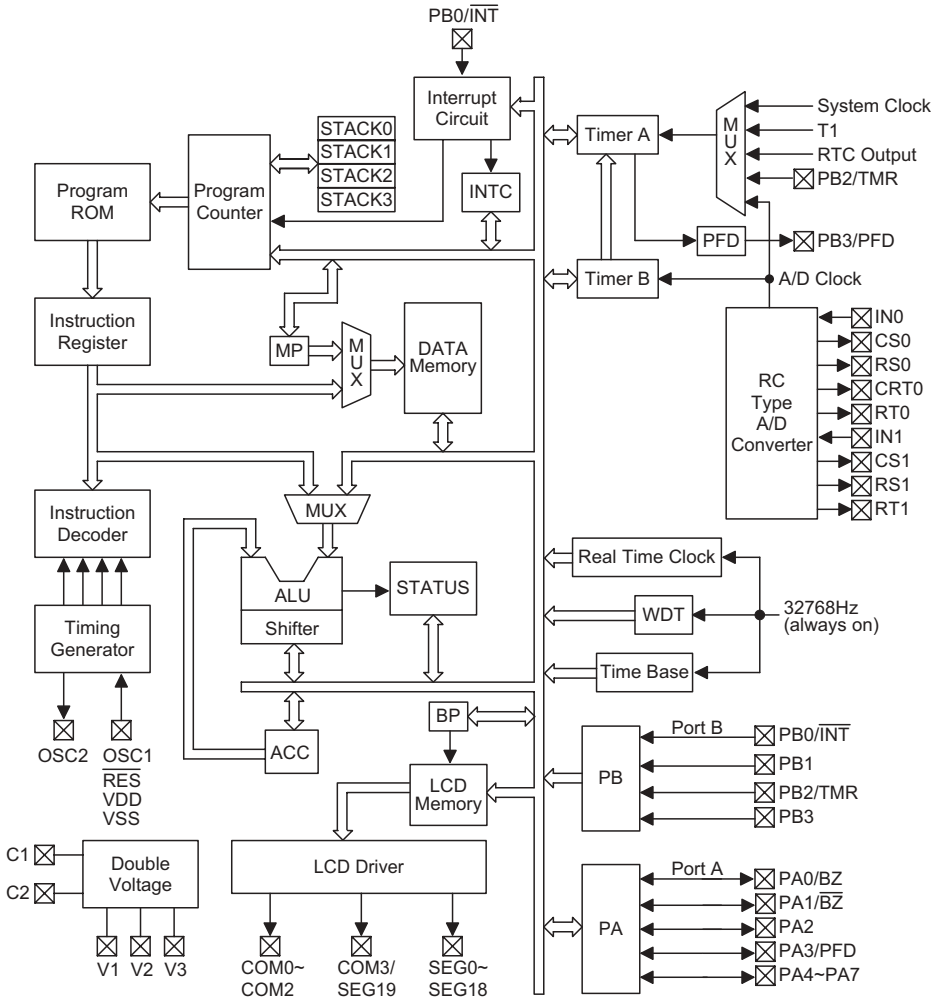
Features

- Operating voltage: 1.2V~2.2V
- Eight bidirectional I/O lines
- Four input lines
- One interrupt input
- One 16-bit programmable timer/event counter with PFD (programmable frequency divider) function
- On-chip 32768Hz crystal oscillator
- Watchdog Timer
- 2K×16 program memory ROM
- 64×8 data memory RAM
- One real time clock (RTC)
- One 8-bit prescaler for real time clock
- One buzzer output
- One low voltage detector
- One low voltage reset circuit
- HALT function and wake-up feature reduce power consumption
- LCD bias C type
- One LCD driver with 20×2 or 20×3 or 19×4 segments
- Two channels RC type A/D converter
- Four-level subroutine nesting
- Bit manipulation instruction
- 16-bit table read instruction
- Up to 122μs instruction cycle with 32768Hz system clock
- All instructions in one or two machine cycles
- 63 powerful instructions

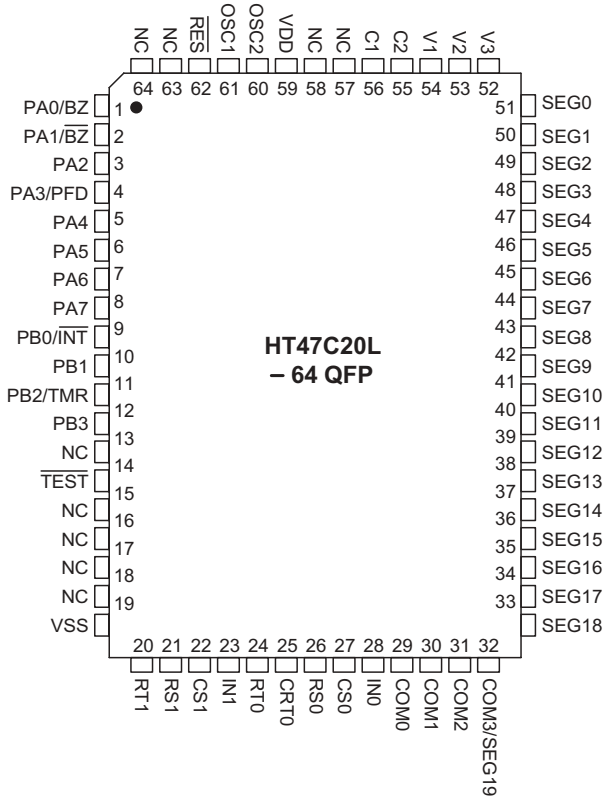
General Description

The HT47C20L is an 8-bit high performance RISC-like microcontroller. Its single cycle instruction and two-stage pipeline architecture make high speed applications. The device is suited for use in multiple LCD low power applications among which are calculators, clock timers, games, scales, toys, thermometers, hygrometers, body thermometers, capacitor scaler, other hand held LCD products, and battery system in particular.

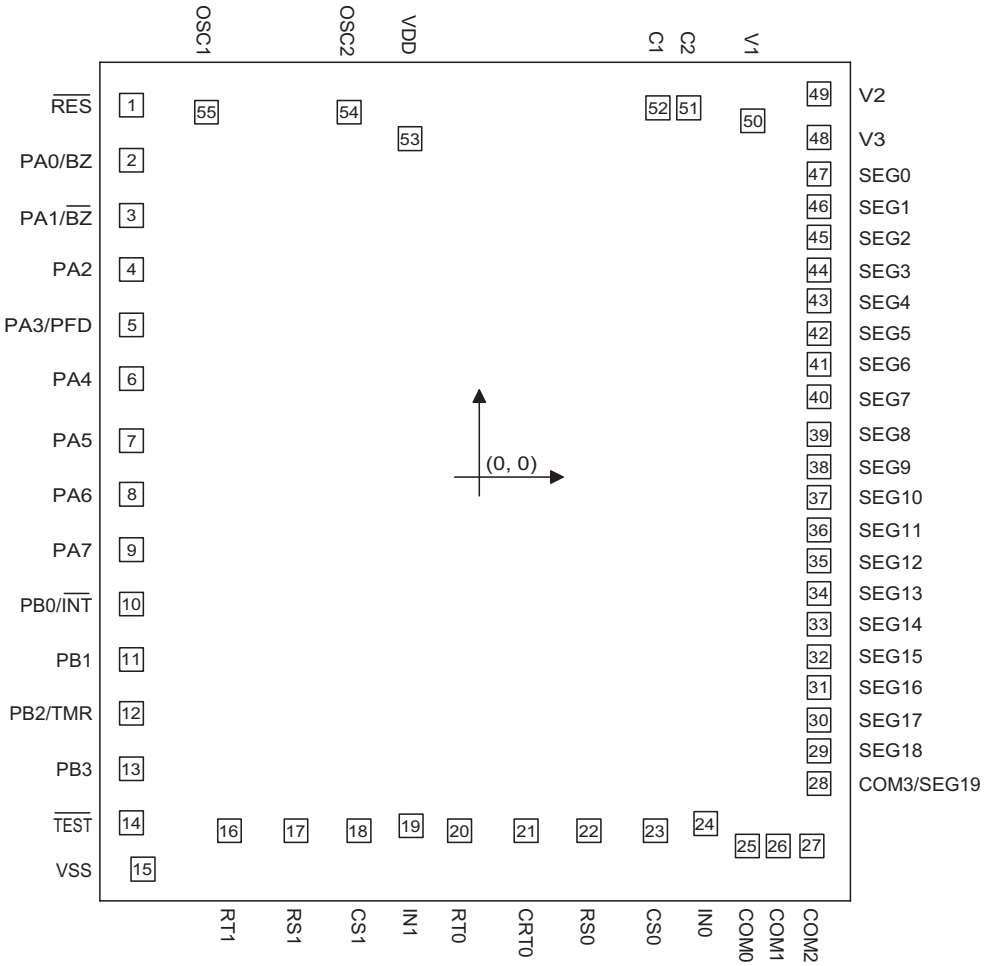
Block Diagram



Pin Assignment



Pad Assignment



* The IC substrate should be connected to VSS in the PCB layout artwork.

