R-Type Blood Pressure Meter Flash MCU

HT45F3W

Revision: V1.50 Date: December 01, 2016

www.holtek.com

Table of Contents

Features **6**
CPU Features 6
Peripheral Features 6
General Description **7**
Block Diagram **7**
Pin Assignment **8**
Pin Description **9**
Absolute Maximum Ratings **11**
D.C. Characteristics (Charge Pump & Regulator off) **12**
A.C. Characteristics **13**
A/D Electrical Characteristics **14**
D/A Electrical Characteristics **15**
Charge Pump, Regulator and PGA Electrical Characteristics **15**
LCD D.C. Characteristics **16**
LVR Electrical Characteristics **17**
Power-on Reset Characteristics **17**
System Architecture **18**
Clocking and Pipelining 18
Program Counter 19
Stack 20
Arithmetic and Logic Unit – ALU 20
Flash Program Memory **21**
Structure 21
Special Vectors 21
Look-up Table 21
Table Program Example 22
In Circuit Programming – ICP 23
In Application Programming – IAP 24
On Chip Debug Support (OCDS) 31
Data Memory **32**
Structure 32
General Purpose Data Memory 33
Special Purpose Data Memory 33
LCD Memory 33
Special Function Register Description **35**
Indirect Addressing Register – IAR0, IAR1 35
Memory Pointers – MP0, MP1 35
Bank Pointer – BP 36
Accumulator – ACC 36

Program Counter Low Register – PCL 36
Look-up Table Registers – TBLP, TBHP, TBLH 37
Status Register – STATUS 37

System Control Register 39

EEPROM Data memory 41
EEPROM Data Memory Structure 41
EEPROM Registers 41

Oscillators 45
Oscillator Overview 45
System Clock Configurations 45
External Crystal/Ceramic Oscillator – HXT 46
External RC Oscillator – ERC 47
Internal RC Oscillator – HIRC 47
External 32.768kHz Crystal Oscillator – LXT 48
LXT Oscillator Low Power Function 49
Internal 32kHz Oscillator – LIRC 49
Supplementary Oscillators 49

Operating Modes and System Clocks 50
System Clocks 50
System Operation Modes 52
Control Register 53
Fast Wake – up 55
Operating Mode Switching 56
Standby Current Considerations 60
Wake-up 60
Programming Considerations 61

Watchdog Timer 62
Watchdog Timer Clock Source 62
Watchdog Timer Control Register 62
Watchdog Timer Operation 63

Reset and Initialisation 65
Reset Functions 65
Reset Initial Conditions 67

Input/Output Ports 71
Pull-high Resistors 71
Port A Wake-up 72
I/O Port Control Registers 73
SCKO Function 74
I/O Pin Structures 74
Programming Considerations 75

Timer Modules – TM 76
Introduction 76
TM Operation 76

TM Clock Source 77
TM Interrupts 77
TM External Pins 77
Programming Considerations 78

Compact Type TM – CTM 79
Compact TM Operation 79
Compact Type TM Register Description 80
Compact Type TM Operating Modes 84

Standard Type TM – STM 90
Standard TM Operation 90
Standard Type TM Register Description 91
Standard Type TM Operating Modes 94

Analog to Digital Converter – ADC 104
A/D Overview 104
A/D Converter Data Registers – ADRL, ADRH 104
A/D Converter Control Registers – ADCR, ACSR, ANCSR 105
A/D Operation 107
Conversion Accumulation Mode 107
A/D Input Pins 108
Summary of A/D Conversion Steps 109
Programming Considerations 110
A/D Transfer Function 110
A/D Programming Example 111

Digital to Analog Converter – DAC 113
DAC Operation 113

Serial Interface – SPI0, SPI1 114
SPI Interface Operation 114
SPI Control Register 115
SPI Communication 119

Interrupts 120
Interrupt Registers 121
Interrupt Operation 125
External Interrupt 127
A/D Converter Interrupt 128
Multi-function Interrupt 128
Time Base Interrupt 128
SPI Interrupts 129
LVD Interrupt 130
EEPROM Interrupt 130
TM Interrupts 130
SCF Interrupt 130
Charge Pump Interrupt 131
Interrupt Wake-up Function 131
Programming Considerations 131

Low Voltage Detector – LVD 132
LVD Register 132
LVD Operation 133
Programmable Gain Amplifier and Switched Capacitor Filter 134
PGA Operation 134
Switched Capacitor Filter 135
PGA Control Register 136
Sensor Constant Current Generator Circuit 138
Constant Current Generator Operation 138
Pressure Sensor Constant Current Control Register 139
Charge Pump and Voltage Regulator 140
Operation 140
LCD Driver 143
LCD Display Memory 143
LCD Control Register 144
Clock Source 146
LCD Driver Output 146
LCD Voltage Source and Biasing 147
Programming Considerations 148
VDDSPI Pin Function 150
Configuration Option 151
Instruction Set 152
Introduction 152
Instruction Timing 152
Moving and Transferring Data 152
Arithmetic Operations 152
Logical and Rotate Operation 153
Branches and Control Transfer 153
Bit Operations 153
Table Read Operations 153
Other Operations 153
Instruction Set Summary 154
Table Conventions 154
Instruction Definition 156
Package Information 165
64-pin LQFP (7mm×7mm) Outline Dimensions 166

Features

CPU Features

- Operating Voltage:
  - f_{SYS} = 4MHz: 2.2V~5.5V
  - f_{SYS} = 8MHz: 3.0V~5.5V
  - f_{SYS} = 12MHz: 4.5~5.5V
- Up to 0.33μs instruction cycle with 12MHz system clock at V_{DD}= 5V
- Power down and wake-up functions to reduce power consumption
- Five oscillators:
  - External Crystal – HXT
  - External 32.768kHz Crystal – LXT
  - External RC – ERC
  - Internal RC – HIRC
  - Internal 32kHz RC – LIRC
- Multi-mode operation: NORMAL, SLOW, IDLE and SLEEP
- Fully integrated internal 8MHz oscillator requires no external components
- All instructions executed in one or two instruction cycles
- Table read instructions
- 63 powerful instructions
- Up to 12-level subroutine nesting
- Bit manipulation instruction

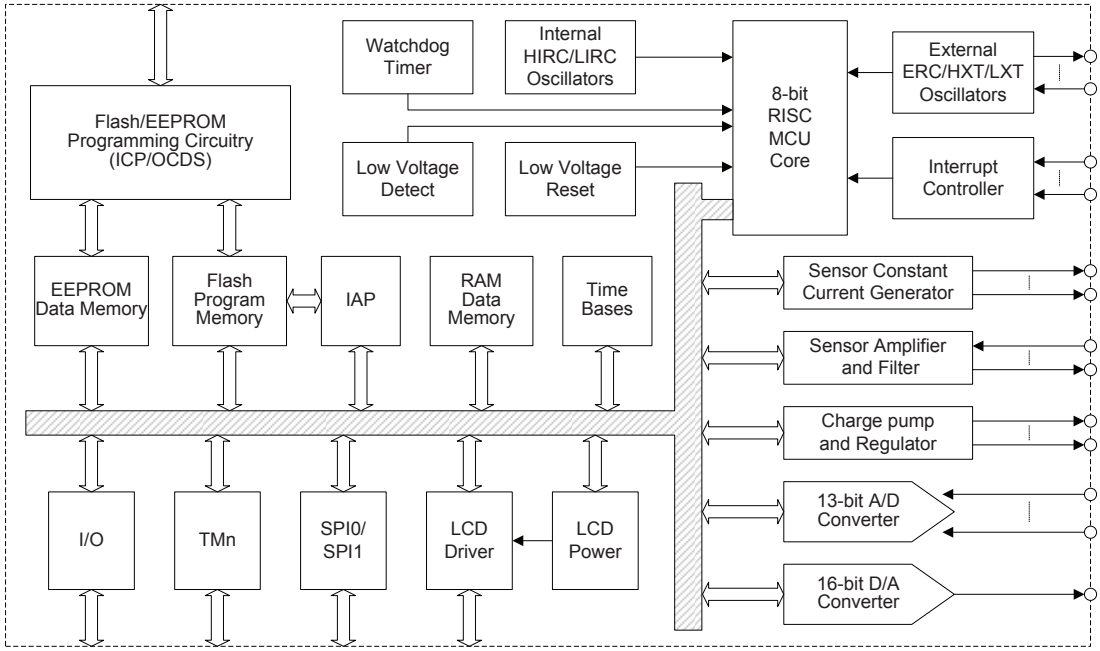
Peripheral Features

- Flash Program Memory: 16K × 16
- RAM Data Memory: 512 × 8
- True EEPROM Memory: 64 × 8
- Watchdog Timer function
- Up to 29 bidirectional I/O lines
- Two pin-shared external interrupts
- Multiple Timer Module for time measure, input capture, compare match output, PWM output or single pulse output function
- Two Serial SPI Interfaces
- Dual Time-Base functions for generation of fixed time interrupt signals
- 8-channel 13-bit resolution A/D converter
- One channel 16-bit D/A converter
- Low voltage reset function
- Low voltage detect function
- LCD driver function
- PGA and OPA Modules
- Charge pump and 3.3V Regulator for Analog Circuit
- I/O Power for SPI0 and SPI1
- Package types: 64-pin LQFP

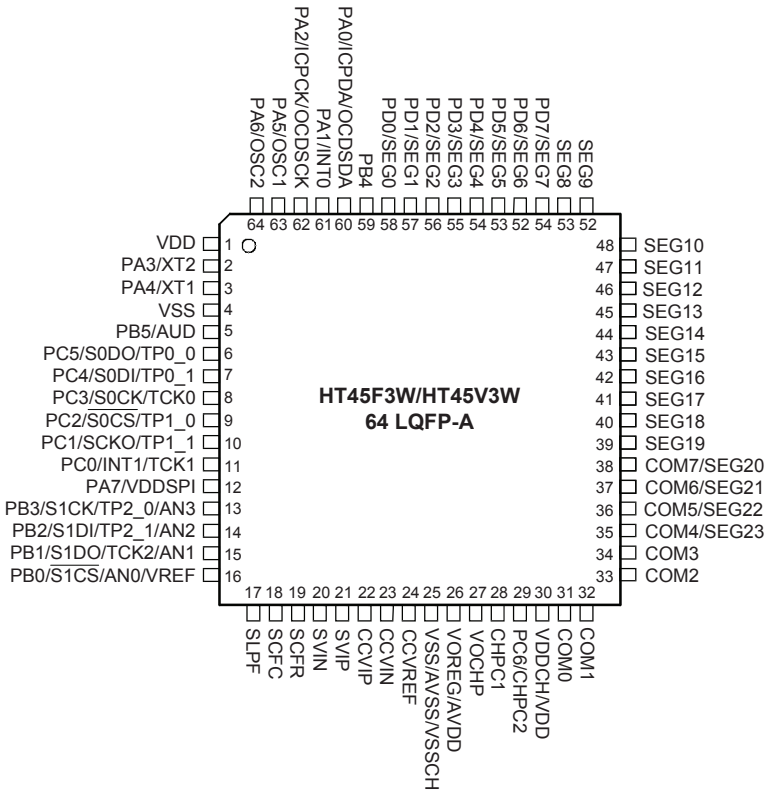
General Description

The device is an 8-bit high performance RISC architecture microcontroller specifically designed for Blood-Pressure Meter applications. As most of the functions required to implement a blood pressure meter application, such as amplifiers, charge pump etc. are integrated within the device, such applications can be implemented with a minimum of external components. The additional advantages of low power consumption, I/O flexibility, oscillator options, power down and wake-up functions, watchdog timer and low cost, enhance the versatility of the device to enable quick and cost efficient Blood-Pressure Meter applications.

Block Diagram



Pin Assignment



Note: The OCDSDA and OCDSCK pins are the OCDS dedicated pins and only available for the HT45V3W device which is the OCDS EV chip for the HT45F3W device.

Pin Description

Pin Name	Function	OPT	I/T	O/T	Description
PA0/ICPDA/ OCDSDA	PA0	PAPU PAWK	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	ICPDA	—	ST	CMOS	ICP Address/Data
	OCDSDA	—	ST	CMOS	OCDS Address/Data, for EV chip only.
PA1/INT0	PA1	PAPU PAWK	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	INT0	—	ST	—	External Interrupt 0
PA2/ICPCK/ OCDSCK	PA2	PAPU PAWK	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	ICPCK	—	ST	—	ICP Clock pin
	OCDSCK	—	ST	—	OCDS Clock pin, for EV chip only.
PA3/XT2	PA3	PAPU PAWK	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	XT2	CO	—	LXT	Low frequency crystal pin
PA4/XT1	PA4	PAPU PAWK	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	XT1	CO	LXT	—	Low frequency crystal pin
PA5/OSC1	PA5	PAPU PAWK	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	OSC1	CO	OSC	—	Oscillator pin
PA6/OSC2	PA6	PAPU PAWK	ST	CMOS	General purpose I/O. Register enabled pull-high and wake-up.
	OSC2	CO	—	OSC	Oscillator pin
PA7/VDDSPI	PA7	PAWK	ST	CMOS	General purpose I/O. Register enabled wake-up.
	VDDSPI	VDDSPICR	VI	—	SPI0/SPI1 external voltage input pin
PB0/$\overline{\text{S1CS}}$/ AN0/VREF	PB0	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-high. If the pull-high is enabled, the pin will be connected via a resistor to VOCHP. (It is recommended that the charge pump is enabled to prevent leakage.)
	$\overline{\text{S1CS}}$	SPI1C0	ST	—	SPI1 Slave Select
	AN0	ADCR	AN	—	A/D channel 0
	VREF	ACSR	VI	—	ADC reference voltage input pin
PB1/S1DO/ TCK2/AN1	PB1	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-high. If the pull-high is enabled, the pin will be connected via a resistor to VOCHP. (It is recommended that the charge pump is enabled to prevent leakage.)
	S1DO	SPI1C0	—	CMOS	SPI1 Data output
	TCK2	TM2C0	ST	—	TM2 input
	AN1	ADCR	AN	—	A/D channel 1
PB2/S1DI/ TP2_1/AN2	PB2	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-high. If the pull-high is enabled, the pin will be connected via a resistor to VOCHP. (It is recommended that the charge pump is enabled to prevent leakage.)
	S1DI	SPI1C0	ST	—	SPI1 Data input
	TP2_1	—	ST	CMOS	TM2 output
	AN2	ADCR	AN	—	A/D channel 2

Pin Name	Function	OPT	I/T	O/T	Description
PB3/S1CK/ TP2_0/AN3	PB3	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-high. If the pull-high is enabled, the pin will be connected via a resistor to VOCHP. (It is recommended that the charge pump is enabled to prevent leakage.)
	S1CK	SPI1C0	ST	—	SPI1 Serial Clock
	TP2_0	—	ST	CMOS	TM2 output
	AN3	ADCR	AN	—	A/D channel 3
PB4	PB4	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-high.
PB5/AUD	PB5	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-high.
	AUD	—	—	CMOS	Audio DAC output
PC0/INT1/ TCK1	PC0	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high. If the pull-high is enabled, the pin will be connected via a resistor to VOCHP. (It is recommended that the charge pump is enabled to prevent leakage.)
	INT1	—	ST	—	External Interrupt 1
	TCK1	TM1C0	ST	—	TM1 input
PC1/SCKO/ TP1_1	PC1	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high. If the pull-high is enabled, the pin will be connected via a resistor to VOCHP. (It is recommended that the charge pump is enabled to prevent leakage.)
	SCKO	SCKOC	—	CMOS	SCK output frequency pin
	TP1_1	—	ST	CMOS	TM1 output
PC2/$\overline{\text{S0CS}}$/ TP1_0	PC2	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high. If the pull-high is enabled, the pin will be connected via a resistor to VOCHP. (It is recommended that the charge pump is enabled to prevent leakage.)
	$\overline{\text{S0CS}}$	SPI0C0	ST	—	SPI0 Slave Select
	TP1_0	—	ST	CMOS	TM1 output
PC3/S0CK/ TCK0	PC3	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high. If the pull-high is enabled, the pin will be connected via a resistor to VOCHP. (It is recommended that the charge pump is enabled to prevent leakage.)
	S0CK	SPI0C0	ST	—	SPI0 Serial Clock
	TCK0	TM0C0	ST	—	TM0 input
PC4/S0DI/ TP0_1	PC4	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high. If the pull-high is enabled, the pin will be connected via a resistor to VOCHP. (It is recommended that the charge pump is enabled to prevent leakage.)
	S0DI	SPI0C0	ST	—	SPI0 Data input
	TP0_1	—	ST	CMOS	TM0 output
PC5/S0DO/ TP0_0	PC5	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high. If the pull-high is enabled, the pin will be connected via a resistor to VOCHP. (It is recommended that the charge pump is enabled to prevent leakage.)
	S0DO	SPI0C0	—	CMOS	SPI0 Data output
	TP0_0	—	ST	CMOS	TM0 output
PC6/CHPC2	PC6	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-high.
	CHPC2	—	—	—	Capacitor connected between CHPC1 and CHPC2
PD0/SEG0~ PD7/SEG7	PDn	PDPU	ST	CMOS	General purpose I/O. Register enabled pull-high.
	SEGn	SEGCR	—	SEG	LCD SEG0~SEG7 output

Pin Name	Function	OPT	I/T	O/T	Description
SEG8~SEG19	SEGn	—	—	SEG	LCD SEG8~SEG19 output
COM0~COM3	COMn	—	—	COM	LCD COM0~COM3 output
COM4/SEG23~ COM7/SEG20	COMn	LCDCTRL	—	COM	LCD COM4~COM7 output for 1/8 duty
	SEGn	LCDCTRL	—	SEG	LCD SEG23~SEG20 output for 1/4 and 1/6 duty
VDD	VDD	—	PWR	—	Power Supply
AVDD	AVDD	—	PWR	—	ADC Power Supply
VSS	VSS	—	PWR	—	Ground
AVSS	AVSS	—	PWR	—	ADC Ground
VSSCH	VSSCH	—	PWR	—	Charge Pump negative power supply. AVSS and VSSCH are bonded together
VDDCH	VDDCH	—	PWR	—	Charge Pump positive power supply. Connected to VDD in the application
CHPC1	CHPC1	—	—	—	Capacitor connected between CHPC1 and CHPC2
VOCHP	VOCHP	—	—	PWR	Charge Pump output
VOREG	VOREG	—	—	PWR	Regulator output. AVDD and VOREG are bonded together
CCVREF	CCVREF	—	—	—	VREF output pin – capacitor connected between this pin and VSS
CCVIP	CCVIP	—	—	—	OPA non inverting input pin
CCVIN	CCVIN	—	—	—	OPA inverting input pin
SVIP	SVIP	—	—	—	Pressure sensor positive input pin
SVIN	SVIN	—	—	—	Pressure sensor negative input pin
SLPF	SLPF	—	—	—	PGA2 input pin
SCFR	SCFR	—	—	—	PGA1 output pin.
SCFC	SCFC	—	—	—	SCF input pin

Note: I/T: Input type O/T: Output type

OPT: Optional by configuration option (CO) or register option

PWR: Power CO: Configuration option ST: Schmitt Trigger input

CMOS: CMOS output AN: Analog input pin VI: Voltage input

HXT: High frequency crystal oscillator

LXT: Low frequency crystal oscillator

Absolute Maximum Ratings

Supply Voltage ..... V_{SS}−0.3V to V_{SS}+6.0V
Input Voltage ..... V_{SS}−0.3V to V_{DD}+0.3V
Storage Temperature..... -50˚C to 125˚C
Operating Temperature..... -10˚C to 50˚C

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

D.C. Characteristics (Charge Pump & Regulator off)

Ta= 25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	f_{SYS}=4MHz	2.2	—	5.5	V
			f_{SYS}=8MHz	3.0	—	5.5	V
			f_{SYS}=12MHz	4.5	—	5.5	V
I_{DD1}	Operating Current (HXT/ERC)	3V	No load, f_{SYS}=4MHz ADC disable	—	0.8	1.2	mA
		5V		—	2.1	3.15	mA
I_{DD2}	Operating Current (HXT/ERC/HIRC)	3V	No load, f_{SYS}=8MHz ADC disable	—	1.4	2.1	mA
		5V		—	2.8	4.2	mA
I_{DD3}	Operating Current (HXT/ERC)	5V	No load, f_{SYS}=12MHz ADC disable	—	4	6	mA
I_{DD4}	Operating Current (HIRC+LXT, Slow Mode)	3V	No load, f_{SYS}=32768Hz ADC disable (LXT on LVR enable)	—	45	70	μA
		5V		—	60	90	μA
I_{STB}	Standby current (LIRC/LXT on)	3V	No load, System halt Ta=25°C	—	—	5	μA
		5V		—	—	10	μA
V_{IL}	Input Low Voltage for I/O Ports, TCKn and INTn	—	—	0	—	$0.2V_{DD}$	V
		5V	—	0	—	1.5	V
V_{IH}	Input High Voltage for I/O Ports, TCKn and INTn	5V	—	3.5	—	5	V
		—	—	$0.8V_{DD}$	—	V_{DD}	V
V_{LVD1}	Low Voltage Detector Voltage	—	LVD voltage 2.0V option	-5%	2.0	+5%	V
V_{LVD2}		—	LVD voltage 2.2V option		2.2		V
V_{LVD3}		—	LVD voltage 2.4V option		2.4		V
V_{LVD4}		—	LVD voltage 2.7V option		2.7		V
V_{LVD5}		—	LVD voltage 3.0V option		3.0		V
V_{LVD6}		—	LVD voltage 3.3V option		3.3		V
V_{LVD7}		—	LVD voltage 3.6V option		3.6		V
V_{LVD8}		—	LVD voltage 4.0V option		4.0		V
I_{OH1}	I/O source current (PA,PB,PC)	3V	V_{OH}=0.9V_{DD}	-2	-4	—	mA
		5V		-5	-10	—	mA
I_{OL1}	I/O sink current (PA,PB,PC)	3V	V_{OL}=0.1V_{DD}	4	8	—	mA
		5V		10	20	—	mA
I_{OL2}	PA7 sink current	5V	V_{OL}=0.1V_{DD}	2	3	—	mA
R_{PH}	Pull-high Resistance of I/O Ports	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ

A.C. Characteristics

Ta= 25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS}	System clock (HXT)	2.2V~5.5V	—	400	—	4000	kHz
		3.0V~5.5V	—	400	—	8000	kHz
		4.5V~5.5V	—	400	—	12000	kHz
f_{HIRC}	System clock (HIRC)	3.0V/5.0V	Ta=25°C	-2%	8	+2%	MHz
		3.0V/5.0V	Ta=0~70°C	-5%	8	+5%	MHz
		3.0V~5.5V	Ta=0~70°C	-8%	8	+8%	MHz
		3.0V~5.5V	Ta=-40~85°C	-12%	8	+12%	MHz
f_{ERC}	System clock (ERC)	5.0V	Ta=25°C, R=120KΩ	-2%	4	+2%	MHz
		5.0V	Ta=0~70°C, R=120KΩ	-5%	4	+5%	MHz
		5.0V	Ta=-40~85°C, R=120KΩ	-7%	4	+7%	MHz
		2.2V~5.5V	Ta=-40~85°C, R=120KΩ	-11%	4	+11%	MHz
f_{LXT}	System clock (LXT)	—	—	—	32768	—	Hz
f_{TIMER}	Timer input frequency (TCKn)	2.2V~5.5V	—	0	—	4000	kHz
		3.0V~5.5V	—	0	—	8000	kHz
		4.5V~5.5V	—	0	—	12000	kHz
t_{WDTOSC}	Watchdog oscillator period	3.0V	—	45	90	180	μs
		5.0V	—	32	65	130	μs
t_{SST}	System start-up timer period	—	Wake-up from HALT	—	1024	—	t_{SYS}
t_{INT}	Interrupt pulse width	—	—	1	—	—	μs

A/D Electrical Characteristics

Ta= 25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
AV_{DD}	Analog Operating Voltage	—	$V_{REF}=AV_{DD}$	2.7	—	5.5	V
V_{AD}	A/D Input Voltage	—	—	0	—	AV_{DD}/ V_{REF}	V
V_{REF}	A/D Input Reference Voltage	—	AV_{DD}=3V	2	—	AV_{DD} +0.1	V
		—	AV_{DD}=5V	2	—	AV_{DD} +0.1	V
DNL	Differential Non-linearity	3V	$V_{REF}=AV_{DD}=V_{DD}$, t_{AD}=0.5μs	-2	—	+2	LSB
		5V		-2	—	+2	LSB
		5V	Regulator on, charge pump off, $V_{REF}=AV_{DD}=V_{OREG}$, t_{AD}=0.5μs	-2	—	+2	LSB
		3V	Regulator & charge pump on, $V_{REF}=AV_{DD}=V_{OREG}$, t_{AD}=0.5μs	-2	—	+2	LSB
INL	Integral Non-linearity	3V	$V_{REF}=AV_{DD}=V_{DD}$, t_{AD}=0.5μs	-4	—	+4	LSB
		5V		-4	—	+4	LSB
		5V	Regulator on, charge pump off, $V_{REF}=AV_{DD}=V_{OREG}$, t_{AD}=0.5μs	-4	—	+4	LSB
		3V	Regulator & charge pump on, $V_{REF}=AV_{DD}=V_{OREG}$, t_{AD}=0.5μs	-4	—	+4	LSB
I_{ADC}	Additional Power Consumption if A/D Converter is used	3V	No load, t_{AD}=0.5μs	—	0.5	—	mA
		5V		—	0.6	—	mA
t_{AD}	A/D Clock Period	2.7~5.5V	—	0.5	—	10	μs
t_{ADC}	A/D Conversion Time	2.7~5.5V	12-bit A/D mode (ACCM=0)	—	16	—	t_{AD}
t_{ON2ST}	A/D on to ADC Start	2.7~5.5V	—	2	—	—	us

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

D/A Electrical Characteristics

Ta= 25˚C

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

Charge Pump, Regulator and PGA Electrical Characteristics

Ta= 25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
Charge Pump and Regulator							
V_{CHPI}	Input Voltage	—	Charge Pump on	2.2	—	3.6	V
			Charge Pump off	3.7	—	6.0	V
V_{REGO}	Output Voltage	—	No load Ta=25°C & 85°C	3.0	3.3	3.6	V
			No load Ta=-40°C	2.7	3.3	3.9	V
V_{REGDP1}	Regulator Output Voltage Drop (Compare with No load)	3.7~5.5V	Charge Pump off Current≤15mA + all circuits which are powered by the Regulator	—	100	—	mV
V_{REGDP2}	Regulator Output Voltage Drop (Compare with No load)	2.4~3.6V	Charge Pump on Current≤15mA + all circuits which are powered by the Regulator	—	100	—	mV
PGA, OP Amplifier and SCF PGA							
V_{ADOFF}	Input Offset Range	—	V_{REGO}=3.3V	—	500	800	μV
V_{RFGTC}	Reference Generator Temperature Coefficient	—	V_{REGO}=3.3V	—	50	—	Ppm/C
V_{ICMR}	Common Mode Input Range	—	Amplifier, no load	0.2	—	V_{REGO}-1	V
			Integrator, no load	1	—	V_{REGO}-0.2	V

LCD D.C. Characteristics

Ta= 25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{STB1}	Standby Current Only LCD on, LCD clock=4kHz	—	No load, Ta=25°C V_{LCD}=4.5, 1/3 bias (R_{BIAS}=144kΩ)	—	30	60	μA
I_{STB2}			No load, Ta=25°C V_{LCD}=4.5, 1/3 bias (R_{BIAS}=72kΩ)	—	60	120	μA
I_{STB3}			No load, Ta=25°C V_{LCD}=4.5, 1/3 bias (R_{BIAS}=36kΩ)	—	120	240	μA
I_{STB4}			No load, Ta=25°C V_{LCD}=4.5, 1/3 bias (R_{BIAS}=18kΩ)	—	240	480	μA
I_{STB5}			No load, Ta=25°C V_{LCD}=4.5, 1/4 bias (R_{BIAS}=192kΩ)	—	25	50	μA
I_{STB6}			No load, Ta=25°C V_{LCD}=4.5, 1/4 bias (R_{BIAS}=96kΩ)	—	45	90	μA
I_{STB7}			No load, Ta=25°C V_{LCD}=4.5, 1/4 bias (R_{BIAS}=48kΩ)	—	90	180	μA
I_{STB8}			No load, Ta=25°C V_{LCD}=4.5, 1/4 bias (R_{BIAS}=24kΩ)	—	180	360	μA
I_{STB9}			No load, Ta=-40~85°C V_{LCD}=4.5, 1/3 bias (R_{BIAS}=144kΩ)	—	45	105	μA
I_{STB10}			No load, Ta=-40~85°C V_{LCD}=4.5, 1/3 bias (R_{BIAS}=72kΩ)	—	90	180	μA
I_{STB11}			No load, Ta=-40~85°C V_{LCD}=4.5, 1/3 bias (R_{BIAS}=36kΩ)	—	180	360	μA
I_{STB12}			No load, Ta=-40~85°C V_{LCD}=4.5, 1/3 bias (R_{BIAS}=18kΩ)	—	360	720	μA
I_{STB13}			No load, Ta=-40~85°C V_{LCD}=4.5, 1/4 bias (R_{BIAS}=192kΩ)	—	37.5	75	μA
I_{STB14}			No load, Ta=-40~85°C V_{LCD}=4.5, 1/4 bias (R_{BIAS}=96kΩ)	—	67.5	135	μA
I_{STB15}			No load, Ta=-40~85°C V_{LCD}=4.5, 1/4 bias (R_{BIAS}=48kΩ)	—	145	290	μA
I_{STB16}			No load, Ta=-40~85°C V_{LCD}=4.5, 1/4 bias (R_{BIAS}=24kΩ)	—	270	540	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

LVR Electrical Characteristics

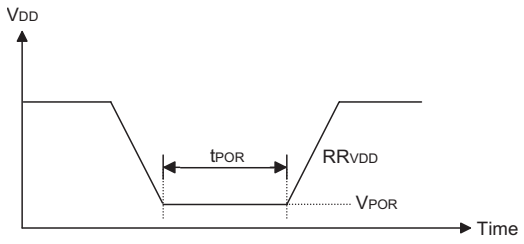
Ta= 25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{LVR1}	Low Voltage Reset Voltage	—	LVR enable, 2.10V option	-5%	2.1	+5%	V
V_{LVR2}			LVR enable, 2.55V option		2.55		V
V_{LVR3}			LVR enable, 3.15V option		3.15		V
V_{LVR4}			LVR enable, 3.80V option		3.8		V
t_{LVR}	Low Voltage Width to Reset	—	—	120	240	480	μs
t_{SRESET}	Software Reset Width to Reset	—	—	45	90	120	μs

Power-on Reset Characteristics

Ta= 25°C

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

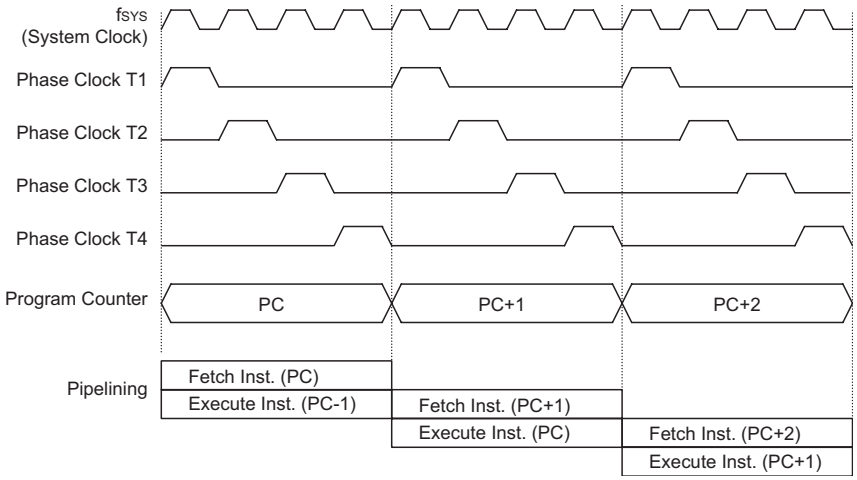


System Architecture

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

Clocking and Pipelining

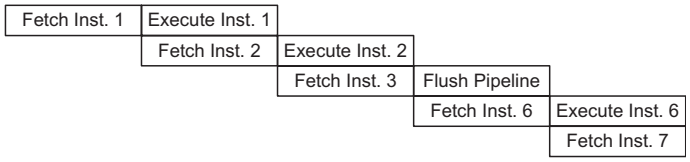
The main system clock, derived from either a HXT, LXT, HIRC, LIRC or ERC oscillator is subdivided into four internally generated non-overlapping clocks, T1~T4. The Program Counter is incremented at the beginning of the T1 clock during which time a new instruction is fetched. The remaining T2~T4 clocks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



System Clocking and Pipelining

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

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



Instruction Fetching

Program Counter

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

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

Program Counter	
Program Counter High Byte	**PCL Register**
PMBP0, PC12~PC8	PCL7~PCL0

Program Counter

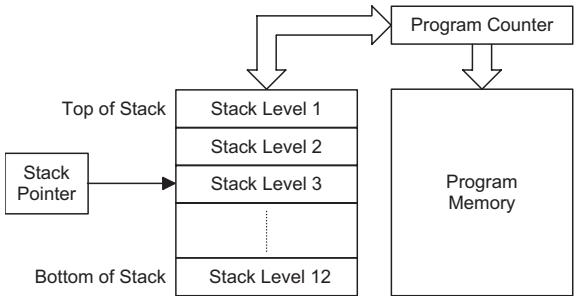
The lower byte of the Program Counter, known as the Program Counter Low register or PCL, is available for program control and is a readable and writeable register. By transferring data directly into this register, a short program jump can be executed directly. However, as only this low byte is available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

Stack

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

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

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



Arithmetic and Logic Unit – ALU

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

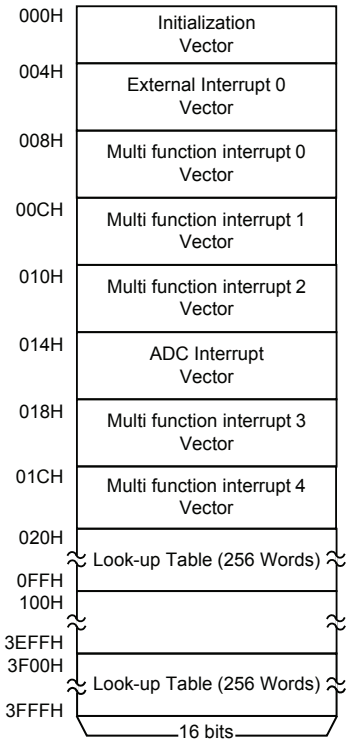
- Arithmetic operations: ADD, ADDM, ADC, ADCM, SUB, SUBM, SBC, SBCM, DAA
- Logic operations: AND, OR, XOR, ANDM, ORM, XORM, CPL, CPLA
- Rotation RRA, RR, RRCA, RRC, RLA, RL, RLCA, RLC
- Increment and Decrement INCA, INC, DECA, DEC
- Branch decision, JMP, SZ, SZA, SNZ, SIZ, SDZ, SIZA, SDZA, CALL, RET,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

Flash Program Memory

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

Structure

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



Program Memory Structure

Special Vectors

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

Look-up Table

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

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

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

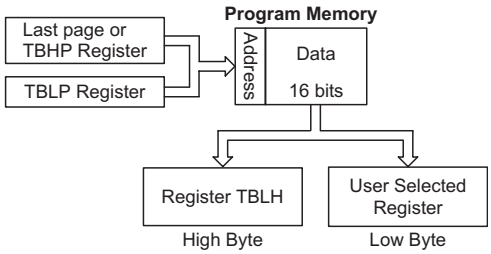


Table Program Example

The following example shows how the table pointer and table data is defined and retrieved from the microcontroller. This example uses raw table data located in the last page which is stored there using the "ORG" and "rombank" statements. The value at this ORG statement is "1F00H" which is located in ROM Bank 1 which refers to the start address of the last page within the 16K words Program Memory. The table pointer low byte register is setup here to have an initial value of "06H". This will ensure that the first data read from the data table will be at the Program Memory address "3F06H" or 6 locations after the start of the last page. Note that the value for the table pointer is referenced to the first address of the present page if the "TABRD [m]" instruction is being used. The high byte of the table data which in this case is equal to zero will be transferred to the TBLH register automatically when the "TABRD [m]" instruction is executed.

