Features

Patent Pending: 87208775 (R.O.C.), 98214300.1 (P.O.C)

- Operating voltage: 2.0V~5.2V
- Operation current<1mA max. @V_{DD}=2.5V
- Standby current<10μA (32768Hz ON and system oscillator OFF) @V_{DD}=2.5V
- Dialer mode interrupt for $\overline{\text{HKS}}$, HFI, $\overline{\text{HDI}}$ pins
- 26 I/O pins structure depends on mask option in either dialer mode or normal mode
- One 8-bit and one 16-bit programmable timer with overflow interrupts
- 8/16 timer clock sources selectable via 32768Hz external source or 1/4 system clock internal source
- Watchdog timer (external source 32768Hz only)
- Built-in RC system oscillator. System clock can be selected as 256kHz, 512kHz, 1MHz and 1.5MHz via metal option
- 4K×15 program memory ROM
- 224×8 data memory RAM
- Built-in 32768Hz DTMF generator for dialer application
- Built-in option resistor for dialer application
- 63 powerful instructions
- All instructions in 1 or 2 machine cycles
- 15-bit table read instructions
- 4-level subroutine nesting
- Bit manipulation instructions

General Description

The HT99U210 is an 8-bit high performance RISC-like microcontroller used for cored or cordless phone applications.

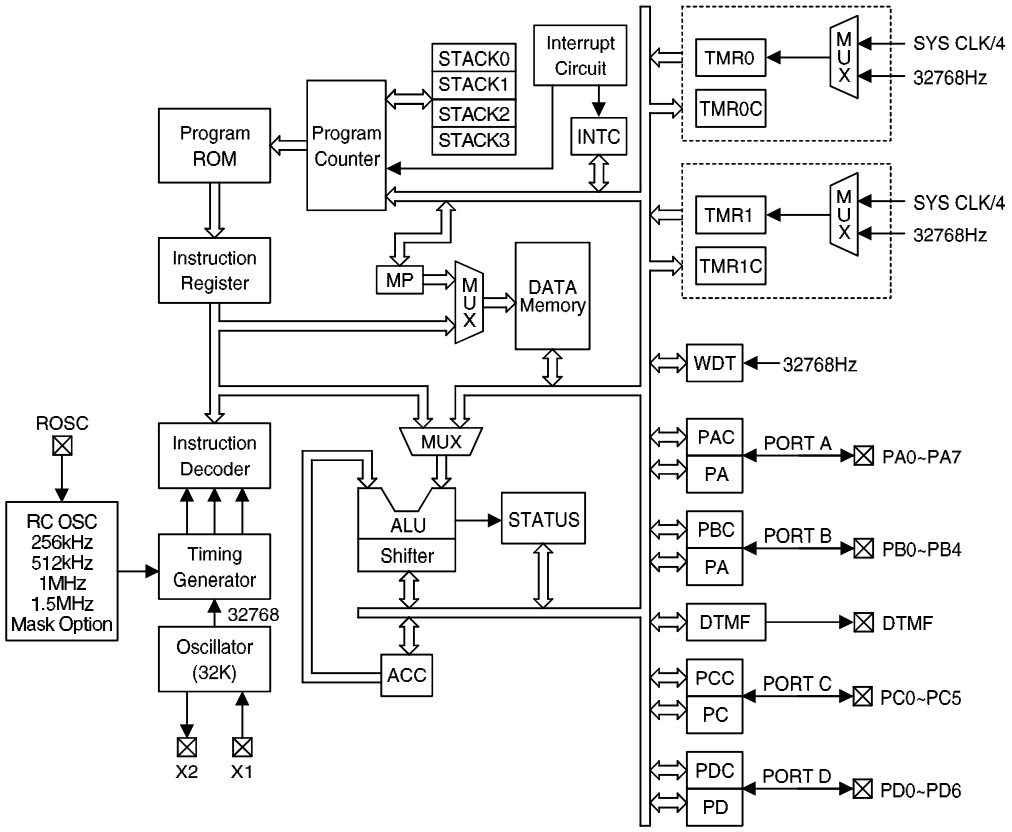
The HT99U210 consists of a built-in DTMF generator. Together with option resistors and option I/O ports, this IC provides a very flexible programmable telecommunication control device.

The device is particularly suitable for use in products such as cordless phone controllers, μC dialers, feature phone controllers and various subsystem controllers. A halt feature is included to reduce power consumption.

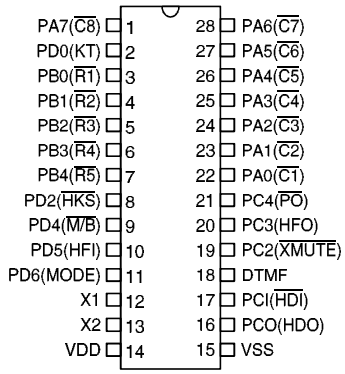
Selection Table

Function / Part No.	ROM (Bits)	RAM (Bits)	I/O (Lines)	DTMF Gen.	PFD	CPT Interrupt	RTC
HT99U210	4K×15	224×8	26	√	—	—	—
HT99U410	4K×15	224×8	32	√	√	√	√
HT99U810	8K×16	1536×8	56	√	√	√	√

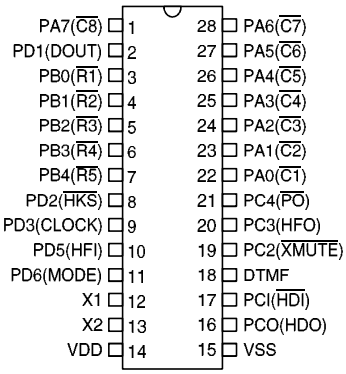
Block Diagram



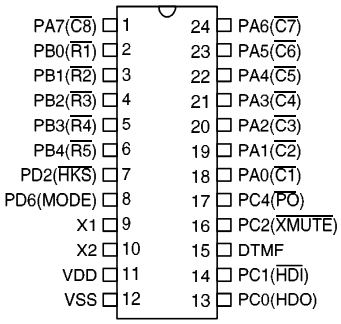
Pin Assignment



HT99U210A
– 28 SDIP/SOP



HT99U210B
– 28 SDIP/SOP



HT99U210C
– 24 SDIP/SOP

Pad Description

Pad No.	Pad Name	I/O	Mask Option	Description
	PA0~PA3	I/O	Wake-up or None	These pins have schmitt trigger with pull-high resistor (fixed) input and open drain output. Generally, we used these pins together with PB0~PB3 to form a keyboard matrix. In addition, they have a special function keyboard wherein various software specifications can be selected by adding resistors across keyboard matrix pins.
	PA4~PA7	I/O	Wake-up or None Schmitt Trigger with Pull-high or None Dialer Mode or Normal Mode	These pins have schmitt trigger with pull-high (option) input and either open drain output in dialer mode or CMOS output in normal mode. The purpose of these pin in dialer mode is the same as PA0~PA3 otherwise it is a general I/O.
	PB0~PB3	I/O	—	These pins have schmitt trigger with pull-high resistor (fixed) input and open drain output. Generally, we used these pins together with PA0~PA3 to form a keyboard matrix. In addition, they have a special function keyboard wherein various software specifications can be selected by adding resistors across keyboard matrix pins.

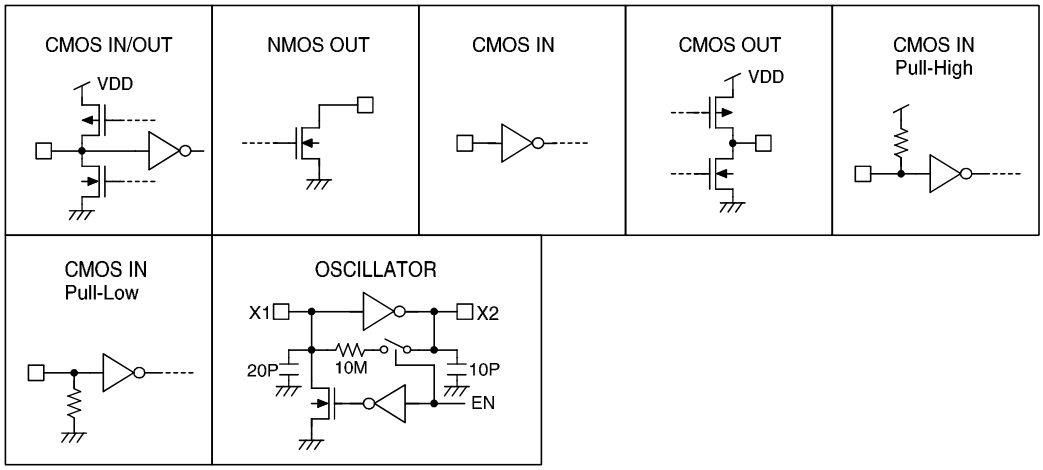
Pad No.	Pad Name	I/O	Mask Option	Description
	PB4	I/O	Schmitt Trigger with Pull-high or None Dialer Mode or Normal Mode	This pin has schmitt trigger with pull-high (option) input and either open drain output in dialer mode or CMOS output in normal mode. The purpose of this pin in dialer mode is the same as PB0~PB3 otherwise it is a general I/O.
	X1	I	—	The time base oscillator consists of an inverter, a bias resistor and the necessary on chip load capacitor. Connect a standard 32768Hz crystal to X1, X2.
	X2	O		
	PC5	I/O	—	This pin has schmitt trigger input and open drain output.
	PC4	I/O	Schmitt Trigger with Pull-high or None Dialer Mode or Normal Mode	This pin has a dialer function which is the same as $\overline{PO}$ in dialer mode. It will be controlled directly by PD2 ($\overline{HKS}$), PC1 ($\overline{HDI}$) and PD5 (HFI) when these pins are also set in dialer mode. For example, when PD2 ($\overline{HKS}$) is low or PC1 ($\overline{HDI}$) is low or PD5 (HFI) is low, these situations will let PC4 ($\overline{PO}$) go high to make the line. PC4 is CMOS output. If this pin is set in normal mode, it only has a general I/O function.
	PC3	I/O	Schmitt Trigger with Pull-high or None Dialer Mode or Normal Mode	This pin has a dialer function which is the same as HFO in dialer mode. It will be controlled directly by PD5 (HFI) when this pin is also set in dialer mode. For example, when PD5 (HFI) is in positive transition then PC3 (HFO) is set to high and the status can be monitored in embedded register [Addr=21H] which is a read only register. When PC3 (HFO) is set to high, it cannot be changed by programming until PD5 (HFI) is toggle changed to another positive transition or PC1 ($\overline{HDI}$) is activated or PD2 ($\overline{HKS}$) is activated then PC3 (HFO) is from high to low. PC3 is CMOS output. If this pin is set in normal mode, it only has a general I/O function.
	PC2	I/O	—	This pin always has a dialer function which is the same as $\overline{XMUTE}$ function. This pin has open drain output and controlled by programming.

