HT9280/HT9380
Pager Controller

Features

- Operating voltage: 2.2V~3.5V
- Low power crystal oscillator control
  - 512, 1200 or 2400 bps data rate
- 8K×16 program ROM
- 320×8 data RAM
- 35×4 LCD display
- 7 input lines, and 10 bidirectional I/O lines
- 8-bit programmable timer for RTC interrupt
- Battery fail interrupt and data ready interrupt
- 8-bit programmable timer/event counter and an overflow interrupt
- 8-bit programmable tone generator with buzzer output
- Watchdog timer
- Halt function and wake-up feature reduces power consumption
- 63 powerful instructions, most instructions in one machine cycle
- 8-level subroutine nesting
- Table read instruction
- Decodes CCIR Radio-Paging Code No.1 (POCSAG Code)
- 4-bit burst error correction for address codeword
- 2-bit or 4-bit burst error correction for message codeword
- Improved synchronization algorithm
- Supports 4 user addresses in 2 independent frames
- 3 RF power-on timing control pins
- Inverted or non-inverted input signal selection for decoder input
- 80-pin LQFP package

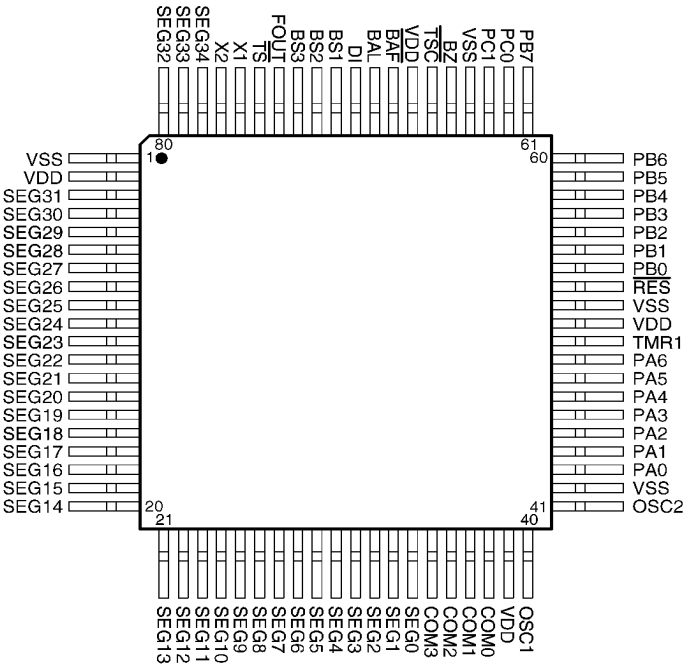
General Description

The HT9280/HT9380 is a high performance pager controller. The built-in single cycle instructions (16-bit wide) and 2-stage pipeline architecture of the HT9280/HT9380 account for its high performance. The controller contains a full function pager decoder (POCSAG code) at 512, 1200, or 2400 bits per second data rate and an LCD driver with a 35×4 dot output.

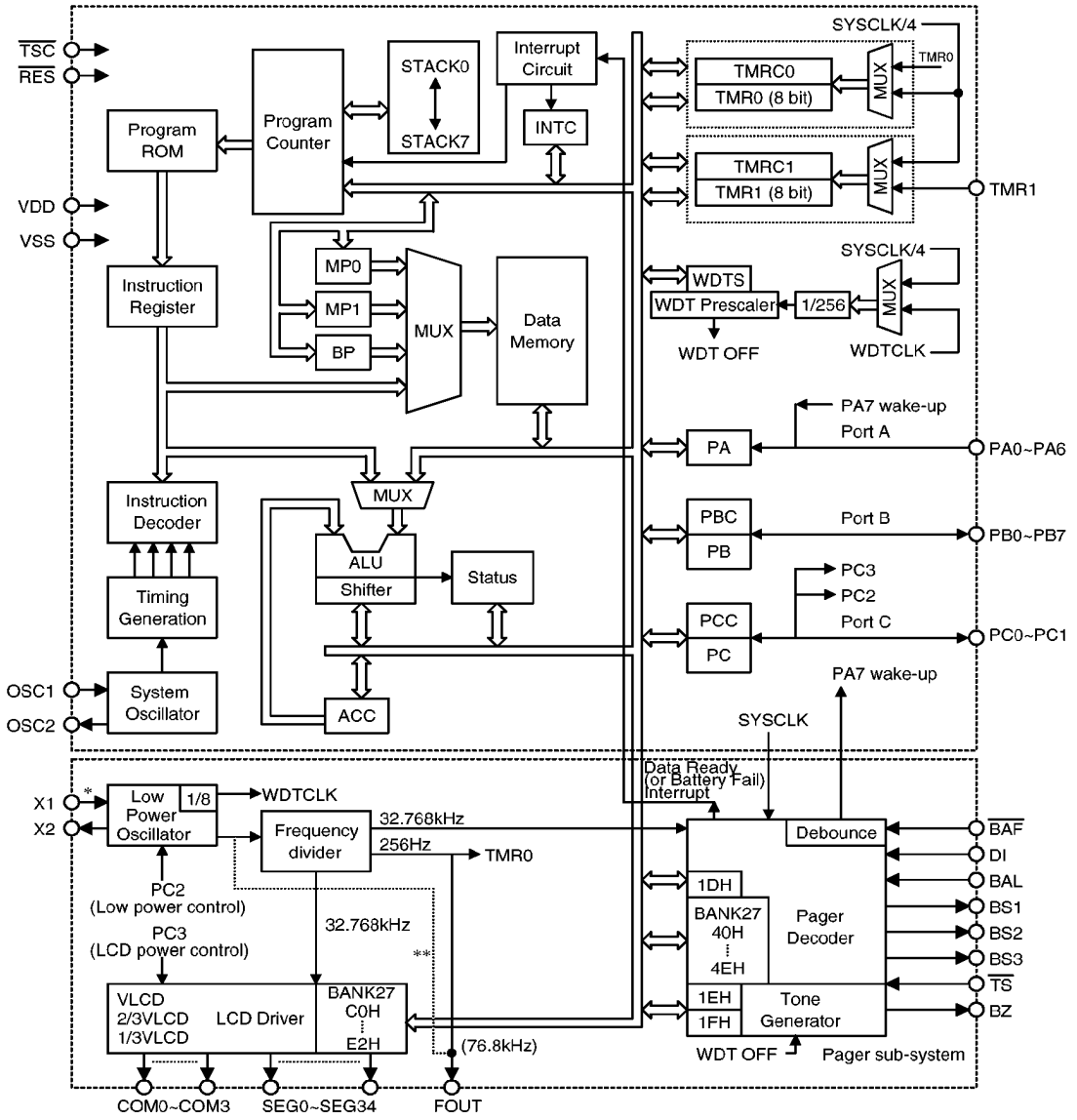
Selection Table

	System OSC	OSC Start-up	f_{OUT}
HT9280	Typ. 256kHz (RC, X'tal)	None (RC OSC)	256Hz
		1024 System Clock (X'tal)	
HT9380	Typ. 256kHz, RC	None	76.8kHz
	Typ. 76.8kHz, X'tal	None	

Pin Assignment



Block Diagram



Note:
*: Assumes X1, X2 uses a 76.8kHz crystal
**: The solid line (FOUT) is for HT9280 while the dotted line (FOUT) is for HT9380

Pin Description

Pin No	Pin Name	I/O	Function
43~49	PA0~PA6	I	7-bit input ports, with pull-high resistors Each bit can be configured as a wake-up input by mask option
54~61	PB0~PB7	I/O	Bidirectional 8-bit input/output ports, pull-high mask option The output structures, whether tri-state or CMOS, are determined by software instructions
62~63	PC0~PC1	O	Bidirectional 2-bit input/output ports, pull-high mask option The output structures, whether tri-state or CMOS, are determined by software instructions
1, 42, 52, 64	VSS		Negative power supply (GND)
76 77	X1 X2	I O	X1 and X2 are connected to an external crystal to form an internal low power oscillator clock
40 41	OSC1 OSC2	I O	OSC1 and OSC2 are connected to an RC network or a crystal (determined by mask option) to form the system clock oscillator. For RC operation, OSC2 is the output terminal of the system clock
53	$\overline{\text{RES}}$	I	Schmitt trigger reset input, active low
68	$\overline{\text{BAF}}$	I	Battery fail interrupt with debounce circuit input
50	TMR1	I	Schmitt trigger input for timer/event counter
2, 39, 51 67	VDD		Positive power supply
65	BZ	O	Buzzer non-inverting BZ output The BZ pin outputs a "high" at buzzer off (by setting the value 00H of 1DH)
3~34 78~80	SEG31~SEG0 SEG34~SEG32	O	LCD driver outputs for LCD panel segments
35~38	COM3~COM0	O	Outputs for LCD panel common connections
66	$\overline{\text{TSC}}$	I	µC test mode input pin, active low with pull-high resistor
75	$\overline{\text{TS}}$	I	Decoder test mode input pin, active low with pull-high resistor
69	BAL	I	Battery low indication input, active high without pull-high resistor
70	DI	I	POCSAG code input serial data (inverting or non-inverting determined by SPF32) CMOS input without pull-high resistor
71	BS1	O	Pager receiver power control enable output, CMOS output
72	BS2	O	RF DC level adjustment pin, CMOS output
73	BS3	O	PLL control pin, CMOS output
74	FOUT	O	Frequency reference output pin. The FOUT output pin produces a 256Hz/76.8kHz signal with a 1/2 duty cycle reference frequency for HT9280/HT9380 respectively

Absolute Maximum Ratings*

Supply Voltage ..... –0.3V to 5.5V
Storage Temperature..... –50°C to 125°C
Input Voltage..... V_{SS}–0.3V to V_{DD}+0.3V
Operating Temperature..... –25°C to 85°C

*Note: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only. Functional operation of this device at these or any other conditions above those indicated in the operational sections of this specification is not implied and exposure to absolute maximum rating conditions for extended periods 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	—	3V application	2.2	3.0	3.5	V
I_{DD}	Operating Current	3V	No load, fsys=256kHz	—	300	—	μA
I_{STB1}	Standby Current 1	3V	No load, System HALT (Watchdog ON)	—	200	—	μA
I_{STB2}	Standby Current 2	3V	No load, System HALT (Watchdog OFF)	—	—	1	μA
V_{IL}	Input Low Voltage for Input Port and I/O Port	3V	—	0	—	1	V
V_{IH}	Input High Voltage for Input Port and I/O Port	3V	—	2.2	—	3	V
V_{IL1}	Input Low Voltage ($\overline{RES}$,TMR1,BAL)	3V	—	0	—	1	V
V_{IH1}	Input High Voltage ($\overline{RES}$,TMR1,BAL)	3V	—	2.2	—	3	V
V_{IL2}	Input Low Voltage ($\overline{BAF}$)	3V	—	0	—	0.9	V
V_{IH2}	Input High Voltage ($\overline{BAF}$)	3V	—	1.3	—	3	V
I_{OL}	I/O Port Sink Current	3V	V_{OL}=0.3V	1.7	3.4	—	mA
I_{OH}	I/O Port Source Current	3V	V_{OH}=2.7V	–1	–1.9	—	mA
I_{OL}	Segment0-34 Output Sink Current	3V	V_{OL}=0.3V	20	44	—	μA
I_{OH}	Segment0-34 Output Source Current	3V	V_{OH}=2.7V	–20	–38	—	μA
I_{OL}	BZ, Sink Current	3V	V_{OL}=0.3V	1	2.5	—	mA
I_{OH}	BZ, Source Current	3V	V_{OH}=2.7V	–1	–2	—	mA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{OL}	PC0~PC1 Sink Current	3V	V_{OL}=0.3V	1.7	3.4	—	mA
I_{OH}	PC0~PC1 Source Current if Pull-High Mask Option	3V	V_{OH}=2.7V	–1	–1.9	—	mA
I_{OL}	BS1, BS2, BS3, FOUT Sink Current	3V	V_{OL}=0.3V	350	—	—	μA
I_{OH}	BS1, BS2, BS3, FOUT Source Current	3V	V_{OH}=2.7V	–0.9	—	—	mA
R_{PH}	Pull-High Resistance of I/O Ports	3V	—	100	200	500	kΩ

