TinyPower™ Flash MCU with LCD & EEPROM

HT67F370/HT69F340/HT69F350/HT69F360

Revision: V1.62 Date: February 26, 2025

www.holtek.com

Features

CPU Features

- Operating voltage
  - f_{SYS}=4MHz: 1.8V~5.5V
  - f_{SYS}=8MHz (HT69F340/350/360): 2.0V~5.5V
  - f_{SYS}=8MHz (HT67F370): 1.8V~5.5V
  - f_{SYS}=12MHz: 2.7V~5.5V
  - f_{SYS}=16MHz (HT69F340/350/360): 4.5V~5.5V
  - f_{SYS}=16MHz (HT67F370): 3.3V~5.5V
- Up to 0.25μs instruction cycle with 16MHz system clock at V_{DD}=5V
- Power down and wake-up functions to reduce power consumption
- Oscillator types
  - External High Speed Crystal – HXT
  - Internal High Speed 4/8/12MHz RC – HIRC
  - External Low Speed 32.768kHz Crystal – LXT
  - Internal Low Speed 32kHz – LIRC
- Fully integrated internal oscillators require no external components
- Multi-mode operation: NORMAL, SLOW, IDLE and SLEEP
- All instructions executed in one to three instruction cycles
- Table read instructions
- 115 powerful instructions
- 8-level subroutine nesting
- Bit manipulation instruction

Peripheral Features

- Flash Program Memory: 4K×16~32K×16
- RAM Data Memory: 256×8~2048×8
- True EEPROM Memory: 64×8~256×8
- In Application Programming function – IAP
- Watchdog Timer function
- Up to 63 bidirectional I/O lines
- LCD driver function
- Two external interrupt lines shared with I/O pins
- Multiple Timer Modules for time measure, input capture, compare match output, PWM output function or single pulse output function
- Dual Time Base functions for generation of fixed time interrupt signals
- Serial Interfaces Module – SIM for SPI or I^2C
- Single Fully-duplex Universal Asynchronous Receiver and Transmitter Interface – UART (For HT69F360 and HT67F370)
- 12 external channel 12-bit resolution A/D converter with internal reference voltage V_{VR}
- Low voltage reset function
- Low voltage detect function
- Flash program memory can be re-programmed up to 10,000 times

- Flash program memory data retention > 10 years
- True EEPROM data memory can be re-programmed up to 100,000 times
- True EEPROM data memory data retention > 10 years
- Package types: 48/64/80-pin LQFP

Development Tools

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

https://www.holtek.com/page/tool-detail/dev_plat/ha/IR_Remote_Controller_Workshop

General Description

The series of devices are Flash Memory A/D type 8-bit high performance RISC architecture microcontrollers, designed for applications which require an LCD interface.

For memory features, the Flash Memory offers users the convenience of multi-programming 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. By using the In Application Programming technology, users have a convenient means to directly store their measured data in the Flash Program Memory as well as having the ability to easily update their application programs.

Analog features include a multi-channel 12-bit AD converter. Multiple and extremely flexible Timer Modules provide timing, pulse generation, capture input, compare match output and PWM generation functions. Communication with the outside world is catered for by including fully integrated SPI, I^2C and UART interface functions, three 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 external, internal, high and low speed oscillators is provided including two fully integrated system oscillators which requires no external components for their 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 minimize power consumption.

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

Selection Table

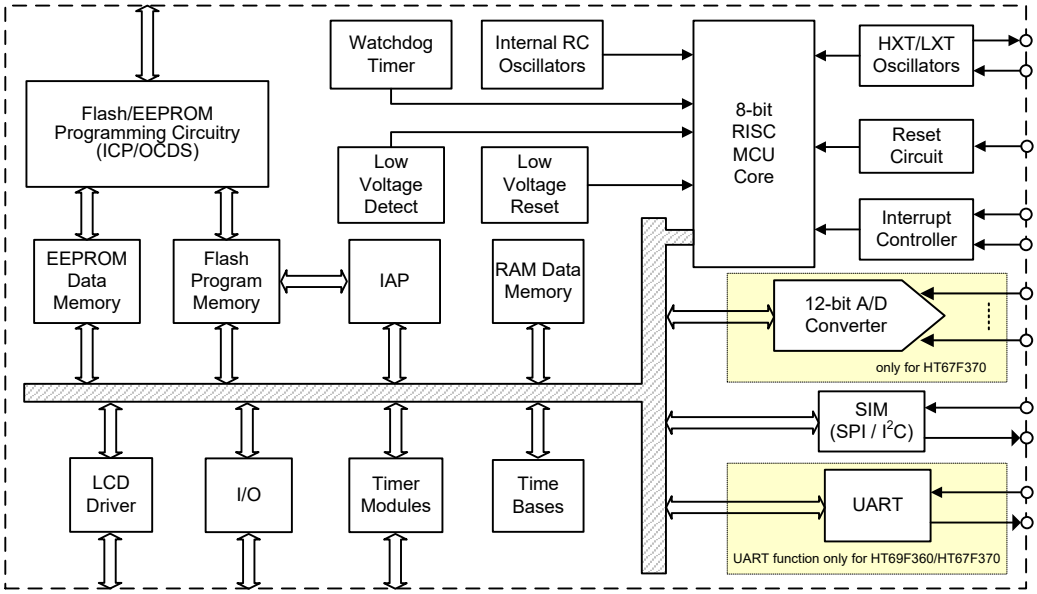
Most features are common to all devices. The main features distinguishing them are Program Memory, Data Memory capacity, EEPROM capacity, I/O count, TM features, A/D converter, LCD display and package types. The following table summarises the main features of each device.

Part No.	V_{DD}	Program Memory	Data Memory	Data EEPROM	I/O	Ext. Int.	Stacks
HT67F370	1.8V~5.5V	32K × 16	2048 × 8	256 × 8	63	2	8
HT69F340	1.8V~5.5V	4K × 16	256 × 8	64 × 8	39	2	8
HT69F350	1.8V~5.5V	8K × 16	512 × 8	64 × 8	55	2	8
HT69F360	1.8V~5.5V	16K × 16	1024 × 8	128 × 8	63	2	8

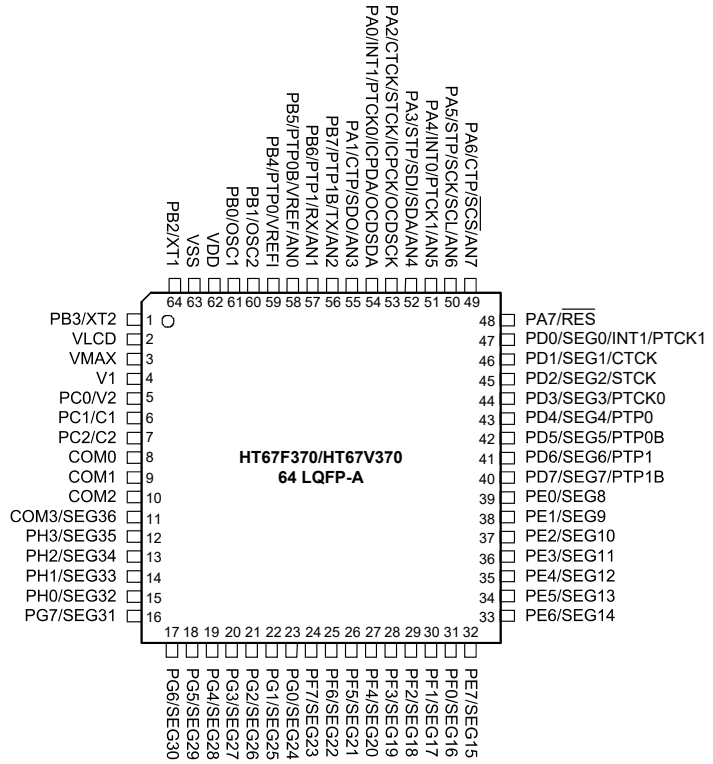
Part No.	Timer Module	Interface (SPI/I²C)	UART	A/D Converter	Time Base	LCD Driver	Package
HT67F370	10-bit CTM × 1 16-bit STM × 1 10-bit PTM × 2	√	√	12-bit × 12	2	48×4/49×3	64/80LQFP
HT69F340	10-bit CTM × 1 10-bit PTM × 1	√	—	—	2	24×4/25×3	48LQFP
HT69F350	10-bit CTM × 1 16-bit STM × 1 10-bit PTM × 1	√	—	—	2	36×4/37×3	48/64LQFP
HT69F360	10-bit CTM × 1 16-bit STM × 1 10-bit PTM × 2	√	√	—	2	48×4/49×3	64/80LQFP

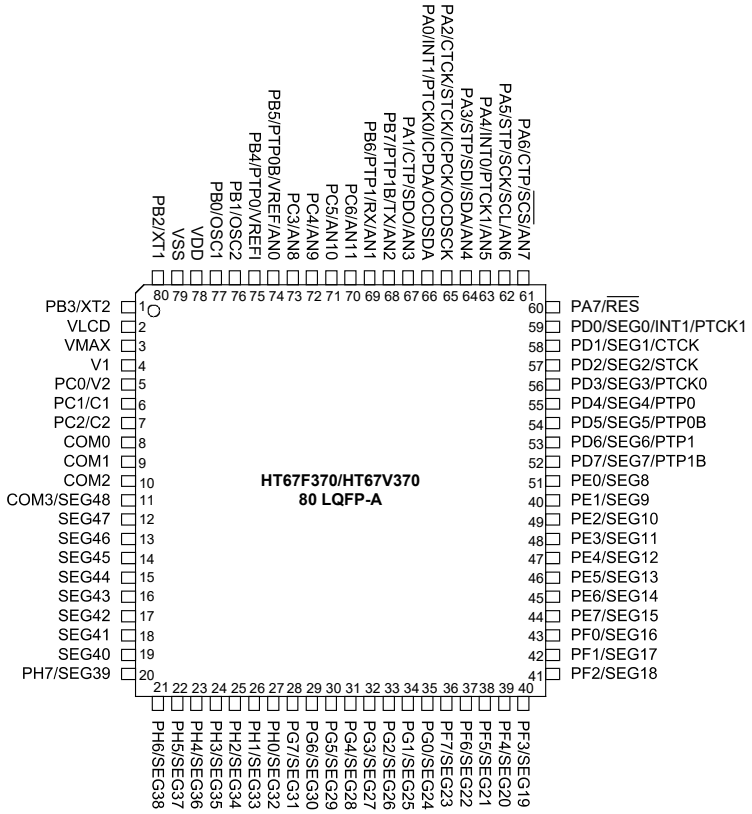
Note: As devices exist in more than one package format, the table reflects the situation for the package with the most pins.

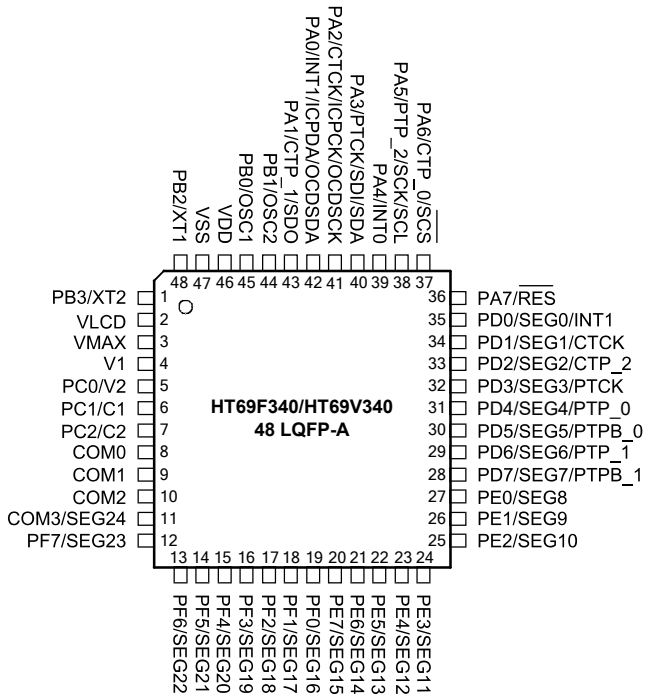
Block Diagram

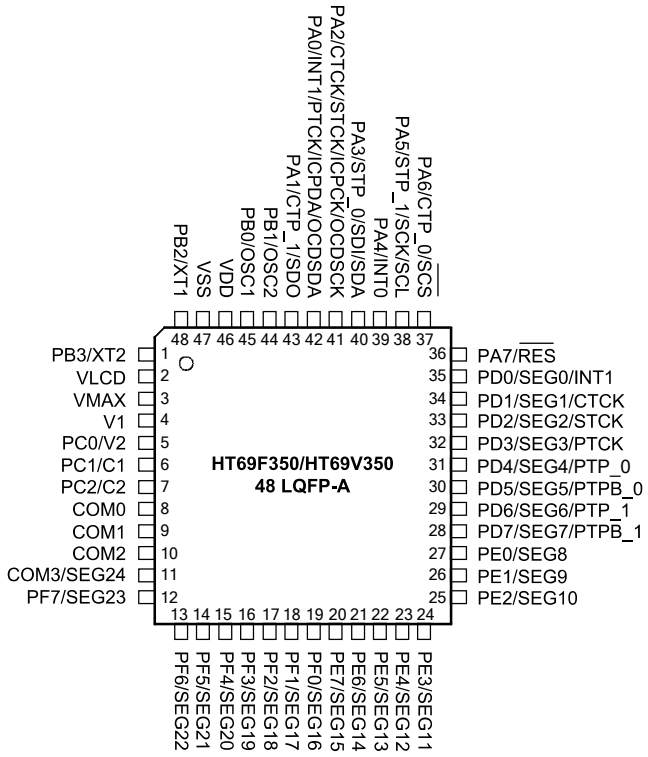


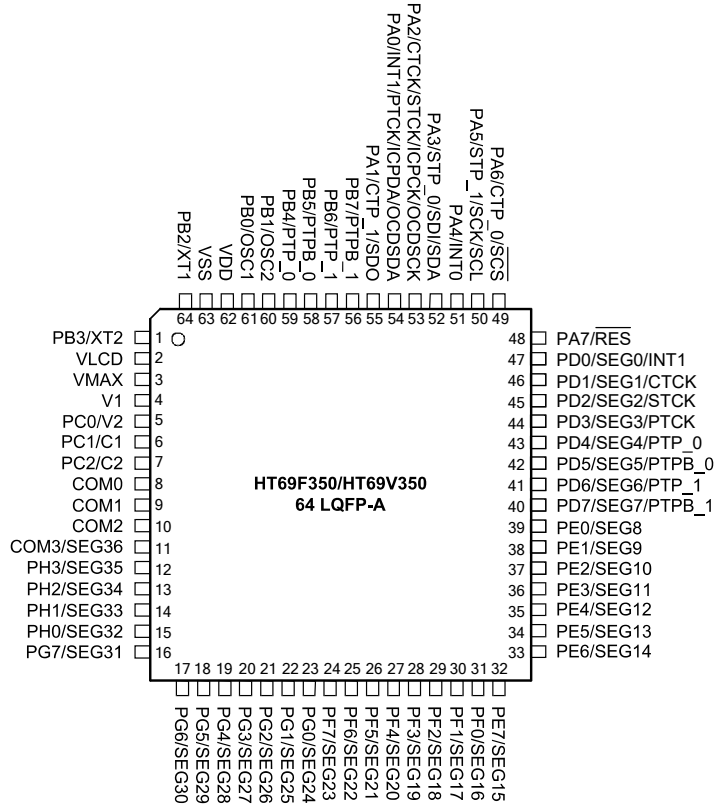
Pin Assignment

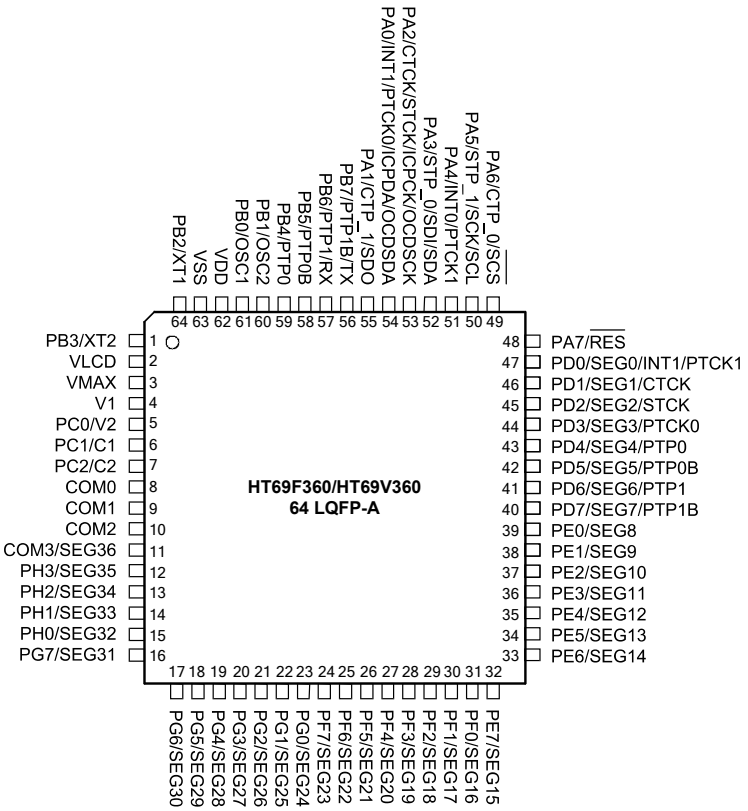


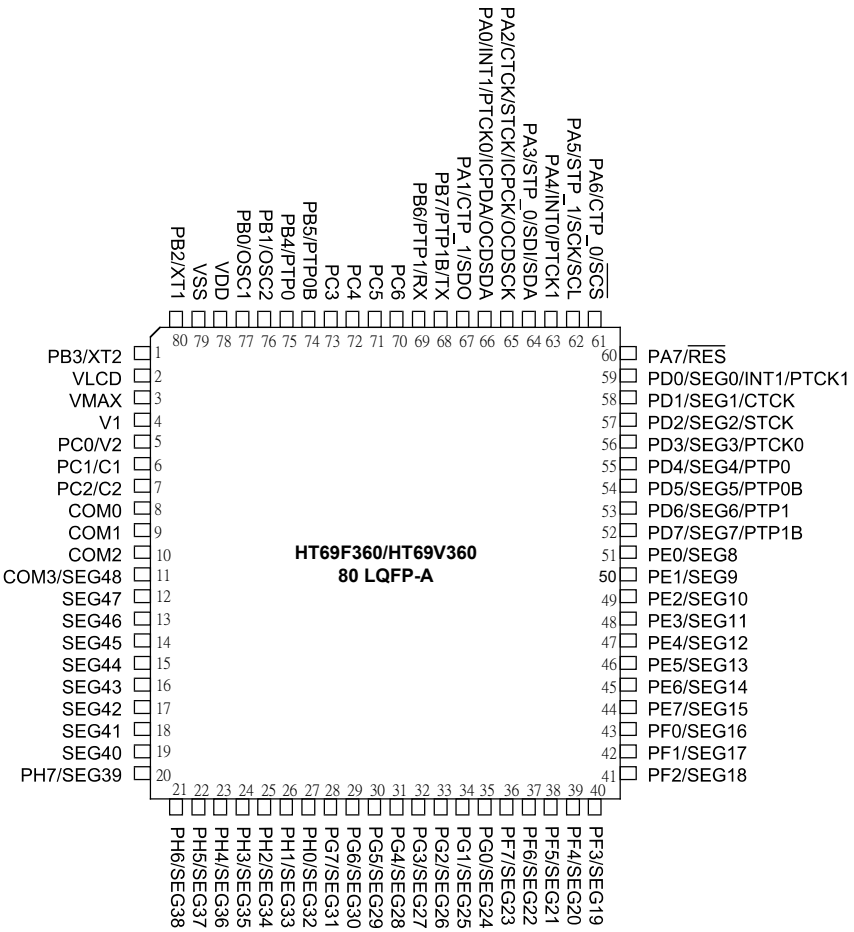



PB2/XT1
VSS
VDD
PB0/OSC1
PB1/OSC2
PA1/CTP_1/SDO
PA0/INT1/ICPDA/OCDSDA
PA2/CTCK/ICPCK/OCDSCK
PA3/PTCK/SDI/SDA
PA4/INT0
PA5/PTP_2/SCK/SCL
PA6/CTP_0/SCS
48 47 46 45 44 43 42 41 40 39 38 37
PB3/XT2 1
VLCD 2
VMAX 3
V1 4
PC0/V2 5
PC1/C1 6
PC2/C2 7
COM0 8
COM1 9
COM2 10
COM3/SEG24 11
PF7/SEG23 12
HT69F340/HT69V340
48 LQFP-A
36 PA7/RES
35 PD0/SEG0/INT1
34 PD1/SEG1/CTCK
33 PD2/SEG2/CTP_2
32 PD3/SEG3/PTCK
31 PD4/SEG4/PTP_0
30 PD5/SEG5/PTPB_0
29 PD6/SEG6/PTP_1
28 PD7/SEG7/PTPB_1
27 PE0/SEG8
26 PE1/SEG9
25 PE2/SEG10
13 14 15 16 17 18 19 20 21 22 23 24
PF6/SEG22
PF5/SEG21
PF4/SEG20
PF3/SEG19
PF2/SEG18
PF1/SEG17
PF0/SEG16
PE7/SEG15
PE6/SEG14
PE5/SEG13
PE4/SEG12
PE3/SEG11


PB2/XT1
VSS
VDD
PB0/OSC1
PB1/OSC2
PA1/CTP_1/SDO
PA0/INT1/PTCK/ICPDA/OCDSDA
PA2/CTCK/STCK/ICPCK/OCDSCK
PA3/STP_0/SDI/SDA
PA4/INT0
PA5/STP_1/SCK/SCL
PA6/CTP_0/SCS
48 47 46 45 44 43 42 41 40 39 38 37
PB3/XT2 1
VLCD 2
VMAX 3
V1 4
PC0/V2 5
PC1/C1 6
PC2/C2 7
COM0 8
COM1 9
COM2 10
COM3/SEG24 11
PF7/SEG23 12
HT69F350/HT69V350
48 LQFP-A
36 PA7/RES
35 PD0/SEG0/INT1
34 PD1/SEG1/CTCK
33 PD2/SEG2/STCK
32 PD3/SEG3/PTCK
31 PD4/SEG4/PTP_0
30 PD5/SEG5/PTPB_0
29 PD6/SEG6/PTP_1
28 PD7/SEG7/PTPB_1
27 PE0/SEG8
26 PE1/SEG9
25 PE2/SEG10
13 14 15 16 17 18 19 20 21 22 23 24
PF6/SEG22
PF5/SEG21
PF4/SEG20
PF3/SEG19
PF2/SEG18
PF1/SEG17
PF0/SEG16
PE7/SEG15
PE6/SEG14
PE5/SEG13
PE4/SEG12
PE3/SEG11


PB2/XT1
VSS
VDD
PB0/OSC1
PB1/OSC2
PB4/PTP_0
PB5/PTPB_0
PB6/PTP_1
PB7/PTPB_1
PA1/CTP_1/SDO
PA0/INT1/PTCK/ICPDA/OCDSCK
PA2/CTCK/STCK/ICPCK/OCDSCK
PA3/STP_0/SDI/SDA
PA4/INT0
PA5/STP_1/SCK/SCL
PA6/CTP_0/SCS
64 63 62 61 60 59 58 57 56 55 54 53 52 51 50 49
PB3/XT2 1
VLCD 2
VMAX 3
V1 4
PC0/V2 5
PC1/C1 6
PC2/C2 7
COM0 8
COM1 9
COM2 10
COM3/SEG36 11
PH3/SEG35 12
PH2/SEG34 13
PH1/SEG33 14
PH0/SEG32 15
PG7/SEG31 16
HT69F350/HT69V350
64 LQFP-A
48 PA7/RES
47 PD0/SEG0/INT1
46 PD1/SEG1/CTCK
45 PD2/SEG2/STCK
44 PD3/SEG3/PTCK
43 PD4/SEG4/PTP_0
42 PD5/SEG5/PTPB_0
41 PD6/SEG6/PTP_1
40 PD7/SEG7/PTPB_1
39 PE0/SEG8
38 PE1/SEG9
37 PE2/SEG10
36 PE3/SEG11
35 PE4/SEG12
34 PE5/SEG13
33 PE6/SEG14
17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
PG6/SEG30
PG5/SEG29
PG4/SEG28
PG3/SEG27
PG2/SEG26
PG1/SEG25
PG0/SEG24
PF7/SEG23
PF6/SEG22
PF5/SEG21
PF4/SEG20
PF3/SEG19
PF2/SEG18
PF1/SEG17
PF0/SEG16
PE7/SEG15


PB2/XT1
VSS
VDD
PB0/OSC1
PB1/OSC2
PB4/PTP0
PB5/PTP0B
PB6/PTP1/RX
PB7/PTP1B/TX
PA1/CTP_1/SDO
PA0/INT1/PTCK0/ICPDA/OCDSCK
PA2/CTCK/STCK/ICPCK/OCDSCK
PA3/STP_0/SDI/SDA
PA4/INT0/PTCK1
PA5/STP_1/SCK/SCL
PA6/CTP_0/SCS
64 63 62 61 60 59 58 57 56 55 54 53 52 51 50 49
PB3/XT2 1
VLCD 2
VMAX 3
V1 4
PC0/V2 5
PC1/C1 6
PC2/C2 7
COM0 8
COM1 9
COM2 10
COM3/SEG36 11
PH3/SEG35 12
PH2/SEG34 13
PH1/SEG33 14
PH0/SEG32 15
PG7/SEG31 16
HT69F360/HT69V360
64 LQFP-A
48 PA7/RES
47 PD0/SEG0/INT1/PTCK1
46 PD1/SEG1/CTCK
45 PD2/SEG2/STCK
44 PD3/SEG3/PTCK0
43 PD4/SEG4/PTP0
42 PD5/SEG5/PTP0B
41 PD6/SEG6/PTP1
40 PD7/SEG7/PTP1B
39 PE0/SEG8
38 PE1/SEG9
37 PE2/SEG10
36 PE3/SEG11
35 PE4/SEG12
34 PE5/SEG13
33 PE6/SEG14
17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
PG6/SEG30
PG5/SEG29
PG4/SEG28
PG3/SEG27
PG2/SEG26
PG1/SEG25
PG0/SEG24
PF7/SEG23
PF6/SEG22
PF5/SEG21
PF4/SEG20
PF3/SEG19
PF2/SEG18
PF1/SEG17
PF0/SEG16
PE7/SEG15



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

2. The OCDSDA and OCDSCK pins are the OCDS dedicated pins and only available for the EV chip of the series of devices. The EV chip supports the "On-Chip-Debug" function for debugging during development using the OCDSDA and OCDSCK pins connected to the Holtek HT-IDE development tools.

3. For the less pin count package type there will be unbounded pins which should be properly configured to avoid unwanted power consumption resulting from floating input conditions. Refer to the "Standby Current Considerations" and "Input/Output Ports" sections.

Pin Descriptions

With the exception of the power pins and some relevant transformer control 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 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 each Pin Description table shows the situation for the package with the most pins, not all pins in the tables will be available on smaller package sizes.

HT69F340

Pin Name	Function	OPT	I/T	O/T	Description
PA0/INT1/ICPDA/ OCDSDA	PA0	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	INT1	INTEG INTC0 SFS	ST	—	External Interrupt 1
	ICPDA	—	ST	CMOS	ICP address/data
	OCDSDA	—	ST	CMOS	OCDS address/data, for EV chip only
PA1/CTP_1/SDO	PA1	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	CTP_1	PAFS	—	CMOS	CTM output
	SDO	PAFS	—	CMOS	SPI data output
PA2/CTCK/ ICPCK/OCDSCK	PA2	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	CTCK	SFS	ST	—	CTM clock input
	ICPCK	—	ST	—	ICP clock pin
	OCDSCK	—	ST	—	OCDS clock pin, for EV chip only
PA3/PTCK/SDI/SDA	PA3	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	PTCK	SFS	ST	—	PTM clock input
	SDI	PAFS	ST	—	SPI data input
	SDA	PAFS	ST	NMOS	I²C data line
PA4/INT0	PA4	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	INT0	INTEG INTC0	ST	—	External Interrupt 0
PA5/PTP_2/SCK/SCL	PA5	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	PTP_2	PAFS	—	CMOS	PTM output
	SCK	PAFS	ST	CMOS	SPI Serial Clock
	SCL	PAFS	ST	NMOS	I²C Clock
PA6/CTP_0/$\overline{SCS}$	PA6	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	CTP_0	PAFS	—	CMOS	CTM output
	$\overline{SCS}$	PAFS	ST	CMOS	SPI Slave select
PA7/$\overline{RES}$	PA7	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	$\overline{RES}$	CO	ST	—	External reset pin

Pin Name	Function	OPT	I/T	O/T	Description
PB0/OSC1	PB0	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	OSC1	SCC HXTC PBFS	HXT	—	HXT oscillator pin
PB1/OSC2	PB1	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	OSC2	SCC HXTC PBFS	—	HXT	HXT oscillator pin
PB2/XT1	PB2	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	XT1	SCC LXTC PBFS	LXT	—	LXT oscillator pin
PB3/XT2	PB3	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	XT2	SCC LXTC PBFS	—	LXT	LXT oscillator pin
PC0/V2	PC0	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	V2	—	—	AO	LCD voltage pump
PC1/C1	PC1	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	C1	—	—	AO	LCD voltage pump
PC2/C2	PC2	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	C2	—	—	AO	LCD voltage pump
PD0/SEG0/INT1	PD0	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG0	PDFS	—	AO	LCD driver output for LCD panel segment
	INT1	SFS	ST	—	External Interrupt 1
PD1/SEG1/CTCK	PD1	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG1	PDFS	—	AO	LCD driver output for LCD panel segment
	CTCK	SFS	ST	—	CTM clock input
PD2/SEG2/CTP_2	PD2	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG2	PDFS SFS	—	AO	LCD driver output for LCD panel segment
	CTP_2	PDFS SFS	—	CMOS	CTM output
PD3/SEG3/PTCK	PD3	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG3	PDFS	—	AO	LCD driver output for LCD panel segment
	PTCK	SFS	ST	—	PTM clock input
PD4/SEG4/PTP_0	PD4	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG4	PDFS SFS	—	AO	LCD driver output for LCD panel segment
	PTP_0	PDFS SFS	—	CMOS	PTM output

Pin Name	Function	OPT	I/T	O/T	Description
PD5/SEG5/PTPB_0	PD5	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG5	PDFS SFS	—	AO	LCD driver output for LCD panel segment
	PTPB_0	PDFS SFS	—	CMOS	PTM inverted output
PD6/SEG6/PTP_1	PD6	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG6	PDFS SFS	—	AO	LCD driver output for LCD panel segment
	PTP_1	PDFS SFS	—	CMOS	PTM output
PD7/SEG7/PTPB_1	PD7	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG7	PDFS SFS	—	AO	LCD driver output for LCD panel segment
	PTPB_1	PDFS SFS	—	CMOS	PTM inverted output
PE0/SEG8	PE0	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG8	PEFS	—	AO	LCD driver output for LCD panel segment
PE1/SEG9	PE1	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG9	PEFS	—	AO	LCD driver output for LCD panel segment
PE2/SEG10	PE2	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG10	PEFS	—	AO	LCD driver output for LCD panel segment
PE3/SEG11	PE3	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG11	PEFS	—	AO	LCD driver output for LCD panel segment
PE4/SEG12	PE4	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG12	PEFS	—	AO	LCD driver output for LCD panel segment
PE5/SEG13	PE5	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG13	PEFS	—	AO	LCD driver output for LCD panel segment
PE6/SEG14	PE6	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG14	PEFS	—	AO	LCD driver output for LCD panel segment
PE7/SEG15	PE7	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG15	PEFS	—	AO	LCD driver output for LCD panel segment
PF0/SEG16	PF0	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG16	PFFS	—	AO	LCD driver output for LCD panel segment
PF1/SEG17	PF1	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG17	PFFS	—	AO	LCD driver output for LCD panel segment
PF2/SEG18	PF2	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG18	PFFS	—	AO	LCD driver output for LCD panel segment

Pin Name	Function	OPT	I/T	O/T	Description
PF3/SEG19	PF3	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG19	PFFS	—	AO	LCD driver output for LCD panel segment
PF4/SEG20	PF4	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG20	PFFS	—	AO	LCD driver output for LCD panel segment
PF5/SEG21	PF5	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG21	PFFS	—	AO	LCD driver output for LCD panel segment
PF6/SEG22	PF6	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG22	PFFS	—	AO	LCD driver output for LCD panel segment
PF7/SEG23	PF7	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG23	PFFS	—	AO	LCD driver output for LCD panel segment
COM0~COM2	COMn	—	—	AO	LCD driver output for LCD panel common
COM3/SEG24	COM3	—	—	AO	LCD driver output for LCD panel common
	SEG24	—	—	AO	LCD driver output for LCD panel segment
V1	V1	—	—	AO	LCD voltage pump
VLCD	VLCD	—	PWR	—	LCD power supply
VMAX	VMAX	—	PWR	—	IC maximum voltage, connect to VDD, VLCD or V1
VDD	VDD	—	PWR	—	Power Supply
VSS	VSS	—	PWR	—	Ground

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

HT69F350

Pin Name	Function	OPT	I/T	O/T	Description
PA0/INT1/PTCK/ ICPDA/OCDSDA	PA0	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	INT1	INTEG INTC0 SFSR	ST	—	External Interrupt 1
	PTCK	SFSR	ST	—	PTM clock input
	ICPDA	—	ST	CMOS	ICP address/data
	OCDSDA	—	ST	CMOS	OCDS address/data, for EV chip only
PA1/CTP_1/SDO	PA1	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	CTP_1	PAFS	—	CMOS	CTM output
	SDO	PAFS	—	CMOS	SPI Data output

Pin Name	Function	OPT	I/T	O/T	Description
PA2/CTCK/STCK/ ICPCK/OCDSCK	PA2	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	CTCK	SFSR	ST	—	CTM clock input
	STCK	SFSR	ST	—	STM clock input
	ICPCK	—	ST	—	ICP clock pin
	OCDSCK	—	ST	—	OCDS clock pin, for EV chip only
PA3/STP_0/SDI/SDA	PA3	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	STP_0	—	—	CMOS	STM output
	SDI	PAFS	ST	—	SPI data input
	SDA	PAFS	ST	NMOS	I²C data line
PA4/INT0	PA4	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	INT0	INTEG INTC0	ST	—	External Interrupt 0
PA5/STP_1/SCK/SCL	PA5	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	STP_1	—	—	CMOS	STM output
	SCK	PAFS	ST	CMOS	SPI Serial Clock
	SCL	PAFS	ST	NMOS	I²C Clock
PA6/CTP_0/$\overline{SCS}$	PA6	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	CTP_0	—	—	CMOS	CTM output
	$\overline{SCS}$	PAFS	ST	CMOS	SPI Slave select
PA7/$\overline{RES}$	PA7	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	$\overline{RES}$	CO	ST	—	External reset pin
PB0/OSC1	PB0	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	OSC1	SCC HXTC PBFS	HXT	—	HXT oscillator pin
PB1/OSC2	PB1	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	OSC2	SCC HXTC PBFS	—	HXT	HXT oscillator pin
PB2/XT1	PB2	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	XT1	SCC LXTC PBFS	LXT	—	LXT oscilliator pin
PB3/XT2	PB3	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	XT2	SCC LXTC PBFS	—	LXT	LXT oscilliator pin
PB4/PTP_0	PB4	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	PTP_0	PBFS	—	CMOS	PTM output
PB5/PTPB_0	PB5	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	PTPB_0	PBFS	—	CMOS	PTM inverted output

Pin Name	Function	OPT	I/T	O/T	Description
PB6/PTP_1	PB6	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	PTP_1	PBFS	—	CMOS	PTM output
PB7/PTPB_1	PB7	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	PTPB_1	PBFS	—	CMOS	PTM inverted output
PC0/V2	PC0	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	V2	—	—	AO	LCD voltage pump
PC1/C1	PC1	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	C1	—	—	AO	LCD voltage pump
PC2/C2	PC2	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	C2	—	—	AO	LCD voltage pump
PD0/SEG0/INT1	PD0	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG0	PDFS	—	AO	LCD driver output for LCD panel segment
	INT1	INTEG INTC0 SFSR	ST	—	External Interrupt 1
PD1/SEG1/CTCK	PD1	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG1	PDFS	—	AO	LCD driver output for LCD panel segment
	CTCK	SFSR	ST	—	CTM clock input
PD2/SEG2/STCK	PD2	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG2	PDFS	—	AO	LCD driver output for LCD panel segment
	STCK	SFSR	ST	—	STM clock input
PD3/SEG3/PTCK	PD3	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG3	PDFS	—	AO	LCD driver output for LCD panel segment
	PTCK	SFSR	ST	—	PTM clock input
PD4/SEG4/PTP_0	PD4	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG4	PDFS SFSR	—	AO	LCD driver output for LCD panel segment
	PTP_0	PDFS SFSR	—	CMOS	PTM output
PD5/SEG5/PTPB_0	PD5	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG5	PDFS SFSR	—	AO	LCD driver output for LCD panel segment
	PTPB_0	PDPS SFSR	—	CMOS	PTM inverted output
PD6/SEG6/PTP_1	PD6	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG6	PDFS SFSR	—	AO	LCD driver output for LCD panel segment
	PTP_1	PDPS SFSR	—	CMOS	PTM output
PD7/SEG7/PTPB_1	PD7	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG7	PDFS SFSR	—	AO	LCD driver output for LCD panel segment
	PTPB_1	PDPS SFSR	—	CMOS	PTM inverted output

Pin Name	Function	OPT	I/T	O/T	Description
PE0/SEG8	PE0	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG8	PEFS	—	AO	LCD driver output for LCD panel segment
PE1/SEG9	PE1	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG9	PEFS	—	AO	LCD driver output for LCD panel segment
PE2/SEG10	PE2	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG10	PEFS	—	AO	LCD driver output for LCD panel segment
PE3/SEG11	PE3	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG11	PEFS	—	AO	LCD driver output for LCD panel segment
PE4/SEG12	PE4	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG12	PEFS	—	AO	LCD driver output for LCD panel segment
PE5/SEG13	PE5	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG13	PEFS	—	AO	LCD driver output for LCD panel segment
PE6/SEG14	PE6	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG14	PEFS	—	AO	LCD driver output for LCD panel segment
PE7/SEG15	PE7	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG15	PEFS	—	AO	LCD driver output for LCD panel segment
PF0/SEG16	PF0	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG16	PFFS	—	AO	LCD driver output for LCD panel segment
PF1/SEG17	PF1	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG17	PFFS	—	AO	LCD driver output for LCD panel segment
PF2/SEG18	PF2	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG18	PFFS	—	AO	LCD driver output for LCD panel segment
PF3/SEG19	PF3	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG19	PFFS	—	AO	LCD driver output for LCD panel segment
PF4/SEG20	PF4	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG20	PFFS	—	AO	LCD driver output for LCD panel segment
PF5/SEG21	PF5	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG21	PFFS	—	AO	LCD driver output for LCD panel segment
PF6/SEG22	PF6	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG22	PFFS	—	AO	LCD driver output for LCD panel segment
PF7/SEG23	PF7	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG23	PFFS	—	AO	LCD driver output for LCD panel segment
PG0/SEG24	PG0	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG24	PGFS	—	AO	LCD driver output for LCD panel segment
PG1/SEG25	PG1	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG25	PGFS	—	AO	LCD driver output for LCD panel segment

Pin Name	Function	OPT	I/T	O/T	Description
PG2/SEG26	PG2	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG26	PGFS	—	AO	LCD driver output for LCD panel segment
PG3/SEG27	PG3	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG27	PGFS	—	AO	LCD driver output for LCD panel segment
PG4/SEG28	PG4	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG28	PGFS	—	AO	LCD driver output for LCD panel segment
PG5/SEG29	PG5	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG29	PGFS	—	AO	LCD driver output for LCD panel segment
PG6/SEG30	PG6	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG30	PGFS	—	AO	LCD driver output for LCD panel segment
PG7/SEG31	PG7	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG31	PGFS	—	AO	LCD driver output for LCD panel segment
PH0/SEG32	PH0	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG32	PHFS	—	AO	LCD driver output for LCD panel segment
PH1/SEG33	PH1	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG33	PHFS	—	AO	LCD driver output for LCD panel segment
PH2/SEG34	PH2	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG34	PHFS	—	AO	LCD driver output for LCD panel segment
PH3/SEG35	PH3	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG35	PHFS	—	AO	LCD driver output for LCD panel segment
COM3/SEGn	COM3	—	—	AO	LCD driver outputs for LCD panel common
	SEGn	—	—	AO	LCD driver output for LCD panel segment. n=24 for 48-pin package, n=36 for 64-pin package
COM0~COM2	COMn	—	—	AO	LCD driver outputs for LCD panel common
V1	V1	—	—	AO	LCD voltage pump
VMAX	VMAX	—	PWR	—	IC maximum voltage, connect to VDD, VLCD or V1
VLCD	VLCD	—	PWR	—	LCD power supply
VDD	VDD	—	PWR	—	Power supply
VSS	VSS	—	PWR	—	Ground

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

HT69F360

Pin Name	Function	OPT	I/T	O/T	Description
PA0/INT1/PTCK0/ ICPDA/OCDSDA	PA0	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	INT1	INTEG INTC0 SFSR	ST	—	External Interrupt 1
	PTCK0	SFSR	ST	—	PTM0 clock input
	ICPDA	—	ST	CMOS	ICP address/data
	OCDSDA	—	ST	CMOS	OCDS address/data, for EV chip only.
PA1/CTP_1/SDO	PA1	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	CTP_1	PAFS	—	CMOS	CTM output
	SDO	PAFS	—	CMOS	SPI Data output
PA2/CTCK/STCK/ ICPCK/OCDSCK	PA2	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	CTCK	SFSR	ST	—	CTM clock input
	STCK	SFSR	ST	—	STM clock input
	ICPCK	—	ST	—	ICP clock pin
	OCDSCK	—	ST	—	OCDS clock pin, for EV chip only
PA3/STP_0/SDI/SDA	PA3	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	STP_0	PAFS	—	CMOS	STM output
	SDI	PAFS	ST	—	SPI data input
	SDA	PAFS	ST	NMOS	I^2C data line
PA4/INT0/PTCK1	PA4	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	INT0	INTEG INTC0	ST	—	External Interrupt 0
	PTCK1	—	ST	—	PTM1 clock input
PA5/STP_1/SCK/SCL	PA5	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	STP_1	PAFS	—	CMOS	STM output
	SCK	PAFS	ST	CMOS	SPI Serial Clock
	SCL	PAFS	ST	NMOS	I^2C Clock
PA6/CTP_0/$\overline{SCS}$	PA6	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	CTP_0	PAFS	—	CMOS	CTM output
	$\overline{SCS}$	PAFS	ST	CMOS	SPI Slave select
PA7/$\overline{RES}$	PA7	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up
	$\overline{RES}$	CO	ST	—	External reset pin
PB0/OSC1	PB0	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	OSC1	SCC HXTC PBFS	HXT	—	HXT oscillator pin
PB1/OSC2	PB1	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	OSC2	SCC HXTC PBFS	—	HXT	HXT oscillator pin

Pin Name	Function	OPT	I/T	O/T	Description
PB2/XT1	PB2	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	XT1	SCC LXTC PBFS	LXT	—	LXT oscillator pin
PB3/XT2	PB3	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	XT2	SCC LXTC PBFS	—	LXT	LXT oscillator pin
PB4/PTP0	PB4	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	PTP0	PBFS	—	CMOS	PTM0 output
PB5/PTP0B	PB5	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	PTP0B	PBFS	—	CMOS	PTM0 inverted output
PB6/PTP1/RX	PB6	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	PTP1	PBFS	—	CMOS	PTM1 output
	RX	SFSR1	ST	—	UART data received input
PB7/PTP1B/TX	PB7	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	PTP1B	PBFS	—	CMOS	PTM1 inverted output
	TX	PBFS SFSR1	—	CMOS	UART data transmission output
PC0/V2	PC0	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	V2	—	—	AO	LCD voltage pump
PC1/C1	PC1	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	C1	—	—	AO	LCD voltage pump
PC2/C2	PC2	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high
	C2	—	—	AO	LCD voltage pump
PC3~PC6	PC3~PC6	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high
PD0/SEG0/INT1/ PTCK1	PD0	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG0	PDFS	—	AO	LCD driver output for LCD panel segment
	INT1	INTEG INTC0 SFSR	ST	—	External Interrupt 1
	PTCK1	SFSR1	ST	—	PTM1 clock input
PD1/SEG1/CTCK	PD1	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG1	PDFS	—	AO	LCD driver output for LCD panel segment
	CTCK	SFSR	ST	—	CTM clock input
PD2/SEG2/STCK	PD2	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG2	PDFS	—	AO	LCD driver output for LCD panel segment
	STCK	SFSR	ST	—	STM clock input
PD3/SEG3/PTCK0	PD3	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG3	PDFS	—	AO	LCD driver output for LCD panel segment
	PTCK0	SFSR	ST	—	PTM0 clock input

Pin Name	Function	OPT	I/T	O/T	Description
PD4/SEG4/PTP0	PD4	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG4	PDFS SFSR	—	AO	LCD driver output for LCD panel segment
	PTP0	PDFS SFSR	—	CMOS	PTM0 output
PD5/SEG5/PTP0B	PD5	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG5	PDFS SFSR	—	AO	LCD driver output for LCD panel segment
	PTP0B	PDPS SFSR	—	CMOS	PTM0 inverted output
PD6/SEG6/PTP1	PD6	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG6	PDFS SFSR	—	AO	LCD driver output for LCD panel segment
	PTP1	PDPS SFSR	—	CMOS	PTM1 output
PD7/SEG7/PTP1B	PD7	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG7	PDFS SFSR	—	AO	LCD driver output for LCD panel segment
	PTP1B	PDPS SFSR	—	CMOS	PTM1 inverted output
PE0/SEG8	PE0	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG8	PEFS	—	AO	LCD driver output for LCD panel segment
PE1/SEG9	PE1	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG9	PEFS	—	AO	LCD driver output for LCD panel segment
PE2/SEG10	PE2	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG10	PEFS	—	AO	LCD driver output for LCD panel segment
PE3/SEG11	PE3	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG11	PEFS	—	AO	LCD driver output for LCD panel segment
PE4/SEG12	PE4	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG12	PEFS	—	AO	LCD driver output for LCD panel segment
PE5/SEG13	PE5	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG13	PEFS	—	AO	LCD driver output for LCD panel segment
PE6/SEG14	PE6	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG14	PEFS	—	AO	LCD driver output for LCD panel segment
PE7/SEG15	PE7	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG15	PEFS	—	AO	LCD driver output for LCD panel segment
PF0/SEG16	PF0	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG16	PFFS	—	AO	LCD driver output for LCD panel segment
PF1/SEG17	PF1	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG17	PFFS	—	AO	LCD driver output for LCD panel segment

Pin Name	Function	OPT	I/T	O/T	Description
PF2/SEG18	PF2	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG18	PFFS	—	AO	LCD driver output for LCD panel segment
PF3/SEG19	PF3	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG19	PFFS	—	AO	LCD driver output for LCD panel segment
PF4/SEG20	PF4	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG20	PFFS	—	AO	LCD driver output for LCD panel segment
PF5/SEG21	PF5	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG21	PFFS	—	AO	LCD driver output for LCD panel segment
PF6/SEG22	PF6	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG22	PFFS	—	AO	LCD driver output for LCD panel segment
PF7/SEG23	PF7	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG23	PFFS	—	AO	LCD driver output for LCD panel segment
PG0/SEG24	PG0	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG24	PGFS	—	AO	LCD driver output for LCD panel segment
PG1/SEG25	PG1	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG25	PGFS	—	AO	LCD driver output for LCD panel segment
PG2/SEG26	PG2	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG26	PGFS	—	AO	LCD driver output for LCD panel segment
PG3/SEG27	PG3	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG27	PGFS	—	AO	LCD driver output for LCD panel segment
PG4/SEG28	PG4	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG28	PGFS	—	AO	LCD driver output for LCD panel segment
PG5/SEG29	PG5	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG29	PGFS	—	AO	LCD driver output for LCD panel segment
PG6/SEG30	PG6	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG30	PGFS	—	AO	LCD driver output for LCD panel segment
PG7/SEG31	PG7	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG31	PGFS	—	AO	LCD driver output for LCD panel segment
PH0/SEG32	PH0	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG32	PHFS	—	AO	LCD driver output for LCD panel segment
PH1/SEG33	PH1	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG33	PHFS	—	AO	LCD driver output for LCD panel segment
PH2/SEG34	PH2	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG34	PHFS	—	AO	LCD driver output for LCD panel segment

Pin Name	Function	OPT	I/T	O/T	Description
PH3/SEG35	PH3	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG35	PHFS	—	AO	LCD driver output for LCD panel segment
PH4/SEG36	PH4	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG36	PHFS	—	AO	LCD driver output for LCD panel segment
PH5/SEG37	PH5	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG37	PHFS	—	AO	LCD driver output for LCD panel segment
PH6/SEG38	PH6	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG38	PHFS	—	AO	LCD driver output for LCD panel segment
PH7/SEG39	PH7	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-high
	SEG39	PHFS	—	AO	LCD driver output for LCD panel segment
SEG40~SEG47	SEGn	—	—	AO	LCD driver output for LCD panel segment
COM3/SEGn	COM3	—	—	AO	LCD driver outputs for LCD panel common
	SEGn	—	—	AO	LCD driver output for LCD panel segment. n=36 for 64-pin package, n=48 for 80-pin package
COM0~COM2	COMn	—	—	AO	LCD driver outputs for LCD panel common
V1	V1	—	—	AO	LCD voltage pump
VMAX	VMAX	—	PWR	—	IC maximum voltage, connect to VDD, VLCD or V1
VLCD	VLCD	—	PWR	—	LCD power supply
VDD	VDD	—	PWR	—	Power supply
VSS	VSS	—	PWR	—	Ground

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

HT67F370

Pin Name	Function	OPT	I/T	O/T	Description
PA0/INT1/PTCK0/ ICPDA/OCDSDA	PA0	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	INT1	INTEG INTC0 SFSR	ST	—	External Interrupt 1
	PTCK0	SFSR	ST	—	PTM0 clock input
	ICPDA	—	ST	CMOS	ICP address/data
	OCDSDA	—	ST	CMOS	OCDS address/data, for EV chip only
PA1/CTP/SDO/ AN3	PA1	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	CTP	PAFS	—	CMOS	CTM output
	SDO	PAFS	—	CMOS	SPI data output
	AN3	PAFS	AN	—	A/D Converter external input channel
PA2/CTCK/STCK/ ICPCK/OCDSCK	PA2	PAWU PAPU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	CTCK	SFSR	ST	—	CTM clock input
	STCK	SFSR	ST	—	STM clock input
	ICPCK	—	ST	—	ICP clock pin
	OCDSCK	—	ST	—	OCDS clock pin, for EV chip only.
PA3/STP/SDI/ SDA/AN4	PA3	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	STP	PAFS	ST	CMOS	STM input/output
	SDI	PAFS	ST	—	SPI serial data input
	SDA	PAFS	ST	NMOS	I^2C data line
	AN4	PAFS	AN	—	A/D Converter external input channel
PA4/INT0/PTCK1/ AN5	PA4	PAWU PAPU SFSR1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	INT0	INTEG INTC0 SFSR1	ST	—	External Interrupt 0
	PTCK1	SFSR1	ST	—	PTM1 clock input
	AN5	SFSR1	AN	—	A/D Converter external input channel
PA5/STP/SCK/ SCL/AN6	PA5	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	STP	PAFS	ST	CMOS	STM input/output
	SCK	PAFS	ST	CMOS	SPI serial clock
	SCL	PAFS	ST	NMOS	I^2C clock line
	AN6	PAFS	AN	—	A/D Converter external input channel
PA6/CTP/$\overline{SCS}$/AN7	PA6	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	CTP	PAFS	—	CMOS	CTM output
	$\overline{SCS}$	PAFS	ST	CMOS	SPI slave select
	AN7	PAFS	AN	—	A/D Converter external input channel

Pin Name	Function	OPT	I/T	O/T	Description
PA7/$\overline{RES}$	PA7	PAWU PAPU CO	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up
	$\overline{RES}$	CO	ST	—	External reset pin
PB0/OSC1	PB0	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	OSC1	SCC HXTC PBFS	HXT	—	HXT oscillator pin
PB1/OSC2	PB1	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	OSC2	SCC HXTC PBFS	—	HXT	HXT oscillator pin
PB2/XT1	PB2	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	XT1	SCC LXTC PBFS	LXT	—	LXT oscillator pin
PB3/XT2	PB3	PBPU PBFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	XT2	SCC LXTC PBFS	—	LXT	LXT oscillator pin
PB4/PTP0/VREFI	PB4	PBPU PBFS SFSR1	ST	CMOS	General purpose I/O. Register enabled pull-up
	PTP0	PBFS SFSR1	ST	CMOS	PTM0 input/output
	VREFI	PBFS SFSR1	AN	—	A/D Converter PGA input
PB5/PTP0B/VREF/AN0	PB5	PBPU PBFS SFSR1	ST	CMOS	General purpose I/O. Register enabled pull-up
	PTP0B	PBFS SFSR1	ST	CMOS	PTM0 input/inverted output
	VREF	PBFS SFSR1	AN	—	A/D Converter external reference voltage input
	AN0	PBFS SFSR1	AN	—	A/D Converter external input channel
PB6/PTP1/RX/AN1	PB6	PBPU PBFS SFSR1	ST	CMOS	General purpose I/O. Register enabled pull-up
	PTP1	PBFS SFSR1	ST	CMOS	PTM1 input/output
	RX	PBFS SFSR1	ST	—	UART data received input
	AN1	PBFS SFSR1	AN	—	A/D Converter external input channel

Pin Name	Function	OPT	I/T	O/T	Description
PB7/PTP1B/TX/AN2	PB7	PBPU PBFS SFSR1	ST	CMOS	General purpose I/O. Register enabled pull-up
	PTP1B	PBFS SFSR1	ST	CMOS	PTM1 input/inverted output
	TX	PBFS SFSR1	—	CMOS	UART data transmission output
	AN2	PBFS SFSR1	AN	—	A/D Converter external input channel
PC0/V2	PC0	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	V2	—	—	AN	LCD voltage pump
PC1/C1	PC1	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	C1	—	—	AN	LCD voltage pump
PC2/C2	PC2	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	C2	—	—	AN	LCD voltage pump
PC3/AN8	PC3	PCPU PCFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN8	PCFS	AN	—	A/D Converter external input channel
PC4/AN9	PC4	PCPU PCFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN9	PCFS	AN	—	A/D Converter external input channel
PC5/AN10	PC5	PCPU PCFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN10	PCFS	AN	—	A/D Converter external input channel
PC6/AN11	PC6	PCPU PCFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	AN11	PCFS	AN	—	A/D Converter external input channel
PD0/SEG0/INT1/PTCK1	PD0	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG0	PDFS	—	AN	LCD driver output for LCD panel segment
	INT1	INTEG INTC0 PDFS SFSR	ST	—	External Interrupt 1
	PTCK1	PDFS SFSR1	ST	—	PTM1 clock input
PD1/SEG1/CTCK	PD1	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG1	PDFS	—	AN	LCD driver output for LCD panel segment
	CTCK	PDFS SFSR	ST	—	CTM clock input
PD2/SEG2/STCK	PD2	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG2	PDFS	—	AN	LCD driver output for LCD panel segment
	STCK	PDFS SFSR	ST	—	STM clock input
PD3/SEG3/PTCK0	PD3	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG3	PDFS	—	AN	LCD driver output for LCD panel segment
	PTCK0	PDFS SFSR	ST	—	PTM0 clock input

Pin Name	Function	OPT	I/T	O/T	Description
PD4/SEG4/PTP0	PD4	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG4	PDFS SFSR	—	AN	LCD driver output for LCD panel segment
	PTP0	PDFS SFSR	ST	CMOS	PTM0 input/output
PD5/SEG5/PTP0B	PD5	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG5	PDFS SFSR	—	AN	LCD driver output for LCD panel segment
	PTP0B	PDFS SFSR	ST	CMOS	PTM0 input/inverted output
PD6/SEG6/PTP1	PD6	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG6	PDFS SFSR	—	AN	LCD driver output for LCD panel segment
	PTP1	PDFS SFSR	ST	CMOS	PTM1 input/output
PD7/SEG7/PTP1B	PD7	PDPU PDFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG7	PDFS SFSR	—	AN	LCD driver output for LCD panel segment
	PTP1B	PDFS SFSR	ST	CMOS	PTM1 input/inverted output
PE0/SEG8	PE0	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG8	PEFS	—	AN	LCD driver output for LCD panel segment
PE1/SEG9	PE1	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG9	PEFS	—	AN	LCD driver output for LCD panel segment
PE2/SEG10	PE2	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG10	PEFS	—	AN	LCD driver output for LCD panel segment
PE3/SEG11	PE3	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG11	PEFS	—	AN	LCD driver output for LCD panel segment
PE4/SEG12	PE4	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG12	PEFS	—	AN	LCD driver output for LCD panel segment
PE5/SEG13	PE5	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG13	PEFS	—	AN	LCD driver output for LCD panel segment
PE6/SEG14	PE6	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG14	PEFS	—	AN	LCD driver output for LCD panel segment
PE7/SEG15	PE7	PEPU PEFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG15	PEFS	—	AN	LCD driver output for LCD panel segment
PF0/SEG16	PF0	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG16	PFFS	—	AN	LCD driver output for LCD panel segment

Pin Name	Function	OPT	I/T	O/T	Description
PF1/SEG17	PF1	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG17	PFFS	—	AN	LCD driver output for LCD panel segment
PF2/SEG18	PF2	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG18	PFFS	—	AN	LCD driver output for LCD panel segment
PF3/SEG19	PF3	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG19	PFFS	—	AN	LCD driver output for LCD panel segment
PF4/SEG20	PF4	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG20	PFFS	—	AN	LCD driver output for LCD panel segment
PF5/SEG21	PF5	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG21	PFFS	—	AN	LCD driver output for LCD panel segment
PF6/SEG22	PF6	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG22	PFFS	—	AN	LCD driver output for LCD panel segment
PF7/SEG23	PF7	PFPU PFFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG23	PFFS	—	AN	LCD driver output for LCD panel segment
PG0/SEG24	PG0	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG24	PGFS	—	AN	LCD driver output for LCD panel segment
PG1/SEG25	PG1	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG25	PGFS	—	AN	LCD driver output for LCD panel segment
PG2/SEG26	PG2	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG26	PGFS	—	AN	LCD driver output for LCD panel segment
PG3/SEG27	PG3	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG27	PGFS	—	AN	LCD driver output for LCD panel segment
PG4/SEG28	PG4	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG28	PGFS	—	AN	LCD driver output for LCD panel segment
PG5/SEG29	PG5	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG29	PGFS	—	AN	LCD driver output for LCD panel segment
PG6/SEG30	PG6	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG30	PGFS	—	AN	LCD driver output for LCD panel segment
PG7/SEG31	PG7	PGPU PGFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG31	PGFS	—	AN	LCD driver output for LCD panel segment
PH0/SEG32	PH0	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG32	PHFS	—	AN	LCD driver output for LCD panel segment
PH1/SEG33	PH1	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG33	PHFS	—	AN	LCD driver output for LCD panel segment

Pin Name	Function	OPT	I/T	O/T	Description
PH2/SEG34	PH2	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG34	PHFS	—	AN	LCD driver output for LCD panel segment
PH3/SEG35	PH3	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG35	PHFS	—	AN	LCD driver output for LCD panel segment
PH4/SEG36	PH4	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG36	PHFS	—	AN	LCD driver output for LCD panel segment
PH5/SEG37	PH5	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG37	PHFS	—	AN	LCD driver output for LCD panel segment
PH6/SEG38	PH6	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG38	PHFS	—	AN	LCD driver output for LCD panel segment
PH7/SEG39	PH7	PHPU PHFS	ST	CMOS	General purpose I/O. Register enabled pull-up
	SEG39	PHFS	—	AN	LCD driver output for LCD panel segment
SEG40~SEG47	SEGn	—	—	AN	LCD driver output for LCD panel segment
COM0~COM2	COMn	—	—	AN	LCD driver outputs for LCD panel common
COM3/SEGn	COM3	—	—	AN	LCD driver outputs for LCD panel common
	SEGn	—	—	AN	LCD driver output for LCD panel segment. n=36 for 64-pin package, n=48 for 80-pin package
V1	V1	—	—	AN	LCD voltage pump
VMAX	VMAX	—	PWR	—	IC maximum voltage, connect to VDD, VLCD or V1
VLCD	VLCD	—	PWR	—	LCD power supply
VDD	VDD	—	PWR	—	Power supply
VSS	VSS	—	PWR	—	Negative power supply, ground

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

Absolute Maximum Ratings

Supply Voltage ..... V_{SS}-0.3V to V_{SS}+6.0V
Input Voltage ..... V_{SS}-0.3V to V_{DD}+0.3V
Storage Temperature..... -60°C to 150°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 the device. Functional operation of this device at other conditions beyond those listed in the specification is not implied and prolonged exposure to extreme conditions may affect device reliability.

