_{SYS}/128$
These three bits are used to select the clock source for the A/D converter.

A/D Converter Operation

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

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

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

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

A/D Clock Period Examples

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

A/D Converter Reference Voltage

The reference voltage supply to the A/D converter can be supplied from the positive power supply pin, AVDD, or from an external reference source supplied on pin VREF. The desired selection is made using the SAVRS1 and SAVRS0 bits. When the SAVRS bit field is set to "01", the A/D converter reference voltage will come from the AVDD pin. Otherwise, if the SAVRS bit field is set to any other value except "01", the A/D converter reference voltage will come from the VREF pin. As the VREF pin is pin-shared with other functions, when the VREF pin is selected as the reference voltage supply pin, the VREF pin control bit VREFS should first be set high to enable the VREF pin function then the other pin functions will be disabled automatically. However, if the internal A/D converter power AV_{DD} is selected as the reference voltage, the VREF pin must not be configured as the reference voltage input function to avoid the internal connection between the VREF pin to A/D converter power AV_{DD}. The analog input values must not be allowed to exceed the value of the selected A/D reference voltage.

A/D Converter Input Signals

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

There are four internal analog signals derived from the high performance bandgap reference voltage V_{BGREF}, the SD operational amplifier 0 output signal OPA0O and the SD operational amplifier 1 output signal OPA1O, which can be connected to the A/D converter as the analog input signal by configuring the SAINS2~SAINS0 bits. If the external channel input is selected to be converted, the SAINS2~SAINS0 bits should be set to "000" or "101~111" and the SACS3~SACS0 bits can determine which external channel is selected. If the internal analog signal is selected to be converted, the SACS3~SACS0 bits must be configured with an appropriate value to switch off the external analog channel input. Otherwise, the internal analog signal will be connected together with the external channel input. This will result in unpredictable situations.

This V_{BGREF} is high performance bandgap voltage reference with driver capability. It has accurate voltage reference output when input supply voltage V_{DD} changes or temperature varies. And, this bandgap will startup at low voltage supply voltage. Therefore, this voltage reference has high power supply rejection ratio (PSRR) for low dropout regulator (LDO) is presented.

SAINS[2:0]	SACS[3:0]	Input Signals	Description
000, 101~111	0000~0100	AN0~AN4	External pin analog input
	0101~1111	—	Non-existed channel, input is floating.
001	0101~1111	V_{BGREF}	Internal high performance Bandgap reference voltage
010	0101~1111	OPA0O	Internal SD operational amplifier 0 output signal
011	0101~1111	OPA1O	Internal SD operational amplifier 1 output signal
100	0101~1111	GND	Unused, connected to ground

A/D Converter Input Signal Selection – HT66F4530

SAINS[2:0]	SACS[3:0]	Input Signals	Description
000, 101~111	0000~0111	AN0~AN7	External pin analog input
	1000~1111	—	Non-existed channel, input is floating.
001	1000~1111	V_{BGREF}	Internal high performance Bandgap reference voltage
010	1000~1111	OPA0O	Internal SD operational amplifier 0 output signal
011	1000~1111	OPA1O	Internal SD operational amplifier 1 output signal
100	1000~1111	GND	Unused, connected to ground

A/D Converter Input Signal Selection – HT66F4540/HT66F4550

VBGRC Register

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

Bit 7~1 Unimplemented, read as "0"

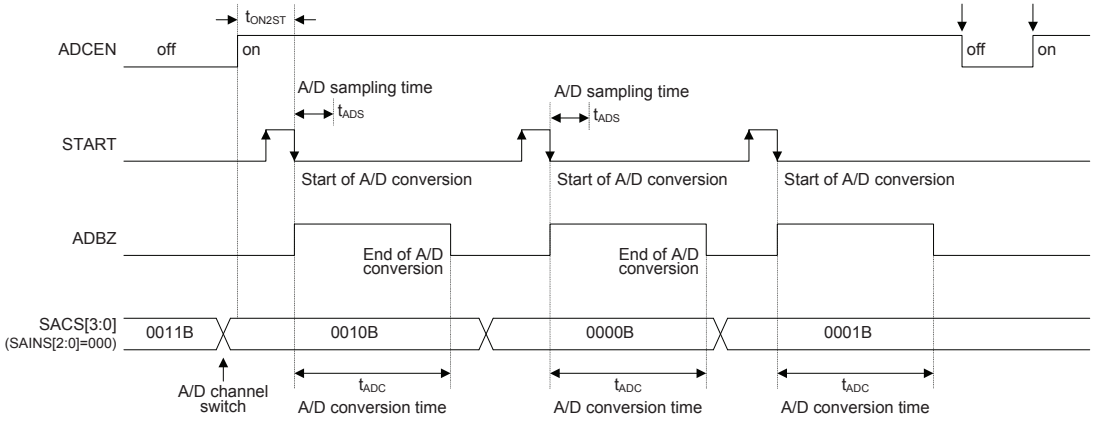
Bit 0 **VBGREN**: Bandgap enable or disable control
0: Disable
1: Enable
When the VBGREN bit is cleared to zero, the V_{BGREF} is in a high impedance state.

Conversion Rate and Timing Diagram

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

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

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



A/D Conversion Timing

Summary of A/D Conversion Steps

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

- Step 1
  Select the required A/D conversion clock by correctly programming bits SACKS2~SACKS0 in the SADC1 register.
- Step 2
  Enable the A/D by setting the ADCEN bit in the SADC0 register to one.
- Step 3
  Select which signal is to be connected to the internal A/D converter by correctly configuring the SAINS2~SAINS0 bits
  Select the external channel input to be converted, go to Step 4.
  Select the internal analog signal to be converted, go to Step 5.
- Step 4
  If the A/D input signal comes from the external channel input selected by configuring the SAINS2~SAINS0 bits field, the corresponding pins should be configured as A/D input function by configuring the relevant pin-shared function control bits. The desired analog channel then should be selected by configuring the SACS3~SACS0 bits field. After this step, go to Step 6.
- Step 5
  Before the A/D input signal is selected to come from the internal analog signal by configuring the SAINS2~SAINS0 bits field, the corresponding external input pin must be switched to a non-existed channel input by properly configured the SACS3~SACS0 bits. The desired internal analog signal then can be selected by configuring the SAINS2~SAINS0 bits field. After this step, go to Step 6.
- Step 6
  Select the reference voltage source by configuring the SAVRS1~SAVRS0 bits in the SADC1 register.
- Step 7
  Select A/D converter output data format by setting the ADRFS bit in the SADC0 register.
- Step 8
  If A/D conversion interrupt is used, the interrupt control registers must be correctly configured to ensure the A/D interrupt function is active. The master interrupt control bit, EMI, and the A/D conversion interrupt control bit, ADE, must both be set high in advance.
- Step 9
  The A/D conversion procedure can now be initialized by setting the START bit from low to high and then low again.
- Step 10
  If A/D conversion is in progress, the ADBZ flag will be set high. After the A/D conversion process is complete, the ADBZ flag will go low and then the output data can be read from SADOH and SADOL registers.

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

Programming Considerations

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

A/D Conversion Function

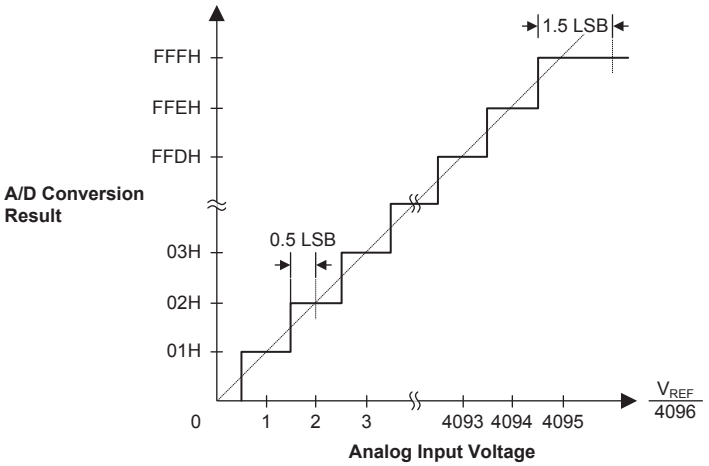
As these devices contain a 12-bit A/D converter, its full-scale converted digitised value is equal to FFFH. Since the full-scale analog input value is equal to the actual A/D converter reference voltage, V_{REF}, this gives a single bit analog input value of V_{REF} divided by 4096.

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

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

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

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



Ideal A/D Transfer Function

A/D Conversion Programming Examples

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

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

```
clr  ADE                ; disable ADC interrupt
mov  a,03H
mov  SADC1,a            ; select fSYS/8 as A/D clock
set  ADCEN
mov  a,10h              ; setup PAS1 register to configure pin AN0
mov  PAS1,a
mov  a,20h
mov  SADC0,a            ; enable and connect AN0 channel to A/D converter
:
start_conversion:
clr  START              ; high pulse on start bit to initiate conversion
set  START              ; reset A/D
clr  START              ; start A/D
polling_EOC:
sz   ADBZ               ; poll the SADC0 register ADBZ bit to detect end of A/D
conversion
jmp  polling_EOC        ; continue polling
mov  a,SADOL            ; read low byte conversion result value
mov  SADOL_buffer,a     ; save result to user defined register
mov  a,SADOH            ; read high byte conversion result value
mov  SADOH_buffer,a     ; save result to user defined register
:
:
jmp  start_conversion   ; start next A/D conversion
```

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

```
clr  ADE                ; disable ADC interrupt
mov  a,03H
mov  SADC1,a            ; select fSYS/8 as A/D clock
set  ADCEN
mov  a,10h              ; setup PAS1 register to configure pin AN0
mov  PAS1,a
mov  a,20h
mov  SADC0,a            ; enable and connect AN0 channel to A/D converter
Start_conversion:
clr  START              ; high pulse on START bit to initiate conversion
set  START              ; reset A/D
clr  START              ; start A/D
clr  ADF                ; clear ADC interrupt request flag
set  ADE                ; enable ADC interrupt
set  EMI                ; enable global interrupt
:
:
; ADC interrupt service routine
ADC_ISR:
mov  acc_stack,a        ; save ACC to user defined memory
mov  a,STATUS
mov  status_stack,a     ; save STATUS to user defined memory
:
:
mov  a,SADOL            ; read low byte conversion result value
mov  SADOL_buffer,a     ; save result to user defined register
mov  a,SADOH            ; read high byte conversion result value
mov  SADOH_buffer,a     ; save result to user defined register
:
:
EXIT_INT_ISR:
mov  a,status_stack
mov  STATUS,a           ; restore STATUS from user defined memory
mov  a,acc_stack        ; restore ACC from user defined memory
reti
```

Serial Interface Module – SIM

These devices contain a Serial Interface Module, which includes both the four line SPI interface and the two line I^2C interface types, to allow an easy method of communication with external peripheral hardware. Having relatively simple communication protocols, these serial interface types allow the microcontroller to interface to external SPI or I^2C based hardware such as sensors, Flash or EEPROM memory, etc. The SIM interface pins are pin-shared with other I/O pins 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 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 these devices can be either master or slave. Although the SPI interface specification can control multiple slave devices from a single master, but these devices provide 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.

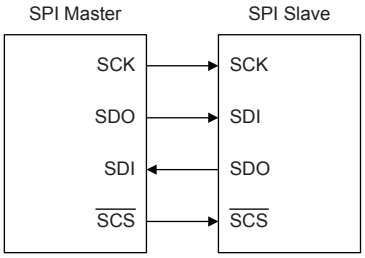
SPI Interface Operation

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, the SCK pin 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. 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 these devices only contain 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.

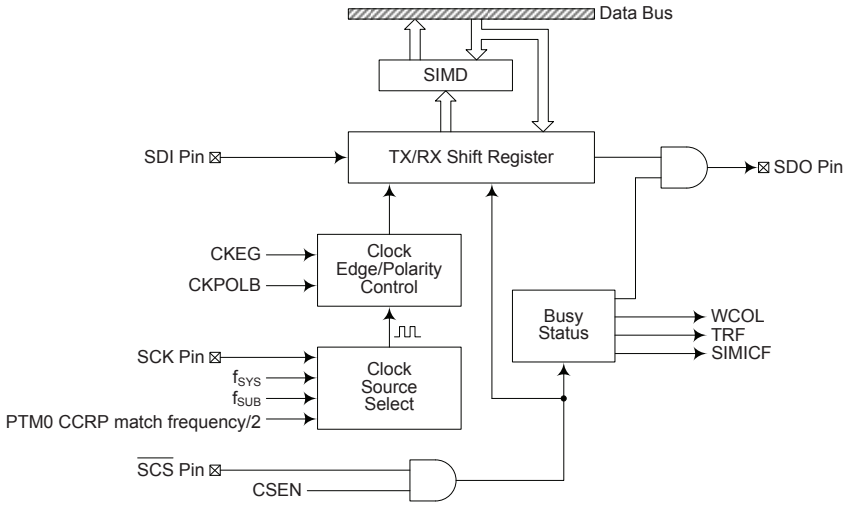
The SPI function in these devices offer the following features:

- 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 these devices are in the master or slave mode and upon the condition of certain control bits such as CSEN and SIMEN.



SPI Master/Slave Connection



SPI Block 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 control registers, SIMC0 and SIMC2.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	—	SIMDEB1	SIMDEB0	SIMEN	SIMICF
SIMC2	D7	D6	CKPOLB	CKEG	MLS	CSEN	WCOL	TRF
SIMD	D7	D6	D5	D4	D3	D2	D1	D0

SPI Registers List

SPI Data Register

The SIMD register is used to store the data being transmitted and received. The same register is used by both the SPI and I^2C functions. Before these 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, these 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

Bit 7~0 **D7~D0**: SIM data register bit 7 ~ bit 0

SPI Control Registers

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 SIMC0 register is used to control the enable/disable function and to set the data transmission clock frequency. The SIMC2 register 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	SIMICF
R/W	R/W	R/W	R/W	—	R/W	R/W	R/W	R/W
POR	1	1	1	—	0	0	0	0

Bit 7~5 **SIM2~SIM0**: SIM Operating Mode Control
000: SPI master mode; SPI clock is $f_{SYS}/4$
001: SPI master mode; SPI clock is $f_{SYS}/16$
010: SPI master mode; SPI clock is $f_{SYS}/64$
011: SPI master mode; SPI clock is f_{SUB}
100: SPI master mode; SPI clock is PTM0 CCRP match frequency/2
101: SPI slave mode
110: I²C slave mode
111: Unused mode
These bits setup the overall operating mode of the SIM function. As well as selecting if the I²C or SPI function, they are used to control the SPI Master/Slave selection and the SPI Master clock frequency. The SPI clock is a function of the system clock but can also be chosen to be sourced from PTM0 and f_{SUB}. 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**: I²C Debounce Time Selection
The SIMDEB1~SIMDEB0 bits are only used in the I²C mode and the detailed definition is described in the I²C section.

Bit 1 **SIMEN**: SIM Enable 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 I²C function and the SIM operating current will be reduced to a minimum value. When the bit is high the SIM interface is enabled. If the SIM is configured to operate as an SPI interface via the SIM2~SIM0 bits, the contents of the SPI control registers will remain at the previous settings when the SIMEN bit changes from low to high and should therefore be first initialised by the application program. If the SIM is configured to operate as an I²C interface via the SIM2~SIM0 bits and the SIMEN bit changes from low to high, the contents of the I²C control bits such as HTX and TXAK will remain at the previous settings and should therefore be first initialised by the application program while the relevant I²C flags such as HCF, HAAS, HBB, SRW and RXAK will be set to their default states.

Bit 0 **SIMICF**: SIM SPI Incomplete Flag
0: SIM SPI incomplete condition is not occurred
1: SIM SPI incomplete condition is occurred
This bit is only available when the SIM is configured to operate in an SPI slave mode. If the SPI operates in the slave mode with the SIMEN and CSEN bits both being set to 1 but the $\overline{SCS}$ line is pulled high by the external master device before the SPI data transfer is completely finished, the SIMICF bit will be set to 1 together with the TRF bit. When this condition occurs, the corresponding interrupt will occur if the interrupt function is enabled. However, the TRF bit will not be set to 1 if the SIMICF bit is set to 1 by software application program.

- **SIMC2 Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	CKPOLB	CKEG	MLS	CSEN	WCOL	TRF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **D7~D6**: Undefined bits

These bits can be read or written by the application program.

Bit 5 **CKPOLB**: SPI clock line base condition selection

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**: SPI SCK clock active edge type selection

CKPOLB=0

0: SCK is high base level when the clock is inactive and data capture at SCK rising edge.

1: SCK is high base level when the clock is inactive and data capture at SCK falling edge.

CKPOLB=1

0: SCK is low base level when the clock is inactive and data capture at SCK falling edge.

1: SCK is low base level when the clock is inactive 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 first

1: MSB first

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

Bit 2 **CSEN**: SPI $\overline{\text{SCS}}$ pin control

0: Disable

1: Enable

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

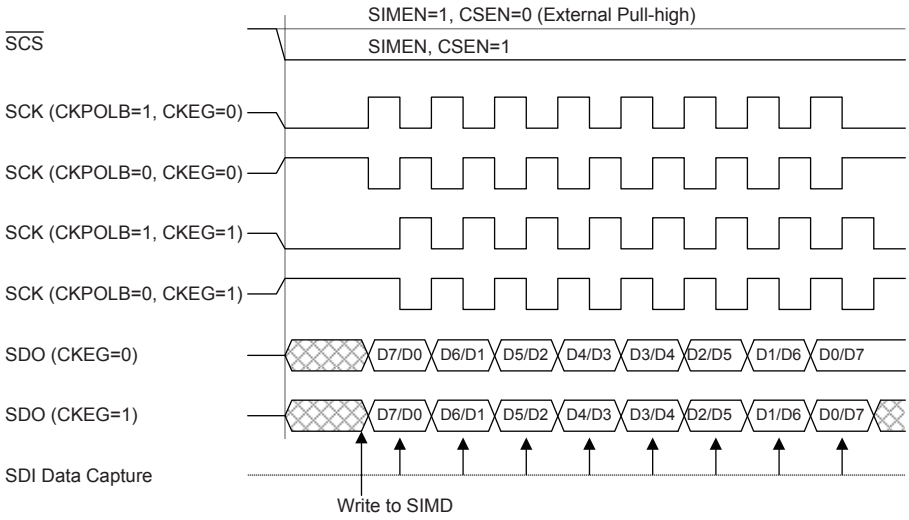
1: SPI data transmission is completed

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.