A.C. Characteristics

(Ta=25°C)

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS1}	System Clock (RC OSC)	3V	—	76.8	256	1000	kHz
f_{SYS2}	System Clock (Crystal OSC)	3V	—	76.8	256	1000	kHz
f_{SUBSYS}	Pager Subsystem Clock (Crystal OSC)	3V	—	32.768	76.8	153.6	kHz
f_{TIMER}	Timer I/P Frequency (TMR1)	3V	—	0	—	1000	kHz
t_{RES}	External Reset Low Pulse Width	—	—	1	—	—	μs
t_{INT}	Interrupt Pulse Width	—	—	1	—	—	μs
t_{SST}*	System Start-up Time	—	Power-On	—	1024	—	t_{SYS}

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

*: t_{SST} is implemented in HT9280 only

System Architecture

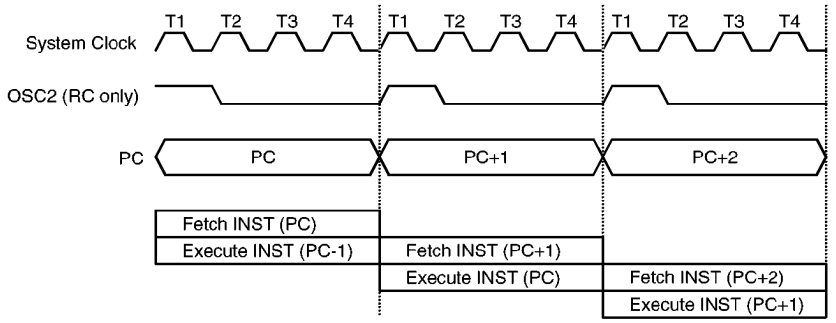
Execution flow

The HT9280/HT9380 system clock can be derived from either a crystal or an RC oscillator. It is internally divided into four non-overlapping clocks denoted by P1, P2, P3, and P4. Each instruction cycle consists of T1 to T4.

Instruction fetching and execution are pipelined in such a way that a fetch takes an instruction cycle while decoding and execution takes the next instruction cycle. The pipelining scheme causes each instruction to effectively execute within a cycle. If an instruction changes the contents of the program counter, two cycles are required to complete the instruction.

Program counter – PC

The program counter (PC) is 13-bit wide and controls the sequence execution. The contents of the PC can specify a maximum of 8192 addresses.



Execution flow

Mode	Program Counter												
	*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
Data ready interrupt and battery fail interrupt	0	0	0	0	0	0	0	0	0	0	1	0	0
Programmable timer interrupt	0	0	0	0	0	0	0	0	0	1	0	0	0
Timer/event Counter interrupt	0	0	0	0	0	0	0	0	0	1	1	0	0
Skip	PC+2												
Loading PCL	*12	*11	*10	*9	*8	@7	@6	@5	@4	@3	@2	@1	@0
Jump, call branch	#12	#11	#10	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
Return from subroutine	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

Program counter

Notes:

*12~*0: Bit of program counter

#12~#0: Bits of instruction code

S12~S0: Bits of stack register

@7~@0: Bits of PCL

The PC value is incremented by one after a program memory word is accessed in order to fetch an instruction code. The PC then points to a memory word with the next instruction code.

The PC loads the address corresponding to each instruction and then manipulates program transfer while executing a jump instruction, conditional skip execution, loading a PCL, a register, a subroutine call, an initial reset, an internal interrupt, an external interrupt, or returning from a subroutine.

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

The low byte of the PC (PCL) is a readable and writable register (06H). Moving data into the PCL performs a short jump. The destination is within 256 locations.

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

Program memory – ROM

The program memory (ROM) is used to store the program instructions. It consists of data, table(s), and interrupt entries, and is organized into 8192×16 bits, which are addressed by the PC and table pointer.

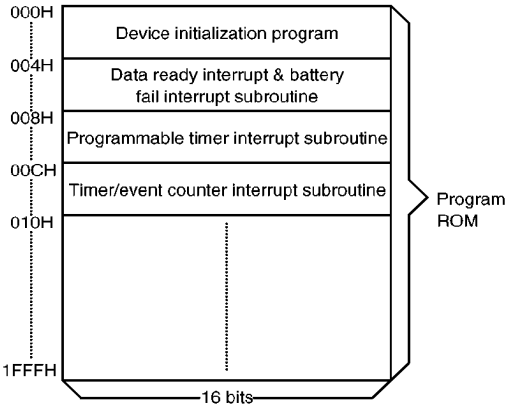
Certain locations in the ROM are reserved for specific usage:

- Location 0000H

  Location 0000H is reserved for program initialization. The program always begins execution at this location each time the chip is reset.

- Location 0004H

  Location 0004H is reserved for the data ready interrupt and battery fail interrupt service programs. If an interrupt results from a pager decoder interrupt request or from a battery fail interrupt request, and the interrupt is enabled, and the stack is not full, the program begins execution at location 0004H. The occurrence of the data ready interrupt or the battery fail interrupt is detected by checking the battery fail interrupt bit (1EH-bit 4, $\overline{BF}$ flag) and the data ready interrupt bit (1EH-bit 7, $\overline{DR}$ flag). The interrupt should be carefully processed if both interrupt bits are active.



Program memory

- Location 0008H

  Location 0008H is reserved for the programmable timer interrupt service program. If an interrupt results from a programmable timer interrupt request (its source is from 256Hz divided by N, where the value of N ranges from 1 to 256), and the interrupt is enabled, and the stack is not full, the program begins execution at location 0008H.

- Location 000CH

  Location 000CH is reserved for the timer/event counter interrupt service program. If a timer interrupt results from a timer/event counter overflow, and the interrupt is enabled, and the stack is not full, the program begins execution at location 000CH.

- Look-up tables XX00H~XXFFH

  The ROM is composed of 32 groups (each group contains 256 continuous words) which can be used as look–up tables. The instructions "TABRDC [m]" (the current table) and "TABRDL [m]" (the last table) transfer the contents of the low-order byte to the specified data memory, and the contents of the high-order byte to TBLH (Table High-order Byte Reg-

ister) (08H). Only the destination of the low-order byte in the table is well-defined, the other bits of the table word are all transferred to the low portion of TBLH. TBLH is read only while the table pointer (TBLP) is a readable/writable register (07H) used to indicate the table location. Before accessing the table, the location should be placed in TBLP. All of the table related instructions require 2 cycles to complete the operation. This feature is efficient only for the movement of data blocks, which may function as look-up tables or as a normal program memory depending upon the requirements.

Stack register – STACK

The stack register is a special memory port of the used to save the contents of the PC. It is divided into 8 levels. The stack register is neither part of the data nor part of the program, and is neither readable nor writable. The activated level of the stack register is indexed by the stack pointer (SP), and also cannot be read or written to. At the commencement of a subroutine call or an interrupt acknowledge, the contents of the PC is pushed onto the stack. At the end of the subroutine or the interrupt routine, as signaled by a return instruction (RET or RETI), the contents of the PC 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 occurs, the interrupt request flag is recorded but acknowledging is inhibited until the value of the SP is decremented (by RET or RETI), allowing that interrupt to be serviced. As this feature can prevent a stack overflow, the use of the structure becomes much easier. In a similar case, if the stack is full, and a "CALL" is subsequently executed, a stack overflow occurs and the first entry is lost (only the most recent eight return addresses are stored).

Data memory – RAM

The data memory (RAM) is designed in three banks, i.e., bank 0, bank 1, and bank 27, and comprised of four functional groups namely, special function registers (of 22×8 bits; 1×4 bit; 1×2 bit in bank 0), data memory (of 320×8 bits; 128×8 in bank 0; 192×8 in bank 1), LCD display mapping memory (of 35×4 bits), and decoder configuration RAM mapping memory (of 15×8 bits). Most of the these groups are readable/writable but some are read only.

Of the four functional groups, the special function registers of bank 0 consist of an indirect addressing registers (IAR0;00H, IAR1;02H), memory pointer registers (MP0;01H, MP1;03H), a memory bank pointer register (BP;04H), an accumulator (ACC;05H), a program counter low byte register (PCL;06H), a table pointer (TBLP;07H), a table high-order part register (TBLH;08H), a watchdog timer option setting register (WDTS;09H), a status register (STATUS;0AH), an interrupt control register (INTC;0BH), a programmable timer counter (TMR0;0DH), a programmable timer counter control register (TMRC0;0EH), a timer/event counter (TMR1;10H), a timer/event counter control register (TMRC1;11H), an input port, two I/O ports (PA;12H, PB;14H, PC;16H), two I/O control register (PBC;15H, PCC;17H), a tone control register (1DH), a pager control register (1EH), and a pager data register (1FH). The special function registers are located from 00H to 1FH whereas the 32 global data regis-

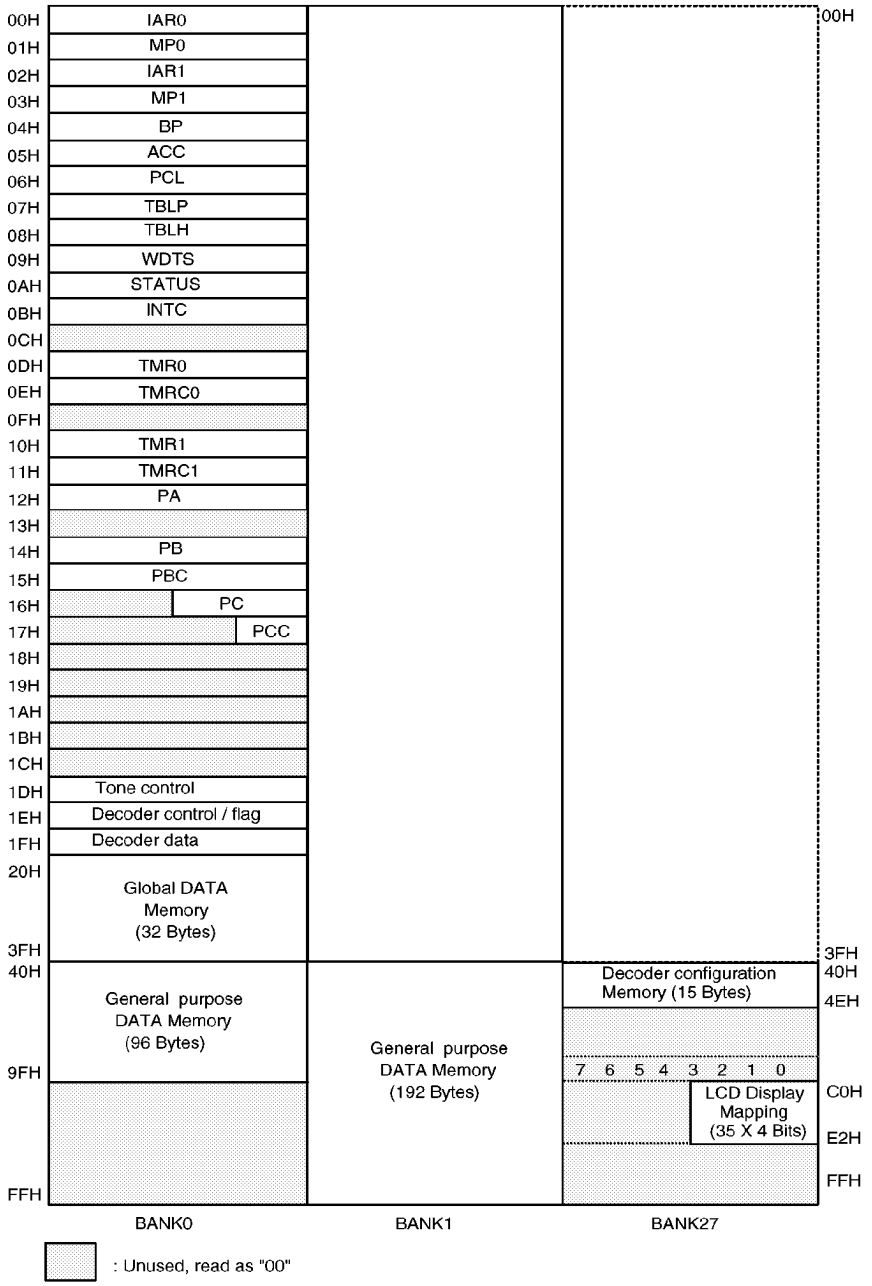
Instruction(s)	Table Location												
	*12	*11	*10	*9	*8	*7	*6	*5	*4	*3	*2	*1	*0
TABRDC [m]	P12	P11	P10	P9	P8	@7	@6	@5	@4	@3	@2	@1	@0
TABRDL [m]	1	1	1	1	1	@7	@6	@5	@4	@3	@2	@1	@0

Notes:

*12~*0: Bits of table location

P12~P8: Bits of current program counter

@7~@0: Bits of table pointer



RAM mapping

ters are from 20H to 3FH, where each bank points to the same location. The other spaces namely, 0CH, 0FH, 13H, the high nibble of 16H, 17H, and 18H~1CH, are all reserved for future expansion usage; reading these locations will get a "00H" value.