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

Table Read Program Example

```
rombank 1 code1
rombank 1 code1
ds .section 'data'
tempreg1 db ?          ; temporary register #1
tempreg2 db ?          ; temporary register #2
:
:
code0 .section 'code'
mov a,06h              ; initialise table pointer - note that this address is referenced
mov tblp,a             ; to the last page or the page that tbhp pointed
mov a,3fh              ; 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 referred by table pointer to tempreg1
                       ; data at program memory address 3F06H transferred to
                       ; tempreg1 and TBLH
dec tblp               ; reduce value of table pointer by one
tabrd tempreg2
tabrdl tempreg2        ; transfers value in table referenced by table pointer to tempreg2
                       ; data at program memory address 3F05H transferred to tempreg2
                       ; and TBLH
                       ; In this example the data "1AH" is transferred to tempreg1
                       ; and data "0FH" to tempreg2
                       ; the value "00H" will be transferred to the high byte register TBLH
:
:
code1 .section 'code'
org 1F00h              ; sets initial address of last page
dc 00Ah,00Bh,00Ch,00Dh,00Eh,00Fh,01Ah,01Bh
:
```

In Circuit Programming – ICP

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

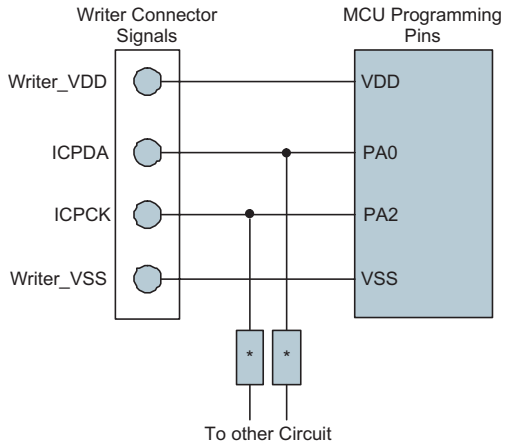
The Holtek Flash MCU to Writer Programming Pin correspondence table is as follows:

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

The Program Memory and EEPROM data Memory can both be programmed serially in-circuit using this 4-wire interface. Data is downloaded and uploaded serially on a single pin with an additional line for the clock. Two additional lines are required for the power supply. The technical details

regarding the in-circuit programming of the device are beyond the scope of this document and will be supplied in supplementary literature.

During the programming process, taking control of the PA0 and PA2 I/O pins for data and clock programming purposes. The user must there take care to ensure that no other outputs are connected to these two pins.



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

In Application Programming – IAP

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

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

In Application Programming Control Register

The Address register, FARL/FARH, and the Data register, FD0L/FD0H, FD1L/FD1H, FD2L/FD2H and FD3L/FD3H, located in Data Memory Bank 0, together with the Control register, FC0, FC1 and FC2, located in Data Memory Bank 1 are the corresponding Flash access registers for IAP. As indirect addressing is the only way to access the FC0, FC1 and FC2 registers, all read and write operations to the registers must be performed using the Indirect Addressing Register, IAR1, and the Memory Pointer, MP1. Because the FC0, FC1 and FC2 control registers are located at the address of 43H~45H in Data Memory Bank 1, the MP1 Memory Pointer must first be set to the value 43H~45H and the Bank Pointer set to “1”.

FC0 Register

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

Bit 7 **CFWEN**: Confirm the FWEN status
0: Flash ROM write disable
1: software write 1 no action
When this bit is clear by firmware, the IAP controller exits FWEN mode.The user can read this bit to confirm the FWEN status.

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

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

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

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

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

FC1 Register

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

FC2 Register

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

Bit 7~1 Unimplemented, read as "0"

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

FARL Register

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

Bit 7~0 **A7~A0**: The flash address[7:0]

FARH Register

Bit	7	6	5	4	3	2	1	0
Name	A15	A14	A13	A12	A11	A10	A9	A8
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 **A15~A8**: The flash address[15:8]

FD0L Register

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

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

FD0H Register

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

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

FD1L Register

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

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

FD1H Register

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

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

FD2L Register

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

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

FD2H Register

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

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

FD3L Register

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

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

FD3H Register

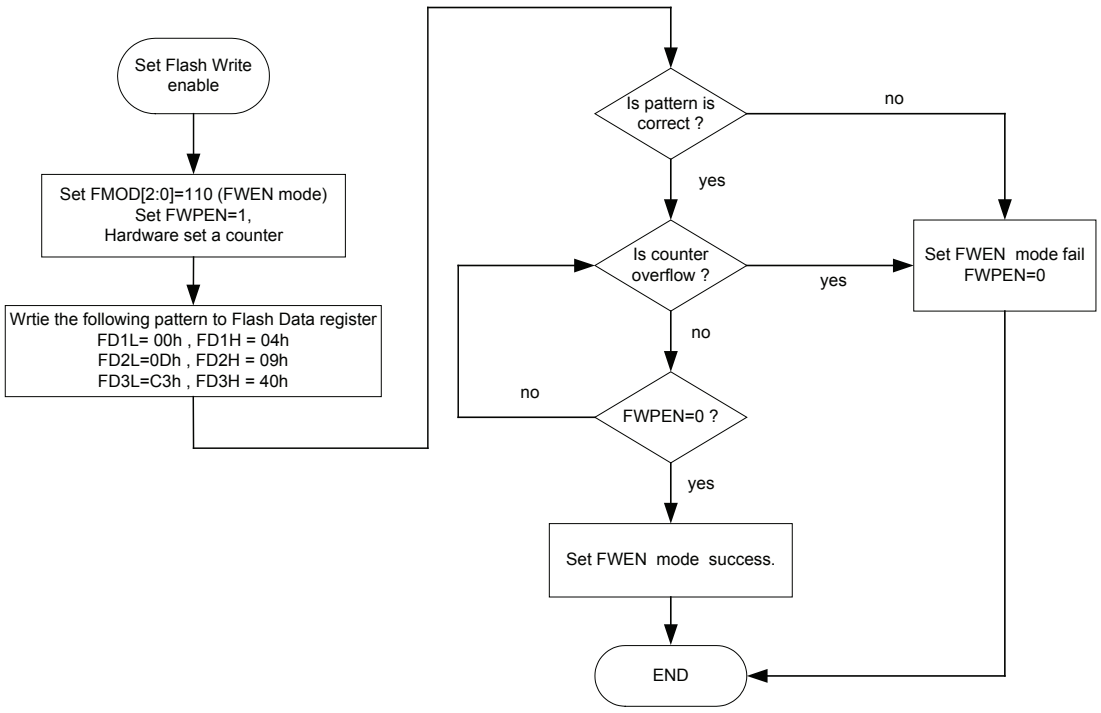
Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

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

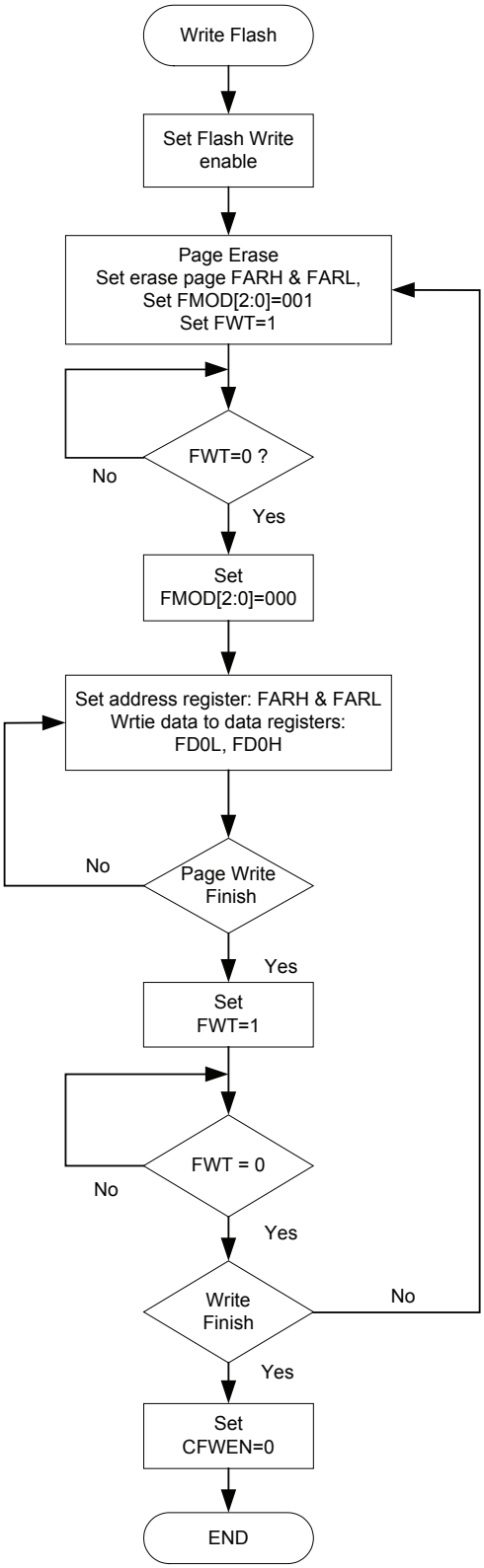
Set Flash Write enable (FWEN)

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

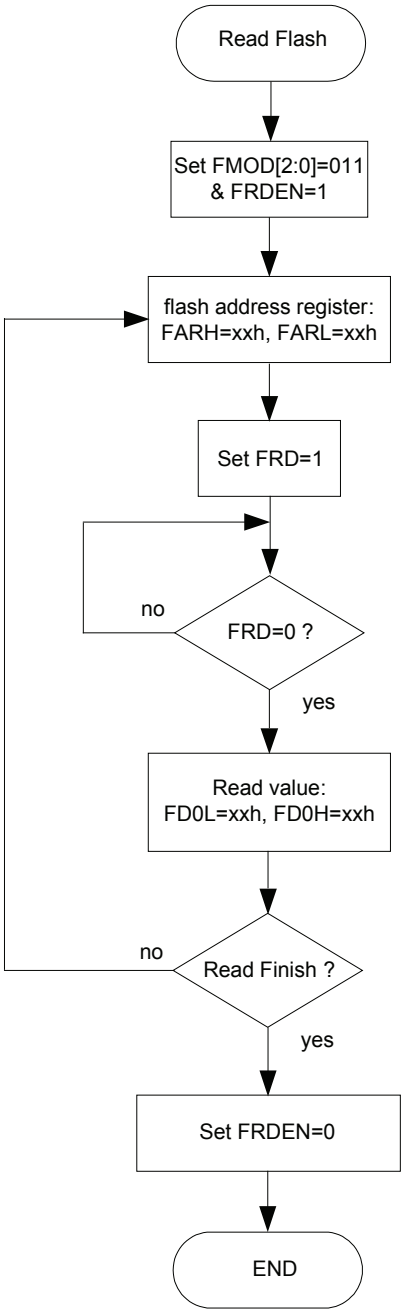
- Write 110 to the FMOD2~FMOD0 for setting FWEN mode.
- Set FWPEN bit to 1. The step 1 and step 2 can set simultaneously.
- The Hardware will start a 300μs counter to allow the user writing the correct pattern data to FD1L/FD1H~FD3L/FD3H. (The clock of 300μs counter is from LIRC.)
- Once 300μs counter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



Set Flash Write Enable Procedure



Write Flash Program Procedure



Read Flash Program Procedure

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

On Chip Debug Support (OCDS)

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

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

Data Memory

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

Divided into three sections, the first of these is an area of RAM where special function registers are located. These registers have fixed locations and are necessary for correct operation of the device. Many of these registers can be read from and written to directly under program control, however, some remain protected from user manipulation.

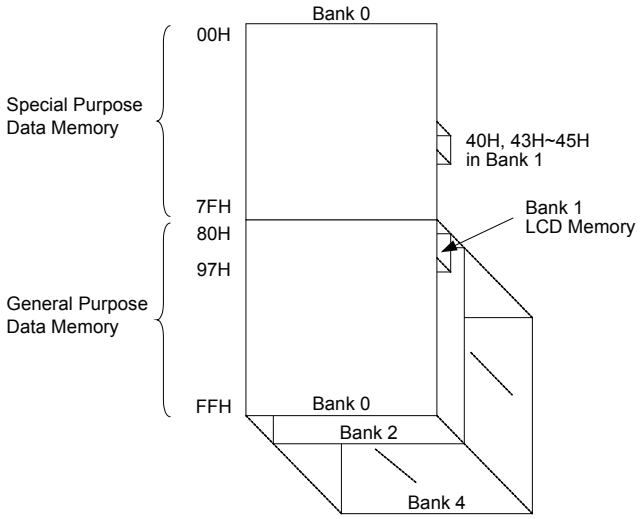
The second area of Data Memory is reserved for general purpose use. All locations within this area are read and write accessible under program control.

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

Structure

The Data Memory is subdivided into five banks, known as Bank 0 to Bank 4, all of which are implemented in 8-bit wide RAM. The Data Memory located in Bank 0 is subdivided into two sections, the Special Purpose Data Memory and the General Purpose Data Memory.

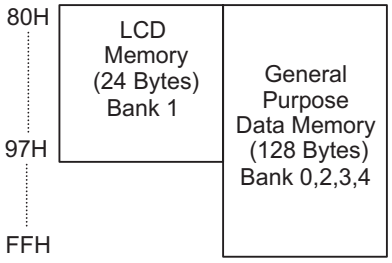
The start address of the Data Memory for the device is the address 00H. Registers which are common to all microcontrollers, such as ACC, PCL, etc., have the same Data Memory address. The LCD Memory is mapped into Bank 1. The Banks 2 to 4 contain only General Purpose Data Memory for the device. As the Special Purpose Data Memory registers are mapped into all bank areas, they can subsequently be accessed from any bank location.



Data Memory Structure

General Purpose Data Memory

All microcontroller programs require an area of read/write memory where temporary data can be stored and retrieved for use later. It is this area of RAM memory that is known as General Purpose Data Memory. This area of Data Memory is fully accessible by the user programing for both reading and writing operations. By using the "SET [m].i" and "CLR [m].i" 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.



General Purpose Data Memory

Special Purpose Data Memory

To ensure successful operation of the microcontroller, certain internal registers are implemented in the Data Memory area. These registers ensure correct operation of internal functions such as timers, interrupts, etc., as well as external functions such as I/O data control. The location of these registers within the Data Memory begins at the address 00H. Any unused Data Memory locations between these special function registers and the point where the General Purpose Memory begins is reserved and attempting to read data from these locations will return a value of 00H.

LCD Memory

The data to be displayed on the LCD is also stored in an area of fully accessible Data Memory. By writing to this area of RAM, the LCD display output can be directly controlled by the application program. As the LCD Memory exists in Bank 1, but have addresses which map into the General Purpose Data Memory, it is necessary to first ensure that the Bank Pointer bits, BP.2, BP.1 and BP.0 are set to the values "001", to ensure that the LCD Memory is accessed. The LCD Memory can only be accessed indirectly using the memory pointer MP1 and the indirect addressing register IAR1.

	Bank 0~4		Bank 0, 2,3,4	Bank 1
00H	IAR0	40H	Unused	EEC
01H	MP0	41H	EEA	Unused
02H	IAR1	42H	EED	Unused
03H	MP1	43H	Unused	FC0
04H	BP	44H	Unused	FC1
05H	ACC	45H	LVRC	FC2
06H	PCL	46H	TM0C0	
07H	TBLP	47H	TM0C1	
08H	TBLH	48H	TM0DL	
09H	TBHP	49H	TM0DH	
0AH	STATUS	4AH	TM0AL	
0BH	INTC0	4BH	TM0AH	
0CH	INTC1	4CH	TM1C0	
0DH	MFIC0	4DH	TM1C1	
0EH	MFIC1	4EH	TM1DL	
0FH	TBC	4FH	TM1DH	
10H	PA	50H	TM1AL	
11H	PAC	51H	TM1AH	
12H	PAPU	52H	TM2C0	
13H	PAWK	53H	TM2C1	
14H	PB	54H	TM2DL	
15H	PBC	55H	TM2DH	
16H	PBPU	56H	TM2AL	
17H	PC	57H	TM2AH	
18H	PCC	58H	TM2RP	
19H	PCPU	59H	SMOD0	
1AH	CTRL2	5AH	Unused	
1BH	CTRL3	5BH	SMOD1	
1CH	CTRL1	5CH	SCKOC	
1DH	Unused	5DH	CHPRC	
1EH	Unused	5EH	FARL	
1FH	VDDSPICR	5FH	FARH	
20H	ADRL	60H	FD0L	
21H	ADRH	61H	FD0H	
22H	ADCR	62H	FD1L	
23H	ACSR	63H	FD1H	
24H	WDTC	64H	FD2L	
25H	PD	65H	FD2H	
26H	PDC	66H	FD3L	
27H	PDPU	67H	FD3H	
28H	MFIC2	68H	Unused	
29H	MFIC3	69H	Unused	
2AH	MFIC4	6AH	Unused	
2BH	ANCSR	6BH	Unused	
2CH	DAL	6CH	Unused	
2DH	DAH	6DH	Unused	
2EH	VOL	6EH	Unused	
2FH	LVDC	6FH	Unused	
30H	CCVREFC	70H	Unused	
31H	PGAC0	71H	Unused	
32H	PGAC1	72H	Unused	
33H	PGAC2	73H	Unused	
34H	SCFC0	74H	Unused	
35H	SCFC1	75H	Unused	
36H	SCFCKD	76H	Unused	
37H	LCDCTRL	77H	Unused	
38H	SEGCR	78H	Unused	
39H	VLCDC	79H	Unused	
3AH	SPI0C0	7AH	Unused	
3BH	SPI0C1	7BH	Unused	
3CH	SPI0D	7CH	Unused	
3DH	SPI1C0	7DH	Unused	
3EH	SPI1C1	7EH	Unused	
3FH	SPI1D	7FH	Unused	

:Unimplemented, read as "0"

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

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

Memory Pointers – MP0, MP1

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

Indirect Addressing Program Example

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

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

Bank Pointer – BP

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

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

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

BP Register

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

Bit 7~6 Unimplemented, read as "0"

Bit 5 **PMBP0**: Select Program Memory Banks
0: Bank 0, Program Memory Address is from 0000H ~ 1FFFH
1: Bank 1, Program Memory Address is from 2000H ~ 3FFFH

Bit 4~3 Unimplemented, read as "0"

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

Accumulator – ACC

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

Program Counter Low Register – PCL

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

Look-up Table Registers – TBLP, TBHP, TBLH

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

Status Register – STATUS

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

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

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

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

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

STATUS Register

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

"x" unknown

Bit 7~6 Unimplemented, read as "0"

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

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

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

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

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

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

C is also affected by a rotate through carry instruction.

System Control Register

CTRL1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	TPIOS2	TPIOS1	TPIOS0	TPOS2	TPOS1	TPOS0
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 **TPIOS2**: output selection of I/O and TP2_0/TP2_1
0: Normal I/O
1: TP2_0/TP2_1 (timer function)

Bit 4 **TPIOS1**: output selection of I/O and TP1_0/TP1_1
0: Normal I/O
1: TP1_0/TP1_1 (timer function)

Bit 3 **TPIOS0**: output selection of I/O and TP0_0/TP0_1
0: Normal I/O
1: TP0_0/TP0_1 (timer function)

Bit 2 **TPOS2**: output selection of TP2_0 and TP2_1
0: TP2_0
1: TP2_1

Bit 1 **TPOS1**: output selection of TP1_0 and TP1_1
0: TP1_0
1: TP1_1

Bit 0 **TPOS0**: output selection of TP0_0 and TP0_1
0: TP0_0
1: TP0_1

Note: 1. Priority of Pin Sharing Functions: SPI > Timer > I/O
2. Procedure for SPI Pin Sharing Functions: SPI0EN=1 or SPI1EN=1
3. Procedure for Timer Pin Sharing Functions: SPI0EN=0 or SPI1EN=0, and TPIOS0=1 or TPIOS1=1 or TPIOS2=1 (Note: When TPIOS0 is set to 1, the output of timer is TP0_1. The pin TP0_0 could be SPI or I/O functions. If TP0_0 be used SPI function, the SPI0EN is set to 1.)

CTRL2 Register

Bit	7	6	5	4	3	2	1	0
Name	DACEN	CPDEN	INT1ES1	INT1ES0	INT0ES1	INT0ES0	CPF	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	—
POR	0	0	1	0	1	0	0	—

Bit 7 **DACEN**: DAC disable/enable control
0: DAC disable
1: DAC enable
The DAC circuit is disabled when system enters the HALT mode.

Bit 6 **CPDEN**: Charge pump input voltage detector circuit disable/enable control
0: disable
1: enable
If CPDEN=0, the Charge pump voltage input detector circuit is disabled, CPF will remain at 0, and the detector circuit consumes no power.

Bit 5~4 **INT1ES1, INT1ES0**: external interrupt 1 edge selection
00: disable
01: rising edge trigger
10: falling edge trigger
11: dual edge trigger

Bit 3~2 **INT0ES1, INT0ES0**: external interrupt 0 edge selection
00: disable
01: rising edge trigger
10: falling edge trigger
11: dual edge trigger

Bit 1 **CPF**: Charge pump input voltage detector flag
0: Charge pump input voltage is above 3.8V
1: Charge pump input voltage is equal to or under 3.8V
When CPDEN and REGCEN are enabled, the CPF will be working.

Bit 0 Unimplemented, read as "0"

CTRL3 Register

Bit	7	6	5	4	3	2	1	0
Name	SCFVREF4	SCFVREF3	SCFVREF2	SCFVREF1	SCFVREF0	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

Bit 7~3 **SCFVREF4~SCFVREF0**: SCF Offset Voltage selected
SCF Offset Voltage = ([SCFVREF4:0] / 32) * VOREG

Bit 2~0 Unimplemented, read as "0"

EEPROM Data memory

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

EEPROM Data Memory Structure

The EEPROM Data Memory capacity is 64x8 bits for the device. 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 Bank 0 and a single control register in Bank 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 Bank 0, they can be directly accessed in the same was as any other Special Function Register. The EEC register however, being located in Bank1, cannot be directly addressed directly and can only be read from or written to indirectly using the MP1 Memory Pointer and Indirect Addressing Register, IAR1. Because the EEC control register is located at address 40H in Bank 1, the MP1 Memory Pointer must first be set to the value 40H and the Bank Pointer register, BP, set to the value, 01H, before any operations on the EEC register are executed.

EEPROM Register List

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

EEA Register

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	—	—	x	x	x	x	x	x

"x" unknown

Bit 7~6 Unimplemented, read as "0"

