HT46RU66/HT46CU66 A/D Type 8-Bit MCU with LCD

Technical Document

- Tools Information
- FAQs
- Application Note
  - HA0003E Communicating between the HT48 & HT46 Series MCUs and the HT93LC46 EEPROM
  - HA0004E HT48 & HT46 MCU UART Software Implementation Method
  - HA0005E Controlling the I2C bus with the HT48 & HT46 MCU Series
  - HA0047E An PWM application example using the HT46 series of MCUs
  - HA0075E MCU Reset and Oscillator Circuits Application Note

Features

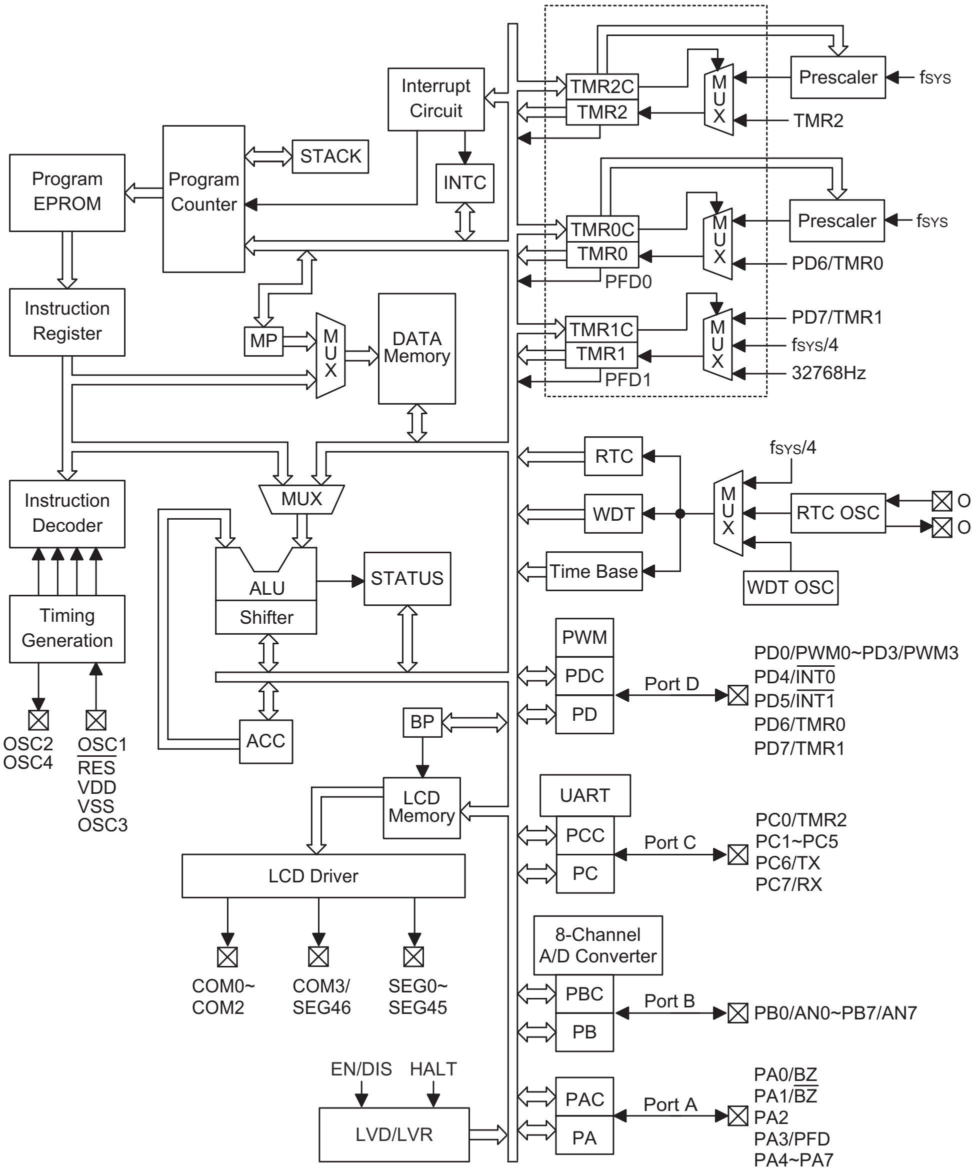
- Operating voltage:
  f_{SYS}=4MHz: 2.2V~5.5V
  f_{SYS}=8MHz: 3.3V~5.5V
- 32 bidirectional I/O lines
- Two external interrupt inputs
- Two 16-bit programmable timer/event counters with Programmable Frequency Divider, PFD, function
- One 8-bit programmable timer/event counter with source clock prescaler
- 47×3 or 46×4 segment LCD driver with logic output option for SEG0~SEG23)
- 16K×16 program memory
- 576×8 data memory RAM
- Universal Asynchronous Receiver Transmitter (UART)
- PFD function for sound generation
- Real Time Clock - RTC
- 8-bit RTC prescaler
- Watchdog Timer
- Buzzer output function
- Crystal, RC and 32768Hz crystal system oscillator option
- Power down and wake-up functions reduce power consumption
- 16-level subroutine nesting
- 8-channel 12-bit resolution A/D converter
- 4-channel PWM output shared with 4 I/O lines
- Bit manipulation instruction
- 16-bit table read instruction
- Up to 0.5μs instruction cycle with 8MHz system clock
- 63 powerful instructions
- Instruction execution within 1 or 2 machine cycles
- Low voltage reset/detector function
- 52-pin QFP, 56-pin SSOP, 100-pin QFP packages

General Description

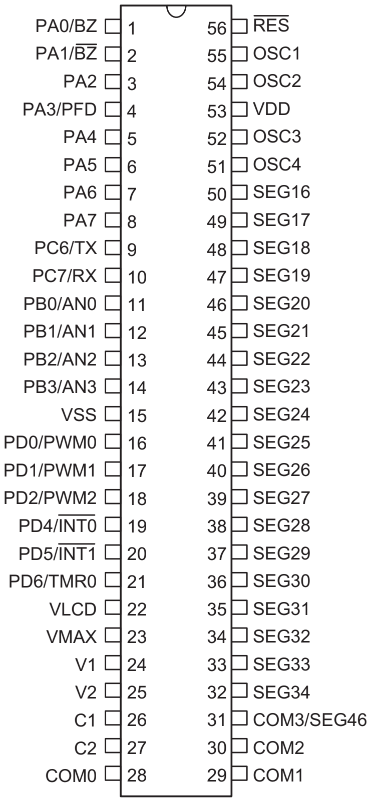
The HT46RU66/HT46CU66 are 8-bit, high performance, RISC architecture microcontroller devices specifically designed for A/D product applications that interface directly to analog signals and which require an LCD Interface. The HT46CU66 mask version device, is fully pin and functionally compatible with its sister HT46RU66 OTP device.

The advantages of low power consumption, I/O flexibility, timer functions, oscillator options, multi-channel A/D Converter, Pulse Width Modulation function, Power Down and Wake-up functions, in addition to a flexible and configurable LCD interface enhance the versatility of these devices to control a wide range of applications requiring analog signal processing and LCD interfacing, such as electronic metering, environmental monitoring, handheld measurement tools, motor driving, etc. for both the industrial and home appliance application areas.

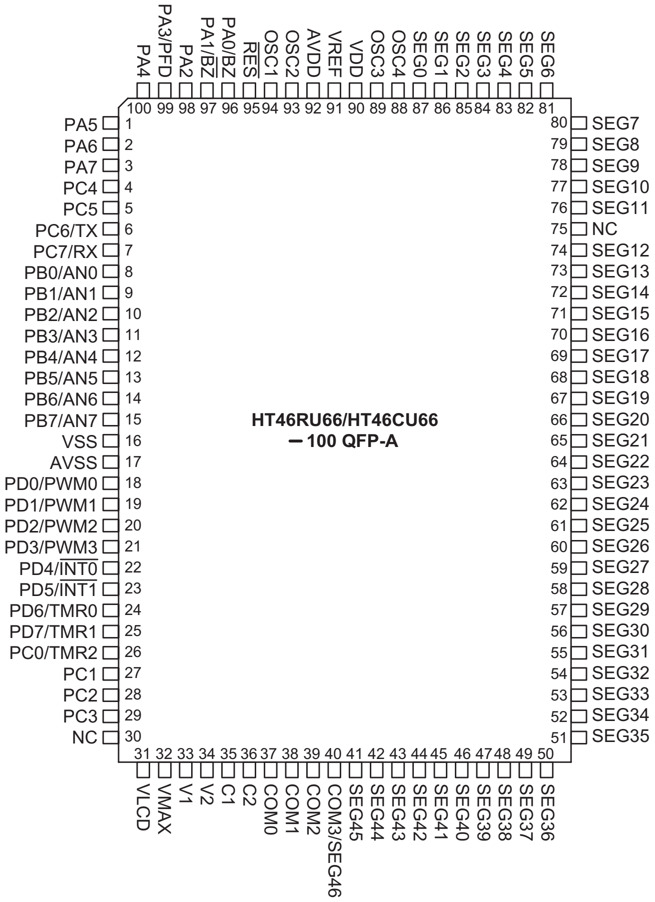
Block Diagram

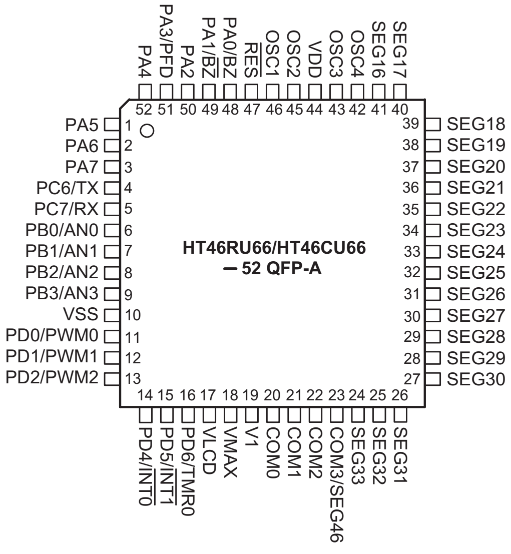


Pin Assignment



HT46RU66/HT46CU66
— 56 SSOP-A





Pin Description

Pin Name	I/O	Configuration Option	Description
PA0/BZ PA1/$\overline{\text{BZ}}$ PA2 PA3/PFD PA4~PA7	I/O	Pull-high Wake-up 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 the port have pull-high resistors. Pins PA0, PA1 and PA3 are pin-shared with BZ, $\overline{\text{BZ}}$ and PFD respectively, the function of which is chosen via configuration options.
PB0/AN0~ PB7/AN7	I/O	Pull-high	Bidirectional 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/TMR2 PC1~PC5 PC6/TX PC7/RX	I/O	Pull-high	Bidirectional 8-bit input/output port. Software instructions determine if the pin is a CMOS output or Schmitt Trigger input. A Configuration option determines if the port has pull-high resistors. Pin PC0 is pin-shared with the timer input pin TMR2. Pins PC6 and PC7 are pin-shared with the UART pins TX and RX.
PD0/PWM0 PD1/PWM1 PD2/PWM2 PD3/PWM3 PD4/$\overline{\text{INT0}}$ PD5/$\overline{\text{INT1}}$ PD6/TMR0 PD7/TMR1	I/O	Pull-high PWM Interrupt	Bidirectional 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. PD0~PD3 are pin-shared with PWM0~PWM3, the function of each pin is selected via a configuration option. Pins PD4 and PD5 are pin-shared with external interrupt input pins $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ respectively. Configuration options determine the interrupt enable/disable and the interrupt low/high trigger type. Pins PD6 and PD7 are pin-shared with the external timer input pins TMR0 and TMR1 respectively.
OSC1 OSC2	I O	Crystal or RC	OSC1 and OSC2 are connected to an external RC network or external crystal, determined by configuration option, for the internal system clock. For external RC system clock operation, OSC2 is an output pin where the system frequency can be monitored, at a frequency of 1/4 system clock. If an RTC oscillator on pins OSC3 and OSC4 is used as a system clock, then the OSC1 and OSC2 pins should be left floating.
OSC3 OSC4	I O	RTC or System Clock	OSC3 and OSC4 are connected to a 32768Hz crystal to form a real time clock for timing purposes or to form a system clock.
VLCD	—	—	LCD power supply
VMAX	—	—	IC maximum voltage, connect to V_{DD}, V_{LCD} or V1
V1, V2, C1, C2	I	—	LCD voltage pump
SEG0~SEG7	O	SEG0~SEG7 or CMOS output	LCD driver outputs for LCD panel segments. A configuration option can select all pins to be used as segment drivers or all pins to be used as CMOS outputs.
SEG8~SEG15	O	SEG8~SEG15 or CMOS output	LCD driver outputs for LCD panel segments. A configuration option can select all pins to be used as segment drivers or all pins to be used as CMOS outputs.
SEG16~SEG23	O	SEG16~SEG23 CMOS output	LCD driver outputs for LCD panel segments. Configuration options can select each pin to be used as either a segment driver or each pin to be used as a CMOS output.
SEG24~SEG45	O	—	LCD driver outputs for LCD panel segments
COM0~COM2 COM3/SEG46	O	1/3 or 1/4 Duty COM3 or SEG46	An LCD duty-cycle configuration option determines if SEG46 is configured as a segment driver or as a common output driver for the LCD panel. COM0~COM2 are the LCD common outputs.

Pin Name	I/O	Configuration Option	Description
VREF	I	—	Reference voltage input pin.
$\overline{\text{RES}}$	I	—	Schmitt Trigger reset input. Active low.
VDD	—	—	Positive power supply
VSS	—	—	Negative power supply, ground
AVDD	I	—	Positive analog power supply
AVSS	I	—	Negative analog power supply, ground

Note: Each pin on Port A can be programmed through a configuration option to have a wake-up function.
Individual pins can be selected to have a pull-high resistor.
Pins V2, C1, C2 and segment pin SEG34 are not available on the 52-pin QFP package.
Pins PB4/AN4~PB7/AN7 only exist on the 100-pin QFP package.
Pins PC0~PC5 only exist on the 100-pin QFP package.
Pin PD3/PWM3 only exists on the 100-pin QFP package.
Pins PD7/TMR1 and PC0/TMR2 only exist on the 100-pin QFP package. The 56-pin SSOP and 52-pin QFP packages have only one external timer input TMR0.
Segment pins SEG0~SEG15 and SEG35~SEG45 only exist on the 100-pin QFP package.
For the 52-pin QFP and 56-pin SSOP, the VREF, AVDD are bonded together with VDD pin.
For the 52-pin QFP and 56-pin SSOP, the AVSS is bonded together with VSS pin.