Pad No.	Pad Name	I/O	Mask Option	Description
	PC1	I/O	Schmitt Trigger with Pull-high or None Dialer Mode or Normal Mode	This pin has a dialer function which is the same as $\overline{\text{HDI}}$ in dialer mode. It will control the PC4 ($\overline{\text{PO}}$) and PC0 (HDO) directly, when these pins are also set in dialer mode. It is an important property of PC1 ($\overline{\text{HDI}}$) pin that this pin will generate an interrupt signal when a negative transition occured, then PC0 (HDO) is set to high and the status can be monitored in embedded register [Addr=21H] which is a read only register. PC1 is CMOS output. If this pin is set in normal mode, it only has a general I/O function.
	PC0	I/O	Dialer Mode or Normal Mode	This pin has a dialer function which is the same as HDO in dialer mode. It will be controlled directly by PC1 ($\overline{\text{HDI}}$) when this pin is also set in dialer mode. For example, when PC1 ($\overline{\text{HDI}}$) has negative transition then PC0 (HDO) is set to high and the status can be monitored in embedded register [Addr=21H] which is a read only register. When PC0 (HDO) is set to high, it cannot be changed by programming until PC1 ($\overline{\text{HDI}}$) is toggle changed to another negative transition or PD5 (HFI) is activated or PD2 ($\overline{\text{HKS}}$) is activated then PC0 (HDO) is from high to low. PC0 is CMOS output. If this pin is set in normal mode, it only has a general I/O function.
	DTMF	O	—	This pin generates an expected muti-tone through a programming embedded register [Addr=20H] and output control by embedded register [Addr=21H] TP bit. When TP=0 disable DTMF generator. TP=1 enable DTMF generator.
	PD0	I/O	Schmitt Trigger with Pull-high or None	This pin has schmitt trigger with pull-high resistor (option) input and CMOS output.
	PD1	I/O	—	This pin is an NMOS open drain output pin and its function is the same as DOUT in dialer mode. Through programming, it outputs the BCD code of the dialing digits to the LCD driver chip (HT16XX series) or to the μC to display the dialed number.

Pad No.	Pad Name	I/O	Mask Option	Description
	PD2	I/O	Schmitt Trigger with Pull-high or None Dialer Mode or Normal Mode	This pin has a dialer function which is the same as $\overline{HKS}$ in dialer mode. It will control the PC4 ($\overline{PO}$) directly, when this pin is also set in dialer mode. It is an important property of PD2 ($\overline{HKS}$) pin that this pin will generate a reset and interrupt signals when this pin is either negative or positive transition. When PD2 ($\overline{HKS}$)=V_{DD} then PC4 ($\overline{PO}$)=low PD2 ($\overline{HKS}$)=V_{SS} then PC4 ($\overline{PO}$)=high PD2 is CMOS output. If this pin is set in normal mode, it only has a general I/O function.
	PD3	I/O	—	This pin is an NMOS open drain output pin and its function is the same as CLOCK in dialer mode. Through programming, during dialing, this pin generates a series of pulse trains for PD1 (DOUT) data synchronization.
	PD4	I/O	Schmitt Trigger with Pull-high or None Dialer Mode or Normal Mode	This pin has a dialer function which is the same as M/B in dialer mode. This pin has a tristate input to control an embedded register [Addr=22H] through programming, to determine make and break ratio. If this pin is set in normal mode, it only has a general I/O function.
	PD5	I/O	Schmitt Trigger with Pull-high or None Dialer Mode or Normal Mode	This pin has a dialer function which is the same as HFI in dialer mode. It will control the PC4 (PO) and PC3 (HFO) directly, when these pins are also set in dialer mode. It is an important property of PD5 (HFI) pin that this pin will generate reset and interrupt signals when positive transition occured, then PC3 (HFO) is set to high and the status can be monitored in embedded register [Addr=21H] which is a read only register. PD5 is CMOS output. If this pin is set in normal mode, it only has a general I/O function.

Pad No.	Pad Name	I/O	Mask Option	Description
	PD6	I/O	Schmitt Trigger with Pull-high or None Dialer Mode or Normal Mode	This pin has a dialer function which is the same as MODE in dialer mode. This pin has a tristate input through programming, to control embedded register [Addr=22H] to determine dialing mode specification. If this pin is set in normal mode, it only has a general I/O function.
	VDD	—	—	Positive power supply, 2.0V~5.2V
	VSS	—	—	Negative power supply, GND

Approximate internal connection circuits



Absolute Maximum Ratings*

Supply Voltage ..... –0.3V to 6V

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

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

Operating Temperature ..... –20°C to 75°C

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

Electrical Characteristics

System oscillator=256kHz, Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.0	—	5.2	V
I_{DD}	Operating Current	2.5V	—	—	—	1	mA
I_{STB}	Standby Current	2.5V	No load, system halt	—	—	10	μA
I_{OL1}	Normal I/O Ports Source Current	2.5V	—	1	—	—	mA
I_{OH1}	PA, PB, Normal I/O Ports Sink Current	2.5V	—	0.3	—	—	mA
R_{PH}	Pull-high Resistance of I/O Ports	2.5V	—	30	—	120	kΩ
I_{XMO}	$\overline{XMUTE}$ Leakage Current	—	$V_{\overline{XMUTE}}$=12V No entry	—	—	1	μA
I_{OLXM}	$\overline{XMUTE}$ Sink Current	2.5V	$V_{\overline{XMUTE}}$=0.5V	1	—	—	mA
R_{HFI}	HFI Pull-low Resistance	2.5V	V_{HFI}=2.5V	—	200	—	kΩ
$R_{\overline{HDI}}$	$\overline{HDI}$ Pull-high Resistance	2.5V	$V_{\overline{HDI}}$=0V	—	200	—	kΩ
I_{OH2}	HFO Pin Source Current	2.5V	V_{OH}=2V	–1	—	—	mA
I_{OL2}	HFO Pin Sink Current	2.5V	V_{OL}=0.5V	1	—	—	mA
I_{OH3}	HDO Pin Source Current	2.5V	V_{OH}=2V	–1	—	—	mA
I_{OL3}	HDO Pin Sink Current	2.5V	V_{OL}=0.5V	1	—	—	mA
I_{OH4}	KT Pin Source Current	2.5V	V_{OH}=2V	–1	—	—	mA
I_{OL4}	KT Pin Sink Current	2.5V	V_{OL}=0.5V	1	—	—	mA
I_{OH5}	PO Pin Source Current	2.5V	V_{OH}=2V	–1	—	—	mA
I_{OL5}	PO Pin Sink Current	2.5V	V_{OL}=0.5V	1	—	—	mA
I_{OL6}	DNPO Pin Sink Current	2.5V	V_{OL}=0.5V	1	—	—	mA
I_{OL7}	CLOCK Pin Sink Current	2.5V	V_{OL}=0.5V	1	—	—	mA
I_{OL8}	DOUT Pin Sink Current	2.5V	V_{OL}=0.5V	1	—	—	mA

DTMF generator electrical characteristics Time base=32768Hz, Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{TDC}	DTMF Output DC Level	2.5V	—	$0.45V_{DD}$	—	$0.7V_{DD}$	V
V_{TOL}	DTMF Sink Current	2.5V	V_{DTMF}=0.5V	0.1	—	—	mA
V_{TAC}	DTMF Output AC Level	2.5V	Row Group, R_L=5kΩ	100	—	160	mVrms
R_L	DTMF Output Load	2.5V	THD≤–23dB	5			kΩ
A_{CR}	Column Pre-emphasis	2.5V	Row Group=0dB	1	2	3	dB
THD	Tone Signal Duration	2.5V	R_L=5kΩ	—	–30	–23	dB

THD (Distortion) (dB) = 20 log ($\sqrt{V1^2 + V2^2 + \ldots + Vn^2}$ / $\sqrt{Vi^2 + Vh^2}$)

Vi, Vh: Row group and column group signals

V1, V2, ... Vn: Harmonic signals (BW=300Hz~3500Hz)

Single tone frequency deviation

Frequency	Theoretical Freq.	% (Theoretical Error)	Max. Draft Freq.	%
697Hz	697.191Hz	0.027	—	—
770Hz	771.012Hz	0.131	774.657Hz	0.605
852Hz	851.117Hz	–0.104	836.718Hz	0.62
941Hz	936.229Hz	–0.507	—	—
1209Hz	1213.63Hz	0.383	—	—
1336Hz	1337.469Hz	0.11	1348.48Hz	0.934
1477Hz	1472.719Hz	–0.29	1466.13Hz	–0.736
1633Hz	1638.4Hz	0.331	—	—

*The maximum drift frequency is calculated by the duty cycle 40% drift of crystal.

