HT46R71D
A/D with LCD Type 8-Bit MCU

Technical Document

- Tools Information
- FAQs
- Application Note
  - HA0003E Communicating between the HT48 & HT46 Series MCUs and the HT93LC46 EEPROM
  - HA0004E HT48 & HT46 MCU UART Software Implementation Method
  - HA0005E Controlling the I2C bus with the HT48 & HT46 MCU Series
  - HA0007E Using the MCU Look Up Table Instructions
  - HA0049E Read and Write Control of the HT1380

Features

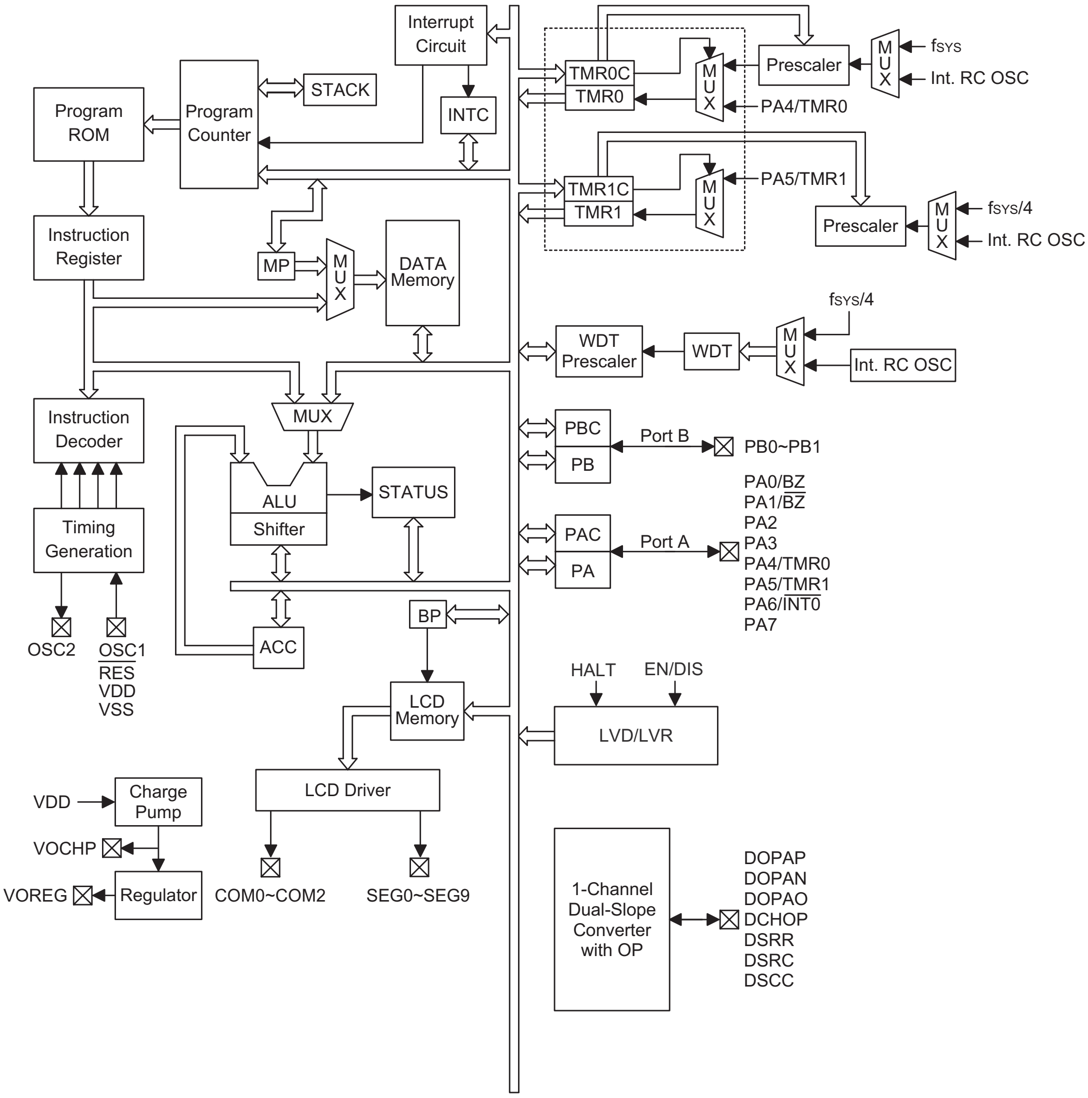
- Operating voltage:
  f_{SYS}=4MHz: 2.2V~5.5V
- 10 bidirectional I/O lines and two ADC input
- One external interrupt input shard with an I/O lines
- One 8-bit and one 16-bit programmable timer/event counter with overflow interrupt a 7-stage pre-scalar
- LCD driver with 10×3 segments
- 2K×14 program memory
- 32×8 data memory RAM
- Single differential input channel dual slope Analog to Digital Converter with Operational Amplifier.
- Watchdog Timer
- Buzzer output
- Internal 12kHz RC oscillator
- RC 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
- 14-bit table read instruction
- Up to 1μs instruction cycle with 4MHz system clock
- 63 powerful instructions
- All instructions in 1 or 2 machine cycles
- Low voltage reset/detector function
- 48-pin SSOP package

General Description

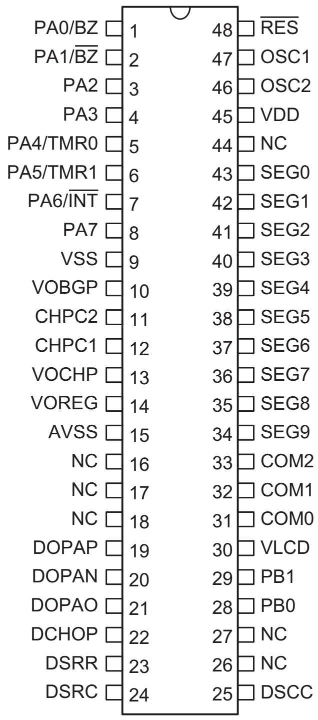
The HT46R71D is an 8-bit, high performance, RISC architecture microcontroller device specifically designed for A/D product applications that interface directly to analog signals and which require an LCD Interface.

The advantages of low power consumption, I/O flexibility, timer functions, oscillator options, multi-channel A/D Converter, LCD display, HALT and wake-up functions, in addition to a flexible and configurable LCD interface enhance the versatility of these devices to control a wide range of applications requiring analog signal processing and LCD interfacing, such as electronic metering, environmental monitoring, handheld measurement tools, motor driving, etc., for both industrial and home appliance application areas.

Block Diagram



Pin Assignment



HT46R71D
— 48 SSOP-A

Pin Description

Pin Name	I/O	Options	Description
PA0/BZ PA1/$\overline{BZ}$ PA2 PA3 PA4/TMR0 PA5/TMR1 PA6/$\overline{INT}$ PA7	I/O	Wake-up Pull-high Buzzer	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 the CMOS output or Schmitt trigger input. Configuration options determine which pin on this port has pull-high resistors. The BZ, $\overline{BZ}$, TMR0, TMR1 and $\overline{INT}$ are pin-shared with PA0, PA1, PA4, PA5 and PA6 respectively.
PB0~PB1	I/O	Pull-high	Bidirectional 2-bit input/output port. Software instructions determine if the pin is a CMOS output or Schmitt trigger input. Configuration options determine which pin on this port have pull-high resistors.
VLCD	I	—	LCD power supply
COM0~COM2	O	1/2 or 1/3 Duty	COM0~COM2 are the common outputs for the LCD panel plate.
SEG0~SEG9	O	Segment Output	LCD driver outputs for the LCD panel segments.
VOBGP	AO	—	Bandgap voltage output pin. (for external use)
VOREG	O	—	Regulator output 3.3V
VOCHP	O	—	Charge pump output (a capacitor is required to be connected)
CHPC1	—	—	Charge pump capacitor, positive
CHPC2	—	—	Charge pump capacitor, negative

Pin Name	I/O	Options	Description
DOPAN, DOPAP, DOPAO, DCHOP	AI/AO	—	Dual Slope converter pre-stage OPA related pins. DOPAN is OPA Negative input pin, DOPAP is OPA Positive input pin, DOPAO is OPA output pin and the DCHOP is OPA Chopper pins.
DSRR, DSRC, DSCC	AI/AO	—	Dual slope AD 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.
OSC1 OSC2	I O	External RC	OSC1, OSC2 are connected to an external RC network for the internal system clock. For external RC system clock operation, OSC2 is an output pin for 1/4 system clock.
$\overline{RES}$	I	—	Schmitt trigger reset input, active low
VDD	—	—	Positive power supply
VSS	—	—	Negative power supply, ground
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

Storage Temperature ....–50°C to 125°C

Operating Temperature...........................–40°C to 85°C

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
I_{DD1}	Operating Current (External RC OSC)	3V	No load, ADC off, f_{SYS}=2MHz	—	0.5	1	mA
		5V		—	1.5	3	mA
I_{DD2}	Operating Current (External RC OSC)	3V	No load, ADC off, f_{SYS}=4MHz	—	0.8	1.5	mA
		5V		—	2.5	4	mA
I_{DD3}	Operating Current (External RC OSC)	5V	No load, ADC on, f_{SYS}=4MHz, ADCCLK=125kHz	—	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
I_{STB4}	Standby Current (WDT Disable Internal RC 12kHz OSC ON)	3V	No load, system HALT, LCD on at HALT, 1/2 bias, VLCD=VDD	—	17	30	μA
		5V		—	34	60	μA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{STB5}	Standby Current (WDT Disable Internal RC 12kHz OSC ON)	3V	No load, system HALT LCD on at HALT, 1/3 bias, VLCD=VDD	—	13	25	μA
		5V		—	28	50	μA
V_{IL1}	Input Low Voltage for I/O Ports, TMR and $\overline{INT}$	—	—	0	—	$0.3V_{DD}$	V
V_{IH1}	Input High Voltage for I/O Ports, TMR and $\overline{INT}$	—	—	$0.7V_{DD}$	—	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_{LVR}	Low Voltage Reset	—	—	2	2.1	2.2	V
V_{LVD}	Low Voltage Detector	—	—	2.2	2.3	2.4	V
I_{OL1}	I/O Port Segment Logic Output Sink Current	3V	$V_{OL}=0.1V_{DD}$	4	8	—	mA
		5V		10	20	—	mA
I_{OH1}	I/O Port Segment Logic Output Source Current	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
R_{PH}	Pull-high Resistance of I/O Ports and $\overline{INT}$	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ
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_{RFGO}	Reference Generator Output	—	@3.3V	1.45	1.5	1.55	V
V_{RFGTC}	Reference Generator Temperature Coefficient	—	@3.3V	—	50	—	Ppm/C
V_{ADOFF}	Input Offset Range	—	—	—	500	800	μV

A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS}	System Clock	—	2.2V~5.5V	400	—	4000	kHz
f_{INRC}	Internal RC OSC	3V	—	—	12	—	kHz
		5V		—	15	—	kHz
f_{TIMER}	Timer I/P Frequency (TMR0/TMR1)	—	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	—	Power-up or wake-up from HALT	—	1024	—	t_{SYS}
t_{INT}	Interrupt Pulse Width	—	—	1	—	—	μs

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

Functional Description

Execution Flow

The system clock is derived from an 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 11 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 2048 addresses.