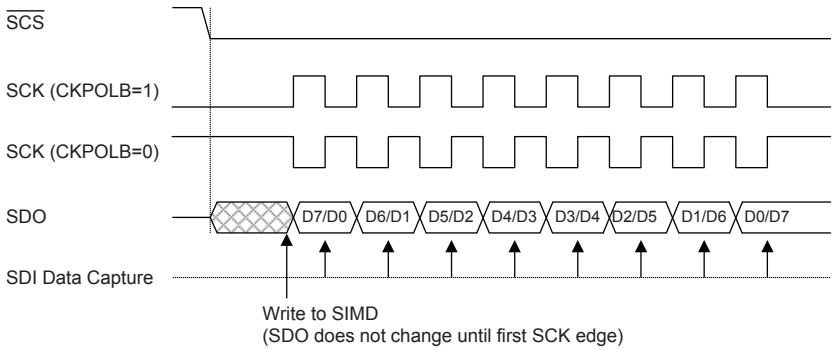
SPI Communication

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.

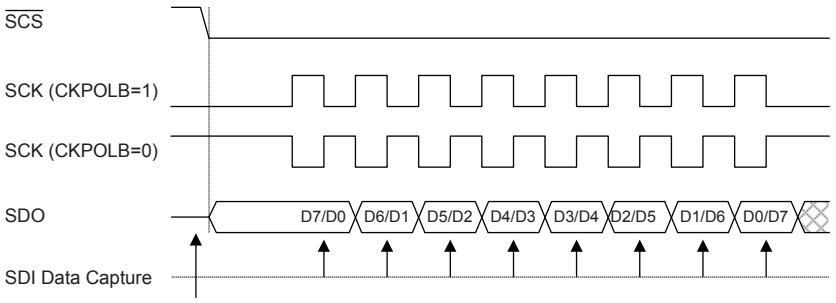
The SPI will continue to function in special IDLE Modes if the clock source used by the SPI interface is still active.



SPI Master Mode Timing

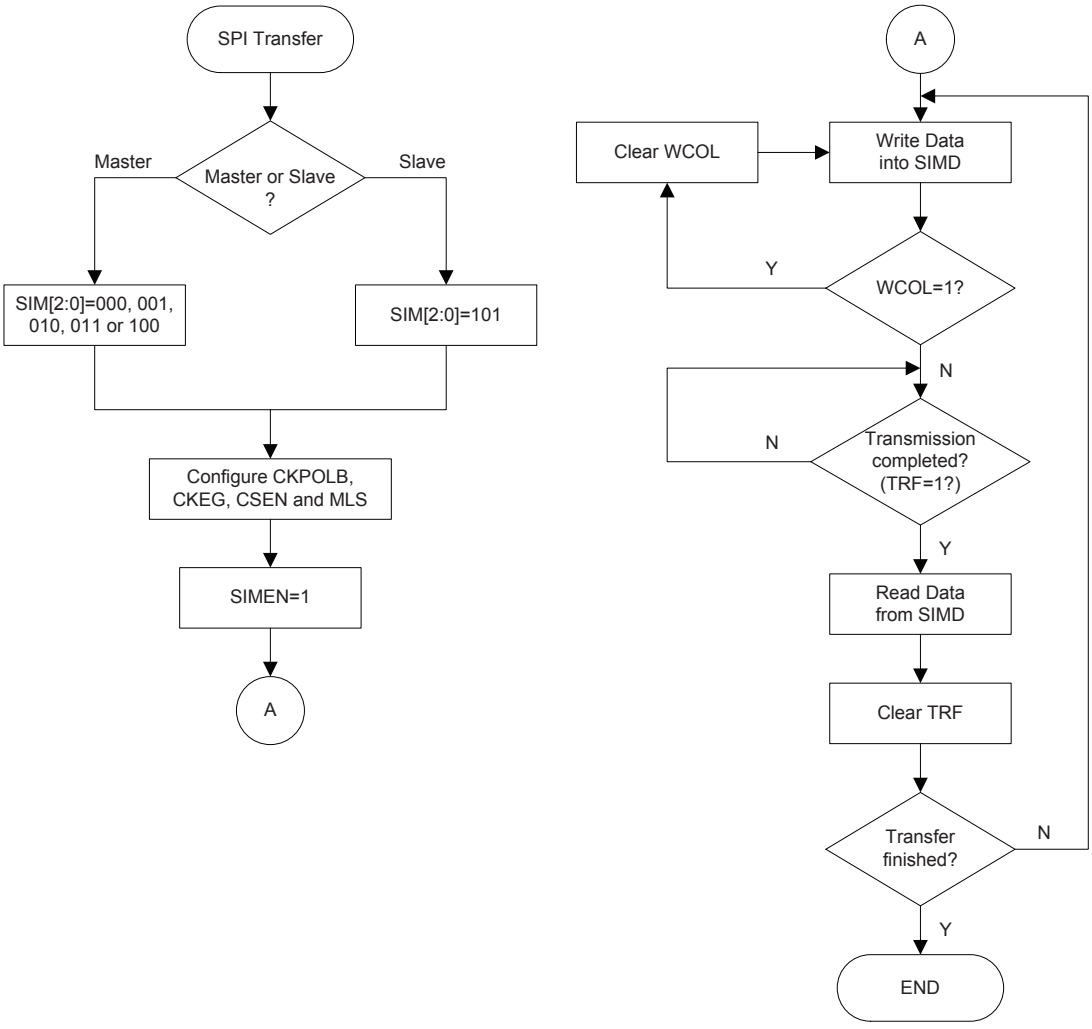


SPI Slave Mode Timing – CKEG=0



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

SPI Slave Mode Timing – CKEG=1



SPI Transfer Control Flowchart

SPI Bus Enable/Disable

To enable the SPI bus, set CSEN=1 and $\overline{SCS}$=0, then wait for data to be written into the SIMD (TXRX buffer) register. For the Master Mode, after data has been written to the SIMD (TXRX buffer) register, then transmission or reception will start automatically. When all the data has been transferred, the TRF bit should be set high. For the Slave Mode, when clock pulses are received on SCK, data in the TXRX buffer will be shifted out or data on SDI will be shifted in.

When the SPI bus is disabled, SCK, SDI, SDO and $\overline{SCS}$ can become I/O pins or other pin-shared functions using the corresponding pin-shared control bits.

SPI Operation

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

The CSEN bit in the SIMC2 register controls the overall function of the SPI interface. Setting this bit high will enable the SPI interface by allowing the $\overline{SCS}$ line to be active, which can then be used to control the SPI interface. If the CSEN bit is low, the SPI interface will be disabled and the $\overline{SCS}$ line will be in a floating condition and can therefore not be used for control of the SPI interface. If the CSEN bit and the SIMEN bit in the SIMC0 are set high, this will place the SDI line in a floating condition and the SDO line high. If in Master Mode the SCK line will be either high or low depending upon the clock polarity selection bit CKPOLB in the SIMC2 register. If in Slave Mode the SCK line will be in a floating condition. If the SIMEN bit is low, then the bus will be disabled and $\overline{SCS}$, SDI, SDO and SCK will all become I/O pins or the other functions using the corresponding pin-shared control bits. 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 SIMD 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 SPI Master mode and clock source using the SIM2~SIM0 bits in the SIMC0 control register.
  - Step 2
    Setup the CSEN bit and setup the MLS bit to choose if the data is MSB or LSB first, this setting must be the same with the Slave devices.
  - Step 3
    Setup the SIMEN bit in the SIMC0 control register to enable the SPI interface.
  - Step 4
    For write operations: write the data to the SIMD register, which will actually place the data into the TXRX buffer. Then use the SCK and $\overline{SCS}$ lines to output the data. After this, go to step5.
    For read operations: the data transferred in on the SDI line will be stored in the TXRX buffer until all the data has been received at which point it will be latched into the SIMD register.
  - Step 5
    Check the WCOL 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 TRF bit or wait for a SPI serial bus interrupt.
  - Step 7
    Read data from the SIMD register.

- Step 8
  Clear TRF.
- Step 9
  Go to step 4.

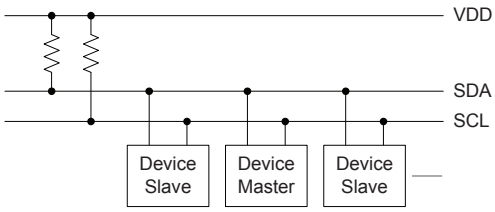
- **Slave Mode:**
  - Step 1
    Select the SPI Slave mode using the SIM2~SIM0 bits in the SIMC0 control register
  - Step 2
    Setup the CSEN bit and setup the MLS bit to choose if the data is MSB or LSB first, this setting must be the same with the Master devices.
  - Step 3
    Setup the SIMEN bit in the SIMC0 control register to enable the SPI interface.
  - Step 4
    For write operations: write the data to the SIMD register, which will actually place the data into the TXRX buffer. Then wait for the master clock SCK and $\overline{SCS}$ signal. After this, go to step5.
    For read operations: the data transferred in on the SDI line will be stored in the TXRX buffer until all the data has been received at which point it will be latched into the SIMD register.
  - Step 5
    Check the WCOL 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 TRF bit or wait for a SPI serial bus interrupt.
  - Step 7
    Read data from the SIMD register.
  - Step 8
    Clear TRF.
  - Step 9
    Go to step 4.

Error Detection

The WCOL bit in the SIMC2 register is provided to indicate errors during data transfer. The bit is set by the SPI 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 SIMD register takes place during a data transfer operation and will prevent the write operation from continuing.

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.

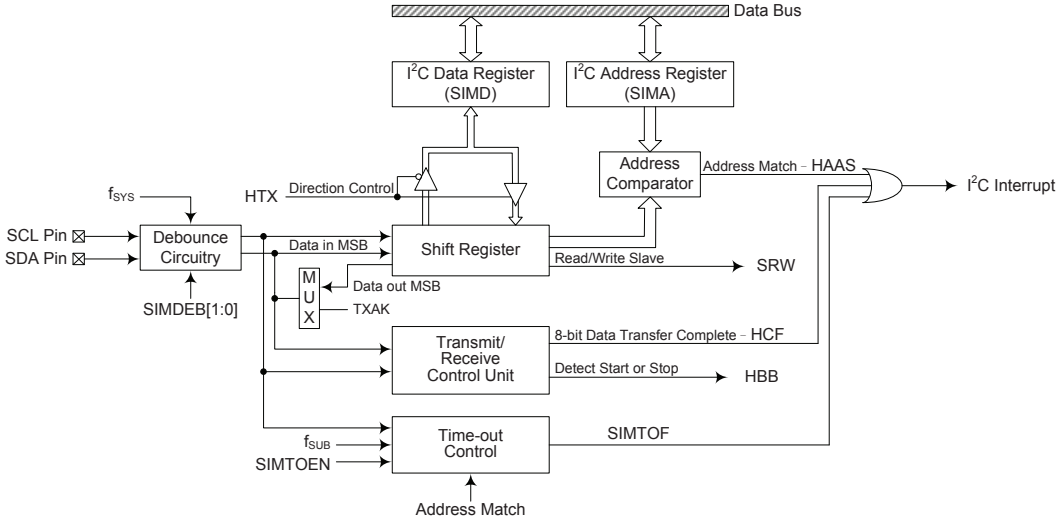


I²C Master Slave Bus Connection

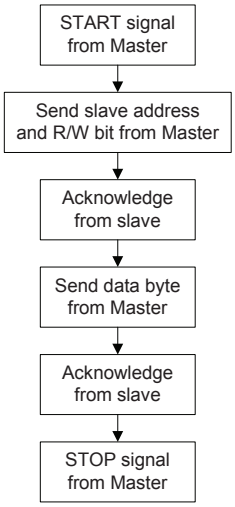
I²C Interface Operation

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.



I²C Block Diagram



The SIMDEB1 and SIMDEB0 bits determine the debounce time of the I²C interface. This uses the internal clock to in effect add a debounce time to the external clock to reduce the possibility of glitches on the clock line causing erroneous operation. The debounce time, if selected, can be chosen to be either 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} > 2 MHz	f_{SYS} > 5 MHz
2 system clock debounce	f_{SYS} > 4 MHz	f_{SYS} > 10 MHz
4 system clock debounce	f_{SYS} > 8 MHz	f_{SYS} > 20 MHz

I²C Minimum f_{SYS} Frequency

I²C Registers

There are three control registers associated with the I²C bus, SIMC0, SIMC1 and SIMTOC, one address register SIMA and one data register, SIMD.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	—	SIMDEB1	SIMDEB0	SIMEN	SIMICF
SIMC1	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
SIMD	D7	D6	D5	D4	D3	D2	D1	D0
SIMA	A6	A5	A4	A3	A2	A1	A0	—
SIMTOC	SIMTOEN	SIMTOF	SIMTOS5	SIMTOS4	SIMTOS3	SIMTOS2	SIMTOS1	SIMTOS0

I²C Registers List

I²C Data Register

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 these devices write 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, these devices can read it from the SIMD register. Any transmission or reception of data from the I²C 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

Bit 7~0 **D7~D0**: SIM data register bit 7 ~ bit 0

I²C Address Register

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 these devices 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.

- **SIMA Register**

Bit	7	6	5	4	3	2	1	0
Name	A6	A5	A4	A3	A2	A1	A0	—
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 7~1 **A6~A0**: I²C slave address
A6~A0 is the I²C slave address bit 6~bit 0.

Bit 0 Unimplemented, read as "0"

I²C Control Registers

There are three control registers for the I²C interface, SIMC0, SIMC1 and SIMTOC. The SIMC0 register is used to control the enable/disable function and to set the data transmission clock frequency. The SIMC1 register contains the relevant flags which are used to indicate the I²C communication status. Another register, SIMTOC, is used to control the I²C time-out function and described in the corresponding section.

- **SIMC0 Register**

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

Bit 7~5 **SIM2~SIM0**: SIM Operating Mode Control
000: SPI master mode; SPI clock is $f_{SYS}/4$
001: SPI master mode; SPI clock is $f_{SYS}/16$
010: SPI master mode; SPI clock is $f_{SYS}/64$
011: SPI master mode; SPI clock is f_{SUB}
100: SPI master mode; SPI clock is PTM0 CCRP match frequency/2
101: SPI slave mode
110: I²C slave mode
111: Unused mode
These bits setup the overall operating mode of the SIM function. As well as selecting if the I²C or SPI function, they are used to control the SPI Master/Slave selection and the SPI Master clock frequency. The SPI clock is a function of the system clock but can also be chosen to be sourced from PTM0 and f_{SUB}. 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**: I²C Debounce Time Selection
00: No debounce
01: 2 system clock debounce
1x: 4 system clock debounce
These bits are used to select the I²C debounce time when the SIM is configured as the I²C interface function by setting the SIM2~SIM0 bits to "110".

Bit 1 **SIMEN**: SIM Enable 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 I²C function and the SIM operating current will be reduced to a minimum value. When the bit is high the SIM interface is enabled. If the SIM is configured to operate as an SPI interface via the SIM2~SIM0 bits, the contents of the SPI control registers will remain at the previous settings when the SIMEN bit changes from low to high and should therefore be first initialised by the application program. If the SIM is configured to operate as an I²C interface via the SIM2~SIM0 bits and the SIMEN bit changes from low to high, the contents of the I²C control bits such as HTX and TXAK will remain at the previous settings and should therefore be first initialised by the application program while the relevant I²C flags such as HCF, HAAS, HBB, SRW and RXAK will be set to their default states.

Bit 0 **SIMICF**: SIM SPI Incomplete Flag
The SIMICF bit is only used in the SPI mode and the detailed definition is described in the SPI section.

- **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**: I²C slave device is transmitter or receiver selection
0: Slave device is the receiver
1: Slave device is the transmitter

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

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

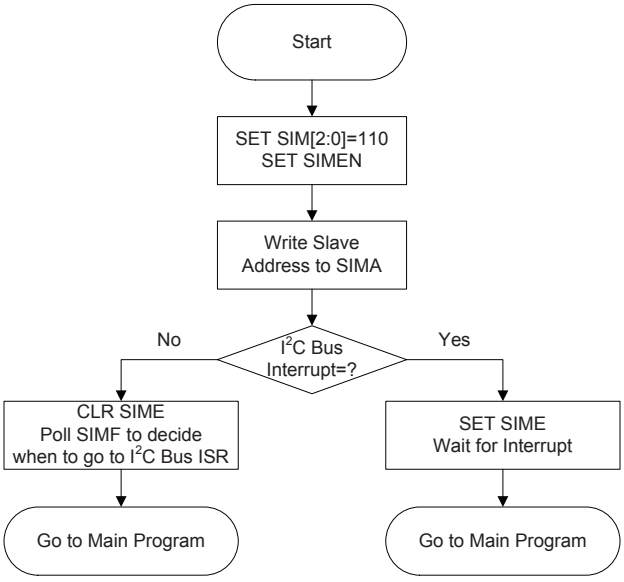
Bit 1 **IAMWU**: I²C Address Match Wake-up control
0: Disable
1: Enable
This bit should be set to 1 to enable the I²C address match wake up from the SLEEP or IDLE Mode. If the IAMWU bit has been set before entering either the SLEEP or IDLE mode to enable the I²C address match wake up, then this bit must 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.

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 and SIMTOF bits to determine whether the interrupt source originates from an address match or from the completion of an 8-bit data transfer completion or I²C bus time-out occurrence. 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 "110" and "1" respectively to enable the I²C bus.
- Step 2
  Write the slave address of these devices to the I²C bus address register SIMA.
- Step 3
  Set the SIME interrupt enable bit of the interrupt control register to enable the SIM 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.

I²C 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 three sources, when the program enters the interrupt subroutine, the HAAS and SIMTOF bits 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 or the I²C bus time-out occurrence. When a slave address is matched, these 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 master device wishes to read data from the I²C bus or write data to the I²C bus. The slave device should examine this bit to determine if it is to be a transmitter or a receiver. If the SRW flag is "1" then this indicates that the master device wishes to read data from the I²C bus, therefore the slave device must be setup to send data to the I²C bus as a transmitter. If the SRW flag is "0" then this indicates that the master wishes to send data to the I²C bus, therefore the slave device must be setup to read data from the I²C bus as a receiver.