Functional Description

Execution flow

The system clock for the HT99U210 is derived from the internal RC oscillator that can be selected from 256kHz, 512kHz, 1MHz and 1.5MHz by metal option. The system clock is internally divided into four non-overlapping clocks. One instruction cycle consists of four system clock cycles.

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

Program counter – PC

The 12-bit program counter (PC) controls the sequence in which the instructions stored in program ROM are executed. Its contents specify a maximum of 4096 addresses.

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

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

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

The lower byte of the program counter (PCL) is a read/write register (06H). Moving data into the PCL performs a short jump. The destination will be within 256 locations.

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

Program memory – ROM

The program memory is used to store the program instructions which are to be executed. It also contains data, table, and interrupt entries, and is organized with 4096×15 bits, addressed by the program counter and table pointer.

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

- Location 000H

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

- Location 004H

  This area is reserved for the dialer mode interrupt service program. If $\overline{\text{HKS}}$, HFI, $\overline{\text{HDI}}$ input pins is activated, then interrupt is enabled and the stack is not full, the program begins execution at location 004H.

- Location 008H

  This area is reserved for the Timer 0 interrupt service program. If a timer interrupt resulting from a Timer 0 overflow occurs, and if the interrupt is enabled and the stack is not full, the program begins execution at location 008H.

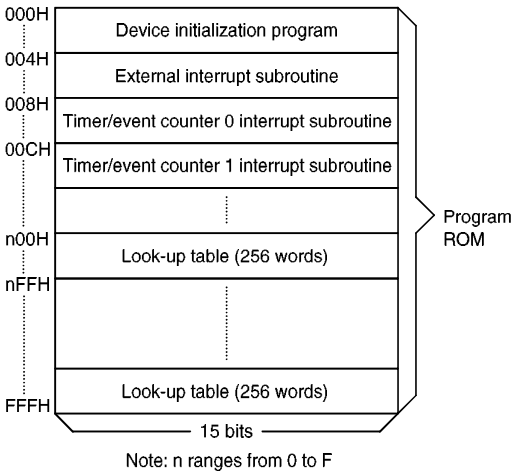
- Location 00CH

  This area is reserved for the Timer 1 interrupt service program. If a timer interrupt resulting from a Timer 1 overflow occurs, and if the interrupt is enabled and the stack is not full, the program begins execution at location 00CH.

- Table location

  Any location in the ROM space can be used as look-up tables. The instructions TABRDC [m] (the current page, 1 page=256 words) and TABRDL [m] (the last page) transfer the contents of the lower-order byte to the specified data memory, and the higher-order byte to TBLH (08H). Only the destination of the lower-order byte in the table is well-defined, the other bits of the table word are transferred to the lower portion of TBLH, the remaining one bit is read as 0. The Table Higher-order byte register (TBLH) is read

only. The TBLH is read only and cannot be restored. If the main routine and the ISR (Interrupt Service Routine) both employ the table read instruction, the contents of the TBLH in the main routine are likely to be changed by the table read instruction used in the ISR. Errors can occur. For this reason using the table read instruction in the main routine and the ISR simultaneously should be avoided. However, if the table read instruction has to be applied in both the main routine and the ISR, the interrupt(s) should be disabled prior to the table read instruction. It (they) will not be enabled again until the TBLH has been backed up. The table pointer (TBLP) is a read/write register (07H), which indicates the table location. Before accessing the table, the location must be placed in TBLP. All table related instructions need 2 cycles to complete the operation. These areas may function as normal program memory depending upon the requirements.



Program memory

Stack register – STACK

This is a special part of the memory which is used to save the contents of the program counter (PC) only. The stack is organized into 4 levels and is neither part of the data nor part of the program space, and is neither readable nor writeable. The activated level is indexed by the stack pointer (SP) and is neither readable nor writeable. At a subroutine call or interrupt acknowledgment, the contents of the program counter are pushed onto the stack. At the end of a subroutine or an interrupt routine, signaled by a return instruction (RET or RETI), the program counter is restored to its previous value from the stack. After a chip reset, the SP will point to the top of the stack.

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

Program counter

Notes: *11~*0: Program counter bits
#11~#0: Instruction code bits
S11~S0: Stack register bits
@7~@0: PCL bits

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

Data memory – RAM

The data memory is divided into three functional groups: special function registers (24×8); embedded control registers (3×8) and general purpose data memory (224×8). Most are read/write but some are read only.

The special function registers include the indirect addressing register 0 (00H), the memory pointer register 0 (MP0;01H), the indirect addressing register 1 (02H), the memory pointer register 1 (MP1;03H), the bank pointer register (04H), the accumulator (ACC;05H), the program counter lower-byte register (PCL;06H), the table pointer (TBLP;07H), the table higher-order byte register (TBLH;08H), the status register (STATUS;0AH), the interrupt control register (INTC;0BH), the timer 0 higher-order byte register (TMR0H;0CH), the timer 0 lower-order byte register (TMR0L;0DH), the timer 0 control register (TMR0C;0EH), the timer 1 (TMR1;10H), the timer 1 control register (TMR1C;11H), the I/O registers (PA;12H, PB;14H, PC;16H, PD;18H) and the I/O control registers (PAC;13H, PBC;15H, PCC;17H, PDC;19H). The embedded control registers are used to control the DTMF generator, I/O pin states of the dialer mode, 32768Hz oscillation speed up, ...etc.

The general purpose data memory includes Bank0, addressed from 40H to CFH; Bank1, addressed from 40H to 80H. They are used for data and control information under instruction command.

All data memory areas can execute arithmetic, logic, increment, decrement and rotate operations directly. Except for some dedicated bits, each bit in the data memory can be set and reset by the SET [m].i and CLR [m].i instructions, respectively. They are also indirectly accessible through the memory pointer registers (MP0;01H, MP1;03H).

Indirect addressing register

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

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

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

Table location

Notes: *11~*0: Table location bits

P11~P8: Current program counter bits

@7~@0: Table pointer bits

Data RAM map

General purpose RAM, special purpose register and embedded control register (Bank 0)

Addr.	Function	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
00	IAR0	Indirect addressing register 0							
01	MP0	Memory pointer register 0							
02	IAR1	Indirect addressing register 1							
03	MP1	Memory pointer register 1							
04	BP	Bank pointer register							
05	ACC	Accumulator							
06	PCL	Program counter low byte							
07	TBLP	Table low-order byte pointer							
08	TBLH	Table higher-order byte data							
09		Unused							
0A	STATUS	Status register							
0B	INTC	Interrupt control register							
0C	TMR0H	Timer 0 16-bit counter high byte							
0D	TMR0L	Timer 0 16-bit counter low byte							
0E	TMR0C	Timer 0 control register							
0F		unused							
10	TMR1	Timer 1 8-bit counter register							
11	TMR1C	Timer 1 control register							
12	PA	Port A							
13	PAC	Port A control							
14	PB	Port B							
15	PBC	Port B control							
16	PC	Port C							
17	PCC	Port C control							
18	PD	Port D							
19	PDC	Port D control							
1A~1F		unused							
20	DTMF GEN	TR4 (r/w)	TR3 (r/w)	TR2 (r/w)	TR1 (r/w)	TC4 (r/w)	TC3 (r/w)	TC2 (r/w)	TC1 (r/w)

Addr.	Function	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
21	STATUS 1		HDO (ro)	HFO (ro)	HKB (ro)	OPTR (r/w)		SPUP (r/w)	TP (r/w)
22	MODE & M/B	PDRESET (r/w)	PDSET (ro)	MBO (ro)	MBEN (r/w)	MBCK (r/w)	MODE (ro)	MDEN (r/w)	MDCK (r/w)
23~3F		unused							
40~CF		General purpose RAM space							

Notes: All the undefined bits [20~22] are read only and read as "0"
(r/w) means that this is a readable and writeable bit
(ro) means that this is a read only bit

General purpose RAM (BANK 1)

Addr.	Function	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
40~8F		General purpose RAM space.							

Bank pointer register

There are two general purpose RAM spaces within the HT99U210. BP (04H) is used to choose the memory bank area. When BP=0 the bank area is Bank 0 RAM addressed from 40H to CFH. When BP=1 the bank area is Bank 1 RAM addressed from 40H to 80H.

Accumulator

The accumulator is closely related to ALU operations. It is also mapped to location 05H of the data memory and can operate with immediate data. All data movements between two data memory locations must pass through the accumulator.

Arithmetic and logic unit – ALU

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

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

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

Status register – STATUS

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

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

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

In addition, on entering the interrupt sequence or executing the subroutine call, the status register will not be pushed onto the stack automatically. If the contents of the status are important and if the

subroutine can corrupt the status register, precautions must be taken to save it properly.

Dialer mode interrupt

The HT99U210 provides a dialer mode interrupt and internal timer interrupts. The dialer mode interrupt contains three interrupt sources. These are $\overline{\text{HKS}}$, HFI, $\overline{\text{HDI}}$, they are all mapped to the same interrupt vector (04H). The interrupt control register (INTC;0BH) contains the interrupt control bits to set the enable/disable and the interrupt request flags.