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
AV_{DD}	Analog Operating Voltage (see Note 5)	—	V_{REF}=AV_{DD}	3.0	—	5.5	V
I_{DD1}	Operating Current (Crystal OSC, RC OSC)	3V	No load, f_{SYS}=4MHz, ADC Off, UART Off	—	1	2	mA
		5V		—	3	5	mA
I_{DD2}	Operating Current (Crystal OSC, RC OSC)	3V	No load, f_{SYS}=4MHz, ADC Off, UART On	—	1.5	3	mA
		5V		—	3	6	mA
I_{DD3}	Operating Current (Crystal OSC, RC OSC)	5V	No load, f_{SYS}=8MHz, ADC Off, UART Off	—	4	8	mA
I_{DD4}	Operating Current (Crystal OSC, RC OSC)	5V	No load, f_{SYS}=8MHz, ADC Off, UART On	—	5	10	mA
I_{DD5}	Operating Current (f_{SYS}=32768Hz)	3V	No load, ADC Off, UART Off	—	0.3	0.6	mA
		5V		—	0.6	1	mA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{STB1}	Standby Current (*f_S=T1)	3V	No load, system HALT, LCD Off at HALT, UART Off	—	—	1	μA
		5V		—	—	2	μA
I_{STB2}	Standby Current (*f_S=RTC OSC)	3V	No load, system HALT, LCD On at HALT, C type, UART Off	—	2.5	5	μA
		5V		—	10	20	μA
I_{STB3}	Standby Current (*f_S=WDT OSC)	3V	No load, system HALT, LCD On at HALT, C type, UART Off	—	2	5	μA
		5V		—	6	10	μA
I_{STB4}	Standby Current (*f_S=RTC OSC)	3V	No load, system HALT, LCD On at HALT, R type, 1/2 bias, V_{LCD}=V_{DD}, UART Off (Low bias current option)	—	17	30	μA
		5V		—	34	60	μA
I_{STB5}	Standby Current (*f_S=RTC OSC)	3V	No load, system HALT, LCD On at HALT, R type, 1/3 bias, V_{LCD}=V_{DD}, UART Off (Low bias current option)	—	13	25	μA
		5V		—	28	50	μA
I_{STB6}	Standby Current (*f_S=WDT OSC)	3V	No load, system HALT, LCD On at HALT, R type, 1/2 bias, V_{LCD}=V_{DD}, `UART Off (Low bias current option)	—	14	25	μA
		5V		—	26	50	μA
I_{STB7}	Standby Current (*f_S=WDT OSC)	3V	No load, system HALT, LCD On at HALT, R type, 1/3 bias, V_{LCD}=V_{DD}, UART Off (Low bias current option)	—	10	20	μA
		5V		—	19	40	μA
V_{IL1}	Input Low Voltage for I/O Ports, TMR0, TMR1, $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$	—	—	0	—	0.3V_{DD}	V
V_{IH1}	Input High Voltage for I/O Ports, TMR0, TMR1, $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$	—	—	0.7V_{DD}	—	V_{DD}	V
V_{IL2}	Input Low Voltage ($\overline{\text{RES}}$)	—	—	0	—	0.4V_{DD}	V
V_{IH2}	Input High Voltage ($\overline{\text{RES}}$)	—	—	0.9V_{DD}	—	V_{DD}	V
V_{LVR}	Low Voltage Reset Voltage	—	—	2.7	3.0	3.3	V
V_{LVD}	Low Voltage Detector Voltage	—	—	3.0	3.3	3.6	V
V_{AD}	A/D Input Voltage	—	52QFP, 56SSOP	0	—	A_{VDD}	V
			100QFP	0	—	V_{REF}	V
V_{REF}	ADC Input Reference Voltage Range	—	AV_{DD}=3V	1.3	—	A_{VDD}	V
			AV_{DD}=5V	1.5	—	A_{VDD}	V
I_{OL1}	I/O Port Segment Logic Output Sink Current	3V	V_{OL}=0.1V_{DD}	6	12	—	mA
		5V		10	25	—	mA
I_{OH1}	I/O Port Segment Logic Output Source Current	3V	V_{OH}=0.9V_{DD}	–2	–4	—	mA
		5V		–5	–8	—	mA
I_{OL2}	LCD Common and Segment Current	3V	V_{OL}=0.1V_{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

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
R_{PH}	Pull-high Resistance of I/O Ports and $\overline{INT0}$, $\overline{INT1}$	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ
I_{ADC}	Additional Power Consumption if A/D Converter is Used	—	No load, t_{AD}=1μs	—	0.5	1	mA
				—	1.5	3	mA
DNL	ADC Differential Non-Linear	—	AV_{DD}=5V, V_{REF}=AV_{DD}, t_{AD}=1μs	—	—	±2	LSB
INL	ADC Integral Non-Linear	—	AV_{DD}=5V, V_{REF}=AV_{DD}, t_{AD}=1μs	—	±2.5	±4	LSB
RESOLU	Resolution	—	—	—	—	12	Bits

Note: 1. ″*f_S″ refer to the WDT clock option

2. I_{STB1}=WDT disable, I_{STB2}~I_{STB7}=WDT enable

3. Voltage level of AV_{DD} and V_{DD} must be the same.

A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS1}	System Clock (Crystal OSC, RC OSC)	—	2.2V~5.5V	400	—	4000	kHz
		—	3.3V~5.5V	400	—	8000	kHz
f_{SYS2}	System Clock (32768Hz Crystal OSC)	—	2.2V~5.5V	—	32768	—	Hz
f_{RTCOSC}	RTC Frequency	—	—	—	32768	—	Hz
f_{TIMER}	Timer I/P Frequency (TMR0/TMR1)	—	2.2V~5.5V	0	—	4000	kHz
		—	3.3V~5.5V	0	—	8000	kHz
t_{WDTOSC}	Watchdog Oscillator Period	3V	—	45	90	180	μs
		5V	—	32	65	130	μs
t_{RES}	External Reset Low Pulse Width	—	—	1	—	—	μs
t_{SST}	System Start-up Timer Period	—	Power-up or wake-up from HALT	—	1024	—	t_{SYS}
t_{INT}	Interrupt Pulse Width	—	—	1	—	—	μs
t_{AD}	A/D Clock Period	—	—	1	—	—	μs
t_{ADC}	A/D Conversion Time	—	—	—	80	—	t_{AD}
t_{ADCS}	A/D Sampling Time	—	—	—	32	—	t_{AD}

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

Functional Description

Execution Flow

The system clock is derived from either a crystal or an RC oscillator or a 32768Hz crystal oscillator. It is internally divided into four non-overlapping clocks. One instruction cycle consists of four system clock cycles.

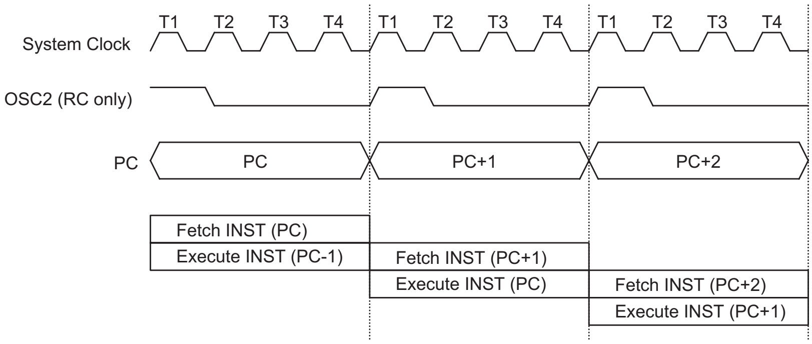
Instruction fetching and execution are pipelined in such a way that a fetch takes one instruction cycle while decoding and execution takes the next instruction cycle. The pipelining scheme allows each instruction to be effectively executed in a cycle. If an instruction changes the value of the program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

Program Counter – PC

The program counter is 14 bits wide and controls the sequence in which the instructions stored in the program ROM are executed. The contents of the PC can specify a maximum of 16384×16 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 instructions requiring jumps to non-consecutive addresses such as a jump instruction,a subroutine call, interrupt or reset, etc., the

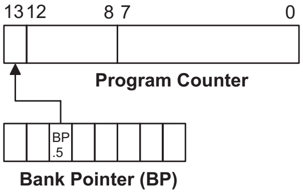


Execution Flow

Mode	Program Counter													
	*13	*12	*11	*10	*9	*8	*7	*6	*5	*4	*3	*2	*1	*0
Initial Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0
External Interrupt 0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
External Interrupt 1	0	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	0	1	1	0	0
Timer/Event Counter 1 Overflow	0	0	0	0	0	0	0	0	0	1	0	0	0	0
UART Bus Interrupt	0	0	0	0	0	0	0	0	0	1	0	1	0	0
Multi-function Interrupt	0	0	0	0	0	0	0	0	0	1	1	0	0	0
Skip	Program Counter + 2 (Within the current bank)													
Loading PCL	*13	*12	*11	*10	*9	*8	@7	@6	@5	@4	@3	@2	@1	@0
Jump, Call Branch	BP.5	#12	#11	#10	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
Return from Subroutine	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

Program Counter

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



microcontroller manages program control by loading the required address into the Program Counter. For conditional skip instructions, once the condition has been met, the next instruction, which has already been fetched during the present instruction execution, is discarded and a dummy cycle takes its place while the correct instruction is obtained.

The lower byte of the Program Counter, known as the Program Counter Low register or PCL, is available for program control and is a readable and writable register. By transferring data directly into this register, a short program jump can be executed directly, however, as only this low byte is available for manipulation, the jumps are limited to the present page of memory, that is 256 locations. When such program jumps are executed it should also be noted that a dummy cycle will be inserted.

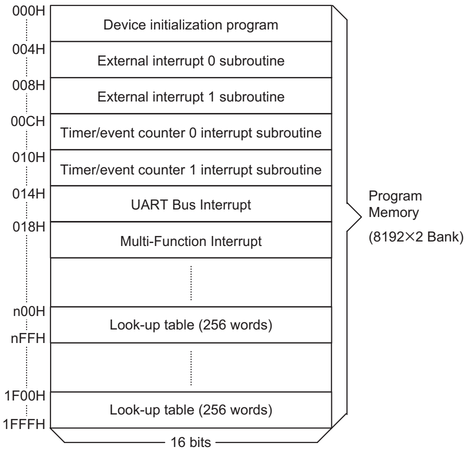
As the Program Memory is stored in two Banks, the Bank selection is under the control of bit 5 of the Bank Pointer. It is this Bank Pointer bit that controls the highest address bit of the Program Counter as shown in the diagram.

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 into a format of 16384×16 bits, which are addressed by the PC and table pointer. The Program Memory is divided into two banks, Bank0 and Bank1. Each bank has a capacity of 8192×16 bits and is selected using bit BP.5 in the the bank pointer register. With BP = 000XXXXXB, Bank0 is selected and with BP = 001XXXXXB, Bank1 is selected. The JMP and CALL instructions provide only 13 bits of address to allow branching within any 8K program memory bank. When executing a JMP or CALL instruction, the user must ensure that the bank pointer bit, BP.5, is programmed so that the desired program memory bank is addressed. If a return from a CALL instruction or interrupt is executed, the entire 14 bit PC is popped off the stack. Therefore, manipulation of the BP.5 is not required when the RET or RETI instructions are executed.

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

- Location 000H
  Location 000H is reserved for program initialisation. After a device reset, the program will jump to this location and begin execution.


000H
004H
008H
00CH
010H
014H
018H
n00H
nFFH
1F00H
1FFFH
Device initialization program
External interrupt 0 subroutine
External interrupt 1 subroutine
Timer/event counter 0 interrupt subroutine
Timer/event counter 1 interrupt subroutine
UART Bus Interrupt
Multi-Function Interrupt
Look-up table (256 words)
Look-up table (256 words)
16 bits
Program Memory
(8192×2 Bank)

Note: n ranges from 0 to 1F

Program Memory

- Location 004H
  Location 004H is reserved for the external interrupt service program. If the $\overline{\text{INT0}}$ input pin is activated, the interrupt is enabled, and the stack is not full, the program begins execution at location 004H.
- Location 008H
  Location 008H is also reserved for the external interrupt service program. If the $\overline{\text{INT1}}$ input pin is activated, the interrupt is enabled, and the stack is not full, the program will jump to this location and begin execution.
- Location 00CH
  Location 00CH is reserved for the Timer/Event Counter 0 interrupt service program. If a timer interrupt results from a Timer/Event Counter 0 overflow, and if the interrupt is enabled and the stack is not full, the program will jump to this location and begin execution.
- Location 010H
  Location 010H is reserved for the Timer/Event Counter 1 interrupt service program. If a timer interrupt results from a Timer/Event Counter 1 overflow, and if the interrupt is enabled and the stack is not full, the program will jump to this location and begin execution.
- Location 014H
  Location 014H is reserved for the UART Bus interrupt service program. If the UART Bus interrupt resulting from a transmission flag or reception is completed, and if the interrupt is enabled and the stack is not full, the program will jump to this location and begin execution.

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

Table Location

Note: *13~*0: Table location bits
@7~@0: Table pointer lower-order bits (TBLP)
TBHP: Table pointer higher-order bits

- Location 018H
  This area is reserved for the Multi-function interrupt service program. If a timer interrupt results from a Timer/Event Counter 2 overflow, or the real time clock time out, or Time base time out, and if the interrupt is enabled and the stack is not full, the program will jump to this location and begin execution.
- Table location
  Any location in the Program Memory can be used as a look-up table. The instructions ″TABRDC [m]″ (page specified by the TBHP) for 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. This is the Table Higher-order byte register (08H). 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 the TBLH. The TBLH is read only, the higher-order byte table pointer TBHP (1FH) and the table pointer, TBLP, is a read/write register (07H), indicating the table location. Before accessing the table, the location should be placed in the TBHP and TBLP registers. All the table related instructions require 2 cycles to complete the operation. These areas may function as a normal Program Memory depending upon the user′s requirements.

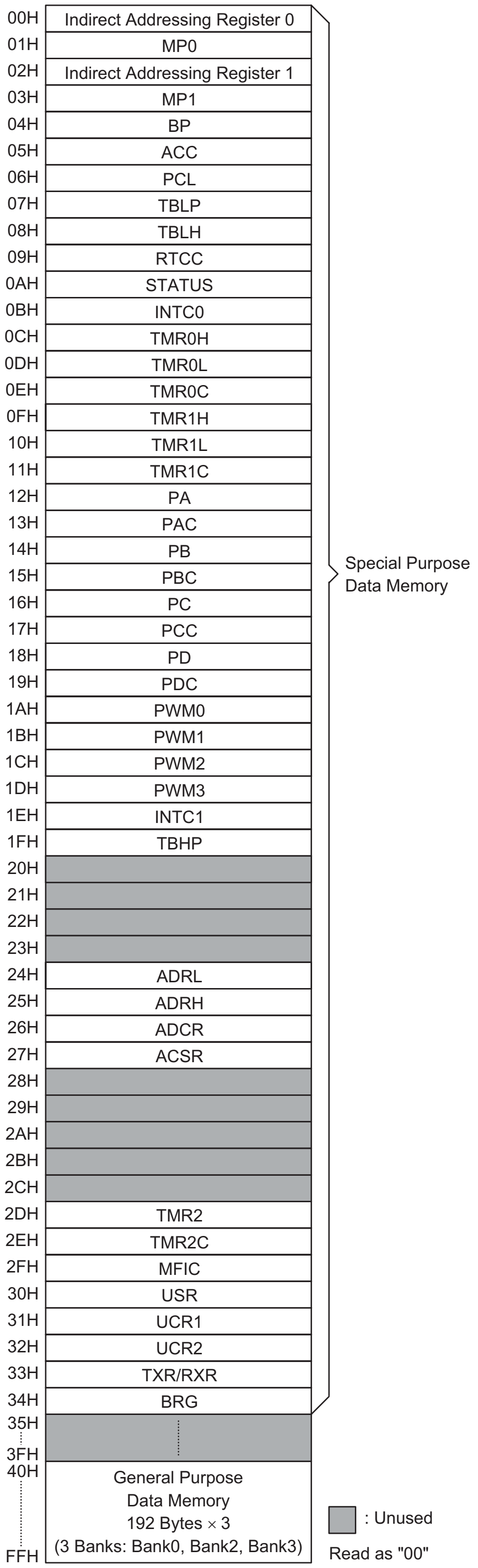
Stack Register – STACK

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, known as SP, which is neither readable nor writeable. At the 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, signaled by a return instruction, RET or RETI, the contents of the program counter is restored to its previous value from the stack. After a device reset, the stack pointer will point to the top of the stack.

If the stack is full and a non-masked interrupt takes place, the interrupt request flag is recorded but the acknowledge signal is still inhibited. Once the SP is decremented, using an RET or RETI instruction, the interrupt is serviced. This feature prevents stack overflow, allowing the programmer to use the structure easily. Likewise, if the stack is full, and a ″CALL″ is subsequently executed, a stack overflow occurs and the first entry is lost as only the most recent 16 return addresses are stored.

Data Memory – RAM

The Data Memory, RAM, has a structure of 620×8 bits, and is divided into two functional groups, namely; special function registers, 44×8 bits, and general purpose data memory (Bank 0: 192×8 bits, Bank 2: 192×8 bits



00H	Indirect Addressing Register 0
01H	MP0
02H	Indirect Addressing Register 1
03H	MP1
04H	BP
05H	ACC
06H	PCL
07H	TBLP
08H	TBLH
09H	RTCC
0AH	STATUS
0BH	INTC0
0CH	TMR0H
0DH	TMR0L
0EH	TMR0C
0FH	TMR1H
10H	TMR1L
11H	TMR1C
12H	PA
13H	PAC
14H	PB
15H	PBC
16H	PC
17H	PCC
18H	PD
19H	PDC
1AH	PWM0
1BH	PWM1
1CH	PWM2
1DH	PWM3
1EH	INTC1
1FH	TBHP
20H	▨
21H	▨
22H	▨
23H	▨
24H	ADRL
25H	ADRH
26H	ADCR
27H	ACSR
28H	▨
29H	▨
2AH	▨
2BH	▨
2CH	▨
2DH	TMR2
2EH	TMR2C
2FH	MFIC
30H	USR
31H	UCR1
32H	UCR2
33H	TXR/RXR
34H	BRG
35H 3FH	▨
40H FFH	General Purpose Data Memory 192 Bytes × 3 (3 Banks: Bank0, Bank2, Bank3)

▨ : Unused
Read as "00"

RAM Mapping

and Bank 3: 192×8 bits). Most of these registers are readable and writeable, although some are read only. The special function registers are overlapped in any bank. Of the two types of functional groups, the special function registers consist of an Indirect addressing register 0 (00H), a Memory pointer register 0 (MP0;01H), an Indirect addressing register 1 (02H), a Memory pointer register 1 (MP1;03H), a Bank pointer (BP;04H), an Accumulator (ACC;05H), a Program counter lower-order byte register (PCL;06H), a table pointer lower-order byte (TBLP;07H), a table pointer higher-order byte (TBHP;1FH), a Table higher-order byte register (TBLH;08H), a Real time clock control register (RTCC;09H), a Status register (STATUS;0AH), an Interrupt control register 0 (INTC0;0BH), a Timer/Event Counter 0 (TMR0H;0CH; TMR0L;0DH), a Timer/Event Counter 0 control register (TMR0C;0EH), a Timer/Event Counter 1 (TMR1H:0FH;TMR1L;10H), a Timer/Event Counter 1 control register (TMR1C;11H), a Timer/Event Counter 2 (TMR2;2DH), a Timer/Event Counter 2 control register (TMR2C;2EH), Interrupt control register 1 (INTC1;1EH), Multi-function Interrupt control register 1 (MFIC;2FH), PWM data register (PWM0;1AH, PWM1;1BH, PWM2;1CH, PWM3;1DH), the A/D result lower-order byte register (ADRL;24H), the A/D result higher-order byte register (ADRH;25H), the A/D control register (ADCR;26H), the A/D clock setting register (ACSR;27H), I/O registers (PA;12H, PB;14H, PC;16H, PD;18H) and I/O control registers (PAC;13H, PBC;15H, PCC;17H, PDC;19H), the UART Bus status register control register (USR;30H), the UART Bus control register 1 (UCR1;31H), the UART Bus control register 2 (UCR2;32H), the UART Bus transmit and receive register (TXR/RXR;33H), the UART Bus Baud Rate Generator register (BRG;34H).