Pin Description

Pin Name	I/O	Mask Option	Function
$\overline{RES}$	I	—	Schmitt trigger reset input. Active low.
PA0/BZ PA1/$\overline{BZ}$ PA2 PA3/PFD PA4~PA7	I/O	Wake-up Pull-high or None CMOS or NMOS	Bidirectional 8-bit input/output port. The low nibble of the PA can be configured as CMOS output or NMOS output with or without pull-high resistors (mask option). NMOS output can be configured as Schmitt trigger input with or without pull-high resistors. Each bit of NMOS output can be configured as wake up input by mask option. Of the eight bits, PA0~PA1 can be set as I/O pins or buzzer outputs by mask option. PA3 can be set as an I/O pin or a PFD output by mask option.
PB0/$\overline{INT}$ PB1 PB2/TMR PB3	I	—	Four-bit Schmitt trigger input port. The PB is configured as with pull-high resistors. Of the four bits, PB0 can be set as an input pin or an external interrupt input pin ($\overline{INT}$) by software application. While PB2 can be set as an input pin or a timer/event counter input pin by software application.
VSS	—	—	Negative power supply, ground
V1~V3, C1~C2	—	—	Voltage pump
SEG19/COM3 COM2~COM0	O	or 1/3 or 1/4 Duty	SEG19/COM3 can be set as a segment or a common output driver for LCD panel by mask option. COM2~COM0 are outputs for LCD panel plate.
SEG18~SEG0	O	—	LCD driver outputs for LCD panel segments
VDD	—	—	Positive power supply
OSC2 OSC1	O I	—	OSC1 and OSC2 are connected to a 32768Hz crystal for the internal system clock and WDT source.
IN0 CS0 RS0 CRT0 RT0	I O O O O	—	Oscillation input pin of channel 0 Reference capacitor connection pin of channel 0 Reference resistor connection pin of channel 0 Resistor/capacitor sensor connection pin for measurement of channel 0 Resistor sensor connection pin for measurement of channel 0
IN1 CS1 RS1 RT1	I O O O	—	Oscillation input pin of channel 1 Reference capacitor connection pin of channel 1 Reference resistor connection pin of channel 1 Resistor sensor connection pin for measurement of channel 1
$\overline{TEST}$	I	—	TEST mode input pin with pull-high resistor. It disconnects in normal operation.

Absolute Maximum Ratings

Supply Voltage .....–0.3V to 2.5V

Input Voltage.....V_{SS}–0.3V to V_{DD}+0.3V

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

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

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

D.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	1.2	1.5	2.2	V
V_{LVD}	Low Voltage Detector Voltage	—	—	1.1	1.2	1.3	V
V_{LVR}	Low Voltage Reset Voltage	—	—	1.0	1.1	1.2	V
I_{DD1}	Operating Current (LVR Disable, LVD Disable)	1.5V	No load, f_{SYS}=32768Hz A/D Off, LVD Off	—	4	8	μA
I_{DD2}	Operating Current (LVR Disable, LVD Enable)	1.5V	No load, f_{SYS}=32768Hz A/D Off, LVD Off	—	9	15	μA
I_{DD3}	Operating Current (LVR Enable, LVD Enable)	1.5V	No load, f_{SYS}=32768Hz A/D Off, LVD Off	—	12	20	μA
I_{STB1}	Standby Current (LVR Disable, LVD Disable, LCD Off)	1.5V	No load, system HALT A/D Off, LVD Off	—	1	2	μA
I_{STB2}	Standby Current (LVR Disable, LVD Enable, LCD On)	1.5V	No load, system HALT A/D Off, LVD Off	—	6	10	μA
I_{STB3}	Standby Current (LVR Enable, LVD Enable, LCD On)	1.5V	No load, system HALT A/D Off, LVD Off	—	9	15	μA
I_{STB4}	Standby Current (LVR Disable, LVD Enable, LCD On)	1.5V	No load, system HALT A/D Off, LVD On	—	8	15	μA
I_{STB5}	Standby Current (LVR Off, LVD Disable, LCD Off)	1.5	No load, system HALT A/D On *R=5.1kΩ, *C=500P	—	270	500	μA
V_{IL}	Input Low Voltage for I/O Ports	1.5V	—	0	—	0.45	V
V_{IH}	Input High Voltage for I/O Ports	1.5V	—	1.05	—	1.5	V
V_{IL1}	Input Low Voltage ($\overline{RES}$)	1.5V	$0.5V_{DD}$	0	—	0.75	V
V_{IL2}	Input Low Voltage ($\overline{INT}$, TMR)	1.5V	$0.3V_{DD}$	0	—	0.45	V
V_{IH1}	Input High Voltage ($\overline{RES}$, $\overline{INT}$, TMR)	1.5V	$0.8V_{DD}$	1.2	—	1.5	V
I_{OL}	I/O Port Sink Current	1.5V	V_{OL}=0.15V	0.3	0.6	—	mA
I_{OH}	I/O Port Source Current	1.5V	V_{OH}=1.35V	–0.2	–0.4	—	mA
I_{OL1}	Common 0~3 Output Sink Current	1.5V	V_{OL}=0.3V (1/2 bias)	120	230	—	μA
I_{OH1}	Common 0~3 Output Source Current	1.5V	V_{OH}=2.7V (1/2 bias)	–50	–100	—	μA
I_{OL2}	Segment 0~19 Output Sink Current	1.5V	V_{OL}=0.3V (1/2 bias)	30	60	—	μA
I_{OH2}	Segment 0~19 Output Source Current	1.5V	V_{OH}=2.7V (1/2 bias)	–20	–30	—	μA
I_{OL3}	Common 0~3 Output Sink Current	1.5V	V_{OL}=0.45V (1/3 bias)	120	220	—	μA
I_{OH3}	Common 0~3 Output Source Current	1.5V	V_{OH}=4.05V (1/3 bias)	–50	–100	—	μA
I_{OL4}	Segment 0~19 Output Sink Current	1.5V	V_{OL}=0.45V (1/3 bias)	30	60	—	μA
I_{OH4}	Segment 0~19 Output Source Current	1.5V	V_{OH}=4.05V (1/3 bias)	–20	–30	—	μA
I_{OL5}	RC oscillation Output Sink Current	1.5V	V_{OL}=0.15V	2	2.7	—	mA
I_{OH5}	RC oscillation Output Source Current	1.5V	V_{OH}=1.35V	–2	–3.1	—	mA
R_{PH1}	Pull-high Resistance of I/O Ports and $\overline{INT}$	1.5V	—	100	150	200	kΩ
R_{PH2}	Pull-high Resistance of $\overline{TEST}$	1.5V	—	10	30	60	kΩ

Note: *R means the resistance of RC type A/D converter

*C means the capacitance of RC type A/D converter

A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS1}	System Clock	1.5V	—	—	32768	—	Hz
f_{TIMER}	Timer I/P Frequency (TMR)	1.5V	—	0	—	32768	Hz
t_{RES}	External Reset Low Pulse Width	—	—	100	—	—	μs
t_{SST}	System Start-up Timer Period	—	—	—	8192	—	t_{SYS}
t_{INT}	Interrupt Pulse Width	1.5V	—	100	—	—	μs
t_{RIS}	Power Supply Rise Time	—	Power-up	—	—	1	s
t_{LVD}	Low Voltage Detector Response Time	1.5V	—	200	—	—	μs
f_{AD}	A/D Converter Frequency	1.5V	—	—	—	500	kHz

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

Functional Description

Execution flow

The HT47C20L system clock is derived from a 32768Hz crystal oscillator. The system clock is internally divided into four non-overlapping clocks (T1, T2, T3 and T4). 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 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.

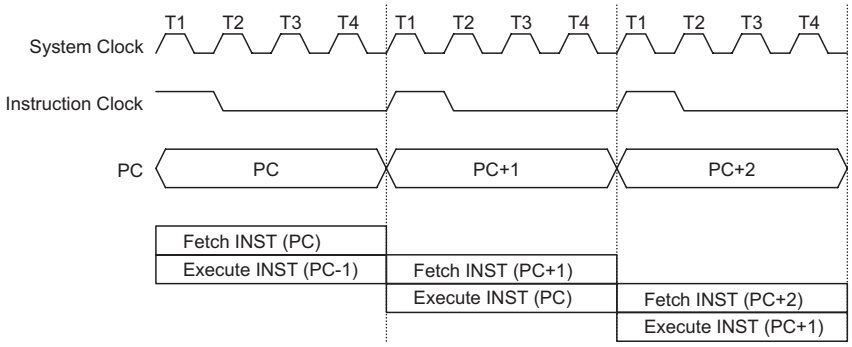
After accessing a program memory word to fetch an instruction code, the contents of the program counter are incremented by 1. The program counter then points to the memory word containing the next instruction code.

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.



Execution flow

Mode	Program Counter										
	*10	*9	*8	*7	*6	*5	*4	*3	*2	*1	*0
Initial reset	0	0	0	0	0	0	0	0	0	0	0
External interrupt	0	0	0	0	0	0	0	0	1	0	0
Time base interrupt	0	0	0	0	0	0	0	1	0	0	0
Real time clock interrupt	0	0	0	0	0	0	0	1	1	0	0
Timer/event counter interrupt	0	0	0	0	0	0	1	0	0	0	0
Skip	PC+2										
Loading PCL	*10	*9	*8	@7	@6	@5	@4	@3	@2	@1	@0
Jump, call branch	#10	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
Return from subroutine	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

Program counter

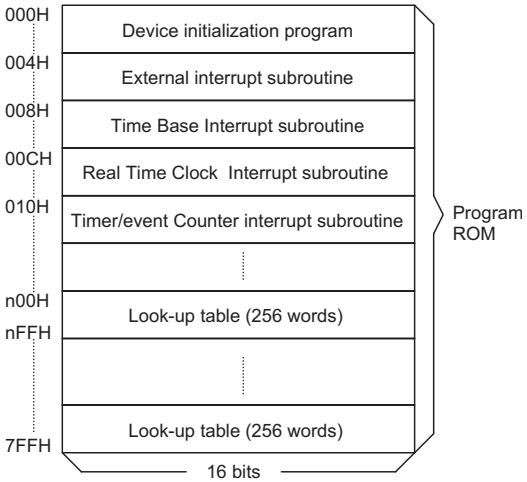
Note: *10~*0: Program counter bits
#10~#0: Instruction code bits
S10~S0: Stack register bits
@7~@0: PCL bits

Program memory – ROM

The program memory is used to store the program instructions which are to be executed. It also contains data, table, and interrupt entries, and is organized into 2048×16 bits, addressed by the program counter and table pointer.

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 the stack is not full, the program begins execution at location 004H.
- Location 008H
  This area is reserved for the time base interrupt service program. If time base interrupt resulting from a time base overflow, and if the interrupt is enabled and the stack is not full, the program begins execution at location 008H.