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

EED Register

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

"x" unknown

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

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

Write Protection

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

EEPROM Interrupt

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

Programming Considerations

Care must be taken that data is not inadvertently written to the EEPROM. Protection can be enhanced by ensuring that the Write Enable bit is normally cleared to zero when not writing. Also the Bank Pointer could be normally cleared to zero as this would inhibit access to Bank 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, 040H              ; setup memory pointer MP1
MOV   MP1, A               ; MP1 points to EEC register
MOV   A, 01H               ; setup Bank Pointer
MOV   BP, A
SET   IAR1.1               ; set RDEN bit, enable read operations
SET   IAR1.0               ; start Read Cycle - set RD bit
BACK:
SZ IAR1.0                  ; check for read cycle end
JMP   BACK
CLR   IAR1                 ; disable EEPROM read/write
CLR   BP
MOV   A, EED               ; move read data to register
MOV   READ_DATA, A
```

  - Writing Data to the EEPROM – polling method

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

Oscillators

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

Oscillator Overview

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

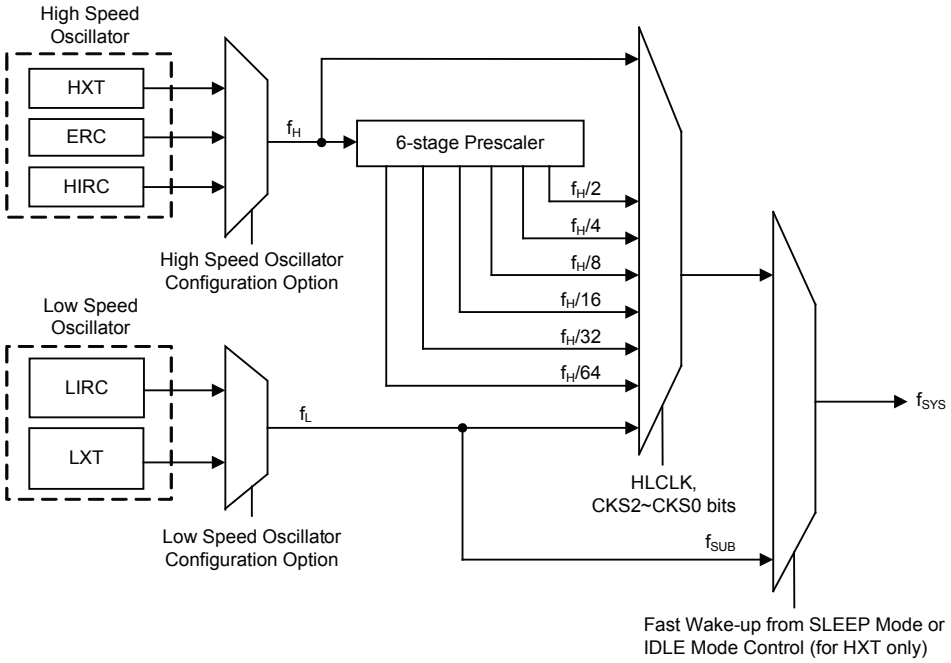
Type	Name	Freq.	Pins
External Crystal	HXT	400kHz~12MHz	OSC1/OSC2
External RC	ERC	4MHz~12MHz	OSC1
Internal High Speed RC	HIRC	8MHz	—
External Low Speed Crystal	LXT	32.768kHz	XT1/XT2
Internal Low Speed RC	LIRC	32kHz	—

Oscillator Types

System Clock Configurations

There are five methods of generating the system clock, three high speed oscillators and two low speed oscillators. The high speed oscillators are the external crystal/ceramic oscillator, external RC oscillator and the internal 8MHz RC oscillator. The two low speed oscillators are the internal 32kHz RC oscillator and the external 32.768kHz crystal oscillator. Selecting whether the low or high speed oscillator is used as the system oscillator is implemented using the HLCLK bit and CKS2~CKS0 bits in the SMOD0 register and as the system clock can be dynamically selected.

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

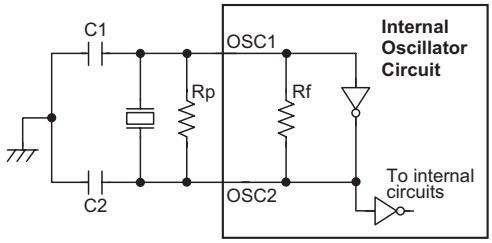


System Clock Configurations

External Crystal/Ceramic Oscillator – HXT

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

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



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

Crystal/Resonator Oscillator – HXT

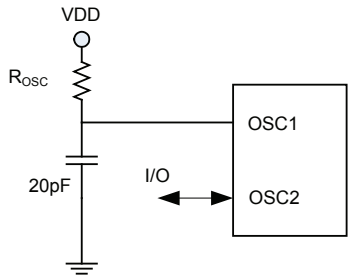
Crystal Oscillator C1 and C2 Values		
Crystal Frequency	C1	C2
12MHz	0pF	0pF
8MHz	0pF	0pF
4MHz	0pF	0pF
1MHz	100pF	100pF
Note: C1 and C2 values are for guidance only.		

Crystal Recommended Capacitor Values

External RC Oscillator – ERC

Using the ERC oscillator only requires that a resistor, with a value between 56kΩ and 2.4MΩ, is connected between OSC1 and VDD, and a capacitor is connected between OSC1 and ground, providing a low cost oscillator configuration. It is only the external resistor that determines the oscillation frequency; the external capacitor has no influence over the frequency and is connected for stability purposes only. Device trimming during the manufacturing process and the inclusion of internal frequency compensation circuits are used to ensure that the influence of the power supply voltage, temperature and process variations on the oscillation frequency are minimised. As a resistance/frequency reference point, it can be noted that with an external 120kΩ resistor connected and with a 5V voltage power supply and temperature of 25°C degrees, the oscillator will have a frequency of 4MHz within a tolerance of 2%. Here only the OSC1 pin is used, which is shared with I/O pin PA5, leaving pin PA6 free for use as a normal I/O pin.

For oscillator stability and to minimise the effects of noise and crosstalk, it is important to locate the capacitor and resistoras close to the MCU as possible.



External RC Oscillator – ERC

Internal RC Oscillator – HIRC

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

External 32.768kHz Crystal Oscillator – LXT

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

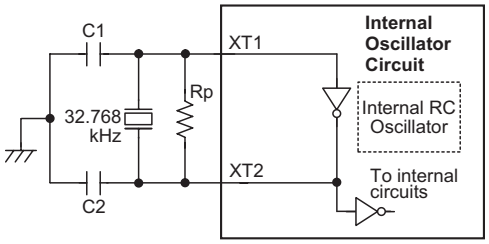
When the microcontroller enters the SLEEP or IDLE Mode, the system clock is switched off to stop microcontroller activity and to conserve power. However, in many microcontroller applications it may be necessary to keep the internal timers operational even when the microcontroller is in the SLEEP or IDLE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

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

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

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

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



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

External LXT Oscillator

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

32.768kHz Crystal Recommended Capacitor Values

LXT Oscillator Low Power Function

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

LXTLP Bit	LXT Mode
0	Quick Start
1	Low-power

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

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

Internal 32kHz Oscillator – LIRC

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

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

Operating Modes and System Clocks

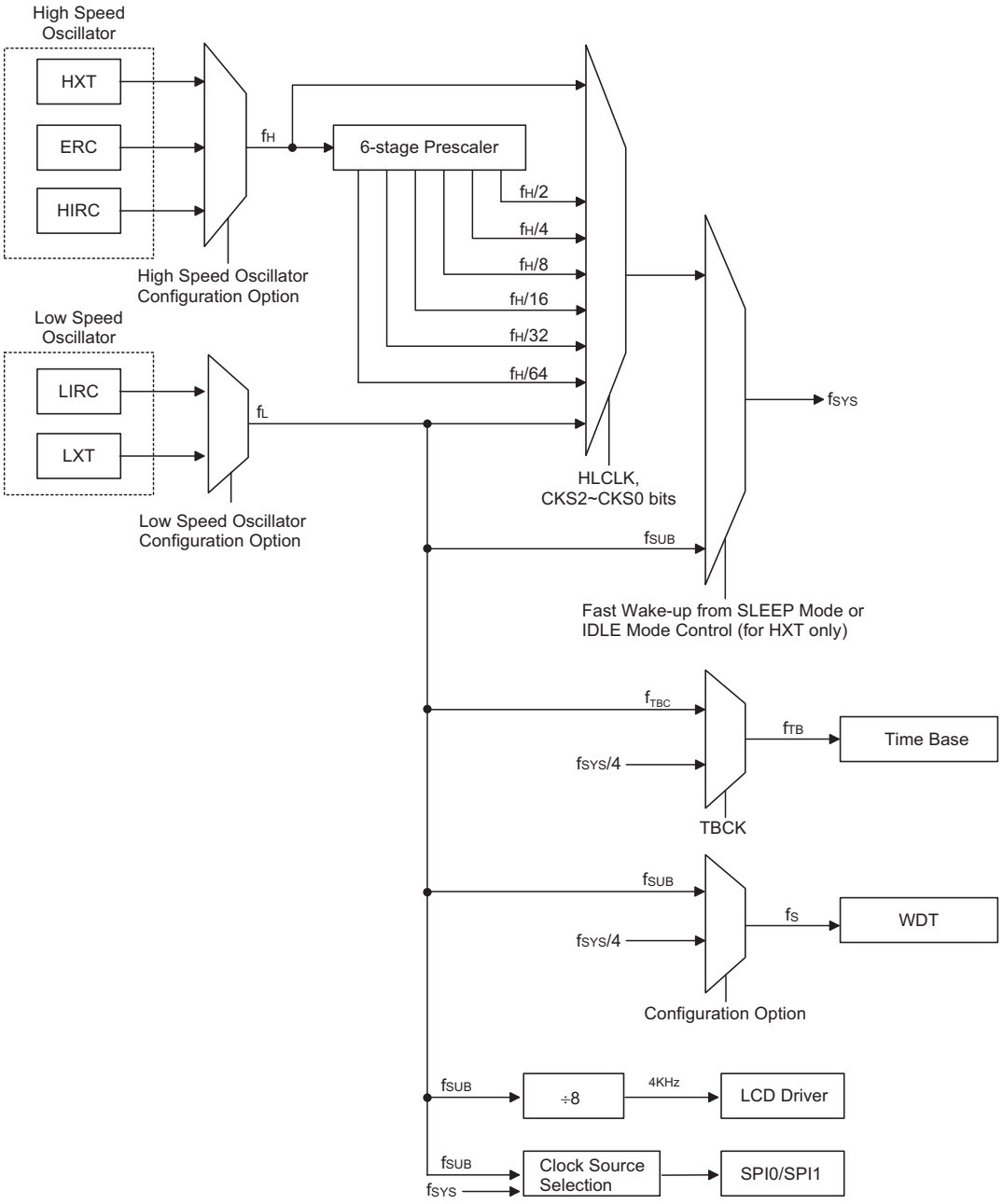
Present day applications require that their microcontrollers have high performance but often still demand that they consume as little power as possible, conflicting requirements that are especially true in battery powered portable applications. The fast clocks required for high performance will by their nature increase current consumption and of course vice-versa, lower speed clocks reduce current consumption. As Holtek has provided the device 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 device has many different clock sources for both the CPU and peripheral function operation. By providing the user with a wide range of clock options using configuration options and register programming, a clock system can be configured to obtain maximum application performance.

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

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



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

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

System Operation Modes

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

Operating Mode	Description				
	CPU	f_{SYS}	f_{SUB}	f_S	f_{TBC}
NORMAL Mode	on	f_H~f_H/64	on	on	on
SLOW Mode	on	f_L	on	on	on
IDLE0 Mode	off	off	on	on/off	on
IDLE1 Mode	off	on	on	on	on
SLEEP0 Mode	off	off	off	off	off
SLEEP1 Mode	off	off	on	on	off

NORMAL Mode

As the name suggests this is one of the main operating modes where the microcontroller has all of its functions operational and where the system clock is provided by one of the high speed oscillators. This mode operates allowing the microcontroller to operate normally with a clock source will come from one of the high speed oscillators, either the HXT, ERC or HIRC oscillators. The high speed oscillator will however first be divided by a ratio ranging from 1 to 64, the actual ratio being selected by the CKS2~CKS0 and HLCLK bits in the SMOD0 register. Although a high speed oscillator is used, running the microcontroller at a divided clock ratio reduces the operating current.

SLOW Mode

This is also a mode where the microcontroller operates normally although now with a slower speed clock source. The clock source used will be from one of the low speed oscillators, either the LXT or the LIRC. Running the microcontroller in this mode allows it to run with much lower operating currents. In the SLOW Mode, the f_H is off.

SLEEP0 Mode

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

SLEEP1 Mode

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

IDLE0 Mode

The IDLE0 Mode is entered when a HALT instruction is executed and when the IDLEN bit in the SMOD0 register is high and the FSYSON bit in the SMOD1 register is low. In the IDLE0 Mode the system oscillator will be inhibited from driving the CPU but some peripheral functions will remain operational such as the Watchdog Timer, TMs, LCD driver and SPI. In the IDLE0 Mode, the system oscillator will be stopped. In the IDLE0 Mode the Watchdog Timer clock, f_S, will either be on or off depending upon the f_S clock source. If the source is $f_{SYS}/4$ then the f_S clock will be off, and if the source comes from f_{SUB} then f_S will be on.

IDLE1 Mode

The IDLE1 Mode is entered when an HALT instruction is executed and when the IDLEN bit in the SMOD0 register is high and the FSYSON bit in the SMOD1 register is high. In the IDLE1 Mode the system oscillator will be inhibited from driving the CPU but may continue to provide a clock source to keep some peripheral functions operational such as the Watchdog Timer, TMs, LCD driver and SPI. In the IDLE1 Mode, the system oscillator will continue to run, and this system oscillator may be high speed or low speed system oscillator. In the IDLE1 Mode the Watchdog Timer clock, f_S, will be on. If the source is $f_{SYS}/4$ then the f_S clock will be on, and if the source comes from f_{SUB} then f_S will be on.

Control Register

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

SMOD0 Register

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

Bit 7~5 **CKS2~CKS0**: The system clock selection when HLCLK is "0"
000: f_L (f_{LXT} or f_{LIRC})
001: f_L (f_{LXT} or f_{LIRC})
010: $f_H/64$
011: $f_H/32$
100: $f_H/16$
101: $f_H/8$
110: $f_H/4$
111: $f_H/2$

These three bits are used to select which clock is used as the system clock source. In addition to the system clock source, which can be either the LXT or LIRC, a divided version of the high speed system oscillator can also be chosen as the system clock source.

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

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

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

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

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

This is the high speed system oscillator ready flag which indicates when the high speed system oscillator is stable. This flag is cleared to "0" by hardware when the device is powered on and then changes to a high level after the high speed system oscillator is stable.

Therefore this flag will always be read as "1" by the application program after device power-on. The flag will be low when in the SLEEP or IDLE0 Mode but after a wake-up has occurred, the flag will change to a high level after 1024 clock cycles if the HXT oscillator is used and after 15~16 clock cycles if the ERC or HIRC oscillator is used.

Bit 1 **IDLEN**: IDLE Mode control
0: Disable
1: Enable

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

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

This bit is used to select if the f_H clock or the $f_H/2 \sim f_H/64$ or f_L clock is used as the system clock. When the bit is high the f_H clock will be selected and if low the $f_H/2 \sim f_H/64$ or f_L clock will be selected. When system clock switches from the f_H clock to the f_L clock and the f_H clock will be automatically switched off to conserve power.

Fast Wake – up

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

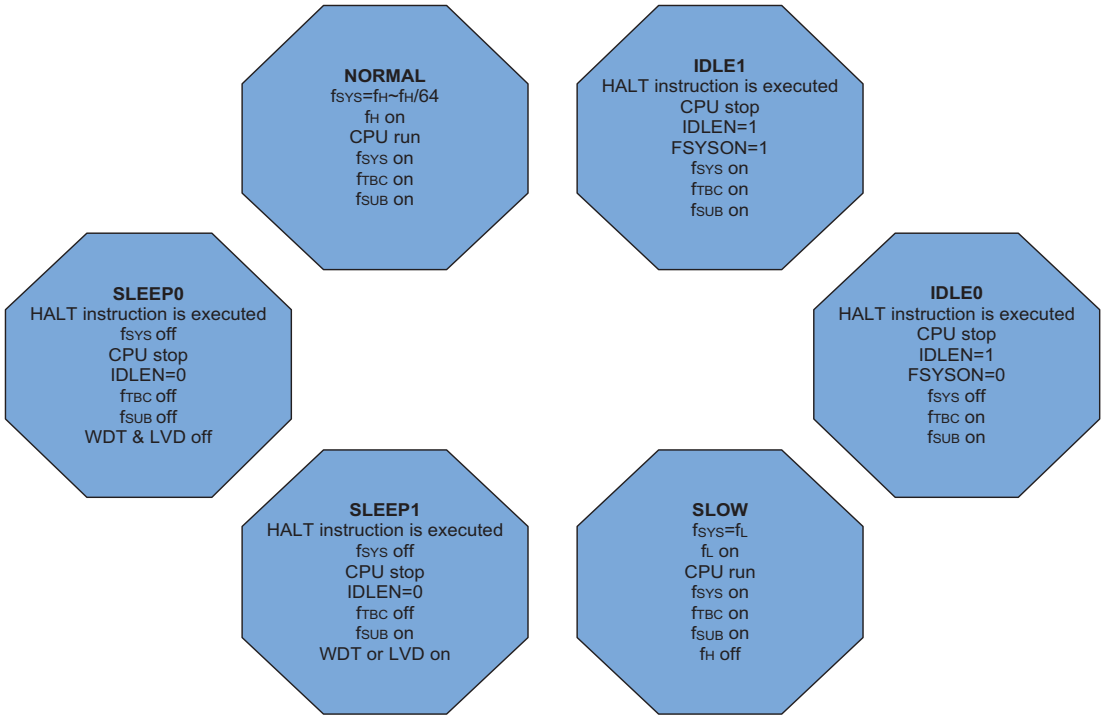
If the HXT oscillator is selected as the NORMAL Mode system clock, and if the Fast Wake-up function is enabled, then it will take one to two tSUB clock cycles of the LIRC or LXT oscillator for the system to wake-up. The system will then initially run under the f_{SUB} clock source until 1024 HXT clock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

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

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

Wake-Up Times

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



Operating Mode Switching

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

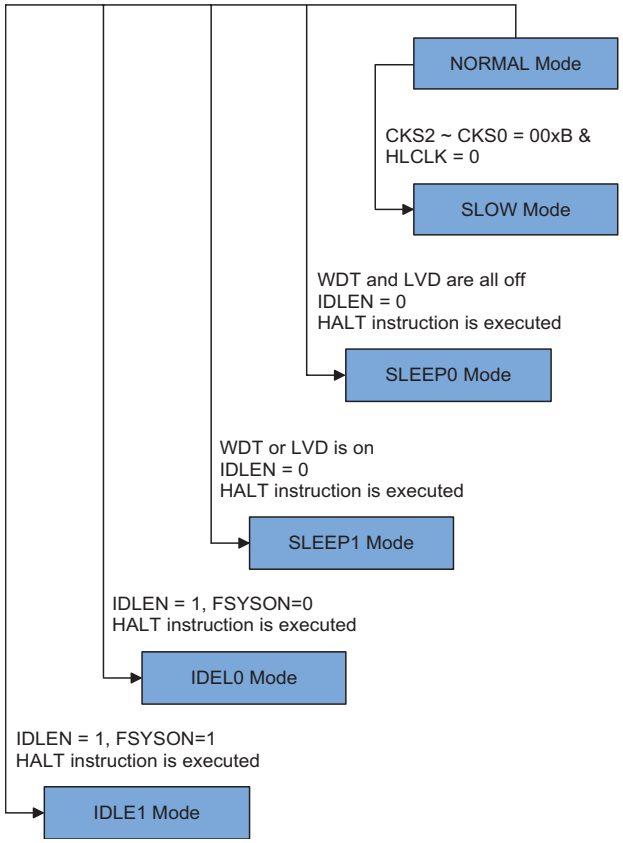
In simple terms, Mode Switching between the NORMAL Mode and SLOW Mode is executed using the HLCLK bit and CKS2~CKS0 bits in the SMOD0 register while Mode Switching from the NORMAL/SLOW Modes to the SLEEP/IDLE Modes is executed via the HALT instruction. When a HALT instruction is executed, whether the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

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

NORMAL Mode to SLOW Mode Switching

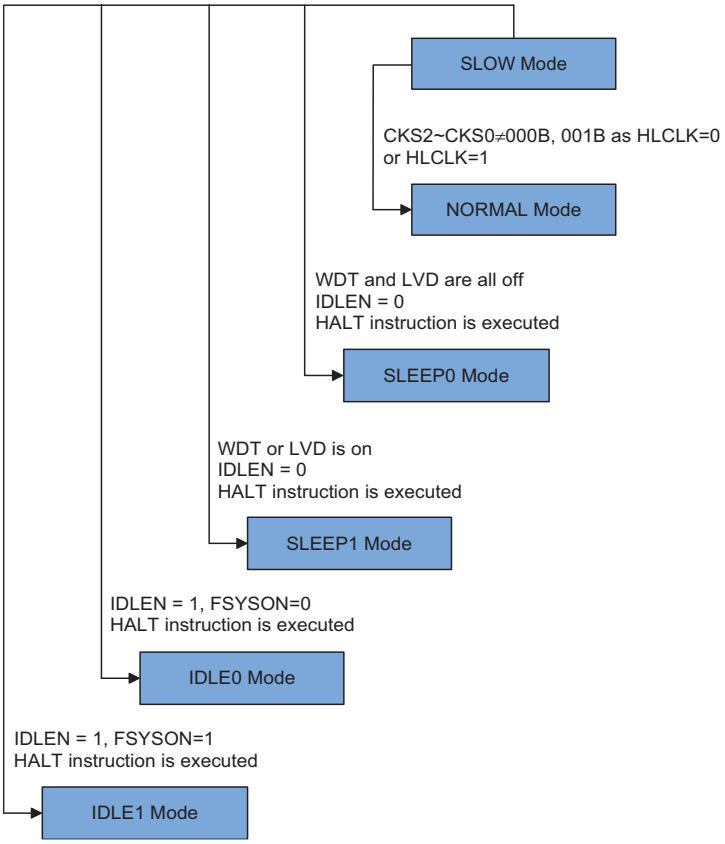
When running in the NORMAL Mode, which uses the high speed system oscillator, and therefore consumes more power, the system clock can switch to run in the SLOW Mode by set the HLCLK bit to "0" and set the CKS2~CKS0 bits to "000" or "001" in the SMOD0 register. This will then use the low speed system oscillator which will consume less power. Users may decide to do this for certain operations which do not require high performance and can subsequently reduce power consumption.

The SLOW Mode is sourced from the LXT or the LIRC oscillators and therefore requires these oscillators to be stable before full mode switching occurs. This is monitored using the LTO bit in the SMOD0 register.



SLOW Mode to NORMAL Mode Switching

In SLOW Mode the system uses either the LXT or LIRC low speed system oscillator. To switch back to the NORMAL Mode, where the high speed system oscillator is used, the HLCLK bit should be set to "1" or HLCLK bit is "0", but CKS2~CKS0 is set to "010", "011", "100", "101", "110" or "111". As a certain amount of time will be required for the high frequency clock to stabilise, the status of the HTO bit is checked. The amount of time required for high speed system oscillator stabilization depends upon which high speed system oscillator type is used.



Entering the SLEEP0 Mode

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

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

Entering the SLEEP1 Mode

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

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

Entering the IDLE0 Mode

There is only one way for the device to enter the IDLE0 Mode and that is to execute the "HALT" instruction in the application program with the IDLEN bit in SMOD0 register equal to "1" and the FSYSON bit in SMOD1 register equal to "0". When this instruction is executed under the conditions described above, the following will occur:

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

Entering the IDLE1 Mode

There is only one way for the device to enter the IDLE1 Mode and that is to execute the "HALT" instruction in the application program with the IDLEN bit in SMOD0 register equal to "1" and the FSYSON bit in SMOD1 register equal to "1". When this instruction is executed under the conditions described above, the following will occur:

- The system clock and Time Base clock and f_{SUB} clock will be on and the application program will stop at the "HALT" instruction.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting if the WDT is enabled regardless of the WDT clock source which originates from the f_{SUB} clock or from the system clock.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Standby Current Considerations

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

Care must also be taken with the loads, which are connected to I/O pins, which are setup as outputs. These should be placed in a condition in which minimum current is drawn or connected only to external circuits that do not draw current, such as other CMOS inputs. Also note that additional standby current will also be required if the configuration options have enabled the LXT or LIRC oscillator.

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

Wake-up

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

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

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

Programming Considerations

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

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

Watchdog Timer

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

Watchdog Timer Clock Source

The Watchdog Timer clock source is provided by the internal clock, f_S, which is in turn supplied by one of two sources selected by configuration option: f_{SUB} or $f_{SYS}/4$. The f_{SUB} clock can be sourced from either the LXT or LIRC oscillators, again chosen via a configuration option. The LIRC internal oscillator has an approximate period of 32kHz at a supply voltage of 5V.

However, it should be noted that this specified internal clock period can vary with V_{DD}, temperature and process variations. The LXT oscillator is supplied by an external 32.768kHz crystal. The other Watchdog Timer clock source option is the $f_{SYS}/4$ clock. The Watchdog Timer clock source can originate from its own internal LIRC oscillator, the LXT oscillator or $f_{SYS}/4$. It is divided by a value of 2^8 to 2^{18}, using the WS2~WS0 bits in the WDTC register to obtain the required Watchdog Timer time-out period.

Watchdog Timer Control Register

A single register, WDTC, controls the required timeout period as well as the enable reset operation. This register together with several configuration options control the overall operation of the Watchdog Timer.

WDTC Register

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

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

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

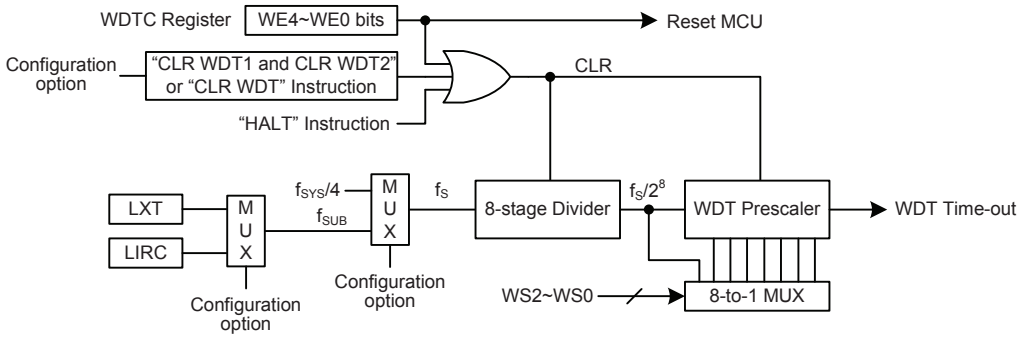
Watchdog Timer Operation

The Watchdog Timer operates by providing a device reset when its timer overflows. This means that in the application program and during normal operation the user has to strategically clear the Watchdog Timer before it overflows to prevent the Watchdog Timer from executing a reset. This is done using the clear watchdog instructions.

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

There are two methods of using software instructions to clear the Watchdog Timer, one of which must be chosen by configuration option. The first option is to use the single "CLR WDT" instruction while the second is to use the two commands "CLR WDT1" and "CLR WDT2". For the first option, a simple execution of CLR WDT will clear the WDT while for the second option, both "CLR WDT1" and "CLR WDT2" must both be executed alternately to successfully clear the Watchdog Timer. Note that for this second option, if "CLR WDT1" is used to clear the Watchdog Timer, successive executions of this instruction will have no effect, only the execution of a "CLR WDT2" instruction will clear the Watchdog Timer. Similarly after the "CLR WDT2" instruction has been executed, only a successive "CLR WDT1" instruction can clear the Watchdog Timer.

The maximum time out period is when the 2^{18} division ratio is selected. As an example, with a 32.768kHz LXT oscillator as its source clock, this will give a maximum watchdog period of around 8 second for the 2^{18} division ratio, and a minimum timeout of 7.8ms for the 2^8 division ration. If the $f_{SYS}/4$ clock is used as the Watchdog Timer clock source, it should be noted that when the system enters the SLEEP or IDLE0 Mode, then the instruction clock is stopped and the Watchdog Timer may lose its protecting purposes. For systems that operate in noisy environments, using the f_{SUB} clock source is strongly recommended.



Watchdog Timer

SMOD1 Register

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

"x" unknown

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

Bit 6~3 Unimplemented, read as "0"

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

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

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

Reset and Initialisation

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

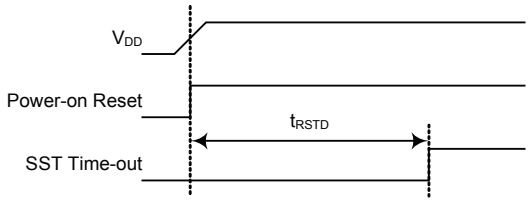
Another type of reset is when the Watchdog Timer overflows and resets. All types of reset operations result in different register conditions being setup. Another reset exists in the form of a Low Voltage Reset, LVR, where a full reset, is implemented in situations where the power supply voltage falls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

Reset Functions

There are several ways in which a reset can occur, each of which will be described as follows.

Power-on Reset

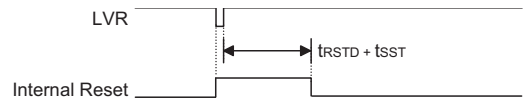
The most fundamental and unavoidable reset is the one that occurs after power is first applied to the microcontroller. As well as ensuring that the Program Memory begins execution from the first memory address, a power-on reset also ensures that certain other registers are preset to known conditions.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



Power-On Reset Timing Chart

Low Voltage Reset – LVR