The Data Memory space before address 40H is reserved for future expansion usage and reading these locations will retrieve a value of ″00H″. The space before 40H overlaps in each bank. The general purpose data memory, addressed from 40H to FFH (Bank0; BP=0, Bank2; BP=2 or Bank3; BP=3), is used for data and control information under instruction commands. All of the data memory areas can directly handle arithmetic, logic, increment, decrement and rotate operations . Except for some dedicated bits, each bit in the data memory can be set and reset by ″SET [m].i″ and ″CLR [m].i″. They are also indirectly accessible through the memory pointer registers, MP0;01H and MP1;03H.

Indirect Addressing Register

Location 00H and 02H are indirect addressing registers that are not physically implemented. Any read/write operation of [00H] and [02H] accesses the Data Memory pointed to by MP0 and MP1 respectively. Reading location 00H or 02H indirectly returns the result 00H. Writing to it indirectly results 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 used to access the Data Memory in combination with their corresponding indirect addressing registers. MP0 can only be used to access data from Bank 0, while MP1 can be used to access data from all banks.

Accumulator – ACC

The accumulator, ACC, is related to the ALU operations. It is also mapped to location 05H of the Data Memory, 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 (0AH) 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 operational sequence.

Except for the TO and PDF flags, bits in the status register can be altered by instructions similar to the other registers. Data written into the status register does not alter the TO or PDF flags. Operations related to the status register, however, may yield different results from those intended. The TO and PDF flags can only be changed by a Watchdog Timer overflow, device power-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 is important, and if the subroutine is likely to corrupt the status register, the programmer should take precautions to save it properly.

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

Interrupts – INTC0, INTC1, MFIC

The device provides two external interrupts, two Internal Timer/Event Counter (0/1) interrupts, a UART Bus interrupt, and a Multi-function interrupt. The Multi-function interrupt includes the internal Timer/Event Counter 2 interrupt, the internal real time clock interrupt, and the internal time base interrupt . The Interrupt Control register 0, INTC0;0BH, interrupt control register 1, INTC1;1EH, and the Multi-Function interrupt control register, MFIC;2FH, contain the interrupt control bits that are used to set the enable/disable status and interrupt request flags.

Once an interrupt subroutine is serviced, other interrupts are all blocked automatically as the EMI bit is cleared. This scheme may prevent any further interrupt nesting. Other interrupt requests may take place during this interval, but only the interrupt request flag will be recorded. If a certain interrupt requires servicing within the service routine, the EMI bit and the corresponding bits in the INTC0, INTC1 and MFIC registers may be set in order to allow interrupt nesting. Once the stack is full, the interrupt request will not be acknowledged, even if the related interrupt is enabled, until the SP is decremented. If immediate service is desired, the stack should be prevented from becoming full.

All these interrupts can support 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 register or the status register is altered by the interrupt service program which corrupts the desired control sequence, the contents should be saved in advance.

External interrupts are triggered by an edge transition on pins $\overline{\text{INT0}}$ or $\overline{\text{INT1}}$. A configuration option exists to select one of three transition types, either high to low, low to high or both. The related interrupt request flag, EIF0; bit 4 of the INTC0 register and EIF1; bit 5 of the INTC0 register, will be set when an external interrupt occurs. After the interrupt is enabled, the stack is not full, and the external interrupt is active, a subroutine call to location 04H or 08H occurs. The interrupt request flag, EIF0 or EIF1 and the EMI bits are all cleared to disable other maskable interrupts.

The internal Timer/Event Counter 0 interrupt is initialised by setting the Timer/Event Counter 0 interrupt request flag, T0F; bit 6 of the INTC0 register, which is normally caused by a timer overflow. After the interrupt is enabled, and the stack is not full, and the T0F bit is set, a subroutine call to location 0CH occurs. The related interrupt request flag, T0F, is reset, and the EMI bit is cleared to disable other maskable interrupts.

The Timer/Event Counter 1 and Timer/Event Counter 2 operates in the same manner, The Timer/Event Counter 1 related interrupt request flag is T1F, bit 4 of the INTC1 register, and its subroutine call location is 010H. The Timer/Event Counter 2 related interrupt request flags are MFF, bit 6 of the INTC1 register, and T2F, bit 4 of the MFIC register, and its subroutine call location is 018H. The related interrupt request flags, T1F and MFF, will be reset and the EMI bit cleared to disable further interrupts. T2F, bit 4 of the MFIC register, will not be cleared automatically, and should be cleared by the user.

The UART Bus interrupt is initialized by setting the UART Bus interrupt request flag, URF; bit 5 of the INTC1 register, caused by transmit data register empty (TXIF), received data available(RXIF), transmission idle (TIDLE), Over run error (OERR) or Address detected. When the interrupt is enabled, the stack is not full and the TXIF, RXIF, TIDLE, OERR bit is set or an address is detected, a subroutine call to location 014H will occur. The related interrupt request flag, URF, will be reset and the EMI bit cleared to disable further interrupts.

The Multi-Function Interrupt, MFI, is initialised by setting the interrupt request flag, MFF; bit 6 of the INTC1 register, that is caused by a Timer 2 overflow, T2F; bit 4 of the MFIC register, caused by a regular real time clock time-out, RTF; bit 6 of the MFIC register or caused by a time base time-out, TBF; bit5 of the MFIC register. After the interrupt is enabled, EMFI=1, the stack is not full, and the MFF bit is set, a subroutine call to location 018H will occur. The related interrupt request flag, MFF, is reset and the EMI bit is cleared to disable further maskable interrupts. T2F, TBF and RTF indicate that a related interrupt has occurred. As these flags will not be cleared automatically after reading, they should be cleared by the user.

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 may be executed. RETI sets the EMI bit and enables an interrupt service, but RET does not.

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

Bit No.	Label	Function
0	EMI	Controls the master (global) interrupt (1=enable; 0= disable)
1	EEI0	Controls the external interrupt 0 (1=enable; 0=disable)
2	EEI1	Controls the external interrupt 1 (1=enable; 0=disable)
3	ET0I	Controls the Timer/Event Counter 0 interrupt (1=enable; 0=disable)
4	EIF0	External interrupt 0 request flag (1=active; 0=inactive)
5	EIF1	External interrupt 1 request flag (1=active; 0=inactive)
6	T0F	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	Controls the Timer/Event Counter 1 interrupt (1=enable; 0=disable)
1	EURI	Control the UART Bus interrupt (1=enable; 0=disable)
2	EMFI	Control the Multi-function interrupt (1=enable; 0=disable)
3, 7	—	Unused bit, read as ″0″
4	T1F	Internal Timer/Event Counter 1 request flag (1=active; 0=inactive)
5	URF	UART Bus request flag (1=active; 0=inactive)
6	MFF	Multi-function interrupt request flag (1=active; 0=inactive)

INTC1 (1EH) Register

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

MFIC (2FH) Register

Interrupt Source	Priority	Vector
External Interrupt 0	1	04H
External Interrupt 1	2	08H
Timer/Event Counter 0 Overflow	3	0CH
Timer/Event Counter 1 Overflow	4	10H
UART Bus Interrupt	5	14H
Multi-function Interrupt (Timer/Event Counter 2 / Real time clock / Time base overflow)	6	18H

The Timer/Event Counter 0 interrupt request flag, T0F, the external interrupt 1 request flag, EIF1, the external interrupt 0 request flag, EIF0, the enable Timer/Event Counter0 interrupt bit, ET0I, the enable external interrupt 1 bit, EEI1, the enable external interrupt 0 bit, EEI0, and the enable master interrupt bit, EMI, make up the Interrupt Control register 0, INTC0, which is located at 0BH in the Program Memory.

The multi-function interrupt request flag, MFF, the UART interrupt request flag, URF, the Timer/Event Counter 1 interrupt request flag, T1F, the enable multi-function interrupt bit, EMFI, and the enable UART interrupt bit, EURI, and the enable Timer/Event Counter 1 interrupt bit, ET1I, constitute the Interrupt Control register 1, INTC1, which is located at 1EH in the Program Memory.

The Time base interrupt request flag, TBF, the real time interrupt request flag, RTF, the Timer/Event Counter 2 interrupt request flag, T2F, the enable time base interrupt bit, ETBI, the enable real time interrupt bit, ERTI, and the enable Timer/Event counter 2 interrupt bit, ET2I, constitute the Multi-function Interrupt Control register, MFIC, which is located at 2FH in the Program Memory.

The EMI, EEI0, EEI1, ET0I, ET1I, EURI and EMFI bits are used to control the enabling/disabling of interrupts. These bits prevent the requested interrupt from being serviced. Once the interrupt request flags, EIF0, EIF1, T0F, T1F, URF, MFF, are set, they will remain in the INTC0 and INTC1 registers until the interrupts are serviced or cleared by a software instruction.

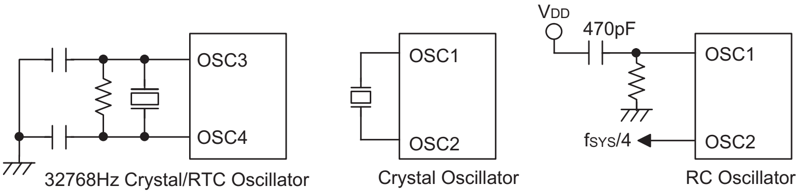
The Timer/Event Counter 2 overflow interrupt flag, T2F; bit 4 of the MFIC register, the real time clock interrupt flag, RTF; bit 6 of the MFIC register, the time base interrupt flag, TBF; bit 5 of the MFIC register, indicate that a related interrupt has occurred. As these flags will not be cleared automatically, they should be cleared by the user. The enable control Timer 2 interrupt, ET2I, the enable time base interrupt, ETBI, the enable real time clock interrupt, ERTI, constitute the Interrupt Control Register 2, MFIC, which is located at 2FH in the Program Memory.

It is recommended that a program does not use the ″CALL″ instruction within the interrupt subroutine. Interrupts often occur in an unpredictable manner or need to be serviced immediately in some applications. If only one stack is left and the interrupt enabling is not well controlled, the original control sequence may be damaged if a ″CALL″ is executed.

Oscillator Configuration

The device provides three oscillator circuits for system clocks. These are an RC oscillator, a crystal oscillator and a 32768Hz crystal oscillator, the choice of which is determined by a configuration option. The Power Down mode will stop the system oscillator, if it is an RC or crystal oscillator type and will ignore external signals in order to conserve power. If the 32768Hz crystal oscillator is selected as the system oscillator, it will continue to run in the Power Down mode, but the instruction execution will be stopped. Since the 32768Hz oscillator is also designed for timing purposes, the internal timing (RTC, time base, WDT) operation still runs even if the system enters the Power Down mode.

Of the three oscillators, if the RC oscillator is used, an external resistor between OSC1 and VSS is required, whose range should be within 24kΩ to 1MΩ. The system clock frequency divided by 4, can be monitored on pin OSC2 if a pull-high resistor is added. This can be used to synchronise external logic. The RC oscillator provides the most cost effective solution. However, the frequency of the oscillation may vary with VDD, temperature, and the chip itself due to process variations. It is therefore, not suitable for timing sensitive operations where accurate an oscillator frequency is desired.



System Oscillator

Note: 32768Hz crystal enable condition: For WDT clock source or for system clock source.

The external resistor and capacitor components connected to the 32768Hz crystal are not necessary to provide oscillation. For applications where precise RTC frequencies are essential, these components may be required to provide frequency compensation due to different crystal manufacturing tolerances.

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

The other oscillator circuit is for the real time clock or RTC, which has a fixed frequency of 32.768kHz. A 32.768kHz crystal oscillator should be connected between OSC3 and OSC4 for its implementation.

The RTC oscillator circuit has a quick start up function which can be activated by setting the ″QOSC″ bit, which is bit 4 of the RTCC register. It is recommended to turn this bit on at power on, and then turn it off after 2 seconds to conserve power.

The WDT oscillator is a free running on-chip RC oscillator, which requires no external components. When the system enters the power down mode, the system clock will stop, but the WDT oscillator will continue to operate, with a period of approximately 65μs at 5V. The WDT oscillator can be disabled by a configuration option to conserve power.

Watchdog Timer – WDT

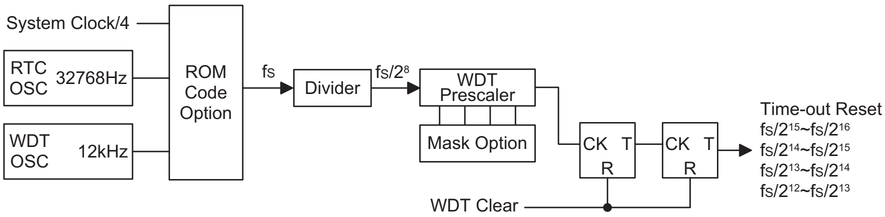
The WDT clock source can come from its own dedicated internal WDT oscillator, from the instruction clock (system clock/4), or from the real time clock oscillator (RTC oscillator). The timer is designed to prevent software malfunctions or sequences from jumping to unknown locations with unpredictable results. The WDT can be disabled by a configuration option. If the WDT is disabled, all executions related to the WDT result in no operation.

The WDT clock source is divided by 2^{12}~2^{15} , the actual value chosen by a configuration option, to get the WDT time-out period. For the WDT internal oscillator, the minimum WDT time-out period is about 300ms~600ms. This time-out period may vary with temperature, VDD and process variations. By using configuration options to set the WDT prescaler, longer time-out periods can be realised. If the WDT time-out is selected as 2^{15}, the maximum time-out period is divided by 2^{15}~2^{16}. This will give a time of about 2.1s~4.3s for the internal WDT oscillator. If the WDT oscillator is disabled, the WDT clock may still come from the instruction clock. The WDT will operate in the same manner except that in the Power Down mode, the WDT will stop counting and lose its protecting purpose. In this situation the system can only be restarted by external logic. If the device operates in a noisy environment, using the on-chip WDT internal oscillator is strongly recommended, since the Power Down mode will stop the system clock.

A WDT overflow under normal operation initialises a ″device reset″ and sets the status bit ″TO″. In the Power Down mode, the overflow initialises a ″warm reset″, and only the program counter and stack pointer are reset to zero. To clear the WDT contents, three methods can be adopted. These are an external reset which is a low level to $\overline{\text{RES}}$, a software instruction, and a ″HALT″ instruction. There are two types of software instructions; the single ″CLR WDT″ instruction and the instruction pair – ″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 options – ″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. In the case that ″CLR WDT1″ and ″CLR WDT2″ are 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

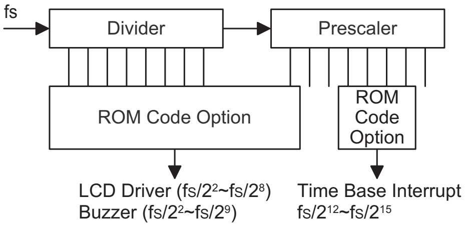
The device provides a multi-function timer for the WDT, time base and the RTC but with different time-out periods. The multi-function timer consists of an 8-stage divider and a 7-bit prescaler, with the clock source coming from the WDT OSC or RTC OSC or the instruction clock (i.e., system clock divided by 4). The multi-function timer also provides a selectable frequency signal (ranging from $f_S/2^2$ to $f_S/2^8$) for the LCD driver circuits, and a selectable frequency signal, ranging from $f_S/2^2$ to $f_S/2^9$, for the buzzer output, setup by configuration options. It is recommended to select a frequency as near to 4kHz as possible for the LCD driver circuits for clarity.



