HT49RU80/HT49CU80
LCD Type 8-Bit MCU

Technical Document

- Tools Information
- FAQs
- Application Note
  - HA0017E Controlling the Read/Write Function of the HT24 Series EEPROM Using the HT49 Series MCUs
  - HA0024E Using the RTC in the HT49 MCU Series
  - HA0025E Using the Time Base in the HT49 MCU Series
  - HA0026E Using the I/O Ports on the HT49 MCU Series
  - HA0027E Using the Timer/Event Counter in the HT49 MCU Series

Features

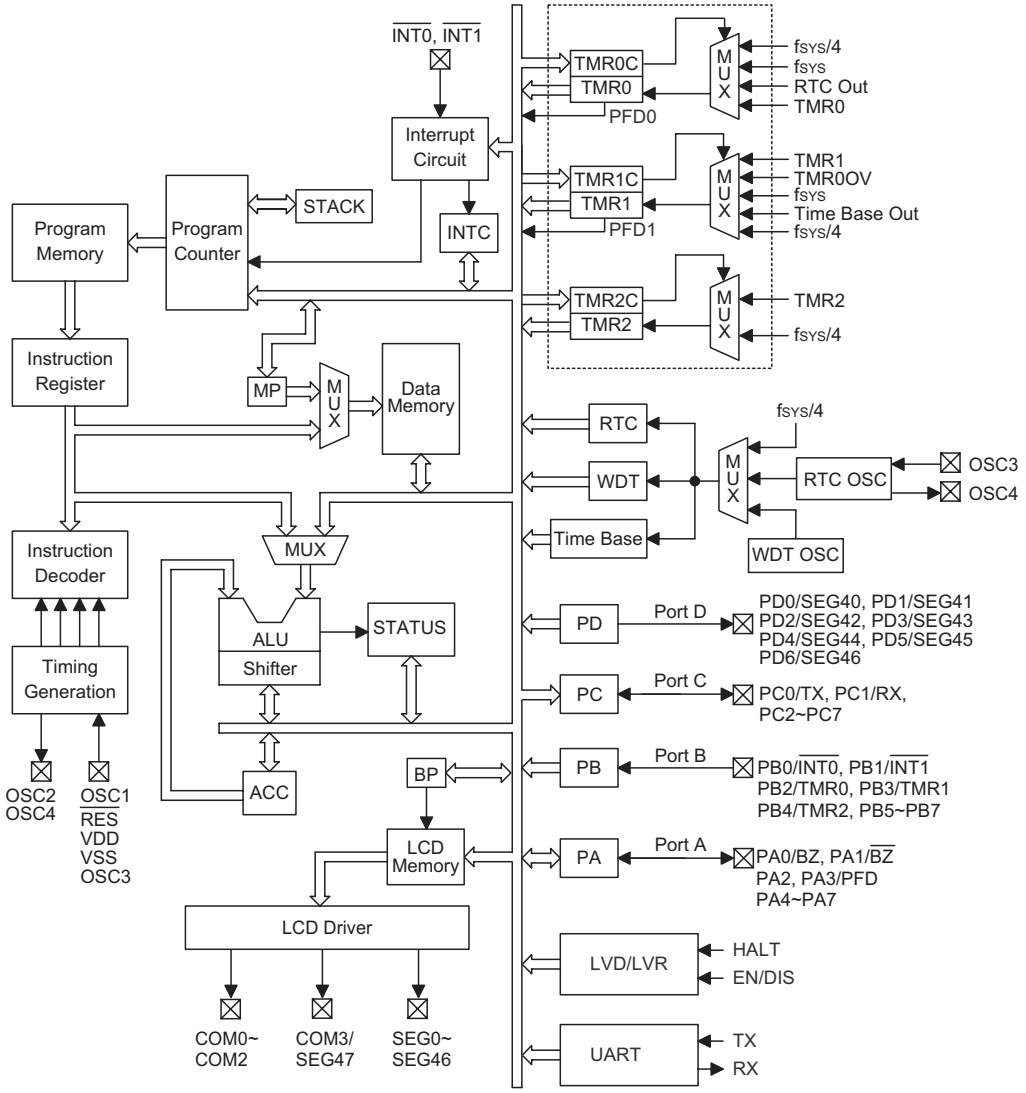
- Operating voltage:
  f_{SYS}=4MHz: 2.2V~5.5V
  f_{SYS}=8MHz: 3.3V~5.5V
- 8 input lines and 7 output lines
- 16 bidirectional I/O lines
- Two external interrupt inputs
- One 8-bit and two 16-bit programmable timer/event counters with PFD - programmable frequency divider function
- LCD driver with 48×2, 48×3 or 47×4 segments
- 16K×16 program memory
- 576×8 data memory RAM
- Real Time Clock - RTC
- RTC 8-bit prescaler
- Watchdog Timer
- Buzzer output
- On-chip crystal, RC and 32768Hz crystal oscillator
- HALT function and wake-up feature reduce power consumption
- 16-level subroutine nesting
- UART - Universal Asynchronous Receiver Transmitter
- Bit manipulation instruction
- 16-bit table read instruction
- Up to 0.5μs instruction cycle with 8MHz system clock
- 63 powerful instructions
- All instructions executed within 1 or 2 machine cycles
- Low voltage reset/detector functions
- 100-pin QFP package

General Description

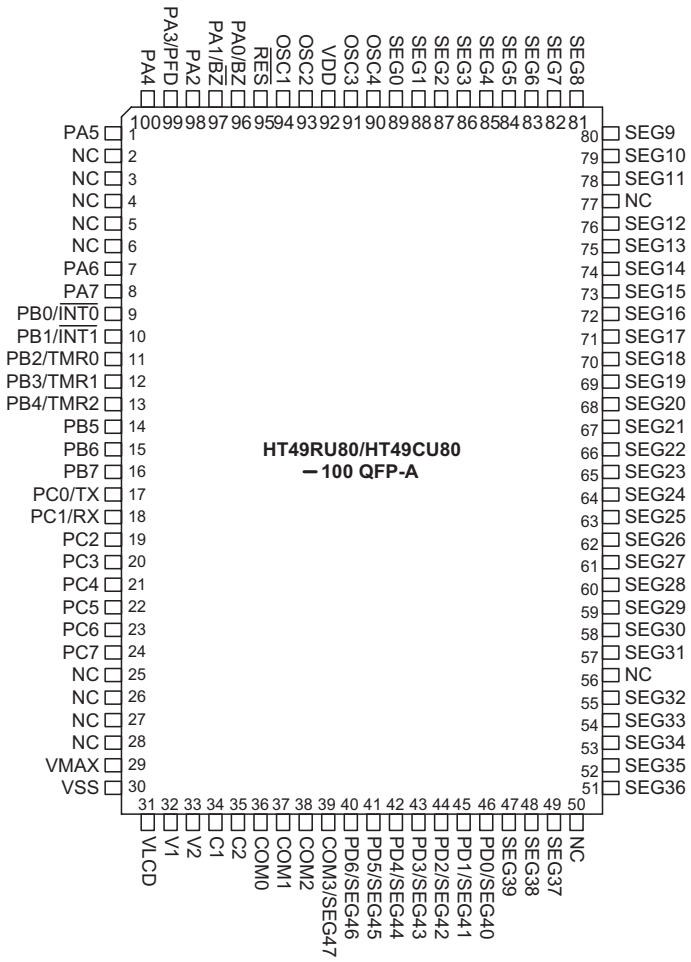
These devices are 8-bit, high performance, RISC architecture microcontrollers specifically designed for a wide range of LCD applications. The mask version, the HT49CU80, is fully pin and functionally compatible with the OTP version HT49RU80 device.

The advantages of low power consumption, I/O flexibility, programmable frequency divider, timer functions, oscillator options, power-down and wake-up functions and buzzer driver in addition to a flexible and configurable LCD interface, enhance the versatility of these devices to control a wide range of LCD-based application possibilities such as measuring scales, electronic multimeters, gas meters, timers, calculators, remote controllers and many other LCD-based industrial and home appliance applications.

Block Diagram



Pin Assignment



Pad Description

Pad Name	I/O	Options	Description
PA0/BZ PA1/$\overline{BZ}$ PA2 PA3/PFD PA4~PA7	I/O	Wake-up CMOS or NMOS Pull-high PA0/PA1 or BZ/$\overline{BZ}$ PA3 or PFD	Bidirectional 8-bit input/output port. Each pin on this port can be configured as a wake-up input by a configuration option. Configuration options determine whether pins PA0~PA3 are configured as CMOS outputs or NMOS input/output pins. If PA0~PA3 are configured as NMOS input/output pins, then pull-high options are available but apply to all 4 pins, not individual pins. Pins PA4~PA7 are always configured as NMOS input/output pins with pull-high resistors connected. All inputs are Schmitt Trigger types. Pins PA0, PA1 and PA3 are pin-shared with BZ, $\overline{BZ}$ and PFD respectively, the function of which is chosen via configuration options.
PB0/$\overline{INT0}$ PB1/$\overline{INT1}$ PB2/TMR0 PB3/TMR1 PB4/TMR2 PB5~PB7	I	—	8-bit Schmitt Trigger input port. Each input pin is connected to an internal pull-high resistor. Pins PB0 and PB1 are pin-shared with $\overline{INT0}$ and $\overline{INT1}$ respectively. Pins PB2, PB3 and PB4 are pin-shared with TMR0, TMR1 and TMR2 respectively.
PC0/TX PC1/RX PC2~PC7	I/O	CMOS or NMOS Pull-high	Bidirectional 8-bit input/output port. Two configuration options determine whether the four pins PC0~PC3 and the four pins PC4~PC7 are configured as CMOS outputs or NMOS input/output pins. Pins must be configured as CMOS outputs or NMOS input/output pins in blocks of four pins, individual pins cannot be selected. If pins PC0~PC3 or PC4~PC7 are configured as NMOS input/output pins, then a pull-high option is available for each block of four pins. Individual pins cannot be selected to have a pull-high option. All inputs are Schmitt Trigger types. Pins PC0 and PC1 are pin-shared with UART pins TX and RX respectively.
PD0/SEG40~ PD6/SEG46	O	CMOS Output or SEG Output	7-bit output port. Each pin can be setup as either a CMOS output or a SEG output via configuration options.
VLCD	I	—	LCD power supply. This pad is implemented for LCD power only. The VLCD levels can be greater or less than the VDD levels.
VMAX	—	—	IC maximum voltage, connect to VDD, VLCD or V1.
V1, V2, C1, C2	I	—	LCD voltage pump
COM0~COM2 COM3/SEG47	O	1/2, 1/3 or 1/4 Duty	The 1/4 LCD duty cycle configuration option will determine whether pin COM3/SEG47 is configured as a SEG47 segment driver or as a common COM3 output driver for the LCD panel. COM0~COM2 are the LCD common outputs.
SEG0~SEG39	O	—	LCD driver outputs for LCD panel segments
OSC1 OSC2	I O	Crystal or RC	OSC1 and OSC2 are connected to an external RC network or external crystal (determined by configuration option) for the internal system clock. For external RC system clock operation, OSC2 is an output pin for 1/4 system clock. If an RTC oscillator on pins OSC3 and OSC4 is used as a system clock, then the OSC1 and OSC2 pins should be left floating.
OSC3 OSC4	I O	RTC or System Clock	OSC3 and OSC4 are connected to a 32768Hz crystal to form a Real Time Clock for timing purposes or to form a system clock.
$\overline{RES}$	I	—	Schmitt Trigger reset input, active low.
VDD	—	—	Positive power supply
VSS	—	—	Negative power supply, ground

Absolute Maximum Ratings

Supply Voltage .....V_{SS}–0.3V to V_{SS}+6.0V
Input Voltage.....V_{SS}–0.3V to V_{DD}+0.3V
I_{OL} Total 150mA
Total Power Dissipation 500mW

Storage Temperature .....–50°C to 125°C
Operating Temperature.....–40°C to 85°C
I_{OH} Total.....–100mA

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

D.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	LVR disabled, f_{SYS}=4MHz	2.2	—	5.5	V
			LVR disabled, f_{SYS}=8MHz	3.3	—	5.5	V
V_{LCD}	LCD Power Supply (Note *)	—	VA≤5.5V	2.2	—	5.5	V
I_{DD1}	Operating Current (Crystal OSC, RC OSC)	3V	No load, f_{SYS}=4MHz, UART Off	—	1	2	mA
		5V		—	3	5	mA
I_{DD2}	Operating Current (Crystal OSC, RC OSC)	3V	No load, f_{SYS}=4MHz, UART On	—	1.5	3.0	mA
		5V		—	3	6	mA
I_{DD3}	Operating Current (Crystal OSC, RC OSC)	5V	No load, f_{SYS}=8MHz, UART Off	—	4	8	mA
I_{DD4}	Operating Current (Crystal OSC, RC OSC)	5V	No load, f_{SYS}=8MHz, UART On	—	5	10	mA
I_{DD5}	Operating Current (f_{SYS}=RTC OSC)	3V	No load, UART Off	—	0.3	0.6	mA
		5V		—	0.6	1	mA
I_{STB1}	Standby Current (*f_S=f_{SYS}/4)	3V	No load, system HALT, LCD Off at HALT, UART Off	—	—	1	μA
		5V		—	—	2	μA
I_{STB2}	Standby Current (*f_S=RTC OSC)	3V	No load, system HALT, LCD On at HALT, C type, UART Off	—	2.5	5.0	μA
		5V		—	10	20	μA
I_{STB3}	Standby Current (*f_S=WDT OSC)	3V	No load, system HALT LCD On at HALT, C type, UART Off	—	2	5	μA
		5V		—	6	10	μA
I_{STB4}	Standby Current (*f_S=RTC OSC)	3V	No load, system HALT, LCD On at HALT, R type, 1/2 bias, UART Off	—	17	30	μA
		5V		—	34	60	μA
I_{STB5}	Standby Current (*f_S=RTC OSC)	3V	No load, system HALT, LCD On at HALT, R type, 1/3 bias, UART Off	—	13	25	μA
		5V		—	26	50	μA
I_{STB6}	Standby Current (*f_S=WDT OSC)	3V	No load, system HALT, LCD On at HALT, R type, 1/2 bias, UART Off	—	14	25	μA
		5V		—	28	50	μA
I_{STB7}	Standby Current (*f_S=WDT OSC)	3V	No load, system HALT, LCD On at HALT, R type, 1/3 bias, UART Off	—	10	20	μA
		5V		—	20	40	μA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{IL1}	Input Low Voltage for I/O Ports, TMR0, TMR1, TMR2, $\overline{INT0}$ and $\overline{INT1}$	—	—	0	—	$0.3V_{DD}$	V
V_{IH1}	Input High Voltage for I/O Ports, TMR0, TMR1, TMR2, $\overline{INT0}$ and $\overline{INT1}$	—	—	$0.7V_{DD}$	—	V_{DD}	V
V_{IL2}	Input Low Voltage ($\overline{RES}$)	—	—	0	—	$0.4V_{DD}$	V
V_{IH2}	Input High Voltage ($\overline{RES}$)	—	—	$0.9V_{DD}$	—	V_{DD}	V
I_{OL1}	I/O Port Sink Current	3V	$V_{OL}=0.1V_{DD}$	6	12	—	mA
		5V		10	25	—	mA
I_{OH1}	I/O Port Source Current	3V	$V_{OH}=0.9V_{DD}$	–2	-4	—	mA
		5V		–5	–8	—	mA
I_{OL2}	LCD Common and Segment Current	3V	$V_{OL}=0.1V_A$	210	420	—	μA
		5V		350	700	—	μA
I_{OH2}	LCD Common and Segment Current	3V	$V_{OH}=0.9V_A$	–80	–160	—	μA
		5V		–180	–360	—	μA
R_{PH}	Pull-high Resistance	3V	—	20	60	100	kΩ
		5V		10	30	50	kΩ
V_{LVR}	Low Voltage Reset Voltage	—	—	2.7	3.0	3.3	V
V_{LVD}	Low Voltage Detector Voltage	—	—	3.0	3.3	3.6	V

Note: "*" for the value of VA refer to the LCD driver section.

"*f_S" please refer to the WDT clock option

A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS1}	System Clock (Crystal OSC, RC OSC)	—	2.2V~5.5V	400	—	4000	kHz
		—	3.3V~5.5V	400	—	8000	kHz
f_{SYS2}	System Clock (32768Hz Crystal OSC)	—	—	—	32768	—	Hz
f_{RTCOSC}	RTC Frequency	—	—	—	32768	—	Hz
f_{TIMER}	Timer I/P Frequency (50% duty)	—	2.2V~5.5V	0	—	4000	kHz
		—	3.3V~5.5V	0	—	8000	kHz
t_{WDTOSC}	Watchdog Oscillator Period	3V	—	45	90	180	μs
		5V	—	32	65	130	μs
t_{RES}	External Reset Low Pulse Width	—	—	1	—	—	μs
t_{SST}	System Start-up Timer Period	—	Wake-up from HALT	—	1024	—	$*t_{SYS}$
t_{LVR}	Low Voltage Width to Reset	—	—	0.25	1	2	ms
t_{INT}	Interrupt Pulse Width	—	—	1	—	—	μs

Note: $*t_{SYS}= 1/f_{SYS1}$ or $1/f_{SYS2}$

Functional Description

Execution Flow

The system clock is derived from either a crystal or an RC oscillator or a 32768Hz crystal oscillator. It 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. The pipelining scheme makes it possible for each instruction to be effectively executed in a cycle. If an instruction changes the value of the program counter, two cycles are required to complete the instruction.

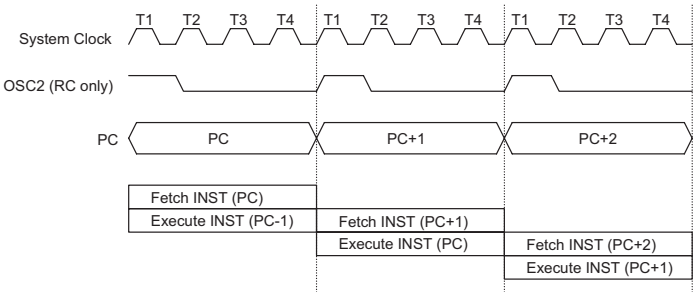
Program Counter – PC

The program counter (PC) is 14 bits wide and controls the sequence in which the instructions stored in the program ROM are executed. The contents of the PC can specify a maximum of 16384 addresses.

After accessing a program memory word to fetch an instruction code, the value of the PC is incremented by ″1″. The PC then points to the memory word containing the next instruction code.