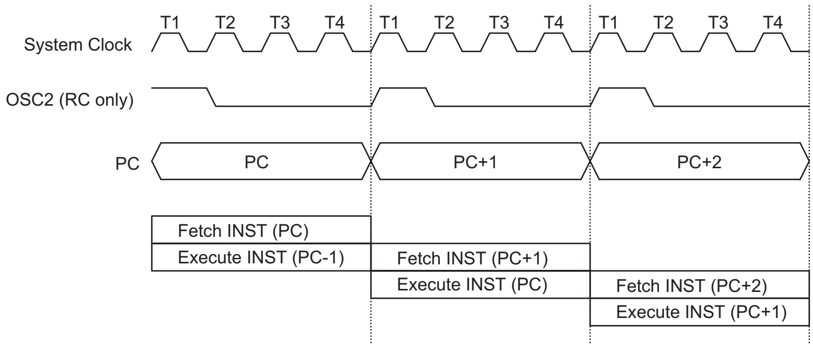
After accessing a program memory word to fetch an instruction code, the value of the PC is incremented by 1. The PC then points to the memory word containing the next instruction code.

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.

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



Execution Flow

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

Program Counter

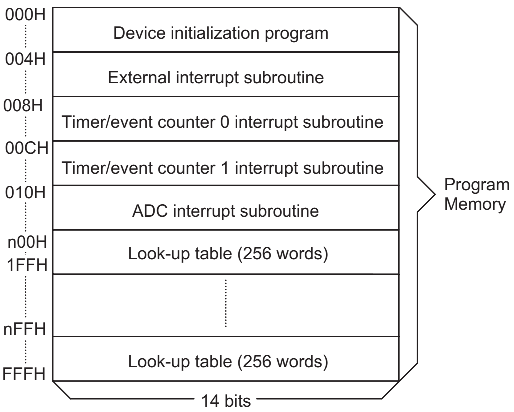
Note: *10~*0: Program counter bits
#10~#0: Instruction code bits
S10~S0: Stack register bits
@7~@0: PCL bits

Program Memory – EPROM

The program memory (EPROM) is used to store the program instructions which are to be executed. It also contains data, table, and interrupt entries, and is organized into 2048×14 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 $\overline{INT}$ 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.



Note: n ranges from 1 to 6

Program Memory

- Location 010H
  Location 010H 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.
- 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.

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).

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

Table Location

Note: *10~*0: Table location bits
@7~@0: Table pointer bits
P10~P8: Current program counter bits

Data Memory – RAM

The data memory (RAM) is designed with 57×8 bits, and is divided into two functional groups, namely; special function registers 25×8 bit and general purpose data memory, 32×8 bit most of which are readable/writable, although some are read only. The special function register are overlapped in any banks.

Of the two types of functional groups, the special function registers consist of an Indirect addressing register 0 (00H), a Memory pointer register 0 (MP0;01H), an Indirect addressing register 1 (02H), a Memory pointer register 1 (MP1;03H), a Bank pointer (BP;04H), an Accumulator (ACC;05H), a Program counter lower-order byte register (PCL;06H), a Table pointer (TBLP;07H), a Table higher-order byte register (TBLH;08H), a Status register (STATUS;0AH), an Interrupt control register 0 (INTC0;0BH), a Timer/Event Counter 0 (TMR0; 0DH), a Timer/Event Counter 0 control register (TMR0C;0EH), a Timer/Event Counter 1 (TMR1H:0FH;TMR1L:10H), a Timer/Event Counter 1 control register (TMR1C;11H), I/O registers (PA;12H, PB;14H) and I/O control registers (PAC;13H, PBC;15H), an ADC control register (ADCR;18H), an ADC chopper divider register (ADCD;1AH), an Interrupt control register 1 (INTC1;1EH) and Charge Pump & Regulator Control Register (CHPRC;1FH).

00H	Indirect Addressing Register 0	Special Purpose Data Memory
01H	MP0	
02H	Indirect Addressing Register 1	
03H	MP1	
04H	BP	
05H	ACC	
06H	PCL	
07H	TBLP	
08H	TBLH	
09H	MODE	
0AH	STATUS	
0BH	INTC0	
0CH		
0DH	TMR0	
0EH	TMR0C	
0FH	TMR1H	
10H	TMR1L	
11H	TMR1C	
12H	PA	
13H	PAC	
14H	PB	
15H	PBC	
16H		
17H		
18H	ADCR	
19H	Reserved	
1AH	ADCD	
1BH		
1CH		
1DH		
1EH	INTC1	
1FH	CHPRC	
20H ~ 3FH	General Purpose Data Memory (32 Bytes)	

: Unused
Read as "00"

RAM Mapping

The remaining space before the 20H is reserved for future expanded usage and reading these locations will get ″00H″. The general purpose data memory, addressed from 20H to 3FH , 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 ″SET [m].i″ and ″CLR [m].i″. They are also indirectly accessible through memory pointer registers (MP0;01H or MP1:03H).

After first setting up BP to the value of ″01H″ to access Bank 1 , these banks 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 20H~3FH 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.

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, MP1) are 7-bit registers.

The function of data movement between two indirect addressing registers is not supported. The memory pointer registers, MP0 and MP1, are both 7-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, two internal timer/event counter interrupts and the ADC 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 these interrupts can support a wake-up function. As an interrupt is serviced, a control transfer occurs by pushing the contents of the program counter onto the stack followed by a branch to a subroutine at the specified location in the ROM. Only the contents of the program counter is pushed onto the stack. If the contents of the register or of the status register (STATUS) is altered by the interrupt service program which corrupts the desired control sequence, the contents should be saved in advance.

External interrupts is triggered by an edge transition of $\overline{INT}$ (Configuration option: high to low, low to high, both low to high and high to low), and the related interrupt request flag (EIF0; 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 (EIF0) and EMI bits are all cleared to disable other maskable interrupts.

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

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 0CH occurs. The related interrupt request flag (T0F) is reset, and the EMI bit is cleared to disable other maskable interrupts. Timer/Event Counter 1 is operated in the same manner but its related interrupt request flag is T1F (bit 6 of INTC0) and its subroutine call location is 0CH.

The A/D Converter interrupt is initialized by setting the A/D Converter clock interrupt request flag (ADF; bit 4 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 10H 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.

Interrupt Source	Priority	Vector
External interrupt 0	1	04H
Timer/Event Counter 0 overflow	2	08H
Timer/Event Counter 1 overflow	3	0CH
ADC interrupt	4	10H

The Timer/Event Counter 0 interrupt request flag (T0F), external interrupt 1 request flag (EIF1), external interrupt 0 request flag (EIF0), enable Timer/Event Counter 0 interrupt bit (ET0I), enable external interrupt 1 bit (EEI1), enable external interrupt 0 bit (EEI0) and enable master interrupt bit (EMI) make up of the Interrupt Control register 0 (INTC0) which is located at 0BH in the RAM. The ADC interrupt request flag (ADF). Timer/Event Counter 1 interrupt request flag (T1F), enable ADC interrupt bit (ADI), enable Timer/Event Counter 1 interrupt bit (ET1I) on the other hand, constitute the Interrupt Control register 1 (INTC1) which is located at 1EH in the RAM. EMI, EEI0, EEI1, ET0I, ET1I and EADI are all used to control the enable/disable status of interrupts. These bits prevent the requested interrupt from being serviced. Once the interrupt request flags (ADF, T0F, T1F, EIF1, EIF0) 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.

Bit No.	Label	Function
0	EMI	Controls the master (global) interrupt (1=enabled; 0=disabled)
1	EEI0	Controls the external interrupt 0 (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	EIF0	External interrupt 0 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.

INTC 0 (0BH) Register

Bit No.	Label	Function
0	EADI	Controls the ADC interrupt (1=enabled; 0:disabled)
1~3, 5~7	—	Unused bit, read as ″0″
4	ADF	ADC request flag (1=active; 0=inactive)

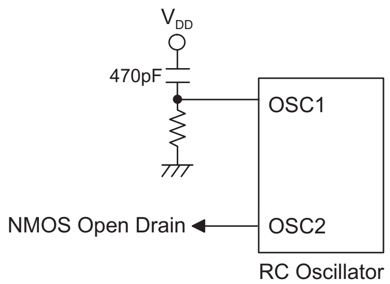
INTC 1 (1EH) Register

Oscillator Configuration

The device provides two oscillator circuits, an external RC oscillator and an internal RC 12kHz oscillator (Int.RCOSC). The external RC oscillator signal is used for the system clock while the Internal 12kHz RC oscillator is designated for timing purposes.

In the IDLE mode, the system oscillator will stop running, but if bit IRCC = 1,to enable the IRC clock source, the internal RC oscillator (Int.RCOSC) will continue to free run. In the HALT mode, if the IRC clock source is disabled, with bit IRCC=0, both the system oscillator and the internal RC oscillator will stop running. However, if the WDT is enabled, the internal RC oscillator will continuously free run. The system can be woken-up from either the IDLE or HALT mode by the occurrence of an interrupt, a high to low transition on any of the Port A pins, a WDT overflow or a timer overflow and request flag is set ($0 \rightarrow 1$). If an external RC oscillator is used, an external resistor between OSC1 and VSS is required to achieve oscillation, the value of which must be between 100kΩ to 2.4MΩ. The system clock, divided by 4, is available for external logic synchronization purposes on pin OSC2.

The Internal RC oscillator (Int.RCOSC) is a free running on-chip RC oscillator, requiring no external components. Even if the system enters the Power Down Mode, and the system clock is stopped, the internal RC oscillator continues to run with a period of approximately 65μs at 5V if either the WDT or IRC clock is enabled. The internal RC oscillator can be disabled by a configuration option and by clearing the IRCC bit to ″0″ to conserve power.



System Oscillator

Watchdog Timer – WDT

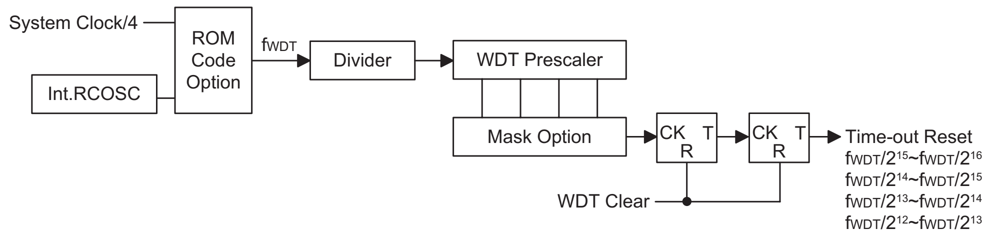
The WDT is implemented either using a dedicated internal RC oscillator (Int.RCOSC) or the instruction clock (system clock/4). The timer is designed to prevent a software malfunction or sequence from jumping to an unknown location with unpredictable results. The WDT can be disabled by a configuration option, however if the WDT is disabled, all executions related to the WDT will result in no operation.

