HT46RU75D-1

Dual Slope A/D Type MCU with LCD

Technical Document

- Tools Information
- FAQs
- Application Note

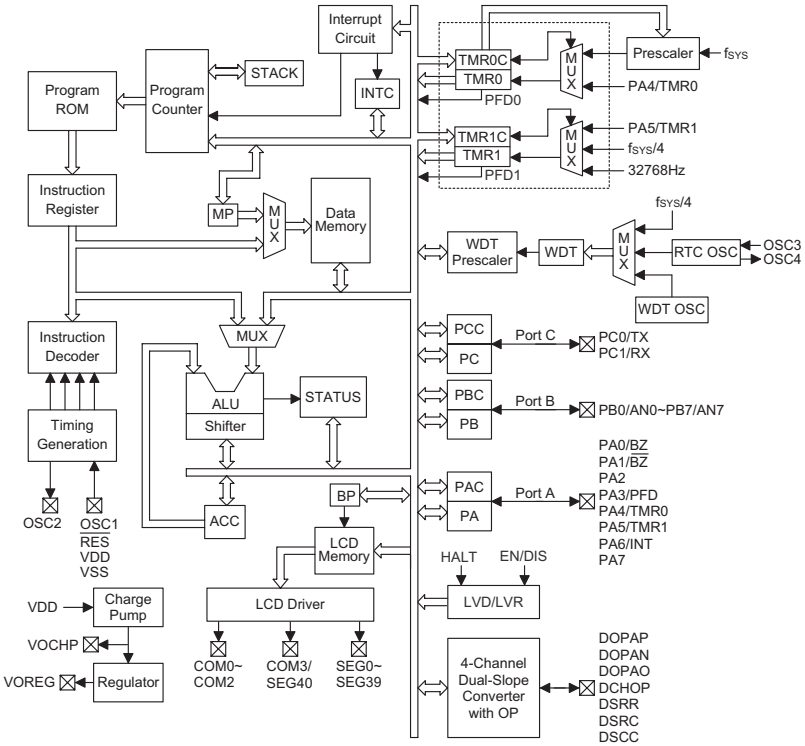
Features

- Operating voltage:
  f_{SYS} = 4MHz: 2.2V~5.5V
  f_{SYS} = 8MHz: 3.3V~5.5V
- 18 bidirectional I/O lines and two ADC inputs
- Single external interrupt input shared with I/O line
- One 16-bit and one 18-bit programmable timer/event counter with overflow interrupt and prescaler
- LCD driver with 40×4, 41×3 or 41×2 segments
- 8K×16 program memory with partial lock function
- 160×8 data memory RAM
- Four differential input channel dual slope Analog to Digital Converter with Operational Amplifier
- Watchdog Timer with regulator power supply
- Buzzer output
- External 32768Hz RTC oscillator
- Integrated RC or crystal oscillator
- Power-down and wake-up functions reduce power consumption
- Voltage regulator (3.3v) and charge pump
- Embedded voltage reference generator (1.5V)
- 16-level subroutine nesting
- Universal Asynchronous Receiver Transmitter - UART
- Bit manipulation instruction
- 16-bit table read instruction
- Up to 0.5μs instruction cycle with 8MHz system clock at V_{DD}=5V
- 63 powerful instructions
- All instructions in 1 or 2 machine cycles
- Low voltage reset/detector function
- 100-pin QFP package

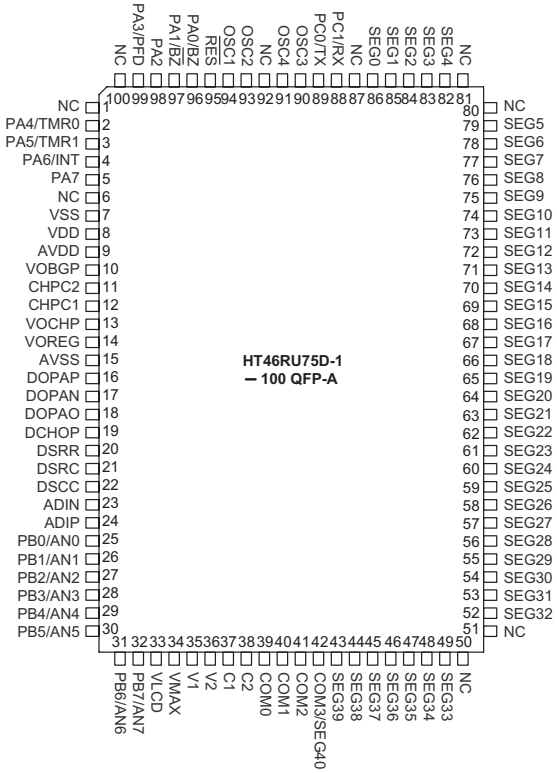
General Description

The HT46RU75D-1 is an 8-bit high performance, RISC architecture microcontroller device specifically designed for A/D with LCD applications that interface directly to analog signals, such as those from sensors. The advantages of low power consumption, I/O flexibility, timer functions, oscillator options, Dual slope A/D converter, LCD display, UART function, HALT and wake-up functions, watchdog timer, as well as low cost, provide the flexibility to suit a wide range of AD with LCD application possibilities such as sensor signal processing, scales, consumer products, subsystem controllers, etc.

Block Diagram



Pin Assignment



Pin Description

Pin Name	I/O	Options	Description
PA0/BZ PA1/$\overline{BZ}$ PA2 PA3/PFD PA4/TMR0 PA5/TMR1 PA6/INT PA7	I/O	Wake-up Pull-high Buzzer PFD	Bidirectional 8-bit input/output port. Each individual pin on this port can be configured as a wake-up input by a configuration option. Software instructions determine if the pin is a CMOS output or Schmitt trigger input. Configuration options determine which pins on this port have pull-high resistors. The BZ/$\overline{BZ}$, PFD, TMR0/TMR1 and INT pins are shared with PA0/1, PA3, PA4/5, and PA6, respectively.
PB0/AN0~ PB7/AN7	I/O	Pull-high	Bi-directional 8-bit input/output port. Software instructions determine if the pin is a CMOS output or Schmitt Trigger input. Configuration options determine which pins on the port have pull-high resistors. PB is pin-shared with the A/D input pins. The A/D inputs are selected via software instructions Once selected as an A/D input, the I/O function and pull-high resistor functions are disabled automatically.
PC0/TX PC1/RX	I/O	Pull-high	Bi-directional 8-bit input/output port. Software instructions determine if the pin is a CMOS output or Schmitt Trigger input. Configuration options determine which pins on the port have pull-high resistors. PC0 can be chosen as an I/O pin or as a UART TX output by software. PC1 can be chosen as an I/O pin or as a UART RX input by software.
VLCD	—	—	LCD power supply
VMAX	—	—	IC maximum voltage. Connected to VDD, VLCD or V1
V1, V2, C1, C2	—	—	LCD voltage pump
COM0~COM2 COM3/SEG40	O	1/2, 1/3 or 1/4 Duty	COM0~COM3 are the LCD common outputs. A configuration option selects the LCD duty-cycle. When either 1/3 or 1/2 duty is selected, the COM3/SEG40 pin will be configured as SEG40.
SEG0~SEG39	O	Segment Output	LCD driver outputs for the LCD panel segments.
VOBGP	AO	—	Band gap voltage output pin - for internal use
VOREG	O	—	Regulator output - 3.3V
VOCHP	O	—	Charge pump output - requires external capacitor
CHPC1	—	—	Charge pump capacitor, positive
CHPC2	—	—	Charge pump capacitor, negative
ADIN ADIP	AO	—	OP external Resistor Analog output connection. ADIN connects to DOPAN by a resistor. ADIP connects to DOPAP by a resistor.
DOPAN, DOPAP, DOPAO, DCHOP	AI/AO	—	Dual Slope converter pre-stage OPA related pins. DOPAN is the OPA Negative input pin, DOPAP is the OPA Positive input pin, DOPAO is the OPA output pin and DCHOP is the OPA Chopper pins.
DSRR, DSRC, DSCC	AI/AO	—	Dual slope AD converter main function RC circuit. DSRR is the input or reference signal, DSRC is the Integrator negative input, and DSCC is the comparator negative input.
OSC1 OSC2	I O	Crystal or RC	OSC1, OSC2 are connected to an external RC network or crystal for the internal system clock. If the RC system clock option is selected, pin OSC2 can be used to monitor the system clock at 1/4 frequency.
OSC3 OSC4	I O	RTC or System Clock	OSC3, 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
AVDD	—	—	Analog positive power supply
AVSS	—	—	Analog 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	—	f_{SYS}=4MHz	2.2	—	5.5	V
		—	f_{SYS}=8MHz	3.3	—	5.5	V
I_{DD1}	Operating Current (Crystal OSC)	3V	No load, f_{SYS}=4MHz Analog block off	—	0.6	1.6	mA
		5V		—	2	4	mA
I_{DD2}	Operating Current (RC OSC)	3V	No load, f_{SYS}=4MHz Analog block off	—	0.8	1.5	mA
		5V		—	2.5	4	mA
I_{DD3}	Operating Current (RC OSC)	3V	No load, f_{SYS}=8MHz Analog block off	—	2	4	mA
		5V		—	4	8	mA
I_{DD4}	Operating Current (Crystal OSC)	5V	No load, f_{SYS}=8MHz Analog block off	—	4	8	mA
I_{DD5}	Operating Current (RTC OSC)	3V	No load, f_{SYS}=32768Hz	—	0.3	0.6	mA
		5V		—	0.6	1	mA
I_{DD6}	Operating Current (ADC On)	5V	V_{REGO}=3.3V, f_{SYS}=4MHz ADC on, ADCCLK=125kHz (all other analog devices off)	—	3	5	mA
I_{STB1}	Standby Current (*f_S=f_{SYS}/4)	3V	No load, system HALT, Analog block off, LCD off	—	—	1	μA
		5V		—	—	2	μA
I_{STB2}	Standby Current (*f_S=RTC OSC)	3V	No load, system HALT, Analog block off, LCD off	—	2.5	5	μA
		5V		—	10	20	μA
I_{STB3}	Standby Current (*f_S=WDT OSC)	3V	No load, system HALT, Analog block off, LCD off	—	2	5	μA
		5V		—	6	10	μA
I_{STB4}	Standby Current (*f_S=RTC OSC)	3V	No load, system HALT, Analog block off, LCD on 1/2 bias, V_{LCD}=V_{DD} (Low bias current option)	—	17	30	μA
		5V		—	34	60	μA
I_{STB5}	Standby Current (*f_S=RTC OSC)	3V	No load, system HALT, Analog block off, LCD on 1/3 bias, V_{LCD}=V_{DD} (Low bias current option)	—	13	25	μA
		5V		—	28	50	μA
I_{STB6}	Standby Current (*f_S=WDT OSC)	3V	No load, system HALT, Analog block off, LCD on 1/2 bias, V_{LCD}=V_{DD}	—	14	25	μA
		5V		—	26	50	μA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{STB7}	Standby Current (*f_S=WDT OSC)	3V	No load, system HALT, Analog block off, LCD on 1/3 bias, $V_{LCD}=V_{DD}$	—	10	20	μA
		5V		—	19	40	μA
I_{STB8}	Standby Current (*f_{SYS}=4MHZ X′TAL/RC) (WDT Disabled), f_S=T1	3V	No load, system HALT, LCD off, UART enable System oscillator enabled by option	—	—	1200	μA
		5V		—	—	2500	μA
I_{STB9}	Standby Current (*f_{SYS}=8MHZ X′TAL/RC) (WDT Disabled), f_S=T1	3V	No load, system HALT, LCD off, UART enable System oscillator enabled by option	—	—	2000	μA
		5V		—	—	4000	μA
V_{IL1}	Input Low Voltage for I/O Ports, TMR0, TMR1 and $\overline{INT}$	—	—	0	—	$0.3V_{DD}$	V
V_{IH1}	Input High Voltage for I/O Ports, TMR0, TMR1 and $\overline{INT}$	—	—	$0.7V_{DD}$	—	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
V_{LCD}	LCD Highest Voltage	—	—	0	—	V_{DD}	V
V_{LVR1}	Low Voltage Reset 1	—	LVR option= 2.2V	1.98	2.1	2.22	V
V_{LVR2}	Low Voltage Reset 2	—	LVR option= 3.3V	2.98	3.15	3.32	V
V_{LVD1}	Low Voltage Detector 1	—	LVR option= 2.2V LVD option= LVR+0.2	2.15	2.3	2.45	V
V_{LVD2}	Low Voltage Detector 2	—	LVR option= 3.3V LVD option= LVR+0.2	3.2	3.35	3.5	V
I_{OL1}	I/O Port Segment Logic Output Sink Current	3V	$V_{OL}=0.1V_{DD}$	4	8	—	mA
		5V		10	20	—	mA
I_{OH1}	I/O Port Segment Logic Output Source Current	3V	$V_{OH}=0.9V_{DD}$	–2	–4	—	mA
		5V		–5	–10	—	mA
I_{OL2}	LCD Common and Segment Current	3V	$V_{OL}=0.1V_{DD}$	210	420	—	μA
		5V		350	700	—	μA
I_{OH2}	LCD Common and Segment Current	3V	$V_{OH}=0.9V_{DD}$	–80	–160	—	μA
		5V		–180	–360	—	μA
R_{PH}	Pull-high Resistance of I/O Ports	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ
Charge Pump and Regulator							
V_{CHPI}	Input Voltage	—	Charge pump on	2.2	—	3.6	V
			Charge pump off	3.7	—	5.5	V
V_{REGO}	Output Voltage	—	No load	3	3.3	3.6	V

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{REGDP1}	Regulator Output Voltage Drop (Compare with No Load)	—	V_{DD}=3.7V~5.5V Charge pump off Current≤10mA	—	100	—	mV
V_{REGDP2}		—	V_{DD}=2.4V~3.6V Charge pump on Current≤6mA	—	100	—	mV
Dual Slope AD, Amplifier and Band Gap							
V_{RFGO}	Reference Generator Output	—	@3.3V	1.45	1.5	1.55	V
V_{RFGTC}	Reference Generator Temperature Coefficient	—	@3.3V	—	50	—	Ppm/C
V_{ICMR}	Common Mode Input Range	—	Amplifier, no load	0.2	—	V_{REGO}–1	V
		—	Integrator, no load	1	—	V_{REGO}–0.2	V
V_{ADOFF}	Input Offset Range	—	—	—	500	800	μV

A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS}	System Clock (RC OSC)	—	2.2V~5.5V	400	—	4000	kHz
	System Clock (Crystal OSC)	—	2.2V~5.5V	400	—	4000	kHz
		—	3.3V~5.5V	400	—	8000	kHz
f_{INRC}	Internal RC OSC	3V	—	—	12	—	kHz
		5V		—	15	—	kHz
f_{TIMER}	Timer I/P Frequency (TMR0/TMR1)	—	2.2V~5.5V	0	—	4000	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	—	Power-up or 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_{SYS}

Functional Description

Execution Flow

The system clock is derived from either a crystal or an external RC 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 one cycle. If an instruction changes the value of the program counter, two cycles are required to complete the instruction.

Program Counter – PC

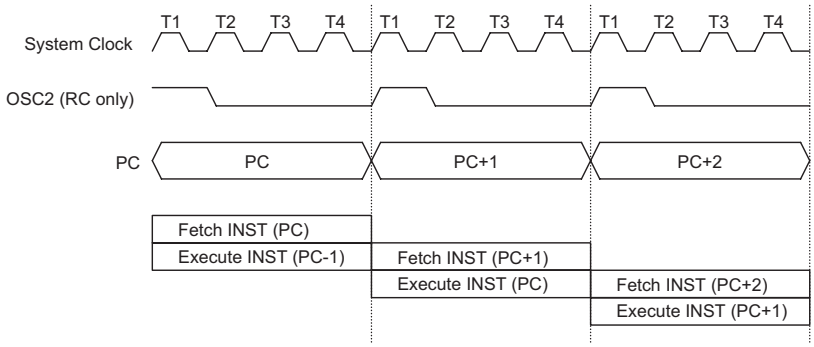
The program counter is 13 bits wide and controls the sequence in which the instructions stored in the program ROM are executed. The contents of the PC can specify a maximum of 8192 addresses.

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, conditional skip execution, loading the PCL register, a subroutine call, an initial reset, an internal interrupt, an external interrupt, or returning from a subroutine, the PC manipulates 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 with the next instruction.

The lower byte of the Program Counter, PCL, is a readable and writeable register. Moving data into the PCL register performs a short jump. The destination must be within 256 locations.