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

The conditional skip is activated by instructions. Once the condition is met, the next instruction, fetched during the current instruction execution, is discarded and a dummy cycle replaces it to get a proper instruction; otherwise the program proceeds to the next instruction.

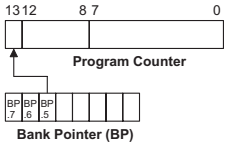


Execution Flow

Mode	Program Counter									
	*13	*12~*8	*7	*6	*5	*4	*3	*2	*1	*0
Initial Reset	0	00000	0	0	0	0	0	0	0	0
External Interrupt 0	0	00000	0	0	0	0	0	1	0	0
External Interrupt 1	0	00000	0	0	0	0	1	0	0	0
Timer/Event Counter 0 Overflow	0	00000	0	0	0	0	1	1	0	0
Timer/Event Counter 1 Overflow	0	00000	0	0	0	1	0	0	0	0
UART Interrupt	0	00000	0	0	0	1	0	1	0	0
Multi Function Interrupt	0	00000	0	0	0	1	1	0	0	0
Skip	Program Counter + 2 (within the current bank)									
Loading PCL	*13	*12~*8	@7	@6	@5	@4	@3	@2	@1	@0
Jump, Call Branch	BP.5	#12~#8	#7	#6	#5	#4	#3	#2	#1	#0
Return from Subroutine	S13	S12~S8	S7	S6	S5	S4	S3	S2	S1	S0

Program Counter

Note: *13~*0: Program counter bits
#12~#0: Instruction code bits
S13~S0: Stack register bits
@7~@0: PCL bits



The lower byte of the PC, known as PCL, is a readable and writeable register. Moving data into the PCL performs a short jump. The destination is within 256 locations.

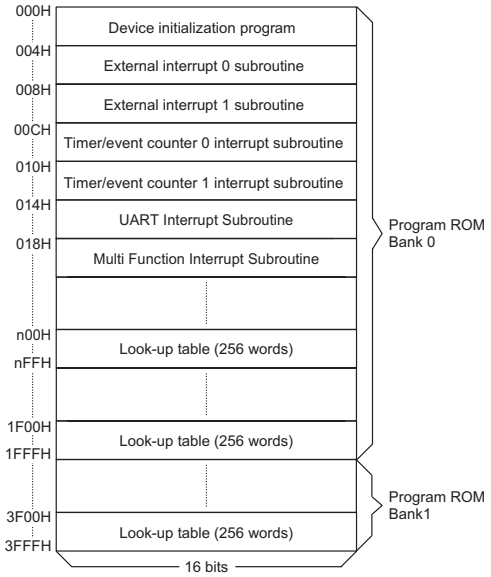
When a control transfer takes place, an additional dummy cycle is required.

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 organised into 8192×16×2 banks which are addressed by the program counter and table pointer.

The BP register bits5~bits7 are used to select the Program Memory bank. When BP.7~BP.5 = 000B, Program Memory bank 0 is selected and ranges from 0000H to 1FFFH. When BP.7~BP.5 = 001B, Program Memory bank 1 is selected which ranges from 2000H to 3FFFH.

The CALL and JMP instruction provide for a full 13 bits of addressing to allow branching anywhere within the 8K program memory bank. When executing a CALL or JMP instruction, the highest bit of the address is provided by BP.5. When executing a CALL or JMP instruction, the bank select bit must be correctly programmed to ensure that the desired program memory bank is addressed. If a return from a CALL instruction or an interrupt is executed, the entire 14-bit program counter is popped off the stack.



Note: n ranges from 0 to 1F

Program Memory

Certain locations in the ROM are reserved for special usage:

- Location 000H
  Location 000H is reserved for program initialisation. After a chip reset, the program always begins execution at this location.
- Location 004H
  Location 004H is reserved for the external interrupt service program. If the $\overline{INT0}$ input pin is activated, and the interrupt is enabled, and the stack is not full, the program will jump to this location and begin execution.
- Location 008H
  Location 008H is reserved for the external interrupt service program also. If the $\overline{INT1}$ input pin is activated, and the interrupt is enabled, and the stack is not full, the program will jump to this location and begin execution.
- Location 00CH
  Location 00CH is reserved for the Timer/Event Counter 0 interrupt service program. If a timer interrupt results from a Timer/Event Counter 0 overflow, and if the interrupt is enabled and the stack is not full, the program will jump to this location and begin execution.
- Location 010H
  Location 010H is reserved for the Timer/Event Counter 1 interrupt service program. If a timer interrupt results from a Timer/Event Counter 1 overflow, and if the interrupt is enabled and the stack is not full, the program will jump to this location and begin execution.
- Location 014H
  This location is reserved for the UART interrupt service program. If a UART interrupt results from a UART TX or RX, and the interrupt is enabled and the stack is not full, the program will jump to this location and begin execution.
- Location 018H
  This location is reserved for the multi function interrupt service program. If a multi function interrupt results from a Timer/Event Counter 2 overflow, a time base interrupt occurs, or an RTC counter overflow, and the related interrupts are enabled and the multi function interrupt is enabled and the stack is not full, the program will jump to this location and begin execution.

Instruction(s)	Table Location								
	*13~*8	*7	*6	*5	*4	*3	*2	*1	*0
TABRDC [m]	TBHP	@7	@6	@5	@4	@3	@2	@1	@0
TABRDL [m]	111111	@7	@6	@5	@4	@3	@2	@1	@0

Table Location

Note: *13~*0: Table location bits
@7~@0: Bits of TBLP
TBHP: Table pointer higher-order bits

- Table location
  Any location in the program memory can be used as look-up tables. The instructions "TABRDC [m]" (page specified by TBHP and TBLP) 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). The higher-order byte table pointer TBHP (1FH) and lower-order byte table pointer TBLP (07H) are read/write registers, which indicate the table locations. Before accessing the table data, the location has to be placed in the TBHP and TBLP. The TBLH register is read only and cannot be restored. If the main routine and the interrupt service routine both employ the table read instruction, the contents of the TBLH register in the main routine are likely to be changed by the table read instruction used in the interrupt service routine. If this happens errors can occur. Therefore, using the table read instruction in the main routine and in the interrupt service routine simultaneously should be avoided. However, if the table read instruction has to be used in both the main routine and in the interrupt service routine the interrupt should be disabled prior to executing the table read instruction. It should not be re-enabled until TBLH in the main routine has been backed up. All table related instructions require 2 cycles to execute.

Stack Register – STACK

The stack register is a special part of the memory used to save the contents of the program counter. The stack is organised into 16 levels and is neither part of the data memory nor part of the program memory, and is neither readable nor writeable. Its activated level is indexed by a stack pointer, known as SP, and is neither readable nor writeable. At the start of a subroutine call or an interrupt acknowledge, the contents of the program counter are pushed onto the stack. At the end of the subroutine or interrupt routine, indicated by a return instruction, RET or RETI, the contents of the program counter are restored to their previous value from the stack. After a chip reset, the SP will point to the top of the stack.

If the stack is full and a non-masked interrupt takes place, the interrupt request flag is recorded but the acknowledge is still inhibited. Once the Stack Pointer is decremented, by RET or RETI, the interrupt is serviced. This feature prevents stack overflow, allowing the programmer to use the structure easily. Likewise, 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 16 return addresses are stored.

Data Memory – RAM

Not including the LCD memory, the data memory, has a total capacity of 608×8 bits, and is divided into two functional groups, namely, the special function registers and the general purpose data memory most of which are readable/ writeable, although some are read only. The General Purpose Data Memory is subdivided into three banks, Banks 0, 2 and 3 each of which has a capacity of 192 × 8bits. The bank pointer, BP, selects which bank is to be used, however care should be exercised when manipulating the bank pointer as it is also used to select the Program Memory bank.

BP	RAM Bank
00000	0
00001	1
00010	2
00011	3

The general purpose data memory, addressed from 40H to FFH (bank0, 2, 3), is used for data and control information under instruction commands.

The RAM areas can directly handle arithmetic, logic, increment, decrement, and rotate operations. Except for some dedicated bits, each bit in the RAM can be set and reset by the bit manipulation instructions "SET [m].i" and "CLR [m].i". They are also indirectly accessible through Memory pointer register 0, MP0, or Memory pointer register 1, MP1.

There is also a special part of memory for the LCD memory. Bits in this special part memory are mapped to the LCD pixel one by one. This LCD memory is located in data memory bank 1.

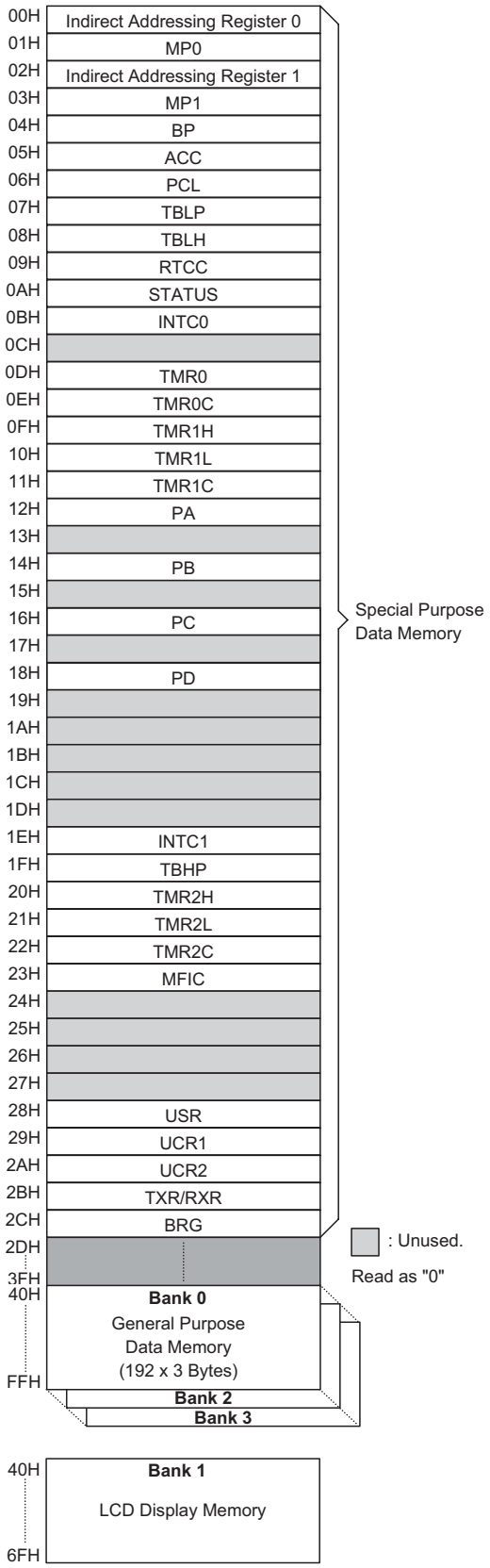
Indirect Addressing Registers

Locations 00H and 02H are indirect addressing registers that are not physically implemented. Any read/write operation of [00H] and [02H] accesses the data memory pointed to by MP0 and MP1, respectively. Reading locations 00H or 02H indirectly returns the result 00H. Writing to it indirectly leads to no operation.

The direct transfer of data between two indirect addressing registers is not supported. The memory pointer registers, MP0 and MP1, are both 8-bit registers used to access the data memory by combining the corresponding indirect addressing registers. MP0 can only be applied to memory located at bank 0, while MP1 can be applied to data memory from bank 0, bank 2 and bank 3 as well as the LCD display memory which is located in bank 1.

Accumulator – ACC

The accumulator, ACC, is related to ALU operations. It is also mapped to location 05H in the data memory and is capable of operating with immediate data. Any data transfers between two data memory locations must pass through the ACC.



RAM Mapping

Arithmetic and Logic Unit – ALU

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

- Arithmetic operations - ADD, ADC, SUB, SBC, DAA
- Logical operations - AND, OR, XOR, CPL
- Rotations - RL, RR, RLC, RRC
- Increment and Decrement - INC, DEC
- Branch decisions - SZ, SNZ, SIZ, SDZ etc.

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

Status Register – STATUS

The status register is 8 bits wide and contains, a carry flag (C), an auxiliary carry flag (AC), a zero flag (Z), an overflow flag (OV), a power down flag (PDF), and a watchdog time-out flag (TO). It also records the status information and controls the operation sequence.

Except for the TO and PDF flags, bits in the status register can be altered by instructions similar to other registers. Data written into the status register does not alter the TO or PDF flags. Operations related to the status register, however, may yield different results from those intended. The TO and PDF flags can only be changed by a Watchdog Timer overflow, device power-on, or clearing the Watchdog Timer and executing the ″HALT″ instruction. The Z, OV, AC, and C flags reflect the status of the latest operations.

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

Interrupts

The device provides two external interrupts, three internal timer/event counters interrupts, an internal time base interrupt, an internal real time clock interrupt, and an UART TX/RX interrupt. The interrupt control register 0, INTC0, and interrupt control register 1, INTC1, both contain the interrupt control bits that are used to set the enable/disable status and to record the interrupt request flags.

Once an interrupt subroutine is serviced, other interrupts are all blocked, by clearing the EMI bit. This scheme may prevent any further interrupt nesting. Other interrupt requests may take place during this interval, but only the interrupt request flag will be recorded. If a certain interrupt requires servicing within the service routine, the EMI bit and the corresponding bit in the INTC0 or of INTC1 register may be set in order to allow interrupt nesting. Once 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.

Bit No.	Label	Function
0	C	C is set if an 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.
1	AC	AC is set if an 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.
2	Z	Z is set if the result of an arithmetic or logic operation is zero, otherwise Z is cleared.
3	OV	OV is set if an 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.
4	PDF	PDF is cleared by either a system power-up or executing the ″CLR WDT″ instruction. PDF is set by executing the ″HALT″ instruction.
5	TO	TO is cleared by a system power-up or executing the ″CLR WDT″ or ″HALT″ instruction. TO is set by a WDT time-out.
6, 7	—	Unused bit, read as ″0″

STATUS (0AH) Register

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

External interrupts are triggered by a high to low transition on the $\overline{\text{INT0}}$ or $\overline{\text{INT1}}$ pins, which will result in their their related interrupt request flags, EIF0 and EEF1, being set. After the interrupt is enabled, the stack is not full, and a high to low transition occurs on the external interrupt pins, a subroutine call to location 04H or 08H occurs. When the interrupt service routine is serviced, the interrupt request flags, EIF0 and EIF1, and the global enable bit, EMI, are all cleared to disable other interrupts.

The internal Timer/Event Counter 0 interrupt is initialised by setting the Timer/Event Counter 0 interrupt request flag, T0F. This will occur when the timer overflows. After the interrupt is enabled, and the stack is not full, and T0F bit is set, a subroutine call to location 0CH occurs. The related interrupt request flag, T0F, is reset, and the EMI bit is cleared to disable further interrupts. The Timer/Event Counter 1 is operated in the same manner but its related interrupt request flag is T1F, and its subroutine call location is 10H.

The UART interrupt is initialised by setting the interrupt request flag, URF, that is caused by a regular UART receive signal, caused by a UART transmit signal. After the interrupt is enabled, the stack is not full, and the URF bit is set, a subroutine call to location 14H occurs. The related interrupt request flag, URF, is reset and the EMI bit is cleared to disable further interrupts.

The multi function interrupt is initialised by setting the interrupt request flag, MFF, that is caused by a regular internal Timer/Event Counter 2 overflow, caused by a time base signal or caused by a real time clock signal. After the interrupt is enabled, the stack is not full, and the MFF bit is set, a subroutine call to location 18H occurs. The related interrupt request flag, MFF, is reset and the EMI bit is cleared to disable further interrupts.

During the execution of an interrupt subroutine, other interrupt acknowledgments are all held until a ″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″ may be executed. RETI sets the EMI bit and enables an interrupt service, but RET does not.

Interrupts occurring in the interval between the rising edges of two consecutive T2 pulses are 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.

Interrupt Source	Priority	Vector
External interrupt 0	1	004H
External interrupt 1	2	008H
Timer/Event Counter 0 overflow	3	00CH
Timer/Event Counter 1 overflow	4	010H
UART interrupt	5	014H
Multi function interrupt (Timer 2, Time base, RTC)	6	018H

It is recommended that a program should not use the ″CALL subroutine″ within the interrupt subroutine. It's because interrupts often occur in an unpredictable manner or require to be serviced immediately in some applications. During that period, if only one stack is left, and enabling the interrupt is not well controlled, operation of the ″call″ in the interrupt subroutine may damage the original control sequence.

Bit No.	Label	Function
0	EMI	Controls the master (global) interrupt (1=enable; 0=disable)
1	EEI0	Controls the external interrupt 0 (1=enable; 0=disable)
2	EEI1	Controls the external interrupt 1 (1=enable; 0=disable)
3	ET0I	Controls the Timer/Event Counter 0 overflow interrupt (1=enable; 0=disable)
4	EIF0	External interrupt 0 request flag (1=active; 0=inactive)
5	EIF1	External interrupt 1 request flag (1=active; 0=inactive)
6	T0F	Timer/Event Counter 0 overflow request flag (1=active; 0=inactive)
7	—	Unused bit, read as ″0″

INTC0 (0BH) Register

Bit No.	Label	Function
0	ET1I	Controls the Timer/Event Counter 1 overflow interrupt (1=enable; 0=disable)
1	EURI	Controls the UART TX or RX interrupt (1=enable; 0:disable)
2	EMFI	Controls the multi-function interrupt (1=enable; 0:disable)
3, 7	—	Unused bit, read as ″0″
4	T1F	Timer/Event Counter 1 overflow request flag (1=active; 0=inactive)
5	URF	UART TX or RX interrupt request flag (1=active; 0=inactive)
6	MFF	Multi function interrupt request flag (1=active; 0=inactive)

INTC1 (1EH) Register

Bit No.	Label	Function
0	ET2I	Controls the Timer/Event Counter 2 overflow interrupt (1=enable; 0=disable)
1	ETBI	Controls the time base interrupt (1=enable; 0=disable)
2	ERTI	Controls the real time clock interrupt (1=enable; 0=disable)
3, 7	—	Unused bit, read as ″0″
4	T2F	Timer/Event Counter 2 interrupt request flag (1=active; 0=inactive)
5	TBF	Time base interrupt request flag (1=active; 0=inactive)
6	RTF	Real time clock interrupt request flag (1=active; 0=inactive)

MFIC (23H) Register

Oscillator Configuration

These devices contain three kinds of system clocks, an RC oscillator, a crystal oscillator and a 32768Hz crystal oscillator, the choice of which is determined by configuration options. No matter what type of oscillator is selected, the signal is used for the system clock. The power down mode stops the system oscillator if it is an RC or crystal oscillator type. The 32768Hz crystal system oscillator will continue to run even if in the power down mode. If the 32768Hz crystal oscillator is selected as the system oscillator, the system oscillator will continue to run but f_{SYS} and instruction execution will cease. Since the system oscillator is also designed for timing purposes, the internal timing such as that for the RTC, the time base and WDT operation continues to run even if the system enters the power down mode.