D.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage (HXT)	—	$f_{SYS}=f_{HXT}$=4MHz	1.8	—	5.5	V
			$f_{SYS}=f_{HXT}$=8MHz (HT67F370)	1.8	—	5.5	V
			$f_{SYS}=f_{HXT}$=8MHz (HT69F340/350/360)	2.0	—	5.5	V
			$f_{SYS}=f_{HXT}$=12MHz	2.7	—	5.5	V
			$f_{SYS}=f_{HXT}$=16MHz (HT67F370)	3.3	—	5.5	V
			$f_{SYS}=f_{HXT}$=16MHz (HT69F340/350/360)	4.5	—	5.5	V
	Operating Voltage (HIRC)	—	$f_{SYS}=f_{HIRC}$=4MHz	1.8	—	5.5	V
			$f_{SYS}=f_{HXT}$=8MHz (HT67F370)	1.8	—	5.5	V
			$f_{SYS}=f_{HIRC}$=8MHz (HT69F340/350/360)	2.0	—	5.5	V
			$f_{SYS}=f_{HIRC}$=12MHz	2.7	—	5.5	V
V_{IL}	Input Low Voltage for I/O Ports	5V	—	0.0	—	1.5	V
		—	—	0	—	$0.2V_{DD}$	V
	Input Low Voltage ($\overline{RES}$)	—	—	0	—	$0.4V_{DD}$	V
V_{IH}	Input High Voltage for I/O Ports	5V	—	3.5	—	5.0	V
		—	—	$0.8V_{DD}$	—	V_{DD}	V
	Input High Voltage ($\overline{RES}$)	—	—	$0.9V_{DD}$	—	V_{DD}	V

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{DD}	Operating Current (HXT)	3V	No load, all peripherals off, f_H=4MHz	—	0.50	0.75	mA
		5V		—	1.0	1.5	mA
		3V	No load, all peripherals off, f_H=8MHz	—	1.0	1.5	mA
		5V		—	2	3	mA
		3V	No load, all peripherals off, f_H=12MHz	—	1.50	2.75	mA
		5V		—	3.0	4.5	mA
		5V	No load, all peripherals off, f_H=16MHz	—	4	6	mA
	Operating Current (HIRC)	3V	No load, all peripherals off, f_H=4MHz	—	400	600	μA
		5V		—	0.8	1.2	mA
		3V	No load, all peripherals off, f_H=8MHz	—	0.8	1.2	mA
		5V		—	1.6	2.4	mA
		3V	No load, all peripherals off, f_H=12MHz	—	1.2	1.8	mA
		5V		—	2.4	3.6	mA
	Operating Current (LXT)	3V	No load, all peripherals off, f_{SYS}=32768Hz	—	10	20	μA
		5V		—	30	50	μA
	Operating Current (LIRC)	3V	No load, all peripherals off, f_{SYS}=32kHz	—	10	20	μA
		5V		—	30	50	μA
I_{STB}	Standby Current (SLEEP)	3V	No load, all peripherals off, WDT off	—	0.2	0.8	μA
		5V		—	0.5	1.0	μA
		3V	No load, all peripherals off, WDT on	—	1.5	3.0	μA
		5V		—	3	5	μA
	Standby Current (IDLE0)	3V	No load, all peripherals off, f_{SUB} on	—	3	5	μA
		5V		—	5	10	μA
	Standby Current (IDLE1, HIRC)	3V	No load, all peripherals off, f_{SUB} on, f_H=4MHz	—	180	270	μA
		5V		—	400	600	μA
		3V	No load, all peripherals off, f_{SUB} on, f_H=8MHz	—	360	500	μA
		5V		—	600	800	μA
		3V	No load, all peripherals off, f_{SUB} on, f_H=12MHz	—	540	750	μA
		5V		—	800	1200	μA
	Standby Current (IDLE1, HXT)	3V	No load, all peripherals off, f_{SUB} on, f_H=4MHz (HT67F370)	—	180	250	μA
		5V		—	400	600	μA
		3V	No load, all peripherals off, f_{SUB} on, f_H=4MHz (HT69F340/350/360)	—	180	270	μA
		5V		—	400	600	μA
		3V	No load, all peripherals off, f_{SUB} on, f_H=8MHz	—	360	500	μA
		5V		—	600	800	μA
		3V	No load, all peripherals off, f_{SUB} on, f_H=12MHz	—	540	750	μA
		5V		—	800	1200	μA
		5V	No load, all peripherals off, f_{SUB} on, f_H=16MHz (HT67F370)	—	1.4	2.0	mA
		5V	No load, all peripherals off, f_{SUB} on, f_H=16MHz (HT69F340/350/360)	—	1.10	1.65	mA
I_{OL}	Sink Current for I/O Ports (HT67F370)	1.8V	V_{OL}=0.1V_{DD}	7	14	—	mA
		3V		16	32	—	
		5V		32	64	—	
	Sink Current for I/O Ports (HT69F340/350/360)	3V	V_{OL}=0.1V_{DD}	15	30	—	mA
		5V		30	60	—	mA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{OH}	Source Current for I/O Ports (HT67F370)	1.8V	V_{OH}=0.9V_{DD}	-1.4	-2.8	—	mA
		3V		-4	-8	—	
		5V		-10	-20	—	
	Source Current for I/O Ports (HT69F340/350/360)	3V	V_{OH}=0.9V_{DD}	3.5	7.0	—	mA
		5V		7.5	15.0	—	mA
R_{PH}	Pull-high Resistance for I/O Ports (HT67F370)	3V	LVPU=0	40	120	200	kΩ
		5V		30	60	120	
		3V	LVPU=1	10	30	50	kΩ
		5V		5	15	25	
	Pull-high Resistance of I/O Ports (HT69F340/350/360)	3V	LVPU=0	60	120	240	kΩ
		5V		30	60	120	kΩ
		3V	LVPU=1	15	30	60	kΩ
		5V		7.5	15	30	kΩ
I_{LEAK}	Input Leakage Current	5V	V_{IN}=V_{DD} or V_{IN}=V_{SS}	—	—	±1	μA
V_{OH}	Output High Voltage for I/O Ports	3V	I_{OH}=2mA	2.7	—	—	V
		5V	I_{OH}=5mA	4.5	—	—	V
V_{OL}	Output Low Voltage for I/O Ports	3V	I_{OL}=4mA	—	—	0.3	V
		5V	I_{OL}=10mA	—	—	0.5	V

A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Condition				
f_{SYS}	System clock (HXT)	1.8V~5.5V	—	0.4	—	4.0	MHz
		2.0V~5.5V		0.4	—	8.0	MHz
		2.7V~5.5V		0.4	—	12.0	MHz
		4.5V~5.5V		0.4	—	16.0	MHz
	System clock (HIRC)	3V/5V	Ta=25°C	-2%	4	+2%	MHz
		3V/5V	Ta=25°C	-2%	8	+2%	MHz
		5V	Ta=25°C	-2%	12	+2%	MHz
		2.2V~3.6V	Ta=-40°C~85°C	-8%	4	+8%	MHz
		3.0V~5.5V	Ta=-40°C~85°C	-8%	4	+8%	MHz
		2.2V~3.6V	Ta=-40°C~85°C	-8%	8	+8%	MHz
		3.0V~5.5V	Ta=-40°C~85°C	-8%	8	+8%	MHz
		3.0V~5.5V	Ta=-40°C~85°C	-11%	12	+11%	MHz
	System Clock (LIRC)	5V	Ta=25°C	-10%	32	+10%	kHz
		2.2V~5.5V	Ta=-40°C~85°C	-50%	32	+60%	kHz
t_{EERD}	EEPROM Read Time	—	—	1	2	4	t_{SYS}
t_{EEWR}	EEPROM Write Time	—	—	1	2	4	ms
t_{TIMER}	xTCKn and TM Capture Input Minimum Pulse Width	—	—	0.3	—	—	μs
t_{RES}	External Reset Pin Minimum Pulse Width	—	—	10	—	—	μs
t_{INT}	External Interrupt Minimum Pulse Width	—	—	10	—	—	μs

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Condition				
t_{SST}	System Start-up Timer Period (Wake-up from HALT, f_{SYS} off at HALT state, Slow Mode → Normal Mode, Normal Mode → Slow Mode)	—	$f_{SYS}=f_{HXT}$ or f_{LXT} (Slow Mode→Normal Mode(HXT), Normal Mode→Slow Mode(LXT))	1024	—	—	t_{SYS}
			$f_{SYS}=f_{HXT}$ (Wake-up from HALT, f_{SYS} off at HALT state)	128	—	—	t_{SYS}
			$f_{SYS}=f_{LXT}$ (Wake-up from HALT, f_{SYS} off at HALT state)	1024	—	—	t_{SYS}
		—	$f_{SYS}=f_{HIRC}$	16	—	—	t_{SYS}
		—	$f_{SYS}=f_{LIRC}$	2	—	—	t_{SYS}
	System Start-up Timer Period (Wake-up from HALT, f_{SYS} on at HALT state)	—	—	2	—	—	t_{SYS}
t_{RSTD}	System Reset Delay Time (Power On Reset, LVR reset, LVR S/W reset(LVRC), WDT S/W reset(WDTC))	—	—	25	50	100	ms
	System Reset Delay Time ($\overline{RES}$ reset, WDT normal reset)	—	—	8.3	16.7	33.3	ms

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

2. To maintain the accuracy of the internal HIRC oscillator frequency, a 0.1μF decoupling capacitor should be connected between VDD and VSS and located as close to the device as possible.

A/D Converter Electrical Characteristics – HT67F370

Ta=-40°C~85°C, unless otherwise specify

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	1.8	—	5.5	V
V_{ADI}	Input Voltage	—	—	0	—	V_{REF}	V
V_{REF}	Reference Voltage	—	—	1.8	—	V_{DD}	V
DNL	Differential Non-linearity	1.8V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=2.0μs	—	—	±3	LSB
		2V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=0.5μs				
		3V					
		5V					
		1.8V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=10μs				
		3V					
		5V					
INL	Integral Non-linearity	1.8V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=2.0μs	—	—	±4	LSB
		2V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=0.5μs				
		3V					
		5V					
		1.8V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, $V_{REF}=V_{DD}$, t_{ADCK}=10μs				
		3V					
		5V					

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{ADC}	Additional Current for A/D Converter Enable	1.8V	No load, t_{ADCK}=2.0μs	—	0.5	1.0	mA
		3V	No load, t_{ADCK}=0.5μs	—	1.0	2.0	
		5V	No load, t_{ADCK}=0.5μs	—	1.5	3.0	
t_{ADCK}	Clock Period	—	1.8V≤V_{DD}<2.0V	2.0	—	10	μs
			2.0V≤V_{DD}≤5.5V	0.5	—	10	
t_{ON2ST}	A/D Converter On-to-Start Time	—	—	4	—	—	μs
t_{ADS}	Sampling Time	—	—	—	4	—	t_{ADCK}
t_{ADC}	Conversion Time (Including A/D Sample and Hold Time)	—	—	—	16	—	t_{ADCK}
I_{PGA}	Additional Current for PGA Enable	2.2V	No load	—	250	400	μA
		3V		—	300	450	
		5V		—	400	500	
V_{CM}	PGA Common Mode Voltage Range	2.2V	—	V_{SS}	—	V_{DD}-1.4	V
		3V		V_{SS}	—	V_{DD}-1.4	
		5V		V_{SS}	—	V_{DD}-1.4	
V_{OR}	PGA Maximum Output Voltage Range	2.2V	—	V_{SS}+0.1	—	V_{DD}-0.1	V
		3V		V_{SS}+0.1	—	V_{DD}-0.1	
		5V		V_{SS}+0.1	—	V_{DD}-0.1	
V_{VR}	Fix Voltage Output of PGA	2.2V~5.5V	Ta=-40°C~85°C	-1%	2	+1%	V
		3.2V~5.5V		-1%	3	+1%	
		4.2V~5.5V		-1%	4	+1%	

LVD & LVR Electrical Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{LVR}	Low Voltage Reset Voltage (HT67F370)	—	LVR enable, voltage select 1.7V	-5%	1.7	+5%	V
			LVR enable, voltage select 1.9V	-5%	1.9	+5%	
			LVR enable, voltage select 2.55V	-3%	2.55	+3%	
			LVR enable, voltage select 3.15V	-3%	3.15	+3%	
			LVR enable, voltage select 3.8V	-3%	3.8	+3%	
	Low Voltage Reset Voltage (HT69F340/350/360)	—	LVR Enable, voltage select 1.70V	-5%	1.7	+5%	V
			LVR Enable, voltage select 1.90V		1.9		
			LVR Enable, voltage select 2.55V		2.55		
			LVR Enable, voltage select 3.15V		3.15		
			LVR Enable, voltage select 3.80V		3.8		
V_{LVD}	Low Voltage Detector Voltage	—	LVDEN=1, V_{LVD}=1.8V	-5%	1.8	+5%	V
			LVDEN=1, V_{LVD}=2.0V		2.0		V
			LVDEN=1, V_{LVD}=2.4V		2.4		V
			LVDEN=1, V_{LVD}=2.7V		2.7		V
			LVDEN=1, V_{LVD}=3.0V		3.0		V
			LVDEN=1, V_{LVD}=3.3V		3.3		V
			LVDEN=1, V_{LVD}=3.6V		3.6		V
			LVDEN=1, V_{LVD}=4.0V		4.0		V

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{LVR}	Additional Power Consumption if LVR is used (HT67F370)	5V	LVD disable, PGAIS=0	—	—	8	μA
	Additional Power Consumption if LVR is used (HT69F340/350/360)	3V	LVR disable → LVR enable	—	10	20	μA
		5V		—	15	30	μA
I_{LVD}	Additional Power Consumption if LVD is used (HT67F370)	5V	LVR disable, PGAIS=0	—	—	8	μA
	Additional Power Consumption if LVD is used (HT69F340/350/360)	3V	LVD disable → LVD enable (LVR disable)	—	10	20	μA
		5V		—	15	30	μA
		3V	LVD disable → LVD enable (LVR enable)	—	1	2	μA
		5V		—	2	4	μA
t_{LVR}	Minimum Low Voltage Width to Reset	—	—	120	240	480	μs
t_{LVD}	Minimum Low Voltage Width to Interrupt	—	—	60	120	240	μs
t_{LVDS}	LVDO Stable Time (HT67F370)	—	For LVR enable, PGAIS=0, LVD off → on	—	—	15	μs
			For LVR disable, PGAIS=0, LVD off → on	—	—	150	
	LVDO Stable Time (HT69F340/350/360)	—	For LVR enable, LVD off → on	—	—	15	μs
		—	For LVR disable, LVD off → on	—	—	150	μs
$I_{LVRLVDBG}$	Operating Current (HT67F370)	3V	LVD enable, LVR enable, V_{LVR}=1.9V, V_{LVD}=2V, PGAIS=0	—	—	10	μA
		5V		—	8	15	
		3V	LVD enable, LVR enable, V_{LVR}=1.9V, V_{LVD}=2V, PGAIS=1	—	—	100	μA
		5V		—	80	115	
t_{SRESET}	Minimum Software Reset Width to Reset	—	—	45	90	120	μs

Internal Reference Voltage Electrical Characteristics – HT67F370

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{BG}	Bandgap Reference Voltage	—	—	-3%	1.2	+3%	V
I_{BG}	Additional Current for Bandgap Reference Enable	—	LVR disable, LVD disable	—	—	110	μA
t_{BGS}	V_{BG} Turn On Stable Time	—	No load	—	—	50	μs

Note: 1. All the above parameters are measured under conditions of no load condition unless otherwise described.

2. A 0.1μF ceramic capacitor should be connected between VDD and GND.

3. The V_{BG} voltage is used as the A/D converter reference voltage input.

LCD D.C. Characteristics

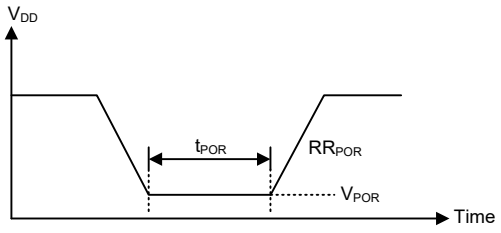
Ta=25°C

Symbol	Parameter	Test Condition		Min.	Typ.	Max.	Unit
		V_{DD}	Condition				
I_{STB}	Standby Current (Idle) (f_{SYS}, f_{WDT} off, f_S=f_{SUB}=32768 or 32K RC OSC)	3V	No load, system HALT, LCD on, WDT off, C type, V_{LCD}=V_{DD}, 1/2 Bias	—	6	10	μA
		5V		—	10	15	μA
	Standby Current (Idle) (f_{SYS}, f_{WDT} off, f_S=f_{SUB}=32768 or 32K RC OSC)	3V	No load, system HALT, LCD on, WDT off, C type, V_{LCD}=V_{DD}, 1/3 Bias	—	6	10	μA
		5V		—	10	15	μA
	Standby Current (Idle) (f_{SYS}, f_{WDT} off, f_S=f_{SUB}=32768 or 32K RC OSC)	3V	No load, system HALT, LCD on, WDT off, R type, V_{LCD}=V_{DD}, 1/2 bias (I_{BIAS}=7.5μA)	—	13.5	20.0	μA
		5V		—	22.5	40.0	μA
	Standby Current (Idle) (f_{SYS}, f_{WDT} off, f_S=f_{SUB}=32768 or 32K RC OSC)	3V	No load, system HALT, LCD on, WDT off, R type, V_{LCD}=V_{DD}, 1/2 bias (I_{BIAS}=15μA)	—	21	40	μA
		5V		—	35	60	μA
	Standby Current (Idle) (f_{SYS}, f_{WDT} off, f_S=f_{SUB}=32768 or 32K RC OSC)	3V	No load, system HALT, LCD on, WDT off, R type, V_{LCD}=V_{DD}, 1/2 bias (I_{BIAS}=45μA)	—	51	80	μA
		5V		—	85	160	μA
	Standby Current (Idle) (f_{SYS}, f_{WDT} off, f_S=f_{SUB}=32768 or 32K RC OSC)	3V	No load, system HALT, LCD on, WDT off, R type, V_{LCD}=V_{DD}, 1/2 bias (I_{BIAS}=90μA)	—	96	160	μA
		5V		—	160	320	μA
	Standby Current (Idle) (f_{SYS}, f_{WDT} off, f_S=f_{SUB}=32768 or 32K RC OSC)	3V	No load, system HALT, LCD on, WDT off, R type, V_{LCD}=V_{DD}, 1/3 bias (I_{BIAS}=7.5μA)	—	11	20	μA
		5V		—	18.3	40.0	μA
	Standby Current (Idle) (f_{SYS}, f_{WDT} off, f_S=f_{SUB}=32768 or 32K RC OSC)	3V	No load, system HALT, LCD on, WDT off, R type, V_{LCD}=V_{DD}, 1/3 bias (I_{BIAS}=15μA)	—	16	25	μA
		5V		—	26.6	50.0	μA
	Standby Current (Idle) (f_{SYS}, f_{WDT} off, f_S=f_{SUB}=32768 or 32K RC OSC)	3V	No load, system HALT, LCD on, WDT off, R type, V_{LCD}=V_{DD}, 1/3 bias (I_{BIAS}=45μA)	—	36	50	μA
		5V		—	60	100	μA
	Standby Current (Idle) (f_{SYS}, f_{WDT} off, f_S=f_{SUB}=32768 or 32K RC OSC)	3V	No load, system HALT, LCD on, WDT off, R type, V_{LCD}=V_{DD}, 1/3 bias (I_{BIAS}=90μA)	—	66	100	μA
		5V		—	110	200	μA
I_{OL}	LCD Common and Segment Sink Current	3V	V_{OL}=0.1V_{LCD}	210	420	—	μA
		5V		350	700	—	μA
I_{OH}	LCD Common and Segment Source Current	3V	V_{OH}=0.9V_{LCD}	-80	-160	—	μA
		5V		-180	-360	—	μA

Power-on Reset Electrical Characteristics

Ta=25°C

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

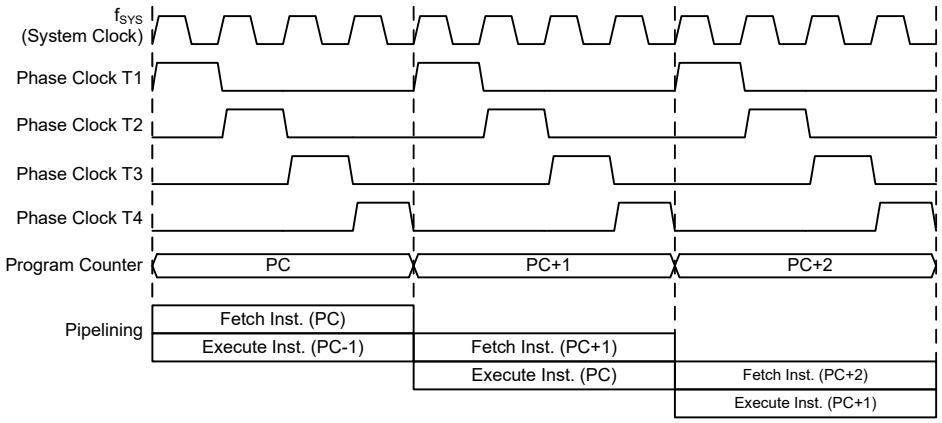


System Architecture

A key factor in the high-performance features of the Holtek range of microcontrollers is attributed to their internal system architecture. The series of devices take advantage of the usual features found within RISC microcontrollers providing increased speed of operation and Periodic performance. The pipelining scheme is implemented in such a way that instruction fetching and instruction execution are overlapped, hence instructions are effectively executed in one or two cycles for most of the standard or extended instructions respectively. with the exception of 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 (HT67F370 only) control system with maximum reliability and flexibility. This makes the devices suitable for low-cost, high-volume production for controller applications.

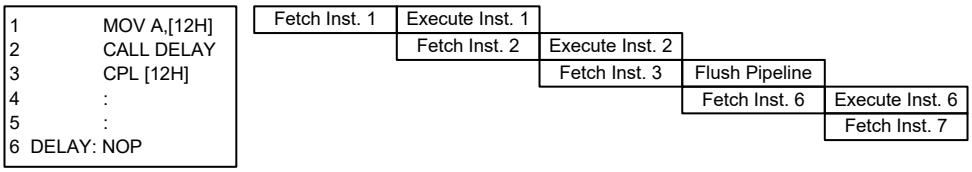
Clocking and Pipelining

The main system clock, derived from either a 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 Clock and Pipelining

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



Instruction Fetching

Program Counter

During program execution, the Program Counter is used to keep track of the address of the next instruction to be executed. It is automatically incremented by one each time an instruction is executed except for instructions, such as "JMP" or "CALL" that demand a jump to a non-consecutive Program Memory address. As the device memory capacity is greater than 8K words, the Program Memory address may be located in a certain program memory bank which is selected by the program memory bank pointer bits, PBP1~PBP0. 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.

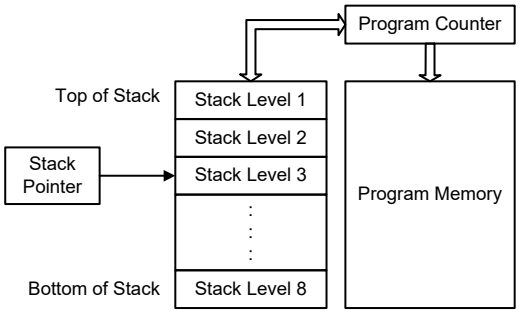
The lower byte of the Program Counter, known as the Program Counter Low register or PCL, is available for program control and is a readable and writeable register. By transferring data directly into this register, a short program jump can be executed directly, however, as only this low byte is available 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.

Device	Program Counter	
	Program Counter High Byte	PCL Register
HT69F340	PC11~PC8	PCL7~PCL0
HT69F350	PC12~PC8	PCL7~PCL0
HT69F360	PBP0, PC12~PC8	PCL7~PCL0
HT67F370	PBP1~PBP0, PC12~PC8	PCL7~PCL0

Stack

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

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



Arithmetic and Logic Unit – ALU

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

- Arithmetic operations: ADD, ADDM, ADC, ADCM, SUB, SUBM, SBC, SBCM, DAA, LADD, LADDM, LADC, LADCM, LSUB, LSUBM, LSBC, LSBCM, LDAA
- Logic operations: AND, OR, XOR, ANDM, ORM, XORM, CPL, CPLA, LAND, LANDM, LOR, LORM, LXOR, 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, LSNZ, LSZ, LSZA, LSIZ, LSIZA, LSDZ, LSDZA

Flash Program Memory

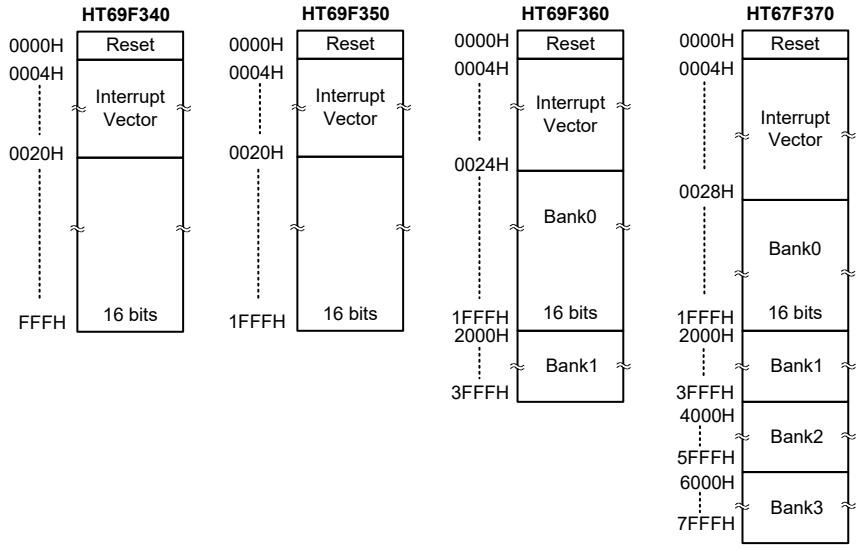
The Program Memory is the location where the user code or program is stored. For these devices the Program Memory is Flash type, which means it can be programmed and re-programmed a large number of times, allowing the user the convenience of code modification on the same device. By using the appropriate 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.

Structure

The Program Memory has a capacity of 4K×16 to 32K×16 bits. The Program Memory is addressed by the Program Counter and also contains data, table information and interrupt entries. Table data, which can be setup in any location within the Program Memory, is addressed by separate table pointer registers.

Device	Capacity	Banks
HT69F340	4K×16	—
HT69F350	8K×16	—
HT69F360	16K×16	0~1
HT67F370	32K×16	0~3

The HT69F360 device has its Program Memory divided into two Banks, Bank0~Bank1. The required Bank is selected using the Bit 0 of the PBP Register. The HT67F370 device has its Program Memory divided into four Banks, Bank0~Bank3. The required Bank is selected using the Bit 0~1 of the PBP 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 the device reset for program initialisation. After a device reset is initiated, the program will jump to this location and begin execution.

Look-up Table

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

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

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

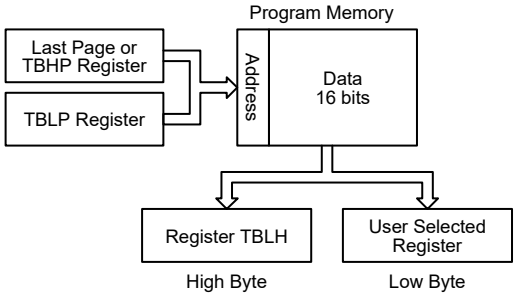


Table Program Example

The following example shows how the table pointer and table data is defined and retrieved from the microcontroller. This example uses raw table data located in the Program Memory which is stored there using the ORG statement. The value at this ORG statement is "1F00H" which refers to the start address of the last page within the 32K words Program Memory of the HT67F370. The table pointer is setup here to have an initial value of "06H". This will ensure that the first data read from the data table will be at the Program Memory address "7F06H" or 6 locations after the start of the specified page. Note that the value for the table pointer is referenced to the specific address pointed by TBHP and TBLP 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

```
rombank 3 code3
ds .section 'data'
tempreg1 db ?      ; temporary register #1 in current page
tempreg2 db ?      ; temporary register #2 in current page
     :
     :
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,7fh          ; initialise high table pointer
mov tbhp,a         ; it is not necessary to set tbhp if executing tabrdl
     :
     :
tabrd tempreg1
tabrdl tempreg1    ; transfers value in table referenced by table pointer data at program
                   ; memory address "7F06H" transferred to tempreg1 and TBLH
dec tblp           ; reduce value of table pointer by one
tabrd tempreg2
tabrdl tempreg2    ; transfers value in table referenced by table pointer data at program
                   ; memory address "7F05H" transferred to tempreg2 and TBLH. In this
                   ; example the data "1AH" is transferred to tempreg1 and data "0FH" to
                   ; register tempreg2 while the value "00H" will be transferred to the
                   ; high byte register TBLH
     :
     :
org 1F00h          ; sets initial address of program memory
dc 000Ah, 000Bh, 000Ch, 000Dh, 000Eh, 000Fh, 001Ah, 001Bh
     :
     :
```

In Circuit Programming – ICP

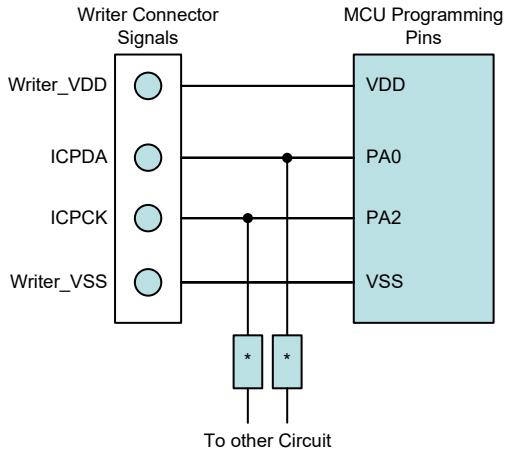
The provision of Flash type Program Memory provides the user with a means of convenient and easy upgrades and modifications to their programs on the same device.

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

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

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

During the programming process, the user must take control 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 HT69V3x0/HT67V370 which is used to emulate the real MCU device HT69F3x0/HT67F370. 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 the "On-Chip Debug" function. Users can use the EV chip device to emulate the real chip device behavior by connecting the OCDSDA and OCDSCK pins to the Holtek HT-IDE development tools. The OCDSDA pin is the OCDS Data/Address input/output pin while the OCDSCK pin is the OCDS clock input pin. When users use the EV chip for debugging, other functions which are shared with the OCDSDA and OCDSCK pins in the read MCU device will have no effect in the EV chip. However, the two OCDS pins which are pin-shared with the ICP programming pins are still used as the Flash Memory programming pins for ICP. For a more detailed OCDS description, refer to the corresponding document named "Holtek e-Link for 8-bit MCU OCDS User's Guide".

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

In Application Programming – IAP

This series devices offer IAP function to update data or application program to Flash ROM. Users can define any ROM location for IAP, but there are some features which users must notice in using IAP function. Note that the HT69F340 device supports the "Block Erase" function instead of the "Page Erase" function.

HT69F340 IAP Configurations		HT69F350 IAP Configurations	
Erase Block	256 words/block	Erase Page	32 words/page
Writing Word	4 words/time	Writing Word	32 words/time
Reading Word	1 word/time	Reading Word	1 word/time
HT69F360/HT67F370 IAP Configurations			
Erase Page	64 words/page		
Writing Word	64 words/time		
Reading Word	1 word/time		

In Application Programming Control Registers

The Address registers, FARL and FARH, and the Data registers, FD0L/FD0H, FD1L/FD1H, FD2L/FD2H, FD3L/FD3H, together with the Control registers, FC0, FC1 and FC2, located in Data Memory all sectors, are the corresponding Flash access registers for IAP. If using the indirect addressing method to access the FC0, FC1 and FC2 registers in other sectors except in sector 0, all read and write operations to these registers must be performed using the Indirect Addressing Register, IAR1 or IAR2, and the Memory Pointer pairs, MP1L/MP1H or MP2L/MP2H . For example, if the control registers FC0, FC1 and FC2 in sector 1 are wanted, as these registers are located at the address of 67H~69H in Data Memory sector 1, the desired value ranged from 67H to 69H must be written into the MP1L or MP2L Memory Pointer low byte and the value “01H” must also be written into the MP1H or MP2H Memory Pointer high byte.

Register Name	Bit							
	7	6	5	4	3	2	1	0
FC0	CFWEN	FMOD2	FMOD1	FMOD0	FWPEN	FWT	FRDEN	FRD
FC1	D7	D6	D5	D4	D3	D2	D1	D0
FC2 (HT69F350/360/ HT67F370)	—	—	—	—	—	—	—	CLWB
FARL	A7	A6	A5	A4	A3	A2	A1	A0
FARH (HT69F340)	—	—	—	—	A11	A10	A9	A8
FARH (HT69F350)	—	—	—	A12	A11	A10	A9	A8
FARH (HT69F360/ HT67F370)	—	—	A13	A12	A11	A10	A9	A8
FD0L	D7	D6	D5	D4	D3	D2	D1	D0
FD0H	D15	D14	D13	D12	D11	D10	D9	D8
FD1L	D7	D6	D5	D4	D3	D2	D1	D0
FD1H	D15	D14	D13	D12	D11	D10	D9	D8
FD2L	D7	D6	D5	D4	D3	D2	D1	D0
FD2H	D15	D14	D13	D12	D11	D10	D9	D8
FD3L	D7	D6	D5	D4	D3	D2	D1	D0
FD3H	D15	D14	D13	D12	D11	D10	D9	D8

IAP Register List

• **FC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	CFWEN	FMOD2	FMOD1	FMOD0	FWPEN	FWT	FRDEN	FRD
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	1	1	1	0	0	0	0

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

Bit 6~4 **FMOD2~FMOD0**: Mode selection
000: Write program memory
001: Block/Page erase program memory
010: Reserved
011: Read program memory
100: Reserved
101: Reserved
110: FWEN mode – Flash memory write function enable mode
111: Reserved
When these bits are set to "001", the "Block erase" mode is selected for the HT69F340, while the "Page erase" mode is selected for the HT69F350/HT69F360/HT67F370.

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

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

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

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

Note: 1. The FWT, FRDEN and FRD bits cannot be set to "1" at the same time with a single instruction.
2. Ensure that the f_{SUB} clock is stable before executing the erase or write operation.
3. Note that the CPU will be stopped when a read, write or erase operation is successfully activated.
4. Ensure that the read, erase or write operation is totally complete before executing other operations.

• **FC1 Register**

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

Bit 7~0 **D7~D0**: Whole chip reset pattern
When users write a specific value of "55H" to this register, it will generate a reset signal to reset whole chip.

• **FC2 Register – HT69F350/HT69F360/HT67F370**

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

Bit 7~1 Unimplemented, read as "0"

Bit 0 **CLWB**: Flash Memory Write buffer clear control
0: Do not initiate Write Buffer Clear or Write Buffer Clear process is completed
1: Initiate Write Buffer Clear process
Before page write action, users must be set CLWB bit to clear Write Buffer.
This bit is set by software and cleared by hardware when the Write Buffer Clear process is completed.

• **FARL Register**

Bit	7	6	5	4	3	2	1	0
Name	A7	A6	A5	A4	A3	A2	A1	A0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 Flash Memory Address [7:0]

• **FARH Register – HT69F340**

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

Bit 7~4 Unimplemented, read as "0"

Bit 3~0 Flash Memory Address [11:8]

• **FARH Register – HT69F350**

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

Bit 7~5 Unimplemented, read as "0"

Bit 4~0 Flash Memory Address [12:8]

• **FARH Register – HT69F360/HT67F370**

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

Bit 7~6 Unimplemented, read as "0"

Bit 5~0 Flash Memory Address [13:8]

• **FD0L Register**

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

Bit 7~0 The first Flash Memory data [7:0]

• **FD0H Register**

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

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

• **FD1L Register**

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

Bit 7~0 The second Flash Memory data [7:0]

• **FD1H Register**

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

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

• **FD2L Register**

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

Bit 7~0 The third Flash Memory data [7:0]

• **FD2H Register**

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

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

• **FD3L Register**

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

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

• **FD3H Register**

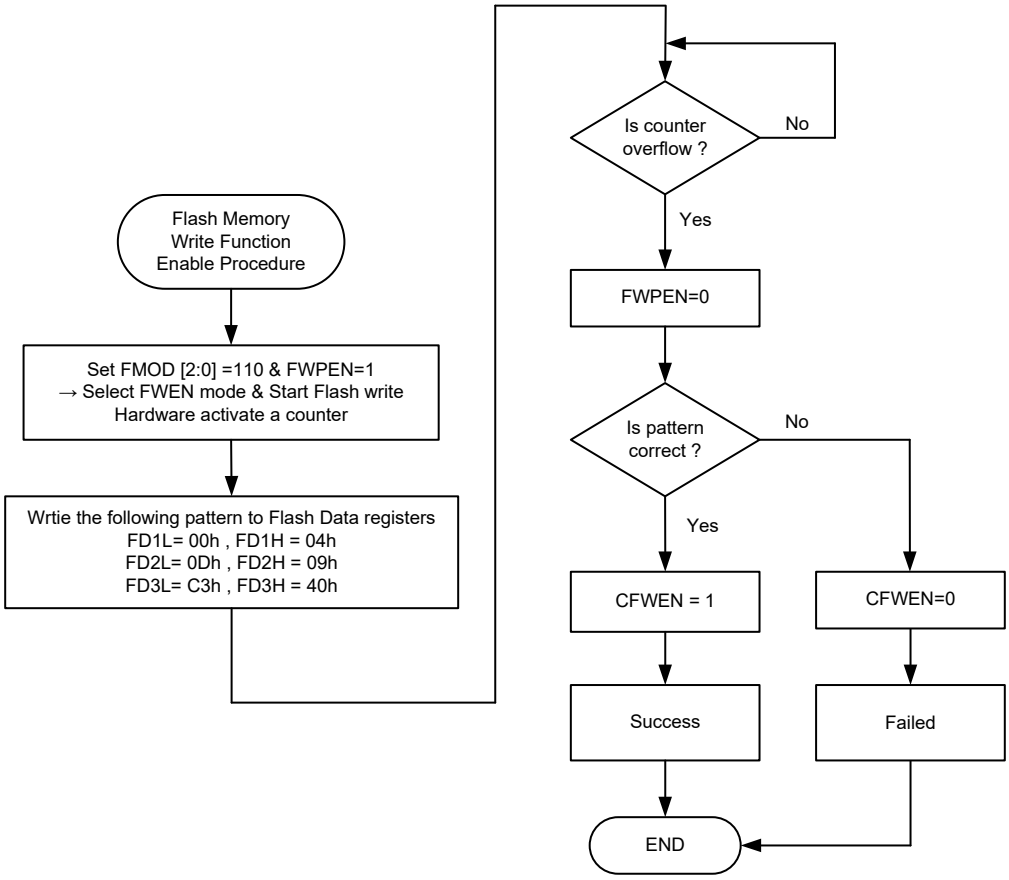
Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

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

Flash Memory Write Function Enable Procedure

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

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



Flash Memory Write Function Enable Procedure

Flash Memory Write Procedure

After the Flash memory write function is successfully enabled through the preceding IAP procedure, users must first erase the corresponding Flash memory block and then initiate the Flash memory write operation. For the HT69F340 device, the number of the block erase operation is 256 words per block, the available block erase address is only specified by FARH register and the content in the FARL register is not used to specify the block address. For the HT69F350 device, since the number of the page erase operation is 32 words per page, the available page erase address is specified by FARH register and the content of bit 7~bit 5 in the FARL register. For the HT69F360/HT67F370 device, since the number of the page erase operation is 64 words per page, the available page erase address is specified by FARH register and the content of bit 7 ~ bit 6 in the FARL register.

Erase Block	HT69F340	
	FARH [3:0]	FARL [7:0]
0	0000	xxxx xxxx
1	0001	xxxx xxxx
2	0010	xxxx xxxx
3	0011	xxxx xxxx
4	0100	xxxx xxxx
5	0101	xxxx xxxx
:	:	:

Erase Block	HT69F340	
	FARH [3:0]	FARL [7:0]
12	1100	xxxx xxxx
13	1101	xxxx xxxx
14	1110	xxxx xxxx
15	1111	xxxx xxxx

"x": don't care

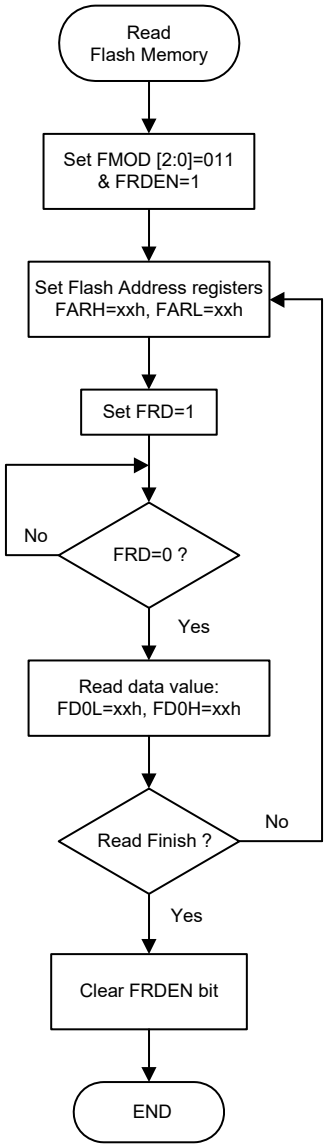
HT69F340 Erase Block Number and Selection

Erase Page	HT69F350		HT69F360/HT67F370	
	FARH	FARL	FARH	FARL
0	0000 0000	000x xxxx	0000 0000	00xx xxxx
1	0000 0000	001x xxxx	0000 0001	01xx xxxx
2	0000 0000	010x xxxx	0000 0010	10xx xxxx
3	0000 0000	011x xxxx	0000 0011	11xx xxxx
4	0000 0000	100x xxxx	0000 0100	00xx xxxx
5	0000 0000	101x xxxx	0000 0101	01xx xxxx
6	0000 0000	110x xxxx	0000 0110	10xx xxxx
7	0000 0000	111x xxxx	0000 0111	11xx xxxx
8	0000 0001	000x xxxx	0000 1000	00xx xxxx
9	0000 0001	001x xxxx	0000 1001	01xx xxxx
:	:	:	:	:
:	:	:	:	:
252	0001 1111	100x xxxx	0011 1111	00xx xxxx
253	0001 1111	101x xxxx	0011 1111	01xx xxxx
254	0001 1111	110x xxxx	0011 1111	10xx xxxx
255	0001 1111	111x xxxx	0011 1111	11xx xxxx

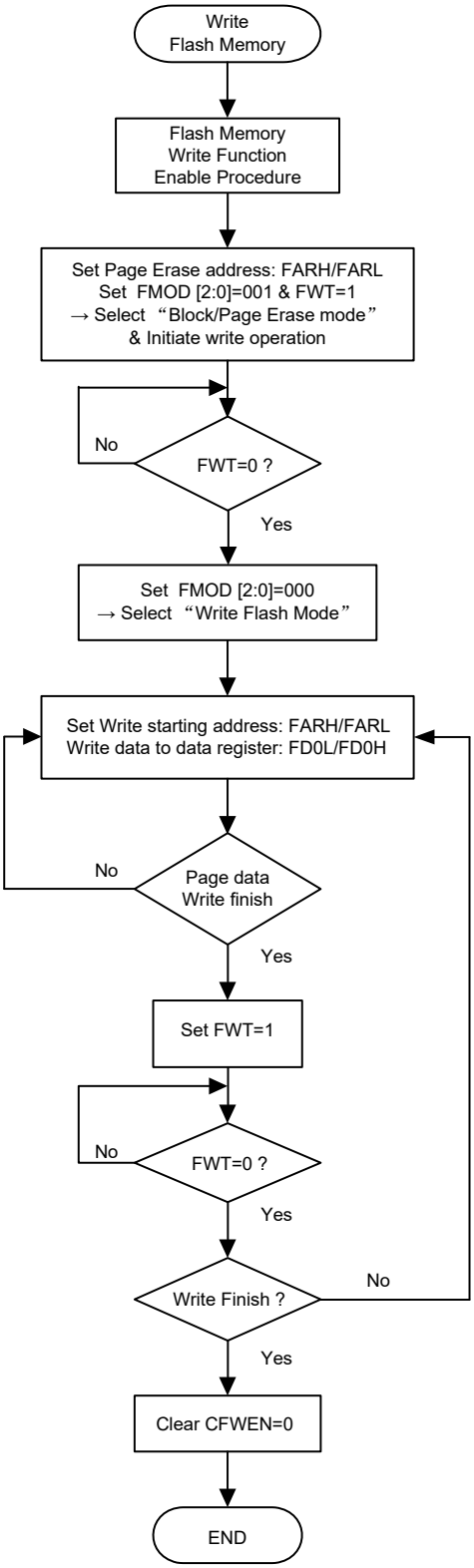
"x": don't care

Note: There are 256 IAP erase pages in the HT69F350/HT69F360/HT67F370 devices.

HT69F350/360/HT67F370 Erase Block Number and Selection



Read Flash Memory Procedure



Write Flash Memory Procedure

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

Data Memory

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

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

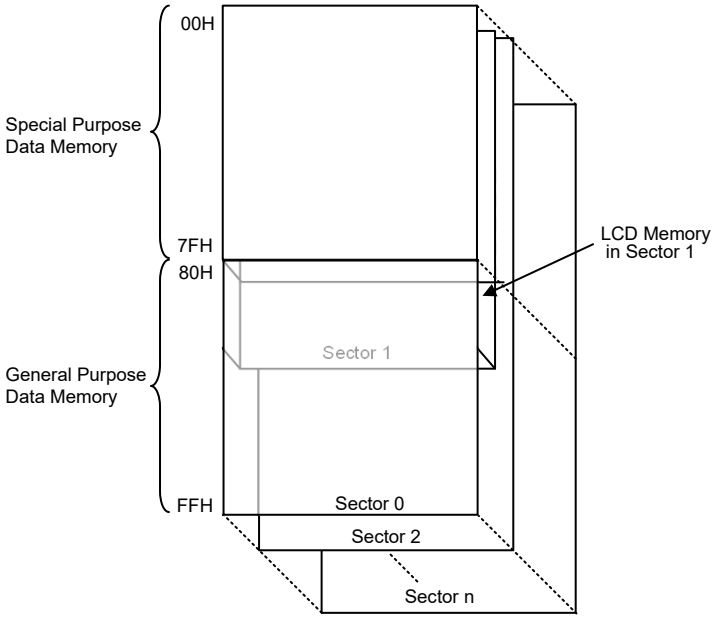
Structure

The Data Memory is subdivided into several sectors, all of which are implemented in 8-bit wide RAM. Each of the Data Memory Sectors is categorized into two types, the special Purpose Data Memory and the General Purpose Data Memory. While the 80H~98H of HT69F340, the 80H~A4H of HT69F350 or 80H~B0H of HT69F360/HT67F370 of Sector 1 is LCD Memory.

