HT48R054-1/HT48R054-2
8-Bit OTP Microcontroller

Features

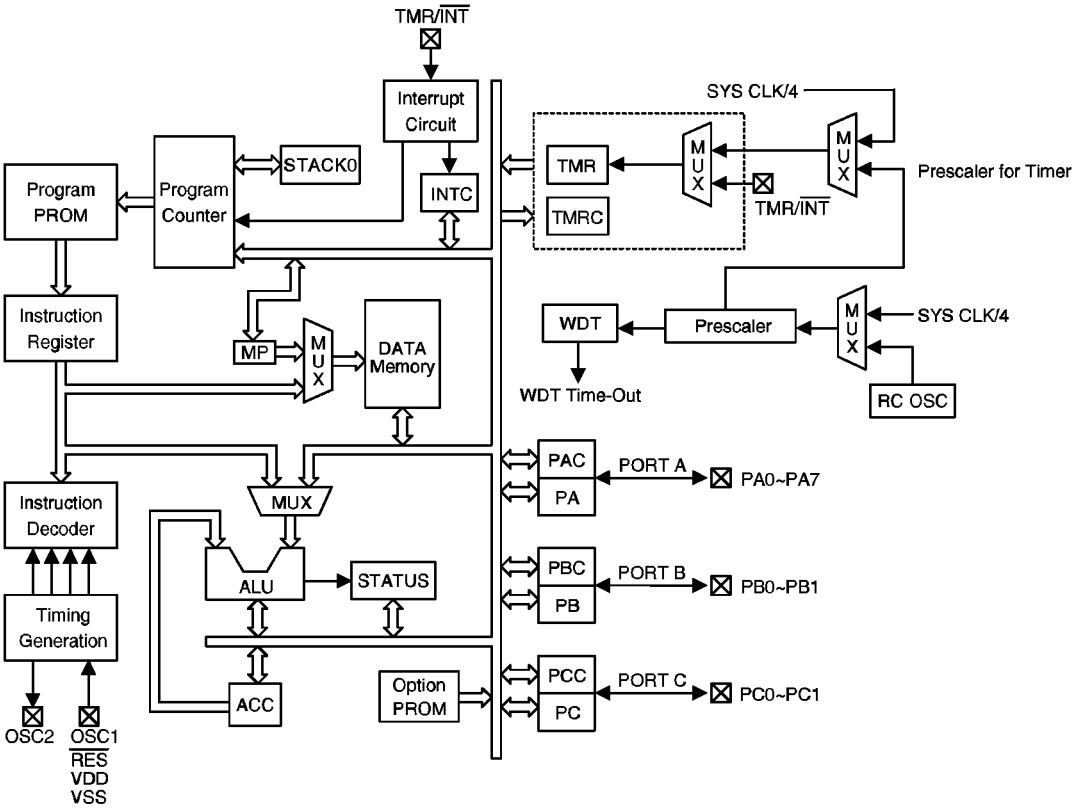
- Operating voltage:
  - HT48R054-1: 3.0V~5.2V
  - HT48R054-2: 2.4V~4.0V
- 12 bidirectional I/O lines
- One interrupt input
- One 8-bit programmable timer/event counter with overflow interrupt
- 512×14 program memory PROM
- 3 × 14 option memory PROM
- 32×8 data memory RAM
- Watchdog timer
- On-chip crystal and RC oscillator
- Halt function and wake-up feature reduce power consumption
- One-level subroutine nesting
- Bit manipulation instructions
- Up to 1μs instruction cycle with 4MHz system clock at V_{DD}=5V (HT48R054-1)
- 56 powerful instructions
- All instructions in 1 or 2 machine cycles

General Description

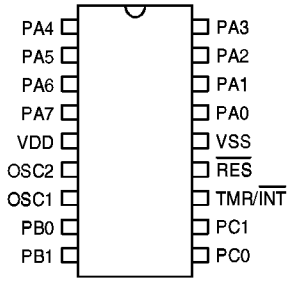
The HT48R054-1/HT48R054-2 is an 8-bit high performance RISC-like microcontroller specifically designed for multiple I/O product applications. The device is particularly suitable for use in products such as remote controllers, fan/light controllers, washing machine controllers, scales, toys and various subsystem controllers. A halt feature is included to reduce power consumption.

The program and option PROM can be electrically programmed. The PROM type program and option memory make the HT48R054-1/HT48R054-2 suitable for use in product developments.

Block Diagram



Pin Assignment



HT48R054-1/HT48R054-2
– 18PDIP-A-0

*Notes: The TMR and $\overline{INT}$ are the same pad.

The TMR/$\overline{INT}$ pin is without pull-high resistor.

Pin Description

Pin Name	I/O	ROM Code Option	Function
PA0~PA7	I/O	Wake-up pull-high or none	Bidirectional 8-bit Input/Output port. Each bit can be configured as a wake-up input by ROM code option. Software instructions determine the CMOS output or schmitt trigger input with or without pull-high resistor (ROM code option).
PB0~PB1	I/O	Pull-high or none	Bidirectional 2-bit Input/Output port. Software instructions determine the CMOS output or schmitt trigger input with or without pull-high resistor (ROM code option).
PC0~PC1	I/O	Pull-high or none	Bidirectional 2-bit Input/Output port. Software instructions determine the CMOS output or schmitt trigger input with or without pull-high resistor (ROM code option).
VSS	—	—	Negative power supply, GND.
TMR/$\overline{INT}$	I	—	External interrupt and external event counter schmitt trigger input. Edge triggered activated on a high to low transition for external interrupt.
$\overline{RES}$	I	—	Schmitt trigger reset input. Active low.
VDD	—	—	Positive power supply.
OSC1 OSC2	I O	Crystal or RC	OSC1, OSC2 are connected to the RC network or to the crystal (determined by ROM code option) for the internal system clock. In the case of RC operation, OSC2 is the output terminal for 1/4 system clock (NMOS open drain output).

Absolute Maximum Ratings*

Supply Voltage ..... V_{DD}–0.3V to 5.5V
Input Voltage ..... VSS–0.3V to V_{DD}+0.3V
Storage Temperature ..... –50°C to 125°C
Operating Temperature ..... –25°C to 70°C

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

D.C. Characteristics

HT48R054-1 Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Uint
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	3.0	—	5.2	V
I_{DD1}	Operating Current (Crystal OSC)	3V	No load, f_{SYS}=2MHz	—	0.4	1	mA
		5V		—	1.2	2	mA
I_{DD2}	Operating Current (Crystal OSC)	3V	No load, f_{SYS}=32768Hz	—	0.1	0.2	mA
		5V		—	0.4	0.6	mA
I_{DD3}	Operating Current (RC OSC)	3V	No load, f_{SYS}=2MHz	—	0.4	1	mA
		5V		—	1	2	mA
I_{STB1}	Standby Current (WDT Enabled)	3V	No load, System HALT	—	—	5	μA
		5V		—	—	10	μA
I_{STB2}	Standby Current (WDT Disabled)	3V	No load, System HALT	—	—	1	μA
		5V		—	—	2	μA
V_{IL1}	Input Low Voltage for I/O Ports	3V	—	0	—	0.9	V
		5V	—	0	—	1.5	V
V_{IH1}	Input High Voltage for I/O Ports	3V	—	2.1	—	3	V
		5V	—	3.5	—	5	V
V_{IL2}	Input Low Voltage ($\overline{INT}$, TMR)	3V	—	0	—	0.7	V
		5V	—	0	—	1.3	V
V_{IH2}	Input High Voltage ($\overline{INT}$, TMR)	3V	—	2.3	—	3	V
		5V	—	3.8	—	5	V
V_{IL3}	Input Low Voltage ($\overline{RES}$)	3V	—	—	1.5	—	V
		5V	—	—	2.5	—	V

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Uint
		V_{DD}	Conditions				
V_{IH3}	Input High Voltage ($\overline{RES}$)	3V	—	—	2.4	—	V
		5V	—	—	4.0	—	V
I_{OL}	I/O Ports Sink Current	3V	V_{OL}=0.3V	3	6	—	mA
		5V	V_{OL}=0.5V	8	16	—	mA
I_{OH}	I/O Ports Source Current	3V	V_{OH}=2.7V	–1	–1.5	—	mA
		5V	V_{OH}=4.5V	–2	–4	—	mA
R_{PH}	Pull-high Resistance of I/O Ports	3V	—	40	60	80	kΩ
		5V	—	10	30	50	kΩ

