HTG2170
8-bit LCD Microcontroller

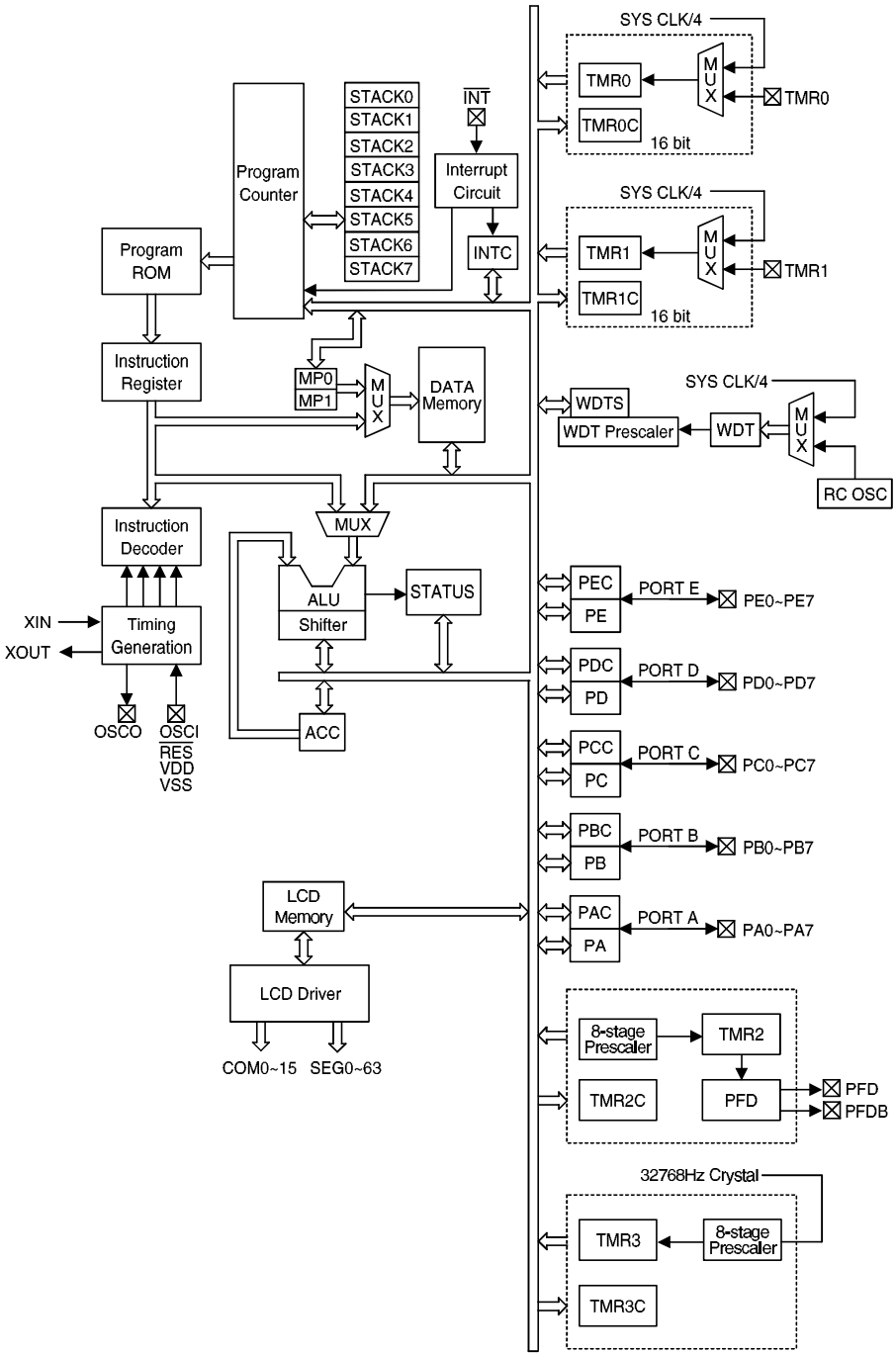
Features

- Operating voltage: 2.4V~5.2V
- 40 bidirectional I/O lines (Max.)
- One interrupt input
- Two 8K × 16 program ROM Bank
- (208 + 192) × 8 data RAM
- 1/5 bias, 16 common × 64 segment LCD driver (Max.)
- One 8-bit programmable Timer with 8 stage prescaler for PFD
- One 8-bit programmable Timer with 8 stage prescaler for Time base
- Two 16-bit programmable timer/event counters with overflow interrupts
- On-chip crystal and RC oscillator for system clock and 32768Hz Crystal oscillator for Timebase
- Watchdog timer
- Halt function and wake-up feature reduce power consumption
- Eight-level subroutine nesting
- Bit manipulation instructions
- 63 powerful instructions