Execution Flow

Mode	Program Counter												
	*12	*11	*10	*9	*8	*7	*6	*5	*4	*3	*2	*1	*0
Initial Reset	0	0	0	0	0	0	0	0	0	0	0	0	0
External Interrupt	0	0	0	0	0	0	0	0	0	0	1	0	0
UART Interrupt	0	0	0	0	0	0	0	0	0	1	0	0	0
Timer/Event Counter 0 Overflow	0	0	0	0	0	0	0	0	0	1	1	0	0
Timer/Event Counter 1 Overflow	0	0	0	0	0	0	0	0	1	0	0	0	0
ADC Interrupt	0	0	0	0	0	0	0	0	1	0	1	0	0
RTC Interrupt	0	0	0	0	0	0	0	0	1	1	0	0	0
Skip	Program Counter+2												
Loading PCL	*12	*11	*10	*9	*8	@7	@6	@5	@4	@3	@2	@1	@0
Jump, Call Branch	#12	#11	#10	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
Return from Subroutine	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

Program Counter

Note: *12~*0: Program counter bits
#12~#0: Instruction code bits
S12~S0: Stack register bits
@7~@0: PCL bits

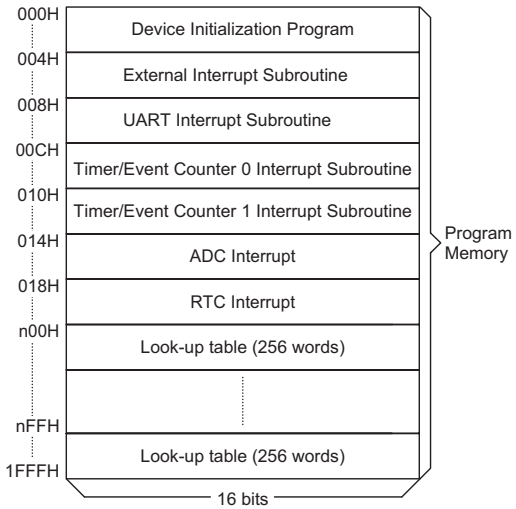
When a control transfer takes place, an additional dummy cycle is required.

Program Memory

The program memory is used to store the program instructions which are to be executed. It also contains data, table, and interrupt entries, and is organized with a structure of 8192×16 bits which are addressed by the program counter and table pointer.

Certain locations in the Program Memory are reserved for special usage:

- Location 000H
  Location 000H is reserved for program initialization. After a chip reset, the program always begins execution at this location.
- Location 004H
  Location 004H is reserved for the INT external interrupt service program. If the INT input pin is activated, and the interrupt is enabled, and the stack is not full, the program begins execution at location 004H.
- Location 008H
  This location is reserved for the UART interrupt service program. If the UART interrupt resultis from a UART TX or RX, and the interrupt is enabled and the stack is not full, the program begins execution at this location.



Note: n ranges from 0 to 1F

Program Memory

- 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 begins execution at location 00CH.
- 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 begins execution at location 010H.
- Location 014H
  Location 014H is reserved for the ADC interrupt service program. If an ADC interrupt occurs, and the interrupt is enabled, and the stack is not full, the program begins execution at location 014H.
- Location 018H
  Location 018H is reserved for the real time clock interrupt service program. If a real time clock interrupt occurs, and the interrupt is enabled, and the stack is not full, the program begins execution at location 018H.
- Table location
  Any location in the Program Memory can be used as a look-up table. The instructions "TABRDC [m]" (the current page, 1 page=256 words) and "TABRDL [m]" (the last page) transfer the contents of the lower-order byte to the specified data memory, and the contents of the higher-order byte to the TBLH register, which is the Table high order byte register. Only the destination of the lower-order byte in the table is well-defined; the other bits of the table word are all transferred to the lower portion of TBLH. The TBLH register is read only, and the table pointer, TBLP, is a read/write register, and is used to indicate the table location. Before accessing the table, the location should be placed into the TBLP register. All the table related instructions require 2 cycles to complete their operation. These areas may function as normal Program Memory depending upon the user's requirements.

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 organized into 16 levels and is neither part of the data nor part of the program, and is neither readable nor writeable. Its activated level is indexed by a stack pointer, SP, and is neither readable nor writeable. At the

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

Table Location

Note: *12~*0: Table location bits
P12~P8: Current program counter bits
@7~@0: Table pointer bits

start of a subroutine call or an interrupt acknowledgment, the contents of the program counter is 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 is restored to its previous value from the stack. After a chip reset, the SP will point to the top of the stack.

If the stack is full and a non-masked interrupt takes place, the interrupt request flag is recorded but the acknowledgment is still inhibited. Once the SP is decremented using a RET or RETI instruction, the interrupt is serviced. This feature prevents a 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 as only the most recent 16 return addresses are stored.

Data Memory

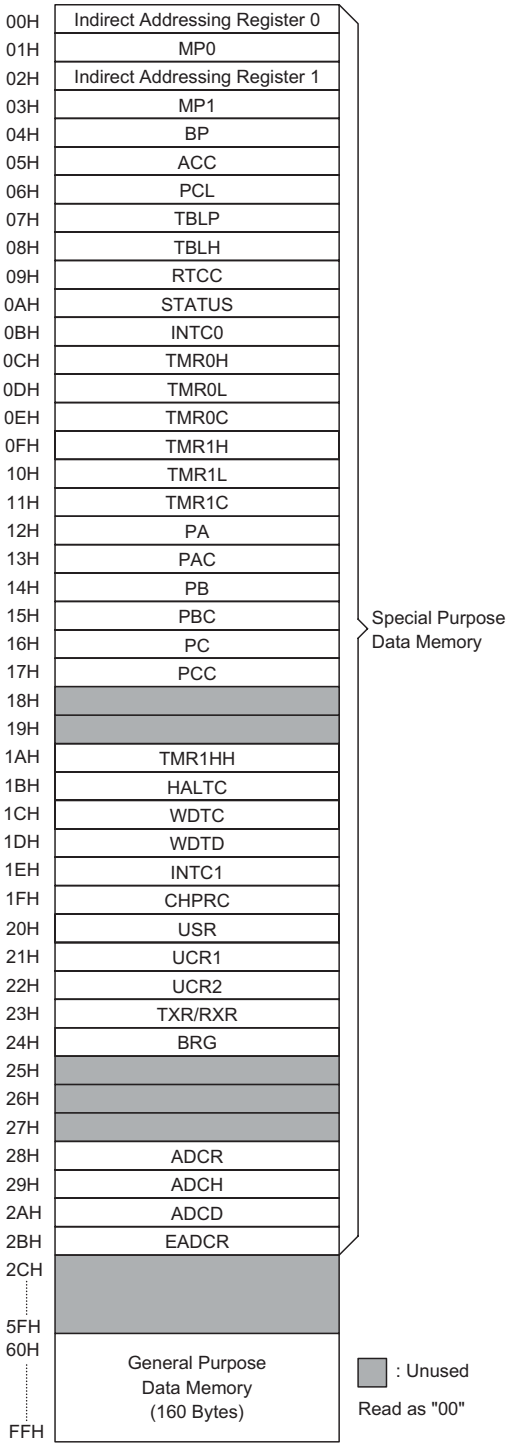
Bank 0 of the data memory has a capacity of 199×8 bits, and is divided into two functional groups, namely the special function registers, which have a 39×8 bit capacity and the general purpose data memory which have a 160×8 bit capacity. Most locations are readable/writable, although some are read only. The special function registers are overlapped in all banks.

Any unused locations before 60H will return a zero result if read. The general purpose data memory, addressed from 60H to FFH , is used for data and control information under instruction commands. All of the data memory areas can handle arithmetic, logic, increment, decrement and rotate operations directly. Except for some dedicated bits, each bit in the data memory can be set and reset by the "SET [m].i" and "CLR [m].i" instructions. They are also indirectly accessible through the memory pointer registers, MP0 and MP1.

Bank 1 contains the LCD Data Memory locations. After first setting up the Bank Pointer, BP, to the value of "01H" to access Bank 1, this bank must then be accessed indirectly using Memory Pointer MP1. With BP set to a value of "01H", using MP1 to indirectly read or write to the data memory areas with addresses from 40H~68H will result in operations to Bank 1. Directly addressing the Data Memory will always result in Bank 0 being accessed irrespective of the value of BP.

Indirect Addressing Register

Locations 00H and 02H are the indirect addressing registers, with the names of IAR0 and IAR2. These registers are not physically implemented, and any read/write operations to [00H] and [02H] accesses the Data Memory locations pointed to by MP0 and MP1 respectively. Reading locations 00H or 02H indirectly returns the result 00H. Writing to them indirectly leads to no operation. The function of data movement between two indirect addressing registers is not supported. The memory pointer registers, MP0 and MP1, are both 8-bit registers and are used to access the Data Memory in combination with the indirect addressing registers. MP0 can only be used to access the data memory, while MP1 can be used to access both the data memory and the LCD display memory.



RAM Mapping

Accumulator – ACC

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

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
- Logic operations - AND, OR, XOR, CPL
- Rotation - RL, RR, RLC, RRC
- Increment and Decrement - INC, DEC
- Branch decision - 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, the status register bits 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, a device power-up, 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 a subroutine call, the status register will not be automatically pushed onto the stack. If the contents of the status register is important, and if the subroutine is likely to corrupt the status register, the programmer should take precautions and save it properly.

Interrupts

The device provides one external interrupt, one UART interrupt, two internal timer/event counter interrupts and an ADC interrupt. The interrupt control register INTC0, and interrupt control register INTC1, both contain the interrupt control bits that are used to set the enable/disable status and record the interrupt request flags.

Once an interrupt subroutine is serviced, the other interrupts will be disabled, as the EMI bit will be automatically cleared, preventing 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 of INTC0 or INTC1 may be set in order to permit interrupt nesting to take place. Once the stack is full, the interrupt request will not be acknowledged, even if the related interrupt is enabled, until the Stack Pointer is decremented. If immediate service is desired, the stack should be prevented from becoming full.

All interrupts will 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 the specified location in the Program Memory. Only the contents of the program counter is pushed onto the stack. If the contents of the accumulator or of the status register is altered by the interrupt service program, this may corrupt the desired control sequence, therefore the contents should be saved in advance.

External interrupts are triggered by an edge transition on the INT pin. A configuration option determines the type of edge transition, high to low, low to high, or both

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

low to high and high to low. Its related interrupt request flag, EIF0; bit 4 of INTC0, must also be set. After the interrupt is enabled, if the stack is not full and the external interrupt is active, a subroutine call to location 04H occurs. The interrupt request flag, EIF0, and EMI bits will be cleared to disable other maskable interrupts.

The UART interrupt is initialised by setting the interrupt request flag, which is the URF; bit 5 in the INTC0 register. This is caused by a regular UART receive signal or a 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 08H occurs. The related interrupt request flag, URF, will be reset and the EMI bit will be cleared to disable other interrupts.

The internal Timer/Event Counter 0 interrupt is generated when the Timer/Event Counter 0 interrupt request flag is set, which is bit T0F; bit 6 of the INTC0 register. This occurs when the timer overflows. After the interrupt is enabled, if the stack is not full, and the T0F bit is set, a subroutine call to location 0CH occurs. The related interrupt request flag, T0F, will be reset, and the EMI bit will be cleared to disable other maskable interrupts. The interrupt for Timer/Event Counter 1 operates in a similar manner but its related interrupt request flag is T1F, which is bit 4 of INTC1, and its subroutine call location is 10H.

The A/D converter interrupt is generated when the A/D converter interrupt request flag, ADF; bit 5 of INTC1 is set. This occurs when an A/D conversion process has completed. After the interrupt is enabled, if the stack is not full, and the ADF bit is set, a subroutine call to location 14H occurs. The related interrupt request flag, ADF, is reset and the EMI bit is cleared to disable further maskable interrupts.

The real time clock interrupt is generated when the real time clock interrupt request flag, RTF; bit 6 of INTC1, is set. After the interrupt is enabled, if the stack is not full, and the RTF bit is set, a subroutine call to location 18H occurs. The related interrupt request flag, RTF, is reset and the EMI bit is cleared to disable further maskable interrupts.

During the execution of an interrupt subroutine, other maskable interrupt acknowledgments are all held until the ″RETI″ instruction is executed or the EMI bit and the related interrupt control bit are set both to 1 (if the stack is not full). To return from the interrupt subroutine, a ″RET″ or ″RETI″ instruction should be executed. A RETI instruction sets the EMI bit and enables an interrupt service, but a RET instruction does not.

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

Bit No.	Label	Function
0	EMI	Control the master (global) interrupt (1=enabled; 0=disabled)
1	EEI	Control the external interrupt (1=enabled; 0=disabled)
2	EURI	Controls the UART TX or RX interrupt (1/0: enable/disable)
3	ET0I	Control the Timer/Event Counter 0 interrupt (1=enabled; 0=disabled)
4	EIF0	External interrupt 0 request flag (1=active; 0=inactive)
5	URF	UART TX or RX interrupt request flag (1=active; 0=inactive)
6	T0F	Internal Timer/Event Counter 0 request flag (1=active; 0=inactive)
7	—	For test mode used only. Must be written as ″0″; otherwise may result in unpredictable operation.

INTC0 (0BH) Register

Bit No.	Label	Function
0	ET1I	Control the Timer/Event Counter 1 interrupt (1=enabled; 0=disabled)
1	EADI	Control the ADC interrupt (1=enabled; 0:disabled)
2	ERTI	Control the real time clock interrupt (1=enabled; 0:disabled)
3, 7	—	Unused bit, read as ″0″
4	T1F	Internal Timer/Event Counter 1 request flag (1=active; 0=inactive)
5	ADF	ADC request flag (1=active; 0=inactive)
6	RTF	Real time clock request flag (1=active; 0=inactive)

INTC1 (1EH) Register

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	1	04H
UART interrupt	2	08H
Timer/Event Counter 0 overflow	3	0CH
Timer/Event Counter 1 overflow	4	10H
ADC interrupt	5	14H
Real time clock interrupt	6	18H

Once an interrupt request flag has been set, it remains in the INTC1 or INTC0 register until the interrupt is serviced or cleared by a software instruction.

It is recommended that a program should not use the "CALL subroutine" within the interrupt subroutine. This is 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 interrupts is not well controlled, executing a "call" in the interrupt subroutine may damage the original control sequence.

Oscillator Configuration

The device provides three oscillator circuits for the system clock. These are an RC oscillator, a crystal oscillator and a 32768Hz crystal oscillator, the choice of which is determined by configuration options. The Power-down mode will stop the system oscillator, if the RC or crystal oscillator type has been selected, in order to conserve power. The 32768Hz crystal oscillator will continue running even when in the Power-down mode. If the 32768Hz crystal oscillator is selected as the system oscillator, the system oscillator is not stopped; but instruction execution is stopped. Since the 32768Hz oscillator is also designed for timing purposes, the internal timing (RTC, time base, WDT) operation keeps running 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 resistance should range from 30kΩ to 750kΩ. The system clock, divided by 4, is available on OSC2 with a pull-high resistor added, which can be used to synchronise external logic. The RC oscillator provides the most cost effective clock implementation, however, the frequency of the oscillation may vary with VDD, temperature and process variations. It is therefore, not suitable for timing sensitive operations where an accurate oscillator frequency is desired.

If the crystal oscillator is selected, a crystal across 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 obtain a frequency reference, but two external capacitors connected between OSC1, OSC2 and ground are required.

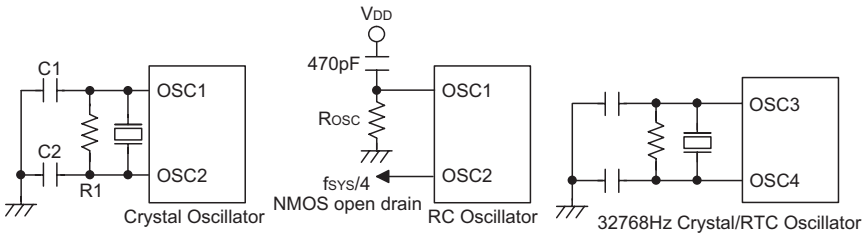
Another oscillator circuit is supplied for the real time clock. For this oscillator only a 32.768kHz crystal oscillator can be used, and should be connected between pins OSC3 and OSC4.

The RTC oscillator circuit can be made to start up quickly by setting the "QOSC" bit, which is bit 4 in the RTCC register. It is recommended to turn on the quick oscillating function when power is first applied, and then turn it off after 2 seconds.

