HT49RV3/HT49CV3
VFD Type 8-Bit MCU

Technical Document

- Tools Information
- FAQs
- Application Note
  - HA0077E HT49CVX Remote Control Receiver SWIP Design Note
  - HA0078E HT49CVX Display SWIP Design Note

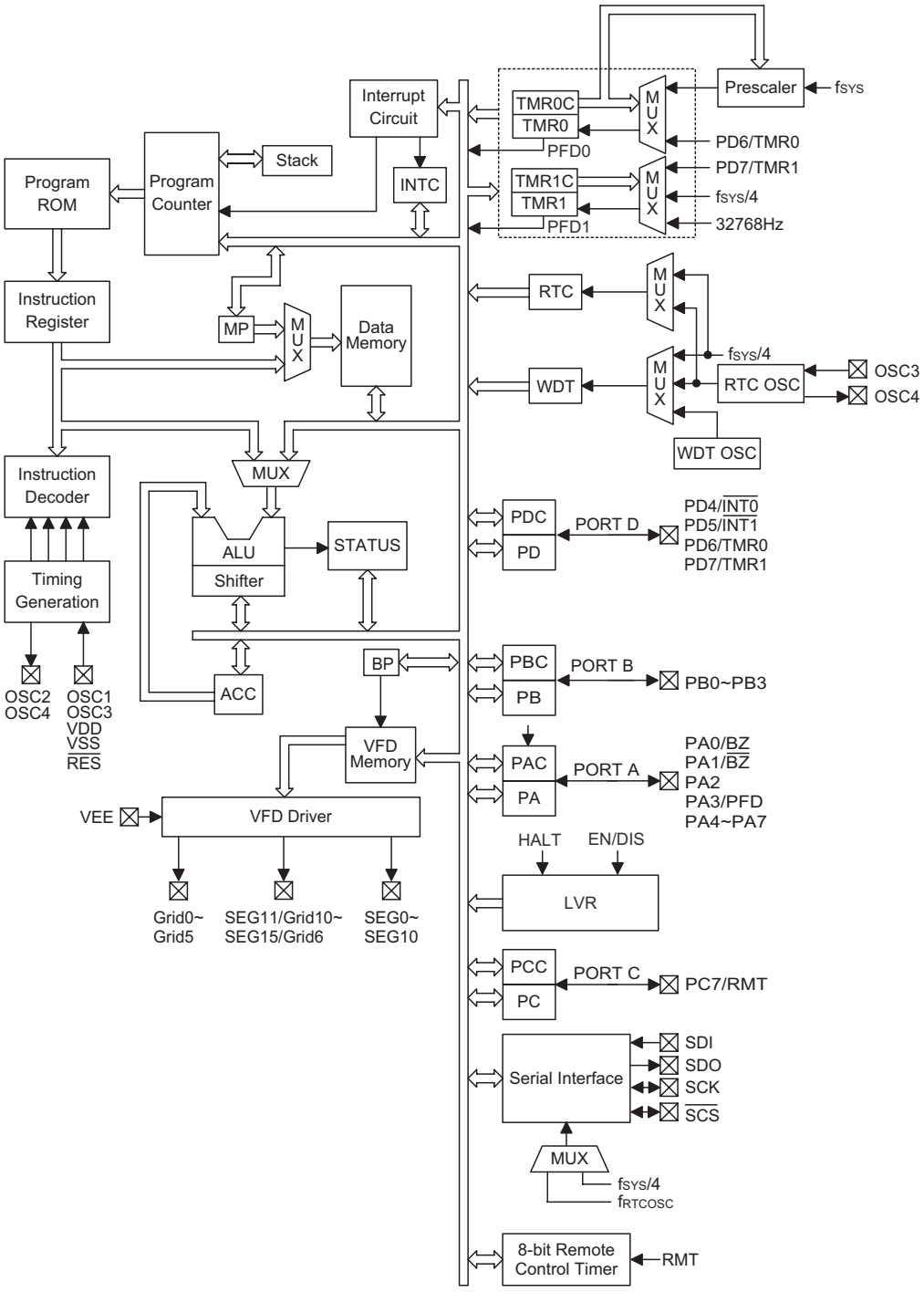
Features

- Operating voltage:
  f_{SYS}=4MHz: 2.2V~5.5V
  f_{SYS}=8MHz: 3.3V~5.5V
- 17 bidirectional I/O lines (PA, PB, PC, PD)
- Two external interrupt inputs
- Two 16-bit programmable timer/event counters with PFD (programmable frequency divider) function
- One 8-bit Remote Control Timer (RMT), pin-shared with PC7
- Single channel serial interface
- VFD driver with 11×11 segments
  (16-segment & 4-grid to 11-segment & 11-grid)
- 2K×16 program memory
- 96×8 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
- Low voltage reset function
- 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
- 52-pin QFP package

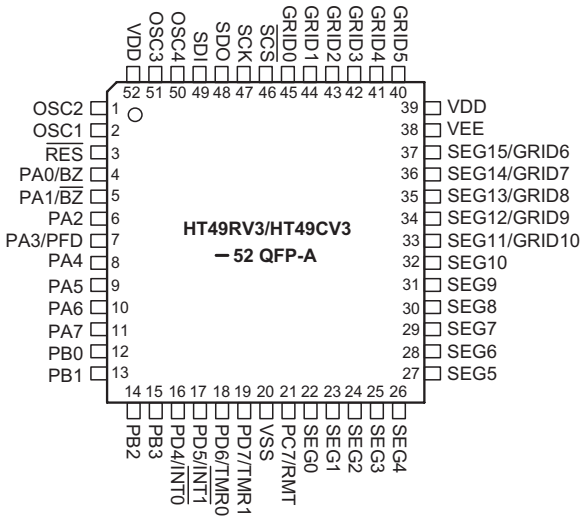
General Description

The HT49RV3/HT49CV3 are 8-bit high performance single chip MCUs. Their single cycle instruction and 2-stage pipeline architecture make them suitable for high speed applications. As the devices include an VFD driver they are suitable for use in products which require a front panel for their operation such as DVDs, VCDs, mini-component audio systems, cassette decks, tuners, CD players, other home appliances, etc.

Block Diagram



Pin Assignment



Note: Each V_{DD} (V_{SS}) pins must be connected to the power (ground) of the system.

Pin Description

Pin Name	I/O	Options	Description
PA0/BZ PA1/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 a wake-up input by configuration option. Software instructions determine if the pin is a CMOS output or Schmitt trigger input with or without pull-high resistor (determined by pull-high option: bit option). Pins PA0, PA1 and PA3 are pin-shared with BZ, BZ and PFD, respectively.
PB0~PB3	I/O	Pull-high	Bidirectional 4-bit input/output port. Software instructions determine if the pin is a CMOS output or Schmitt trigger input with or without pull-high resistor (determined by pull-high option: bit option).
PC7/RMT	I/O	Pull-high	Bidirectional 1-bit input/output port. Software instructions determine if the pin is a CMOS output or Schmitt trigger input with or without pull-high resistor (determined by pull-high option: bit option). RMT with wake-up function (both rising and falling edge) and Schmitt trigger input with or without a pull-high resistor (determined by pull-high option). Note: The RMT is pin-shared with PC7. When this pin is used as an RMT input, the pin should be setup as an input, to prevent the I/O pin influencing the operation of the RMT.
PD4/INT0 PD5/INT1 PD6/TMR0 PD7/TMR1	I/O	Pull-high	Bidirectional 4-bit input/output port. Software instructions determine if the pin is a CMOS output or Schmitt trigger input with or without pull-high resistor (determined by pull-high option: bit option). Pins PD4~PD7 are pin-shared with INT0, INT1, TMR0 & TMR1, respectively (determined by software control).
VSS	—	—	Negative power supply, ground
VEE	—	—	VFD Negative power supply
SEG0~SEG10	O	—	High voltage segment output for VFD panel.
SEG11/Grid10~ SEG15/Grid6	O	—	High voltage output for VFD panel. These pins are selectable for segment or grid output.
Grid0~Grid5	O	—	High voltage grid output for VFD panel.
SDI	I	—	Serial interface serial data input

Pin Name	I/O	Options	Description
SDO	O	—	Serial interface serial data output
SCK	I/O	—	Serial interface serial clock input/output (initial "input").
$\overline{SCS}$	I/O	—	Serial interface chip select pin, output for master mode, input for slave mode.
OSC4 OSC3	O I	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
OSC2 OSC1	O I	Crystal or RC	OSC1 and OSC2 are connected to an RC network or a crystal (by options) for the internal system clock. For RC operation, OSC2 is an output pin 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 left floating.
$\overline{RES}$	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
I_{OL} Total 150mA
Total Power Dissipation 500mW

Storage Temperature ..... –50°C to 125°C
Operating Temperature ..... –40°C to 85°C
I_{OH} Total ..... –100mA