The start address of the Data Memory for all devices is the address 00H while the start address of the General Purpose Data Memory is the address 80H. The special purpose registers which are addressed from 00H to 7FH in Data Memory are common to all sectors and are accessible in all sectors except EEC register which can only be addressed in Sector 1.

Device	Special Purpose Data Memory		LCD Display Data Memory		General Purpose Data Memory	
	Capacity	Sectors	Capacity	Sectors	Capacity	Sectors
HT69F340	128×8	0~2: 00H~7FH	25×8	1: 80H~98H	256×8	0: 80H~FFH 2: 80H~FFH
HT69F350	128×8	0~4: 00H~7FH	37×4	1: 80H~A4H	512×8	0: 80H~FFH 2: 80H~FFH 3: 80H~FFH
HT69F360	128×8	0~8: 00H~7FH	49×4	1: 80H~B0H	1024×8	0: 80H~FFH 2: 80H~FFH : 8: 80H~FFH
HT67F370	128×8	0~16: 00H~7FH	49×8	1: 80H~B0H	2048×8	0: 80H~FFH 2: 80H~FFH : 16: 80H~FFH

Data Memory Summary



Note: n=2 for HT69F340, n=4 for HT69F350, n=8 for HT69F360, n=16 for HT67F370

Data Memory Structure

Data Memory Addressing

For these devices that support the extended instructions, there is no Bank Pointer for Data Memory. The Program memory Bank Pointer, PBP, is only available for Program Memory of the HT69F360/HT67F370 device. For Data Memory the desired Sector is pointed by the MP1H or MP2H register and a 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 can be up to 13 valid bits, the high byte indicates a sector and the low byte indicates a specific address.

General Purpose Data Memory

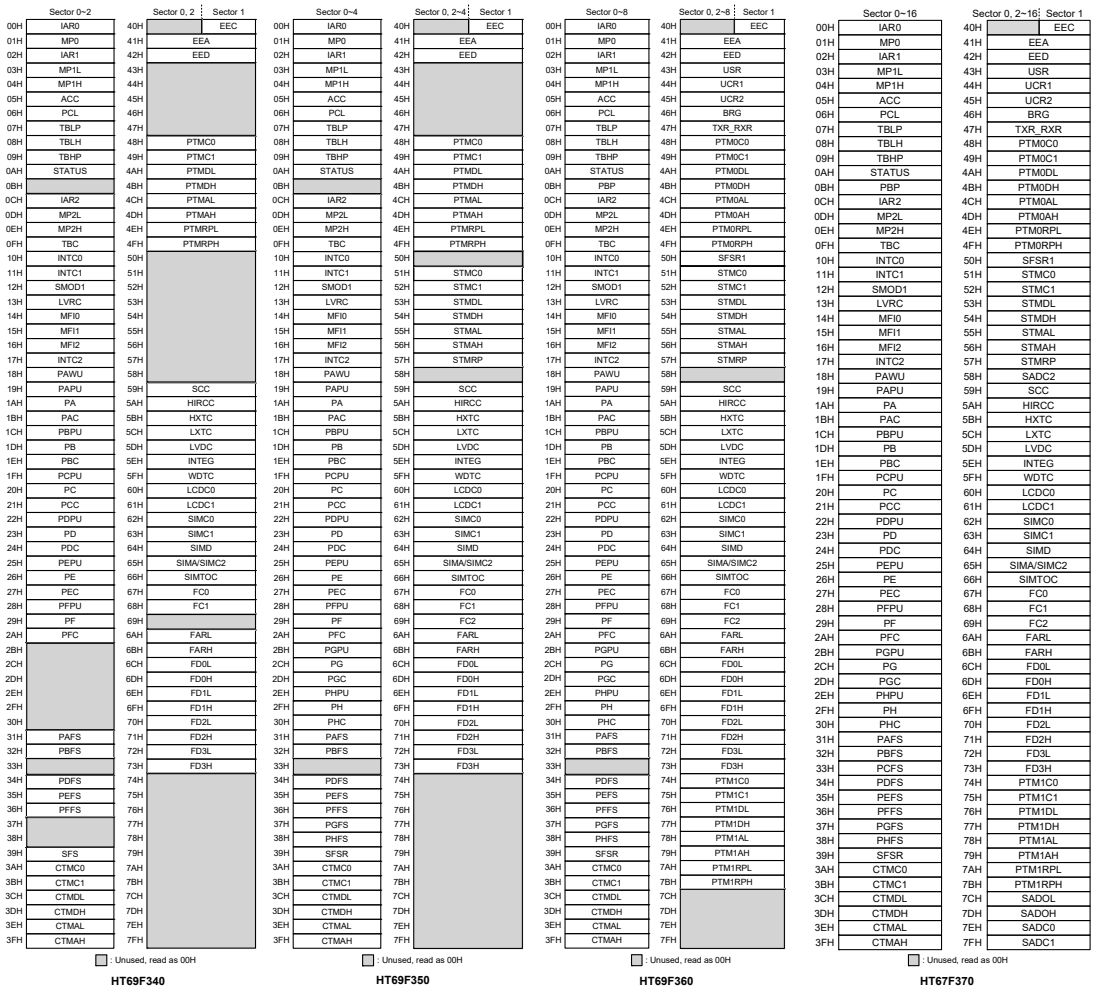
All microcontroller programs require an area of read/write memory where temporary data can be stored and retrieved for use later. It is this area of RAM memory that is known as General Purpose Data Memory. This area of Data Memory is fully accessible by the user programming for both reading and writing 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”.

LCD Display Data Memory

The data to be displayed on the LCD display is stored in an area of fully accessible Data Memory. By writing to this area of RAM, the display output can be directly controlled by the application program. As the LCD Memory addresses overlap those of the General Purpose Data Memory, it is stored in its own independent Sector 1 area for LCD display. This area is located in the 80H~B0H at Sector 1 for HT69F360/HT67F370. To access the Display Memory therefore requires first that Sector 1 is selected by writing a value of 01H to MP1H or MP2H. After this, the memory can then be accessed by using indirect addressing through the use of MP1L or MP2L. With Sector 1 selected, then using MP1L or MP2L to read or write to the memory area, from 80H to B0H, will result in operations to the LCD memory. Directly addressing the Display Memory is not applicable and will result in a data access to the Sector 0 General Purpose Data Memory.



Special Purpose Data Memory

Special Function Register Description

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

Indirect Addressing Register – 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 MP1L/MP1H register pair and IAR2 register together with 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 indirectly will return a result of "00H" and writing to the registers indirectly will result in no operation.

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

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

The following example shows how to clear a sector 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 at 01F0H ´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.

Program Memory Bank Pointer – PBP

For this device the Program Memory is divided into several banks. Selecting the required Program Memory area is achieved using the Program Memory Bank Pointer, PBP. The PBP register should be properly configured before the device executes the “Branch” operation using the “JMP” or “CALL” instruction. After that a jump to a non-consecutive Program Memory address which is located in a certain bank selected by the program memory bank pointer bits will occur.

• **PBP Register – HT67F370**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PBP1	PBP0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~3 Unimplemented, read as "0"

Bit 1~0 **PBP1~PBP0**: Program Memory Bank Pointer bit 1 ~ bit 0
00: Bank 0
01: Bank 1
10: Bank 2
11: Bank 3

• **PBP Register – HT69F360**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	PBP0
R/W	—	—	—	—	—	—	—	R/W
POR	—	—	—	—	—	—	—	0

Bit 7~1 Unimplemented, read as "0"

Bit 0 **PBP0**: Program Memory Bank Pointer bit 0
0: Bank 0
1: Bank 1

Accumulator – ACC

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

Program Counter Low Register – PCL

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

Look-up Table Registers – TBLP, 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.

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

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

• STATUS Register

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

"x": unknown

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

Bit 6 **CZ**: The the operational result of different flags for different 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

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

EEPROM Data Memory Structure

The EEPROM Data Memory capacity is 64×8 bits for the HT69F340 and HT69F350, 128×8 bits for the HT69F360 and 256×8 bits for the HT67F370. 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 data register in all sectors 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 all sectors, they can be directly accessed in the same was as any other Special Function Register. The EEC register however, being located in Sector 1, cannot be directly addressed directly and can only be read from or written to indirectly using the MP1L/MP1H or MP2L/MP2H Memory Pointer and Indirect Addressing Register, IAR1 or IAR2. Because the EEC control register is located at address 40H in Sector 1, the MP1L or MP2L Memory Pointer low byte 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 (HT67F370)	D7	D6	D5	D4	D3	D2	D1	D0
EEA (HT69F340/350)	—	—	D5	D4	D3	D2	D1	D0
EEA (HT69F360)	—	D6	D5	D4	D3	D2	D1	D0
EED	D7	D6	D5	D4	D3	D2	D1	D0
EEC	—	—	—	—	WREN	WR	RDEN	RD

EEPROM Control Register List

• **EEA Register – HT67F370**

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 address bit 7 ~ bit 0

• **EEA Register – HT69F340/HT69F350**

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

Bit 7~6 Unimplemented, read as "0"

Bit 5~0 **D5~D0**: Data EEPROM address
Data EEPROM address bit 5 ~ bit 0

• **EEA Register – HT69F360**

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

Bit 7 Unimplemented, read as "0"

Bit 6~0 **D6~D0**: Data EEPROM address
Data EEPROM address bit 6 ~ 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 can not be set to "1" at the same time in one instruction. The WR and RD can not be set to "1" at the same time.

Reading Data from the EEPROM

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

Writing Data to the EEPROM

The EEPROM address of the data to be written must first be placed in the EEA register and the data placed in the EED register. To write data to the EEPROM, the write enable bit, WREN, in the EEC register must first be set high to enable the write function. After this, the WR bit in the EEC register must be immediately set high to initiate a write cycle. These two instructions must be executed consecutively. The global interrupt bit EMI should also first be cleared before implementing any

write operations, and then set again after the write cycle has started. Setting the WR bit high will not initiate a write cycle if the WREN bit has not been set. As the EEPROM write cycle is controlled using an internal timer whose operation is asynchronous to microcontroller system clock, a certain time will elapse before the data will have been written into the EEPROM. Detecting when the write cycle has finished can be implemented either by polling the WR bit in the EEC register or by using the EEPROM interrupt. When the write cycle terminates, the WR bit will be automatically cleared to zero by the microcontroller, informing the user that the data has been written to the EEPROM. The application program can therefore poll the WR bit to determine when the write cycle has ended.

Write Protection

Protection against inadvertent write operation is provided in several ways. After the device is powered-on the Write Enable bit in the control register will be cleared preventing any write operations. Also at power-on MP1L/MP1H and MP2L/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. However, as the EEPROM is contained within a Multi-function Interrupt, the associated multi-function interrupt enable bit must also be set. When an EEPROM write cycle ends, the DEF request flag and its associated multi-funtion interrupt request flag will both be set. If the global, EEPROM and multi-function interrupts are enabled and the stack is not full, a jump to the associated Interrupt vector will take place. When the interrupt is serviced only the multi-function interrupt flag will be automatically reset, the EEPROM interrupt flag must be manually reset by the application program. More details can be obtained in the Interrupt section.

Programming Considerations

Care must be taken that data is not inadvertently written to the EEPROM. Protection can be Periodic by ensuring that the Write Enable bit is normally cleared to zero when not writing. Also the Memory pointer high byte registers could be normally cleared to zero as this would inhibit access to Sector 1 where the EEPROM control register exist. Although certainly not necessary, consideration might be given in the application program to the checking of the validity of new write data by a simple read back process. When writing data the WR bit must be set high immediately after the WREN bit has been set high, to ensure the write cycle executes correctly. The global interrupt bit EMI should also be cleared before a write cycle is executed and then re-enabled after the write cycle starts. Note that the device 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, 40H                   ; 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, 40H                ; setup memory pointer MP1L
MOV  MP1L, A               ; MP1L points to EEC register
MOV  A, 01H                ; setup memory pointer MP1H
MOV  MP1H, A
CLR  EMI                   ; diable global interrupt
SET  IAR1.3                ; set WREN bit, enable write operations
SET  IAR1.2                ; start Write Cycle - set WR bit - executed immediately
                           ; after setting WREN bit
SET  EMI                   ; enable global interrupt
BACK:
SZ   IAR1.2                ; check for write cycle end
JMP  BACK
CLR  IAR1                  ; disable EEPROM 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 some registers.

Oscillator Overview

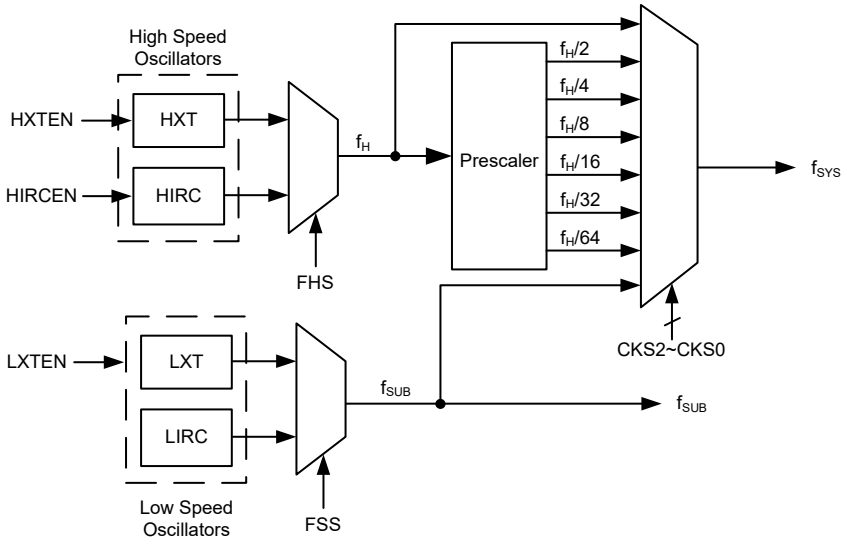
In addition to being the source of the main system clock the oscillators also provide clock sources for the Watchdog Timer and Time Base Interrupts. External oscillators requiring some external components as well as fully integrated internal oscillators, requiring no external components, are provided to form a wide range of both fast and slow system oscillators. 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	Freq.	Pins
External High Speed Crystal	HXT	400kHz~16MHz	OSC1/OSC2
Internal High Speed RC	HIRC	4/8/12MHz	—
External Low Speed Crystal	LXT	32.768kHz	XT1/XT2
Internal Low Speed RC	LIRC	32kHz	—

Oscillator Types

System Clock Configurations

There are four methods of generating the system clock, two high speed oscillators and two low speed oscillators. The high speed oscillators are the external crystal/ceramic oscillator, HXT, and the internal 4/8/12MHz RC oscillator, HIRC. The two low speed oscillators are the internal 32kHz RC oscillator and the external 32.768kHz crystal oscillator. Selecting whether the low or high speed oscillator is used as the system oscillator is implemented using the CKS2~CKS0 bits in the SCC register and as the system clock can be dynamically selected. It is not possible to choose a no-oscillator selection for either the high or low speed oscillator.

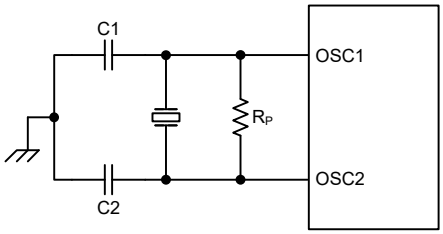


System Clock Configurations

External Crystal/Ceramic Oscillator – HXT

The External Crystal/Ceramic System Oscillator is one of the high frequency oscillator choices, which is selected by the FHS bit and enabled by HXTEN bit. 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**
16MHz	0pF	0pF
12MHz	0pF	0pF
8 MHz	0pF	0pF
4 MHz	0pF	0pF
1 MHz	100pF	100pF
Note: C1 and C2 values are for guidance only.		

Crystal Recommended Capacitor Values

Internal RC Oscillator – HIRC

The internal RC oscillator is a fully integrated system oscillator requiring no external components. The internal RC oscillator is one of the high frequency oscillator choices, which is selected by the FHS bit and is enabled by HIRCEN bit. The HIRC oscillator has three frequencies of 4MHz, 8MHz or 12MHz, selected by HIRC1~HIRC0 bits in HIRCC register. These bits must also be setup to match the selected configuration option frequency to ensure that the HIRC frequency accuracy specified in the A.C. Characteristics is achieved. Device trimming during the manufacturing process and the inclusion of internal frequency compensation circuits are used to ensure that the influence of the power supply voltage, temperature and process variations on the oscillation frequency are minimised. As a result, at a power supply of 5V and at temperature of 25°C degrees, the fixed oscillation frequency of the HIRC will have a tolerance within 2%. Note that if this internal system clock option is selected, as it requires no external pins for its operation.

External 32.768kHz Crystal Oscillator – LXT

The External 32.768kHz Crystal System Oscillator is one of the low frequency oscillator choices, which is selected by FSS bit and enabled by LXTEN bit. 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.

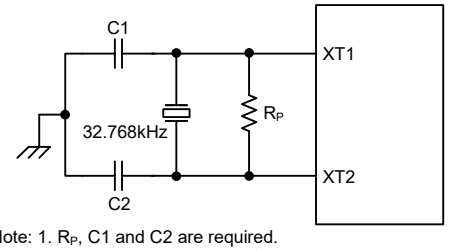
When the microcontroller enters the SLEEP or IDLE Mode, the system clock is switched off to stop microcontroller activity and to conserve power. However, in many microcontroller applications it may be necessary to keep the internal timers operational even when the microcontroller is in the SLEEP or IDLE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

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

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

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

For oscillator stability and to minimise the effects of noise and crosstalk, it is important to ensure 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=5MΩ~10MΩ is recommended.		

32.768kHz Crystal Recommended Capacitor Values

LXT Oscillator Low Power Function

The LXT oscillator can function in one of two modes, the Speed-Up Mode and the Low-Power Mode. The mode selection is executed using the LXTSP bit in the LXTC register

LXTSP Bit	LXT Operating Mode
0	Low Power
1	Speed up

When the LXTSP bit is set to high, the LXT Speed Up Mode will be enabled. In the Speed-Up Mode the LXT oscillator will power up and stabilise quickly. However, after the LXT oscillator has fully powered up, it can be placed into the Low-Power Mode by clearing the LXTSP bit to zero and the oscillator will continue to run but with reduced current consumption. It is important to note that the LXT operating mode switching must be properly controlled before the LXT oscillator clock is selected as the system clock source. Once the LXT oscillator clock is selected as the system clock source using the CKS bit field and FSS bit in the SCC register, the LXT oscillator operating mode can not be changed.

It should be note that no matter what condition the LXTSP is set to, the LXT oscillator will always function normally. The only difference is that it will take more time to start up if in the Low Power Mode.

Internal 32kHz Oscillator – LIRC

The Internal 32kHz System Oscillator is one of the low frequency oscillator choices, which is selected via FSS bit. It is a fully integrated RC oscillator with a typical frequency of 32kHz at 5V, requiring no external components for its implementation. Device trimming during the manufacturing process and the inclusion of internal frequency compensation circuits are used to ensure that the influence of the power supply voltage, temperature and process variations on the oscillation frequency are minimised. As a result, at a power supply of 5V and at a temperature of 25°C degrees, the fixed oscillation frequency of 32kHz will have a tolerance within 10%.

Supplementary Oscillator

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

Operating Modes and System Clocks

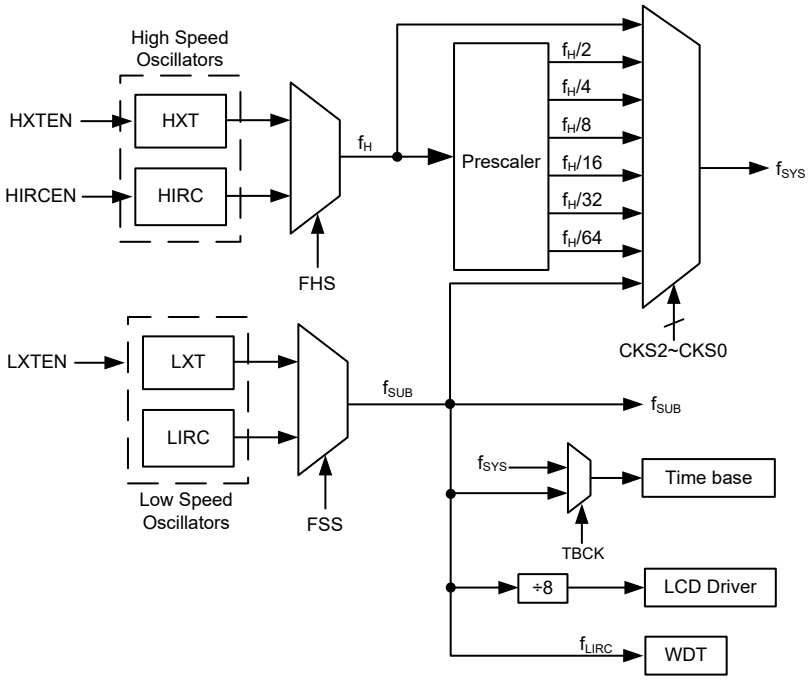
Present day applications require that their microcontrollers have high performance but often still demand that they consume as little power as possible, conflicting requirements that are especially true in battery powered portable applications. The fast clocks required for high performance will by their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

System Clocks

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

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

The HXT, HIRC and LXT oscillators will turn on by setting their related enable bits HXTEN, HIRCEN and LXTEN. The LIRC will turn off when CPU enters the SLEEP Mode and the WDT will turn off.



Device Clock Configuration

Note: When the system clock source f_{SYS} is switched to f_{SUB} from f_H, the high speed oscillator can be stopped to conserve the power. Therefore there is no f_H~$f_H/64$, 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 NORMAL 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	Related Register Setting			f_{SYS}	f_H	f_{SUB}
		FHIDEN	FSIDEN	CKS2~CKS0			
NORMAL	On	x	x	000~110	f_H~$f_H/64$	On	On
SLOW	On	x	x	111	f_{SUB}	On/Off (1)	On
IDLE0	Off	0	1	000~110	Off	Off	On
				111	f_{SUB}		
IDLE1	Off	1	1	xxx	On	On	On
IDLE2	Off	1	0	000~110	On	On	On/Off (2)
				111	f_{SUB} (2)		
SLEEP	Off	0	0	xxx	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_{SUB} clock can be switched on or off which is controlled by the WDT function being enabled or disabled in the IDLE2/SLEEP mode.

NORMAL Mode

As the name suggests this is one of the main operating modes where the microcontroller has all of its functions operational and where the system clock is provided by one of the high speed oscillators. This mode operates allowing the microcontroller to operate normally with a clock source will come from one of the high speed oscillators, HIRC or HXT. 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 f_{SUB}. The clock source is derived from the low speed oscillator LIRC or LXT oscillator. Running the microcontroller in this mode allows it to run with much lower operating currents.

SLEEP Mode

The SLEEP Mode is entered when a HALT instruction is executed and when the FHIDEN and FSIDEN bits are both low. In the SLEEP mode the CPU will be stopped. And the f_{SUB} clock to peripheral will be on or off which is controlled by the WDT function being enabled or disabled in this mode.

IDLE0 Mode

The IDLE0 Mode is entered when a HALT instruction is executed and when the FSIDEN bit in the SCC register is high and the FHIDEN bit in the SCC register is low. 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 a HALT instruction is executed and when the FHIDEN and FSIDEN bits in the SCC register are both high. In the IDLE1 Mode the CPU will be switched off but both the high and low speed oscillator will be turned on to provide a clock source to keep some peripheral functions operational.

IDLE2 Mode

The IDLE2 Mode is entered when a 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. And if the WDT is enabled, the f_{SUB} will also be on to provide a clock for the WDT. If the WDT is disabled, the f_{SUB} will also be off.

Control Register

The register, SCC, is used for overall control of the internal clocks within these devices. The register HXTC is used for HXT oscillator control while the HIRCC register is for the HIRC oscillator control.

• 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**: The 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 off
0: Disable
1: Enable

Bit 0 **FSIDEN**: Low frequency oscillator control when CPU is off
0: Disable
1: Enable

This FSIDEN bit is used to control whether a low speed oscillator is activated or stopped when the CPU is switched off by executing a HALT instruction. When the LIRC oscillator is selected to be the low speed oscillator, the LIRC oscillator will also be controlled by this bit together with the WDT function enable control bit. When this bit is cleared to 0, but the WDT function is enabled, the LIRC oscillator will be enabled.

Note: A certain delay is required before the relevant clock is successfully switched to the target clock source after any clock switching setup using the CKS2~CKS0 bits, FHS bit or FSS bit. A proper delay time must be arranged before executing the following operations which require immediate reaction with the target clock source.

Clock switching delay time = $4\times t_{SYS} + [0 \sim (1.5\times t_{Curr.}+0.5\times t_{Tar.})]$,

where $t_{Curr.}$ indicates the current clock period, $t_{Tar.}$ indicates the target clock period and t_{SYS} indicates the current system clock period.

• HXTC Register

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

Bit 7~3 Unimplemented, read as "0"

Bit 2 **HXTM**: HXT mode selection
0: HXT frequency ≤ 10MHz (sink/source current is smaller)
1: HXT frequency > 10MHz (sink/source current is larger)

Note that this bit should be configured correctly according to the used HXT frequency. If HXTM=0 while the HXT frequency is larger than 10MHz, the oscillation performance at a low voltage condition may be not well. If HXTM=1 while the HXT frequency is less than 10MHz, the oscillator frequency and the current may be obnormal.

This bit must be properly configured before the HXT is enabled. When the OSC1 and OSC2 pin functions have been enabled using relevant pin-shared control bits and the HXTEN bit has been set to 1 to enable the HXT oscillator, it is invalid to change the value of the HXTM bit. When the OSC1 or OSC2 pin function is disabled, then the HXTM bit can be changed by software, regardless of the HXTEN bit value.

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

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

• **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: 4MHz
01: 4MHz
10: 8MHz
11: 12MHz
When the HIRC oscillator is enabled and the HIRC frequency is changed by application program, the clock frequency will automatically be changed after the HIRCF flag is set to 1.
It is recommended that the HIRC frequency selected by these two bits should be the same with the frequency determined by the configuration option to achieve the HIRC frequency accuracy specified in the A.C. Characteristics.

Bit 1 **HIRCF**: HIRC clock stable flag
0: Unstable
1: Stable
This bit is used to indicate whether the HIRC oscillator is stable. When HIRC frequency is exchanged, the CPU will stop 16 HIRC clocks. When bit HIRCEN is set high to enable the HIRC oscillator, this bit will be cleared to"0" and then will be set high after HIRC clock is stable. The HIRC stable time will spend 16 clocks when bit HIRCEN is enabled.

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

• **LXTC Register**

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

Bit 7~3 Unimplemented, read as "0"

Bit 2 **LXTSP**: LXT Speed up control
0: Disable-Low power
1: Enable-Speed up
This bit is used to control whether the LXT oscillator is operating in the low power or Speed-Up mode.When the LXTSP bit is set high, the LXT oscillator will oscillate quickly but consume more power. If the LXTSP bit is cleared to zero, the LXT oscillator will consume less power but take longer time to stablise. It is important to

note that this bit can not be changed after the LXT oscillator is selected as the system clock source using the CKS2~CKS0 and FSS bits in the SCC register.

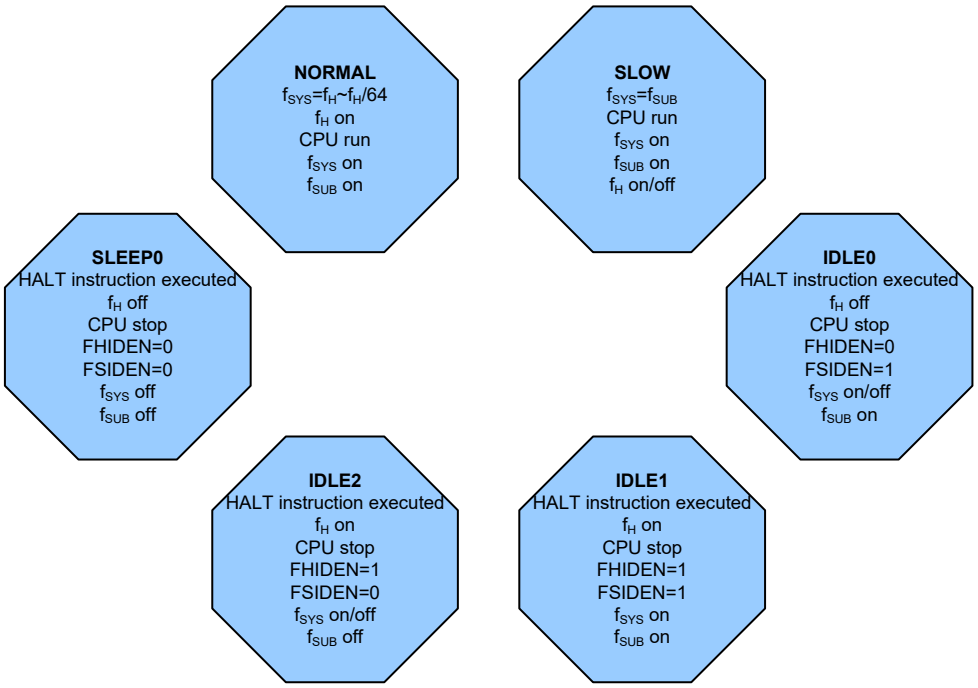
Bit 1 **LXTF**: LXT clock stable flag
0: Unstable
1: Stable
This bit is used to indicate whether the LXT oscillator is stable or not. When the LXTEN bit is set high to enable the LXT oscillator, this LXTF bit will be cleared to"0" and then will be set high after the LXT clock is stable. The LXT stable time will spend 1024 clock when bit LXT is enabled.

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

Operating Mode Switching

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

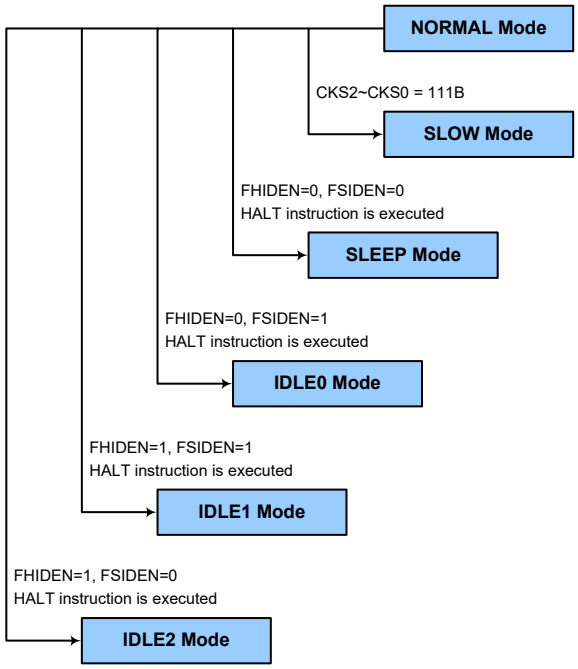
In simple terms, Mode Switching between the NORMAL Mode and SLOW Mode is executed using the CKS2~CKS0 bits in the SCC register while Mode Switching from the NORMAL/SLOW Modes to the IDLE Modes is executed via the HALT instruction. When a HALT instruction is executed, whether the device enters the IDLE0 Mode, IDLE1 Mode, IDLE2 Mode or the SLEEP Mode is determined by the condition of the FHIDEN and FSIDEN bits in the SCC register.



NORMAL Mode to SLOW Mode Switching

When running in the NORMAL Mode, which uses the high speed system oscillator, and therefore consumes more power, the system clock can switch to run in the SLOW Mode by set the 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.

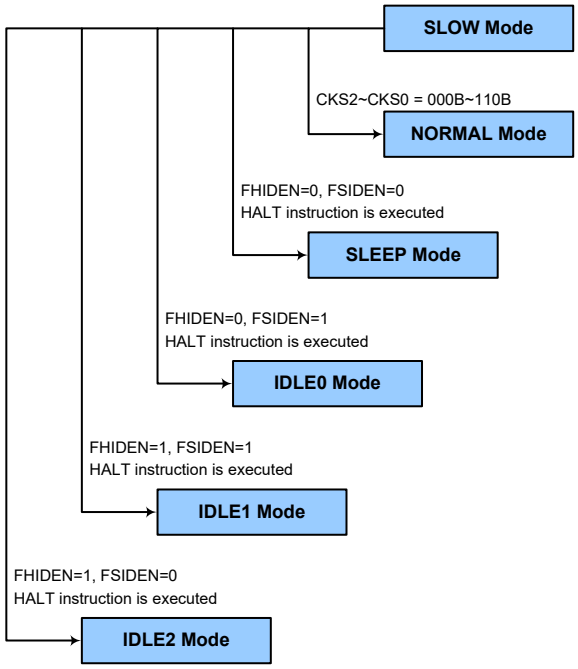
The SLOW Mode is sourced from the LIRC or LXT oscillator and therefore requires the oscillators to be stable before full mode switching occurs.



SLOW Mode to NORMAL Mode Switching

In SLOW mode the system clock is from f_{SUB}. When system clock is switched back to the NORMAL mode from f_{SUB}, the CKS [2:0] bits should be set to "000"~"110", and then the system clock will be switched to f_H~f_H/64.

However, if the f_H clock is not used in the SLOW mode, it will take some time to re-oscillate and stabilize. This is monitored using the HXTF bit in the HXTC register or the HIRCF bit in the HIRCC register. The amount of time required for high speed system oscillator stabilization depends upon which high speed system oscillator type is used.



Entering the SLEEP Mode

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

- The system clock will be stopped and the application program will stop at the "HALT" instruction.
- 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 is disabled then WDT will be cleared and stopped.

Entering the IDLE0 Mode

There is only one way for the device to enter the IDLE0 Mode and that is to execute the "HALT" instruction in the application program with the FSIDEN bit in the SCC register equal to "1" and the FHIDEN bit in 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 off and the f_{SUB} clock will be on and the application program will stop at the "HALT" instruction.
- The Data Memory contents and registers will maintain their present condition.
- The 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 is disabled then WDT will be cleared and stopped.

Entering the IDLE1 Mode

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

- The f_H and the f_{SUB} clock will be on and the application program will stop at the "HALT" instruction.
- The Data Memory contents and registers will maintain their present condition.
- The 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 is disabled then WDT will be cleared and stopped.

Entering the IDLE2 Mode

There is only one way for the device to enter the IDLE2 Mode and that is to execute the "HALT" instruction in the application program with the FHIDEN bit in SCC register equal to "1" and the FSIDEN bit in 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 while 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 is disabled then WDT will be cleared and stopped.

Standby Current Considerations

As the main reason for entering the SLEEP or IDLE Mode is to keep the current consumption of the device to as low a value as possible, perhaps only in the order of several micro-amps except in the IDLE1 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 the device. All high-impedance input pins must be connected to either a fixed high or low level as any floating input pins could create internal oscillations and result in increased current consumption. This also applies to devices which have different package types, as there may be unbounded 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 LIRC or LXT oscillator has enabled.

In the IDLE1 and IDLE2 Mode the high speed oscillator is on, if the system oscillator is 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 the device can enter the SLEEP or IDLE Mode, where the CPU will be switched off. However, when the device is woken up again, it can take a considerable time for the original system oscillator to restart, stabilise and allow normal operation to resume.

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

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

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

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

Watchdog Timer

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

Watchdog Timer Clock Source

The Watchdog Timer clock source is provided by the f_S clock, which is in turn supplied by the f_{SUB} clock. The f_{SUB} clock can be sourced from either the LXT or LIRC oscillator selected by the FSS bit in the SCC register. The LIRC internal oscillator has an approximate frequency of 32kHz and this specified internal clock period can vary with V_{DD}, temperature and process variations. The LXT oscillator is supplied by an external 32.768kHz crystal. The Watchdog Timer source clock is then subdivided by a ratio of 2^8 to 2^{18} to give longer timeouts, the actual value being chosen using the WS2~WS0 bits in the WDTC register.

Watchdog Timer Control Register

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

• **WDTC Register**

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

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

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

• **SMOD1 Register**

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

"x": unknown

Bit 7~3 Unimplemented, read as "0"

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

Bit 1 **LRF**: LVRC register software reset flag
Describe elsewhere

Bit 0 **WRF**: WDTC 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 WDT instructions will not be executed in the correct manner, in which case the Watchdog Timer will overflow and reset the device.

With regard to the Watchdog Timer enable/disable function, there are five bits, WE4~WE0, in the WDTC register to additional enable/disable and reset control of the Watchdog Timer. The WDT

function will be disabled when the WE4~WE0 bits are set to a value of 10101B while the WDT function will be enabled if the WE4~WE0 bits are equal to 01010B. If the WE4~WE0 bits are set to any other values, other than 01010B and 10101B, it will reset the device after 2~3 f_{SUB} clock cycles. After power on these bits will have a value of 01010B.

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

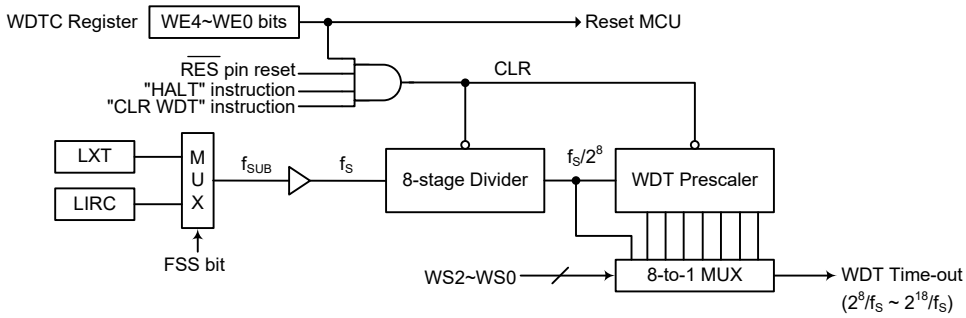
Watchdog Timer Enable/Disable Control

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 and PDF bits 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 value other than 01010B and 10101B written into the WE4~WE0 field, the second is using the Watchdog Timer software clear instructions, the third is via a HALT instruction and the fourth is an external hardware reset, which means a low level on the $\overline{RES}$ 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 seconds for the 2^{18} division ratio, and a minimum timeout of 7.8ms for the 2^8 division ration.



Watchdog Timer

Reset and Initialisation

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

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

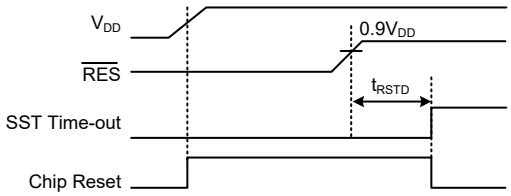
Another type of reset is when the Watchdog Timer overflows and resets the microcontroller. All types of reset operations result in different register conditions being setup. Another reset exists in the form of a Low Voltage Reset, LVR, where a full reset, similar to the $\overline{RES}$ reset is implemented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

Reset Functions

There are 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.



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

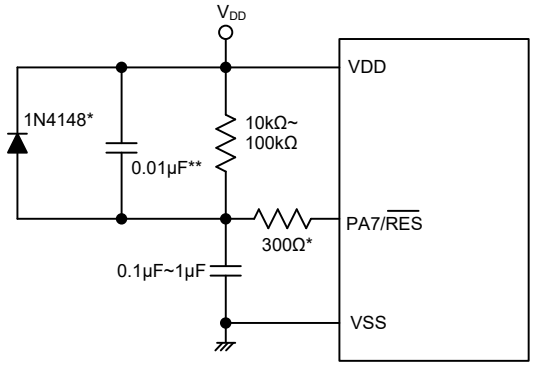
Power-On Reset Timing Chart

$\overline{RES}$ Pin

As the reset pin is shared with an I/O pin, the reset function must be selected using a configuration option. Although the microcontroller has an internal RC reset function, if the V_{DD} power supply rise time is not fast enough or does not stabilise quickly at power-on, the internal reset function may be incapable of providing proper reset operation. For this reason it is recommended that an external RC network is connected to the $\overline{RES}$ pin, whose additional time delay will ensure that the $\overline{RES}$ 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 $\overline{RES}$ line reaches a certain voltage value, the reset delay time t_{RSTD} is invoked to provide an extra delay time after which the will begin normal operation. The abbreviation SST in the figures stands for System Start-up Timer.

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

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

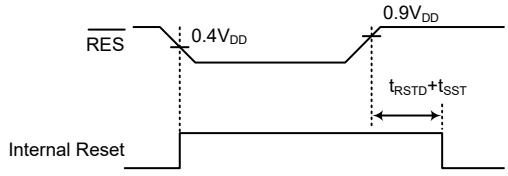


External $\overline{RES}$ Circuit

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

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

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

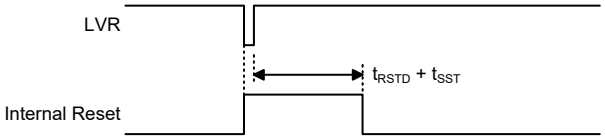


Note: t_{RSTD} is power-on delay, typical time=16.7ms.

$\overline{RES}$ Reset Timing Chart

Low Voltage Reset – LVR

The microcontroller contains a low voltage reset circuit in order to monitor the supply voltage of the device. Writing a corresponding value in LVRC register, the LVR function can be enabled during the normal and slow modes with a specific LVR voltage, V_{LVR}. If the supply voltage of the device drops to within a range of 0.9V~V_{LVR} such as might occur when changing the battery, the LVR will automatically reset the device internally and the LVRF bit in the SMOD1 register will also be set to1. For a valid LVR signal, a low voltage, i.e., a voltage in the range between 0.9V~V_{LVR} must exist for greater than the value t_{LVR} specified in the LVD&LVR Electrical characteristics. If the low voltage state does not exceed this value, the LVR will ignore the low supply voltage and will not perform a reset function. The actual V_{LVR} can be selected by the LVS bits in the LVRC register. If the LVS7~LVS0 bits are changed to some certain values, which may occur due to adverse environmental conditions such as noise, the LVR will reset the device after 2~3 f_{SUB} clock cycles. When this happens, the LRF bit in the SMOD1 register will be set to 1. After power on the register will have the value of 01100110B. Note that the LVR function will be automatically disabled when the device enters the SLEEP/IDLE mode.



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

Low Voltage Reset Timing Chart

- **LVRC Register**

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

Bit 7~0 **LVS7~LVS0**: LVR Voltage Select control
01100110: 1.7V (default)
01010101: 1.9V
00110011: 2.55V
10011001: 3.15V
10101010: 3.8V
11110000: LVR disable
Other values: Generates MCU reset – register is reset to POR value

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

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

- **SMOD1 Register**

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

"x": unknown

Bit 7~3 Unimplemented, read as "0"

Bit 2 **LVRF**: LVR function reset flag
0: Not occurred
1: Occurred

This bit is set to 1 when a specific Low Voltage Reset situation condition occurs. This bit can only be cleared to 0 by the application program.

Bit 1 **LRF**: LVRC register software reset flag
0: Not occurred
1: Occurred

This bit is set to 1 if the LVRC register contains any non defined LVR voltage register values. This in effect acts like a software reset function. This bit can only be cleared to 0 by the application program.

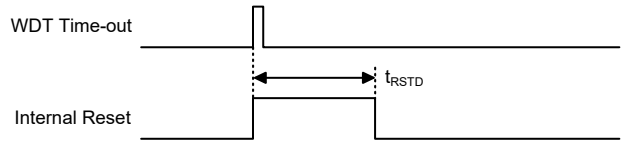
Bit 0 **WRF**: WDTC register software reset flag
Describe elsewhere

IAP Reset

When a specific value of "55H" is written into the FC1 register, a reset signal will be generated to reset the whole device. Refer to the IAP section for more associated details.

Watchdog Time-out Reset during Normal Operation

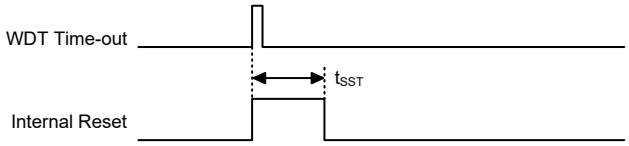
The Watchdog time-out Reset during normal operation is the same as a hardware $\overline{RES}$ pin reset except that the Watchdog time-out flag TO will be set to "1".



WDT Time-out Reset during Normal Operation Timing Chart

Watchdog Time-out Reset during SLEEP or IDLE Mode

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



WDT Time-out Reset during SLEEP or IDLE Timing Chart

Reset Initial Conditions

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

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

Note: "u" stands for unchanged

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

Item	Condition After RESET
Program Counter	Reset to zero
Interrupts	All interrupts will be disabled
WDT, 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 the microcontroller internal registers. Note that where more than one package type exists the table will reflect the situation for the larger package type.

HT69F340

Register	Power-On Reset	$\overline{RES}$ or LVR Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
IAR0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
IAR1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	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	---- xxxx	---- uuuu	---- uuuu	---- uuuu	---- uuuu
STATUS	xx00 xxxx	uuuu uuuu	uu01 uuuu	xx1u uuuu	uu11 uuuu
IAR2	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP2L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
TBC	0011 -111	0011 -111	0011 -111	0011 -111	uuuu -uuu
INTC0	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SMOD1	---- -x00	---- -x00	---- -x00	---- -x00	---- -uuu
LVRC	0110 0110	0110 0110	0110 0110	0110 0110	uuuu uuuu
MFI0	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
MFI1	--00 --00	--00 --00	--00 --00	--00 --00	--uu –-uu
MFI2	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
INTC2	---0 ---0	---0 ---0	---0 ---0	---0 ---0	---u ---u
PAWU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBPU	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu
PB	---- 1111	---- 1111	---- 1111	---- 1111	---- uuuu
PBC	---- 1111	---- 1111	---- 1111	---- 1111	---- uuuu
PCPU	---- -000	---- -000	---- -000	---- -000	---- -uuu
PC	---- -111	---- -111	---- -111	---- -111	---- -uuu
PCC	---- -111	---- -111	---- -111	---- -111	---- -uuu
PDPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PE	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	Power-On Reset	$\overline{RES}$ or LVR Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
PF	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAFS	0000 -000	0000 -000	0000 -000	0000 -000	uuuu -uuu
PBFS	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu
PDFS	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEFS	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFFS	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
SFS	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMC0	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMC1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMDL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMDH	---- --00	---- --00	---- --00	---- --00	---- --uu
CTMAL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMAH	---- --00	---- --00	---- --00	---- --00	---- --uu
EEA	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
EED	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMC0	0000 0---	0000 0---	0000 0---	0000 0---	uuuu u---
PTMC1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDH	---- --00	---- --00	---- --00	---- --00	---- --uu
PTMAL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMAH	---- --00	---- --00	---- --00	---- --00	---- --uu
PTMRPL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMRPH	---- --00	---- --00	---- --00	---- --00	---- --uu
SCC	000- 0000	000- 0000	000- 0000	000- 0000	uuu- uuuu
HIRCC	---- 0001	---- 0001	---- 0001	---- 0001	---- uuuu
HXTC	---- -000	---- -000	---- -000	---- -000	---- -uuu
LXTC	---- -000	---- -000	---- -000	---- -000	---- -uuu
LVDC	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
INTEG	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu
WDTC	0101 0011	0101 0011	0101 0011	0101 0011	uuuu uuuu
LCDC0	0-0- 0000	0-0- 0000	0-0- 0000	0-0- 0000	u-u- uuuu
LCDC1	---- -000	---- -000	---- -000	---- -000	---- -uuu
SIMC0	111- 0000	111- 0000	111- 0000	111- 0000	uuu- uuuu
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
FC0	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARH	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu
FD0L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD0H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD1L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	Power-On Reset	$\overline{RES}$ or LVR Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
FD1H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
EEC	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu

HT69F350

Register	Power-On Reset	$\overline{RES}$ or LVR Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
IAR0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
IAR1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	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	---x xxxx	---u uuuu	---u uuuu	---u uuuu	---u uuuu
STATUS	xx00 xxxx	uuuu uuuu	uu01uuuu	xx1u uuuu	uu11 uuuu
IAR2	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP2L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
TBC	0011 -111	0011 -111	0011 -111	0011 -111	uuuu -uuu
INTC0	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SMOD1	---- -x00	---- -x00	---- -x00	---- -x00	---- -uuu
LVRC	0110 0110	0110 0110	0110 0110	0110 0110	uuuu uuuu
MFI0	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI1	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
MFI2	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
INTC2	---0 ---0	---0 ---0	---0 ---0	---0 ---0	---u ---u
PAWU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCPU	---- -000	---- -000	---- -000	---- -000	---- -uuu
PC	---- -111	---- -111	---- -111	---- -111	---- -uuu
PCC	---- -111	---- -111	---- -111	---- -111	---- -uuu
PDPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PE	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu

Register	Power-On Reset	$\overline{RES}$ or LVR Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
PEC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PF	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PGPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PG	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PGC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PHPU	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu
PH	---- 1111	---- 1111	---- 1111	---- 1111	----- uuuu
PHC	---- 1111	---- 1111	---- 1111	---- 1111	---- uuuu
PAFS	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBFS	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDFS	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEFS	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFFS	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PGFS	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PHFS	---- 1111	---- 1111	---- 1111	---- 1111	---- uuuu
SFSR	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMC0	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMC1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMDL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMDH	---- --00	---- --00	---- --00	---- --00	---- --uu
CTMAL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMAH	---- --00	---- --00	---- --00	---- --00	---- --uu
EEA	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
EED	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMC0	0000 0---	0000 0---	0000 0---	0000 0---	uuuu u---
PTMC1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDH	---- --00	---- --00	---- --00	---- --00	---- --uu
PTMAL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMAH	---- --00	---- --00	---- --00	---- --00	---- --uu
PTMRPL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMRPH	---- --00	---- --00	---- --00	---- --00	---- --uu
STMC0	0000 0---	0000 0---	0000 0---	0000 0---	uuuu u---
STMC1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDH	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAH	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMRP	0000 0000	0000 0000	0000 0000	0000 0000	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	---- -000	---- -000	---- -000	---- -000	---- -uuu
LVDC	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
INTEG	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu

Register	Power-On Reset	$\overline{RES}$ or LVR Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
WDTC	0101 0011	0101 0011	0101 0011	0101 0011	uuuu uuuu
LCDC0	0-0- 0000	0-0- 0000	0-0- 0000	0-0- 0000	u-u- uuuu
LCDC1	---- -000	---- -000	---- -000	---- -000	---- -uuu
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
FC0	0111 0000	0111 0000	0111 0000	0111 0000	uuuu uuuu
FC1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC2	---- ---0	---- ---0	---- ---0	---- ---0	---- ---u
FARL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARH	---0 0000	---0 0000	---0 0000	---0 0000	---u uuuu
FD0L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD0H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD1L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD1H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
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

HT69F360

Register	Power-On Reset	$\overline{RES}$ or LVR Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
IAR0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
IAR1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	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	--xx xxxx	--uu uuuu	--uu uuuu	--uu uuuu	--uu uuuu
STATUS	xx00 xxxx	uuuu uuuu	uu01uuuu	xx1u uuuu	uu11 uuuu
PBP	---- ---0	---- ---0	---- ---0	---- ---0	---- ---u
IAR2	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu

Register	Power-On Reset	$\overline{RES}$ or LVR Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
MP2L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
TBC	0011 -111	0011 -111	0011 -111	0011 -111	uuuu -uuu
INTC0	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SMOD1	---- -x00	---- -x00	---- -x00	---- -x00	---- -uuu
LVRC	0110 0110	0110 0110	0110 0110	0110 0110	uuuu uuuu
MFI0	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI2	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
INTC2	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
PAWU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCPU	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
PC	-111 1111	-111 1111	-111 1111	-111 1111	-uuu uuuu
PCC	-111 1111	-111 1111	-111 1111	-111 1111	-uuu uuuu
PDPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PE	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PF	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PGPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PG	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PGC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PHPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PH	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PHC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAFS	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBFS	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDFS	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEFS	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFFS	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PGFS	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PHFS	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
SFSR	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMC0	0000 0000	0000 0000	0000 0000	0000 0000	Uuuu uuuu
CTMC1	0000 0000	0000 0000	0000 0000	0000 0000	Uuuu uuuu
CTMDL	0000 0000	0000 0000	0000 0000	0000 0000	Uuuu uuuu

Register	Power-On Reset	$\overline{RES}$ or LVR Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
CTMDH	---- --00	---- --00	---- --00	---- --00	---- --uu
CTMAL	0000 0000	0000 0000	0000 0000	0000 0000	Uuuu uuuu
CTMAH	---- --00	---- --00	---- --00	---- --00	---- --uu
EEA	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
EED	0000 0000	0000 0000	0000 0000	0000 0000	uuuu 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
BRG	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TXR_RXR	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu 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
SFSR1	---- -000	---- -000	---- -000	---- -000	---- -uuu
STMC0	0000 0---	0000 0---	0000 0---	0000 0---	uuuu u---
STMC1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDH	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAH	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMRP	0000 0000	0000 0000	0000 0000	0000 0000	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	---- -000	---- -000	---- -000	---- -000	---- -uuu
LVDC	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
INTEG	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu
WDTC	0101 0011	0101 0011	0101 0011	0101 0011	uuuu uuuu
LCDC0	0-0- 0000	0-0- 0000	0-0- 0000	0-0- 0000	u-u- uuuu
LCDC1	---- -000	---- -000	---- -000	---- -000	---- -uuu
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
FC0	0111 0000	0111 0000	0111 0000	0111 0000	uuuu uuuu
FC1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC2	---- ---0	---- ---0	---- ---0	---- ---0	---- ---u
FARL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARH	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
FD0L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	Power-On Reset	$\overline{RES}$ or LVR Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
FD0H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD1L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD1H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3L	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3H	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
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

