Dual Slope A/D Type 8-Bit OTP MCU with LCD

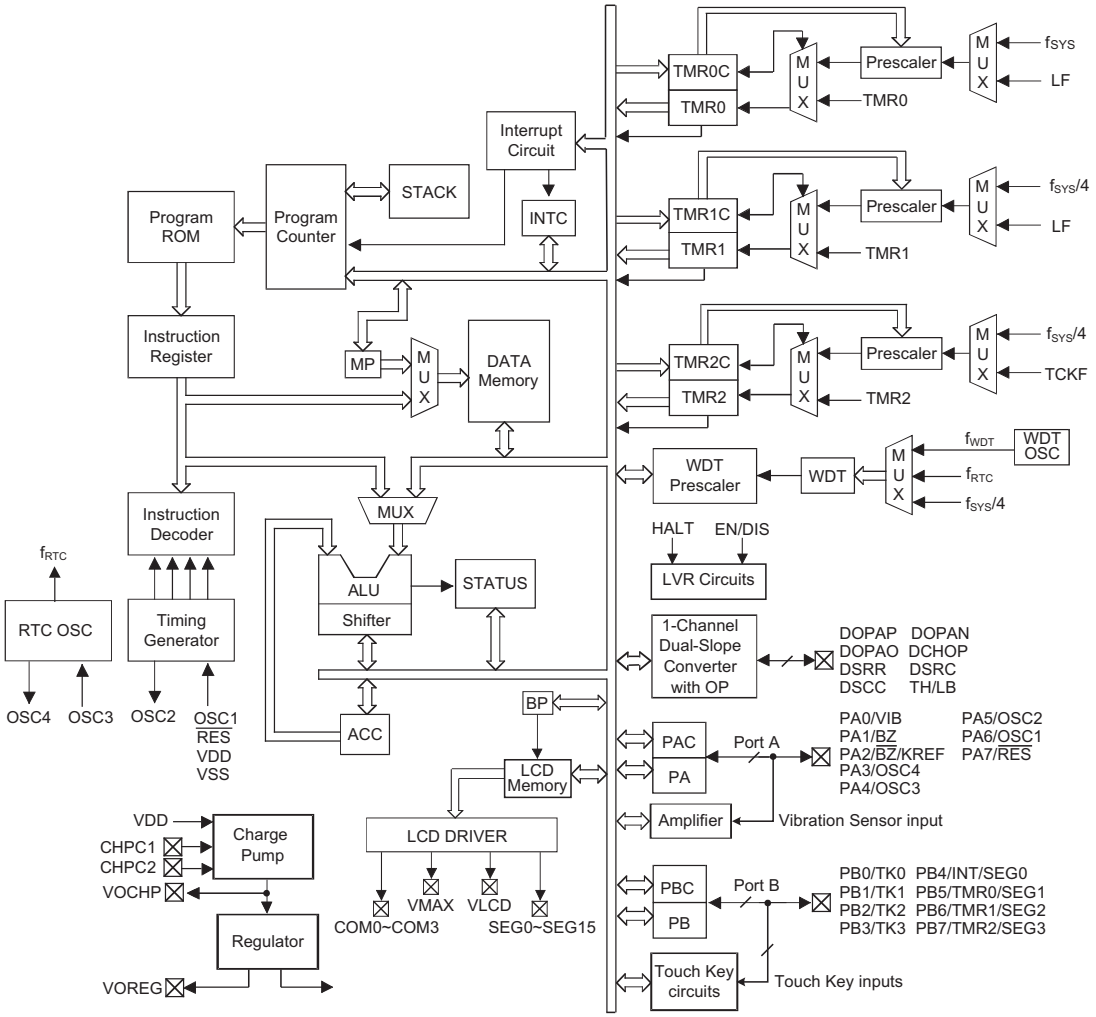
Features

- Operating voltage:
  f_{SYS}=4MHz: 2.2V~5.5V
  f_{SYS}=8MHz: 3.3V~5.5V
- Three system oscillators:
  External Crystal oscillator
  External RC oscillator
  Internal RC oscillator
- Up to 16 bidirectional I/O lines
- One external interrupt input shard with an I/O lines
- One 8-bit and two 16-bit programmable timer/event counter with overflow interrupt a 8-stage pre-scalar
- LCD driver with 16×4 segments
- 4K×15 program memory
- 128×8 data memory RAM
- Single differential input channel dual slope Analog to Digital Converter with Operational Amplifier.
- Watchdog Timer with regulator power
- Buzzer output
- Internal 12kHz RC oscillator
- External 32.768kHz Crystal oscillator
- HALT function and wake-up feature reduce power consumption
- Voltage regulator (3.3V) and charge pump
- Embeded voltage reference generator (1.5V)
- 4-level subroutine nesting
- Bit manipulation instruction
- 15-bit table read instruction
- Up to 0.5μs instruction cycle with 8MHz system clock at V_{DD}=5V
- 63 powerful instructions
- All instructions in 1 or 2 machine cycles
- Low voltage reset function
- One vibration sensor input
- Four touch-key inputs
- 52-pin QFP package

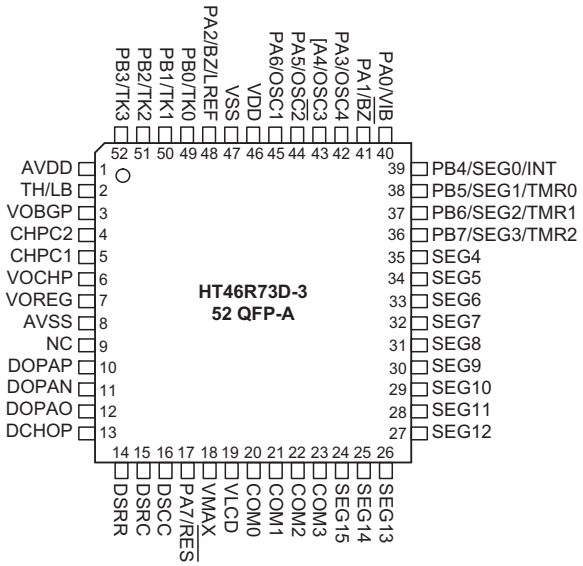
General Description

The HT46R73D-3 is an 8-bit high performance, RISC architecture microcontroller device specifically designed for A/D with LCD applications that interface directly to analog signals, such as those from sensors. The advantages of low power consumption, I/O flexibility, timer functions, oscillator options, Dual slope A/D converter, LCD display, HALT and wake-up functions, watchdog timer, as well as low cost, enhance the versatility of these devices to suit for a wide range of AD with LCD application possibilities such as sensor signal processing, scales, consumer products, subsystem controllers, etc.

Block Diagram



Pin Assignment



Pin Description

Pin Name	I/O	Options	Description
PA0/VIB PA1/BZ PA2/BZ/KREF PA3/OSC4 PA4/OSC3 PA5/OSC2 PA6/OSC1 PA7/RES	I/O	Pull-high Wake-up Buzzer 32.768kHz Crystal System oscillator RES	Bidirectional 8-bit input/output port. Each individual bit on this port can be configured to have a wake-up function using a configuration option. Software instructions determine if the pin is a CMOS output or Schmitt Trigger input. Configuration options determine which pins on this port have pull-high resistors except for PA7. VIB is the vibration sensor analog input which is pin-shared with PA0. BZ and BZ are buzzer outputs pin-shared with PA1 and PA2 and are to be used as buzzer outputs or normal I/O functions determined by configuration options. KREF is the reference oscillator input for the touch key function. OSC1 and OSC2 can be used as system oscillator pins which are pin-shared with PA6 and PA5. Configuration options determine if these pins are used as I/O pins or system oscillator pins. OSC3 and OSC4 can be configured to be used as the 32.768kHz oscillator pins or as the normal I/O pins named PA4 and PA3 using a configuration option. RES is pin-shared with PA7 determined by a configuration option. When PA7 is configured as an I/O pin, software instructions determine if this pin is open drain output or Schmitt Trigger input without pull-high resistor. For PA2/BZ/KREF pin, KREF has a higher priority than BZ if both of them are enabled at same time.
PB0/TK0 PB1/TK1 PB2/TK2 PB3/TK3 PB4/INT/SEG0 PB5/TMR0/SEG1 PB6/TMR1/SEG2 PB7/TMR2/SEG3	I/O	Pull-high	Bidirectional 8-bit input/output port. Software instructions determine if the pin is a CMOS output or Schmitt Trigger input. Configuration options determine which pins on this port have pull-high resistors. TK0~TK3 are touch sensor input pins which are pin-shared with PB0~PB3. PB4~PB7 are pin-shared with INT, TMR0, TMR1 and TMR2 and also with the LCD segments SEG0~SEG3 respectively which are selected by Software instructions. Once these pins are selected as segments, the I/O function including Schmitt trigger input and pull-high are all disabled. However, these pins will default to an input mode with pull-high resistors after a reset.
SEG4~SEG15	O	—	LCD segment outputs
COM0~COM3	O	—	LCD common outputs
VMAX	—	—	IC maximum voltage, connect to VDD or VLCD.
VLCD	I	—	LCD power supply
VOBGP	AO		Band gap voltage output pin. (for internal use)
VOREG	O	—	Charge pump capacitor (Negative)
VOCHP	O	—	Regulator output 3.3V
CHPC1	—	—	Charge pump output - a capacitor is required to be connected
CHPC2	—	—	Charge pump capacitor (Positive)
DOPAN, DOPAP, DOPAO, DCHOP	AI/AO	—	Dual Slope A/D converter pre-stage OPA related pins. DOPAN is the OPA Negative input pin, DOPAP is the OPA Positive input pin, DOPAO is the OPA output pin and DCHOP is the OPA Chopper pins.
TH/LB			Temperature sensor/Low battery voltage input pin.

Pin Name	I/O	Options	Description
DSRR, DSRC, DSCC	AI/AO	—	Dual slope A/D converter main function RC circuit. DSRR is the input or reference signal, DSRC is the Integrator negative input, and DSCC is the comparator negative input.
VDD	—	—	Digital positive power supply
VSS	—	—	Digital Negative Power supply, ground
AVDD	—	—	Analog positive power supply
AVSS	—	—	Analog negative power supply, ground

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
I_{OL} Total 150mA
Total Power Dissipation 500mW

Storage Temperature ..... –50°C to 125°C
Operating Temperature ..... –20°C to 85°C
I_{OH} Total ..... –100mA

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

D.C. Characteristics

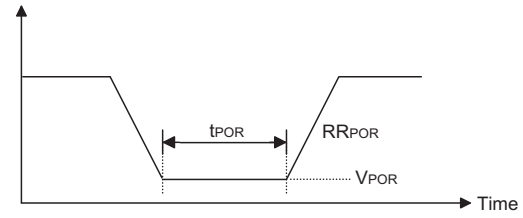
Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	f_{SYS}=4MHz	2.2	—	5.5	V
		—	f_{SYS}=8MHz	3.3	—	5.5	V
I_{DD1}	Operating Current (Crystal OSC, Ext. RC OSC, Int. RC OSC)	5V	No load, f_{SYS}=8MHz, analog block off	—	4	8	mA
I_{DD2}	Operating Current (Crystal OSC, Ext. RC OSC, Int. RC OSC)	3V	No load, f_{SYS}=4MHz, ADC block off	—	0.8	1.5	mA
		5V		—	2.5	4	mA
I_{DD3}	Operating Current (Crystal OSC, Ext. RC OSC)	3V	No load, f_{SYS}=2MHz, ADC block off	—	0.5	1	mA
		5V		—	1.5	3	mA
I_{DD4}	Operating Current (Crystal OSC, Ext. RC OSC)	5V	V_{REGO}=3.3V, f_{SYS}=4MHz, ADC on, ADCCCLK= 125kHz (all other analog devices off)	—	3	5	mA
I_{STB1}	Standby Current (WDT Disable)	3V	No load, system HALT, LCD off at HALT	—	—	1	μA
		5V		—	—	2	μA
I_{STB2}	Standby Current (WDT Enable)	3V	No load, system HALT, LCD off at HALT, ADC off	—	2.5	5	μA
		5V		—	8	15	μA
I_{STB3}	Standby Current (WDT Disable Internal RC 12kHz OSC ON)	3V	No load, system HALT, LCD off at HALT, ADC off	—	2	5	μA
		5V		—	6	10	μA

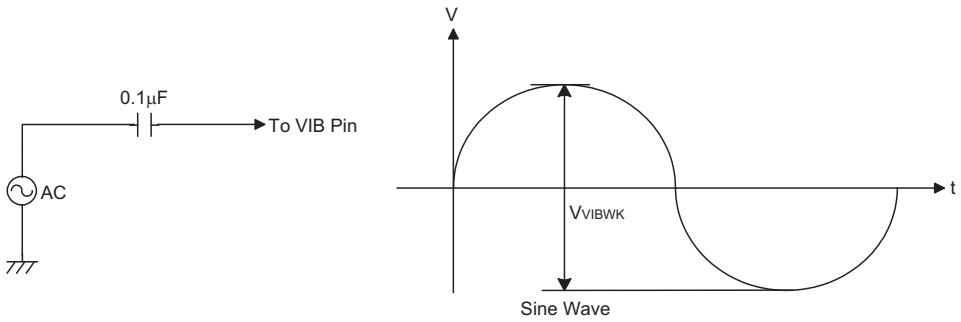
Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{STB4}	Standby Current (WDT Disable)	3V	No load, system osc HALT, internal RC 12kHz OSC On, ADC block Off, LCD ON (1/3 bias) at HALT, $V_{LCD}=V_{DD}$	—	43	55	μA
		5V		—	58	80	μA
I_{STB5}	Standby Current (Internal RC 12kHz OSC Off, RTC On)	3V	No load, system HALT RTC osc slowly start-up	—	—	5	μA
		5V		—	—	15	μA
I_{STB6}	Standby Current (WDT Disable)	3V	No load, system osc Off, RTC OSC On, ADC block Off, LCD On (1/3 bias), $V_{LCD}=V_{DD}$	—	30	60	μA
		5V		—	60	120	μA
I_{STB7}	Standby Current (WDT Disable)	3V	No load, Only vibration sensor turn on & VIB pin connected a 0.1μF cap to VSS	—	2	4	μA
		5V		—	8	16	μA
V_{IL1}	Input Low Voltage for I/O Ports, TMR0, TMR1, TMR2 and INT pins	—	—	0	—	$0.3V_{DD}$	V
V_{IH1}	Input High Voltage for I/O Ports, TMR0, TMR1, TMR2 and INT pins	—	—	$0.7V_{DD}$	—	V_{DD}	V
V_{IL2}	Input Low Voltage ($\overline{RES}$)	—	—	0	—	$0.4V_{DD}$	V
V_{IH2}	Input High Voltage ($\overline{RES}$)	—	—	$0.9V_{DD}$	—	V_{DD}	V
V_{LCD}	LCD Highest Voltage	—	—	0	—	V_{DD}	V
V_{LVR}	Low Voltage Reset	—	Configuration option: 2.1V	1.98	2.10	2.22	V
			Configuration option: 3.15V	2.98	3.15	3.32	V
			Configuration option: 4.2V	3.98	4.20	4.42	V
V_{LVD}	Low Voltage Detector	—	—	2.2	2.3	2.4	V
I_{OL1}	Sink Current for I/O ports except PA7	3V	$V_{OL}=0.1V_{DD}$	4	8	—	mA
		5V		10	20	—	mA
I_{OH1}	Source Current for I/O ports except PA7	3V	$V_{OH}=0.9V_{DD}$	–2	–4	—	mA
		5V		–5	–10	—	mA
I_{OL2}	LCD Common and Segment Current	3V	$V_{OL}=0.1V_{DD}$	210	420	—	μA
		5V		350	700	—	μA
I_{OH2}	LCD Common and Segment Current	3V	$V_{OH}=0.9V_{DD}$	–80	–160	—	μA
		5V		–180	–360	—	μA
I_{OL3}	Sink Current for PA7	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
V_{POR}	VDD Start Voltage to ensure Power-on Reset	—	—	—	—	100	mV
R_{POR}	VDD Rise Rate to ensure Power-on Reset	—	—	0.035	—	—	V/ms
t_{POR}	Power-on Reset Low Pulse Width	—	—	1	—	—	ms