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

D.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	f_{SYS}=4MHz	2.2	—	5.5	V
		—	f_{SYS}=8MHz	3.3	—	5.5	V
V_{EE}	VFD Supply Voltage	—	—	0	—	V_{DD}–30	V
I_{DD1}	Operating Current (Crystal OSC)	3V	No load, VFD off, f_{SYS}=4MHz	—	2.0	3.0	mA
		5V		—	5.0	8.0	mA
I_{DD2}	Operating Current (RC OSC)	3V	No load, VFD off, f_{SYS}=4MHz	—	1.8	2.7	mA
		5V		—	4.6	7.5	mA
I_{DD3}	Operating Current (f_{SYS}=32768Hz)	3V	No load, VFD off	—	1.2	2	mA
		5V		—	4	7	mA
I_{DD4}	Operating Current (Crystal OSC)	3V	No load, VFD on, f_{SYS}=4MHz	—	3.5	4.5	mA
		5V		—	7.5	12	mA
I_{STB1}	Standby Current (*f_S=T1)	3V	No load, system HALT VFD off at HALT	—	—	1	μA
		5V		—	—	2	μA
I_{STB2}	Standby Current (*f_S=32768Hz OSC)	3V	No load, system HALT VFD off at HALT	—	4	10	μA
		5V		—	14	20	μA
V_{IL1}	Input Low Voltage for I/O Ports, TMR and $\overline{INT}$	3V	—	0	—	$0.2V_{DD}$	V
		5V					

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{IH1}	Input High Voltage for I/O Ports, TMR and $\overline{INT}$	3V	—	$0.8V_{DD}$	—	V_{DD}	V
		5V					
V_{IL2}	Input Low Voltage ($\overline{RES}$)	3V	—	0	—	$0.4V_{DD}$	V
		5V					
V_{IH2}	Input High Voltage ($\overline{RES}$)	3V	—	$0.9V_{DD}$	—	V_{DD}	V
		5V					
I_{OL}	I/O Port Sink Current	3V	$V_{OL}=0.1V_{DD}$	6	12	—	mA
		5V		10	25	—	mA
I_{OH1}	I/O Port Source Current	3V	$V_{OH}=0.9V_{DD}$	–2	–4	—	mA
		5V		–5	–8	—	mA
I_{OH2}	Grid Source Current	5V	$V_{OH}=V_{DD}-2V$	–15	—	—	mA
I_{OH3}	Segment Source Current	5V	$V_{OH}=V_{DD}-2V$	–3	—	—	mA
R_{PH}	Pull-high Resistance of I/O Ports and $\overline{INT0}$, $\overline{INT1}$	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ
R_{PL}	VFD Driver Output Pull-low Resistor	5V	—	50	100	150	kΩ
V_{LVR}	Low Voltage Reset Voltage	—	LVR enabled	2.7	3.0	3.3	V

Note: "*f_S" Refer to WDT clock option

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 (TMR0/TMR1)	—	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_{INT}	Interrupt Pulse Width	—	—	1	—	—	μs

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 ensures that instructions are effectively executed in one cycle. Exceptions to this are instructions that change the contents of the program counter, such as subroutine calls or jumps, in which case, two cycles are required to complete the instruction.

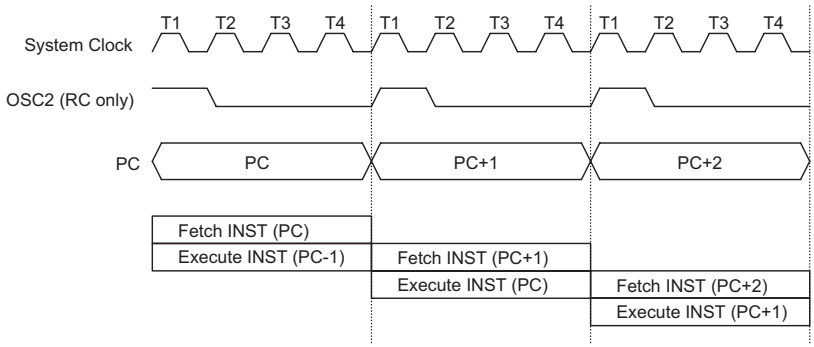
Program Counter – PC

The 11-bit program counter (PC) controls the sequence in which the instructions stored in the program ROM are executed and its contents specify a maximum of 2048 addresses.

After accessing a program memory word to fetch an instruction code, the contents of the program counter are incremented by one. The program counter then points to the memory word containing the next instruction code.

When executing a jump instruction, conditional skip execution, loading PCL register, subroutine call, initial reset, internal interrupt, external interrupt or return from subroutine, etc., the microcontroller manages program control by loading the address corresponding to each instruction.

For conditional skip instructions, once the condition has been met, the next instruction, which has already been fetched during the current instruction execution, is discarded and a dummy cycle replaces it while the proper instruction is obtained. Otherwise proceed with 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 0 Overflow	0	0	0	0	0	0	0	1	1	0	0
Timer/Event Counter 1 Overflow	0	0	0	0	0	0	1	0	0	0	0
Serial Interface Interrupt	0	0	0	0	0	0	1	0	1	0	0
Multi-function 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 program counter (PCL) is available for program control and is a readable and writeable register (06H). Moving data into the PCL performs a short jump. The destination will be within 256 locations.

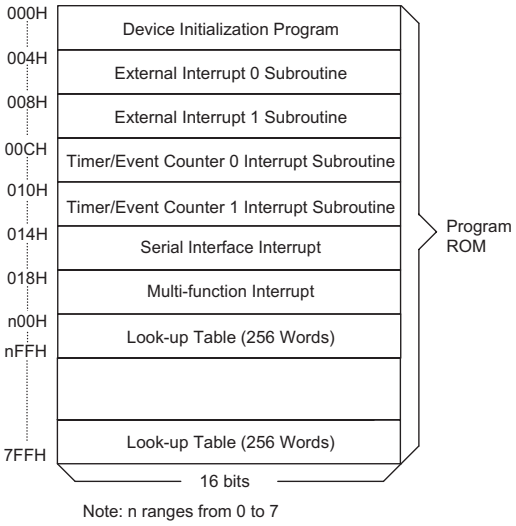
When a control transfer takes place, an additional dummy cycle is required.

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 2048×16 bits format. The program counter is composed of 11 bits, so it can directly access the whole program memory without changing banks.

Certain locations in the ROM are reserved for special usage:

- Location 000H
  This area is reserved for use by the chip reset for program initialization. After a chip reset is initiated, the program will jump to this location and begin execution.
- Location 004H
  This area 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 will jump to this location and begin execution.



Note: n ranges from 0 to 7

Program Memory

- Location 008H
  This area is reserved for the external interrupt service program. If the $\overline{INT1}$ input pin is activated, and the interrupt is enabled, and the stack is not full, the program will jump to this location and begin execution.
- Location 00CH
  This area is reserved for the Timer/Event Counter 0 interrupt service program. If a timer interrupt results from a Timer/Event Counter 0 overflow, and if the interrupt is enabled and the stack is not full, the program will jump to this location and begin execution.
- Location 010H
  This area is reserved for the Timer/Event Counter 1 interrupt service program. If a timer interrupt results from a Timer/Event Counter 1 overflow, and if the interrupt is enabled and the stack is not full, the program will jump to this location and begin execution.
- Location 014H
  This area is reserved for the Serial Interface interrupt service program. If 8 bits of data have been received or transmitted successfully from the serial interface, and the interrupt is enabled, and the stack is not full, the program will jump to this location and begin execution.
- Location 018H
  This area is reserved for the multi-function interrupt. If a real time clock interrupt occurs, or if a rising edge is detected from the RMT input pin, or if a falling edge is detected from the RMT input pin, or if the RMT overflow and the related interrupts are enabled, and the stack is not full, the program will jump to this location and begin execution.
- Table location
  Any location within the program memory can be used as a look-up table where programmers can store fixed data. The instructions "TABRDC [m]" (the current page, 1 page=256 words) and "TABRDL [m]" (the last page) transfer the contents of the lower-order byte to the specified data memory, and the contents of the higher-order byte to TBLH (Table Higher-order byte register) (08H). Only the destination of the lower-order byte in the table is well-defined, the other bits of the table word are all transferred to the lower portion of TBLH. The TBLH is a read only register and the table pointer (TBLP) is a read/write register (07H), which indicates the table location. Before accessing the table, the location must be placed in the TBLP. All table related instructions require two cycles to complete the operation. These areas may function as normal program memory depending upon 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
P10~P8: Current program counter bits
@7~@0: Table pointer bits

Stack Register – STACK

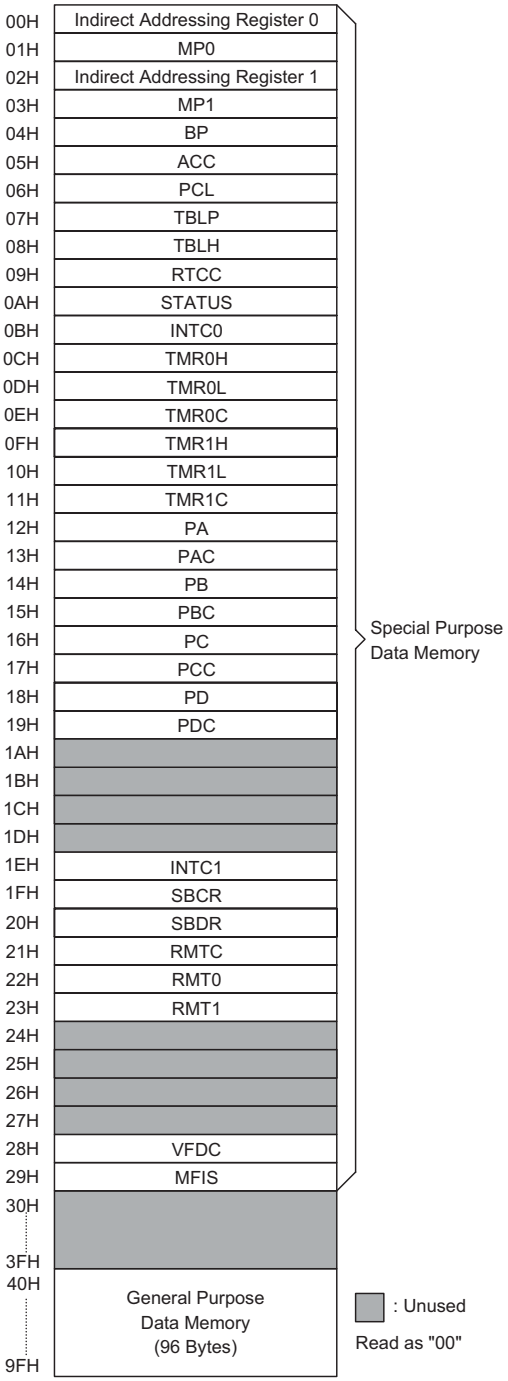
The stack register is a special part of the memory used to save the contents of the Program Counter. The stack is organized into 6 levels and is neither part of the data nor part of the program, and is neither readable nor writeable. Its activated level is indexed by a stack pointer (SP) and is neither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

Data Memory – RAM

The data memory (RAM) has a capacity of 130×8 bits, and is divided into two functional groups, namely; special function registers (34×8 bit) and general purpose data memory (RAM bank contains 96×8 bits) most of which are readable/writeable, but some are read only. The special function registers are overlapped in any banks.