Note: n ranges from 0 to 7

Program memory

- Location 00CH
  This area is reserved for the real time clock interrupt service program. If a real time clock interrupt occurs, and if the interrupt is enabled and the stack is not full, the program begins execution at location 00CH.
- Location 010H
  This area is reserved for the timer/event counter interrupt service program. If timer interrupt results from a Timer/Event Counter A or B overflow, and if the interrupt is enabled and the stack is not full, the program begins execution at location 010H.
- Table location
  Any location in the ROM space can be used as look-up tables. 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 higher-order byte to TBLH (08H). Only the destination of the lower-order byte in the table is well-defined, the higher-order byte of the table word are transferred to the TBLH. The table higher-order byte register (TBLH) is read only. 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 TBLP. The TBLH is read only and cannot be restored. If the main routine and the ISR (interrupt service routine) both employ the table read instruction, the contents of the TBLH in the main routine are likely to be changed by the table read instruction used in the ISR. Errors can occur. In other words using the table read instruction in the main routine and the ISR simultaneously should be avoided. However, if the table read instruction has to be applied in both the main routine and the ISR, the interrupt is supposed to be disabled prior to the table read instruction. It will not be enabled until the TBLH has been backed up. All table related instructions need two cycles to complete the operation. These areas may function as normal program memory depending upon the requirements.

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 four levels 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

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: Bits of table location @7~@0: Bits of table pointer

P10~P8: Bits of current program counter

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.

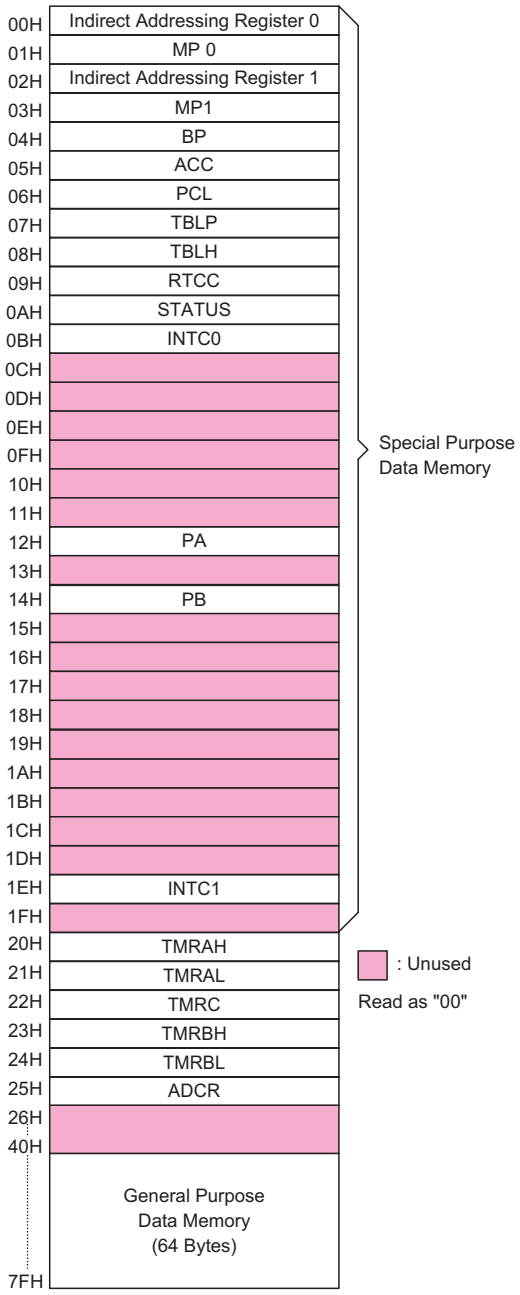
If the stack is full and a non-masked interrupt takes place, the interrupt request flag will be recorded but the acknowledgment will be inhibited. When the stack pointer is decremented (by RET or RETI), the interrupt will be serviced. This feature prevents stack overflow allowing the programmer to use the structure more easily. In a similar case, if the stack is full and a "CALL" is subsequently executed, stack overflow occurs and the first entry will be lost (only the most recent four return addresses are stored).

Data memory – RAM

The data memory is designed with 83×8 bits. The data memory is divided into two functional groups: special function registers and general purpose data memory (64×8). Most are read/write, but some are read only.

The special function registers include the indirect addressing register 0 (00H), the memory pointer register 0 (MP0; 01H), the indirect addressing register 1 (02H), the memory pointer register 1 (MP1;03H), the bank pointer (BP;04H), the accumulator (ACC;05H), the program counter lower-order byte register (PCL;06H), the table pointer (TBLP;07H), the table higher-order byte register (TBLH;08H), the real time clock control register (RTCC;09H), the status register (STATUS;0AH), the interrupt control register 0 (INTC0;0BH), the I/O registers (PA;12H, PB;14H), the interrupt control register 1 (INTC1;1EH), the Timer/Event counter A higher order byte register (TMRAH; 20H), the Timer/Event Counter A lower order byte register (TMRAL; 21H), the timer/event counter control register (TMRC; 22H), the Timer/Event Counter B higher order byte register (TMRBH; 23H), the Timer/Event Counter B lower-order byte register (TMRBL; 24H), and the RC oscillator type A/D converter control register (ADCR; 25H). The remaining space before the 40H are reserved for future expanded usage and reading these location will return the result 00H. The general purpose data memory, addressed from 40H to 7FH, is used for data and control information under instruction command.

All data memory areas can handle arithmetic, logic, increment, decrement and rotate operations. 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 instruction, respectively. They are also indirectly accessible through memory pointer registers (MP0;01H, MP1;03H).



RAM mapping (bank 0)

Indirect addressing register

Location 00H and 02H are indirect addressing registers that are not physically implemented. Any read/write operation of [00H] and [02H] access data memory pointed to by MP0 (01H) and MP1 (03H) respectively. Reading location 00H or 02H indirectly will return the result 00H. Writing indirectly results in no operation.

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

MP0 only can be applied to data memory, while MP1 can be applied to data memory and LCD display memory.

Accumulator

The accumulator is closely related to ALU operations. It is also mapped to location 05H of the data memory and is capable of carrying 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, DAA)
- 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 can change 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 most other registers. 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, system 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 automatically pushed onto the stack. If the contents of status are important and if the subroutine can corrupt the status register, precautions must be taken to save it properly.

Interrupts

The HT47C20L provides an external interrupt, an internal timer/event counter interrupt, an internal time base interrupt, and an internal real time clock interrupt. The interrupt control register 0 (INTC0;0BH) and interrupt control register 1 (INTC1;1EH) both contain the interrupt control bits to set the enable/disable and 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 a certain interrupt needs servicing within the service routine, the programmer may set the EMI bit and the corresponding bit of INTC0 or INTC1 allow interrupt nesting. If 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 must be prevented from becoming full.

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. C is also 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 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 0; 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	Unused bit, read as ″0″
—	7	Unused bit, read as ″0″

STATUS register

All these kinds of interrupt have a wake-up capability. As an interrupt is serviced, a control transfer occurs by pushing the program counter onto the stack, followed by a branch to a subroutine at specified locations in the program memory. Only the program counter is pushed onto the stack. If the contents of the register and status register (STATUS) is altered by the interrupt service program which corrupts the desired control sequence, the contents must be saved first.

External interrupt is triggered by a high to low transition of $\overline{INT}$ and the related interrupt request flag (EIF; bit 4 of INTC0) 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 4 of INTC1), caused by a timer A or timer B overflow. When the interrupt is enabled, and the stack is not full and the TF bit is set, a subroutine call to location 10H will occur. The related interrupt request flag (TF) will be reset and the EMI bit cleared to disable further interrupts.

The time base interrupt is initialized by setting the time base interrupt request flag (TBF; bit 5 of INTC0), caused by a regular time base signal. When the interrupt is enabled, and the stack is not full and the TBF bit is set, a subroutine call to location 08H will occur. The related interrupt request flag (TBF) will be reset and the EMI bit cleared to disable further interrupts.

The real time clock interrupt is initialized by setting the real time clock interrupt request flag (RTF; bit 6 of INTC0), caused by a regular real time clock signal. When the interrupt is enabled, and the stack is not full and the RTF bit is set, a subroutine call to location 0CH will occur. The related interrupt request flag (RTF) 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 or the EMI bit and the related interrupt control bit are set to 1 (if the stack is not full). To return from the interrupt subroutine, RET or RETI instruction may be invoked. RETI will set the EMI bit to enable an interrupt service, but RET does 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 following table shows the priority that is applied. These can be masked by resetting the EMI bit.

No.	Interrupt Source	Priority	Vector
a	External interrupt	1	04H
b	Time base interrupt	2	08H
c	Real time clock interrupt	3	0CH
d	Timer/event counter interrupt	4	10H

Register	Bit No.	Label	Function
INTC0 (0BH)	0	EMI	Control the master (global) interrupt (1=enabled; 0=disabled)
	1	EEI	Control the external interrupt (1=enabled; 0=disabled)
	2	ETBI	Control the time base interrupt (1=enabled; 0=disabled)
	3	ERTI	Control the real time clock interrupt (1=enabled; 0=disabled)
	4	EIF	External interrupt request flag (1=active; 0=inactive)
	5	TBF	Time base interrupt request flag (1=active; 0=inactive)
	6	RTF	Real time clock interrupt request flag (1=active; 0=inactive)
	7	—	Unused bit, read as ″0″
INTC1 (1EH)	0	ETI	Control the timer/event counter interrupt (1=enabled; 0=disabled)
	1	—	Unused bit, read as ″0″
	2	—	Unused bit, read as ″0″
	3	—	Unused bit, read as ″0″
	4	TF	Internal timer/event counter interrupt request flag (1=active; 0=inactive)
	5	—	Unused bit, read as ″0″
	6	—	Unused bit, read as ″0″
	7	—	Unused bit, read as ″0″

INTC Register

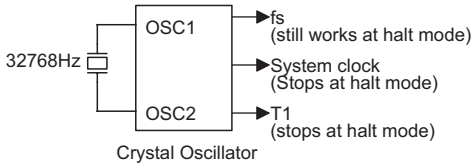
The external interrupt request flag (EIF), real time clock interrupt request flag (RTF), time base interrupt request flag (TBF), enable external interrupt bit (EEI), enable real time clock interrupt bit (ERTI), enable time base interrupt bit (ETBI), and enable master interrupt bit (EMI) constitute an interrupt control register 0 (INTC0) which is located at 0BH in the data memory. The timer/event counter interrupt request flag (TF), enable timer/event counter interrupt bit (ETI) on the other hand, constitute an interrupt control register 1 (INTC1) which is located at 1EH in the data memory. EMI, EEI, ETI, ETBI, and ERTI are used to control the enabling/disabling of interrupts. These bits prevent the requested interrupt being serviced. Once the interrupt request flags (RTF, TBF, TF, EIF) are set, they remain in the INTC1 or INTC0 respectively until the interrupts are serviced or cleared by a software instruction.

