HOLTEK

HT46R62/HT46C62
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
  - HA0047E An PWM application example using the HT46 series of MCUs

Features

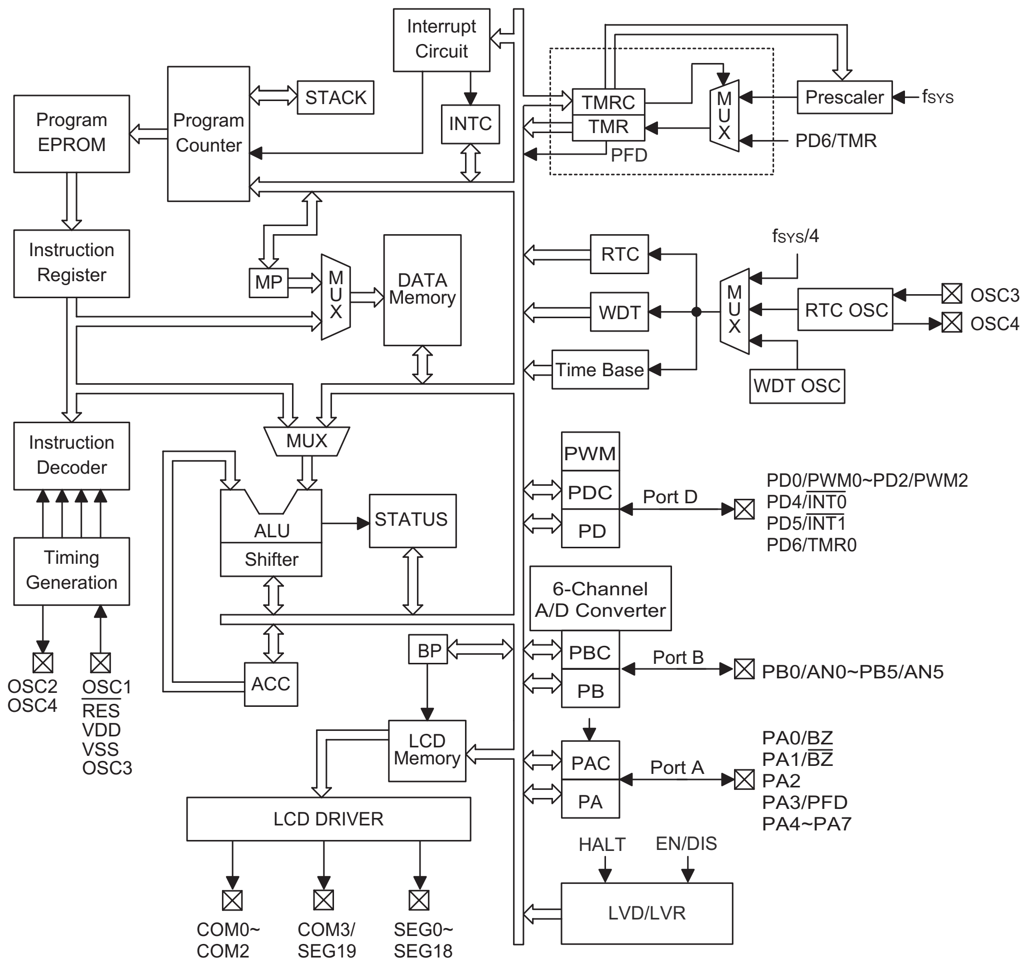
- Operating voltage:
  f_{SYS}=4MHz: 2.2V~5.5V
  f_{SYS}=8MHz: 3.3V~5.5V
- 20 bidirectional I/O lines
  (PA, PB0~PB5, PD0~PD2, PD4~PD6)
- Two external interrupt input
- One 8-bit programmable timer/event counter with PFD (programmable frequency divider) function
- LCD driver with 20×3 or 19×4 segments
  (logical output option for SEG0~SEG15)
- 2K×14 program memory
- 88×8 data memory RAM
- Supports PFD for sound generation
- Real Time Clock (RTC)
- 8-bit prescaler for RTC
- Watchdog Timer
- Buzzer output
- On-chip crystal, RC and 32768Hz crystal oscillator
- HALT function and wake-up feature reduce power consumption
- 6-level subroutine nesting
- 6 channels 9-bit resolution A/D converter
- 3-channel 8-bit PWM output shared with 3 I/O lines
- Bit manipulation instruction
- 16-bit table read instruction
- Up to 0.5μs instruction cycle with 8MHz system clock
- 63 powerful instructions
- All instructions in 1 or 2 machine cycles
- Low voltage reset/detector function
- 52-pin QFP, 56-pin SSOP packages

General Description

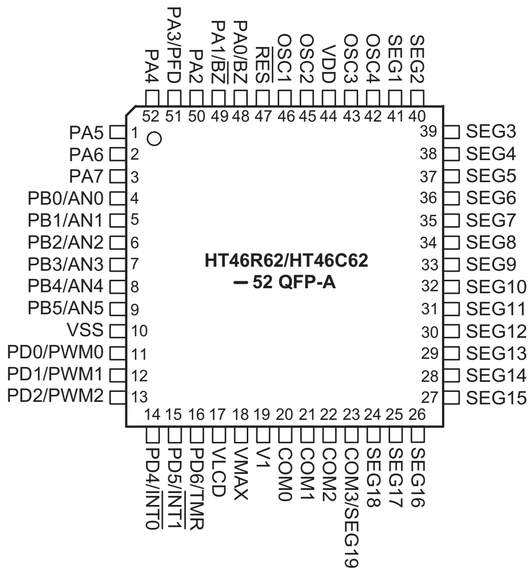
The HT46R62/HT46C62 are 8-bit, high performance, RISC architecture microcontroller devices specifically designed for A/D product applications that interface directly to analog signals and which require LCD Interface. The mask version HT46C62 is fully pin and functionally compatible with the OTP version HT46R62 device.

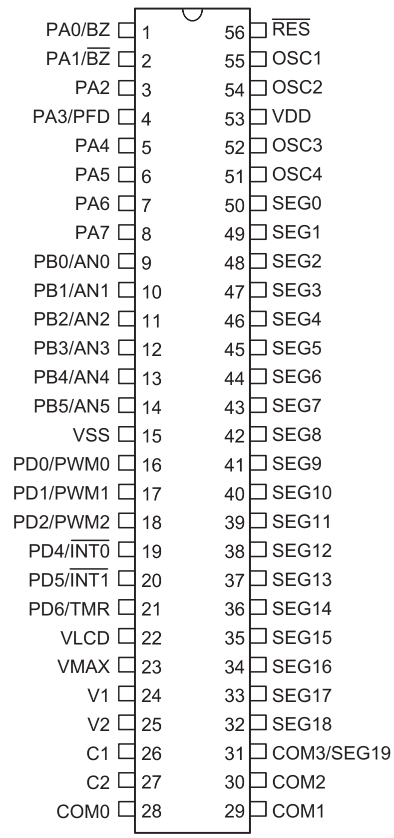
The advantages of low power consumption, I/O flexibility, timer functions, oscillator options, multi-channel A/D Converter, Pulse Width Modulation function, HALT and wake-up functions, in addition to a flexible and configurable LCD interface enhance the versatility of these devices to control a wide 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





HT46R62/HT46C62
— 56 SSOP-A

Note: The 52-pin QFP package does not support the charge pump (C type bias) of the LCD. The LCD bias type must select the R type by option.

Pin Description

Pin Name	I/O	Options	Description
PA0/BZ PA1/$\overline{BZ}$ PA2 PA3/PFD PA4~PA7	I/O	Wake-up Pull-high Buzzer PFD	Bidirectional 8-bit input/output port. Each bit can be configured as wake-up input by option. Software instructions determine the CMOS output or Schmitt Trigger input with or without pull-high resistor (determined by pull-high options: bit option). The BZ, $\overline{BZ}$ and PFD are pin-shared with PA0, PA1 and PA3, respectively.
PB0/AN0 PB1/AN1 PB2/AN2 PB3/AN3 PB4/AN4 PB5/AN5	I/O	Pull-high	Bidirectional 6-bit input/output port. Software instructions determine the CMOS output, Schmitt trigger input with or without pull-high resistor (determined by pull-high option: bit option) or A/D input. Once a PB line is selected as an A/D input (by using software control), the I/O function and pull-high resistor are disabled automatically.
PD0/PWM0 PD1/PWM1 PD2/PWM2	I/O	Pull-high PWM	Bidirectional 3-bit input/output port. Software instructions determine the CMOS output, Schmitt trigger input with or without a pull-high resistor (determined by pull-high option: bit option). The PWM0/PWM1/PWM2 output function are pin-shared with PD0/PD1/PD2 (dependent on PWM options).
PD4/$\overline{INT0}$ PD5/$\overline{INT1}$ PD6/TMR	I/O	Pull-high	Bidirectional 3-bit input/output port. Software instructions determine the CMOS output, Schmitt trigger input with or without a pull-high resistor (determined by pull-high option: bit option). The $\overline{INT0}$, $\overline{INT1}$ and TMR are pin-shared with PD4/PD5/PD6.
VSS	—	—	Negative power supply, ground
VLCD	I	—	LCD power supply
VMAX	I	—	IC maximum voltage connect to VDD, VLCD or V1
V1, V2, C1, C2	I	—	Voltage pump
COM0~COM2 COM3/SEG19	O	1/2, 1/3 or 1/4 Duty	SEG19 can be set as a segment or as a common output driver for LCD panel by options. COM0~COM2 are outputs for LCD panel plate.
SEG0~SEG18	O	Logical Output	LCD driver outputs for LCD panel segments. SEG0~SEG15 can be optioned as logical outputs.
OSC1 OSC2	I O	Crystal or RC	OSC1 and OSC2 are connected to an RC network or a crystal (by options) for the internal system clock. In the case of RC operation, OSC2 is the output terminal for 1/4 system clock. The system clock may come from the RTC oscillator. If the system clock comes from RTCOSC, these two pins can be floating.
OSC3 OSC4	I O	RTC or System Clock	Real time clock oscillators. OSC3 and OSC4 are connected to a 32768Hz crystal oscillator for timing purposes or to a system clock source (depending on the options). No built-in capacitor
VDD	—	—	Positive power supply
$\overline{RES}$	I	—	Schmitt trigger reset input, active low