HT67F370

Register	Power On Reset	$\overline{RES}$ Reset (Normal Operation)	WDT Time-out (Normal Operation)	WDT Time-out (IDLE/SLEEP)
IAR0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
IAR1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1L	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP1H	0000 0000	0000 0000	0000 0000	uuuu uuuu
ACC	xxxx xxxx	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	xxxx xxxx	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	xxxx xxxx	uuuu uuuu	uuuu uuuu
TBHP	-xxx xxxx	-uuu uuuu	-uuu uuuu	-uuu uuuu
STATUS	xx00 xxxx	uuuu uuuu	xx1u uuuu	uu11 uuuu
PBP	---- --00	---- --00	---- --00	---- --uu
IAR2	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP2L	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2H	0000 0000	0000 0000	0000 0000	uuuu uuuu
TBC	0011 -111	0011 -111	0011 -111	uuuu -uuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
SMOD1	---- -x00	---- -x00	---- -x00	---- -uuu
LVRC	0110 0110	0110 0110	0110 0110	uuuu uuuu
MFI0	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI1	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI2	--00 --00	--00 --00	-00 --00	--uu --uu
INTC2	-000 -000	-000 -000	-000 -000	-uuu -uuu
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu

Register	Power On Reset	$\overline{\text{RES}}$ Reset (Normal Operation)	WDT Time-out (Normal Operation)	WDT Time-out (IDLE/SLEEP)
PBPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCPU	-000 0000	-000 0000	-000 0000	-uuu uuuu
PC	-111 1111	-111 1111	-111 1111	-uuu uuuu
PCC	-111 1111	-111 1111	-111 1111	-uuu uuuu
PDPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PE	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PF	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PGPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PG	1111 1111	1111 1111	1111 1111	uuuu uuuu
PGC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PHPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PH	1111 1111	1111 1111	1111 1111	uuuu uuuu
PHC	1111 1111	1111 1111	1111 1111	1111 1111
PAFS	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBFS	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCFS	-000 0---	-000 0---	-000 0---	-uuu u---
PDFS	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEFS	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFFS	1111 1111	1111 1111	1111 1111	uuuu uuuu
PGFS	1111 1111	1111 1111	1111 1111	uuuu uuuu
PHFS	1111 1111	1111 1111	1111 1111	uuuu uuuu
SFSR	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMC0	0000 0000	0000 0000	0000 0000	Uuuu uuuu
CTMC1	0000 0000	0000 0000	0000 0000	Uuuu uuuu
CTMDL	0000 0000	0000 0000	0000 0000	Uuuu uuuu
CTMDH	---- --00	---- --00	---- --00	---- --uu
CTMAL	0000 0000	0000 0000	0000 0000	Uuuu uuuu
CTMAH	---- --00	---- --00	---- --00	---- --uu
EEA	0000 0000	0000 0000	0000 0000	uuuu uuuu
EED	0000 0000	0000 0000	0000 0000	uuuu uuuu
USR	0000 1011	0000 1011	0000 1011	uuuu uuuu
UCR1	0000 00x0	0000 00x0	0000 00x0	uuuu uuuu
UCR2	0000 0000	0000 0000	0000 0000	uuuu uuuu
BRG	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TXR_RXR	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PTM0C0	0000 0---	0000 0---	0000 0---	uuuu u---
PTM0C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0DH	---- --00	---- --00	---- --00	---- --uu
PTM0AL	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	Power On Reset	RES Reset (Normal Operation)	WDT Time-out (Normal Operation)	WDT Time-out (IDLE/SLEEP)
PTM0AH	---- --00	---- --00	---- --00	---- --uu
PTM0RPL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0RPH	---- --00	---- --00	---- --00	---- --uu
SFSR1	--00 0000	--00 0000	--00 0000	--uu uuuu
STMC0	0000 0---	0000 0---	0000 0---	uuuu u---
STMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDH	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAH	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMRP	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADC2	0--0 0000	0--0 0000	0--0 0000	u--u uuuu
SCC	000- 0000	000- 0000	000- 0000	uuu- uuuu
HIRCC	---- 0001	---- 0001	---- 0001	---- uuuu
HXTC	---- -000	---- -000	---- -000	---- -uuu
LXTC	---- -000	---- -000	---- -000	---- -uuu
LVDC	--00 0000	--00 0000	--00 0000	--uu uuuu
INTEG	---- 0000	---- 0000	---- 0000	---- uuuu
WDTC	0101 0011	0101 0011	0101 0011	Uuuu uuuu
LCDC0	0-0- 0000	0-0- 0000	0-0- 0000	u-u- uuuu
LCDC1	---- -000	---- -000	---- -000	---- -uuu
SIMC0	111- 0000	111- 0000	111- 0000	uuu- uuuu
SIMC1	1000 0001	1000 0001	1000 0001	uuuu uuuu
SIMD	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SIMA/SIMC2	0000 0000	0000 0000	0000 0000	uuuu uuuu
SIMTOC	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC0	0111 0000	0111 0000	0111 0000	uuuu uuuu
FC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC2	---- ---0	---- ---0	---- ---0	---- ---u
FARL	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARH	--00 0000	--00 0000	--00 0000	--uu uuuu
FD0L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD0H	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD1L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD1H	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2H	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3L	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3H	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1C0	0000 0---	0000 0---	0000 0---	uuuu u---
PTM1C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1DH	---- --00	---- --00	---- --00	---- --uu
PTM1AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1AH	---- --00	---- --00	---- --00	---- --uu
PTM1RPL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1RPH	---- --00	---- --00	---- --00	---- --uu

Register	Power On Reset	RES Reset (Normal Operation)	WDT Time-out (Normal Operation)	WDT Time-out (IDLE/SLEEP)
SADOL	xxxx ----	xxxx ----	xxxx ----	uuuu ---- (ADRFS=0)
				uuuu uuuu (ADRFS=1)
SADOH	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu (ADRFS=0)
				---- uuuu (ADRFS=1)
SADC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADC1	0000 -000	0000 -000	0000 -000	uuuu -uuu
EEC	---- 0000	---- 0000	---- 0000	---- uuuu

Note: "-" not implement
"u" stands for "unchanged"
"x" stands for "unknown"

Input/Output Ports

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

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

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PBPU	—	—	—	—	PBPU3	PBPU2	PBPU1	PBPU0
PB	—	—	—	—	PB3	PB2	PB1	PB0
PBC	—	—	—	—	PBC3	PBC2	PBC1	PBC0
PCPU	—	—	—	—	—	PCPU2	PCPU1	PCPU0
PC	—	—	—	—	—	PC2	PC1	PC0
PCC	—	—	—	—	—	PCC2	PCC1	PCC0
PDPU	PDPU7	PDPU6	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
PD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
PDC	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
PEPU	PEPU7	PEPU6	PEPU5	PEPU4	PEPU3	PEPU2	PEPU1	PEPU0
PE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
PEC	PEC7	PEC6	PEC5	PEC4	PEC3	PEC2	PEC1	PEC0
PFPU	PFPU7	PFPU6	PFPU5	PFPU4	PFPU3	PFPU2	PFPU1	PFPU0
PF	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
PFC	PFC7	PFC6	PFC5	PFC4	PFC3	PFC2	PFC1	PFC0

I/O Logic Function Register List – HT69F340

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PBPU	PBPU7	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PB	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PBC	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PCPU	—	—	—	—	—	PCPU2	PCPU1	PCPU0
PC	—	—	—	—	—	PC2	PC1	PC0
PCC	—	—	—	—	—	PCC2	PCC1	PCC0
PDPU	PDPU7	PDPU6	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
PD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
PDC	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
PEPU	PEPU7	PEPU6	PEPU5	PEPU4	PEPU3	PEPU2	PEPU1	PEPU0
PE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
PEC	PEC7	PEC6	PEC5	PEC4	PEC3	PEC2	PEC1	PEC0
PFPU	PFPU7	PFPU6	PFPU5	PFPU4	PFPU3	PFPU2	PFPU1	PFPU0
PF	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
PFC	PFC7	PFC6	PFC5	PFC4	PFC3	PFC2	PFC1	PFC0
PGPU	PGPU7	PGPU6	PGPU5	PGPU4	PGPU3	PGPU2	PGPU1	PGPU0
PG	PG7	PG6	PG5	PG4	PG3	PG2	PG1	PG0
PGC	PGC7	PGC6	PGC5	PGC4	PGC3	PGC2	PGC1	PGC0
PHPU	—	—	—	—	PHPU3	PHPU2	PHPU1	PHPU0
PH	—	—	—	—	PH3	PH2	PH1	PH0
PHC	—	—	—	—	PHC3	PHC2	PHC1	PHC0

I/O Logic Function Register List – HT69F350

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PBPU	PBPU7	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PB	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PBC	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PCPU	—	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PC	—	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	—	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PDPU	PDPU7	PDPU6	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
PD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
PDC	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
PEPU	PEPU7	PEPU6	PEPU5	PEPU4	PEPU3	PEPU2	PEPU1	PEPU0
PE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
PEC	PEC7	PEC6	PEC5	PEC4	PEC3	PEC2	PEC1	PEC0
PFPU	PFPU7	PFPU6	PFPU5	PFPU4	PFPU3	PFPU2	PFPU1	PFPU0

Register Name	Bit							
	7	6	5	4	3	2	1	0
PF	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
PFC	PFC7	PFC6	PFC5	PFC4	PFC3	PFC2	PFC1	PFC0
PGPU	PGPU7	PGPU6	PGPU5	PGPU4	PGPU3	PGPU2	PGPU1	PGPU0
PG	PG7	PG6	PG5	PG4	PG3	PG2	PG1	PG0
PGC	PGC7	PGC6	PGC5	PGC4	PGC3	PGC2	PGC1	PGC0
PHPU	PHPU7	PHPU6	PHPU5	PHPU4	PHPU3	PHPU2	PHPU1	PHPU0
PH	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0
PHC	PHC7	PHC6	PHC5	PHC4	PHC3	PHC2	PHC1	PHC0

I/O Logic Function Register List – HT69F360/HT67F370

"—" Unimplemented, read as "0"

PAWUn: Port A pin wake-up function control
0: Disable
1: Enable

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

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

PAPUn/PBPUn/PCPUn/PDPUn/PEPUn/PFPUn/PGPUn/PHPUn: I/O pin pull-high function control
0: Disable
1: Enable

Pull-high Resistors

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

The series of devices provide two resistor selections, which can be 60kΩ in normal voltage mode or 15kΩ in low voltage mode. The resistor selection is executed using the LVPU bit in the LVDC register.

LVPU Bit	Pull-high Resistor
0	60kΩ (Normal Mode)
1	15kΩ (Low voltage Mode)

Port A Wake-up

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

• **PAWU Register**

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

Bit 7~0 I/O Port A bit 7 ~ bit 0 Wake Up Control
0: Disable
1: Enable

I/O Port Control Registers

Each I/O port has its own control register known as PAC~PHC, to control the input/output configuration. With this control register, each I/O pin with or without pull-high resistors 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.

Pin-shared Functions

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

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAFS	PAFS7	PAFS6	PAFS5	PAFS4	PAFS3	PAFS2	PAFS1	PAFS0
PBFS	PBFS7	PBFS6	PBFS5	PBFS4	PBFS3	PBFS2	PBFS1	PBFS0
PCFS	—	PCFS6	PCFS5	PCFS4	PCFS3	—	—	—
PDFS	PDFS7	PDFS6	PDFS5	PDFS4	PDFS3	PDFS2	PDFS1	PDFS0
PEFS	PEFS7	PEFS6	PEFS5	PEFS4	PEFS3	PEFS2	PEFS1	PEFS0
PFFS	PFFS7	PFFS6	PFFS5	PFFS4	PFFS3	PFFS2	PFFS1	PFFS0
PGFS	PGFS7	PGFS6	PGFS5	PGFS4	PGFS3	PGFS2	PGFS1	PGFS0
PHFS	PHFS7	PHFS6	PHFS5	PHFS4	PHFS3	PHFS2	PHFS1	PHFS0
SFSR	SFS7	SFS6	SFS5	SFS4	SFS3	SFS2	SFS1	SFS0
SFSR1	—	—	PA4FS	PB4FS	PB5FS	PB7FS	PB6FS	SFS8

Pin-shared Function Selection Register List – HT67F370

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAFS	PAFS7	PAFS6	PAFS5	PAFS4	—	PAFS2	PAFS1	PAFS0
PBFS	—	—	—	—	PBFS3	PBFS2	PBFS1	PBFS0
PDFS	PDFS7	PDFS6	PDFS5	PDFS4	PDFS3	PDFS2	PDFS1	PDFS0
PEFS	PEFS7	PEFS6	PEFS5	PEFS4	PEFS3	PEFS2	PEFS1	PEFS0
PFFS	PFFS7	PFFS6	PFFS5	PFFS4	PFFS3	PFFS2	PFFS1	PFFS0
SFS	SFS7	SFS6	SFS5	SFS4	SFS3	SFS2	SFS1	SFS0

Pin-shared Function Selection Register List – HT69F340

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAFS	PAFS7	PAFS6	PAFS5	PAFS4	PAFS3	PAFS2	PAFS1	PAFS0
PBFS	PBFS7	PBFS6	PBFS5	PBFS4	PBFS3	PBFS2	PBFS1	PBFS0
PDFS	PDFS7	PDFS6	PDFS5	PDFS4	PDFS3	PDFS2	PDFS1	PDFS0
PEFS	PEFS7	PEFS6	PEFS5	PEFS4	PEFS3	PEFS2	PEFS1	PEFS0
PFFS	PFFS7	PFFS6	PFFS5	PFFS4	PFFS3	PFFS2	PFFS1	PFFS0
PGFS	PGFS7	PGFS6	PGFS5	PGFS4	PGFS3	PGFS2	PGFS1	PGFS0
PHFS	—	—	—	—	PHFS3	PHFS2	PHFS1	PHFS0
SFSR	SFS7	SFS6	SFS5	SFS4	SFS3	SFS2	SFS1	SFS0

Pin-shared Function Selection Register List – HT69F350

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAFS	PAFS7	PAFS6	PAFS5	PAFS4	PAFS3	PAFS2	PAFS1	PAFS0
PBFS	PBFS7	PBFS6	PBFS5	PBFS4	PBFS3	PBFS2	PBFS1	PBFS0
PDFS	PDFS7	PDFS6	PDFS5	PDFS4	PDFS3	PDFS2	PDFS1	PDFS0
PEFS	PEFS7	PEFS6	PEFS5	PEFS4	PEFS3	PEFS2	PEFS1	PEFS0
PFFS	PFFS7	PFFS6	PFFS5	PFFS4	PFFS3	PFFS2	PFFS1	PFFS0
PGFS	PGFS7	PGFS6	PGFS5	PGFS4	PGFS3	PGFS2	PGFS1	PGFS0
PHFS	PHFS7	PHFS6	PHFS5	PHFS4	PHFS3	PHFS2	PHFS1	PHFS0
SFSR	SFS7	SFS6	SFS5	SFS4	SFS3	SFS2	SFS1	SFS0
SFSR1	—	—	—	—	—	PB7FS	PB6FS	SFS8

Pin-shared Function Selection Register List – HT69F360

- **PAFS Register – HT67F370**

Bit	7	6	5	4	3	2	1	0
Name	PAFS7	PAFS6	PAFS5	PAFS4	PAFS3	PAFS2	PAFS1	PAFS0
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 **PAFS7~PAFS6**: PA6 Pin-Shared function selection
00: PA6
01: AN7
10: CTP
11: $\overline{\text{SCS}}$

Bit 5~4 **PAFS5~PAFS4**: PA5 Pin-Shared function selection
00: PA5
01: AN6
10: STP
11: SCK/SCL

Bit 3~2 **PAFS3~PAFS2**: PA3 Pin-Shared function selection
00: PA3
01: AN4
10: STP
11: SDI/SDA

Bit 1~0 **PAFS1~PAFS0**: PA1 Pin-Shared function selection
00: PA1
01: AN3
10: CTP
11: SDO

• PAFS Register − HT69F340

Bit	7	6	5	4	3	2	1	0
Name	PAFS7	PAFS6	PAFS5	PAFS4	—	PAFS2	PAFS1	PAFS0
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 7~6 **PAFS7~PAFS6**: PA6 Pin-shared function selection
00: I/O
01: I/O
10: CTP_0
11: $\overline{SCS}$

Bit 5~4 **PAFS5~PAFS4**: PA5 Pin-shared function selection
00: I/O
01: I/O
10: PTP_2
11: SCK/SCL

Bit 3 Unimplemented, read as “0”

Bit 2 **PAFS2**: PA3 Pin-shared function selection
0: I/O
1: SDI/SDA

Bit 1~0 **PAFS1~PAFS0**: PA1 Pin-shared function selection
00: I/O
01: I/O
10: CTP_1
11: SDO

• **PAFS Register − HT69F350/HT69F360**

Bit	7	6	5	4	3	2	1	0
Name	PAFS7	PAFS6	PAFS5	PAFS4	PAFS3	PAFS2	PAFS1	PAFS0
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 **PAFS7~PAFS6**: PA6 Pin-shared function selection
00: I/O
01: I/O
10:CTP_0
11: SCS

Bit 5~4 **PAFS5~PAFS4**: PA5 Pin-shared function selection
00: I/O
01: I/O
10: STP_1
11: SCK/SCL

Bit 3~2 **PAFS3~PAFS2**: PA3 Pin-shared function selection
00: I/O
01: I/O
10: STP_0
11: SDI/SDA

Bit 1~0 **PAFS1~PAFS0**: PA1 Pin-shared function selection
00: I/O
01: I/O
10: CTP_1
11: SDO

• **PBFS Register – HT67F370**

Bit	7	6	5	4	3	2	1	0
Name	PBFS7	PBFS6	PBFS5	PBFS4	PBFS3	PBFS2	PBFS1	PBFS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **PBFS7**: PB7 Pin-Shared function selection, together with the PB7FS bit in the SFSR1 register [PBFS7:PB7FS]=
00: PB7
01: AN2
10: PTP1B
11: TX

Bit 6 **PBFS6**: PB6 Pin-Shared function selection, together with the PB6FS bit in the SFSR1 register [PBFS6:PB6FS]=
00: PB6
01: AN1
10: PTP1
11: RX

Bit 5 **PBFS5**: PB5 Pin-Shared function selection, together with the PB5FS bit in the SFSR1 register [PB5FS:PBFS5]=
00: PB5
01: PTP0B
10: VREF
11: AN0

Bit 4 **PBFS4**: PB4 Pin-Shared function selection, together with the PB4FS bit in the SFSR1 register [PB4FS:PBFS4]=
00: PB4
01: PTP0
10: VREFI
11: PB4

Bit 3~2 **PBFS3~PBFS2**: PB3 or PB2 Pin-Shared function selection
00: PB3 or PB2
01: PB3 or PB2
10: PB3 or PB2
11: XT2 or XT1

Bit 1~0 **PBFS1~PBFS0**: PB1 or PB0 Pin-Shared function selection
00: PB1 or PB0
01: PB1 or PB0
10: PB1 or PB0
11: OSC2 or OSC1

• PBFS Register − HT69F340

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	PBFS3	PBFS2	PBFS1	PBFS0
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 **PBFS3~PBFS2**: PB3/PB2 pin-shared function selection
00: I/O
01: I/O
10: I/O
11: XT2/XT1

Bit 1~0 **PBFS1~PBFS0**: PB1/PB0 pin-shared function selection
00: I/O
01: I/O
10: I/O
11: OSC2/OSC1

• PBFS Register − HT69F350/HT69F360

Bit	7	6	5	4	3	2	1	0
Name	PBFS7	PBFS6	PBFS5	PBFS4	PBFS3	PBFS2	PBFS1	PBFS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **PBFS7**: PB7 pin-shared function selection
0: I/O
1: PTPB_1

Bit 6 **PBFS6**: PB6 pin-shared function selection
0: I/O
1: PTP_1

Bit 5 **PBFS5**: PB5 pin-shared function selection
0: I/O
1: PTPB_0

Bit 4 **PBFS4**: PB4 pin-shared function selection
0: I/O
1: PTP_0

Bit 3~2 **PBFS3~PBFS2**: PB3/PB2 pin-shared function selection.
00: I/O
01: I/O
10: I/O
11: XT2/XT1

Bit 1~0 **PBFS1~PBFS0**: PB1/PB0 pin-shared function selection
00: I/O
01: I/O
10: I/O
11: OSC2/OSC1

• **PCFS Register – HT67F370**

Bit	7	6	5	4	3	2	1	0
Name	—	PCFS6	PCFS5	PCFS4	PCFS3	—	—	—
R/W	—	R/W	R/W	R/W	R/W	—	—	—
POR	—	0	0	0	0	—	—	—

Bit 7 Unimplemented, read as "0"

Bit 6 **PCFS6**: PC6 Pin-Shared function selection
0: PC6
1: AN11

Bit 5 **PCFS5**: PC5 Pin-Shared function selection
0: PC5
1: AN10

Bit 4 **PCFS4**: PC4 Pin-Shared function selection
0: PC4
1: AN9

Bit 3 **PCFS3**: PC3 Pin-Shared function selection
0: PC3
1: AN8

Bit 2~0 Unimplemented, read as "0"

• **PDFS Register – HT67F370**

Bit	7	6	5	4	3	2	1	0
Name	PDFS7	PDFS6	PDFS5	PDFS4	PDFS3	PDFS2	PDFS1	PDFS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 7 **PDFS7**: PD7 Pin-Shared function selection
0: PD7
1: SEG7/PTP1B

Bit 6 **PDFS6**: PD6 Pin-Shared function selection
0: PD6
1: SEG6/PTP1

Bit 5 **PDFS5**: PD5 Pin-Shared function selection
0: PD5
1: SEG5/PTP0B

Bit 4 **PDFS4**: PD4 Pin-Shared function selection
0: PD4
1: SEG4/PTP0

Bit 3 **PDFS3**: PD3 Pin-Shared function selection
0: PD3/PTCK0
1: SEG3

Bit 2 **PDFS2**: PD2 Pin-Shared function selection
0: PD2/STCK
1: SEG2

Bit 1 **PDFS1**: PD1 Pin-Shared function selection
0: PD1/CTCK
1: SEG1

Bit 0 **PDFS0**: PD0 Pin-Shared function selection
0: PD0/INT1/PTCK1
1: SEG0

• **PDFS Register – HT69F340**

Bit	7	6	5	4	3	2	1	0
Name	PDFS7	PDFS6	PDFS5	PDFS4	PDFS3	PDFS2	PDFS1	PDFS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 7 **PDFS7**: PD7 pin-shared function selection
0: I/O
1: SEG7/PTPB_1

Bit 6 **PDFS6**: PD6 pin-shared function selection
0: I/O
1: SEG6/PTP_1

Bit 5 **PDFS5**: PD5 pin-shared function selection
0: I/O
1: SEG5/PTPB_0

Bit 4 **PDFS4**: PD4 pin-shared function selection
0: I/O
1: SEG4/PTP_0

Bit 3 **PDFS3**: PD3 pin-shared function selection
0: I/O
1: SEG3

Bit 2 **PDFS2**: PD2 pin-shared function selection
0: I/O
1: SEG2/CTP_2

Bit 1 **PDFS1**: PD1 pin-shared function selection
0: I/O
1: SEG1

Bit 0 **PDFS0**: PD0 pin-shared function selection
0: I/O
1: SEG0

• **PDFS Register – HT69F350**

Bit	7	6	5	4	3	2	1	0
Name	PDFS7	PDFS6	PDFS5	PDFS4	PDFS3	PDFS2	PDFS1	PDFS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 7 **PDFS7**: PD7 pin-shared function selection
0: I/O
1: SEG7/PTPB_1

Bit 6 **PDFS6**: PD6 pin-shared function selection
0: I/O
1: SEG6/PTP_1

Bit 5 **PDFS5**: PD5 pin-shared function selection
0: I/O
1: SEG5/PTPB_0

Bit 4 **PDFS4**: PD4 pin-shared function selection
0: I/O
1: SEG4/PTP_0

Bit 3 **PDFS3**: PD3 pin-shared function selection
0: I/O
1: SEG3

Bit 2 **PDFS2**: PD2 pin-shared function selection
0: I/O
1: SEG2

Bit 1 **PDFS1**: PD1 pin-shared function selection
0: I/O
1: SEG1

Bit 0 **PDFS0**: PD0 pin-shared function selection
0: I/O
1: SEG0

- **PDFS Register – HT69F360**

Bit	7	6	5	4	3	2	1	0
Name	PDFS7	PDFS6	PDFS5	PDFS4	PDFS3	PDFS2	PDFS1	PDFS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 7 **PDFS7**: PD7 pin-shared function selection
0: I/O
1: SEG7/PTP1B

Bit 6 **PDFS6**: PD6 pin-shared function selection
00: I/O
01: SEG6/PTP1

Bit 5 **PDFS5**: PD5 pin-shared function selection
00: I/O
01: SEG5/PTP0B

Bit 4 **PDFS4**: PD4 pin-shared function selection
00: I/O
01: SEG4/PTP0

Bit 3 **PDFS3**: PD3 pin-shared function selection
0: I/O
1: SEG3

Bit 2 **PDFS2**: PD2 pin-shared function selection
00: I/O
01: SEG2

Bit 1 **PDFS1**: PD1 pin-shared function selection
00: I/O
01: SEG1

Bit 0 **PDFS0**: PD0 pin-shared function selection
00: I/O
01: SEG0

• **PEFS Register – HT67F370**

Bit	7	6	5	4	3	2	1	0
Name	PEFS7	PEFS6	PEFS5	PEFS4	PEFS3	PEFS2	PEFS1	PEFS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 7 **PEFS7**: PE7 Pin-Shared function selection
0: PE7
1: SEG15

Bit 6 **PEFS6**: PE6 Pin-Shared function selection
0: PE6
1: SEG14

Bit 5 **PEFS5**: PE5 Pin-Shared function selection
0: PE5
1: SEG13

Bit 4 **PEFS4**: PE4 Pin-Shared function selection
0: PE4
1: SEG12

Bit 3 **PEFS3**: PE3 Pin-Shared function selection
0: PE3
1: SEG11

Bit 2 **PEFS2**: PE2 Pin-Shared function selection
0: PE2
1: SEG10

Bit 1 **PEFS1**: PE1 Pin-Shared function selection
0: PE1
1: SEG9

Bit 0 **PEFS0**: PE0 Pin-Shared function selection
0: PE0
1: SEG8

• **PEFS Register – HT69F340/HT69F350/HT69F360**

Bit	7	6	5	4	3	2	1	0
Name	PEFS7	PEFS6	PEFS5	PEFS4	PEFS3	PEFS2	PEFS1	PEFS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 7 **PEFS7**: PE7 pin-shared function selection
0: I/O
1: SEG15

Bit 6 **PEFS6**: PE6 pin-shared function selection
0: I/O
1: SEG14

Bit 5 **PEFS5**: PE5 pin-shared function selection
0: I/O
1: SEG13

Bit 4 **PEFS4**: PE4 pin-shared function selection
0: I/O
1: SEG12

Bit 3 **PEFS3**: PE3 pin-shared function selection
0: I/O
1: SEG11

Bit 2 **PEFS2**: PE2 pin-shared function selection
0: I/O
1: SEG10

Bit 1 **PEFS1**: PE1 pin-shared function selection
0: I/O
1: SEG9

Bit 0 **PEFS0**: PE0 pin-shared function selection
0: I/O
1: SEG8

• **PFFS Register – HT67F370**

Bit	7	6	5	4	3	2	1	0
Name	PFFS7	PFFS6	PFFS5	PFFS4	PFFS3	PFFS2	PFFS1	PFFS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 7 **PFFS7**: PF7 Pin-Shared function selection
0: PF7
1: SEG23

Bit 6 **PFFS6**: PF6 Pin-Shared function selection
0: PF6
1: SEG22

Bit 5 **PFFS5**: PF5 Pin-Shared function selection
0: PF5
1: SEG21

Bit 4 **PFFS4**: PF4 Pin-Shared function selection
0: PF4
1: SEG20

Bit 3 **PFFS3**: PF3 Pin-Shared function selection
0: PF3
1: SEG19

Bit 2 **PFFS2**: PF2 Pin-Shared function selection
0: PF2
1: SEG18

Bit 1 **PFFS1**: PF1 Pin-Shared function selection
0: PF1
1: SEG17

Bit 0 **PFFS0**: PF0 Pin-Shared function selection
0: PF0
1: SEG16

• **PFFS Register – HT69F340/HT69F350/HT69F360**

Bit	7	6	5	4	3	2	1	0
Name	PFFS7	PFFS6	PFFS5	PFFS4	PFFS3	PFFS2	PFFS1	PFFS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 7 **PFFS7**: PF7 pin-shared function selection
0: I/O
1: SEG23

Bit 6 **PFFS6**: PF6 pin-shared function selection
0: I/O
1: SEG22

Bit 5 **PFFS5**: PF5 pin-shared function selection
0: I/O
1: SEG21

Bit 4 **PFFS4**: PF4 pin-shared function selection
0: I/O
1: SEG20

Bit 3 **PFFS3**: PF3 pin-shared function selection
0: I/O
1: SEG19

Bit 2 **PFFS2**: PF2 pin-shared function selection
0: I/O
1: SEG18

Bit 1 **PFFS1**: PF1 pin-shared function selection
0: I/O
1: SEG17

Bit 0 **PFFS0**: PF0 pin-shared function selection
0: I/O
1: SEG16

• **PGFS Register – HT67F370**

Bit	7	6	5	4	3	2	1	0
Name	PGFS7	PGFS6	PGFS5	PGFS4	PGFS3	PGFS2	PGFS1	PGFS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 7 **PGFS7**: PG7 Pin-Shared function selection
0: PG7
1: SEG31

Bit 6 **PGFS6**: PG6 Pin-Shared function selection
0: PG6
1: SEG30

Bit 5 **PGFS5**: PG5 Pin-Shared function selection
0: PG5
1: SEG29

Bit 4 **PGFS4**: PG4 Pin-Shared function selection
0: PG4
1: SEG28

Bit 3 **PGFS3**: PG3 Pin-Shared function selection
0: PG3
1: SEG27

Bit 2 **PGFS2**: PG2 Pin-Shared function selection
0: PG2
1: SEG26

Bit 1 **PGFS1**: PG1 Pin-Shared function selection
0: PG1
1: SEG25

Bit 0 **PGFS0**: PG0 Pin-Shared function selection
0: PG0
1: SEG24

• **PGFS Register – HT69F350/HT69F360**

Bit	7	6	5	4	3	2	1	0
Name	PGFS7	PGFS6	PGFS5	PGFS4	PGFS3	PGFS2	PGFS1	PGFS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 7 **PGFS7**: PG7 pin-shared function selection
0: I/O
1: SEG31

Bit 6 **PGFS6**: PG6 pin-shared function selection
0: I/O
1: SEG30

Bit 5 **PGFS5**: PG5 pin-shared function selection
0: I/O
1: SEG29

Bit 4 **PGFS4**: PG4 pin-shared function selection
0: I/O
1: SEG28

Bit 3 **PGFS3**: PG3 pin-shared function selection
0: I/O
1: SEG27

Bit 2 **PGFS2**: PG2 pin-shared function selection
0: I/O
1: SEG26

Bit 1 **PGFS1**: PG1 pin-shared function selection
0: I/O
1: SEG25

Bit 0 **PGFS0**: PG0 pin-shared function selection
0: I/O
1: SEG24

• **PHFS Register – HT67F370**

Bit	7	6	5	4	3	2	1	0
Name	PHFS7	PHFS6	PHFS5	PHFS4	PHFS3	PHFS2	PHFS1	PHFS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 7 **PHFS7**: PH7 Pin-Shared function selection
0: PH7
1: SEG39

Bit 6 **PHFS6**: PH6 Pin-Shared function selection
0: PH6
1: SEG38

Bit 5 **PHFS5**: PH5 Pin-Shared function selection
0: PH5
1: SEG37

Bit 4 **PHFS4**: PH4 Pin-Shared function selection
0: PH4
1: SEG36

Bit 3 **PHFS3**: PH3 Pin-Shared function selection
0: PH3
1: SEG35

Bit 2 **PHFS2**: PH2 Pin-Shared function selection
0: PH2
1: SEG34

Bit 1 **PHFS1**: PH1 Pin-Shared function selection
0: PH1
1: SEG33

Bit 0 **PHFS0**: PH0 Pin-Shared function selection
0: PH0
1: SEG32

• **PHFS Register – HT69F350**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	PHFS3	PHFS2	PHFS1	PHFS0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	1	1	1	1

Bit 7~4 Unimplemented, read as "0"

Bit 3 **PHFS3**: PH3 pin-shared function selection
0: I/O
1: SEG35

Bit 2 **PHFS2**: PH2 pin-shared function selection
0: I/O
1: SEG34

Bit 1 **PHFS1**: PH1 pin-shared function selection
0: I/O
1: SEG33

Bit 0 **PHFS0**: PH0 pin-shared function selection
0: I/O
1: SEG32

• **PHFS Register – HT69F360**

Bit	7	6	5	4	3	2	1	0
Name	PHFS7	PHFS6	PHFS5	PHFS4	PHFS3	PHFS2	PHFS1	PHFS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 3 **PHFS7**: PH7 pin-shared function selection
0: I/O
1: SEG39

Bit 2 **PHFS6**: PH6 pin-shared function selection
0: I/O
1: SEG38

Bit 1 **PHFS5**: PH5 pin-shared function selection
0: I/O
1: SEG37

Bit 0 **PHFS4**: PH4 pin-shared function selection
0: I/O
1: SEG36

Bit 3 **PHFS3**: PH3 pin-shared function selection
0: I/O
1: SEG35

Bit 2 **PHFS2**: PH2 pin-shared function selection
0: I/O
1: SEG34

Bit 1 **PHFS1**: PH1 pin-shared function selection
0: I/O
1: SEG33

Bit 0 **PHFS0**: PH0 pin-shared function selection
0: I/O
1: SEG32

• **SFSR Register – HT67F370**

Bit	7	6	5	4	3	2	1	0
Name	SFS7	SFS6	SFS5	SFS4	SFS3	SFS2	SFS1	SFS0
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 **SFS7**: STCK source selection
0: PA2
1: PD2

Bit 6 **SFS6**: PTCK0 source selection
0: PA0
1: PD3

Bit 5 **SFS5**: CTCK source selection
0: PA2
1: PD1

Bit 4 **SFS4**: INT1 source selection
0: PA0
1: PD0

Bit 3 **SFS3**: PD7 Special function selection
0: SEG7
1: PTP1B

Bit 2 **SFS2**: PD6 Special function selection
0: SEG6
1: PTP1

Bit 1 **SFS1**: PD5 Special function selection
0: SEG5
1: PTP0B

Bit 0 **SFS0**: PD4 Special function selection
0: SEG4
1: PTP0

• **SFS Register – HT69F340**

Bit	7	6	5	4	3	2	1	0
Name	SFS7	SFS6	SFS5	SFS4	SFS3	SFS2	SFS1	SFS0
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 **SFS7**: PD2 Special function selection
0: SEG2
1: CTP_2

Bit 6 **SFS6**: PTCK source selection
0: PA3
1: PD3

Bit 5 **SFS5**: CTCK source selection
0: PA2
1: PD1

Bit 4 **SFS4**: INT1 source selection
0: PA0
1: PD0

Bit 3 **SFS3**: PD7 Special function selection
0: SEG7
1: PTPB_1

Bit 2 **SFS2**: PD6 Special function selection
0: SEG6
1: PTP_1

Bit 1 **SFS1**: PD5 Special function selection
0: SEG5
1: PTPB_0

Bit 0 **SFS0**: PD4 Special function selection
0: SEG4
1: PTP_0

- **SFSR Register – HT69F350**

Bit	7	6	5	4	3	2	1	0
Name	SFS7	SFS6	SFS5	SFS4	SFS3	SFS2	SFS1	SFS0
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 **SFS7**: STCK source selection
0: PA2
1: PD2

Bit 6 **SFS6**: PTCK source selection
0: PA0
1: PD3

Bit 5 **SFS5**: CTCK source selection
0: PA2
1: PD1

Bit 4 **SFS4**: INT1 source selection
0: PA0
1: PD0

Bit 3 **SFS3**: PD7 Special function selection
0: SEG7
1: PTPB_1

Bit 2 **SFS2**: PD6 Special function selection
0: SEG6
1: PTP_1

Bit 1 **SFS1**: PD5 Special function selection
0: SEG5
1: PTPB_0

Bit 0 **SFS0**: PD4 Special function selection
0: SEG4
1: PTP_0

• **SFSR Register – HT69F360**

Bit	7	6	5	4	3	2	1	0
Name	SFS7	SFS6	SFS5	SFS4	SFS3	SFS2	SFS1	SFS0
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 **SFS7**: STCK source selection
0: PA2
1: PD2

Bit 6 **SFS6**: PTCK0 source selection
0: PA0
1: PD3

Bit 5 **SFS5**: CTCK source selection
0: PA2
1: PD1

Bit 4 **SFS4**: INT1 source selection
0: PA0
1: PD0

Bit 3 **SFS3**: PD7 Special function selection
0: SEG7
1: PTP1B

Bit 2 **SFS2**: PD6 Special function selection
0: SEG6
1: PTP1

Bit 1 **SFS1**: PD5 Special function selection
0: SEG5
1: PTP0B

Bit 0 **SFS0**: PD4 Special function selection
0: SEG4
1: PTP0

• **SFSR1 Register – HT67F370**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PA4FS	PB4FS	PB5FS	PB7FS	PB6FS	SFS8
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 **PA4FS**: PA4 Pin-Shared function selection
0: PA4/INT0/PTCK1
1: AN5

Bit 4 **PB4FS**: PB4 Pin-Shared function selection, together with the PBFS4 bit in the PBFS register
[PB4FS:PBFS4]=
00: PB4
01: PTP0
10: VREFI
11: PB4

Bit 3 **PB5FS**: PB5 Pin-Shared function selection, together with the PBFS5 bit in the PBFS register
[PB5FS:PBFS5]=
00: PB5
01: PTP0B
10: VREF
11: AN0

Bit 2 **PB7FS**: PB7 Pin-Shared function selection, together with the PBFS7 bit in the PBFS register [PBFS7:PB7FS]=
00: PB7
01: AN2
10: PTP1B
11: TX

Bit 1 **PB6FS**: PB6 Pin-Shared function selection, together with the PBFS6 bit in the PBFS register [PBFS6:PB6FS]=
00: PB6
01: AN1
10: PTP1
11: RX

Bit 0 **SFS8**: PTCK1 source selection
0: PA4
1: PD0

- **SFSR1 Register – HT69F360**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	PB7FS	PB6FS	SFS8
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 Unimplemented, read as "0"

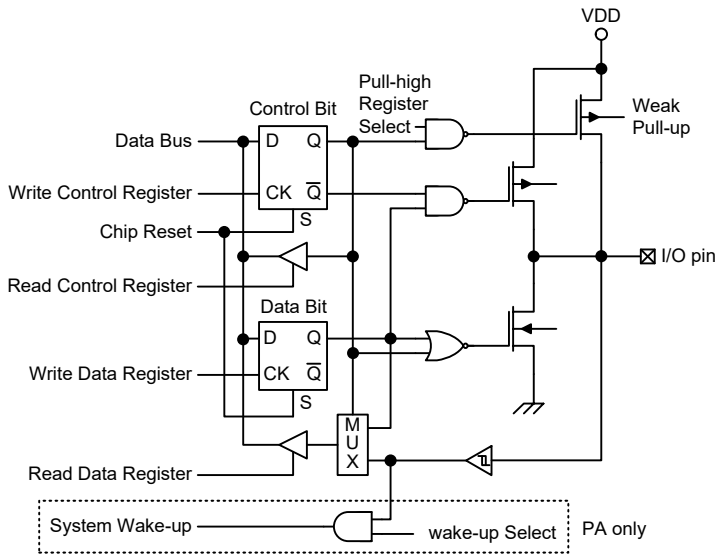
Bit 2 **PB7FS**: PB7 pin-shared function selection
0: PTP1B
1: UART TX

Bit 1 **PB6FS**: PB6 pin-shared function selection
0: PTP1
1: UART RX

Bit 0 **SFS8**: PTCK1 source selection
0: PA4
1: PD0

I/O Pin Structures

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



Generic Input/Output Structure

Programming Considerations

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

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

Timer Modules – TM

One of the most fundamental functions in any microcontroller device is the ability to control and measure time. To implement time related functions the devices include 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 Compact, Standard and Periodic TM sections.

Introduction

The devices contain two to four TMs depending upon which device is selected. The HT69F340 device contains a 10-bit Compact TM-CTM and a 10-bit Periodic TM-PTM. The HT69F350 device contains a 10-bit Compact TM-CTM, a 16-bit Standard TM-STM and a 10-bit Periodic TM-PTM. The HT69F360/HT67F370 device contains a 10-bit Compact TM-CTM, a 16-bit Standard TM-STM and two10-bit Periodic TMs-PTM0 and PTM1.Although similar in nature, the different TM types vary in their feature complexity. The common features to the Compact, Standard and Periodic TMs will be described in this section and the detailed operation will be described in corresponding sections. The main features and differences between the three types of TMs are summarised in the accompanying table.

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

TM Function Summary

Device	CTM	STM	PTM
HT69F340	10-bit CTM	—	10-bit PTM
HT69F350	10-bit CTM	16-bit STM	10-bit PTM
HT69F360/ HT67F370			10-bit PTM0 10-bit PTM1

TM Type Reference

TM Operation

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

TM External Pins

Each of the TMs, irrespective of what type, has one TM input pin, with the label xTCKn. The TM input pin, is essentially a clock source for the TM 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. This external TM input pin is shared with other functions but will be connected to the internal TM if selected using the xTnCK2~xTnCK0 bits. The TM input pin can be chosen to have either a rising or falling active edge. The xTCKn pins are also used as the external trigger input pin in single pulse output mode for the xTMn.

The TMs each have one or more output pins with the label xTPn and PTPnB. The PTPnB is the inverted signal of the PTPn output. 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. The xTPn pin acts as an input when the TM is setup to operate in the Capture Input Mode. As the xTPn pins are pin-shared with other functions, the xTPn output function must first be setup using registers. A single bit in one of the registers determines if its associated pin is to be used as an external TM output pin or if it is to have another function.

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

Device	CTM	STM	PTM0	PTM1
HT69F340	CTP_0, CTP_1, CTP_2	—	PTP_0, PTP_1, PTP_2, PTPB_0, PTPB_1	—
HT69F350	CTP_0, CTP_1	STP_0, STP_1	PTP_0, PTP_1 PTPB_0, PTPB_1	—
HT69F360/ HT67F370	CTP_0, CTP_1	STP_0, STP_1	PTP0 PTP0B	PTP1 PTP1B

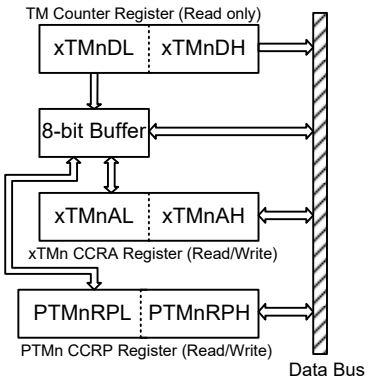
TM Output Pins

Programming Considerations

The TM Counter Registers and the Capture/Compare CCRA or CCRP register, being either 10-bit or 16-bit, all have a low and high byte structure. The high bytes can be directly accessed, but as the low bytes can only be accessed via an internal 8-bit buffer, reading or writing to these register pairs must be carried out in a specific way. The important point to note is that data transfer to and from the 8-bit

buffer and its related low byte only takes place when a write or read operation to its corresponding high byte is executed.

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



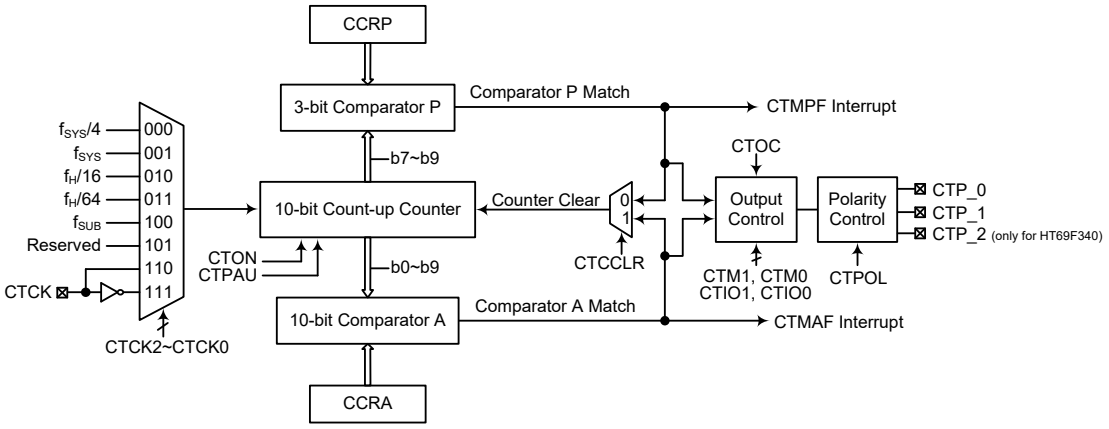
The following steps show the read and write procedures:

- Writing Data to CCRA or CCRP
  - Step 1. Write data to Low Byte 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 and 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.

Compact Type TM – CTM

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

Device	Name	TM Input Pin	TM Output Pin
HT69F340	10-bit CTM	CTCK	CTP_0,CTP_1,CTP_2
HT69F350	10-bit CTM	CTCK	CTP_0, CTP_1
HT69F360/ HT67F370	10-bit CTM	CTCK	CTP_0, CTP_1



Note: The CTM external pins are pin-shared with other functions, so before using the CTM function, ensure that the pin-shared function registers have been set properly to enable the CTM pin function. The CTCK pin, if used, must also be set as an input by setting the corresponding bits in the port control register.

Compact Type TM Block Diagram

Compact TM Operation

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

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

Compact Type TM Register Description

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

Register Name	Bit							
	7	6	5	4	3	2	1	0
CTMC0	CTPAU	CTCK2	CTCK1	CTCK0	CTON	CTRP2	CTRP1	CTRP0
CTMC1	CTM1	CTM0	CTIO1	CTIO0	CTOC	CTPOL	CTDPX	CTCCLR
CTMDL	D7	D6	D5	D4	D3	D2	D1	D0
CTMDH	—	—	—	—	—	—	D9	D8
CTMAL	D7	D6	D5	D4	D3	D2	D1	D0
CTMAH	—	—	—	—	—	—	D9	D8

Compact TM Register List

- **CTMC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	CTPAU	CTCK2	CTCK1	CTCK0	CTON	CTRP2	CTRP1	CTRP0
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 **CTPAU**: CTM Counter Pause Control
0: Run
1: Pause

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

Bit 6~4 **CTCK2~CTCK0**: Select CTM Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: Reserved
110: CTCK rising edge clock
111: CTCK falling edge clock

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

Bit 3 **CTON**: CTM Counter On/Off Control
0: Off
1: On

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

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

Bit 2~0 **CTRP2~CTRP0**: CTM CCRP 3-bit register, compared with the CTM Counter bit 9~bit 7
Comparator P Match Period
000: 1024 CTM clocks
001: 128 CTM clocks
010: 256 CTM clocks
011: 384 CTM clocks
100: 512 CTM clocks
101: 640 CTM clocks
110: 768 CTM clocks
111: 896 CTM 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 CTCCLR bit is set to zero. Setting the CTCCLR 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.

• CTMC1 Register

Bit	7	6	5	4	3	2	1	0
Name	CTM1	CTM0	CTIO1	CTIO0	CTOC	CTPOL	CTDPX	CTCCLR
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 **CTM1~CTM0**: Select CTM Operating Mode
00: Compare Match Output Mode
01: Undefined
10: PWM Mode
11: Timer/Counter Mode
These bits setup the required operating mode for the TM. To ensure reliable operation the TM should be switched off before any changes are made to the bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4 **CTIO1~CTIO0**: Select CTM External Pins Function
Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output
PWM Mode
00: PWM output inactive state
01: PWM output active state
10: PWM output
11: Undefined
Timer/counter Mode
Unused
These two bits are used to determine how the TM external pin changes state when a certain condition is reached. The function that these bits select depends upon in which mode the TM is running.
In the Compare Match Output Mode, the CTIO1~CTIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the CTIO1~CTIO0 bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the CTOC bit. Note that the output level requested by the CTIO1~CTIO0 bits must be different from the initial value setup using the CTOC

bit otherwise no change will occur on the TM output pin when a compare match occurs. After the TM output pin changes state it can be reset to its initial level by changing the level of the CTON bit from low to high.

In the PWM Mode, the CTIO1 and CTIO0 bits determine how the TM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to change the values of the CTIO1 and CTIO0 bits only after the TM has been switched off. Unpredictable PWM outputs will occur if the CTIO1 and CTIO0 bits are changed when the TM is running.

Bit 3 **CTOC**: CTM Output control bit

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

PWM Mode
0: Active low
1: Active high

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

Bit 2 **CTPOL**: CTM Output polarity Control
0: Non-invert
1: Invert

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

Bit 1 **CTDPX**: CTM PWM period/duty Control
0: CCRP – period; CCRA – duty
1: CCRP – duty; CCRA – period

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

Bit 0 **CTCCLR**: Select CTM Counter clear condition
0: CTM Comparatror P match
1: CTM Comparatror A match

This bit is used to select the method which clears the counter. Remember that the Compact TM contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the CTCCLR 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 CTCCLR bit is not used in the PWM Mode.

• **CTMDL 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**: CTM Counter Low Byte Register bit 7 ~ bit 0
CTM 10-bit Counter bit 7 ~ bit 0

• **CTMDH 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**: CTM Counter High Byte Register bit 1 ~ bit 0
CTM 10-bit Counter bit 9 ~ bit 8

• **CTMAL 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**: CTM CCRA Low Byte Register bit 7 ~ bit 0
CTM 10-bit CCRA bit 7 ~ bit 0

• **CTMAH 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**: CTM CCRA High Byte Register bit 1 ~ bit 0
CTM 10-bit CCRA bit 9 ~ bit 8

Compact Type TM Operating Modes

The Compact Type TM can operate in one of three operating modes, Compare Match Output Mode, PWM Output Mode or Timer/Counter Mode. The operating mode is selected using the CTM1 and CTM0 bits in the CTMC1 register.

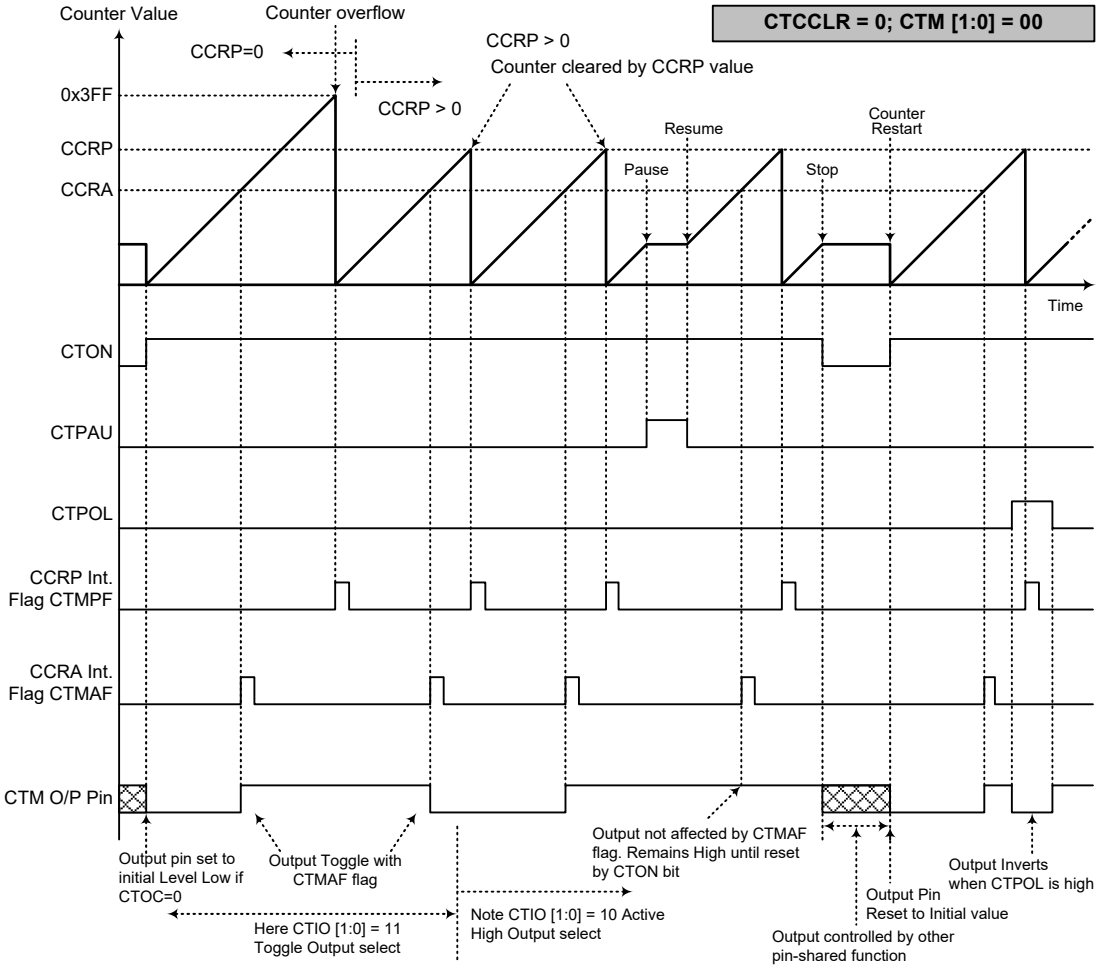
Compare Match Output Mode

To select this mode, bits CTM1 and CTM0 in the CTMC1 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 CTCCLR 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 CTMAF and CTMPF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

If the CTCCLR bit in the CTMC1 register is high then the counter will be cleared when a compare

match occurs from Comparator A. However, here only the CTMAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when CTCCLR is high no CTMPF interrupt request flag will be generated. If the CCRA bits are all zero, the counter will overflow when its reaches its maximum 10-bit, 3FF Hex, value, however here the CTMAF interrupt request flag will not be generated.