Once the internal RC oscillator, which has a nominal period of 65μs, is selected, it is then divided by a value which ranges from 2^{12}~2^{15} the exact value of which is determined by a configuration option, to obtain the actual WDT time-out period. The minimum period of the WDT time-out period is about 300ms~600ms. This time-out period may vary with temperature, VDD and process variations. By using the related WDT configuration option, longer time-out periods can be realized. If the WDT time-out is selected to be 2^{15}, the maximum time-out period is divided by 2^{15}~2^{16}which will give a time-out period of about 2.3s~4.7s.

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 HALT mode the WDT may stop counting and lose its protecting purpose. In this situation the logic can only be restarted by external logic. If the device operates in a noisy environment, using the on-chip RC oscillator (Int.RC OSC) 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 HALT or IDLE 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 $\overline{\text{RES}}$), 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

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, PA0 and PA1. A configuration option is used to select from one of three buzzer options. The first option is for both pins PA0 and PA1 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 PA0 pin to be used as a BZ buzzer pin with the PA1 pin retaining its normal I/O pin function. Note that the $\overline{BZ}$ pin is the inverse of the BZ pin which together generate a differential output which can supply more power to connected interfaces such as buzzers.

The buzzer is driven by the internal clock source, f_S, which then passes through a divider, the division ratio of which is selected by configuration options to provide a range of buzzer frequencies from $f_S/2^2$ to $f_S/2^9$. The clock source that generates f_S, which in turn controls the buzzer frequency, can originate from two different sources, the Int.RCOSC (Internal RC oscillator) or the System oscillator/4, the choice of which is determined by the f_S clock source configuration option. Note that the buzzer frequency is controlled by configuration options, which select both the source clock for the internal clock f_S and the internal division ratio. There are no internal registers associated with the buzzer frequency.

If the configuration options have selected both pins PA0 and PA1 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 PAC0 and PAC1 of the PAC port control register to zero. The PA0 data bit in the PA data register must also be set high to enable the buzzer outputs, if set low, both pins PA0 and PA1 will remain low. In this way the single bit PA0 of the PA register can be used as an on/off control for both the BZ and $\overline{BZ}$ buzzer pin outputs. Note that the PA1 data bit in the PA register has no control over the $\overline{BZ}$ buzzer pin PA1.

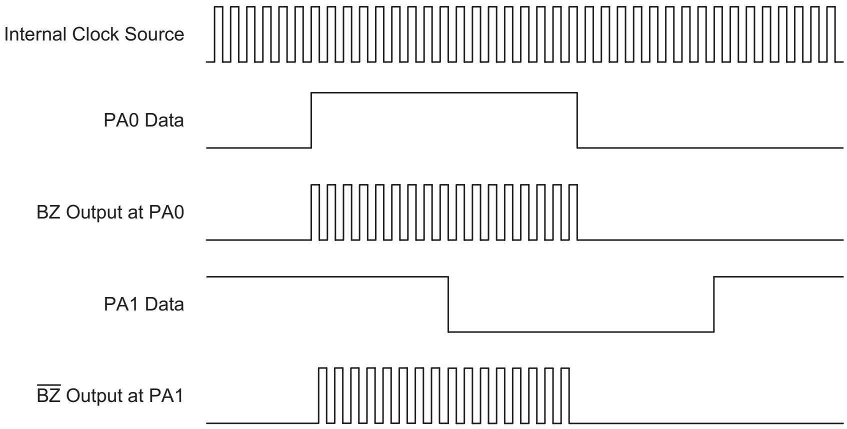
If configuration options have selected that only the PA0 pin is to function as a BZ buzzer pin, then the PA1 pin can be used as a normal I/O pin. For the PA0 pin to function as a BZ buzzer pin, PA0 must be setup as an output by setting bit PAC0 of the PAC port control register to zero. The PA0 data bit in the PA data register must also be set high to enable the buzzer output, if set low pin PA0 will remain low. In this way the PA0 bit can be used as an on/off control for the BZ buzzer pin PA0. If the PAC0 bit of the PAC port control register is set high, then pin PA0 can still be used as an input even though the configuration option has configured it as a BZ buzzer output.

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.

PAC Register PAC.0	PAC Register PAC.1	PA data Register PA.0	PA data Register PA.1	Output Function
0	0	0	X	PA0=0, PA1=0
0	0	1	X	PA0=BZ, PA1=$\overline{BZ}$
0	1	0	X	PA0=0, PA1=Input
0	1	1	X	PA0=BZ, PA1=Input
1	0	0	X	PA0=Input, PA1=0
1	1	X	X	PA0=Input, PA1=Input

PA0/PA1 Pin Function Control

Note: ″X″ stands for don′t care



Buzzer Output Pin Control

Note:The above drawing shows the situation where both pins PA0 and PA1 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 PA1 has no effect on the buzzer outputs.

Power Down Operation – HALT

The HALT mode is initialized by the ″HALT″ instruction and results in the following.

- The system oscillator turns off but the Internal oscillator (Int.RCOSC) keeps running (if the Internal oscillator is selected).
- The contents of the on-chip RAM and of the registers remain unchanged.
- The WDT is cleared and start recounting (if the WDT clock source is from the Internal RC oscillator).
- All I/O ports maintain their original status.
- The PDF flag is set but the TO flag is cleared.
- LCD driver keeps running (if the IRC clock is enabled; IRCC=1).

The system quits the HALT or IDLE mode by means of an external reset, an interrupt, an external falling edge 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 at their original state.

The port A wake-up and interrupt methods can be considered as a continuation of normal execution. Each bit in port A can be independently selected to wake-up the device by options. Awakening from an I/O port stimulus, the program resumes execution of the next instruction. On the other hand, awakening from an interrupt, two sequence may occur. If the related interrupt is disabled or the interrupt is enabled but the stack is full, the program resumes 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.

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 HALT or IDLE 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 the ″initial condition″ 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

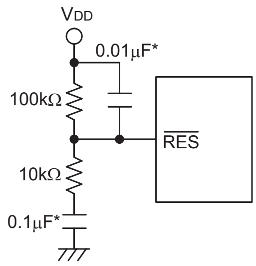
Note: ″u″ stands for unchanged

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.

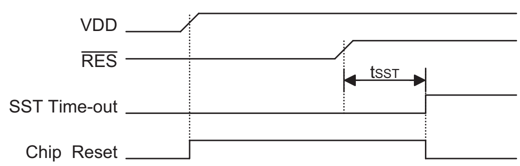
The functional unit chip reset status is shown below.

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

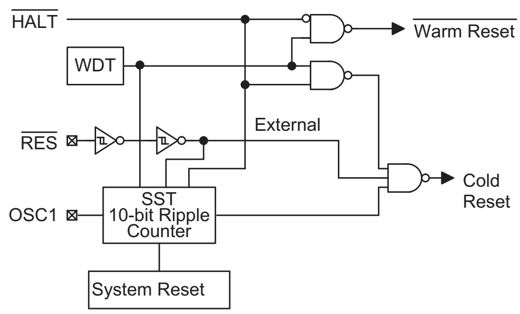


Reset Circuit

Note: ″*″ Make the length of the wiring, which is connected to the $\overline{RES}$ pin as short as possible, to avoid noise interference.



Reset Timing Chart



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	1xxx xxxx	1uuu uuuu	1uuu uuuu	1uuu uuuu	1uuu uuuu
MP1	1xxx xxxx	1uuu uuuu	1uuu uuuu	1uuu uuuu	1uuu uuuu
BP	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
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	--xx xxxx	--uu uuuu	--uu uuuu	--uu uuuu	--uu uuuu
MODE	--0- 00--	--0- 00--	--0- 00--	--0- 00--	--u- uu--
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC0	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
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
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	---- --11	---- --11	---- --11	---- --11	---- --uu
PBC	---- --11	---- --11	---- --11	---- --11	---- --uu
ADCR	00-- x000	00-- x000	00-- x000	00-- x000	00-- x000
ADCD	---- -111	---- -111	---- -111	---- -111	---- -uuu
INTC1	---0 ---0	---0 ---0	---0 ---0	---0 ---0	---u ---u
CHPRC	0000 0-00	0000 0-00	0000 0-00	0000 0-00	uuuu u-uu

Note: ″*″ stands for warm reset

″u″ stands for unchanged

″x″ stands for unknown

Timer/Event Counter

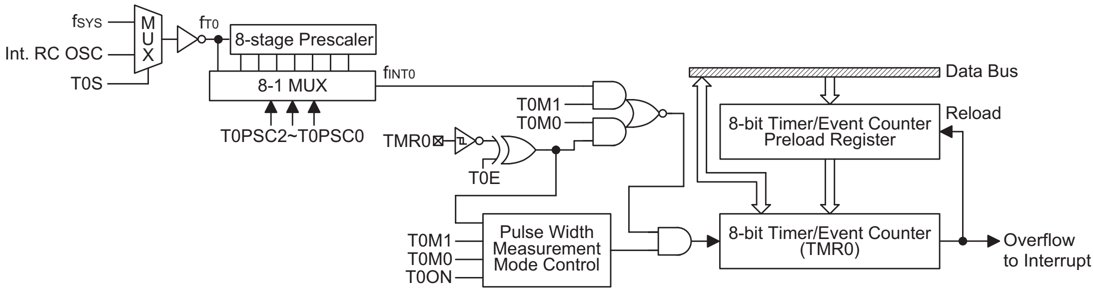
Two timer/event counters (TMR0,TMR1) are implemented in the microcontroller. The Timer/Event Counter 0 contains a 8-bit programmable count-up counter and the clock may come from an external source or an internal clock source. An internal clock source comes from f_{SYS} or Internal RC. The Timer/Event Counter 1 contains a 16-bit programmable count-up counter and the clock may come from an external source or an internal clock source. An internal clock source comes from $f_{SYS}/4$ or Internal RC selected by special function register option. 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 ([0DH]) and TMR0C ([0EH]). 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 some options. There are three registers related to the Timer/Event Counter 1; TMR1H (0FH), TMR1L (10H) and TMR1C (11H). 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.

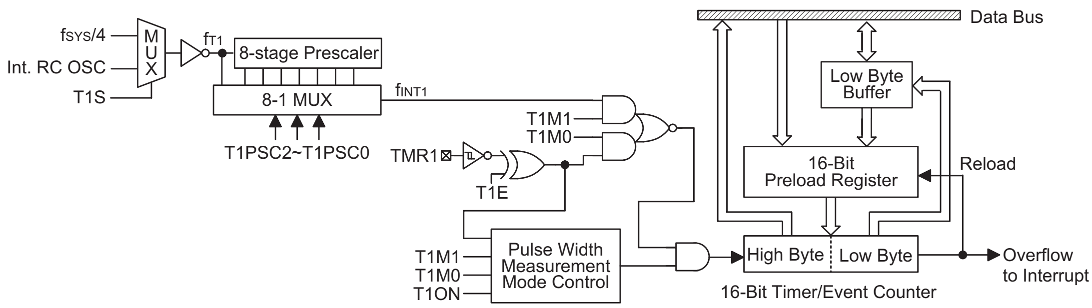
The Timer/Event Counter 1 preload register is changed every time there is a write operation to TRM1H. Reading TMR1H will latch the contents of TMR1H and TMR1L counters to the destination and the lower-order byte buffer, respectively. Reading TMR1L will read the contents of the lower-order byte buffer. The TMR1C is the Timer/Event Counter 1 control register, which defines the operating mode, counting enable or disable and an active edge.