Watchdog Timer

Time Base

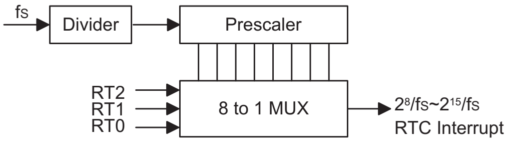
The time base offers a periodic time-out period to generate a regular internal interrupt. Its time-out period ranges from $2^{12}/f_S$ to $2^{15}/f_S$ selected by options. If a time base time-out occurs, the related interrupt request flags, TBF; bit 5 of the MFIC register, and MFF; bit 6 of the INTC1 register, will be set. If the interrupt is enabled, and the stack is not full, a subroutine call to location 18H occurs. The time base time-out signal can also be applied as a clock source for the Timer/Event Counter 1 to obtain longer time-out periods.



Time Base

Real Time Clock – RTC

The real time clock operates in the same manner as the time base in that it is used to supply a regular internal interrupt. Its time-out period ranges from $f_S/2^8$ to $f_S/2^{15}$, the actual value of which is setup by software programming. Writing data to the RT2, RT1 and RT0 bits in the RTCC register, provides various time-out periods. If an RTC time-out occurs, the related interrupt request flag, RTF; bit 6 of the MFIC and MFF; bit 6 of the INTC1, is set. If the interrupt is enabled, and the stack is not full, a subroutine call to location 18H occurs. The real time clock time-out signal can also be applied as a clock source for Timer/Event Counter 0 in order to get longer time-out period.



Real Time Clock

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

Note: ″*″ not recommended to be used

Power Down Operation – HALT

The Power Down mode is initialised by the ″HALT″ instruction and results in the following.

- The system oscillator turns off but the WDT oscillator keeps running if the internal WDT oscillator or the real time clock is selected.
- The contents of the on-chip RAM and of the registers remain unchanged.
- The WDT is cleared and starts recounting, if the WDT clock source comes 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 way of an external reset, an interrupt, an external falling edge signal on port A, or a WDT overflow. An external reset causes device initialisation, and a WDT overflow performs a ″warm reset″. After examining the TO and PDF flags, the reason behind the chip reset can be determined. The PDF flag is cleared by a system power-up or by executing the ″CLR WDT″ instruction, and is set by executing the ″HALT″ instruction. The TO flag is set if a WDT time-out occurs, and causes a wake-up that only resets the program counter and stack pointer, and leaves the other registers in their original state.

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

When an interrupt request flag is set before entering the ″HALT″ state, the system cannot be awakened using that interrupt.

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

To minimize power consumption, all the I/O pins should be carefully managed before entering the Power Down state.

Reset

There are several ways in which a reset may occur.

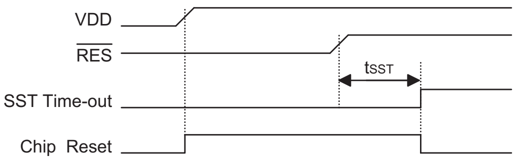
- $\overline{\text{RES}}$ is reset during normal operation
- Power on reset
- $\overline{\text{RES}}$ is reset during a Power Down
- WDT time-out is reset during normal operation
- WDT time-out during a Power Down

A WDT time-out when the device is in the Power Down mode differs from the 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 the other reset conditions. Most registers are reset to their initial condition once the reset conditions are met. By examining the PDF and TO flags, the program can distinguish between the different types of resets.

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

Note: ″u″ stands for unchanged

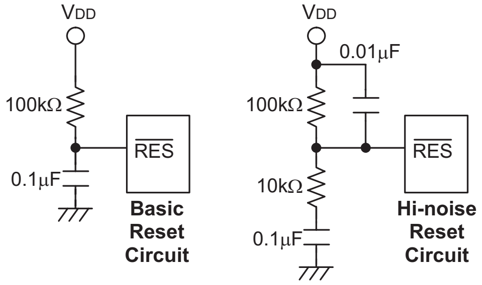
To guarantee that the system oscillator has started and stabilized, the SST (System Start-up Timer) provides an extra delay of 1024 system clock pulses when the system awakens from the Power Down mode or during power up.



Reset Timing Chart

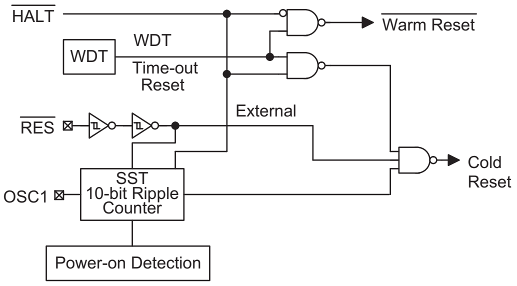
The functional unit chip reset status is shown below.

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



Reset Circuit

Note: Most applications can use the Basic Reset Circuit as shown, however for applications with extensive noise, it is recommended to use the Hi-noise Reset Circuit.



Reset Configuration

The register states are summarized below:

Register	Reset (Power On)	WDT Time-out (Normal Operation)	$\overline{\text{RES}}$ Reset (Normal Operation)	$\overline{\text{RES}}$ Reset (HALT)	WDT Time-out (HALT)*
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
BP	0000 0000	0000 0000	0000 0000	0000 0000	00u0 00uu
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	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
TMR0H	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR0L	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR0C	00-0 1000	00-0 1000	00-0 1000	00-0 1000	uu-u uuuu
TMR1H	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1L	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1C	0000 1---	0000 1---	0000 1---	0000 1---	uuuu u---
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PWM0	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PWM1	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PWM2	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PWM3	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
INTC1	-000 -000	-000 -000	-000 -000	-000 -000	-uuu -uuu
TBHP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
ADRL	xxxx ----	xxxx ----	xxxx ----	xxxx ----	uuuu ----
ADRH	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCR	0100 0000	0100 0000	0100 0000	0100 0000	uuuu uuuu
ACSR	1--- --00	1--- --00	1--- --00	---- --00	u--- --uu
TMR2	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR2C	00-0 1000	00-0 1000	00-0 1000	00-0 1000	uu-u uuuu
MFIC	-000 -000	-000 -000	-000 -000	-000 -000	-uuu -uuu
USR	0000 1011	0000 1011	0000 1011	0000 1011	uuuu uuuu

Register	Reset (Power On)	WDT Time-out (Normal Operation)	$\overline{\text{RES}}$ Reset (Normal Operation)	$\overline{\text{RES}}$ Reset (HALT)	WDT Time-out (HALT)*
UCR1	0000 00x0	0000 00x0	0000 00x0	0000 00x0	uuuu uuuu
UCR2	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
TXR/RXR	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
BRG	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu

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

Timer/Event Counter

Three timer/event counters are implemented in this microcontroller. The Timer/Event Counter 0 contains a 16-bit programmable count-up counter whose clock may come from an external or internal source. Its internal clock source comes from f_{SYS}. The Timer/Event Counter 1 contains a 16-bit programmable count-up counter whose clock may come from an external or internal source. Its internal clock source comes from either $f_{SYS}/4$ or the 32768Hz RTC oscillator selected via configuration option. The Timer/Event Counter 2 contains an 8-bit programmable count-up counter whose clock may come from an external or internal source. Its internal clock source comes from f_{SYS}. The external clock input allows the user to count external events, measure time intervals or pulse widths, or generate an accurate time base.

There are eight registers related to the Timer/Event Counter 0; TMR0H (0CH), TMR0L (0DH), TMR0C (0EH) and the Timer/Event Counter 1; TMR1H (0FH), TMR1L (10H), TMR1C (11H) and the Timer/Event Counter 2; TMR2 (2CH) TMR2C (2DH). Writing to TMR0L (TMR1L) will only put the written data into an internal lower-order byte buffer (8-bit) while writing to TMR0H (TMR1H) will transfer the specified data and the contents of the lower-order byte buffer into the TMR0H (TMR1H) and TMR0L (TMR1L) registers, respectively. The Timer/Event Counter 1/0 preload register is changed by each write operation to TMR0H (TMR1H). Reading from the TMR0H (TMR1H) will latch the contents of the TMR0H (TMR1H) and TMR0L (TMR1L) counters to the destination and the lower-order byte buffer, respectively. Reading from TMR0L (TMR1L) will read the contents of the lower-order byte buffer. Writing to TMR2 places the start value into the Timer/Event Counter 2 preload register, and reading from TMR2 retrieves the contents of the Timer/Event Counter 2. The TMR0C (TMR1C,TMR2C) register is the Timer/Event Counter 0 (1, 2) control register, which defines the operating mode, enable or disable function and the active edge.

The T0M0, T0M1 (TMR0C), T1M0, T1M1 (TMR1C) and T2M0, T2M1 (TMR2C) bits define the operational mode. The event count mode is used to count external events, which means that the clock source comes from the external, TMR0, TMR1 or TMR2 pin. 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 count the duration of a high or low level external signal on pin TMR0, TMR1 or TMR2. The counting is based on the internally selected clock source.

In the event count or timer mode, the Timer/Event Counter 0 (1) starts counting at the current contents in the timer/event counter and ends at FFFFH. Timer/Event Counter 2 starts counting at the current contents in the timer/event counter and ends at FFH. Once an overflow occurs, the counter is reloaded from the timer/event counter preload register, and generates an interrupt request flag which are T0F; bit 6 of INTC0, T1F; bit 4 of INTC1, T2F; bit 4 of MFIC and bit 6 of INTC1.

To enable the pulse width measurement mode, the operating mode select bits should both be set high. After the TMR0/TMR1/TMR2 pin has received a transient from low to high, or high to low if the T0E/T1E/T2E bit is ″0″, it will start counting until the TMR0/TMR1/ TMR2 pin returns to its original level a which point the T0ON/T1ON/T2ON bit will be auomatically reset.

The measured result remains in the timer/event counter even if the activated transient occurs again. In other words, only a single shot measurement can be made. Not until the T0ON/T1ON/T2ON bit is again set by the program, can further pulse width measurements be made. In this operation mode, the timer/event counter begins counting not according to the logic level but to the transient edges. In the case of counter overflows, the counter is reloaded from the timer/event counter register and issues an interrupt request, as in the other two modes, i.e., event and timer modes.

To enable the counting operation, the Timer ON bit, T0ON; bit 4 of TMR0C, T1ON; bit 4 of TMR1C, or T2ON; bit 4 of TMR2C, should be set to 1. In the pulse width measurement mode, the T0ON/T1ON/ T2ON is automatically cleared after a measurement cycle is completed. But in the other two modes, the T0ON/T1ON/T2ON can only be reset by instructions.

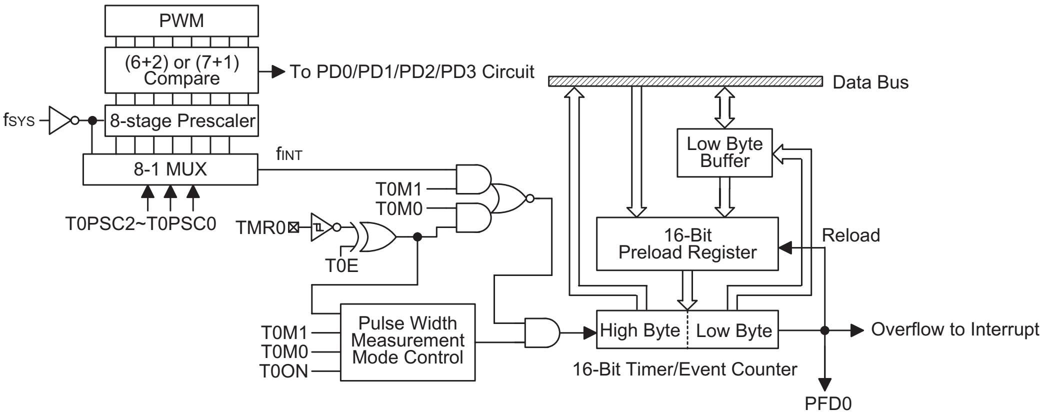
The overflow of the Timer/Event Counter 0/1/2 is one of the wake-up sources. The Timer/Event Counter 0/1 can also be applied to a PFD or Programmable Frequency Divider whose output is on pin PA3 via a configuration option. Only one PFD (PFD0 or PFD1) can be applied to PA3 by options. No matter what the operation mode is, writing a ″0″ to ET0I, ET1I or ET2I disables the related interrupt service. When the PFD function is selected, executing the ″SET [PA].3″ instruction will enable the PFD output and executing the ″CLR [PA].3″ instruction will disable the PFD output.

If the timer/event counter is not running, writing data to the timer/event counter preload register will also reload that data to the timer/event counter. But if the timer/event counter running, data written to the timer/event counter is kept only in the timer/event counter preload register. The timer/event counter continues to operate until an overflow occurs at which point the new data will be loaded from the preload register into the timer/event counter.

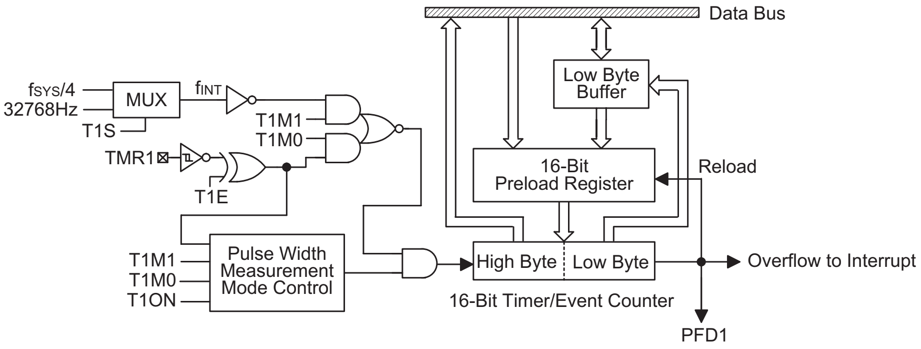
After the timer has been initialised the timer can be turned on and off by controlling the enable bit in the timer control register. Note that setting the timer enable bit high to turn the timer on, should only be executed after the timer mode bits have been properly setup. Setting the timer enable bit high together with a mode bit modification, may lead to improper timer operation if executed as a single timer control register byte write instruction.

When the timer/event counter is read, the clock is blocked to avoid errors. As this may results in a counting error, blocking of the clock should be taken into account by the programmer.

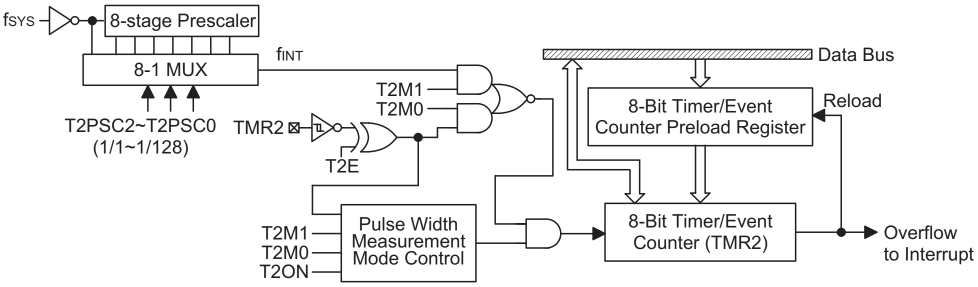
It is strongly recommended to load a desired value into the TMR0/TMR1/TMR2 registers first, before turning on the related timer/event counter, for proper operation since the initial value of the TMR0/TMR1/TMR2 registers are unknown. Due to the timer/event counter scheme, the programmer should pay special attention to the instruction to enable then disable the timer for the first time, whenever there is a need to use the timer/event counter function, to avoid unpredictable result. After this procedure, the timer/event counter function can be operated normally.



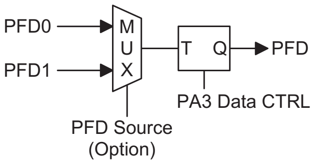
Timer/Event Counter 0



Timer/Event Counter 1



Timer/Event Counter 2



PFD Source Option

The bit0~bit2 of the TMR0C/TMR2C (T0PSC2~0/ T2PSC2~0) can be used to define the pre-scaling stages of the internal clock sources of the timer/event counter. The overflow signal of the timer/event counter can be used to generate the PFD signal. The timer prescaler is also used as the PWM counter.