HT48R054-2 Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Uint
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.4	—	4.0	V
I_{DD1}	Operating Current (Crystal OSC)	3V	No load, f_{SYS}=2MHz	—	0.4	1	mA
I_{DD2}	Operating Current (Crystal OSC)	3V	No load, f_{SYS}=32768Hz	—	0.1	0.2	mA
I_{DD3}	Operating Current (RC OSC)	3V	No load, f_{SYS}=2MHz	—	0.4	1	mA
I_{STB1}	Standby Current (WDT Enabled)	3V	No load, System HALT	—	—	5	μA
I_{STB2}	Standby Current (WDT Disabled)	3V	No load, System HALT	—	—	1	μA
V_{IL1}	Input Low Voltage for I/O Ports	3V	—	0	—	0.9	V
V_{IH1}	Input High Voltage for I/O Ports	3V	—	2.1	—	3	V
V_{IL2}	Input Low Voltage ($\overline{INT}$, TMR)	3V	—	0	—	0.7	V
V_{IH2}	Input High Voltage ($\overline{INT}$, TMR)	3V	—	2.3	—	3	V
V_{IL3}	Input Low Voltage ($\overline{RES}$)	3V	—	—	1.5	—	V
V_{IH3}	Input High Voltage ($\overline{RES}$)	3V	—	—	2.4	—	V
I_{OL}	I/O Ports Sink Current	3V	V_{OL}=0.3V	3	6	—	mA
I_{OH}	I/O Ports Source Current	3V	V_{OH}=2.7V	–1	–1.5	—	mA
R_{PH}	I/O Ports Pull-high Resistance	3V	—	40	60	80	kΩ

A.C. Characteristics

HT48R054-1 Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS1}	System Clock (Crystal OSC)	3V	ROM code option (Crystal)	400	—	2000	kHz
		5V		400	—	4000	kHz
f_{SYS2}	System Clock (Crystal OSC)	3V	ROM code option (32768Hz crystal)	—	32768	—	Hz
		5V		—	32768	—	Hz
f_{SYS3}	System Clock (RC OSC)	3V	—	400	—	2000	kHz
		5V	—	400	—	3000	kHz
f_{TIMER}	Timer I/P Frequency (TMR)	3V	—	0	—	2000	kHz
		5V	—	0	—	4000	kHz
t_{WDTOSC}	Watchdog Oscillator	3V	—	45	90	180	µs
		5V	—	35	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: $t_{SYS}=1/f_{SYS}$

HT48R054-2 Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS1}	System Clock (Crystal OSC)	3V	ROM code option (Crystal)	400	—	2000	kHz
f_{SYS2}	System Clock (Crystal OSC)	3V	ROM code option (32768Hz crystal)	—	32768	—	Hz
f_{SYS3}	System Clock (RC OSC)	3V	—	400	—	2000	kHz
f_{TIMER}	Timer I/P Frequency (TMR)	3V	—	0	—	2000	kHz
t_{WDTOSC}	Watchdog Oscillator	3V	—	45	90	180	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 for the HT48R054-1/HT48R054-2 is derived from either a crystal or an RC oscillator. The system clock 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. However, the pipelining scheme causes each instruction to effectively execute in one cycle. If an instruction changes the program counter, two cycles are required to complete the instruction.

Program counter – PC

The 9-bit program counter (PC) controls the sequence in which the instructions stored in the program PROM are executed and its contents specify a maximum of 512 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, the PC manipulates the program transfer by loading the address corresponding to each instruction.

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

The lower byte of the program counter (PCL) 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 – PROM

The program memory is used to store the program instructions which are to be executed. It also contains data, interrupt entries, and is organized into 512×14 bits, addressed by the program counter.

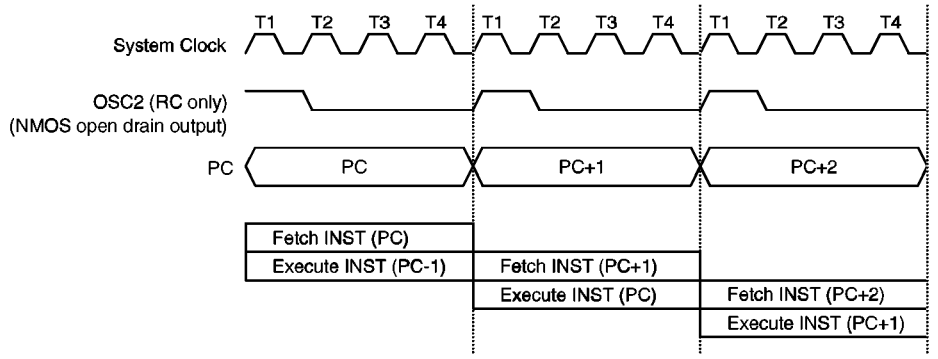
Certain locations in the program memory are reserved for special usage:

- Location 000H

  This area is reserved for the initialization program. After chip reset, the program always begins execution at location 000H.

- Location 004H

  This area is reserved for the external interrupt service program. If the $\overline{INT}$ input pin is activated, and the interrupt is enabled and

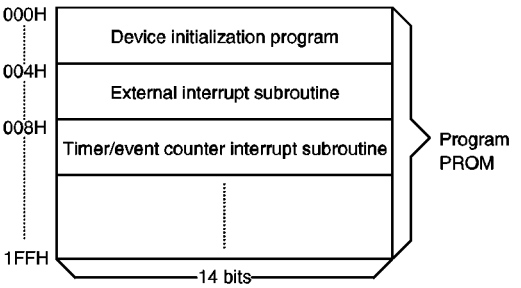


Execution flow

the stack is not full, the program begins execution at location 004H.

- Location 008H

  This area is reserved for the timer/event counter interrupt service program. If timer interrupt results from a timer/event counter overflow, and if the interrupt is enabled and the stack is not full, the program begins execution at location 008H.



Program memory

Stack register – STACK

This is a special part of the memory which is used to save the contents of the program counter (PC) only. The stack is organized into 1 level and is neither part of the data nor part of the program space, and is neither readable nor writeable. The activated level is indexed by the stack pointer (SP) and is neither readable nor writeable. At a subroutine call or interrupt acknowledgment, the contents of the program counter are pushed onto the stack. At the end of a subroutine or an interrupt routine, signaled by a return instruction (RET or RETI), the program counter is restored to its previous value from the stack. After a 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 will be recorded but the acknowledgment will be inhibited. When the stack pointer is decremented (by RET or RETI), the interrupt will be serviced. This feature prevents stack overflow allowing the programmer to use the structure more easily. In a similar case, if the stack is full and a "CALL" is subsequently executed, stack overflow occurs and the previous entry will be lost (only the most recent return addresses are stored).

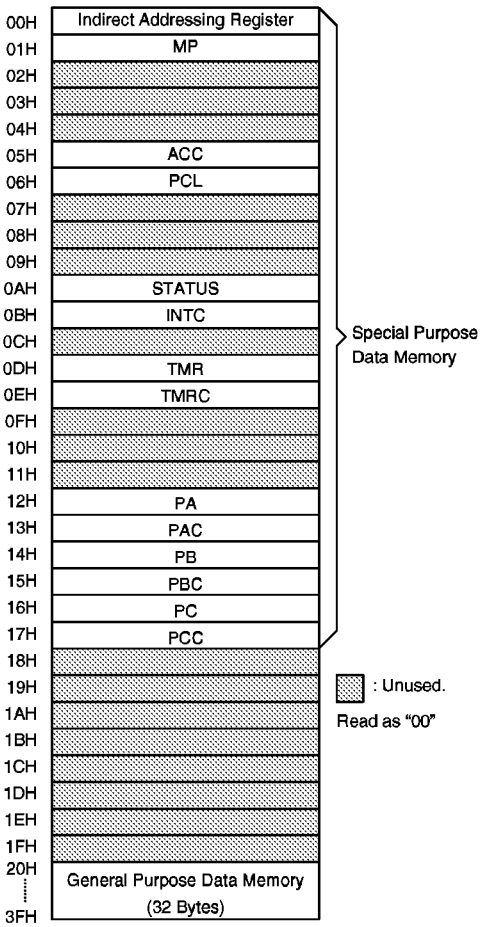
Data memory – RAM

The data memory is designed with 45×8 bits. The data memory is divided into two functional groups: special function registers and general purpose data memory (32×8). Most of them are read/write, but some are read only.

Mode	Program Counter								
	*8	*7	*6	*5	*4	*3	*2	*1	*0
Initial reset	0	0	0	0	0	0	0	0	0
External interrupt	0	0	0	0	0	0	1	0	0
Timer/event counter overflow	0	0	0	0	0	1	0	0	0
Skip	PC+2								
Loading PCL	*8	@7	@6	@5	@4	@3	@2	@1	@0
Jump, call branch	#8	#7	#6	#5	#4	#3	#2	#1	#0

Program counter

Notes: *8~*0: Program Counter Bits
#8~#0: Instruction Code Bits
S8~S0: Stack Register Bits
@7~@0: PCL Bits



RAM mapping

The special function registers include the indirect addressing register (00H), the timer/event counter (TMR;0DH), the timer/event counter control register (TMRC;0EH), the program counter lower-order byte register (PCL;06H), the memory pointer register (MP;01H), the accumulator (ACC;05H), the status register (STATUS;0AH), the interrupt control register (INTC;0BH), the I/O registers (PA;12H, PB;14H, PC;16H) and the I/O control registers (PAC;13H, PBC;15H, PCC;17H). The remaining space before the 20H is reserved for future expanded usage and reading these locations will return the result 00H. The general purpose data memory, addressed from 20H to 3FH, is used for data and control information under instruction command.