On the other hand, the general purpose data memory, divided into three banks (bank 0, bank 1, and bank 27), is used for data, control information, and LCD display control under instruction commands. The banks in the RAM are all addressed from 40H to FFH, and are selected by setting the value ("00H": bank 0; "01H": bank 1; "1BH": bank 27) of the bank pointer (BP;04H). The bank 27 memory is used for LCD display mapping and the decoder configuration RAM mapping. The spaces from 4FH to BFH and from E3H to FFH, and the high nibble part from C0H to E2H in bank 27 are all reserved for future expansion usage; reading these locations will derive "00H".

The special registers, global data registers and general data memory can directly perform arithmetic, logic, increment, decrement, and rotate operations. Each bit in the RAM can be set and reset by "SET [m].i" and "CLR [m].i", and can also be indirectly accessible through the memory pointer registers (MP0;01H, MP1;03H).

Of the special addresses, 1DH and 1FH cannot directly do all these operations, because they are not read and write accessible addresses. 1DH is a write-only address, 1FH a read-only address but these two addresses, 1DH and 1FH can only perform operations by using the "MOV" instruction.

Indirect addressing register

IARx (IAR0;00H, IAR1;02H) are indirect address registers that are not physically implemented. Any read/write operation of the IARx accesses the data memory pointed to by MPx (MP0;01H, MP1;03H). Reading the indirect addressing register itself will indirectly derive 00H, while writing to the indirect addressing register indirectly will have no effect. (IAR0, MP0) is indirectly addressable in bank 0, but (IAR1, MP1) is available for all banks.

Accumulator – ACC

The accumulator (ACC) relates to the ALU operations. It is also mapped to location 05H of the data memory and is capable of carrying out immediate data. Data movement between these two data memories has to pass through the ACC.

Arithmetic and logic unit – ALU

This circuit performs 8-bit arithmetic and logic operations, and provides the following functions:

- Arithmetic operation (ADD, ADC, SUB, SBC, DAA)
- Logic operation (AND, OR, XOR, CPL)
- Rotation (RL, RR, RLC, RRC)
- Increment & decrement (INC, DEC)
- Branch decision (SZ, SNZ, SIZ, SDZ, etc.)

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

Status register – STATUS

The status register (0AH) is 8-bit wide and it contains a zero flag (Z), a carry flag (C), an auxiliary carry flag (AC), an overflow flag (OV), a power down flag (PD), and a WDT time-out flag (TO). The status register not only records the status information, but also controls the operation sequence.

The status register, like most other registers, can be altered by instructions except for the TO and PD flags. Any data written into the status register will not change TO or PD. It should be noted that operations related to the status register may derive different results from those intended. For example, clearing the status register CLR [0AH] has no effect on the TO and PD flags, and the value of the zero flag is also "1", i.e., UU0100 is the data in the register, where the value of U is an unchanged value.

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

On entering an interrupt sequence or executing a subroutine call, the status register will not be automatically pushed onto the stack. If the contents of the status is important, and if the subroutine may corrupt the status register, the programmer should take precautions to save it properly.

Labels	Bits	Function
C	0	C is set if the operation results in a carry out in addition or if a borrow does not take place in subtraction; otherwise C is cleared. Also C is affected by a rotation through carry instructions.
AC	1	AC is set if the operation results in a carry out of the low nibbles in addition or if a borrow from the high nibble into the low nibble doesn't take place in subtraction; otherwise AC is cleared.
Z	2	Z is set if the result of an arithmetic or a logic operation is zero; otherwise Z is cleared.
OV	3	OV is set if the operation results in a carry into the high-order bit but not a carry out of the high-order bit, or vice versa; otherwise OV is cleared.
PD	4	PD is cleared during power up, and set by a "HALT" instruction.
TO	5	TO is cleared during power up or by a "CLR WDT" instruction and a "HALT" instruction. TO is set by a current timer time-out.
–	6	Undefined, read as "0"
–	7	Undefined, read as "0"

STATUS register

Interrupts

The HT9280/HT9380 provide an internal programmable timer interrupt, an internal data ready interrupt, a timer/event counter interrupt, and a battery fail interrupt. The internal data ready interrupt and the battery fail interrupt employ the same jump location (04H). The interrupt control register (INTC;0BH) contains interrupt control bits to set not only the enable/disable status but also the interrupt request flags.

Once an interrupt subroutine is serviced, the other interrupts will all be blocked (by clearing the EMI bit). This scheme may prevent any further interrupt nesting. Other interrupt requests may occur during this interval, but only the interrupt request flag is recorded. If a certain interrupt requires servicing within the service routine, the EMI bit and the corresponding bit of the INTC register may be set to permit interrupt nesting. When the stack is full, the interrupt request will not be acknowledged even if the related interrupt is enabled, until the SP is decremented. If immediate service is desired, the stack should be prevented from becoming full.

All of these interrupts can support the wake-up function. As an interrupt is serviced, a control transfer occurs by pushing the contents of the PC onto the stack, followed by a branch to a subroutine at the specified location in the program memory. Only the contents of the PC is pushed onto the stack. If the contents of the register or of the status register (STATUS) is altered by the interrupt service program which corrupts the desired control sequence, the contents should saved in advance.

The data ready interrupt and battery fail interrupt share the same subroutine call location 04H. Checking the battery fail interrupt bit ($\overline{BF}$;bit 4 of 1EH) and the data ready interrupt bit ($\overline{DR}$; bit 7 of 1EH) can determine which kind of interrupt has occurred. The value of 1EH-bit 7 $\overline{DR}$ is cleared "0" by the decoder data ready interrupt signal, and is set to "1" when the µC sets this bit high. Both interrupt bits are active low.

The data ready interrupt is generated by the pager decoder after a valid call is received, and is initialized by setting the data ready interrupt request flag (EIF; bit 4 of INTC) and the data

ready interrupt bit ($\overline{DR}$; bit 7 of 1EH). Once the data ready interrupt is triggered, the stack is not full, and the EMI bit is set, a subroutine call to location 04H will occur. The related interrupt request flag (EIF) will, however, be reset, and the EMI bit cleared to disable further interrupts. This interrupt should be processed carefully if the battery fail interrupt is activated as well.

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

The programmable timer interrupt is automatically triggered at a rate of 256Hz/N (where the value of N ranges from 1 to 256), and then the interrupt request flag (T0F; bit 5 of INTC) is set. When the timer interrupt is enabled, the stack is not full, and the programmable timer interrupt is activated, a subroutine call to location 08H will occur. Then, the related interrupt request flag (T0F) will be reset, and the EMI bit cleared to disable other interrupts.

The timer/event counter interrupt is initialized by setting the timer/event counter interrupt request flag (T1F; bit 6 of INTC), which is normally caused by a timer overflow. When the interrupt is enabled, the stack is not full, and the T1F bit is set, a subroutine call to location 0CH will occur. The related interrupt request flag (T1F) will be reset, and the EMI bit cleared to disable further interrupts.

During the execution of an interrupt subroutine, other interrupt acknowledgments are all held until the "RETI" instruction is executed, or the EMI bit and the related interrupt control bit are both set to 1 (if the stack is not full). To return from the interrupt subroutine, a "RET" or "RETI" instruction may be invoked. RETI will set the EMI bit to enable an interrupt service, but RET will not.

The interrupts are serviced between the rising edges of the two adjacent T2 clocks. In case of simultaneous requests, the following table shows the priority that is applied. These can be masked by resetting the EMI bit.

Register	Bit No.	Label	Function
INTC (0BH)	0	EMI	Controls the master (global) interrupt (1=enabled; 0=disabled)
	1	EEI	Controls the data ready and battery fail interrupts (1=enabled; 0=disabled)
	2	ET0I	Controls the programmable timer interrupt (1=enabled; 0=disabled)
	3	ET1I	Controls the timer/event counter interrupt (1=enabled, 0=disabled)
	4	EIF	Internal data ready and battery fail interrupt request flag (1=active; 0=inactive)
	5	T0F	Internal programmable timer interrupt request flag (1=active; 0=inactive)
	6	T1F	Timer/event counter request flag (1=active; 0=inactive)
	7	--	Unused bit, read as "0"

INTC register

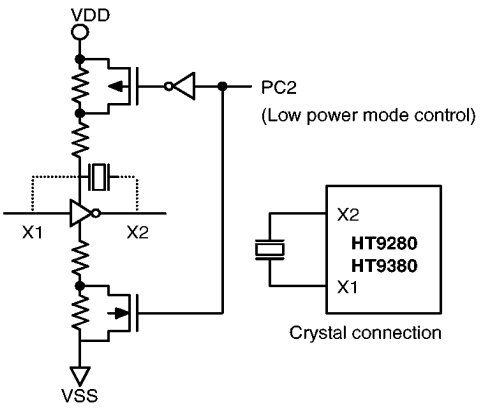
NO.	Interrupt Source	Priority	Vector
a	Data ready interrupt and battery fail interrupt	1	04H
c	Programmable timer interrupt	2	08H
d	Timer/event counter overflow	3	0CH

The register INTC which is located at 0BH in the data memory is made up of the programmable timer interrupt request flag (T0F), timer/event counter interrupt request flag (T1F), data ready interrupt and battery fail interrupt request flag (EIF), enable timer/event counter bit (ET1I), enable data ready interrupt bit (EEI), and enable programmable timer interrupt bit (ET0I). The EEI, ET0I, ET1I, and EMI bits are all used to control the enable/disable status of the interrupts, preventing the requested interrupt from being serviced. Once the interrupt request flags (T0F, T1F, and EIF) are set, they will remain in the INTC register until the interrupts are serviced or cleared by a software instruction.

A "CALL subroutine" in the interrupt subroutine should not be used. This is because interrupts often occur in an unpredictable manner or need to be immediately serviced in some applications. At this time, if only one stack is left, and enabling the interrupt is not well controlled, the operation of a "CALL subroutine" in the interrupt service routine is quite likely to upset the original control sequence.

Oscillator configuration

The system core and the pager subsystem of the HT9280/HT9380 are clocked by different oscillators. The system oscillator can be either a crystal type or an RC type. The subsystem low power oscillator, on the other hand, is a crystal type which is designed with the power on start-up function to reduce the stabilization time of the oscillator. This start-up function is enabled by PC2 which is initially set high at power on reset, and should be cleared so as to enable the low-power oscillator function. The oscillator configuration is running in the low power mode.

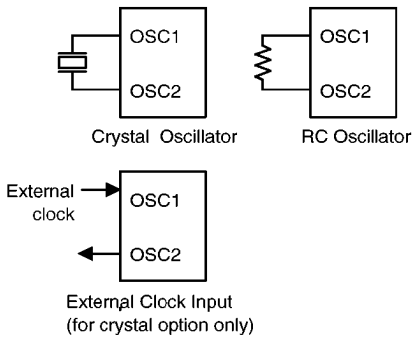


Low power oscillator

The system oscillator can be configured as either an RC or a crystal type of oscillator, determined by mask option. No matter what kind of oscillator type is selected, the signal provides a system clock. The system clock may also be externally connected. The HALT mode stops the system oscillator and ignores external signals so as to conserve power.