The T0M0, T0M1 (TMR0C) and T1M0, T1M1 (TMR1C) bits define the operation mode. The event count mode is used to count external events, which means that the clock source must come from an external (TMR0, TMR1) 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 TMR0 or TMR1, with the timing based on the internal selected clock source.

In the event count or timer mode, the Timer/Event Counter 0 (1) starts counting at the current contents in the Timer/Event Counter 0 (1) and ends at FFH (FFFFH). Once an overflow occurs, the counter is reloaded from the timer/event counter preload register, and generates an interrupt request flag (T0F; bit 5 of INTC0, T1F; bit6 of INTC0). In the pulse width measurement mode with the values of the T0ON/T1ON and T0E/T1E bits equal to 1, after the TMR0 (TMR1) has received a transient from low to high (or high to low if the TE bit is "0"), it will start counting until the TMR0 (TMR1) pin returns to the original level and resets the T0ON/T1ON bit. The measured result remains in the timer/event counter even if the activated transient occurs again. In other words, only a 1-cycle measurement can be made until the T0ON/T1ON is set. The cycle measurement will



Timer/Event Counter 0



Timer/Event Counter 1

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=Int.RCOSC (Internal RC OSC))
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=Int.RCOSC (Internal RC OSC))
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

re-function as long as it receives further transient pulse. 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 ON bit (T0ON; bit 4 of TMR0C or T1ON bit 4 of TMR1C) should be set to 1. In the pulse width measurement mode, the T0ON (T1ON) is automatically cleared after the measurement cycle is completed. But in the other two modes, the T0ON (T1ON) can only be reset by instructions. The overflow of the Timer/Event Counter 0/1 is one of the wake-up sources. No matter what the operation mode is, writing a 0 to ET0I or ET1I 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 (reading TMR0/TMR1) is read, the clock is blocked to avoid errors, however this may results in a counting error, something that should be taken into account by the programmer. It is strongly recommended to load a desired value into the TMR0/TMR1 register first, before turning on the related timer/event counter, for proper operation since the initial value of TMR0/TMR1 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 TMR1C can be used to define the pre-scaling stages of the internal clock sources of Timer/Event Counter 1.

Input/Output Ports

There are 10 bidirectional input/output lines in the microcontroller, labeled as PA and PB, which are mapped to the data memory of [12H] and [14H] respectively. 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]″ (m=12H or 14H). 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) 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 floating state (depending on pull-high 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.

Each I/O port has a pull-high option. Once the pull-high option is selected, the I/O port has a pull-high resistor, otherwise, there′s none. Take note that a non-pull-high I/O port operating in input mode will cause a floating state.

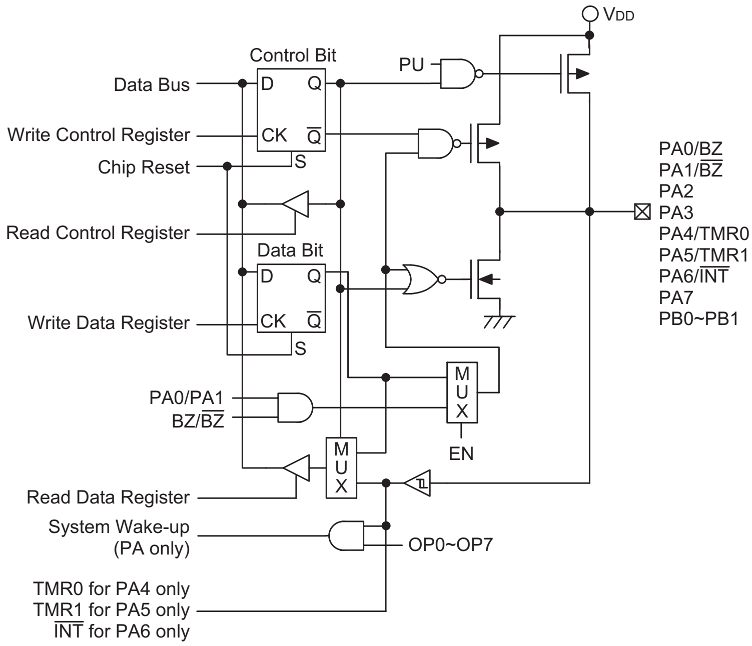
The PA0, PA1, PA4, PA5 and PA6 are pin-shared with BZ, $\overline{BZ}$, TMR0, TMR1 and $\overline{INT}$ pins respectively.

PA0 and PA1 are pin-shared with BZ and $\overline{BZ}$ signal, respectively. If the BZ/$\overline{BZ}$ option is selected, the output signals in the output mode of PA0/PA1 will be the buzzer signal generated by multi-function timer. The input mode always remains in its original function. Once the BZ/$\overline{BZ}$ option is selected, the buzzer output signals are controlled by the PA0, PA1 data register only.

The I/O function of PA0/PA1 are shown below.

PA0 I/O	I	I	O	O	O	O	O	O	O	O
PA1 I/O	I	O	I	I	I	O	O	O	O	O
PA0 Mode	X	X	C	B	B	C	B	B	B	B
PA1 Mode	X	C	X	X	X	C	C	C	B	B
PA0 Data	X	X	D	0	1	D_0	0	1	0	1
PA1 Data	X	D	X	X	X	D1	D	D	X	X
PA0 Pad Status	I	I	D	0	B	D_0	0	B	0	B
PA1 Pad Status	I	D	I	I	I	D_1	D	D	0	B

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



Input/Output Ports

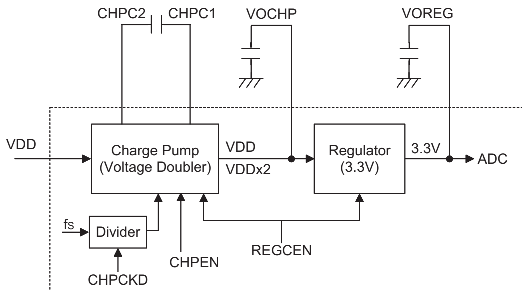
"C" CMOS output

It is recommended that unused or not bonded out I/O lines should be set as output pins by software instructions to avoid consuming power when in an input 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 $V_{DD}\times2$. The regulator can generate a stable voltage of 3.3V, for 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 over 3.6V to ensure that the regulator generates the required 3.3V voltage output. The block diagram of this module is shown below.



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.

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	—	Reserved
3~7	CHPCKD0~ CHPCKD4	The Charge pump clock divider. This 5 bits can form the clock divide by 1~31. 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	$2\times V_{DD}$	ON	3.3V	Active	Use for V_{DD} is less than 3.6V (V_{DD}=2.2V~3.6V)

The CHPCKD4~0 bits are use to set the clock divider to generate the desired clock frequency to provide the charge pump working. The actual frequency is decide by the following formula.

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

The suggestion clock frequency of the charge pump is 20kHz. Application need to set the correct value to get the desired clock frequency. e.g. for the 4MHz application, the CHPCKD should be set to 12, and for 2MHz application, the correct CHPCKD is 6.

The REGCEN bit in the CHPRC 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 a Power Down Mode ignoring the CHPEN setting. The ADC and OPA will also be disabled to reduce power.

If REGCEN is set to logic ″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 of the regulator. If CHPEN is set to logic ″0″, the charge pump is disabled and the charge pump output will be equal to the charge pump input (VDD).

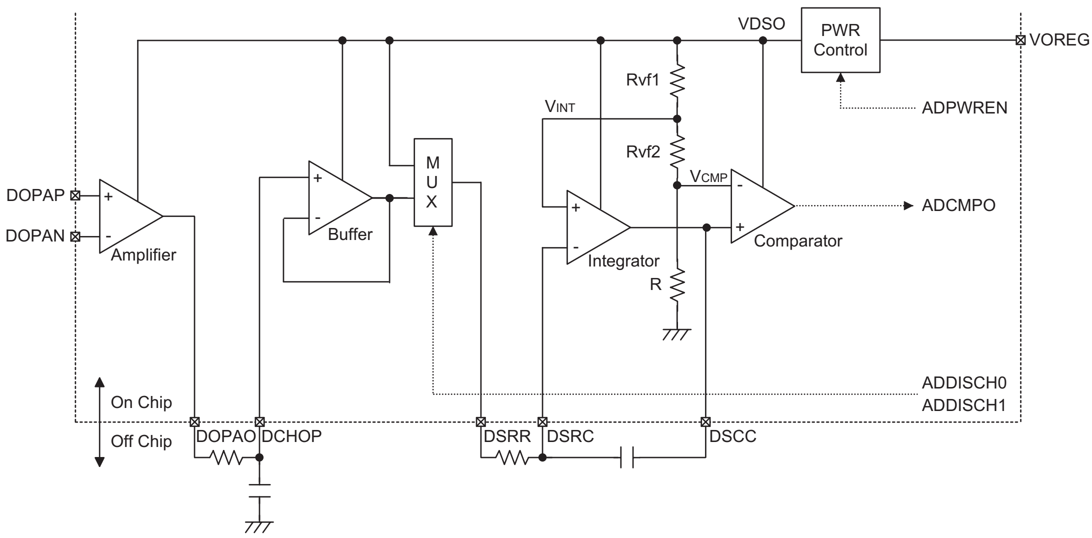
Users need to take care of the V_{DD} voltage, if the voltage is under 3.6V, then CHPEN should be set to 1 to enable the charge pump, otherwise CHPEN should be set to zero. If the Charge pump is disabled and V_{DD} is under 3.6V then the output voltage of the regulator will not be guaranteed.