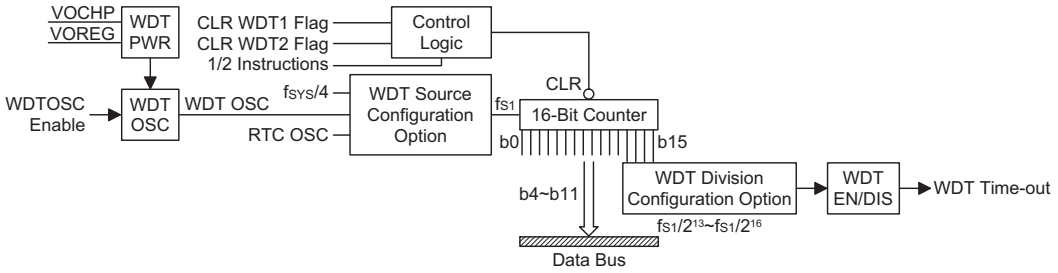
The WDT oscillator is a free running on-chip RC oscillator for which no external components are required. Although when the system enters the power down mode, the system clock stops, the WDT oscillator keeps running with a period of approximately 65μs at 5V. The WDT oscillator can be disabled by a configuration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

Watchdog Timer – WDT

The WDT clock is sourced from a dedicated RC oscillator (WDT oscillator) or instruction clock (system clock divided by 4) or a real time clock oscillator (RTC oscillator) decided by options. The WDT is provided to prevent software malfunctions or a sequence from jumping to an unknown location with unpredictable results. The watchdog timer can be disabled by a configuration option. If the watchdog timer is disabled, the WDT timer will have the same operation as if it were enabled except that the timeout signal will not generate a device reset. So when the watchdog timer is disabled, the WDT timer counter can still be read out and can still be cleared. This function is used to permit the application program to access the WDT frequency to obtain the temperature coefficient



System Oscillator



Watchdog Timer

for analog component adjustment. The WDT oscillator needs to be disabled/enabled using its registers, WDTC and WDTOSC, to minimise power consumption.

There are 2 registers related to the watchdog function, WDTC and WDTD. The WDTC register controls the WDT oscillator enable/disable function and the WDT power source. The WDTD register is the WDT counter readout register.

The WDTPWR bits can be used to choose the WDT power source; the default source is VOCHP. The main purpose of the regulator is for WDT Temperature-coefficient adjustment. In this case, the application program should enable the regulator before switching to the regulator source. The WDTOSC bits can be used to enable or disable the internal WDT OSC (12kHz). If the application does not use the WDT OSC, then it needs to disable it in order to reduce power consumption. When the WDTOSC is disabled, it is actually turned off.

If the internal RC oscillator, which has a nominal period of 65μs, is selected, it is first divided by a value which ranges from 2^{12}~2^{15} the exact value of which is determined by a configuration option, to obtain the actual WDT time-out period. The minimum period of the WDT time-out period is about 300ms~600ms. This time-out period may vary with temperature, VDD and process variations. By using the related WDT configuration option, longer time-out periods can be implemented. If the WDT time-out is selected to be 2^{15}, the maximum time-out period is divided by 2^{15}~2^{16}which will give a time-out period of about 2.3s~4.7s.

The WDT clock source may also come from the instruction clock, in which case the WDT will operate in the same manner except that in the Power Down mode the WDT will stop counting and lose its protecting function. If the device operates in a noisy environment, using the internal WDT oscillator is strongly recommended, since the HALT instruction will stop the system clock.

Under normal operation, a WDT overflow initialises a device reset and sets the status bit ″TO″. In the HALT or IDLE mode, the overflow initialises a ″warm reset″, and

Bit No.	Label	Function
0~1	WDTPWR0~ WDTPWR1	WDT Power source selection. 01: WDT power comes from VOCHP 10: WDT power comes from the regulator 00/11:Reserved
2~3	WDTOSC0~ WDTOSC1	WDT oscillator enable/disable (WDTOSC1:0)= 01: WDT OSC disable 10: WDT OSC enable 00/11: Reserved
4~7	—	Reserved

WDTC (1CH) Register

Note: The WDTOSC registers initial value will be set to enable (1,0), if both ″WDT option enable″ and ″WDT clock option are set to WDT″. Otherwise, it will be set to disable (0,1)

Bit No.	Label	Function
0~7	WDTD0~ WDTD7	The WDT counter data value. Read only - used for temperature adjustment.

WDTD (1DH) Register

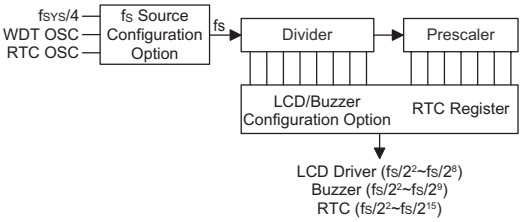
The WDT clock (f_{S1}) is further divided by an internal counter to give longer watchdog time-outs. The division ratio can be varied by selecting different configuration options to give a 2^{13} to 2^{16} division ration range.

only the PC and SP are reset to zero. There are three methods to clear the contents of the WDT, an external low level on $\overline{RES}$, a software instruction or a "HALT" instruction. There are two types of software instructions; the single "CLR WDT" instruction, or the pair of instructions - "CLR WDT1" and "CLR WDT2".

Of these two types of instruction, only one type of instruction can be active at a time depending on the configuration option - "CLR WDT" times selection option. If the "CLR WDT" is selected (i.e., CLR WDT times equal one), any execution of the "CLR WDT" instruction clears the WDT. If the "CLR WDT1" and "CLR WDT2" option is chosen (i.e., CLR WDT times equal two), these two instructions have to be executed to clear the WDT, otherwise the WDT may reset the device due to a time-out.

Multi-function Timer

The device provides a multi-function timer for the RTC, LCD and buzzer functions 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, which ranges 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 using configuration options. It is recommended to select a frequency as close as possible to 4kHz signal for the LCD driver circuits to ensure good display clarity.



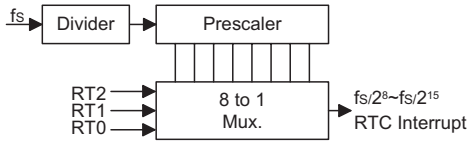
For the Charge Pump and the ADC chopper , the clock is independent of the multi-function timer. The clock is always sourced from the system clock, which is either an RC or Crystal clock.

Real Time Clock – RTC

The real time clock, 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 value being chosen by software programming. Writing data to RT2, RT1 and RT0, which are bits 2, 1, 0 of the RTCC register, provides various time-out periods. If an RTC time-out occurs, the related interrupt request flag, RTF; bit 6 of INTC1, is set. But if the interrupt is enabled, and if the stack is not full, a subroutine call to location 18H occurs.

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 for use



Real Time Clock

Buzzer Output

The Buzzer function provides a means of producing a variable frequency output, suitable for applications such as Piezo-buzzer driving or other external circuits that require a precise frequency generator. The BZ and $\overline{BZ}$ pins form a complimentary pair, and are pin-shared with I/O pins, PA0 and PA1. A configuration option is used to select from one of three buzzer options. The first option is for both pins PA0 and PA1 to be used as normal I/Os, the second option is for both pins to be configured as BZ and $\overline{BZ}$ buzzer pins, the third option selects only the PA0 pin to be used as a BZ buzzer pin with the PA1 pin retaining its normal I/O pin function. Note that the $\overline{BZ}$ pin is the inverse of the BZ pin which together form a differential output pair which can supply increased power to connected interfaces such as buzzers.

The buzzer functions is driven by the internal clock source, f_S, which then passes through a divider, the division ratio of which is selected by configuration options to provide a range of buzzer frequencies from $f_S/2^2$ to $f_S/2^9$.

The clock source that generates f_S, which in turn controls the buzzer frequency, can originate from two different sources, the Int.RCOSC (Internal RC oscillator) or the System oscillator/4, the choice of which is determined by the f_S clock source configuration option. Note that the buzzer frequency is controlled by configuration options, which select both the source clock for the internal clock f_S and the internal division ratio. There are no internal registers associated with the buzzer frequency.

If the configuration options have selected both pins PA0 and PA1 to function as a BZ and $\overline{BZ}$ complementary pair of buzzer outputs, then for correct buzzer operation it is essential that both pins must be setup as outputs by set-

ting bits PAC0 and PAC1 of the PAC port control register to zero. The PA0 data bit in the PA data register must also be set high to enable the buzzer outputs, if set low, both pins PA0 and PA1 will remain low. In this way the single bit PA0 of the PA register can be used as an on/off control for both the BZ and $\overline{BZ}$ buzzer pin outputs. Note that the PA1 data bit in the PA register has no control over the $\overline{BZ}$ buzzer pin PA1.

If the configuration options have selected that only the PA0 pin is to function as a BZ buzzer pin, then the PA1 pin can be used as a normal I/O pin. For the PA0 pin to function as a BZ buzzer pin, PA0 must be setup as an output by setting bit PAC0 of the PAC port control register to zero. The PA0 data bit in the PA data register must also be set high to enable the buzzer output, if set low pin PA0 will remain low. In this way the PA0 bit can be used as an on/off control for the BZ buzzer pin PA0. If the PAC0 bit of the PAC port control register is set high, then pin PA0 can still be used as an input even though the configuration option has configured it as a BZ buzzer output.

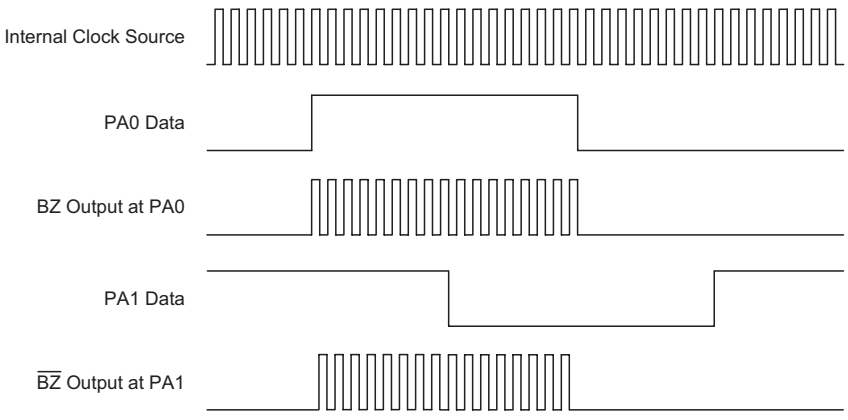
Note that no matter what configuration option is chosen for the buzzer, if the port control register has setup the pin to function as an input, then this will override the configuration option selection and force the pin to always behave as an input pin. This arrangement enables the pin to be used as both a buzzer pin and as an input pin, so regardless of the configuration option chosen; the actual function of the pin can be changed dynamically by the application program by programming the appropriate port control register bit.

The timing diagram shows the situation where both pins PA0 and PA1 are selected by a configuration option to be BZ and $\overline{BZ}$ buzzer pin outputs. The Port Control Register of both pins must have already been setup as outputs. The data setup on pin PA1 has no effect on the buzzer outputs.

PAC Register PAC.0	PAC Register PAC.1	PA data Register PA.0	PA data Register PA.1	Output Function
0	0	0	X	PA0=0, PA1=0
0	0	1	X	PA0=BZ, PA1=$\overline{BZ}$
0	1	0	X	PA0=0, PA1=Input
0	1	1	X	PA0=BZ, PA1=Input
1	0	0	X	PA0=Input, PA1=0
1	1	X	X	PA0=Input, PA1=Input

PA0/PA1 Pin Control Function

Note: "X" stands for don't care



Buzzer Output Pin Control

Power Down Operation – HALT

The Power-down mode is initialised by a ″HALT″ instruction and results in the following.

- The system oscillator turns off but the WDT oscillator keeps running if the WDT oscillator or the real time clock is selected.
- The contents of the Data Memory and the registers remain unchanged.
- The WDT is cleared and starts recounting if the WDT clock 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 leaves the Power-down mode by means of an external reset, an interrupt, an external falling edge signal on port A, or by 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 the device 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. The TO flag is set if a WDT time-out occurs, and causes a wake-up that only resets the program counter and the SP, and leaves the others in their original state.

A port A wake-up and interrupt methods can be considered as a continuation of normal execution. Each pin of port A can be independently selected to wake-up the device using configuration options. After awakening from an I/O port stimulus, the program will resume execution at the next instruction. However, if 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 will resume execution at the next instruction. But if the interrupt is enabled, and the stack is not full, a 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 events occur, 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 minimise power consumption, all the I/O pins should be carefully managed before entering the Power-down mode.

When a HALT instruction is executed, the CPU will stop running, and the related OSC and peripheral clocks will be set by the HALTC register. The HALTC register will only take effect when the system clock (f_{SYS}) is set to OSC.

Note: HALTC has no effect if the 32K oscillator is chosen as the system clock.

Bit No.	Label	Function
0	LCDON	Specifies the LCD condition in the Power-down mode 1: LCD module remains on (if OSCON=1) and ignores the configuration option setting 0: LCD condition decided by the LCD_ON configuration option
1	UARTON	Defines the UART state when in the Power-down mode 1: UART module remains on (if OSCON=1) 0: UART off
2~6	—	Unused bit, read as ″0″
7	OSCON	System clock oscillator On/off setting when in the Power-down mode 0: Oscillator stops running. All related peripherals will lose their clock and stop functioning. (Register bit 0 will be ignored) 1: Oscillator keeps running. (All peripheral keep running, except for the special setting of Bit 0)

HALTC (17H) Register

Reset

There are three ways in which a reset may occur.

- $\overline{RES}$ is reset during a normal operation
- $\overline{RES}$ is reset during Power-down
- WDT time-out is reset during normal operation

The WDT time-out during when in the Power-down mode differs from other reset conditions, as it performs 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 any other reset conditions. Most registers are reset to their initial conditions once the reset conditions are met. By examining the PDF and TO flags, the program can distinguish between different reset types.

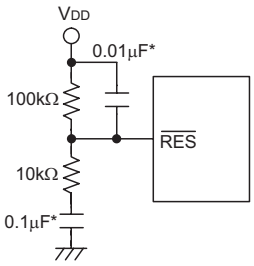
TO	PDF	RESET Conditions
0	0	$\overline{RES}$ reset during power-up
u	u	$\overline{RES}$ reset during normal operation
0	1	$\overline{RES}$ Wake-up HALT
1	u	WDT time-out during normal operation
1	1	WDT Wake-up HALT

Note: "u" stands for unchanged

To guarantee that the system oscillator has started and has 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 or during power-up. When awakening from the Power-down mode or during a system power-up, the SST delay is added.

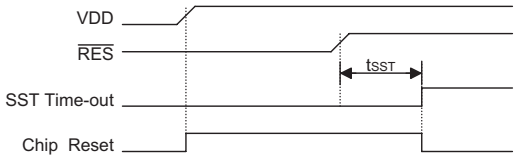
The functional unit chip reset status is shown below.

Program Counter	000H
Interrupt	Disabled
Prescaler, Divider	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

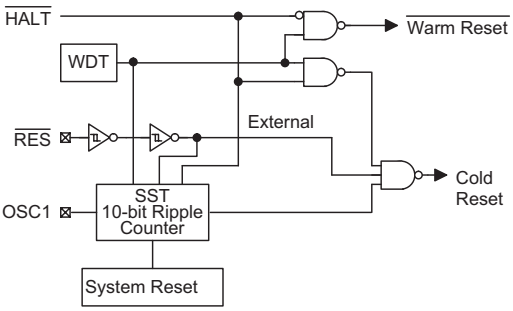


Reset Circuit

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

The register states are summarised below:

Register	Reset (Power On)	WDT Time-out (Normal Operation)	$\overline{RES}$ Reset (Normal Operation)	$\overline{RES}$ 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	---- ---0	---- ---0	---- ---0	---- ---0	---- ---u
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	--00 0111	--00 0111	--00 0111	--00 0111	--uu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC0	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
TMR0H	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR0L	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR0C	00-0 1000	00-0 1000	00-0 1000	00-0 1000	uu-u uuuu
TMR1H	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR1L	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR1C	0000 1----	0000 1----	0000 1----	0000 1----	uuuu u---
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PC	---- --11	---- --11	---- --11	---- --11	---- --uu
PCC	---- --11	---- --11	---- --11	---- --11	---- --uu
TMR1HH	---- --xx	---- --uu	---- --uu	---- --uu	---- --uu
HALTC	0--- --00	0--- --00	0--- --00	0--- --00	u--- --uu
WDTC	---- ss01	---- ss01	---- ss01	---- ss01	---- uuuu
WDTD	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000
INTC1	-000 -000	-000 -000	-000 -000	-000 -000	-uuu -uuu
CHPRC	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
USR	0000 1011	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
UCR1	0000 0x00	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
UCR2	0000 0000	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TXR/RXR	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
BRG	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
ADCR	-000 x000	-000 x000	-000 x000	-000 x000	-000 x000
ADCH	-000 --00	-000 --00	-000 --00	-000 --00	-000 --00
ADCD	---- -111	---- -111	---- -111	---- -111	---- -uuu
EADCR	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu

Note: "*" stands for warm reset

"u" stands for unchanged

"x" stands for unknown

"s" for special case, it depends on the option table (please see the WDT chapter for the detail)

Timer/Event Counter

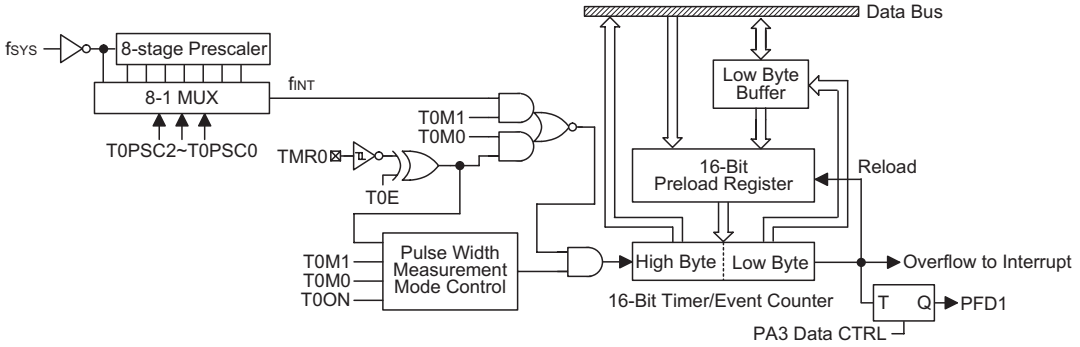
Two timer/event counters are integrated within the microcontroller. Timer/Event Counter 0 contains a 16-bit programmable count-up counter whose clock may come from an external source or an internal clock source. The internal clock source comes from f_{SYS}. Timer/Event Counter 1 contains an 18-bit programmable count-up counter whose clock may come from an external source or an internal clock source. The internal clock source comes from $f_{SYS}/4$ or 32768Hz selected by configuration option. The external clock input allows external events to be counted, time intervals or pulse widths to be measured, or to generate an accurate time base.