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{VIBWK}	Minimum Voltage to Wake MCU by the Vibration Sensor Input	—	100Hz~1KHz sine wave (note)	250	—	—	mV
Charge Pump and Regulator							
V_{CHPI}	Input Voltage	—	Charge pump on	2.2	—	3.6	V
			Charge pump off	3.7	—	5.5	V
V_{REGO}	Output Voltage	—	No load	3	3.3	3.6	V
V_{REGDP1}	Regulator Output Voltage Drop (Compare with No Load)	—	V_{DD}=3.7V~5.5V Charge pump off Current≤10mA	—	100	—	mV
V_{REGDP2}		—	V_{DD}=2.4V~3.6V Charge pump on Current≤6mA	—	100	—	mV
Dual Slope AD, Amplifier and Band Gap							
V_{RFGTC}	Reference Generator Temperature Coefficient	—	@3.3V	—	50	—	Ppm/C
V_{ADOFF}	Input Offset Range	—	—	—	500	800	μV
V_{ICMR}	Common Mode Input Range	—	Amplifier, no load	0.2	—	V_{REGO}–1.2	V
		—	Integrator, no load	1.2	—	V_{REGO}–0.2	V

Note: 1. V_{DD}



2. Test Circuits for V_{VMBWK}



A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS}	System Clock (External RC OSC)	2.2V~5.5V	—	400	—	4000	kHz
	System Clock (Crystal OSC)	2.2V~5.5V	—	400	—	4000	kHz
		3.3V~5.5V	—	400	—	8000	kHz
		4.5V~5.5V	—	400	—	12000	kHz
f_{HIRC}	Internal RC OSC	3V/5V	Ta=25°C	-2%	4/8	+2%	MHz
		5V	Ta=25°C	-2%	12	+2%	MHz
		3V/5V	Ta=0~70°C	-5%	4/8	+5%	MHz
		5V	Ta=0~70°C	-5%	12	+5%	MHz
		2.2V~3.6V	Ta=0~70°C	-8%	4	+8%	MHz
		3.0V~5.5V	Ta=0~70°C	-8%	4/8	+8%	MHz
		4.5V~5.5V	Ta=0~70°C	-8%	12	+8%	MHz
		2.2V~3.6V	Ta=-40~85°C	-12%	4	+12%	MHz
		3.0V~5.5V	Ta=-40~85°C	-12%	4/8	+12%	MHz
		4.5V~5.5V	Ta=-40~85°C	-12%	12	+12%	MHz
f_{ERC}	External RC OSC	5V	Ta=25°C, R=120kΩ	-2%	4	-2%	MHz
		5V	Ta=0~70°C, R=120kΩ	-5%	4	-5%	MHz
		5V	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_{TIMER}	Timer I/P Frequency (TMR0/TMR1/TMR2)	2.2V~5.5V	—	0	—	4000	kHz
t_{WDTOSC}	Watchdog Oscillator Period	3V	—	45	90	180	μs
		5V	—	32	65	130	μs
t_{RES}	External Reset Low Pulse Width	—	—	1	—	—	μs
t_{SST}	System Start-up Timer Period (Wake-up from HALT)	—	f_{SYS}=Crystal Oscillator	—	1024	—	t_{SYS}
			f_{SYS}= f_{ERC} or f_{HIRC}	—	1024 *	—	t_{SYS}
t_{INT}	Interrupt Pulse Width	—	—	1	—	—	ms
t_{LVR}	Low Voltage Width to Reset	—	—	0.25	1.00	2.00	ms

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

"*" When the system clock comes from the external RC or internal RC oscillator, the system start-up time period can be 2 or 1024 clock cycles determined by a configuration option.

Functional Description

Execution Flow

The system clock is derived from a crystal, an external RC or internal RC oscillator. It is internally divided into four non-overlapping clocks. One instruction cycle consists of four system clock cycles.

Instruction fetching and execution are pipelined in such a way that a fetch takes one instruction cycle while decoding and execution takes the next instruction cycle. The pipelining scheme makes it possible for each instruction to be effectively executed in a cycle. If an instruction changes the value of the program counter, two cycles are required to complete the instruction.

Program Counter – PC

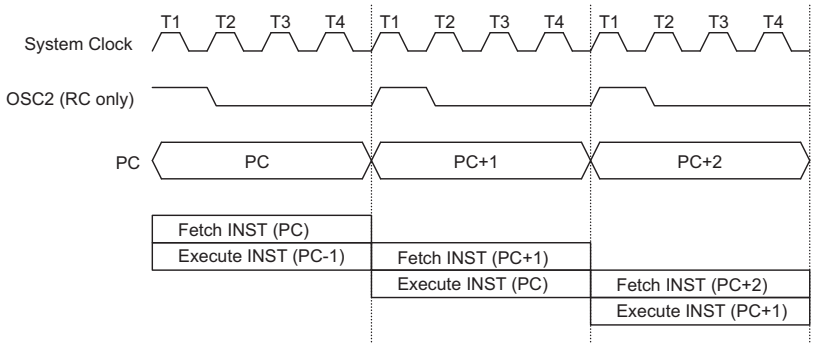
The program counter (PC) is 12 bits wide and it controls the sequence in which the instructions stored in the program ROM are executed. The contents of the PC can specify a maximum of 4096 addresses.

After accessing a program memory word to fetch an instruction code, the value of the PC is incremented by 1. The PC then points to the memory word containing the next instruction code.

When executing a jump instruction, conditional skip execution, loading a PCL register, a subroutine call, an initial reset, an internal interrupt, an external interrupt, or returning from a subroutine, the PC manipulates the program transfer by loading the address corresponding to each instruction.

The conditional skip is activated by instructions. Once the condition is met, the next instruction, fetched during the current instruction execution, is discarded and a dummy cycle replaces it to get a proper instruction; otherwise proceed to the next instruction.

The lower byte of the PC (PCL) is a readable and writeable register (06H). Moving data into the PCL performs a short jump. The destination is within 256 locations.



Execution Flow

Mode	Program Counter											
	*11	*10	*9	*8	*7	*6	*5	*4	*3	*2	*1	*0
Initial Reset	0	0	0	0	0	0	0	0	0	0	0	0
External Interrupt	0	0	0	0	0	0	0	0	0	1	0	0
Timer/Event Counter 0 Overflow	0	0	0	0	0	0	0	0	1	0	0	0
Timer/Event Counter 1 Overflow	0	0	0	0	0	0	0	0	1	1	0	0
Timer/Event Counter 2 Overflow	0	0	0	0	0	0	0	1	0	0	0	0
ADC Interrupt	0	0	0	0	0	0	0	1	0	1	0	0
Touch Key interrupt	0	0	0	0	0	0	0	1	1	0	0	0
Skip	Program Counter+2											
Loading PCL	*11	*10	*9	*8	@7	@6	@5	@4	@3	@2	@1	@0
Jump, Call Branch	#11	#10	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
Return From Subroutine	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

Program Counter

Note: *11~*0: Program counter bits
#11~#0: Instruction code bits
S11~S0: Stack register bits
@7~@0: PCL bits

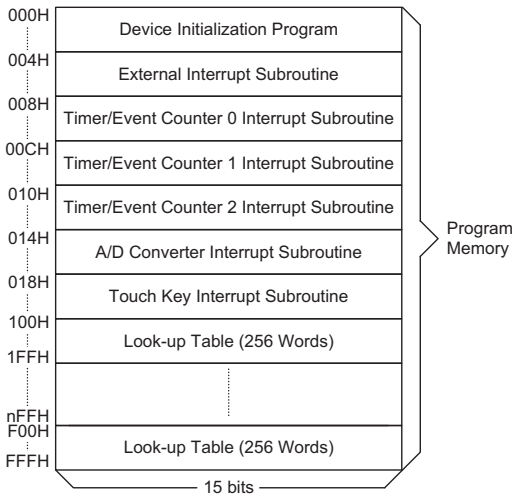
When a control transfer takes place, an additional dummy cycle is required.

Program Memory

The program memory is used to store the program instructions which are to be executed. It also contains data, table, and interrupt entries, and is organized into 4096×15 bits which are addressed by the program counter and table pointer.

Certain locations in the ROM are reserved for special usage:

- Location 000H
  Location 000H is reserved for program initialization. After chip reset, the program always begins execution at this location.
- Location 004H
  Location 004H is reserved for the external interrupt service program. If the INT input pin is activated, and the interrupt is enabled, and the stack is not full, the program begins execution at location 004H.
- Location 008H
  Location 008H is reserved for the Timer/Event Counter 0 interrupt service program. If a timer interrupt results from a Timer/Event Counter 0 overflow, and if the interrupt is enabled and the stack is not full, the program begins execution at location 008H.
- Location 00CH
  Location 00CH is reserved for the Timer/Event Counter 1 interrupt service program. If a timer interrupt results from a Timer/Event Counter 1 overflow, and if the interrupt is enabled and the stack is not full, the program begins execution at location 00CH.
- Location 010H
  Location 010H is reserved for the Timer/Event Counter 2 interrupt service program. If a timer interrupt results from a Timer/Event Counter 2 overflow, and if the interrupt is enabled and the stack is not full, the program begins execution at this location.
- Location 014H
  Location 014H is reserved for the ADC interrupt service program. If an ADC interrupt occurs, and if the interrupt is enabled and the stack is not full, the program begins execution at this location.
- Location 018H
  Location 018H is reserved for the touch key interrupt service program. If a touch key interrupt occurs, and if the interrupt is enabled and the stack is not full, the program begins execution at this location.
- Table location
  Any location in the ROM can be used as a look-up table. The instructions "TABRDC [m]" (the current page, 1 page=256 words) and "TABRDL [m]" (the last page) transfer the contents of the lower-order byte to the specified data memory, and the contents of the higher-order byte to TBLH (Table Higher-order byte register) (08H). Only the destination of the lower-order byte in the table is well-defined; the other bits of the table word are all transferred to the lower portion of TBLH. The TBLH is read only, and the table pointer (TBLP) is a read/write register (07H), indicating the table location. Before accessing the table, the location should be placed in TBLP. All the table related instructions require 2 cycles to complete the operation. These areas may function as a normal ROM depending upon the user's requirements.



Program Memory

Instruction(s)	Table Location											
	*11	*10	*9	*8	*7	*6	*5	*4	*3	*2	*1	*0
TABRDC [m]	P11	P10	P9	P8	@7	@6	@5	@4	@3	@2	@1	@0
TABRDL [m]	1	1	1	1	@7	@6	@5	@4	@3	@2	@1	@0

Table Location

Note: *11~*0: Table location bits
P11~P8: Current program counter bits
@7~@0: Table pointer bits

Stack Register – STACK

The stack register is a special part of the memory used to save the contents of the program counter. The stack is organized into 4 levels and is neither part of the data nor part of the program, and is neither readable nor writeable. Its activated level is indexed by a stack pointer (SP) and is neither readable nor writeable. At the start of a subroutine call or an interrupt acknowledgment, the contents of the program counter is pushed onto the stack. At the end of the subroutine or interrupt routine, signaled by a return instruction (RET or RETI), the contents of the program counter is restored to its previous value from the stack. After chip reset, the SP will point to the top of the stack.

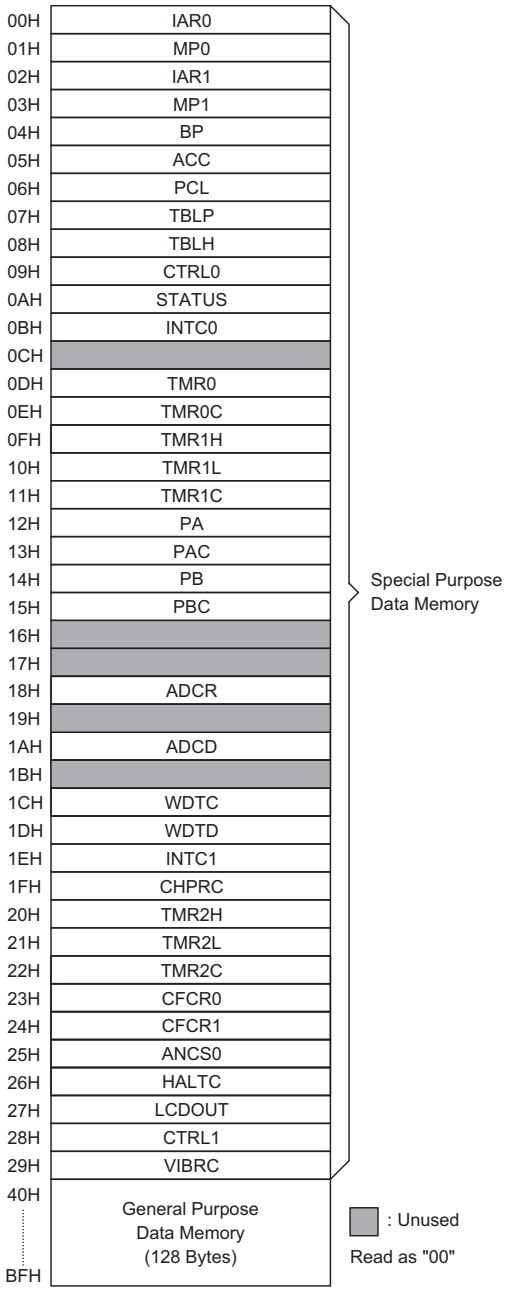
If the stack is full and a non-masked interrupt takes place, the interrupt request flag is recorded but the acknowledgment is still inhibited. Once the SP is decremented (by RET or RETI), the interrupt is serviced. This feature prevents stack overflow, allowing the programmer to use the structure easily. Likewise, if the stack is full, and a "CALL" is subsequently executed, a stack overflow occurs and the first entry is lost (only the most recent 4 return addresses are stored).

Data Memory – RAM

Bank 0 of the data memory has a capacity of 128×8 bits, and is divided into two functional groups, namely the special function registers of 37×8 bit capacity and the general purpose data memory of 96×8 bit capacity. Most locations are readable/writable, although some are read only. The special function register are overlapped in all banks.

Any unused space before 40H is reserved for future expanded usage, reading these locations will get "00H". The general purpose data memory, addressed from 40H to BFH , is used for data and control information under instruction commands. All of the data memory areas can handle arithmetic, logic, increment, decrement and rotate operations directly. Except for some dedicated bits, each bit in the data memory can be set and reset by the "SET [m].i" and "CLR [m].i" instructions. They are also indirectly accessible through the memory pointer registers, MP0 and MP1.

Bank 1 contains the LCD Data Memory locations. After first setting up BP to the value of "01H" to access Bank 1 this bank must then be accessed indirectly using the Memory Pointer MP1. With BP set to a value of "01H", using MP1 to indirectly read or write to the data memory areas with addresses from 40H~4FH will result in operations to Bank 1. Directly addressing the Data Memory will always result in Bank 0 being accessed irrespective of the value of BP.



RAM Mapping

Indirect Addressing Register

Location 00H and 02H are indirect addressing registers that are not physically implemented. Any read/write operation of [00H] and [02H] accesses the RAM pointed to by MP0 (01H) and MP1 (03H) respectively. Reading location 00H or 02H indirectly returns the result 00H. While, writing it indirectly leads to no operation. The memory pointer register, MP0 and MP1, are 8-bit registers.