If the system oscillator is an RC type oscillator, an external resistor between OSC1 and OSC2 is required. The system clock is available on OSC2, which can be used to synchronize external logic. An RC oscillator provides the most cost-effective solution. The frequency of oscillation may vary with power, temperature, and the chip itself due to process variations. The RC oscillator is, therefore, not suitable for timing sensitive operations where an accurate oscillator frequency is desired.

On the other hand, if a crystal type oscillator is used, a crystal across OSC1 and OSC2 is required to provide the feedback and phase shift for oscillation, and no other external components are required. A ceramic resonator can replace the crystal connected between OSC1 and OSC2 to derive a frequency reference. In this case, two external capacitors at OSC1 and OSC2 are required.



System clock oscillator

An external clock can also be applied to OSC1. In this application, the mask option of the crystal type oscillator should be selected, and OSC2 kept open.

The low power crystal oscillator is designed for the pager subsystem and is used to clock the frequency divider, pager decoder, and LCD driver. When the system enters the power down mode the crystal oscillator for the pager subsystem keeps running.

Watchdog timer – WDT

The clock source of the watchdog timer (WDT) is implemented by a subsystem clock (WDTCLK from the pager subsystem which remains running during a system halt) or by an instruction clock (the system clock divided by 4), that is decided by mask option. The value of WDTCLK can be set as 153.6kHz/8, 76.8kHz/8, or 32.768kHz/8, depending upon the different crystal type. The WDT is designed to avoid software malfunctions or the program sequence from jumping to an unknown location with unpredictable results. It can be disabled by mask option. If the WDT is disabled, all the executions related to the WDT produce no effect.

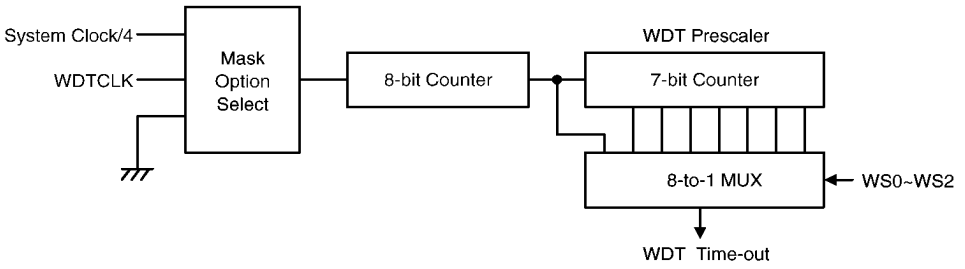
If the subsystem clock is selected, it is first divided by 256 (8 stages) to get the nominal time-out period. Longer time-outs can be realized by invoking the WDT prescaler. Writing data to WS2, WS1, and WS0 (bits 2,1,0 of the WDTS) can yield different time-out periods. If the values of WS2, WS1, and WS0 are all equal to 1, the division ratio is up to 1:128.

On the other hand, if the instruction clock is applied, the WDT operates in the same manner as the case when the subsystem clock is chosen, except that in the HALT state the WDT stops counting and lose its protection purpose. In this situation, WDT logic can be restarted by external logic. The high nibble and bit 3 of the WDTS is reserved for user defined flags, which can be used to indicate some specified status.

The overflow of the WDT under normal operation not only initializes the "chip reset", but also sets the status bit "TO". An overflow in the HALT mode initializes a "warm reset" only

Division Ratio Option				Crystal Type and Time-Out Period		
WS2	WS1	WS0	Division Ratio	153.6kHz	76.8kHz	32.768kHz
0	0	0	1:1	13.3ms	26.7ms	62.5ms
0	0	1	1:2	26.7ms	53.3ms	125ms
0	1	0	1:4	53.3ms	106.7ms	250ms
0	1	1	1:8	106.7ms	213.3ms	500ms
1	0	0	1:16	213.3ms	426.7ms	1000ms
1	0	1	1:32	426.7ms	853.3ms	2000ms
1	1	0	1:64	853.3ms	1706.7ms	4000ms
1	1	1	1:128	1706.7ms	3413.3ms	8000ms

WDTs register



Watchdog timer

when the PC and SP are reset to zero. To clear the contents of the WDT (including the WDT prescaler), there are three methods to be adopted namely, external reset (a low level to $\overline{RES}$), software instruction(s), and a "HALT" instruction. There are two types of software instructions, "CLR WDT" and "CLR WDT1" / "CLR WDT2". But only one of these two types of instructions can be active at a time 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 clears the WDT. In the case that "CLR WDT1" and "CLR WDT2" are chosen (i.e., CLRWDR times equal two), these two instructions should be executed to clear the WDT; otherwise, the WDT may reset the chip due to a time-out.

Power down operation – HALT

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

The system turns off. The low power oscillator, tone generator, LCD driver, pager decoder, and WDT oscillator all keep running (if the WDT oscillator is selected).

The contents of the on–chip RAM and of the registers remain unchanged.

The WDT and the WDT prescaler are cleared and counted again (if the WDT clock is from the WDT oscillator).

All the I/O ports remain in their original status.

The PD flag is set but the TO flag is cleared.

The system can quit the HALT mode by an external reset, an interrupt, an external falling edge signal on port A, or a WDT overflow. An external reset leads to device initialization and the WDT overflow performs a "warm reset". After the TO and PD flags are examined, the reason for the chip reset is determined. The PD flag that is cleared on power-up is set after the "HALT" instruction is executed. The TO flag is set when the WDT time-out occurs, which causes a wake-up that resets only the PC and SP, and leaves the others in their original status.

The port A wake-up and interrupt methods can be considered as a continuation of normal execution. Every bit in port A can be independently selected to wake up the device by mask option. Awakening from an I/O port stimulation, the program resumes execution of the next instruction. However, if the program awakens from an interrupt, two sequences may occur. The program will resume execution of the next instruction if the related interrupt(s) is (are) disabled or the interrupt(s) is (are) enabled but the stack is full. A regular interrupt response, on the other hand, may take place if the interrupt is enabled and the stack is not full.

If the wake-up event(s) occur(s) and the wake-up results from an interrupt acknowledgment, the actual interrupt subroutine execution is delayed by one or more cycles. On the other hand, if the wake-up brings about the next instruction execution, the actual interrupt subroutine is executed immediately after the dummy period is completed.

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

Reset

There are five ways in which a reset can occur:

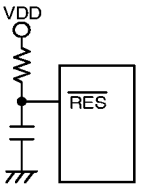
- Power on reset (POR)
- $\overline{RES}$ reset during normal operation
- $\overline{RES}$ reset during HALT
- WDT time-out reset during normal operation
- WDT time-out reset during HALT

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

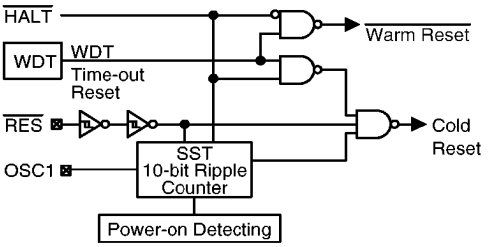
TO	PD	RESET Conditions
0	0	Power on reset
u	u	$\overline{RES}$ reset during normal operation
0	1	$\overline{RES}$ wake-up HALT
1	u	WDT time-out during normal operation
1	1	WDT wake-up HALT

Note:"u" means "unchanged"

To guarantee that the system oscillator has started and stabilized, the SST (System Start-up Timer) provides an extra-delay to delay 1024 system clock pulse when the system power is up or the system awakes from the HALT state. The above SST function is only available on crystal mode of HT9280 application. But when the reset comes from the normal operation, the SST delay is disabled.



Reset circuit



Note: The SST delay is only implemented on HT9280 (crystal mode).

Reset configuration

If the RC oscillator is selected, reset duration comes from $\overline{RES}$ only, the SST delay is not needed. If the Crystal oscillator is used, the extra SST delay is added during the reset period only on HT9280 application.

The functional unit chip reset status is shown in the following the table.

PC	0000H
Interrupt	Disabled
Prescaler	Cleared
WDT	Cleared. After master reset, WDT starts counting.
Programmable timer Counter	Off
Timer/event Counter	Off
Programmable Tone Generator	Off
Pager Decoder	Off
Input/output Ports	input mode
SP	Points to the top of the stack

Programmable timer counter and timer/event counter

The programmable timer counter (TMR0) and timer/event counter (TMR1) are constructed using the same structure. Both counters contain an 8–bit programmable count-up counter, whose clocks may come from an external source or from the system clock divided by 4.

If the internal instruction clock is selected, only one reference time-base is provided. The external clock input allows the user to count external events, measure time intervals or pulse widths, or generate an accurate time base. The clock of the programmable timer counter should come from the external clock of the 256Hz Real Time Clock (RTC).

There are two sets of registers related to the programmable timer counter and to the timer/event counter, namely TMR0 (0DH) and TMRC0 (0EH) and TMR1 (10H) and TMRC1 (11H). There are also two physical registers mapped to the TMR0 and TMR1 locations: writing TMR0 and TMR1 puts the starting value in the programmable timer counter and in the timer/event counter preload registers, while reading them gets the contents of the two counters. TMRC0 and TMRC1 are control registers used to define some timer options.

The TM0 and TM1 bits define the operation mode. The event count mode is used to count external events, which means that the clock source may come from either a 256Hz generator (for TMR0) or an external pin (for TMR1). The timer mode functions as a normal timer, with the clock source coming from the instruction clock or from the outputs of the TMR1 prescaler (TMR0 cannot be used in this mode). The pulse width measurement mode can be used to count the high or low level duration of the external signal TMR1, TMR0 is disabled in this mode. The counting is based on the system clock.

In the event count or timer mode, once the programmable timer counter or timer/event counter starts counting, it will count from the current contents in the counter to FFH. Once an overflow occurs, the counter is reloaded from its counter preload register and generates an interrupt request flag (T0F; bit 5 of INTC and T1F; bit 6 of INTC for programmable timer counter and timer/event counter, respectively).

On the other hand, in the pulse width measurement mode with the TON bit equal to one, when the TMR1 receives a transient from low to high (or high to low depending upon the TE bit) it will start counting until the TMR1 returns to the original level and resets the TON as well.

Labels (TMRC0 and TMRC1)	Bits	Function
—	0~2	Unused bits, read as “0”
TE	3	To define the TMR0 and TMR1 active edge of programmable timer counter and 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 operation mode 01=Event count mode (external clock) 10=Timer mode (internal clock) 11=Pulse width measurement mode 00=Unused

TMRC register

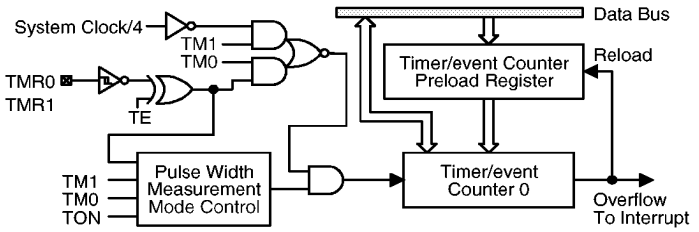
The states of the registers are summarized below.

Register	Power-on reset (POR)	WDT time-out (normal operation)	$\overline{RES}$ reset (normal operation)	$\overline{RES}$ reset (HALT)	WDT time-out (HALT)*
TMR0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMRC0	00-0 1---	00-0 1---	00-0 1---	00-0 1---	uu-u u--
TMR1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMRC1	00-0 1---	00-0 1---	00-0 1---	00-0 1---	uu-u u--
PC	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
INTC	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
WDTS	0000 0111	0000 0111	0000 0111	0000 0111	uuuu uuuu
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	---- uuuu
PCC	---- 1111	---- 1111	---- 1111	---- 1111	---- uuuu

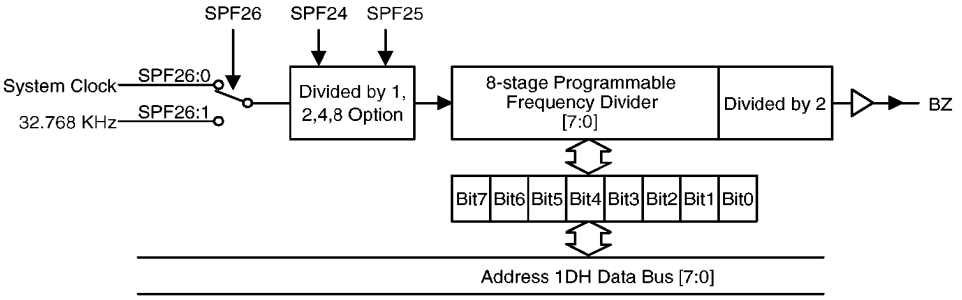
Note: "*" means "warm reset"
"u" means "unchanged"
"x" means "unknown"

The measured result will remain in the timer/event counter even when the activated transient occurs again. In other words, only one cycle measurement can be made until the TON is set. The cycle measurement will re-function as long as further transient pulses are received. Note that, in this operation mode, the timer/event counter starts counting not according to the logic level but to the transient edges. In the case of counting overflows, the counter is re-loaded from its counter preload register and issues an interrupt request, similar to the other two modes.