There are three registers related to Timer/Event Counter 0; TMR0H,TMR0L and TMR0C. Writing to TMR0L will only write the data into an internal lower-order byte 8-bit buffer while writing to TMR0H will transfer the specified data and the contents of the lower-order byte buffer into the TMR0H and TMR0L registers, respectively. The Timer/Event Counter 0 preload register is changed with each TMR0H write operation. Reading TMR0H will latch the contents of TMR0H and TMR0L counters to the destination and the lower-order byte buffer, respectively. Reading the TMR0L will only read the contents of the lower-order byte buffer. The TMR0C register is the Timer/Event Counter 0 control register, which defines the operating mode, counting enable or disable and the active edge.

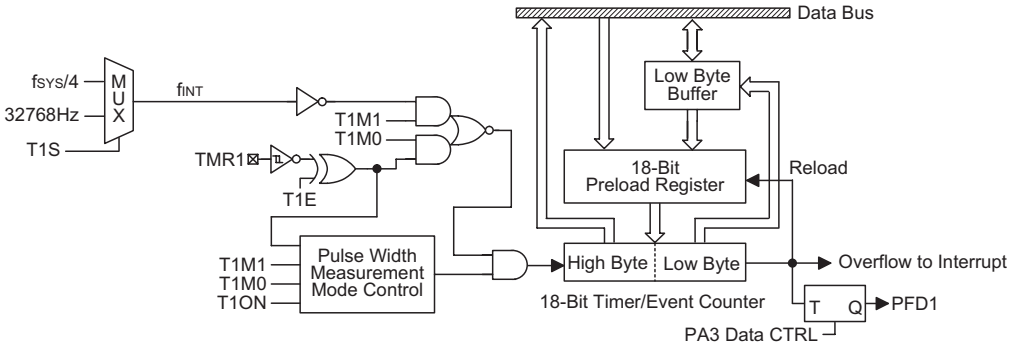
There are four registers related to Timer/Event Counter 1, TMR1HH, TMR1H, TMR1L and TMR1C. Writing to TMR1L and TMR1H will only put the required data into two internal lower-order byte buffers, each of which is 8-bits wide. Writing to TMR1HH will transfer the specified data and the contents of the lower-order byte buffers into the TMR1HH, TMR1H and TMR1L registers respectively. The Timer/Event Counter 1 preload register is changed with each write operation to the TRM1HH register. Reading TMR1HH will latch the contents of TMR1HH to the destination and latch the TMR1H and TMR1L counters to the lower-order byte buffers, respectively. Reading the TMR1H and TMR1L registers will read the contents of the lower-order byte buffers. TMR1C is the Timer/ Event Counter 1 control register, which defines the operating mode, counting enable or disable and the active edge.

The T0M0, T0M1 (TMR0C) and T1M0, T1M1 (TMR1C) bits define the operation mode. The event count mode is used to count external events, which means that the clock source comes from an external pin, TMR0 or TMR1. The timer mode functions as a normal timer with the clock source coming from the internally selected clock source. Finally, the pulse width measurement mode can be used to measure a high or low level duration of an external signal on pin TMR0 or TMR1. This measurement uses the internally selected clock source.

To enable a counting operation, the Timer ON bit - T0ON: bit 4 of TMR0C; T1ON: 4 bit of TMR1C - should



Timer/Event Counter 0



Timer/Event Counter 1

Bit No.	Label	Function
0 1 2	T0PSC0 T0PSC1 T0PSC2	To define the prescaler stages. T0PSC2, T0PSC1, T0PSC0= 000: $f_{INT}=f_{SYS}$ 001: $f_{INT}=f_{SYS}/2$ 010: $f_{INT}=f_{SYS}/4$ 011: $f_{INT}=f_{SYS}/8$ 100: $f_{INT}=f_{SYS}/16$ 101: $f_{INT}=f_{SYS}/32$ 110: $f_{INT}=f_{SYS}/64$ 111: $f_{INT}=f_{SYS}/128$
3	T0E	Defines the timer/event counter TMR0 pin active edge type: In the Event Counter Mode - T0M1,T0M0 = 0,1: 1:count on falling edge; 0:count on rising edge In the Pulse Width measurement mode - T0M1,T0M0 = 1,1 1: start counting on rising edge, stop on falling edge; 0: start counting on falling edge, stop on rising edge
4	T0ON	Enable/disable timer counting - 0=disabled; 1=enabled
5	—	Unused bit, read as ″0″
6 7	T0M0 T0M1	Operation Mode Definition bits 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	T132KON	Defines if the 32768 Oscillator is running or not - see note 0: 32768 Oscillator turned off if not being used by other peripherals 1: 32768 Oscillator starts running or keeps running.
1~2	—	Unused bit, read as ″0″
3	T1E	Defines the timer/event counter TMR1 active edge: In the Event Counter Mode - T1M1,T1M0= 0,1: 1:count on falling edge; 0:count on rising edge In the Pulse Width measurement mode - T1M1,T1M0 = 1,1: 1: start counting on rising edge, stop on falling edge; 0: start counting on falling edge, stop on rising edge
4	T1ON	Enable/disable timer counting - 0= disabled; 1= enabled
5	T1S	Defines the TMR1 internal clock source - 0=$f_{SYS}/4$; 1=32768Hz
6 7	T1M0 T1M1	Operation Mode Definition bits T0M1, T0M0: 01= Event count mode - External clock 10= Timer mode - Internal clock 11= Pulse Width measurement mode - External clock 00= Unused

TMR1C (11H) Register

Note: The 32768Hz oscillator enable will be a logical OR function of the T132KON bit and any configuration option that chooses the 32768Hz oscillator. That is, the 32768Hz OSC will be enabled if any related function enables it, and will be turned off if no function enables it.

be set to 1. In the pulse width measurement mode, the T0ON/T1ON bit is automatically cleared after the measurement cycle is completed. But in the other two modes, the T0ON/T1ON bits can only be reset using instructions. The Timer/Event Counter overflow is one of the wake-up sources. The timers can also be used as the source clock for the PFD - Programmable Frequency Divider - output on PA3. This function is selected by a configuration option. Only one Timer/Event Counter clock source (PFD0 or PFD1) can be used as the PFD clock source, chosen by a configuration option. If PA3 is selected to be a PFD output, there are two types of selections. One is to use PFD0 as the PFD output, the other is to use PFD1 as the PFD output. PFD0 and PFD1 are the timer overflow signals of the Timer/Event Counter 0 and Timer/Event Counter 1 respectively. No matter what the operation mode is, writing a 0 to ET0I or ET1I disables the related interrupt service. When the PFD function is selected, executing a ″SET [PA].3″ instruction will enable the PFD output while executing a ″CLR [PA].3″ instruction will disable the PFD output.

In the case of a 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 running, data written to the timer/event counter will be loaded only to the timer/event counter preload register. The timer/event counter still continues its operation until an overflow occurs.

When the timer/event counter is read, the clock will be blocked to avoid errors, which may result in a counting error, and should be taken into account by the programmer. It is recommended to load a desired value into the TMR0/TMR1 registers first, before turning on the related timer/event counter, since the initial value of the TMR0/TMR1 registers are unknown. After this procedure, the timer/event function can be operated normally.

Bit0~bit2 of TMR0C can be used to define the pre-scaling stages of the timer/event counter internal clock. The overflow signal of timer/event counter can be used to generate the PFD signal.

Input/Output Ports

There are 18 bi-directional input/output lines in the mirocontroller, labeled as PA, PB and PC. All of these I/O ports can be used for input and output operations. For input operation, these ports are non-latching, that is, the inputs must be ready at the T2 rising edge of instruction ″MOV A,[m]″. For output operation, all the data is latched and remains unchanged until the output latch is rewritten.

Each I/O line has its own control register, known as PAC, PBC and PCC, to control the input/output configuration. With this control register, either CMOS outputs or Schmitt trigger inputs with or without pull-high resistor structures can be reconfigured dynamically under software control. To function as an input, the corresponding latch of the control register must be written with a ″1″. The input source also depends on the control register. If the control register bit is ″1″, the input will read the pad state. To function as an output the the control register bit should be set to ″0″. The latter is possible in the ″read-modify-write″ instruction.

After a device reset, these input/output lines will default to inputs and remain at a high level or in a floating state, depending upon the pull-high configuration options. Each bit of these input/output latches can be set or cleared by a ″SET [m].i″ and ″CLR [m].i″ 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 the accumulator.

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

Each I/O port has a pull-high option. Once a pull-high option is selected, the I/O port has a pull-high resistor connected. Take note that a non-pull-high I/O port setup as an input mode will be in a floating condition.

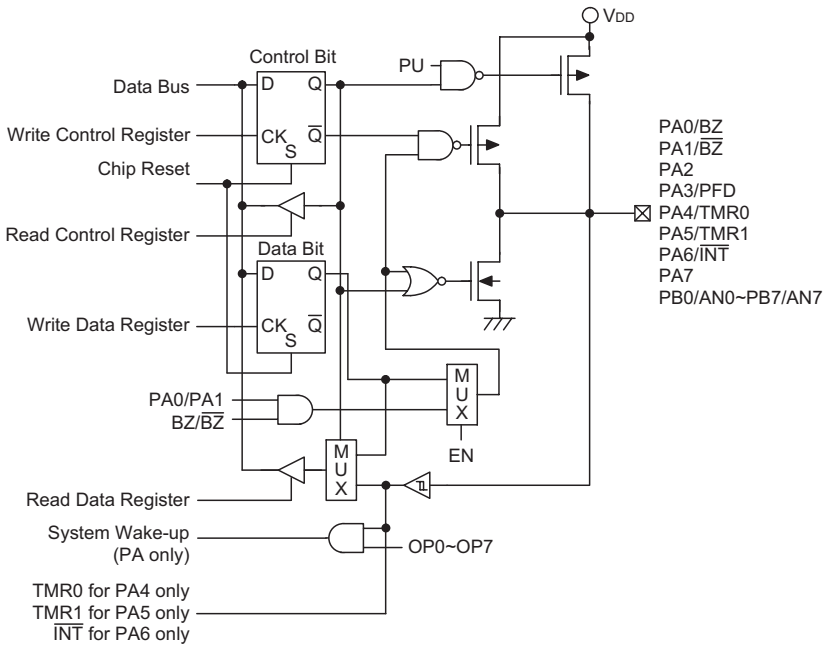
Pins PA0, PA1, PA3, PA4, PA5 and PA6 are pin-shared with BZ, $\overline{BZ}$, PFD, TMR0, TMR1 and INT pins respectively.

PA0 and PA1 are pin-shared with the BZ and $\overline{BZ}$ signals, respectively. If the BZ/$\overline{BZ}$ configuration option is selected, then if PA0/PA1 are setup as outputs, the buzzer signal will appear on these pins. The input mode always retains its original function. Once the BZ/$\overline{BZ}$ configuration option is selected, the buzzer output signals are controlled by the PA0 data register.

The PA0/PA1 I/O function is shown below.

PA0 I/O	I	I	O	O	O	O	O	O	O	O
PA1 I/O	I	O	I	I	I	O	O	O	O	O
PA0 Mode	X	X	C	B	B	C	B	B	B	B
PA1 Mode	X	C	X	X	X	C	C	C	B	B
PA0 Data	X	X	D	0	1	D_0	0	1	0	1
PA1 Data	X	D	X	X	X	D1	D	D	X	X
PA0 Pad Status	I	I	D	0	B	D_0	0	B	0	B
PA1 Pad Status	I	D	I	I	I	D_1	D	D	0	B

Note: ″I″ input; ″O″ output
″D, D0, D1″ Data
″B″ buzzer option, BZ or $\overline{BZ}$
″X″ don't care
″C″ CMOS output



Input/Output Ports

The PA3 pin is pin-shared with the PFD signal. If the PFD option is selected, the output signal in the output mode of PA3 will be the PFD signal generated by the timer/event counter overflow signal. The input mode always retains its original functions. Once the PFD option is selected, the PFD output signal is controlled by the PA3 data register only. Writing a "1" to PA3 data register will enable the PFD output function and writing a "0" will force the PA3 pin to remain at "0". The I/O functions of PA3 are shown below.

I/O Mode	I/P (Normal)	O/P (Normal)	I/P (PFD)	O/P (PFD)
PA3	Logical Input	Logical Output	Logical Input	PFD (Timer on)

Note: The PFD frequency is the timer/event counter overflow frequency divided by 2.

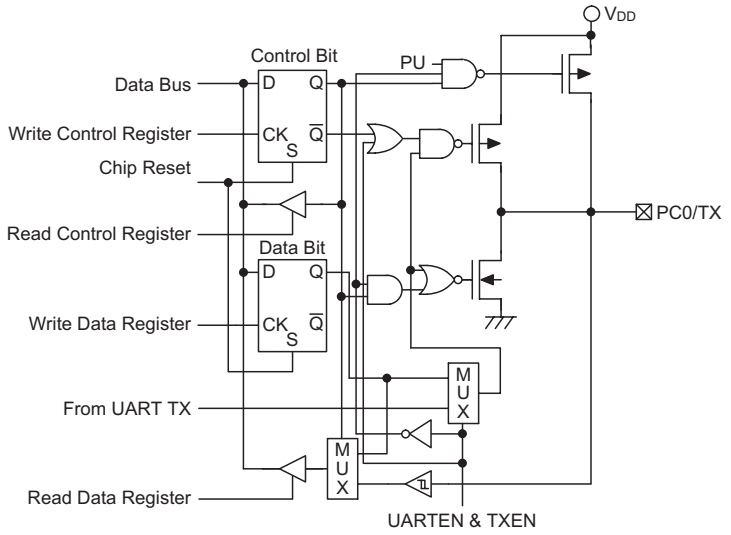
The PA0, PA1, PA3, PD4, PD5, PD6 and PD7 pins are pin-shared with BZ, BZ, PFD, INT0, INT1, TMR0 and TMR1 pins, respectively.

The descriptions of the PFD control signal and PFD output frequency are listed in the following table.

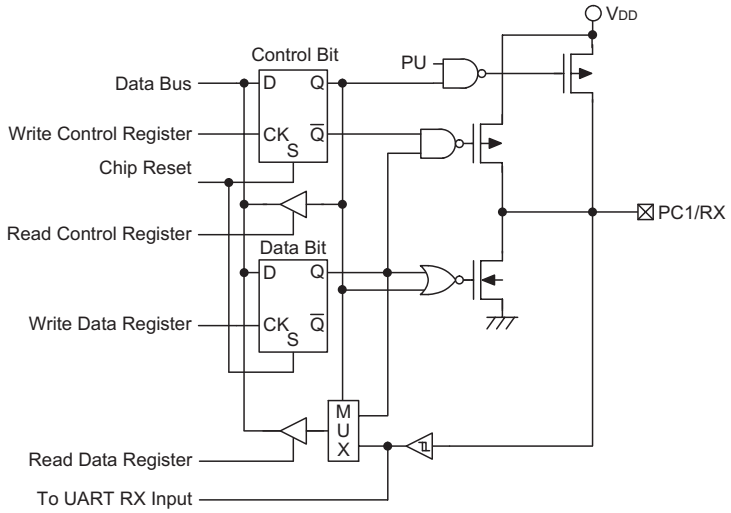
Timer	Timer Preload Value	PA3 Data Register	PA3 Pad State	PFD Frequency
OFF	X	0	0	X
OFF	X	1	U	X
ON	N	0	0	X
ON	N	1	PFD	$f_{TMR}/[2\times(M-N)]$

Note: "X" stands for unused
"U" stands for unknown
"M" is "65536" for PFD0 or PFD1
"N" is the preload value for the timer/event counter
"f_{TMR}" is input clock frequency for the timer/event counter

It is recommended that if there are unused lines then they should be setup as output pins using software instructions to avoid consuming power. If setup as inputs and left floating this may result in unnecessary increased power consumption.