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
		—	f_{SYS}=8MHz	3.3	—	5.5	V
I_{DD1}	Operating Current (Crystal OSC, RC OSC)	3V	No load, ADC Off, f_{SYS}=4MHz	—	1	2	mA
		5V		—	3	5	mA
I_{DD2}	Operating Current (Crystal OSC, RC OSC)	5V	No load, ADC Off, f_{SYS}=8MHz	—	4	8	mA
I_{DD3}	Operating Current (f_{SYS}=32768Hz)	3V	No load, ADC Off	—	0.3	0.6	mA
		5V		—	0.6	1	mA
I_{STB1}	Standby Current (*f_S=T1)	3V	No load, system HALT, LCD Off at HALT	—	—	1	μA
		5V		—	—	2	μA
I_{STB2}	Standby Current (*f_S=RTC OSC)	3V	No load, system HALT, LCD On at HALT, C type	—	2.5	5	μA
		5V		—	10	20	μA
I_{STB3}	Standby Current (*f_S=WDT OSC)	3V	No load, system HALT, LCD On at HALT, C type	—	2	5	μA
		5V		—	6	10	μA
I_{STB4}	Standby Current (*f_S=RTC OSC)	3V	No load, system HALT, LCD On at HALT, R type, 1/2 bias, V_{LCD}=V_{DD} (Low bias current option)	—	17	30	μA
		5V		—	34	60	μA
I_{STB5}	Standby Current (*f_S=RTC OSC)	3V	No load, system HALT, LCD On at HALT, R type, 1/3 bias, V_{LCD}=V_{DD} (Low bias current option)	—	13	25	μA
		5V		—	28	50	μA
I_{STB6}	Standby Current (*f_S=WDT OSC)	3V	No load, system HALT, LCD On at HALT, R type, 1/2 bias, V_{LCD}=V_{DD} (Low bias current option)	—	14	25	μA
		5V		—	26	50	μA
I_{STB7}	Standby Current (*f_S=WDT OSC)	3V	No load, system HALT, LCD On at HALT, R type, 1/3 bias, V_{LCD}=V_{DD} (Low bias current option)	—	10	20	μA
		5V		—	19	40	μA
V_{IL1}	Input Low Voltage for I/O Ports, TMR, $\overline{INT0}$, $\overline{INT1}$	—	—	0	—	0.3V_{DD}	V
V_{IH1}	Input High Voltage for I/O Ports, TMR, $\overline{INT0}$, $\overline{INT1}$	—	—	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 Voltage	—	—	2.7	3.0	3.3	V
V_{LVD}	Low Voltage Detector Voltage	—	—	3.0	3.3	3.6	V
I_{OL1}	I/O Port Segment Logic Output Sink Current	3V	V_{OL}=0.1V_{DD}	6	12	—	mA
		5V		10	25	—	mA
I_{OH1}	I/O Port Segment Logic Output Source Current	3V	V_{OH}=0.9V_{DD}	–2	–4	—	mA
		5V		–5	–8	—	mA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
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 $\overline{INT0}$, $\overline{INT1}$	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ
V_{AD}	A/D Input Voltage	—	—	0	—	V_{DD}	V
E_{AD}	A/D Conversion Integral Nonlinearity Error	—	—	—	±0.5	±1	LSB
I_{ADC}	Additional Power Consumption if A/D Converter is Used	3V	—	—	0.5	1	mA
		5V		—	1.5	3	mA

Note: "*f_S" please refer to clock option of Watchdog Timer

A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS1}	System Clock	—	2.2V~5.5V	400	—	4000	kHz
		—	3.3V~5.5V	400	—	8000	kHz
f_{SYS2}	System Clock (32768Hz Crystal OSC)	—	2.2V~5.5V	—	32768	—	Hz
f_{RTCOSC}	RTC Frequency	—	—	—	32768	—	Hz
f_{TIMER}	Timer I/P Frequency	—	2.2V~5.5V	0	—	4000	kHz
		—	3.3V~5.5V	0	—	8000	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_{LVR}	Low Voltage Width to Reset	—	—	1	—	—	ms
t_{INT}	Interrupt Pulse Width	—	—	1	—	—	μs
t_{AD}	A/D Clock Period	—	—	1	—	—	μs
t_{ADC}	A/D Conversion Time	—	—	—	76	—	t_{AD}
t_{ADCS}	A/D Sampling Time	—	—	—	32	—	t_{AD}

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

Functional Description

Execution Flow

The system clock is derived from either a crystal or an RC oscillator or a 32768Hz crystal 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.

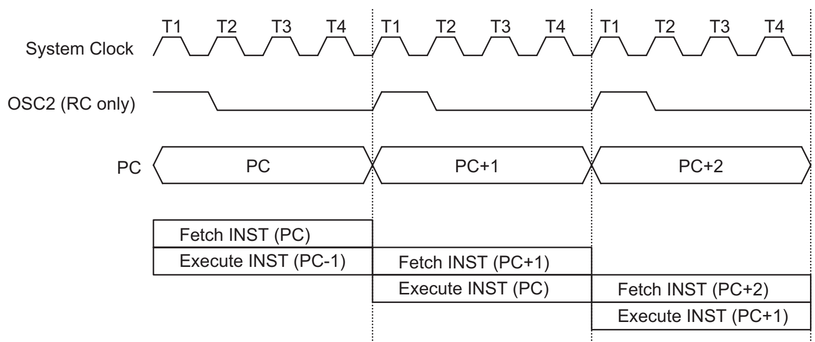
Program Counter – PC

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.



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	0	1	0	0
External Interrupt 1	0	0	0	0	0	0	0	1	0	0	0
Timer/Event Counter Overflow	0	0	0	0	0	0	0	1	1	0	0
Time Base Interrupt	0	0	0	0	0	0	1	0	1	0	0
RTC Interrupt	0	0	0	0	0	0	1	1	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

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.

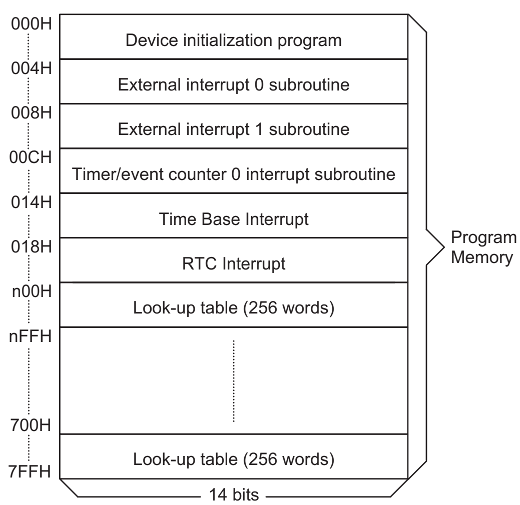
When a control transfer takes place, an additional dummy cycle is required.

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 $\overline{INT0}$ input pin is activated, and the interrupt is enabled, and the stack is not full, the program begins execution at location 004H.



Note: n ranges from 0 to 7

Program Memory

- Location 008H
  Location 008H is reserved for the external interrupt service program also. If the $\overline{INT1}$ input pin is activated, and 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 interrupt service program. If a timer interrupt results from a Timer/Event Counter overflow, and if the interrupt is enabled and the stack is not full, the program begins execution at location 00CH.
- Location 014H
  Location 014H is reserved for the Time Base interrupt service program. If a Time Base interrupt occurs, and the interrupt is enabled, and the stack is not full, the program begins execution at location 014H.
- Location 018H
  Location 018H is reserved for the real time clock interrupt service program. If a real time clock interrupt occurs, and the interrupt is enabled, and the stack is not full, the program begins execution at location 018H.
- 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 and the remaining 1 bit is read as "0". 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.

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

Stack Register – STACK

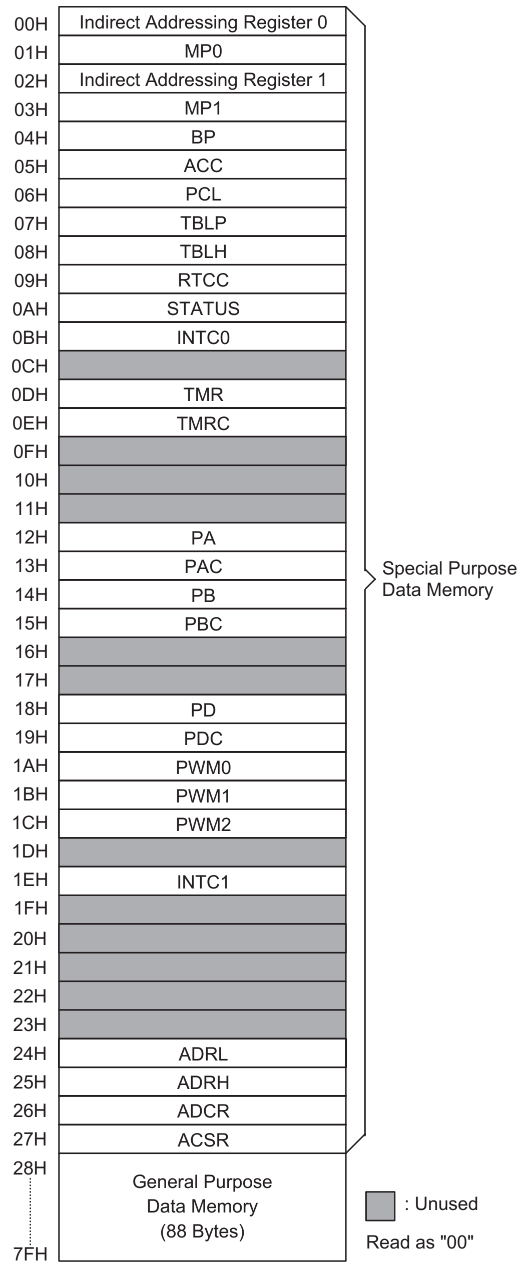
The stack register is a special part of the memory used to save the contents of the program counter. The stack is organized into 6 levels and is neither part of the data nor part of the program, and is neither readable nor writeable. Its activated level is indexed by a stack pointer (SP) and is neither 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 sixteen return addresses are stored).