The function of data movement between two indirect addressing registers is not supported. The memory pointer registers, MP0 and MP1, are both 8-bit registers used to access the RAM by combining corresponding indirect addressing registers. MP0 can only be applied to data memory, while MP1 can be applied to data memory and LCD display memory.

Accumulator – ACC

The accumulator (ACC) is related to the ALU operations. It is also mapped to location 05H of the RAM and is capable of operating with immediate data. The data movement between two data memory locations must pass through the ACC.

Arithmetic and Logic Unit – ALU

This circuit performs 8-bit arithmetic and logic operations and provides the following functions:

- Arithmetic operations (ADD, ADC, SUB, SBC, DAA)
- Logic operations (AND, OR, XOR, CPL)
- Rotation (RL, RR, RLC, RRC)
- Increment and Decrement (INC, DEC)
- Branch decision (SZ, SNZ, SIZ, SDZ etc.)

The ALU not only saves the results of a data operation but also changes the status register.

Status Register – STATUS

The status register (0AH) is 8 bits wide and contains, a carry flag (C), an auxiliary carry flag (AC), a zero flag (Z), an overflow flag (OV), a power down flag (PDF), and a watchdog time-out flag (TO). It also records the status information and controls the operation sequence.

Except for the TO and PDF flags, bits in the status register can be altered by instructions similar to other registers. Data written into the status register does not alter the TO or PDF flags. Operations related to the status register, however, may yield different results from those intended. The TO and PDF flags can only be changed by a Watchdog Timer overflow, chip power-up, or clearing the Watchdog Timer and executing the ″HALT″ instruction. The Z, OV, AC, and C flags reflect the status of the latest operations.

On entering the interrupt sequence or executing the subroutine call, the status register will not be automatically pushed onto the stack. If the contents of the status is important, and if the subroutine is likely to corrupt the status register, the programmer should take precautions and save it properly.

Interrupts

The device provides one external interrupts, three internal timer/event counter interrupts, an ADC interrupt and touch key interrupt. The interrupt control register 0 (INTC0;0BH) and interrupt control register 1 (INTC1;1EH) both contain the interrupt control bits that are used to set the enable/ disable status and interrupt request flags.

Once an interrupt subroutine is serviced, other interrupts are all blocked, by clearing the EMI bit). This scheme may prevent any further interrupt nesting. Other interrupt requests may take place during this interval, but only the interrupt request flag will be recorded. If a certain interrupt requires servicing within the service routine, the EMI bit and the corresponding bit of the INTC0 or of INTC1 may be set in order to allow interrupt nesting. Once the stack is full, the interrupt request will not be acknowledged, even if the related interrupt is enabled, until the SP is decremented. If immediate service is desired, the stack should be prevented from becoming full.

All interrupts will provide a wake-up function. As an interrupt is serviced, a control transfer occurs by pushing the contents of the program counter onto the stack fol-

Bit No.	Label	Function
0	C	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.
1	AC	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.
2	Z	Z is set if the result of an arithmetic or logic operation is zero; otherwise Z is cleared.
3	OV	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.
4	PDF	PDF is cleared by either a system power-up or executing the ″CLR WDT″ instruction. PDF is set by executing the ″HALT″ instruction.
5	TO	TO is cleared by a system power-up or executing the ″CLR WDT″ or ″HALT″ instruction. TO is set by a WDT time-out.
6~7	—	Unused bit, read as ″0″

Status (0AH) Register

lowed by a branch to a subroutine at the specified location in the Program Memory. Only the contents of the program counter is pushed onto the stack. If the contents of the register or of the status register is altered by the interrupt service program which corrupts the desired control sequence, the contents should be saved in advance.

An external interrupt is triggered by an edge transition on INT (A configuration option selects: high to low, low to high, both low to high and high to low), and the related interrupt request flag (EIF; bit 4 of INTC0) is set as well. After the interrupt is enabled, the stack is not full, and the external interrupt is active, a subroutine call to location 04H occurs. The interrupt request flag (EIF) and EMI bits are all cleared to disable other maskable interrupts.

The internal Timer/Event Counter 0 interrupt is initialized by setting the Timer/Event Counter 0 interrupt request flag (T0F; bit 5 of INTC0), which is normally caused by a timer overflow. After the interrupt is enabled, and the stack is not full, and the T0F bit is set, a subroutine call to location 08H occurs. The related interrupt request flag (T0F) is reset, and the EMI bit is cleared to disable other maskable interrupts. Timer/Event Counter 1 and Timer/Event Counter 2 are operated in the same manner but its related interrupt request flag is T1F and T2F (bit 6 of INTC0 and bit 4 of INTC1) and its subroutine call location is 0CH and 10H.

The A/D Converter interrupt is initialized by setting the A/D Converter interrupt request flag (ADF; bit 5 of INTC1), that is caused by an A/D conversion done signal. After the interrupt is enabled, and the stack is not full, and the ADF bit is set, a subroutine call to location 14H occurs. The related interrupt request flag (ADF) is reset and the EMI bit is cleared to disable further maskable interrupts.

During the execution of an interrupt subroutine, other maskable interrupt acknowledgments are all held until the "RETI" instruction is executed or the EMI bit and the related interrupt control bit are set both to 1 (if the stack is not full). To return from the interrupt subroutine, "RET" or "RETI" may be invoked. RETI sets the EMI bit and enables an interrupt service, but RET does not.

Interrupts occurring in the interval between the rising edges of two consecutive T2 pulses are serviced on the latter of the two T2 pulses if the corresponding interrupts are enabled. In the case of simultaneous requests, the priorities in the following table apply. These can be masked by resetting the EMI bit.

Bit No.	Label	Function
0	EMI	Controls the master (global) interrupt (1=enabled; 0=disabled)
1	EEI	Controls the external interrupt (1=enabled; 0=disabled)
2	ET0I	Controls the Timer/Event Counter 0 interrupt (1=enabled; 0=disabled)
3	ET1I	Controls the Timer/Event Counter 1 interrupt (1=enabled; 0=disabled)
4	EIF	External interrupt request flag (1=active; 0=inactive)
5	T0F	Internal Timer/Event Counter 0 request flag (1=active; 0=inactive)
6	T1F	Internal Timer/Event Counter 1 request flag (1=active; 0=inactive)
7	—	For test mode used only. Must be written as "0"; otherwise may result in unpredictable operation.

INTC0 Register

Bit No.	Label	Function
0	ET2I	Control the Timer/Event Counter 2 interrupt (1=enabled; 0=disabled)
1	EADI	Control the ADC interrupt (1=enabled; 0=disabled)
2	TKE	Control touch key interrupt (1=enabled; 0=disabled)
3	—	Unused bit, read as "0"
4	T2F	Internal Timer/Event Counter 2 request flag (1=active; 0=inactive)
5	ADF	ADC request flag (1=active; 0=inactive)
6	TKF	Touch key interrupt (1=active; 0=inactive)
7	—	Unused bit, read as "0"

INTC1 Register

Interrupt Source	Priority	Vector
External interrupt	1	04H
Timer/Event Counter 0 overflow	2	08H
Timer/Event Counter 1 overflow	3	0CH
Timer/Event Counter 2 overflow	4	10H
A/D converter interrupt	5	14H
Touch Key interrupt	6	18H

Once the interrupt request flags (TKF, ADF, T2F, T1F, T0F and EIF) are all set, they remain in the INTC1 or INTC0 respectively until the interrupts are serviced or cleared by a software instruction.

It is recommended that a program should not use the "CALL subroutine" within the interrupt subroutine. It's because interrupts often occur in an unpredictable manner or require to be serviced immediately in some applications. During that period, if only one stack is left, and enabling the interrupt is not well controlled, operation of the "call" in the interrupt subroutine may damage the original control sequence.

Interrupts for Touch Key interrupt

The Touch Key interrupt is initialised by setting the Touch Key interrupt request flag, TKF, bit 6 of INTC1. This is caused by a signal completion of the Touch Key sensor. After the interrupt is enabled, and the stack is not full, and the TKF bit is set, a subroutine call to location 18H occurs. The related interrupt request flag, TKF, will be reset and the EMI bit is cleared to disable further maskable interrupts.

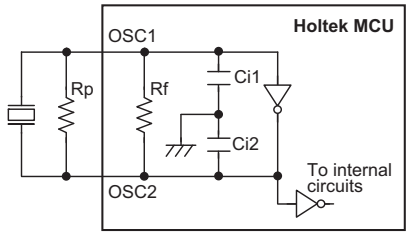
Oscillator Configuration

The device provides three system oscillator circuits known as a crystal oscillator (HXT), an external RC oscillator (ERC) and an internal high speed RC oscillator (HIRC) which are used for the system clock. There are also an internal 12kHz RC (LIRC) and a 32.768kHz crystal oscillator (LXT) which can provide a source clock for the WDT clock named f_S, the LCD driver clock named f_{SUB} and the Timer/Event counters low frequency clock named f_L for various timing purposes.

In the Power down mode, the system oscillator, the internal 12kHz RC oscillator (LIRC) or the external 32.768kHz crystal oscillator (LXT) may be enabled or disabled depending upon the corresponding clock control bit described in the relevant sections. The system can be woken-up from the Power down mode by the occurrence of an interrupt, a transition determined by configuration options on any of the Port A pins, a WDT overflow or a timer overflow.

External Crystal/ Ceramic Oscillator – HXT

The External Crystal/Ceramic System Oscillator is one of the system oscillator choices, which is selected via configuration options. 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, if a resonator instead of crystal is connected between OSC1 and OSC2, 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 for oscillation to occur. The values of C1 and C2 should be selected in consultation with the crystal or resonator manufacturer's specification.

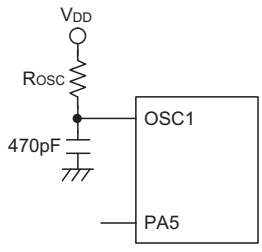


Note: 1. Rp is normally not required.
2. Although not shown OSC1/OSC2 pins have a parasitic capacitance of around 7pF.

External Crystal/Ceramic Oscillator

External RC Oscillator – ERC

Using the ERC oscillator only requires that a resistor, with a value between 24kΩ and 1.5MΩ, 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 PA6, leaving pin PA5 free for use as a normal I/O pin.



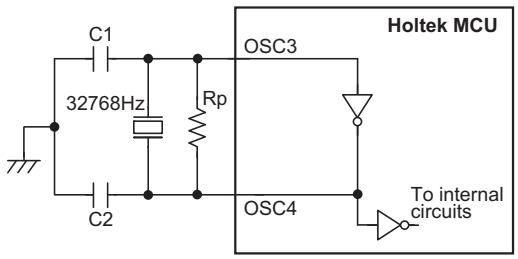
External RC Oscillator – ERC

Internal RC Oscillator – HIRC

The internal RC oscillator is a fully integrated system oscillator requiring no external components. The internal RC oscillator has three fixed frequencies of either 4MHz, 8MHz or 12MHz. Device trimming during the manufacturing process and the inclusion of internal frequency compensation circuits are used to ensure that the influence of the power supply voltage, temperature and process variations on the oscillation frequency are minimised. As a result, at a power supply of either 3V or 5V and at a temperature of 25°C degrees, the fixed oscillation frequency of 4MHz, 8MHz or 12MHz will have a tolerance within 2%. Note that if this internal system clock option is selected, as it requires no external pins for its operation, I/O pins PA5 and PA6 are free for use as normal I/O pins.

External 32.768kHz Crystal Oscillator – LXT

The External 32.768kHz Crystal Oscillator is one of the low frequency oscillator choices, which is selected via a configuration option. This clock source has a fixed frequency of 32.768kHz and requires a 32.768kHz crystal to be connected between pins OSC3 and OSC4. 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.



Note: 1. RTC OSC: without build-in RC
2. Rp, C1 and C2 are required.
3. Although not shown pins have a parasitic capacitance of around 7pF.

External 32.768kHz Oscillator – LXT

When the microcontroller enters the Power down 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 Power down Mode. To do this, another clock, independent of the system clock, must be provided.

The exact values of C1 and C2 should be selected in consultation with the crystal or resonator manufacturer's specification. The external parallel feedback resistor, Rp, is required.

LXT Oscillator C1 and C2 Values		
Crystal Frequency	**C1**	**C2**
32768Hz	8pF	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 QOSC bit in the CTRL0 register.

QOSC Bit	LXT Mode
0	Quick Start
1	Low-power

After power on the QOSC 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 QOSC 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 QOSC bit high about 2 seconds after power-on.

It should be noted that, no matter what condition the QOSC 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 12kHz Oscillator – LIRC

The Internal 12kHz RC 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 period of approximately 65□gs at 5V, requiring no external components for its implementation. If the system enters the Power Down Mode, the internal RC oscillator can still continue to run if its clock is necessary to be used to clock the functions for timing purpose such as the WDT function, LCD Driver or Timer/Event Counters. The internal RC oscillator can be disabled only when it is not used as the clock source for all the peripheral functions determined by the configuration options of the WDT function and the relevant control bits which determine the clock is enabled or disabled for related peripheral functions.

Watchdog Timer – WDT

The WDT is implemented using an internal 12kHz RC oscillator known as LIRC, an external 32.768kHz crystal oscillator or the instruction clock which is the system clock divided by 4. The timer is designed to prevent a software malfunction or sequence from jumping to an unknown location with unpredictable results. The watchdog timer can be disabled by a configuration option. If the watchdog timer is disabled, the WDT timer will have the same manner as in the enable-mode except that the timeout signal will not generate a chip reset. So in the watchdog timer disable mode, the WDT timer counter can be read out and can be cleared. This function is used for the application program to access the WDT frequency to get the temperature coefficient for analog component adjustment. The LIRC oscillator can be disabled or enabled by the oscillator enable control bits WDTOSC1 and WDTOSC0 in the WDT control register WDTC for power saving reasons.

There are 2 registers related to the WDT function named WDTC and WDTD. The WDTC register can control the WDT oscillator enable/disable and the WDT power source. The WDTD register is the WDT counter content register and this register is read only.

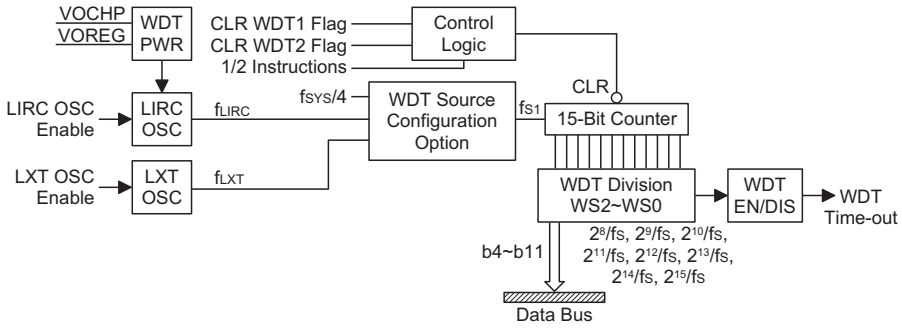
The WDT power source selection bits named WDTPWR1 and WDTPWR0 can be used to choose the WDT power source, the WDT default power source is from VOCHP. The main purpose of the regulator is to be used for the WDT Temperature-coefficient adjustment. In this case, the application program should enable the regulator before switching to the Regulator source. The WDTOSC1 and WDTOSC0 bits can be used to enable or disable the LIRC oscillator (12kHz). If the application does not use the LIRC oscillator, then it needs to disable it in order to save power. When the LIRC oscillator is disabled, then it is actually turned off, regardless of the setting of the relevant control bits which select the LIRC oscillator as its clock source. When the LIRC oscillator is enabled, it can be used as the clock source in the Power Down mode defined by the corresponding control bits of the peripheral functions.