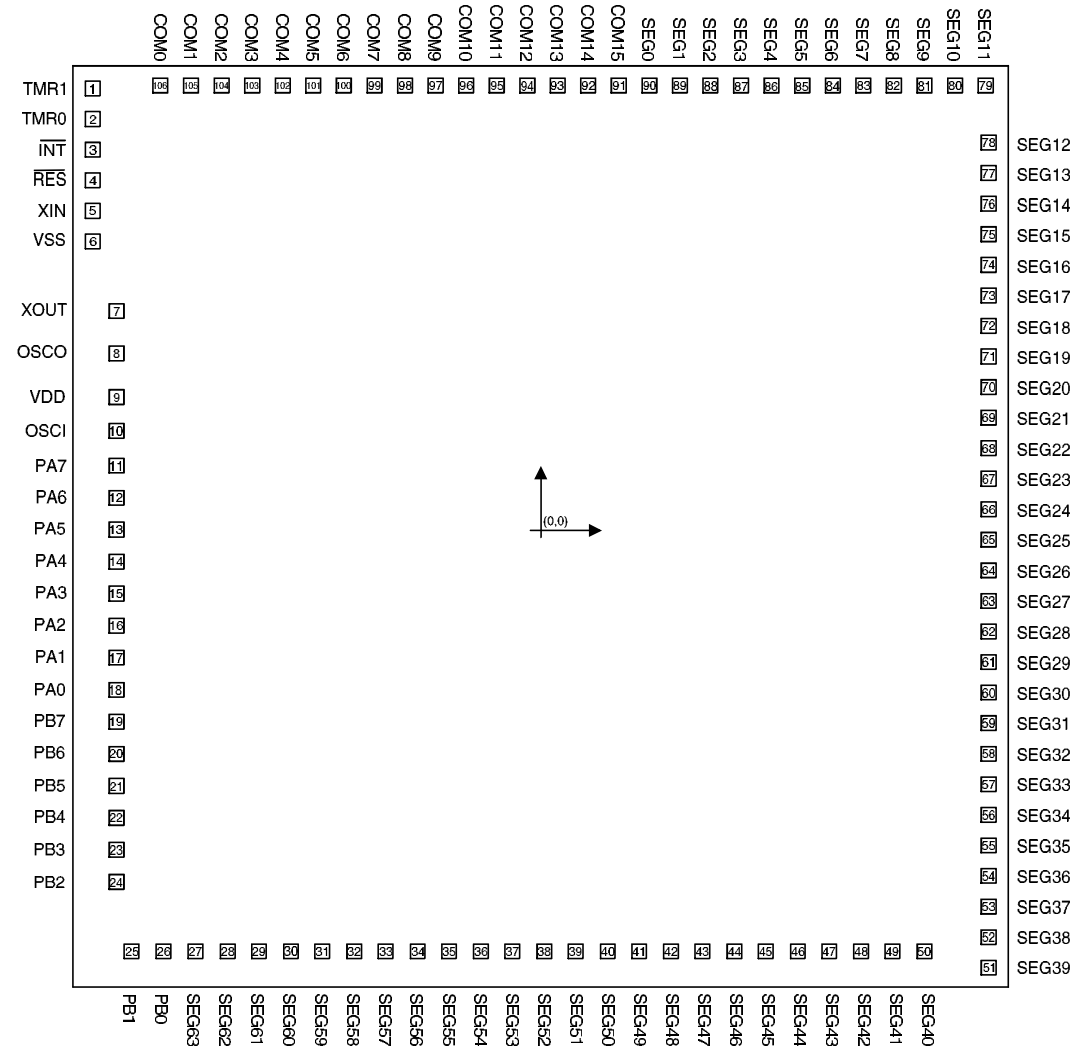
General Description

The HTG2170 is an 8-bit high performance RISC-like microcontroller. The single cycle instruction and two-stage pipeline architecture make it suitable for high speed application. The device is ideally suited for multiple LCD low power application among which are calculators, clock timer, game, scales, toys and hand held LCD products, as well as for battery systems.

Block Diagram



Pad Assignment



Chip size: $3910 \times 3950\ (\mu m)^2$

* The IC substrate should be connected to V_{SS} in the PCB layout artwork.

Pad Coordinates

Unit: μm

Pad No.	X	Y	Pad No.	X	Y
1	–1832.40	1807.45	54	1832.40	–1413.95
2	–1832.40	1682.45	55	1832.40	–1288.95
3	–1832.40	1557.45	56	1832.40	–1163.95
4	–1832.40	1432.45	57	1832.40	–1038.95
5	–1832.40	1307.45	58	1832.40	–913.95
6	–1832.40	1182.45	59	1832.40	–788.95
7	–1734.90	897.05	60	1832.40	–663.95
8	–1734.90	724.45	61	1832.40	–538.95
9	–1734.90	545.25	62	1832.40	–413.95
10	–1734.90	406.65	63	1832.40	–288.95
11	–1734.90	264.75	64	1832.40	–163.95
12	–1734.90	133.75	65	1832.40	–38.95
13	–1734.90	2.75	66	1832.40	86.05
14	–1734.90	–128.25	67	1832.40	211.05
15	–1734.90	–259.25	68	1832.40	336.05
16	–1734.90	–390.25	69	1832.40	461.05
17	–1734.90	–521.25	70	1832.40	586.05
18	–1734.90	–652.25	71	1832.40	711.05
19	–1734.90	–783.25	72	1832.40	836.05
20	–1734.90	–914.25	73	1832.40	961.05
21	–1734.90	–1045.25	74	1832.40	1086.05
22	–1734.90	–1176.25	75	1832.40	1211.05
23	–1734.90	–1307.25	76	1832.40	1336.05
24	–1734.90	–1438.25	77	1832.40	1461.05
25	–1673.00	–1722.45	78	1832.40	1586.05
26	–1542.00	–1722.45	79	1819.90	1819.95
27	–1411.00	–1722.45	80	1694.90	1819.95
28	–1280.00	–1722.45	81	1569.90	1819.95
29	–1152.00	–1722.45	82	1444.90	1819.95
30	–1021.00	–1722.45	83	1319.90	1819.95
31	–893.00	–1722.45	84	1194.90	1819.95
32	–762.00	–1722.45	85	1069.90	1819.95
33	–634.00	–1722.45	86	944.90	1819.95
34	–503.00	–1722.45	87	819.90	1819.95
35	–375.00	–1722.45	88	694.90	1819.95
36	–244.00	–1722.45	89	569.90	1819.95
37	–116.00	–1722.45	90	444.90	1819.95
38	15.00	–1722.45	91	319.90	1819.95
39	143.00	–1722.45	92	194.90	1819.95
40	274.00	–1722.45	93	69.90	1819.95

Pad No.	X	Y	Pad No.	X	Y
41	402.00	–1722.45	94	–55.10	1819.95
42	533.00	–1722.45	95	–180.10	1819.95
43	661.00	–1722.45	96	–305.10	1819.95
44	792.00	–1722.45	97	–430.10	1819.95
45	920.00	–1722.45	98	–555.10	1819.95
46	1051.00	–1722.45	99	–680.10	1819.95
47	1179.00	–1722.45	100	–805.10	1819.95
48	1310.00	–1722.45	101	–930.10	1819.95
49	1438.00	–1722.45	102	–1055.10	1819.95
50	1569.00	–1722.45	103	–1180.10	1819.95
51	1832.40	–1788.95	104	–1305.10	1819.95
52	1832.40	–1663.95	105	–1430.10	1819.95
53	1832.40	–1538.95	106	–1555.10	1819.95

Pad Description

Pad Name	I/O	Internal Connection	Description
TMR1	I	—	Schmitt trigger input for timer/event counter 1
TMR0	I	—	Schmitt trigger input for timer/event counter 0
$\overline{\text{INT}}$	I	—	External interrupt schmitt trigger input with pull-high resistor. Edge triggered activated on a high to low transition.
$\overline{\text{RES}}$	I	—	Schmitt trigger reset input. Active low.
XIN XOUT	I O	Enable/disable	Crystal oscillator (32.768kHz) for Timer 3 and LCD clock.
VSS		—	Negative power supply, GND
VDD		—	Positive power supply.
OSCI OSCO	I O	Crystal or RC	OSCI, OSCO are connected to the RC network or the Crystal (determined by mask option) for the internal system clock. In the case of RC operation, OSC2 is the output terminal for 1/4 system clock.
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 mask option. Software instructions determine the CMOS output or schmitt trigger input with or without pull high resistor (mask option).