Data Memory – RAM

The data memory (RAM) is designed with 116×8 bits, and is divided into two functional groups, namely; special function registers 28×8 bit and general purpose data memory, 88×8 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 Real time clock control register (RTCC;09H), a Status register (STATUS;0AH), an Interrupt control register 0 (INTC0;0BH), Interrupt control register 1 (INTC1;1EH) , PWM data register (PWM0;1AH, PWM1;1BH, PWM2;1CH), the A/D result lower-order byte register (ADRL;24H), the A/D result higher-order byte register (ADRH;25H), the A/D control register (ADCR;26H), the A/D clock setting register (ACSR;27H), I/O registers (PA;12H, PB;14H, PD;18H) and I/O control registers (PAC;13H, PBC;15H, PDC;19H). The space before 28H is overlapping in each bank. The general purpose data memory, addressed from 28H to 7FH, 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/MP1;03H). The space before 28H is overlapping in each bank.



RAM Mapping

Indirect Addressing Register

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

The 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. The bit 7 of MP0 and MP1 are always ″1″. 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 two external interrupts, one internal timer/event counter interrupts, an internal time base interrupt, and an internal real time clock 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.

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

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 are triggered by a an edge transition of $\overline{\text{INT0}}$ or $\overline{\text{INT1}}$ (option: high to low, low to high, low to high or high to low), and the related interrupt request flag (EIF0; bit 4 of INTC0, EIF1; bit 5 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 or 08H occurs. The interrupt request flag (EIF0 or EIF1) and EMI bits are all cleared to disable other maskable interrupts.

The internal Timer/Event Counter interrupt is initialized by setting the Timer/Event Counter interrupt request flag (TF; bit 6 of INTC0), which is normally caused by a timer overflow. After the interrupt is enabled, and the stack is not full, and the TF bit is set, a subroutine call to location 0CH occurs. The related interrupt request flag (TF) is reset, and the EMI bit is cleared to disable further interrupts.

The time base interrupt is initialized by setting the time base interrupt request flag (TBF; bit 5 of INTC1), that is caused by a regular time base signal. After the interrupt is enabled, and the stack is not full, and the TBF bit is set, a subroutine call to location 14H occurs. The related interrupt request flag (TBF) is reset and the EMI bit is cleared to disable further maskable interrupts.

The real time clock interrupt is initialized by setting the real time clock interrupt request flag (RTF; bit 6 of INTC1), that is caused by a regular real time clock signal. After the interrupt is enabled, and the stack is not full, and the RTF bit is set, a subroutine call to location 18H occurs. The related interrupt request flag (RTF) is reset and the EMI bit is cleared to disable further interrupts.

During the execution of an interrupt subroutine, other interrupt acknowledgments are all held until the ″RETI″ instruction is executed or the EMI bit and the related in-

Bit No.	Label	Function
0	EMI	Control the master (global) interrupt (1=enabled; 0=disabled)
1	EEI0	Control the external interrupt 0 (1=enabled; 0=disabled)
2	EEI1	Control the external interrupt 1 (1=enabled; 0=disabled)
3	ETI	Control the Timer/Event Counter interrupt (1=enabled; 0=disabled)
4	EIF0	External interrupt 0 request flag (1=active; 0=inactive)
5	EIF1	External interrupt 1 request flag (1=active; 0=inactive)
6	TF	Internal Timer/Event Counter request flag (1=enable; 0=disable)
7	—	For test mode used only. Must be written as ″0″; otherwise may result in unpredictable operation.

INTC0 (0BH) Register

Bit No.	Label	Function
0	—	Unused bit, read as ″0″
1	ETBI	Control the time base interrupt (1=enabled; 0:disabled)
2	ERTI	Control the real time clock interrupt (1=enabled; 0:disabled)
3, 4	—	Unused bit, read as ″0″
5	TBF	Time base request flag (1=active; 0=inactive)
6	RTF	Real time clock request flag (1=active; 0=inactive)
7	—	Unused bit, read as ″0″

INTC1 (1EH) Register

terrupt 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
External interrupt 1	2	08H
Timer/Event Counter overflow	3	0CH
Time base interrupt	4	14H
Real time clock interrupt	5	18H

The Timer/Event Counter interrupt request flag (TF), external interrupt 1 request flag (EIF1), external interrupt 0 request flag (EIF0), enable Timer/Event Counter interrupt bit (ETI), 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 real time clock interrupt request flag (RTF), time base interrupt request flag (TBF), enable real time clock interrupt bit (ERTI), and enable time base interrupt bit (ETBI), on the other hand, constitute the Interrupt Control register 1 (INTC1) which is located at 1EH in the RAM. EMI, EEI0, EEI1, ETI, ET1I, ETBI and ERTI 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 (RTF, TBF, TF, 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.

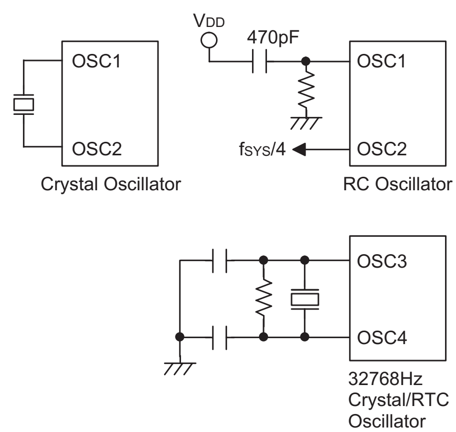
Oscillator Configuration

The device provides three oscillator circuits for system clocks, i.e., RC oscillator, crystal oscillator and 32768Hz crystal oscillator, determined by options. No matter what type of oscillator is selected, the signal is used for the system clock. The HALT mode stops the system oscillator (RC and crystal oscillator only) and ignores external signal in order to conserve power. The 32768Hz crystal oscillator still runs at HALT mode. If the 32768Hz crystal oscillator is selected as the system oscillator, the system oscillator is not stopped; but the instruction execution is stopped. Since the 32768Hz oscillator is also designed for timing purposes, the internal timing (RTC, time base, WDT) operation still runs even if the system enters the HALT mode.

Of the three oscillators, if the RC oscillator is used, an external resistor between OSC1 and VSS is required, and the range of the resistance should be from 30kΩ to 750kΩ. The system clock, divided by 4, is available on OSC2 with pull-high resistor, which can be used to synchronize external logic. The RC oscillator provides the most cost effective solution. However, the frequency of the oscillation may vary with VDD, temperature, and the chip itself due to process variations. It is therefore, not suitable for timing sensitive operations where accurate oscillator frequency is desired.

On the other hand, if the crystal oscillator is selected, a crystal across OSC1 and OSC2 is needed to provide the feedback and phase shift required for the oscillator, and no other external components are required. A resonator may be connected between OSC1 and OSC2 to replace the crystal and to get a frequency reference, but two external capacitors in OSC1 and OSC2 are required.

There is another oscillator circuit designed for the real time clock. In this case, only the 32.768kHz crystal oscillator can be applied. The crystal should be connected between OSC3 and OSC4.



System Oscillator

Note: 32768Hz crystal enable condition: For WDT clock source or for system clock source.

The external resistor and capacitor components connected to the 32768Hz crystal are not necessary to provide oscillation. For applications where precise RTC frequencies are essential, these components may be required to provide frequency compensation due to different crystal manufacturing tolerances.

The RTC oscillator circuit can be controlled to oscillate quickly by setting the ″QOSC″ bit (bit 4 of RTCC). It is recommended to turn on the quick oscillating function upon power on, and then turn it off after 2 seconds.

The WDT oscillator is a free running on-chip RC oscillator, and no external components are required. Although the system enters the power down mode, the system clock stops, and the WDT oscillator still works with a period of approximately 65μs@5V. The WDT oscillator can be disabled by options to conserve power.

Watchdog Timer – WDT

The WDT clock source is implemented by a dedicated RC oscillator (WDT oscillator) or an instruction clock (system clock/4) or a real time clock oscillator (RTC oscillator). 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 options. But if the WDT is disabled, all executions related to the WDT lead to no operation.

Once an internal WDT oscillator (RC oscillator with period 65μs@5V normally) is selected, it is divided by 2^{12}~2^{15} (by option to get the WDT time-out period). The minimum period of WDT time-out period is about 300ms~600ms. This time-out period may vary with temperature, VDD and process variations. By selection the WDT option, longer time-out periods can be realized. If the WDT time-out is selected 2^{15}, the maximum time-out period is divided by 2^{15}~2^{16}about 2.1s~4.3s. If the WDT oscillator is disabled, the WDT clock may still come from the instruction clock and operate in the same manner except that in the halt state 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 (WDT OSC) is strongly recommended, since the HALT will stop the system clock.