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

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

When the $\overline{\text{HKS}}$: rising/falling edges, HFI: rising edge and $\overline{\text{HDI}}$: falling edge, interrupts can occur and the related interrupt request flag (EIF; bit 4 of INTC) will be set. When the interrupt is enabled, and the stack is not full and the dialer mode interrupt is active, a subroutine call to location 04H will occur. The interrupt request flag (EIF) and EMI bits will be cleared to disable other interrupts.

The internal Timer 0 interrupt is initialized when the Timer 0 interrupt request flag (T0F; bit 5 of

Labels	Bits	Function
C	0	C is set if the operation results in a carry during an addition operation or if a borrow does not take place during a subtraction operation; otherwise C is cleared. C is also affected by a rotate through carry instruction.
AC	1	AC is set if the operation results in a carry out of the low nibbles in addition or no borrow from the high nibble into the low nibble in subtraction; otherwise AC is cleared.
Z	2	Z is set if the result of an arithmetic or logic operation is zero; otherwise Z is cleared.
OV	3	OV is set if the operation results in a carry into the highest-order bit but not a carry out of the highest-order bit, or vice versa; otherwise OV is cleared.
PD	4	PD is cleared when either a system power-up or executing the CLR WDT instruction. PD is set by executing the HALT instruction.
TO	5	TO is cleared by a system power-up or executing the CLR WDT or HALT instruction. TO is set by a WDT time-out.
—	6	Undefined, read as 0
—	7	Undefined, read as 0

STATUS register

the INTC) is set. This happens when a Timer 0 overflow occurs. When the interrupt is enabled, and the stack is not full and the T0F bit is set, a subroutine call to location 08H will occur. The related interrupt request flag (T0F) will be reset and the EMI bit cleared to disable further interrupts.

The Timer 1 interrupt operates in the same manner as Timer 0. The related interrupt control bits ET1I and T1F of Timer 1 are bit 3 and bit 6 of the INTC respectively.

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

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

No.	Interrupt Source	Priority	Vector
a	Dialer Mode Interrupt	1	04H
b	Timer 0 Overflow	2	08H
c	Timer 1 Overflow	3	0CH

The Timer 0/1 interrupt request flag (T0F/T1F), External interrupt request flag (EIF), Enable Timer 0/1 bit (ET0I/ET1I), Enable external interrupt bit (EEI) and Enable master interrupt bit (EMI) constitute an interrupt control register (INTC) which is located at 0BH in the data memory. EMI, EEI, ET0I, ET1I are used to control the enabling/disabling of interrupts. These bits prevent the requested interrupt from being serviced. Once the interrupt request flags (T0F, T1F, EIF) are set, they will remain in the INTC register until the interrupts are serviced or cleared by a software instruction.

It is recommended that a “CALL subroutine” is not used within the interrupt subroutine. Interrupts often occur in an unpredictable manner or need to be serviced immediately in some applications. In this case, if only one stack is left and enabling the interrupt is not well controlled, a

Register	Bit No.	Label	Function
INTC (0BH)	0	EMI	Controls the master (global) interrupt (1=enabled; 0=disabled)
	1	EEI	Controls the dialer mode interrupt (1=enabled; 0=disabled)
	2	ET0I	Controls the Timer 0 interrupt (1=enabled; 0=disabled)
	3	ET1I	Controls the Timer 1 interrupt (1=enabled; 0=disabled)
	4	EIF	Dialer mode interrupt request flag (1=active; 0=inactive)
	5	T0F	Internal Timer 0 request flag (1=active; 0=inactive)
	6	T1F	Internal Timer 1 request flag (1=active; 0=inactive)
	7	—	Unused bit, is read as 0

INTC register

"CALL subroutine" operating within the interrupt subroutine can upset the original control sequence.

Watchdog timer – WDT

The clock source of the WDT is implemented by using an external 32768Hz crystal oscillator. This timer is designed to prevent software malfunctions or the program sequence from jumping to an unknown location with unpredictable results. The watchdog timer can be disabled by mask option. If the watchdog timer is disabled, all the executions related to the WDT result in no operation.

A WDT overflow under normal operation will initialize a "chip reset" and set the status bit "TO". In the HALT mode, the overflow will initialize a "warm reset". Here, only the PC and SP are reset to zero. To clear the contents of the WDT, three methods are adopted; external reset (dialer mode reset), software instruction(s), or a HALT instruction. There are two types of software instruction(s). One type is the single instruction "CLR WDT", the other type comprises two instructions, "CLR WDT1" and "CLR WDT2". Of these two types of instruction, only one can be active depending on the mask option – "CLR WDT times selection option". If the "CLR WDT" is selected (i.e.. CLR WDT times equal one), any execution of the CLR WDT instruction will clear the WDT. In case "CLR WDT1" and "CLR WDT2" are chosen (i.e.. CLRWDT times equal two), these two instructions must be executed to clear the WDT; otherwise, the WDT may reset the chip due to a time-out.

Power down operation – HALT

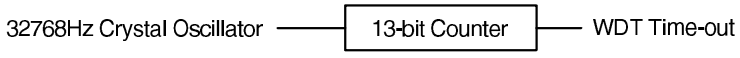
The HALT mode is initialized by the HALT instruction and results in the following...

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

The system can leave the HALT mode by means of an external reset, an interrupt, an external falling edge signal on port A or a WDT overflow. An external reset causes a device initialization and the WDT overflow performs a "warm reset". Examining the TO and PD flags, the reason for chip reset can be determined. The PD flag is cleared when the system powers-up or executing the CLR WDT instruction and is set when the HALT instruction is executed. The TO flag is set if the WDT time-out occurs, and causes a wake-up that only resets the PC and SP, the others keep their original status.

The port A wake-up and interrupt methods can be considered as a continuation of normal execution. Each bit in port A can be independently selected to wake-up the device by mask option. Awakening from an I/O port stimulus, the program will resume execution of the next instruction. If awakening from an interrupt, two sequences may happen. If the related interrupts are disabled or the interrupts are enabled but the stack is full, the program will resume execution at the next instruction. If the interrupt is enabled and the stack is not full, the regular interrupt response takes place.

Once the wake-up event(s) occurs, and if the system clock comes from a crystal, it takes 1024 t_{SYS} (system clock period) to resume normal operation. In other words, the HT99U210 will insert a dummy period after the wake-up. If the system clock comes from an RC oscillator, it continues operating immediately. If the wake-up results from an interrupt acknowledgment, the actual interrupt subroutine execution will be delayed by one more cycle. If the wake-up



Watchdog timer

results in next instruction execution, this will execute immediately after the dummy period has finished.

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

Reset

There are 2 rest sources in the dialer mode and three ways in which a reset can occur:

- $\overline{\text{HKS}}$: Off-hook state
- HFI: Hand-free on state
- Reset during normal operation
- Reset during HALT
- WDT time-out reset during normal operation

The WDT time-out during a HALT is different from other chip reset conditions, since it can perform a warm reset that just resets PC and SP, leaving the other circuits to maintain their original state. Some registers remain unchanged during other reset conditions. Most registers are reset to the "initial condition" when the reset conditions are met. By examining the PD flag and TO flag, the program can distinguish between different "chip resets".

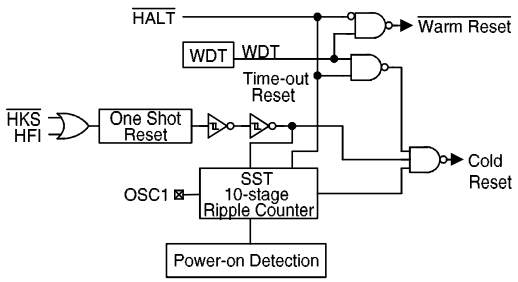
TO	PD	RESET Conditions
0	0	Reset during power-up
u	u	Reset during normal operation
0	1	Reset wake-up HALT
1	u	WDT time-out during normal operation
1	1	WDT wake-up HALT

Note: "u" means "unchanged"

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

When a system power-up occurs, the SST delay is added during the reset period.

The functional unit chip reset status are shown below.



Reset configuration

PC	000H
Dialer mode interrupt	Disable
Prescaler	Clear
WDT	Clear. After master reset, WDT begins counting
Timer (0/1)	Off
Input/output ports	Input mode
SP	Points to the top of the stack

Timer

Two Timers are implemented in the HT99U210. The Timer 0 and Timer 1 contain 16-bit and 8-bit programmable count-up counters respectively. The clock source can be selected to be either 32768Hz time base or 1/4 system clock by TMR0C and TMRIC timer control registers.

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

Reading TMR0H will also latch the TMR0L into the low byte buffer to avoid the false timing problem. Reading TMR0L returns the content of the low byte buffer. In other words, the low byte of Timer 0 can not be read directly. It must read the TMR0H first to latch the low byte contents of Timer 0 latched into the buffer.

There are two registers related to Timer 1; TMR1 (10H), TMR1C (11H). Writing to TMR1 places the starting value in the Timer 1 Preload register and reading TMR1 gets the contents of the Timer 1 Preload register.

TMR0C is the Timer 0 control register, which defines the Timer 0 options. Timer 1 has the same options as the Timer 0 and is defined by TMR1C.

The Timer control registers define the operating mode, counting enable or disable and active edge.

The TM0, TM1 bits define the operating mode.