ADC – Dual Slope

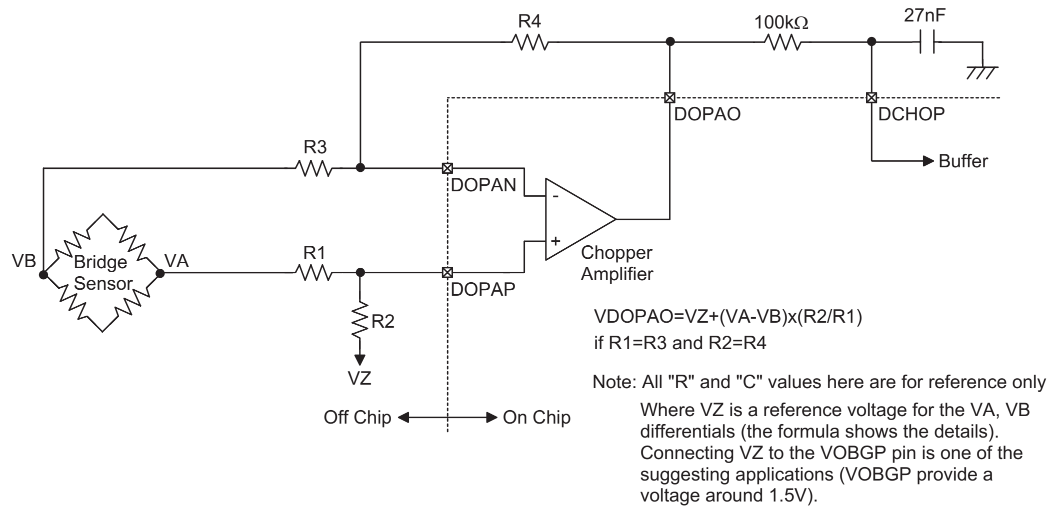
A Dual Slope A/D converter is implemented in this microcontroller. The dual slope module includes an Operational Amplifier and a buffer for the amplification of differential signals, an Integrator and a comparator for the main dual slope AD converter.

In addition, there is also an integrated 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.

There are 2 special function registers related to this block including: 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 is the A/D Chopper clock divider register, which define the chopper clock to the ADC module.

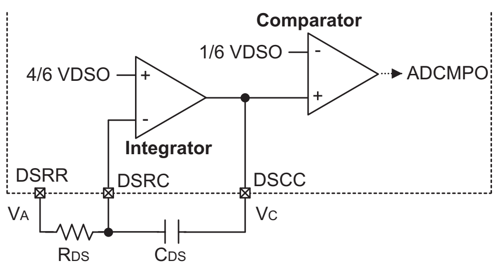


Note: V_{INT}, V_{CMP} signal can come from different R groups which are selected by software registers.



The ADPWREN bit defined in ADCR register is used to control the on/off function of the ADC module. The ADCCKEN bit defined in the ADCR register is used to control the chopper clock on/off. When ADCCKEN is set to logic ″1″ it will enable the Chopper clock, with the clock frequency defined by the ADCD registers. The ADC module includes the OPA, buffer, integrator and Comparator, however the Band gap 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 idle to conserve power. The charge/discharge control bits (ADDISCH1~0) are used to control the Dual slope circuit charging and discharging behavior. The ADCMPO bit is read only for the comparator output, and the ADCMPO changing falling edge will trigger a dual slope ADC interrupt.

The following descriptions are base on the ADRR0=0



The combination of the amplifier and buffer forms a differential input pre-amplifier which amplifies signals from the sensor. The amplification is controlled by the ratio of R1~R4 as shown in the block diagram:

The combinations of the Integrator, the Comparator and the resistor between DSRR and ADRC(Rds) and the capacity between DSRC and DSCC (Cds) form the main body of the Dual slope ADC.

The ″Integrator″ integrates the output voltage increase or decrease controlled by ″Switch Circuit″ (refer to the block diagram). The integrated and de-integrated curves are illustrated in the following:

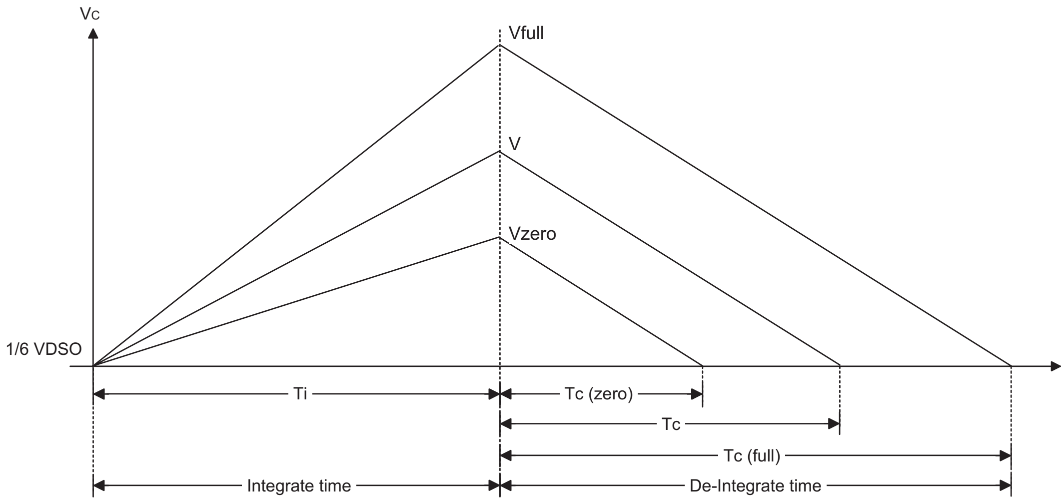
The ″Integrator″ integrates the output voltage increments or decrements controlled by the ″Switch Circuit″ (refer to the block diagram). The integrated and de-integrated curves are illustrated by the following.

The ″comparator″ will switch the state from high to low when the VC (the DSCC pin voltage) drop under the 1/6 VDSO.

In general applications, the application program will switch the ADC to charging mode for a fixed time called Ti (integrating time), and then switch to the dis-charging mode, wait for the V_C drop under the 1/6VDSO (the comparator will change state), keep the time Tc (de-integrating time). And then follow the formula 1 to get the input voltage V_A.

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

Application hints: Application users need to choose the correctly R_{DS}, C_{DS} and the Ti let the V_C work between 6/5VDSO and 1/6VDSO. (e.g. Vfull can□t be over the 5/6VDSO and Vzero can′t be under 1/6VDSO)



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	—	Reserved
6	ADCCKEN	ADC OP chopper clock source on/off switching. 0: disable 1: enable (clock value is defined by ADCD register)
7	ADRR0	ADC resisters selection 0: (V_{INT}, V_{CMP})= (4/6 VOREG, 1/6 VOREG) 1: (V_{INT}, V_{CMP})= (4.4/6 VOREG, 1/6 VOREG)

ADCR (18H) Register

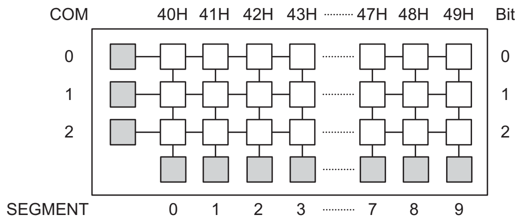
Bit No.	Label	Function
0 1 2	ADCD0 ADCD1 ADCD2	Define the chopper clock (ADCCKEN should be enable), the suggestion clock is around 10kHz. The chopper clock define : 0: clock= (f_{SYS}/32)/1 1: clock= (f_{SYS}/32)/2 2: clock= (f_{SYS}/32)/4 3: clock= (f_{SYS}/32)/8 4: clock= (f_{SYS}/32)/16 5: clock= (f_{SYS}/32)/32 6: clock= (f_{SYS}/32)/64 7: clock= (f_{SYS}/32)/128
3~7	—	Reserved

ADCD (1AH) Register

LCD Display Memory

The device provides an area of embedded data memory for LCD display. This area is located from 40H to 49H of the RAM at Bank 1. Bank pointer (BP; located at 04H of the RAM) is the switch between the RAM and the LCD display memory. When the BP is set as ″1″, any data written into 40H~49H will effect the LCD display. When the BP is cleared to ″0″ or ″1″, any data written into 40H~49H means to access the general purpose data memory. The LCD display memory can be read and written to only by indirect addressing mode using MP1. When data is written into the display data area, it is automatically read by the LCD driver which then generates the corresponding LCD driving signals. To turn the display on or off, a ″1″ or a ″0″ is written to the corresponding bit of the display memory, respectively. The figure illustrates the mapping between the display memory and LCD pattern for the device.

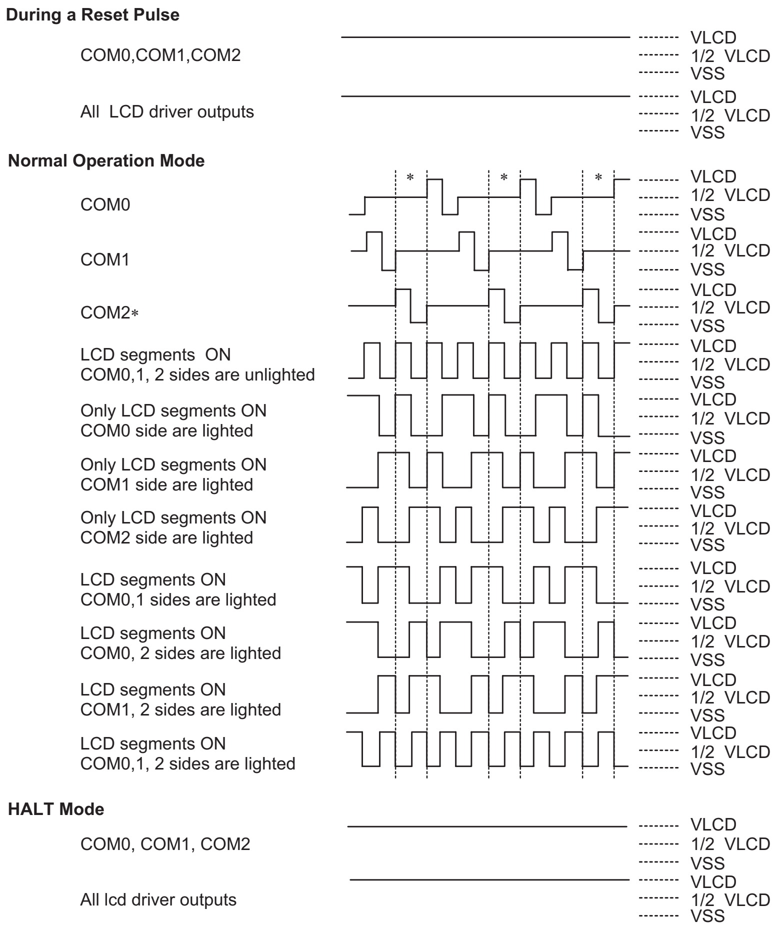
The LCD clock is driven by the IRC clock, which then passes through a divider, the division ratio of which is selected by configuration options to provide a range of LCD frequencies from Int.RCOSC/3 to Int.RCOSC/4. Note that the LCD frequency is controlled by configuration options, which select the internal division ratio. There are no internal registers associated with the buzzer frequency.



Display Memory

LCD Driver Output

The output number of the device LCD driver can be 10×3 by configuration option (i.e., 1/2 duty or 1/3 duty). The bias type LCD driver is R type only. The LCD driver bias voltage can be 1/2 bias or 1/3 bias by option.



Note: "*" Omit the COM2 signal, if the 1/2 duty LCD is used.

LCD Driver Output (1/3 Duty, 1/2 Duty, R Type)

Low Voltage Reset/Detector Functions

There is a low voltage detector (LVD) and a low voltage reset circuit (LVR) implemented in the microcontroller. These two functions can be enabled/disabled by options. Once the LVD options is enabled, the user can use the MODE.3 to enable/disable (1/0) the LVD circuit and read the LVD detector status (0/1) from MODE.5; otherwise, the LVD function is disabled.