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), a Timer/Event Counter 0 (TMR0H:0CH; TMR0L:0DH), a Timer/Event Counter 0 control register (TMR0C;0EH), a Timer/Event Counter 1 (TMR1H:0FH;TMR1L:10H), a Timer/Event Counter 1 control register (TMR1C; 11H), Interrupt control register 1 (INTC1;1EH), Serial Bus control register (SBCR;1FH), Serial Bus data register (SBDR; 20H), Remote timer control register (RMTC;21H), Remote control capture register 0 (RMT0;22H), Remote control capture register 1 (RMT1;23H), Multi-function interrupt Status register (MFIS;29H), VFD control register (VFDC; 28H), I/O registers (PA;12H, PB;14H, PC;16H, PD;18H) and I/O control registers (PAC;13H, PBC;15H, PCC;17H, PDC;19H).



RAM Mapping

The remaining space before 40H is reserved for future expansion usage and reading these locations will return the result "00H". The space before 40H overlaps in each bank. The general-purpose data memory, addressed from 40H to 9FH, is used for data and control information under instruction commands.

All 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).

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. Writing 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 8-bit registers used to access the RAM by combining corresponding indirect addressing registers. MP0 can only be applied to data memory, while MP1 can be applied to data memory and VFD display memory.

Accumulator – ACC

The Accumulator (ACC) is closely related with operations carried out by the ALU. It is 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 a ″HALT″ instruction. The Z, OV, AC, and C flags reflect the status of the latest operations.

On entering an interrupt sequence or executing a 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 HT49RV3/HT49CV3 provides two external interrupts, two internal timer/event counter interrupts, three remote control timer interrupts, an internal real time clock interrupt and serial interface 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, all other interrupts are 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 another interrupt requires servicing while the program is in the interrupt service routine, the programmer should set the EMI bit and the corresponding bit of the INTC0 or INTC1 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.

Bit No.	Label	Function
0	EMI	Controls the master or global interrupt (1=enable; 0=disable)
1	EEI0	Controls the external interrupt 0 (1=enable; 0=disable)
2	EEI1	Controls the external interrupt 1 (1=enable; 0=disable)
3	ET0I	Controls the Timer/Event Counter 0 interrupt (1=enable; 0=disable)
4	EIF0	External interrupt 0 request flag (1=active; 0=inactive)
5	EIF1	External interrupt 1 request flag (1=active; 0=inactive)
6	T0F	Internal Timer/Event Counter 0 request flag (1=active; 0=inactive)
7	—	Unused bit, read as ″0″

INTC0 (0BH) Register

Bit No.	Label	Function
0	ET1I	Controls the Timer/Event Counter 1 interrupt (1=enable; 0=disable)
1	ESII	Controls the serial interface interrupt (1=enabled; 0:disabled)
2	EMFI	Controls the real multi-function interrupt (1=enable; 0:disable)
3, 7	—	Unused bit, read as ″0″
4	T1F	Internal Timer/Event Counter 1 request flag (1=active; 0=inactive)
5	SIF	Serial interface data transferred or data received interrupt request flag (1=active; 0=inactive)
6	MFF	Multi-function interrupt request flag (1=active; 0=inactive)

INTC1 (1EH) Register

Bit No.	Label	Function
0	—	Unused bit, read as ″0″
1	ERMT0	Controls the remote control timer rising edge interrupt (1=enable; 0=disable)
2	ERMT1	Controls the remote control timer falling edge interrupt (1=enable; 0=disable)
3	ERMTV	Controls the remote control timer overflow interrupt (1=enable; 0=disable)
4	RME	Controls the remote control timer (1=enable; 0=disable) 1=enable & start counting; 0= disable & clear counter to 0
5	RMCS	Selects the remote control timer clock source f_X (1=f_{SYS}; 0=f_{SYS}/4)
6 7	RMS0 RMS1	Selects the remote control timer clock 00=$f_X/2^5$ 01= $f_X/2^6$ 10= $f_X/2^7$ 11= $f_X/2^8$

RMTC (21H) Register

Bit No.	Label	Function
0	RMTVF	Remote control timer overflow interrupt flag (1=indicates that an overflow has occurred; 0=indicates that an overflow has not occurred)
1	RTF	Real time clock interrupt flag (1=indicates that an RTC interrupt has occurred; 0=indicates that an RTC interrupt has not occurred)
2	RMT0F	Remote control timer rising edge interrupt flag (1=indicates that a rising edge interrupt has occurred; 0=indicates that a rising edge interrupt has not occurred)
3	RMT1F	Remote control timer falling edge interrupt flag (1=indicates that a falling edge interrupt has occurred; 0=indicates that a falling edge interrupt has not occurred)
4	ERTI	Controls the real time clock interrupt (1=enable; 0=disable)
5~7	—	Unused bit, read as ″0″

MFIS (29H) Register

External interrupts are triggered by an edge transition of $\overline{INT0}$ or $\overline{INT1}$ (configuration option: high to low, low to high, low to high or high to low), and the related interrupt request flag (EIF0; bit 4 of the INTC0, EIF1; bit 5 of the 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 0 interrupt is initialized by setting the Timer/Event Counter 0 interrupt request flag (T0F; bit 6 of the INTC0), which is normally caused by a timer overflow. After the interrupt is enabled, and the stack is not full, and the T0F bit is set, a subroutine call to location 0CH occurs. The related interrupt request flag (T0F) is reset, and the EMI bit is cleared to disable other maskable interrupts. Timer/Event Counter 1 is operated in the same manner but its related interrupt request flag is T1F (bit 4 of the INTC1) and its subroutine call location is 10H.

The Serial Interface interrupt is initialized by setting the interrupt request flag (SIF; bit 5 of the INTC1), which is caused by completely receiving or transferring 8 bits of data from a serial interface. After the interrupt is enabled, and the stack is not full, and the SIF bit is set, a subroutine call to location 14H occurs. The related interrupt request flag (SIF) is reset and the EMI bit is cleared to disable further maskable interrupts.

The multi-function interrupt is initialized by setting the interrupt request flag (MFF; bit 6 of the INTC1), which is caused by a regular real time clock signal, or caused by a rising edge of RMT, or caused by a falling edge of RMT, or caused by an RMT overflow. After the interrupt is enabled, and the stack is not full, and the MFF bit is set, a subroutine call to location 18H occurs. The related interrupt request flag (MFF) is reset and the EMI bit is cleared to disable further maskable interrupts.

During the execution of an interrupt subroutine, other maskable interrupt acknowledgments are all held until the ″RETI″ instruction is executed or the EMI bit and the related interrupt control bit are both set 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 following table shows the priority that is applied. 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 0 overflow	3	0CH
Timer/Event Counter 1 overflow	4	10H
Serial Interface interrupt	5	14H
Multi-function interrupt	6	18H

The RMT overflow interrupt flag (RMTVF; bit 0 of the MFIS), real time clock interrupt flag (RTF; bit 1 of the MFIS), the RMT rising edge interrupt flag (RMT0F; bit 2 of the MFIS) and the RMT falling edge interrupt flag (RMT1F; bit 3 of the MFIS) indicate that a related interrupt has occurred. After reading these flags, these flags will not be cleared automatically, they should be cleared by the user.

The serial interface interrupt is indicated by the interrupt flag (SIF; bit 5 of the INTC1), that is caused by receiving or transferring a complete 8-bit data transfer between the HT49RV3/HT49CV3 and an external device. After the interrupt is enabled (by setting ESII; bit 1 of the INTC1), and the stack is not full, a subroutine call to location 14H occurs.

The Timer/Event Counter 0 interrupt request flag (T0F), external interrupt 1 request flag (EIF1), external interrupt 0 request flag (EIF0), enable Timer/Event Counter0 interrupt bit (ET0I), enable external interrupt 1 bit (EEI1), enable external interrupt 0 bit (EEI0), and enable master

interrupt bit (EMI) constitute the Interrupt Control register 0 (INTC0) which is located at 0BH in the RAM.

The multi-function interrupt request flag (MFF), serial interface interrupt request flag (SIF), Timer/Event Counter 1 interrupt request flag (T1F), enable multi-function interrupt (EMFI), enable serial interface interrupt bit (ESII), and enable Timer/Event Counter 1 interrupt bit (ET1I), constitute the Interrupt Control register 1 (INTC1) which is located at 1EH in the RAM.