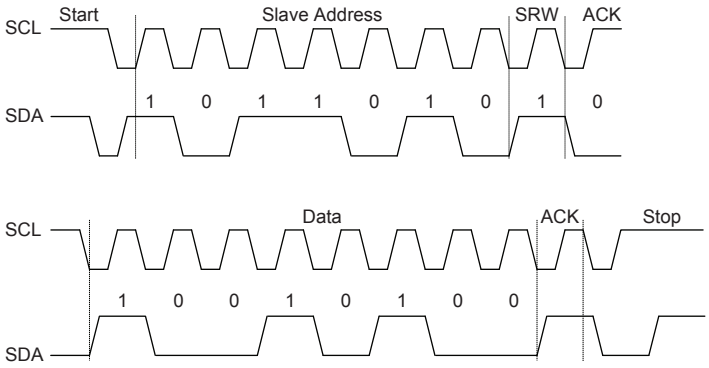
I²C Bus Slave Address Acknowledge Signal

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

I²C Bus Data and Acknowledge Signal

The transmitted data is 8-bit 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.

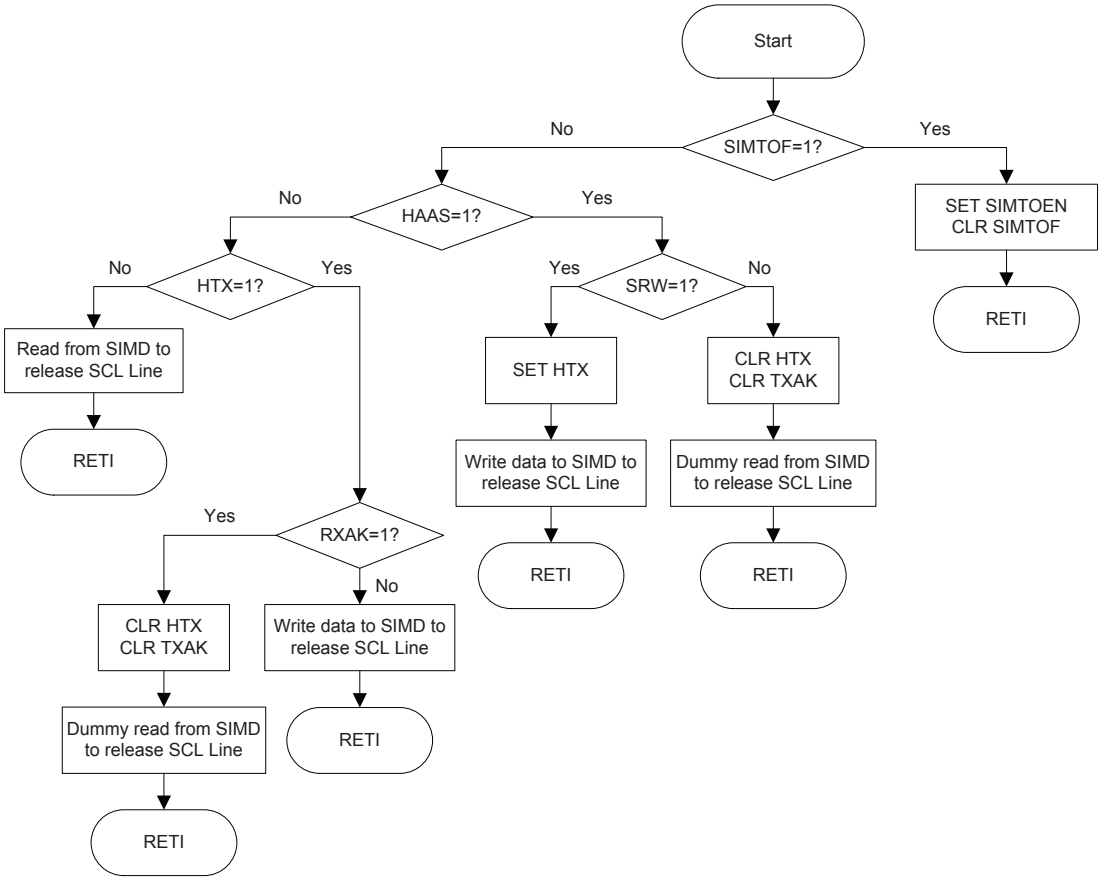


S=Start (1 bit)
SA=Slave Address (7 bits)
SR=SRW bit (1 bit)
M=Slave device send acknowledge bit (1 bit)
D=Data (8 bits)
A=ACK (RXAK bit for transmitter, TXAK bit for receiver, 1 bit)
P=Stop (1 bit)

S	SA	SR	M	D	A	D	A	·····	S	SA	SR	M	D	A	D	A	·····	P

I²C Communication Timing Diagram

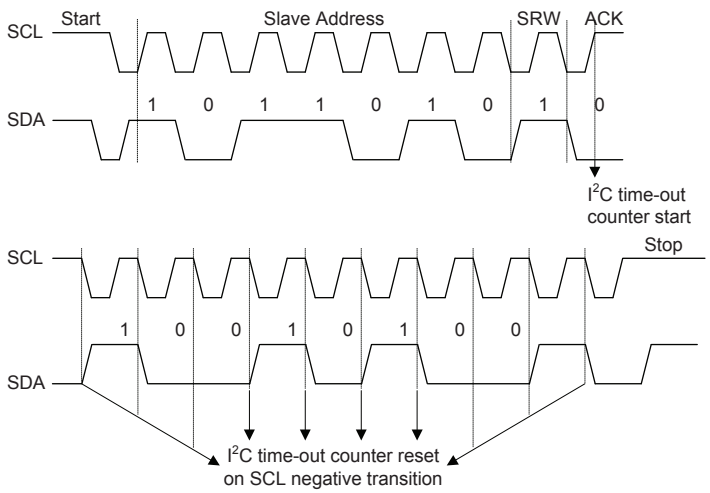
Note: *When a slave address is matched, these devices must be placed in either the transmit mode and then write data to the SIMD register, or in the receive mode where it must implement a dummy read from the SIMD register to release the SCL line.



I²C Bus ISR Flow Chart

I²C Time-out Control

In order to reduce the problem of I²C lockup due to reception of erroneous clock sources, a time-out function is provided. If the clock source to the I²C 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 counting 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 setup by the SIMTOC 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 SIMTOEN bit will be cleared to zero and the SIMTOF 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 interrupt vector. When an I²C time-out occurs, the I²C internal circuitry will be reset and the registers will be reset into the following condition:

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

I²C Registers after Time-out

The SIMTOF flag can be cleared by the application program. There are 64 time-out periods which can be selected using SIMTOS bit field in the SIMTOC register. The time-out time is given by the formula: ((1~64) × 32) / f_{SUB}. This gives a time-out period which ranges from about 1ms to 64ms.

- **SIMTOC Register**

Bit	7	6	5	4	3	2	1	0
Name	SIMTOEN	SIMTOF	SIMTOS5	SIMTOS4	SIMTOS3	SIMTOS2	SIMTOS1	SIMTOS0
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 **SIMTOEN**: SIM I²C Time-out control
0: Disable
1: Enable

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

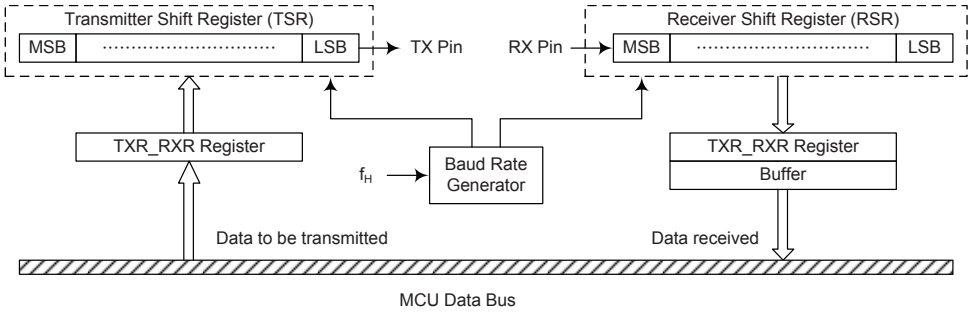
Bit 5~0 **SIMTOS5~SIMTOS0**: SIM I²C Time-out period selection
I²C time-out clock source is $f_{SUB}/32$.
I²C time-out time is equal to (SIMTOS[5:0]+1) × ($32/f_{SUB}$).

UART Interface – HT66F4540/HT66F4550

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

The integrated UART function contains the following features:

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



UART Data Transfer Block Diagram

UART External Pins

To communicate with an external serial interface, the internal UART has two external pins known as TX and RX. The TX and RX pins are the UART transmitter and receiver pins respectively. The TX and RX pin function should first be selected by the corresponding pin-shared function selection register before the UART function is used. Along with the UARTEN bit, the TXEN and RXEN bits, if set, will setup these pins to their respective TX output and RX input conditions and disable any pull-high resistor option which may exist on the TX and RX pins. When the TX or RX pin function is disabled by clearing the UARTEN, TXEN or RXEN bit, the TX or RX pin will be set to a floating state. At this time whether the internal pull-high resistor is connected to the TX or RX pin or not is determined by the corresponding I/O pull-high function control bit.

UART Data Transfer Scheme

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

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

It should be noted that the actual register for data transmission and reception only exists as a single shared register in the Data Memory. This shared register known as the TXR_RXR register is used for both data transmission and data reception.

UART Status and Control Registers

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

Register Name	Bit							
	7	6	5	4	3	2	1	0
USR	PERR	NF	FERR	OERR	RIDLE	RXIF	TIDLE	TXIF
UCR1	UARTEN	BNO	PREN	PRT	STOPS	TXBRK	RX8	TX8
UCR2	TXEN	RXEN	BRGH	ADDEN	WAKE	RIE	TIIE	TEIE
TXR_RXR	TXRX7	TXRX6	TXRX5	TXRX4	TXRX3	TXRX2	TXRX1	TXRX0
BRG	BRG7	BRG6	BRG5	BRG4	BRG3	BRG2	BRG1	BRG0

UART Registers List

USR Register

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

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

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

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

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

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

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

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

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

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

Bit 3 **RIDLE**: Receiver status
0: Data reception is in progress (Data being received)
1: No data reception is in progress (Receiver is idle)

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

Bit 2 **RXIF**: Receive TXR_RXR data register status
0: TXR_RXR data register is empty
1: TXR_RXR data register has available data
The RXIF flag is the receive data register status flag. When this read only flag is "0", it indicates that the TXR_RXR read data register is empty. When the flag is "1", it indicates that the TXR_RXR read data register contains new data. When the contents of the shift register are transferred to the TXR_RXR register, an interrupt is generated if RIE=1 in the UCR2 register. If one or more errors are detected in the received word, the appropriate receive-related flags NF, FERR, and/or PERR are set within the same clock cycle. The RXIF flag is cleared when the USR register is read with RXIF set, followed by a read from the TXR_RXR register, and if the TXR_RXR register has no data available.

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

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

UCR1 Register

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

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

"x" unknown

Bit 7 **UARTEN**: UART function enable control
0: Disable UART. TX and RX pins are used as I/O or other pin-shared functional pins
1: Enable UART. TX and RX pins function as UART pins
The UARTEN bit is the UART enable bit. When this bit is equal to "0", the UART will be disabled and the RX pin as well as the TX pin will be as General Purpose I/O or other pin-shared functional pins. When the bit is equal to "1", the UART will be enabled and the TX and RX pins will function as defined by the TXEN and RXEN enable control bits.

When the UART is disabled, it will empty the buffer so any character remaining in the buffer will be discarded. In addition, the value of the baud rate counter will be reset. If the UART is disabled, all error and status flags will be reset. Also the TXEN, RXEN, TXBRK, RXIF, OERR, FERR, PERR and NF bits will be cleared, while the TIDLE, TXIF and RIDLE bits will be set. Other control bits in UCR1, UCR2 and BRG registers will remain unaffected. If the UART is active and the UARTEN bit is cleared, all pending transmissions and receptions will be terminated and the module will be reset as defined above. When the UART is re-enabled, it will restart in the same configuration.

Bit 6 **BNO**: Number of data transfer bits selection
0: 8-bit data transfer
1: 9-bit data transfer

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

Bit 5 **PREN**: Parity function enable control
0: Parity function is disabled
1: Parity function is enabled

This is the parity enable bit. When this bit is equal to "1", the parity function will be enabled. If the bit is equal to "0", then the parity function will be disabled. Replace the most significant bit position with a parity bit.

Bit 4 **PRT**: Parity type selection bit
0: Even parity for parity generator
1: Odd parity for parity generator

This bit is the parity type selection bit. When this bit is equal to "1", odd parity type will be selected. If the bit is equal to "0", then even parity type will be selected.

Bit 3 **STOPS**: Number of Stop bits selection
0: One stop bit format is used
1: Two stop bits format is used

This bit determines if one or two stop bits are to be used. When this bit is equal to "1", two stop bits format are used. If this bit is equal to "0", then only one stop bit format is used.

Bit 2 **TXBRK**: Transmit break character
0: No break character is transmitted
1: Break characters transmit

The TXBRK bit is the Transmit Break Character bit. When this bit is "0", there are no break characters and the TX pin operates normally. When the bit is "1", there are transmit break characters and the transmitter will send logic zeros. When this bit is equal to "1", after the buffered data has been transmitted, the transmitter output is held low for a minimum of a 13-bit length and until the TXBRK bit is reset.

Bit 1 **RX8**: Receive data bit 8 for 9-bit data transfer format (read only)

This bit is only used if 9-bit data transfers are used, in which case this bit location will store the 9th bit of the received data known as RX8. The BNO bit is used to determine whether data transfers are in 8-bit or 9-bit format.

Bit 0 **TX8**: Transmit data bit 8 for 9-bit data transfer format (write only)

This bit is only used if 9-bit data transfers are used, in which case this bit location will store the 9th bit of the transmitted data known as TX8. The BNO bit is used to determine whether data transfers are in 8-bit or 9-bit format.

UCR2 Register

The UCR2 register is the second of the two UART control registers and serves several purposes. One of its main functions is to control the basic enable/disable operation of the UART Transmitter and Receiver as well as enabling the various UART interrupt sources. The register also serves to control the baud rate speed, receiver wake-up enable and the address detect enable. Further explanation on each of the bits is given below:

Bit	7	6	5	4	3	2	1	0
Name	TXEN	RXEN	BRGH	ADDEN	WAKE	RIE	TIIE	TEIE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **TXEN**: UART Transmitter enabled control
0: UART transmitter is disabled
1: UART transmitter is enabled

The bit named TXEN is the Transmitter Enable Bit. When this bit is equal to "0", the transmitter will be disabled with any pending data transmissions being aborted. In addition the buffers will be reset. In this situation the TX pin will be used as an I/O or other pin-shared functional pin.

If the TXEN bit is equal to "1" and the UARTEN bit is also equal to "1", the transmitter will be enabled and the TX pin will be controlled by the UART. Clearing the TXEN bit during a transmission will cause the data transmission to be aborted and will reset the transmitter. If this situation occurs, the TX pin will be used as an I/O or other pin-shared functional pin.

Bit 6 **RXEN**: UART Receiver enabled control
0: UART receiver is disabled
1: UART receiver is enabled

The bit named RXEN is the Receiver Enable Bit. When this bit is equal to "0", the receiver will be disabled with any pending data receptions being aborted. In addition the receive buffers will be reset. In this situation the RX pin will be used as an I/O or other pin-shared functional pin. If the RXEN bit is equal to "1" and the UARTEN bit is also equal to "1", the receiver will be enabled and the RX pin will be controlled by the UART. Clearing the RXEN bit during a reception will cause the data reception to be aborted and will reset the receiver. If this situation occurs, the RX pin will be used as an I/O or other pin-shared functional pin.

Bit 5 **BRGH**: Baud Rate speed selection
0: Low speed baud rate
1: High speed baud rate

The bit named BRGH selects the high or low speed mode of the Baud Rate Generator. This bit, together with the value placed in the baud rate register BRG, controls the Baud Rate of the UART. If this bit is equal to "1", the high speed mode is selected. If the bit is equal to "0", the low speed mode is selected.

Bit 4 **ADDEN**: Address detect function enable control
0: Address detect function is disabled
1: Address detect function is enabled

The bit named ADDEN is the address detect function enable control bit. When this bit is equal to "1", the address detect function is enabled. When it occurs, if the 8th bit, which corresponds to RX7 if BNO=0 or the 9th bit, which corresponds to RX8 if BNO=1, has a value of "1", then the received word will be identified as an address, rather than data. If the corresponding interrupt is enabled, an interrupt request will be generated each time the received word has the address bit set, which is the 8th or 9th bit depending on the value of BNO. If the address bit known as the 8th or 9th bit of the received word is "0" with the address detect function being enabled, an interrupt will not be generated and the received data will be discarded.

Bit 3 **WAKE**: RX pin wake-up UART function enable control
0: RX pin wake-up UART function is disabled
1: RX pin wake-up UART function is enabled
This bit is used to control the wake-up UART function when a falling edge on the RX pin occurs. Note that this bit is only available when the UART clock (f_H) is switched off. There will be no RX pin wake-up UART function if the UART clock (f_H) exists. If the WAKE bit is set to 1 as the UART clock (f_H) is switched off, a UART wake-up request will be initiated when a falling edge on the RX pin occurs. When this request happens and the corresponding interrupt is enabled, an RX pin wake-up UART interrupt will be generated to inform the MCU to wake up the UART function by switching on the UART clock (f_H) via the application program. Otherwise, the UART function cannot resume even if there is a falling edge on the RX pin when the WAKE bit is cleared to 0.

Bit 2 **RIE**: Receiver interrupt enable control
0: Receiver related interrupt is disabled
1: Receiver related interrupt is enabled
This bit enables or disables the receiver interrupt. If this bit is equal to "1" and when the receiver overrun flag OERR or receive data available flag RXIF is set, the UART interrupt request flag will be set. If this bit is equal to "0", the UART interrupt request flag will not be influenced by the condition of the OERR or RXIF flags.

Bit 1 **TIIE**: Transmitter Idle interrupt enable control
0: Transmitter idle interrupt is disabled
1: Transmitter idle interrupt is enabled
This bit enables or disables the transmitter idle interrupt. If this bit is equal to "1" and when the transmitter idle flag TIDLE is set, due to a transmitter idle condition, the UART interrupt request flag will be set. If this bit is equal to "0", the UART interrupt request flag will not be influenced by the condition of the TIDLE flag.

Bit 0 **TEIE**: Transmitter Empty interrupt enable control
0: Transmitter empty interrupt is disabled
1: Transmitter empty interrupt is enabled
This bit enables or disables the transmitter empty interrupt. If this bit is equal to "1" and when the transmitter empty flag TXIF is set, due to a transmitter empty condition, the UART interrupt request flag will be set. If this bit is equal to "0", the UART interrupt request flag will not be influenced by the condition of the TXIF flag.