Pad Name	I/O	Internal Connection	Description
PB0~PB7	I/O	Pull-high or None	Bidirectional 8-bit Input/Output port. Software instructions determine the CMOS output or schmitt trigger input with or without pull high resistor (mask option). PB0 and PB1 can be selected as PFD and PFDB respectively by mask option.
PC0~PC7 PD0~PD7 PE0~PE7	I/O	Pull-high or None I/O pins or segment	Three bidirectional 8-bit Input/Output ports. Software instructions determine the CMOS output or schmitt trigger input with or without pull high resistor (mask option). PC, PD, PE can option for segment output individually by mask option. PC0~PC7 share pins with SEG56~SEG63, PD0~PD7 share pins with SEG48~SEG55, and PE0~PE7 share pins with SEG40~SEG47.
SEG39~SEG0	O	—	Segment signal
COM15~COM0	O	—	Common signal

Absolute Maximum Ratings*

Supply Voltage .....–0.3V to 5.5V
Input Voltage .....V_{SS}–0.3V to V_{DD}+0.3V
Storage Temperature .....–50°C to 125°C
Operating Temperature.....0°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

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.4	—	5.2	V
I_{DD1}	Operating Current (Crystal OSC)	3V	No load f_{SYS}= 4MHz	—	1	2	mA
		5V		—	2	4	mA
I_{DD2}	Operating Current (RC OSC)	3V	No load f_{SYS}= 2MHz	—	0.7	1.5	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	—	—	2	μA
		5V		—	—	3	μA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{STB3}	Standby Current (WDT Disabled) With 10μA LCD Bias Option (RTC ON, LCD ON)	3V	No load, HALT mode	—	30	—	μA
		5V	No load, HALT mode	—	50	—	μA
I_{STB4}	Standby Current (WDT Disabled) With 10μA LCD Bias Option (RTC ON, LCD OFF)	3V	No load, HALT mode	—	2	—	μA
		5V	No load, HALT mode	—	3	—	μA
V_{IL1}	Input Low Voltage for I/O Ports	3V	—	0	—	0.7	V
		5V	—	0	—	1.3	V
$V_{IH}1$	Input High Voltage for I/O Ports	3V	—	2.3	—	3	V
		5V	—	3.8	—	5	V
V_{IL2}	Input Low Voltage (TMR0,TMR1,$\overline{INT}$)	3V	—	0	—	0.7	V
		5V	—	0	—	1.3	V
V_{IH2}	Input High Voltage (TMR0,TMR1,$\overline{INT}$)	3V	—	2.3	—	3	V
		5V	—	3.8	—	5	V
V_{IL3}	Input Low Voltage ($\overline{RES}$)	3V	—	0	—	0.9	V
		5V	—	0	—	2	V
V_{IH3}	Input High Voltage ($\overline{RES}$)	3V	—	2.5	—	3	V
		5V	—	4.2	—	5	V
I_{OL1}	I/O Ports Sink Current	3V	V_{OL}=0.3V	1.5	4	—	mA
		5V	V_{OL}=0.5V	4	10	—	mA
I_{OH1}	I/O Ports Source Current	3V	V_{OH}=2.7V	–1	–2	—	mA
		5V	V_{OH}=4.5V	–2	–4.5	—	mA
I_{OL2}	Segment, Common Output Sink Current	3V	V_{OL}=0.3V	100	200	—	μA
		5V	V_{OL}=0.5V	250	500	—	μA
I_{OH2}	Segment, Common Output Source Current	3V	V_{OH}=2.7V	75	150	—	μA
		5V	V_{OH}=4.5V	200	400	—	μA
R_{PH}	Pull-high Resistance of I/O Ports, TMR0, TMR1 and $\overline{INT}$	3V	—	20	40	60	kΩ
		5V	—	10	25	50	kΩ

A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS1}	System Clock (Crystal OSC)	3V	—	400	—	4000	kHz
		5V	—	400	—	8000	kHz
f_{SYS2}	System Clock (RC OSC)	3V	—	400	—	2000	kHz
		5V	—	400	—	3000	kHz
f_{TIMER}	Timer I/P Frequency (TMR0, TMR1)	3V	—	0	—	4000	kHz
		5V	—	0	—	4000	kHz
t_{WDTOSC}	Watchdog Oscillator	3V	—	45	90	180	μs
		5V		35	65	130	μs
t_{WDT1}	Watchdog Time-out Period (RC)	3V	Without WDT prescaler	12	23	45	ms
		5V		9	17	35	ms
t_{WDT2}	Watchdog Time-out Period (System Clock)	—	Without WDT prescaler	—	1024	—	t_{SYS}
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 HTG2170 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 14-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 16384 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.

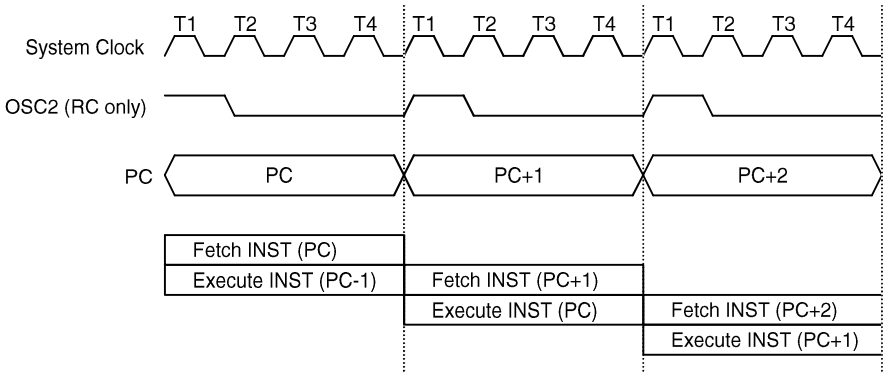
When a control transfer takes place, an additional dummy cycle is required.

Program memory – 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 16384 × 16 bits, addressed by the program counter and table pointer.

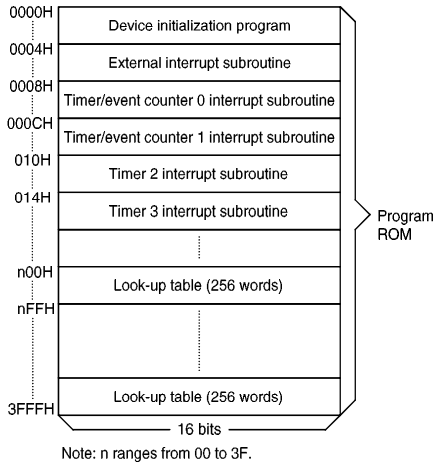
Certain locations in the program memory are reserved for special usage:

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



Execution flow

- 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/00CH
  This area is reserved for the timer/event counter 0/1 interrupt service program. If a timer interrupt results from a timer/event counter 0/1 overflow, and if the interrupt is enabled and the stack is not full, the program begins execution at location 008H/00CH.
- Location 010H/014H
  This area is reserved for the timer 2/3 interrupt service program. If a timer interrupt results from a timer 2/3 overflow, and if the interrupt is enabled and the stack is not full, the program begins execution at location 010H/014H.
- Location 020H
  For the best condition, the location is the begin of writing program.