To enable the counting operation, the value of the timer on bit (TON; bit 4 of TMRC0 and TMRC1) is "1". In the pulse width measurement mode, the TON is automatically cleared after the measurement cycle is completed. In the other two modes, the event count or timer mode, the TON can be reset only by instructions. The overflow of the programmable timer counter and of the timer/event counter can be configured as one of the wake-up sources. No matter what type of operation mode is chosen, writing a 0 to ET0I and ET1I disables the interrupt service of the programmable timer counter and the timer/event counter, respectively.



Timer/event counter



Programmable tone generator

In the case of the programmable timer counter and a timer/event counter OFF condition, writing data to their preload registers also reloads that data to their counters. But if the programmable timer counter or the timer/event counter is turned on, data written to the counter is kept only in its preload register, and the counter still goes on operating until an overflow occurs.

After the counter (reading TMR0 or TMR1) is read, the clock is blocked to avoid errors. The programmer should take clock blocking into consideration, since this may result in timing counting errors.

Programmable tone generator

The programmable tone generator is implemented in the HT9280/HT9380. The programmable tone generator contains an 8-stage programmable frequency divider (mapping to the 1DH address of the µC), a 4-stage programmable frequency prescaler (set by SPF24 and SPF25), and a frequency source selector (set by SPF26). When 1DH=00H, the tone generator is disabled and BZ outputs high. But when 1DH is of any value greater than zero the generator is enabled. The value of the frequency divider, ranging from 2~256, is always greater than the assigned value by 1. The output of the 8-stage divider is divided by 2 to generate an output of 1:1 duty cycle on BZ. The 4-stage programmable frequency prescaler is shown below.

SPF24	SPF25	Prescaler Divider Factor
0	0	1
0	1	2
1	0	4
1	1	8

The above setting of the prescaler divider factor is designed for applications on melodies or sound effects.

The frequency source selector is set by SPF26. When SPF26=0, the value of the frequency source selector is the system clock. On the other hand, when SPF26=1, the value of the selector turns out to be 32.768kHz. For instance, if the desired output of BZ is 2.73kHz, the frequency source is 32.768kHz, the values of SPF24 and SPF25 are both set to 0, and the value of the programmable frequency divider is set to 5.

Input/output ports

There are 7 input lines, and 10 input/output lines in the HT9280/HT9380, which are labeled as PA, and PB; PC (PC0, PC1). These are mapped to [12H], and [14H]; [16H] of the data memory, respectively. Port A is an input port only while Port B and Port C (PC0 and PC1) are bidirectional I/O ports. For input operation, the ports A, B, and C are non-latched, i.e., the inputs have to be ready at the T2 rising edge of the instruction "MOV A, [m]" (m=12H, 14H, 16H). For output operation, data is latched and then remains unchanged until the output latch is rewritten.

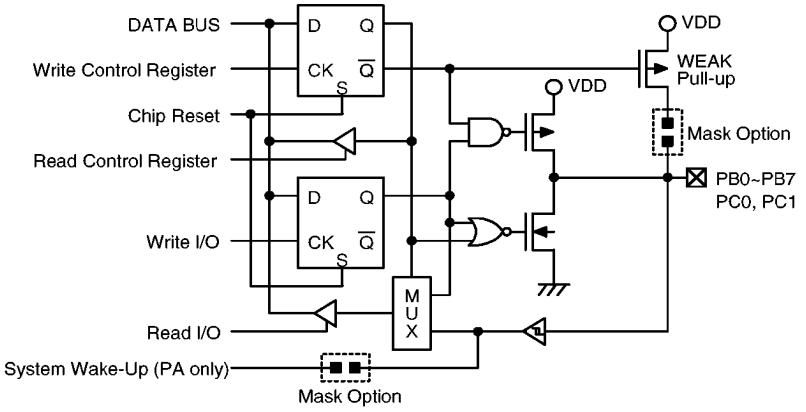
The PB and PC (PC0, PC1) I/O lines have their own control registers (PBC, PCC) to control the input/output configuration. These control registers, tri-state (control register=1) or CMOS (control register=0) with pull-high (option) structures can be reconfigured dynamically (i.e., on-the-fly) by the software control. To function as an input, the corresponding I/O latch and related bit of the control register should be written "1" to avoid external logic violation. These control registers are mapped to location 15H, and 17H (bit 0 and bit 1 of 17H).

After a chip reset, these input/output lines stay at high levels or floating (by mask option). They are defined as input types by writing "1" to the control registers and as output types by writing "0" to the control registers. Each bit of these input/output latches can be set or cleared by "SET [m].i" and "CLR [m].i" (m=14H only) instructions.

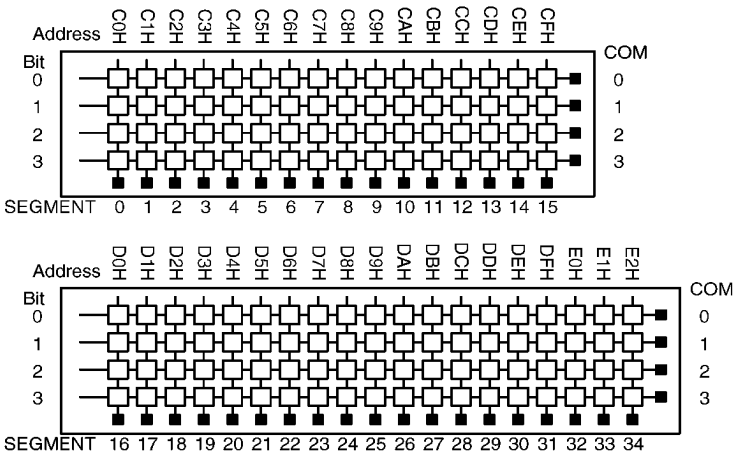
Some instructions first input data and then follow the output operations. For example, "SET [m].i", "CLR [m].i", "CPL [m]", "CPLA [m]" read the entire port states into the CPU, execute the defined operation (bit-operation), and then write the results back to the latches or the accumulator.

Each line of port A is capable of waking up the device (when a falling edge occurs) and is determined by mask option. The highest 4 bits of port C are not physically implemented. Reading them gets a "0", but writing to them leads to no operation.

Bit 7 of port A connects a battery fail interrupt and a wake-up function. Bit 7 of port A wakes up the μC each time a battery is changed. Bit 2 of port C is used for internal subsystem oscillator



Input/output ports



Display memory

low-power function control (1: non-active; 0: active). The value of bit 2 of port C is set as "1" at an initial power on. Bit 3 of port C is used for LCD power control (1:LCD turn-on; 0:LCD turn-off). The value of bit 3 of port C is also set as "1" at the initial power on.

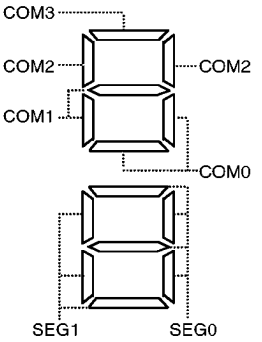
LCD display

The LCD display memory is embedded in the data memory (mapped to the addresses C0H~E2H of bank 27). It can be read and written to as a normal data memory. The figure above illustrates the mapping between the display memory and the LCD pattern.

To turn the display on/off, the programmer writes 1 or 0 to the corresponding bit of the display memory. The LCD display module can be of any form as long as the number of the common doesn't exceed 4 and the number of the segment is not over 35.

The entire number of the LCD driver output is 35×4. The LCD driver can directly drive an LCD of 1/4 duty cycle and 1/3 bias. All of the LCD segments are random at the initial clear mode.

The frequency of the LCD driving clock is fixed at about 256Hz, and cannot be changed. It is set by HOLTEK according to the application.

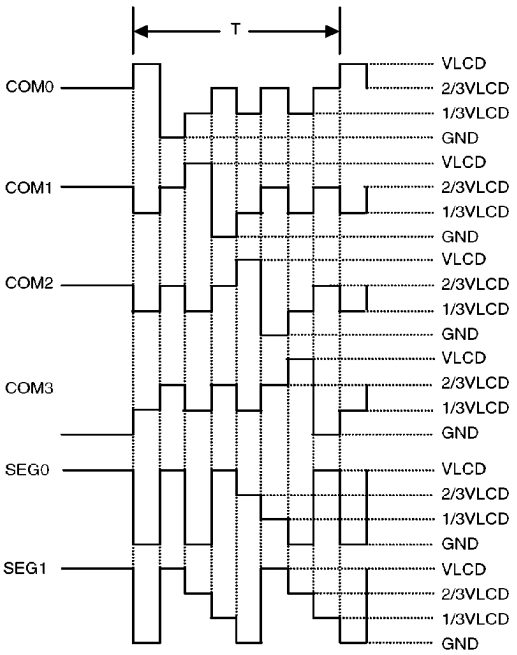


The following is an example of an 8-segment digit display, which shows a waveform of "5".

Pager decoder

The pager decoder is a POCSAG code pager decoder at 512, 1200, or 2400 bps data rate, compatible with CCIR radio paging Code No.1 (POCSAG Code). The decoder supports four user addresses and two independently programmable user frames.

The operation of the decoder is controlled by a pager control address (1EH) in conjunction with a pager data address (1FH). Upon receipt of a valid call the data ready interrupt is generated.



LCD timing

- The POCSAG Paging Code

  The CCIR Radio paging Code No.1 (POCSAG Code) is constructed according to the following rules:

Each transmission starts with a preamble, which is a sequence of at least 576 continually alternating bits (101010....). The preamble precedes a number of batches. The transmission stops right after the last batch is transmitted.

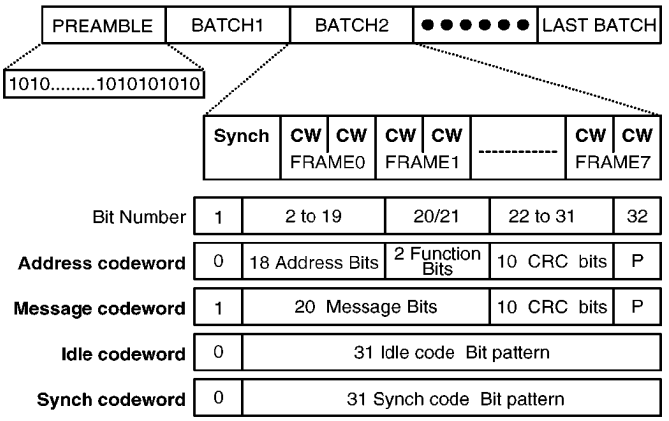
A batch consists of a synchronization codeword followed by 8 frames (numbered from 0 to 7). Each frame contains 2 codewords, and only complete batches can be transmitted.

A codeword is either an address or a message codeword. Idle codewords are transmitted to fill in empty batches or to separate messages.

An address codeword is coded as shown below. Of the 21 bits of user addresses, 18 bits are coded in the codeword itself (bits 2 to 19), which is protected against transmission errors by a number of CRC checkbits (bits 22 to 31). Bit 32 is an overall even-parity bit. The missing three bits of the 21 bits of the user address are coded in the number of the frame, in which the address codeword is transmitted. The two function bits (bits 20 and 21) allow distinction of four different calls to one user address.

An idle codeword is a valid address codeword, which cannot be allocated to the pager.