It is recommended that a program does not use the "CALL subroutine" within the interrupt subroutine. Interrupts often occur in an unpredictable manner or need to be serviced immediately in some applications. If only one stack is left, and enabling the interrupt is not well controlled, the original control sequence will be damaged once the "CALL subroutine" operates in the interrupt subroutine.

Oscillator configuration

The HT47C20L provides one 32768Hz crystal oscillator for real time clock and system clock. The 32768Hz crystal oscillator still work at halt mode. The halt mode stop the system clock and T1 and ignores an external signal to conserve power. The real time clock comes from 32768Hz crystal and still works at halt mode.

A 32768Hz crystal across OSC1 and OSC2 is needed to provide the feedback and phase shift needed for the oscillator, no other external components are needed.



32768Hz crystal

Watchdog Timer – WDT

The clock source of the WDT (f_S) is implemented by a 32768Hz crystal oscillator. The 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 mask option. If the Watchdog Timer is disabled, all the executions related to the WDT result in no operation.

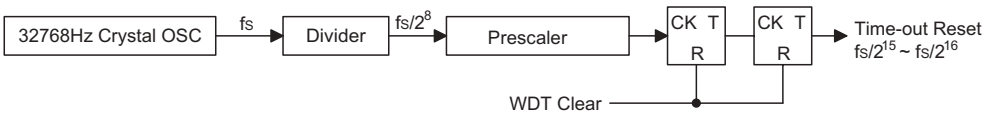
The "HALT" instruction is executed, WDT still counts and can wake-up from halt mode due to the WDT time-out.

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 0. To clear the contents of WDT, three methods are adopted, external reset (a low level to $\overline{RES}$), software instruction, or a HALT instruction. The software instructions are of two sets which include CLR WDT and the other set – CLR WDT1 and CLR WDT2. Of these two types of instruction, only one can be active depending on the mask option – "CLR WDT times selection option". If the "CLR WDT" is selected (i.e., CLR WDT times equal one), any execution of the CLR WDT instruction will clear the WDT. In case "CLR WDT1" and "CLR WDT2" are chosen (i.e. CLR WDT times equal two), these two instructions must be executed to clear the WDT; otherwise, the WDT may reset the chip because of the time-out.

The WDT time-out period ranges from $f_{S/2}{}^{15}$~$f_{S/2}{}^{16}$. The "CLR WDT" or "CLR WDT1" and "CLRWDT2" instruction only clear the last two-stage of the WDT.

Multi-function timer

The HT47C20L provides a multi-function timer for the WDT, time base and real time clock but with different time-out periods. The multi-function timer consists of a 7-stage divider and an 8-bit prescaler, with the clock source coming from the 32768Hz. The multi-function timer also provides a fixed frequency signal (f_S/8) for the LCD driver circuits, and a selectable frequency signal (ranges from $f_S/2^2$ to $f_S/2^9$) for buzzer output by mask option.



Watchdog Timer

Time base

The time base offers a periodic time-out period to generate a regular internal interrupt. Its time-out period ranges from $f_S/2^{12}$ to $f_S/2^{15}$ selected by mask option. If time base time-out occurs, the related interrupt request flag (TBF; bit 5 of INTC0) is set. But if the interrupt is enabled, and the stack is not full, a subroutine call to location 08H occurs.

When the HALT instruction is executed, the time base still works and can wake up from halt mode. If the TBF is set ″1″ before entering the halt mode, the wake up function will be disabled.

Real time clock – RTC

The real time clock is operated in the same manner as the time base that is used to supply a regular internal interrupt. Its time-out period ranges from $f_S/2^8$ to $f_S/2^{15}$ by software programming. Writing data to RT2, RT1 and RT0 (bits 2, 1, 0 of RTCC;09H) yields various time-out periods. If a real time clock time-out occurs, the related interrupt request flag (RTF; bit 6 of INTC0) is set. But if the interrupt is enabled, and the stack is not full, a subroutine call to location 0CH occurs. The real time clock time-out signal can also be applied as a clock source of Timer/Event Counter A, so as to get a longer time-out period.

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}

Power down operation – HALT

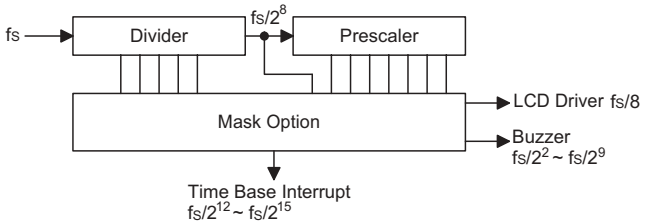
The halt mode is initialized by the HALT instruction and results in the following.

- The 32768Hz crystal oscillator will still work but the system clock and T1 will turn off.
- The contents of the on-chip RAM and registers remain unchanged.
- The WDT will be cleared and recount again.
- All I/O ports maintain their original status.
- The PD flag is set and the TO flag is cleared.
- LCD driver is still running (by mask option).
- The time base and real time clock will still work.

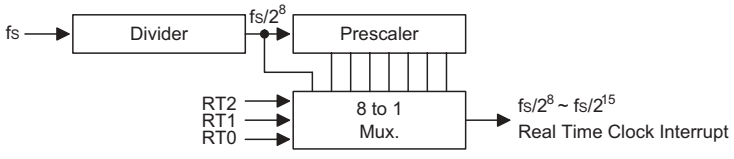
The system can leave the halt mode by means of an external reset, an interrupt, an external falling edge signal on port A or a WDT overflow. An external reset causes a device initialization and the WDT overflow performs a ″warm reset″. Examining the TO and PD flags, the reason for chip reset can be determined. The PD flag is cleared when system power-up or executing 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 maintain 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 mask option. Awakening from an I/O port stimulus, the program will resume execution of the next instruction. If awakening 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, 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.



Time base



Real time clock

If the wake-up results from an interrupt acknowledgment, the actual interrupt subroutine execution will be delayed by more than one cycle. However, if the wake-up results in the next instruction execution, the execution will be performed immediately.

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

Reset

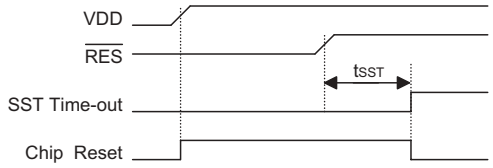
There are three ways in which a reset may occur.

- $\overline{RES}$ reset during normal operation
- $\overline{RES}$ reset during halt mode
- WDT time-out reset during normal operation
- The LVR is enable and the VDD is lower then V_{LVR}

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 in their original state. Some registers remain unchanged during 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 or LVR reset during normal operation
0	1	$\overline{RES}$ reset or LVR reset wake-up from HALT mode
1	u	WDT time-out during normal operation
1	1	WDT wake-up from HALT mode

Note: "u" means "unchanged"

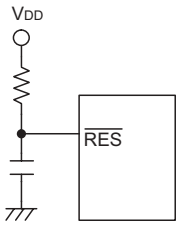


Reset circuit

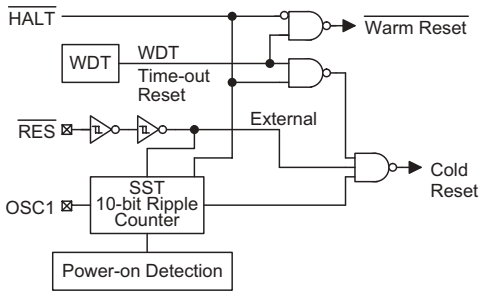
To guarantee that the crystal oscillator has started and stabilized, the SST (system start-up timer) provides an extra delay of 8192 system clock pulses when the system powers up.

The functional unit chip reset status are shown below.

PC	000H
Interrupt	Disabled
Prescaler, divider	Cleared
WDT, real time clock, time base	Clear. After master reset, begin counting
Timer/event counter	Off
Input/output ports	Input mode
SP	Points to the top of the stack



Reset timing chart



Reset configuration

The states of the registers are 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)
TMRAH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMRAL	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMRC	0000 1---	0000 1---	0000 1---	0000 1---	uuuu u---
TMRBH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMRBL	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
ADCR	1xxx --00	1xxx --00	1xxx --00	1xxx --00	uuuu --uu
Program Counter	000H	000H	000H	000H	000H*
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLP	xxxx xxxx	uuuu uuuu	uuu 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
INTC0	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	---0 ---0	---0 ---0	---0 ---0	---0 ---0	---u ---u
RTCC	--xx 0111	--xx 0111	--xx 0111	--xx 0111	--uu uuuu
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu

Note: "*" refers to "warm reset"

"u" means "unchanged"

"x" means "unknown"

Timer/event counter

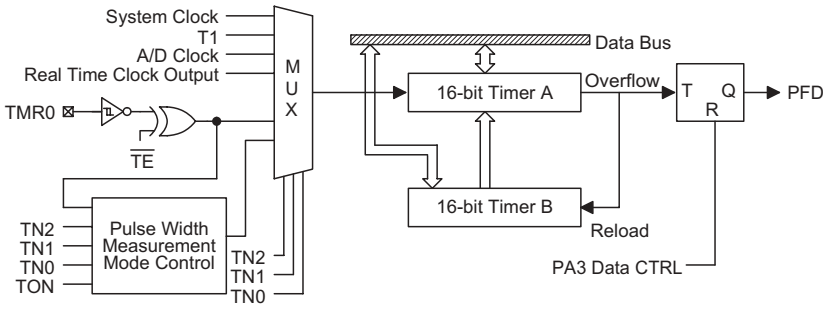
One 16-bit timer/event counter with PFD output or two channels of RC type A/D converter is implemented in the HT47C20L. The ADC/$\overline{TM}$ bit (bit 1 of ADCR register) decides whether timer A and timer B are composed of one 16-bit timer/event counter or timer A and timer B are composed of two channels RC type A/D converter.