Program memory

- Table location
  Any location in the ROM space can be used as look–up tables. The instructions TABRDC [m] (the any page, 1 page=256 words) and TABRDL [m] (the last page) transfer the con-

Mode	Program Counter													
	***13**	***12**	***11**	***10**	***9**	***8**	***7**	***6**	***5**	***4**	***3**	***2**	***1**	***0**
Initial reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0
External interrupt	0	0	0	0	0	0	0	0	0	0	0	1	0	0
Timer/event counter 0 overflow	0	0	0	0	0	0	0	0	0	0	1	0	0	0
Timer/event counter 1 overflow	0	0	0	0	0	0	0	0	0	0	1	1	0	0
Timer 2 overflow	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Timer 3 overflow	0	0	0	0	0	0	0	0	0	1	0	1	0	0
Skip	**PC+2**													
Loading PCL	*13	*12	*11	*10	*9	*8	@7	@6	@5	@4	@3	@2	@1	@0
Jump, call branch	BP.5	#12	#11	#10	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
Return from subroutine	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

Program counter

Notes: *13~*0: Program counter bits
@7~@0: PCL bits
#12~#0: Instruction code bits
S13~S0: Stack register bits
BP.5: Bit 5 of bank point (04H)

tents 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 (TBHP, TBLP) is a read/write register (1FH, 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

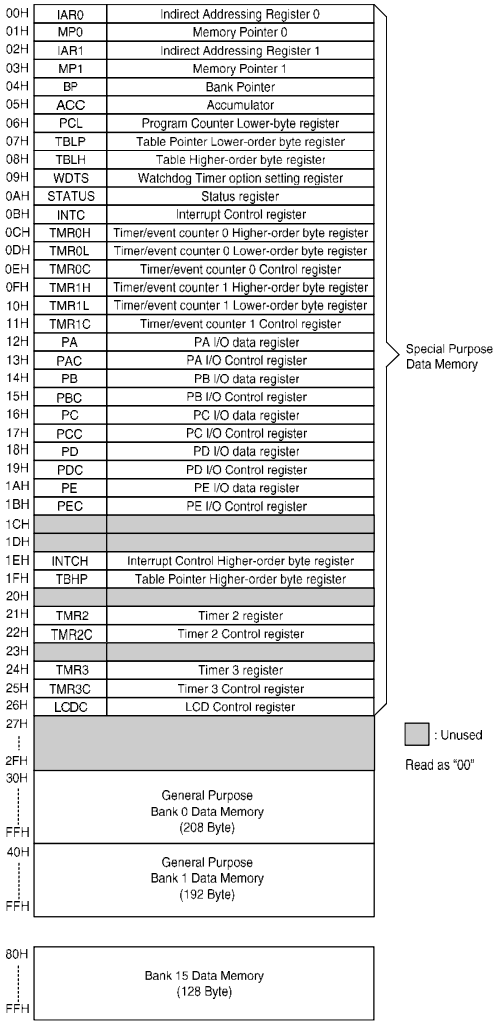
This is a special part of the memory which is used to save the contents of the program counter (PC) only. The stack is organized into eight levels and is neither part of the data nor part of the program space, and is neither readable nor writeable. The activated level is indexed by the stack pointer (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 eight return address are stored).

Instruction(s)	Table Location													
	***13**	***12**	***11**	***10**	***9**	***8**	***7**	***6**	***5**	***4**	***3**	***2**	***1**	***0**
TABRDC [m]	#5	#4	#3	#2	#1	#0	@7	@6	@5	@4	@3	@2	@1	@0
TABRDL [m]	1	1	1	1	1	1	@7	@6	@5	@4	@3	@2	@1	@0

Table location

Notes: @7~@0: TabAdrL bits
#5~#0: TabAdrH bits
*13~*0: Current Program counter bits



RAM mapping

Data memory – RAM

- Bank 0 (BP4~0=00000):
  The Bank 0 data memory include special purpose and general purpose memory. The special purpose memory is addressed from 00H to 26H, while general purpose memory is addressed from 30H to FFH. The remaining space from 27H to 2FH are reserved for future expanded usage and reading these locations will return the result 00H. 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 the memory pointer registers (MP0;01H, MP1;03H)

- Bank 1 (BP4~0=0001B)
  The range of RAM starting from 40H to FFH are for general purpose. Only MP1 can deal with the memory of this range.

- Bank 15 (BP4~0=01111B)
  The range of RAM starts from 80H to FFH. Every bit stands for one dot on the LCD. If the bit is “1”, the light of the dot on the LCD will be turned on. If the bit is “0”, then it will be turned off. Only MP1 can deal with the memory of this range.
  The contrast form of RAM location, COMMON, and SEGMENT is as follows.

LCD display memory: (Bank 15)

Address	80	81	82	83	84	85 ---------- 91	92	-----	-----	-----	BF
COM0	Bit0										
COM1	Bit1										
COM2	Bit2										
COM3	Bit3										
COM4	Bit4										
COM5	Bit5										
COM6	Bit6										
COM7	Bit7										
Address	C0H	C1H	C2H	C3H	C4H	C5H ---------- ----	-----	-----	-----	FEH	FFH
COM8	Bit0										
COM9	Bit1										
COM10	Bit2										
COM11	Bit3										
COM12	Bit4										
COM13	Bit5										
COM14	Bit6										
COM15	Bit7										
	Seg0	Seg1	Seg2	Seg3	Seg4	Seg5 ---------- Seg17	Seg18	-----	-----	-----	Seg63

LCD driver output

The maximum output number of the HTG2170 LCD driver is 16×64. The Common output signal can be selected as 16 com or 8 com by mask option. The LCD driver bias type is "R" type, no external capacitor is required and the bias voltage is 1/5 bias. Some of the Segment outputs share pins with I/O pins, PE0~PE7 (SEG40~47), PD0~PD7 (SEG48~55) and PC0~PC7 (SEG56~63). Whether segment output or I/O pin can be decided by mask option bit by bit individually.

LCD driver output can be enabled or disabled by setting ELCDB (bit 0 of LCDC; 26H) without the influence of the related memory condition.

Register	Bit No.	Label	Function
LCDC	0	ELCDB	Control the LCD output (1=disabled; 0=enabled)
	1	—	Unused bit Should be written as "1"
	2	RTMR0	Select the TMR0 Pull-high resistor (1=with Pull-high; 0=without Pull-high)
	3	RTMR1	Select the TMR1 Pull-high resistor (1=with Pull-high; 0=without Pull-high)
	4~7	—	Unused bits Can be used as general purpose RAM

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, is not supported. The memory pointer registers, MP0 and MP1, are 8-bit registers which can be used to access the data memory by combining corresponding indirect addressing registers but Bank 15 can use MP1 only.

Accumulator

The accumulator is closely related to ALU operations. It is also mapped to location 05H of the data memory and it can carry out immediate data operations. The data movement between two data memories has to 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 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 system power up, watchdog timer overflow, executing the HALT instruction and clearing the watchdog timer.

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.

Interrupt

The HTG2170 provides an external interrupt and internal timer/event counter interrupts. The Interrupt Control register (INTC;0BH, INTCH;1EH) 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 a certain interrupt needs servicing within the service routine, the programmer may set the EMI bit and the corresponding bit of INTC to 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.

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 followed by a branch to subroutines 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) are altered by the interrupt service program which corrupt the desired control sequence, the contents should 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 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 0 interrupt is initialized by setting the timer/event counter 0 interrupt request flag (T0F; bit 5 of INTC), resulting from a timer/event counter 0 overflow. When the interrupt is enabled, and the stack is not full and the T0F bit is set, a subroutine call to location 08H will occur. The related interrupt request flag (T0F) will be reset and the EMI bit cleared to disable further interrupts.

The timer/event counter 1 and Timer 2/3 interrupts are operated in the same manner as timer/event counter 0. The related interrupt control bits ET1I and T1F of timer/event counter 1 are bit 3 and bit 6 of the INTC respectively. While ET2I/ET3I and T2F/T3F are the related control bits and the related request flags of TMR2/TMR3, which locate at bit0/bit1 and bit4/bi5 of the INTCH respectively.