Once the internal 12kHz RC oscillator LIRC with period 65μs normally is selected, it is divided by max. 2^{15} to get the time-out period of approximately 2.15s. This time-out period may vary with temperature, VDD and process variations.

The WDT clock source may also come from the instruction clock, in which case the WDT will operate in the same manner except that in the Power Down mode the WDT may stop counting and lose its protecting purpose. In this situation the device can only be restarted by external logic. If the device operates in a noisy environment, using the on-chip LIRC oscillator is strongly recommended, since the HALT instruction will stop the system clock.

The WDT overflow under normal operation initializes a "chip reset" and sets the status bit "TO". In the Power Down Mode, the overflow initializes a "warm reset", and only the PC and SP are reset to zero. There are three methods to clear the contents of the WDT, an external reset (a low level on $\overline{RES}$), a software instruction or a "HALT" instruction. There are two types of software instructions; the single "CLR WDT" instruction, or the pair of instructions – "CLR WDT1" and "CLR WDT2".

Of these two types of instruction, only one type of instruction can be active at a time depending on the configuration option – "CLR WDT" times selection option. If the "CLR WDT" is selected (i.e., CLR WDT times equal one), any execution of the "CLR WDT" instruction clears the WDT. If the "CLR WDT1" and "CLR WDT2" option is chosen (i.e., CLR WDT times equal two), these two instructions have to be executed to clear the WDT, otherwise the WDT may reset the chip due to a time-out.



Watchdog Timer

<table>
<tr><th>Bit No.</th><th>Label</th><th>Function</th></tr>
<tr><td>0
1</td><td>WDTPWR0~
WDTPWR1</td><td>The WDT Power source selection.
01: WDT power comes from VOCHP
10: WDT power comes from Regulator
00/11: WDT power comes from VOCHP
It is strongly recommend to use ″01″ for VOCHP to prevent the noise to let the WDT lose the power</td></tr>
<tr><td>2
3</td><td>WDTOSC0~
WDTOSC1</td><td>The LIRC oscillator enable/disable control bits
01: LIRC oscillator is disabled
10: LIRC oscillator is enabled
00/11: LIRC oscillator is enabled
It is strongly recommended to use ″10″ for WDT OSC enable</td></tr>
<tr><td>4</td><td>—</td><td>Reserved</td></tr>
<tr><td>5
6
7</td><td>WS0
WS1
WS2</td><td>WS2~WS0: WDT prescaler rate select
<table>
<tr><th>WS2</th><th>WS1</th><th>WS0</th><th>WDT Rate</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>$2^8/f_S$</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>$2^9/f_S$</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>$2^{10}/f_S$</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>$2^{11}/f_S$</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>$2^{12}/f_S$</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>$2^{13}/f_S$</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>$2^{14}/f_S$</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>$2^{15}/f_S$</td></tr>
</table></td></tr>
</table>

WDTC (1CH) Register

Note: The initial value of the WDTOSC1 and WDTOSC0 bits will be set to ″10″ to enable the LIRC oscillator if both the WDT function is enabled and the WDT clock is selected from the LIRC oscillator determined by the configuration options. Otherwise, the initial value of these two bits will be set to ″01″.

The WDT clock (f_S) is further divided by an internal counter to give longer watchdog time-out period. In this device, the division ratio can be varied by selecting different values of WS2~WS0bits to give $2^8/f_S$ to $2^{15}/f_S$ division ratio range.

Bit No.	Label	Function
0~7	WDTD0~ WDTD7	The WDT Counter value (bit4 ~ bit11) This register is read only and used for temperature adjusting.

WDTD (1DH) Register

The WDT clock (f_{S1}) is further divided by an internal counter to give longer watchdog time-outs., In this device, the division ratio can be varied by selecting different configuration options to give 2^{13} to 2^{16} division ration range.

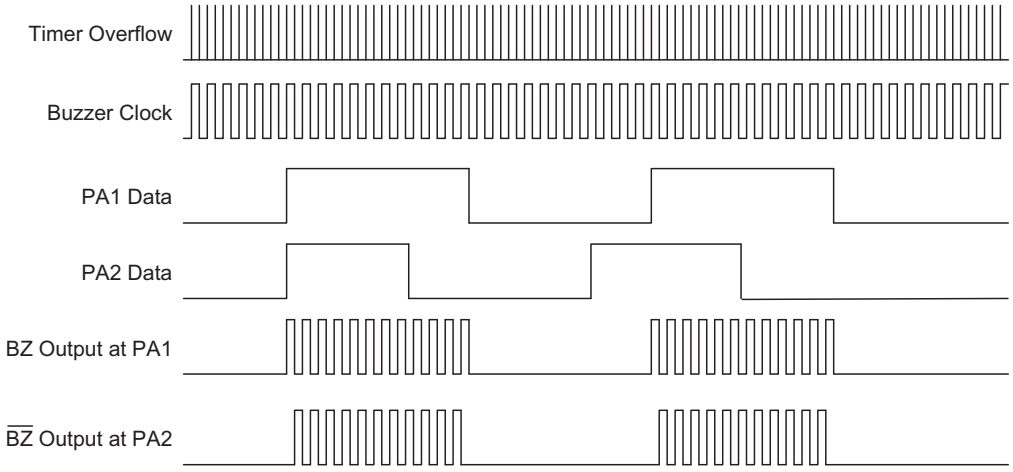
Buzzer Output

The Buzzer function provides a means of producing a variable frequency output, suitable for applications such as Piezo-buzzer driving or other external circuits that require a precise frequency generator. The BZ and $\overline{BZ}$ pins form a complimentary pair, and are pin-shared with I/O pins, PA1 and PA2. Configuration options are used to select from one of three buzzer options. The first option is for both pins PA1 and PA2 to be used as normal I/Os, the second option is for both pins to be configured as BZ and $\overline{BZ}$ buzzer pins, the third option selects only the PA1 pin to be used as a BZ buzzer pin with the PA2 pin retaining its normal I/O pin function. Note that the $\overline{BZ}$ pin is the inverse of the BZ pin which together generates a differential output which can supply more power to connected interfaces such as buzzers.

The buzzer is driven by the Timer/Event Counter 0 or Timer/Event Counter 1 overflow signal divided by 2 selected by the clock source selection bit named BZCS in CTRL1 register.

If the configuration options have selected both pins PA1 and PA2 to function as a BZ and $\overline{BZ}$ complementary pair of buzzer outputs, then for correct buzzer operation it is essential that both pins must be setup as outputs by setting bits PAC1 and PAC2 of the PAC port control register to zero. The PA1 data bit in the PA data register must also be set high to enable the buzzer outputs, if set low, both pins PA1 and PA2 will remain low. In this way the single bit PA1 of the PA data register can be used as an on/off control for both the BZ and $\overline{BZ}$ buzzer pin outputs. Note that the PA2 data bit in the PA data register has no control over the $\overline{BZ}$ buzzer pin PA2.

If configuration options have selected that only the PA1 pin is to function as a BZ buzzer pin, then the PA2 pin can be used as a normal I/O pin. For the PA1 pin to function as a BZ buzzer pin, PA1 must be setup as an output by setting bit PAC1 of the PAC port control register to zero. The PA1 data bit in the PA data register must also be set high to enable the buzzer output, if set low pin PA1 will remain low. In this way the PA1 bit can be used as an on/off control for the BZ buzzer pin PA1. If the PAC1 bit of the PAC port control register is set high, then pin PA1 can still be used as an input even though the configuration option has configured it as a BZ buzzer output.



Buzzer Output Pin Control

PAC Register PAC1	PAC Register PAC2	PA Data Register PA1	PA Data Register PA2	Output Function
0	0	0	X	PA1=″0″, PA2=″0″
0	0	1	X	PA1=BZ, PA2=$\overline{BZ}$
0	1	0	X	PA1=″0″, PA2=Input Line
0	1	1	X	PA1=BZ, PA2=Input Line
1	0	1	X	PA1=Input Line, PA2=$\overline{BZ}$
1	0	0	X	PA1=Input Line, PA2=″0″
1	1	X	X	PA1=Input Line, PA2=Input Line

″X″ stands for don't care

PA1/PA2 Pin Function Control

Note that no matter what configuration option is chosen for the buzzer, if the port control register has setup the pin to function as an input, then this will override the configuration option selection and force the pin to always behave as an input pin. This arrangement enables the pin to be used as both a buzzer pin and as an input pin, so regardless of the configuration option chosen; the actual function of the pin can be changed dynamically by the application program by programming the appropriate port control register bit.

Note: The above drawing shows the situation where both pins PA1 and PA2 are selected by configuration option to be BZ and $\overline{BZ}$ buzzer pin outputs. The Port Control Register of both pins must have already been setup as outputs. The data setup on pin PA2 has no effect on the buzzer outputs.

Power Down Operation – HALT

The Power down mode is initialised by the ″HALT″ instruction and results in the following.

- The system oscillator stops running if the system oscillator is selected to be turned off by clearing the OSCON bit in the HALTC register to zero. Otherwise, the system oscillator will keep running if it is selected to be turned on in the power down mode.
- The contents of the on-chip Data Memory and of the registers remain unchanged.
- The WDT is cleared and starts recounting (if the WDT clock source is from the LIRC or the LXT oscillator).
- All I/O ports maintain their original status.
- The PDF flag is set but the TO flag is cleared.
- The LCD driver keeps running if the LCD clock f_{SUB} is enabled by setting the FSUBC bit to ″1″ and the LCDON bit in the HALTC register is set to ″1″.

The system leaves the Power down mode by means of an external reset, an interrupt, an external transition signal on Port A, or a WDT overflow. An external reset causes device initialisation, and the WDT overflow performs a ″warm reset″. After examining the TO and PDF flags, the reason for chip reset can be determined. The PDF flag is cleared by system power-up or by executing the ″CLR WDT″ instruction, and is set by executing the ″HALT″ instruction. On the other hand, the TO flag is set if WDT time-out occurs, and causes a wake-up that only resets the program counter and SP, and leaves the others in their original state.

The port A wake-up and interrupt methods can be considered as a continuation of normal execution. Each pin of port A can be independently selected to wake-up the device using configuration options. After awakening from an I/O port stimulus, the program will resume execution at the next instruction. However, if awakening from an interrupt, two sequences may occur. If the related interrupt is disabled or the interrupt is enabled but the stack is full, the program will resume execution at the next instruction. But if the interrupt is enabled, and the stack is not full, the regular interrupt response takes place.

When an interrupt request flag is set before entering the ″HALT″ status, the system cannot be awakened using that interrupt.

If a wake-up events occur, it takes 1024 t_{SYS} (system clock periods) to resume normal operation. In other words, a dummy period is inserted after the wake-up. If the wake-up results from an interrupt acknowledgment, the actual interrupt subroutine execution is delayed by more than one cycle. However, if the wake-up results in the next instruction execution, the execution will be performed immediately after the dummy period is finished.

To minimize power consumption, all the I/O pins should be carefully managed before entering the HALT status.

Bit No.	Label	Function
0	LCDON	LCD module state in Power down mode 1: LCD module remains on (if fSUB is active) regardless of the configuration option setting 0: LCD state is determined by the LCD_ON configuration option
1~6		Reserved, read as ″0″
7	OSCON	System oscillator state in Power down mode 1: System oscillator keeps running in Power down mode 0: System oscillator stops running in Power down mode

HALTC Register

Reset

There are three ways in which a reset may occur.

- $\overline{RES}$ is reset during normal operation
- $\overline{RES}$ is reset during HALT
- WDT time-out is reset during normal operation

The WDT time-out during Power Down Mode differs from other chip reset conditions, for it can perform a "warm reset" that resets only the program counter and SP and leaves the other circuits at their original state. Some registers remain unaffected during any other reset conditions. Most registers are reset to their initial conditions once the reset conditions are met. By examining the PDF and TO flags, the program can distinguish between different chip resets.

TO	PDF	RESET Conditions
0	0	$\overline{RES}$ reset during power-up
u	u	$\overline{RES}$ reset during normal operation
0	1	$\overline{RES}$ Wake-up HALT
1	u	WDT time-out during normal operation
1	1	WDT Wake-up HALT

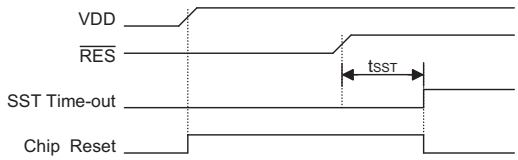
Note: "u" stands for unchanged

To guarantee that the system oscillator is started and stabilized, the SST (System Start-up Timer) provides an extra-delay of 1024 system clock pulses when the system awakes from the HALT state or during power-up. Awaking from the HALT state or system power-up, the SST delay is added.

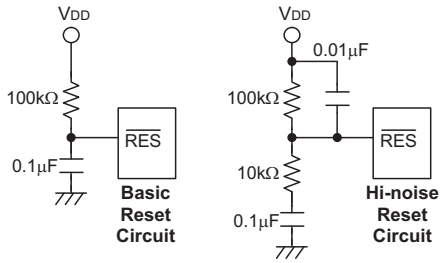
An extra SST delay is added during the power-up period, and any wake-up from HALT may enable only the SST delay.

The functional unit chip reset status is shown below.

Program Counter	000H
Interrupt	Disabled
Prescaler, Divider	Cleared
WDT	Cleared. After master reset, WDT starts counting
Timer/Event Counter	Off
Input/output Ports	Input mode
Stack Pointer	Points to the top of the stack

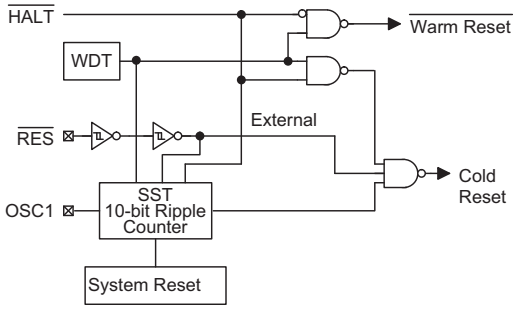


Reset Timing Chart



Reset Circuit

Note: Most applications can use the Basic Reset Circuit as shown, however for applications with extensive noise, it is recommended to use the Hi-noise Reset Circuit.