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 the SET [m].i and CLR [m].i instructions, respectively. They are also indirectly accessible through Memory pointer register (MP;01H).

Indirect addressing register

Location 00H is an indirect addressing register that is not physically implemented. Any read/write operation of [00H] accesses data memory pointed to by MP (01H). Reading location 00H itself indirectly will return the result 00H. Writing indirectly results in no operation.

The memory pointer register MP (01H) is a 7-bit register. The bit 7 of MP is undefined and reading will return the result "1". Any writing operation to MP will only transfer the lower 7-bit data to MP.

Accumulator

The accumulator closely relates to ALU operations. It is also mapped to location 05H of the data memory and it is capable to carry out immediate data operations. The data movement between two data memory locations must pass through the accumulator.

Arithmetic and logic unit – ALU

This circuit performs 8-bit arithmetic and logic operation. The ALU provides the following functions:

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

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

Status register – STATUS

This 8-bit register (0AH) contains the zero flag (Z), carry flag (C), auxiliary carry flag (AC), overflow flag (OV) power down flag (PD) and watchdog time-out flag (TO). It also records the status information and controls the operation sequence.

With the exception of the TO and PD flags, bits in the status register can be altered by instructions like any other register. Any data written into the status register will not change the TO or PD flags. In addition it should be noted that operations related to the status register may give different results from those intended. The TO and PD flags can only be changed by the watchdog timer overflow, chip power-up, clearing the watchdog timer and executing the HALT instruction.

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

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

Interrupt

The HT48R054-1/HT48R054-2 provides an external interrupt and internal timer/event counter interrupts. The interrupt control register (INTC;0BH) contains the interrupt control bits to set the enable/disable and the interrupt request flags.

Once an interrupt subroutine is serviced, all other interrupts will be blocked (by clearing the EMI bit). This scheme may prevent any further interrupt nesting. Other interrupt requests may happen during this interval but only the interrupt request flag is recorded. If the stack is full, the interrupt request will not be acknowledged, even if the related interrupt is enabled, until the SP is decremented.

All these kinds of interrupt have the wake-up capability. As an interrupt is serviced, a control transfer occurs by pushing the program counter onto the stack and then branching to subroutines at specified location(s) in the program memory. Only the program counter is pushed onto the stack. If the contents of the register and Status register (STATUS) are altered by the interrupt service program which corrupt the desired control sequence, the programmer must save these contents first.

Labels	Bits	Function
C	0	C is set if the 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. Also, C is affected by a rotate through carry instruction.
AC	1	AC is set if the operation results in a carry out of the low nibbles in addition or if there's no borrow from the high nibble into the low nibble in subtraction; otherwise AC is cleared.
Z	2	Z is set if the result of an arithmetic or logic operation is zero; otherwise Z is cleared.
OV	3	OV is set if the 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.
PD	4	PD is cleared when either a system power-up or executing the CLR WDT instruction. PD is set by executing the HALT instruction.
TO	5	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	Undefined, read as "0"
—	7	Undefined, read as "0"

STATUS register

External interrupt is triggered by a high to low transition of $\overline{INT}$ and the related interrupt request flag (EIF; bit 4 of INTC) will be set. When the interrupt is enabled, and the stack is not full and the external interrupt is active, a subroutine call to location 04H will occur. The interrupt request flag (EIF) and EMI bits will be cleared to disable other interrupts.

The internal timer/event counter interrupt is initialized by setting the timer/event counter interrupt request flag (TF; bit 5 of INTC), caused by a timer overflow. When the interrupt is enabled, and the stack is not full and the TF bit is set, a subroutine call to location 08H will occur. The related interrupt request flag (TF) will be reset and the EMI bit cleared to disable further interrupts.

During the execution of an interrupt subroutine, other interrupt acknowledgments are held until the RETI instruction is executed. To return from the interrupt subroutine the RET or RETI instruction may be invoked. RETI will set the EMI bit to enable an interrupt service, but RET will not.

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

No.	Interrupt Source	Priority	Vector
a	External interrupt	1	04H
b	Timer/event counter overflow	2	08H

The timer/event counter interrupt request flag (TF), external interrupt request flag (EIF), enable timer/event counter interrupt bit (ETI), enable external interrupt bit (EEI) and enable master interrupt bit (EMI) constitute an interrupt control register (INTC) which is located at 0BH in the data memory. EMI, EEI, ETI are used to control the enabling/disabling of interrupts. These bits prevent the requested interrupt from being serviced. Once the interrupt request flags (TF, EIF) are set, they will remain in the INTC register until the interrupts are serviced or cleared by a software instruction.

It is recommended that a program does not use the "CALL subroutine" within the interrupt subroutine. Because only one stack is in the HT48R054-1/HT48R054-2 and once the "CALL subroutine" operates in the interrupt subroutine, it will damage the original control sequence.

Register	Bit No.	Label	Function
INTC (0BH)	0	EMI	Controls the master (global) interrupt. (1=enabled; 0=disabled)
	1	EEI	Controls the external interrupt. (1=enabled; 0=disabled)
	2	ETI	Controls the timer/event counter interrupt. (1=enabled; 0=disabled)
	3	–	Unused bit, read as "0"
	4	EIF	External interrupt request flag. (1=active; 0=inactive)
	5	TF	Internal timer/event counter request flag. (1=active; 0=inactive)
	6	–	Unused bit, read as "0"
	7	–	Unused bit, read as "0"

INTC register

Oscillator configuration

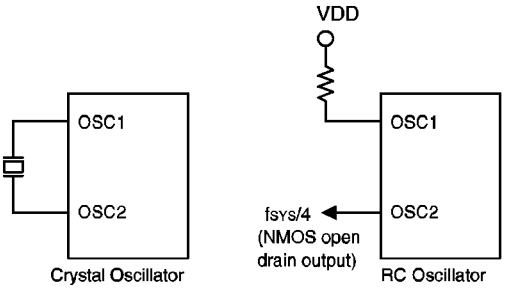
There are two oscillator circuits in the HT48R054-1/HT48R054-2.

Both of them are designed for system clocks; the RC oscillator and the crystal oscillator, which are determined by ROM code options. No matter what oscillator type is selected, the signal provides the system clock. The halt mode stops the system oscillator and ignores the external signal to conserve power.

If an RC oscillator is used, an external resistor between OSC1 and VDD is needed. The system clock, divided by 4, is available on OSC2, 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.

If a crystal oscillator is used, a crystal across OSC1 and OSC2 is needed to provide the feedback and phase shift needed for oscillator, no other external components are needed. Instead of a crystal, the resonator can also be connected between OSC1 and OSC2 to get a frequency reference, but two external capacitors in OSC1 and OSC2 are required.

The WDT oscillator is a free running on-chip RC oscillator, and no external components are required. Even if the system enters the power down mode, the system clock is stopped, but the WDT oscillator still works with a period of approximately 78 μs. The WDT oscillator can be disabled by ROM code option to conserve power.

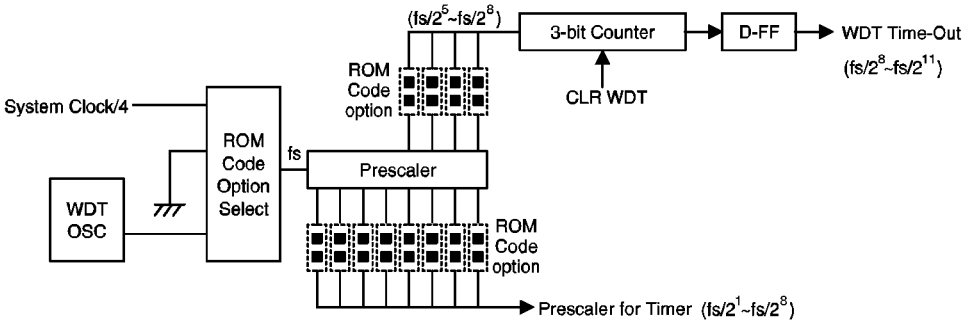


System oscillator

Watchdog timer – WDT

The WDT clock source is implemented by a dedicated RC oscillator (WDT oscillator) or instruction clock (system clock divided by 4), determined by ROM code options. This timer is designed to prevent a software malfunction or sequence jumping to an unknown location with unpredictable results. The watchdog timer can be disabled by a ROM code option. If the watch dog timer is disabled, all the executions related to the WDT result in no operation.