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, 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 applied are shown in the following table. These can be masked by resetting the EMI bit.

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 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 powers up or a CLR WDT instruction is executed. 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.
—	6~7	Undefined bits, read as 0

Status register

No.	Interrupt Source	Priority	Vector
a	External interrupt	1	04H
b	Timer/event counter 0 overflow	2	08H
c	Timer/event counter 1 overflow	3	0CH
d	Timer 2 overflow	4	10H
e	Timer 3 overflow	5	14H

The timer/event counter 0/1 and Timer 2/3 interrupt request flag (T0F/T1F/T2F/T3F), External interrupt request flag (EIF), Enable Timer/event counter 0/1/2/3 bit (ET0I/ET1I/ET2I/ET3I), Enable external interrupt bit (EEI) and Enable master interrupt bit (EMI) constitute an interrupt control register (INTC/INTCH) which is located at 0BH/1EH in the data memory. EMI, EEI, ET0I, ET1I, ET2I, ET3I, are used to control the enabling/disabling of interrupts. These bits prevent the requested interrupt from being serviced. Once

Register	Bit No.	Label	Function
INTC	0	EMI	Controls the (global) interrupt (1= enable; 0= disable)
	1	EEI	Controls the external interrupt (1= enable; 0= disable)
	2	ET0I	Controls the timer/event counter 0 interrupt (1= enable; 0= disable)
	3	ET1I	Controls the timer/event counter 1 interrupt (1= enable; 0= disable)
	4	EIF	External interrupt request flag (1= active; 0= inactive)
	5	T0F	Internal timer/event counter 0 request flag (1= active; 0= inactive)
	6	T1F	Internal timer/event counter 1 request flag (1= active; 0= inactive)
	7	—	Unused bit

INTC register

Register	Bit No.	Label	Function
INTCH	0	ET2I	Controls the Timer 2 interrupt (1= enable; 0= disable)
	1	ET3I	Controls the Timer 3 interrupt (1= enable; 0= disable)
	2~3	—	Should be set “0” always
	4	T2F	Internal Timer 2 request flag (1= active; 0= inactive)
	5	T3F	Internal Timer 3 request flag (1= active; 0= inactive)
	6~7	—	Should be set “0” always

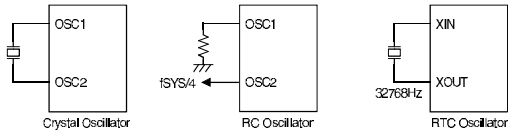
INTCH register

the interrupt request flags (T0F, T1F, T2F, T3F, EIF) are set, they will remain in the INTC/INTCH 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. 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 "CALL subroutine" should not operate in the interrupt subroutine as it will damage the original control sequence.

Oscillator configuration

There are four oscillator circuits in the HTG2170.



System and RTC oscillator

Two of them are designed for system clocks; the RC oscillator and the Crystal oscillator, which are determined by mask 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 V_{SS} is needed and the resistance must range from 51kΩ to 1MΩ. 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 $V_{DD,}$ temperature and the chip itself due to process variations. It is, therefore, not suitable for timing sensitive operations where accurate oscillator frequency is desired.

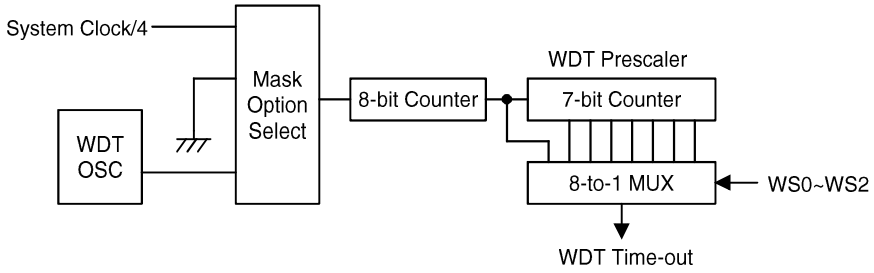
If the Crystal oscillator is used, a crystal across OSC1 and OSC2 is needed to provide the feedback and phase shift needed for oscillator, no other external components are needed. Instead of a crystal, the resonator can also be connected between OSC1 and OSC2 to get a frequency reference, but two external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

Another RTC (Real Time Clock) oscillator (32768Hz) is used to provide clock source for the LCD driver and Timer 3. And it can be enabled or disabled by mask option.

The other 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 mask option to conserve power.

Watchdog timer – WDT

The WDT clock source is implemented by a dedicated RC oscillator (WDT oscillator) or instruction clock (system clock divided by 4), decided by mask option. 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 mask option. If the watchdog timer is disabled, all the executions related to WDT result in no operation.



Watchdog timer

Once the internal WDT oscillator (RC oscillator with 78μs period normally) is selected, it is first divided by 256 (8-stages) to get the nominal time-out period of approximately 20 ms. This time-out period may vary with temperature, V_{DD} and process variations. By invoking the WDT prescaler, longer time-out periods can be realized. Writing data to WS2, WS1, WS0 (bits 2,1,0 of the WDTS) can give different time-out periods. If WS2, WS1, WS0 are all equal to 1, the division ratio is up to 1:128, and the maximum time-out period is 2.6 seconds.

If the WDT oscillator is disabled, the WDT clock may still come from the instruction clock and operate in the same manner except that in the HALT state the WDT may stop counting and lose its protecting purpose. In this situation the logic can only be restarted by an external logic. The high nibble and bit 3 of the WDTS are reserved for user defined flags, and the programmer may use these flags to indicate some specified status.

WS2	WS1	WS0	Division Ratio
0	0	0	1:1
0	0	1	1:2
0	1	0	1:4
0	1	1	1:8
1	0	0	1:16
1	0	1	1:32
1	1	0	1:64
1	1	1	1:128

WDTS register

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 (including the WDT prescaler), three methods are adopted; external reset (a low level to $\overline{RES}$), software instructions, or a HALT instruction. The software instructions include CLR WDT and the other set — CLR WDT1 and CLR WDT2. Of these two types of instructions, only one can be active depending on the mask option — "CLR WDT times selection option". If the "CLR WDT" is selected (i.e.. CLRWDT 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.. CLRWDT times equal two), these two instructions must be executed to clear the WDT; otherwise, the WDT may reset the chip due to time-out.

Power down operation – HALT

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

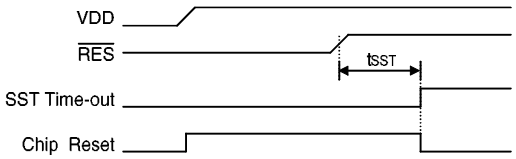
- The system oscillator will turn off but the WDT oscillator keeps running (if the WDT oscillator is selected).
- The contents of the on–chip RAM and registers remain unchanged.
- WDT and WDT prescaler will be cleared and do recounting again (if the clock of WDT comes from the WDT oscillator).
- All 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". By examining the TO and PD flags, the reason for chip reset can be determined. The PD flag is cleared when the system powers up or upon 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, the regular interrupt response takes place.

Once a wake-up event 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 acknowledge, 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.