Bit No.	Label	Function
0 1 2	T0PSC0 T0PSC1 T0PSC2	Defines 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 TMR0 active edge of the timer/event counter: In Event Counter Mode (T0M1,T0M0)=(0,1): 1:count on falling edge; 0:count on rising edge In Pulse Width measurement mode (T0M1,T0M0)=(1,1): 1: start counting on the rising edge, stop on the falling edge; 0: start counting on the falling edge, stop on the rising edge
4	T0ON	Enables/disables the timer counting (0=disable; 1=enable)
5	—	Unused bit, read as ″0″
6 7	T0M0 T0M1	Defines the operating mode, T0M1, T0M0: 01=Event count mode (external clock) 10=Timer mode (internal clock) 11=Pulse width measurement mode 00=Unused

TMR0C (0EH) Register

Bit No.	Label	Function
0~2	—	Unused bit, read as ″0″
3	T1E	Defines the TMR1 active edge of the timer/event counter: In Event Counter Mode (T1M1,T1M0)=(0,1): 1:count on falling edge; 0:count on rising edge In Pulse Width measurement mode (T1M1,T1M0)=(1,1): 1: start counting on the rising edge, stop on the falling edge; 0: start counting on the falling edge, stop on the rising edge
4	T1ON	Enables/disables the timer counting (0=disable; 1=enable)
5	T1S	Defines the TMR1 internal clock source. (0=f_{SYS}/4; 1=32768Hz)
6 7	T1M0 T1M1	Defines the operating mode, T1M1, T1M0: 01=Event count mode (external clock) 10=Timer mode (internal clock) 11=Pulse width measurement mode 00=Unused

TMR1C (11H) Register

Bit No.	Label	Function
0 1 2	T2PSC0 T2PSC1 T2PSC2	Defines the prescaler stages, T2PSC2, T2PSC1, T2PSC0= 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	T2E	Defines the TMR2 active edge of the timer/event counter: In Event Counter Mode (T2M1,T2M0)=(0,1): 1:count on falling edge; 0:count on rising edge In Pulse Width measurement mode (T2M1,T2M0)=(1,1): 1: start counting on the rising edge, stop on the falling edge; 0: start counting on the falling edge, stop on the rising edge
4	T2ON	Enables/disables the timer counting (0=disable; 1=enable)
5	—	Unused bit, read as ″0″
6 7	T2M0 T2M1	Defines the operating mode, T2M1, T2M0: 01=Event count mode (external clock) 10=Timer mode (internal clock) 11=Pulse width measurement mode 00=Unused

TMR2C (2EH) Register

Input/Output Ports

There are 32 bidirectional input/output lines in the device, labeled as PA, PB, PC and PD, which are mapped to the data memory of [12H], [14H], [16H] and [18H] respectively. 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]″ (m=12H, 14H, 16H or 18H). 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 (PAC, PBC, PCC, PDC) to control the input/output configuration. With this control register, a CMOS output or a Schmitt Trigger input 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 setup as 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. If the control register bit is ″0″, the contents of the latches will move to the internal bus. The latter is possible in the ″read-modify-write″ instruction.

For an output function, CMOS is the only configuration. These control registers are mapped to locations 13H, 15H, 17H and 19H.

After a device reset, these I/O lines will default to an input state and will be either high or floating, depending upon the pull-high configuration options. Each bit of these input/output latches can be set or cleared by the ″SET [m].i″ and ″CLR [m].i″ bit manipulation instructions.

Some instructions first input data and then follow the output operations. For example, ″SET [m].i″, ″CLR [m].i″, ″CPL [m]″, ″CPLA [m]″ read the entire port states into the CPU, execute the defined bit operations, 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 the pull-high option is selected, the I/O port has a pull-high resistor, otherwise, there′s none. Take note that a non-pull-high I/O port operating in input mode will cause a floating state.

PA0, PA1, PA3, PD4, PD5, PD6 and PD7 are pin-shared with BZ, $\overline{BZ}$, PFD, $\overline{INT0}$, $\overline{INT1}$, TMR0 and TMR1 pins respectively. The PC0, PC6 and PC7 pins are pin-shared with TMR2, TX and RX.

PA0 and PA1 are pin-shared with the BZ and $\overline{BZ}$ signals, respectively. If the BZ/$\overline{BZ}$ option is selected, the output signal in output mode of PA0/PA1 will be the buzzer signal generated by the Multi-function timer. The input mode always remain in its original functions. Once the BZ/$\overline{BZ}$ option is selected, the buzzer output signals are controlled by the PA0 data register only. The I/O functions of PA0/PA1 are shown below.

PAC Register PAC0	PAC Register PAC1	PA Data Register PA0	PA Data Register PA1	Output Function
0	0	1	x	PA0=BZ, PA1=$\overline{BZ}$
0	0	0	x	PA0=0, PA1=0
0	1	1	x	PA0=BZ, PA1=input
0	1	0	x	PA0=0, PA1=input
1	0	0	x	PA0=input, PA1=0
1	1	x	x	PA0=input, PA1=input

Note: ″x″ stands for don′t care
″D″ stands for Data ″0″ or ″1″

PA3 is pin-shared with the signal. If the PFD option is selected and if PA3 is setup as an output, then the output signal on the PA3 pin will be the PFD signal, generated by the timer/event counter overflow signal. If setup as an input it will function as a normal input pin. Once the PFD option is selected, the PFD output signal is controlled by the PA3 data register only. Writing a ″1″ to the PA3 data register will enable the PFD output function while 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.

Port PB can also be used as A/D converter inputs. There is a PWM function shared with PD0/PD1/PD2/PD3. If the PWM function is enabled, the PWM0/PWM1/PWM2/ PWM3 signal will appear on PD0/PD1/PD2/PD3, if PD0/ PD1/ PD2/PD3 are operating in output mode. Writing ″1″ to the PD0~PD3 data register will enable the PWM output function while writing ″0″ will force the PD0~PD3 to remain at ″0″. The I/O functions of the PD0/PD1/PD2/PD3 are shown below.

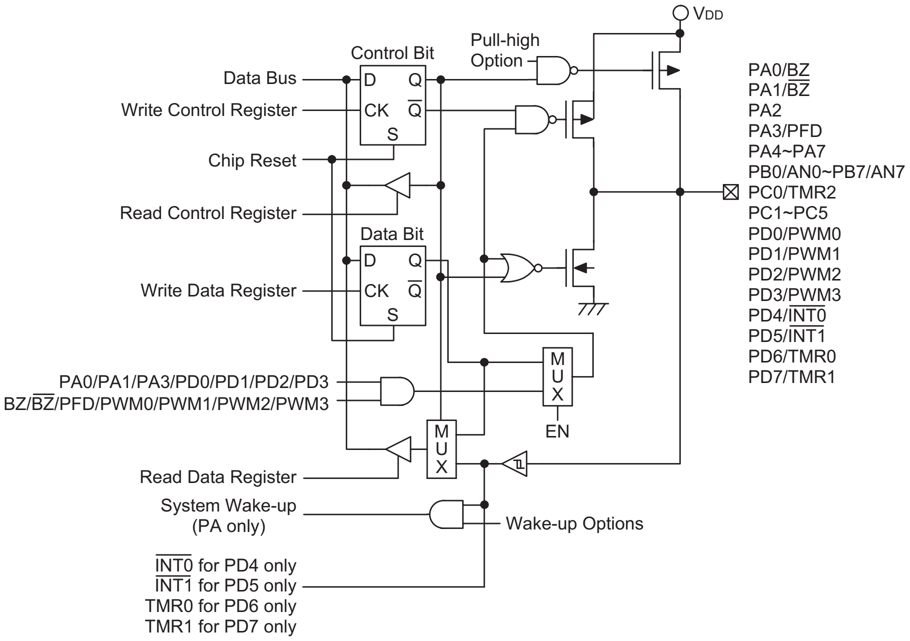
I/O Mode	I/P (Normal)	O/P (Normal)	I/P (PWM)	O/P (PWM)
PD0~PD3	Logical Input	Logical Output	Logical Input	PWM0~PWM3

It is recommended that unused or not bonded out I/O lines should be set as output pins by software instruction to avoid consuming power under input floating state.

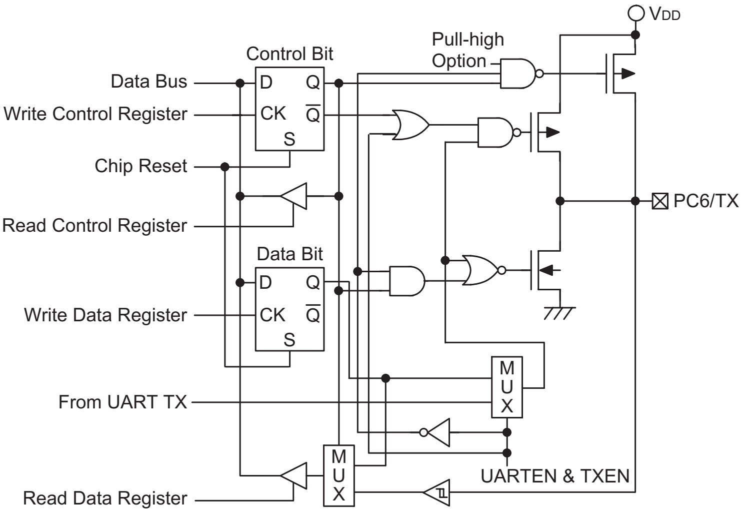
The definitions of the PFD control signal and PFD output frequency are listed in the following table.

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

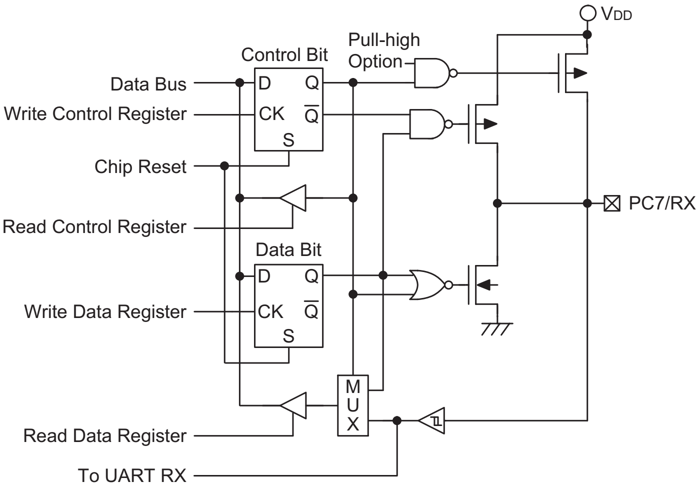
Note: ″X″ stands for unused
″U″ stands for unknown
″M″ is ″65536″ for PFD0 or PFD1
″N″ is the preload value for the timer/event counter
″f_{TMR}″ is input clock frequency for timer/event counter



Input/Output Ports



PC6/TX Input/Output Ports



PC7/RX Input/Output Ports

Pulse Width Modulator

Each devices is provided with either three or four Pulse Width Modulation (PWM) outputs, depending upon which package type is selected. Useful for such applications such as motor speed control, the PWM function provides outputs with a fixed frequency but with a duty cycle that can be varied by setting particular values into the corresponding PWM register.

A single register, located in the Data Memory is assigned to each PWM output. For devices with three PWM outputs, these registers are known as PWM0, PWM1 and PWM2. Devices with four PWM outputs require a further additional register known as PWM3. It is here that the 8-bit value, which represents the overall duty cycle of one modulation cycle of the output waveform, should be placed. To increase the PWM modulation frequency, each modulation cycle is subdivided into two or four individual modulation subsections, known as the 7+1 mode or 6+2 mode respectively. The device can choose which mode to use by selecting the appropriate configuration option. When a mode configuration option is chosen, it applies to all PWM outputs on that device. Note that when using the PWM, it is only necessary to write the required value into the appropriate PWM register and select the required mode configuration option, the subdivision of the waveform into its sub-modulation cycles is done automatically within the microcontroller hardware.

For all devices, the PWM clock source is the system clock f_{SYS}.

Package	Channels	PWM Mode	Output Pin	PWM Register Name
52/56-pin	3	6+2 or 7+1	PD0/PD1/PD2	PWM0/PWM1/PWM2
100-pin	4	6+2 or 7+1	PD0/PD1/PD2/PD3	PWM0/PWM1/PWM2/PWM3

PWM Function Table

This method of dividing the original modulation cycle into a further 2 or 4 sub-cycles enable the generation of higher PWM frequencies which allow a wider range of applications to be served. As long as the periods of the generated PWM pulses are less than the time constants of the load, the PWM output will be suitable as such long time constant loads will average out the pulses of the PWM output. The difference between what is known as the PWM cycle frequency and the PWM modulation frequency should be understood. As the PWM clock is the system clock, f_{SYS}, and as the PWM value is 8-bits wide, the overall PWM cycle frequency is f_{SYS}/256. However, when in the 7+1 mode of operation the PWM modulation frequency will be f_{SYS}/128, while the PWM modulation frequency for the 6+2 mode of operation will be f_{SYS}/64.

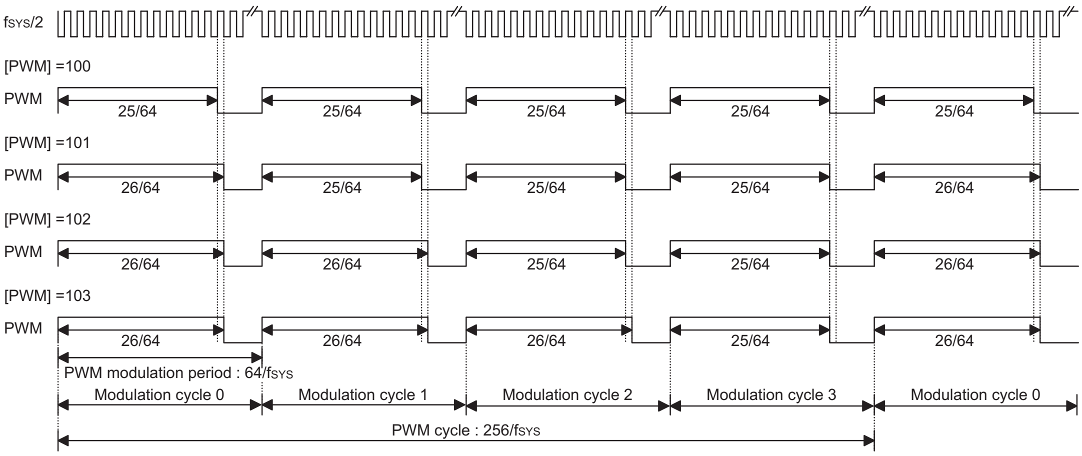
PWM Modulation Frequency	PWM Cycle Frequency	PWM Cycle Duty
f_{SYS}/64 for (6+2) bits mode f_{SYS}/128 for (7+1) bits mode	f_{SYS}/256	[PWM]/256

- 6+2 PWM Mode
  Each full PWM cycle, as it is controlled by an 8-bit PWM register, has 256 clock periods. However, in the 6+2 PWM mode, each PWM cycle is subdivided into four individual sub-cycles known as modulation cycle 0 ~ modulation cycle 3, denoted as "i" in the table. Each one of these four sub-cycles contains 64 clock cycles. In this mode, a modulation frequency increase of four is achieved. The 8-bit PWM register value, which represents the overall duty cycle of the PWM waveform, is divided into two groups. The first group which consists of bit2~bit7 is denoted here as the DC value. The second group which consists of bit0~bit1 is known as the AC value. In the 6+2 PWM mode, the duty cycle value of each of the four modulation sub-cycles is shown in the following table.

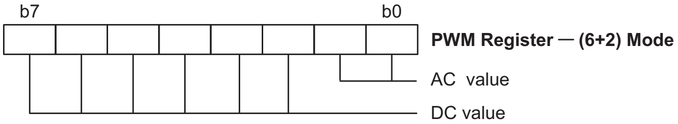
Parameter	AC (0~3)	Duty Cycle
Modulation cycle i (i=0~3)	i<AC	$\frac{DC+1}{64}$
	i≥AC	$\frac{DC}{64}$

6+2 Mode Modulation Cycle Values

The following diagram illustrates the waveforms associated with the 6+2 mode PWM operation. It is important to note how the single PWM cycle is subdivided into 4 individual modulation cycles, numbered from 0~3 and how the AC value is related to the PWM value.



6+2 PWM Mode



PWM Register for 6+2 Mode

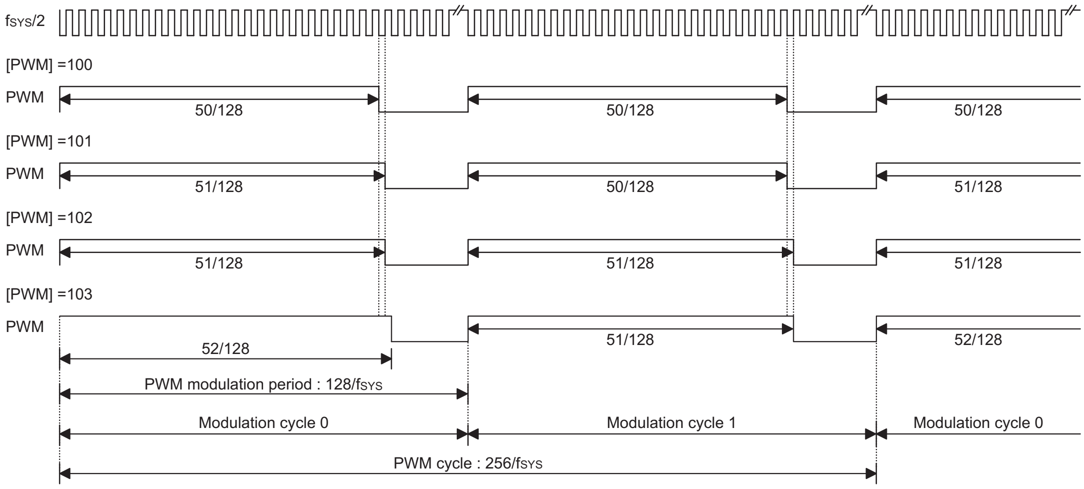
- 7+1 PWM Mode

  Each full PWM cycle, as it is controlled by an 8-bit PWM register, has 256 clock periods. However, in the 7+1 PWM mode, each PWM cycle is subdivided into two individual sub-cycles known as modulation cycle 0 ~ modulation cycle 1, denoted as ″i″ in the table. Each one of these two sub-cycles contains 128 clock cycles. In this mode, a modulation frequency increase of two is achieved. The 8-bit PWM register value, which represents the overall duty cycle of the PWM waveform, is divided into two groups. The first group which consists of bit1~bit7 is denoted here as the DC value. The second group which consists of bit0 is known as the AC value. In the 7+1 PWM mode, the duty cycle value of each of the two modulation sub-cycles is shown in the following table.