Reset Configuration

The register states are summarized below:

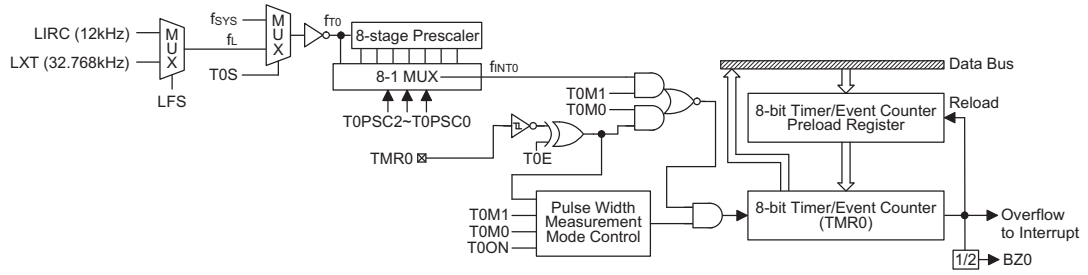
Register	Reset (Power On)	WDT Time-out (Normal Operation)	$\overline{RES}$ Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-out (HALT)*
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuuuuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
BP	---- ---0	---- ---0	---- ---0	---- ---0	---- ---u
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
Program Counter	0000H	0000H	0000H	0000H	0000H
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	-xxx xxxx	-uuu uuuu	-uuu uuuu	-uuu uuuu	-uuu uuuu
CTRL0	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTRL1	---- --01	---- --01	---- --01	---- --01	---- --uu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC0	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	-000 -000	-000 -000	-000 -000	-000 -000	-uuu -uuu
TMR0	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR0C	0000 1000	0000 1000	0000 1000	0000 1000	uuuu uuuu
TMR1H	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1L	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1C	0000 1000	0000 1000	0000 1000	0000 1000	uuuu uuuu
TMR2H	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR2L	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR2C	0000 1000	0000 1000	0000 1000	0000 1000	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
CHPRC	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
WDTC	111- ss01	111- ss01	111- ss01	111- ss01	uuu- uuuu
WDTD	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000
ADCR	-000 x000	-000 x000	-000 x000	-000 x000	-uuu xuuu
ADCD	0--0 0111	0--0 0111	0--0 0111	0--0 0111	u--u uuuu
VIBRC	---- ---0	---- ---0	---- ---0	---- ---0	---- ---u
LCDOUT	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu
CFCR0	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
CFCR1	---- -000	---- -000	---- -000	---- -000	---- -uuu
ANCS0	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu
HALTC	0--- ---0	0--- ---0	0--- ---0	0--- ---0	u--- ---u

Note: "*" stands for warm reset
"u" stands for unchanged
"x" stands for unknown

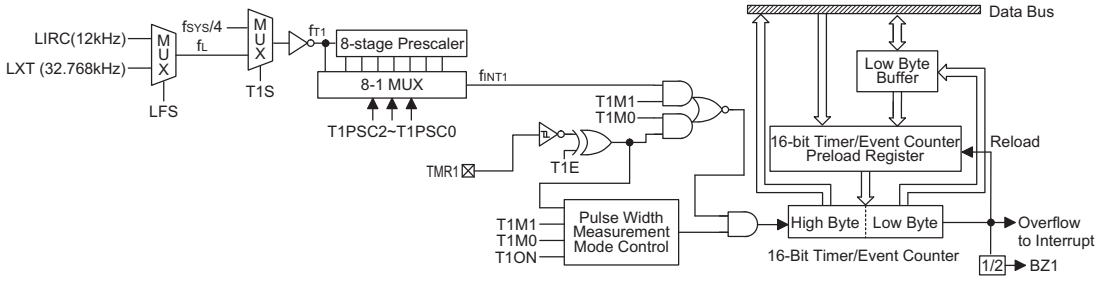
Timer/Event Counter

Three timer/event counters are implemented in the microcontroller. Timer/Event Counter 0 contains an 8-bit programmable count-up counter whose clock may come from an external source or an internal clock source. An internal clock source comes from f_{SYS} or the Internal low frequency clock known as f_L. Timer/Event Counter 1 contains a 16-bit programmable count-up counter whose clock may come from an external source or an internal clock source. An internal clock source comes from $f_{SYS}/4$ or the Internal low frequency clock known as f_L. The clock f_L is derived from the LIRC or LXT oscillator and can be selected by the Low Frequency selection bit LFS bit in the CTRL0 register. The external clock input allows the user to count external events, measure time intervals or pulse widths, or to generate an accurate time base. Timer/Event Counter 2 contains a 16-bit programmable count-up counter whose clock may come from an external source or an internal clock source. An internal clock source comes from $f_{SYS}/4$ or the Timer/Event Counter 2 internal clock f_{TCK}. The clock f_{TCK} may come from the low frequency clock f_L or the clocks generated from the Touch Key module named f_{REF}, f_{SEN} and f_{TMCK} described in the Touch Key Function section. The clock is selected using the Timer/Event Counter 2 clock source selection bits TCKS1 and TCK0 in the CTRL0 register. The external clock input allows the user to count external events, measure time intervals or pulse widths, or to generate an accurate time base.

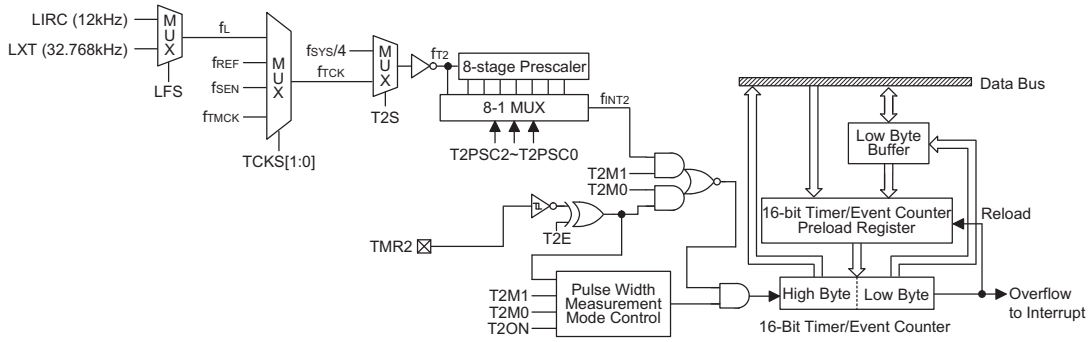
There are two registers related to the Timer/Event Counter 0; TMR0 and TMR0C. Writing to TMR0 puts the starting value in the Timer/Event Counter 0 register and reading TMR0 reads out the contents of Timer/Event Counter 0. The TMR0C is a timer/event counter control register, which defines the overall operations. There are three registers related to the Timer/Event Counter 1; TMR1H, TMR1L and TMR1C. Writing to TMR1L will only put the written data into an internal lower-order byte buffer (8-bit) while writing to TMR1H will transfer the specified data and the contents of the lower-order byte buffer to both the TMR1H and TMR1L registers, respectively. The Timer/Event Counter 1 preload register is changed when each time there is a write operation to TMR1H. Reading TMR1H will latch the contents of TMR1H and TMR1L counters to the destination and the lower-order byte buffer, respectively. Reading TMR1L will read the contents of the lower-order byte buffer. TMR1C is the Timer/Event Counter 1 control register, which defines the operating mode, counting enable or disable, the TMR1 active edge and the prescaler stage selections. Also there are three registers related to the Timer/Event Counter 2 named TMR2H, TMR2L and TMR2C. The operations of reading from and writing to the Timer/Event Counter 2 registers named TMR2H and TMR2L are the same with Timer/Event Counter 1 described above.



Timer/Event Counter 0



Timer/Event Counter 1



Timer/Event Counter 2

Bit No.	Label	Function
0 1 2	T0PSC0 T0PSC1 T0PSC2	To define the prescaler stages, T0PSC2, T0PSC1, T0PSC0= 000: $f_{INT0}=f_{T0}$ 001: $f_{INT0}=f_{T0}/2$ 010: $f_{INT0}=f_{T0}/4$ 011: $f_{INT0}=f_{T0}/8$ 100: $f_{INT0}=f_{T0}/16$ 101: $f_{INT0}=f_{T0}/32$ 110: $f_{INT0}=f_{T0}/64$ 111: $f_{INT0}=f_{T0}/128$
3	T0E	Defines the TMR0 active edge of the timer/event counter: In Event Counter Mode (T0M1,T0M0)=(0,1): 1:count on falling edge; 0:count on rising edge In Pulse Width measurement mode (T0M1,T0M0)=(1,1): 1: start counting on the rising edge, stop on the falling edge; 0: start counting on the falling edge, stop on the rising edge
4	T0ON	Enable/disable timer counting (0=disabled; 1=enabled)
5	T0S	Defines the TMR0 internal clock source 0: f_{SYS} 1: Low Frequency clock f_L
6 7	T0M0 T0M1	Defines the operating mode T0M1, T0M0= 01: Event count mode (External clock) 10: Timer mode (Internal clock) 11: Pulse Width measurement mode (External clock) 00: Unused

TMR0C (0EH) Register

Bit No.	Label	Function
0 1 2	T1PSC0 T1PSC1 T1PSC2	To define the prescaler stages, T1PSC2, T1PSC1, T1PSC0= 000: $f_{INT1}=f_{T1}$ 001: $f_{INT1}=f_{T1}/2$ 010: $f_{INT1}=f_{T1}/4$ 011: $f_{INT1}=f_{T1}/8$ 100: $f_{INT1}=f_{T1}/16$ 101: $f_{INT1}=f_{T1}/32$ 110: $f_{INT1}=f_{T1}/64$ 111: $f_{INT1}=f_{T1}/128$
3	T1E	Defines the TMR1 active edge of the timer/event counter: In Event Counter Mode (T1M1,T1M0)=(0,1): 1: count on falling edge; 0: count on rising edge In Pulse Width measurement mode (T1M1,T1M0)=(1,1): 1: start counting on the rising edge, stop on the falling edge; 0: start counting on the falling edge, stop on the rising edge
4	T1ON	Enable/disable timer counting (0=disabled; 1=enabled)
5	T1S	Defines the TMR1 internal clock source 0: $f_{SYS}/4$ 1: Low Frequency clock f_L
6 7	T1M0 T1M1	Defines the operating mode T1M1, T1M0= 01: Event count mode (External clock) 10: Timer mode (Internal clock) 11: Pulse Width measurement mode (External clock) 00: Unused

TMR1C (11H) Register

Bit No.	Label	Function
0 1 2	T2PSC0 T2PSC1 T2PSC2	To define the prescaler stages, T2PSC2, T2PSC1, T2PSC0= 000: $f_{INT2}=f_{T2}$ 001: $f_{INT2}=f_{T2}/2$ 010: $f_{INT2}=f_{T2}/4$ 011: $f_{INT2}=f_{T2}/8$ 100: $f_{INT2}=f_{T2}/16$ 101: $f_{INT2}=f_{T2}/32$ 110: $f_{INT2}=f_{T2}/64$ 111: $f_{INT2}=f_{T2}/128$
3	T2E	Defines the TMR2 active edge of the timer/event counter: In Event Counter Mode (T2M1,T2M0)=(0,1): 1: count on falling edge; 0: count on rising edge In Pulse Width measurement mode (T2M1,T2M0)=(1,1): 1: start counting on the rising edge, stop on the falling edge; 0: start counting on the falling edge, stop on the rising edge
4	T2ON	Enable/disable timer counting (0=disabled; 1=enabled)
5	T2S	Defines the TMR2 internal clock source 0: $f_{SYS}/4$ 1: f_{TCK}
6 7	T2M0 T2M1	Defines the operating mode T2M1, T2M0= 01: Event count mode (External clock) 10: Timer mode (Internal clock) 11: Pulse Width measurement mode (External clock) 00: Unused

TMR2C Register

The TxM0 and TxM1 bits in TMRxC register where x may be equal to 0, 1 or 2 define the operation mode. The event count mode is used to count external events, which means that the clock source must come from the external (TMR0, TMR1 or TMR2) pin. The timer mode functions as a normal timer with the clock source coming from the internal selected clock source. Finally, the pulse width measurement mode can be used to count a high or low level duration of an external signal on the TMR0, TMR1 or TMR2 pins with the timing based on the internally selected clock source.

In the event count or timer mode, the Timer/Event Counter starts counting at the current contents in the Timer/Event Counter and ends at FFH for -8-bit counter or FFFFH for 16-bit counter. Once an overflow occurs, the counter is reloaded from the timer/event counter preload register, and generates an interrupt request flag, T0F, T1F or T2F. In the pulse width measurement mode with the values of the Timer enable control bit TxON and the active edge control bit TxE equal to ″1″, after the TMRx pin has received a transient from low to high (or high to low if the TxE bit is ″0″), it will start counting until the TMRx pin returns to the original level and resets the TxON bit. The measured result remains in the timer/event counter even if the activated transient occurs again. Therefore, only a 1-cycle measurement can be made until the TxON bit is again set. The cycle measurement will re-function as long as it receives further transient pulses. In this operation mode, the timer/event counter begins counting not according to the logic level but to the transient edges. In the case of counter overflows, the counter is reloaded from the timer/event counter register and issues an interrupt request, as in the other two modes, i.e., event and timer modes.

To enable the counting operation, the Timer enable bit known as TxON in TMRxC where x indicates 0, 1 or 2 should be set to ″1″. In the pulse width measurement mode, the TxON is automatically cleared after the measurement cycle is completed. But in the other two modes, the TxON bit can only be reset by instructions. The overflow of the Timer/Event Counters is one of the wake-up sources. No matter what the operation mode is, writing a ″0″ to the related Timer/Event counter interrupt enable control bit ETxI disables the related interrupt service.

In the case of a Timer/Event Counter OFF condition, writing data to the timer/event counter preload register also reloads that data to the timer/event counter. But if the timer/event counter is turned on, data written to the timer/event counter is kept only in the timer/event counter preload register. The timer/event counter still continues its operation until an overflow occurs.

When the Timer/Event Counter Register TMRx or TMRxH/TMRxL is read, the clock is blocked to avoid errors, however as this may result in a counting error, it should be taken into account by the programmer. It is strongly recommended to load a desired value into the Timer/Event Counter Register TMRx or TMRxH/TMRxL first, before turning on the related timer/event counter, for proper operation since the initial value of TMRx or TMRxH/TMRxL is unknown. Due to the Timer/Event Counter scheme, the programmer should pay special attention to the instructions which enables then disables the timer for the first time, whenever there is a need to use the timer/event counter function, to avoid unpredictable results. After this procedure, the timer/event function can be operated normally.

The bit0~bit2 of the Timer/Event Counter control register TMRxC can be used to define the pre-scaling stages of the internal clock sources of Timer/Event Counters.

Input/Output Ports

There are maximum 16 bidirectional input/output lines in the microcontroller, labeled as PA, PB and PC. All of these I/O ports can be used for input and output operations. For input operation, these ports are non-latching, that is, the inputs must be ready at the T2 rising edge of instruction ″MOV A, [m]″. For output operation, all the data is latched and remains unchanged until the output latch is rewritten.

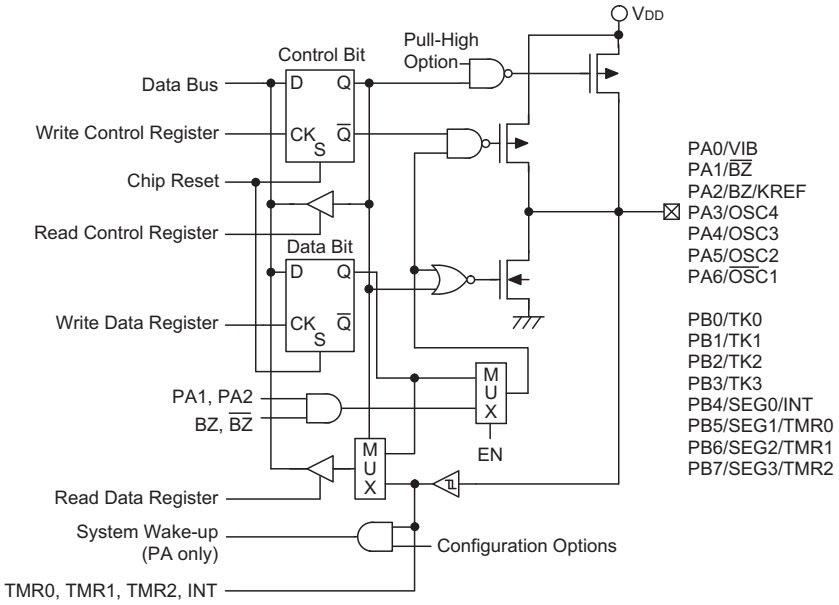
Each I/O line has its own control register, PAC, PBC and PCC, to control the input/output configuration. With this control register, CMOS outputs or Schmitt trigger inputs with or without pull-high resistor structures can be reconfigured dynamically under software control. To function as an input, the corresponding latch of the control register must write ″1″. The input source also depends on the control register. If the control register bit is ″1″, the input will read the pad state. If the control register bit is ″0″, the contents of the latches will move to the internal bus. The latter is possible in the ″read-modify-write″ instruction.