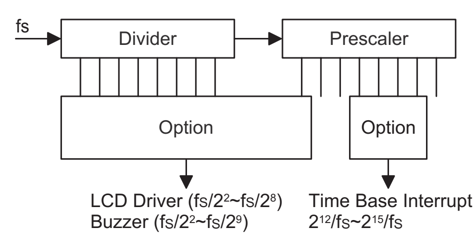
The WDT overflow under normal operation initializes a ″chip reset″ and sets the status bit ″TO″. In the HALT mode, the overflow initializes a ″warm reset″, and only the program counter and SP are reset to zero. To clear the contents of the WDT, there are three methods to be adopted, i.e., external reset (a low level to $\overline{RES}$), software instruction, and a ″HALT″ instruction. There are two types of software instructions; ″CLR WDT″ and the other set – ″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 options – ″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. In the case that ″CLR WDT1″ and ″CLR WDT2″ are 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 time-out.

Multi-function Timer

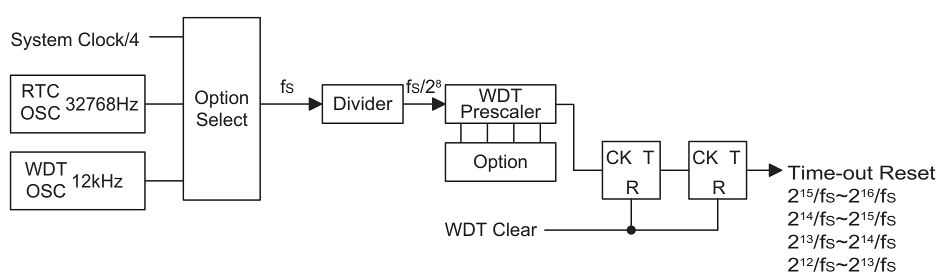
The HT46R62/HT46C62 provides a multi-function timer for the WDT, time base and RTC but with different time-out periods. The multi-function timer consists of an 8-stage divider and a 7-bit prescaler, with the clock source coming from the WDT OSC or RTC OSC or the instruction clock (i.e., system clock divided by 4). The multi-function timer also provides a selectable frequency signal (ranges from $f_S/2^2$ to $f_S/2^8$) for LCD driver circuits, and a selectable frequency signal (ranging from $f_S/2^2$ to $f_S/2^9$) for the buzzer output by options. It is recommended to select a nearly 4kHz signal for the LCD driver circuits to have proper display.

Time Base

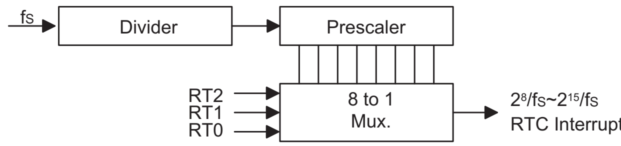
The time base offers a periodic time-out period to generate a regular internal interrupt. Its time-out period ranges from $2^{12}/f_S$ to $2^{15}f_S$ selected by options. If time base time-out occurs, the related interrupt request flag (TBF; bit 5 of INTC1) is set. But if the interrupt is enabled, and the stack is not full, a subroutine call to location 14H occurs.



Time Base



Watchdog Timer



Real Time Clock

Real Time Clock – RTC

The real time clock (RTC) is operated in the same manner as the time base that is used to supply a regular internal interrupt. Its time-out period ranges from $f_S/2^8$ to $f_S/2^{15}$ by software programming. Writing data to RT2, RT1 and RT0 (bit 2, 1, 0 of RTCC;09H) yields various time-out periods. If the RTC time-out occurs, the related interrupt request flag (RTF; bit 6 of INTC1) is set. But if the interrupt is enabled, and the stack is not full, a subroutine call to location 18H occurs.

RT2	RT1	RT0	RTC Clock Divided Factor
0	0	0	2^8*
0	0	1	2^9*
0	1	0	2^{10}*
0	1	1	2^{11}*
1	0	0	2^{12}
1	0	1	2^{13}
1	1	0	2^{14}
1	1	1	2^{15}

Note: ″*″ not recommended to be used

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 WDT oscillator keeps running (if the WDT oscillator or the real time clock 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 WDT oscillator or the real time clock oscillator).
- All I/O ports maintain their original status.
- The PDF flag is set but the TO flag is cleared.
- LCD driver is still running (if the WDT OSC or RTC OSC is selected).

The system quits the HALT mode by an external reset, an interrupt, an external falling edge signal on port A, or a WDT overflow. An external reset causes device initialization, 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 wake-up events occur, it takes 1024 t_{SYS} (system clock period) 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 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 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. Examining the PDF and TO flags, the program can distinguish between different ″chip resets″.

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

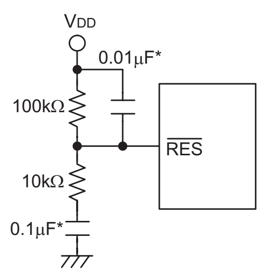
Note: ″u″ stands for unchanged

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.

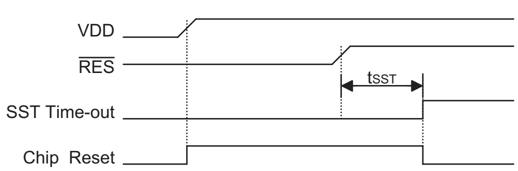
The functional unit chip reset status is shown below.

Program Counter	000H
Interrupt	Disabled
Prescaler, Divider	Cleared
WDT, RTC, Time Base	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

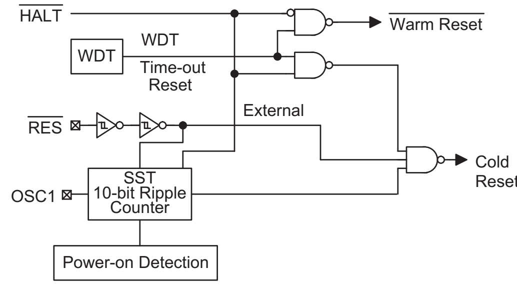


Reset Circuit

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



Reset Timing Chart



Reset Configuration

Timer/Event Counter

One timer/event counters (TMR) are implemented in the microcontroller. The Timer/Event Counter 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}. 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; TMR ([0DH]) and TMRC ([0EH]). Two physical registers are mapped to TMR location; writing TMR puts the starting value in the Timer/Event Counter register and reading TMR takes the contents of the Timer/Event Counter. The TMRC is a timer/event counter control register, which defines some options counting enable or disable and an active edge.

The TM0 and TM1 bits define the operation mode. The event count mode is used to count external events, which means that the clock source is from an external (TMR) 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 the high or low level duration of the external signal (TMR), and the counting is based on the internal selected clock source.

In the event count or timer mode, the timer/event counter starts counting at the current contents in the timer/event counter and ends at FFH. Once an overflow occurs, the counter is reloaded from the timer/event counter preload register, and generates an interrupt request flag (TF; bit 6 of INTC0). In the pulse width measurement mode with the values of the TON and TE bits equal to 1, after the TMR has received a transient from low to high (or high to low if the TE bit is ″0″), it will start counting until the TMR returns to the original level and resets the TON. The measured result remains in the timer/event counter even if the activated transient occurs again. In other words, only 1-cycle measurement can be made until the TON is set. The cycle measurement will 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 over-

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
RTCC	--00 0111	--00 0111	--00 0111	--00 0111	--uu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC0	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
TMR	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMRC	00-0 1000	00-0 1000	00-0 1000	00-0 1000	uu-u 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 1111	--11 1111	--11 1111	--11 1111	--uu uuuu
PBC	--11 1111	--11 1111	--11 1111	--11 1111	--uu uuuu
PD	-111 -111	-111 -111	-111 -111	-111 -111	-uuu -uuu
PDC	-111 -111	-111 -111	-111 -111	-111 -111	-uuu -uuu
PWM0	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PWM1	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PWM2	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
INTC1	-00- -00-	-00- -00-	-00- -00-	-00- -00-	-uu- -uu-
ADRL	x--- ----	x--- ----	x--- ----	x--- ----	u--- ----
ADRH	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCR	0100 0000	0100 0000	0100 0000	0100 0000	uuuu uuuu
ACSR	1--- --00	1--- --00	1--- --00	1--- --00	1--- --uu

Note: "*" stands for warm reset

"u" stands for unchanged

"x" stands for unknown

flows, 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.

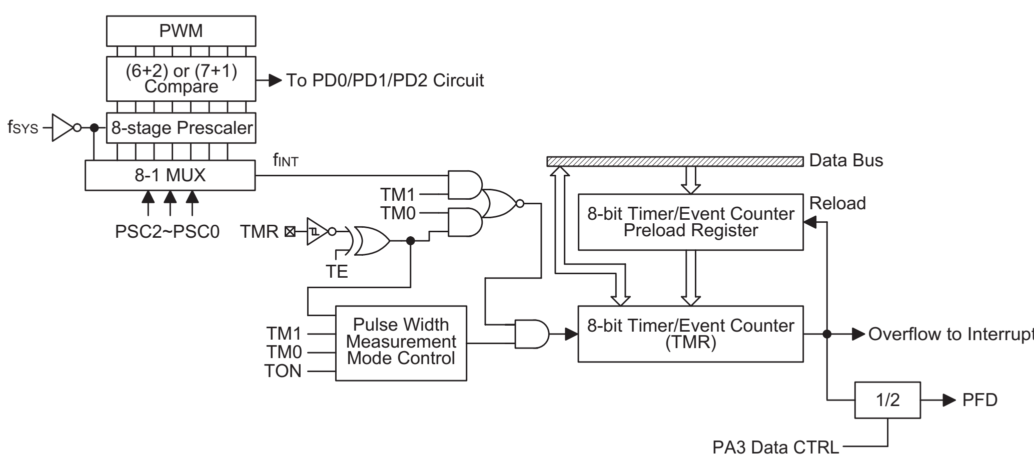
To enable the counting operation, the Timer ON bit (TON; bit 4 of TMRC) should be set to 1. In the pulse width measurement mode, the TON is automatically cleared after the measurement cycle is completed. But in the other two modes, the TON can only be reset by instructions. The overflow of the Timer/Event Counter is one of the wake-up sources and can also be applied to a PFD (Programmable Frequency Divider) output at PA3 by options. Only one PFD can be applied to PA3 by options . No matter what the operation mode is, writing a 0 to ETI disables the related interrupt service. When the PFD function is selected, executing ″SET [PA].3″ instruction to enable PFD output and executing ″CLR [PA].3″ instruction to disable PFD output.