Once the internal WDT oscillator (RC oscillator with period 78μs normally) is selected, it is first divided by 2^{n+5} by ROM code option. The n is from 0 to 3. The WDT prescaler output is divided by 8 (3-stage) to get the WDT time-out. This time-out period may vary with temperature, VDD and process variations. If the n is selected as 3, the maximum time-out period is divided by 2048 or about 160ms.



Watchdog timer

If the WDT oscillator is disabled, the WDT will lose its protecting purpose. In this situation the logic can only be restarted by an 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 will initialize "chip reset" and set the status bit TO. Whereas in the halt mode, the overflow will initialize a "warm reset" only the PC and SP are reset to zero. To clear the WDT contents (only clear the last 3-stage counter), three methods are adopted; external reset (a low level to $\overline{RES}$), software instruction(s), or a "HALT" instruction. The software instruction is CLR WDT. The execution of the CLR WDT instruction will clear the WDT.

Power down operation – HALT

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

- The system oscillator will turn off but the WDT oscillator keeps running (if the WDT oscillator is selected).
- The contents of the on-chip RAM and registers remain unchanged.
- WDT will be cleared and do recounting again (if the WDT clock comes from the WDT oscillator).
- All I/O ports maintain their original status.
- The PD flag is set and the TO flag is cleared.

The system can leave the halt mode by means of an external reset, an interrupt, an external falling edge signal on port A or a WDT overflow. An external reset causes a device initialization and the WDT overflow performs a "warm reset". Examining the TO and PD flags, the reason for chip reset can be determined. The PD flag is cleared when system power-up or execute the CLR WDT instruction and is set when the HALT instruction is executed. The TO flag is set if the WDT time-out occurs, and causes a wake-up that only resets the PC and SP, the others keep 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 the ROM code option. Awakening from an I/O port stimulus, the program will resume execution of the next instruction. If awakening is from an interrupt, two sequences may happen. 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. If the interrupt is enabled and the stack is not full, the regular interrupt response takes place.

Once a wake-up event(s) occurs, it takes 1024 t_{SYS} (system clock period) to resume normal operation. In other words, a dummy cycle period will be inserted after the wake-up. If the wake-up results from an interrupt acknowledgment, the actual interrupt subroutine will be delayed by one more cycle. If the wake-up results in next instruction execution, this will execute immediately after a dummy period has finished. 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.

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

Reset

There are three ways in which a reset can occur:

- $\overline{RES}$ reset during normal operation
- $\overline{RES}$ reset during halt mode
- WDT time-out reset during normal operation

The WDT time-out during halt mode is different from other chip reset conditions, since it can perform a "warm reset" that just resets the PC and SP, leaving the other circuits to keep their state. Some registers remain unchanged during

any other reset conditions. Most registers are reset to the "initial condition" when the reset conditions are met. By examining the PD and TO flags, the program can distinguish between different "chip resets".

TO	PD	RESET Conditions
0	0	System 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

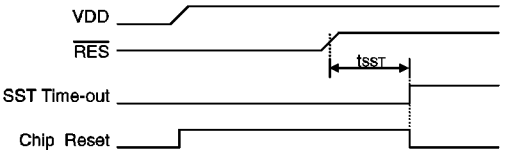
Note: "u" means that "unchanged"

To guarantee that the system oscillator has started and stabilized, the SST (System Start-up Timer) provides an extra delay of 1024 system clock pulses when the system powers up or when the system awakes from the halt mode.

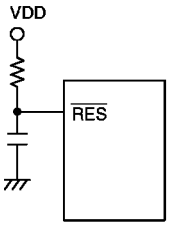
When a system power-up occurs, the SST delay is added during the reset period. But when the reset comes from the $\overline{RES}$ pin, the SST delay is disabled. Any wake-up from halt mode will enable the SST delay.

The functional unit chip reset status are shown below.

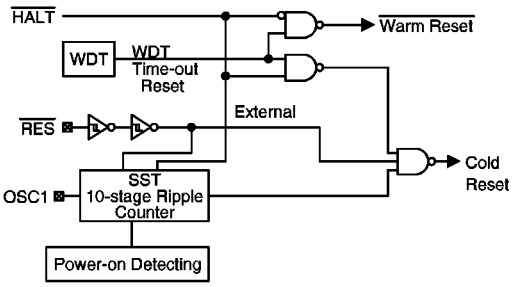
PC	000H
Interrupt	Disable
WDT	Clear. After master reset, WDT begins counting
Timer/event counter	Off
Input/output ports	Input mode
SP	Points to the top of the stack



Reset timing chart



Reset circuit



Reset configuration

The state of the registers is summarized in the following table:

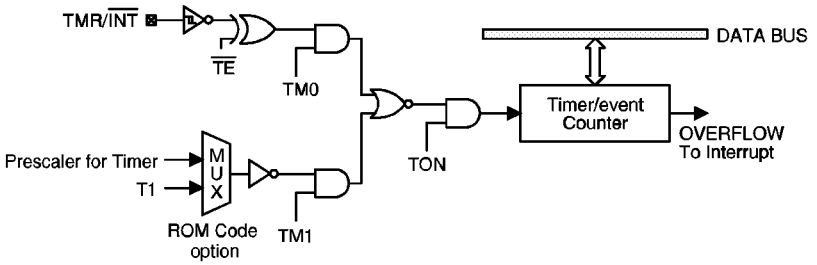
Register	Reset (power on)	WDT time-out (normal operation)	$\overline{RES}$ reset (normal operation)	$\overline{RES}$ reset (HALT)	WDT time-out (HALT)
TMR	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMRC	00-0 1---	00-0 1---	00-0 1---	00-0 1---	uu-u u---
PC	000H	000H	000H	000H	000H*
MP	-xxx xxxx	-uuu uuuu	-uuu uuuu	-uuu uuuu	-uuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC	--00 -000	--00 -000	--00 -000	--00 -000	--uu -uuu
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	---- --11	---- --11	---- --11	---- --11	---- --uu
PBC	---- --11	---- --11	---- --11	---- --11	---- --uu
PC	---- --11	---- --11	---- --11	---- --11	---- --uu
PCC	---- --11	---- --11	---- --11	---- --11	---- --uu

Notes: "*" means "warm reset"
"u" means "unchanged"
"x" means "unknown"

Timer/event counter

One timer/event counter (TMR) is implemented in the HT48R054-1/HT48R054-2. The timer/event counter contains an 8–bit programmable count-up counter and the clock may come from an external source or the internal clock source.

Using the internal clock, there is one ROM code option to select the prescaler output or to select the instruction clock (system clock divided by 4) as the reference time-base . The timer prescaler used the same prescaler with the WDT prescaler. The selection of the timer prescaler is determined by ROM code option. The external clock input allows the user to count external events, or to generate an accurate time base.



Timer/event counter

There are two registers related to the timer/event counter; TMR ([0DH]), TMRC ([0EH]). Only one physical register is mapped to TMR location; writing TMR makes the starting value be placed in the timer/event counter register and reading TMR gets the contents of the timer/event counter. The TMRC is a timer/event counter control register, which defines some options.

The TM0, TM1 bits define the operating mode. The event count mode is used to count external events, which means the clock source comes from an external (TMR) pin. The timer mode functions as a normal timer with the clock source coming from the internal clock.

In the event count or timer mode, once the timer/event counter starts counting, it will count from the current contents in the timer/event counter to FFH. Once overflow occurs, the counter generates the interrupt request flag (TF; bit 5 of INTC) and the timer/event counter is reloaded by the user in the interrupt service routine (MOV TMR, A).

To enable the counting operation, the timer on bit (TON; bit 4 of TMRC) should be set to 1. In these two modes the TON can only be reset by instruction. The overflow of the timer/event counter is one of the wake-up sources. No matter what the operation mode is, writing a 0 to ETI can disable the interrupt service.

When the timer/event counter (reading TMR) is read, the clock will be blocked to avoid errors. As this may result in a counting error, this must be taken into consideration by the programmer.

If the timer clock source comes from the WDT oscillator, the internal timer may still work in the halt mode, and wake-up from the halt mode since timer/event counter overflow occurs.

Input/output ports

There are 12 bidirectional input/output lines in the HT48R054-1/HT48R054-2, labeled from PA to PC, which are mapped to the data memory of [12H], [14H] and [16H] respectively. All these I/O ports can be used for input and output operations. For input operation, these ports are non-latching, that is, the inputs must be ready at the T2 rising edge of instruction MOV A,[m] (m=12H, 14H or 16H). For output operation, all data is latched and remains unchanged until the output latch is rewritten.