As the name of the mode suggests, after a comparison is made, the TM output pin, will change state. The TM output pin condition however only changes state when a CTMAF interrupt request flag is generated after a compare match occurs from Comparator A. The CTMPF interrupt request flag, generated from a compare match occurs from Comparator P, will have no effect on the TM output pin. The way in which the TM output pin changes state are determined by the condition of the CTIO1 and CTIO0 bits in the CTMC1 register. The TM output pin can be selected using the CTIO1 and CTIO0 bits to go high, to go low or to toggle from its present condition when a compare match occurs from Comparator A. The initial condition of the TM output pin, which is setup after the CTON bit changes from low to high, is setup using the CTOC bit. Note that if the CTIO1 and CTIO0 bits are zero then no pin change will take place.

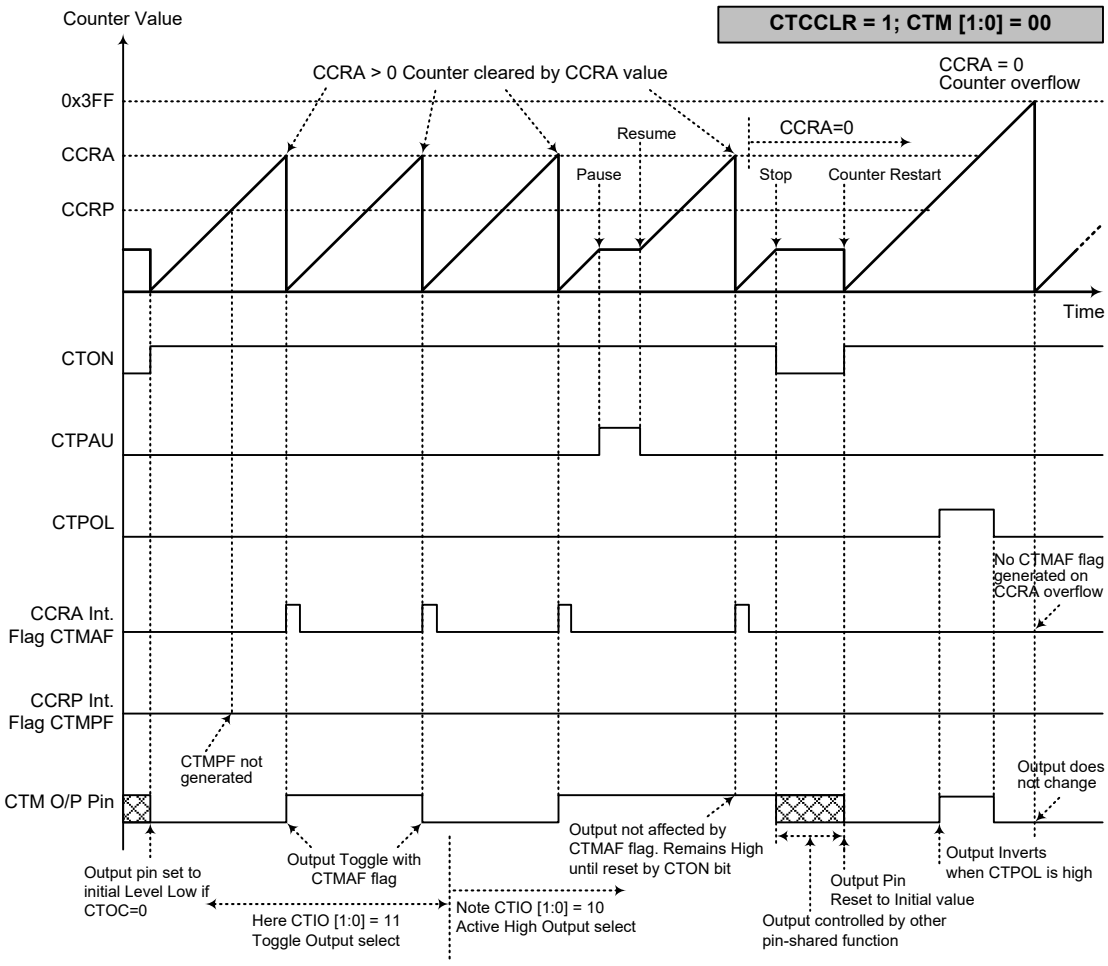


Compare Match Output Mode – CTCCLR=0

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

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

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



Compare Match Output Mode – CTCCLR=1

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

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

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

4. The CTMPF flags is not generated when CTCCLR=1

Timer/Counter Mode

To select this mode, bits CTM1 and CTM0 in the CTMC1 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 CTM1 and CTM0 in the CTMC1 register should be set to 10 respectively. The PWM function within the TM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the TM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM mode, the CTCCLR 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 CTDPX bit in the CTMC1 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 CTOC bit In the CTMC1 register is used to select the required polarity of the PWM waveform while the two CTIO1 and CTIO0 bits are used to enable the PWM output or to force the TM output pin to a fixed high or low level. The CTPOL bit is used to reverse the polarity of the PWM output waveform.

- **10-bit CTM, PWM Mode, Edge-aligned Mode, CTDPX=0**

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

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

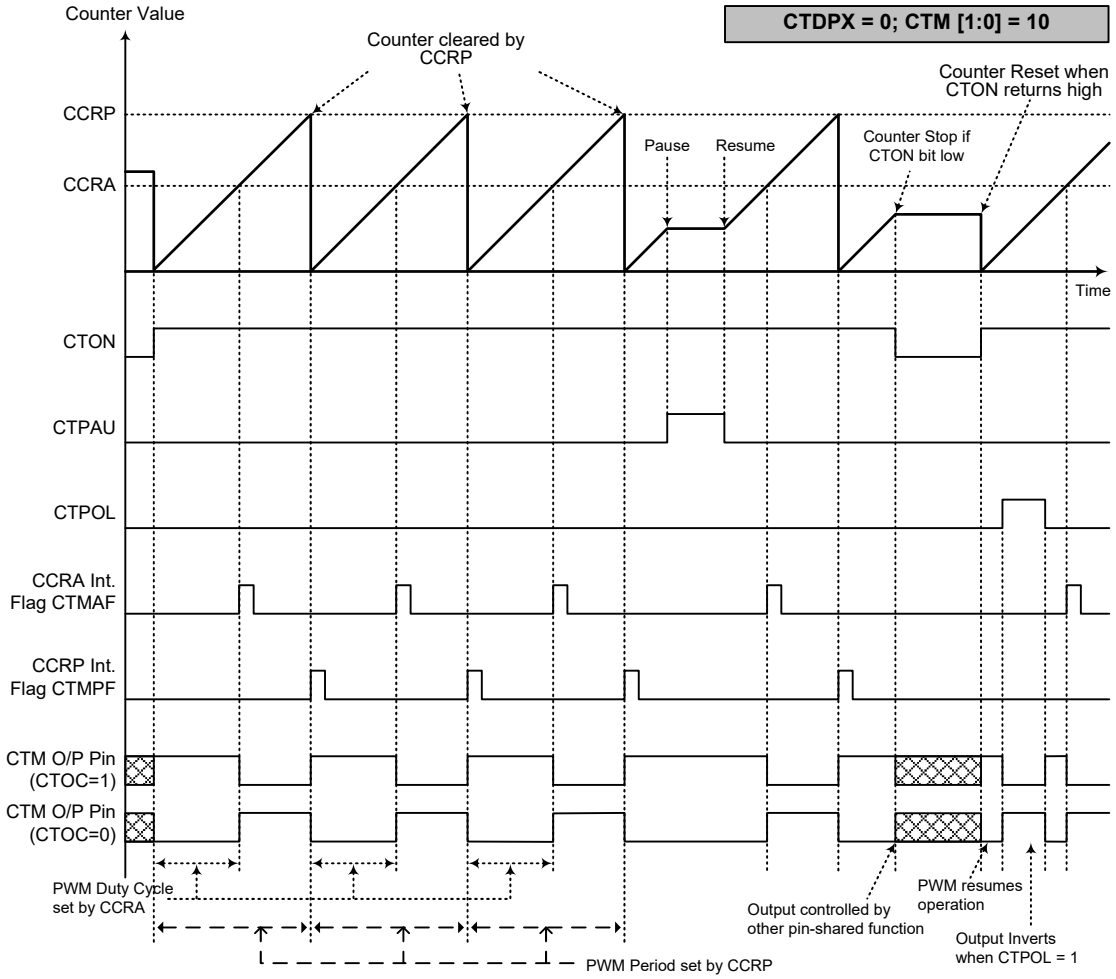
The CTM PWM output frequency=(f_{SYS}/4) / 512=f_{SYS}/2048=7.8125 kHz, duty=128/512=25%.

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

- **10-bit CTM, PWM Mode, Edge-aligned Mode, CTDPX=1**

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

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



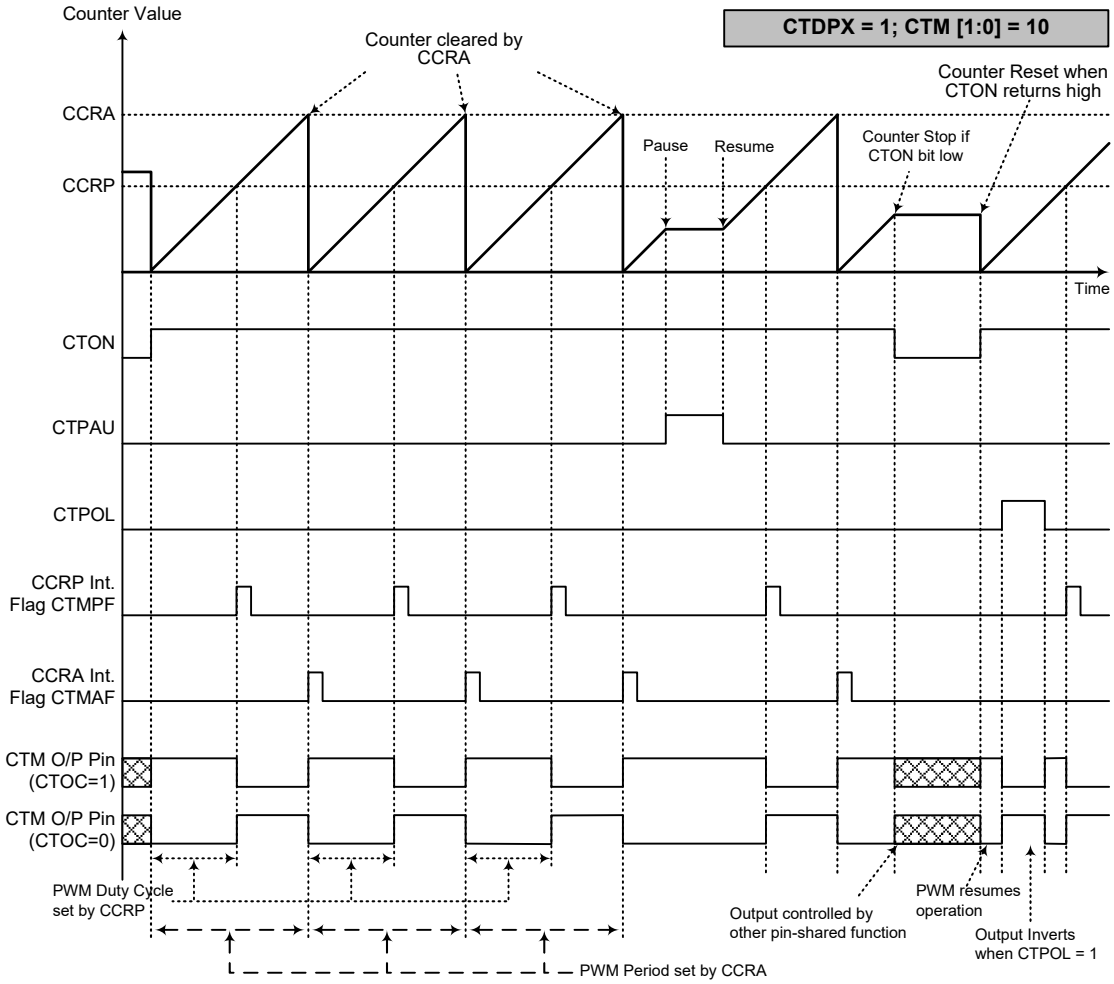
PWM Output Mode – CTDPX=0

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

2. A counter clear sets PWM Period

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

4. The CTCCLR bit has no influence on PWM operation



PWM Output Mode – CTDPX=1

Note: 1. Here CTDPX=1 – Counter cleared by CCRA

2. A counter clear sets PWM Period

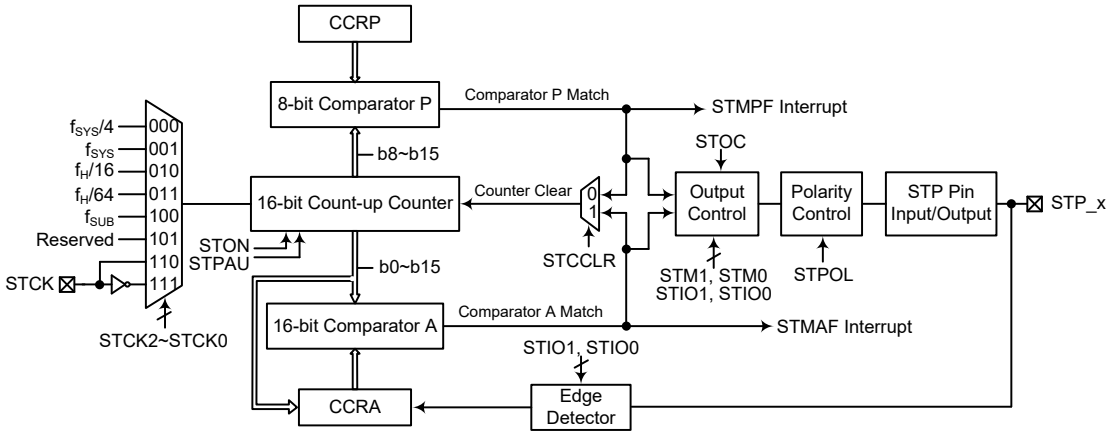
3. The internal PWM function continues even when CTIO[1:0]=00 or 01

4. The CTCCLR bit has no influence on PWM operation

Standard Type TM – STM

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

Device	TM Type	TM Input Pin	TM Output Pin
HT69F350	16-bit STM	STCK	STP_0, STP_1
HT69F360/ HT67F370	16-bit STM	STCK	STP_0, STP_1



Note: The STM external pins are pin-shared with other functions, so before using the STM function, ensure that the pin-shared function registers have been set properly to enable the STM pin function. The STCK pin, if used, must also be set as an input by setting the corresponding bits in the port control register.

Standard Type TM Block Diagram

Standard TM Operation

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

The only way of changing the value of the 16-bit counter using the application program, is to clear the counter by changing the STON bit from low to high. The counter will also be cleared automatically by a counter overflow or a compare match with one of its associated comparators. When these conditions occur, a TM interrupt signal will also usually be generated. The Standard Type TM can operate in a number of different operational modes, can be driven by different clock sources 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 series of registers. A read only register pair exists to store the internal counter 16-bit value, while a read/write register pair exists to store the internal 16-bit CCRA value. The remaining two registers are control registers which setup the different operating and control modes as well as eight CCRP bits.

Register Name	Bit							
	7	6	5	4	3	2	1	0
STMC0	STPAU	STCK2	STCK1	STCK0	STON	—	—	—
STMC1	STM1	STM0	STIO1	STIO0	STOC	STPOL	STDPX	STCCLR
STMDL	D7	D6	D5	D4	D3	D2	D1	D0
STMDH	D15	D14	D13	D12	D11	D10	D9	D8
STMAL	D7	D6	D5	D4	D3	D2	D1	D0
STMAH	D15	D14	D13	D12	D11	D10	D9	D8
STMRP	D7	D6	D5	D4	D3	D2	D1	D0

16-bit Standard TM Register List

- **STMC0 Register**

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

Bit 7 **STPAU**: STM Counter Pause Control
0: Run
1: Pause

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

Bit 6~4 **STCK2~STCK0**: Select STM Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: Reserved
110: STCK rising edge clock
111: STCK falling edge clock

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

Bit 3 **STON**: STM Counter On/Off Control
0: Off
1: On

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

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

Bit 2~0 Unimplemented, read as “0”

• **STMC1 Register**

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

Bit 7~6 **STM1~STM0**: Select STM Operating Mode
00: Compare Match Output Mode
01: Capture Input Mode
10: PWM Mode or Single Pulse Output Mode
11: Timer/Counter Mode
These bits setup the required operating mode for the TM. To ensure reliable operation the TM should be switched off before any changes are made to the bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4 **STIO1~STIO0**: Select STM External Pins Function
Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output
PWM Mode/Single Pulse Output Mode
00: PWM output inactive state
01: PWM output active state
10: PWM output
11: Single pulse output
Capture Input Mode
00: Input capture at rising edge of STP
01: Input capture at falling edge of STP
10: Input capture at falling/rising edge of STP
11: Input capture disabled
Timer/counter Mode
Unused
These two bits are used to determine how the TM external pin changes state when a certain condition is reached. The function that these bits select depends upon in which mode the TM is running.
In the Compare Match Output Mode, the STIO1~STIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the STIO1~STIO0 bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the STOC bit. Note that the output level requested by the STIO1~STIO0 bits must be different from the initial value setup using the STOC bit otherwise no change will occur on the TM output pin when a compare match occurs. After the TM output pin changes state it can be reset to its initial level by changing the level of the STON bit from low to high.
In the PWM Mode, the STIO1 and STIO0 bits determine how the TM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to change the values of the STIO1 and STIO0 bits only after the TM has been switched off. Unpredictable PWM outputs will occur if the STIO1 and STIO0 bits are changed when the TM is running.

Bit 3 **STOC**: STM Output control bit
Compare Match Output Mode
0: Initial low
1: Initial high

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

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

Bit 2 **STPOL**: STM Output polarity Control
0: Non-invert
1: Invert

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

Bit 1 **STDPX**: STM PWM period/duty Control
0: CCRP - period; CCRA - duty
1: CCRP - duty; CCRA - period

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

Bit 0 **STCCLR**: Select STM Counter clear condition
0: TM Comparator P match
1: TM 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 STCCLR bit set high, the counter will be cleared when a compare match occurs from the Comparator A. When the bit is low, the counter will be cleared when a compare match occurs from the Comparator P or with a counter overflow. A counter overflow clearing method can only be implemented if the CCRP bits are all cleared to zero. The STCCLR bit is not used in the PWM, Single Pulse or Input Capture Mode.

• STMDL Register

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

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

• STMDH Register

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

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

• **STMAL Register**

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

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

• **STMAH 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**: STM CCRA High Byte Register bit 7 ~ bit 0
STM 16-bit CCRA bit 15 ~ bit 8

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

Standard Type TM Operating Modes

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

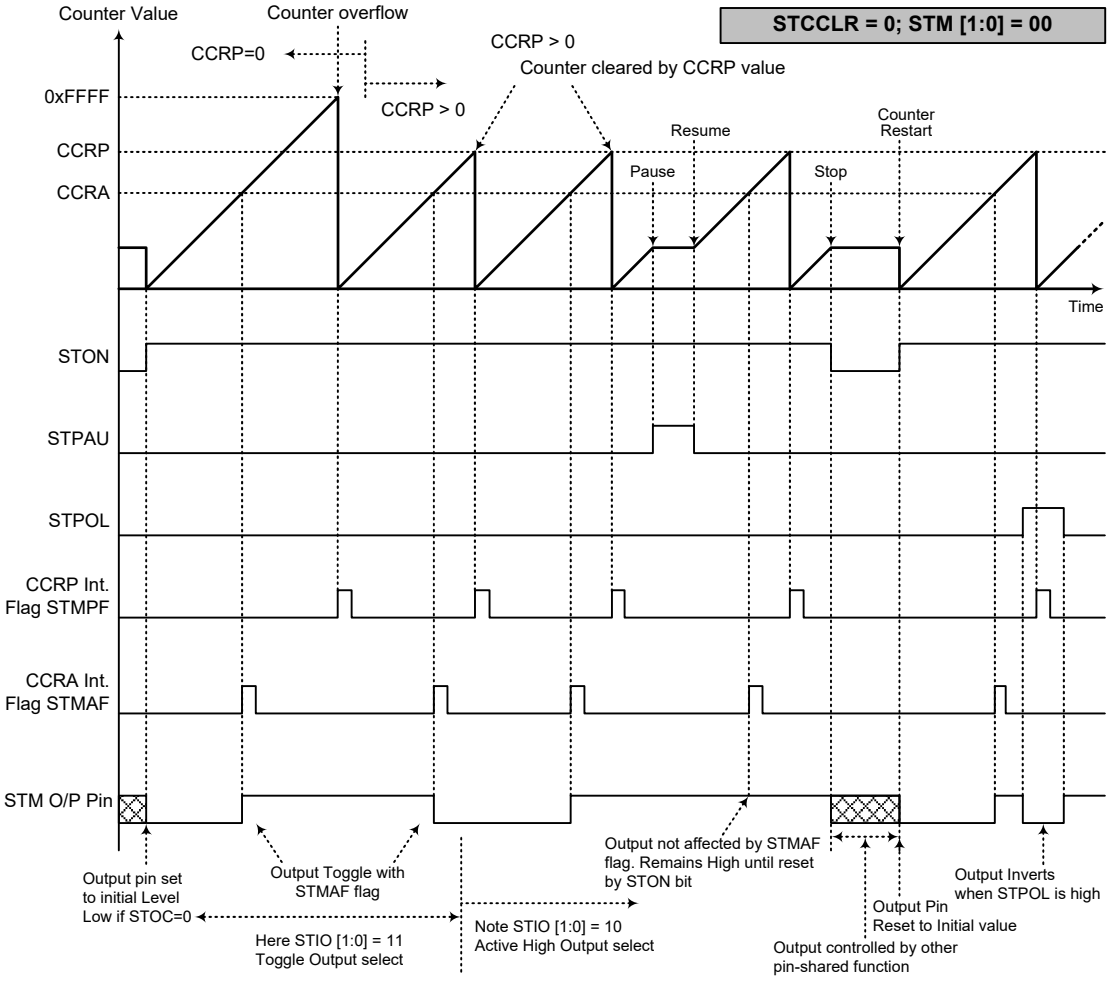
Compare Match Output Mode

To select this mode, bits STM1 and STM0 in the STMC1 register, should be set to 00 respectively. In this mode once the counter is enabled and running it can be cleared by three methods. These are a counter overflow, a compare match from Comparator A and a compare match from Comparator P. When the STCCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both STMAF and STMPF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

If the STCCLR bit in the STMC1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the STMAF interrupt request flag will be

generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when STCCLR is high no STMPF interrupt request flag will be generated. In the Compare Match Output Mode, the CCRA can not be set to "0".

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

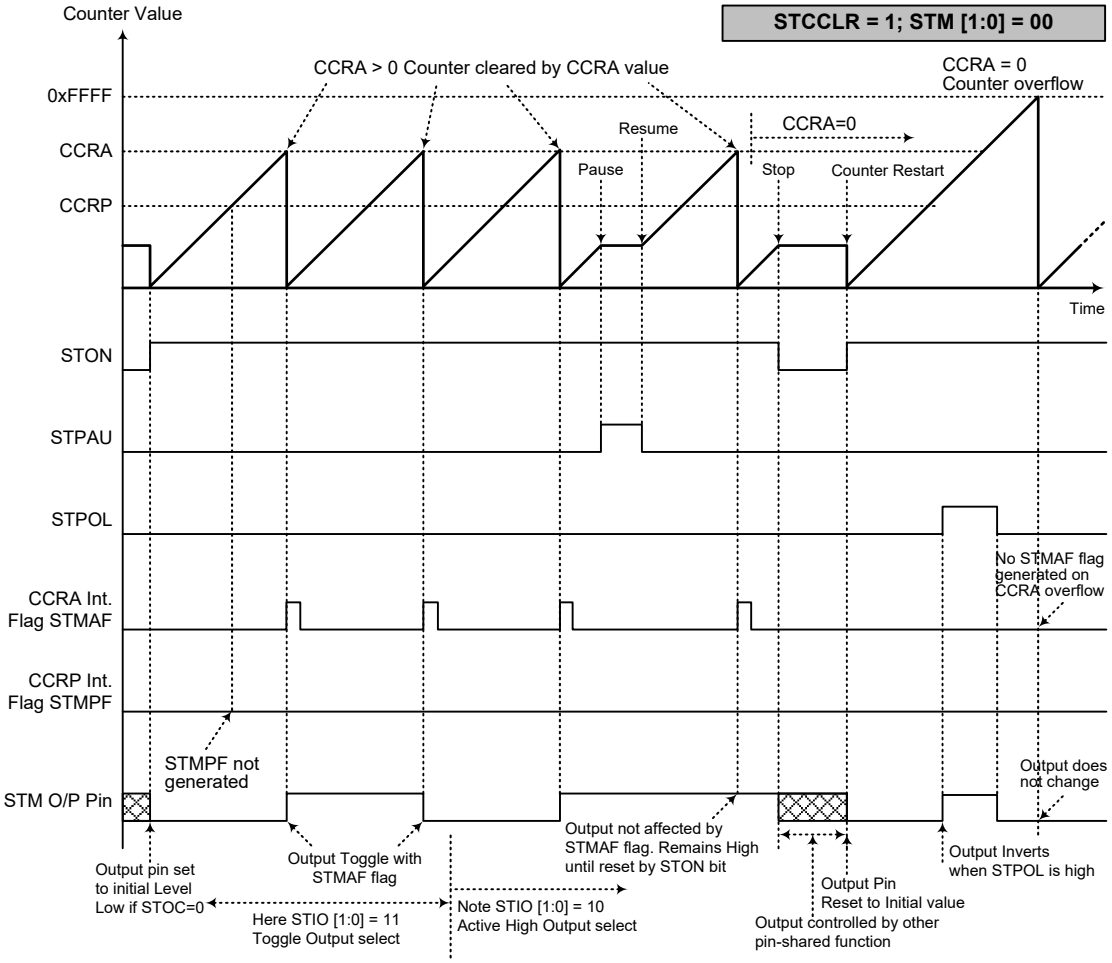


Compare Match Output Mode – STCCLR=0

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

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

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



Compare Match Output Mode – STCCLR=1

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

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

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

4. The STMPF flags is not generated when STCCLR=1

Timer/Counter Mode

To select this mode, bits STM1 and STM0 in the STMC1 register should be set to 11 respectively. The Timer/Counter Mode operates in an identical way to the Compare Match Output Mode generating the same interrupt flags. The exception is that in the Timer/Counter Mode the 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 STM1 and STM0 in the STMC1 register should be set to 10 respectively and also the STIO1 and STIO0 bits should be set to 10 respectively. The PWM function within the TM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the TM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

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

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

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

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

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

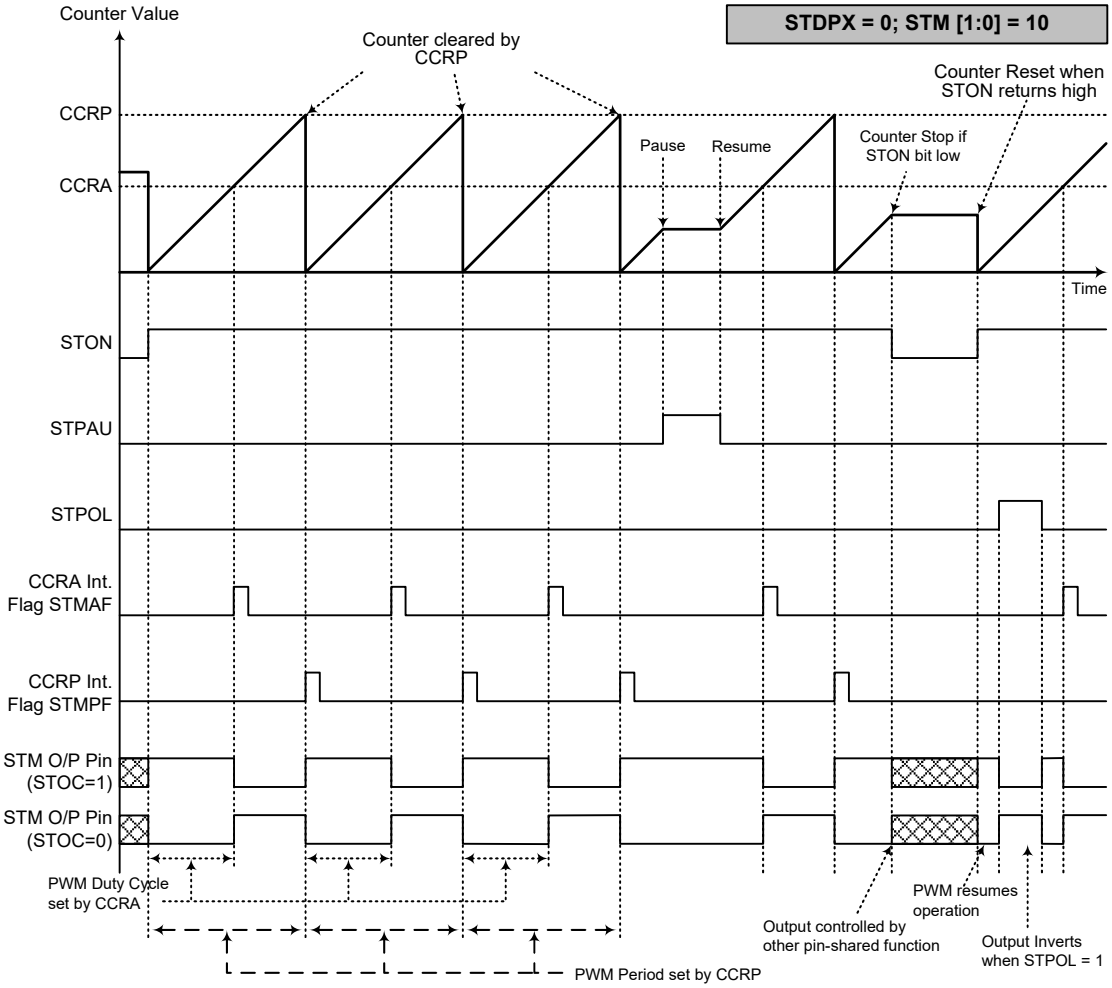
The STM PWM output frequency=(f_{SYS}/4)/512=f_{SYS}/2048=7.8125 kHz, duty=128/512=25%.

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

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

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

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



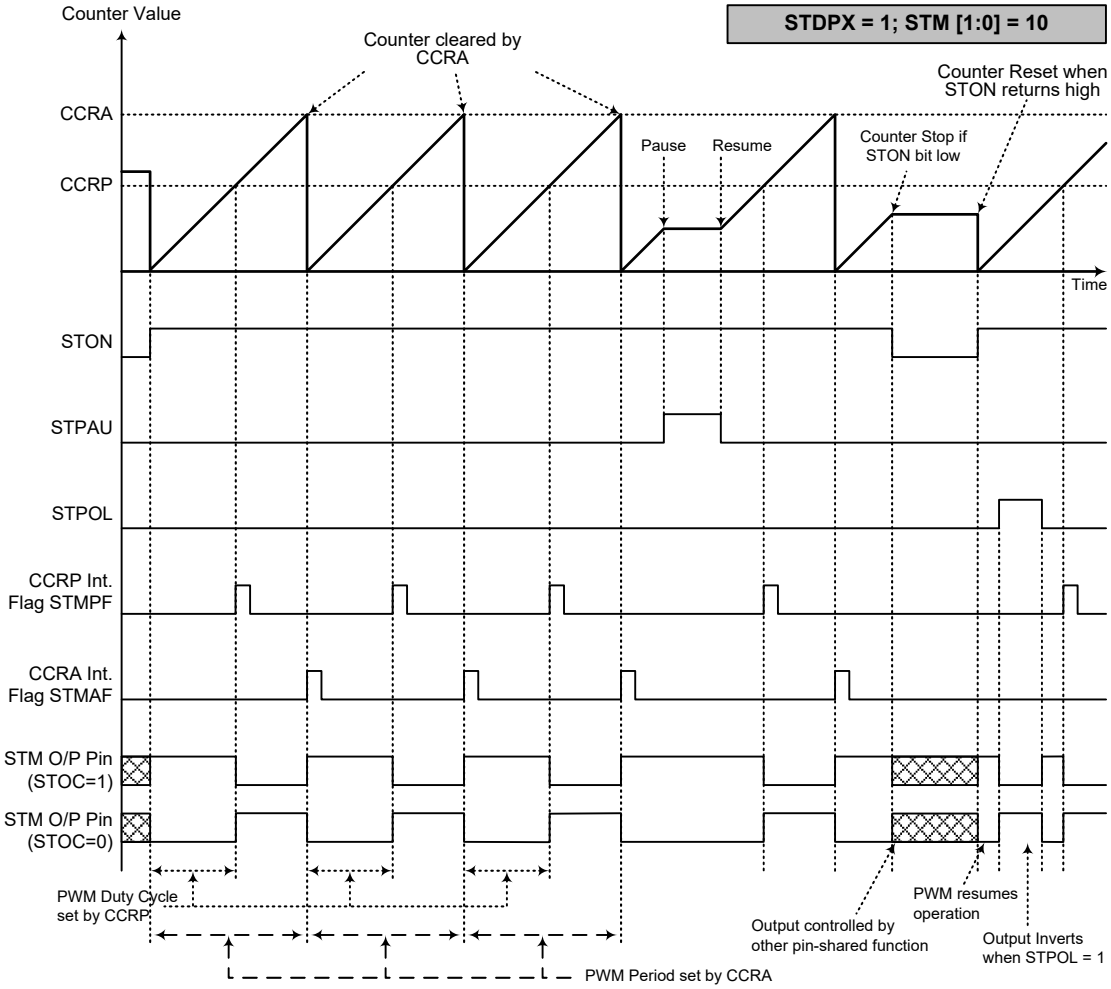
PWM Output Mode – STDPX=0

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

2. A counter clear sets PWM Period

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

4. The STCCLR bit has no influence on PWM operation



PWM Output Mode – STDPX=1

Note: 1. Here STDPX=1 – Counter cleared by CCRA

2. A counter clear sets PWM Period

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

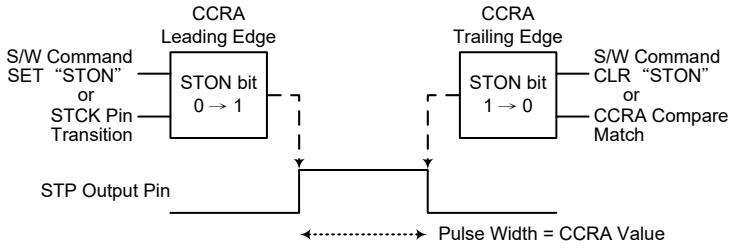
4. The STCCLR bit has no influence on PWM operation

Single Pulse Mode

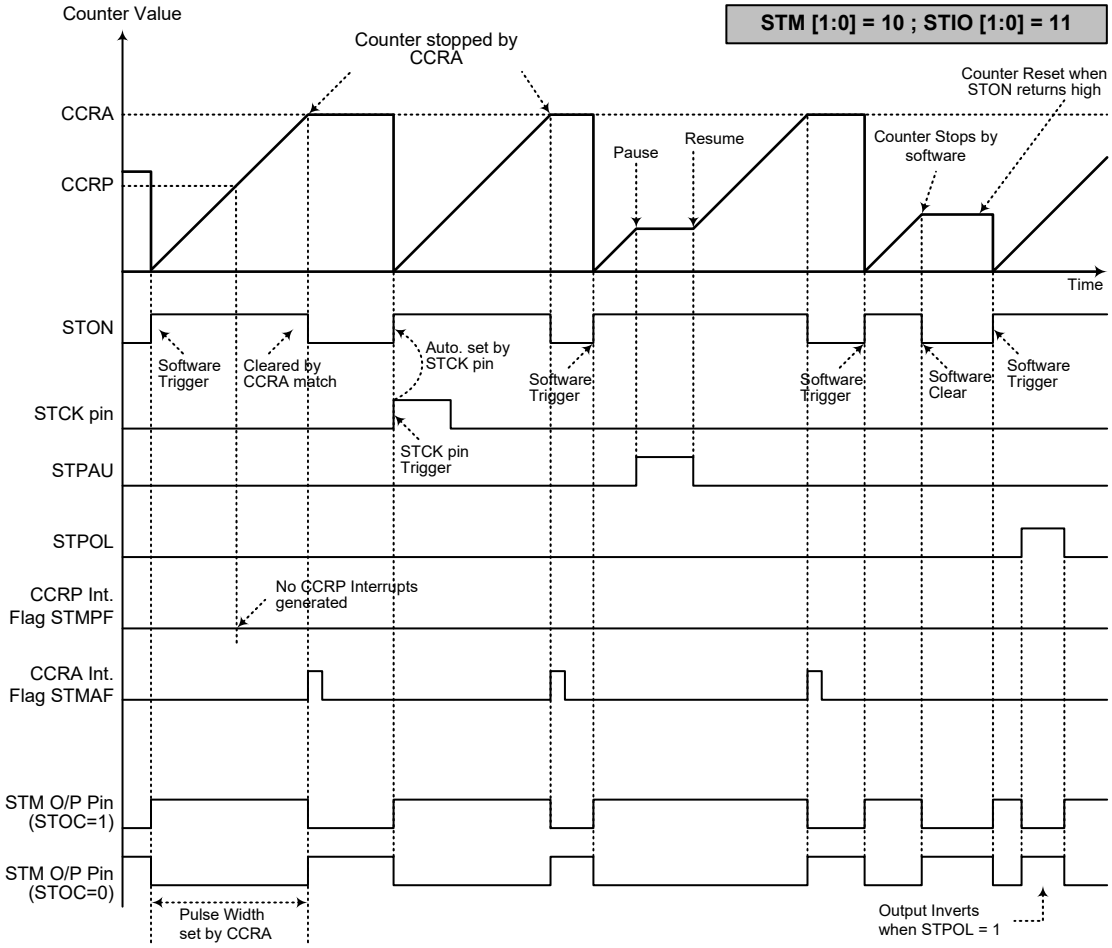
To select this mode, bits STM1 and STM0 in the STMC1 register should be set to 10 respectively and also the STIO1 and STIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the TM output pin.

The trigger for the pulse output leading edge is a low to high transition of the STON bit, which can be implemented using the application program. However in the Single Pulse Mode, the STON bit can also be made to automatically change from low to high using the external STCK pin, which will in turn initiate the Single Pulse output.

When the STON bit transitions to a high level, the counter will start running and the pulse leading edge will be generated. The STON bit should remain high when the pulse is in its active state. The generated pulse trailing edge will be generated when the STON bit is cleared to zero, which can be implemented using the application program or when a compare match occurs from Comparator A.



Single Pulse Generation



Single Pulse Output Mode

Note: 1. Counter stopped by CCRA match

2. CCRP is not used

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

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

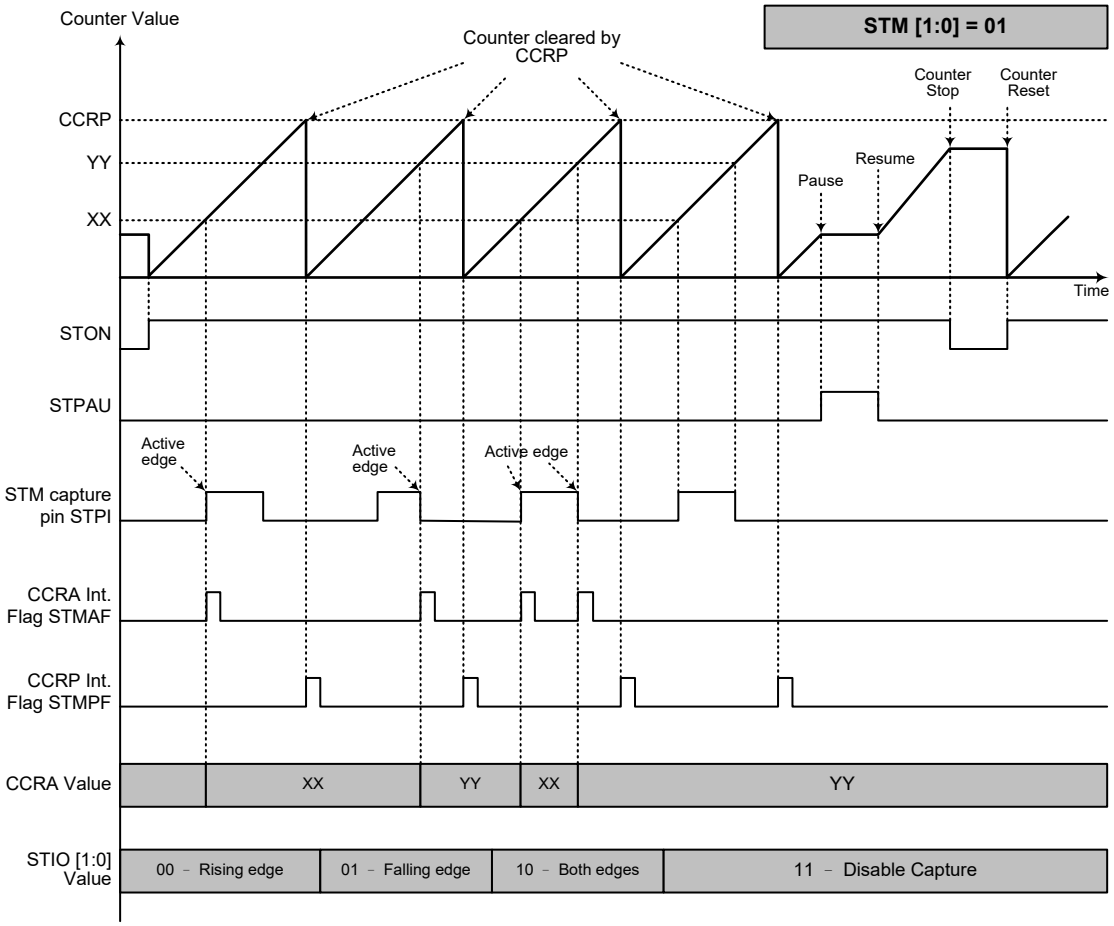
However a compare match from Comparator A will also automatically clear the STON bit and thus generate the Single Pulse output trailing edge. In this way the CCRA value can be used to control the pulse width. A compare match from Comparator A will also generate a TM interrupt. The counter can only be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

Capture Input Mode

To select this mode bits STM1 and STM0 in the STMC1 register should be set to 01 respectively. This mode enables external signals to capture and store the present value of the internal counter and can therefore be used for applications such as pulse width measurements. The external signal is supplied on the STP pin, whose active edge can be either a rising edge, a falling edge or both rising and falling edges; the active edge transition type is selected using the STIO1 and STIO0 bits in the STMC1 register. The counter is started when the STON bit changes from low to high which is initiated using the application program.

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

There are some considerations that should be noted. If STCK is used as the capture input source, then it cannot be selected as the STM clock source. If the captured pulse width is less than 2 timer clock periods, it may be ignored by hardware. After the counter value is latched to CCRA or CCRB registers by an active capture edge, the STMAF flag will be set high after 0.5 timer clock period. The delay time from the active capture edge received to the action of latching counter value to CCRA or CCRB registers is less than 1.5 timer clock periods. The STCCLR and STDPX bits are not used in this Mode.



Capture Input Mode

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

2. A TM Capture input pin active edge transfers the counter value to CCRA
3. The STCCLR bit is not used
4. No output function - STOC and STPOL bits are not used
5. CCRP determines the counter value and the counter has a maximum count value when CCRP is equal to zero.

Periodic Type TM – PTM

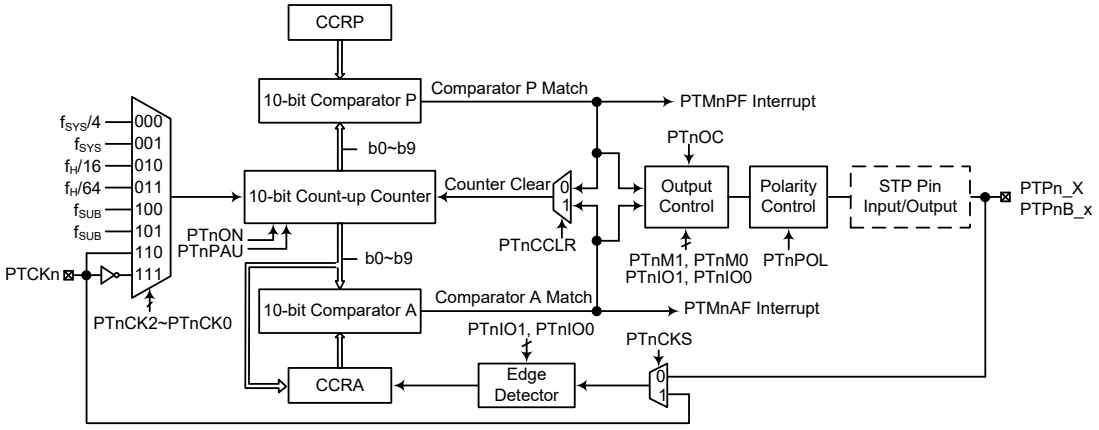
The PTM contains five operating modes, which are Compare Match Output, Timer/Event Counter, Capture Input, Single Pulse Output and PWM Output modes. The P-type TM can also be controlled with an external input pin and can drive two or five output pins.

Device	TM Type	TM Input Pin	TM Output Pin
HT69F340	10-bit PTM	PTCK	PTP_0, PTP_1, PTP_2 PTPB_0, PTPB_1
HT69F350	10-bit PTM	PTCK	PTP_0, PTP_1 PTPB_0, PTPB_1
HT69F360/ HT67F370	10-bit PTM0 10-bit PTM1	PTCK0 PTCK1	PTP0, PTP0B PTP1, PTP1B

Periodic TM Operation

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

The only way of changing the value of the 10 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 TM 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 an output pin. All operating setup conditions are selected using relevant internal registers.



Periodic Type TM Block Diagram (n is only for HT69F360/HT67F370)

PTM register description

Overall operation of the P-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/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	PTnCKS	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 Register List

- **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 TM will remain powered up and continue to consume power. The counter will retain its residual value when this bit changes from low to high and resume counting from this value when the bit changes to a low value again.

Bit 6~4 **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 TM. 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 TM. Setting the bit high enables the counter to run, clearing the bit disables the TM. Clearing this bit to zero will stop the counter from counting and turn off the TM which will reduce its power consumption. When the bit changes state from low to high the internal counter value will be reset to zero, however when the bit changes from high to low, the internal counter will retain its residual value until the bit returns high again. If the TM is in the Compare Match Output Mode then the TM output pin will be reset to its initial condition, as specified by the 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	PTnCKS	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 Mode or Single Pulse Output Mode
11: Timer/Counter Mode

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

Bit 5~4 **PTnIO1~PTnIO0**: Select PTPn External Pins Function

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

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

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

Timer/counter Mode
Unused

These two bits are used to determine how the TM external pin changes state when a certain condition is reached. The function that these bits select depends upon in which mode the TM is running. In the Compare Match Output Mode, the PTnIO1 and PTnIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the 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 TM output pin when a compare match occurs. After the TM output pin changes state, it can be reset to its initial level by changing the level of the PTnON bit from low to high.

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

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

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

This is the output control bit for the TM output pin. Its operation depends upon whether TM is being used in the Compare Match Output Mode or in the PWM Mode/ Single Pulse Output Mode. It has no effect if the TM is in the Timer/Counter Mode. In

the Compare Match Output Mode it determines the logic level of the TM output pin before a compare match occurs. In the PWM Mode it determines if the PWM signal is active high or active low.

Bit 2 **PTnPOL**: 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 TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in theTimer/Counter Mode.

Bit 1 **PTnCKS**: Input Capture trigger source selection
0: External Clock source of Capture Input Mode comes from PTPn
1: External Clock source of Capture Input Mode comes from PTCKn

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 P-type 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, Single Pulse or Input Capture 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/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**: 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/W	R/W
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 (n=0, 1)**

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**: PTMn CCRP 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/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

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

Periodic Type TM Operating Modes

The P-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.

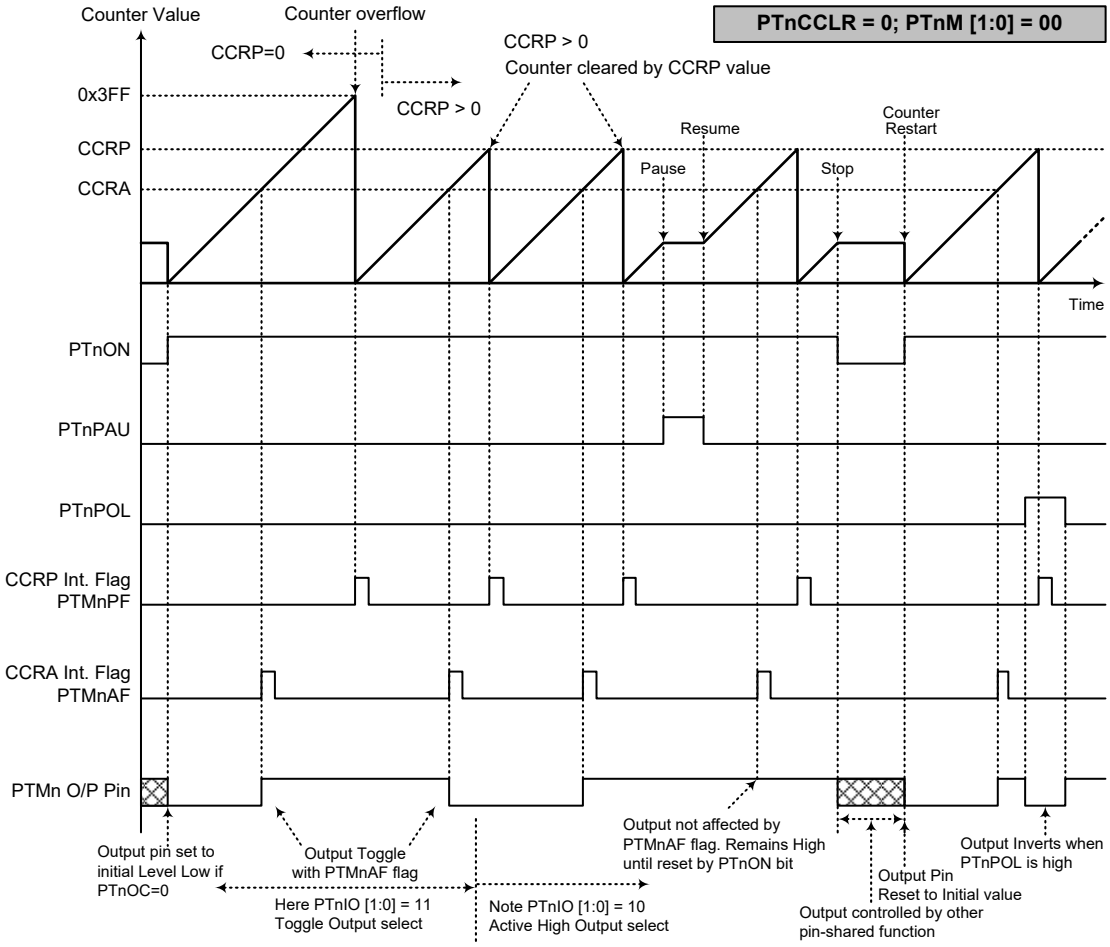
Compare Output Mode

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 set to "0".

As the name of the mode suggests, after a comparison is made, the TM output pin, will change state. The TM output pin condition however only changes state when 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 TM

output pin. The way in which the TM output pin changes state are determined by the condition of the PTnIO1 and PTnIO0 bits in the PTMnC1 register. The TM 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 TM 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.