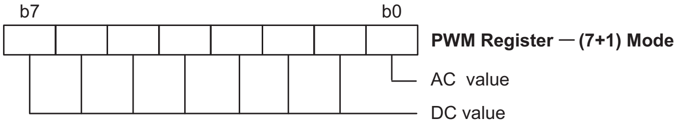
Parameter	AC (0~1)	Duty Cycle
Modulation cycle i (i=0~1)	i<AC	$\frac{DC+1}{128}$
	i≥AC	$\frac{DC}{128}$

7+1 Mode Modulation Cycle Values

The following diagram illustrates the waveforms associated with the 7+1 mode PWM operation. It is important to note how the single PWM cycle is subdivided into 2 individual modulation cycles, numbered 0 and 1 and how the AC value is related to the PWM value.



(7+1) PWM Mode



PWM Register for 7+1 Mode

- PWM Output Control

  On all devices, the PWM outputs are pin-shared with the Port D I/O pins. To operate as PWM outputs and not as I/O pins, the correct PWM configuration options must be selected. A ″0″ must also be written to the corresponding bits in the I/O port control register PDC to ensure that the required PWM output pins are setup as outputs. After these two initial steps have been carried out, and of course after the required PWM value has been written into the PWM register, writing a ″1″ to the corresponding bit in the PD output data register will enable the PWM data to appear on the pin. Writing a ″0″ to the corresponding bit in the PD output data register will disable the PWM output function and force the output low. In this way, the Port D data output register can be used as an on/off control for the PWM function. Note that if the configuration options have selected the PWM function, but a ″1″ has been written to its corresponding bit in the PDC control register to configure the pin as an input, then the pin can still function as a normal input line, with pull-high resistor options.

The following sample program shows how the PWM outputs are setup and controlled, the corresponding PWM output configuration option must first be selected.

```
clr PDC.0               ; set pin PD0 as output
clr PDC.1               ; set pin PD1 as output
clr PDC.2               ; set pin PD2 as output
clr PDC.3               ; set pin PD3 as output

set pd.0                ; PD.0=1; enable pin ″PD0/PWM0″ to be the PWM channel 0
mov a,64h               ; PWM0=100D=64H
mov pwm0,a

set pd.1                ; PD.1=1; enable pin ″PD1/PWM1″ to be the PWM channel 1
mov a,65h               ; PWM1=101D=65H
mov pwm1,a

set pd.2                ; PD.2=1; enable pin ″PD2/PWM2″ to be the PWM channel 2
mov a,66h               ; PWM2=102D=66H
mov pwm2,a

set pd.3                ; PD.3=1; enable pin ″PD3/PWM3″ to be the PWM channel 3
mov a,67h               ; PWM3=103D=67H
mov pwm3,a

clr pd.0                ; disable PWM0 output – PD.0 will remain low
clr pd.1                ; disable PWM1 output – PD.1 will remain low
clr pd.2                ; disable PWM2 output – PD.2 will remain low
clr pd.3                ; disable PWM3 output – PD.3 will remain low
```

A/D Converter

An eight channel and 12 bits resolution A/D converter is implemented in the microcontroller. The reference voltage is VDD. The A/D converter contains four special registers which are; ADRL (24H), ADRH (25H), ADCR (26H) and ACSR (27H). The ADRH and ADRL registers are the A/D result register higher-order byte and lower-order byte and are read-only. After the A/D conversion is completed, the ADRH and ADRL should be read to get the conversion result data. The ADCR is an A/D converter control register, which defines the A/D channel number, analog channel select, start A/D conversion control bit and the end of A/D conversion flag. To start an A/D conversion, the PB configuration must first be defined, the analog channel selected, after which the START bit can supply a rising and falling edge (0→1→0). At the end of A/D conversion, the EOCB bit is cleared. The ACSR register is the A/D clock setting register, which is used to select the A/D clock source.

The A/D converter control register is used to control the A/D converter. Bit2~bit0 are used to select an analog input channel. There are a total of eight channels to select. Bit5~bit3 of the ADCR are used to set the PB configurations. PB can be an analog input or setup as a normal I/O line, the selected function is determined by these 3 bits. Once a PB line is selected as an analog input, the I/O function and pull-high resistor of this I/O line are disabled and the A/D converter circuit is powered-on. The EOCB bit, bit6 of the ADCR is end of A/D conversion flag. This bit can be monitored to know when the A/D conversion has completed. The START bit in the ADCR register is used to start the conversion process of the A/D converter. Giving the START bit a rising edge and falling edge means that the A/D conversion has started. In order to ensure that the A/D conversion is completed, the START bit should remain at ″0″ until the EOCB flag is cleared to ″0″ which indicates the end of the A/D conversion.

Bit 7 of the ACSR register is used for test purposes only and must not be used for other purposes by the application program. Bit1 and bit0 of the ACSR register are used to select the A/D clock source.

The EOCB bit is set to ″1″ when the START bit is set from ″0″ to ″1″.

Important Note for A/D initialisation:
Special care must be taken to initialise the A/D converter each time the Port B A/D channel selection bits are modified, otherwise the EOCB flag may be in an undefined condition. An A/D initialisation is implemented by setting the START bit high and then clearing it to zero within 10 instruction cycles of the Port B channel selection bits being modified. Note that if the Port B channel selection bits are all cleared to zero then an A/D initialisation is not required.

Bit No.	Label	Function
0 1	ADCS0 ADCS1	Selects the A/D converter clock source 00=system clock/2 01=system clock/8 10=system clock/32 11=undefined
2~6	—	Unused bit, read as ″0″
7	TEST	For test mode used only

ACSR (27H) Register

Bit No.	Label	Function
0 1 2	ACS0 ACS1 ACS2	ACS2, ACS1, ACS0: Select A/D channel 0, 0, 0: AN0 0, 0, 1: AN1 0, 1, 0: AN2 0, 1, 1: AN3 1, 0, 0: AN4 1, 0, 1: AN5 1, 1, 0: AN6 1, 1, 1: AN7
3 4 5	PCR0 PCR1 PCR2	Defines the Port B configuration select. If PCR0, PCR1 and PCR2 are all zero, the ADC circuit is powered off to reduce power consumption.
6	EOCB	Indicates end of A/D conversion. (0 = end of A/D conversion) Each time bits 3~5 change state the A/D should be initialised by issuing a START signal, otherwise the EOCB flag may have an undefined condition. See ″Important note for A/D initialisation″.
7	START	Starts the A/D conversion. (0→1→0= start; 0→1= Reset A/D converter and set EOCB to ″1″)

ADCR (26H) Register

PCR2	PCR1	PCR0	7	6	5	4	3	2	1	0
0	0	0	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
0	0	1	PB7	PB6	PB5	PB4	PB3	PB2	PB1	AN0
0	1	0	PB7	PB6	PB5	PB4	PB3	PB2	AN1	AN0
0	1	1	PB7	PB6	PB5	PB4	PB3	AN2	AN1	AN0
1	0	0	PB7	PB6	PB5	PB4	AN3	AN2	AN1	AN0
1	0	1	PB7	PB6	PB5	AN4	AN3	AN2	AN1	AN0
1	1	0	PB7	PB6	AN5	AN4	AN3	AN2	AN1	AN0
1	1	1	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0

Port B Configuration

Register	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADRL	D3	D2	D1	D0	—	—	—	—
ADRH	D11	D10	D9	D8	D7	D6	D5	D4

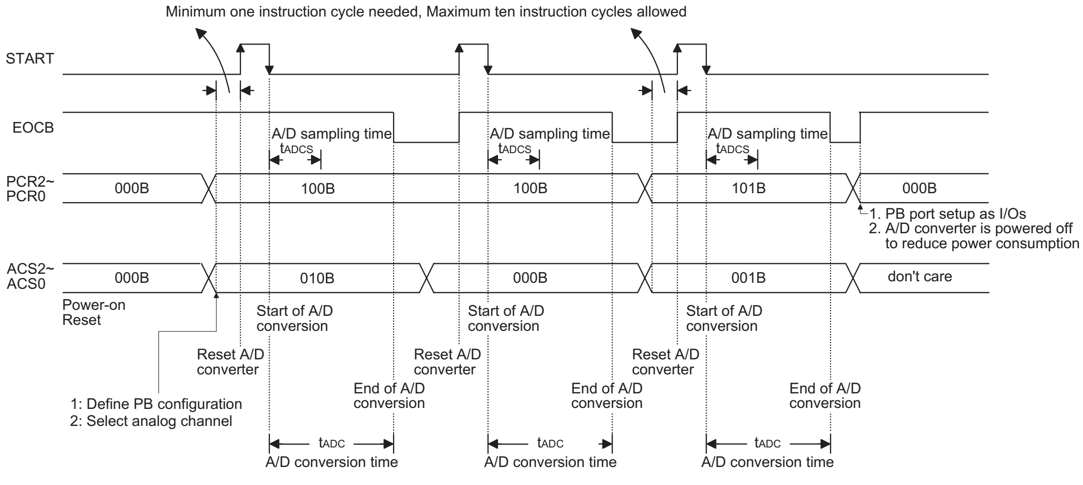
Note: D0~D11 is A/D conversion result data bit LSB~MSB.

ADRL (24H), ADRH (25H) Register

The following programming example illustrates how to setup and implement an A/D conversion. The method of polling the EOCB bit in the ADCR register is used to detect when the conversion cycle is complete.

Example: using EOCB Polling Method to detect end of conversion

```
    mov   a,00000001B
    mov   ACSR,a               ; setup the ACSR register to select fSYS/8 as the A/D clock
    mov   a,00100000B          ; setup ADCR register to configure Port PB0~PB3 as A/D inputs
    mov   ADCR,a               ; and select AN0 to be connected to the A/D converter
          :
          :                    ; As the Port B channel bits have changed the following START
                               ; signal (0-1-0) must be issued within 10 instruction cycles
          :
Start_conversion:
    clr   START
    set   START                ; reset A/D
    clr   START                ; start A/D
Polling_EOC:
    sz    EOCB                 ; poll the ADCR register EOCB bit to detect end of A/D conversion
    jmp   polling_EOC          ; continue polling
    mov   a,ADRH               ; read conversion result high byte value from the ADRH register
    mov   adrh_buffer,a        ; save result to user defined memory
    mov   a,ADRL               ; read conversion result low byte value from the ADRL register
    mov   adrl_buffer,a        ; save result to user defined memory
          :
          :
    jmp   start_conversion     ; start next A/D conversion
```

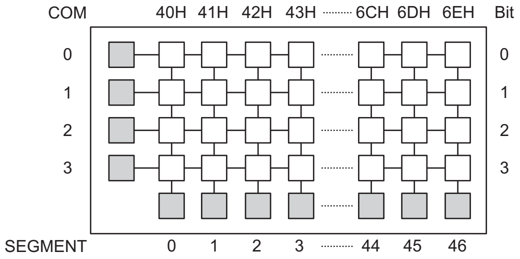


Note: A/D clock must be $f_{SYS}/2$, $f_{SYS}/8$ or $f_{SYS}/32$
$t_{ADCS}=32t_{AD}$
$t_{ADC}=80t_{AD}$

A/D Conversion Timing

LCD Memory

The device provides an area of embedded data memory for LCD display. This area is located from 40H to 6EH of the RAM at Bank 1. Bank pointer (BP; located at 04H of the RAM) is the switch between the RAM and the LCD display memory. When the BP is set as ″1″, any data written into 40H~6EH will affect the LCD display. When the BP is cleared to ″0″, ″2″ or ″3″, any data written into 40H~6EH is map into the general purpose data memory. The LCD display memory can be read and written to only by indirect addressing mode 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.



Display Memory

LCD Driver Output

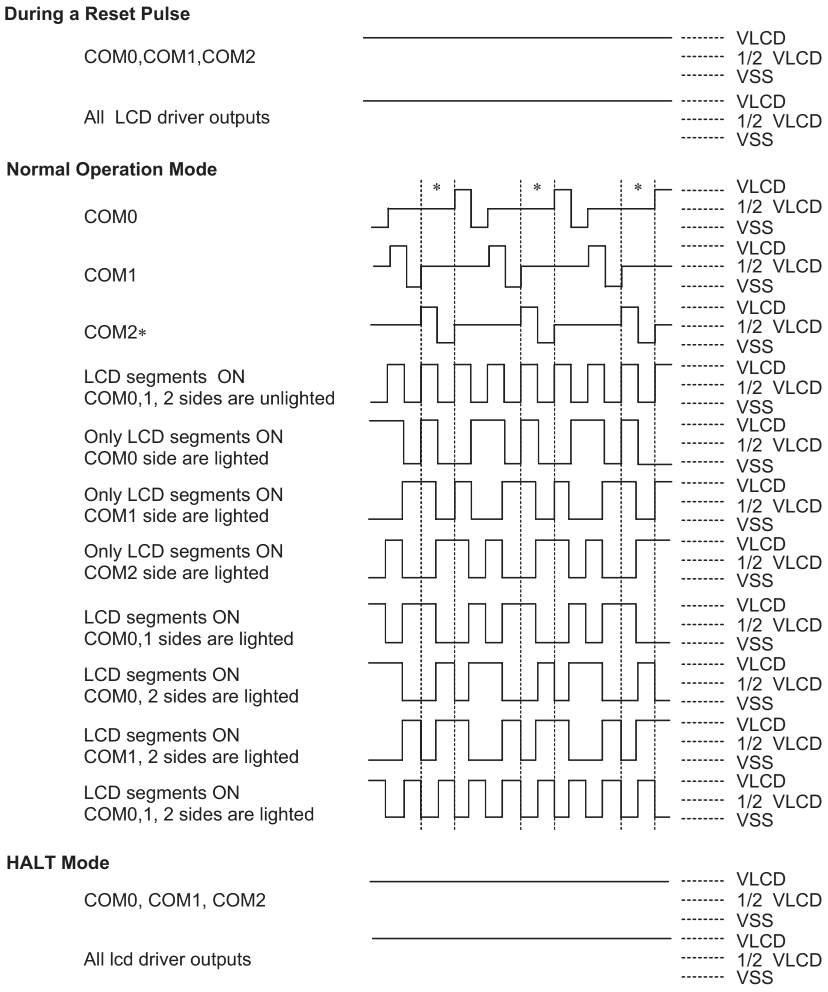
The output number of the device LCD driver can be 47×2 or 47×3 or 46×4 by option (i.e., 1/2 duty, 1/3 duty or 1/4 duty). The bias type LCD driver can be ″R″ type or ″C″ type. If the ″R″ bias type is selected, no external capacitor is required. If the ″C″ bias type is selected, a capacitor mounted between 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.

LCD Segments as Logical Output

The SEG0~SEG23 also can be optioned as logical output, once an LCD segment is optioned as a logical output, the contents of bit0 of the related segment address in the LCD RAM will appear on the segment.

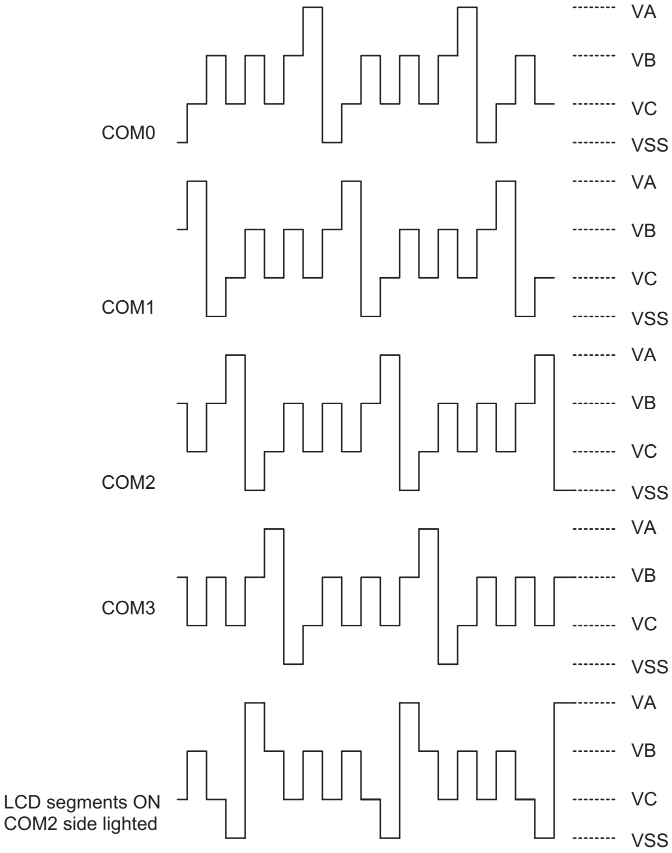
SEG0~SEG7 are all byte optioned as logical outputs, SEG8~SEG15 are also byte optioned as logical outputs, SEG16~SEG23 are individually bit optioned as logical outputs.