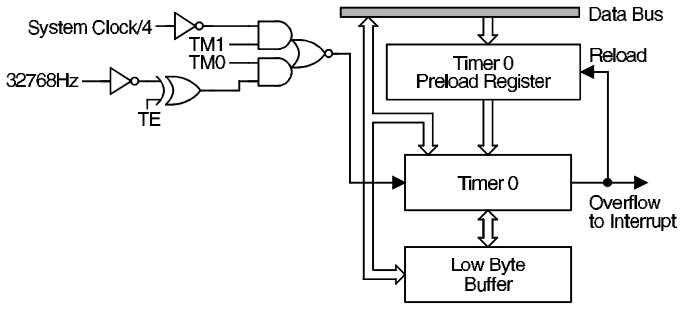
In the Timer mode, once the Timer starts counting, it will count from the current contents in the Timer to FFFFH (TMR0)/FFH (TMR1). Once overflow occurs, the counter is reloaded from the Timer Preload register and generates the corresponding interrupt request flag (T0F/T1F; bit 5/6 of INTC) at the same time.

To enable the counting operation, the Timer ON bit (TON; bit 4 of TMR0C/TMR1C) should be set to 1. In the Timer mode, TON can only be reset by instruction. The overflow of the Timer is one of the wake-up sources. No matter what the operation mode is, writing a 0 to ET0I/ET1I can disable the corresponding interrupt service.

In the case of a Timer OFF condition, writing data to the Timer Preload register will also reload that data to the Timer. But if the Timer is turned on, data written to the Timer will only be kept in the Timer Preload register.

Label	Bits	Function
—	0~2	Unused bits, read as “0”
TE	3	Defines the TMR0/TMR1 active edge of Timer (0=active on low to high; 1=active on high to low)
TON	4	Enable/disable timer counting (0=disabled; 1=enabled)
—	5	Unused bits, is read as “0”
TM0 TM1	6 7	Defines the operating mode 01=Timer mode; 32768Hz time base 10=Timer mode; 1/4 system clock 11=Unused 00=Unused

TMR0C/TMR1C register



Timer 0

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

Register	Reset (Power on)	WDT Time-out (Normal Operation)	Reset (Normal Operation)	Reset (HALT)	WDT Time-out* (HALT)
TMR1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR1C	00-0 1---	00-0 1---	00-0 1---	00-0 1---	uu-u u---
TMR0H	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR0L	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR0C	00-0 1---	00-0 1---	00-0 1---	00-0 1---	uu-u u---
PC	000H	000H	000H	000H	000H
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	-xxx xxxx	-uuu uuuu	-uuu uuuu	-uuu uuuu	-uuu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC	-000 0000	-000 0000	-000 0000	-000 0000	-uuu 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	1111 1111	1111 1111	uuuu uuuu
PCC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu

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

"x" means "unknown" "-" means "undefined"

The bits of the special function registers are denoted as "-" if they are not defined in the microcontrollers.

The Timer will still operate until the overflow occurs.

When the Timer (reading TMR0H/TMR1) is read, the clock will be blocked to avoid errors. As this may result in a counting error, this must be taken into consideration during programming.

Input/output ports

There are 26 input/output lines in the HT99U210, labeled PA, PB, PC and PD, which are mapped to the data memory of [12H], [14H], [16H] and [18H] respectively. Some I/O pins can be selected to be either in normal mode or in dialer mode by mask option. In normal mode, I/O pins can be used for input and output operations. For input operation, these ports are non-latching. This means that the inputs must be ready at the T2 rising edge of instruction MOV A,[m] (m=12H, 14H, 16H or 18H). For output operation, all data is latched and remains unchanged until the output latch is rewritten.

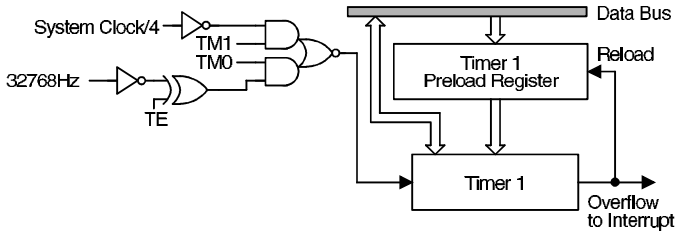
Either in normal mode or in dialer mode, the I/O line has its own control register (PAC, PBC, PCC, PDC) to control the input/output configuration. With this control register, some CMOS output or schmitt trigger input with or without pull-high resistor (mask option) structures can be reconfigured dynamically (i.e., on-the-fly) under software control. To function as an input, the corresponding latch of the control register must write "1". The pull-high resistance will be present automatically if the pull-high option is selected. The input source(s) also depend(s) on the control register. If the control register bit is "1", the input will read the pad state. If the control register bit is "0", the contents of the latches will move to the internal bus. The latter is possible using the "read-modify-write" instruction. For output function, CMOS is the only configuration. These control registers are mapped to locations 13H, 15H, 17H and 19H.

After a chip reset, these input/output lines stay at high levels or floating (mask option). Each bit of these input/output latches can be set or cleared by the SET [m].i or CLR [m].i (m=12H, 14H, 16H or 18H) instruction.

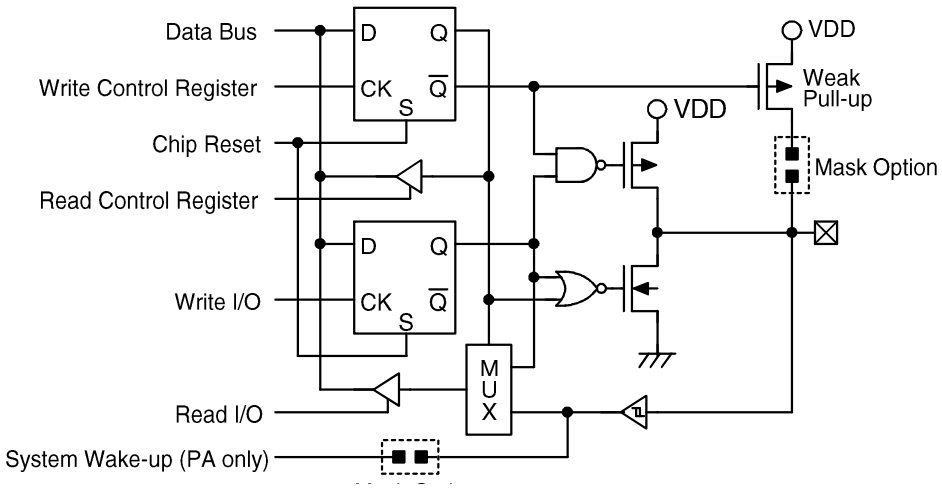
Some instructions first input data and then follow the output operations. For example, the SET [m].i, CLR [m].i, CPL [m] and CPLA [m] instructions read the entire port states into the CPU, execute the defined operations (bit-operation), and then write the results back to the latches or the accumulator.

Each line of port A has the capability to wake-up the device.

The detailed structure of the input/output are illustrated in the pad description and mask option sections.



Timer 1



Input/output ports

DTMF generator

- Using TP (21H) bit to control the DTMF generator enable or disable
- Using TC1~TC4, TR1~TR4 bits to control the DTMF generator dual tone output

Addr.	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
(Bank 0) 21H								TP (r/w)

Notes: TP=0 Disable DTMF Generator
TP=1 Enable DTMF Generator

Addr.	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
(Bank 0) 20H	TR4 (r/w)	TR3 (r/w)	TR2 (r/w)	TR1 (r/w)	TC4 (r/w)	TC3 (r/w)	TC2 (r/w)	TC1 (r/w)

DTMF frequency selection table

Low Group				High Group				Enable	DTMF Output		Key-in
TR4	TR3	TR2	TR1	TC4	TC3	TC2	TC1	TP	Low	High	
X	X	X	X	X	X	X	X	0	"X"	"X"	
0	0	0	1	0	0	0	1	1	697	1209	**1**
0	0	0	1	0	0	1	0	1	697	1336	**2**
0	0	0	1	0	1	0	0	1	697	1477	**3**
0	0	0	1	1	0	0	0	1	697	1633	**A**
0	0	1	0	0	0	0	1	1	770	1209	**4**
0	0	1	0	0	0	1	0	1	770	1336	**5**
0	0	1	0	0	1	0	0	1	770	1477	**6**
0	0	1	0	1	0	0	0	1	770	1633	**B**
0	1	0	0	0	0	0	1	1	852	1209	**7**
0	1	0	0	0	0	1	0	1	852	1336	**8**
0	1	0	0	0	1	0	0	1	852	1477	**9**
0	1	0	0	1	0	0	0	1	852	1633	**C**
1	0	0	0	0	0	0	1	1	941	1209	*****
1	0	0	0	0	0	1	0	1	941	1336	**0**
1	0	0	0	0	1	0	0	1	941	1477	**#**
1	0	0	0	1	0	0	0	1	941	1633	**D**

Hook, hand-free, and hold operation status

Addr.	Function	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
21H	STATUS1		HDO (ro)	HFO (ro)	HKB (ro)				

	1	0
HDO (ro)	Hold function is active	Hold function is not active
HFO (ro)	Hand-free function is active	Hand-free function is not active
HKB (ro)	HKS is high (assuming on-hook)	HKS is low (assuming off-hook)

M/B ratio and mode control

- M/B pad is a tristate input pin of this chip
- Tristate detect:
  - send MBCK=1, MBEN=1, read MBO=1
    send MBCK=0, MBEN=1, read MBO=0 → M/B pad is floating
  - send MBCK=1, MBEN=1, read MBO=0
    send MBCK=0, MBEN=1, read MBO=0 → M/B pad is 0
  - send MBCK=1, MBEN=1, read MBO=1
    send MBCK=0, MBEN=1, read MBO=1 → M/B pad is 1

Addr.	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
(Bank 0) 22H			MBO (ro)	MBEN (r/w)	MBCK (r/w)			

MBEN=1 M/B function Enable
MBEN=0 M/B function Disable

- MODE is a tristate input pin of this chip.
- Tristate detect:
  - send MDCK=1, MDEN=1, read MODE=1
    send MDCK=0, MDEN=1, read MODE=0 → MODE pad is floating
  - send MDCK=1, MDEN=1, read MODE=0
    send MDCK=0, MDEN=1, read MODE=0 → MODE pad is 0
  - send MDCK=1, MDEN=1, read MODE=1
    send MDCK=0, MDEN=1, read MODE=1 → MODE pad is 1

Addr.	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
(Bank 0) 22H						MODE (ro)	MDEN (r/w)	MDCK (r/w)

MDEN (r/w): MDEN=1 MODE function Enable
MDEN=0 MODE function Disable

OSC32K

The 32768Hz oscillator must be brought up to speed automatically after the telephone line is plugged in and an off-hook situation occurs.