Of the three oscillators, if the RC oscillator is used, an external resistor between OSC1 and VSS is required, whose range should be from 24kΩ to 1MΩ. A frequency equal to the system clock divided by 4, is available on pin OSC2. This pin can be used for synchronisation purposes but as it is an open drain output a pull-high resistor should be connected. The RC oscillator provides the most cost effective solution, but, the frequency of the oscillation may vary with VDD, temperature, and process variations. It is therefore, not suitable for timing sensitive operations where accurate oscillator frequencies are desired.

If the crystal oscillator is selected, a crystal connected between OSC1 and OSC2 is needed to provide the feedback and phase shift required for the oscillator. No other external components are required. A resonator may be connected between OSC1 and OSC2 to replace the crystal and to get a frequency reference, but two external capacitors connected between OSC1/OSC2 and ground are required.

A further oscillator circuit designed for the real time clock also exists. This operates at a sole frequency of 32768Hz, for which a 32768Hz crystal should be connected between pins OSC3 and OSC4.

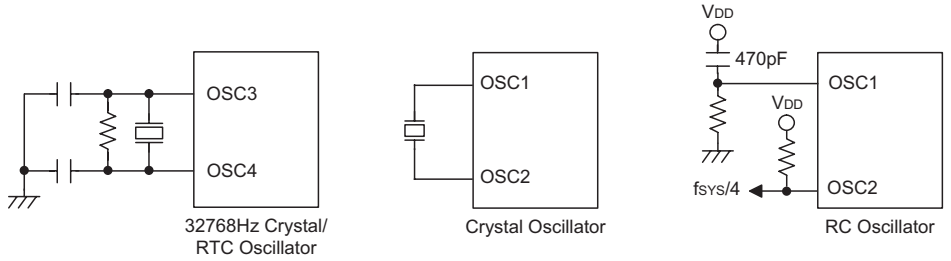
The RTC oscillator circuit can be controlled to start up quickly by clearing the ″QOSC″ bit in the RTCC register. After power on, as the RTC oscillator will be in the quick start up mode, it is recommended that it be turned off after about 2 seconds to conserve power.

The WDT oscillator is a free running on-chip RC oscillator requiring no external components. Although when the system enters the power down mode, the system clock stops, the WDT oscillator will continue to run with a period of approximately 65μs at 5V. The WDT oscillator can be disabled by a configuration option to conserve power.

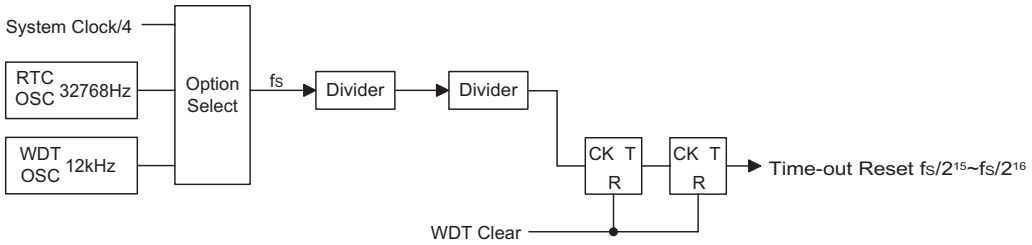
Watchdog Timer – WDT

The WDT clock source is sourced from its own dedicated RC oscillator (WDT oscillator), from the instruction clock (system clock/4) or the real time clock oscillator (RTC oscillator). The timer is designed to prevent a software malfunction or sequence from jumping to an unknown location with unpredictable results. The WDT can be disabled by a configuration option. If the WDT is disabled, all instruction executions relating to the WDT will lead to no operation.

The WDT time-out period is $f_S/2^{15}$~$f_S/2^{16}$.



System Oscillator



Watchdog Timer

If the WDT clock source chooses the internal WDT oscillator as its clock source, the time-out period may vary with temperature, VDD, and process variations. If the clock source is chosen to be the instruction clock, then when the power down mode is entered, it must be noted that the WDT will stop counting and lose its protective function.

When the device operates in a noisy environment, using the WDT oscillator is strongly recommended, since the power down mode will stop the system clock.

The WDT overflow under normal operation initialises a ″chip reset″ and sets the status bit ″TO″. In the power down mode, the overflow initialises a ″warm reset″, and only the program counter and SP are reset to zero. To clear the WDT contents, there are three methods to be adopted. These are an external reset (a low level on the $\overline{\text{RES}}$ pin), a software instruction, and a ″HALT″ instruction.

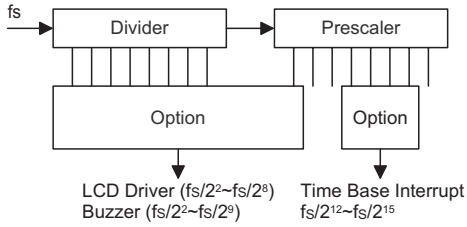
There are two methods of using software instructions to clear the Watchdog Timer, one of which must be chosen by configuration option. The first option is to use the single ″CLR WDT″ instruction while the second is to use the two commands ″CLR WDT1″ and ″CLR WDT2″. For the first option, a simple execution of ″CLR WDT″ will clear the WDT while for the second option, both ″CLR WDT1″ and ″CLR WDT2″ must both be executed to successfully clear the WDT. Note that for this second option, if ″CLR WDT1″ is used to clear the WDT, successive executions of this instruction will have no effect, only the execution of a ″CLR WDT2″ instruction will clear the WDT. Similarly after the ″CLR WDT2″ instruction has been executed, only a successive ″CLR WDT1″ instruction can clear the Watchdog Timer.

Multi-function Timer

These devices provide a multi-function timer for the WDT, time base and RTC but with different time-out periods. The multi-function timer consists of an 8-stage divider and a 7-bit prescaler, with the clock source coming from the WDT OSC, the RTC OSC or the instruction clock which is the system clock divided by 4. The multi-function timer also provides a selectable frequency signal, ranging from $f_S/2^2$ to $f_S/2^8$, for the LCD driver circuits, and a selectable frequency signal, ranging from $f_S/2^2$ to $f_S/2^9$, for the buzzer output selectable by configuration options. To obtain a proper display, it is recommended that a frequency as near as possible to 4kHz is selected for the LCD driver circuits.

Time Base

The time base offers a periodic time-out period to generate a regular internal interrupt. Its time-out period ranges from $/2^{12}$ to $f_S/2^{15}$ selected by a configuration option. If a time base time-out occurs, the related interrupt request flag, TBF, will be set. If the interrupt is enabled, and the stack is not full, a subroutine call to location 18H will take place. The time base time-out signal can also be applied as a clock source to Timer/Event Counter 1 in order to get a longer time-out period.

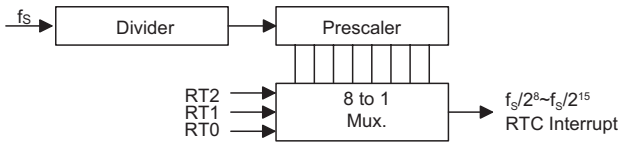


Real Time Clock – RTC

The real time clock, abbreviated as RTC, is operated in the same manner as the time base in that it is used to supply a regular internal interrupt. Its time-out period ranges from $f_S/2^8$ to $f_S/2^{15}$ the actual value setup by software programming . Writing data to the RT2, RT1 and RT0 bits in the RTCC register provides various time-out periods. If an RTC time-out occurs, the related interrupt request flag, RTF, will be set. If the interrupt is enabled, and the stack is not full, a subroutine call to location 18H will take place. The real time clock time-out signal can also be used as a clock source for Timer/Event Counter 0 in order to get longer time-out periods.

RT2	RT1	RT0	RTC Clock Divided Factor
0	0	0	2^8*
0	0	1	2^9*
0	1	0	2^{10}*
0	1	1	2^{11}*
1	0	0	2^{12}
1	0	1	2^{13}
1	1	0	2^{14}
1	1	1	2^{15}

Note: ″*″ not recommended to be used



Real Time Clock

Power Down Mode – HALT

The power down mode is initialised when a ″HALT″ instruction is executed and results in the following.

- The system oscillator and f_{SYS} turn off but the WDT or RTC oscillator keeps running, if the WDT oscillator or the real time clock is selected.
- The contents of the data memory and registers remain unchanged.
- The WDT is cleared and starts recounting, if the WDT clock source is sourced from the WDT oscillator or the real time clock oscillator.
- All I/O ports maintain their original status.
- The PDF flag is set but the TO flag is cleared.
- The LCD driver keeps running, if the WDT OSC or RTC OSC is selected.

The system will exit from power down mode by way of an external reset, an interrupt, an external falling edge signal on port A or a WDT overflow. An external reset causes a device initialisation, while a WDT overflow performs a ″warm reset″. After examining the TO and PDF flags, the reason for a chip reset can be determined. The PDF flag is cleared by a system power-up or by executing the ″CLR WDT″ instruction, and is set by executing the ″HALT″ instruction. However if the TO flag is set and a WDT time-out occurs, the corresponding wake-up only resets the program counter and the stack pointer, and leaves the other registers in their original state.

The port A wake-up and interrupt methods can be considered as a continuation of normal execution. Each pin on port A can be independently selected to wake up the device using configuration options. Awakening from an I/O port stimulus, the program resumes execution at the next instruction. When awakening from an interrupt, two sequences may occur. If the related interrupt is disabled or the interrupt is enabled but the stack is full, the program resumes execution at the next instruction. But if the interrupt is enabled, and the stack is not full, the regular interrupt response takes place.

When an interrupt request flag is set before entering the power down mode, the system cannot be awakened using that interrupt.

If a wake-up event occurs, it takes 1024 t_{SYS} system clock periods to resume normal operation. In other words, a dummy period is inserted after the wake-up. If the wake-up results from an interrupt acknowledgment, the actual interrupt subroutine execution is delayed by more than one cycle. However, if the wake-up results in the next instruction execution, the execution will be performed immediately after the dummy period is finished.

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

Reset

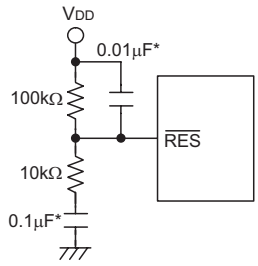
There are three ways in which a reset may occur.

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

The WDT time-out during a power down differs from other chip reset conditions, as it will perform only a ″warm reset″ that resets only the program counter and SP and leaves the other circuits in their original state. Some registers remain unaffected during other reset conditions. Most registers are reset to their ″initial condition″ once the reset conditions are met. By examining the PDF and TO flags, the program can distinguish between different ″chip resets″.

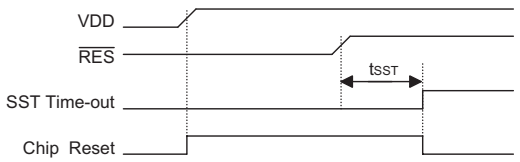
TO	PDF	RESET Conditions
0	0	$\overline{RES}$ reset during power-on
u	u	$\overline{RES}$ reset during normal operation
0	1	$\overline{RES}$ Wake-up HALT
1	u	WDT time-out during normal operation
1	1	WDT Wake-up HALT

Note: ″u″ stands for unchanged

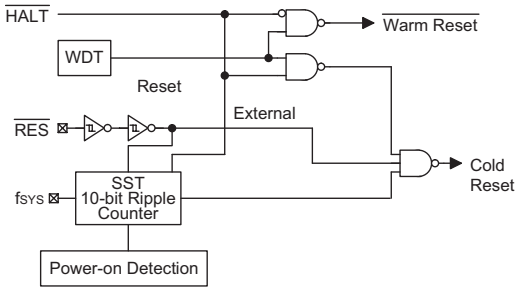


Reset Circuit

Note: ″*″ Make the length of the wiring, which is connected to the $\overline{RES}$ pin as short as possible, to avoid noise interference.



Reset Timing Chart



Reset Configuration

To guarantee that the system oscillator is running and stabilised, the SST (System Start-up Timer) provides an extra delay of 1024 system clock pulses when the system awakes from the power down mode. After awakening from the power down mode, an SST delay is added.

An extra option load time delay is added during a reset and power on.

The functional unit chip reset status is shown below.

Program Counter	000H
Interrupt	Disabled
Prescaler	Cleared
WDT	Cleared. After master reset, WDT starts counting
Timer/event Counter	Off
Input/output Ports	Input mode
Stack Pointer	Points to the top of the stack

Timer/Event Counter

Three timer/event counters are implemented in the device, one 8-bit programmable count-up counter and two 16-bit programmable count-up counter.

The Timer/Event Counter 0 clock source may be sourced from the system clock, the system clock/4, the RTC time-out signal or from an external source. The system clock source or the system clock/4 source is selected by a configuration option.

The Timer/Event Counter 1 clock source may be sourced from the TMR0 overflow, the system clock, the time base time-out signal, the system clock/4 or an external source. The three former clock sources are selected by configuration options. Using the external clock input allows external events to be counted, time intervals or pulse widths to be measured, or an accurate time base to be generated. Using the internal clock allows an accurate time base to be generated.

The Timer/Event Counter 2 contains a 16-bit programmable count-up counter whose clock may be sourced from an external source or an internal clock source. The internal clock source comes from $f_{SYS}/4$. The external clock input allows the user to count external events, measure time intervals or pulse widths, or to generate an accurate time base.

There are two registers related to the Timer/Event Counter 0; TMR0 and TMR0C. Two physical registers are mapped to the TMR0 location. Writing to TMR0 places the starting value in the Timer/Event Counter 0 register while reading TMR0 takes the contents of the Timer/Event Counter 0. The TMR0C register is a timer/event counter control register, which defines the timer options.

There are three registers related to the Timer/Event Counter 1; TMR1H, TMR1L and TMR1C. Writing to TMR1L will only transfer the data into an internal lower-order byte buffer (8-bit) and writing TMR1H will transfer the specified data and the contents of the lower-order byte buffer to TMR1H and TMR1L registers, respectively. The Timer/Event Counter 1 preload register is changed by each writing TRM1H operations. Reading TMR1H will latch the contents of TMR1H and TMR1L counters to the destination and the lower-order byte buffer, respectively. Reading the TMR1L will read the contents of the lower-order byte buffer. The TMR1C is the Timer/Event Counter 1 control register, which defines the operating mode, counting enable or disable and an active edge.

There are three registers related to the Timer/Event Counter 2; TMR2H (20H), TMR2L (21H), TMR2C (22H). Writing TMR2L will only place the written data to an internal lower-order byte buffer (8-bit) and writing TMR2H will transfer the specified data and the contents of the lower-order byte buffer to TMR2H and TMR2L registers, respectively. The Timer/Event Counter 2 preload register is changed by each writing TRM2H operations. Reading TMR2H will latch the contents of the TMR2H and TMR2L counters to the destination and the lower-order byte buffer, respectively. Reading the TMR2L will read the contents of the lower-order byte buffer. The TMR2C is the Timer/Event Counter 2 control register, which defines the operating mode, counting enable or disable and an active edge.

The T0M0, T0M1 (TMR0C), T1M0, T1M1 (TMR1C) and T2M0, T2M1 (TMR2C) bits define the operation mode. The event count mode is used to count external events, which means that the clock source is from an external (TMR0/TMR1/TMR2) pin. The timer mode functions as a normal timer with the clock source coming from the in-

The register states are summarised below:

Register	Reset (Power On)	WDT Time-out (Norma Operation)	$\overline{\text{RES}}$ Reset (Normal Operation)	$\overline{\text{RES}}$ Reset (HALT)	WDT Time-out (HALT)*
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
BP	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
Program Counter	0000H	0000H	0000H	0000H	0000H
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
RTCC	0000 0111	0000 0111	0000 0111	0000 0111	00uu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC0	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
TMR0	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR0C	0000 1---	0000 1---	0000 1---	0000 1---	uuuu u---
TMR1H	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1L	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1C	0000 1---	0000 1---	0000 1---	0000 1---	uuuu u---
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PD	-111 1111	-111 1111	-111 1111	-111 1111	-uuu uuuu
INTC1	-000 -000	-000 -000	-000 -000	-000 -000	-uuu -uuu
TBHP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR2H	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR2L	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR2C	00-0 1---	00-0 1---	00-0 1---	00-0 1---	uu-u u---
MFIC	-000 -000	-000 -000	-000 -000	-000 -000	-uuu -uuu
USR	0000 1011	0000 1011	0000 1011	0000 1011	uuuu uuuu
UCR1	0000 00x0	0000 00x0	0000 00x0	0000 00x0	uuuu uuuu
UCR2	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
TXR/RXR	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
BRG	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu

Note: "*" stands for warm reset
"u" stands for unchanged
"x" stands for unknown

ternal selected clock source. Finally, the pulse width measurement mode can be used to count the high or low level duration of the external signal (TMR0/TMR1/TMR2), and the counting is based on the internal selected clock source.

In the event count or timer mode, the timer/event counter starts counting at the current contents in the timer/event counter and ends at FFH (FFFFH). Once an overflow occurs, the counter is reloaded from the timer/event counter preload register, and generates an interrupt request flag (T0F: bit 6 of the INTC0; T1F: bit 4 of the INTC1; T2F: bit 4 of the MFIC).

In the pulse width measurement mode with the values of the T0ON/T1ON/T2ON and T0E/T1E/T2E bits equal to ″1″, after the TMR0/TMR1/TMR2 has received a transient from low to high (or high to low if the T0E/T1E/T2E bit is ″0″), it will start counting until the TMR0/TMR1/TMR2) returns to the original level and resets the T0ON/T1ON/T2ON. The measured result remains in the timer/event counter even if the activated transient occurs again. In other words, only 1-cycle measurement can be made until the T0ON/T1ON/T2ON is set. The cycle measurement will re-function as long as it receives further transient pulse. In this operation mode, the timer/event counter begins counting not according to the logic level but to the transient edges. In the case of counter overflows, the counter is reloaded from the timer/event counter register and issues an interrupt request, as in the other two modes, i.e., event and timer modes.