The microcontroller contains a low voltage reset circuit in order to monitor the supply voltage of the device. The LVR function is always enabled with a specific LVR voltage, V_{LVR}. If the supply voltage of the device drops to within a range of 0.9V~V_{LVR} such as might occur when changing the battery, the LVR will automatically reset the device internally. The LVR includes the following specifications: For a valid LVR signal, a low voltage, i.e., a voltage in the range between 0.9V~V_{LVR} must exist for greater than the value t_{LVR} specified in the LVR Electrical Characteristics. If the low voltage state does not exceed t_{LVR}, the LVR will ignore it and will not perform a reset function. The actual V_{LVR} value can be selected by the LVS7~LVS0 bits in the LVRC register. If the LVS7~LVS0 bits are changed to some certain values by the environmental noise or software setting, the LVR will reset the device after 2~3 LIRC clock cycles. When this happens, the LRF bit in the SMOD1 register will be set to 1. After power on the register will have the value of 01010101B. Note that the LVR function will be automatically disabled when the device enters the power down mode.



Low Voltage Reset Timing Chart

- **LVRC Register**

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

Bit 7~0 **LVS7~LVS0**: LVR voltage select
01010101: 2.1V (default)
00110011: 2.55V
10011001: 3.15V
10101010: 3.8V
Other values: MCU reset (reset will be active after 2~3 LIRC clock for debounce time).
When an actual low voltage condition occurs, as specified by one of the four defined LVR voltage values above, an MCU reset will be generated. In this situation the register contents will remain the same after such a reset occurs.
Any register value, other than the four defined LVR values above, will also result in the generation of an MCU reset. The reset operation will be activated after 2~3 LIRC clock cycles. However in this situation the register contents will be reset to the POR value.

- **SMOD1 Register**

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

"x" unknown

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

Bit 6~3 Unimplemented, read as "0"

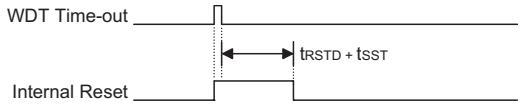
Bit 2 **LVRF**: reset caused by LVR function activation
0: not active
1: active
This bit can be clear to "0", but can not set to "1".

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

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

Watchdog Time-out Reset during Normal Operation

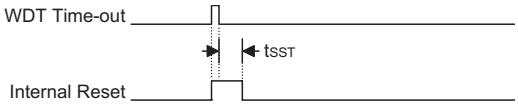
The Watchdog time-out Reset during normal operation is the same as a hardware LVR reset except that the Watchdog time-out flag TO will be set to "1".



WDT Time-out Reset during Normal Operation Timing Chart

Watchdog Time-out Reset during SLEEP or IDLE Mode

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



WDT Time-out Reset during Sleep or IDLE Mode Timing Chart

Reset Initial Conditions

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

TO	PDF	Reset Conditions
0	0	Power-on reset
u	u	LVR reset during Normal or SLOW Mode operation
1	u	WDT time-out reset during Normal or SLOW Mode operation
1	1	WDT time-out reset during IDLE or SLEEP Mode operation

Note: “u” stands for unchanged

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

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

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

Register	Power On Reset	LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
MP0	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
MP1	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
BP	--0- -000	--0- -000	--0- -000	--u- -uuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	--xx xxxx	--uu uuuu	--uu uuuu	--uu uuuu
STATUS	--00 xxxx	--uu uuuu	--1u uuuu	--11 uuuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFIC0	--00 --00	--00 --00	--00 --00	--uu --uu
MFIC1	--00 --00	--00 --00	--00 --00	--uu --uu
TBC	0011 0111	0011 0111	0011 0111	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAPU	-000 0000	-000 0000	-000 0000	-uuu uuuu
PAWK	0000 0000	0000 0000	0000 0000	uuuu uuuu
PB	--11 1111	--11 1111	--11 1111	--uu uuuu
PBC	--11 1111	--11 1111	--11 1111	--uu uuuu
PBPU	--00 0000	--00 0000	--00 0000	--uu uuuu
PC	-111 1111	-111 1111	-111 1111	-uuu uuuu
PCC	-111 1111	-111 1111	-111 1111	-uuu uuuu
PCPU	-000 0000	-000 0000	-000 0000	-uuu uuuu
CTRL2	0010 100-	0010 100-	0010 100-	uuuu uuu-
CTRL3	0000 0---	0000 0---	0000 0---	uuuu u---
CTRL1	--00 0000	--00 0000	--00 0000	--uu uuuu
VDDSPICR	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
ADRL	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADRH	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADCR	011- 0000	011- 0000	011- 0000	uuu- uuuu
ACSR	---0 0000	---0 0000	---0 0000	---u uuuu
WDTC	0101 0011	0101 0011	0101 0011	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFIC2	-000 -000	-000 -000	-000 -000	-uuu -uuu

Register	Power On Reset	LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
MFIC3	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFIC4	0001 0000	0001 0000	0001 0000	uuuu uuuu
ANCSR	---- 0000	---- 0000	---- 0000	---- uuuu
DAL	0000 ----	0000 ----	0000 ----	uuuu ----
DAH	0000 0000	0000 0000	0000 0000	uuuu uuuu
VOL	0000 ----	0000 ----	0000 ----	uuuu ----
LVDC	--00 -000	--00 -000	--00 -000	--uu -uuu
CCVREFC	00-- 0000	00-- 0000	00-- 0000	uu-- uuuu
PGAC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PGAC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PGAC2	0000 0000	0000 0000	0000 0000	uuuu uuuu
SCFC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SCFC1	-0x0 00--	-0x0 00--	-0x0 00--	-uuu uu--
SCFCKD	0000 0000	0000 0000	0000 0000	uuuu uuuu
LCDCTRL	000- 0000	000- 0000	000- 0000	uuu- uuuu
SEGCR	0000 0000	0000 0000	0000 0000	uuuu uuuu
VLCDC	0--- 0000	0--- 0000	0--- 0000	u--- uuuu
SPI0C0	111- --0-	111- --0-	111- --0-	uuu- --u-
SPI0C1	--00 0000	--00 0000	--00 0000	--uu uuuu
SPI0D	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SPI1C0	111- --0-	111- --0-	111- --0-	uuu- --u-
SPI1C1	--00 0000	--00 0000	--00 0000	--uu uuuu
SPI1D	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
EEA	--xx xxxx	--xx xxxx	--xx xxxx	--uu uuuu
EED	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
LVRC	0101 0101	0101 0101	0101 0101	uuuu uuuu
TM0C0	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0DH	---- --00	---- --00	---- --00	---- --uu
TM0AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0AH	---- --00	---- --00	---- --00	---- --uu
TM1C0	0000 0---	0000 0---	0000 0---	uuuu u---
TM1C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1DH	---- --00	---- --00	---- --00	---- --uu
TM1AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1AH	---- --00	---- --00	---- --00	---- --uu
TM2C0	0000 0---	0000 0---	0000 0---	uuuu u---
TM2C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2DL	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	Power On Reset	LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
TM2DH	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2AH	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2RP	0000 0000	0000 0000	0000 0000	uuuu uuuu
SMOD0	0000 0011	0000 0011	0000 0011	uuuu uuuu
SMOD1	0--- -x00	0--- -x00	0--- -x00	u--- -uuu
SCKOC	---0 00-0	---0 00-0	---0 00-0	---u uu-u
CHPRC	0000 0-00	0000 0-00	0000 0-00	uuuu u-uu
FARL	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARH	0000 0000	0000 0000	0000 0000	uuuu 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
EEC	---- 0000	---- 0000	---- 0000	---- uuuu
FC0	0111 0000	0111 0000	0111 0000	uuuu uuuu
FC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC2	---- ---0	---- ---0	---- ---0	---- ---u

Note: “u” stands for unchanged
“x” stands for unknown
“-” stands for unimplemented

Input/Output Ports

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

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

I/O Register List

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAWK	PAWK7	PAWK6	PAWK5	PAWK4	PAWK3	PAWK2	PAWK1	PAWK0
PAPU	—	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	—	—	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PB	—	—	PB5	PB4	PB3	PB2	PB1	PB0
PBC	—	—	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

Pull-high Resistors

Many product applications require pull-high resistors for their switch inputs usually requiring the use of an external resistor. To eliminate the need for these external resistors, all I/O pins, when configured as an input have the capability of being connected to an internal pull-high resistor. These pull-high resistors are selected using registers PAPU~PDPU, and are implemented using weak PMOS transistors. Note that pin PA7 does not have a pull-high resistor selection.

PAPU Register

Bit	7	6	5	4	3	2	1	0
Name	—	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

PBPU Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

PCPU Register

Bit	7	6	5	4	3	2	1	0
Name	—	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

PDPU Register

Bit	7	6	5	4	3	2	1	0
Name	PDPU7	PDPU6	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
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

"—" Unimplemented, read as "0"

PAPUn/PBPUn/PCPUn/PDPUn: I/O Port Pull-high Control

0: Disable

1: Enable

Port A Wake-up

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

PAWK Register

Bit	7	6	5	4	3	2	1	0
Name	PAWK7	PAWK6	PAWK5	PAWK4	PAWK3	PAWK2	PAWK1	PAWK0
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 **PAWK7~PAWK0**: 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~PDC, to control the input/output configuration. Only the output for PA7 is open-drain output. With this control register, each CMOS output or input can be reconfigured dynamically under software control. Each pin of the I/O ports is directly mapped to a bit in its associated port control register. For the I/O pin to function as an input, the corresponding bit of the control register must be written as a "1". This will then allow the logic state of the input pin to be directly read by instructions. When the corresponding bit of the control register is written as a "0", the I/O pin will be setup as a CMOS output. If the pin is currently setup as an output, instructions can still be used to read the output register. However, it should be noted that the program will in fact only read the status of the output data latch and not the actual logic status of the output pin.

PAC Register

Bit	7	6	5	4	3	2	1	0
Name	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

PBC Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	1	1	1	1	1	1

PCC Register

Bit	7	6	5	4	3	2	1	0
Name	—	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	1	1	1	1	1	1	1

PDC Register

Bit	7	6	5	4	3	2	1	0
Name	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
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

"—" Unimplemented, read as "0"

PACn/PBCn/PCCn/PDCn: I/O Port Input/Output Control

0: Output

1: Input

SCKO Function

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

SCKO Operation

The SCKO function is controlled using the SCKOC register. As the pin, SCKO, is shared with PC1, the required pin function is chosen via the SCKC bit in the SCKOC register. The clock source for the SCKO Output is sourced from a divided ratio of the internal f_{SYS} clock. The SCKC bit in is the overall on/off control, setting SCKC bit to "1" enables the SCKO, setting SCKC bit to "0" disables it. The required division ratio of the system clock is selected using the SCKS1 and SCKS0 bits in the same register. If the SCKO is enabled, the output signal in the output mode of PC1 will be the SCKO signal generated by the system clock. The input mode always remains its original functions. Once the SCKO is selected, the SCKO output signal is controlled by PC1 data register only.

SCKOC Register

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

Bit 7~5 Unimplemented, read as "0"

Bit 4 **SCKC**: Select I/O or SCK output function
0: I/O
1: SCK output

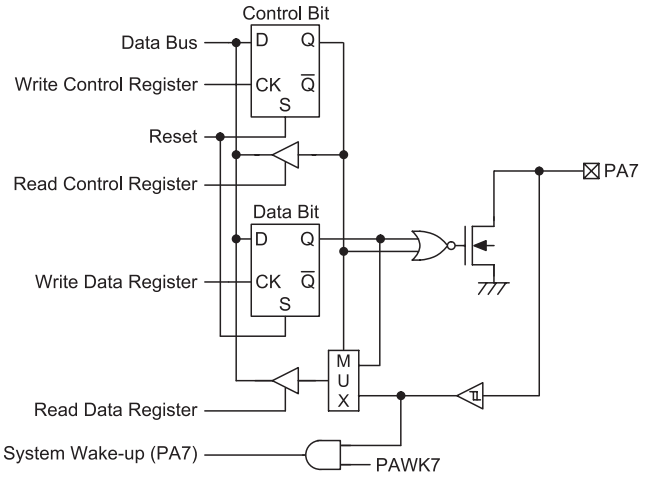
Bit 3~2 **SCKS1, SCKS0**: SCK output frequency selection (SCKO pin)
00: $f_{SYS}/1$
01: $f_{SYS}/2$
10: $f_{SYS}/4$
11: $f_{SYS}/8$

Bit 1 Unimplemented, read as "0"

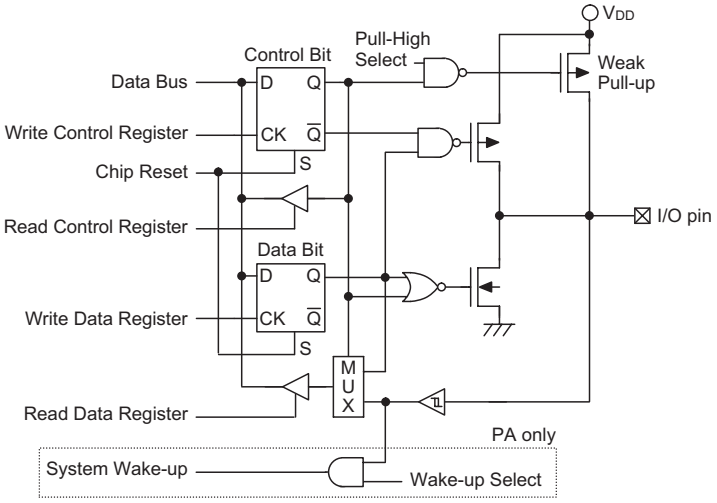
Bit 0 **SCKOS**: PC1/SCKO output sink/source current option
0: normal I/O current
1: normal I/O current/2

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.



Input/output port (PA7)



Generic Input/Output Ports

Programming Considerations

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

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

Timer Modules – TM

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

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

Introduction

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

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

This chip contains a specific number of either Compact Type and Standard Type TM units which are shown in the table together with their individual reference names, TM0~TM2.

Device	TM0	TM1	TM2
HT45F3W	10-bit CTM	10-bit CTM	16-bit STM

TM Name/Type Reference

TM Operation

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

TM Clock Source

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

TM Interrupts

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

TM External Pins

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

The TMs each have two output pins with the label TPn_0 and TPn_1. When the TM is in the Compare Match Output Mode, these pins can be controlled by the TM to switch to a high or low level or to toggle when a compare match situation occurs. The external TPn output pin is also the pin where the TM generates the PWM output waveform. As the TM output pins are pin-shared with other function, the TM output function must first be setup using registers. 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.

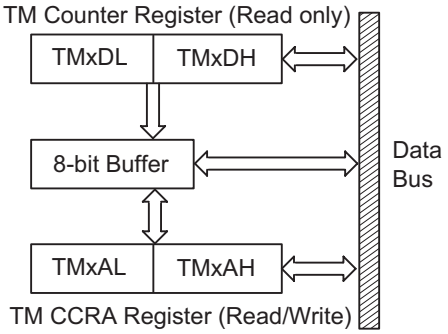
All TM output pin names have a "_n" suffix. Pin names that include a "_0" or "_1" suffix indicate that they are from a TM with multiple output pins. This allows the TM to generate a complimentary output pair, selected using the I/O register data bits.

Device	CTM	STM
HT45F3W	TP0_0, TP0_1,TP1_0, TP1_1	TP2_0, TP2_1

TM Output Pins

Programming Considerations

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



The following steps show the read and write procedures:

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

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

Compact Type TM – CTM

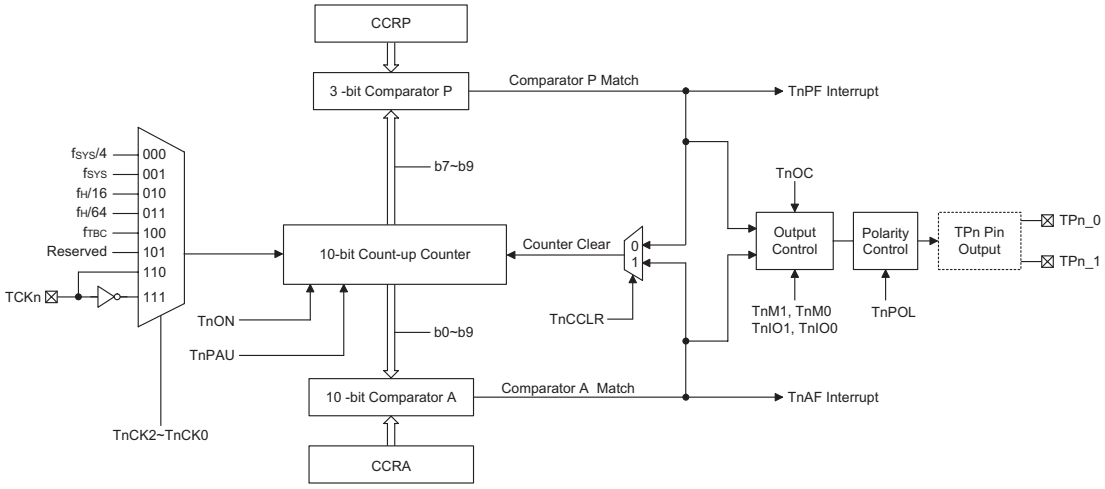
Although the simplest form of the two TM types, the Compact TM type still contains three operating modes, which are Compare Match Output, Timer/Event Counter and PWM Output modes. The Compact TM can also be controlled with an external input pin and can drive two external output pins. These two external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

Name	TM No.	TM Input Pin	TM Output Pin
10-bit CTM	0, 1	TCK0, TCK1	TP0_0, TP0_1 TP1_0, TP1_1

Compact TM Operation

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

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



Compact Type TM Block Digram (n=0 or 1)

Compact Type TM Register Description

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

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

Compact TM Register List (n=0 or 1)

TMnC0 Register (n=0 or 1)

Bit	7	6	5	4	3	2	1	0
Name	T0PAU	T0CK2	T0CK1	T0CK0	T0ON	T0RP2	T0RP1	T0RP0
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 **T0PAU**: TM0 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 **T0CK2~T0CK0**: Select TM0 Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{TBC}
101: Reserved
110: TCK0 rising edge clock
111: TCK0 falling edge clock

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

Bit 3 **T0ON**: TM0 Counter On/Off Control
0: Off
1: On
This bit controls the overall on/off function of the TM. Setting the bit high enables the counter to run, clearing the bit disables the TM. Clearing this bit to zero will stop the counter from counting and turn off the TM which will reduce its power consumption. When the bit changes state from low to high the internal counter value will be reset to zero, however when the bit changes from high to low, the internal counter will retain its residual value. If the TM is in the Compare Match Output Mode then the TM output pin will be reset to its initial condition, as specified by the T0OC bit, when the T0ON bit changes from low to high.

Bit 2~0 **T0RP2~T0RP0**: TM0 CCRP 3-bit register, compared with the TM0 Counter bit 9~bit 7 Comparator P Match Period
000: 1024 TM0 clocks
001: 128 TM0 clocks
010: 256 TM0 clocks
011: 384 TM0 clocks
100: 512 TM0 clocks
101: 640 TM0 clocks
110: 768 TM0 clocks
111: 896 TM0 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 T0CCLR bit is set to zero. Setting the T0CCLR bit to zero ensures that a compare match with the CCRP values will reset the internal counter. As the CCRP bits are only compared with the highest three counter bits, the compare values exist in 128 clock cycle multiples. Clearing all three bits to zero is in effect allowing the counter to overflow at its maximum value.

TMnC1 Register (n=0 or 1)

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

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

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

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

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

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

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

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

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

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

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

TMnDL Register (n=0 or 1)

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

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

TMnDH Register (n=0 or 1)

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

Bit 7~2 Unimplemented, read as "0"

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

TMnAL Register (n=0 or 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**: TMn CCRA Low Byte Register bit 7 ~ bit 0
TMn 10-bit CCRA bit 7 ~ bit 0

TMnAH Register (n=0 or 1)

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

Bit 7~2 Unimplemented, read as "0"

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

Compact Type TM Operating Modes

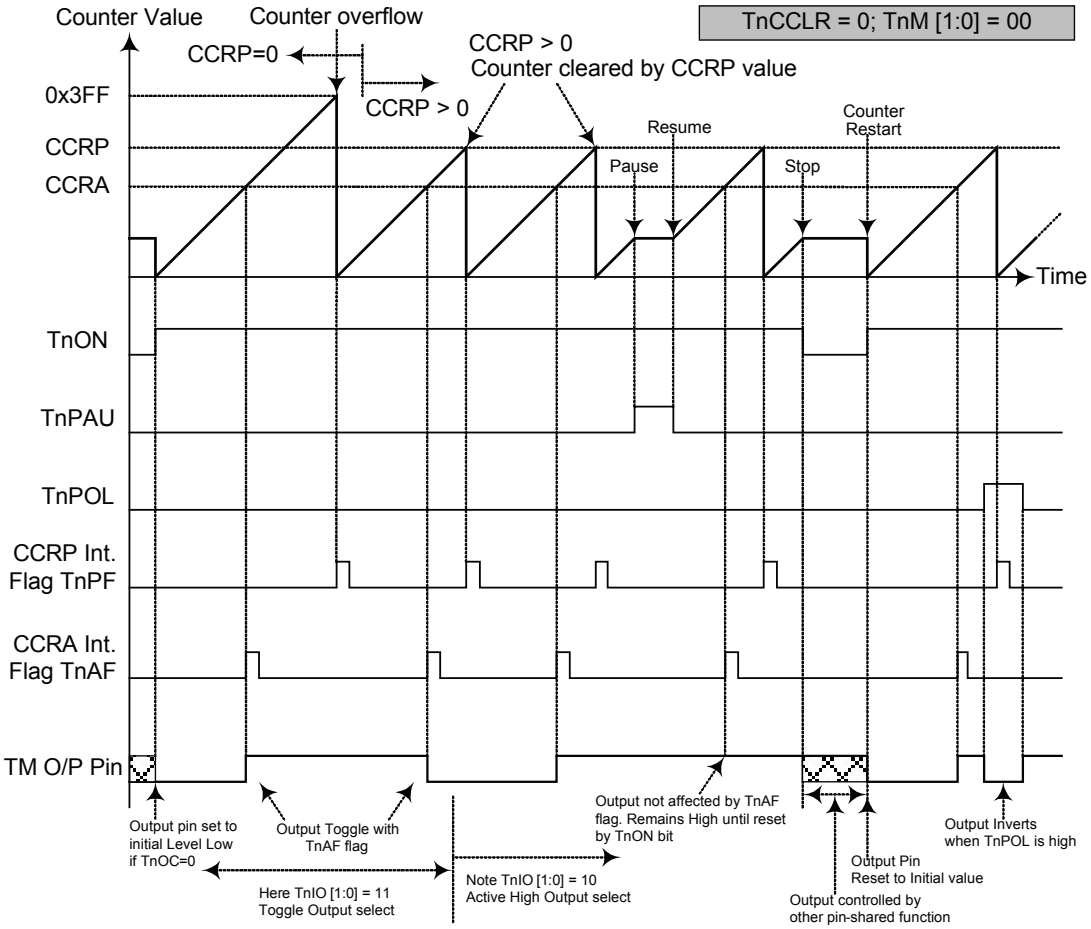
The Compact Type TM can operate in one of three operating modes, Compare Match Output Mode, PWM Mode or Timer/Counter Mode. The operating mode is selected using the TnM1 and TnM0 bits in the TMnC1 register.

Compare Match Output Mode

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

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

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

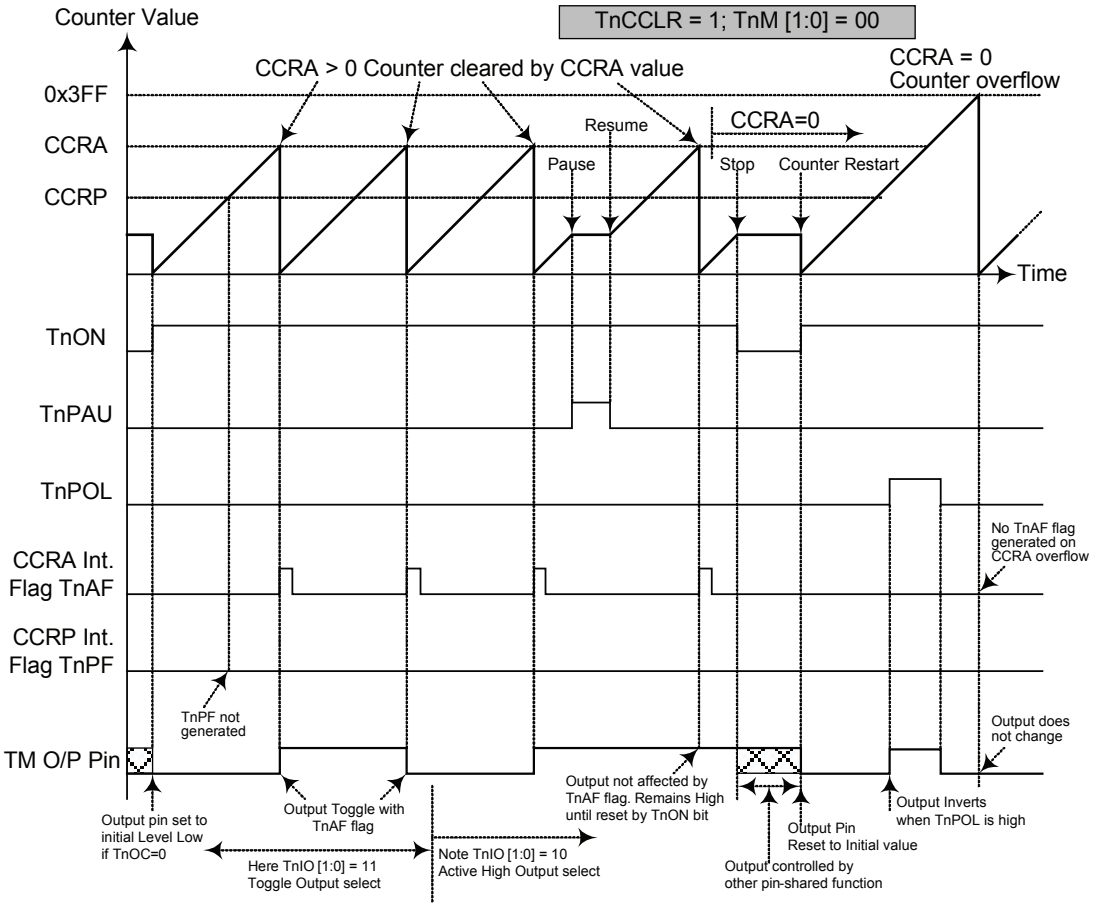


Compare Match Output Mode - TnCCLR = 0 (n=0 or 1)

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

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

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



Compare Match Output Mode - TnCCLR = 1 (n=0 or 1)

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

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

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

4. The TnPF flag is not generated when TnCCLR=1

Timer/Counter Mode

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

PWM Output Mode

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

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

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

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

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

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

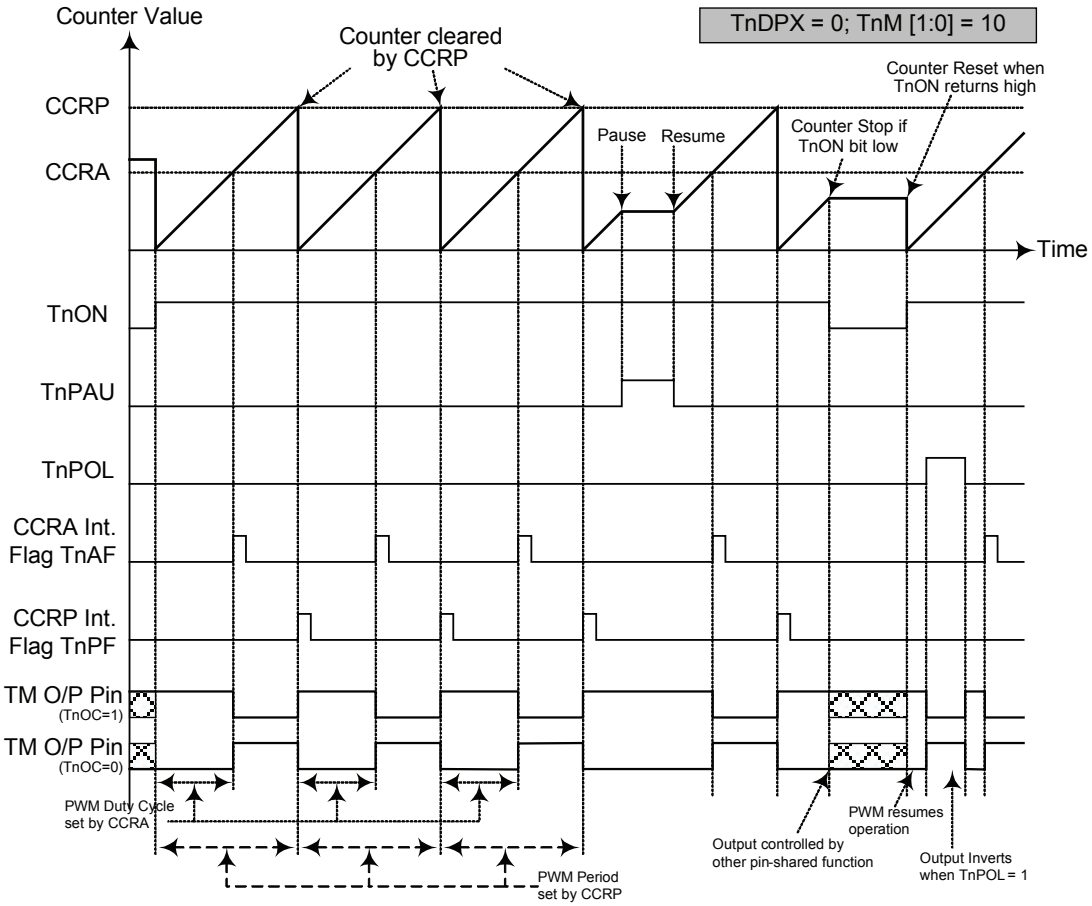
The CTM PWM output frequency = $(f_{SYS}/4)/512 = f_{SYS}/2048$ = 7.8125 kHz, duty = 128/512 = 25%.

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

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

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

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



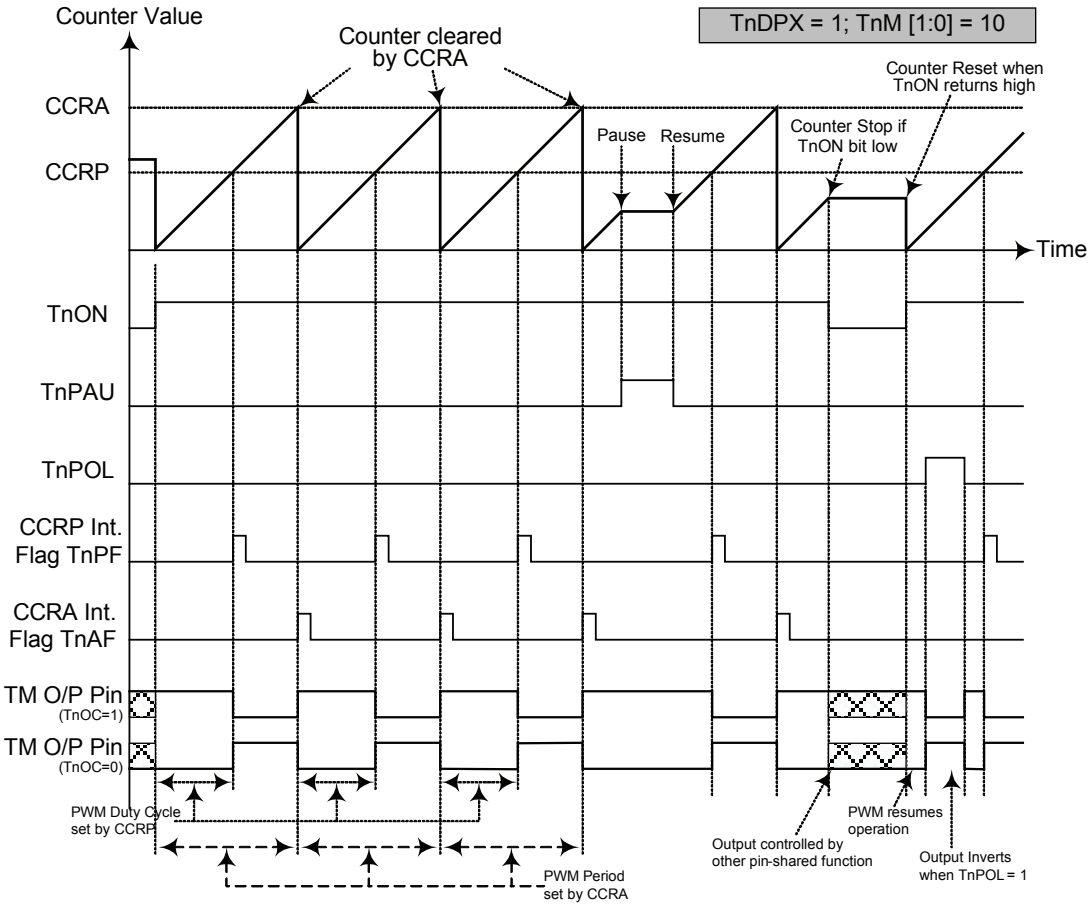
PWM Mode - TnDPX = 0 (n=0 or 1)

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

2. A counter clear sets the PWM Period

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

4. The TnCCLR bit has no influence on PWM operation



PWM Mode - TnDPX = 1 (n=0 or 1)

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

2. A counter clear sets the PWM Period

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

4. The TnCCLR bit has no influence on PWM operation

Standard Type TM – STM

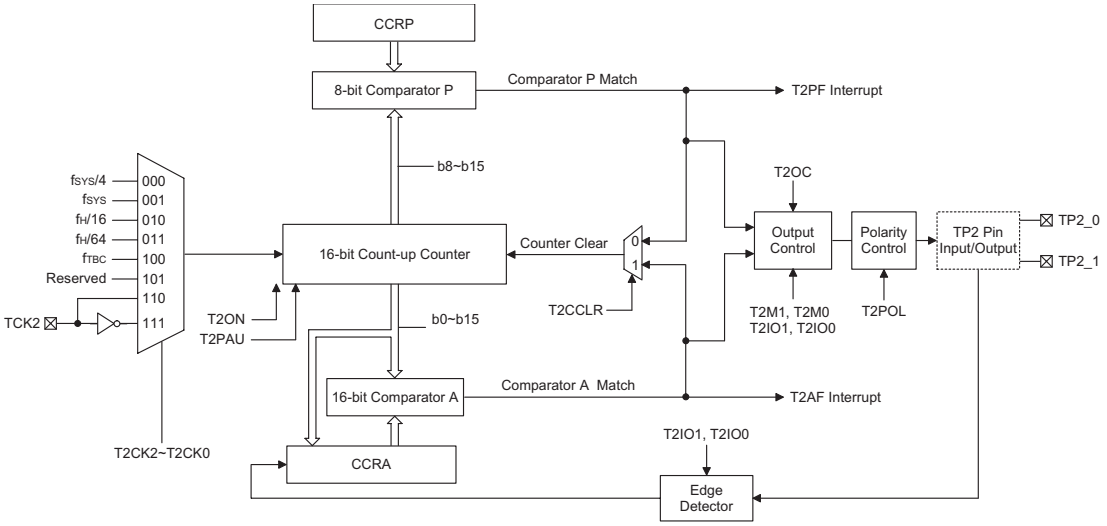
The Standard Type TM contains five operating modes, which are Compare Match Output, Timer/ Event Counter, Capture Input, Single Pulse Output and PWM Output modes. The Standard TM can also be controlled with an external input pin and can drive two external output pins.

Name	TM No.	TM Input Pin	TM Output Pin
16-bit STM	2	TCK2	TP2_0, TP2_1

Standard TM Operation

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

The only way of changing the value of the 16-bit counter using the application program, is to clear the counter by changing the T2ON bit from low to high. The counter will also be cleared automatically by a counter overflow or a compare match with one of its associated comparators. When these conditions occur, a TM interrupt signal will also usually be generated. The Standard Type TM can operate in a number of different operational modes, can be driven by different clock sources including an input pin and can also control an output pin. All operating setup conditions are selected using relevant internal registers.



Standard Type TM Block Digram

Standard Type TM Register Description

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

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