The enable Remote control timer rising edge interrupt (ERMT0), enable Remote control timer falling edge interrupt (ERMT1), enable Remote control timer overflow interrupt (ERMTV), enable Remote control timer start counting (RME), select the Remote control timer clock source (RMCS), and select the Remote control timer clock (RMS0, RMS1) constitute the Remote Timer control Register (RMTC) which is located at 21H in the RAM.

EMI, EEI0, EEI1, ET0I, ET1I, ESII, ERTI, EMFI, ERMT0 and ERMT1 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 (MFF, SIF, T0F, T1F, EIF1, EIF0) are all set, they remain in the INTC0~INTC1 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. This is 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 HT49RV3/HT49CV3 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 signals so as to conserve power. The 32768Hz crystal oscillator still runs in the 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, WDT) operation still runs even if the system enters the HALT mode.

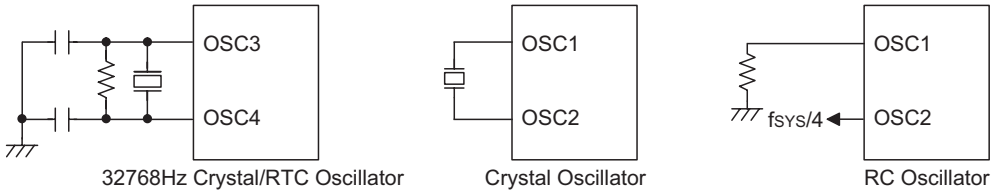
Of the three oscillators, if the RC oscillator is used, an external resistor between OSC1 and VSS is required, and the range of the resistance should be from 56kΩ to 1.5MΩ. The system clock, divided by 4, is available on OSC2 with pull-high resistor, which can be used to synchronize external logic. The 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 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 on OSC1 and OSC2 are required.

There is another oscillator circuit designed for the real time clock. In this case, only a 32768Hz crystal oscillator can be applied. The crystal should be connected between OSC3 and OSC4.

The RTC oscillator circuit can be controlled to start-up quickly by setting the "QOSC" bit (bit 4 of the RTCC). It is recommended to turn on the quick start-up function during 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 at 5V. The WDT oscillator can be disabled by options so as to conserve power.



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.

Watchdog Timer – WDT

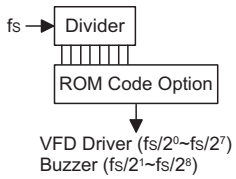
The WDT clock source is implemented by a dedicated RC oscillator (WDT oscillator) or an instruction clock (system clock/4) or a real time clock oscillator (RTC oscillator). The timer is designed to prevent a software malfunction or sequence from jumping to an unknown location with unpredictable results. 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 a period of 65μs at 5V) is selected, it is divided by 2^{12}~2^{15} (by configuration option to get the WDT time-out period). The minimum WDT time-out period is 300ms~600ms. This time-out period may vary with temperature, VDD and process variations. By selecting the WDT configuration 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} which is 2.3s~4.7s. If the WDT oscillator is disabled, the WDT clock may still come from the instruction clock and operates in the same manner except that in the halt state the WDT may stop counting and lose its protecting purposes. 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 is equal to one), any execution of the "CLR WDT" instruction clears the WDT. In the case where the two "CLR WDT1" and "CLR WDT2" instructions are chosen (i.e., CLR WDT times is equal to 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 HT49RV3/HT49CV3 provides a multi-function timer for the WDT 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 RTC OSC or the instruction clock (i.e., system clock divided by 4). The multi-function timer also provides a selectable frequency signal (ranging from $f_S/2^0$ to $f_S/2^7$) for the VFD driver circuits, and a selectable frequency signal (ranging from $f_S/2^1$ to $f_S/2^8$) for the buzzer output by options. It is recommended to select a frequency as close as possible to 32kHz for the VFD driver circuits to obtain good display clarity.

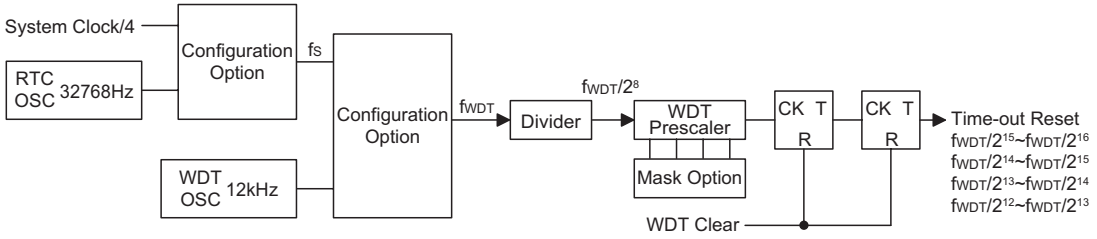


Real Time Clock – RTC

The real time clock (RTC) 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 (bits 2, 1, 0 of RTCC; 09H) yields various time-out periods. If the RTC time-out occurs and the interrupt is enabled, the related interrupt request flag (RTF; bit 1 of the MFIS) is set and the multi-function interrupt request flag (MFF; bit 6 of the INTC1) is set. If the interrupt (EMFI) 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

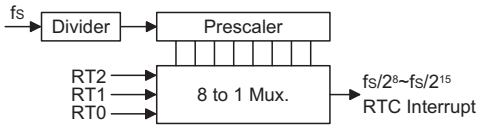


Watchdog Timer

The RTCC register descriptions are listed below.

Bit No.	Label	Read/ Write	Reset	Function
0~2	RT0~ RT2	R/W	1	8 to 1 multiplexer control inputs to select the real time clock prescaler output
3, 5~7	—	—	—	Unused bit, read as ″0″
4	QOSC	R/W	0	32768Hz OSC quick start-up oscillating 0/1: quick/slow start

RTCC (09H) Register



Real Time Clock

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 starts 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.
- The VFD driver keeps running (if the RTC OSC is selected).

The system can leave the HALT mode by means of an external reset, an interrupt, an external falling edge signal on port A, an external rising/falling edge on the RMT pin, or a WDT overflow. An external reset will initialize a chip reset and a WDT overflow will initialize a ″warm reset″. After examining the TO and PDF flags, the source of the reset can be determined. The PDF flag is cleared by a system power-up or by executing the ″CLR WDT″ instruction, and is set by executing the ″HALT″ instruction. The TO flag is set if a WDT time-out occurs, and causes a wake-up that only resets the Program Counter and SP, the other flags remain in their original status.

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 will resume execution at the next instruction. If the system is woken up via an interrupt, two possibilities may occur. If the related interrupt is disabled or the interrupt is enabled but the stack is full, the program will resume execution at the next instruction. But if the interrupt is enabled and the stack is not full, a regular interrupt response takes place.

If an interrupt request flag is set to ″1″ before entering the ″HALT″ mode, the wake-up function of the related interrupt will be disabled.

If a wake-up event occurs, it takes 1024 t_{SYS} (system clock period) to resume normal operation. In other words, a dummy period will be inserted after a wake-up. If the wake-up results from an interrupt acknowledge signal, the actual interrupt subroutine execution will be delayed by one or more cycles. However, if the wake-up results in the next instruction execution, this will be executed immediately after the dummy period is finished.

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

Reset

There are three ways in which a microcontroller reset can occur, through events occurring both internally and externally:

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

The WDT time-out reset during HALT is a little different from other kinds of reset. Most of the conditions remain unchanged except that the Program Counter and Stack Pointer will be cleared to 0 and the TO flag will be set to 1. Most registers are reset to the ″initial condition″ once the reset conditions are met.

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

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

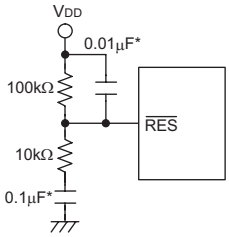
Note: ″u″ stands for unchanged

To ensure 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-on.

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

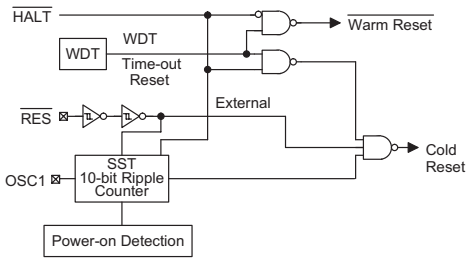
The functional unit chip reset status is shown below.