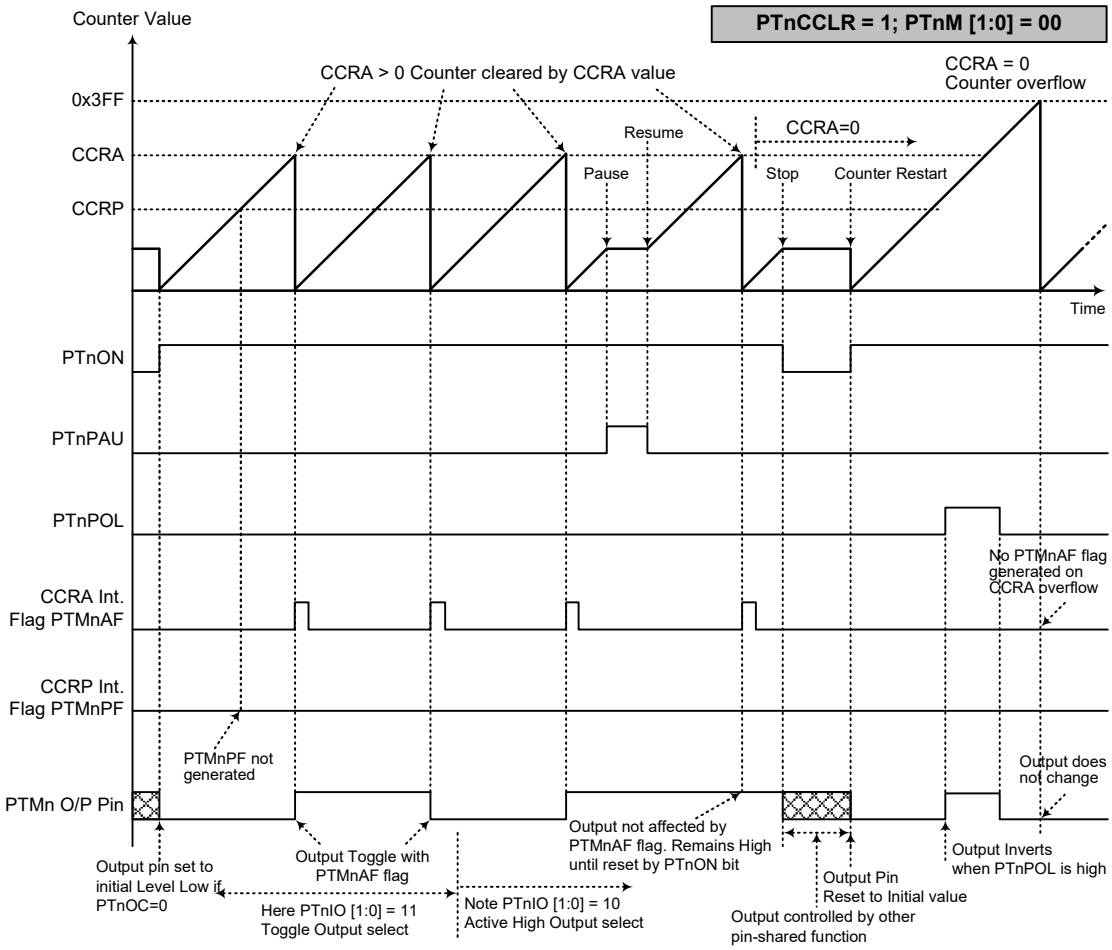


Compare Match Output Mode – PTnCCLR=0 (n=0, 1)

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

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

3. The output pin is reset to itsinitial state by a PTnON bit rising edge



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

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

2. The TM 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 and also the PTnIO1 and PTnIO0 bits should be set to 10 respectively. The PWM function within the TM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the TM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM mode, the PTnCCLR 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. 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 TM output pin to a fixed high or low level. The PTnPOL bit is used to reverse the polarity of the PWM output waveform.

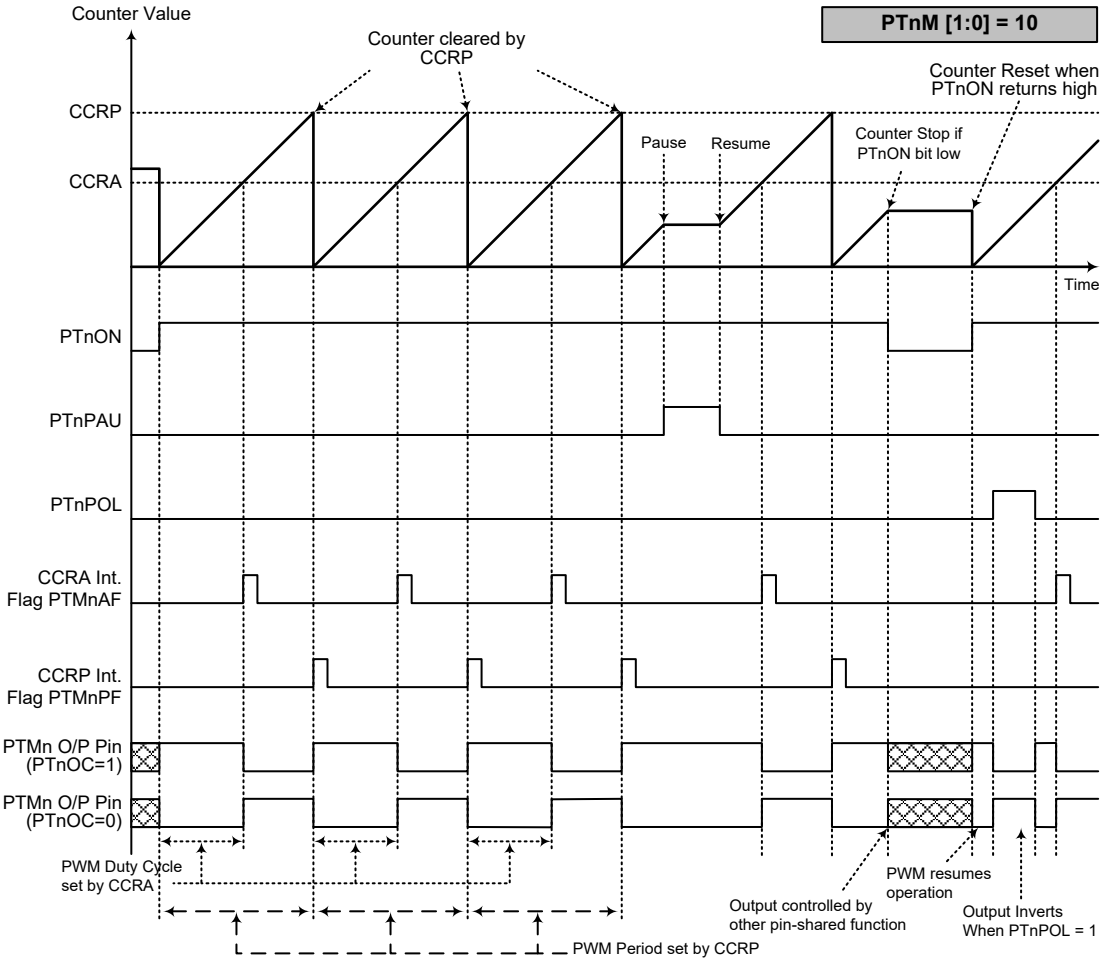
- **10-bit PWM Mode, Edge-aligned Mode**

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

If f_{SYS}=16MHz, TM clock source select $f_{SYS}/4$, CCRP=512 and CCRA=128,

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

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



PWM Output Mode (n=0, 1)

Note: 1. A counter clear sets the PWM Period

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

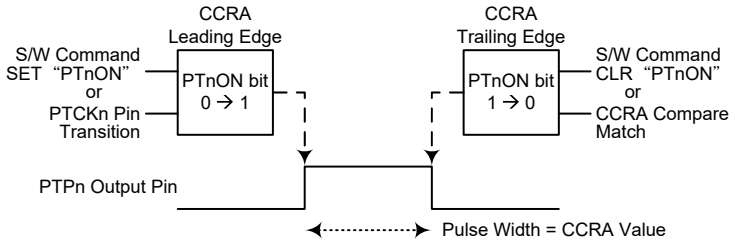
3. The PTnCCLR bit has no influence on PWM operation

Single Pulse Mode

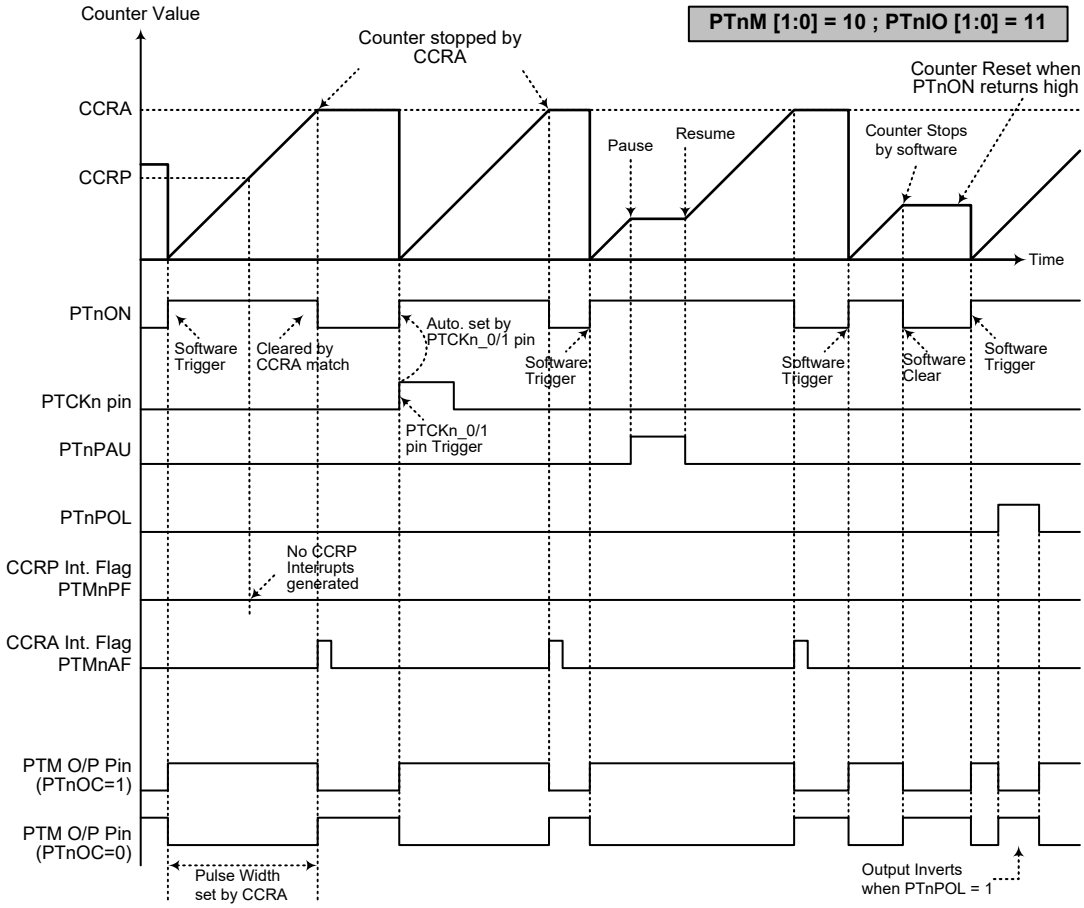
To select this mode, bits PTnM1 and PTnM0 in the PTMnC1 register should be set to 10 respectively and also the PTnIO1 and PTnIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the TM output pin.

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 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 TM 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 Mode CCRP is not used. The PTnCCLR bit is not used in this Mode.



Single Pulse Generation (n=0, 1)



Single Pulse Mode (n=0, 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 hight

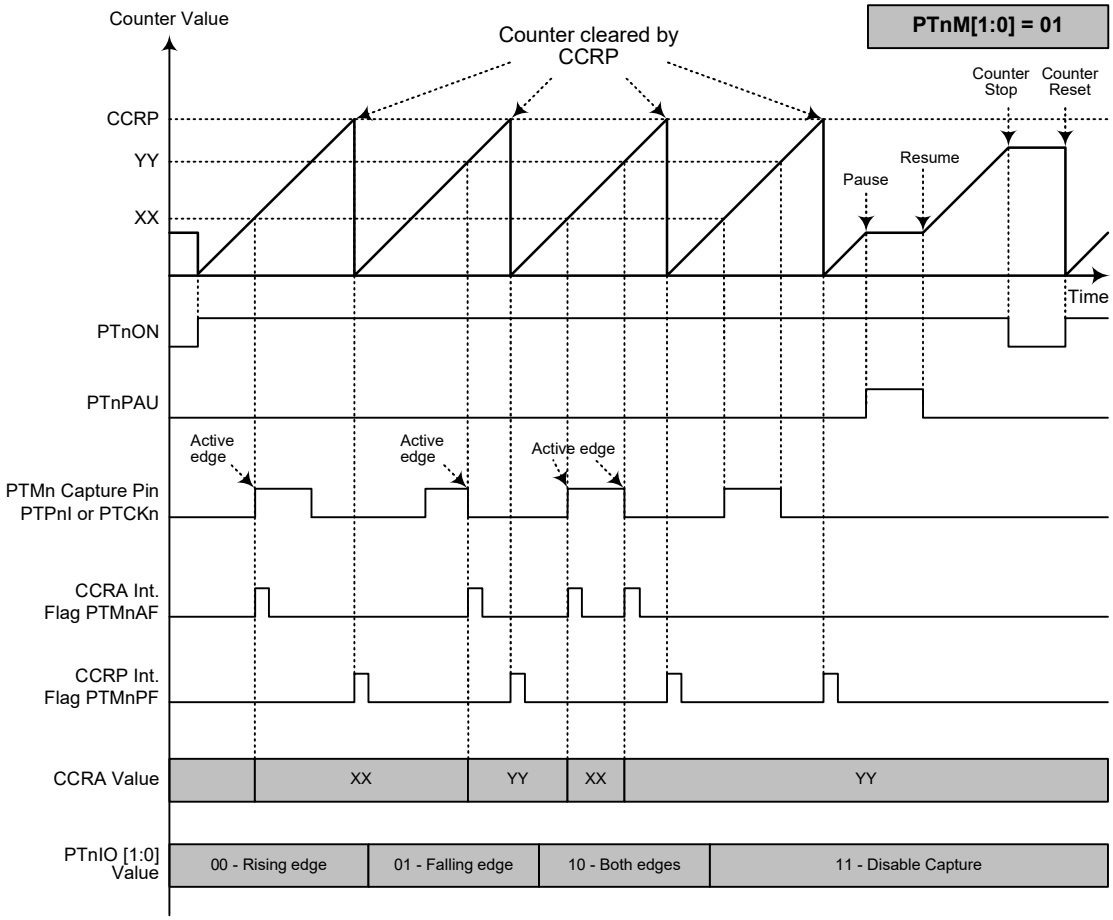
5. In the Single Pulse Mode, PTnIO [1:0] must be set to “11” and can not be changed.

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 PTPn or PTCKn pin, selected by the PTnCKS bit in the PTMnC1 register. The input pin 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 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 PTPn or PTCKn pin the present value in the counter will be latched into the CCRA registers and a TM interrupt generated. Irrespective of what events occur on the PTPn or 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 TM interrupt will also be generated. Counting the number of overflow interrupt signals from the CCRP can be a useful method in measuring long pulse widths. The PTnIO1 and PTnIO0 bits can select the active trigger edge on the PTPn or 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 PTPn or PTCKn pin, however it must be noted that the counter will continue to run.

There are some considerations that should be noted. If PTCKn is used as the capture input source, then it cannot be selected as the PTMn clock source. If the captured pulse width is less than 2 timer clock periods, it may be ignored by hardware. After the counter value is latched to CCRA or CCRB registers by an active capture edge, the PTMnAF flag will be set high after 0.5 timer clock period. The delay time from the active capture edge received to the action of latching counter value to CCRA or CCRB registers is less than 1.5 timer clock periods. The PTnCCLR, PTnOC and PTnPOL bits are not used in this Mode.



Capture Input Mode (n=0, 1)

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

2. A TM 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 – HT67F370

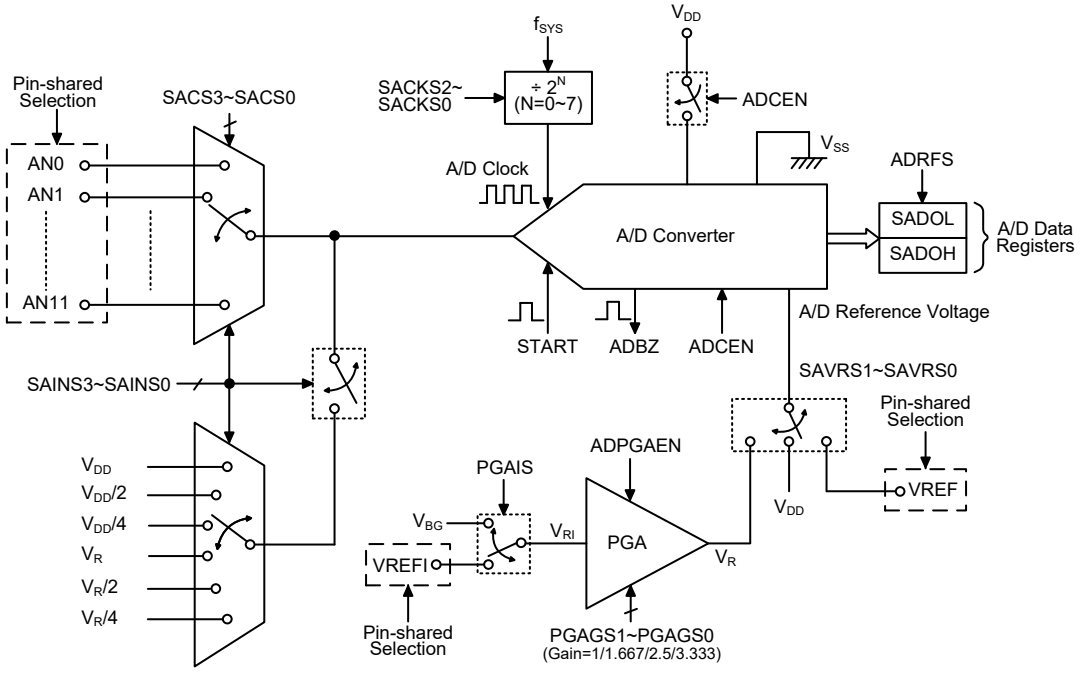
The need to interface to real world analog signals is a common requirement for many electronic systems. However, to properly process these signals by a microcontroller, they must first be converted into digital signals by A/D converters. By integrating the A/D conversion electronic circuitry into the 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

The HT67F370 device contains a multi-channel analog to digital converter which can directly interface to external analog signals, such as that from sensors or other control signals and convert these signals directly into a 12-bit digital value. The external or internal analog signal to be converted is determined by the SAINS3~SAINS0 bits together with the SACS3~SACS0 bits. When the external analog signal is to be converted, the corresponding pin-shared control bits should first be properly configured and then desired external channel input should be selected using the SAINS3~SAINS0 and SACS3~SACS0 bits. Note that when the internal analog signal is to be converted, the external channel input signal will automatically be disconnected regardless of the SACS bit field value. 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 respectively.

External Input Channels	Internal Signal	A/D Channel Select Bits
12: AN0~AN11	6: V_{DD}, $V_{DD}/2$, $V_{DD}/4$, V_R, $V_R/2$, $V_R/4$	SAINS3~SAINS0, SACS3~SACS0

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

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

A/D Converter Register List

A/D Converter Data Registers – SADOL, SADOH

As this device contains 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 Converter Data Registers

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

To control the function and operation of the A/D converter, three control registers known as SADC0~SADC2 are provided. These 8-bit registers define functions such as the selection of which analog channel is connected to the internal A/D converter, the digitised data format, the A/D clock source as well as controlling the start function and monitoring the A/D converter busy status. As the device contains 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 SAINS3~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 A/D converter also contains a programmable gain amplifier, PGA, to generate the A/D converter internal reference voltage. The overall operation of the PGA is controlled using the SADC2 register.

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**

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 the device power consumption. When the A/D converter function is disabled, the contents of the A/D data register pair known as SADOH and SADOL will be unchanged.

Bit 4 **ADRFS**: A/D 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
:
:
1010: AN10
1011: AN11
1100~1111: Non-existed channel, the input will be floating if selected

• **SADC1 Register**

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

Bit 7~4 **SAINS3~SAINS0**: A/D converter input signal select
0000: External input – External analog channel input
0001: Internal input – Internal A/D converter power supply voltage V_{DD}
0010: Internal input – Internal A/D converter power supply voltage $V_{DD}/2$
0011: Internal input – Internal A/D converter power supply voltage $V_{DD}/4$
0100: External input – External analog channel input
0101: Internal input – Internal A/D converter PGA output voltage V_R
0110: Internal input – Internal A/D converter PGA output voltage $V_R/2$
0111: Internal input – Internal A/D converter PGA output voltage $V_R/4$

1000~1011: Reserved, connected to ground
1100~1111: External input – External analog channel input

When the internal analog signal and the external signal are selected to be converted simultaneously, the external channel input signal will automatically be switched off regardless of the SACS3~SACS0 bit field value. It will prevent the external channel input from being connected together with the internal analog signal.

Bit 3 Unimplemented, read as "0"

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.

• SADC2 Register

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

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

When the PGA output V_R is selected as A/D converter input or A/D converter reference voltage, the PGA needs to be enabled by setting this bit high. Otherwise the PGA needs to be disabled by clearing this bit to zero to conserve the power.

Bit 6~5 Unimplemented, read as "0"

Bit 4 **PGAIS**: PGA input (V_{RI}) select
0: External VREFI pin
1: Internal independent reference voltage, V_{BG}

When the external voltage on VREFI pin and the internal independent reference voltage V_{BG} are selected as the PGA input simultaneously, the hardware will only choose the internal voltage V_{BG} as the PGA input.

Bit 3~2 **SAVRS1~SAVRS0**: A/D converter reference voltage select
00: Internal A/D converter power, V_{DD}
01: VREF pin
1x: Internal PGA output voltage, V_R

These bits are used to select the A/D converter reference voltage. When the internal A/D converter power or the internal PGA output voltage and the external input voltage on VREF pin are selected as the reference voltage simultaneously, the hardware will only choose the internal reference voltage as the A/D converter reference voltage.

Bit 1~0 **PGAGS1~PGAGS0**: PGA gain select
00: Gain=1
01: Gain=1.667, V_R=2V for $V_{RI}=V_{BG}$ (PGAIS=1)
10: Gain=2.5, V_R=3V for $V_{RI}=V_{BG}$ (PGAIS=1)
11: Gain=3.333, V_R=4V for $V_{RI}=V_{BG}$ (PGAIS=1)

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 2μs to 10μs for $1.8V \le V_{DD} < 2.0V$, or from 0.5μs to 10μs for $2.0V \le V_{DD} \le 5.5V$, 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 for $2.0V \le V_{DD} \le 5.5V$ and 000~011 or 111 for $1.8V \le V_{DD} < 2.0V$. Doing so will give A/D clock periods that are less than the minimum or larger than the maximum 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 * or # show where, depending upon the device, special care must be taken, as the values may be exceeding the specified A/D Clock Period range.

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 *#
16MHz	62.5ns *#	125ns *#	250ns *#	500ns #	1μs #	2μs	4μs	8μs

Note: # for $1.8V \le V_{DD} < 2.0V$ while * for $2.0V \le V_{DD} \le 5.5V$.

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, VDD, or from an external reference source supplied on pin VREF, or from the internal PGA output voltage, V_R. The desired selection is made using the SAVRS1 and SAVRS0 bits. When the SAVRS bit field is set to "00", the A/D converter reference voltage will come from the VDD pin. If the SAVRS bit field is set to "01", the A/D converter reference voltage will come from the VREF pin. Otherwise, the A/D converter reference voltage will come from the PGA output, V_R. As the VREF pin is pin-shared with other functions, when the VREF pin is selected as the reference voltage supply pin, the VREF pin-shared function control bits should be properly configured to disable other pin functions. In addition, if the program selects an external reference voltage on VREF pin and the internal reference voltage V_{DD} or V_R as the A/D converter reference voltage, then the hardware will only choose the internal reference voltage as the A/D converter reference voltage input. The analog input values must not be allowed to exceed the value of the selected reference voltage, V_{REF}.

The A/D converter also has a VREFI pin which is one of PGA inputs for A/D converter reference. To select this PGA input signal, the PGAIS bit in the SADC2 register must be cleared to zero and the revelent pin-shared control bits should be properly configured. However, the PGA input can be aslo supplied from the internal independent reference voltage, V_{BG}. If the application program selects the external voltage on the VREFI pin and an internal voltage V_{BG} as PGA input simultaneously, then the hardware will only choose the internal voltage V_{BG} as PGA input.

SAVRS[1:0]	Reference	Description
00	V_{DD}	Internal A/D converter power supply voltage
01	VREF pin	External A/D converter reference pin VREF
1x	V_R	Internal A/D converter PGA output voltage

A/D Converter Reference Voltage Selection

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 PxFS and SFSR1 registers 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.

If the SAINS3~SAINS0 bits are set to "0000", "0100", or "1100~1111", the external analog channel input is selected to be converted and the SACS3~SACS0 bits can determine which actual external channel is selected to be converted. If the SAINS3~SAINS0 bits are set to "0001~0011", the V_{DD} voltage with a specific ratio of 1, 1/2 or 1/4 is selected to be converted. If the SAINS3~SAINS0 bits are set to "0101~0111", the PGA output voltage with a specific ratio of 1, 1/2 or 1/4 is selected to be converted. Note that when the programs select external signal (AN0~AN11) and internal signal (V_{DD}, $V_{DD}/2$, $V_{DD}/4$, V_R, $V_R/2$ or $V_R/4$) as an A/D converter input signal simultaneously, then the hardware will only choose the internal signal as an A/D converter input, the external analog signal will be switched off automatically.

SAINS[3:0]	SACS[3:0]	Input Signals	Description
0000, 0100, 1100~1111	0000~1011	AN0~AN11	External pin analog input
	1100~1111	—	Non-existed channel, input is floating
0001	xxxx	V_{DD}	Internal A/D converter power supply voltage
0010	xxxx	$V_{DD}/2$	Internal A/D converter power supply voltage/2
0011	xxxx	$V_{DD}/4$	Internal A/D converter power supply voltage/4
0101	xxxx	V_R	Internal A/D converter PGA output voltage
0110	xxxx	$V_R/2$	Internal A/D converter PGA output voltage/2
0111	xxxx	$V_R/4$	Internal A/D converter PGA output voltage/4
1000~1011	xxxx	—	Reserved, connected to ground.

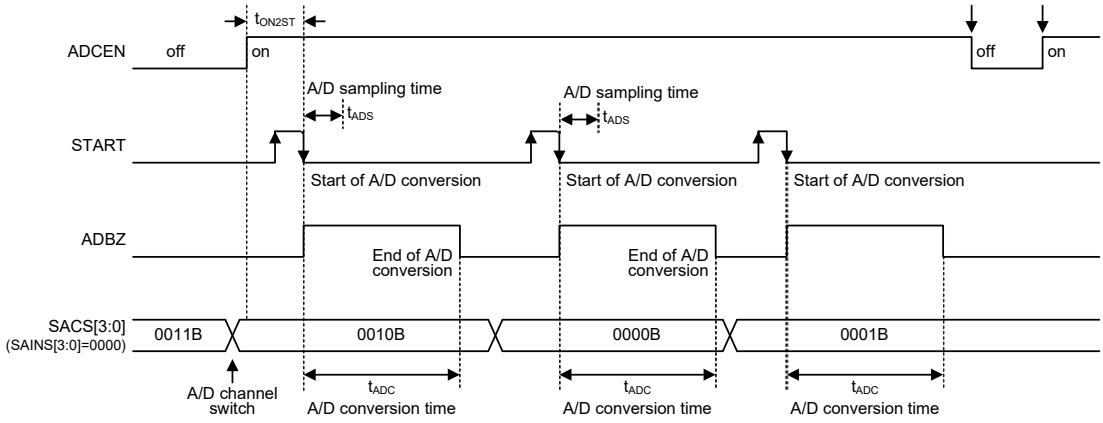
A/D Converter Input Signal Selection

Conversion Rate and Timing Diagram

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

Maximum single A/D conversion rate = 1/(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} where t_{ADCK} is equal to the A/D clock period.



A/D Conversion Timing – External Channel Input

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 SAINS3~SAINS0 bits in the SADC1 register.
  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 SAINS bit 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 SACS bit field. After this step, go to Step 6.
- Step 5
  If the A/D input signal comes from the internal analog signal, the SAINS bit field should be properly configured and then the external channel input will automatically be disconnected regardless of the SACS bit field value. After this step, go to Step 6.
- Step 6
  Select the reference voltage source by configuring the SAVRS1~SAVRS0 bits in the SADC2 register. If the PGA output voltage is selected, the PGA must be enabled and then select the PGA input source by configuring the PGAIS bit in the SADC2 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/Os, then care must be taken as if the input voltage is not at a valid logic level, then this may lead to some increase in power consumption.

A/D Conversion Function

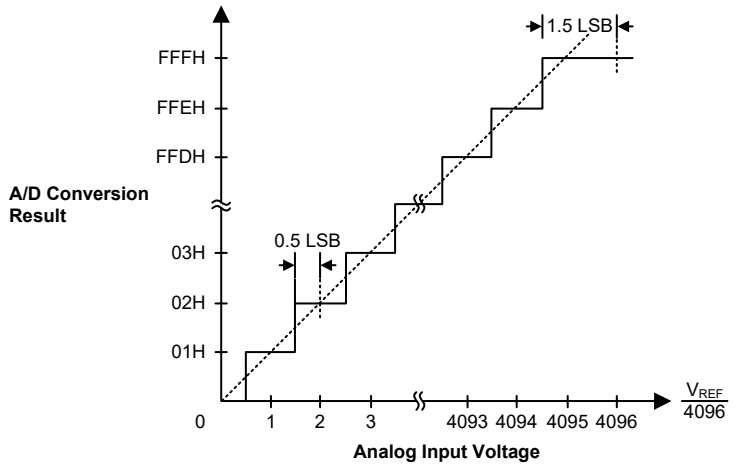
As the device contains a 12-bit A/D converter, its full-scale converted digitised value is equal to 0FFFH. Since the full-scale analog input value is equal to the actual A/D converter reference voltage, V_{REF}, this gives a single bit analog input value of V_{REF} divided by 4096.

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

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

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

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
set PB5FS
set PBFS5             ; setup [PB5FS:PBFS5] to configure pin AN0
mov a,20h
mov SADC0,a
mov a,00h
mov SADC2,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
set PB5FS
set PBFS5             ; setup [PB5FS:PBFS5] to configure pin AN0
mov a,20h
mov SADC0,a
mov a,00h
mov SADC2,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

The series devices contain a Serial Interface Module, which includes both the four-line SPI interface or two-line I²C 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 I²C based hardware such as sensors, Flash or EEPROM memory, etc. The SIM interface pins are pin-shared with other I/O pins and therefore the SIM interface functional pins must first be selected using the corresponding pin-shared function selection bits. As both interface types share the same pins and registers, the choice of whether the SPI or I²C type is used is made using the SIM operating mode control bits, named SIM2~SIM0, in the SIMC0 register. These pull-high resistors of the SIM pin-shared I/O pins are selected using pull-high control registers when the SIM function is enabled and the corresponding pins are used as SIM input pins.

SPI Interface

The SPI interface is often used to communicate with external peripheral devices such as sensors, Flash Memory or EEPROM memory devices etc. Originally developed by Motorola, the four line SPI interface is a synchronous serial data interface that has a relatively simple communication protocol simplifying the programming requirements when communicating with external hardware devices.

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

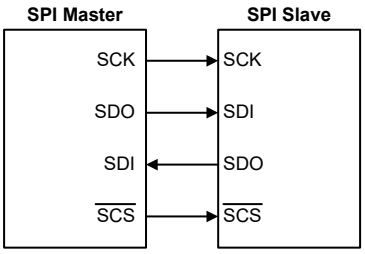
SPI Interface Operation

The SPI interface is a full duplex synchronous serial data link. It is a four line interface with pin names SDI, SDO, SCK and $\overline{SCS}$. Pins SDI and SDO are the Serial Data Input and Serial Data Output lines, SCK is the Serial Clock line and $\overline{SCS}$ is the Slave Select line. As the SPI interface pins are pin-shared with normal I/O pins and with the I²C function pins, the SPI interface pins must first be selected by configuring the pin-shared function selection bits and setting the correct bits in the SIMC0 and SIMC2 registers. After the desired SPI configuration has been set it can also be disabled or enabled using the SIMEN bit in the SIMC0 register. Communication between devices connected to the SPI interface is carried out in a slave/master mode with all data transfer initiations being implemented by the master. The Master also controls the clock signal. As the device only contains a single $\overline{SCS}$ pin only one slave device can be utilized. The $\overline{SCS}$ pin is controlled by software, set CSEN bit to "1" to enable $\overline{SCS}$ pin function, set CSEN bit to "0" the $\overline{SCS}$ pin will be floating state.

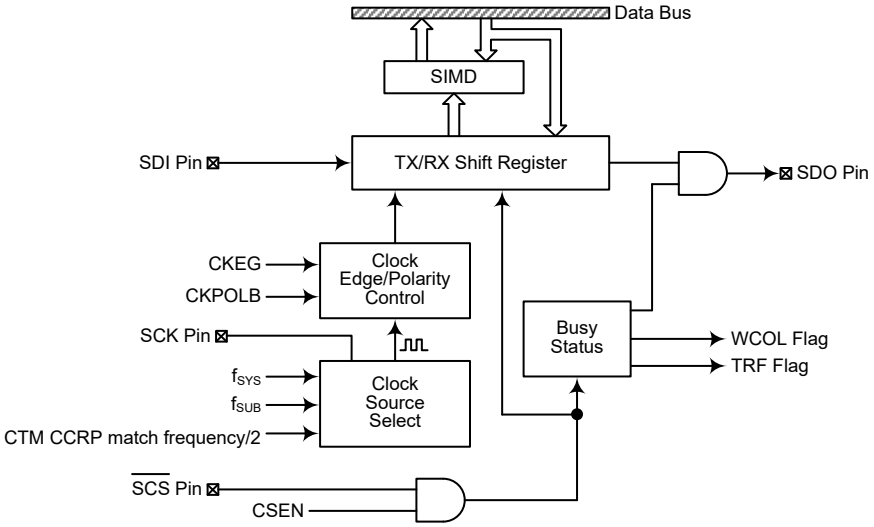
The SPI function in this device offers the following features:

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

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



SPI Master/Slave Connection



SPI Block Diagram

SPI Register

There are three internal registers which control the overall operation of the SPI interface. These are the SIMD data register and two registers SIMC0 and SIMC2.

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

SPI Register List

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

• **SIMD Register**

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

"x": unknown

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

• **SIMC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	—	SIMDEB1	SIMDEB0	SIMEN	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 CTM CCRP match frequency/2
101: SPI slave mode
110: I²C slave mode
111: Unused mode

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

Bit 4 Unimplemented, read as "0"

Bit 3~2 **SIMDEB1~SIMDEB0**: I²C Debounce Time Selection
00: No debounce
01: 2 system clocks debounce
1x: 4 system clocks debounce

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

The bit is the overall on/off control for the SIM interface. When the SIMEN bit is cleared to zero to disable the SIM interface, the SDI, SDO, SCK and $\overline{SCS}$, or SDA and SCL lines will lose their SPI or 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 Incomplete Flag
0: SIM incomplete is not occurred
1: SIM incomplete is occurred
The SIMICF bit is determined by $\overline{SCS}$ pin. When $\overline{SCS}$ pin is set to "1", it will clear the SPI counter. Meanwhile, the interrupt is occurred, if slave device didn't complete data received, then the incomplete flag, SIMICF, is set to "1".

- **SIMC2 Register**

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

Bit 7~6 Undefined bits
These bits can be read or written by user software program.

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

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

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

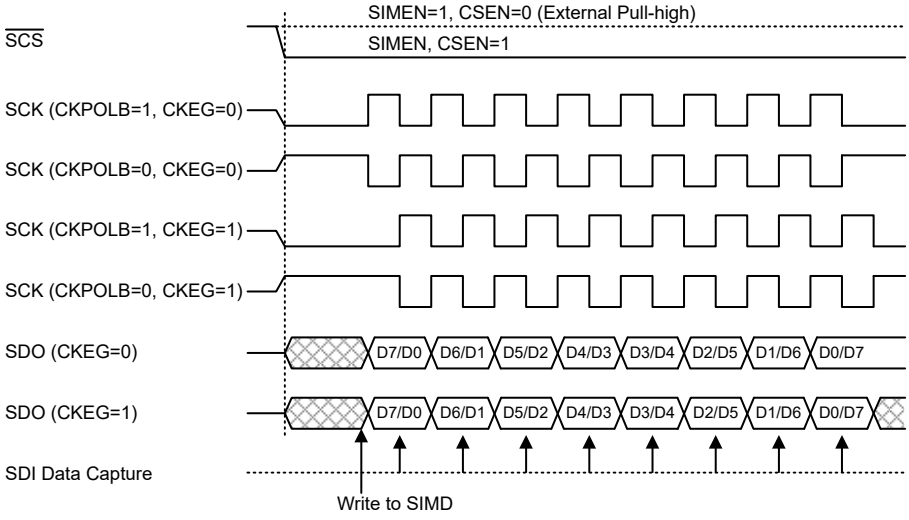
Bit 2 **CSEN**: SPI $\overline{SCS}$ pin Control
0: Disable
1: Enable
The CSEN bit is used as an enable/disable for the $\overline{SCS}$ pin. If this bit is low, then the $\overline{SCS}$ pin will be disabled and placed into I/O pin or the other functions. If the bit is high the $\overline{SCS}$ pin will be enabled and used as a select pin.

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

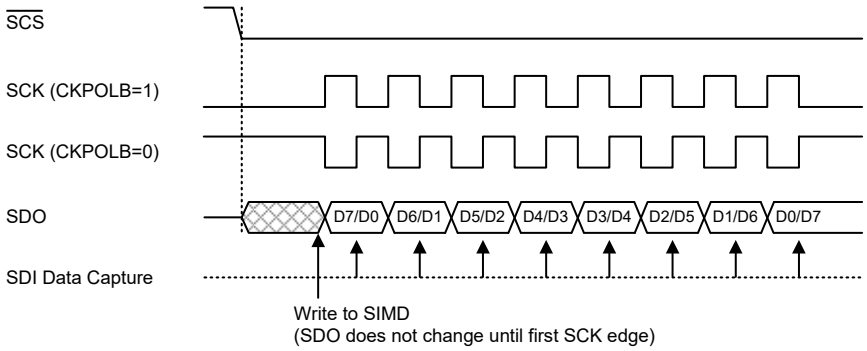
Bit 0 **TRF**: SPI Transmit/Receive Complete flag
0: Data is being transferred
1: SPI data transmission is completed
The TRF bit is the Transmit/Receive Complete flag and is set "1" automatically when an SPI data transmission is completed, but must set to "0" by the application program. It can be used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

SPI Communication

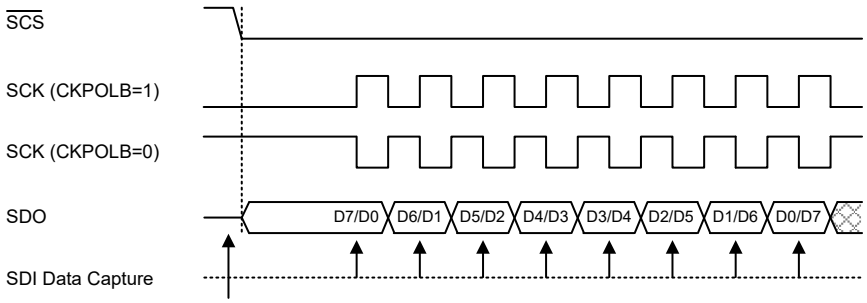
After the SPI interface is enabled by setting the SIMEN bit high, then in the Master Mode, when data is written to the SIMD register, transmission/reception will begin simultaneously. When the data transfer is complete, the TRF flag will be set automatically, but must be cleared using the application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



SPI Master Mode Timing

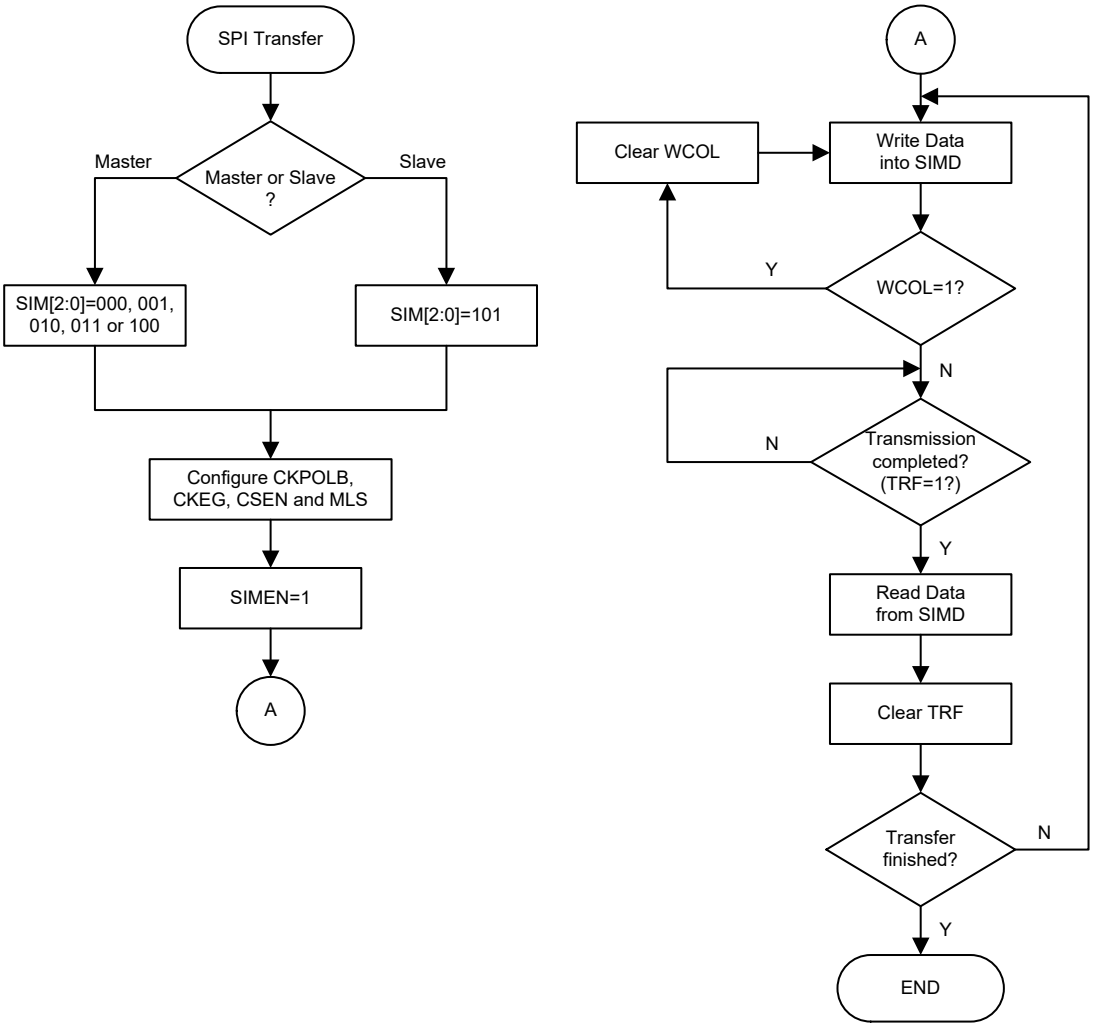


SPI Slave Mode Timing – CKEG=0



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

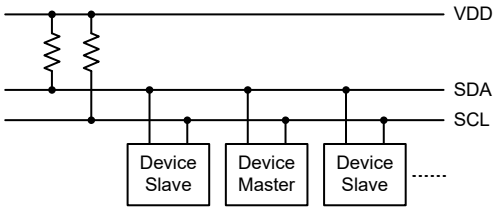
SPI Slave Mode Timing – CKEG=1



SPI Transfer Control Flowchart

I²C Interface

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

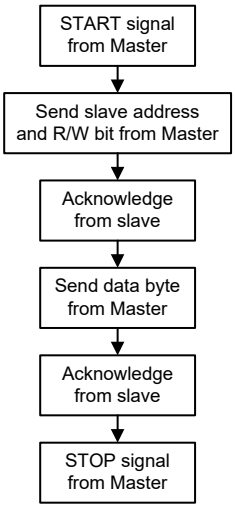


I²C Master Slave Bus Connection

I²C Interface Operation

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

When two devices communicate with each other on the bidirectional I²C bus, one is known as the master device and one as the slave device. Both master and slave can transmit and receive data, however, it is the master device that has overall control of the bus. For these devices, which only operates in slave mode, there are two methods of transferring data on the I²C bus, the slave transmit mode and the slave receive mode. The pull-high control function pin-shared with SCL/SDA pin is still applicable even if I²C device is activated and the related internal pull-high register could be controlled by its corresponding pull-high control register.



I²C Register

There are four control registers associated with the I²C bus, SIMC0, SIMC1, SIMA and SIMTOC and one data register, SIMD. The SIMD register, which is shown in the above SPI section, is used to store the data being transmitted and received on the I²C bus. Before the microcontroller writes data

to the I²C bus, the actual data to be transmitted must be placed in the SIMD register. After the data is received from the I²C bus, the microcontroller can read it from the SIMD register. Any transmission or reception of data from the I²C bus must be made via the SIMD register.

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

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 Register List

- **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 TM0 CCRP match frequency/2
101: SPI slave mode
110: I²C slave mode
111: Unused mode

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

Bit 4 Unimplemented, read as "0"

Bit 3~2 **SIMDEB1~SIMDEB0**: I²C Debounce Time Selection
00: No debounce
01: 2 system clocks debounce
1x: 4 system clocks debounce

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

The bit is the overall on/off control for the SIM interface. When the SIMEN bit is cleared to zero to disable the SIM interface, the SDI, SDO, SCK and $\overline{SCS}$, or SDA and SCL lines will lose their SPI or 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 Incomplete Flag
0: SIM incomplete is not occurred
1: SIM incomplete is occurred
The SIMICF bit is determined by $\overline{SCS}$ pin. When $\overline{SCS}$ pin is set to "1", it will clear the SPI counter. Meanwhile, the interrupt is occurred, if slave device didn't complete data received, then the incomplete flag, SIMICF, is set to "1".

• SIMC1 Register

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

Bit 7 **HCF:** I²C Bus data transfer completion flag
0: Data is being transferred
1: Completion of an 8-bit data transfer
The HCF flag is the data transfer flag. This flag will be zero when data is being transferred. Upon completion of an 8-bit data transfer the flag will go high and an interrupt will be generated.

Bit 6 **HAAS:** I²C Bus address match flag
0: Not address match
1: Address match
The HAAS flag is the address match flag. This flag is used to determine if the slave device address is the same as the master transmit address. If the addresses match then this bit will be high, if there is no match then the flag will be low.

Bit 5 **HBB:** I²C Bus busy flag
0: I²C Bus is not busy
1: I²C Bus is busy
The HBB flag is the I²C busy flag. This flag will be "1" when the I²C bus is busy which will occur when a START signal is detected. The flag will be set to "0" when the bus is free which will occur when a STOP signal is detected.

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

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

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

Bit 1 **IAMWU:** I²C Address Match Wake-up Control
0: Disable
1: Enable – must be cleared by the application program after wake-up
This bit should be set to 1 to enable the I²C address match wake up from the SLEEP or IDLE Mode. If the IAMWU bit has been set before entering 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 do not receive acknowledge flag
The RXAK flag is the receiver acknowledge flag. When the RXAK flag is "0", it means that a acknowledge signal has been received at the 9th clock, after 8 bits of data have been transmitted. When the slave device in the transmit mode, the slave device checks the RXAK flag to determine if the master receiver wishes to receive the next byte. The slave transmitter will therefore continue sending out data until the RXAK flag is "1". When this occurs, the slave transmitter will release the SDA line to allow the master to send a STOP signal to release the I²C Bus.

The SIMD register is used to store the data being transmitted and received. The same register is used by both the SPI and I²C functions. Before the device writes data to the I²C bus, the actual data to be transmitted must be placed in the SIMD register. After the data is received from the I²C bus, the device 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

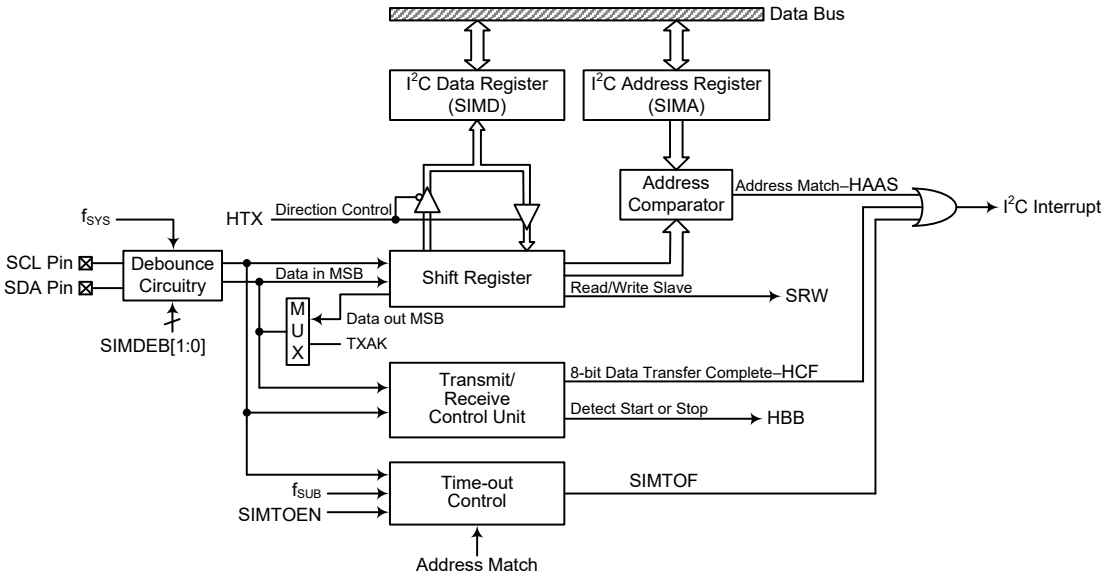
"x": unknown

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

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

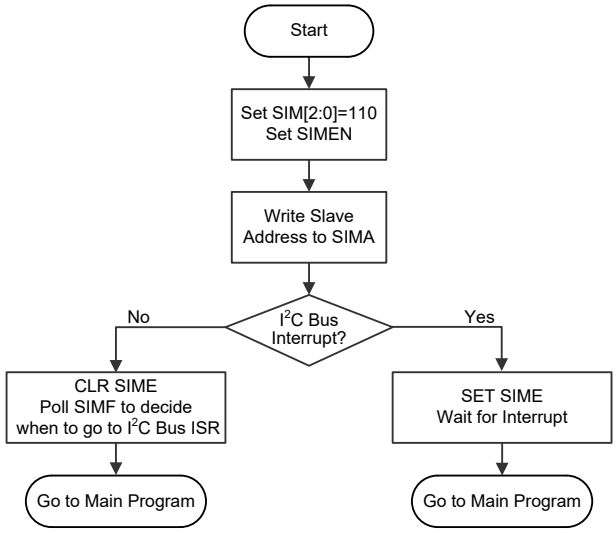


I²C Block Diagram

I²C Bus Communication

Communication on the I²C bus requires four separate steps, a START signal, a slave device address transmission, a data transmission and finally a STOP signal. When a START signal is placed on the I²C bus, all devices on the bus will receive this signal and be notified of the imminent arrival of data on the bus. The first seven bits of the data will be the slave address with the first bit being the MSB. If the address of the slave device matches that of the transmitted address, the HAAS bit in the SIMC1 register will be set and an I²C interrupt will be generated. After entering the interrupt service routine, the slave device must first check the condition of the HAAS bit to determine whether the interrupt source originates from an address match or from the completion of an 8-bit data transfer. During a data transfer, note that after the 7-bit slave address has been transmitted, the following bit, which is the 8th bit, is the read/write bit whose value will be placed in the SRW bit. This bit will be checked by the slave device to determine whether to go into transmit or receive mode. Before any transfer of data to or from the I²C bus, the microcontroller must initialize 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" to enable the I²C bus.
- Step 2
  Write the slave address of the device to the I²C bus address register SIMA.
- Step 3
  Set the SIME 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.