16-bit Standard TM Register List

TM2C0 Register

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

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

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

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

Bit 2~0 Unimplemented, read as "0"

TM2C1 Register

Bit	7	6	5	4	3	2	1	0
Name	T2M1	T2M0	T2IO1	T2IO0	T2OC	T2POL	T2DPX	T2CCLR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **T2M1, T2M0**: Select TM2 Operating Mode
00: Compare Match Output Mode
01: Capture Input Mode
10: PWM Mode or Single Pulse Output Mode
11: Timer/Counter Mode

These bits setup the required operating mode for the TM. To ensure reliable operation the TM should be switched off before any changes are made to the T2M1 and T2M0 bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4 **T2IO1, T2IO0**: Select TP2_0, TP2_1 output function
Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output
PWM Mode/Single Pulse Output Mode
00: PWM Output inactive state
01: PWM Output active state
10: PWM output
11: Single pulse output
Capture Input Mode
00: Input capture at rising edge of TP2_0, TP2_1
01: Input capture at falling edge of TP2_0, TP2_1
10: Input capture at falling/rising edge of TP2_0, TP2_1
11: Input capture disabled
Timer/Counter Mode
Unused

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

In the Compare Match Output Mode, the T2IO1 and T2IO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the T2OC bit in the TM2C1 register. Note that the output level requested by the T2IO1 and T2IO0 bits must be different from the initial value setup using the T2OC bit otherwise no change will occur on the TM output pin when a compare match occurs. After the TM output pin changes state, it can be reset to its initial level by changing the level of the T2ON bit from low to high.

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

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

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

Bit 1 **T2DPX**: TM2 PWM period/duty Control
0: CCRP - period; CCRA - duty
1: CCRP - duty; CCRA - period
This bit, determines which of the CCRA and CCRP registers are used for period and duty control of the PWM waveform.

Bit 0 **T2CCLR**: Select TM2 Counter clear condition
0: TM2 Comparatror P match
1: TM2 Comparatror A match
This bit is used to select the method which clears the counter. Remember that the Compact TM contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the T2CCLR bit set high, the counter will be cleared when a compare match occurs from the Comparator A. When the bit is low, the counter will be cleared when a compare match occurs from the Comparator P or with a counter overflow. A counter overflow clearing method can only be implemented if the CCRP bits are all cleared to zero. The T2CCLR bit is not used in the PWM Mode, Single Pulse or Input Capture Mode.

TM2DL Register

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

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

TM2DH Register

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

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

TM2AL Register

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

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

TM2AH Register

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

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

TM2RP Register

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

Bit 7~0 **D7~D0**: TM2 CCRP Register bit 7 ~ bit 0
TM2 CCRP 8-bit register, compared with the TM2 Counter bit 15 ~ bit 8. Comparator P Match Period
0: 65536 TM2 clocks
1~255: 256×(1~255) TM2 clocks

These eight bits are used to setup the value on the internal CCRP 8-bit register, which are then compared with the internal counter's highest eight bits. The result of this comparison can be selected to clear the internal counter if the T2CCLR bit is set to zero. Setting the T2CCLR bit to zero ensures that a compare match with the CCRP values will reset the internal counter. As the CCRP bits are only compared with the highest eight counter bits, the compare values exist in 256 clock cycle multiples. Clearing all eight bits to zero is in effect allowing the counter to overflow at its maximum value.

Standard Type TM Operating Modes

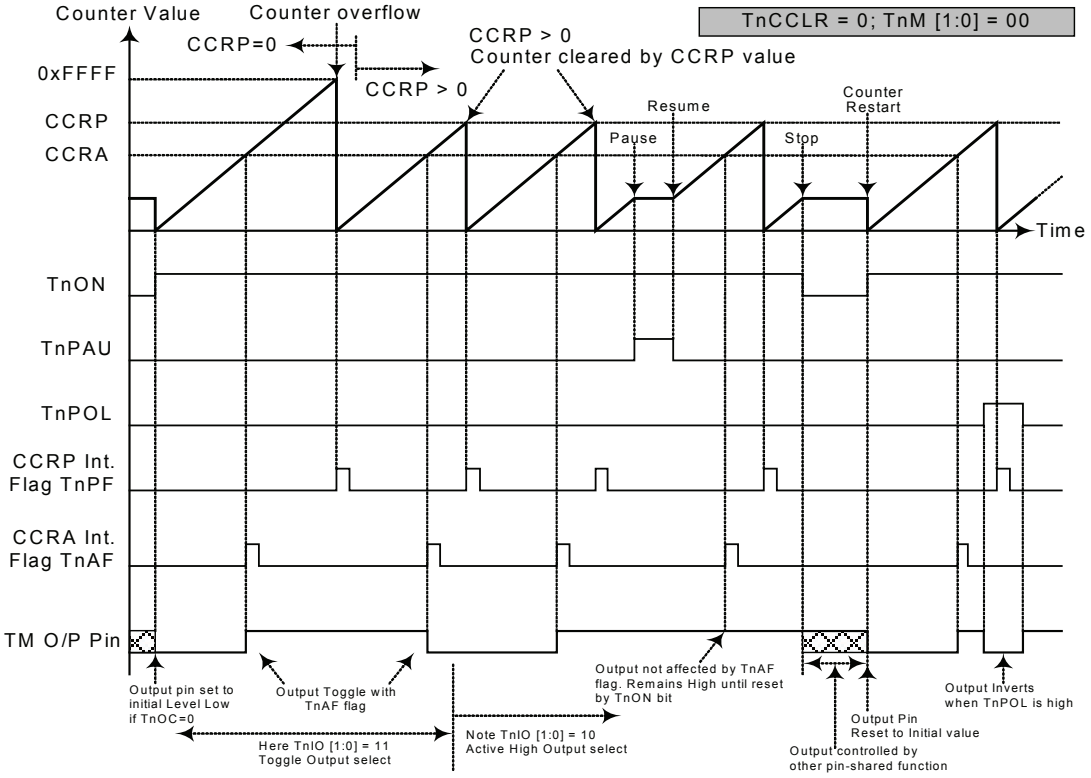
The Standard Type TM can operate in one of five operating modes, Compare Match Output Mode, PWM Output Mode, Single Pulse Output Mode, Capture Input Mode or Timer/Counter Mode. The operating mode is selected using the T2M1 and T2M0 bits in the TM2C1 register.

Compare Output Mode

To select this mode, bits T2M1 and T2M0 in the TM2C1 register, should be set to 00 respectively. In this mode once the counter is enabled and running it can be cleared by three methods. These are a counter overflow, a compare match from Comparator A and a compare match from Comparator P. When the T2CCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both T2AF and T2PF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

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

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

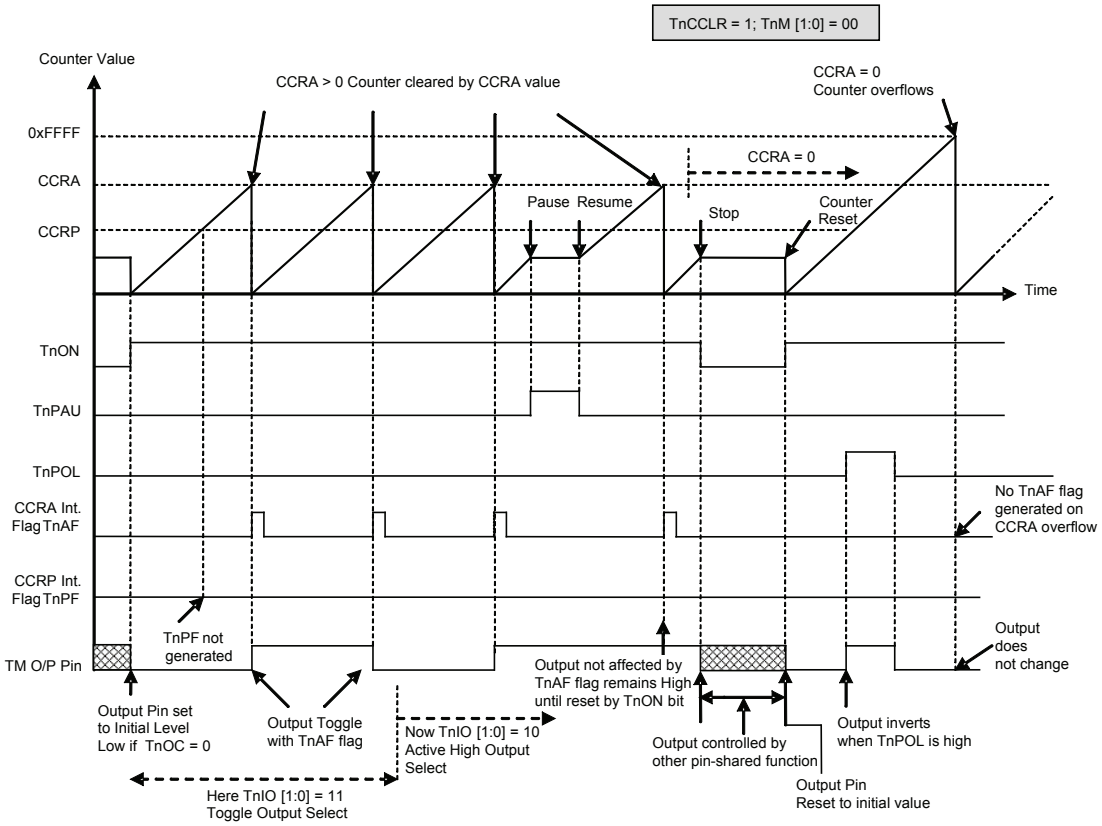


Compare Match Output Mode - TnCCLR = 0 (n=2)

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

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

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



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

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

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

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

4. A TnPF flag is not generated when TnCCLR=1

Timer/Counter Mode

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

PWM Output Mode

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

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM mode, the T2CCLR bit has no effect as the PWM period. Both of the CCRA and CCRP registers are used to generate the PWM waveform, one register is used to clear the internal counter and thus control the PWM waveform frequency, while the other one is used to control the duty cycle. Which register is used to control either frequency or duty cycle is determined using the T2DPX bit in the TM2C1 register. The PWM waveform frequency and duty cycle can therefore be controlled by the values in the CCRA and CCRP registers.

An interrupt flag, one for each of the CCRA and CCRP, will be generated when a compare match occurs from either Comparator A or Comparator P. The T2OC bit in the TM2C1 register is used to select the required polarity of the PWM waveform while the two T2IO1 and T2IO0 bits are used to enable the PWM output or to force the TM output pin to a fixed high or low level. The T2POL bit is used to reverse the polarity of the PWM output waveform.

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

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

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

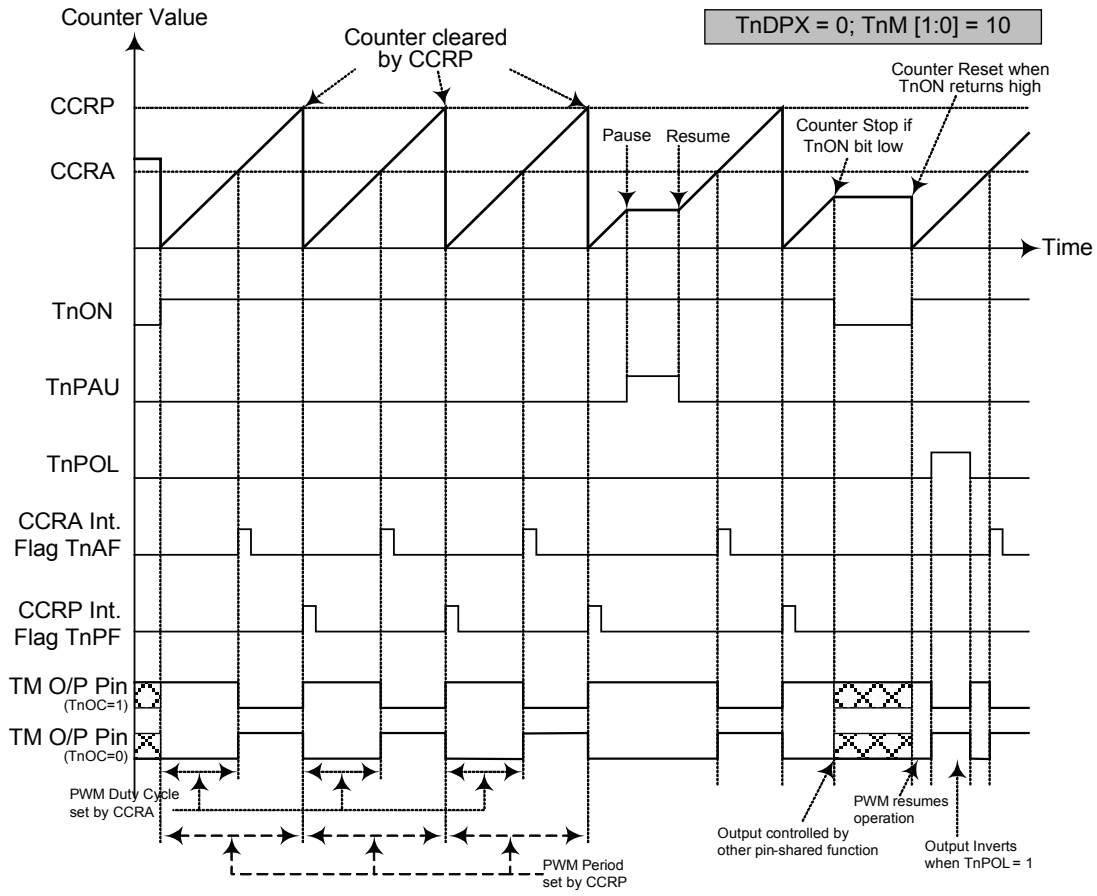
The STM PWM output frequency = $(f_{SYS}/4)/(2\times256) = f_{SYS}/2048$ = 7.8125kHz, duty = $128/(2\times256)$ = 25%.

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

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

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

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



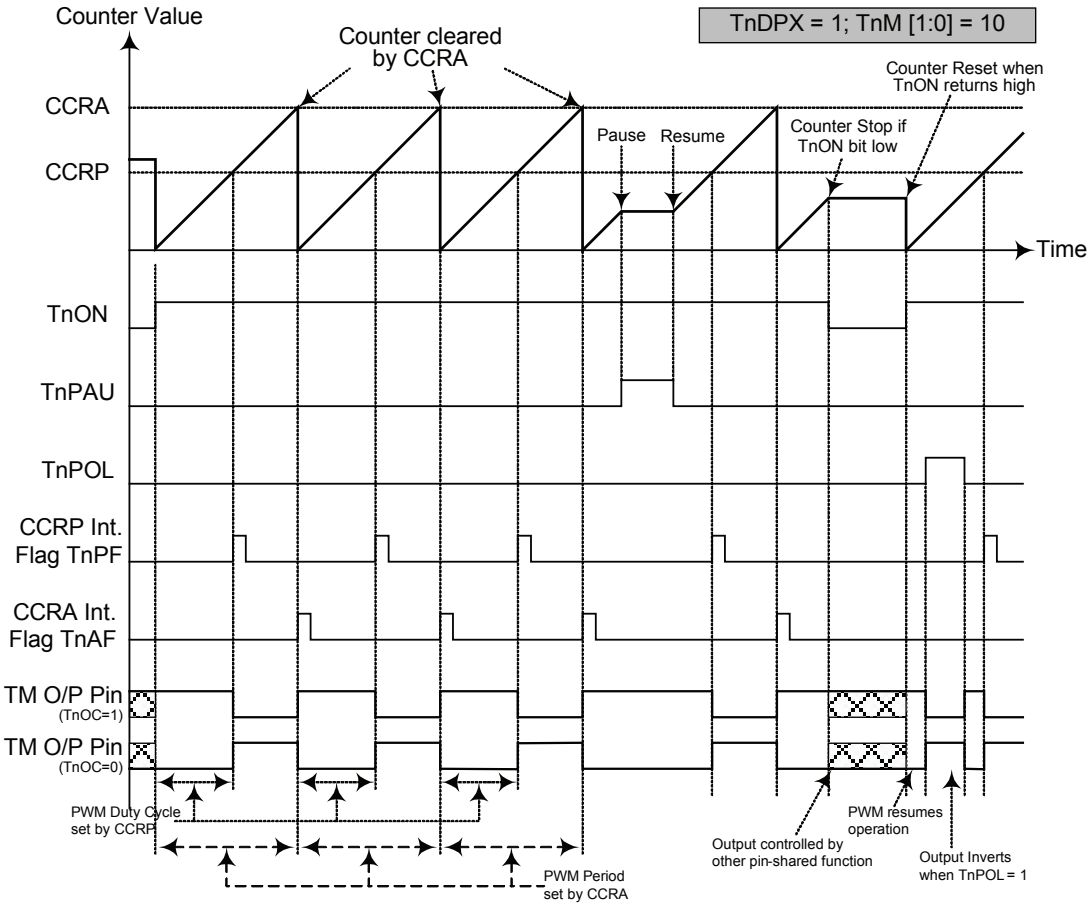
PWM Mode - TnDPX = 0 (n=2)

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

2. A counter clear sets the PWM Period

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

4. The TnCCLR bit has no influence on PWM operation



PWM Mode - TnDPX = 1 (n=2)

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

2. A counter clear sets the PWM Period

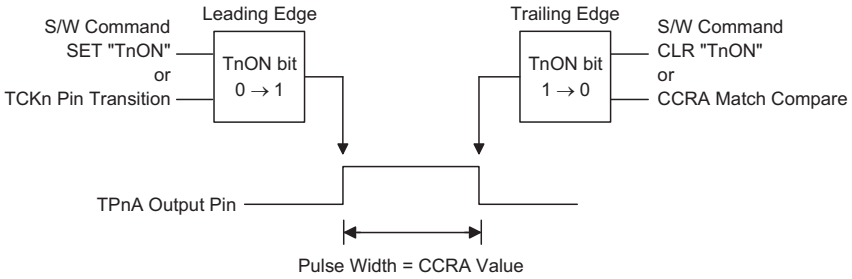
3. The internal PWM function continues even when TnIO [1:0] = 00 or 01

4. The TnCCLR bit has no influence on PWM operation

Single Pulse Mode

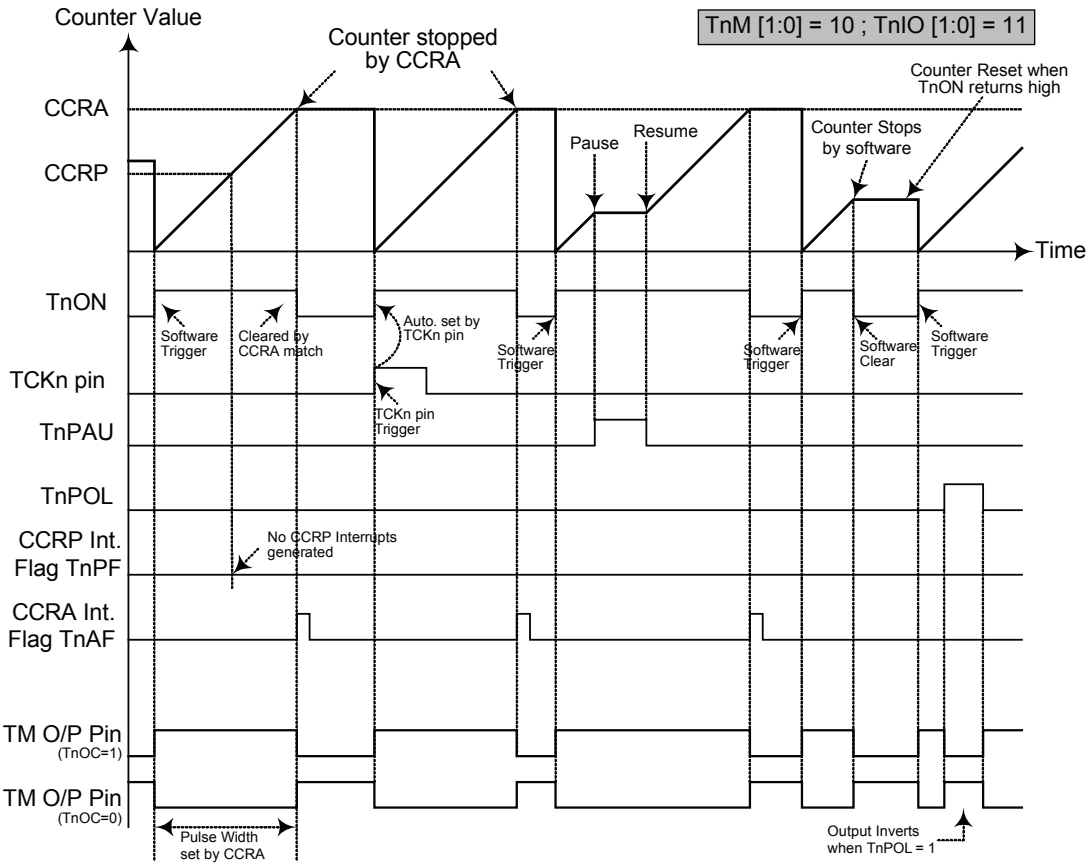
To select this mode, bits T2M1 and T2M0 in the TM2C1 register should be set to 10 respectively and also the T2IO1 and T2IO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the TM output pin.

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



Single Pulse Generation (n=2)

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



Single Pulse Mode (n=2)

Note: 1. Counter stopped by CCRA

2. CCRP is not used

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

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

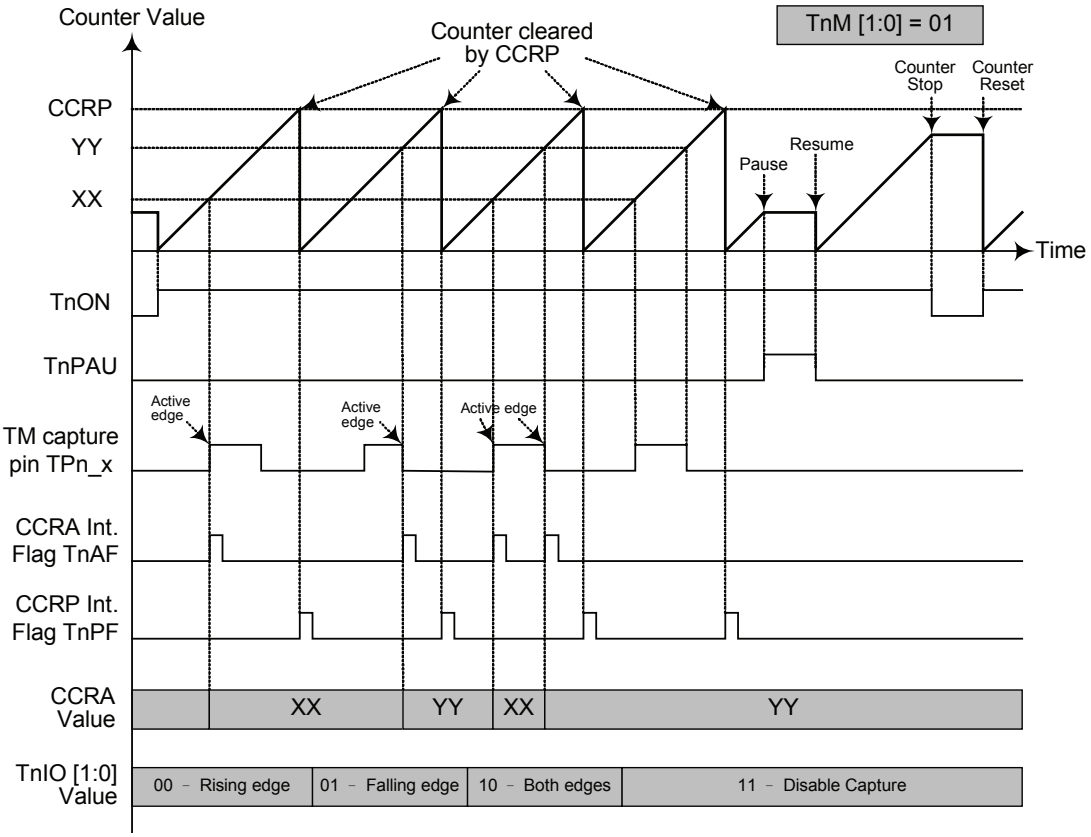
5. In the Single Pulse Mode, TnIO [1:0] must be set to “11” and can not be changed.

Capture Input Mode

To select this mode bits T2M1 and T2M0 in the TM2C1 register should be set to 01 respectively. This mode enables external signals to capture and store the present value of the internal counter and can therefore be used for applications such as pulse width measurements. The external signal is supplied on the TP2_0 or TP2_1 pin, whose active edge can be either a rising edge, a falling edge or both rising and falling edges; the active edge transition type is selected using the T2IO1 and T2IO0 bits in the TM2C1 register. The counter is started when the T2ON bit changes from low to high which is initiated using the application program.

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

As the TP2_0 or TP2_1 pin is pin shared with other functions, care must be taken if the TM is in the Input Capture Mode. This is because if the pin is setup as an output, then any transitions on this pin may cause an input capture operation to be executed. The T2CCLR and T2DPX bits are not used in this Mode.



Capture Input Mode (n=2)

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

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

3. TnCCLR bit not used

4. No output function – TnOC and TnPOL bits are not used

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

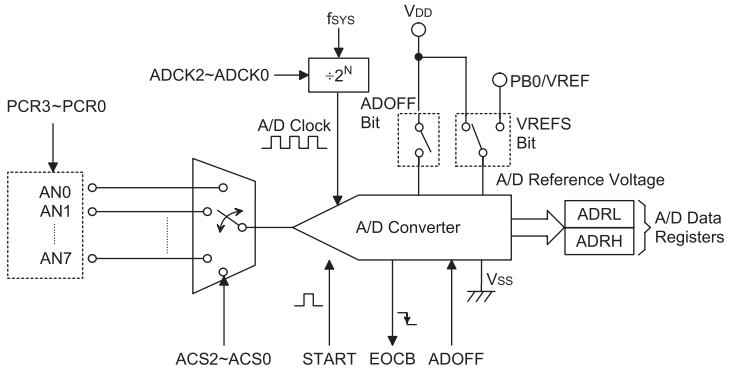
Analog to Digital Converter – ADC

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

A/D Overview

The device contains a multi-channel analog to digital converter, channel 0~3 are for external signal input, channel 4~7 are for the PGA and Battery Detect functions.

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



A/D Converter Structure

A/D Converter Data Registers – ADRL, ADRH

As the internal A/D converter provides a 12 bit digital conversation value it requires two data registers, a high byte register, known as ADRH, and a low byte register, known as ADRL. After the conversion process takes place, these registers can be directly read by the microcontroller to obtain the digitised conversion value. For a single conversion value, that is when the ACCM bit is zero, the 12 bit converted value is stored in bits 0-11. For the conversion accumulation mode, that is when the ACCM bit is high, the full 16-bits are used, therefore the full 16-bits are used.

ADRL 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**: ADC data register bit 7 ~ bit 0

ADRH 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**: ADC data register bit 15 ~ bit 8
When bit ACCM is set to 1, D12 to D15 is valid.

A/D Converter Control Registers – ADCR, ACSR, ANCSR

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

The ANCSR control register contains the PCR3~PCR0 bits which determine which pins on Port B 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. Setting the corresponding bit high will select the A/D input function, clearing the bit to zero will select either the I/O or other pin-shared function. When the pin is selected to be an A/D input, its original function whether it is an I/O or other pin-shared function will be removed. In addition, any internal pull-high resistors connected to these pins will be automatically removed if the pin is selected to be an A/D input.

ADCR Register

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

Bit 7 **START**: Start the A/D conversion
0→1→0: Start
0→1: Reset the A/D converter and set EOCB to "1"
This bit is used to initiate an A/D conversion process. The bit is normally low but if set high and then cleared low again, the A/D converter will initiate a conversion process.
When the bit is set high the A/D converter will be reset.

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

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

Bit 4 Unimplemented, read as "0"

Bit 3 **IDLE_CONV**: MCU halt mode
0: disable
1: enable
When MCU halt mode is enabled, if ADC conversion is in working, the MCU will enter halt mode. And peripheral is not affected.

Bit 2~0 **ACS2~ACS0**: Select A/D channel
000: AN0 (PB0)
001: AN1 (PB1)
010: AN2 (PB2)
011: AN3 (PB3)
100: AN4 (PGA2 output)
101: AN5 (SCF PGA output)
110: AN6 (PGA1 output)
111: AN7 (Battery Voltage Detect input)

ACSR Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	VREFS	ACCM	ADCK2	ADCK1	ADCK0
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 **VREFS**: A/D voltage reference select bit
0: VREF use the internal AVDD, so the external pin is used as an I/O function pin or anlog input.
1: VREF use the external analog input voltage.
Bit 3 **ACCM**: Conversion Accumulation Mode.
0: disable
1: enable
When the Conversion Accumulation function is enabled the A/D interrupt is enabled, the A/D will convert 16 times continuously and the accumulated value will be stored in the full 16-bits of registers ADRL and ADRH. After the 16 conversions have completed, the EOCB bit will be automatically cleared to zero and an A/D converter interrupt generated if it is enabled.
Bit 2~0 **ADCK2~ADCK0**: Select ADC clock source
000: $f_{SYS}/2$
001: $f_{SYS}/8$
010: $f_{SYS}/32$
011: Undefined
100: f_{SYS}
101: $f_{SYS}/4$
110: $f_{SYS}/64$
111: Undefined

ANCSR Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	PCR3	PCR2	PCR1	PCR0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4 Unimplemented, read as "0"
Bit 3 **PCR3**: Define PB3 is A/D input or not
0: Not A/D input
1: A/D input, AN3
Bit 2 **PCR2**: Define PB2 is A/D input or not
0: Not A/D input
1: A/D input, AN2
Bit 1 **PCR1**: Define PB1 is A/D input or not
0: Not A/D input
1: A/D input, AN1
Bit 0 **PCR0**: Define PB0 is A/D input or not
0: Not A/D input
1: A/D input, AN0 or VREF

A/D Operation

The START bit in the ADCR register is used to start and reset the A/D converter. When the microcontroller sets this bit from low to high and then low again, an analog to digital conversion cycle will be initiated. When the START bit is brought from low to high but not low again, the EOCB bit in the ADCR register will be set high and the analog to digital converter will be reset. It is the START bit that is used to control the overall start operation of the internal analog to digital converter.

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

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

Controlling the power on/off function of the A/D converter circuitry is implemented using the ADOFF bit in the ADCR register. This bit must be zero to power on the A/D converter. When the ADOFF bit is cleared to zero to power on the A/D converter internal circuitry a certain delay, as indicated in the timing diagram, must be allowed before an A/D conversion is initiated. Even if no pins are selected for use as A/D inputs by clearing the PCR3~PCR0 bits in the ANCSR registers, if the ADOFF bit is zero then some power will still be consumed. In power conscious applications it is therefore recommended that the ADOFF is set high to reduce power consumption when the A/D converter function is not being used.

The reference voltage supply to the A/D Converter can be supplied from either the positive power supply pin, VDD, or from an external reference sources supplied on pin VREF. The desired selection is made using the VREFS bit. As the VREF pin is pin-shared with other functions, when the VREFS bit is set high, the VREF pin function will be selected and the other pin functions will be disabled automatically.

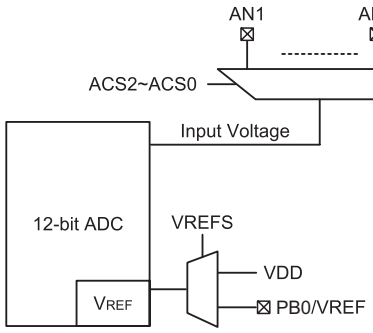
Conversion Accumulation Mode

Each single A/D conversion operation provides a 12-bit wide digital conversion value. If the ACCM bit is zero then only a single A/D conversion process will take place. However if the ACCM bit is set high then the A/D converter will be in the conversion accumulation mode. In this mode, 16 repetitive A/D conversion operations are made with each new converted value being added to the previous accumulated value. After the 16 conversion values, each of 12-bits in length, have completed and added together, a resulting 16-bit wide digital value is obtained and stored in the two A/D data registers. However it must be noted that, although the accumulated result provides a full 16-bit value, only the 13 higher bits, 16 bits will be valid. When in this mode the EOCB bit will be cleared to zero after 16 conversion operations have completed.

A/D Input Pins

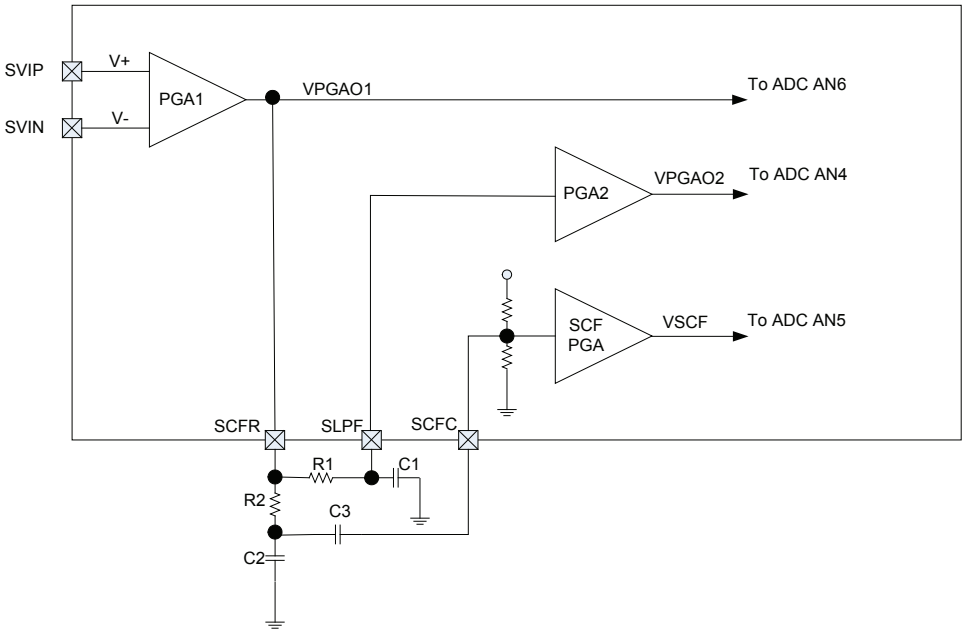
Some of the A/D analog input pins are pin-shared with the I/O pins on Port B as well as other functions. The PCR3~PCR0 bits in the ANCSR register, determine whether the input pins are setup as A/D converter analog inputs or whether they have other functions. If the PCR3~PCR0 bits for its corresponding pin is set high then the pins will be setup to be an A/D converter input and the original pin functions disabled. In this way, pins can be changed under program control to change their function between A/D inputs and other functions. All pull-high resistors, which are setup through register programming, will be automatically disconnected if the pins are setup as A/D inputs. Note that it is not necessary to first setup the A/D pin as an input in the PBC port control register to enable the A/D input as when the PCR3~PCR0 bits enable an A/D input, the status of the port control register will be overridden.

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

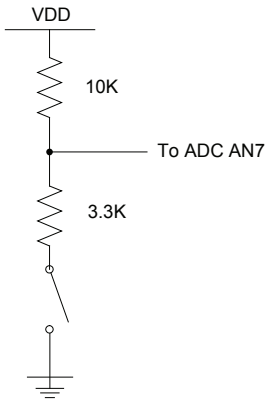


A/D Input Structure

AN4~AN6 are three A/D internal inputs. AN4, AN5 and AN6 are connected to PGA2 output, PGA1output and the SCF PGA output respectively.



AN7 is an internal A/D input which is connected to the battery voltage detector. When the battery voltage detector is enabled the A/D can be used to measure the battery voltage.



Battery Voltage Detector

Summary of A/D Conversion Steps

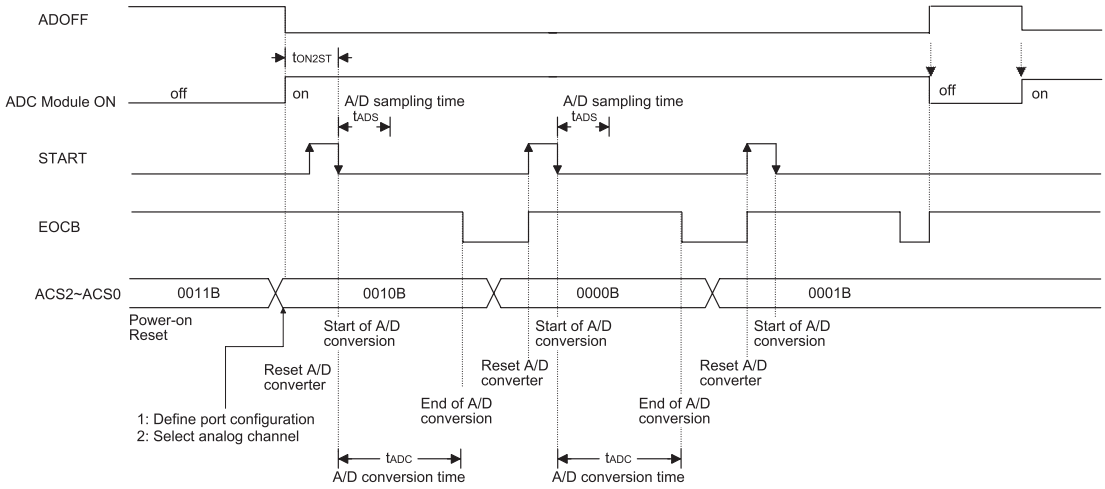
The following summarises the individual steps that should be executed in order to implement an A/D conversion process.

- Step 1
  Select the required A/D conversion clock by correctly programming bits ADCK2~ADCK0 in the ACSR register.
- Step 2
  Enable the A/D by clearing the ADOFF bit in the ADCR register to zero.
- Step 3
  Select which channel is to be connected to the internal A/D converter by correctly programming the ACS2~ACS0 bits which are also contained in the ADCR register.
- Step 4
  Select which pins are to be used as A/D inputs and configure them by correctly programming the PCR3~PCR0 bits in the ANCSR register.
- Step 5
  If the interrupts are to be used, the interrupt control registers must be correctly configured to ensure the A/D converter interrupt function is active. The master interrupt control bit, EMI, and the A/D converter interrupt bit, ADE, must both be set high to do this.
- Step 6
  The analog to digital conversion process can now be initialised by setting the START bit in the ADCR register from low to high and then low again. Note that this bit should have been originally cleared to zero.
- Step 7
  To check when the analog to digital conversion process is complete, the EOCB bit in the ADCR register can be polled. The conversion process is complete when this bit goes low. When this occurs the A/D data registers ADRL and ADRH can be read to obtain the conversion value. As an alternative method, if the interrupts are enabled and the stack is not full, the program can wait for an A/D interrupt to occur.

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

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

Note that when in the Conversion Accumulation Mode, 16 conversion cycles will be required before the EOCB bit is cleared to zero.



A/D Single Conversion Mode Timing

Programming Considerations

During microcontroller operations where the A/D converter is not being used, the A/D internal circuitry can be switched off to reduce power consumption, by setting bit ADOFF high in the ADCR 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.

The power-on reset condition of the A/D converter control registers will ensure that the shared function pins are setup as A/D converter inputs. If any of the A/D converter input pins are to be used for functions, then the A/D converter control register bits must be properly setup to disable the A/D input configuration.

A/D Transfer Function

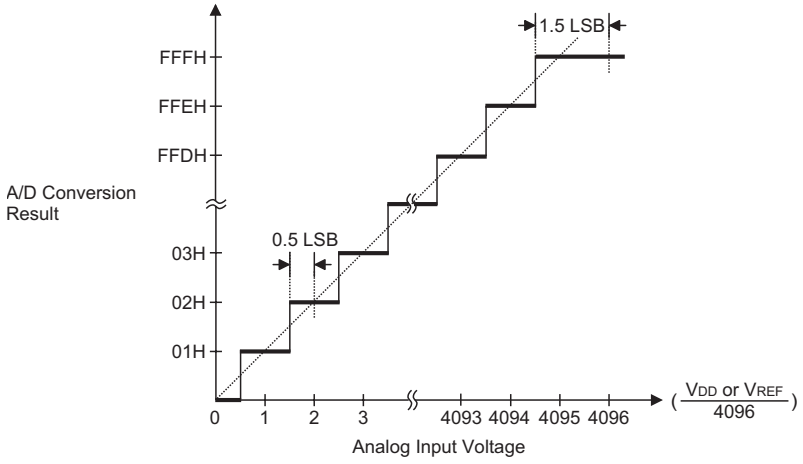
As the device contains a 12-bit A/D converter, its full-scale converted digitised value is equal to FFFH. Since the full-scale analog input value is equal to the V_{DD} or V_{REF} voltage, this gives a single bit analog input value of V_{DD} or V_{REF} divided by 4096.

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

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

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

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



Ideal A/D Transfer Function

A/D Programming Example

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

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