Each I/O line has its own control register (PAC, PBC, PCC) to control the input/output configuration. With this control register, CMOS output or schmitt trigger input with or without pull-high resistor (ROM code option) structures can be reconfigured dynamically (i.e., on-the-fly) under software control. To function as an input, the corresponding latch of the control register

Label (TMRC)	Bits	Function
—	0~2	Unused bits, read as "0"
TE	3	To define the TMR active edge of the timer/event counter (0=active on low to high; 1=active on high to low)
TON	4	To enable/disable the timer counting (0=disabled; 1=enabled)
—	5	Unused bits, read as "0"
TM0 TM1	6 7	To define the operating mode 01=Event count mode (External clock) 10=Timer mode (Internal clock) 00=Unused 11=Unused

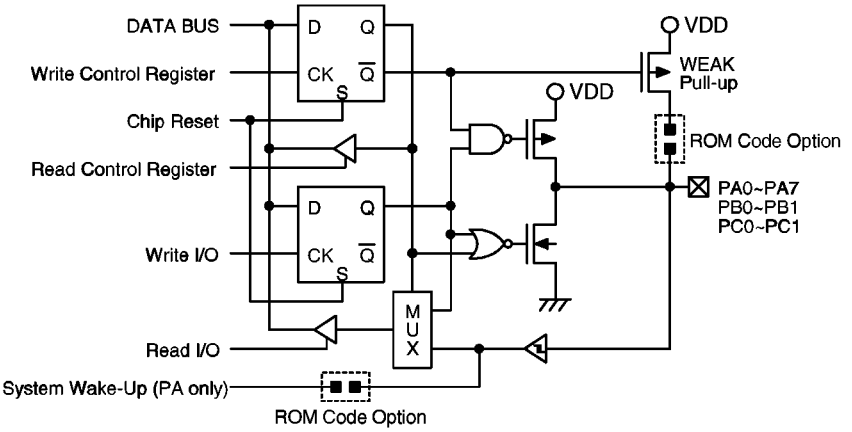
TMRC register

must write a "1". The pull-high resistance will exhibit automatically if the pull-high option is selected. The input source also depends on the control register. If the control register bit is "1", 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 "read-modify-write" instruction. For output function, CMOS is the only configuration. These control registers are mapped to locations 13H, 15H and 17H.

After a chip reset, these input/output lines remain at high levels or floating (ROM code option). Each bit of these input/output latches can be set or cleared by "SET [m].i" and "CLR [m].i" (m=12H, 14H or 16H) instructions.

Some instructions first input data and then follow the output operations. For example, the "SET [m].i", "CPL [m]", "CPLA [m]" and "CLR [m].i" instructions 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 to wake-up the device. The highest six bits of port B and port C are not physically implemented, on reading them a "0" is returned and writing results in no operation.



Input/output ports

ROM code option

The following shows eight kinds of ROM code option in the HT48R054-1/HT48R054-2. All the ROM code options must be defined to ensure proper system function.

No.	ROM code option
1	OSC type selection. This option is to decide if an RC or crystal or 32768Hz crystal oscillator is chosen as system clock.
2	Prescaler source selection. There are two types of selection: on-chip RC oscillator, instruction clock.
3	The WDT time-out period. This option is to decide the WDT time-out period which is the WDT input source divided by 2^{n+8}. The number n is from 0 to 3.
4	Wake-up selection. This option defines the wake-up activity. External I/O pins (PA only) all have the capability to wake-up the chip from a halt mode.
5	Pull-high selection. This option is to decide whether the pull-high resistance is visible or not in the input mode of the I/O ports. Each bit of an I/O port can be independently selected.
6	Prescaler ROM code option for Timer. This option is to decide the prescaler for the timer which is divided by 2^{n+1} . The number n is from 0 to 7.
7	Internal timer clock source ROM code option. There are two types of selection: instruction clock or prescaler for the timer.
8	WDT selection. This option defines the WDT to be enabled or disabled.

ROM code option address table

Address	Bit															
PC[15:0]	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
8000H	LOCK	0	OP13	OP12	X	X	OP9	OP8	OP7	OP6	OP5	OP4	OP3	OP2	OP1	OP0
8001H	LOCK	0	OP27	OP26	OP25	OP24	OP23	OP22	OP21	OP20	OP19	OP18	OP17	OP16	OP15	OP14
8002H	LOCK	0	0	0	0	0	0	0	0	0	0	0	OP31	OP30	OP29	OP28

"X" means "unknown"

Items	Option	Description
1	OP[7:0]	OPA0~OPA7 → PA0~PA7 wake-up pin Bit=0 without wake up Bit=1 with wake up
2	OP[9:8]	System Oscillator 00 → 32768Hz crystal mode 01 → RC mode 10 → RC mode 11 → Crystal mode
3	OP12	Bit=0 RC clock for prescaler source Bit=1 Instruction clock for prescaler source
4	OP13	Bit=1 Disable WDT Bit=0 Enable WDT
5	OP[27:20]	OP20~OP27 → PA0~PA7 input pull-high resistor Bit=0 without pull-high resistor Bit=1 with pull-high resistor
6	OP[29:28]	OP28~OP29 → PB0~PB1 input pull-high resistor Bit=0 without pull-high resistor Bit=1 with pull-high resistor
7	OP[31:30]	OP30~OP31 → PC0~PC1 input pull-high resistor Bit=0 without pull-high resistor Bit=1 with pull-high resistor
8	OP[18:16]	TIMER prescaler option OP18~OP16 → Divided by 2^1~2^8 000B → 2^1 001B → 2^2 010B → 2^3 011B → 2^4 100B → 2^5 101B → 2^6 110B → 2^7 111B → 2^8
9	OP[15:14]	The WDT time-out period option OP15~OP14 → Divided by 2^8~2^{11} 00B → 2^8 01B → 2^9 10B → 2^{10} 11B → 2^{11}
10	OP19	TIMER clock source Bit=0 instruction clock (T1) Bit=1 timer prescaler output
11	LOCK	Bit=0 normal write and read Bit=1 can just write but read out “1” only

EPROM programming and verification

The program memory used in the HT48R054-1/HT48R054-2 is arranged into a 512×14 bits program PROM and a 3×14 bits option PROM. The program code and option code are stored in the program PROM and option PROM. The programming of PROM can be summarized in nine steps as described below:

- Power on
- Set VPP ($\overline{RES}$) to 12.5V
- Set $\overline{CS}$ (PA5) to low

Let PA3~PA0 (AD3~AD0) be the address and data bus and the PA4 (CLK) be the clock input. The data on the AD3~AD0 pins will be clocked into or out the HT48R054-1/HT48R054-2 on the falling edge of PA4 (CLK) for PROM programming and verification.

The address data contains the code address (9 bits) and two option bits. A complete write cycle will contain 4 CLK cycles. The first cycle, bit 0~3 of the address are latched into the HT48R054-1/HT48R054-2. The second and third cycles, bit 4~7 and bit 8 are latched respectively. The fourth cycle, bit 2 is the TSEL option bit and bit 3 is the OSEL option bit. Bit 1~3 in the third cycle and bit 0~1 in the fourth cycle are undefined. If the TSEL is "1" and the OSEL is "0", the TEST memory will be read. If the TSEL is "0" and the OSEL is "1", the option PROM will be accessed. If both the TSEL and OSEL are "0", the program PROM will be managed.

The code data is 14 bits wide. A complete read/write cycle contains 4 CLK cycles. In the first cycle, bit 0~3 of the code data are accessed. In the second and third, bit 4~7 and bit 8~11 are accessed respectively. In the fourth cycle, bit 12~13 are accessed. Bit 14~15 are undefined. During code verification, reading will return the result "00".