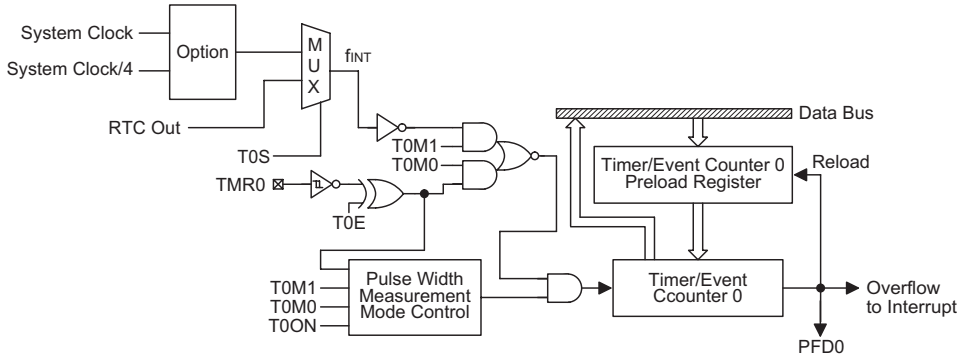
To enable the counting operation, the Timer ON bit (T0ON/T1ON/T2ON; bit 4 of the TMR0C/TMR1C/TMR2C) should be set to ″1″. In the pulse width measurement mode, the T0ON/T1ON/T2ON is automatically cleared after the measurement cycle is completed. But in the other two modes, the T0ON/T1ON/T2ON can only be reset by instructions. The overflow of the Timer/Event Counter 0/1 is one of the wake-up sources and can also be applied to a PFD (Programmable Frequency Divider) output at PA3 by options. Only one PFD (PFD0 or PFD1) can be applied to PA3 by options. No matter what the operation mode is, writing a 0 to ET0I/ET1I/ET2I disables the related interrupt service. When the PFD function is selected, executing ″CLR [PA].3″ instruction to enable PFD output and executing ″SET [PA].3″ instruction to disable PFD output.

In the case of timer/event counter OFF condition, writing data to the timer/event counter preload register also reloads that data to the timer/event counter. But if the timer/event counter is turned on, data written to the timer/event counter is kept only in the timer/event counter preload register. The timer/event counter still continues its operation until an overflow occurs.

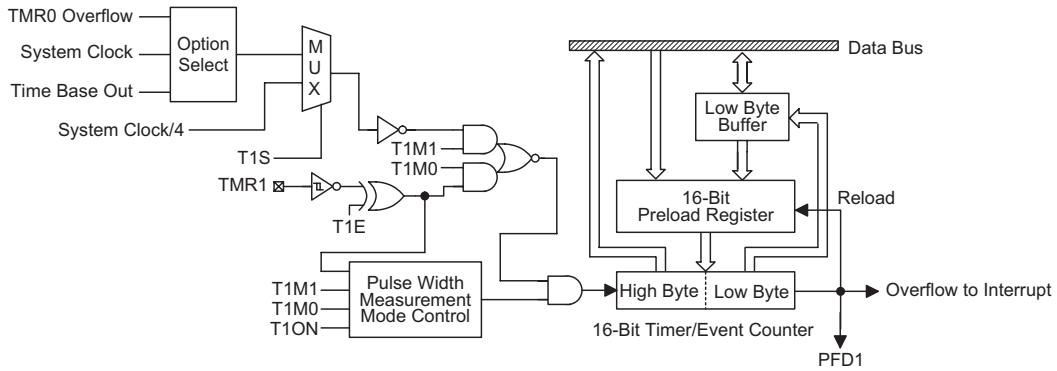
When the timer/event counter (reading TMR0/TMR1/TMR2) is read, the clock is blocked to avoid errors, as this may results in a counting error. Blocking of the clock should be taken into account by the programmer.

It is strongly recommended to load a desired value into the TMR0/TMR1/TMR2 register first, before turning on the related timer/event counter, for proper operation since the initial value of TMR0/TMR1/TMR2 is unknown. Due to the timer/event counter scheme, the programmer should pay special attention on the instruction to enable then disable the timer for the first time, whenever there is a need to use the timer/event counter function, to avoid unpredictable result. After this procedure, the timer/event counter function can be operated normally. The following example is given, using one 8-bit and one 16-bit width Timer (timer 0; timer 1) cascaded into 24-bit width.

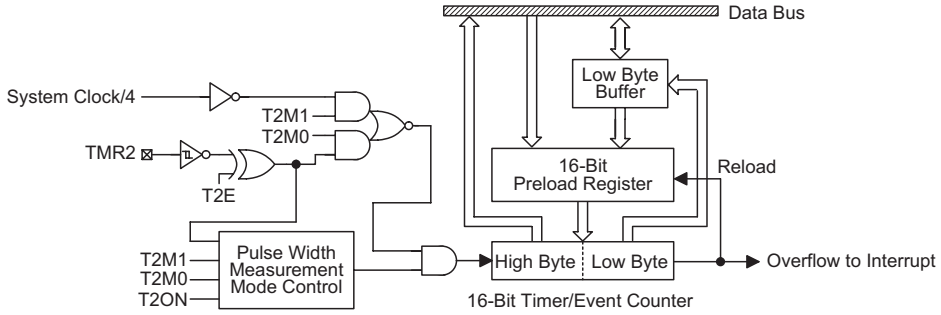
```
START:
    mov    a, 09h    ; Set ET0I & EMI bits to
    mov    intc0, a  ; enable Timer 0 and
                     ; global interrupt

    mov   a, 01h     ; Set ET1I bit to enable
    mov   intc1, a   ; Timer 1 interrupt

    mov   a, 80h     ; Set the operating mode as
    mov   tmr1c, a   ; timer mode and select the mask
                     ; option clock source

    mov  a, 0a0h     ; Set the operating mode as timer
    mov  tmr0c, a    ; mode and select the system
                     ; clock/4


    set    tmr1c.4   ; Enable then disable Timer 1
    clr    tmr1c.4   ; for the first time


    mov  a, 00h      ; Load a desired value into
    mov  tmr0, a     ; the TMR0/TMR1 register
    mov  a, 00h      ;
    mov  tmr1l, a    ;
    mov  tmr1h, a    ;


    set   tmr0c.4    ; Normal operating
    set   tmr1c.4    ;
END
```



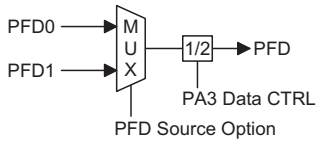
Timer/Event Counter 0



Timer/Event Counter 1



Timer/Event Counter 2



PFD Source Option

Bit No.	Label	Function
0~2	—	Unused bit, read as ″0″
3	T0E	Defines the TMR0 active edge of the timer/event counter: In Event Counter Mode (T0M1,T0M0)=(0,1): 1= count on falling edge; 0= count on rising edge In Pulse Width measurement mode (T0M1,T0M0)=(1,1): 1= start counting on the rising edge, stop on the falling edge; 0= start counting on the falling edge, stop on the rising edge
4	T0ON	Enables/disables the timer counting (0=disable; 1=enable)
5	T0S	2 to 1 multiplexer control inputs which selects the timer/event counter clock source (0=RTC outputs; 1= system clock or system clock/4)
6 7	T0M0 T0M1	Defines the operating mode (T0M1, T0M0) 01= Event count mode (External clock) 10= Timer mode (Internal clock) 11= Pulse Width measurement mode (External clock) 00= Unused

TMR0C (0EH) Register

Bit No.	Label	Function
0~2	—	Unused bit, read as ″0″
3	T1E	Defines the TMR1 active edge of the timer/event counter: In Event Counter Mode (T1M1,T1M0)=(0,1): 1= count on falling edge; 0= count on rising edge In Pulse Width measurement mode (T1M1,T1M0)=(1,1): 1= start counting on the rising edge, stop on the falling edge; 0= start counting on the falling edge, stop on the rising edge
4	T1ON	Enables/disables timer counting (0= disable; 1= enabled)
5	T1S	2 to 1 multiplexer control inputs to select the timer/event counter clock source (0= option clock source; 1= system clock/4)
6 7	T1M0 T1M1	Defines the operating mode (T1M1, T1M0) 01= Event count mode (External clock) 10= Timer mode (Internal clock) 11= Pulse Width measurement mode (External clock) 00= Unused

TMR1C (11H) Register

Bit No.	Label	Function
0~2	—	Unused bit, read as ″0″
3	T2E	Defines the TMR2 active edge of the timer/event counter: In Event Counter Mode (T2M1,T2M0)=(0,1): 1= count on falling edge; 0= count on rising edge In Pulse Width measurement mode (T2M1,T2M0)=(1,1): 1= start counting on the rising edge, stop on the falling edge; 0= start counting on the falling edge, stop on the rising edge
4	T2ON	Enables/disables the timer counting (0=disable; 1=enable)
5	—	Unused bit, read as ″0″
6 7	T2M0 T2M1	Defines the operating mode (T2M1, T2M0): 01=Event count mode (External clock) 10=Timer mode (Internal clock) 11=Pulse width measurement mode (External clock) 00=Unused

TMR2C (22H) Register

Input/Output Ports

There are two 8-bit bidirectional input/output ports, PA and PC and one 8-bit input PB and one 7-bit output PD. PA, PB, PC and PD are mapped to [12H], [14H], [16H] and [18H] of the RAM, respectively. PA0~PA3 can be configured as CMOS (output) or NMOS (input/output) with or without pull-high resistor by options. PA4~PA7 are always pull-high and NMOS (input/output). If NMOS (input) is chosen, each bit on the port (PA0~PA7) can be configured as a wake-up input. PB can only be used for input operation. PC can be configured as CMOS output or NMOS input/output with or without pull-high resistor by options. PD can only be used for CMOS output operation. All the ports for the input operation (PA, PB and PC), are non-latched, that is, the inputs should be ready at the T2 rising edge of the instruction ″MOV A, [m]″ (m=12H, 14H or 16H). For PA, PC, PD output operation, all data are latched and remain unchanged until the output latch is rewritten.

When the PA and PC structures are open drain NMOS type, it should be noted that, before reading data from the pads, a ″1″ should be written to the related bits to disable the NMOS device. That is, executing first the instruction ″SET [m].i″ (i=0~7 for PA) to disable related NMOS device, and then ″MOV A, [m]″ to get stable data.

After a chip reset, these input lines remain at high level or are left floating (by options). Each bit of these output latches can be set or cleared by the ″MOV [m], A″ (m=12H or 16H) instruction.

Some instructions first input data and then follow the output operations. For example, ″SET [m].i″, ″CLR [m].i″, ″CPL [m]″, ″CPLA [m]″ read the entire port states into the CPU, execute the defined operations (bit-operation), and then write the results back to the latches or to the accumulator. When a PA or PC line is used as an I/O line, the related PA or PC line options should be configured as NMOS with or without pull-high resistor. Once a PA or PC line is selected as a CMOS output, the input function cannot be used.

The input state of a PA or PC line is read from the related PA or PC pad. When the PA or PC is configured as NMOS with or without pull-high resistor, one should be careful when applying a read-modify-write instruction to PA or PC. Since the read-modify-write will read the entire port state (pads state) first, execute the specified instruction and then write the result to the port data register. When the read operation is executed, a fault pad state (caused by the load effect or floating state) may be read. Errors will then occur.

There are three function pins that share with the PA port: PA0/BZ, PA1/$\overline{BZ}$ and PA3/PFD.

The BZ and $\overline{BZ}$ are buzzer driving output pair and the PFD is a programmable frequency divider output. If user wants to use the BZ/$\overline{BZ}$ or PFD function, the related PA port should be set as a CMOS output. The buzzer output signals are controlled by PA0 and PA1 data registers as defined in the following table.

PA1 Data Register	PA0 Data Register	PA0/PA1 Pad State
0	0	PA0=BZ, PA1=$\overline{BZ}$
1	0	PA0=BZ, PA1=0
X	1	PA0=0, PA1=0

Note: ″X″ stands for unused

The PFD output signal function is controlled by the PA3 data register and the timer/event counter state. The PFD output signal frequency is also dependent on the timer/event counter overflow period. The definitions of the PFD control signal and PFD output frequency are listed in the following table.

Timer	Timer Preload Value	PA3 Data Register	PA3 Pad State	PFD Frequency
OFF	X	0	U	X
OFF	X	1	0	X
ON	N	0	PFD	f_{INT}/ [2×(256−N)]
ON	N	1	0	X

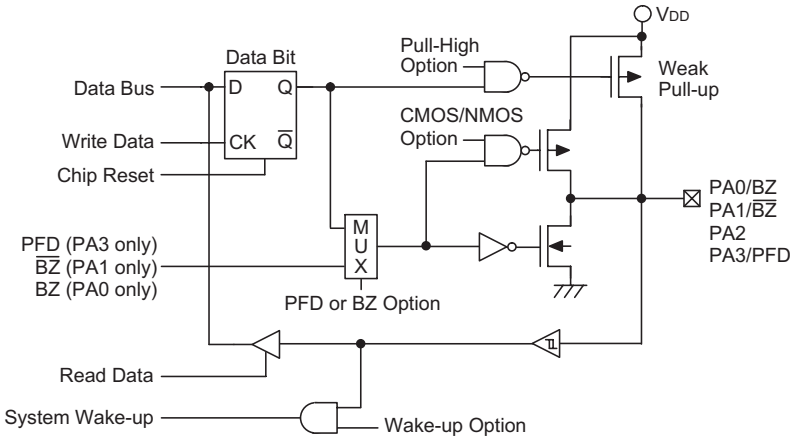
Note: ″X″ stands for unused

″U″ stands for unknown

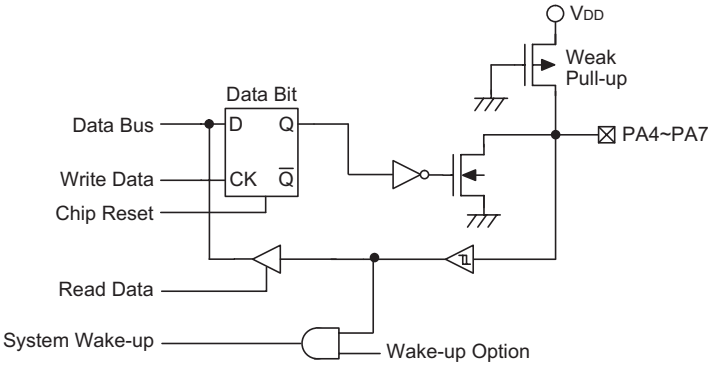
″256″ is for TMR0. If TMR1 is used to generate PFD, the number should be ″65536″.

After a chip reset, these input/output lines remain at high levels (pull-high options) or floating state (non-pull-high options). It is suggested not to apply the ″read-modify-write″ instructions to the I/O port (since a reading error may occur). Using ″MOV″ instruction to avoid the reading error is suggested. The PB is a 8-bit input port and its configuration is Schmitt trigger with pull-high resistors.

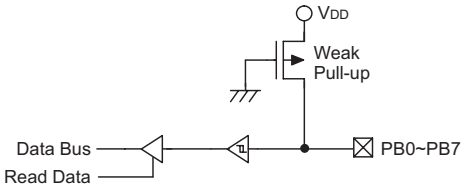
Each line of PA has the capability of waking-up the device. The PB0, PB1, PB2, PB3 and PB4 are pin-shared with $\overline{INT0}$, $\overline{INT1}$, TMR0, TMR1 and TMR2 input functions, respectively.