In the case of 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 turn 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 TMR) is read, the clock is blocked to avoid errors, as this may results

Bit No.	Label	Function
0 1 2	PSC0 PSC1 PSC2	To define the prescaler stages. PSC2, PSC1, PSC0= 000: $f_{INT}=f_{SYS}$ 001: $f_{INT}=f_{SYS}/2$ 010: $f_{INT}=f_{SYS}/4$ 011: $f_{INT}=f_{SYS}/8$ 100: $f_{INT}=f_{SYS}/16$ 101: $f_{INT}=f_{SYS}/32$ 110: $f_{INT}=f_{SYS}/64$ 111: $f_{INT}=f_{SYS}/128$
3	TE	Defines the TMR active edge of the timer/event counter: In Event Counter Mode (TM1,TM0)=(0,1): 1:count on falling edge; 0:count on rising edge In Pulse Width measurement mode (TM1,TM0)=(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	TON	Enable/disable timer counting (0=disabled; 1=enabled)
5	—	Unused bit, read as ″0″
6 7	TM0 TM1	Defines the operating mode (TM1, TM0) 01= Event count mode (External clock) 10= Timer mode (Internal clock) 11= Pulse Width measurement mode (External clock) 00= Unused

TMRC (0EH) Register



Timer/Event Counter

in a counting error. Blocking of the clock should be taken into account by the programmer. It is strongly recommended to load a desired value into the TMR register first, before turning on the related timer/event counter, for proper operation since the initial value of TMR is unknown. Due to the timer/event scheme, the programmer should pay special attention on the instruction to enable then disable the timer for the first time, whenever there is a need to use the timer/event function, to avoid unpredictable result. After this procedure, the timer/event function can be operated normally.

The bit0~bit2 of the TMRC can be used to define the pre-scaling stages of the internal clock sources of timer/event counter. The definitions are as shown. The overflow signal of timer/event counter can be used to generate the PFD signal. The timer prescaler is also used as the PWM counter.

Input/Output Ports

There are 20 bidirectional input/output lines in the microcontroller, labeled as PA, PB0~PB5, PD0~PD2 and PD4~PD6, which are mapped to the data memory of [12H], [14H] and [18H] 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, 14H or 18H). 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, PDC) to control the input/output configuration. With this control register, CMOS output or Schmitt Trigger input 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, 15H and 19H.

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, 14H or 18H) 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 PA3 is pin-shared with the PFD signal. If the PFD option is selected, the output signal in output mode of PA3 will be the PFD signal generated by timer/event counter overflow signal. The input mode always retain its original functions. Once the PFD option is selected, the PFD output signal is controlled by PA3 data register only. Writing ″1″ to PA3 data register will enable the PFD output function and writing 0 will force the PA3 to remain at ″0″. The I/O functions of PA3 are shown below.

I/O Mode	I/P (Normal)	O/P (Normal)	I/P (PFD)	O/P (PFD)
PA3	Logical Input	Logical Output	Logical Input	PFD (Timer on)

Note: The PFD frequency is the timer/event counter overflow frequency divided by 2.

The PA0, PA1, PA3, PD4, PD5 and PD6 are pin-shared with BZ, $\overline{BZ}$, PFD, $\overline{INT0}$, $\overline{INT1}$ and TMR pins respectively.

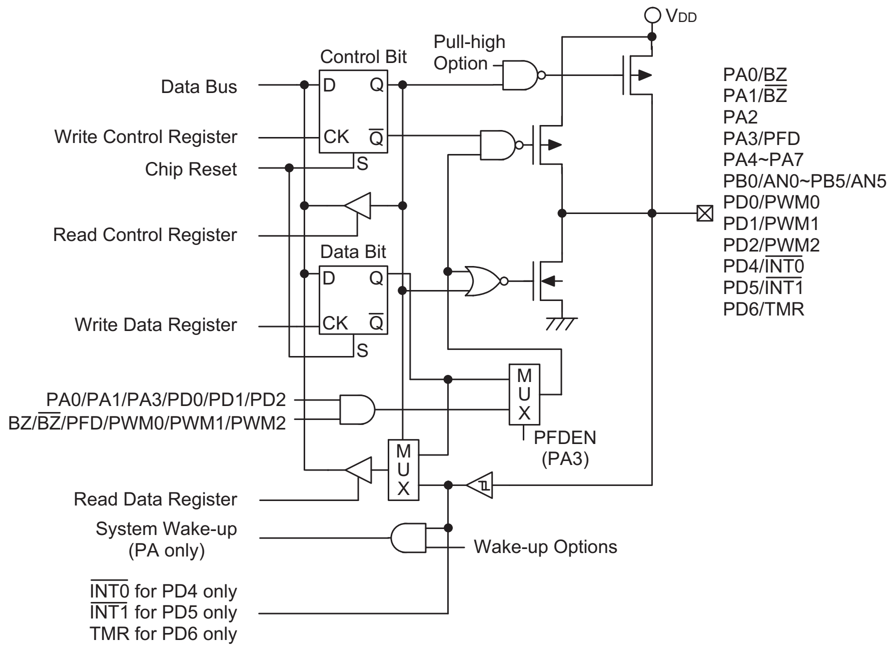
The PA0 and PA1 are pin-shared with BZ and $\overline{BZ}$ signal, respectively. If the BZ/$\overline{BZ}$ option is selected, the output signal in output mode of PA0/PA1 will be the buzzer signal generated by multi-function timer. The input mode always remain in its original function. Once the BZ/$\overline{BZ}$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

The PB can also be used as A/D converter inputs. The A/D function will be described later. There is a PWM function shared with PD0/PD1/PD2. If the PWM function is enabled, the PWM0/PWM1/PWM2 signal will appear



Input/Output Ports

on PD0/PD1/PD2 (if PD0/PD1/PD2 is operating in output mode). The I/O functions of PD0/PD1/PD2 are as shown.

I/O Mode	I/P (Normal)	O/P (Normal)	I/P (PWM)	O/P (PWM)
PD0 PD1 PD2	Logical Input	Logical Output	Logical Input	PWM0 PWM1 PWM2

It is recommended that unused or not bonded out I/O lines should be set as output pins by software instruction to avoid consuming power under input floating state.

The definitions of PFD control signal and PFD output frequency are listed in the following table.

Timer	Timer Preload Value	PA3 Data Register	PA3 Pad State	PFD Frequency
OFF	X	0	0	X
OFF	X	1	U	X
ON	N	0	0	X
ON	N	1	PFD	$f_{TMR}/[2\times(M-N)]$

Note: ″X″ stands for unused
″U″ stands for unknown
″M″ is ″256″ for PFD
″N″ is preload value for timer/event counter
″f_{TMR}″ is input clock frequency for timer/event counter

PWM

The microcontroller provides 3 channels (6+2)/(7+1) (dependent on options) bits PWM output shared with PD0/PD1/PD2. The PWM channels have their data registers denoted as PWM0 (1AH), PWM1 (1BH) and PWM2 (1CH). The frequency source of the PWM counter comes from f_{SYS}. The PWM registers are three 8-bit registers. The waveforms of PWM outputs are as shown. Once the PD0/PD1/PD2 are selected as the PWM outputs and the output function of PD0/PD1/PD2 are enabled (PDC.0/PDC.1/ PDC.2=″0″), writing ″1″ to PD0/PD1/PD2 data register will enable the PWM output function and writing ″0″ will force the PD0/PD1/PD2 to stay at ″0″.

A (6+2) bits mode PWM cycle is divided into four modulation cycles (modulation cycle 0~modulation cycle 3). Each modulation cycle has 64 PWM input clock period. In a (6+2) bit PWM function, the contents of the PWM register is divided into two groups. Group 1 of the PWM register is denoted by DC which is the value of PWM.7~PWM.2. The group 2 is denoted by AC which is the value of PWM.1~PWM.0.

In a (6+2) bits mode PWM cycle, the duty cycle of each modulation cycle is shown in the table.

Parameter	AC (0~3)	Duty Cycle
Modulation cycle i (i=0~3)	i<AC	$\frac{DC+1}{64}$
	i≥AC	$\frac{DC}{64}$

A (7+1) bits mode PWM cycle is divided into two modulation cycles (modulation cycle0~modulation cycle 1). Each modulation cycle has 128 PWM input clock period.

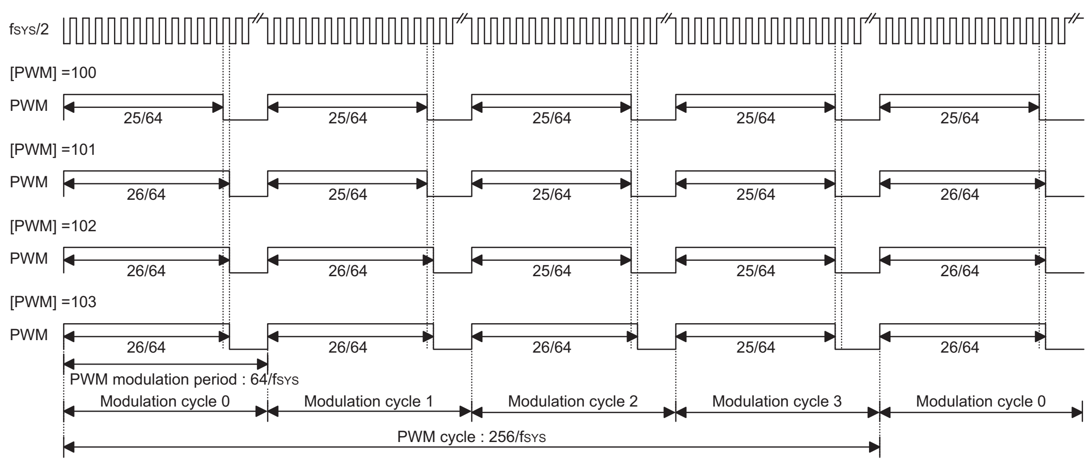
In a (7+1) bits PWM function, the contents of the PWM register is divided into two groups. Group 1 of the PWM register is denoted by DC which is the value of PWM.7~PWM.1. The group 2 is denoted by AC which is the value of PWM.0.