The TMRAL, TMRAH, TMRBL, TMRBH composed of one 16-bit timer/event counter, when ADC/$\overline{TM}$ bit is "0". The TMRBL and TMRBH are timer/event counter preload registers for lower-order byte and higher-order byte respectively.

The timer/event counter clock source may come from system clock or T1 (system clock/4) or real time clock time-out signal or external source.

Using external clock input allows the user to count external events, count external RC type A/D clock, measure time intervals or pulse widths, or generate an accurate time base. While using the internal clock allows the user to generate an accurate time base.

There are six registers related to the timer/event counter operating mode. TMRAH ([20H]), TMRAL ([21H]), TMRC ([22H]), TMRBH ([23H]), TMRBL ([24H]) and ADCR ([25H]). Writing to TMRBL only writes the data into a low



Timer/event counter

Label (TMRC)	Bits	Function
—	0~2	Unused bit, 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 timer counting (0= disabled; 1= enabled)
TN0 TN1 TN2	5 6 7	To define the operating mode (TN2, TN1, TN0) 000= Timer mode (system clock) 001= Timer mode (system clock/4) 010= Timer mode (real time clock output) 011= A/D clock mode (RC oscillation decided by ADCR register) 100= Event counter mode (external clock) 101= Pulse width measurement mode (system clock/4) 110= Unused 111= Unused

TMRC register

byte buffer, and writing to TMRBH will write the data and the contents of the low byte buffer into the time/event counter preload register (16-bit) simultaneously. The timer/event counter preload register is changed by writing to TMRBH operations and writing to TMRBL will keep the timer/event counter preload register unchanged.

Reading TMRAH will also latch the TMRAL into the low byte buffer to avoid the false timing problem. Reading TMRAL returns the contents of the low byte buffer. In other words, the low byte of the timer/event counter can not be read directly. It must read the TMRAH first to make the low byte contents of timer/event counter be latched into the buffer.

If the timer/event counter is on, the TMRAH, TMRAL, TMRBH and TMRBL cannot be read or written to. To avoid conflicting between timer A and timer B, the TMRAH, TMRAL, TMRBH and TMRBL registers should be accessed with ″MOV″ instruction under timer off condition.

The TMRC is the timer/event counter control register, which defines the timer/event counter options.

The timer/event counter control register define the operating mode, counting enable or disable and active edge.

Writing to timer B location puts the starting value in the timer/event counter preload register, while reading timer A yields the contents of the timer/event counter. Timer B is timer/event counter preload register.

The TN0, TN1 and TN2 bits define the operation mode. The event count mode is used to count external events, which means that the clock source comes from an external (TMR) pin. The A/D clock mode is used to count external A/D clock, the RC oscillation mode is decided by ADCR register. The timer mode functions as a normal timer with the clock source coming from the internal selected clock source. Finally, the pulse width measurement mode can be used to count the high or low level duration of the external signal (TMR). The counting is based on the T1 (system clock/4).

In the event count, A/D clock or internal timer mode, once the timer/event counter starts counting, it will count from the current contents in the timer/event counter (TMRAH and TMRAL) to FFFFH. Once overflow occurs, the counter is reloaded from the timer/event counter preload register (TMRBH and TMRBL) and generates the corresponding interrupt request flag (TF; bit 4 of INTC1) at the same time.

In the pulse width measurement mode with the TON and TE bits equal to 1, once the TMR has received a transient from low to high (or high to low if the TE bit is 0) it will start counting until the TMR returns to the original level and resets the TON. The measured result will remain in the timer/event counter even if the activated transient occurs again. In other words, only one cycle measurement can be done. Until setting the TON, the cycle measurement will function again as long as it receives further transient pulse. Note that in this operation mode, the timer/event counter starts counting not according to the logic level but according to the transient edges. In the case of counter overflow, the counter is reloaded from the timer/event counter preload register and issues interrupt request just like the other three modes.

To enable the counting operation, the timer On bit (TON; bit 4 of TMRC) should be set to 1. In the pulse width measurement mode, the TON will automatically be cleared after the measurement cycle is completed. But in the other three modes, the TON can only be reset by instructions. The overflow of the timer/event counter is one of the wake-up sources and can also be applied as a PFD (programmable frequency divider) output at PA3 by mask option. No matter what the operation mode is, writing a 0 to ETI can disable the corresponding interrupt service. When the PFD function is selected, executing ″CLR PA.3″ instruction to enable the PFD output and executing ″SET PA.3″ instruction to disable the PFD output and PA.3 output low level.

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 turns On, data written to the timer/event counter preload register is kept only in the timer/event counter preload register. The timer/event counter will still operate until overflow occurs.

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

It is strongly recommended to load first the desired value into TMRBL, TMRBH, TMRAL, and TMRAH registers then turn on the related timer/event counter for proper operation. Because the initial value of TMRBL, TMRBH, TMRAL and TMRAH are unknown.

If the timer/event counter is on, the TMRAH, TMRAL, TMRBH and TMRBL cannot be read or written to. Only when the timer/event counter is off and when the instruction "MOV" is used could those four registers be read or written to.

Example for Timer/event counter mode (disable interrupt):

```
clr tmrc
clr adcr.1                      ; set timer mode
clr intc1.4                     ; clear timer/event counter interrupt request flag
mov a, low (65536-1000)         ; give timer initial value
mov tmrbl, a                    ; count 1000 time and then overflow
mov a, high (65536-1000)
mov tmrbh, a

mov a, 00110000b                ; timer clock source=T1 and timer on
 tmrc, a

p10:
clr wdt
snz intcl.4                     ; polling timer/event counter interrupt request flag
jmp p10

clr intcl.4                     ; clear timer/event counter interrupt request flag
                                ; program continue
```

A/D converter

Two channels of RC type A/D converter are implemented in the HT47C20L. The A/D converter contains two 16-bit programmable count-up counter and the timer A clock source may come from the system clock, T1 (system clock/4) or real time clock output. The timer B clock source may come from the external RC oscillator. The TMRAL, TMRAH, TMRBL, TMRBH are composed of the A/D converter when ADC/$\overline{TM}$ bit (bit 1 of ADCR register) is ″1″.

The A/D converter timer B clock source may come from channel 0 (IN0 external clock input mode, RS0~CS0 oscillation, RT0~CS0 oscillation, CRT0~CS0 oscillation (CRT0 is a resistor), or RS0~CRT0 oscillation (CRT0 is a capacitor) or channel 1 (RS1~CS1 oscillation, RT1~CS1 oscillation or IN1 external clock input). The timer A clock source is from the system clock, T1 or real time clock prescaler clock output decided by TMRC register.

There are six registers related to A/D converter, i.e., TMRAH, TMRAL, TMRC, TMRBH, TMRBL and ADCR. The internal timer clock is input to TMRAH and TMRAL, the A/D clock is input to TMRBH and TMRBL. The OVB/$\overline{OVA}$ bit (bit 0 of the ADCR register) decides whether timer A overflows or timer B overflows, then the TF bit is set and timer interrupt occurs. When the A/D converter mode timer A or timer B overflows, the TON bit is reset and stop counting. Writing TMRAH/TMRBH puts the starting value in the timer A/timer B and reading TMRAH/TMRBH gets the contents of the timer A/timer B. Writing TMRAL/TMRBL only writes the data into a low byte buffer, and writing TMRAH/TMRBH will write the data and the contents of the low byte buffer into the timer A/timer B (16-bit) simultaneously. The timer A/timer B is change by writing TMRAH/TMRBH operations and writing TMRAL/TMRBL will keep the timer A/timer B unchanged.

Reading TMRAH/TMRBH will also latch the TMRAL/TMRBL into the low byte buffer to avoid the false timing problem. Reading TMRAL/TMRBL returns the contents of the low byte buffer. In other word, the low byte of timer A/timer B can not be read directly. It must read the TMRAH/TMRBH first to make the low byte contents of timer A/timer B be latched into the buffer.

If the A/D converter timer A and timer B are counting, the TMRAH, TMRAL, TMRBH and TMRBL cannot be read or written to. To avoid conflicting between timer A and timer B, the TMRAH, TMRAL, TMRBH and TMRBL registers should be accessed with ″MOV″ instruction under timer A and timer B off condition.

The bit4~bit7 of ADCR decides which resistor and capacitor compose an oscillation circuit and input to TMRBH and TMRBL.