There is a total of 20 bits of caller information to be put into a message codeword (bits 2 to 19), which is protected by the CRC checkbits (bits 22 to 31).



POCSAG code structure

- Decoding of the POCSAG Data Stream

  The POCSAG coded input data received from the RF module is first noise filtered by an internal digital filter in the decoder. From the filtered data, a sampling clock synchronous to the data rate is derived. The decoder supports 512, 1200, and 2400 bits per second data rate, which in turn results in their corresponding sampling clock frequency.

  Upon detection of a valid call, the decoder performs several operations (refer to the following section of the Message Data Transfer).

  Call termination is normally deemed when a valid idle or address codeword is received.

- Erroneous Codewords

  Upon receipt of erroneous uncorrectable codewords, call termination occurs according to the conditions given below:

SPF12	SPF13	Call Termination Event
0	X	Any two consecutive codewords or the codeword directly following the address codeword is in error
1	0	Any single codeword is in error
1	1	Any two consecutive codewords are in error

Decoder interface

The HT9280/HT9380 has two interfaces available. One is the pager control address (1EH), which controls the operation and configuration of the decoder. The other is the pager data address (1FH), which receives the message data of calls in the parallel mode.

- Decoder Control Address

  The decoder control address (1EH) contains a data ready flag ($\overline{DR}$), a battery low flag ($\overline{BL}$), an out of range flag ($\overline{OR}$), a battery fail flag ($\overline{BF}$), a decoder standby flag (STB), a decoder software reset ($\overline{RES}$), and a decoder on/off control bit ($\overline{ON}$). It records the status information and controls the operation of the decoder as well.

  Any data written to the decoder control address cannot change the $\overline{OR}$, $\overline{BF}$, and STB flags.

  If the status of the battery fail ($\overline{BF}$) changes from "1" to "0", the following conditions occur.

- The pager controller generates an interrupt if the value of the data ready interrupt flag is "1".

- The pager controller does not generate an interrupt and no data is transmitted if the value of the data ready interrupt flag is "0".

  On the other hand, if the status of the battery fail ($\overline{BF}$) changes from "0" to "1", the internal node PA7 of the pager controller will supply a wake-up function.

  After the decoder asserts the data transfer request, the data ready interrupt is generated and the $\overline{DR}$ bit (bit 7 of 1EH) is cleared low; then the data ready interrupt subroutine runs to process the call data and resets the $\overline{DR}$ bit high.

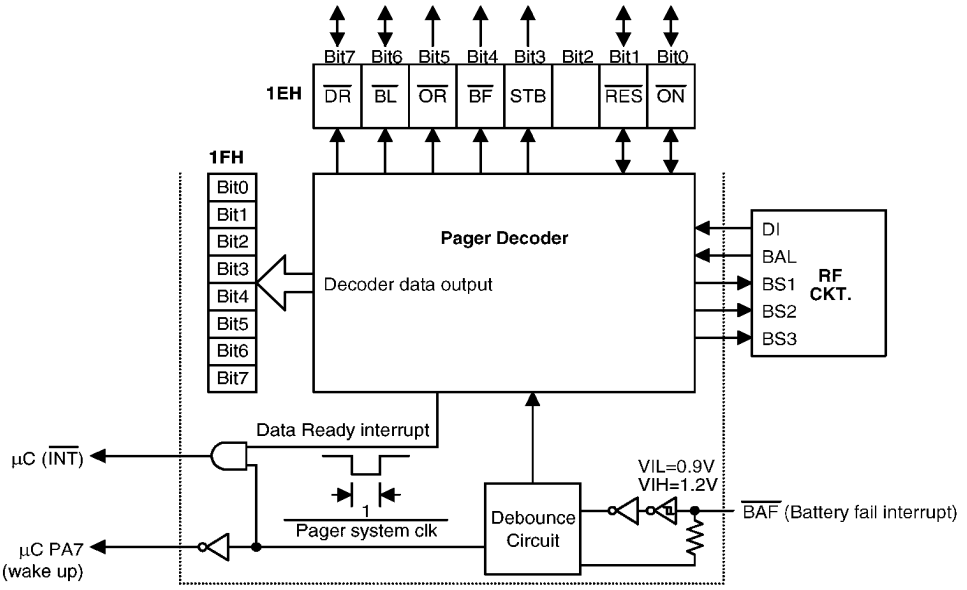
  The function bits ($\overline{ON}$, $\overline{RES}$) and indication bits (STB, $\overline{BF}$, $\overline{OR}$, $\overline{BL}$ and $\overline{DR}$) are all used to control the status of the decoder which is operated through the pager control address as described on the table.

- Pager Data Address

  The pager data address (1FH) are the parallel data lines for decoder data transfer.

Message data transfer

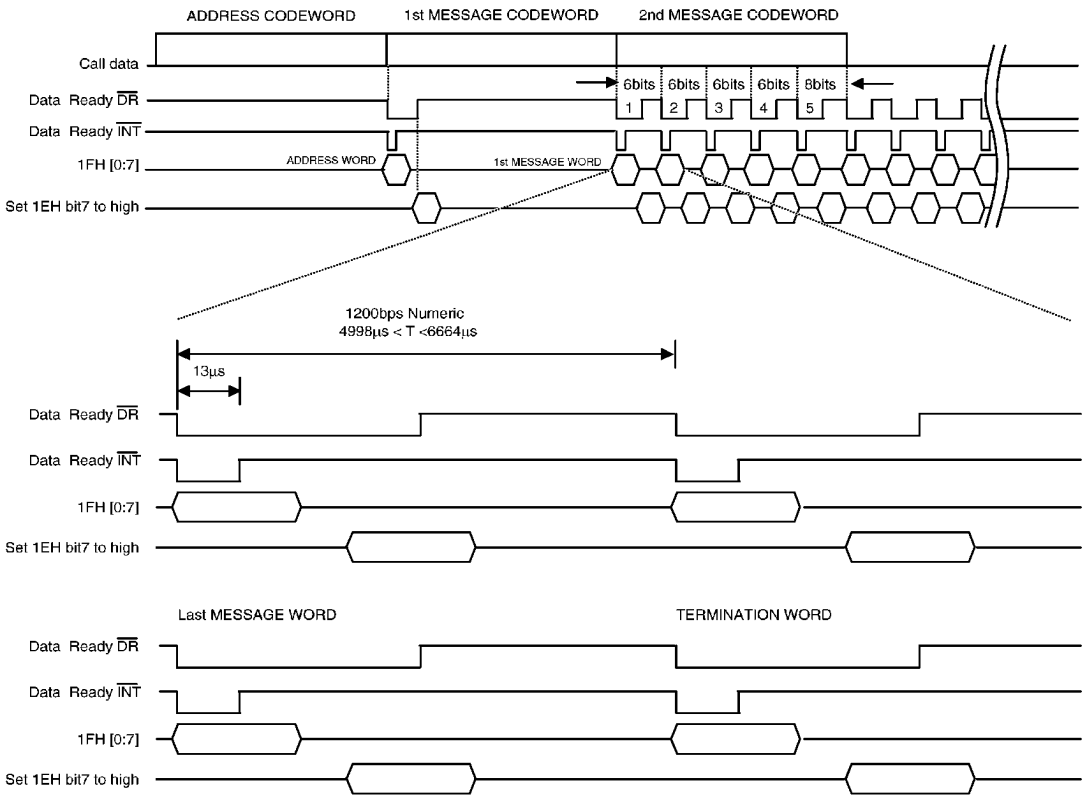
The decoder outputs a deformatted address word and message words upon receipt of a valid call. The message data to be transferred is organized into 8-bit words and transferred through the parallel pager data address (1FH) byte by byte. When a call word starts, the decoder generates a data ready interrupt simultaneously and runs the processing subroutine. The subroutine should read out the word in the pager data address (1FH) before the next call word comes in, i.e., the word should be read in 4mS at 512/1200 bit data rate and in 2mS at 2400 bit data rate. Otherwise, the data in 1FH is overridden by the next word.



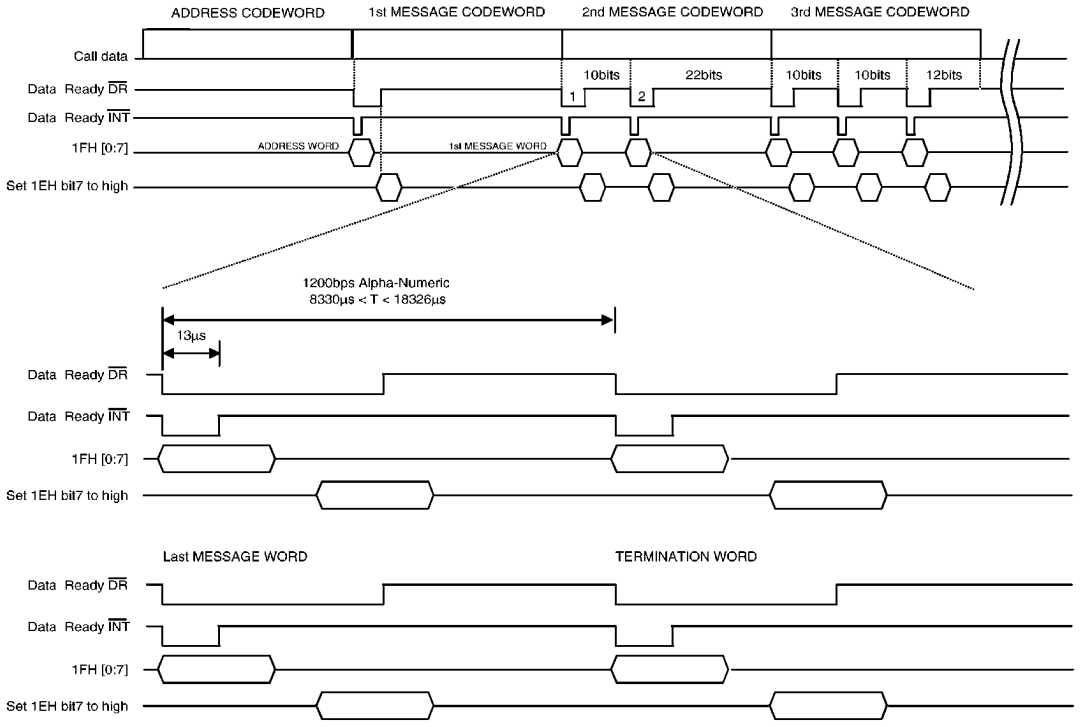
The value of 1EH-bit3 STB is set when decoder enteres the standby mode and cleared when decoder enters the $\overline{ON}$ mode.
The value of 1EH-bit4 $\overline{BF}$ is dependent on the $\overline{BAF}$ pin .
The value of 1EH-bit5 $\overline{OR}$ is always changed by out of range signal.
The value of 1EH-bit6 $\overline{BL}$ is cleared "0" by the decoder Battery Low signal and set "1" when μC sets this bit high.
The value of 1EH-bit7 $\overline{DR}$ is cleared "0" by the decoder Data-Ready interrupt signal and set "1" when μC sets this bit high.

Decoder interface

Symbol	Bit	R/W	Description
$\overline{ON}$	0	R/W	On/off control bit This bit selects the $\overline{ON}$ or STANDBY state of the decoder. 0: $\overline{ON}$ state 1: STANDBY state
$\overline{RES}$	1	R/W	Reset output for the decoder core The µC has to set the $\overline{RES}$ bit low and then high after the pager controller is turned on.
STB	3	R	Standby indication bit When the value of the $\overline{ON}$ bit is 1, the system goes into the STANDBY state. The STANDBY state allows the µC to execute the configuration RAM setting.
$\overline{BF}$	4	R	Battery fail indication bit Once the decoder detects that the battery fail interrupt is low, the $\overline{BF}$ bit will be low but unlatched.
$\overline{OR}$	5	R	Out-of-range indication bit Whenever the decoder detects an out-of-range condition, this bit is cleared low after expiration of the programmed out-of-range hold time that is selected by the configuration registers (SPF06 and SPF07). The out-of-range indication may be tested for an out-of-range condition whenever the interface enable of the decoder is active; otherwise $\overline{OR}$ is normally high . The out-of-range indication is set high by detection of a valid data transmission or by switching the decoder to be in the STANDBY state.
$\overline{BL}$	6	R/W	Battery low indication bit The battery low indication is periodically tested for a battery low condition. If the decoder encounters a battery low condition the battery low indication bit is cleared low. At this time, the µC should set the $\overline{BL}$ bit high.
$\overline{DR}$	7	R/W	Data ready interrupt indication bit When a valid call is detected, data starts to transfer. The $\overline{DR}$ bit becomes low when the serial data is changed to parallel data (1FH). After reading the parallel data, the µC software has to set the $\overline{DR}$ bit high.