TXR_RXR Register

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

"x" unknown

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

BRG Register

Bit	7	6	5	4	3	2	1	0
Name	BRG7	BRG6	BRG5	BRG4	BRG3	BRG2	BRG1	BRG0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x" unknown

Bit 7~0 **BRG7~BRG0**: Baud Rate values
By programming the BRGH bit in UCR2 Register which allows selection of the related formula described above and programming the required value in the BRG register, the required baud rate can be setup.
Note: Baud rate= f_H / [64 × (N+1)] if BRGH=0.
Baud rate= f_H / [16 × (N+1)] if BRGH=1.

Baud Rate Generator

To setup the speed of the serial data communication, the UART function contains its own dedicated baud rate generator. The baud rate is controlled by its own internal free running 8-bit timer, the period of which is determined by two factors. The first of these is the value placed in the baud rate register BRG and the second is the value of the BRGH bit with the control register UCR2. The BRGH bit decides if the baud rate generator is to be used in a high speed mode or low speed mode, which in turn determines the formula that is used to calculate the baud rate. The value N in the BRG register which is used in the following baud rate calculation formula determines the division factor. Note that N is the decimal value placed in the BRG register and has a range of between 0 and 255.

UCR2 BRGH Bit	0	1
Baud Rate (BR)	f_H / [64 (N+1)]	f_H / [16 (N+1)]

By programming the BRGH bit which allows selection of the related formula and programming the required value in the BRG register, the required baud rate can be setup. Note that because the actual baud rate is determined using a discrete value, N, placed in the BRG register, there will be an error associated between the actual and requested value. The following example shows how the BRG register value N and the error value can be calculated.

Calculating the Baud Rate and Error Values

For a clock frequency of 4MHz, and with BRGH cleared to zero determine the BRG register value N, the actual baud rate and the error value for a desired baud rate of 4800.

From the above table the desired baud rate $BR=f_H / [64 (N+1)]$

Re-arranging this equation gives $N=[f_H / (BR\times64)] - 1$

Giving a value for $N=[4000000 / (4800\times64)] - 1=12.0208$

To obtain the closest value, a decimal value of 12 should be placed into the BRG register. This gives an actual or calculated baud rate value of $BR=4000000 / [64 \times (12+1)]=4808$

Therefore the error is equal to (4808 - 4800) / 4800=0.16%

UART Setup and Control

For data transfer, the UART function utilizes a non-return-to-zero, more commonly known as NRZ, format. This is composed of one start bit, eight or nine data bits, and one or two stop bits. Parity is supported by the UART hardware, and can be setup to be even, odd or no parity. For the most common data format, 8 data bits along with no parity and one stop bit, denoted as 8, N, 1, is used as the default setting, which is the setting at power-on. The number of data bits and stop bits, along with the parity, are setup by programming the corresponding BNO, PRT, PREN, and STOPS bits in the UCR1 register. The baud rate used to transmit and receive data is setup using the internal 8-bit baud rate generator, while the data is transmitted and received LSB first. Although the UART transmitter and receiver are functionally independent, they both use the same data format and baud rate. In all cases stop bits will be used for data transmission.

Enabling/Disabling the UART Interface

The basic on/off function of the internal UART function is controlled using the UARTEN bit in the UCR1 register. If the UARTEN, TXEN and RXEN bits are set, then these two UART pins will act as normal TX output pin and RX input pin respectively. If no data is being transmitted on the TX pin, then it will default to a logic high value.

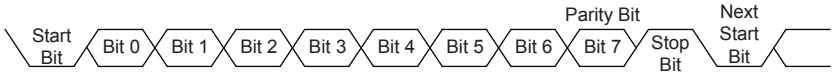
Clearing the UARTEN bit will disable the TX and RX pins and allow these two pins to be used as normal I/O or other pin-shared functional pins by configuring the corresponding pin-shared control bits. When the UART function is disabled the buffer will be reset to an empty condition, at the same time discarding any remaining residual data. Disabling the UART will also reset the error and status flags with bits TXEN, RXEN, TXBRK, RXIF, OERR, FERR, PERR and NF being cleared while bits TIDLE, TXIF and RIDLE will be set. The remaining control bits in the UCR1, UCR2 and BRG registers will remain unaffected. If the UARTEN bit in the UCR1 register is cleared while the UART is active, then all pending transmissions and receptions will be immediately suspended and the UART will be reset to a condition as defined above. If the UART is then subsequently re-enabled, it will restart again in the same configuration.

Data, Parity and Stop Bit Selection

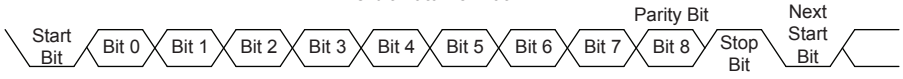
The format of the data to be transferred is composed of various factors such as data bit length, parity on/off, parity type, address bits and the number of stop bits. These factors are determined by the setup of various bits within the UCR1 register. The BNO bit controls the number of data bits which can be set to either 8 or 9, the PRT bit controls the choice of odd or even parity, the PREN bit controls the parity on/off function and the STOPS bit decides whether one or two stop bits are to be used. The following table shows various formats for data transmission. The address bit, which is the MSB of the data byte, identifies the frame as an address character or data if the address detect function is enabled. The number of stop bits, which can be either one or two, is independent of the data length and only used for the transmitter. There is only one stop bit for the receiver.

Start Bit	Data Bits	Address Bit	Parity Bit	Stop Bit
Example of 8-bit Data Formats				
1	8	0	0	1
1	7	0	1	1
1	7	1	0	1
Example of 9-bit Data Formats				
1	9	0	0	1
1	8	0	1	1
1	8	1	0	1

Transmitter Receiver Data FormatThe following diagram shows the transmit and receive waveforms for both 8-bit and 9-bit data formats.



8-bit Data Format



9-bit Data Format

UART Transmitter

Data word lengths of either 8 or 9 bits can be selected by programming the BNO bit in the UCR1 register. When BNO bit is set, the word length will be set to 9 bits. In this case the 9th bit, which is the MSB, needs to be stored in the TX8 bit in the UCR1 register. At the transmitter core lies the Transmitter Shift Register, more commonly known as the TSR, whose data is obtained from the transmit data register, which is known as the TXR_RXR register. The data to be transmitted is loaded into this TXR_RXR register by the application program. The TSR register is not written to with new data until the stop bit from the previous transmission has been sent out. As soon as this stop bit has been transmitted, the TSR can then be loaded with new data from the TXR_RXR register, if it is available. It should be noted that the TSR register, unlike many other registers, is not directly mapped into the Data Memory area and as such is not available to the application program for direct read/write operations. An actual transmission of data will normally be enabled when the TXEN bit is set, but the data will not be transmitted until the TXR_RXR register has been loaded with data and the baud rate generator has defined a shift clock source. However, the transmission can also be initiated by first loading data into the TXR_RXR register, after which the TXEN bit can be set. When a transmission of data begins, the TSR is normally empty, in which case a transfer to the TXR_RXR register will result in an immediate transfer to the TSR. If during a transmission the TXEN bit is cleared, the transmission will immediately cease and the transmitter will be reset. The TX output pin can then be configured as the I/O or other pin-shared function by configuring the corresponding pin-shared control bits.

Transmitting Data

When the UART is transmitting data, the data is shifted on the TX pin from the shift register, with the least significant bit first. In the transmit mode, the TXR_RXR register forms a buffer between the internal bus and the transmitter shift register. It should be noted that if 9-bit data format has been selected, then the MSB will be taken from the TX8 bit in the UCR1 register. The steps to initiate a data transfer can be summarized as follows:

- Make the correct selection of the BNO, PRT, PREN and STOPS bits to define the required word length, parity type and number of stop bits.
- Setup the BRG register to select the desired baud rate.
- Set the TXEN bit to ensure that the TX pin is used as a UART transmitter pin.
- Access the USR register and write the data that is to be transmitted into the TXR_RXR register. Note that this step will clear the TXIF bit.

This sequence of events can now be repeated to send additional data.

It should be noted that when TXIF=0, data will be inhibited from being written to the TXR_RXR register. Clearing the TXIF flag is always achieved using the following software sequence:

1. A USR register access

2. A TXR_RXR register write execution

The read-only TXIF flag is set by the UART hardware and if set indicates that the TXR_RXR register is empty and that other data can now be written into the TXR_RXR register without overwriting the previous data. If the TEIE bit is set then the TXIF flag will generate an interrupt.

During a data transmission, a write instruction to the TXR_RXR register will place the data into the TXR_RXR register, which will be copied to the shift register at the end of the present transmission. When there is no data transmission in progress, a write instruction to the TXR_RXR register will place the data directly into the shift register, resulting in the commencement of data transmission, and the TXIF bit being immediately set. When a frame transmission is complete, which happens after stop bits are sent or after the break frame, the TIDLE bit will be set. To clear the TIDLE bit the following software sequence is used:

1. A USR register access
2. A TXR_RXR register write execution

Note that both the TXIF and TIDLE bits are cleared by the same software sequence.

Transmit Break

If the TXBRK bit is set then break characters will be sent on the next transmission. Break character transmission consists of a start bit, followed by 13×N '0' bits and stop bits, where N=1, 2, etc. If a break character is to be transmitted then the TXBRK bit must be first set by the application program, and then cleared to generate the stop bits. Transmitting a break character will not generate a transmit interrupt. Note that a break condition length is at least 13 bits long. If the TXBRK bit is continually kept at a logic high level then the transmitter circuitry will transmit continuous break characters. After the application program has cleared the TXBRK bit, the transmitter will finish transmitting the last break character and subsequently send out one or two stop bits. The automatic logic high at the end of the last break character will ensure that the start bit of the next frame is recognized.

UART Receiver

The UART is capable of receiving word lengths of either 8 or 9 bits. If the BNO bit is set, the word length will be set to 9 bits with the MSB being stored in the RX8 bit of the UCR1 register. At the receiver core lies the Receive Serial Shift Register, commonly known as the RSR. The data which is received on the RX external input pin is sent to the data recovery block. The data recovery block operating speed is 16 times that of the baud rate, while the main receive serial shifter operates at the baud rate. After the RX pin is sampled for the stop bit, the received data in RSR is transferred to the receive data register, if the register is empty. The data which is received on the external RX input pin is sampled three times by a majority detect circuit to determine the logic level that has been placed onto the RX pin. It should be noted that the RSR register, unlike many other registers, is not directly mapped into the Data Memory area and as such is not available to the application program for direct read/write operations.

Receiving Data

When the UART receiver is receiving data, the data is serially shifted in on the external RX input pin, LSB first. In the read mode, the TXR_RXR register forms a buffer between the internal bus and the receiver shift register. The TXR_RXR register is a two byte deep FIFO data buffer, where two bytes can be held in the FIFO while a third byte can continue to be received. Note that the application program must ensure that the data is read from TXR_RXR before the third byte has been completely shifted in, otherwise this third byte will be discarded and an overrun error OERR will be subsequently indicated. The steps to initiate a data transfer can be summarized as follows:

- Make the correct selection of BNO, PRT and PREN bits to define the word length and parity type.
- Setup the BRG register to select the desired baud rate.
- Set the RXEN bit to ensure that the RX pin is used as a UART receiver pin.

At this point the receiver will be enabled which will begin to look for a start bit.

When a character is received the following sequence of events will occur:

- The RXIF bit in the USR register will be set when the TXR_RXR register has data available. There will be at most one more character available before an overrun error occurs.
- When the contents of the shift register have been transferred to the TXR_RXR register, then if the RIE bit is set, an interrupt will be generated.
- If during reception, a frame error, noise error, parity error, or an overrun error has been detected, then the error flags can be set.

The RXIF bit can be cleared using the following software sequence:

1. A USR register access
2. A TXR_RXR register read execution

Receive Break

Any break character received by the UART will be managed as a framing error. The receiver will count and expect a certain number of bit times as specified by the values programmed into the BNO bit plus one stop bit. If the break is much longer than 13 bit times, the reception will be considered as complete after the number of bit times specified by BNO and STOPS. The RXIF bit is set, FERR is set, zeros are loaded into the receive data register, interrupts are generated if appropriate and the RIDLE bit is set. A break is regarded as a character that contains only zeros with the FERR flag set. If a long break signal has been detected, the receiver will regard it as a data frame including a start bit, data bits and the invalid stop bit and the FERR flag will be set. The receiver must wait for a valid stop bit before looking for the next start bit. The receiver will not make the assumption that the break condition on the line is the next start bit. The break character will be loaded into the buffer and no further data will be received until stop bits are received. It should be noted that the RIDLE read only flag will go high when the stop bits have not yet been received. The reception of a break character on the UART registers will result in the following:

- The framing error flag, FERR, will be set.
- The receive data register, TXR_RXR, will be cleared.
- The OERR, NF, PERR, RIDLE or RXIF flags will possibly be set.

Idle Status

When the receiver is reading data, which means it will be in between the detection of a start bit and the reading of a stop bit, the receiver status flag in the USR register, otherwise known as the RIDLE flag, will have a zero value. In between the reception of a stop bit and the detection of the next start bit, the RIDLE flag will have a high value, which indicates the receiver is in an idle condition.

Receiver Interrupt

The read only receive interrupt flag RXIF in the USR register is set by an edge generated by the receiver. An interrupt is generated if RIE=1, when a word is transferred from the Receive Shift Register, RSR, to the Receive Data Register, TXR_RXR. An overrun error can also generate an interrupt if RIE=1.

Managing Receiver Errors

Several types of reception errors can occur within the UART module, the following section describes the various types and how they are managed by the UART.

Overrun Error – OERR

The TXR_RXR register is composed of a two byte deep FIFO data buffer, where two bytes can be held in the FIFO register, while a third byte can continue to be received. Before this third byte has been entirely shifted in, the data should be read from the TXR_RXR register. If this is not done, the overrun error flag OERR will be consequently indicated.

In the event of an overrun error occurring, the following will happen:

- The OERR flag in the USR register will be set.
- The TXR_RXR contents will not be lost.
- The shift register will be overwritten.
- An interrupt will be generated if the RIE bit is set.

The OERR flag can be cleared by an access to the USR register followed by a read to the TXR_RXR register.

Noise Error – NF

Over-sampling is used for data recovery to identify valid incoming data and noise. If noise is detected within a frame the following will occur:

- The read only noise flag, NF, in the USR register will be set on the rising edge of the RXIF bit.
- Data will be transferred from the Shift register to the TXR_RXR register.
- No interrupt will be generated. However this bit rises at the same time as the RXIF bit which itself generates an interrupt.

Note that the NF flag is reset by a USR register read operation followed by a TXR_RXR register read operation.

Framing Error – FERR

The read only framing error flag, FERR, in the USR register, is set if a zero is detected instead of stop bits. If two stop bits are selected, both stop bits must be high; otherwise the FERR flag will be set. The FERR flag and the received data will be recorded in the USR and TXR_RXR registers respectively, and the flag is cleared in any reset.

Parity Error – PERR

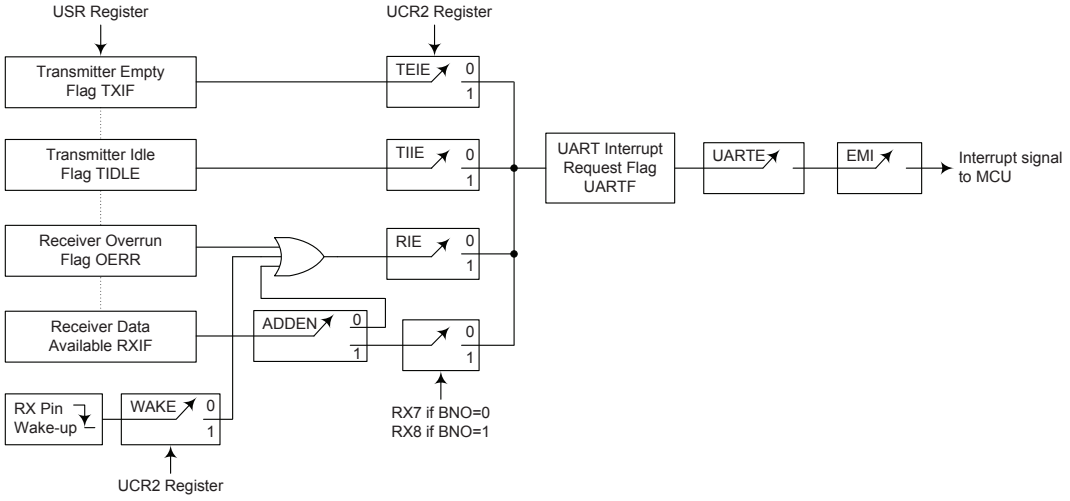
The read only parity error flag, PERR, in the USR register, is set if the parity of the received word is incorrect. This error flag is only applicable if the parity is enabled, PREN=1, and if the parity type, odd or even is selected. The read only PERR flag and the received data will be recorded in the USR and TXR_RXR registers respectively. It is cleared on any reset, it should be noted that the flags, FERR and PERR, in the USR register should first be read by the application program before reading the data word.

UART Interrupt Structure

Several individual UART conditions can generate a UART interrupt. When these conditions exist, a low pulse will be generated to get the attention of the microcontroller. These conditions are a transmitter data register empty, transmitter idle, receiver data available, receiver overrun, address detect and an RX pin wake-up. When any of these conditions are created, if the global interrupt enable bit, multi-function interrupt enable bit and its corresponding interrupt control bit are enabled and the stack is not full, the program will jump to its corresponding interrupt vector where it can be serviced before returning to the main program. Four of these conditions have the corresponding USR register flags which will generate a UART interrupt if its associated interrupt enable control bit in the UCR2 register is set. The two transmitter interrupt conditions have their own corresponding enable control bits, while the two receiver interrupt conditions have a shared enable control bit. These enable bits can be used to mask out individual UART interrupt sources.

The address detect condition, which is also a UART interrupt source, does not have an associated flag, but will generate a UART interrupt when an address detect condition occurs if its function is enabled by setting the ADDEN bit in the UCR2 register. An RX pin wake-up, which is also a UART interrupt source, does not have an associated flag, but will generate a UART interrupt if the UART clock (f_H) source is switched off and the WAKE and RIE bits in the UCR2 register are set when a falling edge on the RX pin occurs.

Note that the USR register flags are read only and cannot be cleared or set by the application program, neither will they be cleared when the program jumps to the corresponding interrupt servicing routine, as is the case for some of the other interrupts. The flags will be cleared automatically when certain actions are taken by the UART, the details of which are given in the UART register section. The overall UART interrupt can be disabled or enabled by the related interrupt enable control bits in the interrupt control registers of the microcontroller to decide whether the interrupt requested by the UART module is masked out or allowed.