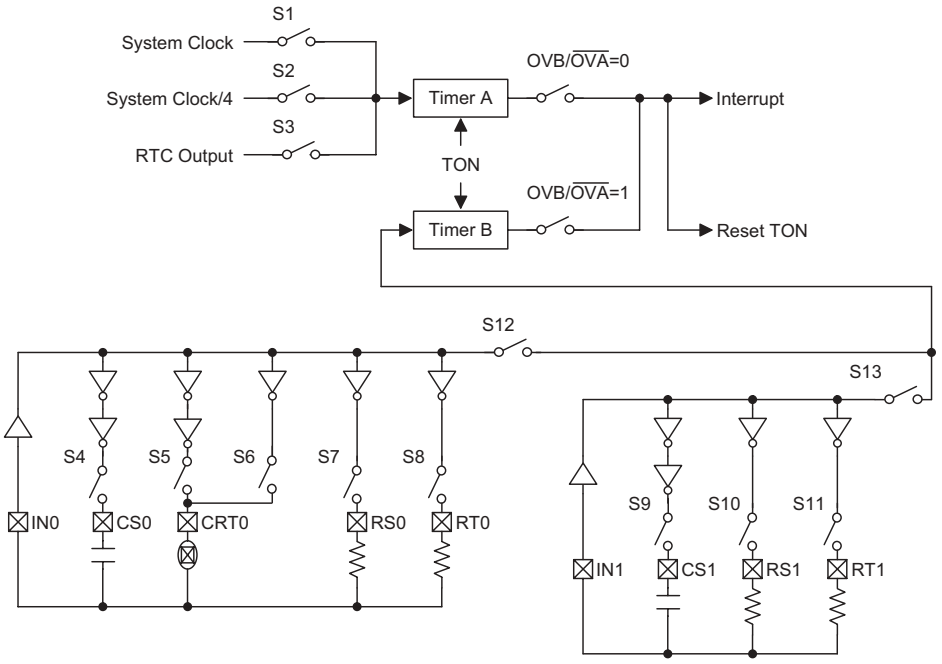
The TN0, TN1 and TN2 bits of TMRC define the clock source of timer A. It is suggested that the clock source of timer A use the system clock, instruction clock or real time clock prescaler clock.

The TON bit (bit 4 of TMRC) is set ″1″, the timer A and timer B will start counting until timer A or timer B overflows, the timer/event counter generates the interrupt request flag (TF ; bit 4 of INTC1) and the timer A and timer B stop counting and reset the TON bit to ″0″ at the same time.

If the TON bit is ″1″, the TMRAH, TMRAL, TMRBH and TMRBL cannot be read or written to. Only when the timer/event counter is off and when the instruction ″MOV″ is used could those four registers be read or written to.

Label (ADCR)	Bits	Function
OVB/$\overline{OVA}$	0	In the RC type A/D converter mode, this bit is used to define the timer/event counter interrupt which comes from timer A overflow or timer B overflow. (0= timer A overflow; 1= timer B overflow) In the timer/event counter mode, this bit is void.
ADC/$\overline{TM}$	1	To define 16-bit timer/event counter or RC type A/D converter is enable. (0= timer/event counter enable; 1= A/D converter is enable)
—	2~3	Unused bits, read as ″0″
M0 M1 M2 M3	4 5 6 7	To define the A/D converter operating mode (M3, M2, M1, M0) 0000= IN0 external clock input mode 0001= RS0~CS0 oscillation (reference resistor and reference capacitor) 0010= RT0~CS0 oscillation (resistor sensor and reference capacitor) 0011= CRT0~CS0 oscillation (resistor sensor and reference capacitor) 0100= RS0~CRT0 oscillation (reference resistor and sensor capacitor) 0101= RS1~CS1 oscillation (reference resistor and reference capacitor) 0110= RT1~CS1 oscillation (resistor sensor and reference capacitor) 0111= IN1 external clock input mode 1XXX= Undefined mode

ADCR register



TN2	TN1	TN0	S1	S2	S3
0	0	0	1	0	0
0	0	1	0	1	0
0	1	0	0	0	1
Other			0	0	0

Note: 0=off, 1=on

M3	M2	M1	M0	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13
0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	1	1	0	0	1	0	0	0	0	1	0
0	0	1	0	1	0	0	0	1	0	0	0	1	0
0	0	1	1	1	0	1	0	0	0	0	0	1	0
0	1	0	0	0	1	0	1	0	0	0	0	1	0
0	1	0	1	0	0	0	0	0	1	1	0	0	1
0	1	1	0	0	0	0	0	0	1	0	1	0	1
0	1	1	1	0	0	0	0	0	0	0	0	0	1
1				0	0	0	0	0	0	0	0	0	0

Note: 0=off, 1=on

RC type A/D converter

Example for RC type AD converter mode (Timer A overflow):

```
clr tmrc
clr adcr.1                      ; set timer mode
clr intc1.4                     ; clear timer/event counter interrupt request flag
mov a, low (65536-1000)         ; give timer A initial value
mov tmrbl, a                    ; count 1000 time and then overflow
mov a, high (65536-1000)
mov tmrbh, a

mov a, 00010010b                ; RS0~CS0; set RC type ADC mode; set Timer A overflow
mov  adcr, a
mov a, 00h                      ; give timer B initial value
mov tmrbl, a
mov a, 00h
mov tmrbh, a

mov a, 00110000b                ; timer A clock source=T1 and timer on
mov tmrc, a

p10:
clr wdt
snz intcl.4                     ; polling timer/event counter interrupt request flag
jmp p10

clr intcl.4                     ; clear timer/event counter interrupt request flag
                                ; program continue
```

Example for RC type AD converter mode (Timer B overflow):

```
clr tmrc
clr adcr.1                      ; set timer mode
clr intc1.4                     ; clear timer/event counter interrupt request flag
 a, 00h                         ; give timer A initial value
mov tmrbl, a
 a, 00h
mov tmrbh, a

mov a, 00010011b                ; RS0~CS0; set RC type ADC mode; set Timer B overflow
mov adcr,a

mov a, low (65536-1000)         ; give timer B initial value
mov tmrbl, a                    ; count 1000 time and then overflow
mov a, high (65536-1000)
mov tmrbh, a

mov a, 00110000b                ; timer A clock source=T1 and timer on
mov tmrc, a

p10:
clr wdt
snz intcl.4                     ; polling timer/event counter interrupt request flag
jmp p10

clr intcl.4                     ; clear timer/event counter interrupt request flag
                                ; program continue
```

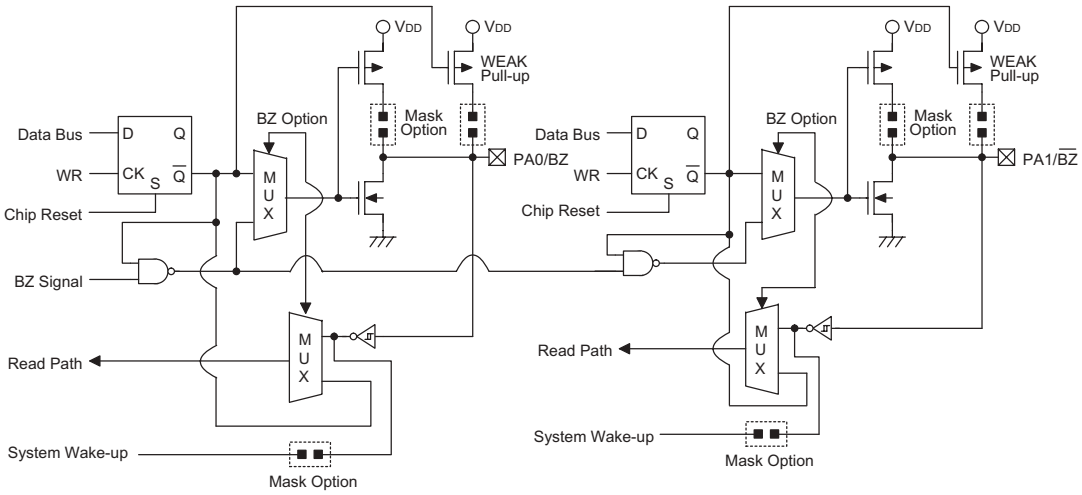
Input/output ports

There are 8-bit bidirectional input/output port and 4-bit input port in the HT47C20L, labeled PA and PB which are mapped to the data memory of [12H] and [14H] respectively. The high nibble of the PA is NMOS output and input with pull-high resistors. The low nibble of the PA can be used for input/output or output operation by selecting NMOS or CMOS output by mask option. Each bit on the PA can be configured as a wake-up input and the low nibble of the PA with or without pull-high resistors by mask option. PB can only be used for input operation, and each bit on the port is with pull high resistor. Both are for the input operation, these ports are non-latched, that is, the inputs should be ready at the T2 rising edge of the instruction "MOV A, [m]" (m=12H or 14H). For PA output operation, all data are latched and remain unchanged until the output latch is rewritten.

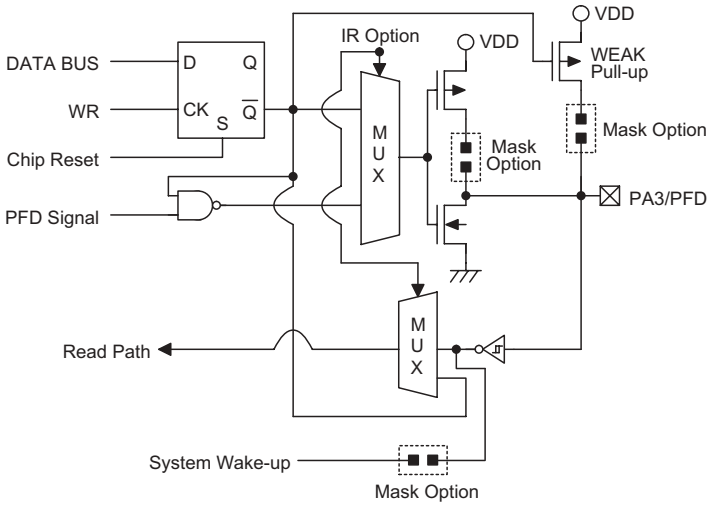
When the structures of PA are open drain NMOS type, it should be noted that, before reading data from the pads a "1" should be written to the related bits to disable the NMOS device. That is done first before executing the instruction "MOV A, 0FFH" and "MOV [12H], A" to disable the related NMOS device, and then "MOV A, [12H]" to get a stable data.

After chip reset, these input lines remain at a high level or are left floating (by mask option).