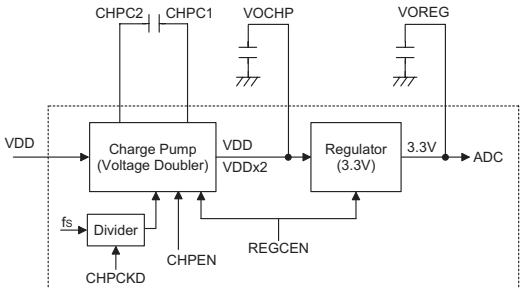
PC0/TX Block Diagram



PC1/RX Block Diagram

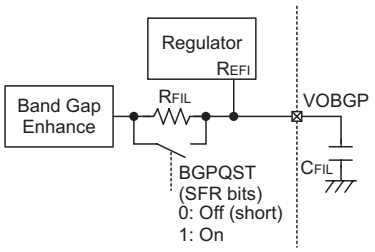
Charge Pump and Voltage Regulator

A charge pump and voltage regulator are integrated within the device. The charge pump can be enabled/disabled by the application program. The charge pump uses VDD as its input, and has the function of doubling the V_{DD} voltage, therefore the charge pump output voltage will be VDD×2. The regulator can generate a stable voltage of 3.3V for the ADC and also can provide an external bridge sensor excitation voltage or supply a reference voltage for other applications. The user needs to ensure that the charge pump output voltage is greater than 3.6V to ensure that the regulator is capable of generating the required 3.3V voltage output.



Additionally, the device also includes a band gap voltage generator for the 1.5V low temperature sensitive reference voltage. This reference voltage is used as the zero adjustment and for a single end type reference voltage.

R_{FIL} has a value of around 100kΩ and the recommend value for C_{FIL} is 10μF.



Note: The VOBGP signal is an internal signal only to which only the recommend C_{FIL}value should be connected.

There is a single register associated with this module named CHPRC. The CHPRC register is the Charge Pump/Regulator Control register, which controls the charge pump on/off function, the regulator on/off functions as well as setting the clock divider value to generate the clock for the charge pump.

The CHPCKD4~CHPCKD0 bits are used to set the clock divider to generate the desired clock frequency for proper charge pump operation. The actual frequency is determined by the following formula.

Actual Charge Pump Clock= (f_{SYS}/16)/(CHPCKD +1).

The suggested charge pump clock frequency is 20kHz. The application needs to set the correct value to get the desired clock frequency. For a 4MHz application, the CHPCKD bits should be set to the decimal value 11, and for a 2MHz application, the bits should be set to 5.

The REGCEN bit in the CHPRC register is the Regulator/Charge-pump module enable/disable control bit. If

Bit No.	Label	Function
0	REGCEN	Enable/disable Regulator/Charge-Pump module. (1=enable; 0=disable)
1	CHPEN	Charge Pump Enable/disable setting. (1=enable; 0=disable) Note: this bit will be ignored if the REGCEN bit is disabled
2	BGPQST	Bandgap quick start-up function 0: R short, quick start up 1: R connected, normal RC filter mode Each time REGCEN changes from 0 to 1, that is when the regulator turns on, this bit should be set to 0 and then set to 1 to ensure a quick start up. The minimum time to keep the bit low should be about 2ms.
3~7	CHPCKD0~ CHPCKD4	Charge pump clock divider. These 5 bits form a clock divider with a division ratio range of 1 to 32. Charge Pump clock = (f_{SYS}/16) / (CHPCKD+1)

CHPRC (1FH) Register

REGCEN	CHPEN	Charge Pump	VOCHP Pin	Regulator	VOREG Pin	OPA ADC	Description
0	X	OFF	V_{DD}	OFF	Hi-Impedance	Disable	Complete module is disabled, OPA/ADC will have no Power
1	0	OFF	V_{DD}	ON	3.3V	Active	Used when V_{DD} is greater than 3.6V
1	1	ON	2×V_{DD}	ON	3.3V	Active	Use whefor V_{DD} is less than 3.6V (V_{DD}=2.2V~3.6V)

this bit is disabled, then the regulator will be disabled and the charge pump will be also be disabled to save power. When REGCEN = 0, the module will enter the Power Down Mode ignoring the CHPEN setting. The ADC and OPA will also be disabled to reduce power.

If REGCEN is set to ″1″, the regulator will be enabled. If the CHPEN bit is enabled, the charge pump will be active and will use VDD as its input to generate the double voltage output. This double voltage will then be used as the input voltage for the regulator. If CHPEN is set to ″0″, the charge pump is disabled and the charge pump output will be equal to the charge pump input, VDD.

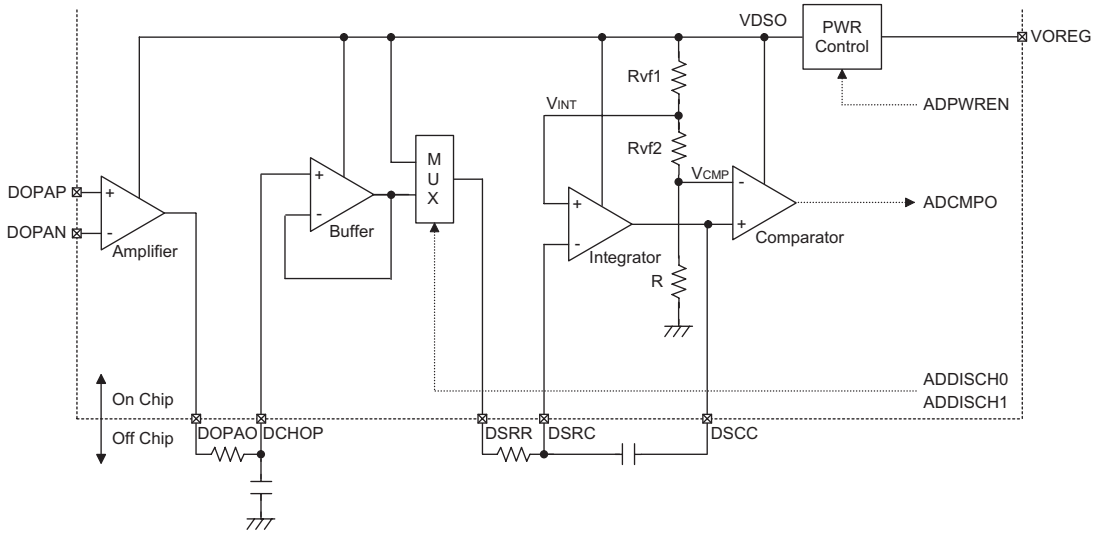
It is necessary to carefully manage the V_{DD} voltage. If the voltage is less than 3.6V, then CHPEN should be set to 1 to enable the charge pump, otherwise CHPEN should be set to zero. If the Charge pump is disabled and V_{DD} is less than 3.6V then the output voltage of the regulator will not be guaranteed.

ADC – Dual Slope

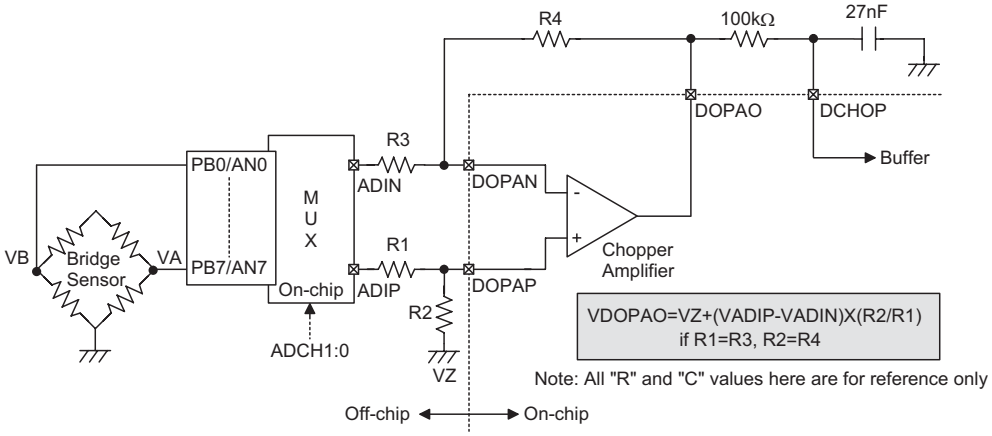
A Dual Slope A/D converter is implemented within the microcontroller. The dual slope module includes an Operational Amplifier and a buffer for the amplification of differential signals and an Integrator and comparator for the main dual slope AD converter.

There are 4 special function registers related to the ADC function known as ADCR, ADCD, ADCH and EADCR. The ADCR register is the A/D control register, which controls the ADC block power on/off, the chopper clock on/off, the charge/discharge control and is also used to read out the comparator output status. The ADCD register is the A/D Chopper clock divider register, which defines the chopper clock to the ADC module. The EADCR register is the enhanced A/D control register, which defines the Auto Mode Dual Slope ADC function. The ADCH A/D Channel selection register, which configure the PB port A/D function enable, and also the input channel selection for the ADC input.

The ADPWREN bit in the ADCR register, is used to control the ADC module on/off function. The ADCCKEN bit in the ADCR register is used to control the chopper clock on/off function. When the ADCCKEN bit is set to ″1″ it

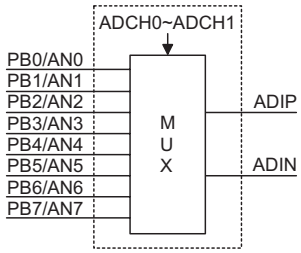


Note: The V_{INT}, V_{CMP} signals can come from different R groups which are selected by software registers.



will enable the Chopper clock, with the clock frequency defined by the ADCD registers. The ADC module includes the OPA, buffer, integrator and comparator, however the Bandgap voltage generator is independent of this module. It will be automatically enabled when the regulator is enabled, and also be disabled when the regulator is disabled. The application program should enable the related power to permit them to function and disable them when idle to conserve power. The charge/discharge control bits, ADDISCH1~ ADDISCH0, are used to control the Dual slope circuit charging and discharging behavior. The ADCMPO bit is read only for the comparator output, while the ADINTM bits can set the ADCMPO trigger mode for interrupt generation.

The PB port can be set as A/D converter inputs or as general purpose bi-directional lines. The PCR2:0 bits define the number of A/D input lines. The ADCH1:0 bits can define the ADIP and ADIN input combination.



Bit No.	Label	Function
0 1	ADCH0 ADCH1	OPA input channel selection register. This register can set the input signal combination. Refer to table below for details.
2~3, 7	—	Reserved
4~6	PCR0~PCR2	Port B: A/D channel (OPA input) / General I/O configuration

ADCH (29H) Register

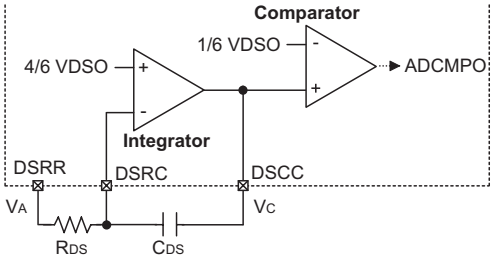
ADCH1~ADCH0	Input Channel (ADIP/ADIN)	Description
00	AN0/AN1	Differential
01	AN2/AN3	Differential
10	AN4/AN5	Differential
11	AN6/AN7	Differential

Table OPA Input Channel Setting

PCR2~PCR0	
000	Port B A/D channels — all off
001	Reserved
010	PB0~PB1 enabled as A/D channels
011	Reserved
100	PB0~PB3 enabled as A/D channels
101	Reserved
110	PB0~PB5 enabled as A/D channels
111	PB0~PB7 enabled as A/D channels

Table Port B Configuration

The following descriptions are based on the fact that ADRR0=0



The amplifier and buffer combination form a differential input pre-amplifier which amplifies the sensor input signal.

The combination of the Integrator, the comparator, the resistor Rds, between DSRR and DSRC and the capacitor Cds, between DSRC and DSCC form the main body of the Dual slope ADC.

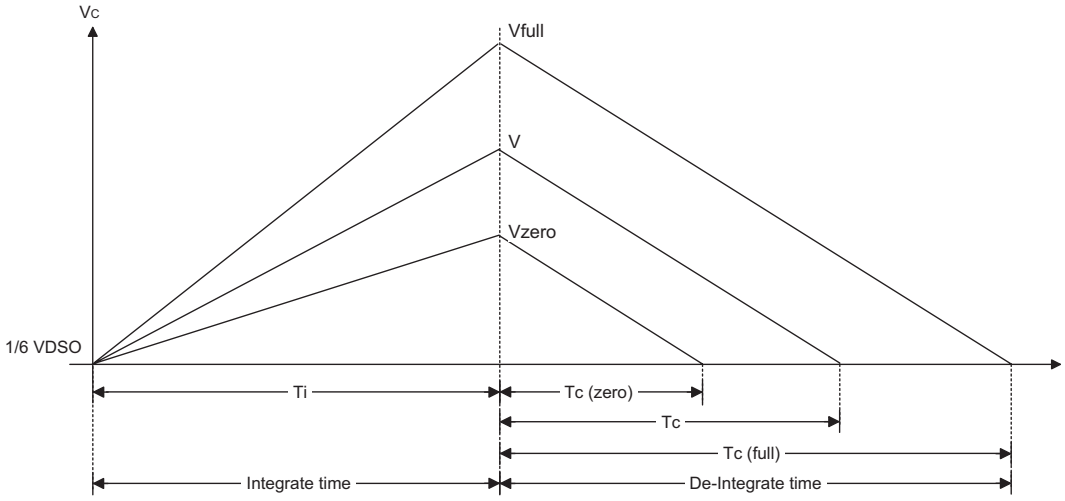
The Integrator integrates the output voltage increase or decrease and is controlled by the ″Switch Circuit″ - refer to the block diagram. The charge and discharge curves are illustrated by the following.

The ″comparator″ will switch the state from high to low when VC, which is the DSCC pin voltage,drops to less than 1/6 VDSO.

In general applications, the application program will switch the ADC to the charging mode for a fixed time called Ti, which is the integrating time. It will then switch to the discharging mode and wait for V_c to drop to less than 1/6VDSO. At this point the comparator will change state and store the time taken, Tc, which is the de-integrating time. The following formula 1 can then be used to calculate the input voltage V_A.

formula 1: $V_A = (1/3) \times VDSO \times (2 - Tc/Ti)$.
(Based on ADRR0=0)

In user applications, it is required to choose the correct value of R_{DS} and C_{DS} to determine the Ti value, to allow the V_C value to operate between 5/6VDSO and 1/6VDSO. Vfull cannot be greater than 5/6VDSO and Vzero cannot be less than 1/6VDSO



Bit No.	Label	Function
0	ADPWREN	Dual slope block - including input OP - power on/off switch. 0: Disable Power 1: Power source is sourced from the regulator.
1~2	ADDISCH0~ ADDISCH1	Defines the ADC discharge/charge. 00: reserved 01: charging - Integrator input is connect to the buffer output 10: discharging - Integrator input is connect to VDSO 11: reserved
3	ADCMPO	Dual Slope ADC - last stage comparator output. Read only bit, write data instructions will be ignored. During the discharging state, when the integrator output is less than the reference voltage, the ADCMPO bit will change from high to low.
4~5	ADINTM0~ ADINTM1	ADC integrator interrupt mode definition. These two bits define the ADCMPO data interrupt trigger mode: 00: no interrupt 01: rising edge 10: falling edge 11: both edge
6	ADCCKEN	ADC OP chopper clock source on/off switch. 0: disable 1: enable - clock value is defined by ADCD register
7	ADRR0	ADC resistor selection 0: (V_{INT}, V_{CMP})= (4/6 VOREG, 1/6 VOREG) 1: (V_{INT}, V_{CMP})= (4.4/6 VOREG, 1/6 VOREG)

ADCR (18H) Register

Bit No.	Label	Function
0 1 2	ADCD0 ADCD1 ADCD2	Defines the chopper clock. ADCCKEN should be enabled. The suggested clock value should be around 10kHz. The chopper clock definitions are: 0: clock= (f_{SYS}/32)/1 1: clock= (f_{SYS}/32)/2 2: clock= (f_{SYS}/32)/4 3: clock= (f_{SYS}/32)/8 4: clock= (f_{SYS}/32)/16 5: clock= (f_{SYS}/32)/32 6: clock= (f_{SYS}/32)/64 7: clock= (f_{SYS}/32)/128
3~7	—	Reserved

ADCD (1AH) Register

Bit No.	Label	Function
0~1	—	Unused bit, read as ″0″
2	CHGTS	Select the ADC charge timer 0: Timer/Event Counter 0 1: Timer/Event Counter 1
3	DISTS	Select the ADC discharge timer 0: Timer/Event Counter 0 1: Timer/Event Counter 1
4	CHGCMP	Select if the charge timer (note 1) will auto-start by the ADCMPO result or not. ASTEN should be enabled. 0: immediately auto start timer counting (note 3) when charging begins 1: Wait for the ADCMPO rising edge to auto start timer counting - see note - when charging begins
5	ASTEN	Dual slope auto start enable. When this function is enabled, the charging timer (note 1) will auto start (note 3) when the user sets ADISCH to the charging mode. The start method is determined by CHGCMP. 0: disable 1: enable this auto function
6	ADISEN	Dual slope auto discharge enable. When this function enabled and ADISCH is set to the charging mode and the charging timer (note 1)) run overflow, the charging timer will auto stop (note 3) and ADISCH will be auto set to the discharging mode (note 4) and the discharging timer (note 2) will auto start counting (note 3). 0: disable 1: enable this auto function
7	AENDEN	Dual slope Auto End Enable. When this function is enabled and ADISCH is set to the discharge mode and detects the ADCMPO falling edge, the discharging timer will auto stop (note 2). 0: disable 1: enable this auto function

Note: 1: Charge timer means the Timer/Event Counter 0 or Timer/Event Counter 1 that is selected by the CHGTS bit.