In a (7+1) bits mode PWM cycle, the duty cycle of each modulation cycle is shown in the table.

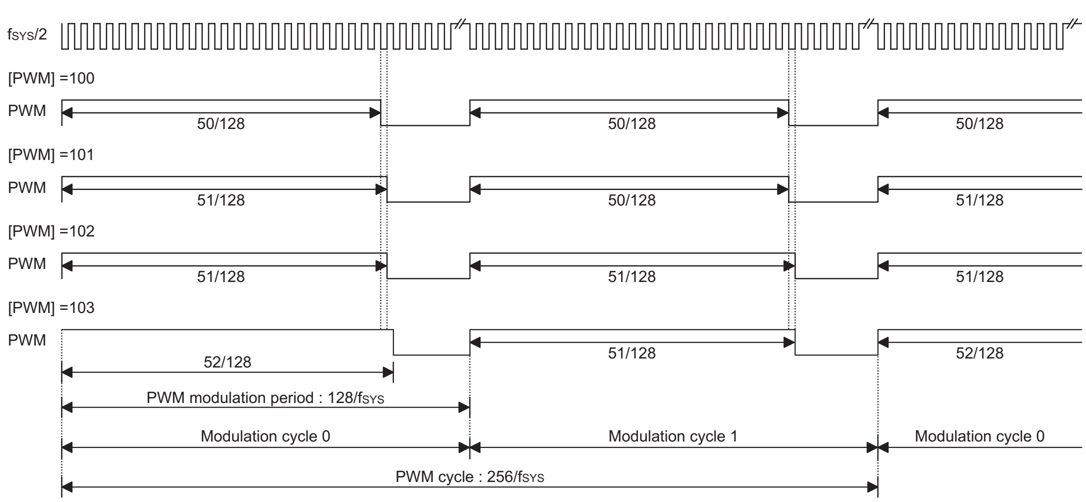
Parameter	AC (0~1)	Duty Cycle
Modulation cycle i (i=0~1)	i<AC	$\frac{DC+1}{128}$
	i≥AC	$\frac{DC}{128}$

The modulation frequency, cycle frequency and cycle duty of the PWM output signal are summarized in the following table.

PWM Modulation Frequency	PWM Cycle Frequency	PWM Cycle Duty
f_{SYS}/64 for (6+2) bits mode f_{SYS}/128 for (7+1) bits mode	f_{SYS}/256	[PWM]/256



(6+2) PWM Mode



(7+1) PWM Mode

A/D Converter

The 6 channels and 9 bits resolution A/D converter are implemented in this microcontroller. The reference voltage is VDD. The A/D converter contains 4 special registers which are; ADRL (24H), ADRH (25H), ADCR (26H) and ACSR (27H). The ADRH and ADRL are A/D result register higher-order byte and lower-order byte and are read-only. After the A/D conversion is completed, the ADRH and ADRL should be read to get the conversion result data. The ADCR is an A/D converter control register, which defines the A/D channel number, analog channel select, start A/D conversion control bit and the end of A/D conversion flag. If the users want to start an A/D conversion, define PB configuration, select the converted analog channel, and give START bit a rising edge and falling edge (0→1→0). At the end of A/D conversion, the EOCB bit is cleared. The ACSR is A/D clock setting register, which is used to select the A/D clock source.

The A/D converter control register is used to control the A/D converter. The bit2~bit0 of the ADCR are used to select an analog input channel. There are a total of six channels to select. The bit5~bit3 of the ADCR are used to set PB configurations. PB can be an analog input or as digital I/O line decided by these 3 bits. Once a PB line is selected as an analog input, the I/O functions and pull-high resistor of this I/O line are disabled and the A/D converter circuit is powered-on. The EOCB bit (bit6 of the ADCR) is end of A/D conversion flag. Check this bit to know when A/D conversion is completed. The START bit of the ADCR is used to begin the conversion of the A/D converter. Giving START bit a rising edge and falling edge means that the A/D conversion has started. In order to ensure that the A/D conversion is completed, the START should remain at ″0″ until the EOCB is cleared to ″0″ (end of A/D conversion).

Bit 7 of the ACSR register is used for test purposes only and must not be used for other purposes by the application program. Bit1 and bit0 of the ACSR register are used to select the A/D clock source.

When the A/D conversion has completed, the A/D interrupt request flag will be set. The EOCB bit is set to ″1″ when the START bit is set from ″0″ to ″1″.

Important Note for A/D initialization:
Special care must be taken to initialize the A/D converter each time the Port B A/D channel selection bits are modified, otherwise the EOCB flag may be in an undefined condition. An A/D initialization is implemented by setting the START bit high and then clearing it to zero within 10 instruction cycles of the Port B channel selection bits being modified. Note that if the Port B channel selection bits are all cleared to zero then an A/D initialization is not required.

Bit No.	Label	Function
0 1	ADCS0 ADCS1	Selects the A/D converter clock source 00= system clock/2 01= system clock/8 10= system clock/32 11= undefined
2~6	—	Unused bit, read as ″0″
7	TEST	For test mode used only

ACSR (27H) Register

Bit No.	Label	Function
0 1 2	ACS0 ACS1 ACS2	Defines the analog channel select.
3 4 5	PCR0 PCR1 PCR2	Defines the port B configuration select. If PCR0, PCR1 and PCR2 are all zero, the ADC circuit is power off to reduce power consumption
6	EOCB	Indicates end of A/D conversion. (0 = end of A/D conversion) Each time bits 3~5 change state the A/D should be initialized by issuing a START signal, otherwise the EOCB flag may have an undefined condition. See ″Important note for A/D initialization″.
7	START	Starts the A/D conversion. (0→1→0= start; 0→1= Reset A/D converter and set EOCB to ″1″)

ADCR (26H) Register

PCR2	PCR1	PCR0	7	6	5	4	3	2	1	0
0	0	0	—	—	PB5	PB4	PB3	PB2	PB1	PB0
0	0	1	—	—	PB5	PB4	PB3	PB2	PB1	AN0
0	1	0	—	—	PB5	PB4	PB3	PB2	AN1	AN0
0	1	1	—	—	PB5	PB4	PB3	AN2	AN1	AN0
1	0	0	—	—	PB5	PB4	AN3	AN2	AN1	AN0
1	0	1	—	—	PB5	AN4	AN3	AN2	AN1	AN0
1	1	0	—	—	AN5	AN4	AN3	AN2	AN1	AN0
1	1	1	—	—	AN5	AN4	AN3	AN2	AN1	AN0

Port B Configuration

ACS2	ACS1	ACS0	Analog Channel
0	0	0	AN0
0	0	1	AN1
0	1	0	AN2
0	1	1	AN3
1	0	0	AN4
1	0	1	AN5
1	1	0	AN5
1	1	1	AN5

Analog Input Channel Selection

Register	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADRL (24H)	D0	—	—	—	—	—	—	—
ADRH (25H)	D8	D7	D6	D5	D4	D3	D2	D1

Note: D0~D8 is A/D conversion result data bit LSB~MSB.

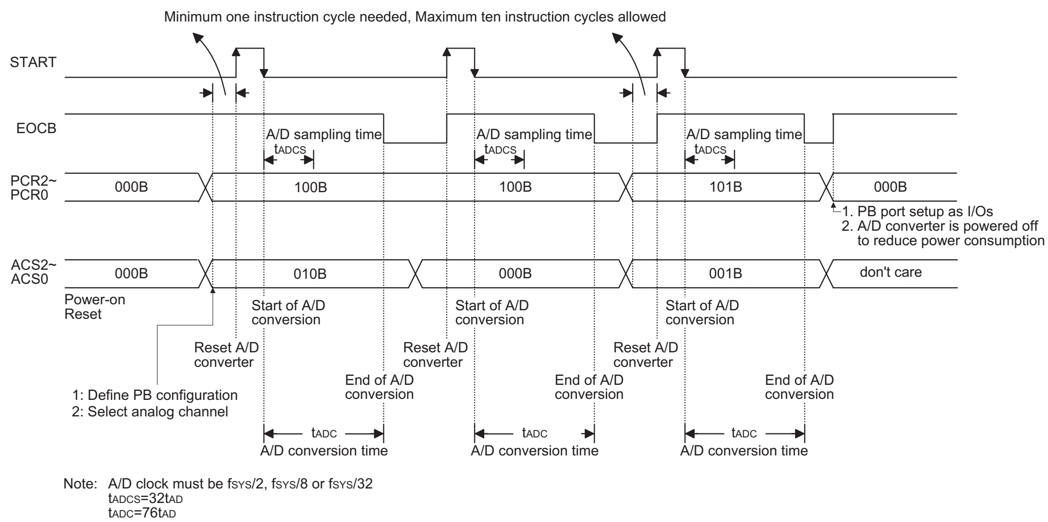
ADRL (24H), ADRH (25H) Register

The following programming example illustrates how to setup and implement an A/D conversion. The method of polling the EOCB bit in the ADCR register is used to detect when the conversion cycle is complete.