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



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

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



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

LCD Driver Output

Low Voltage Reset/Detector Functions

There is a low voltage detector (LVD) and a low voltage reset circuit (LVR) implemented in this microcontroller. These two functions can be enabled/disabled by options. Once the LVD option is enabled, the user can use the RTCC.3 to enable/disable (1/0) the LVD circuit and read the LVD detector status (0/1) from RTCC.5, otherwise, the LVD function is disabled.

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 oscillating 0/1: quick/slow start
5	LVDO	LVD detection output (1/0) 1: low voltage detected, read only
6, 7	—	Unused bit, read as ″0″

Note: ″*″ Once the LVD function is enabled the reference generator should be enabled; otherwise the reference generator is controlled by LVR ROM code option. The relationship between LVR and LVD options and LVDC are as shown.

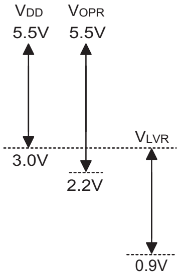
RTCC (09H) Register

The LVR has the same effect or function with the external RES signal which performs a chip reset. During HALT state, both LVR and LVD are disabled.

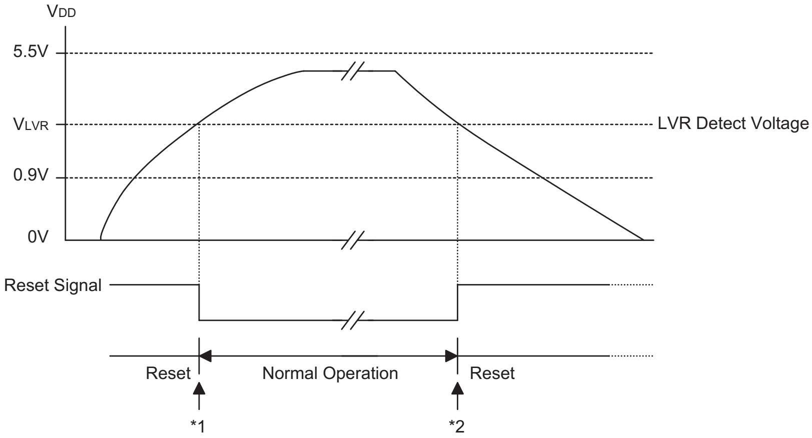
The LVR state requires the following specifications:

- The low voltage (0.9V~V_{LVR}) has to be maintained for more than 1ms, otherwise, the circuits remain in their original state. If the low voltage state does not exceed 1ms, the LVR will ignore it and do not perform a reset function.
- The LVR uses the ″OR″ function with the external $\overline{\text{RES}}$ signal to perform a chip reset.

The relationship between V_{DD} and V_{LVR} is shown below.



Note: V_{OPR} is the voltage range for proper chip operation at 4MHz system clock.



Low Voltage Reset

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

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

UART Bus Serial Interface

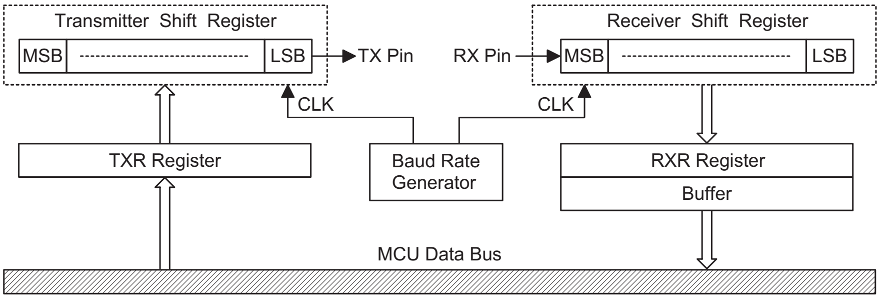
The HT46RU66/HT46CU66 devices contain an integrated full-duplex asynchronous serial communications UART interface that enables communication with external devices that contain a serial interface. The UART function has many features and can transmit and receive data serially by transferring a frame of data with eight or nine data bits per transmission as well as being able to detect errors when the data is overwritten or incorrectly framed. The UART function possesses its own internal interrupt which can be used to indicate when a reception occurs or when a transmission terminates.

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

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

- UART data transfer scheme
  The block diagram shows the overall data transfer structure arrangement for the UART. The actual data to be transmitted from the MCU is first transferred to the TXR register by the application program. The data will then be transferred to the Transmit Shift Register from where it will be shifted out, LSB first, onto the TX pin at a rate controlled by the Baud Rate Generator. Only the TXR register is mapped onto the MCU Data Memory, the Transmit Shift Register is not mapped and is therefore inaccessible to the application program.
  Data to be received by the UART is accepted on the external RX pin, from where it is shifted in, LSB first, to the Receiver Shift Register at a rate controlled by the Baud Rate Generator. When the shift register is full, the data will then be transferred from the shift register to the internal RXR register, where it is buffered and can be manipulated by the application program. Only the RXR register is mapped onto the MCU Data Memory, the Receiver Shift Register is not mapped and is therefore inaccessible to the application program.
  It should be noted that the actual register for data transmission and reception, although referred to in the text, and in application programs, as separate TXR and RXR registers, only exists as a single shared register in the Data Memory. This shared register known as the TXR/RXR register is used for both data transmission and data reception.

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



UART Data Transfer Scheme

- USR register

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

Further explanation on each of the flags is given below:

- ♦ TXIF

The TXIF flag is the transmit data register empty flag. When this read only flag is ″0″ it indicates that the character is not transferred to the transmit shift registers. When the flag is ″1″ it indicates that the transmit shift register has received a character from the TXR data register. The TXIF flag is cleared by reading the UART status register (USR) with TXIF set and then writing to the TXR data register. Note that when the TXEN bit is set, the TXIF flag bit will also be set since the transmit buffer is not yet full.

- ♦ TIDLE

The TIDLE flag is known as the transmission complete flag. When this read only flag is ″0″ it indicates that a transmission is in progress. This flag will be set to ″1″ when the TXIF flag is ″1″ and when there is no transmit data, or break character being transmitted. When TIDLE is ″1″ the TX pin becomes idle. The TIDLE flag is cleared by reading the USR register with TIDLE set and then writing to the TXR register. The flag is not generated when a data character, or a break is queued and ready to be sent.

- ♦ RXIF

The RXIF flag is the receive register status flag. When this read only flag is ″0″ it indicates that the RXR read data register is empty. When the flag is ″1″ it indicates that the RXR read data register contains new data. When the contents of the shift register are transferred to the RXR register, an interrupt is generated if RIE=1 in the UCR2 register. If one or more errors are detected in the received word, the appropriate receive-related flags NF, FERR, and/or PERR are set within the same clock cycle. The RXIF flag is cleared when the USR register is read with RXIF set, followed by a read from the RXR register, and if the RXR register has no data available.

- ♦ RIDLE

The RIDLE flag is the receiver status flag. When this read only flag is ″0″ it indicates that the receiver is between the initial detection of the start bit and the completion of the stop bit. When the flag is ″1″ it indicates that the receiver is idle. Between the completion of the stop bit and the detection of the next start bit, the RIDLE bit is ″1″ indicating that the UART is idle.

- ♦ OERR

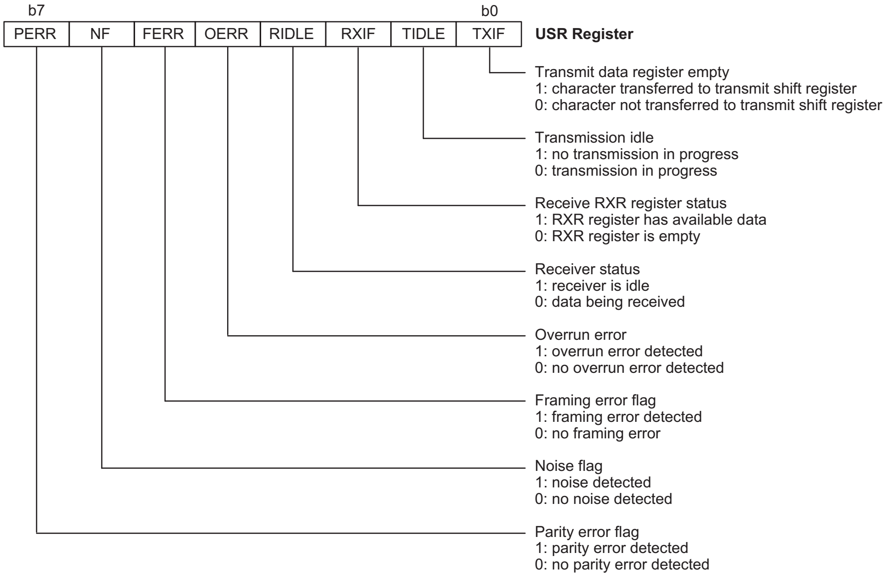
The OERR flag is the overrun error flag, which indicates when the receiver buffer has overflowed. When this read only flag is ″0″ there is no overrun error. When the flag is ″1″ an overrun error occurs which will inhibit further transfers to the RXR receive data register. The flag is cleared by a software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

- ♦ FERR

The FERR flag is the framing error flag. When this read only flag is ″0″ it indicates no framing error. When the flag is ″1″ it indicates that a framing error has been detected for the current character. The flag can also be cleared by a software sequence which will involve a read to the USR status register followed by an access to the RXR data register.

- ♦ NF

The NF flag is the noise flag. When this read only flag is ″0″ it indicates a no noise condition. When the flag is ″1″ it indicates that the UART has detected noise on the receiver input. The NF flag is set during the same cycle as the RXIF flag but will not be set in the case of an overrun. The NF flag can be cleared by a software sequence which will involve a read to the USR status register, followed by an access to the RXR data register.



b7							b0	
PERR	NF	FERR	OERR	RIDLE	RXIF	TIDLE	TXIF	**USR Register**

Transmit data register empty
1: character transferred to transmit shift register
0: character not transferred to transmit shift register

Transmission idle
1: no transmission in progress
0: transmission in progress

Receive RXR register status
1: RXR register has available data
0: RXR register is empty

Receiver status
1: receiver is idle
0: data being received

Overrun error
1: overrun error detected
0: no overrun error detected

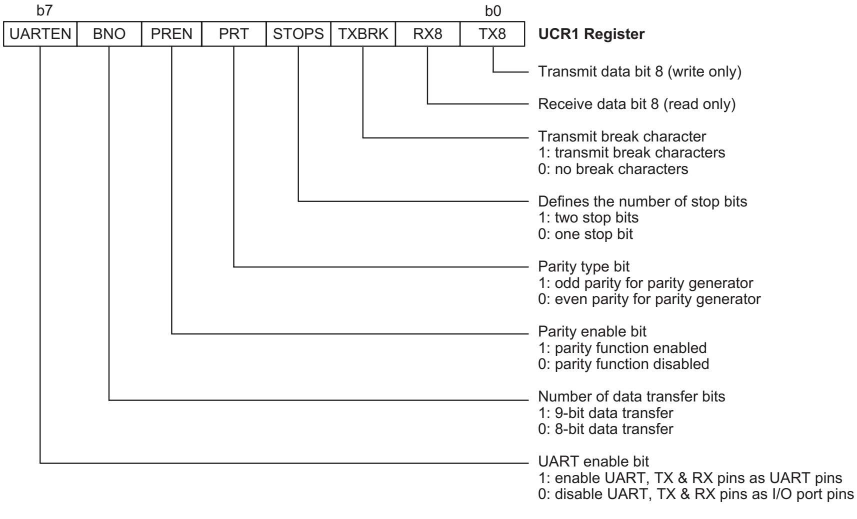
Framing error flag
1: framing error detected
0: no framing error

Noise flag
1: noise detected
0: no noise detected

Parity error flag
1: parity error detected
0: no parity error detected

- 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.


b7
b0
UARTEN
BNO
PREN
PRT
STOPS
TXBRK
RX8
TX8
UCR1 Register
Transmit data bit 8 (write only)
Receive data bit 8 (read only)
Transmit break character
1: transmit break characters
0: no break characters
Defines the number of stop bits
1: two stop bits
0: one stop bit
Parity type bit
1: odd parity for parity generator
0: even parity for parity generator
Parity enable bit
1: parity function enabled
0: parity function disabled
Number of data transfer bits
1: 9-bit data transfer
0: 8-bit data transfer
UART enable bit
1: enable UART, TX & RX pins as UART pins
0: disable UART, TX & RX pins as I/O port pins

- 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

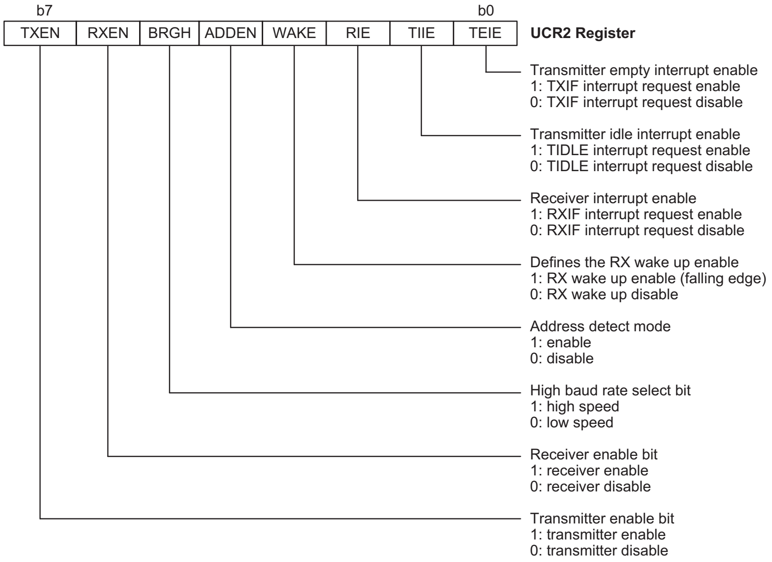
  The ADDEN bit is the address detect mode bit. When this bit is ″1″ the address detect mode is enabled. When this occurs, if the 8th bit, which corresponds to RX7 if BNO=0, or the 9th bit, which corresponds to RX8 if BNO=1, has a value of ″1″ then the received word will be identified as an address, rather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


b7
b0
TXEN
RXEN
BRGH
ADDEN
WAKE
RIE
TIIE
TEIE
UCR2 Register
Transmitter empty interrupt enable
1: TXIF interrupt request enable
0: TXIF interrupt request disable
Transmitter idle interrupt enable
1: TIDLE interrupt request enable
0: TIDLE interrupt request disable
Receiver interrupt enable
1: RXIF interrupt request enable
0: RXIF interrupt request disable
Defines the RX wake up enable
1: RX wake up enable (falling edge)
0: RX wake up disable
Address detect mode
1: enable
0: disable
High baud rate select bit
1: high speed
0: low speed
Receiver enable bit
1: receiver enable
0: receiver disable
Transmitter enable bit
1: transmitter enable
0: transmitter disable

- 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:

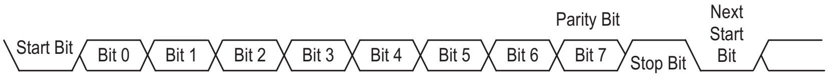
1. A USR register access
2. A TXR register write execution

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.