Numeric message data transfer



Alpha-numeric message data transfer

The address word indicates call address, function bit setting, and decoder flags. The message codewords are received and concatenated to a valid call address word. The message bits are received in the message words consisting of an error flag. The error flag indicates that the message words are derived from un-corrected message codewords.

Data transfer for a received call ends right after the termination word is transferred.

Address word format

Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
Function code		Call address		1	Sync. State	Dup. Call	0

Bit 0: Bit 21 of the address codeword

Bit 1: Bit 20 of the address codeword

Bit 2	Bit 3	Call Address
0	0	RIC A
0	1	RIC B
1	0	RIC C
1	1	RIC D

Bit 5= 1 if an address codeword is received in the data fail mode

Bit 6= 1 if a duplicate call is received

Message word format

Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
LSB			Message Bits			MSB	Error Flag

Bit 7= 1 if the message bits are incorrect

The message data is converted into the ASCII code format.

Termination word format

Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
0	0	1	0	0	0	0	Error Flag

Bit 7= 1 when any call data codeword in the call termination is uncorrectable.

Data conversion

The decoder automatically converts the numerical or alphanumeric format of the received message codewords into the ASCII format. The conversion takes place depending on SPF13 and the function bit setting in the received address codeword.

Function Bits			Message Format
SPF 13	Bit 20	Bit 21	
0	X	X	numeric
1	0	0	numeric
1	0	1	alpha-numeric
1	1	0	alpha-numeric
1	1	1	alpha-numeric

When a conversion from the alpha-numeric format into the ASCII takes place, the received message codewords are split into message blocks of 7 bits in length. In contrast, when a conversion from the numeric format to the ASCII format takes place, the received message codewords are split into blocks of 4 bits in length. Each 4-bit block is converted to a 7-bit block according to the following table. Each message word is comprised of 7-bit block and an error flag.

Numeric format to ASCII conversion:

4-bit block				Character	7-bit block						
msb			lsb		msb						lsb
0	0	0	0	"0"	0	1	1	0	0	0	0
0	0	0	1	"1"	0	1	1	0	0	0	1
0	0	1	0	"2"	0	1	1	0	0	1	0
0	0	1	1	"3"	0	1	1	0	0	1	1
0	1	0	0	"4"	0	1	1	0	1	0	0
0	1	0	1	"5"	0	1	1	0	1	0	1
0	1	1	0	"6"	0	1	1	0	1	1	0
0	1	1	1	"7"	0	1	1	0	1	1	1
1	0	0	0	"8"	0	1	1	1	0	0	0
1	0	0	1	"9"	0	1	1	1	0	0	1
1	0	1	0	"*"	0	1	0	1	0	1	0
1	0	1	1	"U"	1	0	1	0	1	0	1
1	1	0	0	" "	0	1	0	0	0	0	0
1	1	0	1	"-"	0	1	0	1	1	0	1
1	1	1	0	"]"	1	0	1	1	1	0	1
1	1	1	1	"["	1	0	1	1	0	1	1

Configuration RAM organization

The decoder contains a 15-byte RAM to store 4 user addresses, 2 independently programmable frame numbers, and specially programmed function bits (SPF01~SPF32) for the decoder application configuration. The data memory is mapped to the addresses 40H~4EH of bank 27.

• Address register 0:

Bank 27 40H								**Bank 27 41H**							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$\overline{\text{ENA}}$	A00	A01	A02	A03	A04	A05	A06	A07	A08	A09	A10	A11	A12	A13	A14
Bank 27 42H								**Bank 27 43H**							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
A15	A16	A17	$\overline{\text{ENB}}$	B00	B01	B02	B03	B04	B05	B06	B07	B08	B09	B10	B11
Bank27 44H															
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0								
B12	B13	B14	B15	B16	B17										

• Address register 1:

Bank 27 45H								**Bank 27 46H**							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$\overline{ENC}$	C00	C01	C02	C03	C04	C05	C06	C07	C08	C09	C10	C11	C12	C13	C14
Bank 27 47H								**Bank 27 48H**							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
C15	C16	C17	$\overline{END}$	D00	D01	D02	D03	D04	D05	D06	D07	D08	D09	D10	D11
Bank27 49H															
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0								
D12	D13	D14	D15	D16	D17										

• Special register:

Bank 27 4AH								**Bank 27 4BH**							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FR12	FR11	FR10	FR22	FR21	FR20	SPF 01	SPF 02	SPF 03	SPF 04	SPF 05	SPF 06	SPF 07	SPF 08	SPF 09	SPF 10
Bank 27 4CH								**Bank 27 4DH**							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SPF 11	SPF 12	SPF 13	SPF 14	SPF 15	SPF 16	SPF 17	SPF 18	SPF 19	SPF 20	SPF 21	SPF 22	SPF 23	SPF 24	SPF 25	SPF 26
Bank27 4EH															
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0								
SPF 27	SPF 28	SPF 29	SPF 30	SPF 31	SPF 32										

User address format

A user address in the POCSAG code consists of 21 bits. Three of the 21 bits are coded in the frame number and are therefore not explicitly transmitted. In the decoder, the addresses A/B and C/D share the same frame number: addresses A and B reside in frame FR1 (FR10, FR11, FR12), and addresses C and D in frame FR2 (FR20, FR21, FR22). Every address has to be explicitly enabled by resetting the associated enable bit.

Examples:

Address decimal value: RICA=10535

Binary equivalent (14 bits): 10100100100111

Binary equivalent (18+3 bits):
000000010100100100111

Register allocation:

A00	A01	A02	A03	A04	A05	A06	A07	A08
0	0	0	0	0	0	0	1	0
A09	A10	A11	A12	A13	A14	A15	A16	A17
1	0	0	1	0	0	1	0	0

FR12	FR11	FR10
1	1	1

Configuration

The program mode changes to the STANDBY state by setting the $\overline{ON}$ bit high at any time. The configuration RAM can be programmed only when the value of the STB flag is 1. After

the configuration RAM is programmed and the $\overline{ON}$ bit is set low, the system quits the program mode and resumes normal operation.

Test mode

The test mode of the decoder is selected by setting the $\overline{TS}$ pin low at any time. In the test mode, the RF control outputs BS1 and BS3 are set high constantly, but BS2 is set low.

After the $\overline{TS}$ pin is set high the decoder exits the test mode.

RF control

The HT9280/HT9380 provides the BS1-BS3 signals for RF control.

• BS1: Receiver enabled

Receiver establishment time (t_{BS1})		Option	
512 bps	**1200/2400 bps**	**SPF04**	**SPF05**
7.81ms	53.33ms	0	0
15.63ms	6.67ms	0	1
31.25ms	13.33ms	1	0
62.50ms	26.67ms	1	1

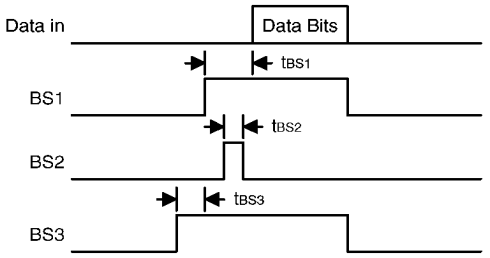
• BS2: Quick charge

RF DC level adjustment time (t_{BS2})		Option	
512 bps	**1200/2400 bps**	**SPF20**	**SPF21**
7.81ms	1.67ms	0	0
11.71ms	6.67ms	0	1
15.63ms	11.67ms	1	0
19.53ms	13.33ms	1	1

• BS3: PLL enabled

PLL establishment time (t_{BS3})		Option	
512 bps	**1200/2400 bps**	**SPF22**	**SPF23**
0ms	0ms	0	0
31.25ms	26.67ms	0	1
46.87ms	40.00ms	1	0
62.50ms	53.33ms	1	1

Timing



Timing

Description of the special programmed function bits (SPF)

The following features can be selected by appropriate programming of the specially programmed function bits:

• SPF01

1: 4-bit burst error correction for message codeword

0: 2-bit burst error correction for message codeword

• SPF02

1: 1200 bit data rate with 76800Hz crystal only

0: 512 bit data rate

When running at a 2400 bit data rate, the value of SPF02 should be 1 (the value of SPF18 should also be set to 1).

• SPF03

1 : 76.8 or 153.6kHz crystal configuration

0 : 32768Hz crystal configuration

• SPF04, SPF05:

Receiver (BS1) establishment time (for the BS2-BS3 options, refer to SPF20-SPF23)

00: 7.81ms/512 53.33ms/1200/2400
01: 15.63ms/512 6.67ms/1200/2400
10: 31.25ms/512 13.33ms/1200/2400
11: 62.50ms/512 26.67ms/1200/2400

- SPF06, SPF07

  Duplicate the call suppress timeout and out-of-range hold-off timeout
  00: 30s/512/1200 15s/2400
  01: 60s/512/1200 30s/2400
  10: 120s/512/1200 60s/2400
  11: 240s/512/1200 120s/2400

- SPF08~SPF11

  Reserved, should be 0

- SPF12, SPF13

  Call termination criteria combination method and message data

  Deformatting method
  0x : Any two consecutive codewords or the codeword directly following the address codeword in error
  10 : Any single codeword in error
  11 : Any two consecutive codewords in error
  x0 : Numeric data deformation
  x1 : Numeric data deformation on function code 00 only

- SPF14

  Reserved, should be 0

- SPF15

  1: Out-of-range Hold-off period according to SPF06 and SPF07
  0: Out-of-range Hold-off period is zero regardless of SPF06 and SPF07

- SPF16

  Reserved, should be 0

- SPF17

  Reserved, should be 0

- SPF18

  1: 2400 bits data rate operation
  0: 512/1200 bits data rate operation

- SPF19

  Reserved, should be 0

- SPF20~SPF21

  RF dc level adjustment (BS2) enable time

  00: 7.81ms/512 1.67ms/1200/2400
  01: 11.71ms/512 6.67ms/1200/2400
  10: 15.63ms/512 11.67ms/1200/2400
  11: 19.53ms/512 13.33ms/1200/2400

- SPF22~SPF23

  PLL (BS3) establishment time

  00: 0ms/512 0ms/1200/2400
  01: 31.25ms/512 26.67ms/1200/2400
  10: 46.87ms/512 40.00ms/1200/2400
  11: 62.50ms/512 53.33ms/1200/2400

- SPF24~SPF25

  Tone generation frequency prescaler divider

  00: Prescaler factor 1
  01: Prescaler factor 2
  10: Prescaler factor 4
  11: Prescaler factor 8