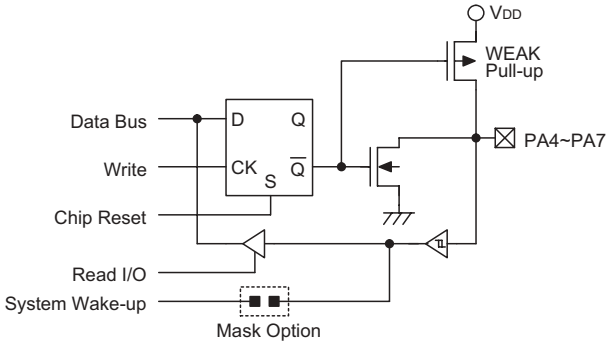
Some instructions first input data and then follow the output operations. For example, "SET [m].i", "CLR [m].i", "CPL [m]", "CPLA [m]" read the entire port states into the CPU, execute the defined operations (bit-operation), and then write the results back to the latches or to the accumulator. Each bit of the PA output latches can not use these instruction, which may change the input lines to output lines (when the input lines are at low level).



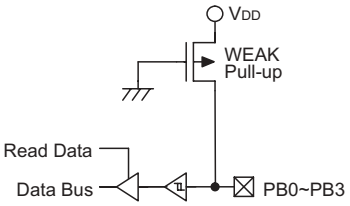
PA0/BZ, PA1/$\overline{BZ}$ input/output lines



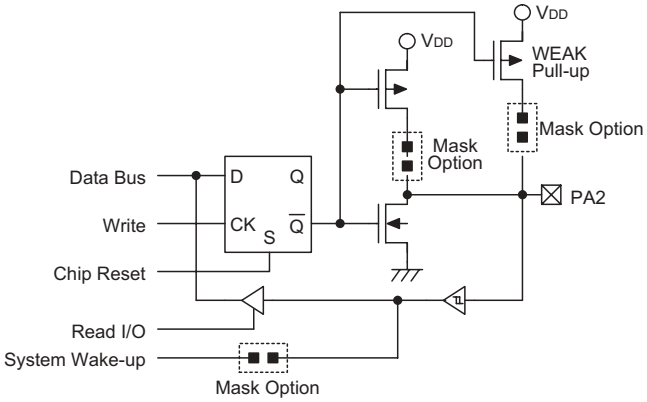
PA3/PFD input/output line



PA4~PA7 input/output line



PB input lines



PA2 input/output lines

LCD display memory

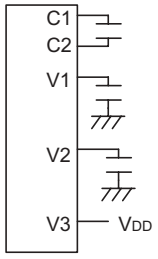
The HT47C20L provides an area of embedded data memory for LCD display. The LCD display memory is designed into 20×4 bits. If the LCD selected 19×4 segments output, the 53H of the LCD display memory can not be accessed. This area is located from 40H to 53H of the RAM at Bank 1. Bank pointer (BP; located at 04H of the data memory) is the switch between the general data memory and the LCD display memory. When the BP is set "1" any data written into 40H~53H will effect the LCD display (indirect addressing mode using MP1). When the BP is cleared "0", any data written into 40H~53H has to access the general purpose data memory. The LCD display memory can be read and written only by indirect addressing mode using MP1. When data is written into the display data area, it is automatically read by the LCD driver which then generates the corresponding LCD driving signals. To turn the display On or Off, a "1" or a "0" is written to the corresponding bit of the display memory, respectively.

The figure illustrates the mapping between the display memory and LCD pattern for the HT47C20L.

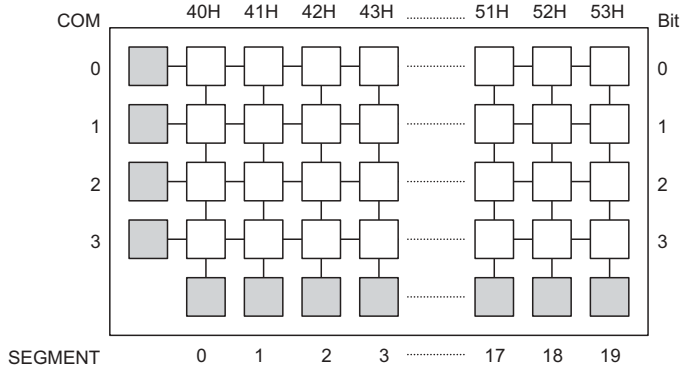
LCD driver output

The output number of the HT47C20L LCD driver can be 20×2 or 20×3 or 19×4 by mask option (i.e.1/ 2duty, 1/3 duty or 1/4 duty).

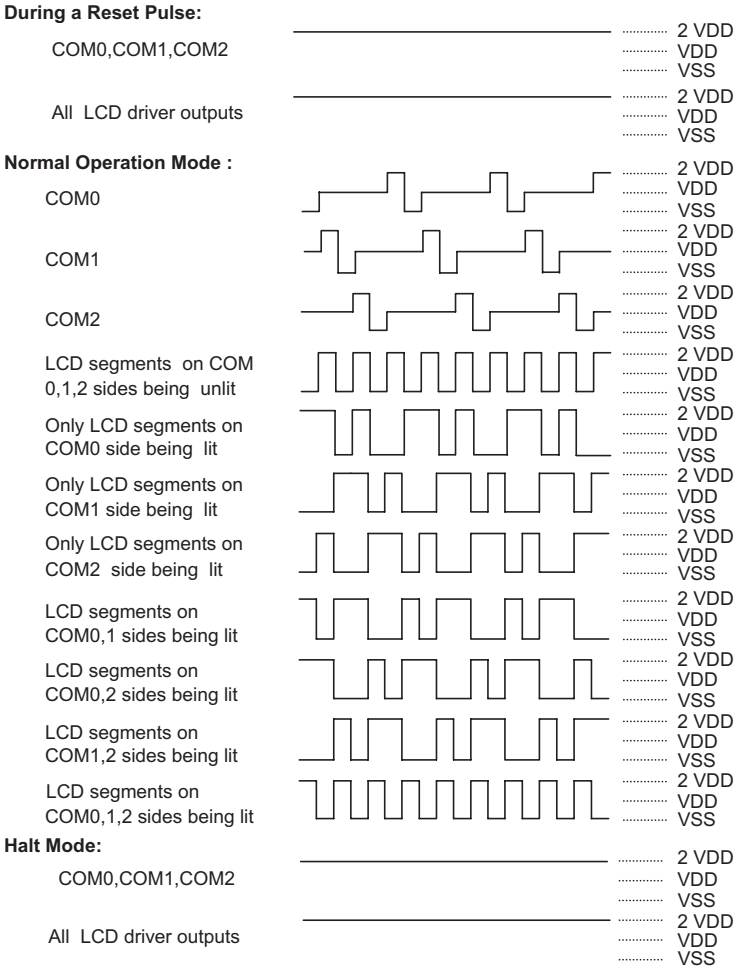
The bias type LCD driver is "C" type. If the duty or 1/3 duty type is selected, the 1/2 bias type is selected. If the 1/4 duty type is selected, the 1/3 bias type is selected. A capacitor has to be connected between C1 and C2. The two kinds of the configurations of V1, V2 and V3 pins are as follows:



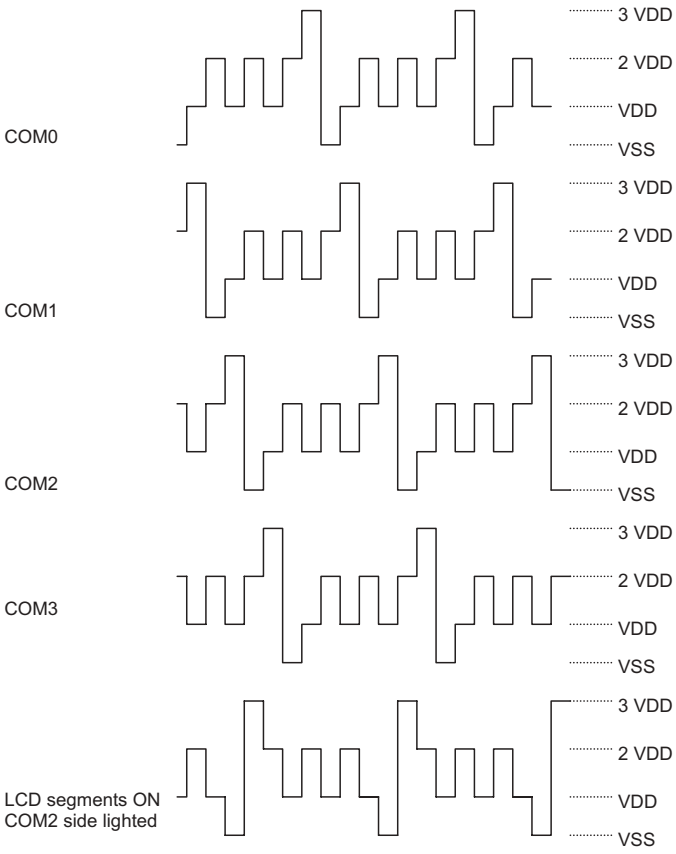
V1, V2, V3 application diagram



Display memory (bank 1)



LCD driver output (1/3 duty, 1/2bias)



LCD driver output (1/4 duty, 1/3 bias)

Voltage low detector

The HT47C20L provides a voltage low detector for battery system application. If the battery voltage is lower than the specified value, the battery low flag (BLF; bit 5 of RTCC) is set. The specified value is 1.2V±0.1V. The voltage low detector circuit can be turn On or Off by writing a ″1″ or a ″0″ to BON (bit 3 of RTCC register). A delay time of 1ms is required to monitor the BLF after setting the BON bit. The BLF is invalid when the BON is cleared as ″0″. The voltage low detector can be disabled by mask option.

Buzzer

HT47C20L provides a pair of buzzer output BZ and $\overline{BZ}$, which share pins with PA0 and PA1 respectively, determined by mask option. Its output frequency can also be selected by mask option.