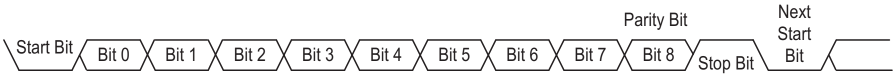
During a data transmission, a write instruction to the TXR register will place the data into the TXR register, which will be copied to the shift register at the end of the present transmission. When there is no data transmission in progress, a write instruction to the TXR register will place the data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

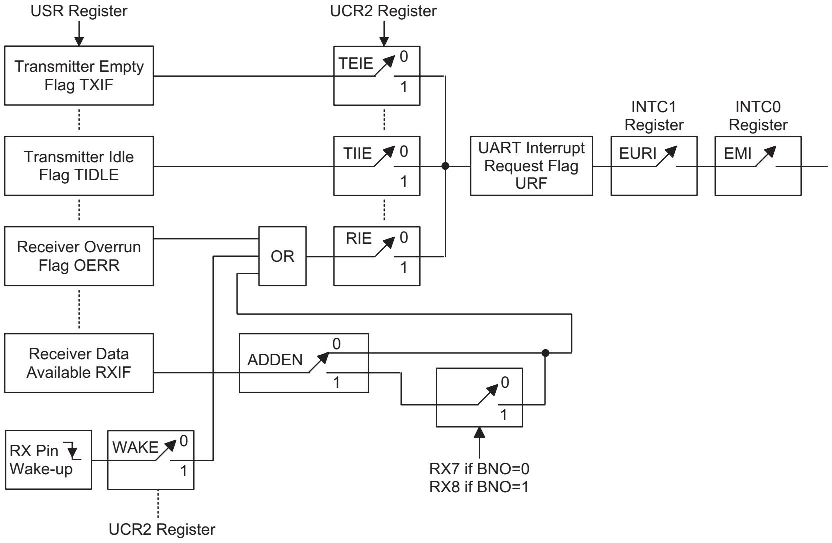
    The read only framing error flag, FERR, in the USR register, is set if a zero is detected instead of stop bits. If two stop bits are selected, both stop bits must be high, otherwise the FERR flag will be set. The FERR flag is buffered along with the received data and is cleared on any reset.

  - 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.

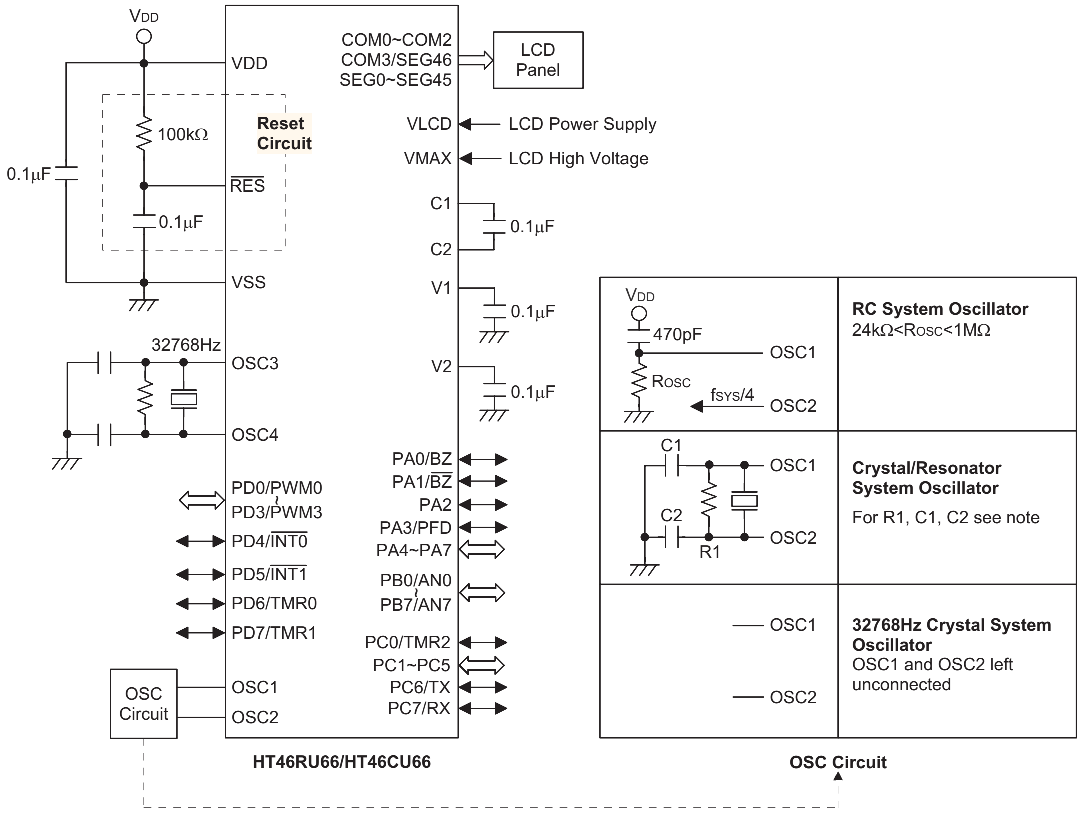
Options

The following shows the options in the device. All these options should be defined in order to ensure having a proper functioning system.

Options
OSC type selection. This option is to determine if an RC or crystal or 32768Hz crystal oscillator is chosen as system clock.
WDT, RTC and time base clock source selection. There are three types of selections: system clock/4 or RTC OSC or WDT OSC.

Options
WDT enable/disable selection. WDT can be enabled or disabled by option.
WDT time-out period selection. There are four types of selection: WDT clock source divided by $2^{12}/f_S$~$2^{13}/f_S$, $2^{13}/f_S$~$2^{14}/f_S$, $2^{14}/f_S$~$2^{15}/f_S$ or $2^{15}/f_S$~$2^{16}/f_S$.
CLR WDT times selection. This option defines the method to clear the WDT by instruction. ″One time″ means that the ″CLR WDT″ can clear the WDT. ″Two times″ means only if both of the ″CLR WDT1″ and ″CLR WDT2″ have been executed, only then can the WDT be cleared.
Time Base time-out period selection. The Time Base time-out period ranges from clock/2^{12} to clock/2^{15}. ″clock″ means the clock source selected by options.
Buzzer output frequency selection. There are eight types of frequency signals for the buzzer output: clock/2^2 ~ clock/2^9. ″clock″ means the clock source selected by options.
Wake-up selection. This option defines the wake-up capability. External I/O pins (PA only) all have the capability to wake-up the chip from a HALT by a falling edge (bit option).
Pull-high selection. This option is to determine whether the pull-high resistance is viable or not in the input mode of the I/O ports. PA, PB, PC and PD can be independently selected (bit option).
I/O pins share with other function selections. PA0/$\overline{\text{BZ}}$, PA1/BZ: PA0 and PA1 can be set as I/O pins or buzzer outputs. PA3/PFD: PA3 can be set as I/O pins or PFD output.
LCD common selection. There are three types of selections: 2 common (1/2 duty) or 3 common (1/3 duty) or 4 common (1/4 duty). If the 4 common is selected, the segment output pin ″SEG46″ will be set as a common output.
LCD bias power supply selection. There are two types of selections: 1/2 bias or 1/3 bias
LCD bias type selection. This option is to determine what kind of bias is selected, R type or C type (Low or high bias current option).
LCD driver clock frequency selection. There are seven types of frequency signals for the LCD driver circuits: $f_S/2^2$~$f_S/2^8$. ″f_S″ stands for the clock source selection by options.
LCD ON/OFF at HALT selection.
LCD Segments as logical output selection, (byte, byte, bit, bit, bit, bit, bit, bit, bit, bit option) [SEG0~SEG7], [SEG8~SEG15], SEG16, SEG17, SEG18, SEG19, SEG20, SEG21, SEG22, or SEG23
LVR selection. LVR has enable or disable options
LVD selection. LVD has enable or disable options
PFD selection. If PA3 is set as PFD output, there are two types of selections; One is PFD0 as the PFD output, the other is PFD1 as the PFD output. PFD0, PFD1 are the timer overflow signals of the Timer/Event Counter 0, Timer/Event Counter 1 respectively.
PWM selection: (7+1) or (6+2) mode PD0: level output or PWM0 output PD1: level output or PWM1 output PD2: level output or PWM2 output PD3: level output or PWM3 output
$\overline{\text{INT0}}$ or $\overline{\text{INT1}}$ triggering edge selection: disable; high to low; low to high; low to high or high to low.

Application Circuits



Note: 1. Crystal/resonator system oscillators

For crystal oscillators, C1 and C2 are only required for some crystal frequencies to ensure oscillation. For resonator applications C1 and C2 are normally required for oscillation to occur. For most applications it is not necessary to add R1. However if the LVR function is disabled, and if it is required to stop the oscillator when VDD falls below its operating range, it is recommended that R1 is added. The values of C1 and C2 should be selected in consultation with the crystal/resonator manufacturer specifications.

2. Reset circuit

The reset circuit resistance and capacitance values should be chosen to ensure that VDD is stable and remains within its operating voltage range before the $\overline{\text{RES}}$ pin reaches a high level. Ensure that the length of the wiring connected to the $\overline{\text{RES}}$ pin is kept as short as possible, to avoid noise interference.

3. For applications where noise may interfere with the reset circuit and for details on the oscillator external components, refer to Application Note HA0075E for more information.

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	1^Note	Z, C, AC, OV
ADD A,x	Add immediate data to ACC	1	Z, C, AC, OV
ADC A,[m]	Add Data Memory to ACC with Carry	1	Z, C, AC, OV
ADCM A,[m]	Add ACC to Data memory with Carry	1^Note	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	1^Note	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	1^Note	Z, C, AC, OV
DAA [m]	Decimal adjust ACC for Addition with result in Data Memory	1^Note	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	1^Note	Z
ORM A,[m]	Logical OR ACC to Data Memory	1^Note	Z
XORM A,[m]	Logical XOR ACC to Data Memory	1^Note	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	1^Note	Z
CPLA [m]	Complement Data Memory with result in ACC	1	Z
Increment & Decrement			
INCA [m]	Increment Data Memory with result in ACC	1	Z
INC [m]	Increment Data Memory	1^Note	Z
DECA [m]	Decrement Data Memory with result in ACC	1	Z
DEC [m]	Decrement Data Memory	1^Note	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	1^{Note}	None
RRCA [m]	Rotate Data Memory right through Carry with result in ACC	1	C
RRC [m]	Rotate Data Memory right through Carry	1^{Note}	C
RLA [m]	Rotate Data Memory left with result in ACC	1	None
RL [m]	Rotate Data Memory left	1^{Note}	None
RLCA [m]	Rotate Data Memory left through Carry with result in ACC	1	C
RLC [m]	Rotate Data Memory left through Carry	1^{Note}	C
Data Move			
MOV A,[m]	Move Data Memory to ACC	1	None
MOV [m],A	Move ACC to Data Memory	1^{Note}	None
MOV A,x	Move immediate data to ACC	1	None
Bit Operation			
CLR [m].i	Clear bit of Data Memory	1^{Note}	None
SET [m].i	Set bit of Data Memory	1^{Note}	None
Branch			
JMP addr	Jump unconditionally	2	None
SZ [m]	Skip if Data Memory is zero	1^{Note}	None
SZA [m]	Skip if Data Memory is zero with data movement to ACC	1^{note}	None
SZ [m].i	Skip if bit i of Data Memory is zero	1^{Note}	None
SNZ [m].i	Skip if bit i of Data Memory is not zero	1^{Note}	None
SIZ [m]	Skip if increment Data Memory is zero	1^{Note}	None
SDZ [m]	Skip if decrement Data Memory is zero	1^{Note}	None
SIZA [m]	Skip if increment Data Memory is zero with result in ACC	1^{Note}	None
SDZA [m]	Skip if decrement Data Memory is zero with result in ACC	1^{Note}	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	2^{Note}	None
TABRDL [m]	Read table (last page) to TBLH and Data Memory	2^{Note}	None
Miscellaneous			
NOP	No operation	1	None
CLR [m]	Clear Data Memory	1^{Note}	None
SET [m]	Set Data Memory	1^{Note}	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	1^{Note}	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 master interrupt global enable bit. 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 ← 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] ← 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] ← [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 ← [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] ← 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 ← 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 ≠ 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

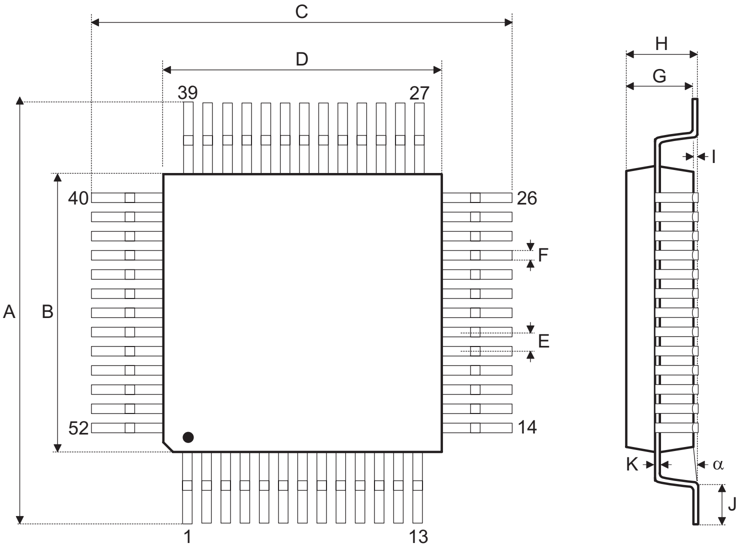
Description Data in the Accumulator and the specified immediate data perform a bitwise logical XOR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″XOR″ x

Affected flag(s) Z

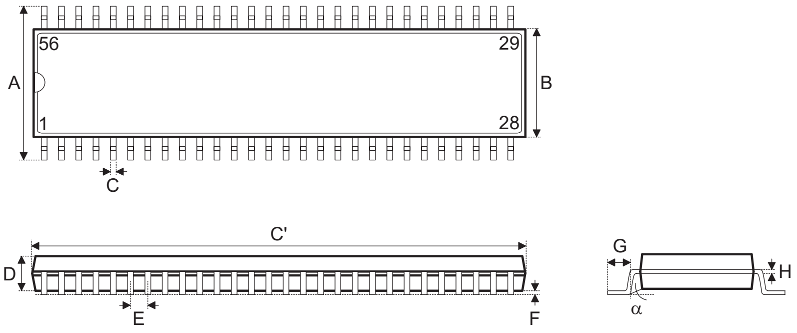
Package Information

52-pin QFP (14×14) Outline Dimensions



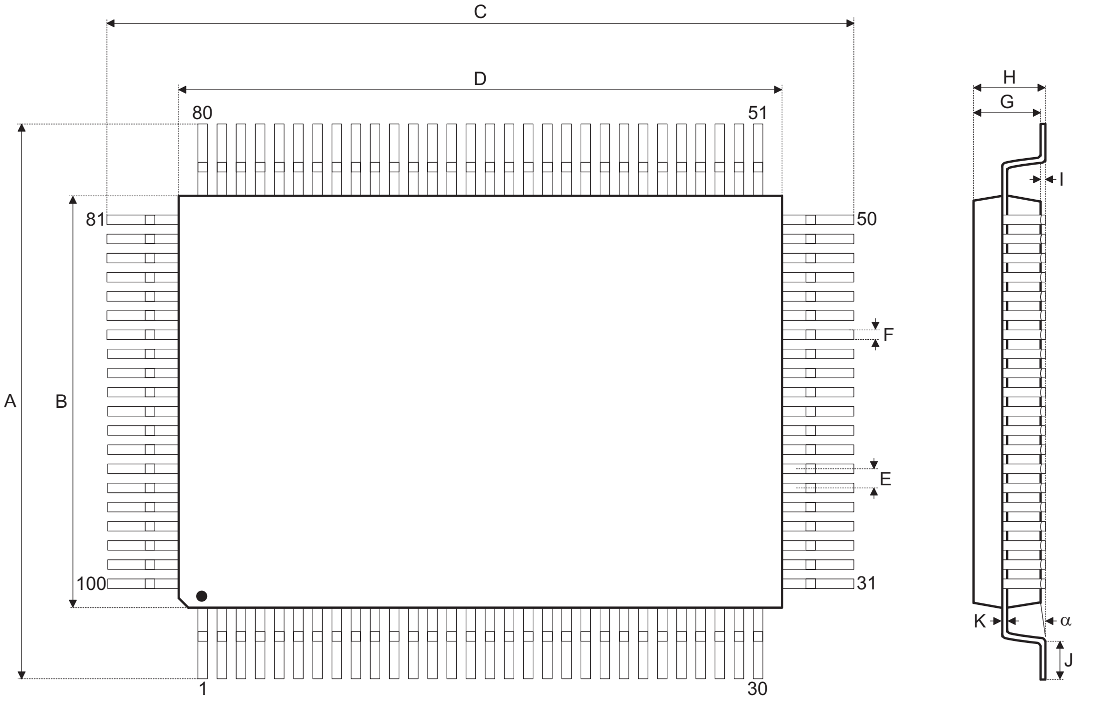
Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	17.3	—	17.5
B	13.9	—	14.1
C	17.3	—	17.5
D	13.9	—	14.1
E	—	1	—
F	—	0.4	—
G	2.5	—	3.1
H	—	—	3.4
I	—	0.1	—
J	0.73	—	1.03
K	0.1	—	0.2
α	0°	—	7°

56-pin SSOP (300mil) Outline Dimensions



Symbol	Dimensions in mil		
	Min.	Nom.	Max.
A	395	—	420
B	291	—	299
C	8	—	12
C′	720	—	730
D	89	—	99
E	—	25	—
F	4	—	10
G	25	—	35
H	4	—	12
α	0°	—	8°

100-pin QFP (14×20) Outline Dimensions



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

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

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

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

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

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

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

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

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