```
clr  ADE                ; disable ADC interrupt
mov  a, 01H
mov  ACSR               ; select fSYS/8 as A/D clock
clr  ADOFF
mov  a, 0FH             ; setup ANCSR to configure pins AN0~AN3
mov  ANCSR,a
mov  a, 00H             ; setup ADCR register to configure Port as A/D inputs
mov  ADCR, a            ; and select AN0 to be connected to the A/D converter
:
:
Start_conversion:
clr  START
set  START              ; reset A/D
clr  START              ; start A/D
Polling_EOC:
sz   EOCB               ; poll the ADCR register EOCB bit to detect end
; of A/D conversion
jmp  polling_EOC        ; continue polling
mov  a, ADRL            ; read low byte conversion result value
mov  adrl_buffer, a     ; save result to user defined register
mov  a, ADRH            ; read high byte conversion result value
mov  adrh_buffer, a     ; save result to user defined register
:
jmp  start_conversion   ; start next A/D conversion
```

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

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

```
clr  ADE                  ; disable ADC interrupt
mov  a, 01H
mov  ACSR                 ; select fSYS/8 as A/D clock
clr  ADOFF
mov  a, 0FH               ; setup ANCSR to configure pins AN0~AN3
mov  ANCSR,a
mov  a, 00H               ; setup ADCR register to configure Port as A/D inputs
mov  ADCR, a              ; and select AN0 to be connected to the A/D
:
:
Start_conversion:
clr  START
set  START                ; reset A/D
clr  START                ; start A/D
clr  ADF                  ; clear ADC interrupt request flag
set  ADE                  ; enable ADC interrupt
set  EMI                  ; enable global interrupt
:
:
; ADC interrupt service routine
ADC_:
mov  acc_stack, a         ; save ACC to user defined memory
mov  a, STATUS
mov  status_stack, a      ; save STATUS to user defined memory
:
:
mov  a, ADRL              ; read low byte conversion result value
mov  adrl_buffer, a       ; save result to user defined register
mov  a, ADRH              ; read high byte conversion result value
mov  adrh_buffer, a       ; save result to user defined register
:
:
EXIT_ISR:
mov  a, status_stack
mov  STATUS, a            ; restore STATUS from user defined memory
mov  a, acc_stack         ; restore ACC from user defined memory
clr  ADF                  ; clear ADC interrupt flag
reti
```

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

Digital to Analog Converter – DAC

The device includes a 16-bit Digital to Analog Converter function. This function allows digital data contained in the device to generate audio signals.

DAC Operation

The voice data must be written into registers DAL and DAH. The audio outputs will be written into the higher nibble of DAL and the whole byte of DAH, and the DAL3~DAL0 is unimplemented and always read as "0H". There are 8 scales of volume controllable level that are provided for the voltage type DAC output. The programmer can change the volume by only writing the volume control data to the higher-nibble of the VOL, and the lower-nibble of VOL is always read as "0H".

The voice control bit DACEN in CTRL2 controls DAC circuit enable/disable. If the DAC circuit is not enabled, any DAH/DAL output is in valid. Writing a "1" to DACEN bit is to enable DAC circuit, and writing a "0" to DACEN bit is to disable DAC circuit.

CTRL2 Register

Bit	7	6	5	4	3	2	1	0
Name	DACEN	CPDEN	INT1ES1	INT1ES0	INT0ES1	INT0ES0	CPF	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	—
POR	0	0	1	0	1	0	0	—

Bit 7 **DACEN**: DAC disable/enable control
0: DAC disable
1: DAC enable
The DAC circuit is disabled when system enters the HALT mode.

Bit 6~0 Described elsewhere

DAL Register

Bit	7	6	5	4	3	2	1	0
Name	DA3	DA2	DA1	DA0	—	—	—	—
R/W	R/W	R/W	R/W	R/W	—	—	—	—
POR	0	0	0	0	—	—	—	—

DAH Register

Bit	7	6	5	4	3	2	1	0
Name	DA11	DA10	DA9	DA8	DA7	DA6	DA5	DA4
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

VOL Register

Bit	7	6	5	4	3	2	1	0
Name	VOL3	VOL2	VOL1	VOL0	—	—	—	—
R/W	R/W	R/W	R/W	R/W	—	—	—	—
POR	0	0	0	0	—	—	—	—

Bit 7~5 **VOL3~VOL1**: Volume control
000: 16-bit DAC D[15:0]=DA11,DA11,DA11,DA11,DA11,DA11,DA11,DA11~DA3
001: 16-bit DAC D[15:0]=DA11,DA11,DA11,DA11,DA11,DA11,DA11~DA2
010: 16-bit DAC D[15:0]=DA11,DA11,DA11,DA11,DA11,DA11~DA1
011: 16-bit DAC D[15:0]=DA11,DA11,DA11,DA11,DA11~DA0
100: 16-bit DAC D[15:0]=DA11,DA11,DA11,DA11~DA0,0
101: 16-bit DAC D[15:0]=DA11,DA11,DA11~DA0,0,0
110: 16-bit DAC D[15:0]=DA11,DA11~DA0,0,0,0
111: 16-bit DAC D[15:0]=DA11~DA0,0,0,0,0

Bit 4 **VOL0**: unused bit, read/writeable
The bit can be used as normal register bit.

Bit 3~0 Unimplemented, read as "0"

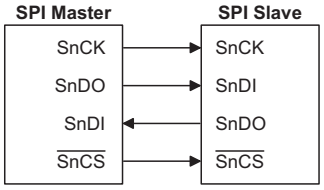
Serial Interface – SPI0, SPI1

The device contains two identical SPI interfaces, known as SPI0 and SPI1. The SPI interface is often used to communicate with external peripheral devices such as sensors, Flash or EEPROM memory devices etc. Originally developed by Motorola, the four line SPI interface is a synchronous serial data interface that has a relatively simple communication protocol simplifying the programming requirements when communicating with external hardware devices.

The communication is full duplex and operates as a slave/master type, where the device can be either master or slave. Although the SPI interface specification can control multiple slave devices from a single master, but this device provides only one $\overline{\text{SnCS}}$ pin. If the master needs to control multiple slave devices from a single master, the master can use an 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 SnDI, SnDO, SnCK and $\overline{\text{SnCS}}$. Pins SnDI and SnDO are the Serial Data Input and Serial Data Output lines, SnCK is the Serial Clock line and $\overline{\text{SnCS}}$ is the Slave Select line. As the SPI interface pins are pin-shared with normal I/O pins, the SPI interface must first be enabled by setting the correct bits in the SPInC0 and SPInC1 registers. Communication between devices connected to the SPI interface is carried out in a slave/master mode with all data transfer initiations being implemented by the master. The Master also controls the clock signal. As the device only contains a single $\overline{\text{SnCS}}$ pin only one slave device can be utilized. The $\overline{\text{SnCS}}$ pin is controlled by software.



SPI Master/Slave Connection

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
- Support IDLE mode, the SPI module can operate after halt instruction is executed

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 CSENn and SPInEN.

There are several configuration options associated with the SPI interface. They are configuration options determine if the CSENn and WCOLn functions are to be used.

Configuration Option	Function
CSENn Function	enable/disable
WCOLn Function	enable/disable

SPI Interface Configuration Options

SPI Control Register

SPI0C0 Register

Bit	7	6	5	4	3	2	1	0
Name	SPI02	SPI01	SPI00	—	—	—	SPI0EN	—
R/W	R/W	R/W	R/W	—	—	—	R/W	—
POR	1	1	1	—	—	—	0	—

Bit 7~5 **SPI02~SPI00**: SPI0 Operating Mode Control
000: SPI0 master mode; SPI0 clock is $f_{SYS}/4$
001: SPI0 master mode; SPI0 clock is $f_{SYS}/16$
010: SPI0 master mode; SPI0 clock is $f_{SYS}/64$
011: SPI0 master mode; SPI0 clock is f_{SUB}
100: SPI0 master mode; SPI0 clock is TM0 CCRP match frequency/4
101: SPI0 slave mode
110: Unused mode
111: Unused mode
These bits setup the overall operating mode of the SPI0 function. As well as selecting if the SPI0 function, they are used to control the SPI0 Master/Slave selection and the SPI0 Master clock frequency. The SPI0 clock is a function of the system clock but can also be chosen to be sourced from the TM0. If the SPI0 Slave Mode is selected then the clock will be supplied by an external Master device.

Bit 4~2 Unimplemented, read as "0"

Bit 1 **SPI0EN**: SPI0 Control
0: Disable
1: Enable
The bit is the overall on/off control for the SPI0 interface. When the SPI0EN bit is cleared to zero to disable the SPI0 interface, the S0DI, S0DO, S0CK and $\overline{\text{S0CS}}$ lines will be in a floating condition and the SPI0 operating current will be reduced to a minimum value. When the bit is high the SPI0 interface is enabled.

Bit 0 Unimplemented, read as "0"

SPI0C1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	CKPOLB0	CKEG0	MLS0	CSEN0	WCOL0	TRF0
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 **CKPOLB0**: Determines the base condition of the SPI0 clock line
0: the S0CK line will be high when the clock is inactive
1: the S0CK line will be low when the clock is inactive
The CKPOLB0 bit determines the base condition of the clock line, if the bit is high then the S0CK line will be low when the clock is inactive. When the CKPOLB0 bit is low then the S0CK line will be high when the clock is inactive.

Bit 4 **CKEG0**: Determines SPI0 S0CK active clock edge type
CKPOLB0=0
0: S0CK is high base level and data capture at S0CK rising edge
1: S0CK is high base level and data capture at S0CK falling edge
CKPOLB0=1
0: S0CK is low base level and data capture at S0CK falling edge
1: S0CK is low base level and data capture at S0CK rising edge
The CKEG0 and CKPOLB0 bits are used to setup the way that the clock signal outputs and inputs data on the SPI0 bus. These two bits must be configured before data transfer is executed otherwise an erroneous clock edge may be generated. The CKPOLB0 bit determines the base condition of the clock line, if the bit is high then the S0CK line will be low when the clock is inactive.
When the CKPOLB0 bit is low then the S0CK line will be high when the clock is inactive. The CKEG0 bit determines active clock edge type which depends upon the condition of CKPOLB0.

Bit 3 **MLS0**: SPI0 Data shift order
0: LSB first
1: MSB first
This is the data shift select bit and is used to select how the data is transferred, either MSB or LSB first. Setting the bit high will select MSB first and low for LSB first.

Bit 2 **CSEN0**: SPI0 $\overline{\text{S0CS}}$ pin Control
0: Disable
1: Enable
The CSEN0 bit is used as an enable/disable for the $\overline{\text{S0CS}}$ pin. If this bit is low then the $\overline{\text{S0CS}}$ pin will be disabled and placed into a floating condition. If the bit is high the $\overline{\text{S0CS}}$ pin will be enabled and used as a select pin. Note that using the CSEN0 bit can be disabled or enabled via configuration option.

Bit 1 **WCOL0**: SPI0 Write Collision flag
0: No collision
1: Collision
The WCOL0 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 SPI0D 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. Note that using the WCOL0 bit can be disabled or enabled via configuration option.

Bit 0 **TRF0**: SPI0 Transmit/Receive Complete flag
0: Data is being transferred
1: SPI0 data transmission is completed
The TRF0 bit is the Transmit/Receive Complete flag and is set "1" automatically when an SPI0 data transmission is completed, but must set to "0" by the application program. It can be used to generate an interrupt.

SPI0D 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**: SPI0 Serial bus data register

SPI1C0 Register

Bit	7	6	5	4	3	2	1	0
Name	SPI12	SPI11	SPI10	—	—	—	SPI1EN	—
R/W	R/W	R/W	R/W	—	—	—	R/W	—
POR	1	1	1	—	—	—	0	—

Bit 7~5 **SPI12~SPI10**: SPI1 Operating Mode Control
000: SPI1 master mode; SPI1 clock is $f_{SYS}/4$
001: SPI1 master mode; SPI1 clock is $f_{SYS}/16$
010: SPI1 master mode; SPI1 clock is $f_{SYS}/64$
011: SPI1 master mode; SPI1 clock is f_{SUB}
100: SPI1 master mode; SPI1 clock is TM0 CCRP match frequency/4
101: SPI1 slave mode
110: Unused mode
111: Unused mode

These bits setup the overall operating mode of the SPI1 function. As well as selecting if the SPI1 function, they are used to control the SPI1 Master/Slave selection and the SPI1 Master clock frequency. The SPI1 clock is a function of the system clock but can also be chosen to be sourced from the f_{SUB} and the TM0. If the SPI1 Slave Mode is selected then the clock will be supplied by an external Master device.

Bit 4~2 Unimplemented, read as “0”

Bit 1 **SPI1EN**: SPI1 Control
0: Disable
1: Enable

The bit is the overall on/off control for the SPI1 interface. When the SPI1EN bit is cleared to zero to disable the SPI1 interface, the S1DI, S1DO, S1CK and $\overline{\text{S1CS}}$ lines will be in a floating condition and the SPI1 operating current will be reduced to a minimum value. When the bit is high the SPI1 interface is enabled.

Bit 0 Unimplemented, read as "0"

SPI1C1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	CKPOLB1	CKEG1	MLS1	CSEN1	WCOL1	TRF1
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 **CKPOLB1**: Determines the base condition of the SPI1 clock line
0: the S1CK line will be high when the clock is inactive
1: the S1CK line will be low when the clock is inactive
The CKPOLB1 bit determines the base condition of the clock line, if the bit is high then the S1CK line will be low when the clock is inactive. When the CKPOLB1 bit is low then the S1CK line will be high when the clock is inactive.

Bit 4 **CKEG1**: Determines SPI1 S1CK active clock edge type
CKPOLB1=0
0: S1CK is high base level and data capture at S1CK rising edge
1: S1CK is high base level and data capture at S1CK falling edge
CKPOLB1=1
0: S1CK is low base level and data capture at S1CK falling edge
1: S1CK is low base level and data capture at S1CK rising edge
The CKEG1 and CKPOLB1 bits are used to setup the way that the clock signal outputs and inputs data on the SPI1 bus. These two bits must be configured before data transfer is executed otherwise an erroneous clock edge may be generated. The CKPOLB1 bit determines the base condition of the clock line, if the bit is high then the S1CK line will be low when the clock is inactive.
When the CKPOLB1 bit is low then the S1CK line will be high when the clock is inactive. The CKEG1 bit determines active clock edge type which depends upon the condition of CKPOLB1.

Bit 3 **MLS1**: SPI1 Data shift order
0: LSB first
1: MSB first
This is the data shift select bit and is used to select how the data is transferred, either MSB or LSB first. Setting the bit high will select MSB first and low for LSB first.

Bit 2 **CSEN1**: SPI1 $\overline{\text{S1CS}}$ pin Control
0: Disable
1: Enable
The CSEN1 bit is used as an enable/disable for the $\overline{\text{S1CS}}$ pin. If this bit is low then the $\overline{\text{S1CS}}$ pin will be disabled and placed into a floating condition. If the bit is high the $\overline{\text{S1CS}}$ pin will be enabled and used as a select pin. Note that using the CSEN1 bit can be disabled or enabled via configuration option.

Bit 1 **WCOL1**: SPI1 Write Collision flag
0: No collision
1: Collision
The WCOL1 flag is used to detect if a data collision has occurred. If this bit is high it means that data has been attempted to be written to the SPI1D register during a data transfer operation. This writing operation will be ignored if data is being transferred. The bit can be cleared by the application program. Note that using the WCOL1 bit can be disabled or enabled via configuration option.

Bit 0 **TRF1**: SPI1 Transmit/Receive Complete flag
0: Data is being transferred
1: SPI1 data transmission is completed
The TRF1 bit is the Transmit/Receive Complete flag and is set "1" automatically when an SPI1 data transmission is completed, but must set to "0" by the application program. It can be used to generate an interrupt.

SPI1D Register

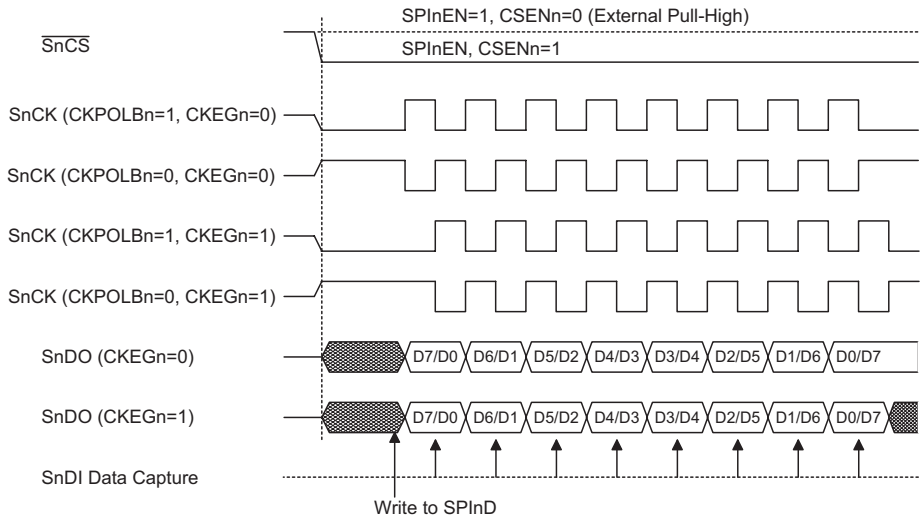
Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x" unknown

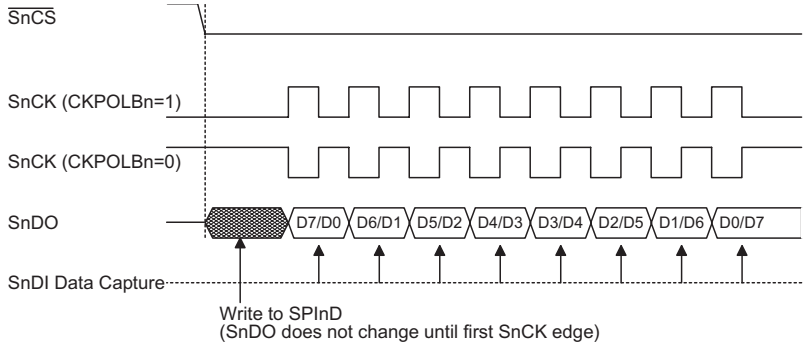
Bit 7~0 **D7~D0**: SPI1 Serial bus data register

SPI Communication

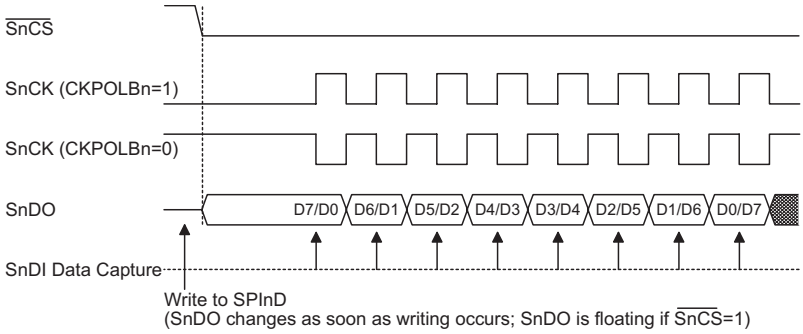
After the SPI interface is enabled by setting the SPInEN bit high, then in the Master Mode, when data is written to the SPInD register, transmission/reception will begin simultaneously. When the data transfer is complete, the TRFn flag will be set automatically, but must be cleared using the application program. In the Slave Mode, when the clock signal from the master has been received, any data in the SPInD register will be transmitted and any data on the SnDI pin will be shifted into the SPInD register. The master should output an $\overline{\text{SnCS}}$ 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 $\overline{\text{SnCS}}$ signal depending upon the configurations of the CKPOLBn bit and CKEGn bit. The accompanying timing diagram shows the relationship between the slave data and $\overline{\text{SnCS}}$ signal for various configurations of the CKPOLBn and CKEGn bits. The SPI will continue to function even in the IDLE Mode.



SPIn Master Mode Timing (n=0 or 1)



SPIn Slave Mode Timing – CKEGn=0 (n=0 or 1)



Note: For SPIn slave mode, if SPInEN=1 and CSENn=0, SPIn is always enabled and ignores the SnCS level.

SPIn Slave Mode Timing – CKEGn=1 (n=0 or 1)

Note: data receiving is still working when MCU enters halt mode

Interrupts

Interrupts are an important part of any microcontroller system. When an external event or an internal function such as a Timer Module or an A/D converter requires microcontroller attention, their corresponding interrupt will enforce a temporary suspension of the main program allowing the microcontroller to direct attention to their respective needs. The device contains several external interrupt and internal interrupts functions. The external interrupt is generated by the action of the external INTn pin, while the internal interrupts are generated by various internal functions such as TMs, Time Base, LVD, SPI, SCF, charge pump and the A/D converter.

No.	Interrupt Source	Priority	Vector
a	External interrupt 0	1	04H
b	Multi function interrupt 0 TM0 comparator A or P match interrupt	2	08H
c	Multi function interrupt 1 TM1 comparator A or P match interrupt	3	0CH
d	Multi function interrupt 2 TM2 comparator A or P match interrupt, EEPROM write interrupt	4	10H
e	A/D interrupt	5	14H
f	Multi-function 3 interrupt (Time Base 0 Interrupt, SPI0 Interrupt, SPI1 Interrupt, INT1 Interrupt)	6	18H
g	Multi-function 4 interrupt (Time Base 1 Interrupt, SCF, LVD Interrupt and Charge pump Interrupt)	7	1CH

Interrupt Subroutine Vector

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 first is the INTC0~INTC1 registers which setup the primary interrupts, the second is the MFIC0~MFIC4 registers which setup the Multi-function interrupts.

Each register contains a number of enable bits to enable or disable individual registers as well as interrupt flags to indicate the presence of an interrupt request. The naming convention of these follows a specific pattern. First is listed an abbreviated interrupt type, then the (optional) number of that interrupt followed by either an "E" for enable/ disable bit or "F" for request flag.

Function	Enable Bit	Request Flag	Notes
Global	EMI	—	—
INTn Pin	INTnE	INTnF	n=0 or 1
A/D Converter	ADE	ADF	—
Multi-function	MFnE	MFnF	n=0~4
Time Base	TBnE	TBnF	n=0 or 1
SPI	SPInE	SPInF	n=0 or 1
LVD	LVE	LVF	—
EEPROM	DEE	DEF	—
SCF interrupt	SCFE	SCFF	—
Charge pump interrupt	CHPE	—	—
TM	TnPE	TnPF	n=0~3
	TnAE	TnAF	n=0~3

Interrupt Register Bit Naming Conventions

INTC0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	MF1F	MF0F	INT0F	MF1E	MF0E	INT0E	EMI
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7 Unimplemented, read as "0"

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

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

Bit 4 **INT0F**: INT0 Interrupt Request Flag
0: No request
1: Interrupt request

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

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

Bit 1 **INT0E**: INT0 Interrupt Control
0: Disable
1: Enable

Bit 0 **EMI**: Global Interrupt Control
0: Disable
1: Enable

INTC1 Register

Bit	7	6	5	4	3	2	1	0
Name	MF4F	MF3F	ADF	MF2F	MF4E	MF3E	ADE	MF2E
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **MF4F**: Multi-function Interrupt 4 Request Flag
0: No request
1: Interrupt request

Bit 6 **MF3F**: Multi-function Interrupt 3 Request Flag
0: No request
1: Interrupt request

Bit 5 **ADF**: A/D Converter Interrupt Request Flag
0: No request
1: Interrupt request

Bit 4 **MF2F**: Multi-function Interrupt 2 Request Flag
0: No request
1: Interrupt request

Bit 3 **MF4E**: Multi-function 4 Interrupt Control
0: Disable
1: Enable

Bit 2 **MF3E**: Multi-function 3 Interrupt Control
0: Disable
1: Enable

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

Bit 0 **MF2E**: Multi-function 2 Interrupt Control
0: Disable
1: Enable

MFIC0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	DEF	T0AF	T0PF	—	DEE	T0AE	T0PE
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 **DEF**: Data EEPROM interrupt request flag
0: No request
1: Interrupt request

Bit 5 **T0AF**: TM0 Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **T0PF**: TM0 Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 3 Unimplemented, read as "0"

Bit 2 **DEE**: Data EEPROM interrupt control
0: Disable
1: Enable

Bit 1 **T0AE**: TM0 Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **T0PE**: TM0 Comparator P match interrupt control
0: Disable
1: Enable

MFIC1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	T1AF	T1PF	—	—	T1AE	T1PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5 **T1AF**: TM1 Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **T1PF**: TM1 Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 3~2 Unimplemented, read as "0"

Bit 1 **T1AE**: TM1 Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **T1PE**: TM1 Comparator P match interrupt control
0: Disable
1: Enable

MFIC2 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	T2AF	T2PF	—	—	T2AE	T2PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5 **T2AF**: TM2 Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **T2PF**: TM2 Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 3~2 Unimplemented, read as "0"

Bit 1 **T2AE**: TM2 Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **T2PE**: TM2 Comparator P match interrupt control
0: Disable
1: Enable

MFIC3 Register

Bit	7	6	5	4	3	2	1	0
Name	INT1F	SPI1F	SPI0F	TB0F	INT1E	SPI1E	SPI0E	TB0E
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 **INT1F**: INT1 Interrupt Request Flag
0: No request
1: Interrupt request

Bit 6 **SPI1F**: SPI1 Interrupt Request Flag
0: No request
1: Interrupt request

Bit 5 **SPI0F**: SPI0 Interrupt Request Flag
0: No request
1: Interrupt request

Bit 4 **TB0F**: Time Base 0 Interrupt Request Flag
0: No request
1: Interrupt request

Bit 3 **INT0E**: INT1 Interrupt Control quest
0: Disable
1: Enable

Bit 2 **SPI1E**: SPI1 Interrupt Control
0: Disable
1: Enable

Bit 1 **SPI0E**: SPI0 Interrupt Control
0: Disable
1: Enable

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

MFIC4 Register

Bit	7	6	5	4	3	2	1	0
Name	LVF	TB1F	—	SCFF	LVE	TB1E	CHPE	SCFE
R/W	R/W	R/W	—	R/W	R/W	R/W	R/W	R/W
POR	0	0	—	1	0	0	0	0

Bit 7 **LVF**: LVD Interrupt Request Flag
0: No request
1: Interrupt request

Bit 6 **TB1F**: Time Base 1 Interrupt Request Flag
0: No request
1: Interrupt request

Bit 5 This bit forbids writing.

Bit 4 **SCFF**: SCF Interrupt Request Flag
0: No request
1: Interrupt request

Bit 3 **LVE**: LVD Interrupt Control
0: Disable
1: Enable

Bit 2 **TB1E**: Time Base 1 Interrupt Control
0: Disable
1: Enable

Bit 1 **CHPE**: Charge pump Interrupt Control
0: Disable
1: Enable

Bit 0 **SCFE**: SCF Interrupt Control
0: Disable
1: Enable

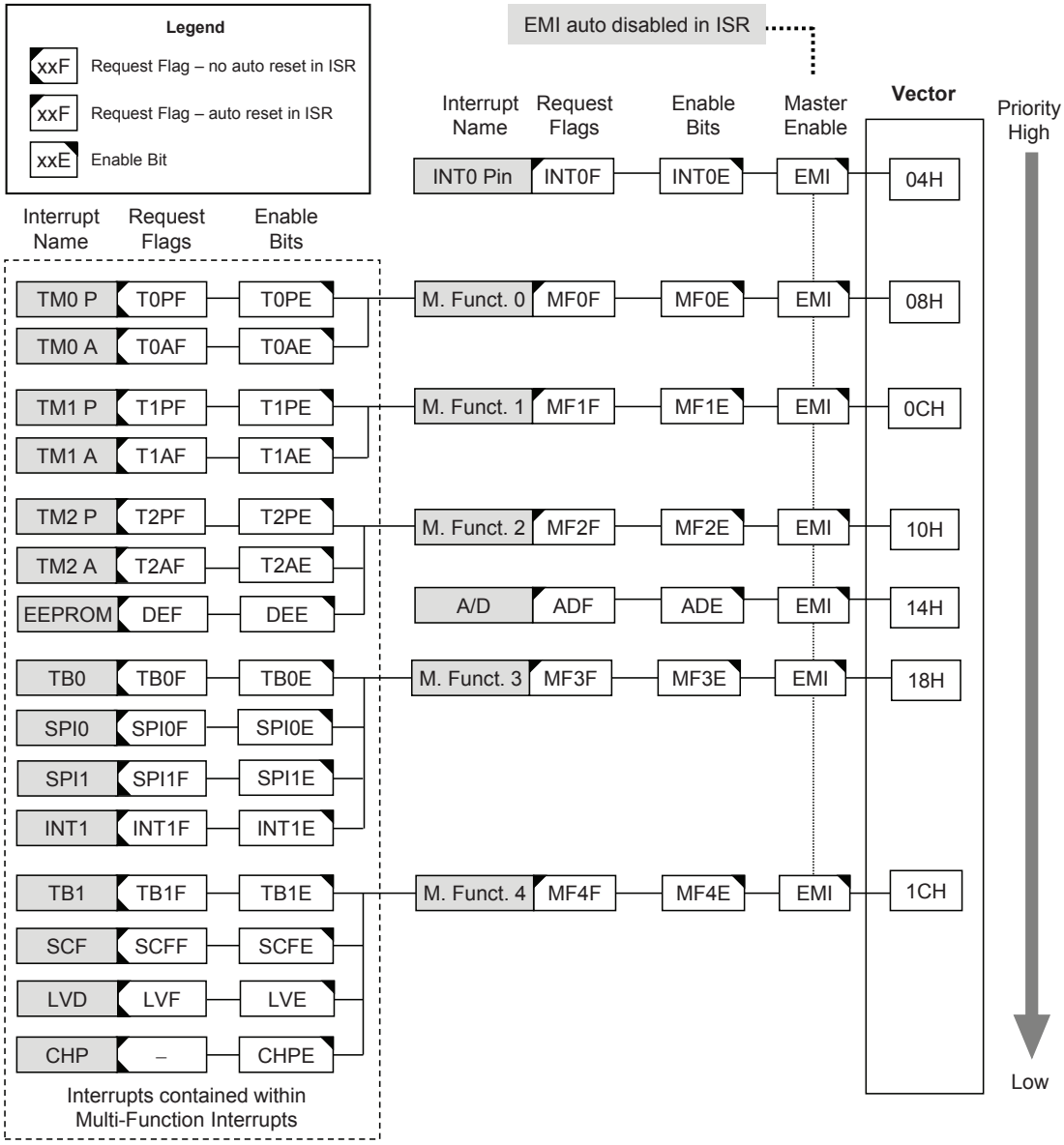
Interrupt Operation

When the conditions for an interrupt event occur, such as a TM Comparator P, Comparator A match or A/D conversion completion etc, the relevant interrupt request flag will be set. Whether the request flag actually generates a program jump to the relevant interrupt vector is determined by the condition of the interrupt enable bit. If the enable bit is set high then the program will jump to its relevant vector, if the enable bit is zero then although the interrupt request flag is set an actual interrupt will not be generated and the program will not jump to the relevant interrupt vector. The global interrupt enable bit, if cleared to zero, will disable all interrupts.

When an interrupt is generated, the Program Counter, which stores the address of the next instruction to be executed, will be transferred onto the stack. The Program Counter will then be loaded with a new address which will be the value of the corresponding interrupt vector. The microcontroller will then fetch its next instruction from this interrupt vector. The instruction at this vector will usually be a "JMP" which will jump to another section of program which is known as the interrupt service routine. Here is located the code to control the appropriate interrupt. The interrupt service routine must be terminated with a "RETI", which retrieves the original Program Counter address from the stack and allows the microcontroller to continue with normal execution at the point where the interrupt occurred.

The various interrupt enable bits, together with their associated request flags, are shown in the Accompanying diagrams with their order of priority. Some interrupt sources have their own individual vector while others share the same multi-function interrupt vector. Once an interrupt subroutine is serviced, all the other interrupts will be blocked, as the global interrupt enable bit, EMI bit will be cleared automatically. This will prevent any further interrupt nesting from occurring. However, if other interrupt requests occur during this interval, although the interrupt will not be immediately serviced, the request flag will still be recorded.