The MODE register definitions are listed below.

Bit No.	Label	Function
0~1	—	Unused bit, read as ″0″
2	IRCC	In HALT mode, IRC clock enable or disable selection bit. 0: IRC clock enable and Int.RCOSC on. 1: IRC clock disabled.
3	LVDC	LVD enable/disable (1/0)
4	—	Unused bit, read as ″0″
5	LVDO	LVD detection output (1/0) 1: low voltage detected, read only. 0: low voltage not detected.
6~7	—	Unused bit, read as ″0″

MODE (09H) Register

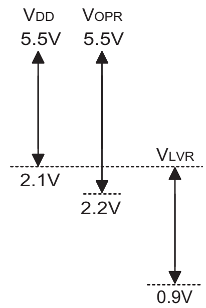
The LVR has the same effect or function with the external RES signal which performs chip reset. During HALT state, LVR is disabled both LVR and LVD are disabled.

The microcontroller provides 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 changing a battery, the LVR will automatically reset the device internally.

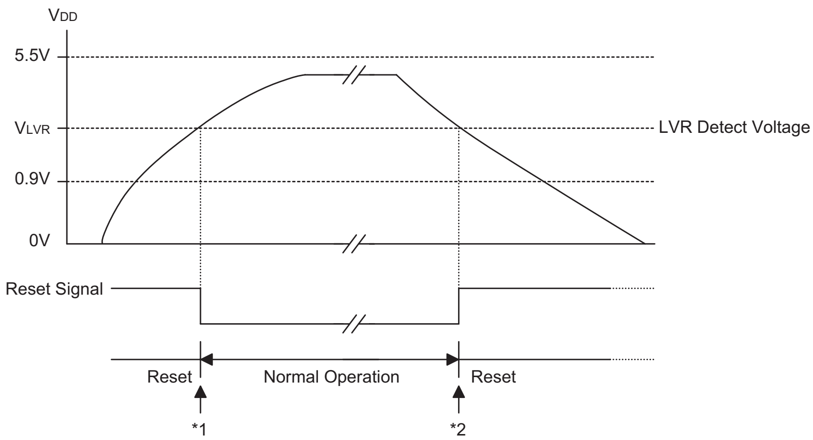
The LVR includes the following specifications:

- The low voltage (0.9V~V_{LVR}) has to remain in their original state to exceed 1ms. If the low voltage state does not exceed 1ms, the LVR will ignore it and do not perform a reset function.
- The LVR uses the ″OR″ function with the external $\overline{RES}$ signal to perform chip reset.

The relationship between V_{DD} and V_{LVR} is shown below.



Note: V_{OPR} is the voltage range for proper chip operation at 4MHz system clock.



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.

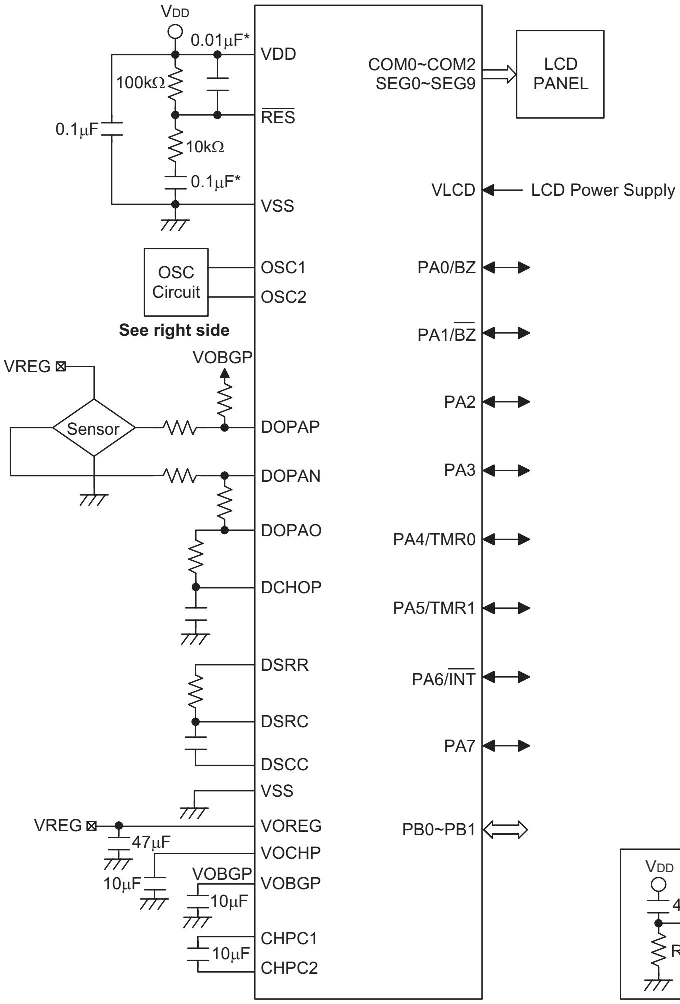
*2: Since a low voltage state has to be maintained in its original state for over 1ms, therefore after 1ms delay, the device enters the reset mode.

Options

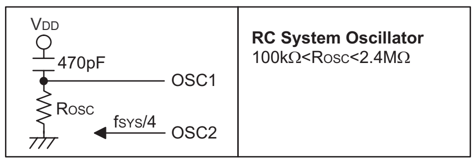
The following shows the options in the device. All these options should be defined in order to ensure proper functioning system.

Options
f_S clock source. There are two types of selections: Int.RCOSC or $f_{SYS}/4$
WDT clock source selection. There are two types of selections: system clock/4 or Int.RCOSC.
WDT enable/disable selection. WDT can be enabled or disabled by option.
WDT time-out period selection. There are four types of selection: WDT clock source divided by $2^{12}/f_S$~$2^{13}/f_S$, $2^{13}/f_S$~$2^{14}/f_S$, $2^{14}/f_S$~$2^{15}/f_S$, $2^{15}/f_S$~$2^{16}/f_S$,
CLR WDT times selection. This option defines the method to clear the WDT by instruction. "One time" means that the "CLR WDT" can clear the WDT. "Two times" means only if both of the "CLR WDT1" and "CLR WDT2" have been executed, the WDT can be cleared.
Buzzer output frequency selection. There are eight types of frequency signals for buzzer output: $f_S/2^2$~$f_S/2^9$. "f_S" means the clock source selected by options.
Wake-up selection. This option defines the wake-up capability. External I/O pins (PA only) all have the capability to wake-up the chip from a HALT by a falling edge (bit option).
Pull-high selection. This option is to decide whether the pull-high resistance is visible or not in the input mode of the I/O ports. PA and PB can be independently selected (bit option).
I/O pins share with other function selections. PA0/$\overline{\text{BZ}}$, PA1/BZ: PA0 and PA1 can be set as I/O pins or buzzer outputs.
LCD common selection. There are three types of selections: 2 common (1/2 duty) or 3 common (1/3 duty).
LCD bias selection. This option is to determine what kind of bias is selected, 1/2 bias or 1/3 bias.
LCD driver clock frequency selection. There are two types of frequency signals for the LCD driver circuits: Int.RCOSC/3~Int.RCOSC/4.
LCD ON/OFF at HALT selection
LVR selection. LVR has enable or disable options
LVD selection. LVD has enable or disable options
$\overline{\text{INT}}$ trigger edge selection: disable; high to low; low to high; low to high or high to low
Partial-lock selection: Page0~3, Page4~6, Page7.

Application Circuits



HT46R71D



OSC Circuit

Note: The resistance and capacitance for reset circuit should be designed in such a way as to ensure that V_{DD} is stable and remains within a valid operating voltage range before bringing $\overline{\text{RES}}$ high.

"*" Make the length of the wiring, which is connected to the $\overline{\text{RES}}$ pin as short as possible, to avoid noise interference.

Instruction Set Summary

Mnemonic	Description	Instruction Cycle	Flag Affected
Arithmetic			
ADD A,[m]	Add data memory to ACC	1	Z,C,AC,OV
ADDM A,[m]	Add ACC to data memory	$1^{(1)}$	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	$1^{(1)}$	Z,C,AC,OV
SUB A,x	Subtract immediate data from 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	$1^{(1)}$	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 and result in data memory	$1^{(1)}$	Z,C,AC,OV
DAA [m]	Decimal adjust ACC for addition with result in data memory	$1^{(1)}$	C
Logic Operation			
AND A,[m]	AND data memory to ACC	1	Z
OR A,[m]	OR data memory to ACC	1	Z
XOR A,[m]	Exclusive-OR data memory to ACC	1	Z
ANDM A,[m]	AND ACC to data memory	$1^{(1)}$	Z
ORM A,[m]	OR ACC to data memory	$1^{(1)}$	Z
XORM A,[m]	Exclusive-OR ACC to data memory	$1^{(1)}$	Z
AND A,x	AND immediate data to ACC	1	Z
OR A,x	OR immediate data to ACC	1	Z
XOR A,x	Exclusive-OR immediate data to ACC	1	Z
CPL [m]	Complement data memory	$1^{(1)}$	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	$1^{(1)}$	Z
DECA [m]	Decrement data memory with result in ACC	1	Z
DEC [m]	Decrement data memory	$1^{(1)}$	Z
Rotate			
RRA [m]	Rotate data memory right with result in ACC	1	None
RR [m]	Rotate data memory right	$1^{(1)}$	None
RRCA [m]	Rotate data memory right through carry with result in ACC	1	C
RRC [m]	Rotate data memory right through carry	$1^{(1)}$	C
RLA [m]	Rotate data memory left with result in ACC	1	None
RL [m]	Rotate data memory left	$1^{(1)}$	None
RLCA [m]	Rotate data memory left through carry with result in ACC	1	C
RLC [m]	Rotate data memory left through carry	$1^{(1)}$	C
Data Move			
MOV A,[m]	Move data memory to ACC	1	None
MOV [m],A	Move ACC to data memory	$1^{(1)}$	None
MOV A,x	Move immediate data to ACC	1	None
Bit Operation			
CLR [m].i	Clear bit of data memory	$1^{(1)}$	None
SET [m].i	Set bit of data memory	$1^{(1)}$	None

Mnemonic	Description	Instruction Cycle	Flag Affected
Branch			
JMP addr	Jump unconditionally	2	None
SZ [m]	Skip if data memory is zero	1(2)	None
SZA [m]	Skip if data memory is zero with data movement to ACC	1(2)	None
SZ [m].i	Skip if bit i of data memory is zero	1(2)	None
SNZ [m].i	Skip if bit i of data memory is not zero	1(2)	None
SIZ [m]	Skip if increment data memory is zero	1(3)	None
SDZ [m]	Skip if decrement data memory is zero	1(3)	None
SIZA [m]	Skip if increment data memory is zero with result in ACC	1(2)	None
SDZA [m]	Skip if decrement data memory is zero with result in ACC	1(2)	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 ROM code (current page) to data memory and TBLH	2(1)	None
TABRDL [m]	Read ROM code (last page) to data memory and TBLH	2(1)	None
Miscellaneous			
NOP	No operation	1	None
CLR [m]	Clear data memory	1(1)	None
SET [m]	Set data memory	1(1)	None
CLR WDT	Clear Watchdog Timer	1	TO,PDF
CLR WDT1	Pre-clear Watchdog Timer	1	TO(4),PDF(4)
CLR WDT2	Pre-clear Watchdog Timer	1	TO(4),PDF(4)
SWAP [m]	Swap nibbles of data memory	1(1)	None
SWAPA [m]	Swap nibbles of data memory with result in ACC	1	None
HALT	Enter Power Down Mode	1	TO,PDF

Note: x: Immediate data

m: Data memory address

A: Accumulator

i: 0~7 number of bits

addr: Program memory address

√: Flag is affected

–: Flag is not affected

(1): If a loading to the PCL register occurs, the execution cycle of instructions will be delayed for one more cycle (four system clocks).