UART Interrupt Structure

Address Detect Mode

Setting the Address Detect Mode bit, ADDEN, in the UCR2 register, enables this special mode. If this bit is enabled then an additional qualifier will be placed on the generation of a Receiver Data Available interrupt, which is requested by the RXIF flag. If the ADDEN bit is enabled, then when data is available, an interrupt will only be generated, if the highest received bit has a high value. Note that the UARTE and EMI interrupt enable bits must also be enabled for correct interrupt generation. This highest address bit is the 9th bit if BNO=1 or the 8th bit if BNO=0. If this bit is high, then the received word will be defined as an address rather than data. A Data Available interrupt will be generated every time the last bit of the received word is set. If the ADDEN bit is not enabled, then a Receiver Data Available interrupt will be generated each time the RXIF flag is set, irrespective of the data last bit status. The address detect mode and parity enable are mutually exclusive functions. Therefore if the address detect mode is enabled, then to ensure correct operation, the parity function should be disabled by resetting the parity enable bit PREN to zero.

ADDEN	Bit 9 if BNO=1, Bit 8 if BNO=0	UART Interrupt Generated
0	0	√
	1	√
1	0	×
	1	√

ADDEN Bit Function

UART Power Down and Wake-up

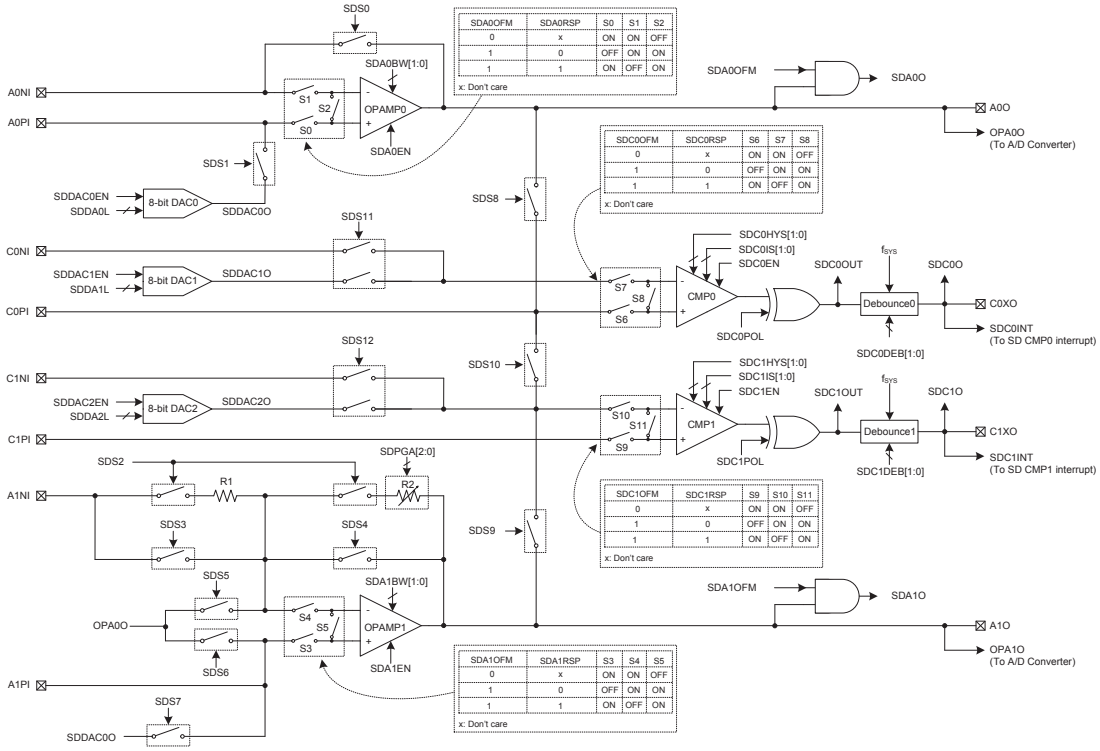
When the UART clock (f_H) is off, the UART will cease to function, all clock sources to the module are shutdown. If the UART clock (f_H) is off while a transmission is still in progress, then the transmission will be paused until the UART clock source derived from the microcontroller is activated. In a similar way, if the MCU enters the Power Down Mode while receiving data, then the reception of data will likewise be paused. When the MCU enters the Power Down Mode, note that the USR, UCR1, UCR2, transmit and receive registers, as well as the BRG register will not be affected. It is recommended to make sure first that the UART data transmission or reception has been finished before the microcontroller enters the Power Down mode.

The UART function contains a receiver RX pin wake-up function, which is enabled or disabled by the WAKE bit in the UCR2 register. If this bit, along with the UART enable bit, UARTEN, the receiver enable bit, RXEN and the receiver interrupt bit, RIE, are all set when the UART clock (f_H) is off, then a falling edge on the RX pin will trigger an RX pin wake-up UART interrupt. Note that as it takes certain system clock cycles after a wake-up, before normal microcontroller operation resumes, any data received during this time on the RX pin will be ignored.

For a UART wake-up interrupt to occur, in addition to the bits for the wake-up being set, the global interrupt enable bit, EMI, and the UART interrupt enable bit, UARTE, must be set. If the EMI and UARTE bits are not set then only a wake up event will occur and no interrupt will be generated. Note also that as it takes certain system clock cycles after a wake-up before normal microcontroller resumes, the UART interrupt will not be generated until after this time has elapsed.

Operational Amplifiers and Comparators

These devices include two fully integrated Operational Amplifiers and two Comparators as well as three 8-bit D/A Converters, which can be used for applications which need to interface to external sensors. The following block diagram illustrates these internal analog functions.



Internal Analog Function Block Diagram

Analog Function Operation

The Operational Amplifiers can be used for signal amplification according to specific user requirements. The 8-bit D/A Converters can provide an accurate reference voltage to the non-inverting input of Operational Amplifiers and the inverting input of Comparators. The comparators are used to compare two analog voltages and provide an output based on their difference and generate an interrupt if the corresponding interrupt control is enabled.

The advantages of multiple switches and input path options, various reference voltage selections, many kinds of internal software gain control, debounce time control, hysteresis, offset reference voltage calibration function and others enhance the flexibility of this circuit to suit a wide range of application possibilities.

Control Registers

Overall operation of the analog circuits is controlled using a series of registers. The SDSW0~SDSW1 registers are used to control the switch on or off and SD Comparator n inverting terminal connection. The SDDAC0C~SDDAC2C registers are used to control the SD DACn enable/disable function. The SDDA0L~SDDA2L registers are used to control the SD DACn output voltage. The SDPAGC register is used to control the PGA R2/R1 rate and SD Comparator n output polarity. The SDA0C~SDA1C registers are used to control the SD OPAn enable/disable and bandwidth functions as well as indicate their output status. The SDA0VOS~SDA1VOS registers are used to control the SD OPAn input offset voltage calibration function. The SDC0C~SDC1C registers are used to control the SD Comparator n enable/disable and debounce time as well as indicate their output status. The SDC0VOS~SDC1VOS are used to control the SD Comparator n input offset voltage calibration function. The SDCHYC register is used to control the SD Comparator n hysteresis function.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SDSW0	SDS7	SDS6	SDS5	SDS4	SDS3	SDS2	SDS1	SDS0
SDSW1	—	—	—	SDS12	SDS11	SDS10	SDS9	SDS8
SDDAC0C	SDDAC0EN	—	—	—	—	—	—	—
SDDAC1C	SDDAC1EN	—	—	—	—	—	—	—
SDDAC2C	SDDAC2EN	—	—	—	—	—	—	—
SDDA0L	D7	D6	D5	D4	D3	D2	D1	D0
SDDA1L	D7	D6	D5	D4	D3	D2	D1	D0
SDDA2L	D7	D6	D5	D4	D3	D2	D1	D0
SDPGAC	SDC0POL	SDC1POL	—	—	—	SDPGA2	SDPGA1	SDPGA0
SDA0C	—	SDA0EN	SDA0O	—	—	—	SDA0BW1	SDA0BW0
SDA1C	—	SDA1EN	SDA1O	—	—	—	SDA1BW1	SDA1BW0
SDA0VOS	SDA0OFM	SDA0RSP	SDA0OF5	SDA0OF4	SDA0OF3	SDA0OF2	SDA0OF1	SDA0OF0
SDA1VOS	SDA1OFM	SDA1RSP	SDA1OF5	SDA1OF4	SDA1OF3	SDA1OF2	SDA1OF1	SDA1OF0
SDC0C	SDC0OUT	SDC0EN	SDC0O	—	SDC0DEB1	SDC0DEB0	SDC0IS1	SDC0IS0
SDC1C	SDC1OUT	SDC1EN	SDC1O	—	SDC1DEB1	SDC1DEB0	SDC1IS1	SDC1IS0
SDC0VOS	—	SDC0OFM	SDC0RSP	SDC0OF4	SDC0OF3	SDC0OF2	SDC0OF1	SDC0OF0
SDC1VOS	—	SDC1OFM	SDC1RSP	SDC1OF4	SDC1OF3	SDC1OF2	SDC1OF1	SDC1OF0
SDCHYC	—	—	—	—	SDC1HYS1	SDC1HYS0	SDC0HYS1	SDC0HYS0

Analog Function Control Registers List

SDSW0 Register

Bit	7	6	5	4	3	2	1	0
Name	SDS7	SDS6	SDS5	SDS4	SDS3	SDS2	SDS1	SDS0
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 **SDS7**: Switch on or off control
0: Off
1: On

Bit 6 **SDS6**: Switch on or off control
0: Off
1: On

Bit 5 **SDS5**: Switch on or off control
0: Off
1: On

Bit 4 **SDS4**: Switch on or off control
0: Off
1: On

Bit 3 **SDS3**: Switch on or off control
0: Off
1: On

Bit 2 **SDS2**: Switch on or off control
0: Off
1: On

Bit 1 **SDS1**: Switch on or off control
0: Off
1: On

Bit 0 **SDS0**: Switch on or off control
0: Off
1: On

SDSW1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	SDS12	SDS11	SDS10	SDS9	SDS8
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	0	0	0	0	0

Bit 7~5 Unimplemented, read as "0"

Bit 4 **SDS12**: SD Comparator 1 inverting terminal connect selection
0: Connect to SD DAC2
1: Connect to C1NI

Bit 3 **SDS11**: SD Comparator 0 inverting terminal connect selection
0: Connect to SD DAC1
1: Connect to C0NI

Bit 2 **SDS10**: Switch on or off control
0: Off
1: On

Bit 1 **SDS9**: Switch on or off control
0: Off
1: On

Bit 0 **SDS8**: Switch on or off control
0: Off
1: On

SDDAC0C Register

Bit	7	6	5	4	3	2	1	0
Name	SDDAC0EN	—	—	—	—	—	—	—
R/W	R/W	—	—	—	—	—	—	—
POR	0	—	—	—	—	—	—	—

Bit 7 **SDDAC0EN**: SD DAC0 enable or disable control
0: Disable
1: Enable

Bit 6~0 Unimplemented, read as "0"

SDDAC1C Register

Bit	7	6	5	4	3	2	1	0
Name	SDDAC1EN	—	—	—	—	—	—	—
R/W	R/W	—	—	—	—	—	—	—
POR	0	—	—	—	—	—	—	—

Bit 7 **SDDAC1EN**: SD DAC1 enable or disable control
0: Disable
1: Enable

Bit 6~0 Unimplemented, read as "0"

SDDAC2C Register

Bit	7	6	5	4	3	2	1	0
Name	SDDAC2EN	—	—	—	—	—	—	—
R/W	R/W	—	—	—	—	—	—	—
POR	0	—	—	—	—	—	—	—

Bit 7 **SDDAC2EN**: SD DAC2 enable or disable control
0: Disable
1: Enable

Bit 6~0 Unimplemented, read as "0"

SDDA0L Register

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

Bit 7~0 **D7~D0**: SD DAC0 output control code
SDDAC0O= (DAC $AV_{DD}/2^8$) × D[7:0]

SDDA1L Register

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

Bit 7~0 **D7~D0**: SD DAC1 output control code
SDDAC1O= (DAC $AV_{DD}/2^8$) × D[7:0]

SDDA2L Register

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

Bit 7~0 **D7~D0**: SD DAC2 output control code
SDDAC2O= (DAC $AV_{DD}/2^8$) × D[7:0]

SDPGAC Register

Bit	7	6	5	4	3	2	1	0
Name	SDC0POL	SDC1POL	—	—	—	SDPGA2	SDPGA1	SDPGA0
R/W	R/W	R/W	—	—	—	R/W	R/W	R/W
POR	0	0	—	—	—	0	0	0

Bit 7 **SDC0POL**: SD Comparator 0 output polarity control
0: Output not inverted
1: Output inverted
This is the SD Comparator 0 polarity bit. If the bit is zero then the SDC0OUT bit will reflect the non-inverted output condition of the comparator. If the bit is high the comparator SDC0OUT bit will be inverted.

Bit 6 **SDC1POL**: SD Comparator 1 output polarity control
0: Output not inverted
1: Output inverted
This is the SD Comparator 1 polarity bit. If the bit is zero then the SDC1OUT bit will reflect the non-inverted output condition of the comparator. If the bit is high the comparator SDC1OUT bit will be inverted.

Bit 5~3 Unimplemented, read as "0"

Bit 2~0 **SDPGA2~SDPGA0**: PGA R2/R1 rate control
000: ×1
001: ×2
010: ×4
011: ×8
100: ×16
101: ×32
110: ×56
111: ×1

SDA0C Register

Bit	7	6	5	4	3	2	1	0
Name	—	SDA0EN	SDA0O	—	—	—	SDA0BW1	SDA0BW0
R/W	—	R/W	R	—	—	—	R/W	R/W
POR	—	0	0	—	—	—	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **SDA0EN**: SD OPA0 enable or disable control
0: Disable
1: Enable

Bit 5 **SDA0O**: SD OPA0 output status (positive logic)
This bit is read only.
When SDA0OFM bit is set to 1, SDA0O is defined as SD OPA0 output status, refer to the "Operational Amplifier Input Offset Calibration" section for the detailed offset calibration procedures.
When SDA0OFM bit is cleared to 0, this bit will be fixed at a low level.

Bit 4~2 Unimplemented, read as "0"

Bit 1~0 **SDA0BW1~SDA0BW0**: SD OPA0 bandwidth control bits
00: 5kHz
01: 40kHz
10: 600kHz
11: 2MHz
Refer to "Operational Amplifier Electrical Characteristics" for details.

SDA1C Register

Bit	7	6	5	4	3	2	1	0
Name	—	SDA1EN	SDA1O	—	—	—	SDA1BW1	SDA1BW0
R/W	—	R/W	R	—	—	—	R/W	R/W
POR	—	0	0	—	—	—	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **SDA1EN**: SD OPA1 enable or disable control
0: Disable
1: Enable

Bit 5 **SDA1O**: SD OPA1 output status (positive logic)
This bit is read only.
When SDA1OFM bit is set to 1, SDA1O is defined as SD OPA1 output status, refer to the "Operational Amplifier Input Offset Calibration" section for the detailed offset calibration procedures.
When SDA1OFM bit is cleared to 0, this bit will be fixed at a low level.

Bit 4~2 Unimplemented, read as "0"

Bit 1~0 **SDA1BW1~SDA1BW0**: SD OPA1 bandwidth control bits
00: 5kHz
01: 40kHz
10: 600kHz
11: 2MHz
Refer to "Operational Amplifier Electrical Characteristics" for details.

SDA0VOS Register

Bit	7	6	5	4	3	2	1	0
Name	SDA0OFM	SDA0RSP	SDA0OF5	SDA0OF4	SDA0OF3	SDA0OF2	SDA0OF1	SDA0OF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	0	0	0	0	0

Bit 7 **SDA0OFM**: SD OPA0 normal operation or input offset voltage cancellation mode selection
0: Normal operation
1: Offset calibration mode

Bit 6 **SDA0RSP**: SD OPA0 input offset voltage calibration reference selection
0: Input reference voltage comes from A0NI
1: Input reference voltage comes from A0PI

Bit 5~0 **SDA0OF5~SDA0OF0**: SD OPA0 input offset voltage calibration control
This 6-bit field is used to perform the operational amplifier input offset calibration operation and the value for the SD OPA0 input offset Calibration can be restored into this bit field. More detailed information is described in the "Operational Amplifier Input Offset Calibration" section.

SDA1VOS Register

Bit	7	6	5	4	3	2	1	0
Name	SDA1OFM	SDA1RSP	SDA1OF5	SDA1OF4	SDA1OF3	SDA1OF2	SDA1OF1	SDA1OF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	0	0	0	0	0

Bit 7 **SDA1OFM**: SD OPA1 normal operation or input offset voltage cancellation mode selection
0: Normal operation
1: Offset calibration mode

Bit 6 **SDA1RSP**: SD OPA1 input offset voltage calibration reference selection
0: Input reference voltage comes from A1NI
1: Input reference voltage comes from A1PI

Bit 5~0 **SDA1OF5~SDA1OF0**: SD OPA1 input offset voltage calibration control
This 6-bit field is used to perform the operational amplifier input offset Calibration operation and the value for the SD OPA1 input offset Calibration can be restored into this bit field. More detailed information is described in the "Operational Amplifier Input Offset Calibration" section.

SDC0C Register

Bit	7	6	5	4	3	2	1	0
Name	SDC0OUT	SDC0EN	SDC0O	—	SDC0DEB1	SDC0DEB0	SDC0IS1	SDC0IS0
R/W	R	R/W	R	—	R/W	R/W	R/W	R/W
POR	0	0	0	—	0	0	0	0

Bit 7 **SDC0OUT**: SD Comparator 0 output bit
If SDC0POL=0,
0: C0NI > C0PI
1: C0PI > C0NI
If SDC0POL=1,
0: C0NI < C0PI
1: C0PI < C0NI
This bit stores the SD Comparator 0 output bit. The polarity of the bit is determined by the voltages on the SD Comparator 0 inputs and by the condition of the SDC0POL bit.

Bit 6 **SDC0EN**: SD Comparator 0 enable or disable control
0: Comparator disable
1: Comparator enable
This is the SD Comparator 0 on/off control bit. The SD Comparator 0 output will be set low when this bit is cleared to zero. Therefore, set SDC0OUT=0 when SDC0POL=0, or set SDC0OUT=1 when SDC0POL=1.

Bit 5 **SDC0O**: SD Comparator 0 debounce output
The SDC0O is de-bounce version of SDC0OUT
If SDC0POL=0, The SDC0O outputs "1" only when the current and previous N samples of SDC0OUT are "1". If SDC0POL=1, The SDC0O outputs "0" only when the current and previous N samples of SDC0OUT are "0". N samples is depend on SDC0DEB[1:0] configuration bits.