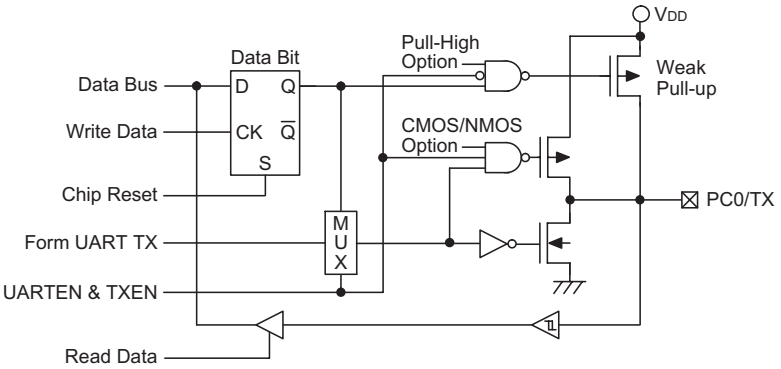
PA0~PA3 Input/Output Ports



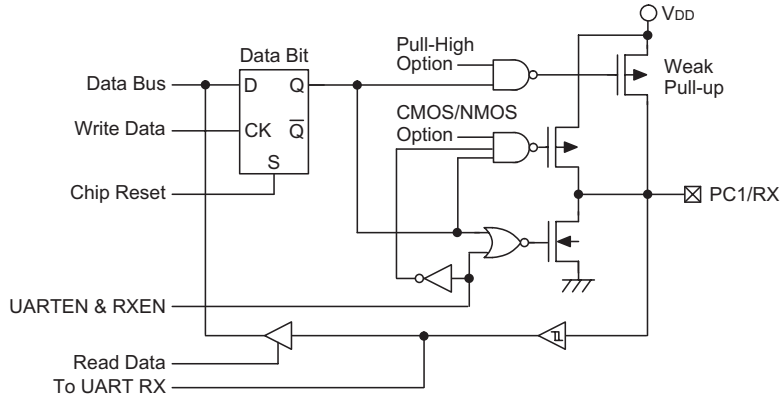
PA4~PA7 Input/Output Ports



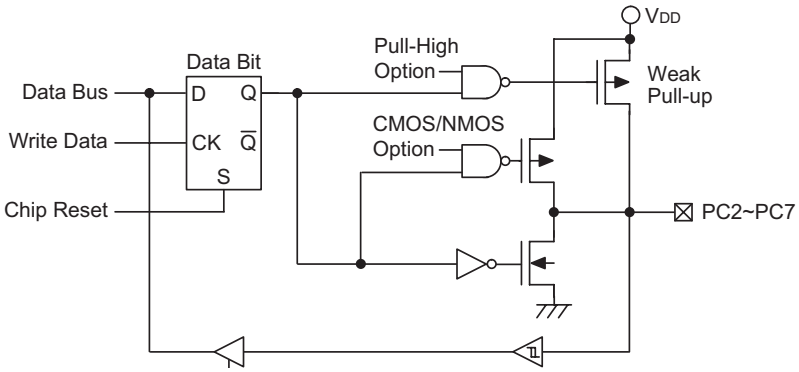
PB Input Port



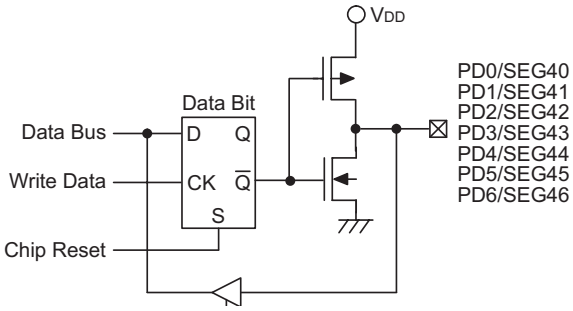
PC0/TX Input/Output Port



PC1/RX Input/Output Port



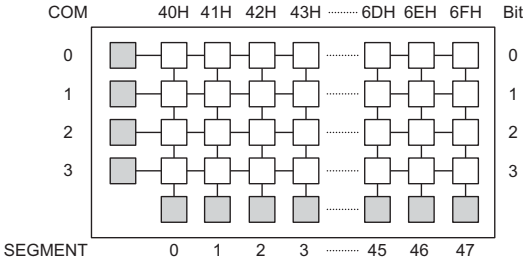
PC2~PC7 Input/Output Ports



PD Output Port

LCD Display Memory

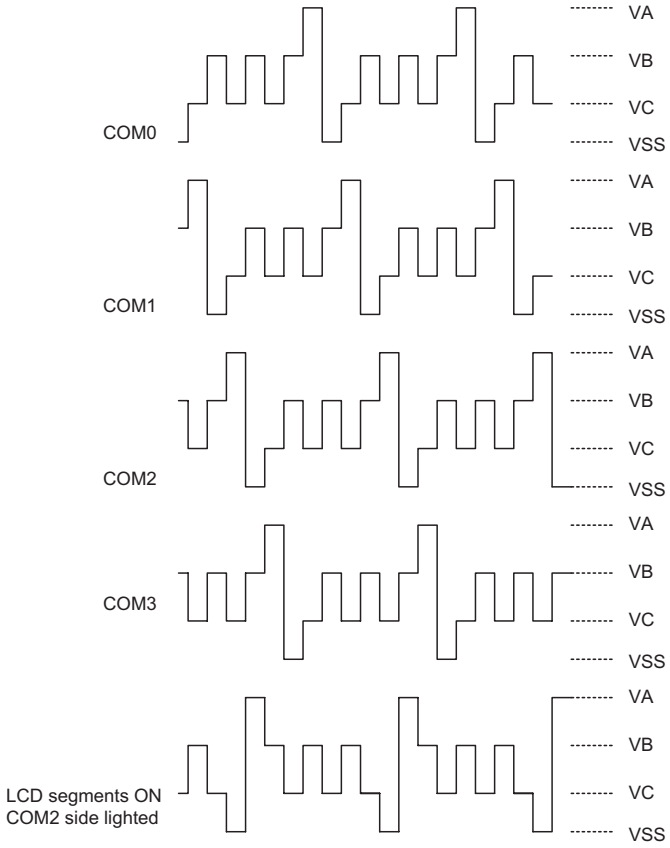
The device provides an area of embedded data memory for the LCD display. This area is located from 40H to 6FH in Bank 1 of the data memory. The bank pointer, BP, is used to switch between the general purpose data memory and the LCD display memory. When BP has the value ″01H″, any data written into the locations 40H~6FH will influence the LCD display. When BP is cleared to ″00H″, any data written into the locations 40H~6FH will access the general purpose data memory. The LCD display memory can be read and written to only using the indirect addressing mode using memory pointer MP1. When data is written into the display data area, it is automatically read by the LCD driver which then generates the corresponding LCD driving signals. To turn the display on or off, a ″1″ or a ″0″ is written to the corresponding bit of the display memory, respectively. The figure illustrates the mapping between the display memory and the LCD pattern for the device.



Display Memory

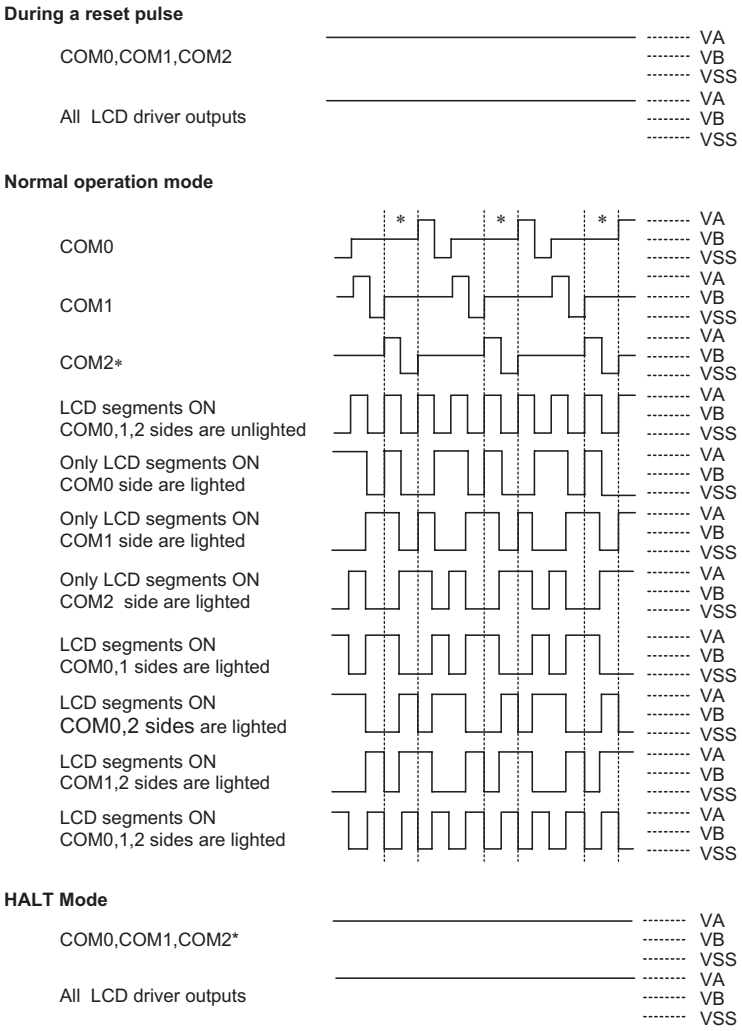
LCD Driver Output

The output number of the LCD driver device can be 48×2, 48×3 or 47×4 selected by configuration options, i.e., 1/2 duty, 1/3 duty or 1/4 duty. There are two types of biasing, ″R″ type or ″C″ type. If the ″R″ bias type is selected, no external capacitors are required. If the ″C″ bias type is selected, a capacitor mounted between C1 and C2 pins is required. If 1/2 bias is selected, a capacitor mounted between the V2 pin and ground is required. If 1/3 bias is selected, two capacitors are required to be connected between the V1 and V2 pins and V_{SS}. A suggested value of 0.1μF is recommended for all the capacitors.



Note: 1/4 duty, 1/3 bias, C type: "VA" 3/2 VLCD, "VB" VLCD, "VC" 1/2 VLCD
1/4 duty, 1/3 bias, R type: "VA" VLCD, "VB" 2/3 VLCD, "VC" 1/3 VLCD

LCD Driver Output



Note: "*" Omit the COM2 signal, if the 1/2 duty LCD is used.
VA=VLCD, VB=1/2 VLCD

LCD Driver Output (1/3 Duty, 1/2 Bias, R/C Type)

The LCD driver requires a clock source for proper operation. The LCD clock source is sourced from the general purpose prescaler whose frequency value is determined by configuration options. The LCD clock frequency should be selected to be as close to 4kHz as possible. The LCD clock frequency options are listed in the following table.

LCD Clock Source	Prescaler Stages
Same as WDT clock source f_S	$f_S/2^2$~$f_S/2^8$

LCD Segments as Output Port

The SEG40~SEG46 lines, via individual configuration options, can be chosen to be either LCD segment outputs or PD outputs. When a segment output is selected, the connection to the VMAX pin depends upon the bias and the voltage that is applied to VLCD. The details are shown in the table. When used as a PD output, V_{MAX} should be connected to V_{DD}.

LCD Type	R Type		C Type	
Bias Type	1/2 Bias	1/3 Bias	1/2 Bias	1/3 Bias
V_{MAX}	If V_{DD}>V_{LCD}, user should connect V_{MAX} to V_{DD}, else connect V_{MAX} to V_{LCD}		If V_{DD}> 3/2V_{LCD}, user should connect V_{MAX} to V_{DD}, else connect V_{MAX} to V_1	

Low Voltage Reset/Detector Functions

The device contains low voltage detector, LVD, and low voltage reset, LVR, circuits. These two functions are enabled or disabled using configuration options. If the configuration options enable the LVD, it can be further enabled or disabled using software, by changing the value of the RTCC.3 bit. The RTCC.5 bit can be used to read the status of the LVD.

The Low Voltate Reset function, LVR, has the same effect as the external $\overline{RES}$ signal which executes a chip reset.Its function is selected via a configuration option. When the device is in the power down mode, the LVR is disabled.

The RTCC register definitions are shown in the table.

Bit No.	Label	Read/Write	Function
0~2	RT0~RT2	R/W	8 to 1 multiplexer control inputs to select the real time clock prescaler output
3	LVDC	R/W	LVD enable/disable (1/0)
4	QOSC	R/W	32768Hz OSC quick start-up - 0/1: quick/slow start
5	LVDO	R	LVD detector output (1/0) - 1: low voltage detected
6, 7	—	—	Unused bit, read as ″0″

RTCC (09H) Register

UART Bus Serial Interface

The HT49RU80/HT49CU80 devices contain an integrated full-duplex asynchronous serial communications UART interface that enables communication with external devices that contain a serial interface. The UART function has many features and can transmit and receive data serially by transferring a frame of data with eight or nine data bits per transmission as well as being able to detect errors when the data is overwritten or incorrectly framed. The UART function possesses its own internal interrupt which can be used to indicate when a reception occurs or when a transmission terminates.

- UART features
  The integrated UART function contains the following features:
  - Full-duplex, asynchronous communication
  - 8 or 9 bits character length
  - Even, odd or no parity options
  - One or two stop bits
  - Baud rate generator with 8-bit prescaler
  - Parity, framing, noise and overrun error detection
  - Support for interrupt on address detect (last character bit=1)
  - Separately enabled transmitter and receiver
  - 2-byte Deep Fifo Receive Data Buffer
  - Transmit and receive interrupts
  - Interrupts can be initialized by the following conditions:
    - Transmitter Empty
    - Transmitter Idle
    - Receiver Full
    - Receiver Overrun
    - Address Mode Detect

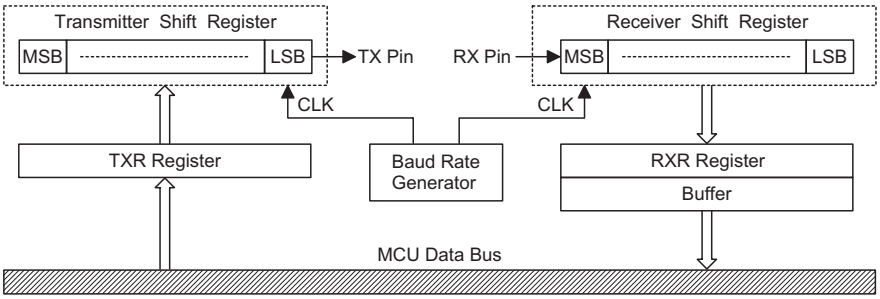
- UART external pin interfacing
  To communicate with an external serial interface, the internal UART has two external pins known as TX and RX. The TX pin is the UART transmitter pin, which can be used as a general purpose I/O pin if the pin is not configured as a UART transmitter, which occurs when the TXEN bit in the UCR2 control register is equal to zero. Similarly, the RX pin is the UART receiver pin, which can also be used as a general purpose I/O pin, if the pin is not configured as a receiver, which occurs if the RXEN bit in the UCR2 register is equal to zero. Along with the UARTEN bit, the TXEN and RXEN bits, if set, will automatically setup these I/O pins to their respective TX output and RX input conditions and disable any pull-high resistor option which may exist on the RX pin.

- UART data transfer scheme
  The block diagram shows the overall data transfer structure arrangement for the UART. The actual data to be transmitted from the MCU is first transferred to the TXR register by the application program. The data will then be transferred to the Transmit Shift Register from where it will be shifted out, LSB first, onto the TX pin at a rate controlled by the Baud Rate Generator. Only the TXR register is mapped onto the MCU Data Memory, the Transmit Shift Register is not mapped and is therefore inaccessible to the application program.

  Data to be received by the UART is accepted on the external RX pin, from where it is shifted in, LSB first, to the Receiver Shift Register at a rate controlled by the Baud Rate Generator. When the shift register is full, the data will then be transferred from the shift register to the internal RXR register, where it is buffered and can be manipulated by the application program. Only the RXR register is mapped onto the MCU Data Memory, the Receiver Shift Register is not mapped and is therefore inaccessible to the application program.

  It should be noted that the actual register for data transmission and reception, although referred to in the text, and in application programs, as separate TXR and RXR registers, only exists as a single shared register in the Data Memory. This shared register known as the TXR/RXR register is used for both data transmission and data reception.

- UART status and control registers
  There are five control registers associated with the UART function. The USR, UCR1 and UCR2 registers control the overall function of the UART, while the BRG register controls the Baud rate. The actual data to be transmitted and received on the serial interface is managed through the TXR/RXR data registers.

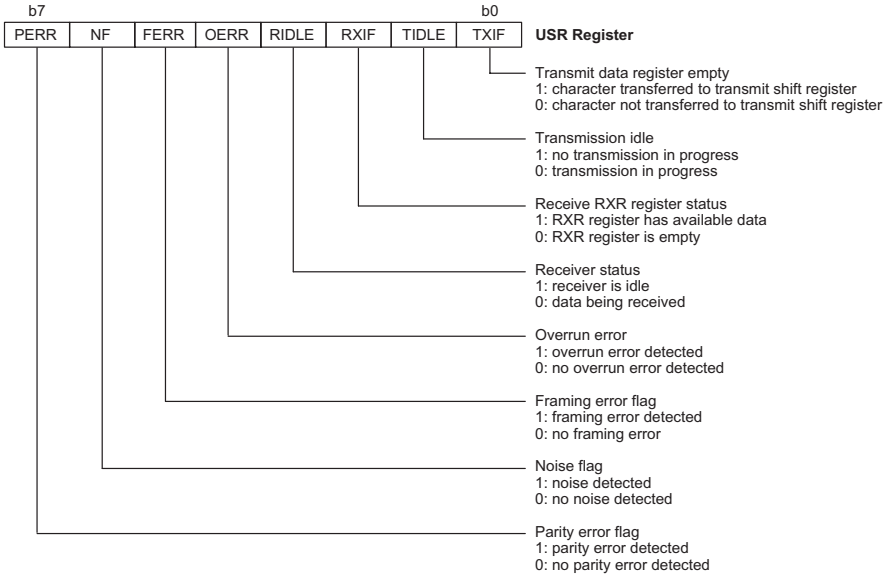


UART Data Transfer Scheme

- USR register
  The USR register is the status register for the UART, which can be read by the program to determine the present status of the UART. All flags within the USR register are read only.

  Further explanation on each of the flags is given below:

  - TXIF
    The TXIF flag is the transmit data register empty flag. When this read only flag is ″0″ it indicates that the character is not transferred to the transmit shift registers. When the flag is ″1″ it indicates that the transmit shift register has received a character from the TXR data register. The TXIF flag is cleared by reading the UART status register (USR) with TXIF set and then writing to the TXR data register. Note that when the TXEN bit is set, the TXIF flag bit will also be set since the transmit buffer is not yet full.
  - TIDLE
    The TIDLE flag is known as the transmission complete flag. When this read only flag is ″0″ it indicates that a transmission is in progress. This flag will be set to ″1″ when the TXIF flag is ″1″ and when there is no transmit data, or break character being transmitted. When TIDLE is ″1″ the TX pin becomes idle. The TIDLE flag is cleared by reading the USR register with TIDLE set and then writing to the TXR register. The flag is not generated when a data character, or a break is queued and ready to be sent.
  - RXIF
    The RXIF flag is the receive register status flag. When this read only flag is ″0″ it indicates that the RXR read data register is empty. When the flag is ″1″ it indicates that the RXR read data register contains new data. When the contents of the shift register are transferred to the RXR register, an interrupt is generated if RIE=1 in the UCR2 register. If one or more errors are detected in the received word, the appropriate receive-related flags NF, FERR, and/or PERR are set within the same clock cycle. The RXIF flag is cleared when the USR register is read with RXIF set, followed by a read from the RXR register, and if the RXR register has no data available.
  - RIDLE
    The RIDLE flag is the receiver status flag. When this read only flag is ″0″ it indicates that the receiver is between the initial detection of the start bit and the completion of the stop bit. When the flag is ″1″ it indicates that the receiver is idle. Between the completion of the stop bit and the detection of the next start bit, the RIDLE bit is ″1″ indicating that the UART is idle.
  - OERR
    The OERR flag is the overrun error flag, which indicates when the receiver buffer has overflowed. When this read only flag is ″0″ there is no overrun error. When the flag is ″1″ an overrun error occurs which will inhibit further transfers to the RXR receive data register. The flag is cleared by a software sequence, which is a read to the status register USR followed by an access to the RXR data register.
  - FERR
    The FERR flag is the framing error flag. When this read only flag is ″0″ it indicates no framing error. When the flag is ″1″ it indicates that a framing error has been detected for the current character. The flag can also be cleared by a software sequence which will involve a read to the USR status register followed by an access to the RXR data register.
  - NF
    The NF flag is the noise flag. When this read only flag is ″0″ it indicates a no noise condition. When the flag is ″1″ it indicates that the UART has detected noise on the receiver input. The NF flag is set during the same cycle as the RXIF flag but will not be set in the case of an overrun. The NF flag can be cleared by a software sequence which will involve a read to the USR status register, followed by an access to the RXR data register.