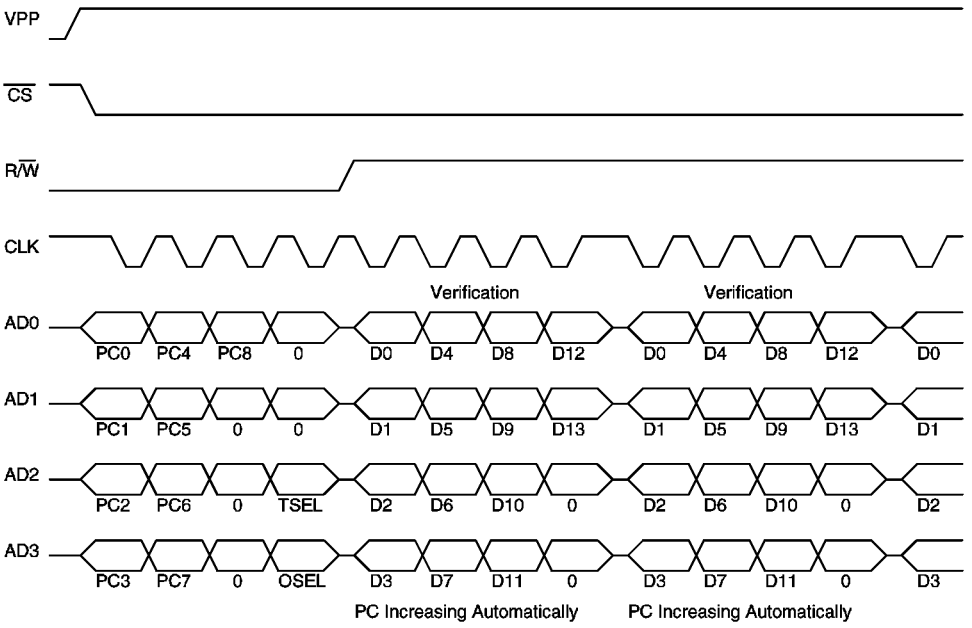
Select the TSEL and OSEL to program and verify the program PROM and option PROM. Use the R/$\overline{W}$ (PA6) to select the programming or verification

The address is incremented by one automatically after a code verification cycle. If the uncontinued address programming or verification is accomplished, the automatic addressing increment is disabled. For the uncontinued address programming and verification, the $\overline{CS}$ pin must return to a high level for programming or verification cycle, that is, if an uncontinued address is managed, the programming or verification cycle must be interrupted and restarted as well.

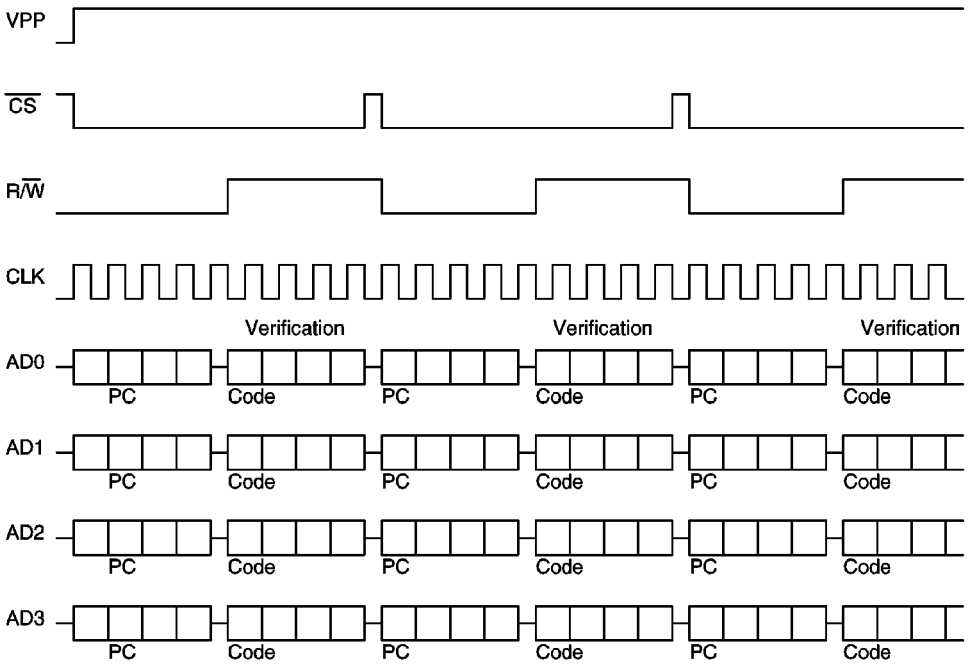
The related pins of PROM programming and verification are listed in the following table.

Pin Name	Function	Description
PA0	AD0	Bit 0 of address/data bus
PA1	AD1	Bit 1 of address/data bus
PA2	AD2	Bit 2 of address/data bus
PA3	AD3	Bit 3 of address/data bus
PA4	CLK	Serial clock input for address and data
PA5	$\overline{CS}$	Chip select, active low
PA6	R/$\overline{W}$	Read/write control input
$\overline{RES}$	VPP	Programming power supply

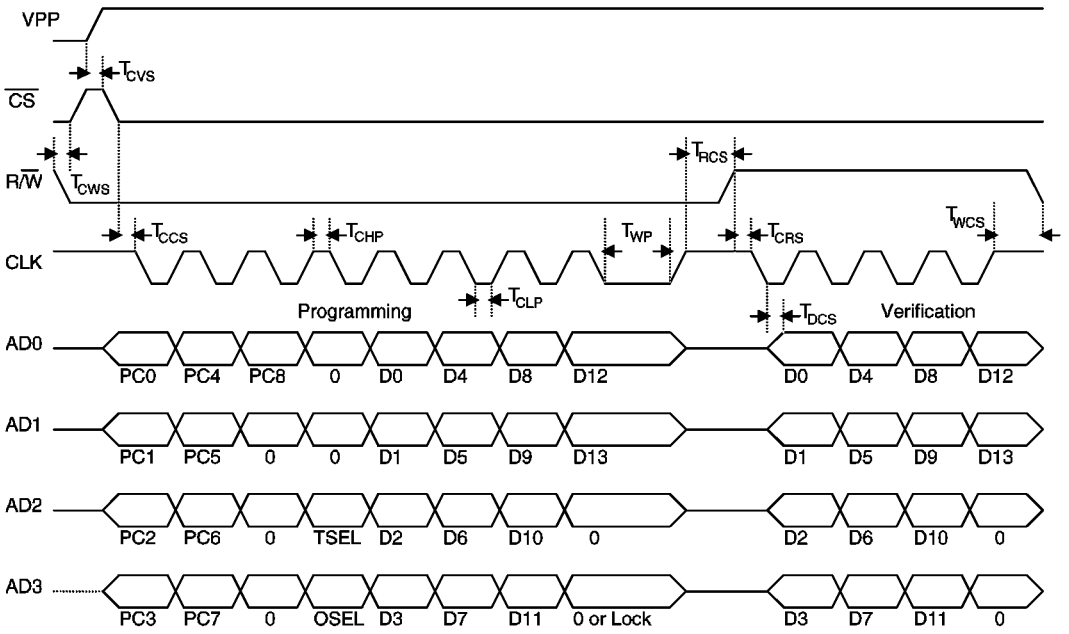
The timing charts of programming and verification are as shown. There is a LOCK signal for code protection. If the LOCK is "1", reading the code will return the result "1". However, if the LOCK is "0", the code protection is disabled and the code can always be read until the LOCK is programmed as "1".