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

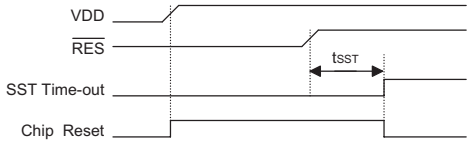


Reset Circuit

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



Reset Configuration



Reset Timing Chart

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)*
TMR0H	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR0L	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR0C	00-0 1000	00-0 1000	00-0 1000	00-0 1000	uu-u uuuu
TMR1H	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR1L	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR1C	0000 1---	0000 1---	0000 1---	0000 1---	uuuu u---
Program Counter	0000H	0000H	0000H	0000H	0000H
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
BP	--0- -000	--0- -000	--0- -000	--0- -000	--u- -uuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu

Register	Reset (Power-on)	WDT Time-out (Normal Operation)	$\overline{RES}$ Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-out (HALT)*
INTC0	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	-000 -000	-000 -000	-000 -000	-000 -000	-uuu -uuu
RMTC	0000 000-	0000 000-	0000 000-	0000 000-	uuuu uuu-
MFIS	---0 0000	---0 0000	---0 0000	---0 0000	---u uuuu
RTCC	---0 -111	---0 -111	---0 -111	---0 -111	---u -uuu
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	---- 1111	---- 1111	---- 1111	---- 1111	---- uuuu
PBC	---- 1111	---- 1111	---- 1111	---- 1111	---- uuuu
PC	1--- ----	1--- ----	1--- ----	1--- ----	u--- ----
PCC	1--- ----	1--- ----	1--- ----	1--- ----	u--- ----
PD	1111 ----	1111 ----	1111 ----	1111 ----	uuuu ----
PDC	1111 ----	1111 ----	1111 ----	1111 ----	uuuu ----
SBCR	0110 0000	0110 0000	0110 0000	0110 0000	uuuu uuuu
SBDR	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
RMT0	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
RMT1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
VFDC	0000 -111	0000 -111	0000 -111	0000 -111	0000 -111

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

Timer/Event Counter

Two timer/event counters (TMR0,TMR1) are implemented in the microcontroller. The Timer/Event Counter 0 contains a 16-bit programmable count-up counter and the clock may come from an external source or an internal clock source. An internal clock source comes from f_{SYS}. The Timer/Event Counter 1 contains a 16-bit programmable count-up counter and the clock may come from an external source or an internal clock source. An internal clock source comes from $f_{SYS}/4$ or 32768Hz selected by option. The external clock input allows the user to count external events, measure time intervals or pulse widths, or to generate an accurate time base.

There are six registers related to the Timer/Event Counter 0 and Timer/Event Counter 1; TMR0H (0CH), TMR0L (0DH), TMR0C (0EH) and the Timer/Event Counter 1, TMR1H (0FH), TMR1L (10H), and TMR1C (11H). Writing TMR0L (TMR1L) will only place the written data to an internal lower-order byte buffer (8-bit) and writing TMR0H (TMR1H) will transfer the specified data and the contents of the lower-order byte buffer to TMR0H (TMR1H) and TMR0L (TMR1L) registers, respectively. The Timer/Event Counter 1/0 preload register is changed by each writing TMR0H (TMR1H) operations. Reading TMR0H (TMR1H) will latch the contents of TMR0H (TMR1H) and TMR0L (TMR1L) counters to the destination and the lower-order byte buffer, respectively. Reading the TMR0L (TMR1L) will read the contents of the lower-order byte buffer. The TMR0C (TMR1C) is the Timer/Event Counter 0 (1) control register, which defines the operating mode, 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 (TMR0, TMR1) pin. The timer mode functions as a normal timer with the clock source coming from the internal selected clock source. Finally, the pulse width measurement mode can be used to count the high or low level duration of the external signal (TMR0, TMR1), 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 FFFFH. Once an overflow occurs, the counter is reloaded from the timer/event counter preload register, and generates an interrupt request flag (T0F; bit 6 of the INTC0, T1F; bit 4 of the INTC1). In the pulse width measurement mode with the values of the TON and TE bits equal to 1, after the TMR0 (TMR1) has received a transient from low to high (or high to low if the TE bit is ″0″), it will start counting until the TMR0 (TMR1) returns to the

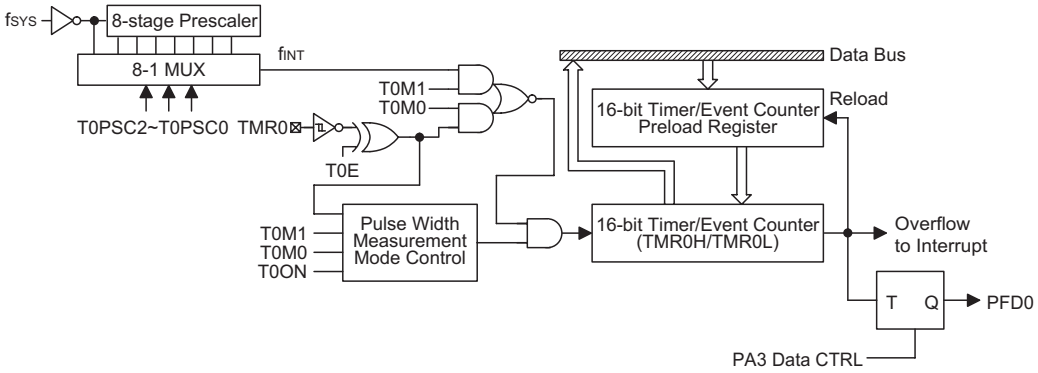
original level and resets the TON. The measured result remains in the timer/event counter even if the activated transient occurs again. In other words, only 1-cycle measurement can be made until the TON bit is set. The cycle measurement will continue as long as it receives further transient pulse. In this operation mode, the timer/event counter begins counting not according to the logic level but to the transient edges. In the case of counter overflows, the counter is reloaded from the timer/event counter register and issues an interrupt request, as in the other two modes, i.e., the event and timer modes.

To enable the counting operation, the Timer ON bit (TON; bit 4 of the TMR0C or TMR1C) should be set to 1. In the pulse width measurement mode, TON is automatically cleared after the measurement cycle is completed. But in the other two modes, TON can only be reset by instructions. The overflow of the Timer/Event Counter 0/1 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 (PFD0 or PFD1) can be applied to PA3 by options. No matter what the operation mode is, writing a 0 to ET0I or ET1I disables the related interrupt service. When the PFD function is selected, executing "SET [PA].3" instruction will enable the PFD output and executing "CLR [PA].3" instruction will disable the PFD output.

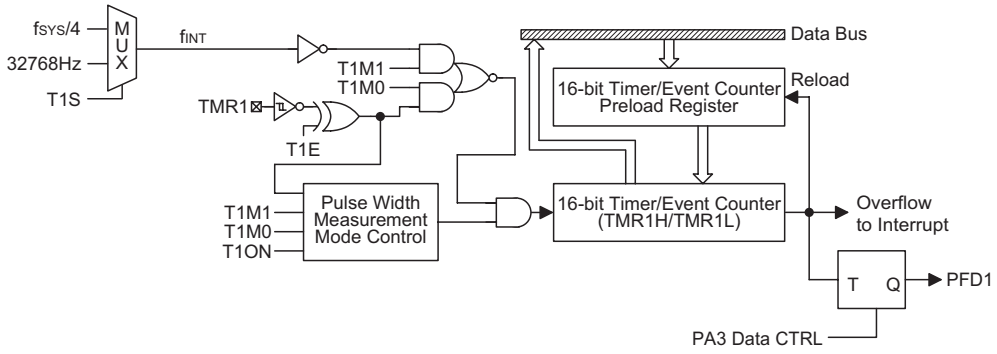
In the case of timer/event counter OFF condition, writing data to the timer/event counter preload register also reloads that data to the timer/event counter. But if the timer/event counter is turned on, data written to the timer/event counter is kept only in the timer/event counter preload register.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

When the timer/event counter (reading TMR0/TMR1) is read, the clock is blocked to avoid errors, as this may results in a counting error. Blocking of the clock should be taken into account by the programmer. It is strongly recommended to load the desired value into the TMR0/TMR1 register first, before turning on the related timer/event counter, for proper operation since the initial value of the TMR0/TMR1 is unknown. Due to the timer/event counter 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 counter function, to avoid unpredictable result. After this procedure, the timer/event function can be operated normally.

Bits 2~0 of the TMR0C can be used to define the prescaling stages of the internal clock sources of the timer/event counter. The definitions are as shown. The overflow signal of the timer/event counter can be used to generate the PFD signal.



Timer/Event Counter 0



Timer/Event Counter 1

Bit No.	Label	Function
0~2	T0PSC0~ T0PSC2	Defines the prescaler stages. T0PSC2, T0PSC1, T0PSC0= 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	T0E	Defines the TMR0 active edge of the timer/event counter: In Event Counter Mode (T0M1,T0M0)=(0,1): 1:count on falling edge; 0:count on rising edge In Pulse Width measurement mode (T0M1,T0M0)=(1,1): 1: start counting on the rising edge, stop on the falling edge; 0: start counting on the falling edge, stop on the rising edge
4	T0ON	Enable/disable timer counting (0=disable; 1=enable)
5	—	Unused bit, read as ″0″
6 7	T0M0 T0M1	Defines the operating mode (T0M1, T0M0) 01= Event count mode (External clock) 10= Timer mode (Internal clock) 11= Pulse Width measurement mode (External clock) 00= Unused

TMR0C (0EH) Register

Bit No.	Label	Function
0~2	—	Unused bit, read as ″0″
3	T1E	Defines the TMR1 active edge of the timer/event counter: In Event Counter Mode (T1M1,T1M0)=(0,1): 1:count on falling edge; 0:count on rising edge In Pulse Width measurement mode (T1M1,T1M0)=(1,1): 1: start counting on the rising edge, stop on the falling edge; 0: start counting on the falling edge, stop on the rising edge
4	T1ON	Enable/disable timer counting (0= disable; 1= enable)
5	T1S	Defines the TMR1 internal clock source (0=$f_{SYS}/4$; 1=32768Hz)
6 7	T1M0 T1M1	Defines the operating mode (T1M1, T1M0) 01= Event count mode (External clock) 10= Timer mode (Internal clock) 11= Pulse Width measurement mode (External clock) 00= Unused

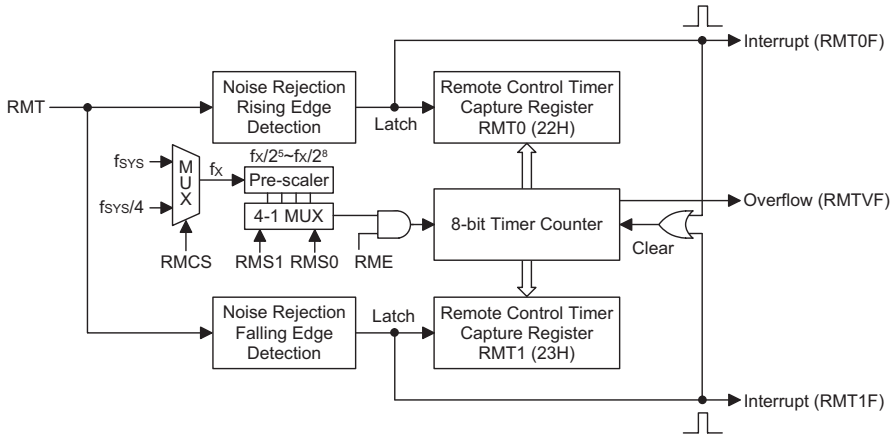
TMR1C (11H) Register

Remote Control Timer – RMT

The HT49RV3/HT49CV3 provides an 8-bit remote control timer that has a pulse width measurement function. Pulse width is measured from a difference in count value when the valid edge (RMT pin) has been detected while the timer operates in the running mode.