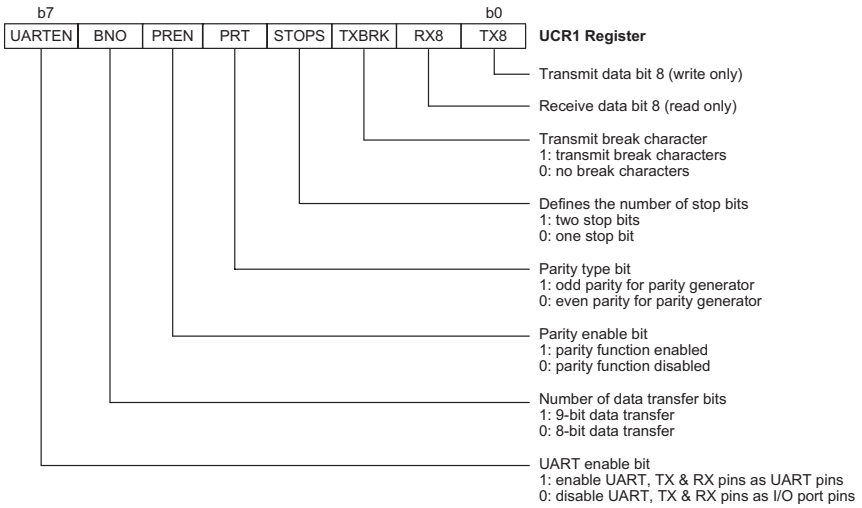


- PERR
  The PERR flag is the parity error flag. When this read only flag is ″0″ it indicates that a parity error has not been detected. When the flag is ″1″ it indicates that the parity of the received word is incorrect. This error flag is applicable only if Parity mode (odd or even) is selected. The flag can also be cleared by a software sequence which involves a read to the USR status register, followed by an access to the RXR data register.

- UCR1 register
  The UCR1 register together with the UCR2 register are the two UART control registers that are used to set the various options for the UART function, such as overall on/off control, parity control, data transfer bit length etc.

  Further explanation on each of the bits is given below:

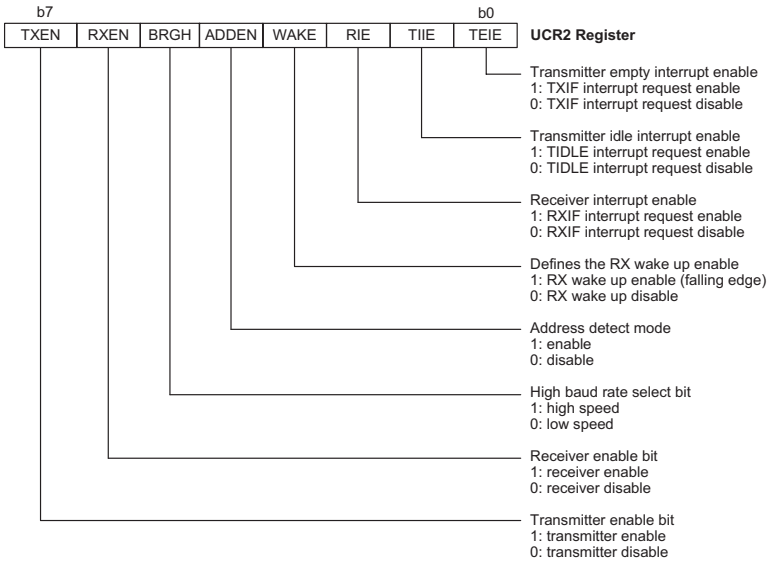
  - TX8
    This bit is only used if 9-bit data transfers are used, in which case this bit location will store the 9th bit of the transmitted data, known as TX8. The BNO bit is used to determine whether data transfers are in 8-bit or 9-bit format.
  - RX8
    This bit is only used if 9-bit data transfers are used, in which case this bit location will store the 9th bit of the received data, known as RX8. The BNO bit is used to determine whether data transfers are in 8-bit or 9-bit format.
  - TXBRK
    The TXBRK bit is the Transmit Break Character bit. When this bit is ″0″ there are no break characters and the TX pin operates normally. When the bit is ″1″ there are transmit break characters and the transmitter will send logic zeros. When equal to ″1″ after the buffered data has been transmitted, the transmitter output is held low for a minimum of a 13-bit length and until the TXBRK bit is reset.
  - STOPS
    This bit determines if one or two stop bits are to be used. When this bit is equal to ″1″ two stop bits are used, if the bit is equal to ″0″ then only one stop bit is used.
  - PRT
    This is the parity type selection bit. When this bit is equal to ″1″ odd parity will be selected, if the bit is equal to ″0″ then even parity will be selected.
  - PREN
    This is parity enable bit. When this bit is equal to ″1″ the parity function will be enabled, if the bit is equal to ″0″ then the parity function will be disabled.
  - BNO
    This bit is used to select the data length format, which can have a choice of either 8-bits or 9-bits. If this bit is equal to ″1″ then a 9-bit data length will be selected, if the bit is equal to ″0″ then an 8-bit data length will be selected. If 9-bit data length is selected then bits RX8 and TX8 will be used to store the 9th bit of the received and transmitted data respectively.
  - UARTEN
    The UARTEN bit is the UART enable bit. When the bit is ″0″ the UART will be disabled and the RX and TX pins will function as General Purpose I/O pins. When the bit is ″1″ the UART will be enabled and the TX and RX pins will function as defined by the TXEN and RXEN control bits. When the UART is disabled it will empty the buffer so any character remaining in the buffer will be discarded. In addition, the baud rate counter value will be reset. When the UART is disabled, all error and status flags will be reset. The TXEN, RXEN, TXBRK, RXIF, OERR, FERR, PERR, and NF bits will be cleared, while the TIDLE, TXIF and RIDLE bits will be set. Other control bits in UCR1, UCR2, and BRG registers will remain unaffected. If the UART is active and the UARTEN bit is cleared, all pending transmissions and receptions will be terminated and the module will be reset as defined above. When the UART is re-enabled it will restart in the same configuration.



- UCR2 register
  The UCR2 register is the second of the two UART control registers and serves several purposes. One of its main functions is to control the basic enable/disable operation of the UART Transmitter and Receiver as well as enabling the various UART interrupt sources. The register also serves to control the baud rate speed, receiver wake-up enable and the address detect enable.

  Further explanation on each of the bits is given below:

  - TEIE
    This bit enables or disables the transmitter empty interrupt. If this bit is equal to ″1″ when the transmitter empty TXIF flag is set, due to a transmitter empty condition, the UART interrupt request flag will be set. If this bit is equal to ″0″ the UART interrupt request flag will not be influenced by the condition of the TXIF flag.
  - TIIE
    This bit enables or disables the transmitter idle interrupt. If this bit is equal to ″1″ when the transmitter idle TIDLE flag is set, the UART interrupt request flag will be set. If this bit is equal to ″0″ the UART interrupt request flag will not be influenced by the condition of the TIDLE flag.
  - RIE
    This bit enables or disables the receiver interrupt. If this bit is equal to ″1″ when the receiver overrun OERR flag or receive data available RXIF flag is set, the UART interrupt request flag will be set. If this bit is equal to ″0″ the UART interrupt will not be influenced by the condition of the OERR or RXIF flags.
  - WAKE
    This bit enables or disables the receiver wake-up function. If this bit is equal to ″1″ and if the MCU is in the Power Down Mode, a low going edge on the RX input pin will wake-up the device. If this bit is equal to ″0″ and if the MCU is in the Power Down Mode, any edge transitions on the RX pin will not wake-up the device.
  - ADDEN
    The ADDEN bit is the address detect mode bit. When this bit is ″1″ the address detect mode is enabled. When this occurs, if the 8th bit, which corresponds to RX7 if BNO=0, or the 9th bit, which corresponds to RX8 if BNO=1, has a value of ″1″ then the received word will be identified as an address, rather than data. If the corresponding interrupt is enabled, an interrupt request will be generated each time the received word has the address bit set, which is the 8 or 9 bit depending on the value of BNO. If the address bit is ″0″ an interrupt will not be generated, and the received data will be discarded.
  - BRGH
    The BRGH bit selects the high or low speed mode of the Baud Rate Generator. This bit, together with the value placed in the BRG register, controls the Baud Rate of the UART. If this bit is equal to ″1″ the high speed mode is selected. If the bit is equal to ″0″ the low speed mode is selected.
  - RXEN
    The RXEN bit is the Receiver Enable Bit. When this bit is equal to ″0″ the receiver will be disabled with any pending data receptions being aborted. In addition the buffer will be reset. In this situation the RX pin can be used as a general purpose I/O pin. If the RXEN bit is equal to ″1″ the receiver will be enabled and if the UARTEN bit is equal to ″1″ the RX pin will be controlled by the UART. Clearing the RXEN bit during a transmission will cause the data reception to be aborted and will reset the receiver. If this occurs, the RX pin can be used as a general purpose I/O pin.



- TXEN
  The TXEN bit is the Transmitter Enable Bit. When this bit is equal to ″0″ the transmitter will be disabled with any pending transmissions being aborted. In addition the buffer will be reset. In this situation the TX pin can be used as a general purpose I/O pin. If the TXEN bit is equal to ″1″ the transmitter will be enabled and if the UARTEN bit is equal to ″1″ the TX pin will be controlled by the UART. Clearing the TXEN bit during a transmission will cause the transmission to be aborted and will reset the transmitter. If this occurs, the TX pin can be used as a general purpose I/O pin.

- Baud rate generator
  To setup the speed of the serial data communication, the UART function contains its own dedicated baud rate generator. The baud rate is controlled by its own internal free running 8-bit timer, the period of which is determined by two factors. The first of these is the value placed in the BRG register and the second is the value of the BRGH bit within the UCR2 control register. The BRGH bit decides, if the baud rate generator is to be used in a high speed mode or low speed mode, which in turn determines the formula that is used to calculate the baud rate. The value in the BRG register determines the division factor, N, which is used in the following baud rate calculation formula. Note that N is the decimal value placed in the BRG register and has a range of between 0 and 255.

UCR2 BRGH Bit	0	1
Baud Rate	$\frac{f_{SYS}}{[64\ (N+1)]}$	$\frac{f_{SYS}}{[16\ (N+1)]}$

By programming the BRGH bit which allows selection of the related formula and programming the required value in the BRG register, the required baud rate can be setup. Note that because the actual baud rate is determined using a discrete value, N, placed in the BRG register, there will be an error associated between the actual and requested value. The following example shows how the BRG register value N and the error value can be calculated.

Calculating the register and error values

For a clock frequency of 8MHz, and with BRGH set to ″0″ determine the BRG register value N, the actual baud rate and the error value for a desired baud rate of 9600.

From the above table the desired baud rate BR

$$= \frac{f_{SYS}}{[64\ (N+1)]}$$

Re-arranging this equation gives $N = \frac{f_{SYS}}{(BRx64)} - 1$

Giving a value for $N = \frac{8000000}{(9600x64)} - 1 = 12.0208$

To obtain the closest value, a decimal value of 12 should be placed into the BRG register. This gives an actual or calculated baud rate value of

$$BR = \frac{8000000}{[64(12+1)]} = 9615$$

Therefore the error is equal to $\frac{9615 - 9600}{9600} = 0.16\%$

The following tables show actual values of baud rate and error values for the two values of BRGH.

Baud Rate K/BPS	Baud Rates for BRGH=0											
	f_{SYS}=8MHz			f_{SYS}=7.159MHz			f_{SYS}=4MHz			f_{SYS}=3.579545MHz		
	BRG	Kbaud	Error	BRG	Kbaud	Error	BRG	Kbaud	Error	BRG	Kbaud	Error
0.3	—	—	—	—	—	—	207	0.300	0.00	185	0.300	0.00
1.2	103	1.202	0.16	92	1.203	0.23	51	1.202	0.16	46	1.19	-0.83
2.4	51	2.404	0.16	46	2.38	-0.83	25	2.404	0.16	22	2.432	1.32
4.8	25	4.807	0.16	22	4.863	1.32	12	4.808	0.16	11	4.661	-2.9
9.6	12	9.615	0.16	11	9.322	-2.9	6	8.929	-6.99	5	9.321	-2.9
19.2	6	17.857	-6.99	5	18.64	-2.9	2	20.83	8.51	2	18.643	-2.9
38.4	2	41.667	8.51	2	37.29	-2.9	1	—	—	1	—	—
57.6	1	62.5	8.51	1	55.93	-2.9	0	62.5	8.51	0	55.93	-2.9
115.2	0	125	8.51	0	111.86	-2.9	—	—	—	—	—	—

Baud Rates and Error Values for BRGH = 0

Baud Rate K/BPS	Baud Rates for BRGH=1											
	f_{SYS}=8MHz			f_{SYS}=7.159MHz			f_{SYS}=4MHz			f_{SYS}=3.579545MHz		
	BRG	Kbaud	Error	BRG	Kbaud	Error	BRG	Kbaud	Error	BRG	Kbaud	Error
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	—	—	—	207	1.202	0.16	185	1.203	0.23
2.4	207	2.404	0.16	185	2.405	0.23	103	2.404	0.16	92	2.406	0.23
4.8	103	4.808	0.16	92	4.811	0.23	51	4.808	0.16	46	4.76	-0.83
9.6	51	9.615	0.16	46	9.520	-0.832	25	9.615	0.16	22	9.727	1.32
19.2	25	19.231	0.16	22	19.454	1.32	12	19.231	0.16	11	18.643	-2.9
38.4	12	38.462	0.16	11	37.287	-2.9	6	35.714	-6.99	5	37.286	-2.9
57.6	8	55.556	-3.55	7	55.93	-2.9	3	62.5	8.51	3	55.930	-2.9
115.2	3	125	8.51	3	111.86	-2.9	1	125	8.51	1	111.86	-2.9
250	1	250	0	—	—	—	0	250	0	—	—	—

Baud Rates and Error Values for BRGH = 1

- Setting up and controlling the UART
  - Introduction
    For data transfer, the UART function utilizes a non-return-to-zero, more commonly known as NRZ, format. This is composed of one start bit, eight or nine data bits, and one or two stop bits. Parity is supported by the UART hardware, and can be setup to be even, odd or no parity. For the most common data format, 8 data bits along with no parity and one stop bit, denoted as 8, N, 1, is used as the default setting, which is the setting at power-on. The number of data bits and stop bits, along with the parity, are setup by programming the corresponding BNO, PRT, PREN, and STOPS bits in the UCR1 register. The baud rate used to transmit and receive data is setup using the internal 8-bit baud rate generator, while the data is transmitted and received LSB first. Although the UART□s transmitter and receiver are functionally independent, they both use the same data format and baud rate. In all cases stop bits will be used for data transmission.
  - Enabling/disabling the UART
    The basic on/off function of the internal UART function is controlled using the UARTEN bit in the UCR1 register. As the UART transmit and receive pins, TX and RX respectively, are pin-shared with normal I/O pins, one of the basic functions of the UARTEN control bit is to control the UART function of these two pins. If the UARTEN, TXEN and RXEN bits are set, then these two I/O pins will be setup as a TX output pin and an RX input pin respectively, in effect disabling the normal I/O pin function. If no data is being transmitted on the TX pin then it will default to a logic high value.
    Clearing the UARTEN bit will disable the TX and RX pins and allow these two pins to be used as normal I/O pins. When the UART function is disabled the buffer will be reset to an empty condition, at the same time discarding any remaining residual data. Disabling the UART will also reset the error and status flags with bits TXEN, RXEN, TXBRK, RXIF, OERR, FERR, PERR and NF being cleared while bits TIDLE, TXIF and RIDLE will be set. The remaining control bits in the UCR1, UCR2 and BRG registers will remain unaffected. If the UARTEN bit in the UCR1 register is cleared while the UART is active, then all pending transmissions and receptions will be immediately suspended and the UART will be reset to a condition as defined above. If the UART is then subsequently re-enabled, it will restart again in the same configuration.
  - Data, parity and stop bit selection
    The format of the data to be transferred, is composed of various factors such as data bit length, parity on/off, parity type, address bits and the number of stop bits. These factors are determined by the setup of various bits within the UCR1 register. The BNO bit controls the number of data bits which can be set to either 8 or 9, the PRT bit controls the choice of odd or even parity, the PREN bit controls the parity on/off function and the STOPS bit decides whether one or two stop bits are to be used. The following table shows various formats for data transmission. The address bit identifies the frame as an address character. The number of stop bits, which can be either one or two, is independent of the data length.

Start Bit	Data Bits	Address Bits	Parity Bits	Stop Bit
Example of 8-bit Data Formats				
1	8	0	0	1
1	7	0	1	1
1	7	1[1]	0	1
Example of 9-bit Data Formats				
1	9	0	0	1
1	8	0	1	1
1	8	1[1]	0	1

Transmitter Receiver Data Format

The following diagram shows the transmit and receive waveforms for both 8-bit and 9-bit data formats.

- UART transmitter
  Data word lengths of either 8 or 9 bits, can be selected by programming the BNO bit in the UCR1 register. When BNO bit is set, the word length will be set to 9 bits. In this case the 9th bit, which is the MSB, needs to be stored in the TX8 bit in the UCR1 register. At the transmitter core lies the Transmitter Shift Register, more commonly known as the TSR, whose data is obtained from the transmit data register, which is known as the TXR register. The data to be transmitted is loaded into this TXR register by the application program. The TSR register is not written to with new data until the stop bit from the previous transmission has been sent out. As soon as this stop bit has been transmitted, the TSR can then be loaded with new data from the TXR register, if it is available. It should be noted that the TSR register, unlike many other registers, is not directly mapped into the Data Memory area and as such is not available to the application program for direct read/write operations. An actual transmission of data will normally be enabled when the TXEN bit is set, but the data will not be transmitted until the TXR register has been loaded with data and the baud rate generator has defined a shift clock source. However, the transmission can also be initiated by first loading data into the TXR register, after which the TXEN bit can be set. When a transmission of data begins, the TSR is normally empty, in which case a transfer to the TXR register will result in an immediate transfer to the TSR. If during a transmission the TXEN bit is cleared, the transmission will immediately cease and the transmitter will be reset. The TX output pin will then return to having a normal general purpose I/O pin function.

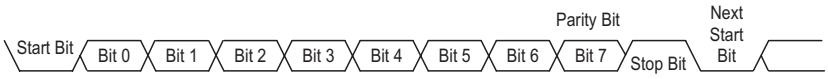
- Transmitting data
  When the UART is transmitting data, the data is shifted on the TX pin from the shift register, with the least significant bit first. In the transmit mode, the TXR register forms a buffer between the internal bus and the transmitter shift register. It should be noted that if 9-bit data format has been selected, then the MSB will be taken from the TX8 bit in the UCR1 register. The steps to initiate a data transfer can be summarized as follows:
  - Make the correct selection of the BNO, PRT, PREN and STOPS bits to define the required word length, parity type and number of stop bits.
  - Setup the BRG register to select the desired baud rate.
  - Set the TXEN bit to ensure that the TX pin is used as a UART transmitter pin and not as an I/O pin.
  - Access the USR register and write the data that is to be transmitted into the TXR register. Note that this step will clear the TXIF bit.
  - This sequence of events can now be repeated to send additional data.