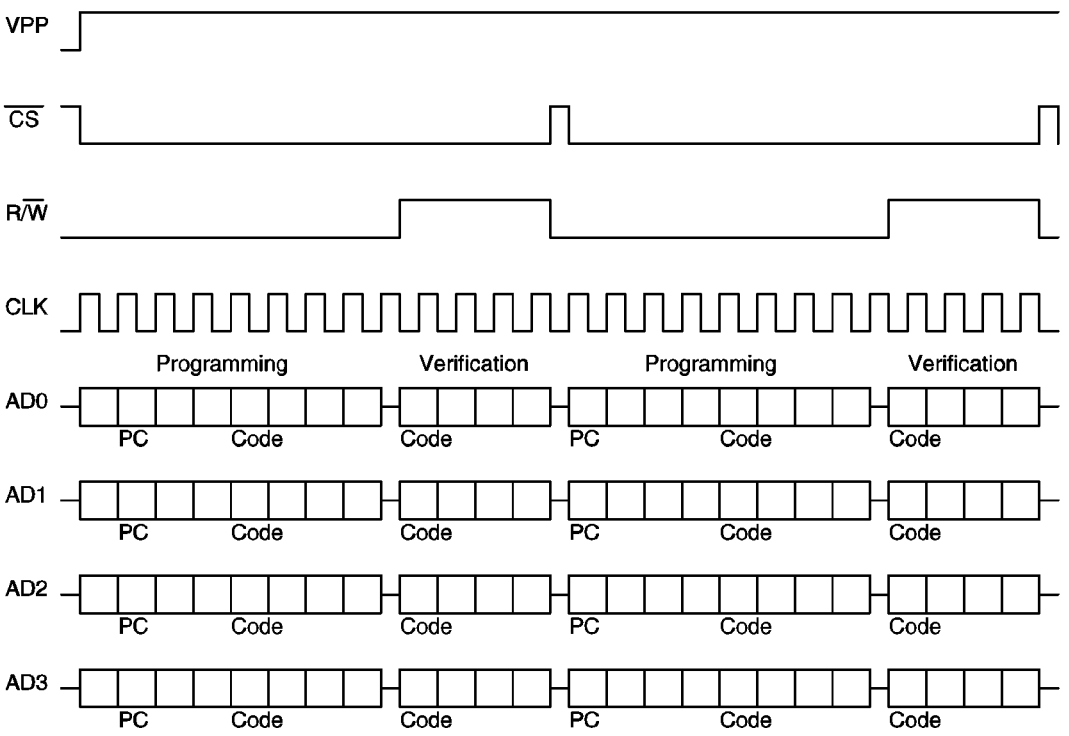
Successive verification



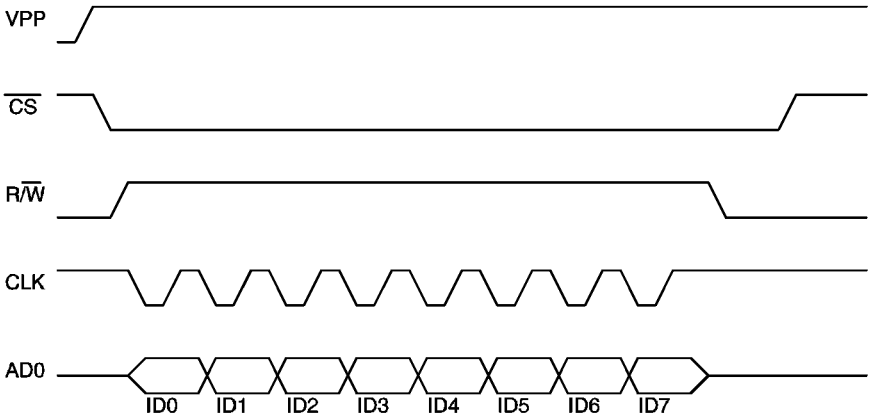
Non-successive verification



Code programming and verification

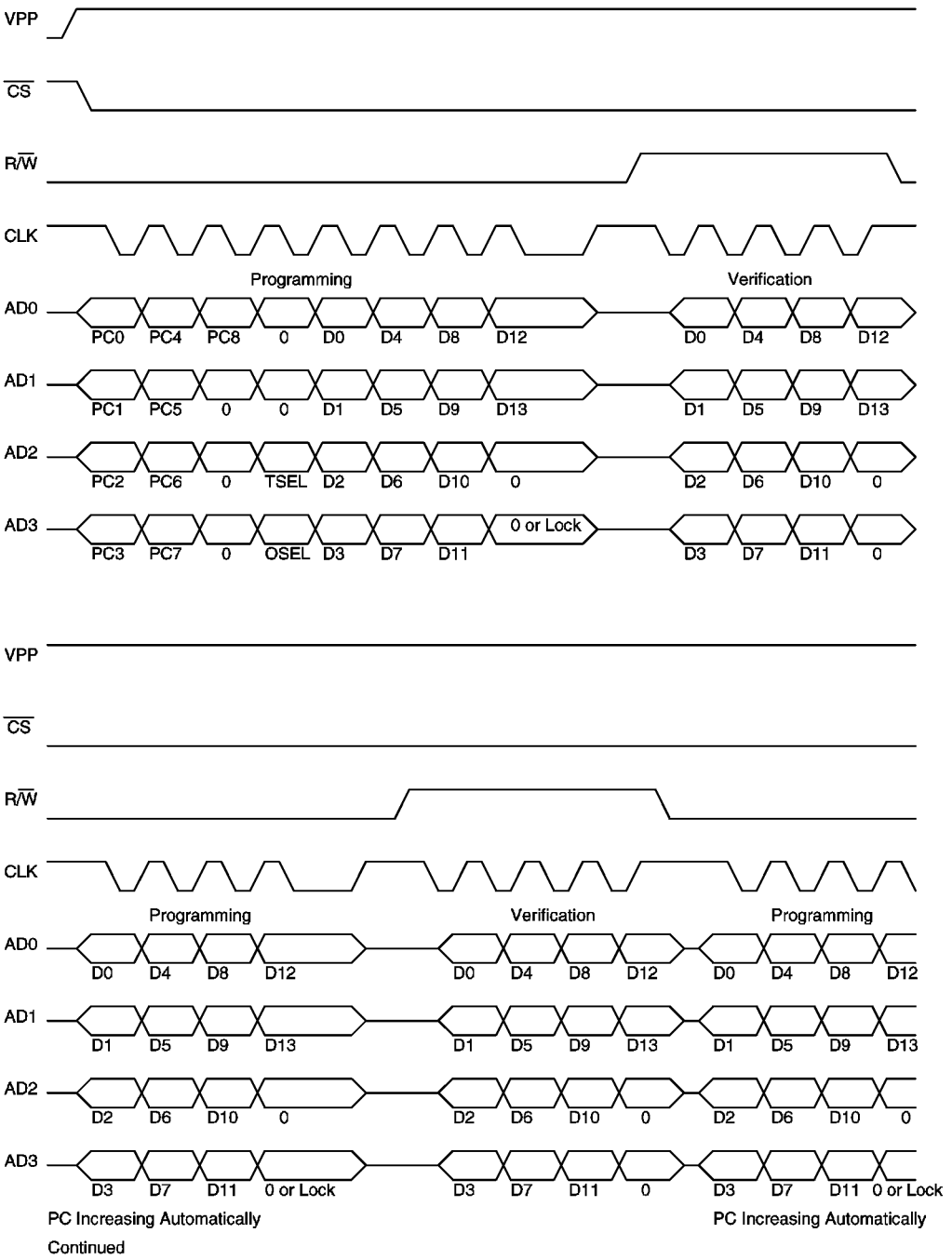


Non-successive programming and verification



AD1, AD2, AD3 : Don't Care

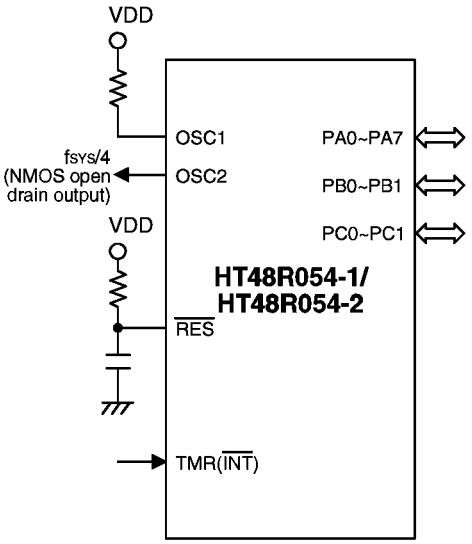
ID code verification



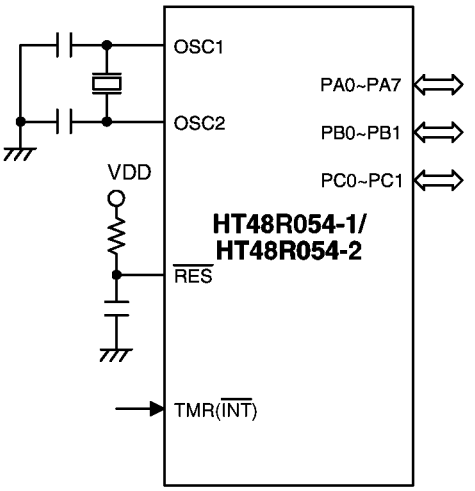
Successive programming and verification

Application Circuits

RC oscillator for multiple I/O applications



Crystal oscillator or ceramic resonator for multiple I/O applications



Instruction Set Summary

Mnemonic	Description	Flag Affected
Arithmetic		
ADD A,[m]	Add data memory to ACC	Z,C,AC,OV
ADDM A,[m]	Add ACC to data memory	Z,C,AC,OV
ADD A,x	Add immediate data to ACC	Z,C,AC,OV
ADC A,[m]	Add data memory to ACC with carry	Z,C,AC,OV
ADCM A,[m]	Add ACC to register with carry	Z,C,AC,OV
SUB A,x	Subtract immediate data from ACC	Z,C,AC,OV
SUB A,[m]	Subtract data memory from ACC	Z,C,AC,OV
SUBM A,[m]	Subtract data memory from ACC with result in data memory	Z,C,AC,OV
SBC A,[m]	Subtract data memory from ACC with carry	Z,C,AC,OV
SBCM A,[m]	Subtract data memory from ACC with carry with result in data memory	Z,C,AC,OV
Logic Operation		
AND A,[m]	AND data memory to ACC	Z
OR A,[m]	OR data memory to ACC	Z
XOR A,[m]	Exclusive-OR data memory to ACC	Z
ANDM A,[m]	AND ACC to data memory	Z
ORM A,[m]	OR ACC to data memory	Z
XORM A,[m]	Exclusive-OR ACC to data memory	Z
AND A,x	AND immediate data to ACC	Z
OR A,x	OR immediate data to ACC	Z
XOR A,x	Exclusive-OR immediate data to ACC	Z
CPL [m]	Complement data memory	Z
CPLA [m]	Complement data memory with result in ACC	Z
Increment & Decrement		
INCA [m]	Increment data memory with result in ACC	Z
INC [m]	Increment data memory	Z
DECA [m]	Decrement data memory with result in ACC	Z
DEC [m]	Decrement data memory	Z
Rotate		
RRA [m]	Rotate data memory right with result in ACC	None
RR [m]	Rotate data memory right	None
RRCA [m]	Rotate data memory right through carry with result in ACC	C
RRC [m]	Rotate data memory right through carry	C
RLA [m]	Rotate data memory left with result in ACC	None
RL [m]	Rotate data memory left	None
RLCA [m]	Rotate data memory left through carry with result in ACC	C
RLC [m]	Rotate data memory left through carry	C