Addr.	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
(Bank 0) 21H							SPUP (r/w)	

SPUP (r/w): SPUP=0 DO NOT SPEED UP OSC32k
SPUP=1 MEANS SPEED UP OSC32k

Power down state control

When a power down state occurs, the power down bit (PDSET) is always set to "1". When the PDRESET bit is set to "1" then the PDSET bit is changed to "0".

Addr.	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
(Bank 0) 22H	PDRESET (r/w)	PDSET (ro)						

PDRESET (r/w): PDRESET=0, PDSET=0 not power on initial
PDRESET=0, PDSET=1 power on initial
PDRESET=1, PDSET=0 power on initial active OK
PDRESET=1, PDSET=1 power on initial active is not OK

Keyboard scan register control

Using OPTR bit to control keyboard resistor scan function enable or disable.

Addr.	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
(Bank 0) 21H					OPTR (r/w)			

OPTR (r/w): OPTR=0 Disable scan keyboard resistor function
OPTR=1 Enable scan keyboard resistor function

Mask option

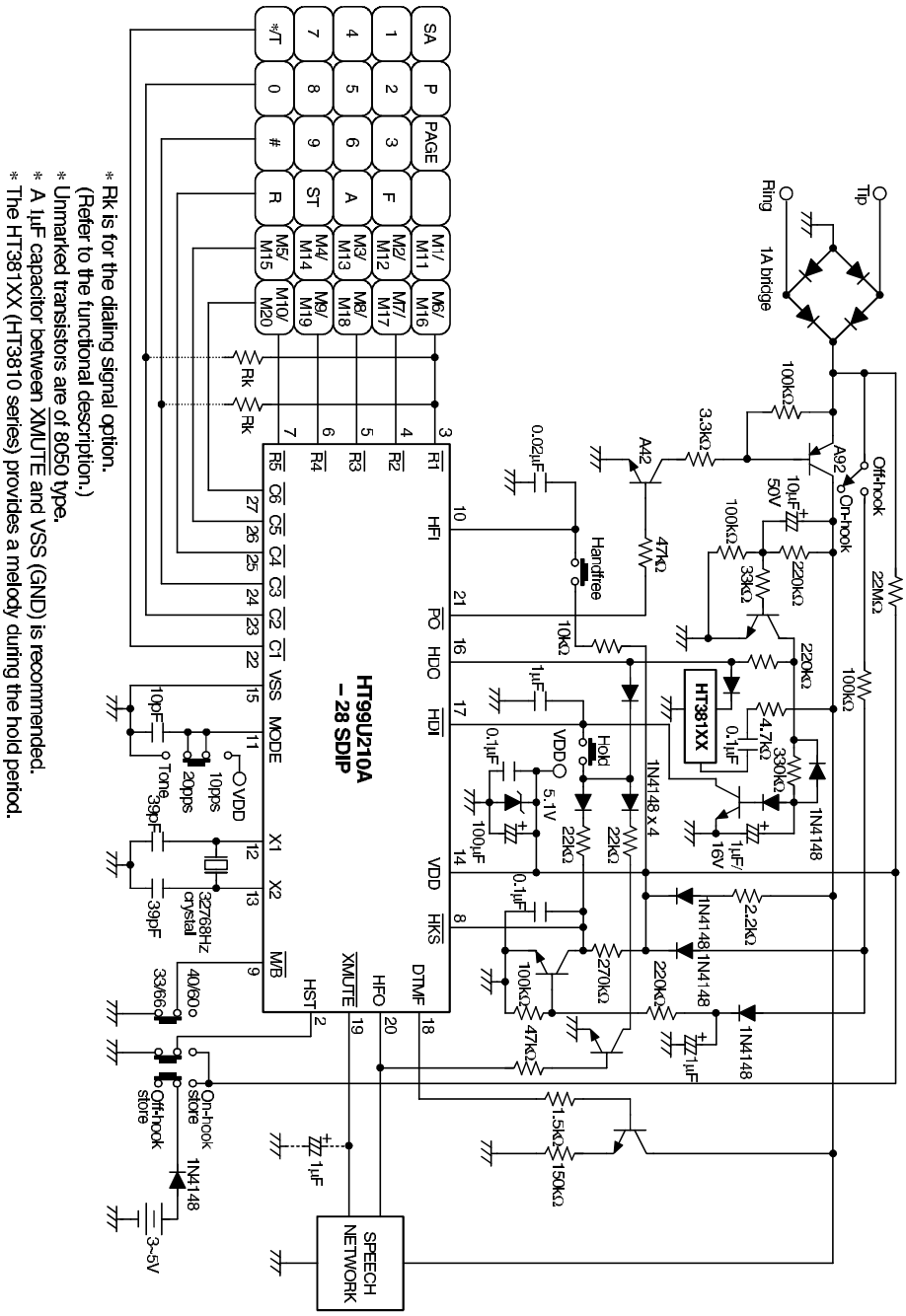
The following shows the many kinds of mask option within the HT99U210. All the mask options must be defined to ensure proper system function.

No.	Pad Name	Mask Option
1	PA0 ($\overline{C1}$)	Wake-up type or none wake-up type.
2	PA1 ($\overline{C2}$)	Wake-up type or none wake-up type.
3	PA2 ($\overline{C3}$)	Wake-up type or none wake-up type.
4	PA3 ($\overline{C4}$)	Wake-up type or none wake-up type.
5	PA4 ($\overline{C5}$)	Wake-up type or none wake-up type. Dialer type or Normal type. Bidirectional I/O with pull-up PMOS pin or general I/O pin without pull-up PMOS.
6	PA5 ($\overline{C6}$)	Wake-up type or none wake-up type. Dialer type or Normal type. Bidirectional I/O with pull-up PMOS pin or general I/O pin without pull-up PMOS.
7	PA6 ($\overline{C7}$)	Wake-up type or none wake-up type. Dialer type or Normal type. Bidirectional I/O with pull-up PMOS pin or general I/O pin without pull-up PMOS.