Slave Address

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

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

I²C Bus Read/Write Signal

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

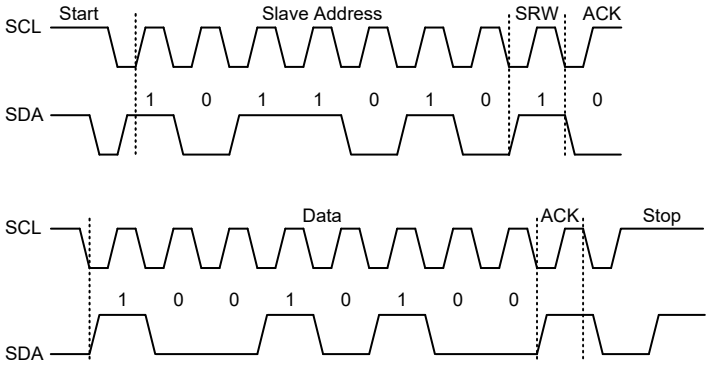
I²C Bus Slave Address Acknowledge Signal

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

I²C Bus Data and Acknowledge Signal

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

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

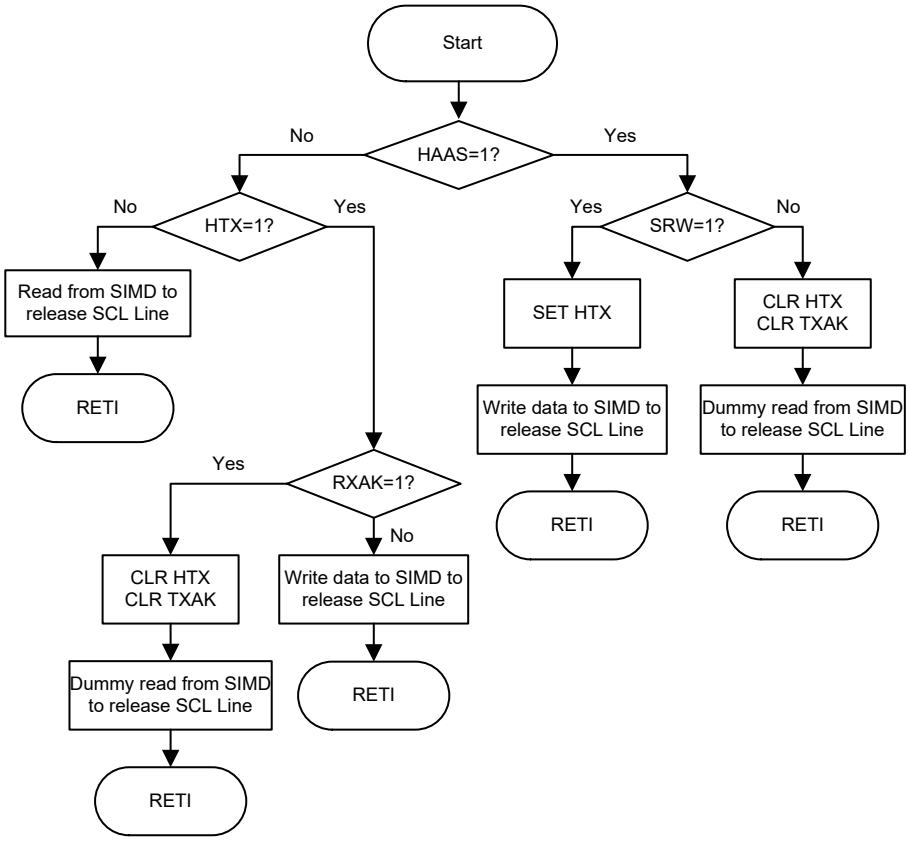


S=Start (1 bit)
SA=Slave Address (7 bits)
SR=SRW bit (1 bit)
M=Slave device send acknowledge bit (1 bit)
D=Data (8 bits)
A=ACK (RXAK bit for transmitter, TXAK bit for receiver, 1 bit)
P=Stop (1 bit)

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

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

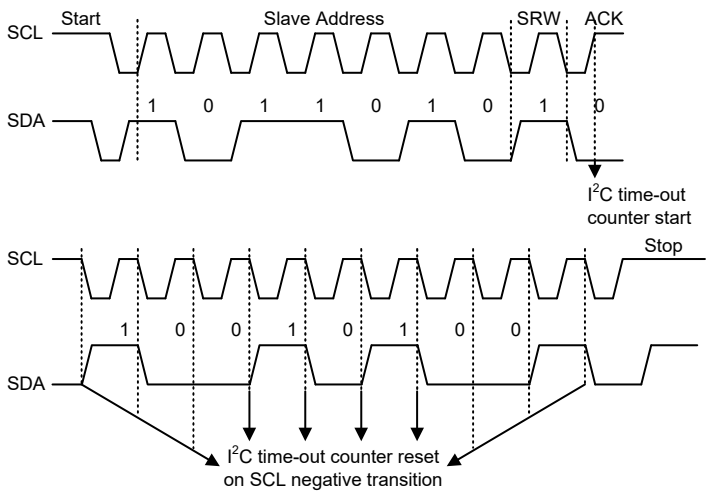
I²C Communication Timing Diagram



I²C Bus ISR Flow Chart

I²C Time-out Control

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

Register	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 period selections which can be selected using the SIMTOS bits in the SIMTOC register. The time-out duration is calculated 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**: I²C Time-out Control
0: Disable
1: Enable

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

Bit 5~0 **SIMTOS5~SIMTOS0**: I²C Time-out Time Selection
I²C Time-out clock source is f_{SUB}/32
I²C Time-out time is given by: (SIMTOS [5:0] +1) × (32/f_{SUB})

UART Interface – HT69F360/HT67F6370

The HT69F360/HT67F370 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, Universal Asynchronous Receiver and Transmitter (UART) 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)
- Transmitter and receiver enabled independently
- 2-byte Deep FIFO Receive Data Buffer
- RX pin Wake-up function
- Transmit and receive interrupts
- Interrupts can be initialized by the following conditions:
  - Transmitter Empty
  - Transmitter Idle
  - Receiver Full
  - Receiver Overrun
  - Address Mode Detect

UART External Interface

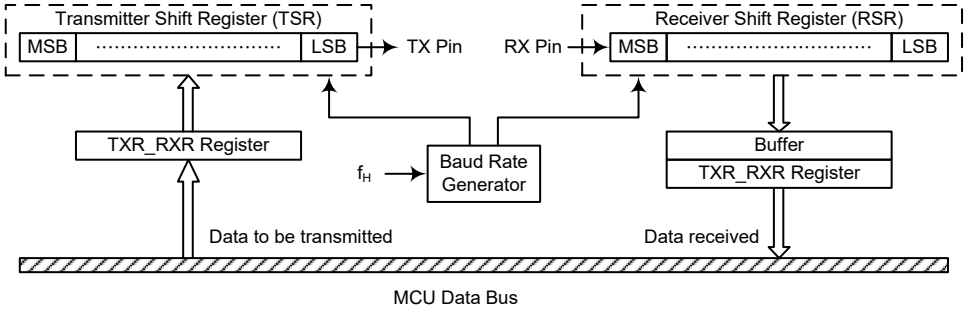
To communicate with an external serial interface, the internal UART has two external pins known as TX and RX,which are pin-shared with I/O or other pin functions. The TX and RX pin function should first be selected by the corresponding pin-shared function selection register before the UART function is used. Along with the UARTEN bit, the TXEN and RXEN bits, if set, will configure these pins to transmitter output and receiver input conditions. At this time the internal pull-high resistor related to the transmitter output pin will be disabled, while the internal pull-high resistor related to the receiver input pin is controlled by the corresponding I/O pull-high function control bit. When the TX or RX pin function is disabled by clearing the UARTEN, TXEN or RXEN bit, the TX or RX pin will be placed into 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 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 register by the application program. The data will then be transferred to the Transmit Shift Register from where it will be shifted out, LSB first, onto the TX pin at a rate controlled by the Baud Rate Generator. Only the TXR register is mapped onto the MCU Data Memory, the Transmit Shift Register is not mapped and is therefore inaccessible to the application program.

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

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



UART Data Transfer Scheme

UART Status and Control Registers

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

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	D7	D6	D5	D4	D3	D2	D1	D0
BRG	D7	D6	D5	D4	D3	D2	D1	D0

UART Register Summary

• 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 RXR data register.

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

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

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

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

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

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

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

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

Bit 2 **RXIF**: Receive RXR data register status
0: RXR data register is empty
1: RXR data register has available data

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

Bit 1 **TIDLE**: Transmission 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 to "1"

when the TXIF flag is "1" and when there is no transmit data or break character being transmitted. When TIDLE is equal to "1", the TX pin becomes idle with the pin state in logic high condition. The TIDLE flag is cleared by reading the USR register with TIDLE set and then writing to the TXR register. The flag is not generated when a data character or a break is queued and ready to be sent.

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

• UCR1 Register

The UCR1 register together with the UCR2 register are the 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 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 are used. If this bit is equal to "0", then only one stop bit 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 falling edge 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	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**: UART Transmit/Receive Data bit 7 ~ bit 0

• **BRG 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**: 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_{SYS}/[64×(N+1)] if BRGH=0.
Baud rate=f_{SYS}/[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 in 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_{SYS} / [64 (N+1)]	f_{SYS} / [16 (N+1)]

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

Calculating the Register and Error Values

For a clock frequency of 4MHz, and with BRGH set to "0" determine the BRG register value N, the actual baud rate and the error value for a desired baud rate of 4800.

From the above table the desired baud rate BR=f_{SYS} / [64 (N+1)]

Re-arranging this equation gives N=[f_{SYS} / (BR×64)] - 1

Giving a value for N=[4000000 / (4800×64)] - 1=12.0208

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

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

The following tables show actual values of baud rate and error values for the two values of BRGH.

Baud Rate K/BPS	Baud Rates for BRGH=0		
	f_{SYS}=16 MHz		
	BRG	Kbaud	Error (%)
0.3	—	—	—
1.2	207	1.202	0.16
2.4	103	2.404	0.16
4.8	51	4.808	0.16
9.6	25	9.615	0.16
19.2	12	19.231	0.16
38.4	6	35.714	-6.99
57.6	3	62.5	8.51
115.2	1	125	8.51
250	0	250	0

Baud Rates and Error Values for BRGH=0

Baud Rate K/BPS	Baud Rates for BRGH=1		
	f_{SYS}=16 MHz		
	BRG	Kbaud	Error (%)
0.3	—	—	—
1.2	—	—	—
2.4	—	—	—
4.8	207	4.808	0.16
9.6	103	9.615	0.16
19.2	51	19.231	0.16
38.4	25	38.462	0.16
57.6	16	58.824	2.12
115.2	8	111.11	-3.55
250	3	250	0

Baud Rates and Error Values for BRGH=1

UART Setup and Control

For data transfer, the UART function utilizes a non-return-to-zero, more commonly known as NRZ, format. This is composed of one start bit, eight or nine data bits, and one or two stop bits. Parity is supported by the UART hardware, and can be setup to be even, odd or no parity. For the most common data format, 8 data bits along with no parity and one stop bit, denoted as 8, N, 1, is used as the default setting, which is the setting at power-on. The number of data bits and stop bits, along with the parity, are setup by programming the corresponding BNO, PRT, PREN, and STOPS bits

in the UCR1 register. The baud rate used to transmit and receive data is setup using the internal 8-bit baud rate generator, while the data is transmitted and received LSB first. Although the 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

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. 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.

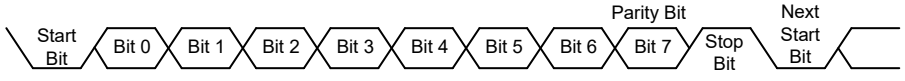
Data, Parity and Stop Bit Selection

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 identifies the frame as an address character. The number of stop bits, which can be either one or two, is independent of the data length.

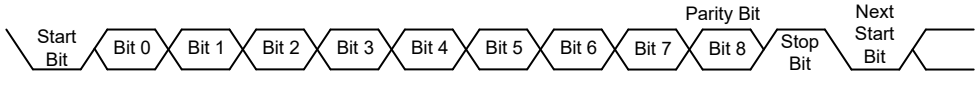
Start Bit	Data Bits	Address Bits	Parity Bits	Stop Bit
Example of 8-bit Data Formats				
1	8	0	0	1
1	7	0	1	1
1	7	1	0	1
Example of 9-bit Data Formats				
1	9	0	0	1
1	8	0	1	1
1	8	1	0	1

Transmitter Receiver Data Format

The following diagram shows the transmit and receive waveforms for both 8-bit and 9-bit data formats.



8-bit data format



9-bit data format

UART Transmitter

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

Transmitting Data

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

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

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

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

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

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

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

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

Transmit Break

If the TXBRK bit is set and the state keeps for a time greater than (BRDn+1)×t_H while TIDLEn=1, 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, 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 highs 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 RXR register forms a buffer between the internal bus and the receiver shift register. The 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 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, PREN and STOPS bits to define the word length, parity type and number of stop bits.
- Setup the BRG register to select the desired baud rate.
- Set the RXEN bit to ensure that the 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 RXR register has data available, at least one character can be read.
- When the contents of the shift register have been transferred to the 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. An 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 and STOPS bits. If the break is much longer than 13 bit times, the reception will be considered as complete after the number of bit times specified by BNO and STOPS. The RXIF bit is set, FERR is set, zeros are loaded into the receive data register, interrupts are generated if appropriate and the RIDLE bit is set. If a long break signal has been detected and the receiver has received a start bit, the data bits and the invalid stop bit, which sets the FERR flag, 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. A break is regarded as a character that contains only zeros with the FERR flag set. The break character will be loaded into the buffer and no further data will be received until stop bits are received. It should be noted that the RIDLE read only flag will go high when the stop bits have not yet been received. The reception of a break character on the UART registers will result in the following:

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

Idle Status

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

RIDLE flag will have a high value, which indicates the receiver is in an idle condition.

Receiver Interrupt

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

Managing Receiver Errors

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

Overrun Error – OERR Flag

The 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 the third byte has been entirely shifted in, the data should be read from the RXR register. If this is not done, the overrun error flag OERR will be consequently indicated.

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

- The OERR flag in the USR register will be set.
- The RXR contents will not be lost.

- The shift register will be overwritten.
- An interrupt will be generated if the RIE bit is set.

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

Noise Error – NF Flag

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

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

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

Framing Error – FERR Flag

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 is buffered along with the received data and is cleared on any reset.

Parity Error – PERR Flag

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 is buffered along with the received data bytes. It is cleared on any reset. It should be noted that the FERR and PERR flags are buffered along with the corresponding word and should be read before reading the data word.

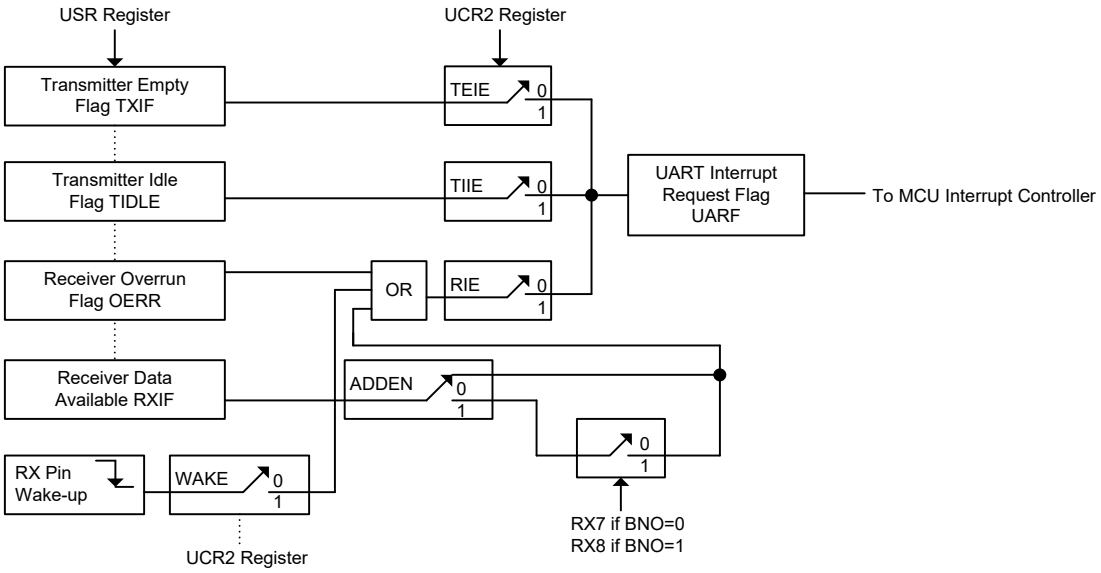
UART Interrupt Structure

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

The address detect condition, which is also a UART interrupt source, does not have an associated flag, but will generate a UART interrupt when an address detect condition occurs if its function is enabled by setting the ADDEN bit in the UCR2 register. An RX pin wake-up, which is also a UART interrupt source, does not have an associated flag, but will generate a UART interrupt if the microcontroller is woken up by a falling edge on the RX pin, if the WAKE and RIE bits in the UCR register are set. Note that in the event of an RX wake-up interrupt occurring, there will be a certain period of delay, commonly known as the System Start-up Time, for the oscillator to restart and stabilize before the system resumes normal operation.

Note that the USR register flags are read only and cannot be cleared or set by the application program, neither will they be cleared when the program jumps to the corresponding interrupt

servicing routine, as is the case for some of the other interrupts. The flags will be cleared automatically when certain actions are taken by the UART, the details of which are given in the UART register section. The overall UART interrupt can be disabled or enabled by the related interrupt enable control bits in the interrupt control registers of the microcontroller to decide whether the interrupt requested by the UART module is masked out or allowed.



UART Interrupt Scheme

Address Detect Mode

Setting the Address Detect 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 MFE, URE 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 to zero.

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

ADDEN Bit Function

UART Power Down and Wake-up

When the f_{SYS} is off, the UART will cease to function. All clock sources to the module are shutdown. If the f_{SYS} 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 before the MCU enters the Power Down Mode, then a falling edge on the RX pin will wake up the MCU from the Power Down Mode. Note that as it takes certain system clock cycles after a wake-up, before normal microcontroller operation resumes, any data received during this time on the RX pin will be ignored.

For a UART wake-up interrupt to occur, in addition to the bits for the wake-up being set, the global interrupt enable bit, EMI, and the UART interrupt enable bit, UARE, must also be set. If these two 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.

Interrupts

Interrupts are an important part of any microcontroller system. When an external event or an internal function such as a Timer Module 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 interrupt is generated by the action of the external INT0~INT1 pins, while the internal interrupts are generated by various internal functions such as the TMs, LVD, EEPROM, SIM, UART, A/D converter and Time Base.

Interrupt Registers

Overall interrupt control, which basically means the setting of request flags when certain microcontroller conditions occur and the setting of interrupt enable bits by the application program, is controlled by a series of registers, located in the Special Purpose Data Memory, as shown in the accompanying table. The number of registers depends upon the device chosen but fall into three categories. The first is the INTC0~INTC2 registers which setup the primary interrupts, the second is the MFI0~MFI2 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
Time Base	TBnE	TBnF	n=0~1
Multi-function	MFnE	MFnF	n=0~2
SIM	SIME	SIMF	—
UART (HT69F360/HT67F370)	URE	URF	—
A/D Converter (HT67F370)	ADE	ADF	—
LVD	LVE	LVF	—
EEPROM	DEE	DEF	—
CTM	CTMPE	CTMPF	—
	CTMAE	CTMAF	—
STM	STMPE	STMPF	—
	STMAE	STMAF	—
PTM (HT69F340/HT69F350)	PTMPE	PTMPF	—
	PTMAE	PTMAF	
PTM (HT69F360/HT67F370)	PTMnPE	PTMnPF	n=0~1
	PTMnAE	PTMnAF	

Interrupt Register Bit Naming Conventions

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	TB0F	INT1F	INT0F	TB0E	INT1E	INT0E	EMI
INTC1	MF2F	MF1F	MF0F	TB1F	MF2E	MF1E	MF0E	TB1E
INTC2	—	ADF	URF	SIMF	—	ADE	URE	SIME
MFI0	STMAF	STMPF	CTMAF	CTMPF	STMAE	STMPE	CTMAE	CTMPE
MFI1	PTM1AF	PTM1PF	PTM0AF	PTM0PF	PTM1AE	PTM1PE	PTM0AE	PTM0PE
MFI2	—	—	DEF	LVF	—	—	DEE	LVE

Interrupt Register List – HT67F370

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	TB0F	INT1F	INT0F	TB0E	INT1E	INT0E	EMI
INTC1	MF2F	MF1F	MF0F	TB1F	MF2E	MF1E	MF0E	TB1E
INTC2	—	—	—	SIMF	—	—	—	SIME
MFI0	—	—	CTMAF	CTMPF	—	—	CTMAE	CTMPE
MFI1	—	—	PTMAF	PTMPF	—	—	PTMAE	PTMPE
MFI2	—	—	DEF	LVF	—	—	DEE	LVE

Interrupt Register List – HT69F340

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	TB0F	INT1F	INT0F	TB0E	INT1E	INT0E	EMI
INTC1	MF2F	MF1F	MF0F	TB1F	MF2E	MF1E	MF0E	TB1E
INTC2	—	—	—	SIMF	—	—	—	SIME
MFI0	STMAF	STMPF	CTMAF	CTMPF	STMAE	STMPE	CTMAE	CTMPE
MFI1	—	—	PTMAF	PTMPF	—	—	PTMAE	PTMPE
MFI2	—	—	DEF	LVF	—	—	DEE	LVE

Interrupt Register List – HT69F350

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	TB0F	INT1F	INT0F	TB0E	INT1E	INT0E	EMI
INTC1	MF2F	MF1F	MF0F	TB1F	MF2E	MF1E	MF0E	TB1E
INTC2	—	—	UARTF	SIMF	—	—	UARTE	SIME
MFI0	STMAF	STMPF	CTMAF	CTMPF	STMAE	STMPE	CTMAE	CTMPE
MFI1	PTM1AF	PTM1PF	PTM0AF	PTM0PF	PTM1AE	PTM1PE	PTM0AE	PTM0PE
MFI2	—	—	DEF	LVF	—	—	DEE	LVE

Interrupt Register List – HT69F360

- **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 edge

- **INTC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	TB0F	INT1F	INT0F	TB0E	INT1E	INT0E	EMI
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **TB0F**: Time Base 0 interrupt request flag
0: No request
1: Interrupt request

Bit 5 **INT1F**: INT1 interrupt request flag
0: No request
1: Interrupt request

Bit 4 **INT0F**: INT0 interrupt request flag
0: No request
1: Interrupt request

Bit 3 **TB0E**: Time Base 0 interrupt control
0: Disable
1: Enable

Bit 2 **INT1E**: INT1 interrupt control
0: Disable
1: Enable

Bit 1 **INT0E**: INT0 interrupt control
0: Disable
1: Enable

Bit 0 **EMI**: Global interrupt control
0: Disable
1: Enable

• **INTC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	MF2F	MF1F	MF0F	TB1F	MF2E	MF1E	MF0E	TB1E
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 **MF2F**: Multi-function interrupt 2 request flag
0: No request
1: Interrupt request

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

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

Bit 4 **TB1F**: Time Base 1 interrupt request flag
0: No request
1: Interrupt request

Bit 3 **MF2E**: Multi-function interrupt 2 control
0: Disable
1: Enable

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

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

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

• **INTC2 Register-HT67F370**

Bit	7	6	5	4	3	2	1	0
Name	—	ADF	URF	SIMF	—	ADE	URE	SIME
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **ADF**: A/D Converter interrupt request flag
0: No request
1: Interrupt request

Bit 5 **URF**: UART interrupt request flag
0: No request
1: Interrupt request

Bit 4 **SIMF**: SIM interrupt request flag
0: No request
1: Interrupt request

Bit 3 Unimplemented, read as "0"

Bit 2 **ADE**: A/D Converter interrupt control
0: Disable
1: Enable

Bit 1 **URE**: UART interrupt control
0: Disable
1: Enable

Bit 0 **SIME**: SIM interrupt control
0: Disable
1: Enable

• **INTC2 Register – HT69F340/HT69F350**

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

Bit 7~5 Unimplemented, read as "0"

Bit 4 **SIMF**: Serial Interface Module request flag
0: No request
1: Interrupt request

Bit 3~1 Unimplemented, read as "0"

Bit 0 **SIME**: Serial Interface Module interrupt control
0: Disable
1: Enable

• **INTC2 Register – HT69F360**

Bit	7	6	5	4	3	2	1	0
Name	—	—	UARTF	SIMF	—	—	UARTE	SIME
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 request flag
0: No request
1: Interrupt request

Bit 4 **SIMF**: Serial Interface Module 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 **SIME**: Serial Interface Module interrupt control
0: Disable
1: Enable

• MFI0 Register – HT69F340

Bit	7	6	5	4	3	2	1	0
Name	—	—	CTMAF	CTMPF	—	—	CTMAE	CTMPE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5 **CTMAF**: CTM Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **CTMPF**: CTM Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 3~2 Unimplemented, read as "0"

Bit 1 **CTMAE**: CTM Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **CTMPE**: CTM Comparator P match interrupt control
0: Disable
1: Enable

• MFI0 Register – HT69F350/HT69F360/HT67F370

Bit	7	6	5	4	3	2	1	0
Name	STMAF	STMPF	CTMAF	CTMPF	STMAE	STMPE	CTMAE	CTMPE
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 **STMAF**: STM Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 6 **STMPF**: STM Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 5 **CTMAF**: CTM Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **CTMPF**: CTM Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 3 **STMAE**: STM Comparator A match interrupt control
0: Disable
1: Enable

Bit 2 **STMPE**: STM Comparator P match interrupt control
0: Disable
1: Enable

Bit 1 **CTMAE**: CTM Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **CTMPE**: CTM Comparator P match interrupt control
0: Disable
1: Enable

• **MFI1 Register – HT69F340/HT69F350**

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

Bit 7~6 Unimplemented, read as "0"

Bit 5 **PTMAF**: PTM Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **PTMPF**: PTM Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 3~2 Unimplemented, read as "0"

Bit 1 **PTMAE**: PTM Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **PTMPE**: PTM Comparator P match interrupt control
0: Disable
1: Enable

• **MFI1 Register – HT69F360/HT67F370**

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

• **MFI2 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	DEF	LVF	—	—	DEE	LVE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 Unimplemented, read as "0"

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

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

Bit 3~2 Unimplemented, read as "0"

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

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

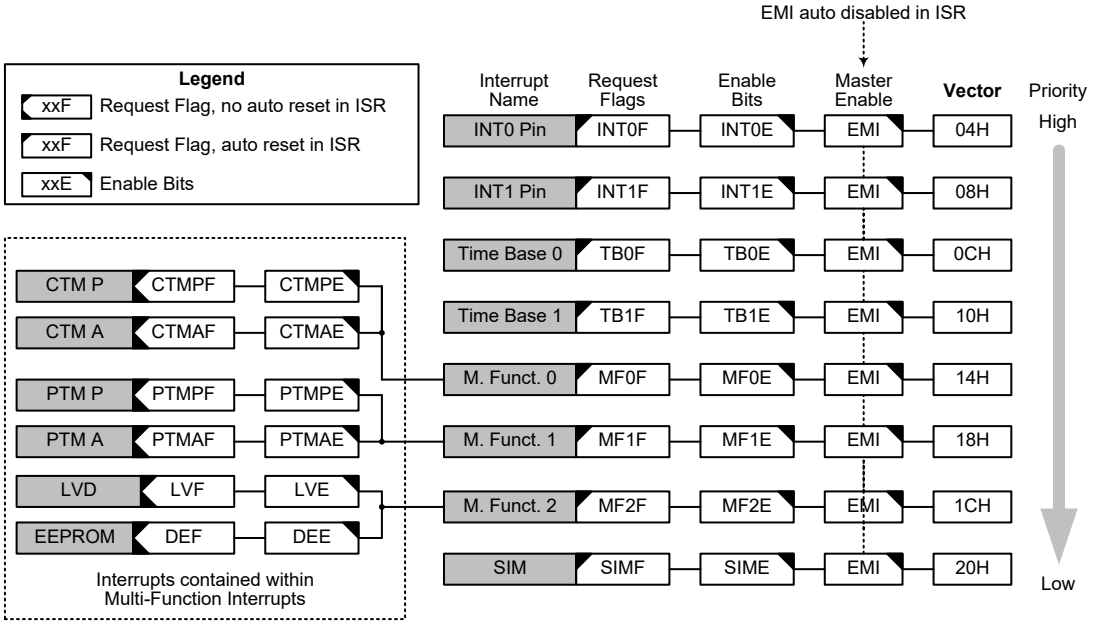
Interrupt Operation

When the conditions for an interrupt event occur, such as a TM Comparator P or Comparator A match, the relevant interrupt request flag will be set. Whether the request flag actually generates a program jump to the relevant interrupt vector is determined by the condition of the interrupt enable bit. If the enable bit is set high then the program will jump to its relevant vector; if the enable bit is zero then although the interrupt request flag is set an actual interrupt will not be generated and the program will not jump to the relevant interrupt vector. The global interrupt enable bit, if cleared to zero, will disable all interrupts.

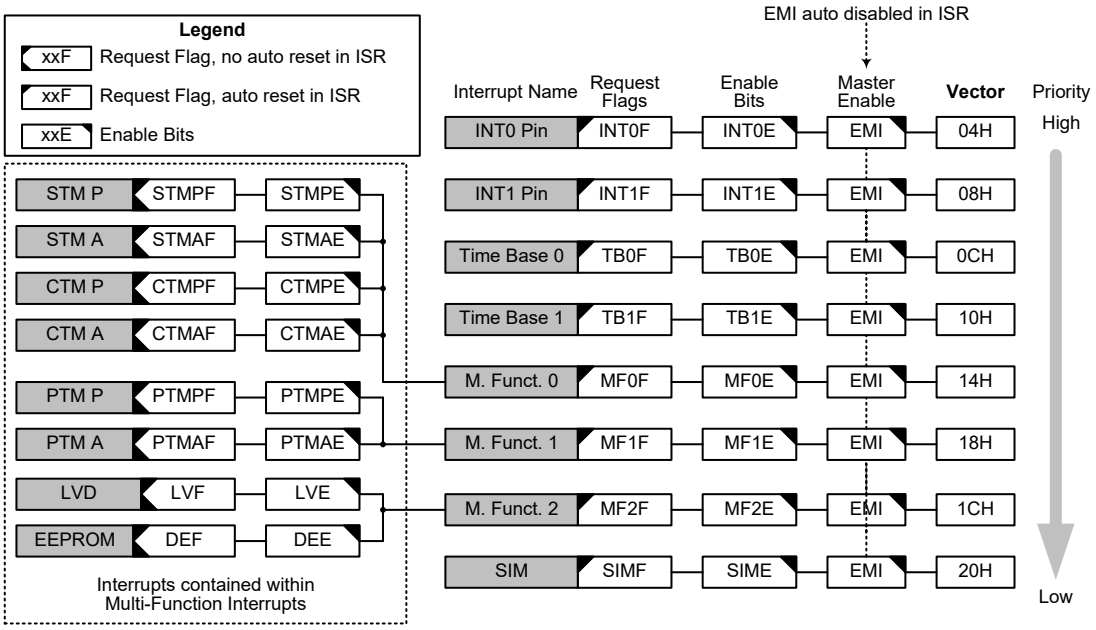
When an interrupt is generated, the Program Counter, which stores the address of the next instruction to be executed, will be transferred onto the stack. The Program Counter will then be loaded with a new address which will be the value of the corresponding interrupt vector. The microcontroller will then fetch its next instruction from this interrupt vector. The instruction at this vector will usually be a "JMP" which will jump to another section of program which is known as the interrupt service routine. Here is located the code to control the appropriate interrupt. The interrupt service routine must be terminated with a "RETI", which retrieves the original Program Counter address from the stack and allows the microcontroller to continue with normal execution at the point where the interrupt occurred.

The various interrupt enable bits, together with their associated request flags, are shown in the accompanying diagrams with their order of priority. Some interrupt sources have their own individual vector while others share the same multi-function interrupt vector. Once an interrupt subroutine is serviced, all the other interrupts will be blocked, as the global interrupt enable bit, EMI bit will be cleared automatically. This will prevent any further interrupt nesting from occurring. However, if other interrupt requests occur during this interval, although the interrupt will not be immediately serviced, the request flag will still be recorded.

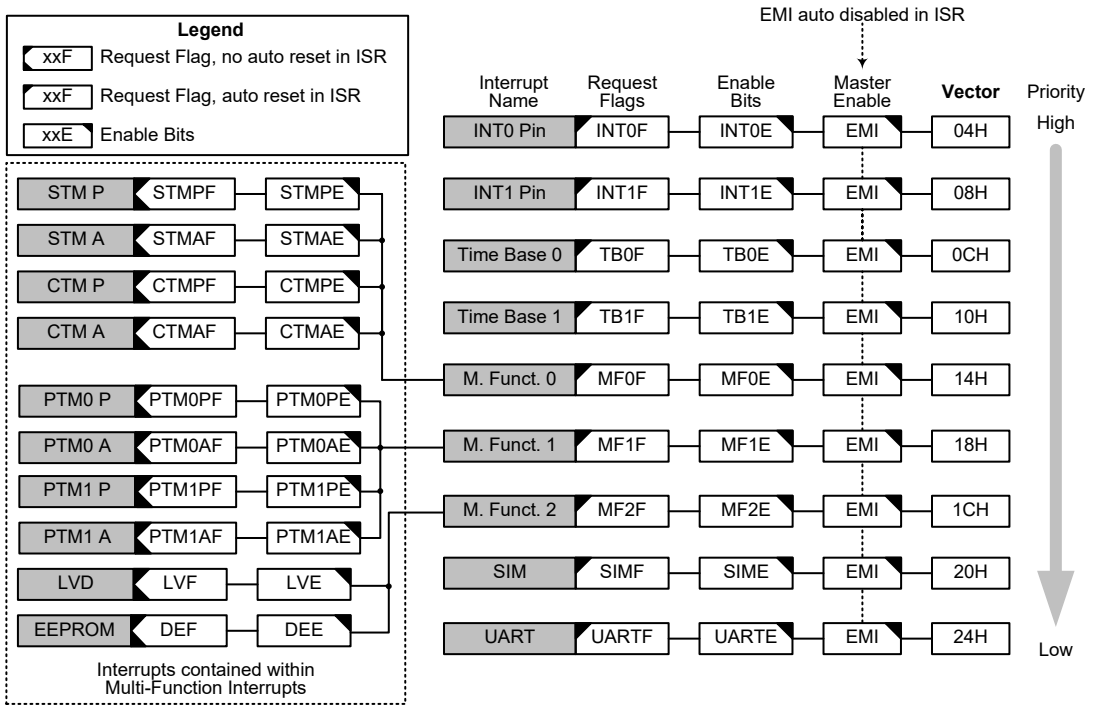
If an interrupt requires immediate servicing while the program is already in another interrupt service routine, the EMI bit should be set after entering the routine, to allow interrupt nesting. If the stack is full, the interrupt request will not be acknowledged, even if the related interrupt is enabled, until the Stack Pointer is decremented. If immediate service is desired, the stack must be prevented from becoming full. In case of simultaneous requests, the accompanying diagram shows the priority that is applied. All of the interrupt request flags when set will wake-up the device if it is in SLEEP or IDLE Mode, however to prevent a wake-up from occurring the corresponding flag should be set before the device is in SLEEP or IDLE Mode.



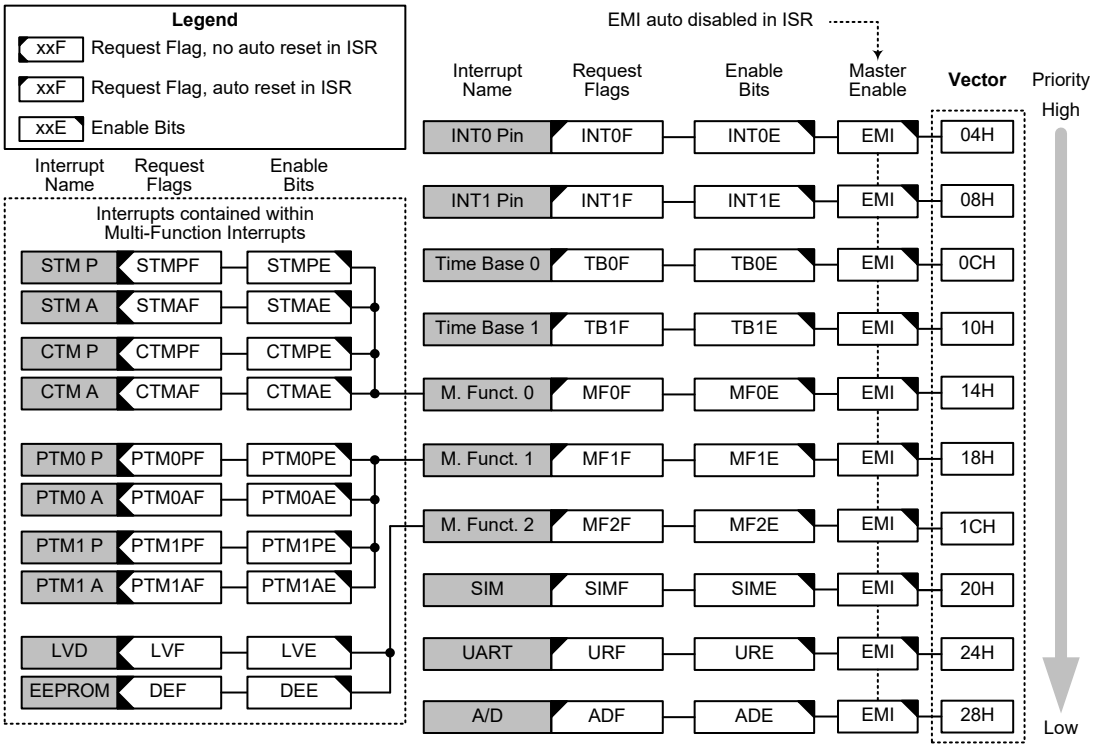
Interrupt Structure – HT69F340



Interrupt Structure – HT69F350



Interrupt Structure – HT69F360



Interrupt Structure – HT67F370

External Interrupt

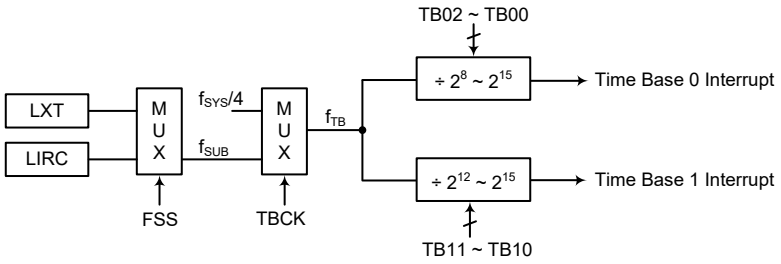
The external interrupt is controlled by signal transitions on the pins INT0~INT1. An external interrupt request will take place when the external interrupt request flags, INT0F~INT1F, are set, which will occur when a transition, whose type is chosen by the edge select bits, appears on the external interrupt pins. To allow the program to branch to the interrupt vector address, the global interrupt enable bit, EMI, and the 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 pin is pin-shared with I/O pins, they can only be configured as external interrupt pins if their external interrupt enable bit in the corresponding interrupt register has been set. The pin must also be setup as an input by setting the corresponding bit in the port control register. When the interrupt is enabled, the stack is not full and the correct transition type appears on the external interrupt pin, a subroutine call to the external interrupt vector, will take place. When the interrupt is serviced, the external interrupt request flags, INT0F~INT1F, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts. Note that the pull-high resistor selection on the external interrupt pin will remain valid even if the pin is used as an external interrupt input.

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

Time Base Interrupts

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

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



Time Base Interrupt

• TBC Register

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

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

Bit 6 **TBCK**: Select f_{TB} Clock
0: f_{SUB}
1: $f_{SYS}/4$

Bit 5~4 **TB11~TB10**: Select Time Base 1 Time-out Period
00: $2^{12}/f_{TB}$
01: $2^{13}/f_{TB}$
10: $2^{14}/f_{TB}$
11: $2^{15}/f_{TB}$

Bit 3 Unimplemented, read as "0"

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

Multi-function Interrupt

Within these devices there are up to three 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, EEPROM Interrupt and LVD Interrupt.

A Multi-function interrupt request will take place when any of the Multi-function interrupt request flags, MF0F~MF2F are set. The Multi-function interrupt flags will be set when any of their included functions generate an interrupt request flag. To allow the program to branch to its respective interrupt vector address, when the Multi-function interrupt is enabled and the stack is not full, and either one of the interrupts contained within each of Multi-function interrupt occurs, a subroutine call to one of the Multi-function interrupt vectors will take place. When the interrupt is serviced, the related Multi-Function request flag, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts.

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

EEPROM Interrupt

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

LVD Interrupt

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

TM Interrupts

The Compact, Standard and Periodic Type TMs each has two interrupts. All of the TM interrupts are contained within the Multi-function Interrupts. For each of the Compact, 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 comparator A match situation happens.

To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the respective TM Interrupt enable bit, and associated Multi-function interrupt enable bit, MFnF, must first be set. When the interrupt is enabled, the stack is not full and a TM comparator match situation occurs, a subroutine call to the relevant TM Interrupt vector locations, will take place. When the TM interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the related MFnF flag will be automatically cleared. As the TM interrupt request flags will not be automatically cleared, they have to be cleared by the application program.

Serial Interface Module Interrupt

The Serial Interface Module Interrupt is also known as the SIM interrupt. A SIM Interrupt request will take place when the SIM Interrupt request flag, SIMF, is set, which occurs when a byte of data has been received or transmitted by the SIM interface. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the Serial Interface Interrupt enable bit, SIME, must first be set. When the interrupt is enabled, the stack is not full and a byte of data has been transmitted or received by the SIM interface, a subroutine call to the respective SIM Interrupt vector, will take place. When the interrupt is serviced, the interrupt request flag, SIMF, will be automatically reset and the EMI bit will be cleared to disable other interrupts.

UART Interrupt

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. 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 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 be cleared automatically when certain actions are taken by the UART, the details of which are given in the UART section.

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.

Interrupt Wake-up Function

Each of the interrupt functions has the capability of waking up the microcontroller when in the SLEEP or IDLE Mode. A wake-up is generated when an interrupt request flag changes from low to high and is independent of whether the interrupt is enabled or not. Therefore, even though these devices are in the SLEEP or IDLE Mode and its system oscillator stopped, situations such as external edge transitions on the external interrupt pins, a low power supply voltage or comparator input change may cause their respective interrupt flag to be set high and consequently generate an interrupt. Care must therefore be taken if spurious wake-up situations are to be avoided. If an interrupt wake-up function is to be disabled then the corresponding interrupt request flag should be

set high before the device enters the SLEEP or IDLE Mode. The interrupt enable bits have no effect on the interrupt wake-up function.

Programming Considerations

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

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

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

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

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

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

Low Voltage Detector – LVD

Each device has a Low Voltage Detector function, also known as LVD. This enables the device to monitor the power supply voltage, V_{DD}, and provides 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	LVPU	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

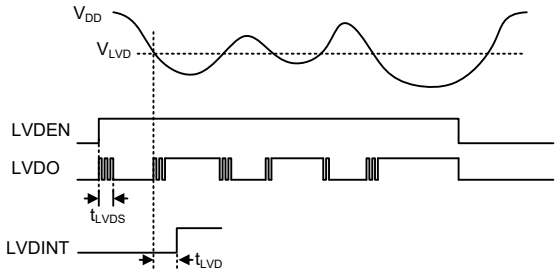
Bit 4 **LVDEN**: Low Voltage Detector Control
0: Disable
1: Enable

Bit 3 **LVPU**: I/O ports Pull-high resistor Control
0: All pin pull-high resistor is 60kΩ @5V (normal mode)
1: All pin pull-high resistor is 15kΩ @5V (low voltage mode)

Bit 2~0 **VLVD2~VLVD0**: Select LVD Voltage
000: 1.8V
001: 2.0V
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 1.8V and 4.0V. When the power supply voltage, V_{DD}, falls below this pre-determined value, the LVDO bit will be set high indicating a low power supply voltage condition. The Low Voltage Detector function is supplied by a reference voltage which will be automatically enabled. When the device is in SLEEP mode the low voltage detector will be automatically disabled. After enabling the Low Voltage Detector, a time delay t_{LVDS} should be allowed for the circuitry to stabilise before reading the LVDO bit. Note also that as the V_{DD} voltage may rise and fall rather slowly, at the voltage nears that of V_{LVD}, there may be multiple bit LVDO transitions.



LVD Operation

The Low Voltage Detector also has its own interrupt which is contained within one of the Multi-function interrupts, providing an alternative means of low voltage detection, in addition to polling the LVDO bit. The interrupt will only be generated after a delay of t_{LVD} after the LVDO bit has been set high by a low voltage condition. When the device is powered down the Low Voltage Detector will remain active if the LVDEN bit is high. In this case, the LVF interrupt request flag will be set, causing an interrupt to be generated if V_{DD} falls below the preset LVD voltage. This will cause the device to wake-up from the SLEEP Mode, however if the Low Voltage Detector wake up function is not required then the LVF flag should be first set high before the device enters the SLEEP Mode.

LCD Driver

For large volume applications, which incorporate an LCD in their design, the use of a custom display rather than a more expensive character based display reduces costs significantly. However, the corresponding COM and SEG signals required, which vary in both amplitude and time, to drive such a custom display require many special considerations for proper LCD operation to occur. The devices contain an LCD Driver function, which with its internal LCD signal generating circuitry and various options, will automatically generate these time and amplitude varying signals to provide a means of direct driving and easy interfacing to a range of custom LCDs.

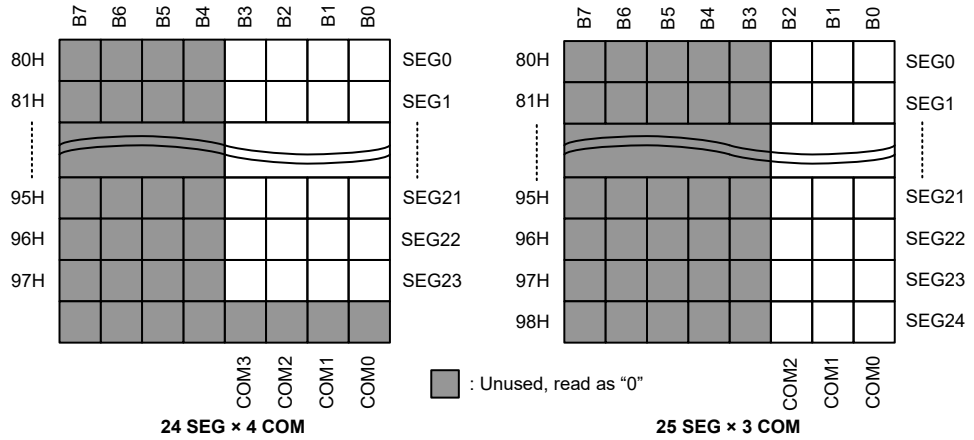
Device	Duty	Driver Output	Bias	Bias Type	Wave Type
HT69F340	1/4 or 1/3	24×4 or 25×3	1/3 or 1/2	C or R	A or B
HT69F350	1/4 or 1/3	36×4 or 37×3	1/3 or 1/2	C or R	A or B
HT69F360/ HT67F370	1/4 or 1/3	48×4 or 49×3	1/3 or 1/2	C or R	A or B

LCD Display Memory

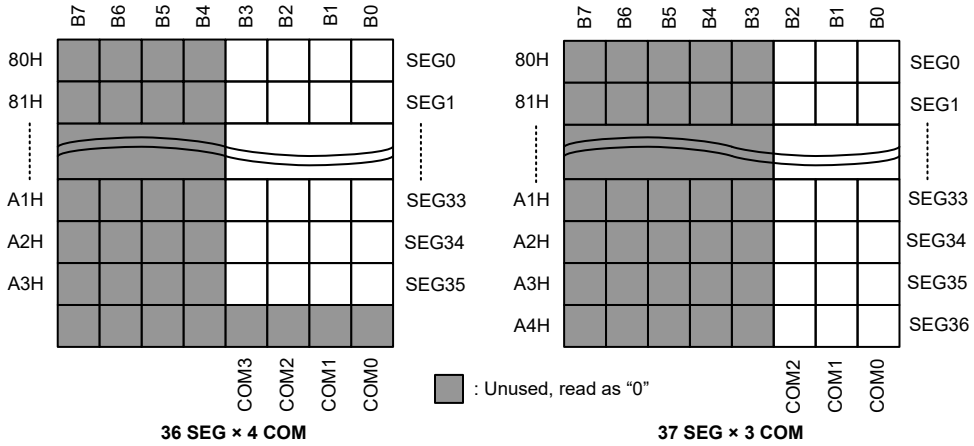
An area of Data Memory is especially reserved for use for the LCD display data. This data area is known as the LCD Memory. Any data written here will be automatically read by the internal display driver circuits, which will in turn automatically generate the necessary LCD driving signals. Therefore any data written into this Memory will be immediately reflected into the actual display connected to the microcontroller.

As the LCD Memory addresses overlap those of the General Purpose Data Memory, it is stored in its own independent Sector 1 area. The Data Memory Sector to be used is chosen by using the Memory Pointer high byte registers, which are special function registers in the Data Memory, with the name, MP1H and MP2H. To access the Display Memory therefore requires first that Sector 1 is selected by writing a value of 01H to MP1H or MP2H register. After this, the memory can then be accessed by using indirect addressing through the use of Memory Pointer low byte MP1L or MP2L. With Sector 1 selected, then using MP1L or MP2L to read or write to the memory area, starting from the address 80H for all the devices, will result in operations to the LCD memory. Directly addressing the Display Memory can be carried out by using the corresponding extended instructions.

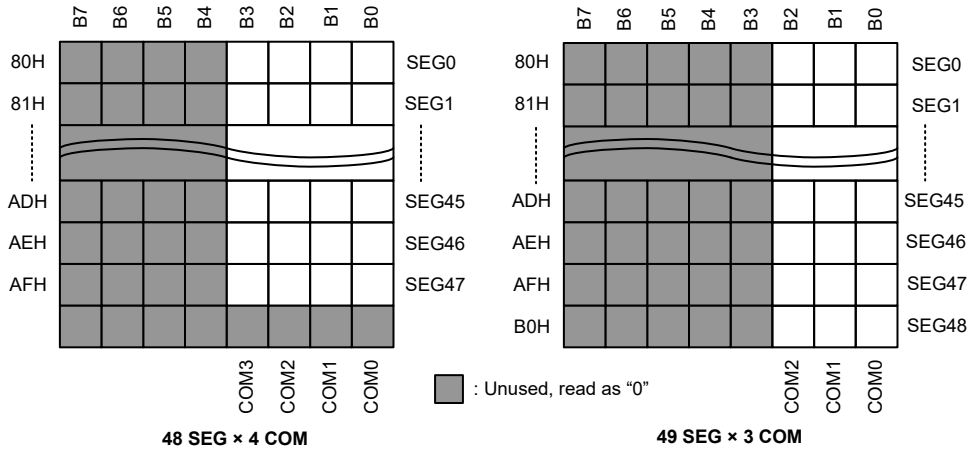
The accompanying LCD Memory Map diagram shows how the internal LCD memory is mapped to the Segments and Commons of the display for the devices. LCD Memory Map for the devices with smaller memory capacities can be extrapolated from the diagrams.



LCD Memory Map – HT69F340



LCD Memory Map – HT69F350



LCD Memory Map – HT69F360/HT67F370

LCD Clock Source

The LCD clock source is the internal clock signal, f_{SUB}, divided by 8, using an internal divider circuit. The f_{SUB} internal clock is supplied by either the LIRC or LXT oscillator, the choice of which is determined by the FSS bit in SCC register. For proper LCD operation, this arrangement is provided to generate an ideal LCD clock source frequency of 4kHz.

LCD Registers

There are two control registers, named as LCDC0 and LCDC1, in the Data Memory used to control the various setup features of the LCD Driver. Various bits in these registers control functions such as wave type, duty type, bias duty, bias resistor selection, R/C type as well as overall LCD enable/disable and LCD power source selection.

The LCDEN bit in the LCDC0 register, which provides the overall LCD enable/disable function, will only be effective when the device is in the Normal, Slow or Idle Mode. If the devices are in the Sleep Mode then the display will always be disabled. RSEL1~RSEL0 in the LCDC0 register select the internal bias resistor to supply the LCD panel with the correct bias voltage. A choice to best match the LCD panel used in the application can be selected also to minimise bias current. The TYPE bit in the same register is used to select whether Type A or Type B LCD control signals are used. If the output function of display pins SEGn are used as segment drivers or I/O or other functions is determined by pin-shared function selection registers.

Register Name	Bit							
	7	6	5	4	3	2	1	0
LCDC0	TYPE	—	DTYC	—	BIAS	RSEL1	RSEL0	LCDEN
LCDC1	—	—	—	—	—	RCT	LCDP1	LCDP0

- **LCDC0 Register**

Bit	7	6	5	4	3	2	1	0
Name	TYPE	—	DTYC	—	BIAS	RSEL1	RSEL0	LCDEN
R/W	R/W	—	R/W	—	R/W	R/W	R/W	R/W
POR	0	—	0	—	0	0	0	0

Bit 7 **TYPE**: LCD Wave Type Control
0: Type A
1: Type B

Bit 6 Unimplemented, read as "0"

Bit 5 **DTYC**: LCD Duty Control
0: 1/3 duty
1: 1/4 duty

Bit 4 Unimplemented, read as "0"

Bit 3 **BIAS**: LCD Bias Control
0: 1/2 bias
1: 1/3 bias

Bit 2~1 **RSEL1~RSEL0**: LCD Bias Resistor Selection
When 1/3 Bias
00: 600kΩ
01: 300kΩ
10: 100kΩ
11: 50kΩ
When 1/2 Bias
00: 400kΩ
01: 200kΩ
10: 67kΩ
11: 34kΩ

Bit 0 **LCDEN**: LCD Enable Control
0: Disable
1: Enable
In the NORNAL, SLOW or IDLE mode, the LCD on/off function can be controlled by this bit. In the SLEEP mode, the LCD is always off.

• **LCDC1 Register**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	RCT	LCDP1	LCDP0
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 Unimplemented, read as "0"

Bit 2 **RCT**: LCD R/C Type Control
0: R Type
1: C Type

Bit 1~0 **LCDP1~LCDP0**: LCD power source select for C type
00: the power source is from VLCD/V1/V2
01: the power source is from VC=DPN Vref (~1.04V)
10: the power source is from VB=V_{DD}
11: the power source is from VA=V_{DD}
If LCDEN=1 and RCT=1,the PC0~2 will be V2, C1, C2 function respectively.

LCD Reset Function

The LCD has an internal reset function that is an OR function of the inverted LCDEN bit in the LCDC0 register and the SLEEP function. When the LCDEN bit is set to 1 to enable the LCD driver function before the device enters the SLEEP mode, the LCD function will be reset after the device enters the SLEEP mode. Clearing the LCDEN bit to zero will also reset the LCD function.

LCDEN	SLEEP Mode	Reset LCD
0	Off	√
0	On	√
1	Off	x
1	On	√

LCD Reset Function

LCD Driver Output

The number of COM and SEG outputs supplied by the LCD driver, as well as the wave type, R/C type, biasing and the duty selections, are dependent upon how the LCD control bits are programmed.

The nature of Liquid Crystal Displays require that only AC voltages can be applied to their pixels as the application of DC voltages to LCD pixels may cause permanent damage. For this reason the relative contrast of an LCD display is controlled by the actual RMS voltage applied to each pixel, which is equal to the RMS value of the voltage on the COM pin minus the voltage applied to the SEG pin. This differential RMS voltage must be greater than the LCD saturation voltage for the pixel to be on and less than the threshold voltage for the pixel to be off.