Mnemonic	Description	Flag Affected
Data Move		
MOV A,[m]	Move data memory to ACC	None
MOV [m],A	Move ACC to data memory	None
MOV A,x	Move immediate data to ACC	None
Bit Operation		
CLR [m].i	Clear bit of data memory	None
SET [m].i	Set bit of data memory	None
Branch		
JMP addr	Jump unconditional	None
SZ [m]	Skip if data memory is zero	None
SZA [m]	Skip if data memory is zero with data movement to ACC	None
SZ [m].i	Skip if bit i of data memory is zero	None
SNZ [m].i	Skip if bit i of data memory is not zero	None
SIZ [m]	Skip if increment data memory is zero	None
SDZ [m]	Skip if decrement data memory is zero	None
SIZA [m]	Skip if increment data memory is zero with result in ACC	None
SDZA [m]	Skip if decrement data memory is zero with result in ACC	None
CALL addr	Subroutine call	None
RET	Return from subroutine and interrupt	None
RET A,x	Return from subroutine and load immediate data to ACC	None
RETI	Return from interrupt	None
Miscellaneous		
NOP	No operation	None
CLR [m]	Clear data memory	None
SET [m]	Set data memory	None
CLR WDT	Clear Watchdog timer	TO,PD
HALT	Enter power down mode	TO,PD

Notes: x = 8 bits immediate data

m = 6 bits data memory address

A = accumulator

i = 0...7 number of bits

addr = 9 bits program memory address

√ = Flag is affected

– = Flag is not affected

Instruction Definition

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

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	√	√	√	√

ADCM A,[m] Add 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)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	√	√	√	√

ADD A,[m] Add data memory to 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)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	√	√	√	√

ADD A,x Add immediate data to 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)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	√	√	√	√

ADDM A,[m] Add accumulator to 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)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	√	√	√	√

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

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

Operation ACC ← ACC "AND" [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	√	√	–	–

AND A,x Logical AND immediate data to accumulator

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

Operation ACC ← ACC "AND" x

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	√	√	–	–

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

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

Operation [m] ← ACC "AND" [m]

Affected flag(s)

TC2	TC1	TO	PD	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 ← PC+1
PC ← addr

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

CLR [m] Clear data memory

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

Operation [m] ← 00H

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

CLR [m].i Clear bit of data memory

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

Operation [m].i ← 0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

CLR WDT Clear watchdog timer

Description The WDT is cleared (the last 3-stage counter re-counting from zero). The power down bit (PD) and time-out bit (TO) are cleared.

Operation WDT last three bits ← 00H
PD & TO ← 0

Affected flag(s)

TC2	TC1	TO	PD	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 contain a one are changed to zero and vice-versa.

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

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	√	–	–

CPLA [m] Complement data memory-place result in accumulator

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

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

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	√	–	–

DEC [m] Decrement data memory

Description Data in the specified data memory is decremented by one.

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

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	√	–	–

DECA [m] Decrement data memory-place result in accumulator

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

Operation $ACC \leftarrow [m]-1$

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	√	–	–

HALT Enter power down mode

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

Operation PC ← PC+1
PD ← 1
TO ← 0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	0	1	–	–	–	–

INC [m] Increment data memory

Description Data in the specified data memory is incremented by one.

Operation [m] ← [m]+1

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	√	–	–

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

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

Operation ACC ← [m]+1

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	√	–	–

JMP ADDR Direct Jump

Description Bits 0~8 of the program counter are replaced with the directly–specified address unconditionally, and control passed to this destination.

Operation PC ← addr

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

MOV A,[m] Move data memory to accumulator

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

Operation ACC ← [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

MOV A,x Move immediate data to accumulator

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

Operation ACC ← x

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

MOV [m],A Move accumulator to data memory

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

Operation [m] ← ACC

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

NOP No operation

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

Operation PC ← PC+1

Affected flag(s)

TC2	TC1	TO	PD	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 memory) performs a bitwise logical_OR operation. The result is stored in the accumulator.

Operation ACC ← ACC "OR" [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	√	–	–

OR A,x Logical OR immediate data to accumulator

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

Operation ACC ← ACC "OR" x

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	√	–	–

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

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

Operation [m] ← ACC "OR" [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	√	–	–

RET Return from subroutine

Description The program counter is restored from the stack. This is a two cycle instruction.

Operation PC ← Stack

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

RET A,x Return and place immediate data in accumulator

Description The program counter is restored from the stack and the accumulator loaded with the specified 8-bit immediate data. This is a two cycle instruction.

Operation PC ← Stack
ACC ← x

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

RETI Return from interrupt

Description The program counter is restored from the stack, and interrupts enabled by setting the EMI bit. EMI is the enable master (global) interrupt bit (bit 0; register INTC).

Operation PC ← Stack
EMI ← 1

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

RL [m] Rotate data memory left

Description The contents of the specified data memory is rotated left one bit 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)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

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

Description Data in the specified data memory is rotated left one bit 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)

TC2	TC1	TO	PD	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 together rotated left one bit. 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)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	√

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

Description Data in the specified data memory and the carry flag are together rotated left one bit. 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)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	√

RR [m] Rotate data memory right

Description The contents of the specified data memory are rotated right one bit 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)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

RRA [m] Rotate right and place result in accumulator

Description Data in the specified data memory is rotated right one bit 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)

TC2	TC1	TO	PD	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 right one bit. 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)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	√

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

Description Data of the specified data memory and the carry flag are together rotated right one bit. 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)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	√

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

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

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

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	√	√	√	√

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

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

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

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	√	√	√	√

SDZ [m] Skip if decrement data memory is zero

Description The contents of the specified data memory are decremented by one. If the result is zero, the next instruction is skipped. If the result is zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle replaced to get the proper instruction. This makes a 2-cycle instruction. Otherwise proceed with the next instruction.

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

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

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

Description The contents of the specified data memory are decremented by one. If the result is zero, the next instruction is skipped. The result is stored in the accumulator but the data memory remains unchanged. If the result is zero ,the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction, that makes a 2-cycle instruction. Otherwise proceed with the next instruction.

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

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

SET [m] Set data memory

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

Operation [m] ← FFH

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

SET [m].i Set bit of data memory

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

Operation [m].i ← 1

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

SIZ [m] Skip if increment data memory is zero

Description The contents of the specified data memory is incremented by one. If the result is zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction. This is a 2 cycle instruction. Otherwise proceed with the next instruction.

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

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

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

Description The contents of the specified data memory is incremented by one. If the result is zero, the next instruction is skipped and the result stored in the accumulator. The data memory remains unchanged. If the result is zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle replaced to get the proper instruction. This is a 2-cycle instruction. Otherwise proceed with the next instruction.

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

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

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

Description If bit i of the specified data memory is not zero, the next instruction is skipped. If bit i of the data memory is not zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction. This is a 2-cycle instruction. Otherwise proceed with the next instruction.

Operation Skip if [m].i≠0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

SUB A,[m] Subtract data memory from 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)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	√	√	√

SUBM A,[m] Subtract data memory from 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)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	√	√	√	√

SUB A,x Subtract immediate data from 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)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	√	√	√	√

SZ [m] Skip if data memory is zero

Description If the contents of the specified data memory is zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction. This is a 2-cycle instruction. Otherwise proceed with the next instruction.

Operation Skip if [m]=0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

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

Description The contents of the specified data memory is copied to accumulator. If the contents is zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction. This is a 2-cycle instruction. Otherwise proceed with the next instruction.

Operation Skip if [m]=0, ACC ← [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

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

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

Operation Skip if [m].i=0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	–	–	–

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

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

Operation ACC ← ACC "XOR" [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	√	–	–

XORM A,[m] Logical XOR data memory with 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 zero flag is affected.

Operation [m] ← ACC "XOR" [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	√	–	–

XOR A,x Logical XOR immediate data to 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 zero flag is affected.

Operation ACC ← ACC "XOR" x

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
–	–	–	–	–	√	–	–