Example: using EOCB Polling Method to detect end of conversion

```
    clr     EADI                ; disable ADC interrupt
    mov     a,00000001B
    mov     ACSR,a              ; setup the ACSR register to select fSYS/8 as the A/D clock
    mov     a,00100000B         ; setup ADCR register to configure Port PB0~PB3 as A/D inputs
    mov     ADCR,a              ; and select AN0 to be connected to the A/D converter
            :
            :                   ; As the Port B channel bits have changed the following START
                                ; signal (0-1-0) must be issued within 10 instruction cycles
            :
Start_conversion:
    clr     START
    set     START               ; reset A/D
    clr     START               ; start A/D
Polling_EOC:
    sz      EOCB                ; poll the ADCR register EOCB bit to detect end of A/D conversion
    jmp     polling_EOC         ; continue polling
    mov     a,ADRH              ; read conversion result high byte value from the ADRH register
```

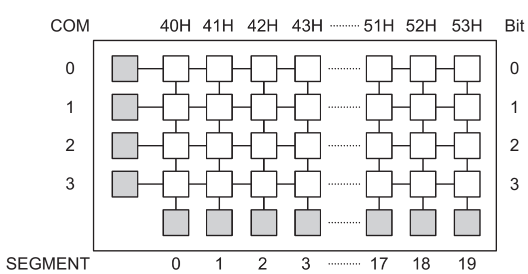
```
mov    adrh_buffer,a        ; save result to user defined memory
mov    a,ADRL               ; read conversion result low byte value from the ADRL register
mov    adrl_buffer,a        ; save result to user defined memory
       :
       :
jmp    start_conversion     ; start next A/D conversion
```



A/D Conversion Timing

LCD Display Memory

The device provides an area of embedded data memory for LCD display. This area is located from 40H to 53H 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~53H will effect the LCD display. When the BP is cleared to ″0″, any data written into 40H~53H 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.



Display Memory

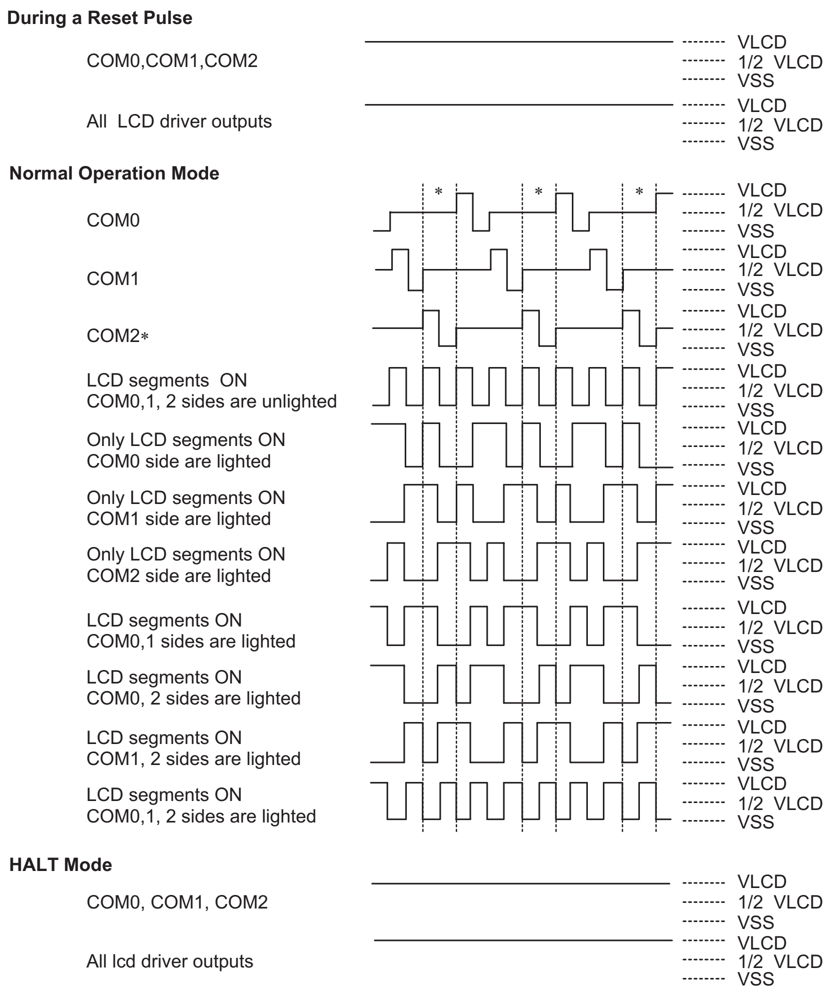
LCD Driver Output

The output number of the device LCD driver can be 20×2 or 20×3 or 19×4 by option (i.e., 1/2 duty, 1/3 duty or 1/4 duty). The bias type LCD driver can be ″R″ type or ″C″ type. If the ″R″ bias type is selected, no external capacitor is required. If the ″C″ bias type is selected, a capacitor mounted between C1 and C2 pins is needed. The LCD driver bias voltage can be 1/2 bias or 1/3 bias by option. If 1/2 bias is selected, a capacitor mounted between V2 pin and ground is required. If 1/3 bias is selected, two capacitors are needed for V1 and V2 pins. Refer to application diagram.

Condition	Option	
	Low Bias Current (Typ.)	High Bias Current (Typ.)
1/3 Bias	$(V_{LCD}/4.5)\times15\mu A$	$(V_{LCD}/4.5)\times45\mu A$
1/2 Bias	$(V_{LCD}/3)\times15\mu A$	$(V_{LCD}/3)\times45\mu A$

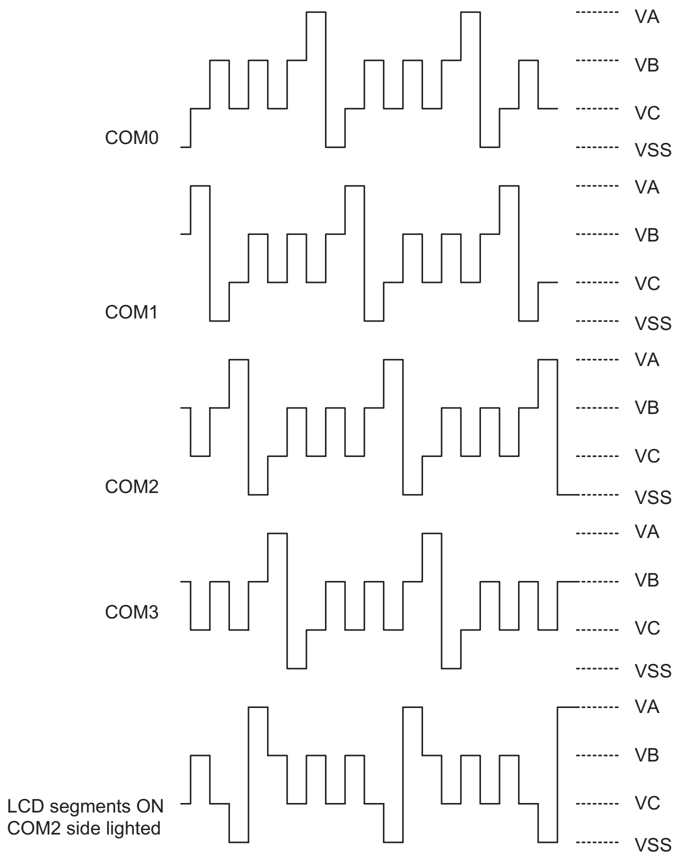
″R″ Type Bias Current

Note: The 52-pin QFP package does not support the charge pump (C type bias) of the LCD. The LCD bias type must select the R type by option.



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

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



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

LCD Driver Output

LCD Segments as Logical Output

The SEG0~SEG15 also can be optioned as logical output, once an LCD segment is optioned as a logical output, the content of bit 0 of the related segment address in LCD RAM will appear on the segment.

SEG0~SEG7 is together byte optioned as logical output, SEG8~SEG15 are bit individually optioned as logical outputs.

LCD Type	**R Type**		**C Type**	
LCD Bias Type	1/2 bias	1/3 bias	1/2 bias	1/3 bias
V_{MAX}	If V_{DD}>V_{LCD}, then V_{MAX} connect to V_{DD}, else V_{MAX} connect to V_{LCD}			If $V_{DD} > \frac{3}{2} V_{LCD}$, then V_{MAX} connect to V_{DD}, else V_{MAX} connect to V1

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 RTCC.3 to enable/disable (1/0) the LVD circuit and read the LVD detector status (0/1) from RTCC.5; otherwise, the LVD function is disabled.

The RTCC register definitions are listed below.

Bit No.	Label	Function
0~2	RT0~RT2	8 to 1 multiplexer control inputs to select the real clock prescaler output
3	LVDC	LVD enable/disable (1/0)
4	QOSC	32768Hz OSC quick start-up oscillating 0/1: quickly/slowly start
5	LVDO	LVD detection output (1/0) 1: low voltage detected, read only
6, 7	—	Unused bit, read as ″0″

RTCC (09H) Register

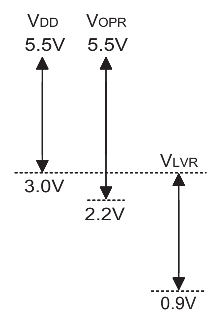
The LVR has the same effect or function with the external RES signal which performs chip reset. During HALT state, LVR is disabled both LVR and LVD are disabled.

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.