2: Discharge timer means the Timer/Event Counter 0 or Timer/Event Counter 1 that is selected by the DISTS bit.

3: Timer auto start will set the T0ON or T1ON bits to 1 and auto stop will set the bits to 0

4: Will auto set the discharge mode by setting the ADDISCH1/0 bits to 1/0.

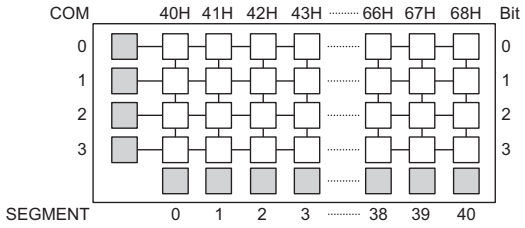
EADCR (1BH) Register

LCD Display Memory

The device provides an area of embedded data memory for the LCD display. This area is located at 40H to 68H in Bank 1 of the Data Memory. The bank pointer BP, enables either the General Purpose Data Memory or LCD Memory to be chosen. When BP is set to ″1″, any data written into location range 40H~68H will affect the LCD display. When the BP is cleared to ″0″, any data written into 40H~68H will access the general purpose data memory. The LCD display memory can be read and written to only indirectly using MP1. When data is written into the display data area, it is automatically read by the LCD driver which then generates the corresponding LCD driving signals. To turn the display on or off, a ″1″ or a ″0″ is written to the corresponding bit of the display memory, respectively. The figure illustrates the mapping between the display memory and LCD pattern for the device.

The LCD clock frequency is determined by configuration options, and has a division ratio range of $fs/2^2$~$fs/2^8$. The LCD clock source frequency should be chosen to be as close as possible to 4kHz.

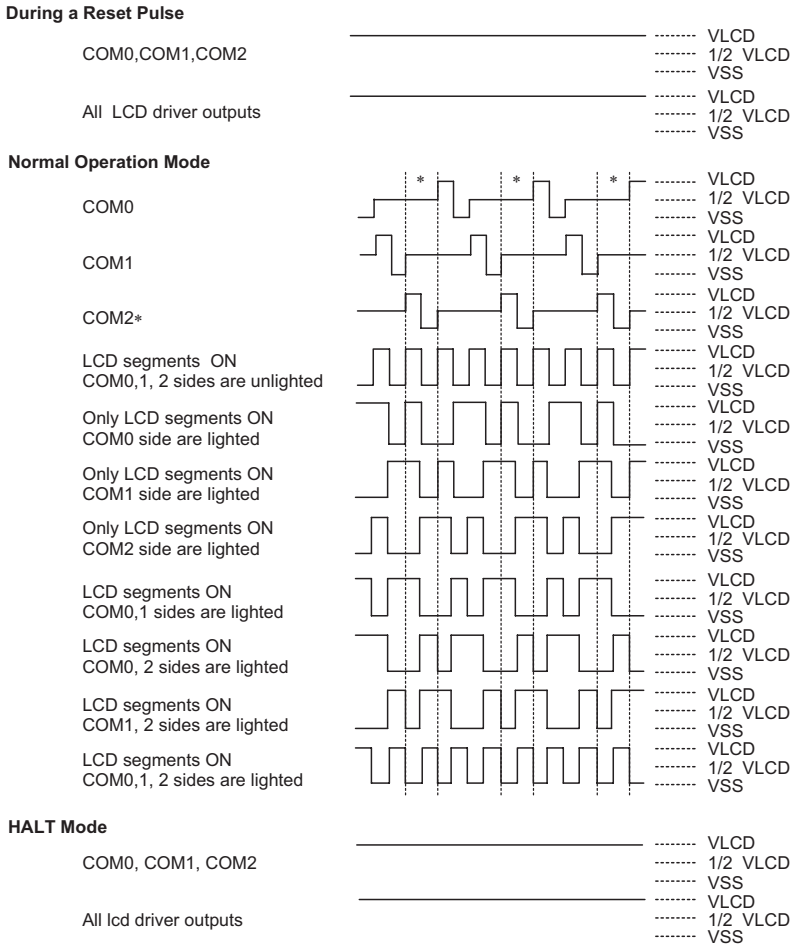
Note that the LCD frequency is controlled by configuration options, which select the internal division ratio.



Display Memory

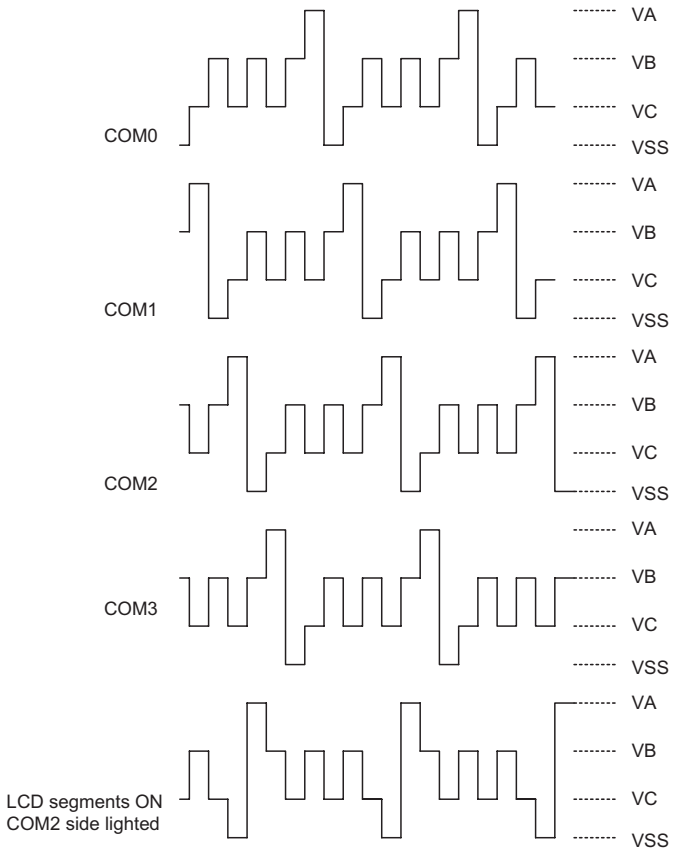
LCD Driver Output

The output number of the device LCD driver can be 41×2, 41×3 or 40×4 by configuration option (i.e., 1/2, 1/3 or 1/4 duty). The bias type LCD driver can be ″R″ type or ″C″ type. If the ″R″ bias type is selected, no external capacitor is required. If the ″C″ bias type is selected, a capacitor mounted between C1 and C2 pins is needed. The LCD driver bias voltage can be 1/2 bias or 1/3 bias by option. If 1/2 bias is selected, a capacitor mounted between V2 pin and ground is required. If 1/3 bias is selected, two capacitors are needed for V1 and V2 pins. Refer to application diagram.



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

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



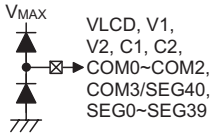
Note: 1/4 duty, 1/3 bias, R type: "VA" VLCD, "VB" 2/3 VLCD, "VC" 1/3 VLCD

LCD Driver Output (1/4 Duty)

LCD Type	R Type		C Type	
LCD Bias Type	1/2 bias	1/3 bias	1/2 bias	1/3 bias
V_{MAX}	If $V_{DD}>V_{LCD}$, then V_{MAX} connect to V_{DD}, else V_{MAX} connect to V_{LCD}		If $V_{DD}>\frac{3}{2}V_{LCD}$, then V_{MAX} connect to V_{DD}, else V_{MAX} connect to V1	

LCD Segment pins used as Logic Outputs

The SEG0~SEG23 pins also can be setup for use logic outputs using configuration options. Once an LCD segment is selected for use as a logic output, the content of bit 0 of the related segment address in the LCD RAM will appear on the segment. SEG0~SEG7 are together byte optioned as logical outputs, SEG8~SEG15 are together byte optioned as logical outputs, and SEG16~SEG23 are bits that can be individually optioned as logical outputs.



Low Voltage Reset/Detector Functions

There is a low voltage detector, LVD, and a low voltage reset circuit, LVR, implemented within the microcontroller. These two functions can be enabled/disabled via configuration options. Once the LVD option is enabled, the user can use the RTCC.3 bit to enable/disable the LVD circuit and read the LVD detector status from the RTCC.5 bit. Otherwise the LVD function is disabled.

When the LVD function configuration option is set to the enable state, the LVD voltage option will decide the detecting voltage. When the LVD configuration option is set to LVR+0.2, the actual LVD voltage depends upon the LVR options and will detect the VDD voltage. When LVD option is set to Regulator+ 0.2, the actual LVD voltage will detect the regulator input voltage.

The RTCC register definitions are listed below.

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

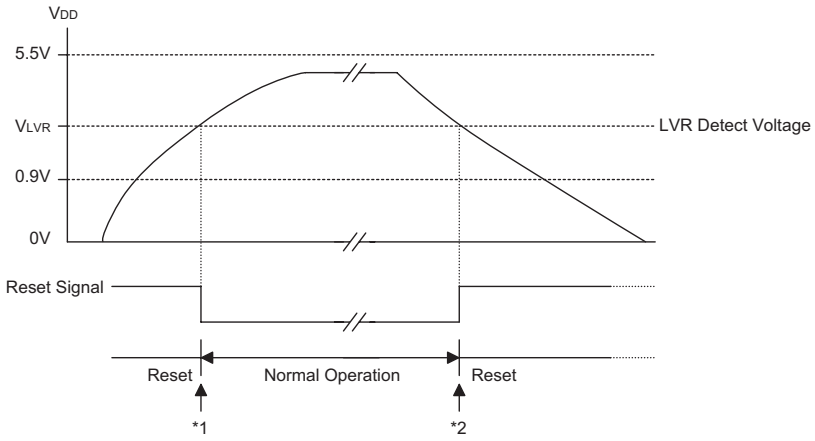
RTCC (09H) Register

The LVR has the same effect or function as the external RES signal which performs a device reset. When in the Power Down Mode, both the LVR and LVD are disabled.

The microcontroller provides a low voltage reset circuit in order to monitor the supply voltage of the device. If the supply voltage of the device is within the range 0.9V~V_{LVR}, such as what might happen when changing a battery, the LVR will automatically reset the device internally.

The LVR includes the following specifications:

- The low voltage, which is specified as 0.9V~V_{LVR}, has to remain within this range for a period of time greater than 1ms. If the low voltage state does not exceed 1ms, the LVR will ignore it will not perform a reset function.
- The LVR has an ″OR″ function with the external $\overline{RES}$ signal to perform a device reset.



Low Voltage Reset

Note: *1: To make sure that the system oscillator has stabilised, the SST provides an extra delay of 1024 system clock pulses before entering normal operation.

*2: Since a low voltage state has to be maintained in its original state for over 1ms, therefore after the 1ms delay, the device enters the reset mode.

UART Bus Serial Interface

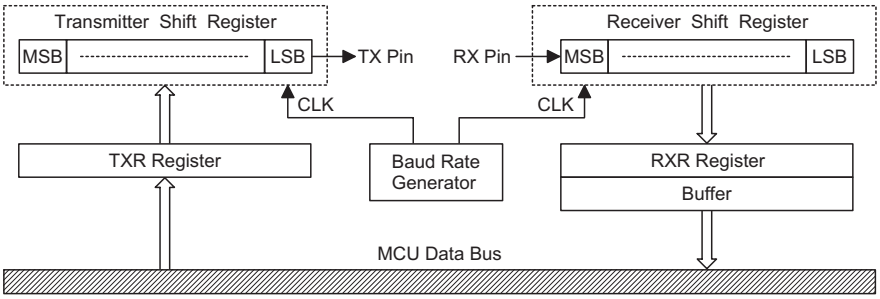
The HT46RU75D-1 device contain an integrated full-duplex asynchronous serial communications UART interface that enables communication with external devices that contain a serial interface. The UART function has many features and can transmit and receive data serially by transferring a frame of data with 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

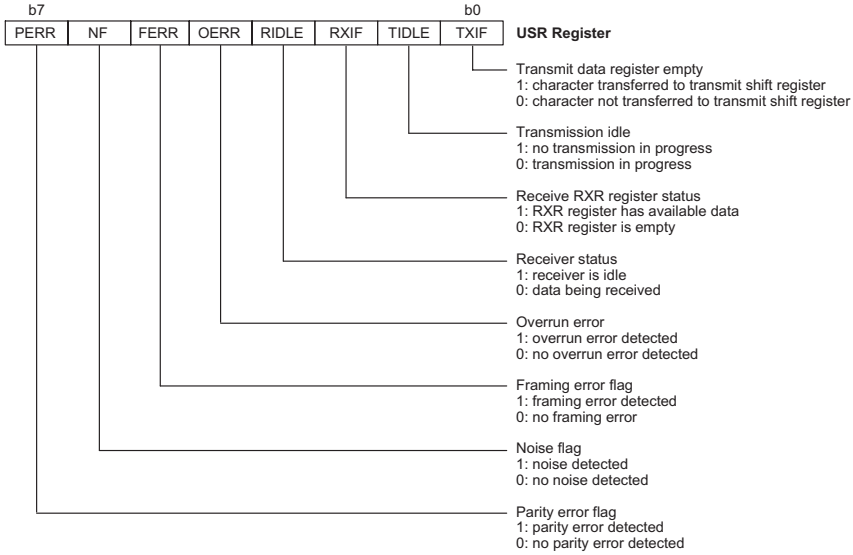
    The OERR flag is the overrun error flag, which indicates when the receiver buffer has overflowed. When this read only flag is ″0″ there is no overrun error. When the flag is ″1″ an overrun error occurs which will inhibit further transfers to the RXR receive data register. The flag is cleared by a software 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

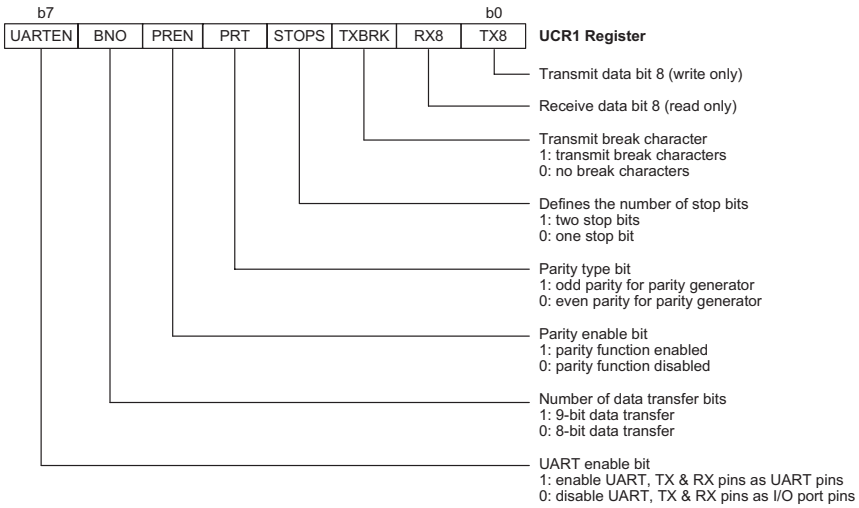
    This is parity enable bit. When this bit is equal to ″1″ the parity function will be enabled, if the bit is equal to ″0″ then the parity function will be disabled.