(2): If a skipping to the next instruction occurs, the execution cycle of instructions will be delayed for one more cycle (four system clocks). Otherwise the original instruction cycle is unchanged.

(3): (1) and (2)

(4): The flags may be affected by the execution status. If the Watchdog Timer is cleared by executing the ″CLR WDT1″ or ″CLR WDT2″ instruction, the TO and PDF are cleared.
Otherwise the TO and PDF flags remain unchanged.

Instruction Definition

ADC A,[m] Add data memory and carry to the accumulator

Description: The contents of the specified data memory, accumulator and the carry flag are added simultaneously, leaving the result in the accumulator.

Operation: ACC ← ACC+[m]+C

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADCM A,[m] Add the accumulator and carry to data memory

Description: The contents of the specified data memory, accumulator and the carry flag are added simultaneously, leaving the result in the specified data memory.

Operation: [m] ← ACC+[m]+C

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADD A,[m] Add data memory to the accumulator

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)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADD A,x Add immediate data to the accumulator

Description: The contents of the accumulator and the specified data are added, leaving the result in the accumulator.

Operation: ACC ← ACC+x

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADDM A,[m] Add the accumulator to the data memory

Description: The contents of the specified data memory and the accumulator are added. The result is stored in the data memory.

Operation: [m] ← ACC+[m]

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

AND A,[m] Logical AND accumulator with data memory

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)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

AND A,x Logical AND immediate data to the accumulator

Description: Data in the accumulator and the specified data perform a bitwise logical_AND operation. The result is stored in the accumulator.

Operation: ACC ← ACC ″AND″ x

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

ANDM A,[m] Logical AND data memory with the accumulator

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)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

CALL addr Subroutine call

Description: The instruction unconditionally calls a subroutine located at the indicated address. The program counter increments once to obtain the address of the next instruction, and pushes this onto the stack. The indicated address is then loaded. Program execution continues with the instruction at this address.

Operation: Stack ← Program Counter+1
Program Counter ← addr

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

CLR [m] Clear data memory

Description: The contents of the specified data memory are cleared to 0.

Operation: [m] ← 00H

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

CLR [m].i Clear bit of data memory

Description: The bit i of the specified data memory is cleared to 0.

Operation: [m].i ← 0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

CLR WDT Clear Watchdog Timer

Description: The WDT is cleared (clears the WDT). The power down bit (PDF) and time-out bit (TO) are cleared.

Operation: WDT ← 00H
PDF and TO ← 0

Affected flag(s)

TO	PDF	OV	Z	AC	C
0	0	—	—	—	—

CLR WDT1 Preclear Watchdog Timer

Description: Together with CLR WDT2, clears the WDT. PDF and TO are also cleared. Only execution of this instruction without the other preclear instruction just sets the indicated flag which implies this instruction has been executed and the TO and PDF flags remain unchanged.

Operation: WDT ← 00H*
PDF and TO ← 0*

Affected flag(s)

TO	PDF	OV	Z	AC	C
0*	0*	—	—	—	—

CLR WDT2 Preclear Watchdog Timer

Description: Together with CLR WDT1, clears the WDT. PDF and TO are also cleared. Only execution of this instruction without the other preclear instruction, sets the indicated flag which implies this instruction has been executed and the TO and PDF flags remain unchanged.

Operation: WDT ← 00H*
PDF and TO ← 0*

Affected flag(s)

TO	PDF	OV	Z	AC	C
0*	0*	—	—	—	—

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)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

CPLA [m] Complement data memory and place result in the accumulator

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

Operation: ACC ← $\overline{[m]}$

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

DAA [m] Decimal-Adjust accumulator for addition

Description: The accumulator value is adjusted to the BCD (Binary Coded Decimal) code. The accumulator is divided into two nibbles. Each nibble is adjusted to the BCD code and an internal carry (AC1) will be done if the low nibble of the accumulator is greater than 9. The BCD adjustment is done by adding 6 to the original value if the original value is greater than 9 or a carry (AC or C) is set; otherwise the original value remains unchanged. The result is stored in the data memory and only the carry flag (C) may be affected.

Operation:
If ACC.3~ACC.0 >9 or AC=1
then [m].3~[m].0 ← (ACC.3~ACC.0)+6, AC1=$\overline{AC}$
else [m].3~[m].0 ← (ACC.3~ACC.0), AC1=0
and
If ACC.7~ACC.4+AC1 >9 or C=1
then [m].7~[m].4 ← ACC.7~ACC.4+6+AC1,C=1
else [m].7~[m].4 ← ACC.7~ACC.4+AC1,C=C

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

DEC [m] Decrement data memory

Description: Data in the specified data memory is decremented by 1.

Operation: [m] ← [m]–1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

DECA [m] Decrement data memory and place result in the accumulator

Description: Data in the specified data memory is decremented by 1, leaving the result in the accumulator. The contents of the data memory remain unchanged.

Operation: ACC ← [m]–1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

HALT Enter Power Down Mode

Description: This instruction stops program execution and turns off the system clock. The contents of the RAM and registers are retained. The WDT and prescaler are cleared. The power down bit (PDF) is set and the WDT time-out bit (TO) is cleared.

Operation: Program Counter ← Program Counter+1
PDF ← 1
TO ← 0

Affected flag(s)

TO	PDF	OV	Z	AC	C
0	1	—	—	—	—

INC [m] Increment data memory

Description: Data in the specified data memory is incremented by 1

Operation: [m] ← [m]+1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

INCA [m] Increment data memory and place result in the accumulator

Description: Data in the specified data memory is incremented by 1, leaving the result in the accumulator. The contents of the data memory remain unchanged.

Operation: ACC ← [m]+1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

JMP addr Directly jump

Description: The program counter are replaced with the directly-specified address unconditionally, and control is passed to this destination.

Operation: Program Counter ←addr

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

MOV A,[m] Move data memory to the accumulator

Description: The contents of the specified data memory are copied to the accumulator.

Operation: ACC ← [m]

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

MOV A,x Move immediate data to the accumulator

Description The 8-bit data specified by the code is loaded into the accumulator.

Operation ACC ← x

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

MOV [m],A Move the accumulator to data memory

Description The contents of the accumulator are copied to the specified data memory (one of the data memories).

Operation [m] ←ACC

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

NOP No operation

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

Operation Program Counter ← Program Counter+1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

OR A,[m] Logical OR accumulator with data memory

Description Data in the accumulator and the specified data memory (one of the data memories) perform a bitwise logical_OR operation. The result is stored in the accumulator.

Operation ACC ← ACC ″OR″ [m]

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

OR A,x Logical OR immediate data to the accumulator

Description Data in the accumulator and the specified data perform a bitwise logical_OR operation. The result is stored in the accumulator.

Operation ACC ← ACC ″OR″ x

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

ORM A,[m] Logical OR data memory with the accumulator

Description Data in the data memory (one of the data memories) 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)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

RET Return from subroutine

Description: The program counter is restored from the stack. This is a 2-cycle instruction.

Operation: Program Counter ← Stack

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RET A,x Return and place immediate data in the accumulator

Description: The program counter is restored from the stack and the accumulator loaded with the specified 8-bit immediate data.

Operation: Program Counter ← Stack
ACC ← x

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RETI Return from interrupt

Description: The program counter is restored from the stack, and interrupts are enabled by setting the EMI bit. EMI is the enable master (global) interrupt bit.

Operation: Program Counter ← Stack
EMI ← 1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RL [m] Rotate data memory left

Description: The contents of the specified data memory are rotated 1 bit left with bit 7 rotated into bit 0.

Operation: [m].(i+1) ← [m].i; [m].i:bit i of the data memory (i=0~6)
[m].0 ← [m].7

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RLA [m] Rotate data memory left and place result in the accumulator

Description: Data in the specified data memory is rotated 1 bit left with bit 7 rotated into bit 0, leaving the rotated result in the accumulator. The contents of the data memory remain unchanged.

Operation: ACC.(i+1) ← [m].i; [m].i:bit i of the data memory (i=0~6)
ACC.0 ← [m].7

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RLC [m] Rotate data memory left through carry

Description: The contents of the specified data memory and the carry flag are rotated 1 bit left. Bit 7 replaces the carry bit; the original carry flag is rotated into the bit 0 position.

Operation:
[m].(i+1) ← [m].i; [m].i:bit i of the data memory (i=0~6)
[m].0 ← C
C ← [m].7

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

RLCA [m] Rotate left through carry and place result in the accumulator

Description: Data in the specified data memory and the carry flag are rotated 1 bit left. Bit 7 replaces the carry bit and the original carry flag is rotated into bit 0 position. The rotated result is stored in the accumulator but the contents of the data memory remain unchanged.

Operation:
ACC.(i+1) ← [m].i; [m].i:bit i of the data memory (i=0~6)
ACC.0 ← C
C ← [m].7

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

RR [m] Rotate data memory right

Description: The contents of the specified data memory are rotated 1 bit right with bit 0 rotated to bit 7.

Operation:
[m].i ← [m].(i+1); [m].i:bit i of the data memory (i=0~6)
[m].7 ← [m].0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RRA [m] Rotate right and place result in the accumulator

Description: Data in the specified data memory is rotated 1 bit right with bit 0 rotated into bit 7, leaving the rotated result in the accumulator. The contents of the data memory remain unchanged.

Operation:
ACC.(i) ← [m].(i+1); [m].i:bit i of the data memory (i=0~6)
ACC.7 ← [m].0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RRC [m] Rotate data memory right through carry

Description: The contents of the specified data memory and the carry flag are together rotated 1 bit right. Bit 0 replaces the carry bit; the original carry flag is rotated into the bit 7 position.

Operation:
[m].i ← [m].(i+1); [m].i:bit i of the data memory (i=0~6)
[m].7 ← C
C ← [m].0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

RRCA [m] Rotate right through carry and place result in the accumulator

Description: Data of the specified data memory and the carry flag are rotated 1 bit right. Bit 0 replaces the carry bit and the original carry flag is rotated into the bit 7 position. The rotated result is stored in the accumulator. The contents of the data memory remain unchanged.