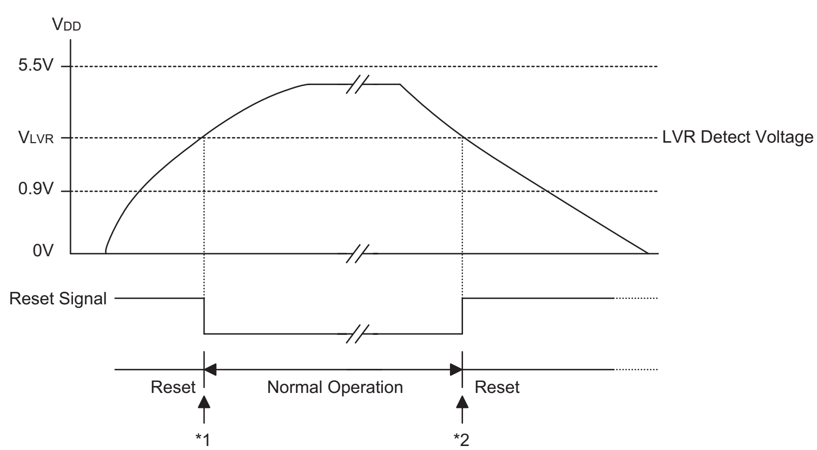
The LVR includes the following specifications:

- 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.

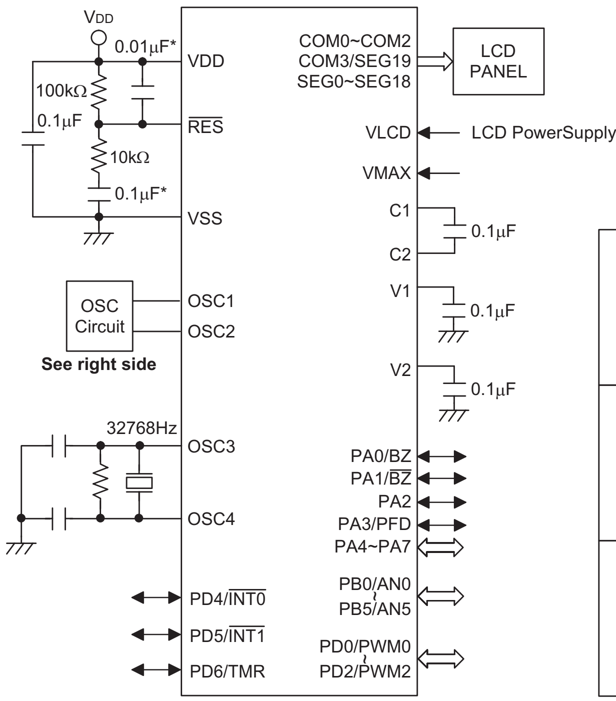
*2: Since low voltage state has to be maintained in its original state for over 1ms, therefore after 1ms delay, the device enters the reset mode.

Options

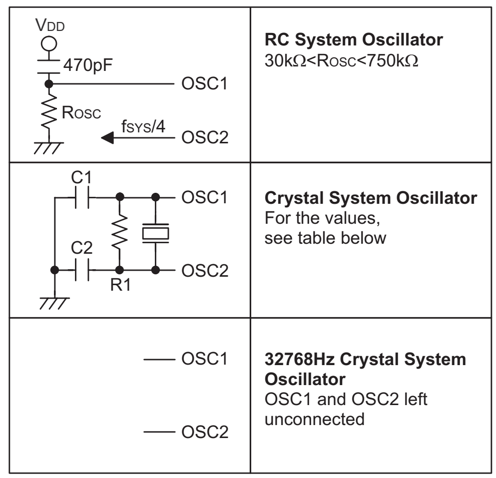
The following shows the options in the device. All these options should be defined in order to ensure proper functioning system.

Options
OSC type selection. This option is to decide if an RC or crystal or 32768Hz crystal oscillator is chosen as system clock.
WDT, RTC and time base clock source selection. There are three types of selections: system clock/4 or RTC OSC or WDT OSC.
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 $2^{12}/f_S$~$2^{13}/f_S$, $2^{13}/f_S$~$2^{14}/f_S$, $2^{14}/f_S$~$2^{15}/f_S$ or $2^{15}/f_S$~$2^{16}/f_S$.
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.
Time Base time-out period selection. The Time Base time-out period ranges from $2^{12}/f_S$ to $2^{15}/f_S$. ″f_S″ means the clock source selected by options.
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, PB and PD can be independently selected (bit option).
I/O pins share with other function selections. PA0/$\overline{BZ}$, 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) or 4 common (1/4 duty). If the 4 common is selected, the segment output pin ″SEG19″ will be set as a common output.
LCD bias power supply selection. There are two types of selections: 1/2 bias or 1/3 bias
LCD bias type selection. This option is to determine what kind of bias is selected, R type or C type.
LCD driver clock frequency selection. There are seven types of frequency signals for the LCD driver circuits: $f_S/2^2$~$f_S/2^8$. ″f_S″ stands for the clock source selection by options.
LCD ON/OFF at HALT selection.
LCD Segments as logical output selection, (byte, bit, bit, bit, bit, bit, bit, bit, bit option) [SEG0~SEG7], SEG8, SEG9, SEG10, SEG11, SEG12, SEG13, SEG14 or SEG15
LVR selection. LVR has enable or disable options
LVD selection. LVD has enable or disable options
PFD selection. If PA3 is set as PFD output, PFD is the timer overflow signal of the Timer/Event Counter respectively.
PWM selection: (7+1) or (6+2) mode PD0: level output or PWM0 output PD1: level output or PWM1 output PD2: level output or PWM2 output
$\overline{INT0}$ or $\overline{INT1}$ triggering edge selection: disable; high to low; low to high; low to high or high to low.

Application Circuits



HT46R62/HT46C62



OSC Circuit

The following table shows the C1, C2 and R1 values corresponding to the different crystal values. (For reference only)

Crystal or Resonator	C1, C2	R1
4MHz Crystal	0pF	10kΩ
4MHz Resonator	10pF	12kΩ
3.58MHz Crystal	0pF	10kΩ
3.58MHz Resonator	25pF	10kΩ
2MHz Crystal & Resonator	25pF	10kΩ
1MHz Crystal	35pF	27kΩ
480kHz Resonator	300pF	9.1kΩ
455kHz Resonator	300pF	10kΩ
429kHz Resonator	300pF	10kΩ
The function of the resistor R1 is to ensure that the oscillator will switch off should low voltage conditions occur. Such a low voltage, as mentioned here, is one which is less than the lowest value of the MCU operating voltage. Note however that if the LVR is enabled then R1 can be removed.		

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

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

"VMAX" connect to VDD or VLCD or V1 refer to the table.

LCD Type	R Type		C Type	
LCD bias type	1/2 bias	1/3 bias	1/2 bias	1/3 bias
VMAX	If V_{DD}>V_{LCD}, then VMAX connect to V_{DD}, else VMAX connect to V_{LCD}			If V_{DD} > 3/2V_{LCD}, then VMAX connect to V_{DD}, else VMAX connect to V1

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 ← 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] ← 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 ← 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 ↔ [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 ← [m].7~[m].4
ACC.7~ACC.4 ← [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.

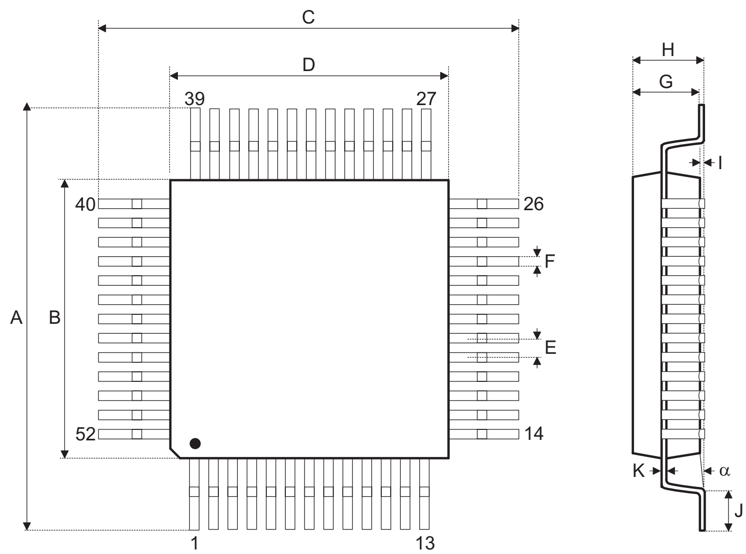
Operation: ACC ← ACC ″XOR″ x

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

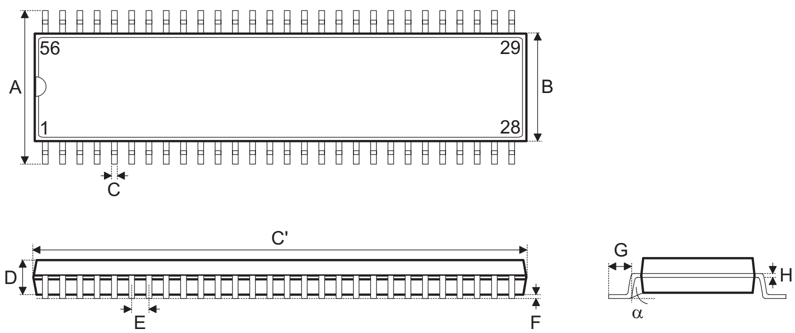
Package Information

52-pin QFP (14×14) Outline Dimensions



Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	17.3	—	17.5
B	13.9	—	14.1
C	17.3	—	17.5
D	13.9	—	14.1
E	—	1	—
F	—	0.4	—
G	2.5	—	3.1
H	—	—	3.4
I	—	0.1	—
J	0.73	—	1.03
K	0.1	—	0.2
α	0°	—	7°

56-pin SSOP (300mil) Outline Dimensions



Symbol	Dimensions in mil		
	Min.	Nom.	Max.
A	395	—	420
B	291	—	299
C	8	—	12
C′	720	—	730
D	89	—	99
E	—	25	—
F	4	—	10
G	25	—	35
H	4	—	12
α	0°	—	8°

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)
46712 Fremont Blvd., Fremont, CA 94538
Tel: 510-252-9880
Fax: 510-252-9885
http://www.holmate.com