The RMT pin with rising/falling edge wake-up function, 8-bit timer counter will be cleared during a chip reset.

And the RMT clock starts to count after the rising/falling edge trigger.



Remote Control Timer

Bit No.	Label	Function
0	—	Unused bit, read as "0"
1	ERMT0	Controls the remote control timer rising edge interrupt (1=enable; 0=disable)
2	ERMT1	Controls the remote control timer falling edge interrupt (1=enable; 0=disable)
3	ERMTV	Controls the remote control timer overflow interrupt (1=enable; 0=disable)
4	RME	Controls the remote control timer (1=enable & start count; 0=disable & clear counter)
5	RMCS	Selects the remote control timer clock source f_X (1=f_{SYS}; 0=f_{SYS}/4)
6~7	RMS0 RMS1	Selects the remote control timer clock 00=$f_X/2^5$ 01= $f_X/2^6$ 10= $f_X/2^7$ 11= $f_X/2^8$

RMTC (21H) Register

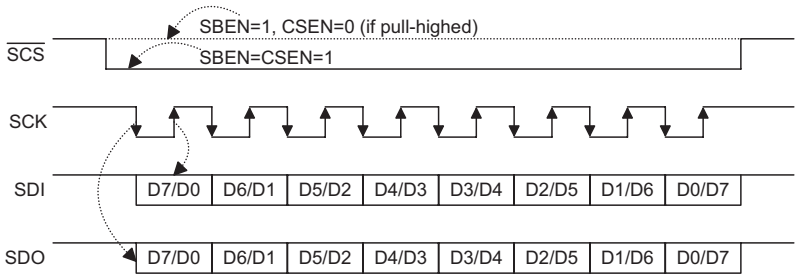
Bit No.	Label	Function
0	RMTVF	Remote control timer overflow interrupt flag (1=indicates that an overflow has occurred; 0=indicates that an overflow has not occurred)
1	RTF	Real time clock interrupt flag (1=indicates that an RTC interrupt has occurred; 0=indicates that an RTC interrupt has not occurred)
2	RMT0F	Remote control timer rising edge interrupt flag (1=indicates that a rising edge interrupt has occurred; 0=indicates that a rising edge interrupt has not occurred)
3	RMT1F	Remote control timer falling edge interrupt flag (1=indicates that a falling edge interrupt has occurred; 0=indicates that a falling edge interrupt has not occurred)
4	ERTI	Controls the Real Time Clock interrupt (1=enable; 0=disable)
5~7	—	Unused bit, read as "0"

MFIS (29H) Register

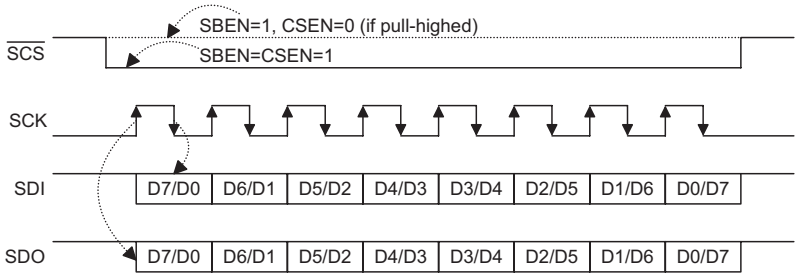
Serial Interface

The Serial Interface function has four basic signals included. They are the SDI (serial data input), SDO (serial data output), SCK (serial clock) and $\overline{SCS}$ (slave select pin).

Two registers (SBCR & SBDR) unique to the serial interface provide control, status and data storage.



SIO Timing (SIOCLK Configuration is Falling Edge)



SIO Timing (SIOCLK Configuration is Rising Edge)

Bit No.	Label	Function
0	TRF	Serial interface data transferred or data received flag 1: data transferred or data received This bit is set by SIO hardware and should be cleared by software after data transferred or data received.
1	WCOL	This bit shows the situation of the buffer SBDR 1: enable (set by SIO) writing to SBDR 0: disable (cleared by user) reading from SBDR
2	CSEN	Serial bus selection signal
3	MLS	Shift first control bit (1: MSB; 0: LSB)
4	SBEN	Serial bus selection (1: enable; 0: disable)
5, 6	M0, M1	Master/slave mode selection: M1, M0= 00: master mode, baud rate=f_{SIO} 01: master mode, baud rate=$f_{SIO}/4$ 10: master mode, baud rate=$f_{SIO}/16$ 11: slave mode
7	CKS	Clock source selection (0: $f_{SYS}/4$; 1: f_{RTCOSC})

SBCR (1FH) Register

- SBCR: Serial bus control register
  - Bit 7 (CKS): clock source selection: $f_{SIO}=f_{SYS}/4$ or f_{RTCOSC}
  - Bit 6 (M1), Bit 5 (M0): master/slave mode & baud rate selection
    - M1, M0:

      00: master mode, baud rate=f_{SIO}

      01: master mode, baud rate=$f_{SIO}/4$

      10: master mode, baud rate=$f_{SIO}/16$

      11: slave mode
  - Bit 4 (SBEN): serial bus enable/disable (1/0)
    - Enable: ($\overline{SCS}$ dependent on CSEN bit)

      Disable → enable: SCK, SDI, SDO, $\overline{SCS}$=0 and waiting for writing data to SBDR (TXRX buffer)

      Master mode: write data to SBDR (TXRX buffer) → start transmission/reception automatically

Master mode: when data has been transferred → set TRF

Slave mode: when a SCK (and $\overline{SCS}$ dependent on CSEN) is received, data in TXRX buffer is shifted-out and data on SDI is shifted-in

– Disable: SCK, SDI, $\overline{SCS}$ floating, SDO output high

- Bit 3 (MLS): MSB or LSB (1/0) shift first control bit
- Bit 2 ($\overline{CSEN}$): serial bus selection signal enable/disable ($\overline{SCS}$), when CSEN=0, $\overline{SCS}$ is floating
- Bit 1 (WCOL): this bit is set to 1 if data is written to SBDR (TXRX buffer) when data is transferred → writing will be ignored if data is written to SBDR (TXRX buffer) when data is transferred
- Bit 0 (TRF): data transferred or data received → used to generate interrupt
  Note: data receiving is still working when the MCU enters the HALT mode

• SBDR: Serial bus data register

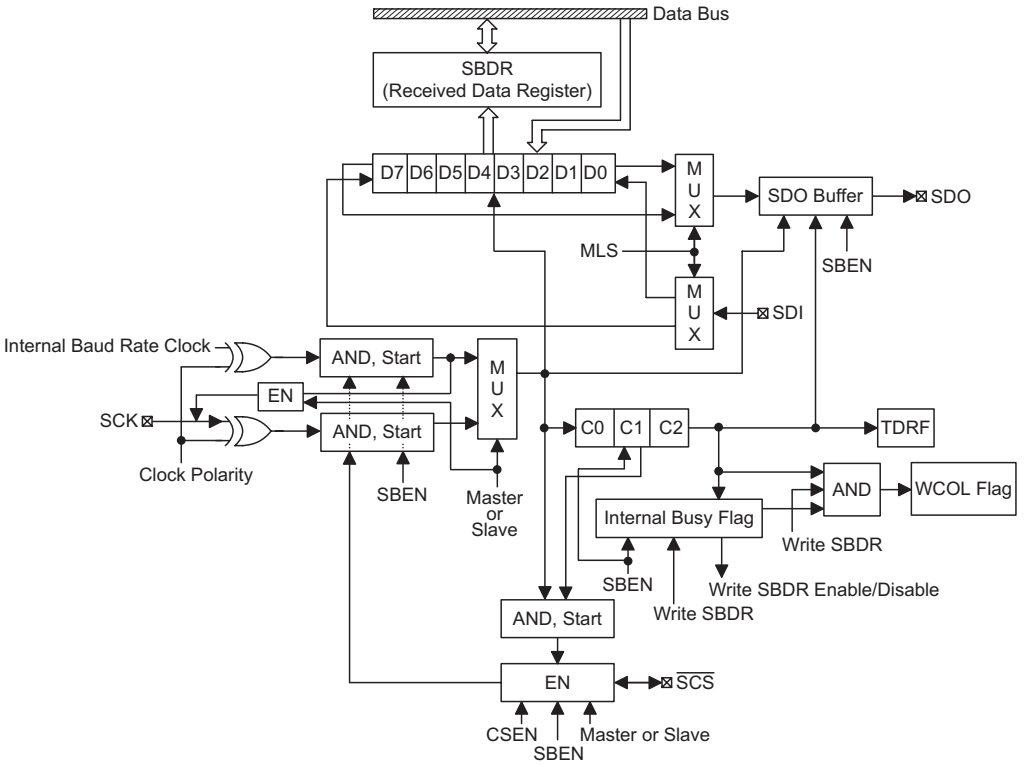
- Data written to SBDR → write data to TXRX buffer only
- Data read from SBDR → read from SBDR only
- Operating Mode description
  – Master transmitter: clock sending and data I/O started by writing SBDR
  – Master clock sending started by writing SBDR
  – Slave transmitter: data I/O started by clock received
  – Slave receiver: data I/O started by clock received