Bit 4 Unimplemented, read as "0"

Bit 3~2 **SDC0DEB1~SDC0DEB0**: SD Comparator 0 debounce time control
00: No debounce
01: (31~32) × 1/ f_{SYS}
10: (63~64) × 1/ f_{SYS}
11: (126~127) × 1/ f_{SYS}

Bit 1~0 **SDC0IS1~SDC0IS0**: SD Comparator 0 current control
Refer to Comparator characteristic for details.

SDC1C Register

Bit	7	6	5	4	3	2	1	0
Name	SDC1OUT	SDC1EN	SDC1O	—	SDC1DEB1	SDC1DEB0	SDC1IS1	SDC1IS0
R/W	R	R/W	R	—	R/W	R/W	R/W	R/W
POR	0	0	0	—	0	0	0	0

Bit 7 **SDC1OUT**: SD Comparator 1 output bit
If SDC1POL=0,
0: C1NI > C1PI
1: C1PI > C1NI
If SDC1POL=1,
0: C1NI < C1PI
1: C1PI < C1NI
This bit stores the SD Comparator 1 output bit. The polarity of the bit is determined by the voltages on the SD Comparator 1 inputs and by the condition of the SDC1POL bit.

Bit 6 **SDC1EN**: SD Comparator 1 enable or disable control
0: Comparator disable
1: Comparator enable
This is the SD Comparator 1 on/off control bit. The SD Comparator 1 output will be set low when this bit is cleared to zero. Therefore, set SDC1OUT=0 when SDC1POL=0, or set SDC1OUT=1 when SDC1POL=1.

Bit 5 **SDC1O**: SD Comparator 1 debounce output
The SDC1O is de-bounce version of SDC1OUT.
If SDC1POL=0, the SDC1O outputs "1" only when the current and previous N samples of SDC1OUT are "1". If SDC1POL=1, the SDC1O outputs "0" only when the current and previous N samples of SDC1OUT are "0". N samples is depend on SDC1DEB[1:0] configuration bits.

Bit 4 Unimplemented, read as "0"

Bit 3~2 **SDC1DEB1~SDC1DEB0**: SD Comparator 1 debounce time control
00: No debounce
01: (31~32) × 1/ f_{SYS}
10: (63~64) × 1/ f_{SYS}
11: (126~127) × 1/ f_{SYS}

Bit 1~0 **SDC1IS1~SDC1IS0**: SD Comparator 1 current control
Refer to Comparator characteristic for details.

SDC0VOS Register

Bit	7	6	5	4	3	2	1	0
Name	—	SDC0OFM	SDC0RSP	SDC0OF4	SDC0OF3	SDC0OF2	SDC0OF1	SDC0OF0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	1	0	0	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **SDC0OFM**: SD Comparator 0 normal operation or input offset voltage cancellation mode selection
0: Normal operation
1: Offset calibration mode

Bit 5 **SDC0RSP**: SD Comparator 0 input offset voltage calibration reference selection
0: Input reference voltage comes from C0NI
1: Input reference voltage comes from C0PI

Bit 4~0 **SDC0OF4~SDC0OF0**: SD Comparator 0 input offset voltage calibration control
This 5-bit field is used to perform the comparator input offset calibration operation and the value for the SD Comparator 0 input offset calibration can be restored into this bit field. More detailed information is described in the "Comparator Input Offset Calibration" section.

SDC1VOS Register

Bit	7	6	5	4	3	2	1	0
Name	—	SDC1OFM	SDC1RSP	SDC1OF4	SDC1OF3	SDC1OF2	SDC1OF1	SDC1OF0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	1	0	0	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **SDC1OFM**: SD Comparator 1 normal operation or input offset voltage cancellation mode selection
0: Normal operation
1: Offset calibration mode

Bit 5 **SDC1RSP**: SD Comparator 1 input offset voltage calibration reference selection
0: Input reference voltage comes from C1NI
1: Input reference voltage comes from C1PI

Bit 4~0 **SDC1OF4~SDC1OF0**: SD Comparator 1 input offset voltage calibration control
This 5-bit field is used to perform the comparator input offset calibration operation and the value for the SD Comparator 1 input offset calibration can be restored into this bit field. More detailed information is described in the "Comparator Input Offset Calibration" section.

SDCHYC Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	SDC1HYS1	SDC1HYS0	SDC0HYS1	SDC0HYS0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4 Unimplemented, read as "0"

Bit 3~2 **SDC1HYS1~SDC1HYS0**: Comparator 1 hysteresis voltage window control
Refer to "Comparator Electrical Characteristics" for details.

Bit 1~0 **SDC0HYS1~SDC0HYS0**: Comparator 0 hysteresis voltage window control
Refer to "Comparator Electrical Characteristics" for details.

Offset Calibration Procedure

To operate in the input offset calibration mode for the SD Operational Amplifier n or the SD Comparator n, the SDAnOFM or SDCnOFM bit should first be set to "1" followed by the reference input selection by configuring the SDAnRSP or SDCnRSP bit. Note that if the SD Operational Amplifier n or the SD Comparator n inputs are pin-shared with I/O pins, they should be configured as the SD Operational Amplifier n inputs or the SD Comparator n inputs first.

Operational Amplifier Input Offset Calibration

Step 1. Set SDAnOFM=1 and SDAnRSP=1, the SD Operational Amplifier n will operate in the operational amplifier input offset Calibration mode, S0 and S2 on or S3 and S5 on. To make sure V_{AnOS} as minimize as possible after calibration, the input reference voltage in calibration should be the same as input DC operating voltage in normal operation.

Step 2. Set SDAnOF[5:0]=000000 and then read the SDAnO bit.

Step 3. Increase the SDAnOF[5:0] value by 1 and then read the SDAnO bit.
If the SDAnO bit state has not changed, then repeat Step 3 until the SDAnO bit state has changed.
If the SDAnO bit state has changed, record the SDAnOF[5:0] value as V_{AnOS1} and then go to Step 4.

Step 4. Set SDAnOF[5:0]=111111 and read the SDAnO bit.

Step 5. Decrease the SDAnOF[5:0] value by 1 and then read the SDAnO bit.
If the SDAnO bit state has not changed, then repeat Step 5 until the SDAnO bit state has changed.
If the SDAnO bit state has changed, record the SDAnOF[5:0] value as V_{AnOS2} and then go to Step 6.

Step 6. Restore the SD Operational Amplifier n input offset calibration value V_{AnOS} into the SDAnOF[5:0] bit field. The offset Calibration procedure is now finished.
Where $V_{AnOS}=(V_{AnOS1}+V_{AnOS2})/2$. If $(V_{AnOS1}+V_{AnOS2})/2$ is not integral, discard the decimal.

Comparator Input Offset Calibration

Step 1. Set SDCnOFM=1 and SDCnRSP=1, the SD Comparator n will now operate in the comparator input offset calibration mode, S6 and S8 on or S9 and S11 on. To make sure V_{CnOS} as minimize as possible after calibration, the input reference voltage in calibration should be the same as input DC operating voltage in normal operation.

Step 2. Set SDCnOF[4:0]=00000 and read the SDCnOUT bit.

Step 3. Increase the SDCnOF[4:0] value by 1 and then read the SDCnOUT bit.
If the SDCnOUT bit state has not changed, then repeat Step 3 until the SDCnOUT bit state has changed.
If the SDCnOUT bit state has changed, record the SDCnOF[4:0] value as V_{CnOS1} and then go to Step 4.

Step 4. Set SDCnOF[4:0]=11111 and then read the SDCnOUT bit.

Step 5. Decrease the SDCnOF[4:0] value by 1 and then read the SDCnOUT bit.
If the SDCnOUT bit state has not changed, then repeat Step 5 until the SDCnOUT bit state has changed.
If the SDCnOUT bit state has changed, record the SDCnOF[4:0] value as V_{CnOS2} and then go to Step 6.

Step 6. Restore the SD Comparator n input offset calibration value V_{CnOS} into the SDCnOF[4:0] bit field. The offset Calibration procedure is now finished.
Where $V_{CnOS}=(V_{CnOS1}+V_{CnOS2})/2$. If $(V_{CnOS1}+V_{CnOS2})/2$ is not integral, discard the decimal.

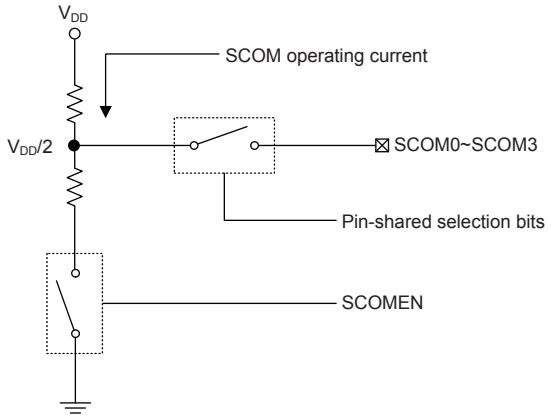
SCOM Function for LCD – HT66F4540/HT66F4550

These devices have the capability of driving external LCD panels. The common pins for LCD driving, SCOM0~SCOM3, are pin shared with certain pin on the I/O ports. The LCD signals (COM and SEG) are generated using the application program.

LCD Operation

An external LCD panel can be driven using these devices by configuring the I/O pins as common pins and 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 Current Control

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

SCOMC Register

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

Bit 7 Unimplemented, read as "0"

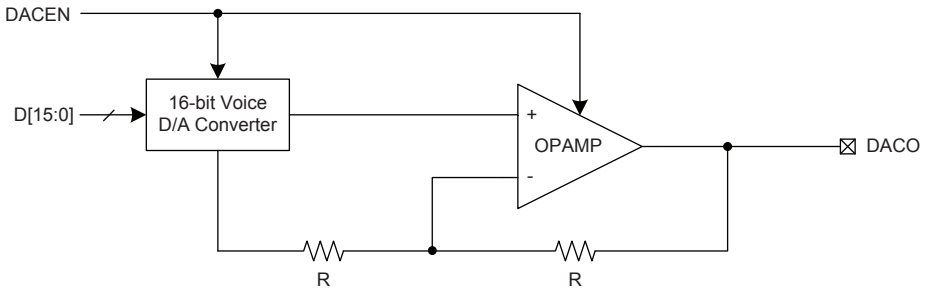
Bit 6~5 **ISEL1~ISEL0**: Select resistor for R type LCD bias current (V_{DD}=5V)
00: 2×100kΩ (1/2 Bias), I_{BIAS}=25μA
01: 2×50kΩ (1/2 Bias), I_{BIAS}=50μA
10: 2×25kΩ (1/2 Bias), I_{BIAS}=100μA
11: 2×12.5kΩ (1/2 Bias), I_{BIAS}=200μA

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

Bit 3~0 Unimplemented, read as "0"

Voice D/A Converter – HT66F4550

The HT66F4550 device contains a fully integrated 16-bit Voice D/A converter circuit complete with buffer output together with an operational amplifier, which can be used for audio application. Its reference comes from analog supply only, and can be power down to save power. The 16-bit D/A Converter is good for voice or audio application. Although this D/A Converter is not general one-to-one digital to analog conversion, it provides not bad and same audio quality no matter what small or big voice. Finally, D/A Converter voltage is amplified and buffer output by the operational amplifier.



Voice D/A Converter Block Diagram

Voice D/A Converter Control Registers

The overall voice D/A Converter function is controlled using several registers. The DACC register is used to control the D/A Converter function enable or disable. The DAH and DAL register pair exists to store a 16-bit wide voice data.

DAH Register

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

Bit 7~0 **D15~D8**: 16-bit Voice D/A Converter data high byte

The 16-bit Voice D/A Converter Data low byte register, known as DAL, should first be modified and then followed by the DAH register modification. Each time when the DAH register is written, the whole 16-bit data will be loaded into the D/A converter and a conversion cycle will be initiated. Note that the D/A converter should first be enabled before the D/A data is updated.

DAL Register

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

Bit 7~0 **D7~D0**: 16-bit Voice D/A Converter data low byte

Writing this register will only write the data to the shadow buffer and writing the DAH register will simultaneously copy the shadow buffer data to the DAL register. Note that the D/A converter should first be enabled before the D/A data is updated.

DACC Register

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

Bit 7~1 Unimplemented, read as "0"

Bit 0 **DACEN**: 16-bit Voice D/A Converter enable or disable control
0: Disable
1: Enable
If the D/A Converter is enabled, user must wait t_{DACS} to ensure the D/A Converter circuit is stable. A time tDACS should be allowed for the D/A Converter circuit to stabilize. And the 16-bit Voice D/A Converter data register should be updated after the D/A Converter circuit is stable.

Low Voltage Detector – LVD

These devices have a Low Voltage Detector function, also known as LVD. This enabled these devices 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	VBGEN	VLVD2	VLVD1	VLVD0
R/W	—	—	R	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 **LVDO**: LVD Output Flag
0: No Low Voltage Detect
1: Low Voltage Detect

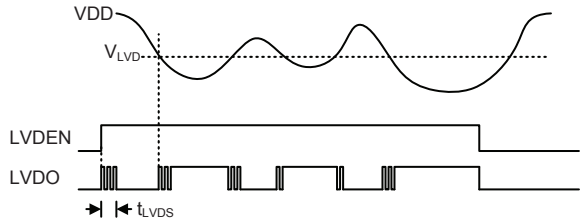
Bit 4 **LVDEN**: Low Voltage Detector Control
0: Disable
1: Enable

Bit 3 **VBGEN**: Bandgap Buffer Control
0: Disable
1: Enable
Note that the Bandgap circuit is enabled when the LVD or LVR function is enabled or when the VBGEN bit is set to 1.

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 these devices are in the SLEEP mode, the low voltage detector will be disabled even if the LVDEN bit is high. After enabling the Low Voltage Detector, a time delay t_{LVDS} should be allowed for the circuitry to stabilise before reading the LVDO bit. Note also that as the V_{DD} voltage may rise and fall rather slowly, at the voltage nears that of V_{LVD}, there may be multiple bit LVDO transitions.



LVD Operation

The Low Voltage Detector also has its own interrupt, providing an alternative means of low voltage detection, in addition to polling the LVDO bit. The interrupt will only be generated after a delay of t_{LVD} after the LVDO bit has been set high by a low voltage condition. In this case, the LVF interrupt request flag will be set, causing an interrupt to be generated if V_{DD} falls below the preset LVD voltage. This will cause these devices 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 these devices enter the SLEEP or IDLE Mode.

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~INT1 pins, while the internal interrupts are generated by various internal functions such as the TMs, Time Bases, LVD, EEPROM and the A/D converter.

Interrupt Registers

Overall interrupt control, which basically means the setting of request flags when certain microcontroller conditions occur and the setting of interrupt enable bits by the application program, is controlled by a series of registers, located in the Special Purpose Data Memory. The number of registers depends upon the selected device but fall into three categories. The first is the INTC0~INTC3 registers which setup the primary interrupts, the second is the MFI0~MFI1 registers which setup the Multi-function interrupts. Finally there is an INTEG register to setup the external interrupt trigger edge type.

Each register contains a number of enable bits to enable or disable individual registers as well as interrupt flags to indicate the presence of an interrupt request. The naming convention of these follows a specific pattern. First is listed an abbreviated interrupt type, then the (optional) number of that interrupt followed by either an "E" for enable/disable bit or "F" for request flag.

Function	Enable Bit	Request Flag	Notes
Global	EMI	—	—
INTn Pin	INTnE	INTnF	n=0~1
SD Comparator	SDCnE	SDCnF	n=0~1
Multi-function	MFnE	MFnF	n=0~1
A/D Converter	ADE	ADF	—
Time Base	TBnE	TBnF	n=0~1
SIM	SIME	SIMF	—
LVD	LVE	LVF	—
EEPROM	DEE	DEF	—
STMn	STMnPE	STMnPF	n=0
	STMnAE	STMnAF	
PTMn	PTMnPE	PTMnPF	n=0
	PTMnAE	PTMnAF	

Interrupt Register Bit Naming Conventions – HT66F4530

Function	Enable Bit	Request Flag	Notes
Global	EMI	—	—
INTn Pin	INTnE	INTnF	n=0~1
SD Comparator	SDCnE	SDCnF	n=0~1
Multi-function	MFnE	MFnF	n=0~1
A/D Converter	ADE	ADF	—
Time Base	TBnE	TBnF	n=0~1
SIM	SIME	SIMF	—
LVD	LVE	LVF	—
EEPROM	DEE	DEF	—
UART	UARTE	UARTF	—
STMn	STMnPE	STMnPF	n=0
	STMnAE	STMnAF	
PTMn	PTMnPE	PTMnPF	n=0~1
	PTMnAE	PTMnAF	

Interrupt Register Bit Naming Conventions – HT66F4540

Function	Enable Bit	Request Flag	Notes
Global	EMI	—	—
INTn Pin	INTnE	INTnF	n=0~1
SD Comparator	SDCnE	SDCnF	n=0~1
Multi-function	MFnE	MFnF	n=0~1
A/D Converter	ADE	ADF	—
Time Base	TBnE	TBnF	n=0~1
SIM	SIME	SIMF	—
LVD	LVE	LVF	—
EEPROM	DEE	DEF	—
UART	UARTE	UARTF	—
STMn	STMnPE	STMnPF	n=0~1
	STMnAE	STMnAF	
PTMn	PTMnPE	PTMnPF	n=0~1
	PTMnAE	PTMnAF	

Interrupt Register Bit Naming Conventions – HT66F4550

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	MF0F	SDC0F	INT0F	MF0E	SDC0E	INT0E	EMI
INTC1	TB1F	TB0F	ADF	MF1F	TB1E	TB0E	ADE	MF1E
INTC2	LVF	SDC1F	SIMF	INT1F	LVE	SDC1E	SIME	INT1E
INTC3	—	—	—	DEF	—	—	—	DEE
MFI0	—	—	PTM0AF	PTM0PF	—	—	PTM0AE	PTM0PE
MFI1	—	—	STM0AF	STM0PF	—	—	STM0AE	STM0PE

Interrupt Registers List – HT66F4530

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	MF0F	SDC0F	INT0F	MF0E	SDC0E	INT0E	EMI
INTC1	TB1F	TB0F	ADF	MF1F	TB1E	TB0E	ADE	MF1E
INTC2	LVF	SDC1F	SIMF	INT1F	LVE	SDC1E	SIME	INT1E
INTC3	—	—	UARTF	DEF	—	—	UARTE	DEE
MFI0	PTM1AF	PTM1PF	PTM0AF	PTM0PF	PTM1AE	PTM1PE	PTM0AE	PTM0PE
MFI1	—	—	STM0AF	STM0PF	—	—	STM0AE	STM0PE