The requirement to limit the DC voltage to zero and to control as many pixels as possible with a minimum number of connections requires that both a time and amplitude signal is generated and applied to the application LCD. These time and amplitude varying signals are automatically generated by the LCD driver circuits in the microcontroller. What is known as the duty determines the number of common lines used, which are also known as backplanes or COMs. The duty, which is chosen by a control bit to have a value of 1/3 or 1/4 and which equates to a COM number of 3

or 4 respectively, therefore defines the number of time divisions within each LCD signal frame. Two types of signal generation are also provided, known as Type A and Type B, the required type is selected via the TYPE bit in the LCDC0 register. Type B offers lower frequency signals, however lower frequencies may introduce flickering and influence display clarity.

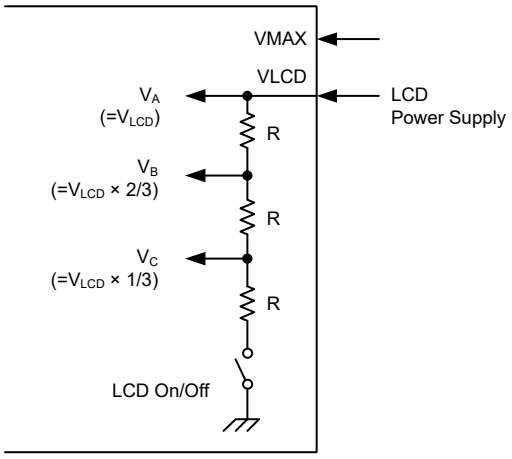
LCD Voltage Source and Biasing

The time and amplitude varying signals generated by the LCD Driver function require the generation of several voltage levels for their operation. The device can have either R type or C type biasing selected via a software control bit RCT in the LCDC1 register. Selecting the C type biasing will enable an internal charge pump circuitry.

R Type Biasing

For R type biasing an external LCD voltage source must be supplied on pin VLCD to generate the internal biasing voltages. This could be the microcontroller power supply or some other voltage source. For the R type 1/2 bias selection, three voltage levels V_{SS}, V_A and V_B are utilised. The voltage V_A is equal to the externally supplied voltage source applied to pin VLCD. The voltage V_B is generated internally by the microcontroller and will have a value equal to $V_{LCD}/2$. For the R type 1/3 bias selection, four voltage levels V_{SS}, V_A, V_B and V_C are utilised. The voltage V_A is equal to V_{LCD}. The voltage V_B is equal to $V_{LCD} \times 2/3$ while the voltage V_C is equal to $V_{LCD} \times 1/3$.

For 1/2 or 1/3 bias, different values of internal bias resistors can be selected using the RSEL0 and RSEL1 bits in the LCDC0 register. This along with the voltage on pin VLCD will determine the bias current. The connection to the VMAX pin depends upon the voltage that is applied to the VLCD pin. If the VDD voltage is greater than the voltage applied to the VLCD pin then the VMAX pin should be connected to VDD, otherwise the VMAX pin should be connected to pin VLCD. Note that no external capacitors or resistors are required to be connected if R type biasing is used.



R Type 1/3 Bias

R Type Bias Voltage Levels

Condition	VMAX Connection
$V_{DD} > V_{LCD}$	Connect VMAX to VDD
Otherwise	Connect VMAX to VLCD

R Type Bias Current VMAX Connection

C Type Biasing

For C type biasing an external pin or internal power source is selected via LCDP1~LCDP0 bit in LCDC1 register. The LCD voltage source is supplied on pin VLCD, V1, V2 or internal power to generate the internal biasing voltages.

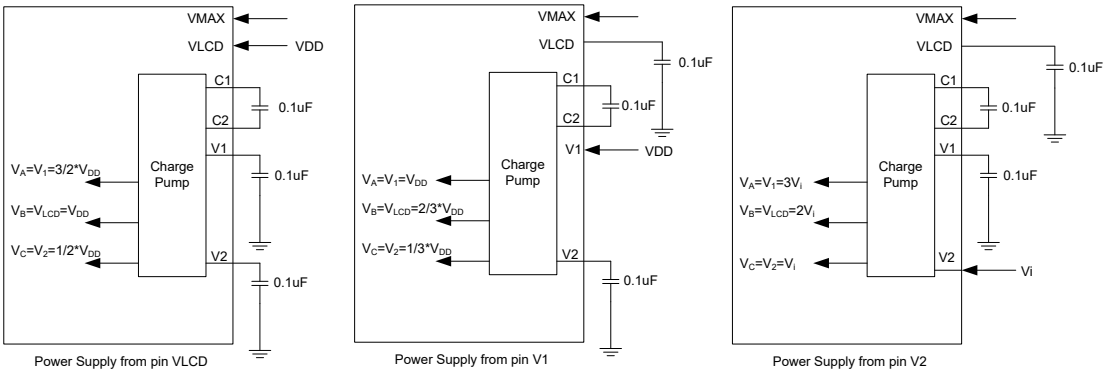
When power source is from pin VLCD, the C type biasing scheme uses an internal charge pump circuit, which in the case of the 1/3 bias selection can generate voltages higher than what is supplied on VLCD. This feature is useful in applications where the microcontroller supply voltage is less than the supply voltage required by the LCD. An additional charge pump capacitor must also be connected between pins C1 and C2 to generate the necessary voltage levels.

The device has a built-in depletion circuit for LCD voltage source. This could be the DPN Vref (~1.04V) or internal power (VDD) to generate biasing voltage when LCDP bit be configured 01, 10 or 11.

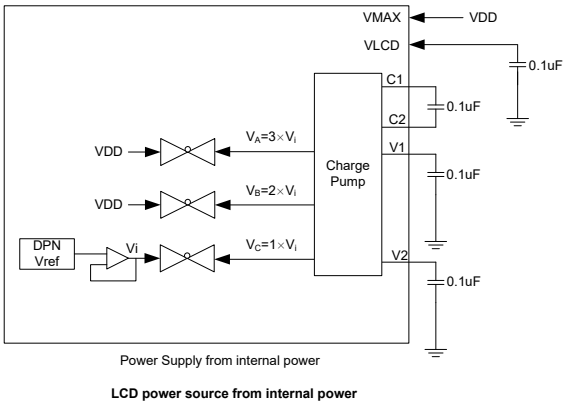
For the C type 1/3 bias selection, four voltage levels V_{SS}, V_A, V_B and V_C are utilised. When power source is from pin V1 or VDD, which is maximum bias. The pin VLCD must connect a 0.1uf to ground. And charge pump will generate V_B and V_C, V_B will have a value equal to $V_A \times 2/3$ and V_C will have a value equal to $V_A \times 1/3$.

When power source is from pin VLCD or VDD, The voltage V_A is generated internally and has a value of $V_{LCD} \times 1.5$. V_B will have a value equal to $V_A \times 2/3$ and V_C will have a value equal to $V_A \times 1/3$.

When power source is from pin V2 or DPN Vref, The voltage V_A is generated internally and has a value of $V_C \times 3$. V_B will have a value equal to $V_C \times 2$ and V_C will have a value equal to V_2 or DPN Vref.



LCD power source from external pin



LCD power source from internal power

Note: The pin VMAX must connect to maximum voltage to prevent pad leakage.

C Type Bias Voltage Levels

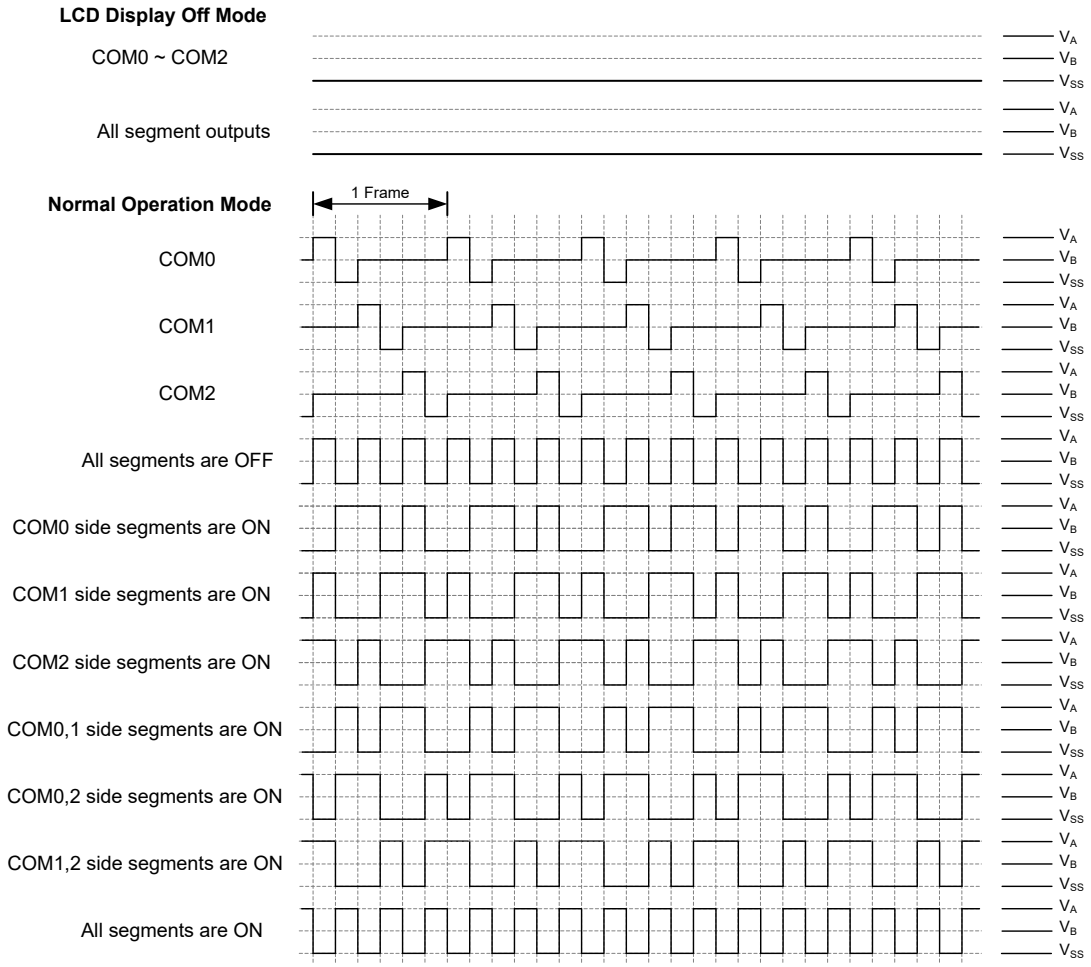
The connection to the VMAX pin depends upon the bias and the voltage that is applied to VLCD, the details are shown in the table. It is extremely important to ensure that these charge pump generated internal voltages do not exceed the maximum VDD voltage of 5.5V.

Biasing Type		VMAX Connection
1/3 Bias	$V_{DD} > V_{LCD} \times 1.5$	Connect VMAX to VDD
	Otherwise	Connect VMAX to V1
1/2 Bias	$V_{DD} > V_{LCD}$	Connect VMAX to VDD
	Otherwise	Connect VMAX to VLCD

C Type Biasing VMAX Connection

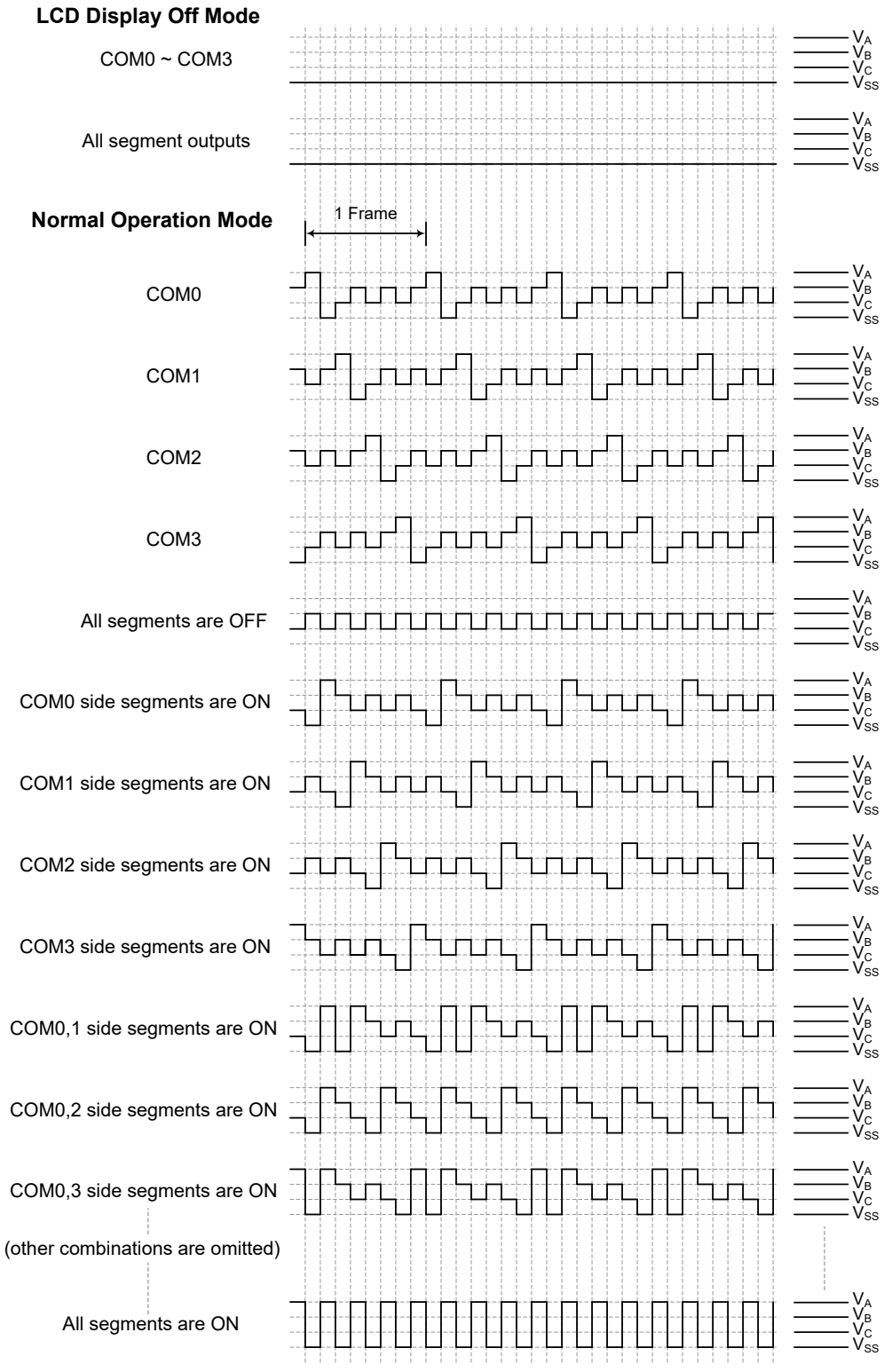
LCD Waveform Timing Diagram

The accompanying timing diagrams depict the display driver signals generated by the microcontroller for various values of duty and bias. The huge range of various permutations only permits a few types to be displayed here.



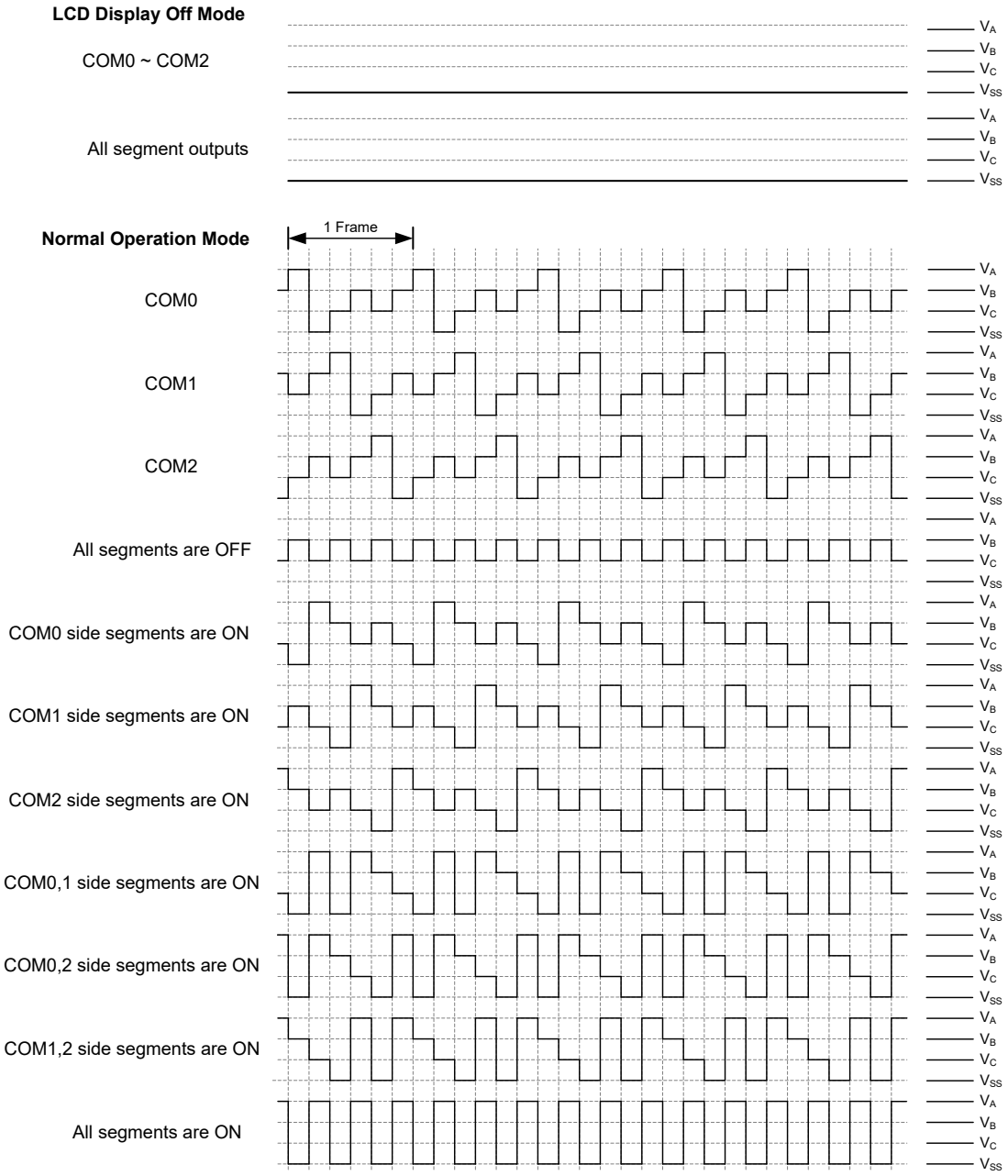
LCD Driver Output – Type A – 1/3 Duty, 1/2 Bias

Note: For 1/2 bias, the $V_A=V_{LCD}$, $V_B=V_{LCD}\times 1/2$ for both R and C type.



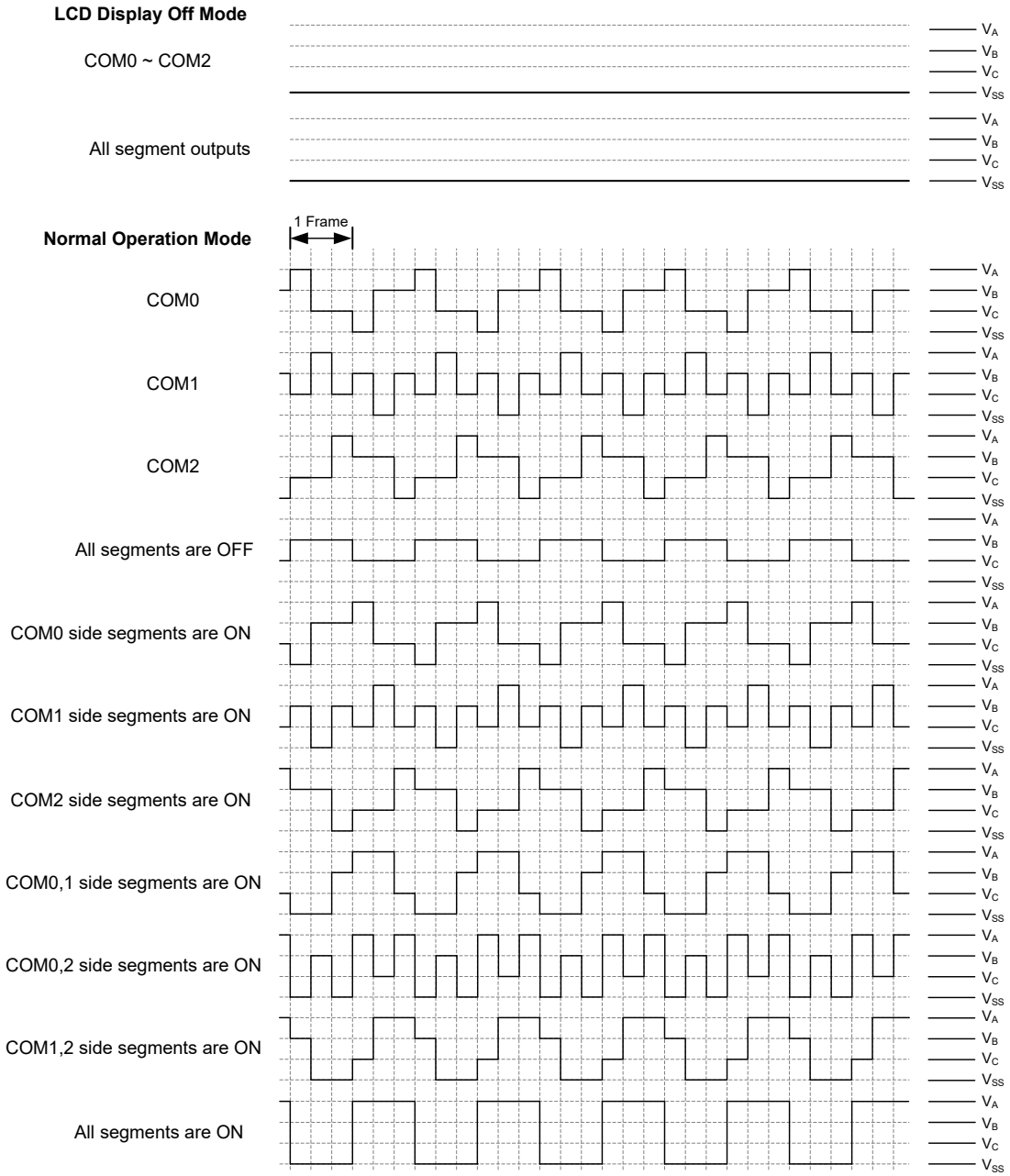
LCD Driver Output – Type A – 1/4 Duty, 1/3 Bias

Note: For 1/3 R type bias, the $V_A=V_{LCD}$, $V_B=V_{LCD} \times 2/3$ and $V_C=V_{LCD} \times 1/3$
For 1/3 C type bias, the $V_A=V_{LCD} \times 1.5$, $V_B=V_{LCD}$ and $V_C=V_{LCD} \times 1/2$



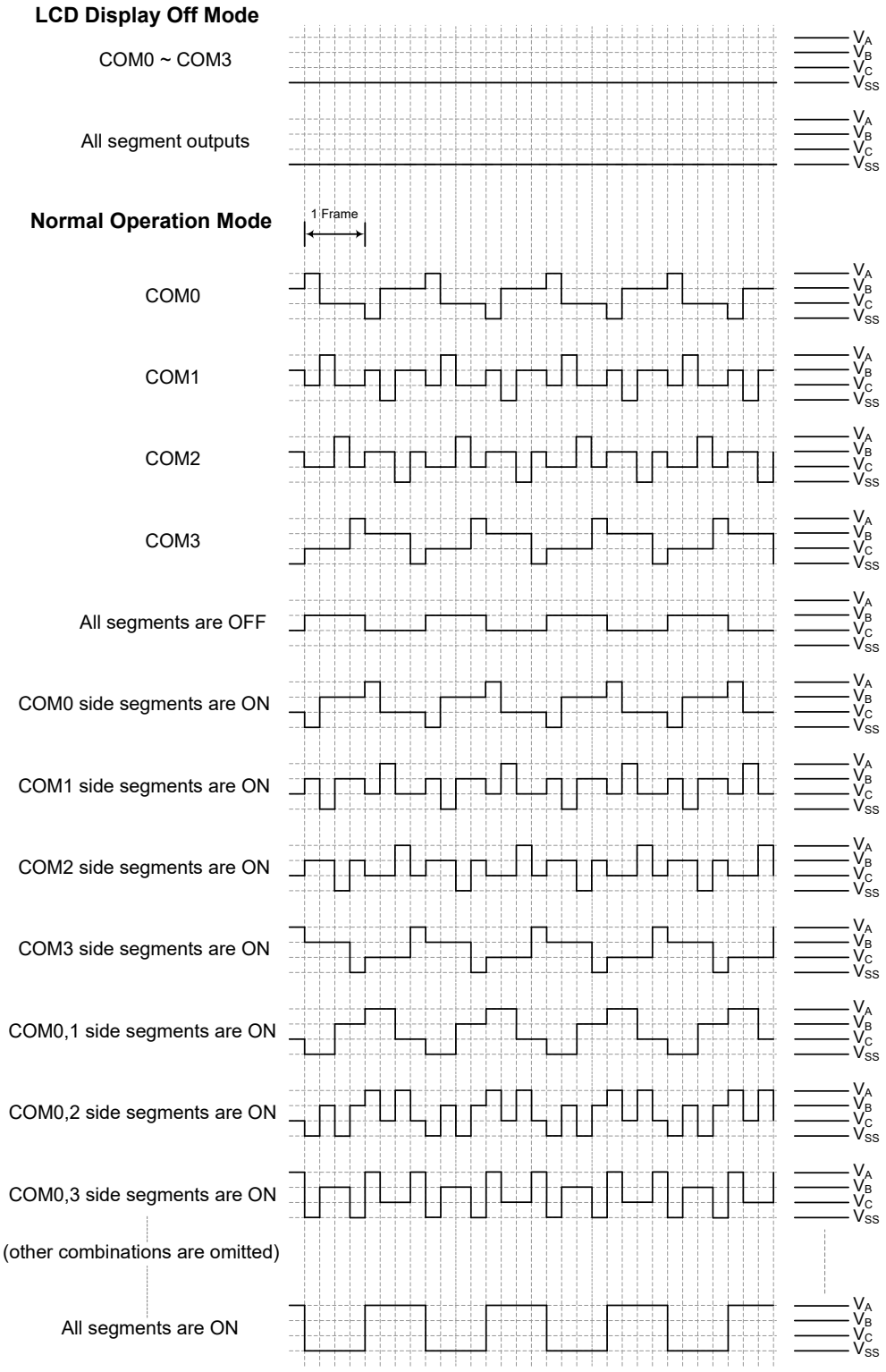
LCD Driver Output – Type A – 1/3 Duty, 1/3 Bias

Note: For 1/3 R type bias, the $V_A=V_{LCD}$, $V_B=V_{LCD} \times 2/3$ and $V_C=V_{LCD} \times 1/3$
For 1/3 C type bias, the $V_A=V_{LCD} \times 1.5$, $V_B=V_{LCD}$ and $V_C=V_{LCD} \times 1/2$



LCD Driver Output – Type B – 1/3 Duty, 1/3 Bias

Note: For 1/3 R type bias, the $V_A=V_{LCD}$, $V_B=V_{LCD} \times 2/3$ and $V_C=V_{LCD} \times 1/3$
For 1/3 C type bias, the $V_A=V_{LCD} \times 1.5$, $V_B=V_{LCD}$ and $V_C=V_{LCD} \times 1/2$



LCD Driver Output – Type B – 1/4 Duty, 1/3 Bias

Note: For 1/3 R type bias, the $V_A=V_{LCD}$, $V_B=V_{LCD} \times 2/3$ and $V_C=V_{LCD} \times 1/3$

For 1/3 C type bias, the $V_A=V_{LCD} \times 1.5$, $V_B=V_{LCD}$ and $V_C=V_{LCD} \times 1/2$

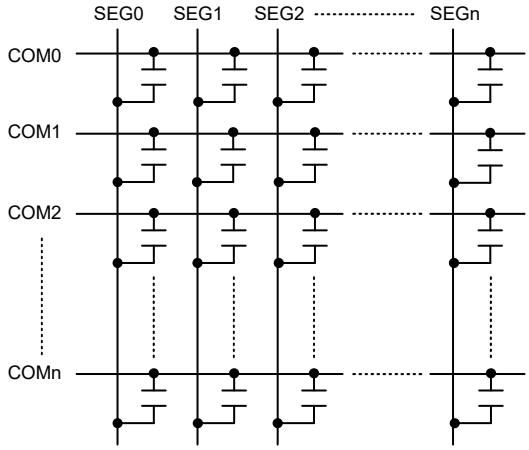
Programming Considerations

Certain precautions must be taken when programming the LCD. One of these is to ensure that the LCD Memory is properly initialized after the microcontroller is powered on. Like the General Purpose Data Memory, the contents of the LCD Memory are in an unknown condition after power-on. As the contents of the LCD Memory will be mapped into the actual display, it is important to initilise this memory area into a known condition soon after applying power to obtain a proper display pattern.

Consideration must also be given to the capacitive load of the actual LCD used in the application. As the load presented to the microcontroller by LCD pixels can be generally modeled as mainly capacitive in nature, it is important that this is not excessive, a point that is particularly true in the case of the COM lines which may be connected to many LCD pixels. The accompanying diagram depicts the equivalent circuit of the LCD.

One additional consideration that must be taken into account is what happens when the microcontroller enters the Idle or Slow Mode. The LCDEN control bit in the LCDC0 register permits the display to be powered off to reduce power consumption. If this bit is zero, the driving signals to the display will cease, producing a blank display pattern but reducing any power consumption associated with the LCD.

After power on, note that as the LCDEN bit will be cleared to zero, the display function will be disabled.



LCD Panel Equivalent Circuit

Configuration Options

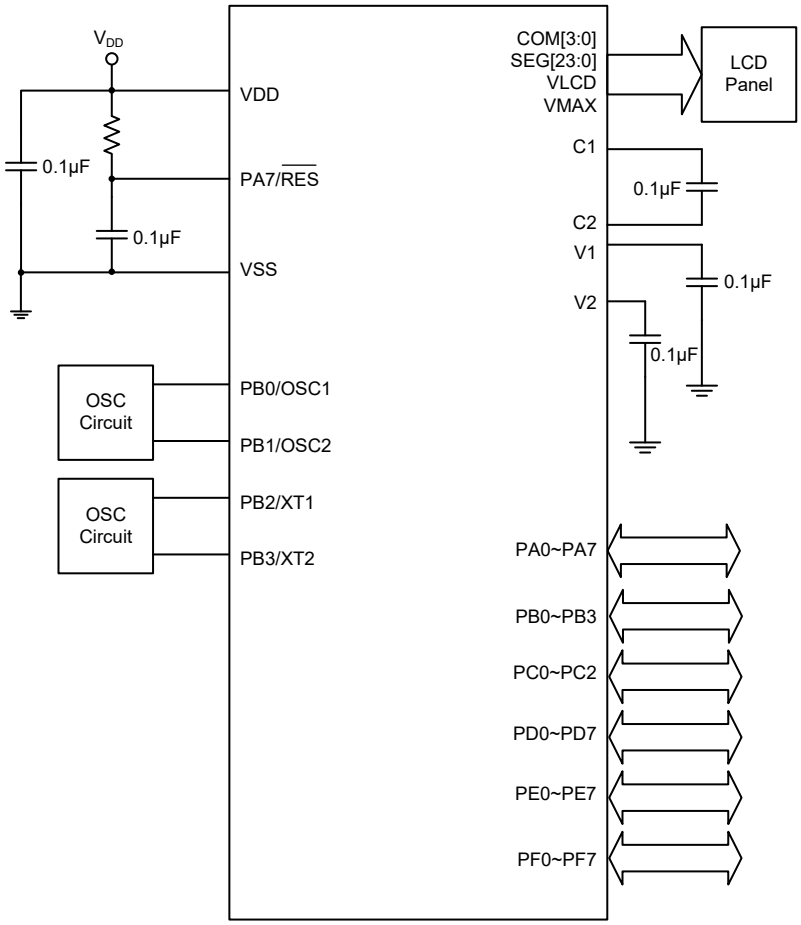
Configuration options refer to certain options within the MCU that are programmed into the device during the programming process. During the development process, these options are selected using the HT-IDE software development tools. As these options are programmed into the device using the hardware 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	PA7 or $\overline{RES}$ pin selection 1. $\overline{RES}$ pin 2. PA7
Oscillator Option	
2	HIRC Frequency Selection – f_{HIRC}: 4MHz, 8MHz or 12MHz

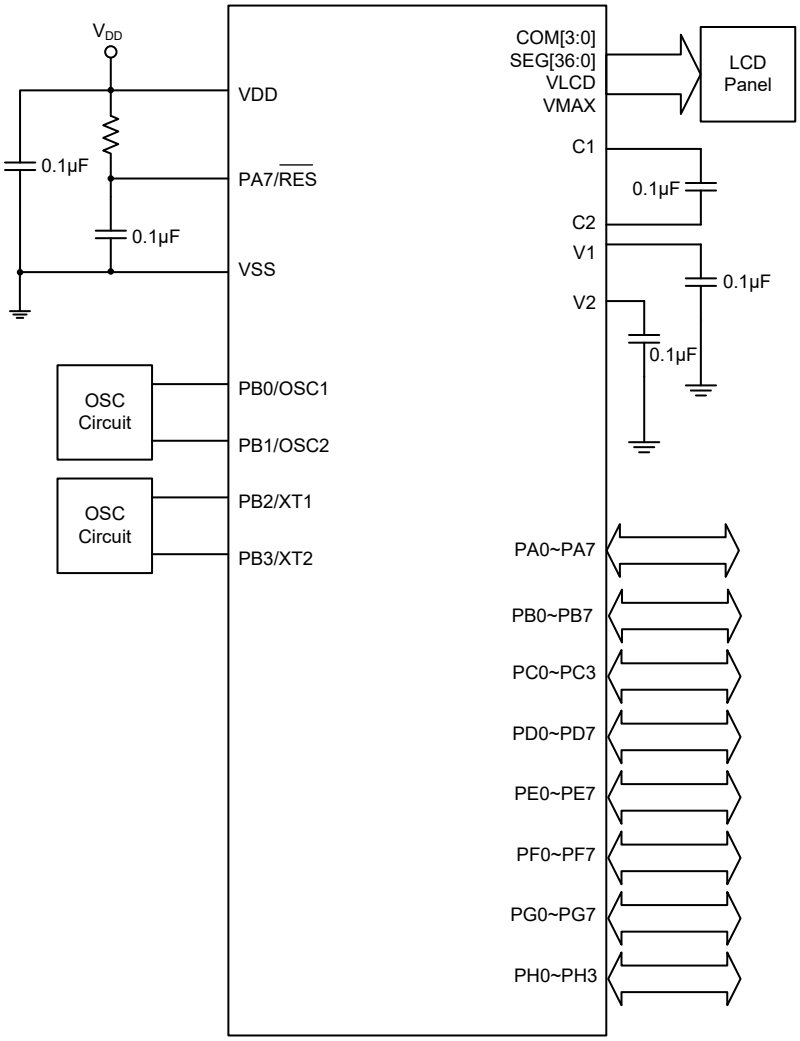
Note: When the HIRC has been configured at a frequency shown in this table, the HIRC1 and HIRC0 bits should also be setup to select the same frequency to achieve the HIRC frequency accuracy specified in the A.C. Characteristics.

Application Circuits

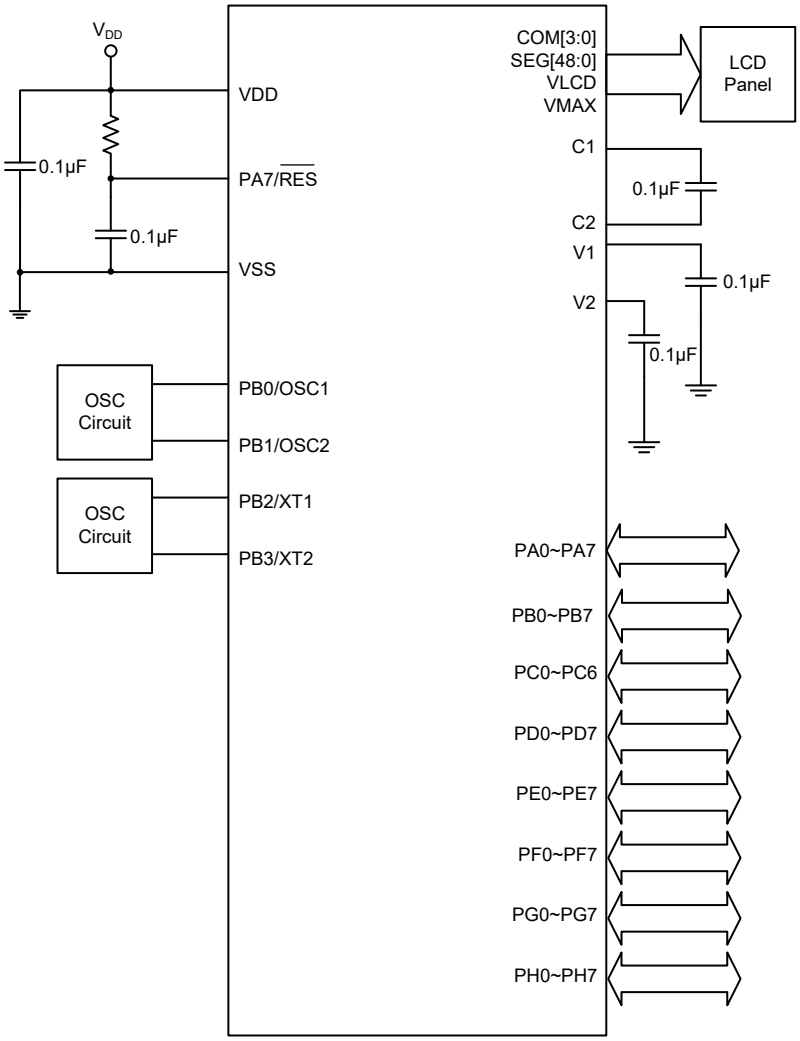
HT69F340



HT69F350


VDD
0.1μF
PA7/RES
0.1μF
VSS
OSC Circuit
PB0/OSC1
PB1/OSC2
OSC Circuit
PB2/XT1
PB3/XT2
COM[3:0]
SEG[36:0]
VLCD
VMAX
LCD Panel
C1
0.1μF
C2
V1
0.1μF
V2
0.1μF
PA0~PA7
PB0~PB7
PC0~PC3
PD0~PD7
PE0~PE7
PF0~PF7
PG0~PG7
PH0~PH3

HT69F360/HT67F370



Instruction Set

Introduction

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

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

Instruction Timing

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

Moving and Transferring Data

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

Arithmetic Operations

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

Logical and Rotate Operation

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

Branches and Control Transfer

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

Bit Operations

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

Table Read Operations

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

Other Operations

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

Instruction Set Summary

The instructions related to the data memory access in the following table can be used when the desired data memory is located in Data Memory sector 0.

Table Conventions

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

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

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

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

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

Extended Instruction Set

The extended instructions are used to support the full range address access for the data memory. When the accessed data memory is located in any data memory sector except sector 0, the extended instruction can be used to directly access the data memory instead of using the indirect addressing access. This can not only reduce the use of Flash memory space but also improve the CPU execution efficiency.

Mnemonic	Description	Cycles	Flag Affected
Arithmetic			
LADD A,[m]	Add Data Memory to ACC	2	Z, C, AC, OV, SC
LADDM A,[m]	Add ACC to Data Memory	2Note	Z, C, AC, OV, SC
LADC A,[m]	Add Data Memory to ACC with Carry	2	Z, C, AC, OV, SC
LADCM A,[m]	Add ACC to Data memory with Carry	2Note	Z, C, AC, OV, SC
LSUB A,[m]	Subtract Data Memory from ACC	2	Z, C, AC, OV, SC, CZ
LSUBM A,[m]	Subtract Data Memory from ACC with result in Data Memory	2Note	Z, C, AC, OV, SC, CZ
LSBC A,[m]	Subtract Data Memory from ACC with Carry	2	Z, C, AC, OV, SC, CZ
LSBCM A,[m]	Subtract Data Memory from ACC with Carry, result in Data Memory	2Note	Z, C, AC, OV, SC, CZ
LDAA [m]	Decimal adjust ACC for Addition with result in Data Memory	2Note	C
Logic Operation			
LAND A,[m]	Logical AND Data Memory to ACC	2	Z
LOR A,[m]	Logical OR Data Memory to ACC	2	Z
LXOR A,[m]	Logical XOR Data Memory to ACC	2	Z
LANDM A,[m]	Logical AND ACC to Data Memory	2Note	Z
LORM A,[m]	Logical OR ACC to Data Memory	2Note	Z
LXORM A,[m]	Logical XOR ACC to Data Memory	2Note	Z
LCPL [m]	Complement Data Memory	2Note	Z
LCPLA [m]	Complement Data Memory with result in ACC	2	Z
Increment & Decrement			
LINCA [m]	Increment Data Memory with result in ACC	2	Z
LINC [m]	Increment Data Memory	2Note	Z
LDECA [m]	Decrement Data Memory with result in ACC	2	Z
LDEC [m]	Decrement Data Memory	2Note	Z
Rotate			
LRRA [m]	Rotate Data Memory right with result in ACC	2	None
LRR [m]	Rotate Data Memory right	2Note	None
LRRCA [m]	Rotate Data Memory right through Carry with result in ACC	2	C
LRRC [m]	Rotate Data Memory right through Carry	2Note	C
LRLA [m]	Rotate Data Memory left with result in ACC	2	None
LRL [m]	Rotate Data Memory left	2Note	None
LRLCA [m]	Rotate Data Memory left through Carry with result in ACC	2	C
LRLC [m]	Rotate Data Memory left through Carry	2Note	C
Data Move			
LMOV A,[m]	Move Data Memory to ACC	2	None
LMOV [m],A	Move ACC to Data Memory	2Note	None
Bit Operation			
LCLR [m].i	Clear bit of Data Memory	2Note	None
LSET [m].i	Set bit of Data Memory	2Note	None

Mnemonic	Description	Cycles	Flag Affected
Branch			
LSZ [m]	Skip if Data Memory is zero	2Note	None
LSZA [m]	Skip if Data Memory is zero with data movement to ACC	2Note	None
LSNZ [m]	Skip if Data Memory is not zero	2Note	None
LSZ [m].i	Skip if bit i of Data Memory is zero	2Note	None
LSNZ [m].i	Skip if bit i of Data Memory is not zero	2Note	None
LSIZ [m]	Skip if increment Data Memory is zero	2Note	None
LSDZ [m]	Skip if decrement Data Memory is zero	2Note	None
LSIZA [m]	Skip if increment Data Memory is zero with result in ACC	2Note	None
LSDZA [m]	Skip if decrement Data Memory is zero with result in ACC	2Note	None
Table Read			
LTABRD [m]	Read table (specific page) to TBLH and Data Memory	3Note	None
LTABRDL [m]	Read table (last page) to TBLH and Data Memory	3Note	None
LITABRD [m]	Increment table pointer TBLP first and Read table (specific page) to TBLH and Data Memory	3Note	None
LITABRDL [m]	Increment table pointer TBLP first and Read table (last page) to TBLH and Data Memory	3Note	None
Miscellaneous			
LCLR [m]	Clear Data Memory	2Note	None
LSET [m]	Set Data Memory	2Note	None
LSWAP [m]	Swap nibbles of Data Memory	2Note	None
LSWAPA [m]	Swap nibbles of Data Memory with result in ACC	2	None

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

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

Instruction Definition

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

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

Operation ACC ← ACC + [m] + C

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

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

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

Operation [m] ← ACC + [m] + C

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

ADD A,[m] Add Data Memory to ACC

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

Operation ACC ← ACC + [m]

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

ADD A,x Add immediate data to ACC

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

Operation ACC ← ACC + x

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

ADDM A,[m] Add ACC to Data Memory

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

Operation [m] ← ACC + [m]

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

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

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

Operation ACC ← ACC "AND" [m]

Affected flag(s) Z

AND A,x Logical AND immediate data to ACC

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

Operation ACC ← ACC "AND" x

Affected flag(s) Z

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

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

Operation [m] ← ACC "AND" [m]

Affected flag(s) Z

CALL addr Subroutine call

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

Operation Stack ← Program Counter + 1
Program Counter ← addr

Affected flag(s) None

CLR [m] Clear Data Memory

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

Operation [m] ← 00H

Affected flag(s) None

CLR [m].i Clear bit of Data Memory

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

Operation [m].i ← 0

Affected flag(s) None

CLR WDT Clear Watchdog Timer

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

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CPL [m] Complement Data Memory

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

Operation [m] ← $\overline{[m]}$

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] − $\overline{C}$

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] − $\overline{C}$

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 (TBLP and TBHP) is moved to the specified Data Memory and the high byte moved to TBLH.

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

Affected flag(s) None

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

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

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

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

Operation ACC ← ACC ″XOR″ [m]

Affected flag(s) Z

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

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

Operation [m] ← ACC ″XOR″ [m]

Affected flag(s) Z

XOR A,x Logical XOR immediate data to ACC

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

Operation ACC ← ACC ″XOR″ x

Affected flag(s) Z

Extended Instruction Definition

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

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

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

Operation ACC ← ACC + [m] + C

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

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

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

Operation [m] ← ACC + [m] + C

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

LADD A,[m] Add Data Memory to ACC

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

Operation ACC ← ACC + [m]

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

LADDM A,[m] Add ACC to Data Memory

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

Operation [m] ← ACC + [m]

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

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

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

Operation ACC ← ACC ″AND″ [m]

Affected flag(s) Z

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

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

Operation [m] ← ACC ″AND″ [m]

Affected flag(s) Z

LCLR [m] Clear Data Memory

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

Operation [m] ← 00H

Affected flag(s) None

LCLR [m].i Clear bit of Data Memory

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

Operation [m].i ← 0

Affected flag(s) None

LCPL [m] Complement Data Memory

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

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

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] − $\overline{C}$

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] − $\overline{C}$

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

LSDZ [m] Skip if decrement Data Memory is 0

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

Operation [m] ← [m] − 1
Skip if [m]=0

Affected flag(s) None

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

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

Operation ACC ← [m] − 1
Skip if ACC=0

Affected flag(s) None

LSET [m] Set Data Memory

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

Operation [m] ← FFH

Affected flag(s) None

LSET [m].i Set bit of Data Memory

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

Operation [m].i ← 1

Affected flag(s) None

LSIZ [m] Skip if increment Data Memory is 0

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

Operation [m] ← [m] + 1
Skip if [m]=0

Affected flag(s) None

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

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

Operation ACC ← [m] + 1
Skip if ACC=0

Affected flag(s) None

LSNZ [m].i Skip if 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 three cycle instruction. If the result is 0 the program proceeds with the following instruction.

Operation Skip if [m].i ≠ 0

Affected flag(s) None

LSNZ [m] Skip if Data Memory is not 0

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

Operation Skip if [m] ≠ 0

Affected flag(s) None

LSUB A,[m] Subtract Data Memory from ACC

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

Operation ACC ← ACC − [m]

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

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

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

Operation [m] ← ACC − [m]

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

LSWAP [m] Swap nibbles of Data Memory

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

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s) None

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

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

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

Affected flag(s) None

LSZ [m] Skip if Data Memory is 0

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

Operation Skip if [m]=0

Affected flag(s) None

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

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

Operation ACC ← [m]
Skip if [m]=0

Affected flag(s) None

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

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

Operation Skip if [m].i=0

Affected flag(s) None

LTABRD [m] Read table (specific page) to TBLH and Data Memory

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

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

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

Operation ACC ← ACC ″XOR″ [m]

Affected flag(s) Z

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

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

Operation [m] ← ACC ″XOR″ [m]

Affected flag(s) Z

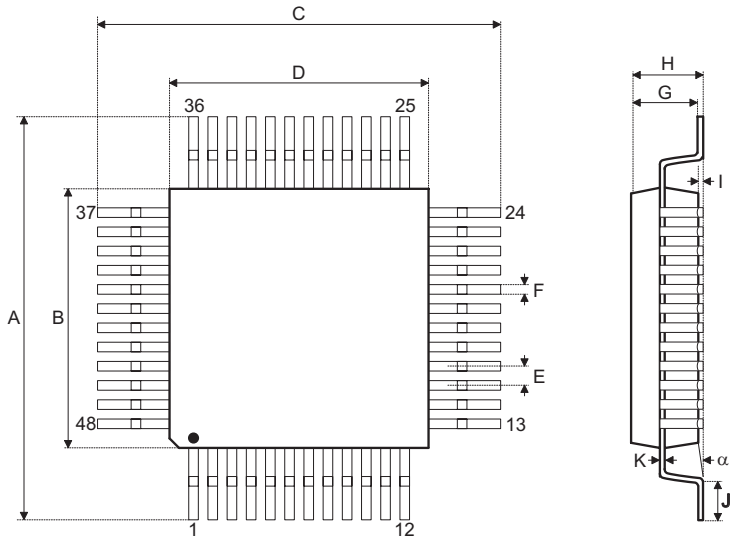
Package Information

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

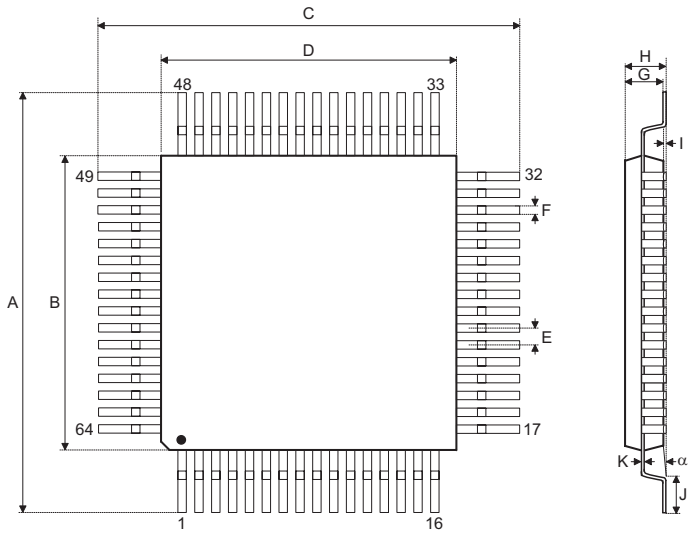
48-pin LQFP (7mm×7mm) Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A		0.354 BSC	
B		0.276 BSC	
C		0.354 BSC	
D		0.276 BSC	
E		0.020 BSC	
F	0.007	0.009	0.011
G	0.053	0.055	0.057
H	—	—	0.063
I	0.002	—	0.006
J	0.018	0.024	0.030
K	0.004	—	0.008
α	0°	—	7°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A		9.00 BSC	
B		7.00 BSC	
C		9.00 BSC	
D		7.00 BSC	
E		0.50 BSC	
F	0.17	0.22	0.27
G	1.35	1.40	1.45
H	—	—	1.60
I	0.05	—	0.15
J	0.45	0.60	0.75
K	0.09	—	0.20
α	0°	—	7°

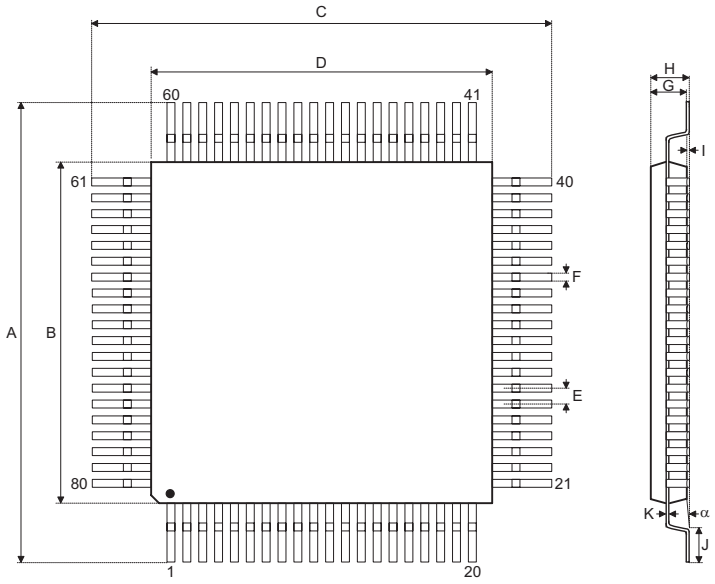
64-pin LQFP (7mm×7mm) Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A		0.354 BSC	
B		0.276 BSC	
C		0.354 BSC	
D		0.276 BSC	
E		0.016 BSC	
F	0.005	0.007	0.009
G	0.053	0.055	0.057
H	—	—	0.063
I	0.002	—	0.006
J	0.018	0.024	0.030
K	0.004	—	0.008
α	0°	—	7°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A		9.00 BSC	
B		7.00 BSC	
C		9.00 BSC	
D		7.00 BSC	
E		0.40 BSC	
F	0.13	0.18	0.23
G	1.35	1.40	1.45
H	—	—	1.60
I	0.05	—	0.15
J	0.45	0.60	0.75
K	0.09	—	0.20
α	0°	—	7°

80-pin LQFP (10mm×10mm) Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A		0.472 BSC	
B		0.394 BSC	
C		0.472 BSC	
D		0.394 BSC	
E		0.016 BSC	
F	0.005	0.007	0.009
G	0.053	0.055	0.057
H	—	—	0.063
I	0.002	—	0.006
J	0.018	0.024	0.030
K	0.004	—	0.008
α	0°	—	7°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A		12.00 BSC	
B		10.00 BSC	
C		12.00 BSC	
D		10.00 BSC	
E		0.40 BSC	
F	0.13	0.18	0.23
G	1.35	1.40	1.45
H	—	—	1.60
I	0.05	—	0.15
J	0.45	0.60	0.75
K	0.09	—	0.20
α	0°	—	7°