For output function, CMOS is the only configuration. These control registers are mapped to locations 13H, and 15H.

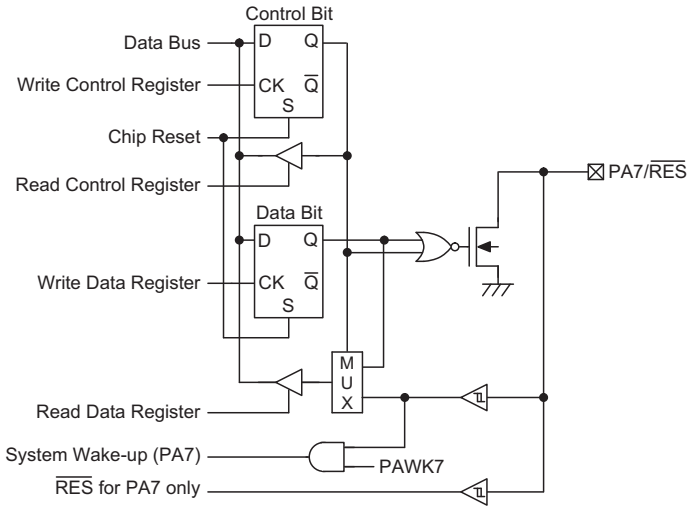
After a chip reset, these input/output lines remain at high levels or in a floating state, depending upon the pull-high configuration options. Each bit of these input/output latches can be set or cleared by ″SET [m].i″ and ″CLR [m].i″ (m=12H or 14H) instructions.

Some instructions first input data and then follow the output operations. For example, ″SET [m].i″, ″CLR [m].i″, ″CPL [m]″, ″CPLA [m]″ read the entire port states into the CPU, execute the defined operations (bit-operation), and then write the results back to the latches or the accumulator.

Each line of port A has the capability of waking-up the device.



Input/Output Ports



PA7 Pin

Each pin of these two I/O ports except PA7 pin has a pull-high resistor determined by a configuration option. Once the pull-high configuration option is selected, the I/O pin has a pull-high resistor connected. Take note that a non-pull-high I/O pin setup as an input will be in a floating condition.

PA1 and PA2 are pin-shared with BZ and $\overline{BZ}$ signal, respectively. If the BZ/$\overline{BZ}$ configuration option is selected, the output signals in the output mode of PA1/PA2 can be the buzzer signal. The input mode always retains its original function. Once the BZ/$\overline{BZ}$ configuration option is selected, the buzzer output signals are controlled by the PA1 data register.

The PA0/PA2 I/O function is shown below.

PA0 I/O	I	I	O	O	O	O	O	O	O	O	I	I
PA2 I/O	I	O	I	I	I	O	O	O	O	O	O	O
PA0 Mode	X	X	C	B	B	C	B	B	B	B	B	B
PA2 Mode	X	C	X	X	X	C	C	C	B	B	B	B
PA0 Data	X	X	D	0	1	D_0	0	1	0	1	0	1
PA2 Data	X	D	X	X	X	D_1	D	D	X	X	X	C
PA0 Pad Status	I	I	D	0	B	D_0	0	B	0	B	I	I
PA2 Pad Status	I	D	I	I	I	D_1	D	D	0	B	0	B

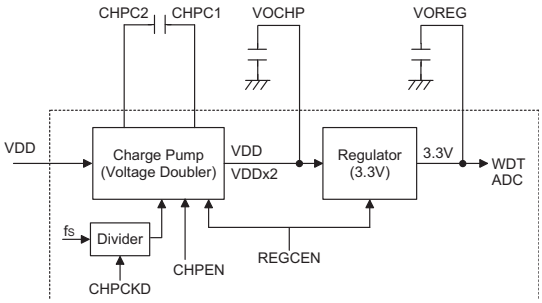
Note: ″I″ input; ″O″ output
″D, D0, D1″ Data
″B″ buzzer option, BZ or $\overline{BZ}$
″X″ don′t care
″C″ CMOS output

It is recommended that unused or not bonded out I/O lines should be set as output pins using software instructions to avoid consuming power when in an input floating state.

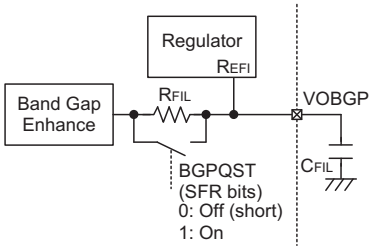
Charge Pump and Voltage Regulator

There is one charge pump and one voltage regulator implement in this device.

The charge pump can be enabled/disabled by the application program. The charge pump uses VDD as its input, and has the function of doubling the V_{DD} voltage. The output voltage of the charge pump will be V_{DD}×2. The regulator can generate a stable voltage of 3.3V, for internal WDT, ADC and also can provide an external bridge sensor excitation voltage or supply a reference voltage for other applications. The user needs to guarantee the charge pump output voltage is greater than 3.6V to ensure that the regulator generates the required 3.3V voltage output. The block diagram of this module is shown below.



Additionally, the device also includes a band gap voltage generator for the 1.5V low temperature sensitive reference voltage. This reference voltage is used as the zero adjustment and for a single end type reference voltage.



R_{FIL} is about 100kΩ and the recommend C_{FIL} is 10μF.

Note: VOBGP signal is only for chip internal used. Don′t connect to external component except the recommend C_{FIL}

There is a single register associated with this module named CHPRC. The CHPRC is the Charge Pump/Regulator Control register, which controls the charge pump on/off, regulator on/off functions as well as setting the clock divider value to generate the clock for the charge pump.

The CHPCKD4~CHPCKD0 bits are use to set the clock divider to generate the desired clock frequency for proper charge pump operation. The actual frequency is determined by the following formula.

Actual Charge Pump Clock= (f_{SYS}/16)/(CHPCKD +1).

Bit No.	Label	Function
0	REGCEN	Enable/disable Regulator/Charge-Pump module. (1=enable; 0=disable)
1	CHPEN	Charge Pump Enable/disable setting. (1=enable; 0=disable) Note: this bit will be ignore if the REGCEN is disable
2	BGPQST	Band gap quickly start-up function 0: R short, quickly start 1: R connected, normal RC filter mode Every time when REGCEN change from 0 to 1 (Regulator turn on) This bit should be set to 0 and then set to 1 to make sure the quickly stable. (the minimum 0 keeping time is about 2ms now)
3~7	CHPCKD0~ CHPCKD4	The Charge pump clock divider. This 5 bits can form the clock divide by 1~32. Following the below equation: Charge Pump clock = (f_{SYS}/16) / (CHPCKD+1)

CHPRC (1FH) Register

REGCEN	CHPEN	Charge Pump	VOCHP Pin	Regulator	VOREG Pin	OPA ADC	Description
0	X	OFF	V_{DD}	OFF	Hi-Impedance	Disable	The whole module is disable, OPA/ADC will lose the Power
1	0	OFF	V_{DD}	ON	3.3V	Active	Use for V_{DD} is greater than 3.6V (V_{DD}>3.6V)
1	1	ON	2×V_{DD}	ON	3.3V	Active	Use for V_{DD} is less than 3.6V (V_{DD}=2.2V~3.6V)

The suggested charge pump clock frequency is 20kHz. The application needs to set the correct value to get the desired clock frequency. For a 4MHz application, the CHPCKD bits should be set to the value 11, and for a 2MHz application, the bits should be set to 5.

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 will be disabled and the charge pump will be also be disabled to save power. When REGCEN = 0, the module will enter the Power Down Mode ignoring the CHPEN setting. The ADC and OPA will also be disabled to reduce power.

If REGCEN is set to ″1″, the regulator will be enabled. If CHPEN is enabled, the charge pump will be active and will use VDD as its input to generate the double voltage output. This double voltage will be used as the input voltage for the regulator. If CHPEN is set to ″0″, the charge pump is disabled and the charge pump output will be equal to the charge pump input, VDD.

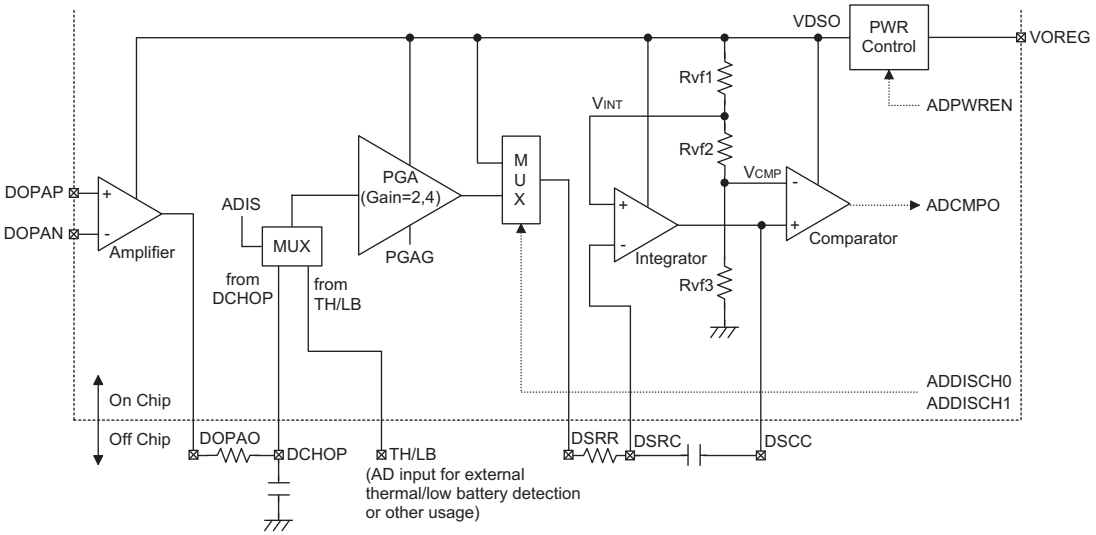
It is necessary to take care of the V_{DD} voltage. If the voltage is less than 3.6V, then CHPEN should be set to 1 to enable the charge pump, otherwise CHPEN should be set to zero. If the Charge pump is disabled and V_{DD} is less than 3.6V then the output voltage of the regulator will not be guaranteed.

ADC – Dual Slope

A Dual Slope A/D converter is implemented in this microcontroller. The dual slope module includes an Operational Amplifier, a Programmable Gain Amplifier PGA for the amplification of differential signals, an Integrator and a comparator for the main dual slope AD converter.

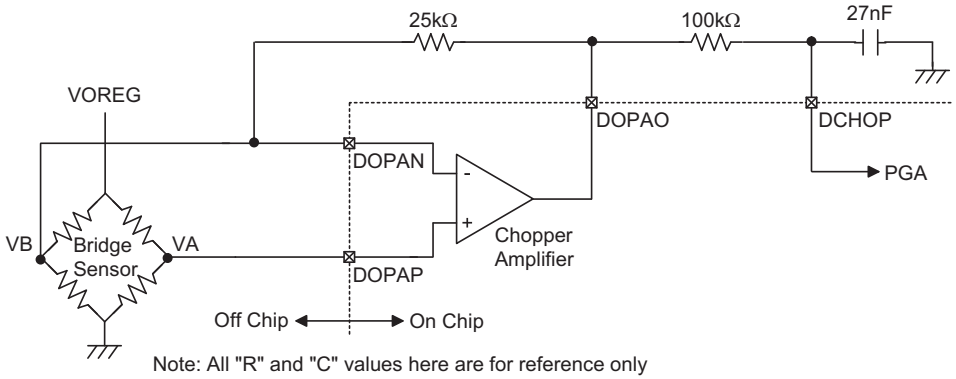
There are 2 special function registers related to this function known as ADCR and ADCD. The ADCR register is the A/D control register, which controls the ADC block power on/off, the chopper clock on/off, the charge/discharge control and is also used to read out the comparator output status. The ADCD register is the A/D Chopper clock divider register, which defines the chopper clock to the ADC module.

The ADPWREN bit, defined in ADCR register, is used to control the ADC module on/off function. The ADCCKEN bit defined in the ADCR register is used to control the chopper clock on/off function. When ADCCKEN is set to ″1″ it will enable the Chopper clock, with the clock frequency defined by the ADCD register. The ADC module includes the OPA, PGA, integrator and comparator. However, the Bandgap voltage generator is independent of this module. It will be automatically enabled when the regulator is enabled, and also be disabled when the regulator is disabled. The application program should enable the related power to permit them to function and disable them when entering the power down mode to conserve power. The charge/discharge control bits,



Note: VINT, VCMP signal can come from different R groups which are selected by software registers.

Dual Slope ADC Structure



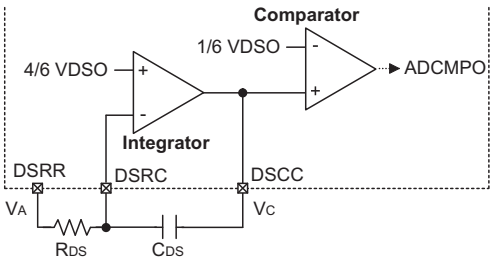
Note: All "R" and "C" values here are for reference only

Dual Slope ADC with Bridge Sensor input

ADDISCH1 and ADDISCH0, are used to control the Dual slope circuit charging and discharging behavior. The ADCMPO bit is read only for the comparator output, while the ADINTM bits can set the ADCMPO trigger mode for interrupt generation. The ADC PGA input signal can come from the DCHOP or TH/LB pin selected by the ADIS selection bit in ADCD register. The PGA gain can be either 2 or 4 determined by the PGAG gain selection bit in the ADCD register. The reference voltages of the ADC integrator and comparator named VINT and VCMP shown in the Dual Slope ADC structure diagram can be selected by the ADRR0 selection bit.

Dual Slope ADC Operation

The following descriptions are based on the fact that the ADRR0 bit is set to "0".



The amplifier and buffer combination, form a differential input pre-amplifier which amplifies the sensor input signal.

The combination of the Integrator, the comparator, the resistor R_{DS}, between DSRR and DSRC and the capacitor C_{DS}, between DSRC and DSCC form the main body of the Dual slope ADC.

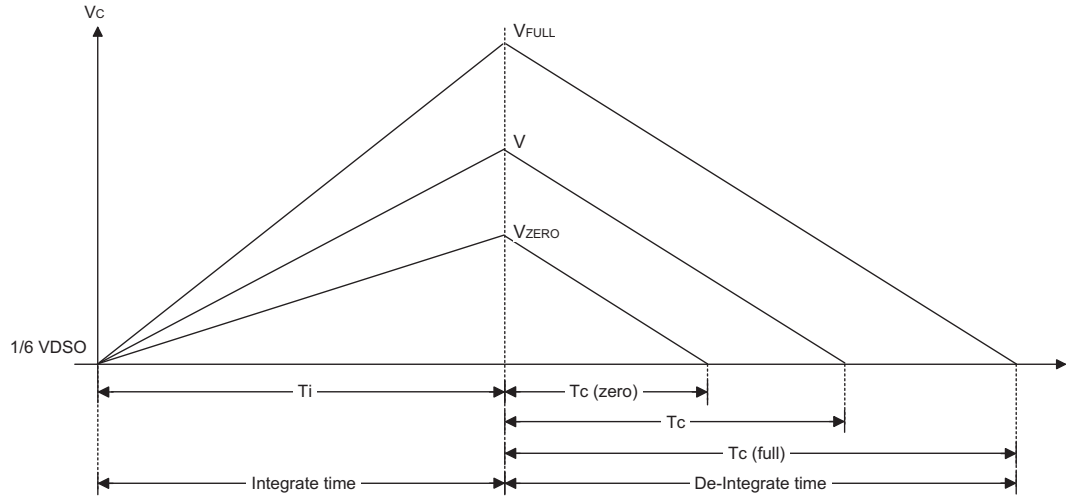
The Integrator integrates the output voltage increase or decrease and is controlled by the ″Switch Circuit″ - refer to the block diagram. The integration and de-integration curves are illustrated by the following.