  It should be noted that when TXIF=0, data will be inhibited from being written to the TXR register. Clearing the TXIF flag is always achieved using the following software sequence:
  1. A USR register access
  2. A TXR register write execution

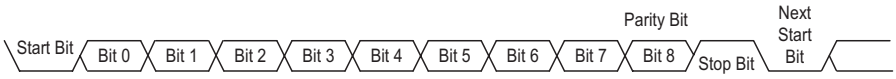
  The read-only TXIF flag is set by the UART hardware and if set indicates that the TXR register is empty and that other data can now be written into the TXR register without overwriting the previous data. If the TEIE bit is set then the TXIF flag will generate an interrupt.

  During a data transmission, a write instruction to the TXR register will place the data into the TXR register, which will be copied to the shift register at the end of the present transmission. When there is no data transmission in progress, a write instruction to the TXR register will place the data directly into the shift register, resulting in the commencement of data transmission, and the TXIF bit being immediately set. When a frame transmission is complete, which happens after stop bits are sent or after the break frame, the TIDLE bit will be set. To clear the TIDLE bit the following software sequence is used:
  1. A USR register access
  2. A TXR register write execution

  Note that both the TXIF and TIDLE bits are cleared by the same software sequence.



8-Bit Data Format



9-Bit Data Format

- Transmit break
  If the TXBRK bit is set then break characters will be sent on the next transmission. Break character transmission consists of a start bit, followed by 13× N ′0′ bits and stop bits, where N=1, 2, etc. If a break character is to be transmitted then the TXBRK bit must be first set by the application program, then cleared to generate the stop bits. Transmitting a break character will not generate a transmit interrupt. Note that a break condition length is at least 13 bits long. If the TXBRK bit is continually kept at a logic high level then the transmitter circuitry will transmit continuous break characters. After the application program has cleared the TXBRK bit, the transmitter will finish transmitting the last break character and subsequently send out one or two stop bits. The automatic logic highs at the end of the last break character will ensure that the start bit of the next frame is recognized.

- UART receiver
  - Introduction
    The UART is capable of receiving word lengths of either 8 or 9 bits. If the BNO bit is set, the word length will be set to 9 bits with the MSB being stored in the RX8 bit of the UCR1 register. At the receiver core lies the Receive Serial Shift Register, commonly known as the RSR. The data which is received on the RX external input pin, is sent to the data recovery block. The data recovery block operating speed is 16 times that of the baud rate, while the main receive serial shifter operates at the baud rate. After the RX pin is sampled for the stop bit, the received data in RSR is transferred to the receive data register, if the register is empty. The data which is received on the external RX input pin is sampled three times by a majority detect circuit to determine the logic level that has been placed onto the RX pin. It should be noted that the RSR register, unlike many other registers, is not directly mapped into the Data Memory area and as such is not available to the application program for direct read/write operations.
  - Receiving data
    When the UART receiver is receiving data, the data is serially shifted in on the external RX input pin, LSB first. In the read mode, the RXR register forms a buffer between the internal bus and the receiver shift register. The RXR register is a two byte deep FIFO data buffer, where two bytes can be held in the FIFO while a third byte can continue to be received. Note that the application program must ensure that the data is read from RXR before the third byte has been completely shifted in, otherwise this third byte will be discarded and an overrun error OERR will be subsequently indicated. The steps to initiate a data transfer can be summarized as follows:
    - Make the correct selection of BNO, PRT, PREN and STOPS bits to define the word length, parity type and number of stop bits.
    - Setup the BRG register to select the desired baud rate.
    - Set the RXEN bit to ensure that the RX pin is used as a UART receiver pin and not as an I/O pin.

    At this point the receiver will be enabled which will begin to look for a start bit.

    When a character is received the following sequence of events will occur:
    - The RXIF bit in the USR register will be set when RXR register has data available, at least one more character can be read.
    - When the contents of the shift register have been transferred to the RXR register, then if the RIE bit is set, an interrupt will be generated.
    - If during reception, a frame error, noise error, parity error, or an overrun error has been detected, then the error flags can be set.

    The RXIF bit can be cleared using the following software sequence:
    1. A USR register access
    2. An RXR register read execution
  - Receive break
    Any break character received by the UART will be managed as a framing error. The receiver will count and expect a certain number of bit times as specified by the values programmed into the BNO and STOPS bits. If the break is much longer than 13 bit times, the reception will be considered as complete after the number of bit times specified by BNO and STOPS. The RXIF bit is set, FERR is set, zeros are loaded into the receive data register, interrupts are generated if appropriate and the RIDLE bit is set. If a long break signal has been detected and the receiver has received a start bit, the data bits and the invalid stop bit, which sets the FERR flag, the receiver must wait for a valid stop bit before looking for the next start bit. The receiver will not make the assumption that the break condition on the line is the next start bit. A break is regarded as a character that contains only zeros with the FERR flag set. The break character will be loaded into the buffer and no further data will be received until stop bits are received. It should be noted that the RIDLE read only flag will go high when the stop bits have not yet been received. The reception of a break character on the UART registers will result in the following:
    - The framing error flag, FERR, will be set.
    - The receive data register, RXR, will be cleared.
    - The OERR, NF, PERR, RIDLE or RXIF flags will possibly be set.
  - Idle status
    When the receiver is reading data, which means it will be in between the detection of a start bit and the reading of a stop bit, the receiver status flag in the USR register, otherwise known as the RIDLE flag, will have a zero value. In between the reception of a stop bit and the detection of the next start bit, the RIDLE flag will have a high value, which indicates the receiver is in an idle condition.

- Receiver interrupt
  The read only receive interrupt flag RXIF in the USR register is set by an edge generated by the receiver. An interrupt is generated if RIE=1, when a word is transferred from the Receive Shift Register, RSR, to the Receive Data Register, RXR. An overrun error can also generate an interrupt if RIE=1.

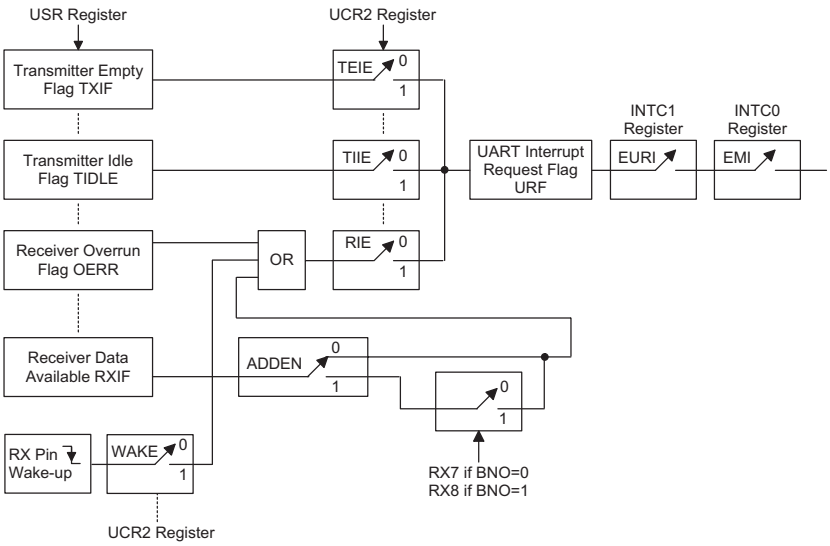
- Managing receiver errors
  Several types of reception errors can occur within the UART module, the following section describes the various types and how they are managed by the UART.
  - Overrun Error – OERR flag
    The RXR register is composed of a two byte deep FIFO data buffer, where two bytes can be held in the FIFO register, while a third byte can continue to be received. Before this third byte has been entirely shifted in, the data should be read from the RXR register. If this is not done, the overrun error flag OERR will be consequently indicated.

    In the event of an overrun error occurring, the following will happen:
    - The OERR flag in the USR register will be set.
    - The RXR contents will not be lost.
    - The shift register will be overwritten.
    - An interrupt will be generated if the RIE bit is set.

    The OERR flag can be cleared by an access to the USR register followed by a read to the RXR register.
  - Noise Error – NF Flag
    Over-sampling is used for data recovery to identify valid incoming data and noise. If noise is detected within a frame the following will occur:
    - The read only noise flag, NF, in the USR register will be set on the rising edge of the RXIF bit.
    - Data will be transferred from the Shift register to the RXR register.
    - No interrupt will be generated. However this bit rises at the same time as the RXIF bit which itself generates an interrupt.

    Note that the NF flag is reset by a USR register read operation followed by an RXR register read operation.
  - Framing Error – FERR Flag
    The read only framing error flag, FERR, in the USR register, is set if a zero is detected instead of stop bits. If two stop bits are selected, both stop bits must be high, otherwise the FERR flag will be set. The FERR flag is buffered along with the received data and is cleared on any reset.
  - Parity Error – PERR Flag
    The read only parity error flag, PERR, in the USR register, is set if the parity of the received word is incorrect. This error flag is only applicable if the parity is enabled, PREN = 1, and if the parity type, odd or even is selected. The read only PERR flag is buffered along with the received data bytes. It is cleared on any reset. It should be noted that the FERR and PERR flags are buffered along with the corresponding word and should be read before reading the data word.

- UART interrupt scheme
  The UART internal function possesses its own internal interrupt and independent interrupt vector. Several individual UART conditions can generate an internal UART interrupt. These conditions are, a transmitter data register empty, transmitter idle, receiver data available, receiver overrun, address detect and an RX pin wake-up. When any of these conditions are created, if the UART interrupt is enabled and the stack is not full, the program will jump to the UART interrupt vector where it can be serviced before returning to the main program. Four of these conditions, have a corresponding USR register flag, which will generate a UART interrupt if its associated interrupt enable flag in



UART Interrupt Scheme

the UCR2 register is set. The two transmitter interrupt conditions have their own corresponding enable bits, while the two receiver interrupt conditions have a shared enable bit. These enable bits can be used to mask out individual UART interrupt sources.

The address detect condition, which is also a UART interrupt source, does not have an associated flag, but will generate a UART interrupt when an address detect condition occurs if its function is enabled by setting the ADDEN bit in the UCR2 register. An RX pin wake-up, which is also a UART interrupt source, does not have an associated flag, but will generate a UART interrupt if the microcontroller is woken up by a low going edge on the RX pin, if the WAKE and RIE bits in the UCR2 register are set. Note that in the event of an RX wake-up interrupt occurring, there will be a delay of 1024 system clock cycles before the system resumes normal operation.

Note that the USR register flags are read only and cannot be cleared or set by the application program, neither will they be cleared when the program jumps to the corresponding interrupt servicing routine, as is the case for some of the other interrupts. The flags will be cleared automatically when certain actions are taken by the UART, the details of which are given in the UART register section. The overall UART interrupt can be disabled or enabled by the EURI bit in the INTC1 interrupt control register to prevent a UART interrupt from occurring.

- Address detect mode
  Setting the Address Detect Mode bit, ADDEN, in the UCR2 register, enables this special mode. If this bit is enabled then an additional qualifier will be placed on the generation of a Receiver Data Available interrupt, which is requested by the RXIF flag. If the ADDEN bit is enabled, then when data is available, an interrupt will only be generated, if the highest received bit has a high value. Note that the EURI and EMI interrupt enable bits must also be enabled for correct interrupt generation. This highest address bit is the 9th bit if BNO=1 or the 8th bit if BNO=0. If this bit is high, then the received word will be defined as an address rather than data. A Data Available interrupt will be generated every time the last bit of the received word is set. If the ADDEN bit is not enabled, then a Receiver Data Available interrupt will be generated each time the RXIF flag is set, irrespective of the data last bit status. The address detect mode and parity enable are mutually exclusive functions. Therefore if the address detect mode is enabled, then to ensure correct operation, the parity function should be disabled by resetting the parity enable bit to zero.

ADDEN	Bit 9 if BNO=1, Bit 8 if BNO=0	UART Interrupt Generated
0	0	√
	1	√
1	0	X
	1	√

ADDEN Bit Function

- UART operation in power down mode
  When the MCU is in the Power Down Mode the UART will cease to function. When the device enters the Power Down Mode, all clock sources to the module are shutdown. If the MCU enters the Power Down Mode while a transmission is still in progress, then the transmission will be terminated and the external TX transmit pin will be forced to a logic high level. In a similar way, if the MCU enters the Power Down Mode while receiving data, then the reception of data will likewise be terminated. When the MCU enters the Power Down Mode, note that the USR, UCR1, UCR2, transmit and receive registers, as well as the BRG register will not be affected.

  The UART function contains a receiver RX pin wake-up function, which is enabled or disabled by the WAKE bit in the UCR2 register. If this bit, along with the UART enable bit, UARTEN, the receiver enable bit, RXEN and the receiver interrupt bit, RIE, are all set before the MCU enters the Power Down Mode, then a falling edge on the RX pin will wake-up the MCU from the Power Down Mode. Note that as it takes 1024 system clock cycles after a wake-up, before normal microcontroller operation resumes, any data received during this time on the RX pin will be ignored.

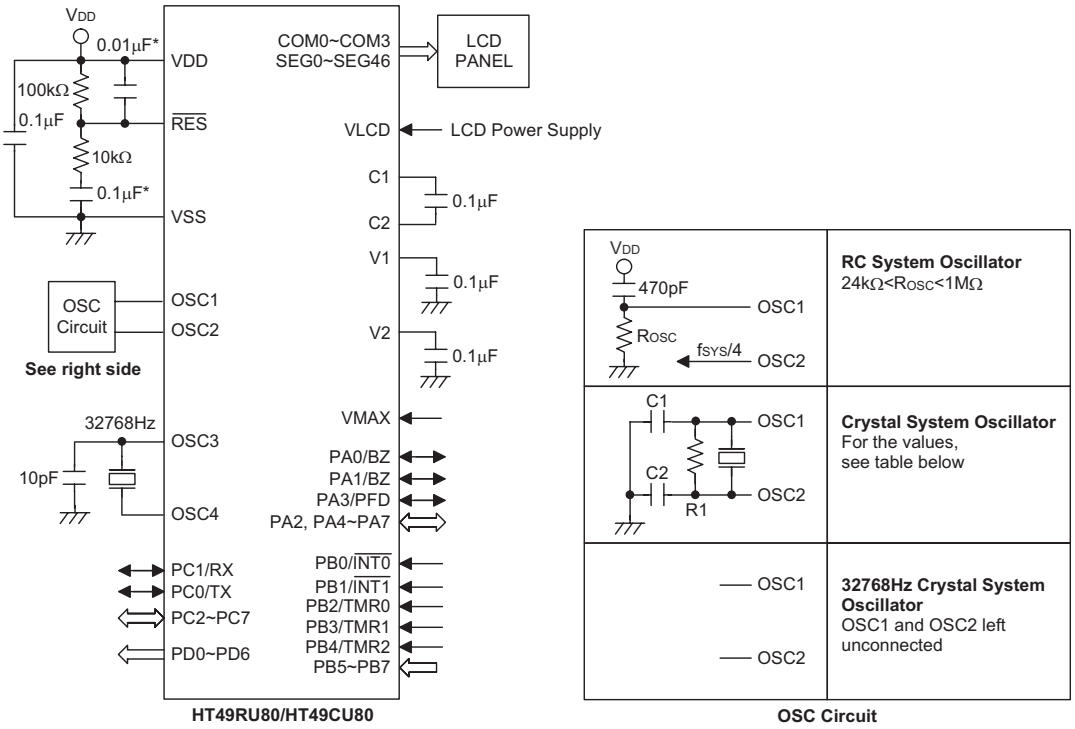
  For a UART wake-up interrupt to occur, in addition to the bits for the wake-up being set, the global interrupt enable bit, EMI, and the UART interrupt enable bit, EURI must also be set. If these two bits are not set then only a wake up event will occur and no interrupt will be generated. Note also that as it takes 1024 system clock cycles after a wake-up before normal microcontroller resumes, the UART interrupt will not be generated until after this time has elapsed.

Configuration Options

The following shows the options in the device. All these options should be defined in order to ensure proper functioning system.Configuration options refer to certain options within the MCU that are programmed into the device during the programming process. During the development process, these options are selected using the HT-IDE software development tools. As these options are programmed into the device using the hardware programming tools, once they are selected they cannot be changed later as the application software has no control over the configuration options. All options must be defined for proper system function, the details of which are shown in the table.

Options
I/O Options
PA0~PA7 wake-up enable/disable PA0~PA3 CMOS/NMOS selection PA0~PA3 pull-high enable/disable PC0~PC3 CMOS/NMOS selection PC4~PC7 CMOS/NMOS selection PC0~PC3 pull-high enable/disable PC4~PC7 pull-high enable/disable
Watchdog Options
WDT function: enable or disable
CLRWDT instructions: one or instructions
Oscillator Options
OSC type selection: RC or crystal
fsys clock source: OSC or RTC
fs internal clock source: $f_{SYS}/4$, RTC OSC or WDT OSC
PFD Options
PFD output enable: enable or disable
PFD clock selection: Timer/Event Counter 0 or Timer/Event Counter 1
Timer Options
Timer/Event Counter 0 clock source: $f_{SYS}/4$ or f_{SYS}
Timer/Event Counter 1 clock source: Timer/Event Counter 0 overflow, Time Base out or f_{SYS}
Timer Base Options
Time Base frequency: $f_S/2^{12}$ ~ $f_S/2^{15}$
Buzzer Options
Buzzer output enable: enable or disable
Buzzer frequency : $f_S/2^2$ ~ $f_S/2^9$
LVD/LVR Options
LVR function reset: enable or disable
Low Voltage Detect: enable or disable
LCD Options
LCD clock: $f_S/2^2$ ~ $f_S/2^8$
LCD duty: 1/2, 1/3, 1/4
LCD bias: 1/2, 1/3
LCD bias type: R type or C type
LCD on/off during power down: enable or disable
LCD segment SEG40~SEG46 output or PD0~PD6 output
HALT mode oscillator enable or disable

Application Circuits



The following table shows the C1, C2 and R1 values corresponding to the different crystal values. (For reference only)

Crystal or Resonator	C1, C2	R1
4MHz Crystal	0pF	12kΩ
4MHz Resonator	10pF	12kΩ
3.58MHz Crystal	0pF	12kΩ
3.58MHz Resonator	25pF	12kΩ
2MHz Crystal & Resonator	25pF	12kΩ
1MHz Crystal	35pF	14kΩ
480kHz Resonator	100pF	14kΩ
455kHz Resonator	200pF	12kΩ
429kHz Resonator	200pF	12kΩ
400kHz Resonator	300pF	12kΩ
The function of the resistor R1 is to ensure that the oscillator will switch off should low voltage conditions occur. Such a low voltage, as mentioned here, is one which is less than the lowest value of the MCU operating voltage. Note however that if the LVR is enabled then R1 can be removed.		