Interrupt Registers List – HT66F4540

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	MF0F	SDC0F	INT0F	MF0E	SDC0E	INT0E	EMI
INTC1	TB1F	TB0F	ADF	MF1F	TB1E	TB0E	ADE	MF1E
INTC2	LVF	SDC1F	SIMF	INT1F	LVE	SDC1E	SIME	INT1E
INTC3	—	—	UARTF	DEF	—	—	UARTE	DEE
MFI0	PTM1AF	PTM1PF	PTM0AF	PTM0PF	PTM1AE	PTM1PE	PTM0AE	PTM0PE
MFI1	STM1AF	STM1PF	STM0AF	STM0PF	STM1AE	STM1PE	STM0AE	STM0PE

Interrupt Registers List – HT66F4550

INTEG Register

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

Bit 7~4 Unimplemented, read as "0"

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

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

INTC0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	MF0F	SDC0F	INT0F	MF0E	SDC0E	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 **MF0F**: Multi-function interrupt 0 request flag
0: No request
1: Interrupt request

Bit 5 **SDC0F**: SD Comparator 0 interrupt request flag
0: No request
1: Interrupt request

Bit 4 **INT0F**: INT0 interrupt request flag
0: No request
1: Interrupt request

Bit 3 **MF0E**: Multi-function interrupt 0 control
0: Disable
1: Enable

Bit 2 **SDC0E**: SD Comparator 0 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	TB1F	TB0F	ADF	MF1F	TB1E	TB0E	ADE	MF1E
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **TB1F**: Time Base 1 interrupt request flag
0: No request
1: Interrupt request

Bit 6 **TB0F**: Time Base 0 interrupt request flag
0: No request
1: Interrupt request

Bit 5 **ADF**: A/D Converter interrupt request flag
0: No request
1: Interrupt request

Bit 4 **MF1F**: Multi-function interrupt 1 request flag
0: No request
1: Interrupt request

Bit 3 **TB1E**: Time Base 1 interrupt control
0: Disable
1: Enable

Bit 2 **TB0E**: Time Base 0 interrupt control
0: Disable
1: Enable

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

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

INTC2 Register

Bit	7	6	5	4	3	2	1	0
Name	LVF	SDC1F	SIMF	INT1F	LVE	SDC1E	SIME	INT1E
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **LVF**: LVD interrupt request flag
0: No request
1: Interrupt request

Bit 6 **SDC1F**: SD Comparator 1 interrupt request flag
0: No request
1: Interrupt request

Bit 5 **SIMF**: SIM interrupt request flag
0: No request
1: Interrupt request

Bit 4 **INT1F**: INT1 interrupt request flag
0: No request
1: Interrupt request

Bit 3 **LVE**: LVD interrupt control
0: Disable
1: Enable

Bit 2 **SDC1E**: SD Comparator 1 interrupt control
0: Disable
1: Enable

Bit 1 **SIME**: SIM interrupt control
0: Disable
1: Enable

Bit 0 **INT1E**: INT1 interrupt control
0: Disable
1: Enable

INTC3 Register – HT66F4530

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

Bit 7~5 Unimplemented, read as "0"

Bit 4 **DEF**: Data EEPROM interrupt request flag
0: No request
1: Interrupt request

Bit 3~1 Unimplemented, read as "0"

Bit 0 **DEE**: Data EEPROM interrupt control
0: Disable
1: Enable

INTC3 Register – HT66F4540/HT66F4550

Bit	7	6	5	4	3	2	1	0
Name	—	—	UARTF	DEF	—	—	UARTE	DEE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5 **UARTF**: UART interrupt request flag
0: No request
1: Interrupt request

Bit 4 **DEF**: Data EEPROM interrupt request flag
0: No request
1: Interrupt request

Bit 3~2 Unimplemented, read as "0"

Bit 1 **UARTE**: UART interrupt control
0: Disable
1: Enable

Bit 0 **DEE**: Data EEPROM interrupt control
0: Disable
1: Enable

MFI0 Register – HT66F4530

Bit	7	6	5	4	3	2	1	0
Name	—	—	PTM0AF	PTM0PF	—	—	PTM0AE	PTM0PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5 **PTM0AF**: PTM0 Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **PTM0PF**: PTM0 Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 3~2 Unimplemented, read as "0"

Bit 1 **PTM0AE**: PTM0 Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **PTM0PE**: PTM0 Comparator P match interrupt control
0: Disable
1: Enable

MFI0 Register – HT66F4540/HT66F4550

Bit	7	6	5	4	3	2	1	0
Name	PTM1AF	PTM1PF	PTM0AF	PTM0PF	PTM1AE	PTM1PE	PTM0AE	PTM0PE
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 **PTM1AF**: PTM1 Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 6 **PTM1PF**: PTM1 Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 5 **PTM0AF**: PTM0 Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **PTM0PF**: PTM0 Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 3 **PTM1AE**: PTM1 Comparator A match interrupt control
0: Disable
1: Enable

Bit 2 **PTM1PE**: PTM1 Comparator P match interrupt control
0: Disable
1: Enable

Bit 1 **PTM0AE**: PTM0 Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **PTM0PE**: PTM0 Comparator P match interrupt control
0: Disable
1: Enable

MFI1 Register – HT66F4530/HT66F4540

Bit	7	6	5	4	3	2	1	0
Name	—	—	STM0AF	STM0PF	—	—	STM0AE	STM0PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5 **STM0AF**: STM0 Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **STM0PF**: STM0 Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 3~2 Unimplemented, read as "0"

Bit 1 **STM0AE**: STM0 Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **STM0PE**: STM0 Comparator P match interrupt control
0: Disable
1: Enable

MFI1 Register – HT66F4550

Bit	7	6	5	4	3	2	1	0
Name	STM1AF	STM1PF	STM0AF	STM0PF	STM1AE	STM1PE	STM0AE	STM0PE
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 **STM1AF**: STM1 Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 6 **STM1PF**: STM1 Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 5 **STM0AF**: STM0 Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **STM0PF**: STM0 Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 3 **STM1AE**: STM1 Comparator A match interrupt control
0: Disable
1: Enable

Bit 2 **STM1PE**: STM1 Comparator P match interrupt control
0: Disable
1: Enable

Bit 1 **STM0AE**: STM0 Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **STM0PE**: STM0 Comparator P match interrupt control
0: Disable
1: Enable

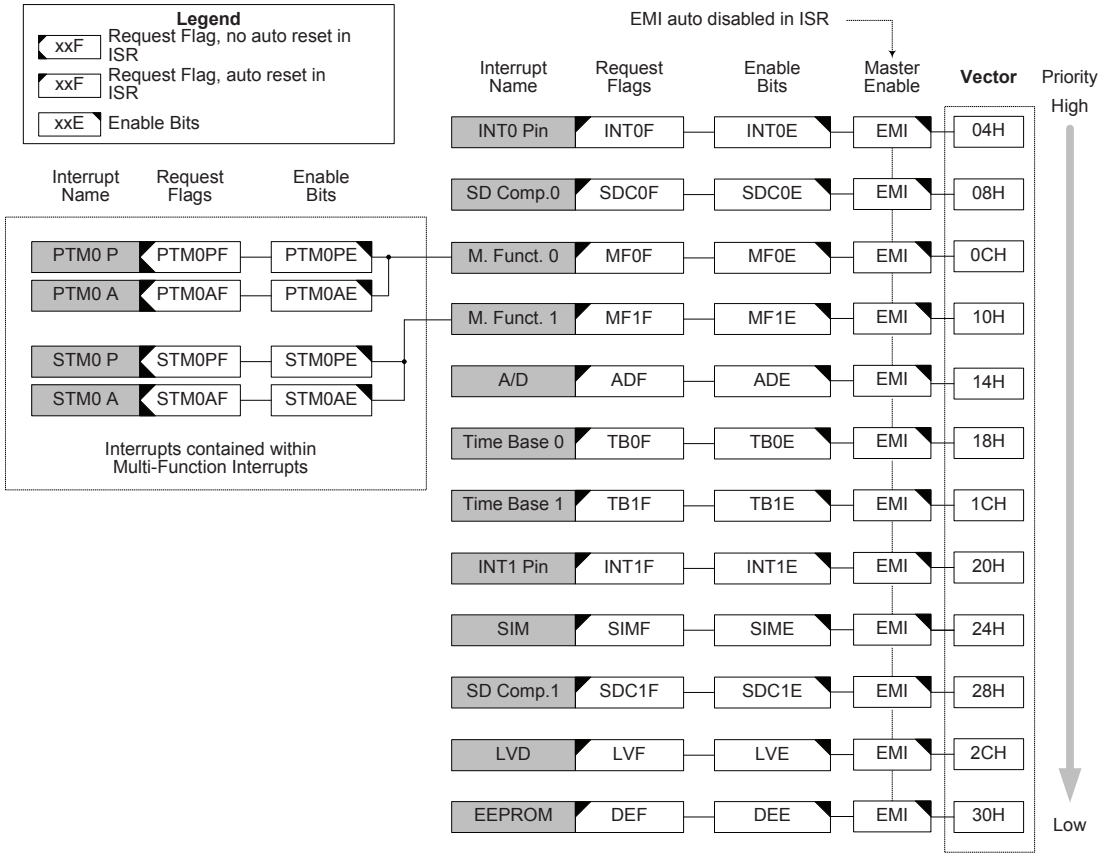
Interrupt Operation

When the conditions for an interrupt event occur, such as a TM Comparator P, Comparator A match or A/D conversion completion etc., the relevant interrupt request flag will be set. Whether the request flag actually generates a program jump to the relevant interrupt vector is determined by the condition of the interrupt enable bit. If the enable bit is set high then the program will jump to its relevant vector; if the enable bit is zero then although the interrupt request flag is set an actual interrupt will not be generated and the program will not jump to the relevant interrupt vector. The global interrupt enable bit, if cleared to zero, will disable all interrupts.

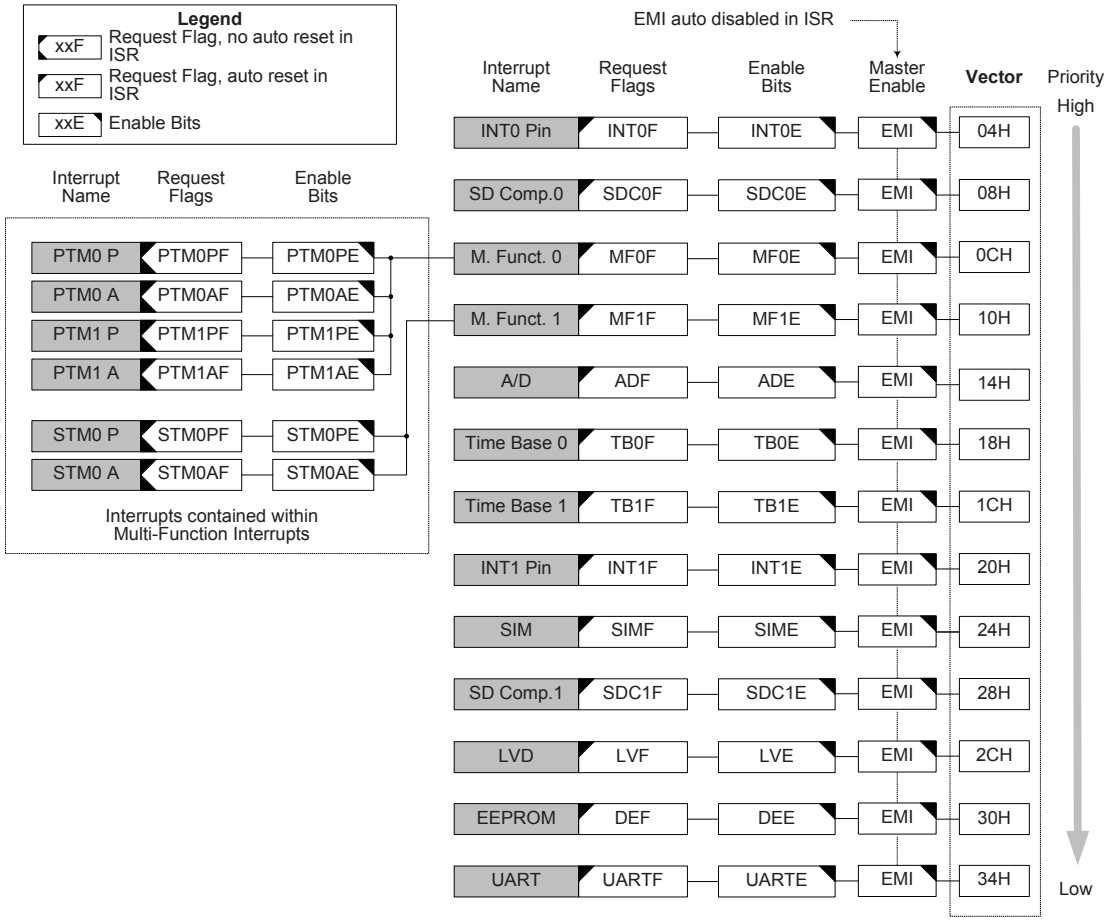
When an interrupt is generated, the Program Counter, which stores the address of the next instruction to be executed, will be transferred onto the stack. The Program Counter will then be loaded with a new address which will be the value of the corresponding interrupt vector. The microcontroller will then fetch its next instruction from this interrupt vector. The instruction at this vector will usually be a "JMP" which will jump to another section of program which is known as the interrupt service routine. Here is located the code to control the appropriate interrupt. The interrupt service routine must be terminated with a "RETI", which retrieves the original Program Counter address from the stack and allows the microcontroller to continue with normal execution at the point where the interrupt occurred.

The various interrupt enable bits, together with their associated request flags, are shown in the accompanying diagrams with their order of priority. Some interrupt sources have their own individual vector while others share the same multi-function interrupt vector. Once an interrupt subroutine is serviced, all the other interrupts will be blocked, as the global interrupt enable bit, EMI bit will be cleared automatically. This will prevent any further interrupt nesting from occurring. However, if other interrupt requests occur during this interval, although the interrupt will not be immediately serviced, the request flag will still be recorded.

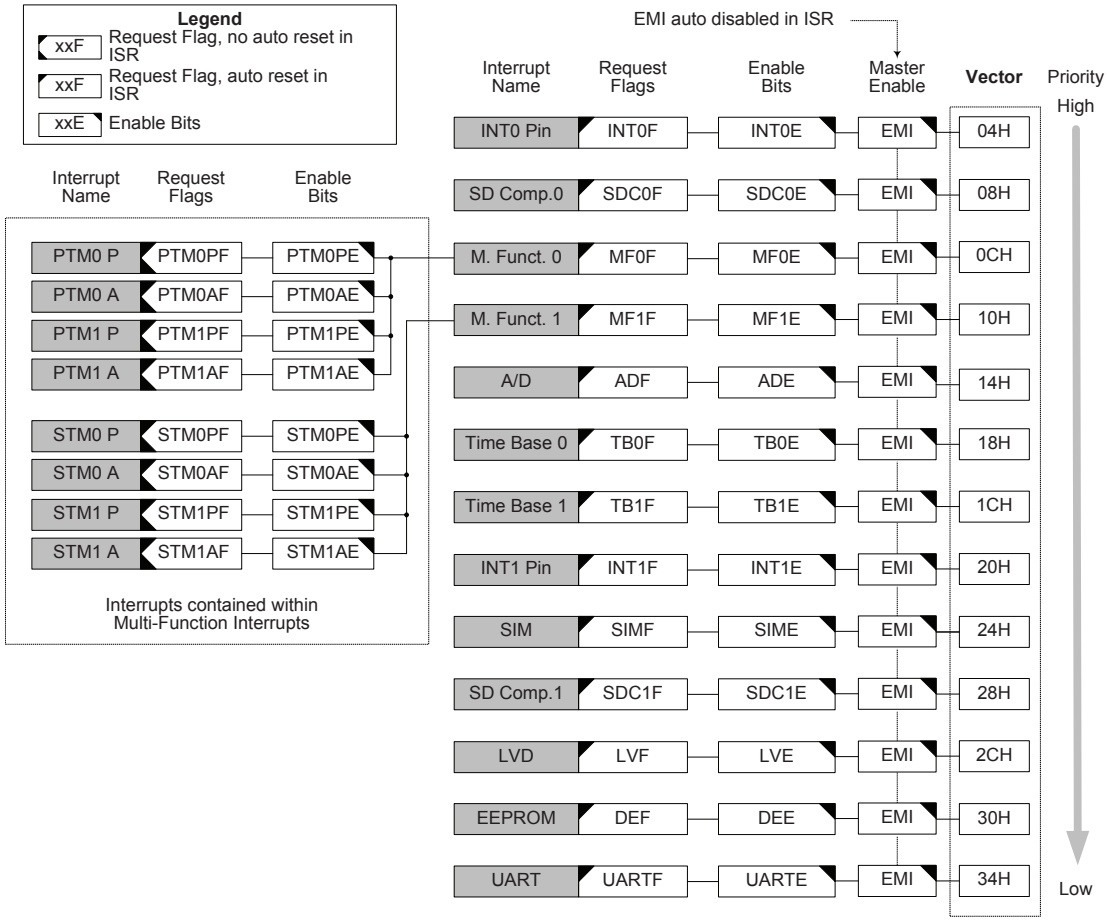
If an interrupt requires immediate servicing while the program is already in another interrupt service routine, the EMI bit should be set after entering the routine, to allow interrupt nesting. If the stack is full, the interrupt request will not be acknowledged, even if the related interrupt is enabled, until the Stack Pointer is decremented. If immediate service is desired, the stack must be prevented from becoming full. In case of simultaneous requests, the accompanying diagram shows the priority that is applied. All of the interrupt request flags when set will wake-up these devices if it is in SLEEP or IDLE Mode, however to prevent a wake-up from occurring the corresponding flag should be set before these devices are in SLEEP or IDLE Mode.