When the buzzer function is selected, setting PA.0 and PA.1 ″0″ simultaneously will enable the buzzer output and setting PA.0 ″1″ will disable the buzzer output and setting PA.0 ″0″ and PA.1 ″1″ will only enable the BZ output and disable the $\overline{BZ}$ output.

PA1	PA0	Function
0 (CLR PA.1)	0 (CLR PA.0)	PA0= BZ PA1= $\overline{BZ}$
1 (SET PA.1)	0 (CLR PA.0)	PA0= BZ PA1= 0
X	1 (SET PA.0)	PA0= 0 PA1= 0

Buzzer enable

Programmable frequency divider – PFD

The PFD output shares pin with PA3 as determined by mask option.

When the PFD option is selected, setting PA3 ″0″ will enable the PFD output and setting PA3 ″1″ will disable the PFD output and PA3 output at low level.

PA3	Function
0 (CLR PA.3)	PA3= PFD Output
1 (SET PA.3)	PA3= 0

PFD output frequency=

$$\frac{1}{2} \times \frac{1}{\text{timer overflow period}}$$

Low voltage reset – LVR

The low voltage reset circuit is used to monitor the power supply of the device. If the power supply voltage of the device is lower than 1.1V±0.1V, the device will automatically reset internally. It is enabled or disabled by mask option.

The LVR includes the following specification:

- The low voltage (lower than 1.1V±0.1V) must be maintained for over 1ms. If the low voltage state does not exceed 1ms, the LVR will ignore it and does not perform the reset function.
- The LVR uses the ″OR″ function with the external $\overline{RES}$ signal to perform chip reset.
- During HALT mode, if the LVR occurs, the device will wake-up and the PD flag will be set as ″1″, the same as the $\overline{RES}$ reset.

Register	Bit No.	Label	Read/Write	Reset	Function
RTCC (09H)	0 1 2	RT0 RT1 RT2	R/W	1 1 1	8 to 1 multiplexer control inputs to select the real time clock prescaler output
	3	BON	R/W	0	Voltage low detector enable/disable control bit ″0″ indicates voltage detector is disabled ″1″ indicates voltage detector is enabled
	4	—	—	—	Unused bit, read as ″unknown″
	5	BLF	R	X	Battery low flag ″0″ indicates that the voltage is not low ″1″ indicates that the voltage is low
	6, 7	—	—	—	Unused bit, read as ″0″

RTCC Register

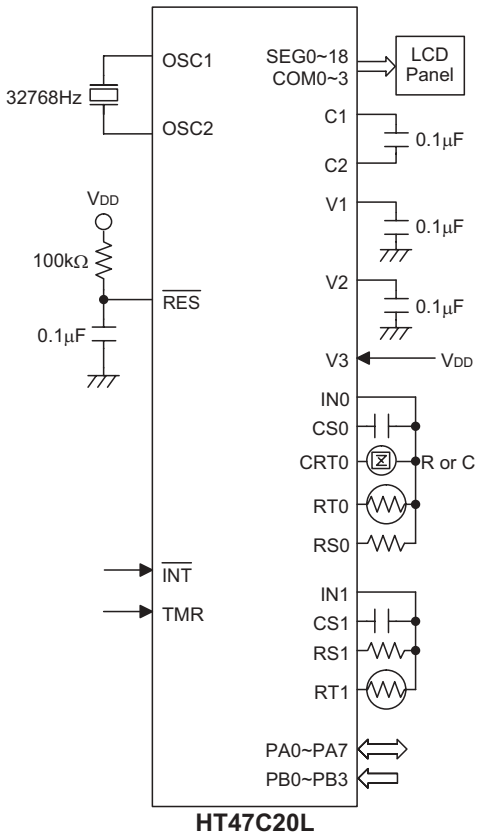
Note: ″X″ means ″invalid″

Mask option

The following shows many kinds of mask options in the HT47C20L. All these options should be defined in order to ensure proper system functioning.

No.	Mask Option
1	WDT enable/disable selection. WDT can be enabled or disabled by mask option.
2	CLR WDT times selection. This option defines how to clear the WDT by instruction. One time means that the ″CLR WDT″ can clear the WDT. ″Two times″ means that only if both of the ″CLR WDT1″ and ″CLR WDT2″ have been executed, then WDT can be cleared.
3	Time base time-out period selection. The time base time-out period ranges from $f_S/2^{12}$ to $f_S/2^{15}$. ″f_S″ stands for the 32768Hz frequency.
4	Buzzer output frequency selection. There are eight types of frequency signals for the buzzer output: $f_S/2^2$~$f_S/2^9$. ″f_S″ stands for the 32768Hz.
5	Wake-up selection. This option defines the wake-up function activity. External I/O pins (PA NMOS output only) all have the capability to wake-up the chip from a halt mode by a following edge.
6	Pull high selection. This option is to decide whether the pull high resistance is viable or not on the low nibble of the PA.
7	PA CMOS or NMOS selection. The structure of the low nibble of the PA can be selected as CMOS or NMOS. When CMOS is selected, the related pins can only be used for output operations. When NMOS is selected, the related pins can be used for input or output operations.
8	I/O pins share with other function selection. PA0/BZ, PA1/$\overline{BZ}$: PA0 and PA1 can be set as I/O pins or buzzer outputs. PA3/PFD: PA3 can be set as I/O pins or PFD output.
9	LCD common selection. There are three types of selection: 2 common (1/2 duty, 1/2bias) 3 common (1/3 duty, 1/2 bias) or 4 common (1/4 duty, 1/3 bias). If the 4 common is selected, the segment output pin ″SEG19/COM3″ will be set as a common output ″COM3″.
10	The low voltage reset and the low voltage detector enable or disable selection. There are three types of selection. The low voltage reset and the voltage detector are both enabled or both disabled or the low voltage reset is disabled but the voltage low detector is enabled.
11	LCD on or LCD off at the halt mode selection. The LCD can be enable or disable at the halt mode by mask option.

Application Circuits



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 data memory 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 and result in data memory	Z, C, AC, OV
DAA [m]	Decimal adjust ACC for addition with result in data memory	C
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
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

Mnemonic	Description	Flag Affected
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	None
RET A,x	Return from subroutine and load immediate data to ACC	None
RETI	Return from interrupt	None
Table Read		
TABRDC [m]	Read ROM code (current page) to data memory and TBLH	None
TABRDL [m]	Read ROM code (last page) to data memory and TBLH	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
CLR WDT1	Pre-clear Watchdog timer	TO*, PD*
CLR WDT2	Pre-clear Watchdog timer	TO*, PD*
SWAP [m]	Swap nibbles of data memory	None
SWAPA [m]	Swap nibbles of data memory with result in ACC	None
HALT	Enter power down mode	TO, PD

Note: x: 8 bits immediate data
m: 7 bits data memory address
A: accumulator
i= 0~7 number of bits
addr: 11 bits program memory address
√: Flag is affected
–: Flag is not affected
*: Flag may be affected by the execution status

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 0.

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 0.

Operation [m].i ← 0

Affected flag(s)

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

CLR WDT Clear watchdog timer

Description The WDT and WDT Prescaler are cleared. The power down bit (PD) and time-out bit (TO) are cleared.

Operation WDT last two bits ← 00H
PD and TO ← 0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	0	0	—	—	—	—

CLR WDT1 Preclear watchdog timer

Description The PD, TO flags and WDT are cleared, if the other preclear WDT instruction had been executed. Only execution of this instruction without the other preclear instruction sets the indicating flag which implies that this instruction was executed and the PD and TO flags remain unchanged.

Operation WDT last two bits ← 00H*
PD & TO ← 0*

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	0*	0*	—	—	—	—

CLR WDT2 Preclear watchdog timer

Description The PD, TO flags and WDT are cleared, if the other preclear WDT instruction had been executed. Only execution of this instruction without the other preclear instruction sets the indicating flag which implies that this instruction was executed and the PD and TO flags remain unchanged.

Operation WDT last two 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 1 are changed to 0 and vice-versa.

Operation [m]← $\overline{[m]}$

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 1 are changed to 0 and vice-versa. The complemented result is stored in the accumulator and the contents of the data memory remains unchanged.

Operation ACC ← $\overline{[m]}$

Affected flag(s)

TC2	TC1	TO	PD	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
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)

TC2	TC1	TO	PD	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)

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

DECA [m] Decrement data memory and place result in 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)

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 1.

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 1, 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 0~10 of the program counter are unconditionally replaced with the directly-specified address, and control is 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 memories).

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 memories) 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 memories) 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.

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 1 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 1 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 1 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 1 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 1 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 1 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 1 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 1 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
—	—	—	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 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 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, 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 1.

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 1.

Operation [m].i ← 1

Affected flag(s)

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

SIZ [m] Skip if increment data memory is 0

Description The contents of the specified data memory is 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. 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 0

Description The contents of the specified data memory is incremented by 1. If the result is 0, the next instruction is skipped and the result 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 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 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. 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
—	—	—	—	√	√	√	√

SWAP [m] Swap nibbles within the data memory

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

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s)

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

SWAPA [m] Swap data memory and place result in 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)

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

SZ [m] Skip if data memory is 0

Description If the contents 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. 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 0

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

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)

TC2	TC1	TO	PD	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)

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
—	—	—	—	—	√	—	—

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

Holtek Semiconductor Inc. (Sales Office)
11F, No.576, Sec.7 Chung Hsiao E. Rd., Taipei, Taiwan
Tel: 886-2-2782-9635
Fax: 886-2-2782-9636
Fax: 886-2-2782-7128 (International sales hotline)

Holtek Semiconductor (Hong Kong) Ltd.
RM.711, Tower 2, Cheung Sha Wan Plaza, 833 Cheung Sha Wan Rd., Kowloon, Hong Kong
Tel: 852-2-745-8288
Fax: 852-2-742-8657

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

Holmate Technology Corp.
48531 Warm Springs Boulevard, Suite 413, Fremont, CA 94539
Tel: 510-252-9880
Fax: 510-252-9885
http://www.holmate.com