Reset timing chart

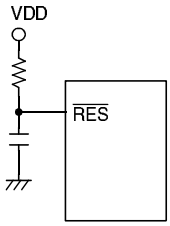
To minimize power consumption, all I/O pins should be carefully managed before entering the HALT status.

Reset

There are three ways in which a reset can occur:

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

The WDT time-out during HALT is different from other chip reset conditions, since it can perform a "warm reset" that just resets 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".



Reset circuit

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

Note: "u" means "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 awakes from the HALT state.

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 will enable the SST delay.

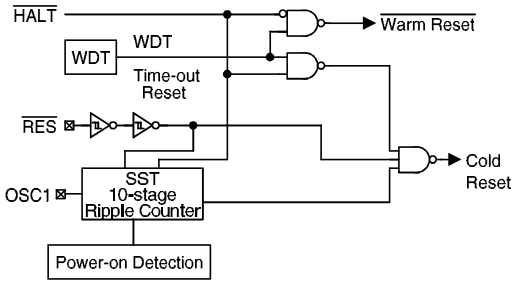
The functional unit chip reset status are shown below.

PC	000H
Interrupt	Disable
Prescaler	Clear
WDT	Clear. After master reset, WDT begins counting
Timer/event counter (0/1/2/3)	Off
LCD Display	Disable
Pull-high of TMR0 /TMR1	with
Input/output Ports	Input mode
SP	Points to the top of the stack

Timer/event counter 0/1

Two timer/event counters are implemented in the HTG2170. The timer/event counter 0 and timer/event counter 1 contain 16-bit programmable count-up counters and the clock may come from an external source or from the system clock divided by 4.

Using the internal instruction clock, there is only one reference time-base. The external clock input allows the user to count external events, measure time intervals or pulse width, or generate an accurate time base.



Reset configuration

There are three registers related to timer/event counter 0; TMR0H (0CH), TMR0L (0DH), TMR0C (0EH). Writing TMR0L only writes the data into a low byte buffer, and writing TMR0H will write the data and the contents of the low byte buffer into the timer/event counter 0 Preload register (16-bit) simultaneously. The timer/event counter 0 preload register is changed by writing TMR0H operations and writing TMR0L will keep the timer/event counter 0 preload register unchanged.

Reading TMR0H will also latch the TMR0L into the low byte buffer to avoid the false timing problem. Reading TMR0L returns the contents of the low byte buffer. In other words, the low byte of timer/event counter 0 cannot be read directly. It must read the TMR0H first to make the low byte contents of timer/event counter 0 be latched into the buffer.

There are three registers related to timer/event counter 1; TMR1H (0FH), TMR1L (10H), TMR1C (11H). The timer/event counter 1 operates in the same manner as timer/event counter 0.

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

Register	Reset (power on)	WDT time-out (normal operation)	$\overline{\text{RES}}$ reset (normal operation)	$\overline{\text{RES}}$ reset (HALT)	WDT time-out (HALT)
TMR1H	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR1L	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR1C	00-0 1---	00-0 1---	00-0 1---	00-0 1---	uu-u u---
TMR0H	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR0L	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR0C	00-0 1---	00-0 1---	00-0 1---	00-0 1---	uu-u u---
TMR2	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR2C	00-0 1000	00-0 1000	00-0 1000	00-0 1000	uu-u uuuu
TMR3	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR3C	00-0 1000	00-0 1000	00-0 1000	00-0 1000	uu-u uuuu
INTCH	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
TBHP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
Program Counter	0000H	0000H	0000H	0000H	0000H*
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu
BP	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
LCDC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
INTC	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
WDTS	0000 0111	0000 0111	0000 0111	0000 0111	uuuu uuuu

Register	Reset (power on)	WDT time-out (normal operation)	$\overline{\text{RES}}$ reset (normal operation)	$\overline{\text{RES}}$ reset (HALT)	WDT time-out (HALT)
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PE	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu

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

The TMR0C is the timer/event counter 0 control register, which defines the timer/event counter 0 options. The timer/event counter 1 has the same options as the timer/event counter 0 and is defined by TMR1C.

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

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 (TMR0/TMR1) pin. The Timer mode functions as a normal timer with the clock source coming from the instruction clock. The pulse width measurement mode can be used to count the high or low level duration of the external signal (TMR0/TMR1). The counting is based on the instruction 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 FFFFH. Once an overflow occurs, the counter is reloaded from the timer/event counter preload register and generates the corresponding interrupt request flag (T0F/T1F; bit 5/6 of INTC) at the same time.

In pulse width measurement mode the TON and TE bits are equal to one, once the TMR0/TMR1 received a transient from low to high (or high to low; if the TE bit is 0) it will start counting until the TMR0/TMR1 returns to the original level and resets the TON. The measured result will remain in the timer/event counter even if the activated transient occurs again. In other words, only one cycle measurements 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 operating mode, the timer/event counter starts counting not according to the logic level but according to the transient edges. In the case of counter overflows, the counter is reloaded from the timer/event counter preload register and issues the interrupt request just like the other two modes.

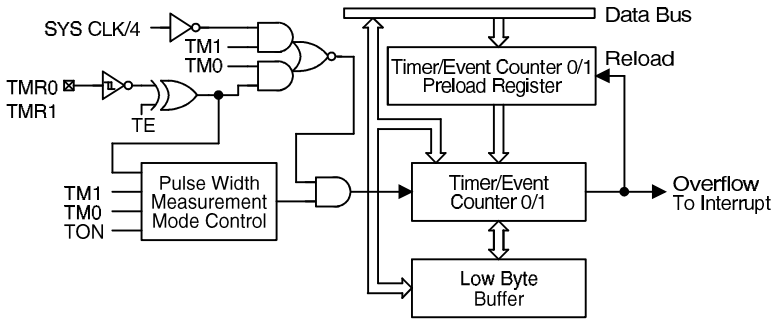
To enable the counting operation, the Timer ON bit (TON; bit 4 of TMR0C/TMR1C) should be set to 1. In the pulse width measurement mode, the TON will be cleared automatically after the measurement cycle is completed. But in the other two modes the TON can only be reset by instruction. The overflow of the timer/event counter is one of the wake-up sources. No matter what the operation mode is, writing a 0 to ET0I/ET1I can disable the corresponding interrupt service.

In the case of timer/event counter OFF condition, writing data to the timer/event counter preload register will also reload that data to the timer/event counter. But if the timer/event counter is turned on, data written to the timer/event counter will only be kept in the timer/event counter preload register. The timer/event counter will still operate until overflow occurs.

When the timer/event counter (reading TMR0H/TMR1H) 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.