Interrupt Structure – HT66F4530



Interrupt Structure – HT66F4540



Interrupt Structure – HT66F4550

External Interrupts

The external interrupts are controlled by signal transitions on the pins INT0~INT1. An external interrupt request will take place when the external interrupt request flags, INT0F~INT1F, are set, which will occur when a transition, whose type is chosen by the edge select bits, appears on the external interrupt pins. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and respective external interrupt enable bit, INT0E~INT1E, must first be set. Additionally the correct interrupt edge type must be selected using the INTEG register to enable the external interrupt function and to choose the trigger edge type. As the external interrupt pins are pin-shared with I/O pins, they can only be configured as external interrupt pins if their external interrupt enable bit in the corresponding interrupt register has been set and the external interrupt pin is selected by the corresponding pin-shared function selection bits. The pin must also be setup as an input by setting the corresponding bit in the port control register. When the interrupt is enabled, the stack is not full and the correct transition type appears on the external interrupt pin, a subroutine call to the external interrupt vector, will take place. When the interrupt is serviced, the external interrupt request flags, INT0F~INT1F, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts. Note that any pull-high resistor selections on the external interrupt pins will remain valid even if the pin is used as an external interrupt input. The INTEG register is used to select the type of active edge that will trigger the external interrupt. A choice of either rising or falling or both edge types can be chosen to trigger an external interrupt. Note that the INTEG register can also be used to disable the external interrupt function.

SD Comparator Interrupts

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

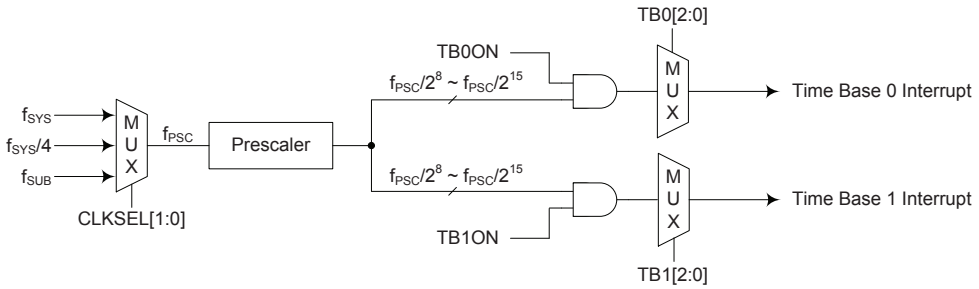
A/D Converter Interrupt

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

Time Base Interrupts

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

The purpose of the Time Base Interrupt is to provide an interrupt signal at fixed time periods. Its clock source, f_{PSC}, originates from the internal clock source f_{SYS}, $f_{SYS}/4$ or f_{SUB} and then passes through a divider, the division ratio of which is selected by programming the appropriate bits in the TB0C and TB1C 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 CLKSEL1~CLKSEL0 bits in the PSCR register.



Time Base Interrupt

PSCR Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CLKSEL1	CLKSEL0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **CLKSEL1~CLKSEL0**: Prescaler clock source selection
00: f_{SYS}
01: $f_{SYS}/4$
1x: f_{SUB}

TB0C 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 Control
0: Disable
1: Enable

Bit 6~3 Unimplemented, read as "0"

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

TB1C 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 Control
0: Disable
1: Enable

Bit 6~3 Unimplemented, read as "0"

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

SIM Interrupt

The Serial Interface Module Interrupt, also known as the SIM interrupt, 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, or an I^2C slave address match occurs, or an I^2C bus time-out occurs. 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, must first be set. When the interrupt is enabled, the stack is not full and any of the above described situations occurs, a subroutine call to the respective Interrupt vector, will take place. When the interrupt is serviced, the Serial Interface Interrupt flag, SIMF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

LVD Interrupt

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

EEPROM Interrupt

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

UART Interrupt

Several individual UART conditions can generate a UART interrupt. When one of these conditions occurs, an interrupt pulse will be generated to get the attention of the microcontroller. These conditions are a transmitter data register empty, transmitter idle, receiver data available, receiver overrun, address detect and an RX pin wake-up. To allow the program to branch to the respective interrupt vector addresses, the global interrupt enable bit, EMI, and UART interrupt enable bit, UARTE, must first be set. When the interrupt is enabled, the stack is not full and any of these conditions are created, a subroutine call to the UART Interrupt vector, will take place. When the UART Interrupt is serviced, the UART Interrupt flag, UARTF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts. However, the USR register flags will only be cleared when certain actions are taken by the UART, the details of which are given in the UART section.

Multi-function Interrupts

Within these devices there are up to two Multi-function interrupts. Unlike the other independent interrupts, these interrupts have no independent source, but rather are formed from other existing interrupt sources, namely the TM Interrupts.

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

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

TM Interrupts

The Standard and Periodic Type TMs have two interrupts, one comes from the comparator A match situation and the other comes from the comparator P match situation. All of the TM interrupts are contained within the Multi-function Interrupts. For each of the Standard and Periodic Type TMs there are two interrupt request flags xTMnPF and xTMnAF and two enable bits xTMnPE and xTMnAE. A TM interrupt request will take place when any of the TM request flags are set, a situation which occurs when a TM comparator P or A match situation happens.

To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, respective TM Interrupt enable bit, and relevant Multi-function Interrupt enable bit, MFnE, must first be set. When the interrupt is enabled, the stack is not full and a TM comparator match situation occurs, a subroutine call to the relevant Multi-function Interrupt vector locations, will take place. When the TM interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the related MFnF flag will be automatically cleared. As the TM interrupt request flags will not be automatically cleared, they have to be cleared by the application program.

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 or a low power supply voltage may cause their respective interrupt flag to be set high and consequently generate an interrupt. Care must therefore be taken if spurious wake-up situations are to be avoided. If an interrupt wake-up function is to be disabled then the corresponding interrupt request flag should be set high before these devices enter 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 SLEEP or IDLE Mode, the wake up being generated when the interrupt request flag changes from low to high. If it is required to prevent a certain interrupt from waking up the microcontroller then its respective request flag should be first set high before enter SLEEP or IDLE Mode.

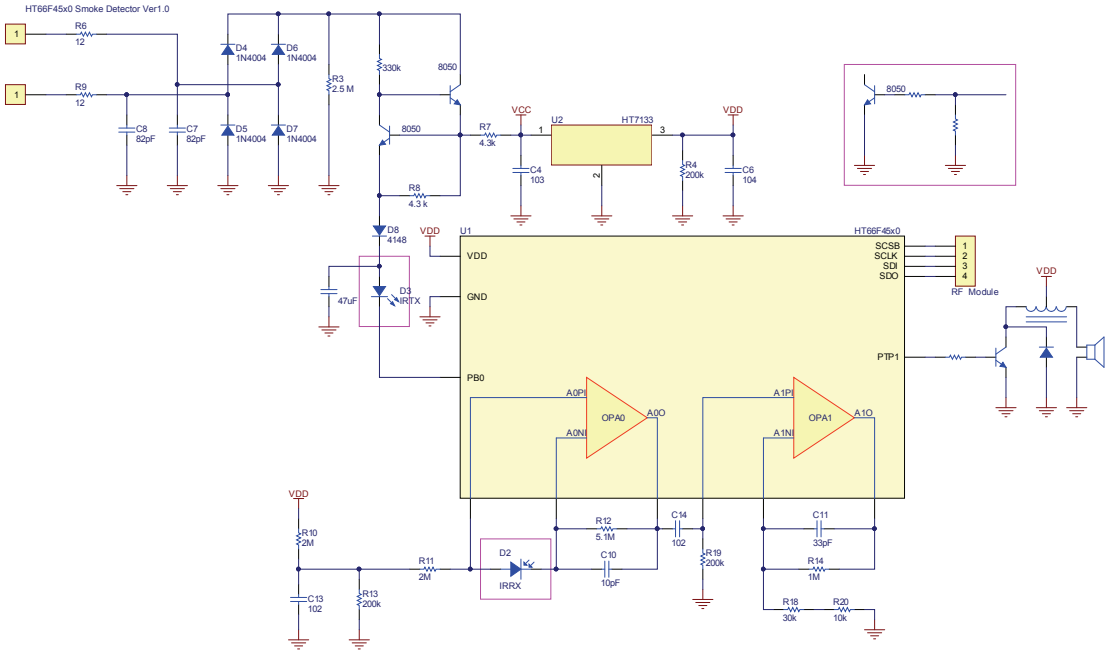
As only the Program Counter is pushed onto the stack, then when the interrupt is serviced, if the contents of the accumulator, status register or other registers are altered by the interrupt service program, their contents should be saved to the memory at the beginning of the interrupt service routine. To return from an interrupt subroutine, either a RET or RETI instruction may be executed. The RETI instruction in addition to executing a return to the main program also automatically sets the EMI bit high to allow further interrupts. The RET instruction however only executes a return to the main program leaving the EMI bit in its present zero state and therefore disabling the execution of further interrupts.

Configuration Option

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 RC Oscillator Frequency Selection: 1. 2MHz 2. 4MHz 3. 8MHz

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 three kinds of MOV instructions, data can be transferred from registers to the Accumulator and vice-versa as well as being able to move specific immediate data directly into the Accumulator. One of the most important data transfer applications is to receive data from the input ports and transfer data to the output ports.

Arithmetic Operations

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

Logical and Rotate Operation

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

Branches and Control Transfer

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

Bit Operations

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

Table Read Operations

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

Other Operations

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

Instruction Set Summary

The instructions related to the data memory access in the following table can be used when the desired data memory is located in Data Memory sector 0.

Table Conventions

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

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

Mnemonic	Description	Cycles	Flag Affected
Data Move			
MOV A,[m]	Move Data Memory to ACC	1	None
MOV [m],A	Move ACC to Data Memory	1Note	None
MOV A,x	Move immediate data to ACC	1	None
Bit Operation			
CLR [m].i	Clear bit of Data Memory	1Note	None
SET [m].i	Set bit of Data Memory	1Note	None
Branch Operation			
JMP addr	Jump unconditionally	2	None
SZ [m]	Skip if Data Memory is zero	1Note	None
SZA [m]	Skip if Data Memory is zero with data movement to ACC	1Note	None
SZ [m].i	Skip if bit i of Data Memory is zero	1Note	None
SNZ [m]	Skip if Data Memory is not zero	1Note	None
SNZ [m].i	Skip if bit i of Data Memory is not zero	1Note	None
SIZ [m]	Skip if increment Data Memory is zero	1Note	None
SDZ [m]	Skip if decrement Data Memory is zero	1Note	None
SIZA [m]	Skip if increment Data Memory is zero with result in ACC	1Note	None
SDZA [m]	Skip if decrement Data Memory is zero with result in ACC	1Note	None
CALL addr	Subroutine call	2	None
RET	Return from subroutine	2	None
RET A,x	Return from subroutine and load immediate data to ACC	2	None
RETI	Return from interrupt	2	None
Table Read Operation			
TABRD [m]	Read table (specific page) to TBLH and Data Memory	2Note	None
TABRDL [m]	Read table (last page) to TBLH and Data Memory	2Note	None
ITABRD [m]	Increment table pointer TBLP first and Read table 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 sector 0, the extended instruction can be used to access the data memory instead of using the indirect addressing access to improve the CPU firmware performance.

Mnemonic	Description	Cycles	Flag Affected
Arithmetic			
LADD A,[m]	Add Data Memory to ACC	2	Z, C, AC, OV, SC
LADDM A,[m]	Add ACC to Data Memory	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

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

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

Affected flag(s) None

RRC [m] Rotate Data Memory right through Carry

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

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

Affected flag(s) C

RRCA [m] Rotate Data Memory right through Carry with result in ACC

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

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

Affected flag(s): C

SBC A,[m] Subtract Data Memory from ACC with Carry

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

Operation: ACC ← ACC − [m] − C

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

SBC A, x Subtract immediate data from ACC with Carry

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

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

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 (specific page) to TBLH and Data Memory

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

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

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

Operation ACC ← ACC ″XOR″ [m]

Affected flag(s) Z

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

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

Operation [m] ← ACC ″XOR″ [m]

Affected flag(s) Z

XOR A,x Logical XOR immediate data to ACC

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

Operation ACC ← ACC ″XOR″ x

Affected flag(s) Z

Extended Instruction Definition

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

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

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

Operation ACC ← ACC + [m] + C

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

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

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

Operation [m] ← ACC + [m] + C

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

LADD A,[m] Add Data Memory to ACC

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

Operation ACC ← ACC + [m]

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

LADDM A,[m] Add ACC to Data Memory

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

Operation [m] ← ACC + [m]

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

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

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

Operation ACC ← ACC ″AND″ [m]

Affected flag(s) Z

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

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

Operation [m] ← ACC ″AND″ [m]

Affected flag(s) Z

LCLR [m] Clear Data Memory

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

Operation [m] ← 00H

Affected flag(s) None

LCLR [m].i Clear bit of Data Memory

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

Operation [m].i ← 0

Affected flag(s) None

LCPL [m] Complement Data Memory

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

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

Affected flag(s) Z

LCPLA [m] Complement Data Memory with result in ACC

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

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

Affected flag(s) Z

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

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

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

Affected flag(s) C

LDEC [m] Decrement Data Memory

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

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

Affected flag(s) Z

LDECA [m] Decrement Data Memory with result in ACC

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

Operation $ACC \leftarrow [m] - 1$

Affected flag(s) Z

LINC [m] Increment Data Memory

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

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

Affected flag(s) Z

LINCA [m] Increment Data Memory with result in ACC

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

Operation $ACC \leftarrow [m] + 1$

Affected flag(s) Z

LMOV A,[m] Move Data Memory to ACC

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

Operation ACC ← [m]

Affected flag(s) None

LMOV [m],A Move ACC to Data Memory

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

Operation [m] ← ACC

Affected flag(s) None

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

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

Operation ACC ← ACC ″OR″ [m]

Affected flag(s) Z

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

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

Operation [m] ← ACC ″OR″ [m]

Affected flag(s) Z

LRL [m] Rotate Data Memory left

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

LRLC [m] Rotate Data Memory left through Carry

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

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

Affected flag(s) C

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

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

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

Affected flag(s) C

LRR [m] Rotate Data Memory right

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

LRRC [m] Rotate Data Memory right through Carry

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

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

Affected flag(s) C

LRRCA [m] Rotate Data Memory right through Carry with result in ACC

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

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

Affected flag(s) C

LSBC A,[m] Subtract Data Memory from ACC with Carry

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

Operation ACC ← 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

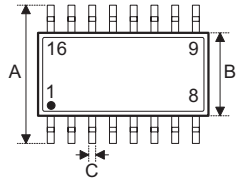
Package Information

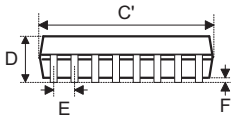
Note that the package information provided here is for consultation purposes only. As this information may be updated at regular intervals users are reminded to consult the Holtek website for the latest version of the Package/Carton Information.

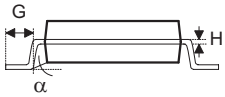
Additional supplementary information with regard to packaging is listed below. Click on the relevant section to be transferred to the relevant website page.

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

16-pin NSOP (150mil) Outline Dimensions



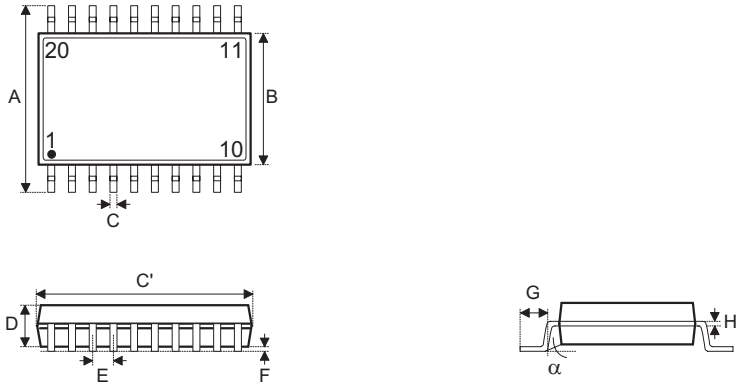




Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.012	—	0.020
C’	—	0.390 BSC	—
D	—	—	0.069
E	—	0.050 BSC	—
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	—	6.00 BSC	—
B	—	3.90 BSC	—
C	0.31	—	0.51
C’	—	9.90 BSC	—
D	—	—	1.75
E	—	1.27 BSC	—
F	0.10	—	0.25
G	0.40	—	1.27
H	0.10	—	0.25
α	0°	—	8°

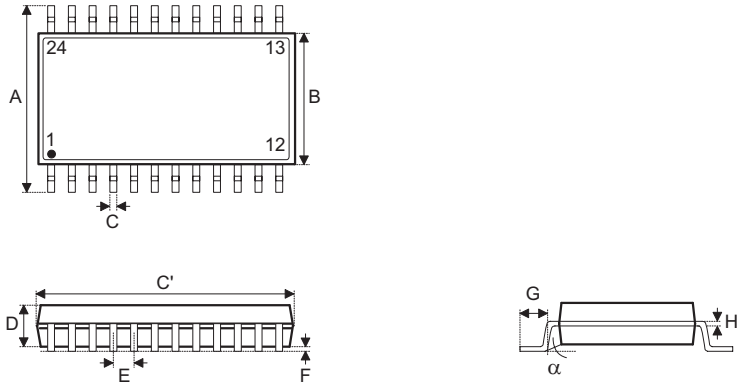
20-pin SSOP (150mil) Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	—	0.236 BSC	—
B	—	0.155 BSC	—
C	0.008	—	0.012
C’	—	0.341 BSC	—
D	—	—	0.069
E	—	0.025 BSC	—
F	0.004	—	0.0098
G	0.016	—	0.05
H	0.004	—	0.01
α	0°	—	8°

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

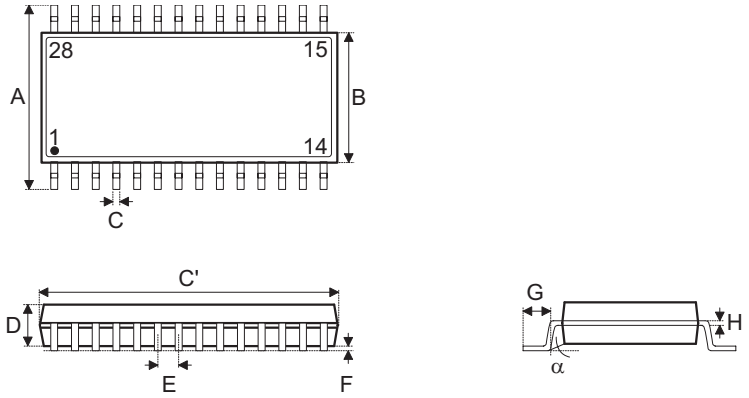
24-pin SSOP (150mil) Outline Dimensions



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

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

28-pin SSOP (150mil) Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.008	—	0.012
C’	—	0.390 BSC	—
D	—	—	0.069
E	—	0.025 BSC	—
F	0.004	—	0.0098
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	—	6.000 BSC	—
B	—	3.900 BSC	—
C	0.20	—	0.30
C’	—	9.900 BSC	—
D	—	—	1.75
E	—	0.635 BSC	—
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
α	0°	—	8°

Copyright© 2016 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.