Clock Polarity=Rising Edge or Falling Edge (Configuration Option)

• Serial interface operation

- Master mode operation

Step1: Select CKS and select M1, M0 = 00, 01, 10
Step2: Select CSEN, MLS (the same as the slave)
Step3: Set SBEN
Step4: Writing data to SBDR
– data is stored in TXRX buffer
– output SCK and $\overline{SCS}$ signals
– go to step 5
Note: SIO internal operation:
* data stored in TXRX buffer, and SDI data is shifted into TXRX buffer
* data transferred, data in TXRX buffer is latched into SBDR
Step5: Check WCOL
– WCOL= 1, clear WCOL and go to step 4
– WCOL= 0, go to step 6
Step6: Check TRF or waiting for serial bus interrupt
Step7: Read data from SBDR
Step8: Clear TRF
Step9: Go to step 4



SIO Block Diagram

- Slave mode operation

  Step1: CKS don't care and select M1, M0 =11

  Step2: Select CSEN, MLS (the same as the master)

  Step3: Set SBEN

  Step4: Writing data to SBDR

  – data is stored in the TXRX buffer

  – waiting for master clock signal (and $\overline{SCS}$): SCK

  – go to step 5

  Note: SIO internal operation:
  * SCK ($\overline{SCS}$) received
  * output data in TXRX buffer and SDI data is shifted into TXRX buffer
  * data transferred, data in TXRX buffer is latched into SBDR

  Step5: Check WCOL

  – WCOL=1, clear WCOL, go to step 4

  – WCOL=0, go to step 6

  Step6: Check TRF or waiting for serial bus interrupt

  Step7: Read data from SBDR

  Step8: Clear TRF

  Step9: Go to step 4

Input/Output Ports

There are 17 bidirectional input/output lines in the microcontroller, labeled as PA, PB, PC and PD, which are mapped to the data memory of [12H], [14H], [16H] 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, 16H 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, PCC, 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 a "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 an output function, CMOS is the only configuration. These control registers are mapped to locations 13H, 15H, 17H 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, 16H 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 pin is pin-shared with the PFD signal. If the PFD option is selected, the output signal in the output mode of PA3 will be the PFD signal generated by the timer/event counter overflow signal. The input mode always retains its original functions. Once the PFD option is selected, the PFD output signal is controlled by the PA3 data register only. Writing a "1" to the PA3 data register will enable the PFD output function and writing a "0" will force the PA3 pin 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)

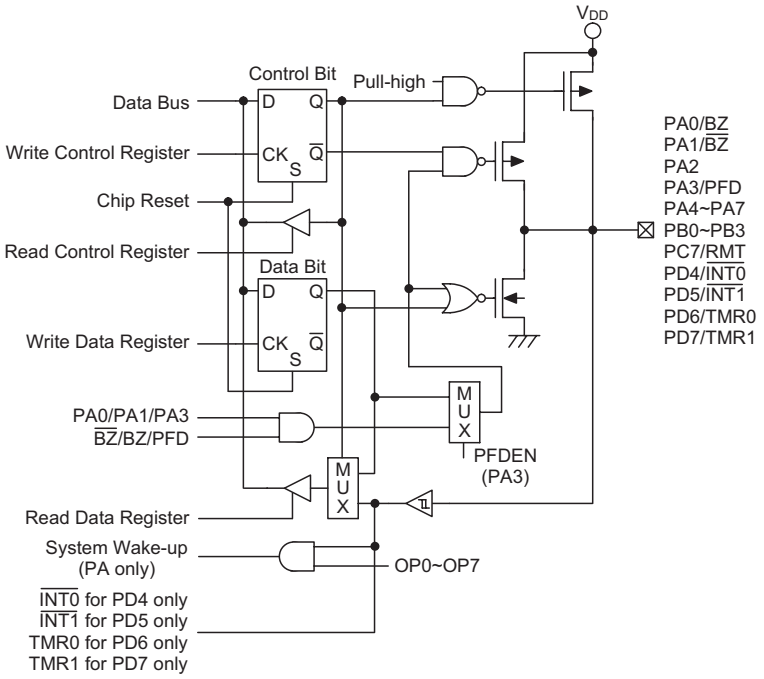
Note: The PFD frequency is the timer/event counter overflow frequency divided by 2.

The descriptions 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 "65536" for PFD0 or PFD1
"N" is preload value for timer/event counter
"f_{TMR}" is input clock frequency for timer/event counter

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



Input/Output Ports

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

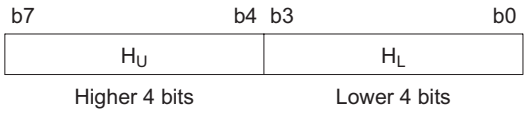
Note: ″I″ input; ″O″ output
″D, D0, D1″ Data
″B″ buzzer option, BZ or $\overline{BZ}$
″X″ don′t care
″C″ CMOS output

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

VFD Display Memory

The HT49RV3/HT49CV3 provides an area of embedded data memory for the VFD display. This area is located from 40H to 55H of the RAM at Bank 1. The Bank Pointer (BP; located at 04H of the RAM) is the switch for the RAM and the VFD display memory. When the BP is set as ″1″, any data written into 40H~55H will affect the VFD display. When the BP is written as ″0″ any data written into 40H~55H is meant to access the general purpose data memory. The VFD display memory can be read and written to only by an indirect addressing mode using MP1. When data is written into the display data area, it is automatically read by the VFD driver which then generates the corresponding VFD 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 VFD pattern for the HT49RV3/HT49CV3.

	SEG15~ SEG12	SEG11~ SEG8	SEG7~ SEG4	SEG3~ SEG0
Grid0	$41H_U$	$41H_L$	$40H_U$	$40H_L$
Grid1	$43H_U$	$43H_L$	$42H_U$	$42H_L$
Grid2	$45H_U$	$45H_L$	$44H_U$	$44H_L$
Grid3	$47H_U$	$47H_L$	$46H_U$	$46H_L$
Grid4	$49H_U$	$49H_L$	$48H_U$	$48H_L$
Grid5	$4BH_U$	$4BH_L$	$4AH_U$	$4AH_L$
Grid6	$4DH_U$	$4DH_L$	$4CH_U$	$4CH_L$
Grid7	$4FH_U$	$4FH_L$	$4EH_U$	$4EH_L$
Grid8	$51H_U$	$51H_L$	$50H_U$	$50H_L$
Grid9	$53H_U$	$53H_L$	$52H_U$	$52H_L$
Grid10	$55H_U$	$55H_L$	$54H_U$	$54H_L$



VFD Display Memory

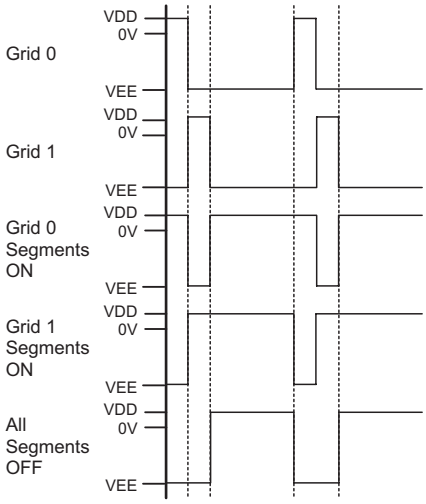
VFD Display Control Register – VFDC

Bit No.	Label	Function
2~0	VGS2~ VGS0	Selects the VFD display mode 000= 4 grids, 16 segments 001= 5 grids, 16 segments 010= 6 grids, 16 segments 011= 7 grids, 15 segments 100= 8 grids, 14 segments 101= 9 grids, 13 segments 110= 10 grids, 12 segments 111= 11 grids, 11 segments
3	—	Unused bit, read as ″0″
4	VFDE	Controls the VFD display (1=enable; 0=disable)
7~5	VDM2~ VDM0	Sets the VFD dimming quantity 000= set pulse width to 1/16. 001= set pulse width to 2/16. 010= set pulse width to 4/16. 011= set pulse width to 10/16. 100= set pulse width to 11/16. 101= set pulse width to 12/16. 110= set pulse width to 13/16. 111= set pulse width to 14/16.

VFDC (28H) Register

At power-on initial, the 11-grid, 11-segment & 1/16 pulse width are set and the VFD display is disabled.

The VFD clock source may come from the RTC or the system clock/4 (f_{SYS}/4).



VFD Driver Output

Low Voltage Reset

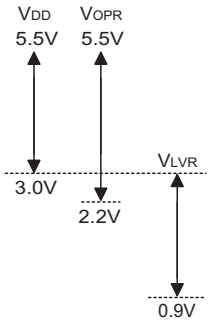
There is a low voltage reset circuit (LVR) implemented in the microcontroller. This function can be enabled/disabled by options.