Label	Bits	Function
–	0~2	Unused bits, read as "0"
TE	3	To define the TMR0/TMR1 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)
–	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) 11= Pulse width measurement mode 00= Unused

TMR0C/TMR1C register



Timer/event counter 0/1

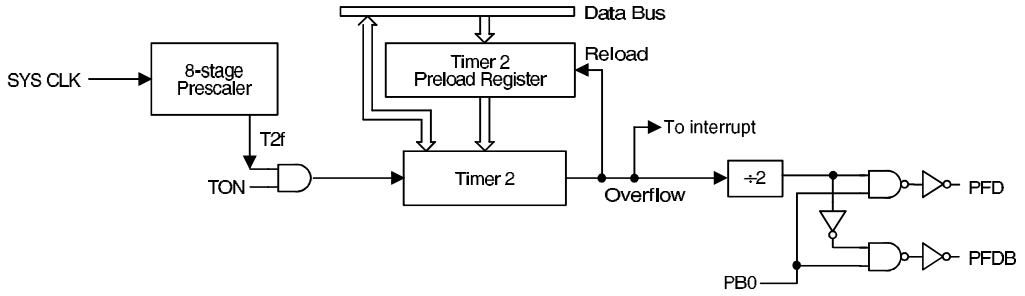
Timer 2/3

Timer 2 is an 8-bit counter, and its clock source comes from the system clock divided by an 8-stage prescaler. There are two registers related to Timer 2 ; TMR2 (21H) and TMR2C (22H). Two physical registers are mapped to TMR2 location; writing TMR2 makes the starting value be put in the Timer 2 preload register and reading the TMR2 gets the contents of the Timer 2 counter. The TMR2C is a control register, which defines the division ratio of the prescaler and counting enable or disable.

Writing data to B2, B1 and B0 (bit 2, 1, 0 of TMR2C) can yield various clock sources.

Once the Timer 2 starts counting, it will count from the current contents in the counter to FFH. Once an overflow occurs, the counter is reloaded from a preload register, and generates an interrupt request flag (T2F; bit 4 of INTCH). To enable the counting operation, the timer On bit (TON ; bit 4 of TMR2C) should be set to "1". For proper operation, bit 6 of TMR2C should be set to "1", and bit 3, bit7 should be set to "0".

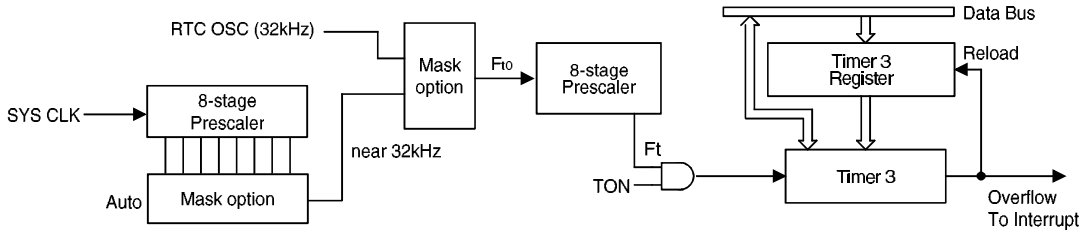
Timer 2 can also be used as buzzer output by setting PB.0 and PB.1 as PFD and PFDB output respectively by mask option. When the PFD/PFDB function is selected, setting PB.0 "1" will enable PFD/PFDB output and setting PB.0 "0" will disable PFD/PFDB output. PFD Frequency= T2f/ [(256-TMR2)×2].



Timer 2

TMR2C			T2f
B2	B1	B0	
0	0	0	SYS CLK/2
0	0	1	SYS CLK/4
0	1	0	SYS CLK/8
0	1	1	SYS CLK/16
1	0	0	SYS CLK/32
1	0	1	SYS CLK/64
1	1	0	SYS CLK/128
1	1	1	SYS CLK/256

TMR2C Bit 4 to enable/disable timer counting (0=disable;1=enable)
TMR2C Bit 6 Always write "1"
TMR2C Bit 7 Always write "0"
TMR2C Bit 3 Always write "0"



Timer 3

TMR3C			Ft
B2	B1	B0	
0	0	0	Ft0/2
0	0	1	Ft0/4
0	1	0	Ft0/8
0	1	1	Ft0/16
1	0	0	Ft0/32
1	0	1	Ft0/64
1	1	0	Ft0/128
1	1	1	Ft0/256

Ft0 can selected 10 Frequency by mask option

Auto Mask option	Ft0
1	Crystal 32768Hz
2	SYS CLK/2
3	SYS CLK/4
4	SYS CLK/8
5	SYS CLK/16
6	SYS CLK/32
7	SYS CLK/64
8	SYS CLK/128
9	SYS CLK/256

Time Base Frequency= Ft / (256 – TMR3)
TMR3C Bit4 to enable/disable timer counting (0=disable;1=enable)
TMR3C Bit6 Always write “1”
TMR3C Bit7 Always write “0”
TMR3C Bit3 Always write “0”

Timer 3 has the same structure and operating manner with Timer 2, except for clock source and PFD function. The Timer 3 can be used as a time base to generate a regular internal interrupt. The clock source of Timer 3 can come from RTC OSC (X'TAL 32kHz) or system clock divided by an 8-stage prescaler. If the RTC mask option is enabled, a 32kHz crystal is needed across XIN and XOUT pins. The 32kHz signal is processed by an 8-stage prescaler to yield various counting clock for Timer 3. There are 2 registers related to Timer 3 ; TMR3 (24H) and TMR3C (25H). Writing data to B2, B1, B0 (bit 2, 1, 0 of TMR3C) can yield various counting clock.

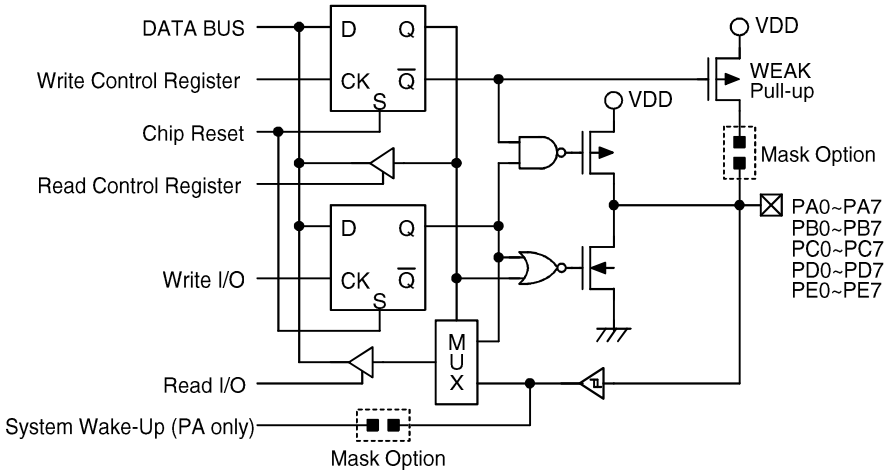
If the RTC mask option is disabled, the clock source comes from the system clock divided by an 8-stage prescaler. The division ratio is automatically determined by mask option to generate a counting clock near 32kHz.

Input/output ports

There are 40 bidirectional input/output lines in the HTG2170, labeled from PA to PE, which are mapped to the data memory of [12H], [14H], [16H], [18H], [1AH], 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, 16H, 18H, 1AH). 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, PDC, PEC) to control the input/output configuration. With this control register, CMOS output or schmitt trigger input with or without pull-high resistor (mask option) structures can be reconfigured dynamically under software control. To function as an input, the corresponding latch of the control register must write "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", the input will read the pad state. If the control register bit is "0", the contents of the latches will move to the internal bus. The latter is possible in "read-modify-write" instruction. For output function, CMOS is the only configuration. These control registers are mapped to locations 13H, 15H, 17H, 19H, 1BH.