If an interrupt requires immediate servicing while the program is already in another interrupt service routine, the EMI bit should be set after entering the routine, to allow interrupt nesting. If the stack is full, the interrupt request will not be acknowledged, even if the related interrupt is enabled, until the Stack Pointer is decremented. If immediate service is desired, the stack must be prevented from becoming full. In case of simultaneous requests, the accompanying diagram shows the priority that is applied. All of the interrupt request flags when set will wake-up the device if it is in SLEEP or IDLE Mode, however to prevent a wake-up from occurring the corresponding flag should be set before the device is in SLEEP or IDLE Mode.



Interrupt Structure

External Interrupt

The device contains two external interrupts, each of which is controlled by signal transitions on the INTn pin. The INT0 interrupt is an independent interrupt, the INT1 interrupt is contained within the Multi-function Interrupt. An external interrupt request will take place when the external interrupt request flag, INTnF, is set, which will occur when a transition, whose type is chosen by the edge select bits, appears on the external interrupt pin. To allow the program to branch to the INT0 interrupt vector address, the global interrupt enable bit, EMI, and respective external interrupt enable bit, INT0E, must first be set. When the interrupt is enabled, the stack is not full and the correct transition type appears on the external interrupt pin, a subroutine call to the external interrupt vector, will take place. When the interrupt is serviced, the external interrupt request flag, INT0F, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts.

To allow the program to branch to the INT1 interrupt vector address, the global interrupt enable bit, EMI, respective external interrupt enable bit, INT1E, and relevant Multi-function Interrupt enable bit must first be set. When the INT1 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 external interrupt request flag, INT1F, will not be automatically cleared, it has to be cleared by the application program. Additionally the correct interrupt edge type must be selected using the related 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 pin, it can only be configured as external interrupt pin if the 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. Note that any pull-high resistor selections on the external interrupt pin will remain valid even if the pin is used as an external interrupt input.

The bits in CTRL2 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 this register can also be used to disable the external interrupt function.

CTRL2 Register

Bit	7	6	5	4	3	2	1	0
Name	DACEN	CPDEN	INT1ES1	INT1ES0	INT0ES1	INT0ES0	CPF	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	—
POR	0	0	1	0	1	0	0	—

Bit 7~6 Described elsewhere

Bit 5~4 **INT1ES1, INT1ES0**: external interrupt 1 edge selection
00: disable
01: rising edge trigger
10: falling edge trigger
11: dual edge trigger

Bit 3~2 **INT0ES1, INT0ES0**: external interrupt 0 edge selection
00: disable
01: rising edge trigger
10: falling edge trigger
11: dual edge trigger

Bit 1~0 Described elsewhere.

A/D Converter Interrupt

The device contains an A/D converter which has its own independent interrupt. The A/D Converter Interrupt is controlled by the termination of an A/D conversion process. An A/D Converter Interrupt request will take place when the A/D Converter Interrupt request flag, ADF, is set, which occurs when the A/D conversion process finishes. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and A/D Interrupt enable bit, ADE, must first be set. When the interrupt is enabled, the stack is not full and the A/D conversion process has ended, a subroutine call to the A/D Converter Interrupt vector, will take place. When the interrupt is serviced, the A/D Converter Interrupt flag, ADF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

Multi-function Interrupt

Within this device there are five Multi-function interrupts. Unlike the other independent interrupts, these interrupts have no independent source, but rather are formed from other existing interrupt sources.

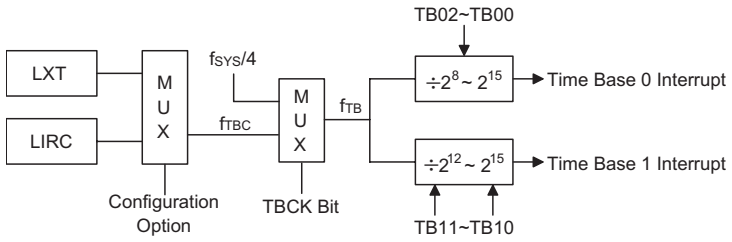
A Multi-function interrupt request will take place the Multi-function interrupt request flag, MFnF is set. The Multi-function interrupt flag will be set when any of its included functions generate an interrupt request flag. To allow the program to branch to its respective interrupt vector address, when 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 the Multi-function interrupt vector will take place. When the interrupt is serviced, the related Multi-Function request flag, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts.

However, it must be noted that, although the Multi-function Interrupt 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 Interrupt, LVD Interrupt, EEPROM Interrupt, Time-base Interrupt, external 1 interrupt, SPIn interrupts, SCF interrupt and Charge Pump interrupt will not be automatically reset and must be manually reset by the application program.

Time Base Interrupt

The Time-base Interrupt, is contained within the Multi-function Interrupt. The function of the Time Base Interrupts is to provide regular time signal in the form of an internal interrupt. They are controlled by the overflow signals from their respective timer functions. When these happens their respective interrupt request flags, TB0F or TB1F will be set. To allow the program to branch to their respective interrupt vector addresses, the global interrupt enable bit, EMI , multi-function enable bit, MFnE and Time Base enable bits, TB0E or TB1E, must first be set. When the interrupt is enabled, the stack is not full and the Time Base overflows, a subroutine call to their respective vector locations will take place. When the interrupt is serviced, the 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 TBnF flag will not be automatically cleared, it has to be cleared by the application program.

The purpose of the Time Base Interrupt is to provide an interrupt signal at fixed time periods. Their clock sources originate from the internal clock source f_{TB}. This f_{TB} input clock passes through a divider, the division ratio of which is selected by programming the appropriate bits in the TBC register to obtain longer interrupt periods whose value ranges. The clock source that generates f_{TB}, which in turn controls the Time Base interrupt period, can originate from several different sources, as shown in the System Operating Mode section.



Time Base Interrupt

TBC Register

Bit	7	6	5	4	3	2	1	0
Name	TBON	TBCK	TB11	TB10	LXTLP	TB02	TB01	TB00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	1	0	1	1	1

Bit 7 **TBON**: TB0 and TB1 Control
0: Disable
1: Enable

Bit 6 **TBCK**: Select f_{TB} Clock
0: f_{TBC}
1: $f_{SYS}/4$

Bit 5~4 **TB11, TB10**: Select Time Base 1 Time-out Period
00: $4096/f_{TB}$
01: $8192/f_{TB}$
10: $16384/f_{TB}$
11: $32768/f_{TB}$

Bit 3 **LXTLP**: LXT Low Power Control
0: Disable (LXT quick start-up)
1: Enable (LXT slow start-up)

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

SPI Interrupts

The SPI1 and SPI0 Interrupts, are contained within the Multi-function Interrupt. When the SPI data transmit has been finished, their respective interrupt request flags, SPInF will be set. To allow the program to branch to their respective interrupt vector addresses, the global interrupt enable bit, EMI, multi-function enable bit, MFnE and SPI interrupt enable bits, SPInE, must first be set. When the interrupt is enabled, the stack is not full and the SPI transmit data finished, a subroutine call to the respective Multi-function Interrupt vector, will take place. When the 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 SPInF flags will not be automatically cleared, they have to be cleared by the application program.

LVD Interrupt

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

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, and 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 EEPROM 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.

TM Interrupts

The Compact and Standard Type TMs have two interrupts each. All of the TM interrupts are contained within the Multi-function Interrupts. For each of the Compact and Standard Type TMs there are two interrupt request flags TnPF and TnAF and two enable bits TnPE and TnAE. A TM interrupt request will take place when any of the TM request flags are set, a situation which occurs when a TM comparator P or A match situation happens.

To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, respective TM Interrupt enable bit, and relevant Multi-function Interrupt enable bit, MFnE, must first be set. When the interrupt is enabled, the stack is not full and a TM comparator match situation occurs, a subroutine call to the relevant Multi-function Interrupt vector locations, will take place. When the TM interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the related MFnF flag will be automatically cleared. As the TM interrupt request flags will not be automatically cleared, they have to be cleared by the application program.

SCF Interrupt

The SCF interrupt, is contained within the Multi-function interrupt. When SCF PGA output a rising edge, its interrupt request flag, SCFF will be set. To allow the program to branch to their respective interrupt vector addresses, the global interrupt enable bit, EMI, multi-function enable bit, MFnE and SCF interrupt enable bit, SCFE, must first be set. When the interrupt is enabled, the stack is not full and SCF PGA output a rising edge, a subroutine call to the respective Multi-function Interrupt vector, will take place. When the 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 SCFF flag will not be automatically cleared, it has to be cleared by the application program.

Charge Pump Interrupt

The charge pump Interrupt, is contained within the Multi-function Interrupt. To allow the program to branch to their respective interrupt vector addresses, the global interrupt enable bit, EMI, multi-function enable bit, MFnE and charge pump interrupt enable bit, CHPE, must first be set. When the interrupt is enabled, the stack is not full and the input voltage of charge pump is greater than 3.8V, a subroutine call to the respective Multi-function Interrupt vector, will take place. When the 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.

Interrupt Wake-up Function

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

Programming Considerations

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

Where a certain interrupt is contained within a Multi-function interrupt, then when the interrupt service routine is executed, as only the Multi-function interrupt request flags, 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 enabled the device to monitor the power supply voltage, V_{DD}, and provide a warning signal should it fall below a certain level. This function may be especially useful in battery applications where the supply voltage will gradually reduce as the battery ages, as it allows an early warning battery low signal to be generated. The Low Voltage Detector also has the capability of generating an interrupt signal.

LVD Register

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

LVDC Register

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

Bit 7~6 Unimplemented, read as "0"

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

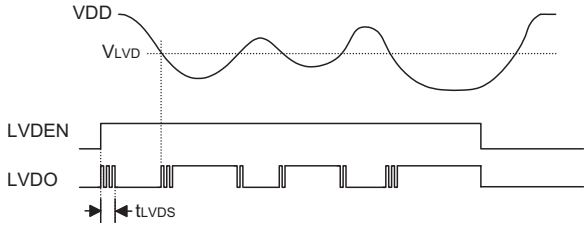
Bit 4 **LVDEN**: Low Voltage Detector Control
0: Disable
1: Enable

Bit 3 Unimplemented, read as “0”

Bit 2~0 **VLVD2~VLVD0**: Select LVD Voltage
000: 2.0V
001: 2.2V
010: 2.4V
011: 2.7V
100: 3.0V
101: 3.3V
110: 3.6V
111: 4.0V

LVD Operation

The Low Voltage Detector function operates by comparing the power supply voltage, V_{DD}, with a pre-specified voltage level stored in the LVDC register. This has a range of between 2.0V and 4.0V. When the power supply voltage, V_{DD}, falls below this pre-determined value, the LVDO bit will be set high indicating a low power supply voltage condition. The Low Voltage Detector function is supplied by a reference voltagewhich will be automatically enabled.When the device is powered down the low voltage detector will remain active if the LVDEN bit is high. After enabling the Low Voltage Detector, a time delay t_{LVDS} should be allowed for the circuitry to stabilise before reading the LVDO bit. Note also that as the V_{DD} voltage may rise and fall rather slowly, at the voltage nears that of V_{LVD}, there may be multiple bit LVDO transitions.



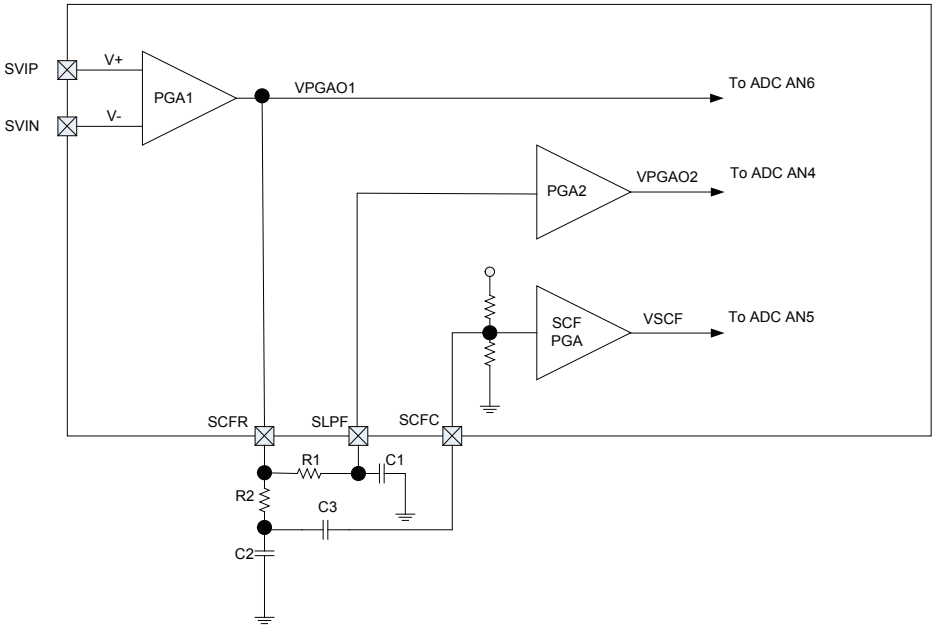
LVD Operation

The Low Voltage Detector also has its own interrupt which is contained within one of the Multi-function interrupts, providing an alternative means of low voltage detection, in addition to polling the LVDO bit. The interrupt will only be generated after a delay of t_{LVD} after the LVDO bit has been set high by a low voltage condition. When the device is powered down the Low Voltage Detector will remain active if the LVDEN bit is high. In this case, the LVF interrupt request flag will be set, causing an interrupt to be generated if V_{DD} falls below the preset LVD voltage. This will cause the device to wake-up from the SLEEP or IDLE Mode, however if the Low Voltage Detector wake up function is not required then the LVF flag should be first set high before the device enters the SLEEP or IDLE Mode.

When LVD function is enabled, it is recommenced to clear LVD flag first, and then enables interrupt function to avoid mistake action.

Programmable Gain Amplifier and Switched Capacitor Filter

The device includes several Programmable Gain Amplifiers, PGAs, as well as a Switched Capacitor Filter, SCF, bandpass filter for sensor signal processing.



PGA Block Diagram

PGA Operation

The overall enable/disable function of the PGA is controlled by the PGAEN bit in the PGAC0 register. When this bit is low the PGA circuits will be powered down, which will be an important consideration in battery powered equipment. The PGA functions include two PGAs, PGA1 and PGA2 as well as a combined PGA and switched capacitor filter. Note: When using the PGA function, the user must wait for about 15ms after enabling the Regulator. Then setup the PGA options and wait for about 400ms, before starting the ADC conversion. In this way the correct ADC conversion value can be obtained.

PGA1

The PGA1 input voltage range is from -20mV to 100mV. The gain of PGA1 is subdivided into two stages. The first stage, known as G11, has a value of 8, 16, 32 or 64 and is controlled by the PGAC0 register. The second stage, known as G12, and has a value of (32~63)/32 and is controlled by the PGAC1 register. The PGA1 Offset Voltage, VOF1, can be (PGAVREF[4:0]/32)×VOREG, giving a range of 32 discrete values, and is controlled by the PGAC0 register. The PGA1 output voltage, VPGAO1, is given by the following formula:

$$VPGAO1 = (V_{+} - V_{-}) \times G11 \times G12 + VOF1$$

PGA2

The gain of PGA2 is fixed at a value of 2. The PGA2 offset voltage, VOF2, is given by the following formula:

$$VOF2 = VREG(3.3V) \times (0\sim255)/256$$

The PGA2 output voltage, VPGA02, is given by the following formula:

$$VPGAO2 = (VPGAO1 \times 2) – VOF2$$

It is the VPGA02 signal that is provided to the A/D converter for blood pressure measuring.

Switched Capacitor Filter

The device includes a band pass switch capacitor filter. The SCF PGA input voltage range, Vin, is 30μV~150μV. In the SCF the Low Pass Filter cut off frequency can be selected to be: 9Hz, 10Hz, 11Hz, 12Hz, selected using the SCFC0 register. The high pass filter cut off frequency is fixed at 0.7Hz. The SCF also includes a PGA function whose gain can be setup using the SCFC0 register to have a value between 56 and 308, selected in 64 discrete stages, using bits in the SCFC0 register. The SCF gain is given by the following formula:

$$\text{SCF Gain} = 56 + 4 \times \text{SCFA}[5:0]$$

The SCF requires a clock with a frequency of about 488Hz. The SCF is sourced from the system clock and subdivided using bits in the SCFC1 and SFCKD registers. The SCF clock is given by the following formula:

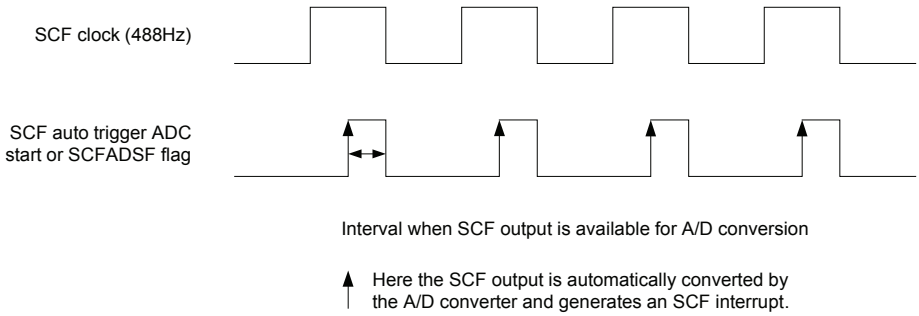
$$\text{SCF clock} = f_{SCF} / (\text{SCFCKD} + 1)$$

where SCFCKD represent the decimal value of the 8 bits in the SCFCKD register, and where f_{SCF} = $f_{SYS}/8$, $f_{SYS}/16$, $f_{SYS}/32$, $f_{SYS}/64$, $f_{SYS}/128$, $f_{SYS}/256$, $f_{SYS}/512$, $f_{SYS}/1024$ selected using bits in the SCFC1 register.

SCF Signal Automatic A/D Conversion

The SCF is connected to the A/D converter and can be setup to execute automatic A/D conversion for the SCF output signal. When the auto measure A/D Converter Enable/Disable bit, SCFADC, is high, then the A/D converter will automatically select the SCF output as its input channel and will override any application program A/D channel selection configuration. Here, an A/D conversion process, initiated by the application program will still be effective. If an A/D conversion process, initiated by either the SCF or application program, is still in progress, and another A/D initiation request (i.e. START) occurs, then this last A/D request will always have priority.

The accompanying diagram shows the relationship between the automatic A/D conversion process and the SCF clock.



Automatic SCF Signal A/D Converter Timing

PGA Control Register

PGAC0 Register

Bit	7	6	5	4	3	2	1	0
Name	PGAEN	PGAA01	PGAA00	PGAVREF4	PGAVREF3	PGAVREF2	PGAVREF1	PGAVREF0
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 **PGAEN**: enable/disable
0: disable – all PGA and SCF circuits powered down
1: enable

Bit 6~5 **PGAA01, PGAA00**: PGA1 first stage gain selection
00: 8
01: 16
10: 32
11: 64

Bit 4~0 **PGAVREF4~PGAVREF0**: PGA1 Offset Voltage selection
PGA1 Offset Voltage = ([PGAVREF4:0] / 32) * VOREG

PGAC1 Register

Bit	7	6	5	4	3	2	1	0
Name	CPEN	BIPGA1	BIPGA0	PGAA14	PGAA13	PGAA12	PGAA11	PGAA10
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 **CPEN**: PGA1's chopper enabled/disabled control
0: disabled
1: enabled
When chopper is enabled, the system will supply a 12kHz to PAG1 for chopper operation.

Bit 6~5 **BIPGA1, BIPGA0**: low-noise OPA bias current control

Bit 4~0 **PGAA14~PGAA10**: PGA1 second stage gain selection
Gain range: 0 ~ 31
Gain = (1 + [PGAA14:PAGA10] / 32)

PGAC2 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**: PAG2 Offset Voltage selection
PGA2 Offset Voltage = (3.3V x [PGAC2[7:0] / 256)

SCFC0 Register

Bit	7	6	5	4	3	2	1	0
Name	SCFLF1	SCFLF0	SCFA5	SCFA4	SCFA3	SCFA2	SCFA1	SCFA0
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 **SCFLF1~SCFLF0**: SCF Low Pass Filter cutoff frequency selection
Band-pass filter – 3dB selection:
00: 9Hz
01: 10Hz
10: 11Hz
11: 12Hz

Bit 5~0 **SCFA5~SCFA0**: SCF Gain selection
Gain range: 0~63
Gain = 56 + 4 x SCFA[5:0]

SCFC1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	SCFADSF	SCFADC	SCFCKP2	SCFCKP1	SCFCKP0	—	—
R/W	—	R	R/W	R/W	R/W	R/W	—	—
POR	—	0	0	0	0	0	—	—

Bit 7 Unimplemented, read as "0"

Bit 6 **SCFADSF**: Enable A/D Converter to measure SCF output signal
0: Disable A/D Converter to measure SCF output signal
1: Enable A/D Converter to measure SCF output signal

Bit 5 **SCFADC**: A/D Converter Enable/Disable auto measure SCF output signal
0: Disable
1: Enable – A/D converter will automatically initiate an SCF signal A/D conversion

Bit 4~2 **SCFCKP2~SCFCKP0**: SCF clock prescaler
000: $f_{SYS}/8$
001: $f_{SYS}/16$
010: $f_{SYS}/32$
011: $f_{SYS}/64$
100: $f_{SYS}/128$
101: $f_{SYS}/256$
110: $f_{SYS}/512$
111: $f_{SYS}/1024$

Bit 1~0 Unimplemented, read as “0”

SCFCKD Register

Bit	7	6	5	4	3	2	1	0
Name	SCFCKD7	SCFCKD6	SCFCKD5	SCFCKD4	SCFCKD3	SCFCKD2	SCFCKD1	SCFCKD0
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

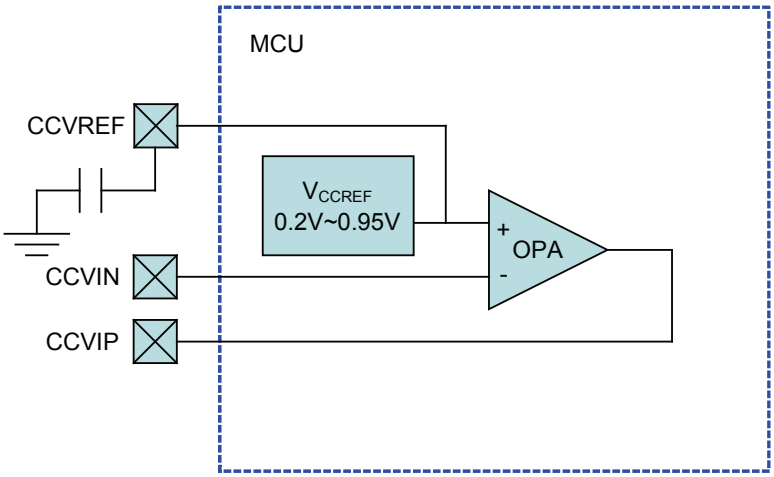
Bit 7~0 **SCFCKD7~SCFCKD0**: The SCF circuit clock divider.
These 8 bits can form the clock divided by 1-256.
Following the below equation: SCF clock = f_{SCF} / (SCFCKD+1)

Sensor Constant Current Generator Circuit

The device includes a constant current generator for driving the pressure bridge sensor. This is formed using a programmable voltage reference and operational amplifier.

Constant Current Generator Operation

The constant current generator must be first enabled by setting the CCVREFEN bit in the CCVREFC register high. If cleared to zero then the circuits will be powered down, which will be an important consideration in battery powered equipment. The VCCREF circuit output voltage can be selected to be between 0.2V~0.95V, in intervals of 0.05V, providing 16 selections using bits in the CCVREFC register. This voltage is available on pin CCVREF to which a capacitor should be connected for stabilisation purposes. This voltage is provided to the positive input of an internal operational amplifier. By connecting an external resistor to the negative operational amplifier input, a constant current can be setup.



Constant Current Generator Block Diagram

Pressure Sensor Constant Current Control Register

CCVREFC Register

Bit	7	6	5	4	3	2	1	0
Name	CCVREFEN	BATEN	—	—	CCVREF3	CCVREF2	CCVREF1	CCVREF0
R/W	R/W	R/W	—	—	R/W	R/W	R/W	R/W
POR	0	0	—	—	0	0	0	0

Bit 7 **CCVREFEN**: Sensor fixed current circuit enable/disable
0: disable – constant current circuit powered down
1: enable

Bit 6 **BATEN**: Battery voltage sense circuit enable/disable
0: disable – battery detect circuit powered down
1: enable

Bit 5~4 Unimplemented, read as "0"

Bit 3~0 **CCVREF3~CCVREF0**: Sensor constant current generator voltage reference selection
0000: 0.2V
0001: 0.25V
0010: 0.3V
0011: 0.35V
0100: 0.4V
0101: 0.45V
0110: 0.5V
0111: 0.55V
1000: 0.6V
1001: 0.65V
1010: 0.7V
1011: 0.75V
1100: 0.8V
1101: 0.85V
1110: 0.9V
1111: 0.95V

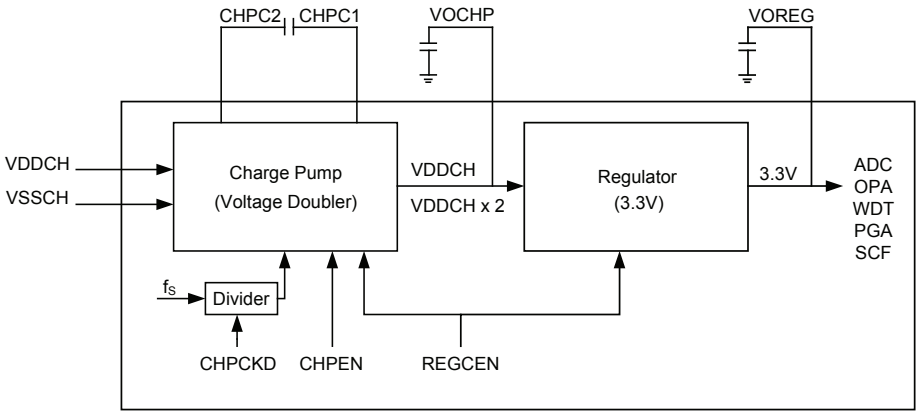
Charge Pump and Voltage Regulator

There device includes a charge pump and one voltage regulator for generation of a constant 3.3V voltage.

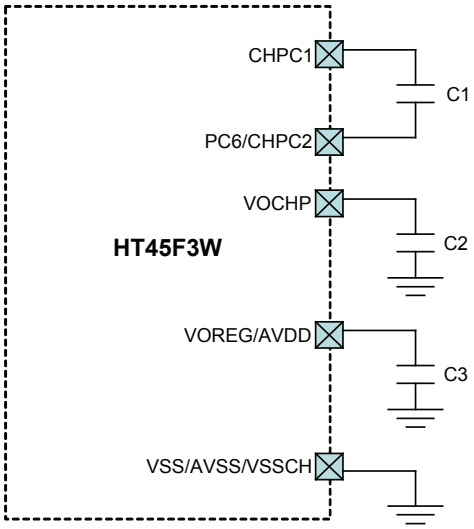
Operation

The charge pump can be enabled or disabled depending on the application. The charge pump acts as a voltage doubler with VDD as its input. The output voltage of charge pump will therefore be VDD×2. This voltage is provided to the regulator input which will in turn generate a stable 3.3V voltage, which can be used by the A/D converter or as an external sensor excitation voltage or other applications. It is necessary that the charge pump provides a minimum output voltage of greater than 3.6V to provide the regulator with its minimum operating voltage for proper 3.3V voltage generation.

The block diagram of these two functions is shown below:



The charge pump application circuit is shown below:



Charge pump Circuit

Note: 1. C1 uses 10μF MLCC capacitor.

2. C2 and C3 use 1μF~10μF Electrolytic capacitor

3. When entering the HALT mode, the hardware will turn off the Charge pump automatically to prevent the charge pump output being in an unknown state. During this time the LCD cannot be used.

A single register, CHPRC, controls overall operation of the charge pump and regulator, controlling functions such as the charge pump on/off, regulator on/off and determines the clock divider value to generate the charge pump clock frequency.

The REGCEN bit in the CHPRC register is the Regulator/Charge-Pump module enable/disable control bit. If this bit is disabled, then the regulator and charge pump will both be disabled to save power. If this bit is set to "0" then the charge pump and regulator module will be powered down irrespective of the CHPEN bit setting. If the REGCEN bit is set to "1", the regulator will be enabled. If the CHPEN is enabled, the charge pump will be active and will use VDD as its input, to generate the double voltage output. The double voltage will be used as the regulator input. If the CHPEN bit is set to "0", the charge pump is disabled and the charge pump output will be equal to the charge pump input VDD.

When using the regulator, it is necessary to take note of the V_{DD} voltage. If the voltage is under 3.6V then, the CHPEN bit should be set high to enable the charge pump; otherwise CHPEN should be cleared to zero. If the charge pump is disabled and the V_{DD} is under 3.6V, then the output voltage of the regulator will not be guaranteed.

CHPRC Register

Bit	7	6	5	4	3	2	1	0
Name	CHPCKD4	CHPCKD3	CHPCKD2	CHPCKD1	CHPCKD0	—	CHPEN	REGCEN
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 7~3 **CHPCKD4~CHPCKD0**: Charge pump clock divider
Clock division select, range of 1-32.
Charge Pump clock = (f_{SYS}/16) / (CHPCKD+1)
Charge Pump clock should be selected to be as near 20kHz as possible.

Bit 2 Unimplemented, read as "0"

Bit 1 **CHPEN**: Charge Pump Enable/Disable Control
0: disable
1: enable

Bit 0 **REGCEN**: Regulator Enable/Disable Control
0: disable
1: enable

REGCEN (control signal)	CHPEN (control signal)	Charge Pump	VOCHP Pin	Regulator	VOREG Pin	Description
0	0	OFF	VDD	OFF	Hi-Impedance	Complete module disabled, OPA/ADC loses power
0	1	OFF	VDD	OFF	VDD	Complete module disabled, OPA/ADC loses power
1	0	OFF	VDD	ON	3.3V	Use when V_{DD} is greater than 3.8V (V_{DD}>3.8V)