If the supply voltage of the device is within the range 0.9V~V_{LVR}, such as when changing a battery, the LVR will automatically reset the device internally.

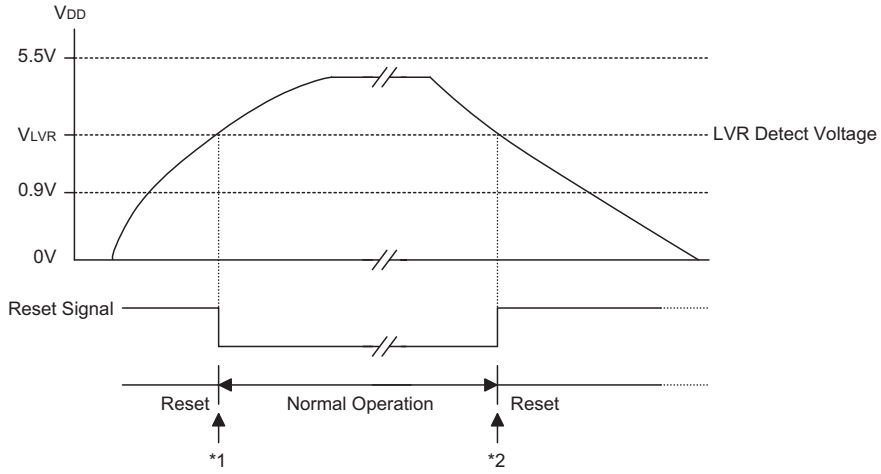
The LVR includes the following specifications:

- The low voltage (0.9V~V_{LVR}) has to remain in its original state for longer than 1ms. If the low voltage state does not exceed 1ms, the LVR will ignore it and will not perform a reset function.
- The LVR uses an "OR" function with the external $\overline{RES}$ signal to perform a 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 starting normal operation.

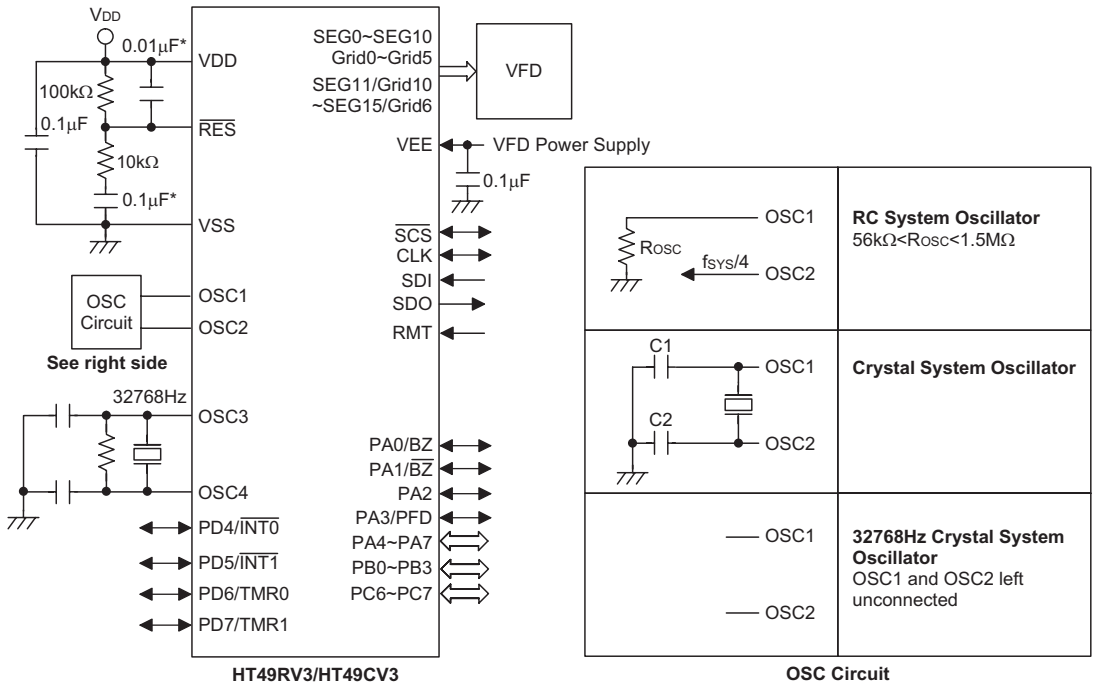
*2: Since the low voltage has to maintain its original state for longer than 1ms, therefore a 1ms delay enters the reset mode.

Options

The following shows the configuration options in the HT49RV3/HT49CV3. All these options should be defined in order to ensure having a properly functioning system.

Options
OSC type selection. This option is to decide if an RC or crystal or 32768Hz crystal oscillator is chosen as the system clock.
f_{WDT}: WDT clock source selection. There are three types of selections: system clock/4, RTC OSC or WDT OSC
f_S: VFD, RTC and buzzer clock source selection. There are two types of selections: system clock/4 or RTC 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 $f_{WDT}/2^{12}$~$f_{WDT}/2^{13}$, $f_{WDT}/2^{13}$~$f_{WDT}/2^{14}$, $f_{WDT}/2^{14}$~$f_{WDT}/2^{15}$ or $f_{WDT}/2^{15}$~$f_{WDT}/2^{16}$
CLR WDT times selection. This option defines the method to clear the WDT by instruction. ″One time″ means that the ″CLR WDT″ instructions can clear the WDT. ″Two times″ means only if both of the ″CLR WDT1″ and ″CLR WDT2″ instructions have been executed, the WDT can be cleared.
Buzzer output frequency selection. There are eight types of frequency signals for buzzer output: $f_S/2^1$~$f_S/2^8$, ″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 determine whether the pull-high resistance is viable or not in the input mode of the I/O ports. PA, PB0~PB3, PC7 and PD4~PD7 can be independently selected. (bit option)
RMT Pull-high selection. This option is to determine whether a pull-high resistance is viable or not in the input pin.
I/O pins shared with other function selections. PA0/BZ, PA1/$\overline{BZ}$, PA3/PFD: PA0, PA1 and PA3 can be set as I/O pins or buzzer outputs.
VFD driver clock selection. There are 8 types of frequency signals for the VFD driver circuits: $f_S/2^0$~$f_S/2^7$. ″f_S″ stands for the clock source selection by options.
VFD ON/OFF at HALT selection
LVR selection: LVR has enable or disable options
PFD selection. If PA3 is set as a PFD output, there are two types of selections; One is PFD0 as the PFD output, the other is PFD1 as the PFD output. PFD0, PFD1 are the timer overflow signals of the Timer/Event Counter 0, Timer/Event Counter 1, respectively.
$\overline{INT0}$ or $\overline{INT1}$ triggering edge: Disable; high to low; low to high; low to high or high to low
SIOCLK: Serial interface clock. There are falling edge or rising edge
CSEN: Serial bus selection: enable or disable
WCOL: SBDR write conflict

Application Circuits



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

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] ← $[\overline{m}]$

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.

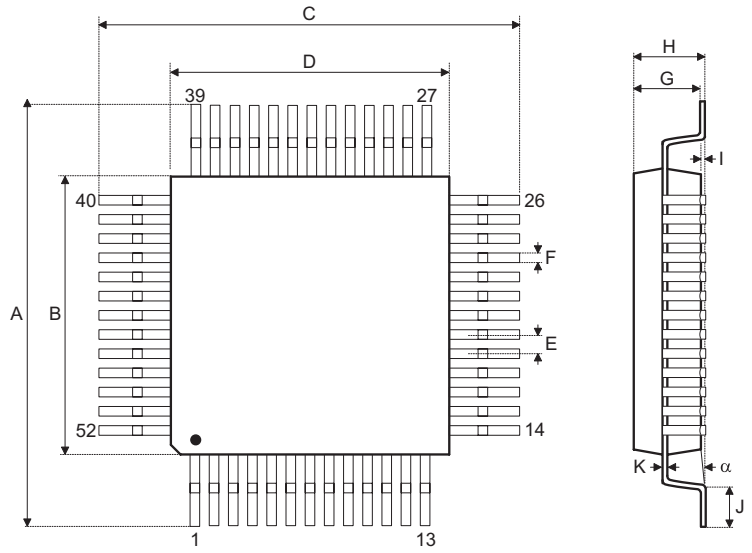
Operation ACC ← ACC ″XOR″ x

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°

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: 86-21-6485-5560
Fax: 86-21-6485-0313
http://www.holtek.com.cn

Holtek Semiconductor Inc. (Shenzhen Sales Office)
5/F, Unit A, Productivity Building, Cross of Science M 3rd Road and Gaoxin M 2nd Road, Science Park, Nanshan District, Shenzhen, China 518057
Tel: 86-755-8616-9908, 86-755-8616-9308
Fax: 86-755-8616-9722

Holtek Semiconductor Inc. (Beijing Sales Office)
Suite 1721, Jinyu Tower, A129 West Xuan Wu Men Street, Xicheng District, Beijing, China 100031
Tel: 86-10-6641-0030, 86-10-6641-7751, 86-10-6641-7752
Fax: 86-10-6641-0125

Holtek Semiconductor Inc. (Chengdu Sales Office)
709, Building 3, Champagne Plaza, No.97 Dongda Street, Chengdu, Sichuan, China 610016
Tel: 86-28-6653-6590
Fax: 86-28-6653-6591

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