After a chip reset, these input/output lines stay at high levels or floating (mask option). Each bit



Input/output ports

of these input/output latches can be set or cleared by the SET [m].i or CLR [m].i (m=12H, 14H, 16H, 18H, 1AH) instruction.

Some instructions first input data and then follow the output operations. For example, the SET [m].i, CLR [m].i, CPL [m] and CPLA [m] 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. PC, PD and PE can be selected as segment output by mask option. If the segment output is selected, the related I/O register (PC, PD and PE) cannot be used as general purpose register. Reading the register will reply an unknown state.

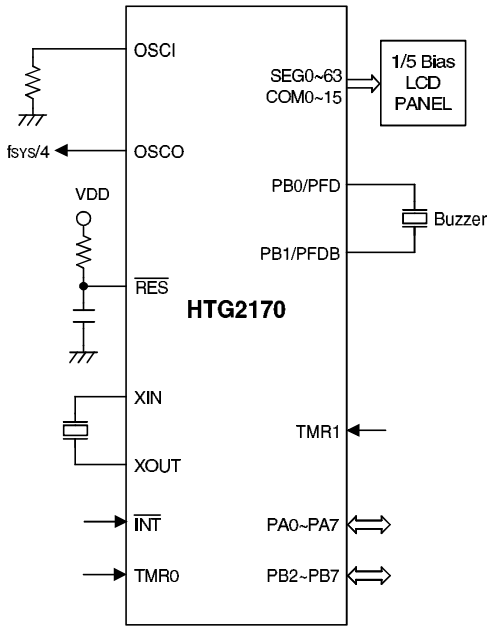
Mask option

The following shows 13 kinds of mask option in the HTG2170. All the mask options must be defined to ensure proper system function.

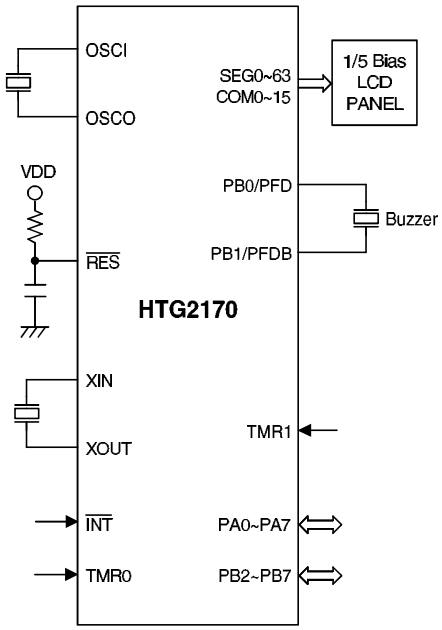
No.	Mask Option
1	OSC type selection. This option is to decide if an RC or Crystal oscillator is chosen as system clock. If the Crystal oscillator is selected, the XST (Crystal Start-up Timer) default is activated, otherwise the XST is disabled.
2	WDT source selection. There are three types of selection: on-chip RC oscillator, instruction clock or disable the WDT.
3	CLRWDT times selection. This option defines how to clear the WDT by instruction. “One time” means that the CLR WDT instruction can clear the WDT. “Two times” means only if both of the CLR WDT1 and CLR WDT2 instructions have been executed, the WDT can be cleared.
4	Wake-up selection. This option defines the wake-up function activity. External I/O pins (PA only) all have the capability to wake-up the chip from a HALT.
5	Pull-high selections. This option is to decide whether the pull-high resistance is viable or not in the input mode of the I/O ports. Each bit of an I/O port can be independently selected.
6	RTC option. This option can enable/disable the crystal (32768 Hz) Oscillator, and select LCD clock and Timer 3 clock Source.
7	Pad option 1: PB0 or PFD
8	Pad option 2: PB1 or PFDB
9	Pad option 3: PE0~PE7 or SEG40~47
10	Pad option 4: PD0~PD7 or SEG48~55
11	Pad option 5: PC0~PC7 or SEG56~63
12	LCD Bias current selection This option decides the LCD bias current. There are two types of selection depending on the common selection.
13	LCD common selection. There are two types of selection: 8 commons or 16 commons

Application Circuits

RC oscillator application



Crystal oscillator application



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
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 and 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

Mnemonic	Description	Flag Affected
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
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

Mnemonic	Description	Flag Affected
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

Notes: x: 8 bits immediate data

m: 8 bits data memory address

A: accumulator

i: 0~7 number of bits

addr: 13 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 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 and the WDT Prescaler are cleared (re-counting from zero). The power down bit (PD) and time-out bit (TO) are cleared.

Operation WDT and WDT Prescaler ← 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, WDT and the WDT Prescaler are cleared (re-counting from zero), if the other preclear WDT instruction had been executed. Execution of this instruction without the other preclear instruction only sets the indicating flag which implies that this instruction was executed and the PD and TO flags remain unchanged.

Operation WDT and WDT Prescaler ← 00H*
PD and 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, WDT and the WDT Prescaler are cleared (re-counting from zero), if the other preclear WDT instruction had been executed. Execution of this instruction without the other preclear instruction, only sets the indicating flag which implies that this instruction was executed and the PD and TO flags remain unchanged.

Operation WDT and WDT Prescaler ← 00H*
PD and 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 and 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
–	–	–	–	–	√	–	–

DAA [m] Decimal-Adjust accumulator for addition

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

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

Affected flag(s)

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] ← [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 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
–	–	–	–	–	√	–	–

HALT Enter power down mode

Description This instruction stops the program execution and turns off the system clock. The contents of the RAM and registers are retained. The WDT and prescaler are cleared. The power down bit (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 0~12 of the program counter are replaced with the directly–specified address unconditionally, 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 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 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 are 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 one bit left, with bit 7 rotated into bit 0.

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

Affected flag(s)

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 one bit left, with bit 7 rotated into bit 0, leaving the rotated result in the accumulator. The contents of the data memory remain unchanged.

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

Affected flag(s)

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 one bit left. Bit 7 replaces the carry bit; the original carry flag is rotated into the bit 0 position.

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

Affected flag(s)

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

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

Affected flag(s)

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

RR [m] Rotate data memory right

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

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

Affected flag(s)

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 one bit right with bit 0 rotated into bit 7, leaving the rotated result in the accumulator. The contents of the data memory remain unchanged.

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

Affected flag(s)

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 one bit right. Bit 0 replaces the carry bit; the original carry flag is rotated into the bit 7 position.

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

Affected flag(s)

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 one bit right. Bit 0 replaces the carry bit and the original carry flag is rotated into the bit 7 position. The rotated result is stored in the accumulator. The contents of the data memory remain unchanged.

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

Affected flag(s)

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 is replaced to get the proper instruction. This makes a two-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 two-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 two-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 is 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 is replaced to get the proper instruction. This is a two-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 two-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 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 two-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 the 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 two-cycle instruction. Otherwise proceed with the next instruction.

Operation Skip if [m]=0

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 two-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 ROM code (current page) to TBLH and data memory

Description The ROM code low byte (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 ROM code low byte (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 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
–	–	–	–	–	√	–	–