Note: The resistance and capacitance for reset circuit should be designed in such a way as to ensure that the VDD is stable and remains within a valid operating voltage range before bringing $\overline{RES}$ to high.

"*" Make the length of the wiring, which is connected to the $\overline{RES}$ pin as short as possible, to avoid noise interference.

Instruction Set Summary

Mnemonic	Description	Instruction Cycle	Flag Affected
Arithmetic			
ADD A,[m]	Add data memory to ACC	1	Z,C,AC,OV
ADDM A,[m]	Add ACC to data memory	1(1)	Z,C,AC,OV
ADD A,x	Add immediate data to ACC	1	Z,C,AC,OV
ADC A,[m]	Add data memory to ACC with carry	1	Z,C,AC,OV
ADCM A,[m]	Add ACC to data memory with carry	1(1)	Z,C,AC,OV
SUB A,x	Subtract immediate data from ACC	1	Z,C,AC,OV
SUB A,[m]	Subtract data memory from ACC	1	Z,C,AC,OV
SUBM A,[m]	Subtract data memory from ACC with result in data memory	1(1)	Z,C,AC,OV
SBC A,[m]	Subtract data memory from ACC with carry	1	Z,C,AC,OV
SBCM A,[m]	Subtract data memory from ACC with carry and result in data memory	1(1)	Z,C,AC,OV
DAA [m]	Decimal adjust ACC for addition with result in data memory	1(1)	C
Logic Operation			
AND A,[m]	AND data memory to ACC	1	Z
OR A,[m]	OR data memory to ACC	1	Z
XOR A,[m]	Exclusive-OR data memory to ACC	1	Z
ANDM A,[m]	AND ACC to data memory	1(1)	Z
ORM A,[m]	OR ACC to data memory	1(1)	Z
XORM A,[m]	Exclusive-OR ACC to data memory	1(1)	Z
AND A,x	AND immediate data to ACC	1	Z
OR A,x	OR immediate data to ACC	1	Z
XOR A,x	Exclusive-OR immediate data to ACC	1	Z
CPL [m]	Complement data memory	1(1)	Z
CPLA [m]	Complement data memory with result in ACC	1	Z
Increment & Decrement			
INCA [m]	Increment data memory with result in ACC	1	Z
INC [m]	Increment data memory	1(1)	Z
DECA [m]	Decrement data memory with result in ACC	1	Z
DEC [m]	Decrement data memory	1(1)	Z
Rotate			
RRA [m]	Rotate data memory right with result in ACC	1	None
RR [m]	Rotate data memory right	1(1)	None
RRCA [m]	Rotate data memory right through carry with result in ACC	1	C
RRC [m]	Rotate data memory right through carry	1(1)	C
RLA [m]	Rotate data memory left with result in ACC	1	None
RL [m]	Rotate data memory left	1(1)	None
RLCA [m]	Rotate data memory left through carry with result in ACC	1	C
RLC [m]	Rotate data memory left through carry	1(1)	C
Data Move			
MOV A,[m]	Move data memory to ACC	1	None
MOV [m],A	Move ACC to data memory	1(1)	None
MOV A,x	Move immediate data to ACC	1	None
Bit Operation			
CLR [m].i	Clear bit of data memory	1(1)	None
SET [m].i	Set bit of data memory	1(1)	None

Mnemonic	Description	Instruction Cycle	Flag Affected
Branch			
JMP addr	Jump unconditionally	2	None
SZ [m]	Skip if data memory is zero	1(2)	None
SZA [m]	Skip if data memory is zero with data movement to ACC	1(2)	None
SZ [m].i	Skip if bit i of data memory is zero	1(2)	None
SNZ [m].i	Skip if bit i of data memory is not zero	1(2)	None
SIZ [m]	Skip if increment data memory is zero	1(3)	None
SDZ [m]	Skip if decrement data memory is zero	1(3)	None
SIZA [m]	Skip if increment data memory is zero with result in ACC	1(2)	None
SDZA [m]	Skip if decrement data memory is zero with result in ACC	1(2)	None
CALL addr	Subroutine call	2	None
RET	Return from subroutine	2	None
RET A,x	Return from subroutine and load immediate data to ACC	2	None
RETI	Return from interrupt	2	None
Table Read			
TABRDC [m]	Read ROM code (current page) to data memory and TBLH	2(1)	None
TABRDL [m]	Read ROM code (last page) to data memory and TBLH	2(1)	None
Miscellaneous			
NOP	No operation	1	None
CLR [m]	Clear data memory	1(1)	None
SET [m]	Set data memory	1(1)	None
CLR WDT	Clear Watchdog Timer	1	TO,PDF
CLR WDT1	Pre-clear Watchdog Timer	1	TO(4),PDF(4)
CLR WDT2	Pre-clear Watchdog Timer	1	TO(4),PDF(4)
SWAP [m]	Swap nibbles of data memory	1(1)	None
SWAPA [m]	Swap nibbles of data memory with result in ACC	1	None
HALT	Enter power down mode	1	TO,PDF

Note: x: Immediate data

m: Data memory address

A: Accumulator

i: 0~7 number of bits

addr: Program memory address

√: Flag is affected

–: Flag is not affected

(1): If a loading to the PCL register occurs, the execution cycle of instructions will be delayed for one more cycle (four system clocks).

(2): If a skipping to the next instruction occurs, the execution cycle of instructions will be delayed for one more cycle (four system clocks). Otherwise the original instruction cycle is unchanged.

(3): (1) and (2)

(4): The flags may be affected by the execution status. If the Watchdog Timer is cleared by executing the CLR WDT1 or CLR WDT2 instruction, the TO and PDF are cleared.
Otherwise the TO and PDF flags remain unchanged.

Instruction Definition

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

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADCM A,[m] Add the 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)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADD A,[m] Add data memory to the 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)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADD A,x Add immediate data to the 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)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADDM A,[m] Add the accumulator to the 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)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

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

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

Operation ACC ← ACC ″AND″ [m]

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

AND A,x Logical AND immediate data to the accumulator

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

Operation ACC ← ACC ″AND″ x

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

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

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

Operation [m] ← ACC ″AND″ [m]

Affected flag(s)

TO	PDF	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 ← Program Counter+1
Program Counter ← addr

Affected flag(s)

TO	PDF	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)

TO	PDF	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)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

CLR WDT Clear Watchdog Timer

Description The WDT is cleared (clears the WDT). The power down bit (PDF) and time-out bit (TO) are cleared.

Operation WDT ← 00H
PDF and TO ← 0

Affected flag(s)

TO	PDF	OV	Z	AC	C
0	0	—	—	—	—

CLR WDT1 Preclear Watchdog Timer

Description Together with CLR WDT2, clears the WDT. PDF and TO are also cleared. Only execution of this instruction without the other preclear instruction just sets the indicated flag which implies this instruction has been executed and the TO and PDF flags remain unchanged.

Operation WDT ← 00H*
PDF and TO ← 0*

Affected flag(s)

TO	PDF	OV	Z	AC	C
0*	0*	—	—	—	—

CLR WDT2 Preclear Watchdog Timer

Description Together with CLR WDT1, clears the WDT. PDF and TO are also cleared. Only execution of this instruction without the other preclear instruction, sets the indicated flag which implies this instruction has been executed and the TO and PDF flags remain unchanged.

Operation WDT ← 00H*
PDF and TO ← 0*

Affected flag(s)

TO	PDF	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 contained a 1 are changed to 0 and vice-versa.

Operation [m] ← $[\overline{m}]$

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

CPLA [m] Complement data memory and place 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 ← $[\overline{m}]$

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

DAA [m] Decimal-Adjust accumulator for addition

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

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

Affected flag(s)

TO	PDF	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)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

DECA [m] Decrement data memory and place 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)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

HALT Enter power down mode

Description This instruction stops 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 (PDF) is set and the WDT time-out bit (TO) is cleared.

Operation Program Counter ← Program Counter+1
PDF ← 1
TO ← 0

Affected flag(s)

TO	PDF	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)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

INCA [m] Increment data memory and place 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)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

JMP addr Directly jump

Description The program counter are replaced with the directly-specified address unconditionally, and control is passed to this destination.

Operation Program Counter ←addr

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

MOV A,[m] Move data memory to the accumulator

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

Operation ACC ← [m]

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

MOV A,x Move immediate data to the accumulator

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

Operation ACC ← x

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

MOV [m],A Move the accumulator to data memory

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

Operation [m] ←ACC

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

NOP No operation

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

Operation Program Counter ← Program Counter+1

Affected flag(s)

TO	PDF	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) perform a bitwise logical_OR operation. The result is stored in the accumulator.

Operation ACC ← ACC ″OR″ [m]

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

OR A,x Logical OR immediate data to the accumulator

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

Operation ACC ← ACC ″OR″ x

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

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

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

Operation [m] ←ACC ″OR″ [m]

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

RET Return from subroutine

Description The program counter is restored from the stack. This is a 2-cycle instruction.

Operation Program Counter ← Stack

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RET A,x Return and place immediate data in the accumulator

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

Operation Program Counter ← Stack
ACC ← x

Affected flag(s)

TO	PDF	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.

Operation Program Counter ← Stack
EMI ← 1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RL [m] Rotate data memory left

Description The contents of the specified data memory are rotated 1 bit left with bit 7 rotated into bit 0.

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

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

Description Data in the specified data memory is rotated 1 bit left with bit 7 rotated into bit 0, leaving the rotated result in the accumulator. The contents of the data memory remain unchanged.

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

Affected flag(s)

TO	PDF	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 rotated 1 bit left. Bit 7 replaces the carry bit; the original carry flag is rotated into the bit 0 position.

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

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

Description Data in the specified data memory and the carry flag are rotated 1 bit left. Bit 7 replaces the carry bit and the original carry flag is rotated into bit 0 position. The rotated result is stored in the accumulator but the contents of the data memory remain unchanged.

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

RR [m] Rotate data memory right

Description The contents of the specified data memory are rotated 1 bit right with bit 0 rotated to bit 7.

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RRA [m] Rotate right and place result in the accumulator

Description Data in the specified data memory is rotated 1 bit right with bit 0 rotated into bit 7, leaving the rotated result in the accumulator. The contents of the data memory remain unchanged.

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

Affected flag(s)

TO	PDF	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 1 bit right. Bit 0 replaces the carry bit; the original carry flag is rotated into the bit 7 position.

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

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

Description Data of the specified data memory and the carry flag are rotated 1 bit right. Bit 0 replaces the carry bit and the original carry flag is rotated into the bit 7 position. The rotated result is stored in the accumulator. The contents of the data memory remain unchanged.

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

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

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

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

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

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

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SDZ [m] Skip if decrement data memory is 0

Description The contents of the specified data memory are decremented by 1. If the result is 0, the next instruction is skipped. If the result is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SDZA [m] Decrement data memory and place result in ACC, skip if 0

Description The contents of the specified data memory are decremented by 1. If the result is 0, the next instruction is skipped. The result is stored in the accumulator but the data memory remains unchanged. If the result is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SET [m] Set data memory

Description Each bit of the specified data memory is set to 1.

Operation [m] ← FFH

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SET [m]. i Set bit of data memory

Description Bit i of the specified data memory is set to 1.

Operation [m].i ← 1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SIZ [m] Skip if increment data memory is 0

Description The contents of the specified data memory are incremented by 1. If the result is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SIZA [m] Increment data memory and place result in ACC, skip if 0

Description The contents of the specified data memory are incremented by 1. If the result is 0, the next instruction is skipped and the result is stored in the accumulator. The data memory remains unchanged. If the result is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SNZ [m].i Skip if bit i of the data memory is not 0

Description If bit i of the specified data memory is not 0, the next instruction is skipped. If bit i of the data memory is not 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation Skip if [m].i≠0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SUB A,[m] Subtract data memory from the 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)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SUBM A,[m] Subtract data memory from the 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)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SUB A,x Subtract immediate data from the 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)

TO	PDF	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 (1 of the data memories) are interchanged.

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

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

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZ [m] Skip if data memory is 0

Description If the contents of the specified data memory are 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation Skip if [m]=0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZA [m] Move data memory to ACC, skip if 0

Description The contents of the specified data memory are copied to the accumulator. If the contents is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation Skip if [m]=0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZ [m].i Skip if bit i of the data memory is 0

Description If bit i of the specified data memory is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation Skip if [m].i=0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

TABRDC [m] Move the ROM code (current page) to TBLH and data memory

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

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

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

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

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

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

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

Operation ACC ← ACC ″XOR″ [m]

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

XORM A,[m] Logical XOR data memory with the 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 0 flag is affected.

Operation [m] ← ACC ″XOR″ [m]

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

XOR A,x Logical XOR immediate data to the accumulator

Description Data in the accumulator and the specified data perform a bitwise logical Exclusive_OR operation. The result is stored in the accumulator. The 0 flag is affected.

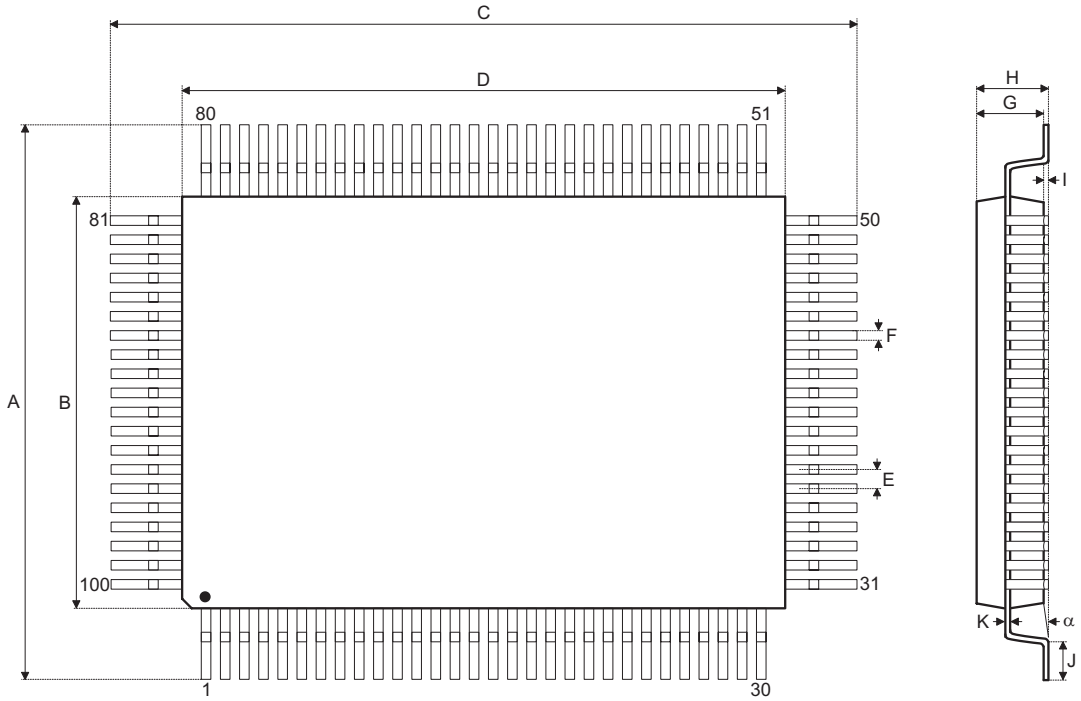
Operation ACC ← ACC ″XOR″ x

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

Package Information

100-pin QFP (14×20) Outline Dimensions



Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	18.50	—	19.20
B	13.90	—	14.10
C	24.50	—	25.20
D	19.90	—	20.10
E	—	0.65	—
F	—	0.30	—
G	2.50	—	3.10
H	—	—	3.40
I	—	0.10	—
J	1	—	1.40
K	0.10	—	0.20
α	0°	—	7°

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

Holtek Semiconductor Inc. (Taipei Sales Office)
4F-2, No. 3-2, YuanQu St., Nankang Software Park, Taipei 115, Taiwan
Tel: 886-2-2655-7070
Fax: 886-2-2655-7373
Fax: 886-2-2655-7383 (International sales hotline)

Holtek Semiconductor Inc. (Shanghai Sales Office)
7th Floor, Building 2, No.889, Yi Shan Rd., Shanghai, China 200233
Tel: 86-21-6485-5560
Fax: 86-21-6485-0313
http://www.holtek.com.cn

Holtek Semiconductor Inc. (Shenzhen Sales Office)
5/F, Unit A, Productivity Building, Cross of Science M 3rd Road and Gaoxin M 2nd Road, Science Park, Nanshan District, Shenzhen, China 518057
Tel: 86-755-8616-9908, 86-755-8616-9308
Fax: 86-755-8616-9722

Holtek Semiconductor Inc. (Beijing Sales Office)
Suite 1721, Jinyu Tower, A129 West Xuan Wu Men Street, Xicheng District, Beijing, China 100031
Tel: 86-10-6641-0030, 86-10-6641-7751, 86-10-6641-7752
Fax: 86-10-6641-0125

Holtek Semiconductor Inc. (Chengdu Sales Office)
709, Building 3, Champagne Plaza, No.97 Dongda Street, Chengdu, Sichuan, China 610016
Tel: 86-28-6653-6590
Fax: 86-28-6653-6591

Holmate Semiconductor, Inc. (North America Sales Office)
46729 Fremont Blvd., Fremont, CA 94538
Tel: 1-510-252-9880
Fax: 1-510-252-9885
http://www.holmate.com

Copyright © 2007 by HOLTEK SEMICONDUCTOR INC.

The information appearing in this Data Sheet is believed to be accurate at the time of publication. However, Holtek assumes no responsibility arising from the use of the specifications described. The applications mentioned herein are used solely for the purpose of illustration and Holtek makes no warranty or representation that such applications will be suitable without further modification, nor recommends the use of its products for application that may present a risk to human life due to malfunction or otherwise. Holtek's products are not authorized for use as critical components in life support devices or systems. Holtek reserves the right to alter its products without prior notification. For the most up-to-date information, please visit our web site at http://www.holtek.com.tw.