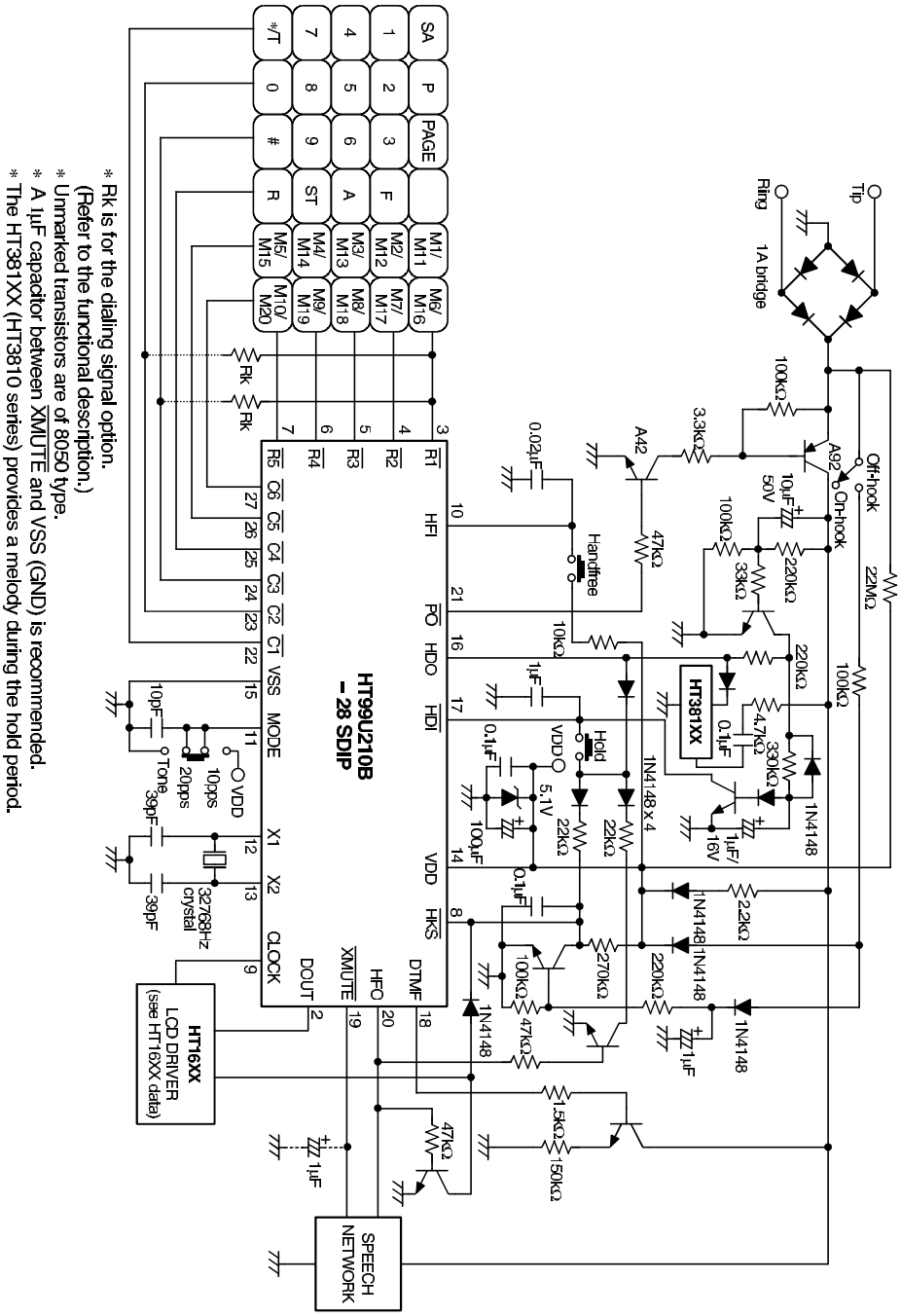
No.	Pad Name	Mask Option
8	PA7 ($\overline{C8}$)	Wake-up type or none wake-up type. Dialer type or Normal type. Bidirectional I/O with pull-up PMOS pin or general I/O pin without pull-up PMOS.
9	PB4 ($\overline{R5}$)	Dialer type or Normal type. Bidirectional I/O with pull-up PMOS pin or general I/O pin without pull-up PMOS.
10	PC0 (HDO)	Dialer type or Normal type. Bidirectional I/O with pull-up PMOS pin or general I/O pin without pull-up PMOS.
11	PC1 ($\overline{HDI}$)	Dialer type or Normal type. Bidirectional I/O with pull-up PMOS pin or general I/O pin without pull-up PMOS.
12	PC3 (HFO)	Dialer type or Normal type. Bidirectional I/O with pull-up PMOS pin or general I/O pin without pull-up PMOS.
13	PC4 ($\overline{PO}$)	Dialer type or Normal type. Bidirectional I/O with pull-up PMOS pin or general I/O pin without pull-up PMOS.
14	PD0 (KT)	Bidirectional I/O with pull-up PMOS pin or general I/O pin without pull-up PMOS.
15	PD2 ($\overline{HKS}$)	Dialer type or Normal type. Bidirectional I/O with pull-up PMOS pin or general I/O pin without pull-up PMOS.
16	PD4 ($\overline{M/B}$)	Dialer type or Normal type. Bidirectional I/O with pull-up PMOS pin or general I/O pin without pull-up PMOS.
17	PD5 (HFI)	Dialer type or Normal type. Bidirectional I/O with pull-up PMOS pin or general I/O pin without pull-up PMOS.
18	PD6 (MODE)	Dialer type or Normal type. Bidirectional I/O with pull-up PMOS pin or general I/O pin without pull-up PMOS.
19	ROSC	RC system clock can be selected to 256kHz, 512kHz, 1MHz and 1.5MHz.

Application Circuits (Wired): Dialer Mode I/O Structure

Application circuit 1

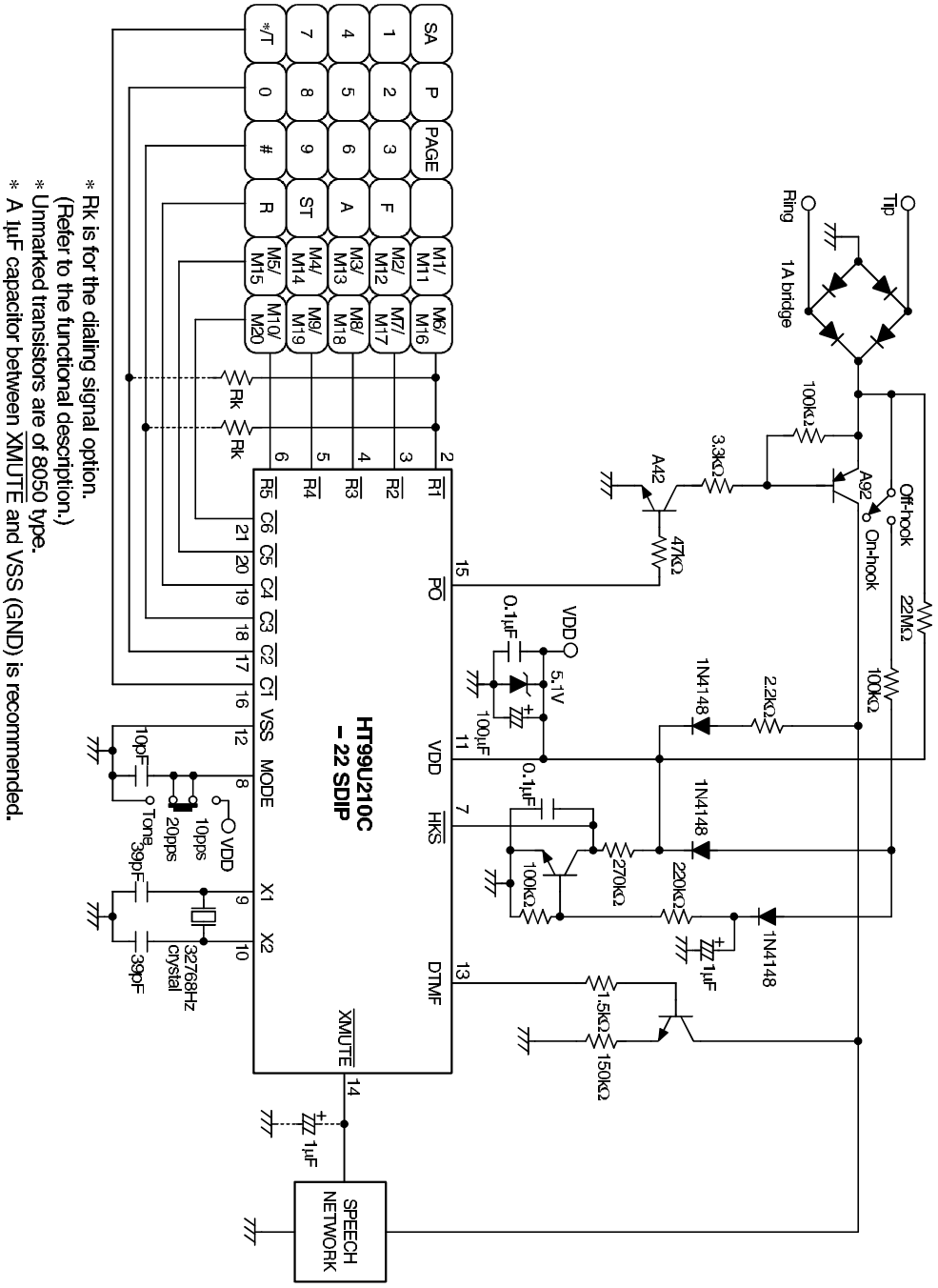


Application circuit 2



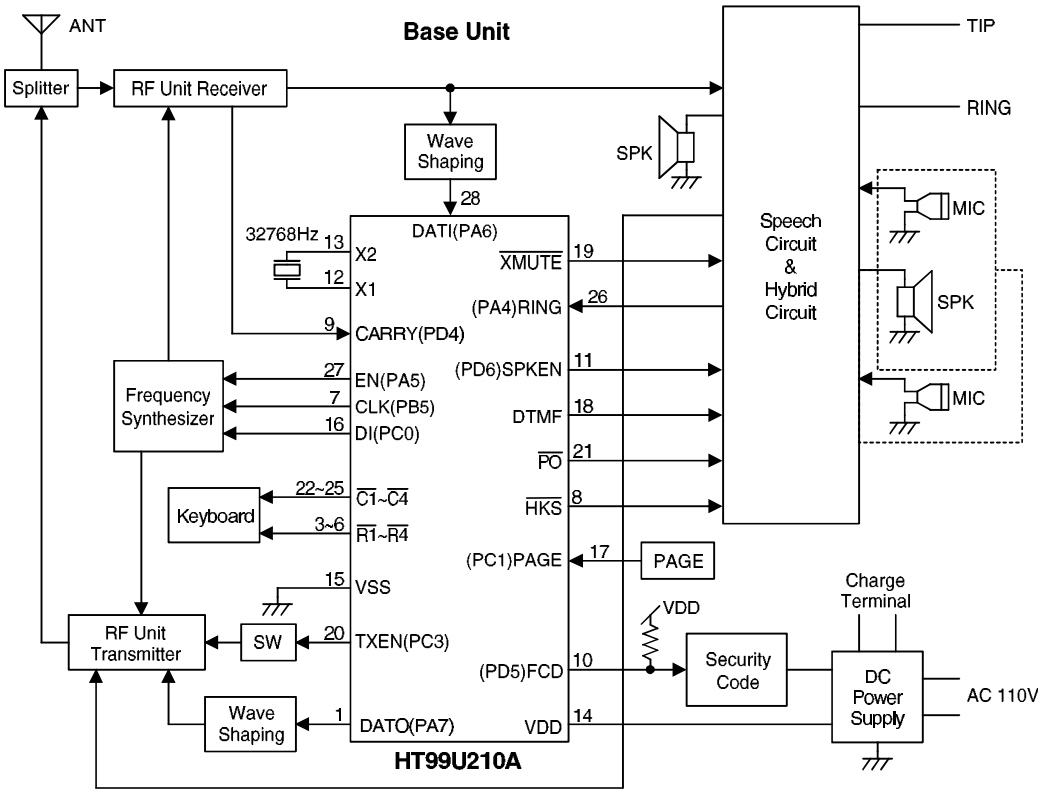
* Rk is for the dialing signal option. (Refer to the functional description.)
* Unmarked transistors are of 8050 type.
* A 1μF capacitor between XMUTE and VSS (GND) is recommended.
* The HT3810XX (HT3810 series) provides a melody during the hold period.