The ″comparator″ will switch the state from high to low when V_C, which is the DSCC pin voltage,drops to less than 1/6 VDSO.

In general applications, the application program will switch the ADC to the charging mode for a fixed time called Ti, which is the integrating time. It will then switch to the dis-charging mode and wait for V_c to drop to less than 1/6 VDSO. At this point the comparator will change state and store the time taken, T_C, which is the de-integrating time. The following formula 1 can then be used to calculate the input voltage V_A.

formula 1: $V_A = (1/3) \times VDSO \times (2 - Tc/Ti)$.
(Based on ADRR0=0)

In user applications, it is required to choose the correct value of R_{DS} and C_{DS} to determine the Ti value, to allow the V_C value to operate between 5/6 VDSO and 1/6 VDSO. V_{FULL} cannot be greater than 5/6 VDSO and V_{ZERO} cannot be less than 1/6 VDSO.



Bit No.	Label	Function
0	ADPWREN	Dual slope block (including input OP) power on/off switching. 0: disable Power 1: Power source comes from the regulator.
1~2	ADDISCH0~ ADDISCH1	Defines the ADC discharge/charge. (ADDISCH1:0) 00: reserved 01: charging (Integrator input connect to buffer output) 10: discharging (Integrator input connect to VDSO) 11: reserved
3	ADCMPO	Dual Slope ADC - last stage comparator output. Read only bit, write data instructions will be ignored. During the discharging state, when the integrator output is less than the reference voltage, the ADCMPO will change from high to low.
4~5	ADINTM0~ ADINTM1	ADC integrator interrupt mode definition. These two bit define the ADCMPO data interrupt trigger mode: (ADINTM1:0)= 00: no interrupt 01: rising edge 10: falling edge 11: both edge
6	ADCCKEN	ADC OP chopper clock source on/off switching. 0: disable 1: enable (clock value is defined by ADCD register)
7	—	Unimplemented, read as ″0″

ADCR (18H) Register

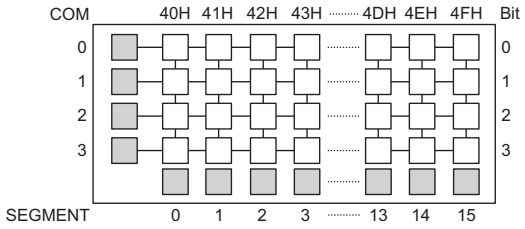
Bit No.	Label	Function
0 1 2	ADCD0 ADCD1 ADCD2	Define the chopper clock (ADCCKEN should be enable), the suggestion clock is around 10kHz. The chopper clock define : 0: clock= (f_{SYS}/32)/1 1: clock= (f_{SYS}/32)/2 2: clock= (f_{SYS}/32)/4 3: clock= (f_{SYS}/32)/8 4: clock= (f_{SYS}/32)/16 5: clock= (f_{SYS}/32)/32 6: clock= (f_{SYS}/32)/64 7: clock= (f_{SYS}/32)/128
3	ADIS	AD PGA input selection 0: from DCHOP pin 1: from TH/LB pin
4	ADRR0	ADC integrator and comparator reference voltage selection 0: (VINT, VCMP) = (4/6 VDSO, 1/6 VDSO) 1: (VINT, VCMP) = (4.4/6 VDSO, 1/6 VDSO)
5~6	—	Unimplemented, read as ″0″
7	PGAG	ADC PGA gain selection 0: gain = 2 1: gain = 4

ADCD (1AH) Register

LCD Display Memory

The device provides an area of embedded data memory for the LCD display. This area is located at 40H to 4FH in Bank 1 of the Data Memory. The bank pointer BP enables either the General Purpose Data Memory or LCD Memory to be chosen. When BP is set to ″1″, any data written into location range 40H~4FH will affect the LCD display. When the BP is cleared to ″0″, any data written into 40H~4FH will access the general purpose data memory. The LCD display memory can be read and written to only indirectly 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.

The LCD clock is driven by the f_{SUB} clock, which then passes through a divider, the division ratio of which is selected by the LCD clock selection bits LCDCK1 and LCDCK0 in the CTRL0 register to provide a LCD clock frequency of f_{SUB}/3, f_{SUB}/4 or f_{SUB}/8. The LCD clock source fSUB can be derived from the LIRC or LXT oscillator selected by the selection bit named FSUBS. Note that the f_{SUB} clock can be enabled or disabled in the power down mode by the f_{SUB} clock control bit FSUBC in the CTRL0 register.



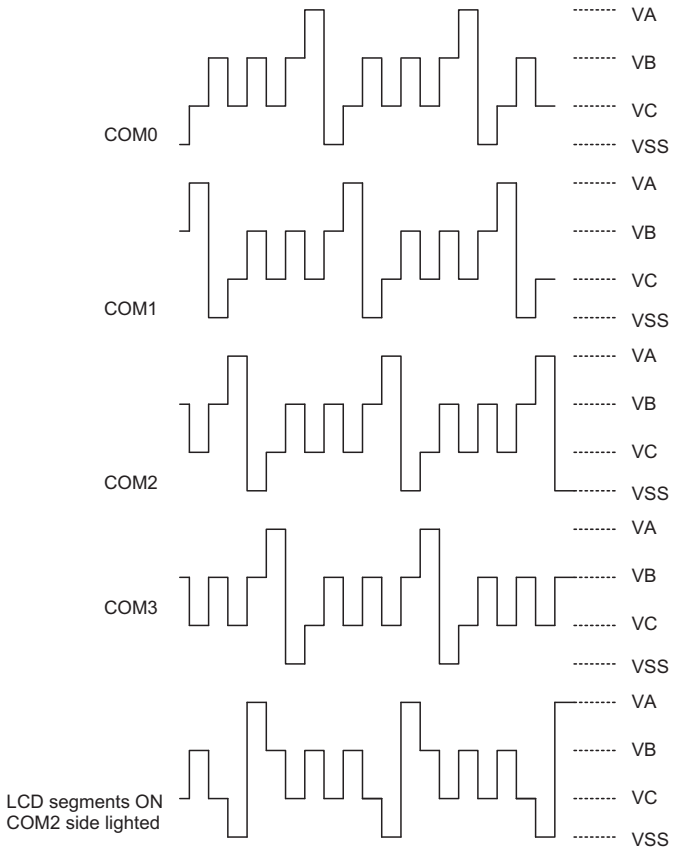
Display Memory

LCD Driver Output

The output structure of the device LCD driver can be 16×4. The LCD driver bias type is R type only. The LCD driver has a fixed 1/3 value.

Low Voltage Reset Function

There is a low voltage reset, LVR, circuit implemented in the microcontroller. The LVR functions can be enabled or disabled by the LVR function configuration option.



Note: 1/4 duty, 1/3 bias, R type: "VA" VLCD, "VB" 2/3 VLCD, "VC" 1/3 VLCD

LCD Driver Output (1/4 Duty)

Bit No.	Label	Function
0	LCDS0	Select SEG 0 or IO. 0/1 : IO/SEG0
1	LCDS1	Select SEG 1 or IO. 0/1 : IO/SEG1
2	LCDS2	Select SEG 2 or IO. 0/1 : IO/SEG2
3	LCDS3	Select SEG 3 or IO. 0/1 : IO/SEG3
4~7	—	Unimplemented, read as ″0″

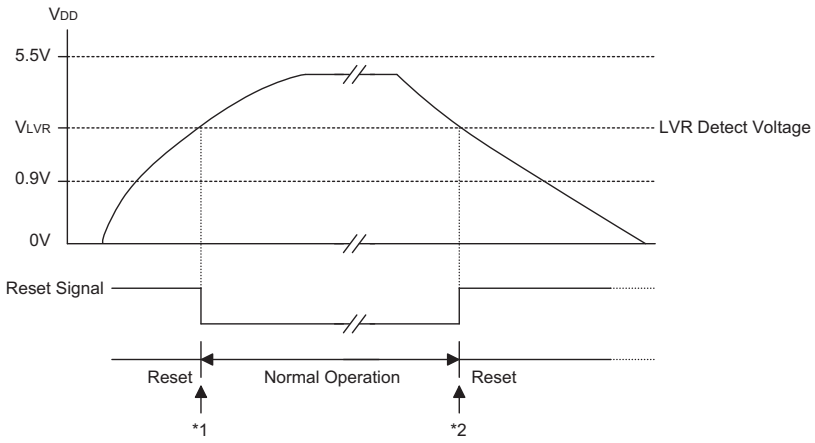
LCDOUT Register

The LVR has the same effect or function as the external RESB signal which performs a device reset. When in the Power Down Mode, the LVR function is disabled.

The microcontroller provides a low voltage reset circuit in order to monitor the supply voltage of the device. If the supply voltage of the device is within the range 0.9V~V_{LVR}, such as what might happen when changing a battery, the LVR will automatically reset the device internally.

The LVR includes the following specifications:

- The low voltage, which is specified as 0.9V~V_{LVR}, has to remain within this range for a period of time greater than 1ms. If the low voltage state does not exceed 1ms, the LVR will ignore it will not perform a reset function.
- The LVR has an "OR" function with the external $\overline{RES}$ signal to perform a chip reset.



Low Voltage Reset

Note: *1: To make sure that the system oscillator has stabilized, the SST provides an extra delay of 1024 system clock pulses before entering the normal operation.

*2: Since a low voltage state has to be maintained in its original state for over 1ms, therefore after 1ms delay, the device enters the reset mode.

Operation Mode

The device has two operational modes. The system clock may come from external RC (ERC), external crystal (HXT) or internal RC (HIRC) oscillator, and whose operational modes can be either Normal Mode or Power down mode. When in the Power down mode, the clocks in this device are all enabled or disabled using software.

HALT Instruction	Mode	System Oscillator	FSUBC	f_{SUB} Clock	RTCEN	RTC Oscillator (OSC3/OSC4)
Not executed	Normal	On	x	Enable	x	On
Executed	Power Down	On (OSCON=1) Off (OSCON=0)	0	Disable	1	On
	Power Down	On (OSCON=1) Off (OSCON=0)	1	Enable	1	On
	Power Down	On (OSCON=1) Off (OSCON=0)	0	Disable	0	Off
	Power Down	On (OSCON=1) Off (OSCON=0)	1	Enable	0	Off

Note: The WDTOSC0:1 register should be set to enable, otherwise, the Int.RCOSC will always be disabled. Refer to the WDT section for the WDTOSC setup details.

Bit No.	Label	Function
0	QOSC	32.765kHz crystal oscillator quick start-up control 0: quick start-up 1: low-power
1	FSUBS	f_{SUB} Clock source selection 0: LIRC oscillator 1: LXT oscillator
2	FSUBC	f_{SUB} Power down mode clock control 0: disabled 1: enabled
3 4	LCDCK0 LCDCK1	To select the LCD driver clock: 00: LCD clock = f_{SUB}/3 01: LCD clock = f_{SUB}/4 10: LCD clock = f_{SUB}/8 11: Reserved
5	LFS	Low Frequency clock source f_L selection 0: LIRC oscillator 1: LXT oscillator
6 7	TCKS0 TCKS1	Timer/Event Counter 2 internal source selection 00: f_L (Low frequency clock) 01: f_{REF} (Reference frequency clock generated from Touch Key module) 10: f_{SEN} (Sensor frequency clock generated from Touch Key module) 11: f_{TMCK} (Gated Sensor frequency clock generated from Touch Key module) If the Touch Key module is disabled, the TCKS1 and TCKS0 bits are always set to 00 and can not be written to.

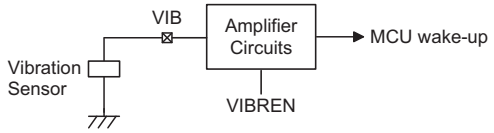
CTRL0 Register

Bit No.	Label	Function
0	RTCEN	32.768kHz oscillator (LXT) control in Power down mode 0: disabled 1: enabled
1	BZCS	Buzzer clock source selection 0: from Timer/Event Counter 0 1: from Timer/Event Counter 1
2~7	—	Unimplemented, read as ″0″

CTRL1 Register

Vibration Sensor Amplifier

The device contains a Vibration Sensor Amplifier to amplify the small electrical signals generated from vibration sensors. When the sensor is connected to the vibration input pin, VIB, and a small signal resulting from a vibration detection is generated on the VIB pin, the internal amplifier will amplify the low amplitude signal which will then be used as a wake-up source when the device is in the Power down mode. The Vibration Sensor Amplifier can be enabled or disable by the control bit, VIBREN, in the VIBRC register for power saving considerations.



Bit No.	Label	Function
0	VIBREN	Vibration Sensor Amplifier control 0: disabled 1: enabled
1~7	—	Unimplemented, read as ″0″

VIBRC Register

Touch Key Module

The device contains a Touch Key Module with four touch key inputs which can detect human body contact using external touch pads. The Touch Key Module includes four touch key inputs, a reference key input, a reference oscillator, a sensor oscillator and a 12-bit Counter as shown in the block diagram.

Touch Key Structure

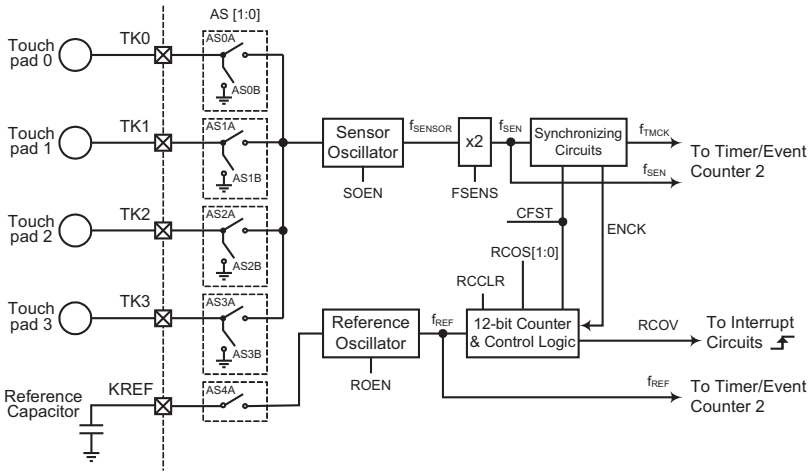
The overall functions of the Touch Key Module are controlled by the Capacitor to Frequency Control Registers CFCR0 and CFCR1 and the Analog Channel Selection register ANCS0.

The reference key named KREF is the reference oscillator input for touch key function. When the reference key input is connected to an external capacitor together with the internal resistor and logic circuits, it forms a reference oscillator which is used to provide a reference frequency for the 12-bit reference counter. The 12-bit reference counter is used to provide a reference period selected by the reference counter overflow period selection bits RCOS1 and RCOS0 in the CFCR0 register. After the reference counter reaches the selected time-out period, the counter will stop counting and generate an interrupt signal. The reference key input KREF is also pin-shared with an I/O pin and controlled by the KREFS bit in the CFCR1 register.

The four touch key inputs named TK0~TK3 are pin-shared with I/O pins and configured by the analog channel selection bits ACS0~ACS3 in the ANCS0 register to determine whether these pins are used as I/O pins or touch key analog inputs. When the four touch keys, TK0~TK3, are configured to function as touch key inputs and connected to external touch pads and combined with the internal resistor and logic circuits, it forms a touch key sensor oscillator. The sensor oscillator will generate a specific frequency different from the reference frequency when the touch key is influenced by human body contact.