  - 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

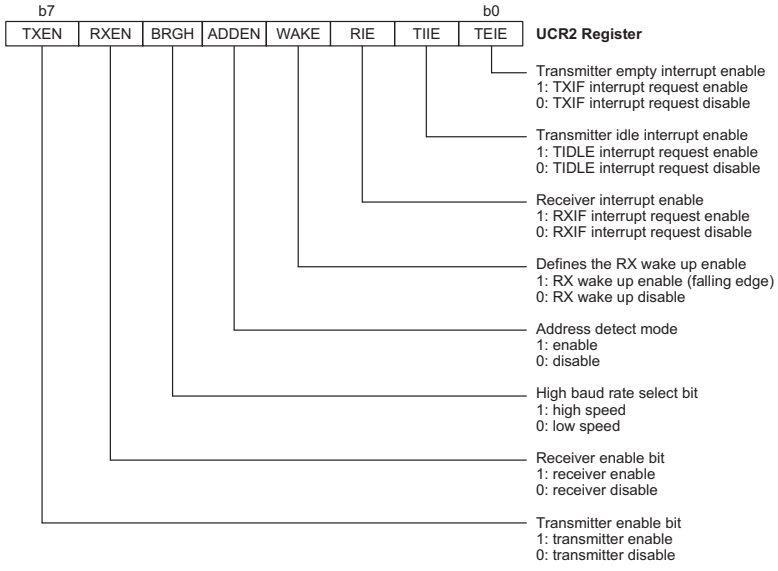
    The ADDEN bit is the address detect mode bit. When this bit is ″1″ the address detect mode is enabled. When this occurs, if the 8th bit, which corresponds to RX7 if BNO=0, or the 9th bit, which corresponds to RX8 if BNO=1, has a value of ″1″ then the received word will be identified as an address, rather 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:

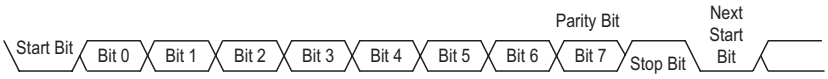
  1. A USR register access
  2. A TXR register write execution

  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.

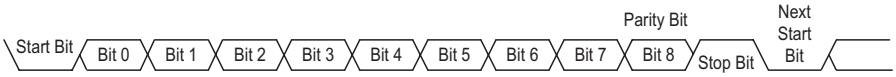
  During a data transmission, a write instruction to the TXR register will place the data into the TXR register, which will be copied to the shift register at the end of the present transmission. When there is no data transmission in progress, a write instruction to the TXR register will place the data 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

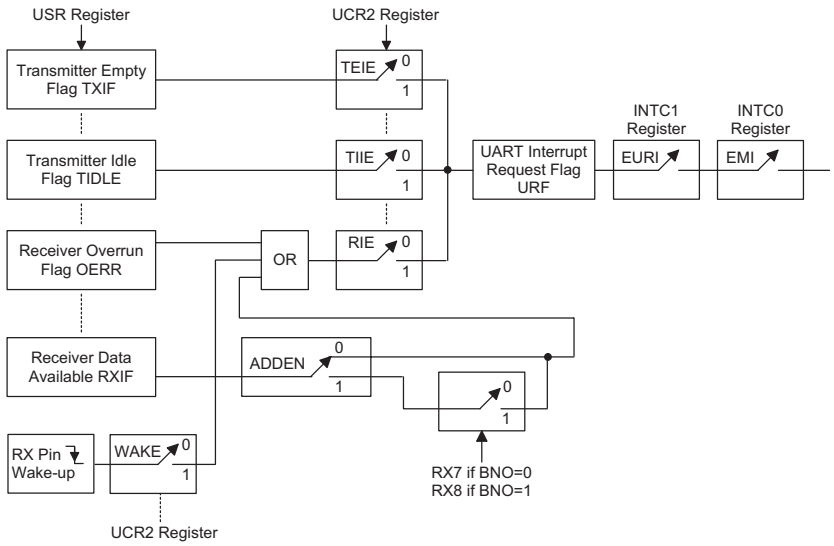
    The read only framing error flag, FERR, in the USR register, is set if a zero is detected instead of stop bits. If two stop bits are selected, both stop bits must be high, otherwise the FERR flag will be set. The FERR flag is buffered along with the received data and is cleared on any reset.

  - 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.

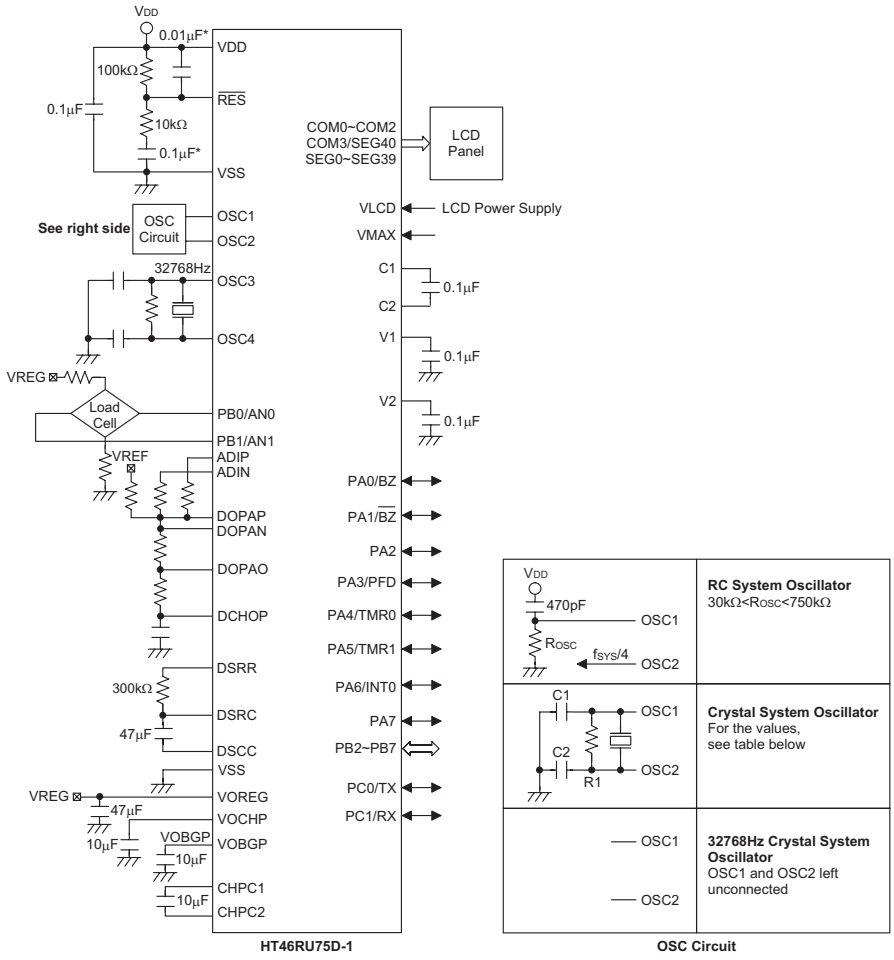
  For a UART wake-up interrupt to occur, in addition to the bits for the wake-up being set, the global interrupt enable bit, EMI, and the UART interrupt enable bit, EURI must also be set. If these two bits are not set then only a wake up event will occur and no interrupt will be generated. Note also that 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.

Options

The following shows the options in the device. All options must be defined for proper device operation.

Options
OSC type selection. There are two types: Crystal OSC or RC OSC
System clock selection: OSC or RTC
f_S clock source. There are two types: $f_{SYS}/4$, WDT or RTC
WDT clock source selection. There are various types: system clock/4, WDT or RTC
WDT enable/disable selection. The WDT function can be enabled or disabled by this configuration option.
WDT time-out period selection. There are four types: WDT clock source divided by $2^{16}/f_S$, $2^{15}/f_S$, $2^{14}/f_S$ or $2^{13}/f_S$
CLR WDT times selection. This option selects the instruction method of clearing the WDT. ″One time″ means that the ″CLR WDT″ instruction can clear the WDT. ″Two times″ means only if both the ″CLR WDT1″ and ″CLR WDT2″ instructions have been executed, can the WDT be cleared.
Buzzer output frequency selection. There are eight types of frequency signals for the buzzer output: $f_S/2^2$~$f_S/2^9$. ″f_S″
Wake-up selection. This option defines the wake-up capability. A falling edge on each external pin on PA has the capability to wake-up the device from a Power Down condition. Bit option.
Pull-high selection. Selects a pull-high resistor when the I/O pin has been setup as an input. Bit options.
I/O pins shared with other function selections. PA0/$\overline{BZ}$, PA1/BZ: PA0 and PA1 can be setup as I/O pins or buzzer outputs. PA3 can be setup as an I/O pin or as a PFD output.
PFD clock source selection: Timer/Event Counter 0 or Timer/Event Counter 1
LCD common selection. There are three types: 2 commons (1/2 duty), 3 commons (1/3 duty) or 4 commons (1/4 duty).
LCD bias selection. This option is to determine what kind of bias is selected, 1/2 bias or 1/3 bias.
LCD segment logic output Determines if pins SEG16~SEG23 are setup as logic outputs or as LCD segment outputs (bit option). Also if SEG0~SEG7 and SEG8~SEG15 are also setup as logic outputs or as LCD segment outputs (byte options.
LCD driver clock frequency selection. There are a range of frequency signals for the LCD driver circuits: $f_S/2^2$~$f_S/2^8$
LCD ON/OFF when in Power Down Mode selection
LCD Bias type selections. This option determines the Bias type - R type or C type
LVR selection. LVR enable or disable option
LVD selection. LVD enable or disable option
LVR voltage selection: 2.1V, 3.15V or 4.2V
LVD voltage selection: LVR+0.2 or regulator+0.2
$\overline{INT}$ trigger edge selection: disable; high to low; low to high; low to high or high to low
Partial-lock selection: Page0~3, Page4~7, Page8~11,....Page24~27, Page28~30, Page31.

Application Circuits



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

Crystal or Resonator	C1, C2	R1
4MHz Crystal	0pF	10kΩ
4MHz Resonator	10pF	12kΩ
3.58MHz Crystal	0pF	10kΩ
3.58MHz Resonator	25pF	10kΩ
2MHz Crystal & Resonator	25pF	10kΩ
1MHz Crystal	35pF	27kΩ
480kHz Resonator	300pF	9.1kΩ
455kHz Resonator	300pF	10kΩ
429kHz Resonator	300pF	10kΩ
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 the 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 $\overline{RES}$ 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

Introduction

Central to the successful operation of any microcontroller is its instruction set, which is a set of program instruction codes that directs the microcontroller to perform certain operations. In the case of Holtek microcontrollers, a comprehensive and flexible set of over 60 instructions is provided to enable programmers to implement their application with the minimum of programming overheads.

For easier understanding of the various instruction codes, they have been subdivided into several functional groupings.

Instruction Timing

Most instructions are implemented within one instruction cycle. The exceptions to this are branch, call, or table read instructions where two instruction cycles are required. One instruction cycle is equal to 4 system clock cycles, therefore in the case of an 8MHz system oscillator, most instructions would be implemented within 0.5μs and branch or call instructions would be implemented within 1μs. Although instructions which require one more cycle to implement are generally limited to the JMP, CALL, RET, RETI and table read instructions, it is important to realize that any other instructions which involve manipulation of the Program Counter Low register or PCL will also take one more cycle to implement. As instructions which change the contents of the PCL will imply a direct jump to that new address, one more cycle will be required. Examples of such instructions would be ″CLR PCL″ or ″MOV PCL, A″. For the case of skip instructions, it must be noted that if the result of the comparison involves a skip operation then this will also take one more cycle, if no skip is involved then only one cycle is required.

Moving and Transferring Data

The transfer of data within the microcontroller program is one of the most frequently used operations. Making use of three kinds of MOV instructions, data can be transferred from registers to the Accumulator and vice-versa as well as being able to move specific immediate data directly into the Accumulator. One of the most important data transfer applications is to receive data from the input ports and transfer data to the output ports.

Arithmetic Operations

The ability to perform certain arithmetic operations and data manipulation is a necessary feature of most microcontroller applications. Within the Holtek microcontroller instruction set are a range of add and subtract instruction mnemonics to enable the necessary arithmetic to be carried out. Care must be taken to ensure correct handling of carry and borrow data when results exceed 255 for addition and less than 0 for subtraction. The increment and decrement instructions INC, INCA, DEC and DECA provide a simple means of increasing or decreasing by a value of one of the values in the destination specified.

Logical and Rotate Operations

The standard logical operations such as AND, OR, XOR and CPL all have their own instruction within the Holtek microcontroller instruction set. As with the case of most instructions involving data manipulation, data must pass through the Accumulator which may involve additional programming steps. In all logical data operations, the zero flag may be set if the result of the operation is zero. Another form of logical data manipulation comes from the rotate instructions such as RR, RL, RRC and RLC which provide a simple means of rotating one bit right or left. Different rotate instructions exist depending on program requirements. Rotate instructions are useful for serial port programming applications where data can be rotated from an internal register into the Carry bit from where it can be examined and the necessary serial bit set high or low. Another application where rotate data operations are used is to implement multiplication and division calculations.

Branches and Control Transfer

Program branching takes the form of either jumps to specified locations using the JMP instruction or to a subroutine using the CALL instruction. They differ in the sense that in the case of a subroutine call, the program must return to the instruction immediately when the subroutine has been carried out. This is done by placing a return instruction RET in the subroutine which will cause the program to jump back to the address right after the CALL instruction. In the case of a JMP instruction, the program simply jumps to the desired location. There is no requirement to jump back to the original jumping off point as in the case of the CALL instruction. One special and extremely useful set of branch instructions are the conditional branches. Here a decision is first made regarding the condition of a certain data memory or individual bits. Depending upon the conditions, the program will continue with the next instruction or skip over it and jump to the following instruction. These instructions are the key to decision making and branching within the program perhaps determined by the condition of certain input switches or by the condition of internal data bits.

Bit Operations

The ability to provide single bit operations on Data Memory is an extremely flexible feature of all Holtek microcontrollers. This feature is especially useful for output port bit programming where individual bits or port pins can be directly set high or low using either the "SET [m].i" or "CLR [m].i" instructions respectively. The feature removes the need for programmers to first read the 8-bit output port, manipulate the input data to ensure that other bits are not changed and then output the port with the correct new data. This read-modify-write process is taken care of automatically when these bit operation instructions are used.

Table Read Operations

Data storage is normally implemented by using registers. However, when working with large amounts of fixed data, the volume involved often makes it inconvenient to store the fixed data in the Data Memory. To overcome this problem, Holtek microcontrollers allow an area of Program Memory to be setup as a table where data can be directly stored. A set of easy to use instructions provides the means by which this fixed data can be referenced and retrieved from the Program Memory.

Other Operations

In addition to the above functional instructions, a range of other instructions also exist such as the "HALT" instruction for Power-down operations and instructions to control the operation of the Watchdog Timer for reliable program operations under extreme electric or electromagnetic environments. For their relevant operations, refer to the functional related sections.

Instruction Set Summary

The following table depicts a summary of the instruction set categorised according to function and can be consulted as a basic instruction reference using the following listed conventions.

Table conventions:

x: Bits immediate data

m: Data Memory address

A: Accumulator

i: 0~7 number of bits

addr: Program memory address

Mnemonic	Description	Cycles	Flag Affected
Arithmetic			
ADD A,[m]	Add Data Memory to ACC	1	Z, C, AC, OV
ADDM A,[m]	Add ACC to Data Memory	1Note	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	1Note	Z, C, AC, OV
SUB A,x	Subtract immediate data from the 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	1Note	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, result in Data Memory	1Note	Z, C, AC, OV
DAA [m]	Decimal adjust ACC for Addition with result in Data Memory	1Note	C
Logic Operation			
AND A,[m]	Logical AND Data Memory to ACC	1	Z
OR A,[m]	Logical OR Data Memory to ACC	1	Z
XOR A,[m]	Logical XOR Data Memory to ACC	1	Z
ANDM A,[m]	Logical AND ACC to Data Memory	1Note	Z
ORM A,[m]	Logical OR ACC to Data Memory	1Note	Z
XORM A,[m]	Logical XOR ACC to Data Memory	1Note	Z
AND A,x	Logical AND immediate Data to ACC	1	Z
OR A,x	Logical OR immediate Data to ACC	1	Z
XOR A,x	Logical XOR immediate Data to ACC	1	Z
CPL [m]	Complement Data Memory	1Note	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	1Note	Z
DECA [m]	Decrement Data Memory with result in ACC	1	Z
DEC [m]	Decrement Data Memory	1Note	Z

Mnemonic	Description	Cycles	Flag Affected
Rotate			
RRA [m]	Rotate Data Memory right with result in ACC	1	None
RR [m]	Rotate Data Memory right	1Note	None
RRCA [m]	Rotate Data Memory right through Carry with result in ACC	1	C
RRC [m]	Rotate Data Memory right through Carry	1Note	C
RLA [m]	Rotate Data Memory left with result in ACC	1	None
RL [m]	Rotate Data Memory left	1Note	None
RLCA [m]	Rotate Data Memory left through Carry with result in ACC	1	C
RLC [m]	Rotate Data Memory left through Carry	1Note	C
Data Move			
MOV A,[m]	Move Data Memory to ACC	1	None
MOV [m],A	Move ACC to Data Memory	1Note	None
MOV A,x	Move immediate data to ACC	1	None
Bit Operation			
CLR [m].i	Clear bit of Data Memory	1Note	None
SET [m].i	Set bit of Data Memory	1Note	None
Branch			
JMP addr	Jump unconditionally	2	None
SZ [m]	Skip if Data Memory is zero	1Note	None
SZA [m]	Skip if Data Memory is zero with data movement to ACC	1note	None
SZ [m].i	Skip if bit i of Data Memory is zero	1Note	None
SNZ [m].i	Skip if bit i of Data Memory is not zero	1Note	None
SIZ [m]	Skip if increment Data Memory is zero	1Note	None
SDZ [m]	Skip if decrement Data Memory is zero	1Note	None
SIZA [m]	Skip if increment Data Memory is zero with result in ACC	1Note	None
SDZA [m]	Skip if decrement Data Memory is zero with result in ACC	1Note	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 table (current page) to TBLH and Data Memory	2Note	None
TABRDL [m]	Read table (last page) to TBLH and Data Memory	2Note	None
Miscellaneous			
NOP	No operation	1	None
CLR [m]	Clear Data Memory	1Note	None
SET [m]	Set Data Memory	1Note	None
CLR WDT	Clear Watchdog Timer	1	TO, PDF
CLR WDT1	Pre-clear Watchdog Timer	1	TO, PDF
CLR WDT2	Pre-clear Watchdog Timer	1	TO, PDF
SWAP [m]	Swap nibbles of Data Memory	1Note	None
SWAPA [m]	Swap nibbles of Data Memory with result in ACC	1	None
HALT	Enter power down mode	1	TO, PDF

Note: 1. For skip instructions, if the result of the comparison involves a skip then two cycles are required, if no skip takes place only one cycle is required.
2. Any instruction which changes the contents of the PCL will also require 2 cycles for execution.
3. For the ″CLR WDT1″ and ″CLR WDT2″ instructions the TO and PDF flags may be affected by the execution status. The TO and PDF flags are cleared after both ″CLR WDT1″ and ″CLR WDT2″ instructions are consecutively executed. Otherwise the TO and PDF flags remain unchanged.