- SPF26

  Tone generation frequency source selector

  0: System clock
  1: 32.768kHz

- SPF27~SPF30

  Spare

- SPF31

  Reserved, should be 0

- SPF32

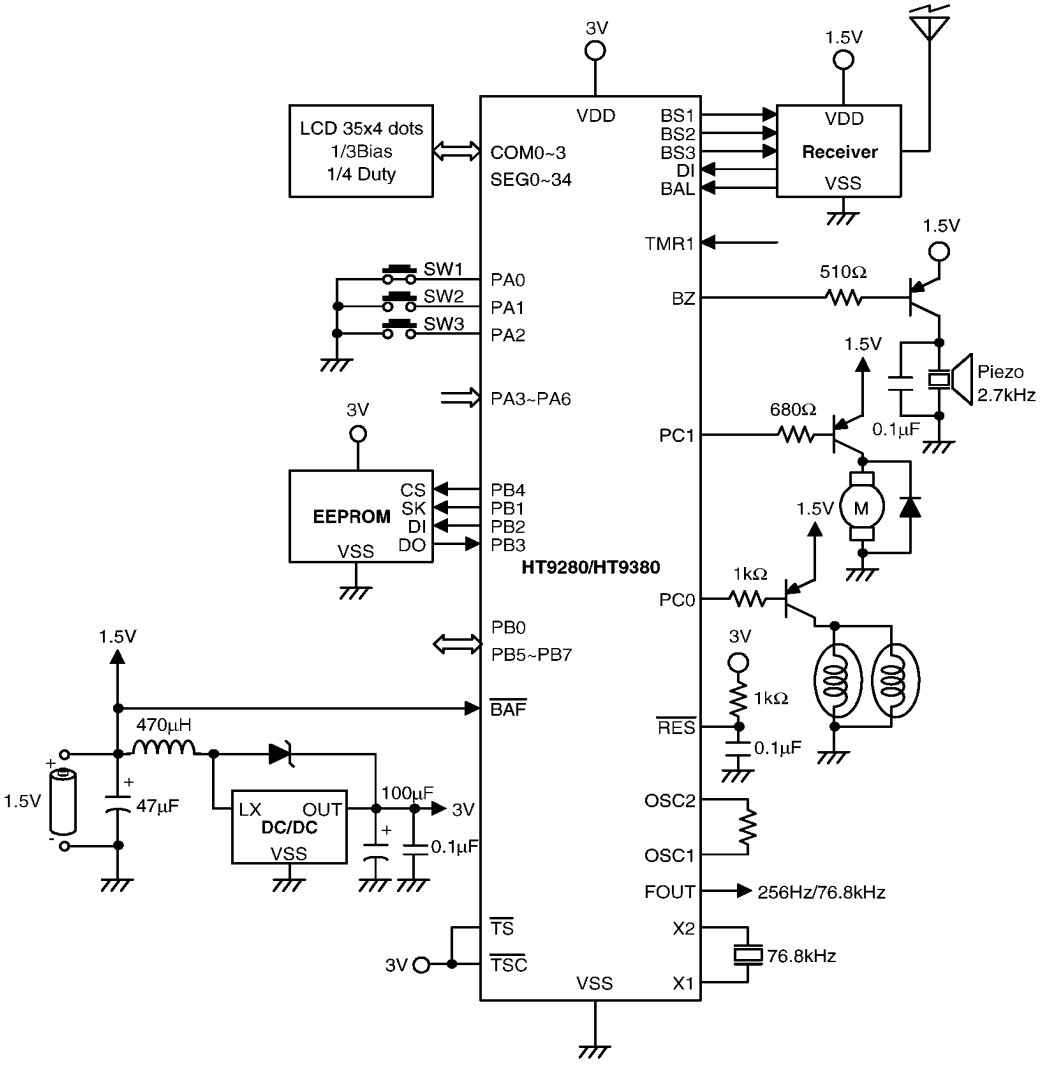
  Non-inversion or inversion data input selection
  1: Inversion input selected for DI from RF circuit, referring to $\overline{DI}$
  0: Non-inversion input selected for DI from RF circuit

Mask option

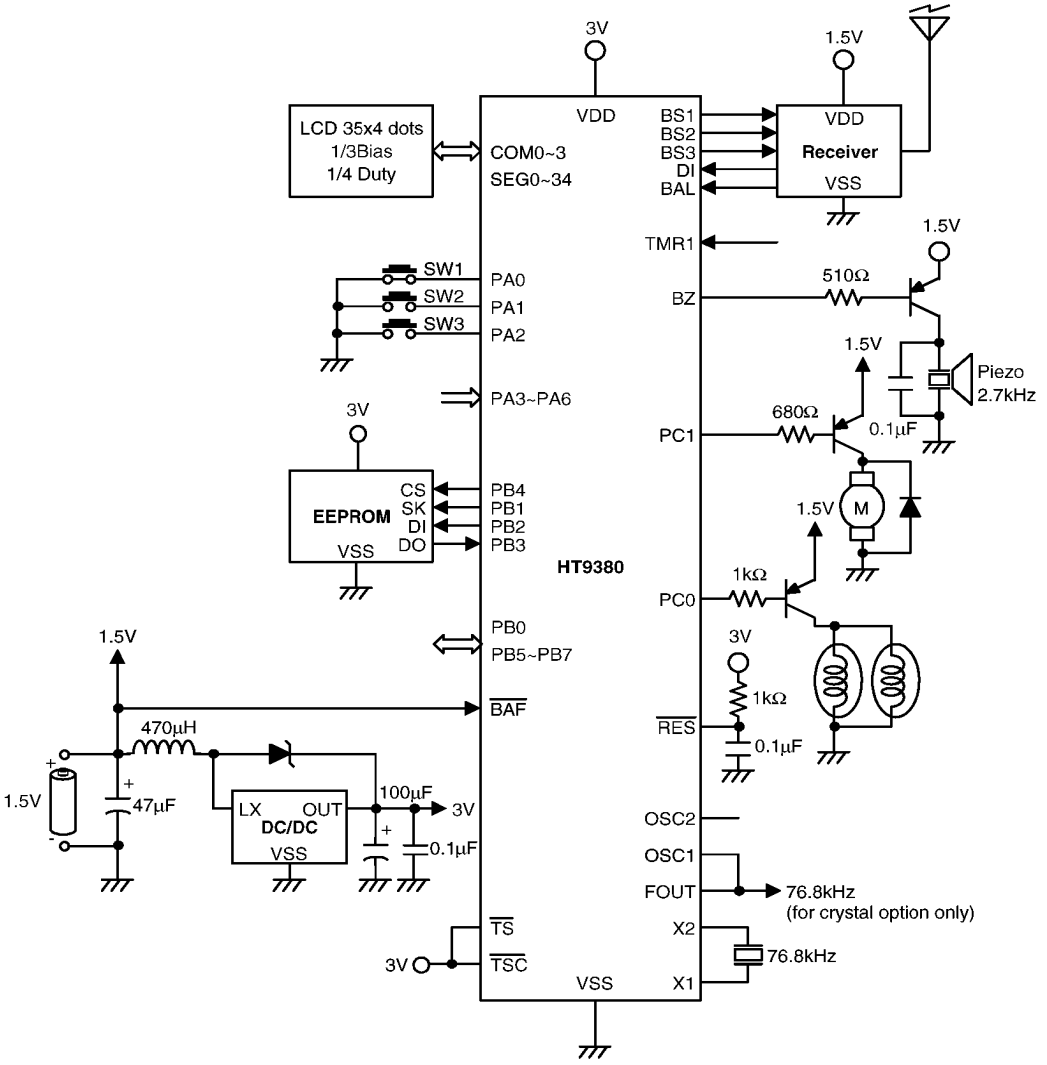
The following table illustrates six kinds of mask options in the HT9280/HT9380. All of the options should be defined to ensure proper system functioning.

No.	Mask Option
1	HALT function selection. This option defines the way of enabling or disabling the HALT function.
2	WDT source selection. This option selects the WDT source, either from the subsystem clock or instruction clock, or opt to disable the WDT function.
3	CLRWDT times selection. This option defines the way of clearing the WDT by instruction. "Once" means that the "CLR WDT" can clear the WDT and "Twice" implies that the "CLR WDT1" and "CLR WDT2" should be executed before the time-out so as to clear the WDT.
4	Wake-up selection. This option defines the wake-up activity. Port A has the capability of waking-up the chip from HALT.
5	PB, PC0, and PC1 pull-high options
6	µC OSC type selection. This option is to decide whether an RC or Crystal oscillator is chosen as the system clock. If the Crystal oscillatior is selected, the SST (Crystal Start-up Timer) default is activated for HT9280 application, otherwise the SST is disabled.

Application Circuit 1



Application Circuit 2



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 & Decrement		
INCA [m]	Increment data memory with result in ACC	Z
INC [m]	Increment data memory	Z
DECA [m]	Decrement data memory with result in ACC	Z
DEC [m]	Decrement data memory	Z
Rotate		
RRA [m]	Rotate data memory right with result in ACC	None
RR [m]	Rotate data memory right	None
RRCA [m]	Rotate data memory right through carry with result in ACC	C
RRC [m]	Rotate data memory right through carry	C
RLA [m]	Rotate data memory left with result in ACC	None
RL [m]	Rotate data memory left	None
RLCA [m]	Rotate data memory left through carry with result in ACC	C
RLC [m]	Rotate data memory left through carry	C

Mnemonic	Description	Flag Affected
Data Move		
MOV A,[m]	Move data memory to ACC	None
MOV [m],A	Move ACC to data memory	None
MOV A,x	Move immediate data to ACC	None
Bit Operation		
CLR [m].i	Clear bit of data memory	None
SET [m].i	Set bit of data memory	None
Branch		
JMP addr	Jump unconditional	None
SZ [m]	Skip if data memory is zero	None
SZA [m]	Skip if data memory is zero with data movement to ACC	None
SZ [m].i	Skip if bit i of data memory is zero	None
SNZ [m].i	Skip if bit i of data memory is not zero	None
SIZ [m]	Skip if increment data memory is zero	None
SDZ [m]	Skip if decrement data memory is zero	None
SIZA [m]	Skip if increment data memory is zero with result in ACC	None
SDZA [m]	Skip if decrement data memory is zero with result in ACC	None
CALL addr	Subroutine call	None
RET	Return from subroutine	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

Notes:

x= 8-bit immediate data

m= 8-bit data memory address

A= accumulator

i= 0...7 number of bits

√= Flag(s) is affected

–= Flag(s) is not affected

*= Flag(s) may be affected by the result of the execution

addr= 13-bit program memory address

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 the WDT Prescaler are cleared (re-counting from 0). 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 0), if the other preclear WDT instruction had been executed. Execution of this instruction only, without the other preclear instruction sets the indicating flag which implies that this instruction was executed. 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 0), if the other preclear WDT instruction had been executed. Execution of this instruction only without the other preclear instruction, sets the indicating flag which implies that this instruction was executed. 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 1 are changed to 0 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 store result in the accumulator

Description Each bit of the specified data memory is logically complemented (1's complement). Bits which previously contained a 1 are changed to 0 and vice-versa. The complemented result is stored in the accumulator and the contents of the data memory remain unchanged.

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

Affected flag(s)

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

DAA [m] Decimal-Adjustment accumulator for addition

Description The accumulator value is adjusted to BCD (Binary Code Decimal) code. The accumulator is divided into 2 nibbles. Each nibble is adjusted to BCD code and an internal carry (AC1) will be created 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 store result in the accumulator

Description Data in the specified data memory is decremented by 1, leaving the result in the accumulator. The contents of the data memory remain unchanged.

Operation ACC ← [m]–1

Affected flag(s)

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 1

Operation [m] ← [m]+1

Affected flag(s)

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

INCA [m] Increment data memory and store result in the accumulator

Description Data in the specified data memory is incremented by 1, leaving the result in the accumulator. The contents of the data memory remain unchanged.

Operation ACC ← [m]+1

Affected flag(s)

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

JMP addr Direct Jump

Description Bits 0~11 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 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 to 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 2-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 is 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 the data memory towards the 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 the data memory towards the left and store the result in the 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 the 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 store result in the 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 the data memory towards the 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 the data memory right then store result in the 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 the 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 then store result in the 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
–	–	–	–	√	√	√	√

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 2-cycle instruction. Otherwise proceed to 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 then store result in ACC, skip if zero

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

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

Operation [m].i ← 1

Affected flag(s)

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

SIZ [m] Skip if increment data memory is zero

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

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

Affected flag(s)

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

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

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

encryption The low-order and high-order nibbles of the specified data memory are interchanged, the result is stored in 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 2-cycle instruction. Otherwise proceed to the next instruction.

Operation Skip if [m]=0

Affected flag(s)

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

SZA [m] Move the 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 2-cycle instruction. Otherwise proceed to the next instruction.

Operation Skip if [m]=0, ACC ← [m]

Affected flag(s)

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

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

Description If bit i of the specified data memory is zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction. This is a 2-cycle instruction. Otherwise proceed to 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 the ROM code (current page) addressed by the table pointer (TBLP) is moved to the specified data memory and the high byte is 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 the ROM code (last page) addressed by the table pointer (TBLP) is moved to the data memory and the high byte is 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 the 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
–	–	–	–	–	√	–	–