Touch Key Operation

Before the Touch Key Module starts to function, both the reference and sensor oscillators should be enabled and the touch key inputs and analog switches should be properly setup. It is important to know that the 12-bit reference counter should first be cleared by setting the reference counter clear control bit RCCLR from ″0″ to ″1″. When the touch key module start bit CFST is set from ″0″ to ″1″, the 12-bit Counter will start to count with the synchronized reference clock f_{REF} to provide a refer-



Touch Key Module Block Diagram

ence period. The reference period can be selected by the Reference Counter Overflow Selection bits RCOS1 and RCOS0 with a period ranged from $512/f_{REF}$ to $4096/f_{REF}$. As the 12-bit reference counter starts to count, Timer/Event Counter 2 will also start to count using its synchronized sensor clock f_{TMCK}. As the selected reference period time has elapsed, the reference counter will stop counting. At the same time, the module will send an interrupt signal to the MCU interrupt circuits and the synchronized sensor clock f_{TMCK} will also be blocked. Since the f_{TMCK} clock is blocked, Timer/Event Counter 2 will also be stopped and the count value during the reference period will be stored in the timer registers, TMR2H and TMR2L. The value obtained when the key is not touched is the untouched reference value of the corresponding key and should be stored in the RAM Data Memory. When the key is touched, the count value stored in the TMR2H and TMR2L registers will be obviously different with the untouched reference value when the key is not influenced by human body contact. By sequentially switching the touch keys and counting, users can know which keys have been touched by comparing the count value with their untouched reference value.

For optimal touch switch operation it is recommended that both the reference and sensor oscillators have a frequency range of 100kHz to 1MHz. It is also recommended that the reference and sensor frequencies are as close to each other as possible for optimal operation. After the external touch key size and layout are defined, their related capacitances will then determine the sensor oscillator frequency. After this frequency is measured the reference oscillator can be adjusted to have a frequency as close as possible to this by adjusting its external reference capacitor value. A simple application program can be written by the user to measure these two internal frequencies.

Touch Key Interrupt

When the 12-bit reference counter overflows, the reference counter overflow flag will be set from ″0″ to ″1″ and a touch key interrupt signal will occur to get the attention of the microcontroller. When a Touch Key interrupt occurs, if the corresponding interrupt in the MCU is enabled and the stack is not full, the program will jump to the corresponding interrupt vector where it can be serviced before returning to the main program. If the related interrupt enable bits are not set, then the interrupt signal will only be a wake-up source and no interrupt will be serviced.

Touch Key Registers

The Capacitor to Frequency Control Register 0 named CFCR0 includes the sensor/reference oscillators enable control, the reference period selection, the touch key analog switch selection and the reference counter clear control bits. Note that when a specific touch key is selected, the other keys will be switched to ground automatically.

Bit No.	Label	Function
0 1	AS0 AS1	Touch Key Analog Switch selection * 00: Touch Key 0 is selected, others switch to the ground. 01: Touch Key 1 is selected, others switch to the ground. 10: Touch Key 2 is selected, others switch to the ground. 11: Touch Key 3 is selected, others switch to the ground.
2	FSENS	f_{SEN} clock source selection 0: $f_{SEN} = f_{SENSOR}$ 1: $f_{SEN} = f_{SENSOR} \times 2$
3 4	RCOS0 RCOS1	Reference Counter Overflow Period Selection 00: $512/f_{REF}$ 01: $1024/f_{REF}$ 10: $2048/f_{REF}$ 11: 4096/f
5	RCCLR	12-bit Reference Counter Clear control 0→1: Clear the counter Others: counter unchanged After this bit is set from ″0″ to ″1″ to clear the counter, users should then reset this bit to ″0″ in preparation for the next clear operation. It is recommended to clear the reference counter first before the touch key module is used.
6	SOEN	Sensor Oscillator enable control 0: disabled 1: enabled
7	ROEN	Reference Oscillator enable control 0: disabled 1: enabled

CFCR0 Register

*: Truth Table of the Touch Key Analog Switch selection when pins are selected as touch key inputs.

AS1	AS0	AS0A	AS0B	AS1A	AS1B	AS2A	AS2B	AS3A	AS3B
0	0	Short	Open	Open	Short	Open	Short	Open	Short
0	1	Open	Short	Short	Open	Open	Short	Open	Short
1	0	Open	Short	Open	Short	Short	Open	Open	Short
1	1	Open	Short	Open	Short	Open	Short	Short	Open

Note: If the TKx pin is selected as an I/O pin, then the analog switches ASxA and ASxB of the TKx pin are both kept open. If the TKx pin is selected as a touch key input, the I/O input, pull-high resistor and output functions are all disabled. The x stands for the pin number from ″0″ to ″3″.

The Capacitor to Frequency Control Register 1 named CFCR1 includes the touch key module start bit, the 12-bit reference counter overflow flag and the reference key function selection bit. Note that when the reference key is selected as an I/O pin, the analog switch AS4A of the reference key shown in the block diagram is kept open.

Bit No.	Label	Function
0	CFST	Touch Key Module Start bit. 0→1: enable the f_{TMCK} output clock When this bit is set from ″0″ to ″1″, it will enable the synchronizing circuits, enable the f_{TMCK} output clock, set the RCOV flag to ″1″ and the 12-bit reference counter will satrt to count. After this bit is set from ″0″ to ″1″ to enable the f_{TMCK} clock, users should remember to reset this bit to 0 before setting this bit to ″1″ again. Note that when the reference counter is counting, it has no operation if this bit is re-triggered.
1	RCOV	12-bit Reference Counter Overflow flag 0: not overflow 1: overflow occurs This bit is read only and set/cleared by hardware automatically. It is set to ″1″ when the 12-bit reference counter overflows and cleared to ″0″ when the touch key module start bit CFST is set from ″0″ to ″1″.
2	KREFS	Reference Key input function selection * 0: I/O pin 1: KREF pin
3~7	—	Unimplemented, read as ″0″

Note: *: If the reference key is selected as an I/O pin, then the KREF pin analog switch AS4A is kept open. If the KREF pin is selected as a reference key input, the I/O input, pull-high resistor and output functions are both disabled.

CFCR1 Register

The Analog Channel Selection register named ANCS0 controls the touch key input function selections. Note that when the touch key TKx is selected to be used as an I/O pin, the analog switches ASxA and ASxB of the related TKx pin, shown in the block diagram, are always kept open where x stands the pin number from 0~3.

Bit No.	Label	Function
0	ACS0	Touch Key input 0 function selection 0: I/O pin 1: TK0 pin
1	ACS1	Touch Key input 1 function selection 0: I/O pin 1: TK1 pin
2	ACS2	Touch Key input 2 function selection 0: I/O pin 1: TK2 pin
3	ACS3	Touch Key input 3 function selection 0: I/O pin 1: TK3 pin
4~7	—	Unimplemented, read as ″0″

ANCS0 Register

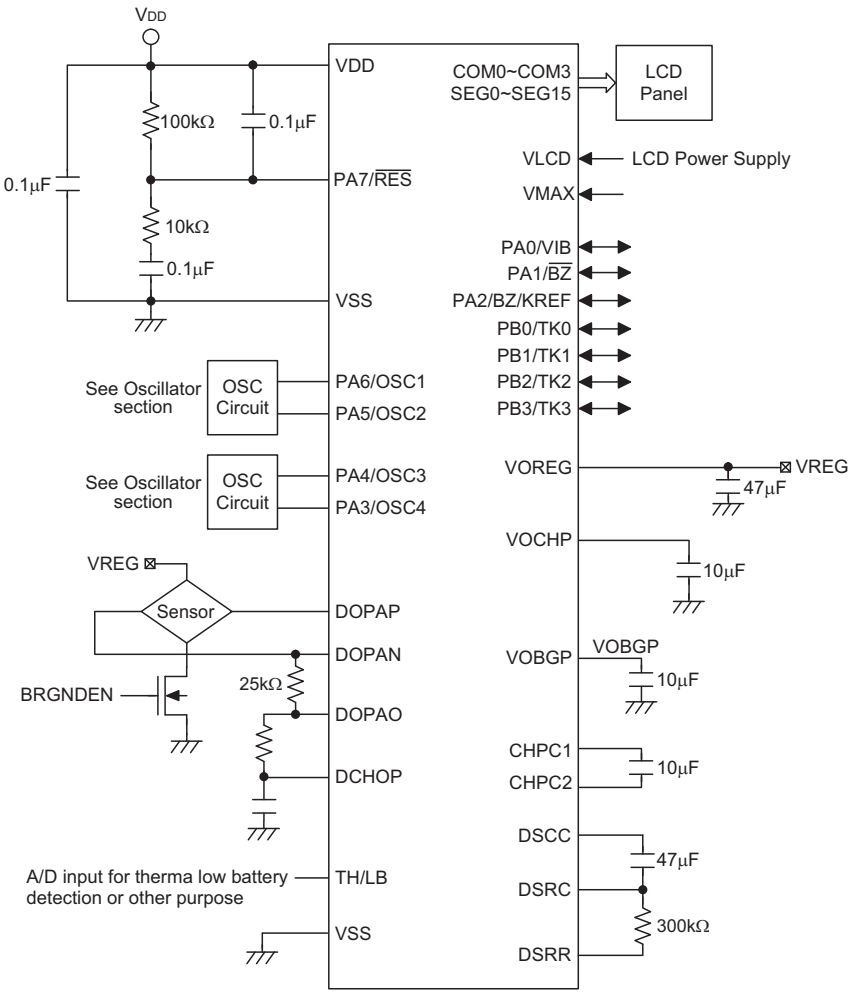
Configuration Options

The following shows the options in the device. All these options should be defined in order to ensure proper functioning system.

No.	Options
I/O Options	
1	Port A wake-up - bit option 1. Enable 2. Disable
2	PA0~PA6 pull-high - bit option 1. Enable 2. Disable
3	Port B pull-high - bit option 1. Enable 2. Disable
4	PA7 Function select 1. I/O 2. $\overline{\text{RES}}$
LCD Options	
5	LCD function in Power down mode 1. Enable 2. Disable
6	R type drive current select 1. 50μA 2. 100μA

No.	Options
Oscillator Options	
7	System oscillator select - f_{SYS} 1. Internal RC 2. External RC 3. External XTAL
8	Internal RC oscillator frequency select 1. 4MHz 2. 8MHz 3. 12MHz
9	External 32KHz oscillator select 1. I/O 2. 32.768kHz external crystal
10	System oscillator SST period selection 1. 1024 clocks 2. 2 clocks
Interrupt Options	
11	INT trigger edge select 1. Disable 2. Falling edge 3. Rising edge 4. Double edge
Watchdog Options	
12	WDT function 1. Enable 2. Disable
13	WDT clock selection - f_S: 1. Internal 12kHz RC oscillator - LIRC 2. $f_{SYS}/4$ 3. 32.768kHz oscillator - LXT
14	CLRWDT instruction select 1. 1 instruction 2. 2 instructions
Buzzer Options	
15	I/O or BZ function select 1. PA1/PA2 2. BZ/PA2 3. BZ/$\overline{BZ}$
LVR Options	
16	LVR function select 1. Disable 2. Enable
17	LVR voltage select 1. 2.1V 2. 3.15V 3. 4.2V

Application Circuits



HT46R73D-3

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 microcontrollers, 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 Operations

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 where 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

Mnemonic	Description	Cycles	Flag Affected
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
Data Move			
MOV A,[m]	Move Data Memory to ACC	1	None
MOV [m],A	Move ACC to Data Memory	1Note	None
MOV A,x	Move immediate data to ACC	1	None
Bit Operation			
CLR [m].i	Clear bit of Data Memory	1Note	None
SET [m].i	Set bit of Data Memory	1Note	None
Branch			
JMP addr	Jump unconditionally	2	None
SZ [m]	Skip if Data Memory is zero	1Note	None
SZA [m]	Skip if Data Memory is zero with data movement to ACC	1note	None
SZ [m].i	Skip if bit i of Data Memory is zero	1Note	None
SNZ [m].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			
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 bitwise 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 $[m] \leftarrow \overline{[m]}$

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

Affected flag(s) C

DEC [m] Decrement Data Memory

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

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

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 $ACC \leftarrow [m] - 1$

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 $TO \leftarrow 0$
$PDF \leftarrow 1$

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 enable master (global) interrupt bit (bit 0; register INTC). 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 $ACC \leftarrow ACC - [m] - \overline{C}$

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 $[m] \leftarrow ACC - [m] - \overline{C}$

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 $[m] \leftarrow [m] - 1$
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 $ACC \leftarrow [m] - 1$
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 $[m] \leftarrow FFH$

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 $[m].i \leftarrow 1$

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 $\neq$ 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

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

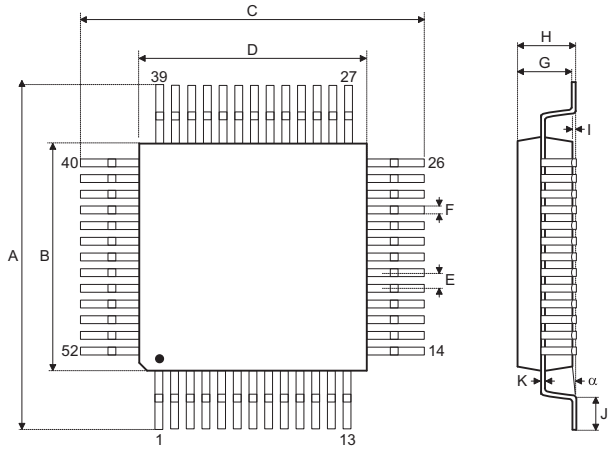
XOR A,[m]	Logical XOR Data Memory to ACC
Description	Data in the Accumulator and the specified Data Memory perform a bitwise logical XOR operation. The result is stored in the Accumulator.
Operation	ACC ← ACC ″XOR″ [m]
Affected flag(s)	Z

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

52-pin QFP (14mm×14mm) Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	0.681	—	0.689
B	0.547	—	0.555
C	0.681	—	0.689
D	0.547	—	0.555
E	—	0.039	—
F	—	0.016	—
G	0.098	—	0.122
H	—	—	0.134
I	—	0.004	—
J	0.029	—	0.041
K	0.004	—	0.008
L	—	0.004	—
α	0°	—	7°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	17.30	—	17.50
B	13.90	—	14.10
C	17.30	—	17.50
D	13.90	—	14.10
E	—	1.00	—
F	—	0.40	—
G	2.50	—	3.10
H	—	—	3.40
I	—	0.10	—
J	0.73	—	1.03
K	0.10	—	0.20
α	0°	—	7°

Holtek Semiconductor Inc. (Headquarters)
No.3, Creation Rd. II, Science Park, Hsinchu, Taiwan
Tel: 886-3-563-1999
Fax: 886-3-563-1189
http://www.holtek.com.tw

Holtek Semiconductor Inc. (Taipei Sales Office)
4F-2, No. 3-2, YuanQu St., Nankang Software Park, Taipei 115, Taiwan
Tel: 886-2-2655-7070
Fax: 886-2-2655-7373
Fax: 886-2-2655-7383 (International sales hotline)

Holtek Semiconductor Inc. (Shenzhen Sales Office)
5F, Unit A, Productivity Building, No.5 Gaoxin M 2nd Road, Nanshan District, Shenzhen, China 518057
Tel: 86-755-8616-9908, 86-755-8616-9308
Fax: 86-755-8616-9722

Holtek Semiconductor (USA), Inc. (North America Sales Office)
46729 Fremont Blvd., Fremont, CA 94538, USA
Tel: 1-510-252-9880
Fax: 1-510-252-9885
http://www.holtek.com