Operation: ACC.i ← [m].(i+1); [m].i:bit i of the data memory (i=0~6)
ACC.7 ← C
C ← [m].0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

SBC A,[m] Subtract data memory and carry from the accumulator

Description: The contents of the specified data memory and the complement of the carry flag are subtracted from the accumulator, leaving the result in the accumulator.

Operation: ACC ← ACC+$\overline{[m]}$+C

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SBCM A,[m] Subtract data memory and carry from the accumulator

Description: The contents of the specified data memory and the complement of the carry flag are subtracted from the accumulator, leaving the result in the data memory.

Operation: [m] ← ACC+$\overline{[m]}$+C

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SDZ [m] Skip if decrement data memory is 0

Description: The contents of the specified data memory are decremented by 1. If the result is 0, the next instruction is skipped. If the result is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation: Skip if ([m]–1)=0, [m] ← ([m]–1)

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SDZA [m] Decrement data memory and place result in ACC, skip if 0

Description: The contents of the specified data memory are decremented by 1. If the result is 0, the next instruction is skipped. The result is stored in the accumulator but the data memory remains unchanged. If the result is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation: Skip if ([m]–1)=0, ACC ← ([m]–1)

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SET [m] Set data memory

Description: Each bit of the specified data memory is set to 1.

Operation: [m] ← FFH

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SET [m]. i Set bit of data memory

Description: Bit i of the specified data memory is set to 1.

Operation: [m].i ← 1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SIZ [m] Skip if increment data memory is 0

Description: The contents of the specified data memory are incremented by 1. If the result is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation: Skip if ([m]+1)=0, [m] ← ([m]+1)

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SIZA [m] Increment data memory and place result in ACC, skip if 0

Description: The contents of the specified data memory are incremented by 1. If the result is 0, the next instruction is skipped and the result is stored in the accumulator. The data memory remains unchanged. If the result is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation: Skip if ([m]+1)=0, ACC ← ([m]+1)

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SNZ [m].i Skip if bit i of the data memory is not 0

Description: If bit i of the specified data memory is not 0, the next instruction is skipped. If bit i of the data memory is not 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation: Skip if [m].i≠0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SUB A,[m] Subtract data memory from the accumulator

Description: The specified data memory is subtracted from the contents of the accumulator, leaving the result in the accumulator.

Operation: ACC $\leftarrow$ ACC+$[\overline{m}]$+1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SUBM A,[m] Subtract data memory from the accumulator

Description: The specified data memory is subtracted from the contents of the accumulator, leaving the result in the data memory.

Operation: [m] $\leftarrow$ ACC+$[\overline{m}]$+1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SUB A,x Subtract immediate data from the accumulator

Description: The immediate data specified by the code is subtracted from the contents of the accumulator, leaving the result in the accumulator.

Operation: ACC $\leftarrow$ ACC+$\overline{x}$+1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SWAP [m] Swap nibbles within the data memory

Description: The low-order and high-order nibbles of the specified data memory (1 of the data memories) are interchanged.

Operation: [m].3~[m].0 $\leftrightarrow$ [m].7~[m].4

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SWAPA [m] Swap data memory and place result in the accumulator

Description: The low-order and high-order nibbles of the specified data memory are interchanged, writing the result to the accumulator. The contents of the data memory remain unchanged.

Operation: ACC.3~ACC.0 $\leftarrow$ [m].7~[m].4
ACC.7~ACC.4 $\leftarrow$ [m].3~[m].0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZ [m] Skip if data memory is 0

Description: If the contents of the specified data memory are 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation: Skip if [m]=0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZA [m] Move data memory to ACC, skip if 0

Description: The contents of the specified data memory are copied to the accumulator. If the contents is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation: Skip if [m]=0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZ [m].i Skip if bit i of the data memory is 0

Description: If bit i of the specified data memory is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation: Skip if [m].i=0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

TABRDC [m] Move the ROM code (current page) to TBLH and data memory

Description: The low byte of ROM code (current page) addressed by the table pointer (TBLP) is moved to the specified data memory and the high byte transferred to TBLH directly.

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

TABRDL [m] Move the ROM code (last page) to TBLH and data memory

Description: The low byte of ROM code (last page) addressed by the table pointer (TBLP) is moved to the data memory and the high byte transferred to TBLH directly.

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

XOR A,[m] Logical XOR accumulator with data memory

Description: Data in the accumulator and the indicated data memory perform a bitwise logical Exclusive_OR operation and the result is stored in the accumulator.

Operation: ACC ← ACC ″XOR″ [m]

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

XORM A,[m] Logical XOR data memory with the accumulator

Description: Data in the indicated data memory and the accumulator perform a bitwise logical Exclusive_OR operation. The result is stored in the data memory. The 0 flag is affected.

Operation: [m] ← ACC ″XOR″ [m]

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

XOR A,x Logical XOR immediate data to the accumulator

Description: Data in the accumulator and the specified data perform a bitwise logical Exclusive_OR operation. The result is stored in the accumulator. The 0 flag is affected.

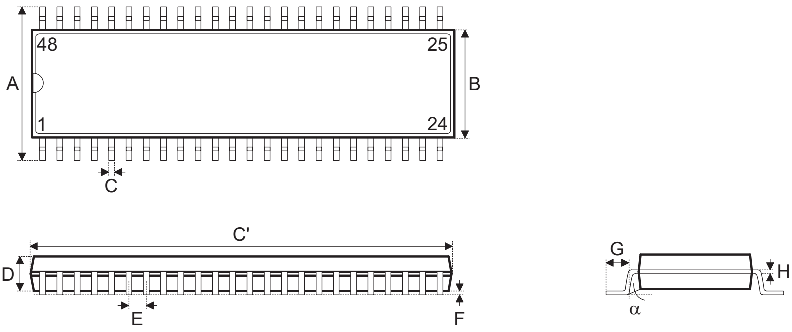
Operation: ACC ← ACC ″XOR″ x

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

Package Information

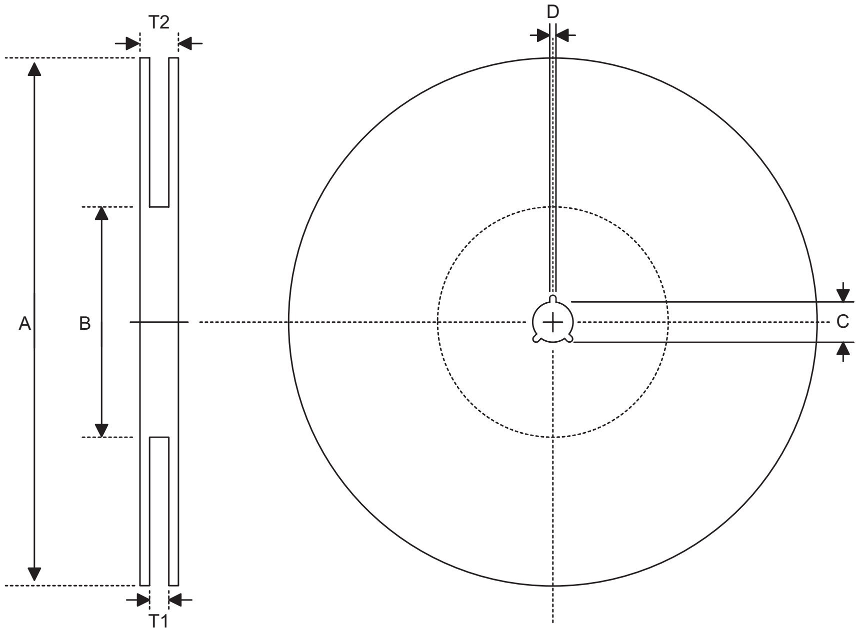
48-pin SSOP (300mil) Outline Dimensions



Symbol	Dimensions in mil		
	Min.	Nom.	Max.
A	395	—	420
B	291	—	299
C	8	—	12
C′	613	—	637
D	85	—	99
E	—	25	—
F	4	—	10
G	25	—	35
H	4	—	12
α	0°	—	8°

Product Tape and Reel Specifications

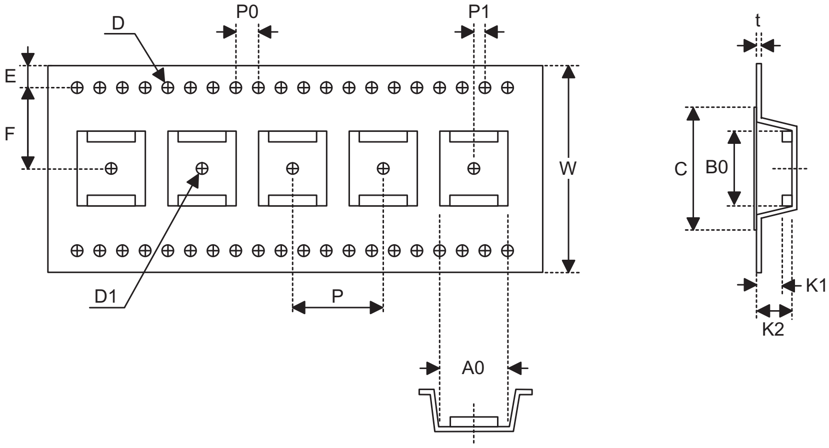
Reel Dimensions



SSOP 48W

Symbol	Description	Dimensions in mm
A	Reel Outer Diameter	330±1
B	Reel Inner Diameter	100±0.1
C	Spindle Hole Diameter	13+0.5 –0.2
D	Key Slit Width	2±0.5
T1	Space Between Flange	32.2+0.3 –0.2
T2	Reel Thickness	38.2±0.2

Carrier Tape Dimensions



SSOP 48W

Symbol	Description	Dimensions in mm
W	Carrier Tape Width	32±0.3
P	Cavity Pitch	16±0.1
E	Perforation Position	1.75±0.1
F	Cavity to Perforation (Width Direction)	14.2±0.1
D	Perforation Diameter	2 Min.
D1	Cavity Hole Diameter	1.5+0.25
P0	Perforation Pitch	4±0.1
P1	Cavity to Perforation (Length Direction)	2±0.1
A0	Cavity Length	12±0.1
B0	Cavity Width	16.2±0.1
K1	Cavity Depth	2.4±0.1
K2	Cavity Depth	3.2±0.1
t	Carrier Tape Thickness	0.35±0.05
C	Cover Tape Width	25.5

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. (Shanghai Sales Office)
7th Floor, Building 2, No.889, Yi Shan Rd., Shanghai, China 200233
Tel: 021-6485-5560
Fax: 021-6485-0313
http://www.holtek.com.cn

Holtek Semiconductor Inc. (Shenzhen Sales Office)
43F, SEG Plaza, Shen Nan Zhong Road, Shenzhen, China 518031
Tel: 0755-8346-5589
Fax: 0755-8346-5590
ISDN: 0755-8346-5591

Holtek Semiconductor Inc. (Beijing Sales Office)
Suite 1721, Jinyu Tower, A129 West Xuan Wu Men Street, Xicheng District, Beijing, China 100031
Tel: 010-6641-0030, 6641-7751, 6641-7752
Fax: 010-6641-0125

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