Application circuit 3



* Rk is for the dialing signal option.
(Refer to the functional description.)
* Unmarked transistors are of 8050 type.
* A 1μF capacitor between $\overline{XMUTE}$ and VSS (GND) is recommended.

Application Circuits (Cordless): Dialer Mode with Normal Mode I/O Structure



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, result in data memory	Z,C,AC,OV
DAA [m]	Decimal adjust ACC for addition with result in data memory	C
Logic Operation		
AND A,[m]	AND data memory to ACC	Z
OR A,[m]	OR data memory to ACC	Z
XOR A,[m]	Exclusive-OR data memory to ACC	Z
ANDM A,[m]	AND ACC to data memory	Z
ORM A,[m]	OR ACC to data memory	Z
XORM A,[m]	Exclusive-OR ACC to data memory	Z
AND A,x	AND immediate data to ACC	Z
OR A,x	OR immediate data to ACC	Z
XOR A,x	Exclusive-OR immediate data to ACC	Z
CPL [m]	Complement data memory	Z
CPLA [m]	Complement data memory with result in ACC	Z
Increment and Decrement		
INCA [m]	Increment data memory with result in ACC	Z
INC [m]	Increment data memory	Z
DECA [m]	Decrement data memory with result in ACC	Z
DEC [m]	Decrement data memory	Z
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 the watchdog timer	TO,PD
CLR WDT1	Pre-clear the watchdog timer	TO*,PD*
CLR WDT2	Pre-clear the 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
addr: 12-bit program memory address
√: Flag(s) is affected
–: Flag(s) is not affected
*: Flag(s) may be affected by the execution status

Instruction Definition

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

Description The contents of the specified data memory, accumulator and the carry flag are added simultaneously, leaving the result in the accumulator.

Operation ACC ← ACC+[m]+C

Affected flag(s)

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

ADCM A,[m] Add accumulator and carry to data memory

Description The contents of the specified data memory, accumulator and the carry flag are added simultaneously, leaving the result in the specified data memory.

Operation [m] ← ACC+[m]+C

Affected flag(s)

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

ADD A,[m] Add data memory to accumulator

Description The contents of the specified data memory and the accumulator are added. The result is stored in the accumulator.

Operation ACC ← ACC+[m]

Affected flag(s)

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

ADD A,x Add immediate data to accumulator

Description The contents of the accumulator and the specified data are added, leaving the result in the accumulator.

Operation ACC ← ACC+x

Affected flag(s)

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

ADDM A,[m] Add accumulator to data memory

Description The contents of the specified data memory and the accumulator are added. The result is stored in the data memory.

Operation [m] ← ACC+[m]

Affected flag(s)

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

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

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

Operation ACC ← ACC "AND" [m]

Affected flag(s)

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

AND A,x Logical AND immediate data to accumulator

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

Operation ACC ← ACC "AND" x

Affected flag(s)

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

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

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

Operation [m] ← ACC "AND" [m]

Affected flag(s)

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

CALL addr Subroutine call

Description The instruction unconditionally calls a subroutine located at the indicated address. The program counter increments once to obtain the address of the next instruction, and pushes this onto the stack. The indicated address is then loaded. Program execution continues with the instruction at this address.

Operation Stack ← PC+1
PC ← addr

Affected flag(s)

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

CLR [m] Clear data memory

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

Operation [m] ← 00H

Affected flag(s)

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

CLR [m].i Clear bit of data memory

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

Operation [m].i ← 0

Affected flag(s)

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

CLR WDT Clear the watchdog timer

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

Operation WDT and WDT Prescaler ← 00H
PD and TO ← 0

Affected flag(s)

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

CLR WDT1 Preclear the watchdog timer

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

Operation WDT and WDT Prescaler ← 00H*
PD and TO ← 0*

Affected flag(s)

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

CLR WDT2 Preclear the watchdog timer

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

Operation WDT and WDT Prescaler ← 00H*
PD and TO ← 0*

Affected flag(s)

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

CPL [m] Complement data memory

Description Each bit of the specified data memory is logically complemented (1's complement). Bits which previously contain a one are changed to zero and vice-versa.

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

Affected flag(s)

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

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

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

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

Affected flag(s)

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

DAA [m] Decimal-Adjust accumulator for addition

Description The accumulator value is adjusted to the BCD (Binary Code Decimal) code. The accumulator is divided into two nibbles. Each nibble is adjusted to BCD code and an internal carry (AC1) will be generated 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 one.

Operation [m] ← [m]–1

Affected flag(s)

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

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

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

Operation ACC ← [m]–1

Affected flag(s)

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

HALT Enter power down mode

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

Operation PC ← PC+1
PD ← 1
TO ← 0

Affected flag(s)

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

INC [m] Increment data memory

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

Operation [m] ← [m]+1

Affected flag(s)

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

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

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

Operation ACC ← [m]+1

Affected flag(s)

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

JMP addr Direct Jump

Description Bits 0~11 of the program counter are replaced with the directly–specified address unconditionally, and control passed to this destination.

Operation PC ← addr

Affected flag(s)

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

MOV A,[m] Move data memory to accumulator

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

Operation ACC ← [m]

Affected flag(s)

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

MOV A,x Move immediate data to accumulator

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

Operation ACC ← x

Affected flag(s)

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

MOV [m],A Move accumulator to data memory

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

Operation [m] ← ACC

Affected flag(s)

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

NOP No operation

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

Operation PC ← PC+1

Affected flag(s)

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

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

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

Operation ACC ← ACC "OR" [m]

Affected flag(s)

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

OR A,x Logical OR immediate data to accumulator

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

Operation ACC ← ACC "OR" x

Affected flag(s)

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

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

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

Operation [m] ← ACC "OR" [m]

Affected flag(s)

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

RET Return from subroutine

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

Operation PC ← Stack

Affected flag(s)

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

RET A,x Return and place immediate data in accumulator

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

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 the interrupts are enabled by setting the EMI bit. EMI is the enable master (global) interrupt bit (bit 0; register INTC).

Operation PC ← Stack
EMI ← 1

Affected flag(s)

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

RL [m] Rotate data memory left

Description The contents of the specified data memory is rotated left one bit, with bit 7 rotated into bit 0.

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

Affected flag(s)

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

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

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

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

Affected flag(s)

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

RLC [m] Rotate data memory left through carry

Description The contents of the specified data memory and the carry flag are together rotated left one bit. Bit 7 replaces the carry bit; the original carry flag is rotated into the bit 0 position.

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

Affected flag(s)

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

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

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

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

Affected flag(s)

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

RR [m] Rotate data memory right

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

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

Affected flag(s)

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

RRA [m] Rotate right and place result in accumulator

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

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

Affected flag(s)

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

RRC [m] Rotate data memory right through carry

Description The contents of the specified data memory and the carry flag are together rotated right one bit. Bit 0 replaces the carry bit. The original carry flag is rotated into the bit 7 position.

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

Affected flag(s)

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

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

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

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

Affected flag(s)

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

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

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

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

Affected flag(s)

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

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

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

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

Affected flag(s)

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

SDZ [m] Skip if decrement data memory is zero

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

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

Affected flag(s)

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

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

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

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

Operation [m] ← FFH

Affected flag(s)

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

SET [m].i Set bit of data memory

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

Operation [m].i ← 1

Affected flag(s)

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

SIZ [m] Skip if increment data memory is zero

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

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

Description The low-order and high-order nibbles of the specified data memory are interchanged, writing the result to the accumulator. The contents of the data memory remain unchanged.

Operation ACC.3~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0

Affected flag(s)

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

SZ [m] Skip if data memory is zero

Description If the contents of the specified data memory is zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction. This is a 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 data memory to ACC, skip if zero

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

Operation Skip if [m]=0

Affected flag(s)

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

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

Description If bit i of the specified data memory is zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction. This is a 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 ROM code (current page) to TBLH and data memory

Description The low byte of ROM code (current page) addressed by the table pointer (TBLP) is moved to the specified data memory and the high byte transferred to TBLH directly.

Operation [m] ← ROM code (low byte)
TBLH ← ROM code (high byte)

Affected flag(s)

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

TABRDL [m] Move ROM code (last page) to TBLH and data memory

Description The low byte of ROM code (last page) addressed by the table pointer (TBLP) is moved to the data memory and the high byte transferred to TBLH directly.

Operation [m] ← ROM code (low byte)
TBLH ← ROM code (high byte)

Affected flag(s)

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

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

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

Operation ACC ← ACC “XOR” [m]

Affected flag(s)

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

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

Description Data in the indicated data memory and the accumulator perform a bitwise logical Exclusive_OR operation. The result is stored in the data memory. The zero flag is affected.

Operation [m] ← ACC "XOR" [m]

Affected flag(s)

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

XOR A,x Logical XOR immediate data to accumulator

Description Data in the 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
–	–	–	–	–	√	–	–