1	1	ON	2xVDD	ON	3.3V	Use when V_{DD} is less than 3.8V (V_{DD}=2.2V~3.8V)

If V_{DD} is greater than 3.8V, the charge pump interrupt flag will be set and the device should turn off the charge pump automatically using the application program.

The Charge Pump clock source comes from the system clock. The CHPCKD[4:0] bits are used to set the clock divider ratio to generate the desired clock frequency to for the charge pump module. The actual frequency is decided by the following formula:

$$\text{Charge Pump Clock Frequency} = (f_{SYS}/16) / (\text{CHPCKD}+1)$$

It is recommended that the charge pump clock frequency kept close to 20kHz. The application program needs to set the correct value to get the desired clock frequency. For example, for a 4MHZ application, the CHPCKD bits should be set to 11, and for a 2MHz application, the correct value of CHPCKD is 5.

The charge pump module contains a voltage detect circuit to detect whether the input voltage is greater than 3.8V. The two related control bits for charge pump input voltage detector is contained in the CTRL2 register. The CPDEN bit is used to enable or disable the input voltage detect function. The other CPF bit, is a read only bit. CPF is a flag bit to record the current charge pump input voltage status. If the input voltage is greater than 3.6V, the CPF will be set high, otherwise, it will be cleared to zero. When the input voltage is over 3.6V, it is also necessary to take note that a charge pump interrupt will be generated. Note: When entering the HALT mode, the hardware will turn off the Charge pump automatically to prevent the charge pump output being in an unknown state. During this time the LCD cannot be used. Note: About 15ms after enabling the regulator, a stable output will be generated. Only after this time can the related analog circuits (ADC, OPA, WDT, PGA, SCF) be setup and used. If the PGA has been enabled, then wait for about 400ms before setting and operating the ADC.

CTRL2 Register

Bit	7	6	5	4	3	2	1	0
Name	DACEN	CPDEN	INT1ES1	INT1ES0	INT0ES1	INT0ES0	CPF	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	—
POR	0	0	1	0	1	0	0	—

Bit 7 Described elsewhere

Bit 6 **CPDEN**: Charge pump input voltage detector circuit disable/enable control
0: disable
1: enable
If CPDEN=0, the Charge pump voltage input detector circuit is disabled, CPF will remain at 0, and the detector circuit consumes no power.

Bit 5~2 Described elsewhere

Bit 1 **CPF**: Charge pump input voltage detector flag
0: Charge pump input voltage is above 3.8V
1: Charge pump input voltage is equal to or under 3.8V
When CPDEN and REGCEN are enabled, the CPF will be working.

Bit 0 Unimplemented, read as "0"

LCD Driver

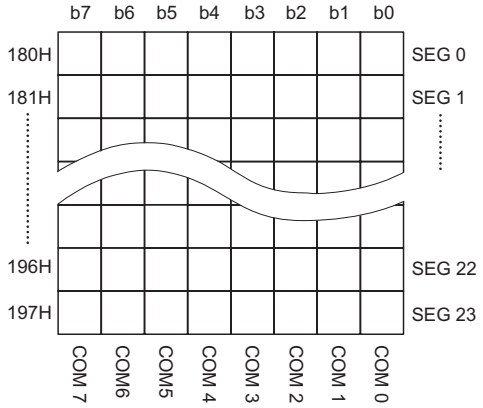
For large volume applications, which incorporate an LCD in their design, the use of a custom display rather than a more expensive character based display reduces costs significantly. However, the corresponding COM and SEG signals required, which vary in both amplitude and time, to drive such a custom display require many special considerations for proper LCD operation to occur. This device contains 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.

Duty	Driver No.	Bias	Bias Type	Wave Type
1/4	24×4	1/3 or 1/4	R	A or B
1/6	22×6			
1/8	20×8			

LCD Display Memory

The device provides an area of embedded data memory for LCD display. This area is located from 80H to 97H of the RAM at Bank 1. Bank pointer (BP) is the switch between the RAM and the LCD display memory. When BP.2~BP.0 is set to "001", data written into 80H~97H will affect the LCD display. If the BP.2~BP.0 is set to a value other than "001", any data written into 80H~FFH is meant to access the general purpose data memory.

The LCD display memory can be read and written to only by indirect addressing mode using MP1. When data is written into the display data area, it is automatically read by the LCD driver which then generates the corresponding LCD driving signals. To turn the display on or off, a "1" or a "0" is written to the corresponding bit of the display memory, respectively. The figure illustrates the mapping between the display memory and LCD pattern for the device.



LCD Memory Map

LCD Control Register

Control Registers in the Data Memory, are used to control the various setup features of the LCD Driver. Various bits in these registers control functions such as duty type, bias type, bias resistor selection as well as overall LCD enable and disable. The LCDEN bit in the LCDCTRL register, which provides the overall LCD enable/disable function, will only be effective when the device is in the Normal, Slow or Idle Mode. If the device is in the Sleep Mode then the display will always be disabled. Bits RSEL0 and RSEL1 in the LCDCTRL register select the internal bias resistors to supply the LCD panel with the correct bias voltages. 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. The SEGCR register is used to determine if the output function of display pins SEG0~SEG7 are used as segment drivers or CMOS outputs. If used as CMOS outputs then the Display Memory is used to determine the logic level of the CMOS output pins.

LCDCTRL Register

Bit	7	6	5	4	3	2	1	0
Name	TYPE	DTYC1	DTYC0	—	BIAS0	RSEL1	RSEL0	LCDEN
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 **TYPE**: LCD Type A or B
0: Type A
1: Type B

Bit 6~5 **DTYC1, DTYC0**: Define LCD Duty
0: 1/4 Duty (24×4)
1: 1/8 Duty (20×8)
2: 1/6 Duty (22×6)
3: 1/8 Duty (20×8)

Bit 4 Unimplemented, read as "0"

Bit 3 **BIAS0**: Define LCD Bias
0: 1/3 Bias
1: 1/4 Bias

Bit 2~1 **RSEL1, RSEL0**: Bias resistor select
1/4 bias
00: 192kΩ– 48kΩ×4
01: 96kΩ – 24kΩ×4
10: 48kΩ – 12kΩ×4
11: 24kΩ – 6kΩ×4
1/3 bias
00: 144kΩ – 48kΩ×3
01: 72kΩ – 24kΩ×3
10: 36kΩ – 12kΩ×3
11: 18kΩ – 6kΩ×3

Bit 0 **LCDEN**: LCD enable/disable control
0: disable
1: enable

SEGCR Register

Bit	7	6	5	4	3	2	1	0
Name	SEG7C	SEG6C	SEG5C	SEG4C	SEG3C	SEG2C	SEG1C	SEG0C
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 **SEG7C**: Select SEG7 or PD7
0: SEG7
1: PD7

Bit 6 **SEG6C**: Select SEG6 or PD6
0: SEG6
1: PD6

Bit 5 **SEG5C**: Select SEG5 or PD5
0: SEG5
1: PD5

Bit 4 **SEG4C**: Select SEG4 or PD4
0: SEG4
1: PD4

Bit 3 **SEG3C**: Select SEG3 or PD3
0: SEG3
1: PD3

Bit 2 **SEG2C**: Select SEG2 or PD2
0: SEG2
1: PD2

Bit 1 **SEG1C**: Select SEG1 or PD1
0: SEG1
1: PD1

Bit 0 **SEG0C**: Select SEG0 or PD0
0: SEG0
1: PD0

VLCDC Register

Bit	7	6	5	4	3	2	1	0
Name	VLCDEN	—	—	—	VLCD3	VLCD2	VLCD1	VLCD0
R/W	R/W	—	—	—	R/W	R/W	R/W	R/W
POR	0	—	—	—	0	0	0	0

Bit 7 **VLCDEN**: LCD power enable or disable control
0: disable – VSS connected to biasing circuits – LCD off
1: enable – VLCD connected to LCD biasing circuits
When the VLCDEN bit has been set to "1", the regulator must be enabled by the program simultaneously, so that the LCD can obtain its power supply.

Bit 6~4 Unimplemented, read as "0"

Bit 3~0 **VLCD3~VLCD0**: LCD voltage select
0000: 3.0V
0001: 3.1V
0010: 3.2V
0011: 3.3V
0100: 3.4V
0101: 3.5V
0110: 3.6V
0111: 3.7V
1000: 3.8V
1001: 3.9V
1010: 4.0V
1011: 4.1V
1100: 4.2V
1101: 4.3V
1110: 4.4V
1111: 4.5V

Clock Source

The LCD clock source is the internal clock signal, f_{SUB}, divided by 8, using an internal divider circuit. The f_{SUB} internal clock is supplied by either the LIRC or LXT oscillator, the choice of which is determined by a configuration option. For proper LCD operation, this arrangement is provided to generate an ideal LCD clock source frequency of 4kHz.

f_{SUB} Clock Source	LCD Clock Frequency
LIRC	4kHz
LXT	4kHz

LCD Clock Source

LCD Driver Output

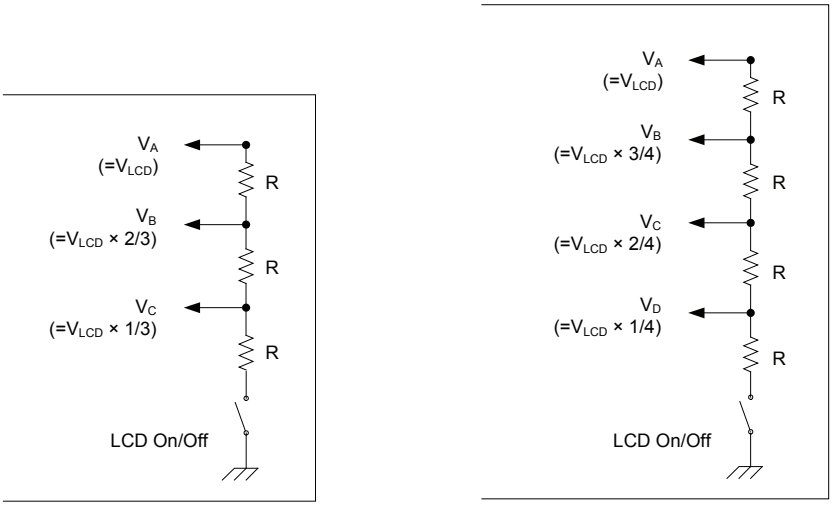
When the SEG7C~SEG0C bits in the SEGCR register are cleared to zero, the SEG7/PD7~SEG0/PD0 lines will be setup as LCD driver pins to drive the LCD display, otherwise, they will have general I/O functions. The biasing and duty of LCD are dependent upon how the related LCD control bits are programmed. 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 two control bits DTYC1 and DTYC0 to have a value of 1/4, 1/6 and 1/8, and which equates to a COM number of 4, 6 and 8, 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 LCDCTRL 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 only has R type biasing. For R type biasing an internal LCD voltage source, V_{LCD} is supplied to generate the internal biasing voltages. The V_{LCD} voltage can be selected to have a range between 3V and 4.5V, in steps of 0.1V, using the VLCD0~VLCD3 bits in the VLCDC register. For the 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}, V_B is equal to V_{LCD} x 2/3 while V_C is equal to V_{LCD} x 1/3. For the 1/4 bias selection, four voltage levels V_{SS}, V_A, V_B, V_C and V_D are utilised. The voltage V_A is equal to V_{LCD}, V_B is equal to V_{LCD} x 3/4, V_C is equal to V_{LCD} x 1/2, while V_D is equal to V_{LCD} x 1/4. In addition to selecting 1/3 or 1/4 bias, several values of bias resistor can be chosen using bits in the LCDCTRL register. Different values of internal bias resistors can be selected using the RSEL0 and RESEL1 bits in the LCDCTRL register.

This along with the value of V_{LCD} will determine the bias current.



R type 1/3 Bias **R type 1/4 Bias**

Bias Voltage Generation

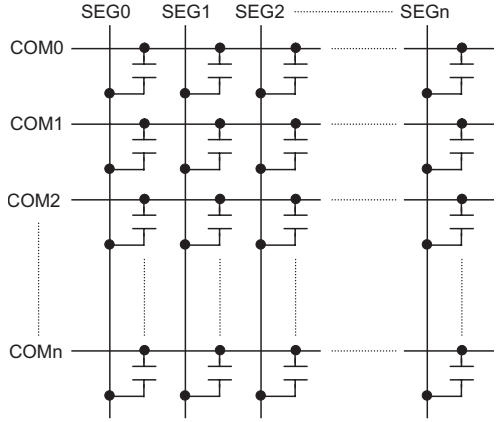
Programming Considerations

Certain precautions must be taken when programming the LCD. One of these is to ensure that the Display Memory is properly initialised after the microcontroller is powered on. Like the General Purpose Data Memory, the contents of the Display Memory are in an unknown condition after power-on. As the contents of the Display Memory will be mapped into the actual display, it is important to initialise this memory area into a known condition soon after applying power to obtain a proper display pattern.

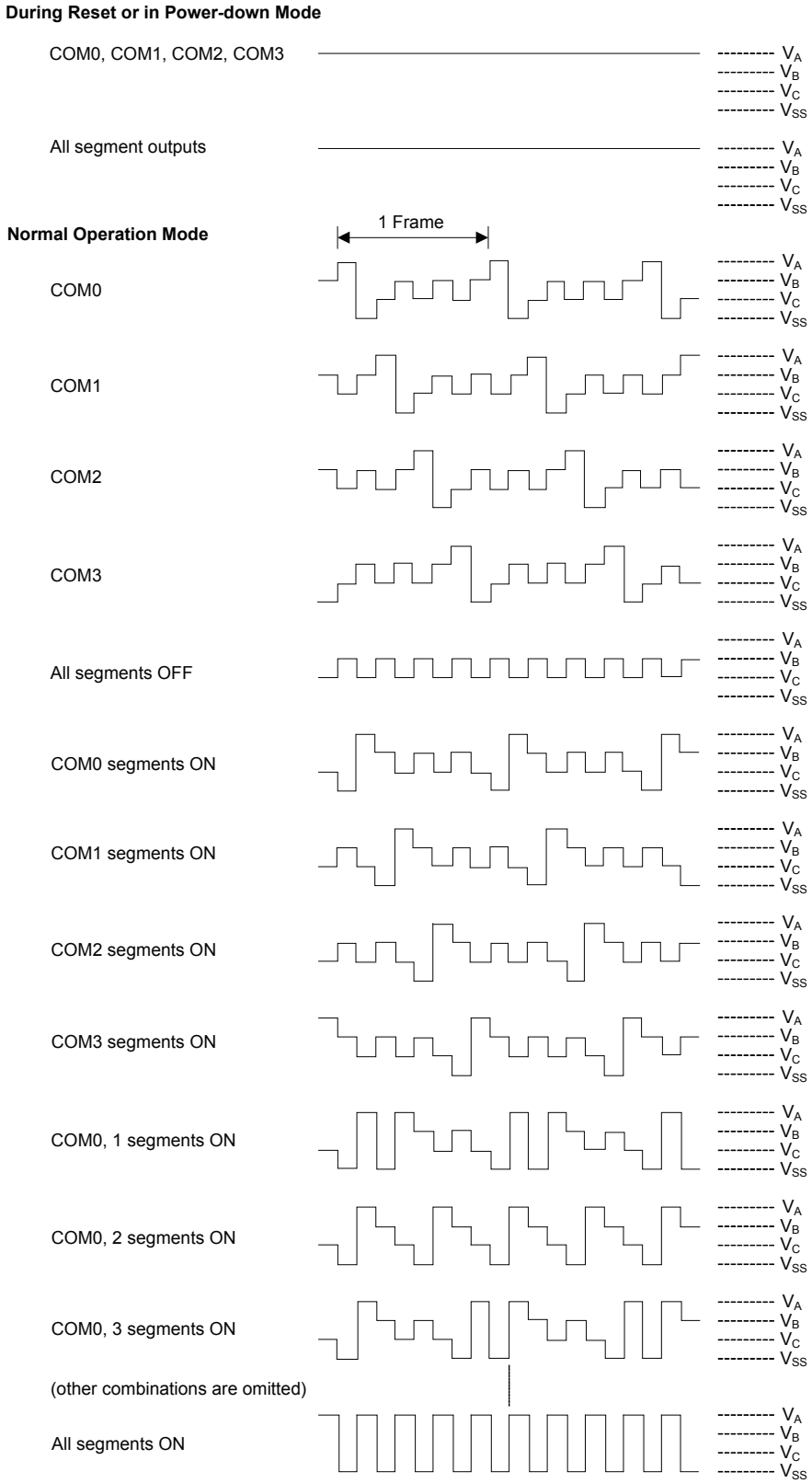
Consideration must also be given to the capacitive load of the actual LCD used in the application. As the load presented to the microcontroller by LCD pixels can be generally modeled as mainly capacitive in nature, it is important that this is not excessive, a point that is particularly true in the case of the COM lines which may be connected to many LCD pixels. The accompanying diagram depicts the equivalent circuit of the LCD.

One additional consideration that must be taken into account is what happens when the microcontroller enters a Power Down condition. The LCDEN control bit in the LEDCTRL 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. 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 does not permit all types to be displayed here.



LCD Panel Equivalent Circuit



LCD Driver Output – Type A -1/4 Duty, 1/3 Bias

VDDSPI Pin Function

As the SPI interfaces connect to external devices, the VDDSPI pin can be connected to the power supply of external devices to ensure the correct I/O voltage level interfacing. As the VDDSPI pin is pin-shared with other functions, it must be enabled as a VDDSPI pin using a bit in the VDDSPICR register. The SPI interfaces can also use the device VDD or VOREG as their power supply, the choice of which is implemented using bits in the VDDSPICR register.

VDDSPICR Register

Bit	7	6	5	4	3	2	1	0
Name	VDDSPIC	—	VDDSCK1	VDDSCK0	VDDSPI11	VDDSPI10	VDDSPI01	VDDSPI00
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 **VDDSPIC**: Select VDDSPI or PA7
0: I/O
1: VDDSPI

Bit 6 Unimplemented, read as "0"

Bit 5~4 **VDDSCK1, VDDSCK0**: INT1 and SCKO power supply select
00: VDD
01: VOREG
10: VDD
11: VDDSPI pin
Users must set the VDDSPIC bit to"1" when choosing the VDDSPI pin.

Bit 3~2 **VDDSPI11, VDDSPI10**: SPI1 power supply select
00: VDD
01: VOREG
10: VDD
11: VDDSPI
Users must set the VDDSPIC bit to "1" when choosing the VDDSPI pin.

Bit 1~0 **VDDSPI01, VDDSPI00**: SPI0 power supply select
00: VDD
01: VOREG
10: VDD
11: VDDSPI
Users must set the VDDSPIC bit to "1" when choosing the VDDSPI pin.

Configuration Option

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
Oscillator Options	
1	High Speed System Oscillator Selection − f_H: 1. HXT 2. ERC 3. HIRC
2	Low Speed System Oscillator Selection − f_{SUB}: 1. LXT 2. LIRC
3	WDT Clock Selection − f_S: 1. f_{SUB} 2. f_{SYS}/4
Note: The f_{SUB} and the f_{TBC} clock source are LXT or LIRC selection by the f_L configuration option.	
Watchdog Options	
4	CLR WDT Instructions Selection: 1. 1 instructions 2. 2 instructions
SPI Options	
5	WCOLn bit: 1. Enable 2. Disable
6	CSENn bit: 1. Enable 2. Disable

Instruction Set

Introduction

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

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

Instruction Timing

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

Moving and Transferring Data

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

Arithmetic Operations

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

Logical and Rotate Operation

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

Branches and Control Transfer

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

Bit Operations

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

Table Read Operations

Data storage is normally implemented by using registers. However, when working with large amounts of fixed data, the volume involved often makes it inconvenient to store the fixed data in the Data Memory. To overcome this problem, Holtek microcontrollers allow an area of Program Memory to be 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 following table depicts a summary of the instruction set categorised according to function and can be consulted as a basic instruction reference using the following listed conventions.

Table Conventions

x: Bits immediate data
m: Data Memory address
A: Accumulator
i: 0~7 number of bits
addr: Program memory address

Mnemonic	Description	Cycles	Flag Affected
Arithmetic			
ADD A,[m]	Add Data Memory to ACC	1	Z, C, AC, OV
ADDM A,[m]	Add ACC to Data Memory	1Note	Z, C, AC, OV
ADD A,x	Add immediate data to ACC	1	Z, C, AC, OV
ADC A,[m]	Add Data Memory to ACC with Carry	1	Z, C, AC, OV
ADCM A,[m]	Add ACC to Data memory with Carry	1Note	Z, C, AC, OV
SUB A,x	Subtract immediate data from the ACC	1	Z, C, AC, OV
SUB A,[m]	Subtract Data Memory from ACC	1	Z, C, AC, OV
SUBM A,[m]	Subtract Data Memory from ACC with result in Data Memory	1Note	Z, C, AC, OV
SBC A,[m]	Subtract Data Memory from ACC with Carry	1	Z, C, AC, OV
SBCM A,[m]	Subtract Data Memory from ACC with Carry, result in Data Memory	1Note	Z, C, AC, OV
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].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
TABRDC [m]	Read table (current page) to TBLH and Data Memory	2Note	None
TABRDL [m]	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
CLR WDT1	Pre-clear Watchdog Timer	1	TO, PDF
CLR WDT2	Pre-clear Watchdog Timer	1	TO, PDF
SWAP [m]	Swap nibbles of Data Memory	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.

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

Instruction Definition

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

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

Operation ACC ← ACC + [m] + C

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

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

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

Operation [m] ← ACC + [m] + C

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

ADD A,[m] Add Data Memory to ACC

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

Operation ACC ← ACC + [m]

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

ADD A,x Add immediate data to ACC

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

Operation ACC ← ACC + x

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

ADDM A,[m] Add ACC to Data Memory

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

Operation [m] ← ACC + [m]

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

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

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

Operation ACC ← ACC "AND" [m]

Affected flag(s) Z

AND A,x Logical AND immediate data to ACC

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

Operation ACC ← ACC "AND" x

Affected flag(s) Z

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

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

Operation [m] ← ACC "AND" [m]

Affected flag(s) Z

CALL addr Subroutine call

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

Operation Stack ← Program Counter + 1
Program Counter ← addr

Affected flag(s) None

CLR [m] Clear Data Memory

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

Operation [m] ← 00H

Affected flag(s) None

CLR [m].i Clear bit of Data Memory

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

Operation [m].i ← 0

Affected flag(s) None

CLR WDT Clear Watchdog Timer

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

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CLR WDT1 Pre-clear Watchdog Timer

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

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CLR WDT2 Pre-clear Watchdog Timer

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

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CPL [m] Complement Data Memory

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

Operation [m] ← $\overline{[m]}$

Affected flag(s) Z

CPLA [m] Complement Data Memory with result in ACC

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

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

Affected flag(s) Z

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

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

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

Affected flag(s) C

DEC [m] Decrement Data Memory

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

Operation [m] ← [m] − 1

Affected flag(s) Z

DECA [m] Decrement Data Memory with result in ACC

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

Operation ACC ← [m] − 1

Affected flag(s) Z

HALT Enter power down mode

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

Operation TO ← 0
PDF ← 1

Affected flag(s) TO, PDF

INC [m] Increment Data Memory

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

Operation [m] ← [m] + 1

Affected flag(s) Z

INCA [m] Increment Data Memory with result in ACC

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

Operation ACC ← [m] + 1

Affected flag(s) Z

JMP addr Jump unconditionally

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

Operation Program Counter ← addr

Affected flag(s) None

MOV A,[m] Move Data Memory to ACC

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

Operation ACC ← [m]

Affected flag(s) None

MOV A,x Move immediate data to ACC

Description The immediate data specified is loaded into the Accumulator.

Operation ACC ← x

Affected flag(s) None

MOV [m],A Move ACC to Data Memory

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

Operation [m] ← ACC

Affected flag(s) None

NOP No operation

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

Operation No operation

Affected flag(s) None

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

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

Operation ACC ← ACC ″OR″ [m]

Affected flag(s) Z

OR A,x Logical OR immediate data to ACC

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

Operation ACC ← ACC ″OR″ x

Affected flag(s) Z

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

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

Operation [m] ← ACC ″OR″ [m]

Affected flag(s) Z

RET Return from subroutine

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

Operation Program Counter ← Stack

Affected flag(s) None

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

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

Operation Program Counter ← Stack
ACC ← x

Affected flag(s) None

RETI Return from interrupt

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

Operation Program Counter ← Stack
EMI ← 1

Affected flag(s) None

RL [m] Rotate Data Memory left

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

RLC [m] Rotate Data Memory left through Carry

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

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

Affected flag(s) C

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

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

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

Affected flag(s) C

RR [m] Rotate Data Memory right

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

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

Affected flag(s) None

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

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

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

Affected flag(s): None

RRC [m] Rotate Data Memory right through Carry

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

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

Affected flag(s): C

RRCA [m] Rotate Data Memory right through Carry with result in ACC

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

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

Affected flag(s): C

SBC A,[m] Subtract Data Memory from ACC with Carry

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

Operation: ACC ← ACC − [m] − C

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

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

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

Operation: [m] ← ACC − [m] − C

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

SDZ [m] Skip if decrement Data Memory is 0

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

Operation:
[m] ← [m] − 1
Skip if [m]=0

Affected flag(s): None

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

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

Operation ACC ← [m] − 1
Skip if ACC=0

Affected flag(s) None

SET [m] Set Data Memory

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

Operation [m] ← FFH

Affected flag(s) None

SET [m].i Set bit of Data Memory

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

Operation [m].i ← 1

Affected flag(s) None

SIZ [m] Skip if increment Data Memory is 0

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

Operation [m] ← [m] + 1
Skip if [m]=0

Affected flag(s) None

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

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

Operation ACC ← [m] + 1
Skip if ACC=0

Affected flag(s) None

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

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

Operation Skip if [m].i ≠ 0

Affected flag(s) None

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

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

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

Operation [m] ← ACC − [m]

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

SUB A,x Subtract immediate data from ACC

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

Operation ACC ← ACC − x

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

SWAP [m] Swap nibbles of Data Memory

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

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s) None

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

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

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

Affected flag(s) None

SZ [m] Skip if Data Memory is 0

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

Operation Skip if [m]=0

Affected flag(s) None

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

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

Operation ACC ← [m]
Skip if [m]=0

Affected flag(s) None

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

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

Operation Skip if [m].i=0

Affected flag(s) None

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

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

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

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

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

Affected flag(s) None

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

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

Operation ACC ← ACC ″XOR″ [m]

Affected flag(s) Z

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

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

Operation [m] ← ACC ″XOR″ [m]

Affected flag(s) Z

XOR A,x Logical XOR immediate data to ACC

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

Operation ACC ← ACC ″XOR″ x

Affected flag(s) Z

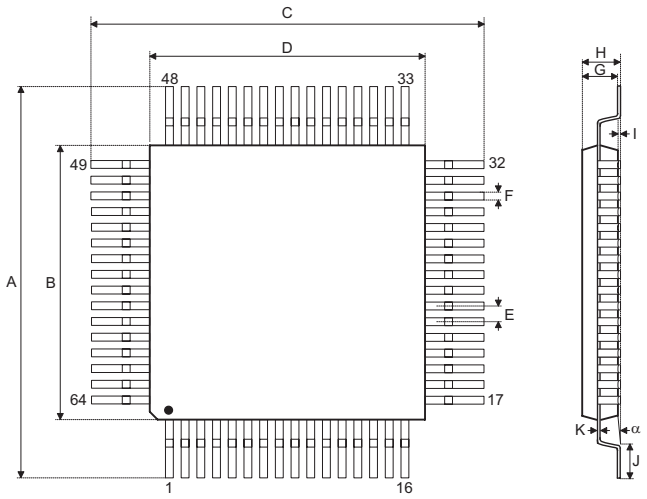
Package Information

Note that the package information provided here is for consultation purposes only. As this information may be updated at regular intervals users are reminded to consult the Holtek website for the latest version of the Package/Carton Information.

Additional supplementary information with regard to packaging is listed below. Click on the relevant section to be transferred to the relevant website page.

- Package Information (include Outline Dimensions, Product Tape and Reel Specifications)
- The Operation Instruction of Packing Materials
- Carton information

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	—	9.0 BSC	—
B	—	7.0 BSC	—
C	—	9.0 BSC	—
D	—	7.0 BSC	—
E	—	0.4 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°