Instruction Definition

ADC A,[m] Add Data Memory to ACC with Carry

Description The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the Accumulator.

Operation ACC ← ACC + [m] + C

Affected flag(s) OV, Z, AC, C

ADCM A,[m] Add ACC to Data Memory with Carry

Description The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the specified Data Memory.

Operation [m] ← ACC + [m] + C

Affected flag(s) OV, Z, AC, C

ADD A,[m] Add Data Memory to ACC

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) OV, Z, AC, C

ADD A,x Add immediate data to ACC

Description The contents of the Accumulator and the specified immediate data are added. The result is stored in the Accumulator.

Operation ACC ← ACC + x

Affected flag(s) OV, Z, AC, C

ADDM A,[m] Add ACC to Data Memory

Description The contents of the specified Data Memory and the Accumulator are added. The result is stored in the specified Data Memory.

Operation [m] ← ACC + [m]

Affected flag(s) OV, Z, AC, C

AND A,[m] Logical AND Data Memory to ACC

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) Z

AND A,x Logical AND immediate data to ACC

Description Data in the Accumulator and the specified immediate data perform a bitwise logical AND operation. The result is stored in the Accumulator.

Operation ACC ← ACC "AND" x

Affected flag(s) Z

ANDM A,[m] Logical AND ACC to Data Memory

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) Z

CALL addr Subroutine call

Description Unconditionally calls a subroutine at the specified address. The Program Counter then increments by 1 to obtain the address of the next instruction which is then pushed onto the stack. The specified address is then loaded and the program continues execution from this new address. As this instruction requires an additional operation, it is a two cycle instruction.

Operation Stack ← Program Counter + 1
Program Counter ← addr

Affected flag(s) None

CLR [m] Clear Data Memory

Description Each bit of the specified Data Memory is cleared to 0.

Operation [m] ← 00H

Affected flag(s) None

CLR [m].i Clear bit of Data Memory

Description Bit i of the specified Data Memory is cleared to 0.

Operation [m].i ← 0

Affected flag(s) None

CLR WDT Clear Watchdog Timer

Description The TO, PDF flags and the WDT are all cleared.

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CLR WDT1 Pre-clear Watchdog Timer

Description The TO, PDF flags and the WDT are all cleared. Note that this instruction works in conjunction with CLR WDT2 and must be executed alternately with CLR WDT2 to have effect. Repetitively executing this instruction without alternately executing CLR WDT2 will have no effect.

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CLR WDT2 Pre-clear Watchdog Timer

Description The TO, PDF flags and the WDT are all cleared. Note that this instruction works in conjunction with CLR WDT1 and must be executed alternately with CLR WDT1 to have effect. Repetitively executing this instruction without alternately executing CLR WDT1 will have no effect.

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

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

Affected flag(s) Z

CPLA [m] Complement Data Memory with result in ACC

Description Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa. The complemented result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

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

Affected flag(s) Z

DAA [m] Decimal-Adjust ACC for addition with result in Data Memory

Description Convert the contents of the Accumulator value to a BCD (Binary Coded Decimal) value resulting from the previous addition of two BCD variables. If the low nibble is greater than 9 or if AC flag is set, then a value of 6 will be added to the low nibble. Otherwise the low nibble remains unchanged. If the high nibble is greater than 9 or if the C flag is set, then a value of 6 will be added to the high nibble. Essentially, the decimal conversion is performed by adding 00H, 06H, 60H or 66H depending on the Accumulator and flag conditions. Only the C flag may be affected by this instruction which indicates that if the original BCD sum is greater than 100, it allows multiple precision decimal addition.

Operation $[m] \leftarrow ACC + 00H$ or
$[m] \leftarrow ACC + 06H$ or
$[m] \leftarrow ACC + 60H$ or
$[m] \leftarrow ACC + 66H$

Affected flag(s) C

DEC [m] Decrement Data Memory

Description Data in the specified Data Memory is decremented by 1.

Operation $[m] \leftarrow [m] - 1$

Affected flag(s) Z

DECA [m] Decrement Data Memory with result in ACC

Description Data in the specified Data Memory is decremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation $ACC \leftarrow [m] - 1$

Affected flag(s) Z

HALT Enter power down mode

Description This instruction stops the program execution and turns off the system clock. The contents of the Data Memory and registers are retained. The WDT and prescaler are cleared. The power down flag PDF is set and the WDT time-out flag TO is cleared.

Operation $TO \leftarrow 0$
$PDF \leftarrow 1$

Affected flag(s) TO, PDF

INC [m]	Increment Data Memory
Description	Data in the specified Data Memory is incremented by 1.
Operation	[m] ← [m] + 1
Affected flag(s)	Z
INCA [m]	Increment Data Memory with result in ACC
Description	Data in the specified Data Memory is incremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.
Operation	ACC ← [m] + 1
Affected flag(s)	Z
JMP addr	Jump unconditionally
Description	The contents of the Program Counter are replaced with the specified address. Program execution then continues from this new address. As this requires the insertion of a dummy instruction while the new address is loaded, it is a two cycle instruction.
Operation	Program Counter ← addr
Affected flag(s)	None
MOV A,[m]	Move Data Memory to ACC
Description	The contents of the specified Data Memory are copied to the Accumulator.
Operation	ACC ← [m]
Affected flag(s)	None
MOV A,x	Move immediate data to ACC
Description	The immediate data specified is loaded into the Accumulator.
Operation	ACC ← x
Affected flag(s)	None
MOV [m],A	Move ACC to Data Memory
Description	The contents of the Accumulator are copied to the specified Data Memory.
Operation	[m] ← ACC
Affected flag(s)	None
NOP	No operation
Description	No operation is performed. Execution continues with the next instruction.
Operation	No operation
Affected flag(s)	None
OR A,[m]	Logical OR Data Memory to ACC
Description	Data in the Accumulator and the specified Data Memory perform a bitwise logical OR operation. The result is stored in the Accumulator.
Operation	ACC ← ACC ″OR″ [m]
Affected flag(s)	Z

OR A,x Logical OR immediate data to ACC

Description: Data in the Accumulator and the specified immediate data perform a bitwise logical OR operation. The result is stored in the Accumulator.

Operation: ACC ← ACC "OR" x

Affected flag(s): Z

ORM A,[m] Logical OR ACC to Data Memory

Description: Data in the specified Data Memory 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): Z

RET Return from subroutine

Description: The Program Counter is restored from the stack. Program execution continues at the restored address.

Operation: Program Counter ← Stack

Affected flag(s): None

RET A,x Return from subroutine and load immediate data to ACC

Description: The Program Counter is restored from the stack and the Accumulator loaded with the specified immediate data. Program execution continues at the restored address.

Operation: Program Counter ← Stack
ACC ← x

Affected flag(s): None

RETI Return from interrupt

Description: The Program Counter is restored from the stack and the interrupts are re-enabled by setting the EMI bit. EMI is the enable master (global) interrupt bit (bit 0; register INTC). If an interrupt was pending when the RETI instruction is executed, the pending Interrupt routine will be processed before returning to the main program.

Operation: Program Counter ← Stack
EMI ← 1

Affected flag(s): None

RL [m] Rotate Data Memory left

Description: The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0.

Operation: [m].(i+1) ← [m].i; (i = 0~6)
[m].0 ← [m].7

Affected flag(s): None

RLA [m] Rotate Data Memory left with result in ACC

Description: The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation: ACC.(i+1) ← [m].i; (i = 0~6)
ACC.0 ← [m].7

Affected flag(s): None

RLC [m] Rotate Data Memory left through Carry

Description The contents of the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into bit 0.

Operation
[m].(i+1) ← [m].i; (i = 0~6)
[m].0 ← C
C ← [m].7

Affected flag(s) C

RLCA [m] Rotate Data Memory left through Carry with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into the bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation
ACC.(i+1) ← [m].i; (i = 0~6)
ACC.0 ← C
C ← [m].7

Affected flag(s) C

RR [m] Rotate Data Memory right

Description The contents of the specified Data Memory are rotated right by 1 bit with bit 0 rotated into bit 7.

Operation
[m].i ← [m].(i+1); (i = 0~6)
[m].7 ← [m].0

Affected flag(s) None

RRA [m] Rotate Data Memory right with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated right by 1 bit with bit 0 rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation
ACC.i ← [m].(i+1); (i = 0~6)
ACC.7 ← [m].0

Affected flag(s) None

RRC [m] Rotate Data Memory right through Carry

Description The contents of the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7.

Operation
[m].i ← [m].(i+1); (i = 0~6)
[m].7 ← C
C ← [m].0

Affected flag(s) C

RRCA [m] Rotate Data Memory right through Carry with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation
ACC.i ← [m].(i+1); (i = 0~6)
ACC.7 ← C
C ← [m].0

Affected flag(s) C

SBC A,[m] Subtract Data Memory from ACC with Carry

Description The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

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

Affected flag(s) OV, Z, AC, C

SBCM A,[m] Subtract Data Memory from ACC with Carry and result in Data Memory

Description The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

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

Affected flag(s) OV, Z, AC, C

SDZ [m] Skip if decrement Data Memory is 0

Description The contents of the specified Data Memory are first decremented by 1. If the result is 0 the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation $[m] \leftarrow [m] - 1$
Skip if [m] = 0

Affected flag(s) None

SDZA [m] Skip if decrement Data Memory is zero with result in ACC

Description The contents of the specified Data Memory are first decremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.

Operation $ACC \leftarrow [m] - 1$
Skip if ACC = 0

Affected flag(s) None

SET [m] Set Data Memory

Description Each bit of the specified Data Memory is set to 1.

Operation $[m] \leftarrow FFH$

Affected flag(s) None

SET [m].i Set bit of Data Memory

Description Bit i of the specified Data Memory is set to 1.

Operation $[m].i \leftarrow 1$

Affected flag(s) None

SIZ [m] Skip if increment Data Memory is 0

Description The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation [m] ← [m] + 1
Skip if [m] = 0

Affected flag(s) None

SIZA [m] Skip if increment Data Memory is zero with result in ACC

Description The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

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

Affected flag(s) None

SNZ [m].i Skip if bit i of Data Memory is not 0

Description If bit i of the specified Data Memory is not 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is 0 the program proceeds with the following instruction.

Operation Skip if [m].i $\neq$ 0

Affected flag(s) None

SUB A,[m] Subtract Data Memory from ACC

Description The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC – [m]

Affected flag(s) OV, Z, AC, C

SUBM A,[m] Subtract Data Memory from ACC with result in Data Memory

Description The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation [m] ← ACC – [m]

Affected flag(s) OV, Z, AC, C

SUB A,x Subtract immediate data from ACC

Description The immediate data specified by the code is subtracted from the contents of the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC – x

Affected flag(s) OV, Z, AC, C

SWAP [m] Swap nibbles of Data Memory

Description The low-order and high-order nibbles of the specified Data Memory are interchanged.

Operation [m].3~[m].0 ↔ [m].7 ~ [m].4

Affected flag(s) None

SWAPA [m] Swap nibbles of Data Memory with result in ACC

Description The low-order and high-order nibbles of the specified Data Memory are interchanged. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

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

Affected flag(s) None

SZ [m] Skip if Data Memory is 0

Description If the contents of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation Skip if [m] = 0

Affected flag(s) None

SZA [m] Skip if Data Memory is 0 with data movement to ACC

Description The contents of the specified Data Memory are copied to the Accumulator. If the value is zero, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation ACC ← [m]
Skip if [m] = 0

Affected flag(s) None

SZ [m].i Skip if bit i of Data Memory is 0

Description If bit i of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.

Operation Skip if [m].i = 0

Affected flag(s) None

TABRDC [m] Read table (current page) to TBLH and Data Memory

Description The low byte of the program code (current page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

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

Affected flag(s) None

TABRDL [m] Read table (last page) to TBLH and Data Memory

Description The low byte of the program code (last page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

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

Affected flag(s) None

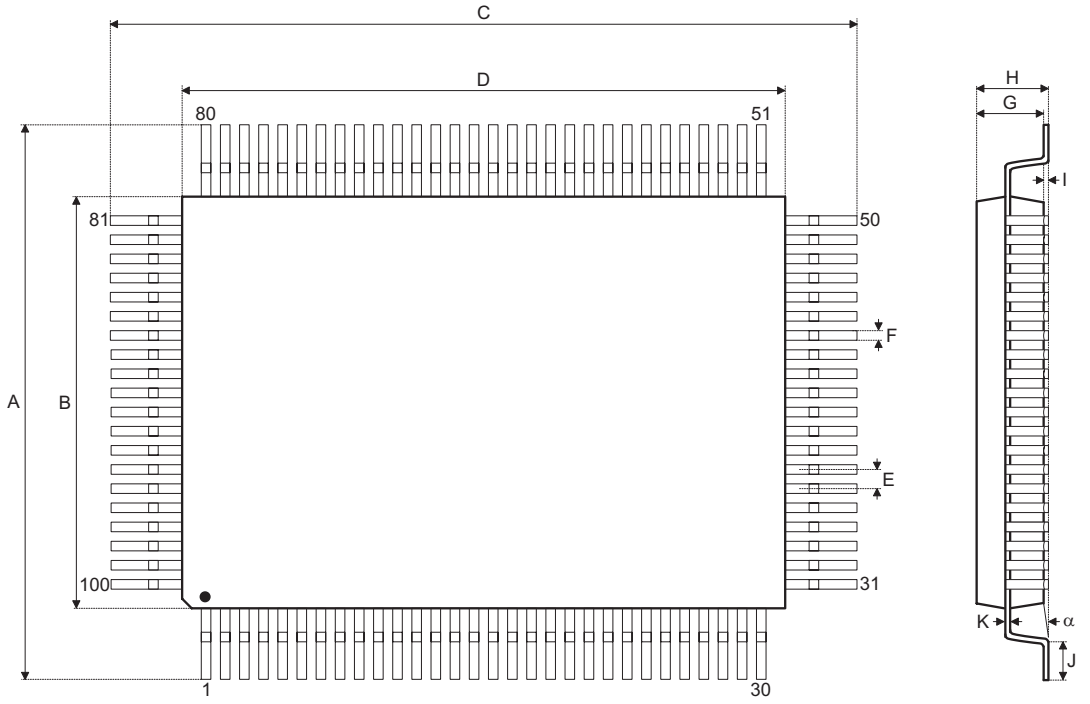
XOR A,[m]	Logical XOR Data Memory to ACC
Description	Data in the Accumulator and the specified Data Memory perform a bitwise logical XOR operation. The result is stored in the Accumulator.
Operation	ACC ← ACC ″XOR″ [m]
Affected flag(s)	Z

XORM A,[m]	Logical XOR ACC to Data Memory
Description	Data in the specified Data Memory and the Accumulator perform a bitwise logical XOR operation. The result is stored in the Data Memory.
Operation	[m] ← ACC ″XOR″ [m]
Affected flag(s)	Z

XOR A,x	Logical XOR immediate data to ACC
Description	Data in the Accumulator and the specified immediate data perform a bitwise logical XOR operation. The result is stored in the Accumulator.
Operation	ACC ← ACC ″XOR″ x
Affected flag(s)	Z

Package Information

100-pin QFP (14×20) Outline Dimensions



Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	18.5	—	19.2
B	13.9	—	14.1
C	24.5	—	25.2
D	19.9	—	20.1
E	—	0.65	—
F	—	0.3	—
G	2.5	—	3.1
H	—	—	3.4
I	—	0.1	—
J	1	—	1.4
K	0.1	—	0.2
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

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